A Literature Survey on Secure Messaging App.

EDWIN DOMINIC
Department of Computer Science and Engineering (Cyber Security) Vimal Jyothi Engineering College Chemperi, Kannur
edwindominic7878@gmail.com

ABHINANDH P NARAYAN
Department of Computer Science and Engineering (Cyber Security) Vimal Jyothi Engineering College Chemperi, Kannur
abhinandhpnarayan@gmail.com

ELIZEBATH JESLIN
Department of Computer Science and Engineering (Cyber Security) Vimal Jyothi Engineering College Chemperi, Kannur elizebathjeslin75@gmail.com

MOHAMMED ADNAN
Department of Computer Science and Engineering (Cyber Security)
Vimal Jyothi Engineering College Chemperi, Kannur zaynadnan014@gmail.com

SR.REEMA JOSE
Assistant Professor
Department of Computer Science and Engineering (Cyber Security)
Vimal Jyothi Engineering College Chemperi, Kannur srreemajose@vjec.ac.in

***Abstract*—With privacy becoming more important today, this paper presents ”ZeroTrace,” a messaging app built to keep your chats safe and temporary. The app starts with a login screen where you can choose between using a phone number or Google to sign in. For the phone option, you enter your number, get a code from Firebase, and type it in to prove it’s you, then set up your profile. With Google, you pick your account and go straight to profile setup, where you add your name, photo, and a short bio, saved in Firestore. The main screen shows your chats and a button to pick contacts, letting you start secure talks in a chat window. You can send texts, files, pictures, videos, or voice notes—all locked with AES-256-GCM encryption, stored in Firestore and Firebase Storage, and unlocked only by the receiver. Your device keeps a copy for later. Once the receiver sees a message or file, it gets a 12-hour timer before it’s deleted by a Firebase tool that checks every hour, keeping things private. You can also manage your profile, delete your account, or log out from a settings page, and if you log in again, Firebase remembers you. By mixing strong encryption and smart storage, ZeroTrace offers a safe, easy way to message, better than most apps, fitting today’s security needs..**

I. Introduction

In the modern world, instant messaging has become a crucial element of communication, influencing both personal exchanges and interactions in professional and governmental settings. As more people rely on messaging applications to exchange sensitive information, concerns about privacy, data breaches, and government surveillance have escalated. High-profile incidents, such as leaks of private communications and the exploitation of centralized systems by malicious actors, underline the vulnerabilities inherent in traditional messaging architectures. Centralized messaging platforms often store user data on single points of control, making them prime targets for cyberattacks and unauthorized access. Additionally, these platforms frequently collect extensive metadata, which, even without message content, can reveal significant details about users’ habits and associations. Amid these challenges, the demand for decentralized, privacy-preserving alternatives has surged. Proposes a decentralized instant messaging system designed to address these critical issues. By leveraging a distributed server model, the system eliminates single points of failure, ensuring higher fault tolerance and scalability. Additionally, it incorporates end-to-end encryption via the Signal Protocol to guarantee that only intended recipients can access the message content. Such an approach aligns with global trends advocating for user empowerment, digital sovereignty, and resilience against cyber threats. As data privacy laws like GDPR and CCPA become stricter, this research contributes to creating messaging systems that respect user autonomy while providing robust security.

II. LITERATURE SURVEY

[1] Dashtinejad and Pejman proposed a system to evaluate the security of mobile messaging apps, key aspects include confidentiality (data access by authorized users), integrity (data modification by authorized users only), and availability (data access when needed). Authentication verifies identity to prevent impersonation. Weak authentication: Uses a single factor (e.g., password), vulnerable to brute force attacks. Strong authentication: Uses multi-factor methods like OTPs or certificate-based authentication (CBA) with public-private key cryptography.

User privacy ensures control over shared information. Many apps collect metadata (e.g., location, contacts) without user awareness, compromising privacy. Secure apps should minimize metadata collection and ensure transparency. App Security Architecture :

- Stage 1: Installation: Apps are downloaded from verified stores and installed automatically.
- Stage 2: Sign-Up: Users create a local profile with an email and PIN (used for encryption). A key pair is

generated, and a certificate is issued by a CA server. Registration with the IM server involves email verification and certificate exchange.
- Stage 3: Partner Verification: Establishes a TLS connection with the server. Facilitates secure certificate-based key exchange between users.

Message Exchange Process :

- Message Creation: A random key encrypts the message. The message hash is encrypted with the sender's private key. The message key is encrypted with the recipient's public key.
- Message Sending: The encapsulated message (encrypted text, key, and hash) is sent via the IM server.
- Message Receiving: The recipient verifies the hash and decrypts the message using the private key.

Implementation and Demo :

- Protocols: XMPP (XML-based, widely supported).
- Cryptography: AES-256 for encryption, RSA-2048 for key generation, HMAC-SHA1-512 for integrity.
- Server: Openfire with MySQL for message storage.
- Mobile Testing: Conducted on Android devices using Eclipse IDE.

Demo Highlights:

- Users add contacts and view their status.
- Initial chats are unencrypted, visible to the admin.
- Encrypted chats ensure privacy, appearing unreadable to the admin.

This system demonstrates a secure, privacy-focused messaging architecture with encryption, robust authentication, and effective key management.

[2] Anurag Mhatre , Nitesh Marchande , Riddhesh Khedekar and Amruta Sankhe proposed an architecture which outlines an online text messaging application designed for efficient and seamless communication between users. To ensure continuous communication, both users must maintain an active internet connection. The application also features cloud storage to enable users to store and retrieve data, and is organized into several modules, each serving a specific function while working together harmoniously. The major modules include:

- Database Module : The database operates in a test mode, which allows for flexibility in adjusting its rules and structure in the future as needed.
- Storage Module : This module enables users to upload and download files to and from Firebase cloud storage. It supports a variety of formats, including images,documents, and videos, helping users save local disk space.
- Registration Module : New users can sign up using their Google account or create one if they do not already have an account. Registration is performed using their Gmail ID.
- Login Module : Users log into the application by entering their Gmail ID and password,with passwords stored securely as hashed values to ensure privacy and security.
- Chat Module : The chat module allows users to initiate and exchange messages with others on the network. Conversations are initiated when a user starts a chat with another user.
- Backend Module : This module is crucial for enabling real-time functionality in the messaging application, ensuring that messages are delivered instantly and the overall experience remains smooth.

[3] Dev Mashru, Ganesha Maruthi Mangipudi, Harivardhan Swamy, Shivakumar Halangaliand Sushma E suggested a system that employs a decentralized architecture to overcome the limitations of traditional centralized systems. Unlike centralized instant messaging services with a single point of failure, this design integrates multiple active servers that communicate with one another. Key Features of the Multiple Server Approach :

- Fault Tolerance : Users select a server at registration from an active list or manually specify one. All communications are routed through this server chosen. If the server crashes, only the users connected to it are affected, while others continue uninterrupted. Users can switch to a different server manually to restore service.
- Server-to-Server Communication : Messages between users connected to different servers are handled by inter-server communication. For instance, if User A sends a message to User B on Server Y, Server X forwards the message to Server Y for delivery. This two-hop mechanism ensures decentralized data flow, eliminating reliance on a central hub.
- User Data Control : Data, including private encryption keys, is stored locally on user devices. Only public keys and essential metadata are shared with servers, minimizing privacy risks. Users can choose privacy-focused servers or host their own, ensuring control over their data.
- Resilience Against Malicious Servers: While a malicious server can collect metadata, it cannot decrypt messages due to end-to-end encryption (E2EE) via the Signal Protocol. This limits the potential for data misuse compared to centralized systems.
- Scalability: New servers can be integrated effortlessly, enabling organizations or individuals to contribute to the ecosystem. This lessens reliance on a singular entity and promotes collaboration.
- Enhanced Privacy via Decentralized Key Exchange: The need for a central key distribution center (KDC) is eliminated. Instead, servers store public encryption keys for their users, and key exchange occurs only when users establish connections (e.g., accepting friend requests).

Implementation Insights: The backend leverages cloud-hosted virtual machines (VMs) distributed across regions to demonstrate decentralization. REST APIs support both client-server and server-server interactions, ensuring smooth communication across the system. A buffer mechanism on the client side handles message delivery confirmations and retries, mitigating issues related to network delays or server downtime. Future Directions: To enhance this multiple-server approach: Automated Server Switching: This allows users to prioritize multiple servers, enabling automatic failover during crashes. Replication and Load Balancing: Introduce replication within servers to increase redundancy, combining decentralization with high availability. Cross-Platform Compatibility: Extend support to other platforms, such as iOS or the web, to broaden accessibility. By decentralizing the architecture and introducing multiple servers, the system achieves fault tolerance, enhanced privacy, and user control, addressing the primary challenges of centralized instant messaging systems.

[4] Nadim Kobeissi, Karthikeyan Bhargavan and Bruno Blanchet proposed a framework for securely implementing and verifying cryptographic messaging protocols, focusing on the Signal Protocol variant (SP). It highlights how protocols like SP ensure message confidentiality, authenticity, forward secrecy, and future secrecy, even if long-term keys are compromised. Key features include authenticated key exchanges using multiple Diffie-Hellman handshakes and a double-ratchet mechanism for key updates. The Signal Protocol is a widely regarded end-to-end encryption (E2EE) protocol that ensures secure communication by encrypting messages on the sender's device and decrypting them on the recipient's device. Its key features include:

- Authenticated Key Exchange: It employs both long-lasting and temporary Diffie-Hellman keys to create secure connections that ensure forward secrecy, ensuring past messages stay protected even if the existing keys are breached.
- Double Ratchet Mechanism: This mechanism ensures key freshness by updating encryption keys with every message sent or received, providing both forward and future secrecy.
- Forward Secrecy: If long-term keys are compromised, previously exchanged messages remain protected, as session keys are regularly updated and discarded. Message Integrity: Messages are authenticated to ensure they are not tampered with during transmission.
- Implementation in Messaging Apps: The protocol underpins popular applications like Signal, WhatsApp, and Facebook Messenger, ensuring secure communication even if the messaging servers are compromised.

The Signal Protocol utilizes contemporary cryptographic methods such as Curve25519 for exchanging keys, AES-GCM for encrypting messages, and HMAC-SHA256 for ensuring integrity. These features, combined with its robust security design, make it a benchmark for secure communication protocols.

[5] Vijay Bhuse suggested that End-to-End Encryption (E2EE) is a security method that secures messages by encrypting them on the sender's device and permitting decryption only on the recipient's device, making them unapproachable to any external parties. The Signal Protocol is a leading implementation of E2EE, using advanced cryptographic techniques such as public-key cryptography for secure key exchange, symmetric-key encryption for efficient message handling, and forward secrecy to protect past conversations even if current keys are compromised. This robust protocol significantly enhances user privacy by safeguarding messages from interception by hackers, governments, or even messaging service providers. Despite the proven security of E2EE systems like the Signal Protocol, many social media companies fail to adopt such measures universally. This paper examines the varying security practices of social media platforms and highlights the importance of implementing stronger privacy protections for users.

[6] Tsai-Yeh Tung, Laurent Lin, D.T.Lee introduced secure instant messaging is essential for protecting sensitive data, yet many messaging services prioritize speed over security, leaving users vulnerable to data breaches. One effective approach to enhancing security is the use of self-destructing messages, which automatically delete after a set time, preventing unauthorized access and long-term storage risks. This method relies on temporary encryption keys, ensuring that messages remain secure even in cloud-based environments. The system operates with four key components: the sender, the recipient, the messaging server, and an optional ephemeral key management system. The sender encrypts the message using a temporary public key obtained from the receiver. Once the receiver decrypts it, the ephemeral private key is securely deleted, making message recovery impossible. The messaging server facilitates communication, while the ephemeral key manager enables offline messaging, synchronization, and push notifications. By integrating self-destructing messages, this architecture enhances confidentiality and prevents data persistence. Automatic deletion reduces exposure to hacking, data leaks, and unauthorized access, making it an effective solution for secure communication. Future improvements should focus on simplifying implementation and optimizing performance for large-scale use while maintaining strong security measures.

[7] Dong Zheng, Liang Xue, Chao Yu, Yannan Li,and Yong Yu suggested that assured data deletion in cloud storage is essential to ensure that once data is deleted, it cannot be retrieved under any circumstances. With the widespread use of cloud services, proper management and disposal of sensitive data have D. Mashru, G. M. Mangipudi, H. Swamy, S. Halangali and S. E, ”A Decentralised Instant Messaging Application with End-to-End Encryption,” 2023 20th Learning and Technology Conference (L and T), Jeddah, Saudi Arabia,

2023, pp. 48-53,become critical for security and compliance. If cloud service providers fail to guarantee complete deletion, they risk reputation loss and legal penalties. Users increasingly demand secure deletion mechanisms to protect their data from unauthorized access. Traditional deletion methods, such as physical destruction, are impractical in cloud environments due to features like virtualization and multi-tenancy, making assured deletion a complex challenge. Trust in cloud providers is crucial, yet there is no definitive technical proof that deleted data is permanently erased. Many users encrypt their data before uploading it to the cloud, leading to solutions focused on deleting encryption keys. However, this approach has drawbacks, including high computational demands, maintenance challenges, and restrictions on data usability. Existing research highlights the risks of incomplete deletions, where residual data may still be accessible even after deletion requests. Mechanisms ensuring complete and verifiable deletion are necessary to mitigate these risks. Various approaches have been explored, but limitations remain, necessitating more efficient and reliable solutions. The proposed techniques aim to provide verifiable deletion while addressing challenges such as multi-tenancy and data deduplication. Practical implementation requires balancing computational efficiency, security, and user acceptance. Ensuring fine-grained access control and proof of deletion is critical to building user confidence in cloud storage. Ultimately, enhancing assured data deletion mechanisms will strengthen cloud security, reduce vulnerabilities, and promote user trust in cloud services, making it imperative for ongoing research and development.

[8] Yuchuan Luo, Ming Xu, Shaojing Fu, and Dongsheng Wang Cloud storage provides an easy method for storing and retrieving data.However, with the convenience of cloud storage comes the risk of data breaches and unauthorized access. Ensuring secure deletion remains a challenge. Since users lack direct control over their data, deletion must be reliable and irreversible. A proposed solution uses encryption: data is encrypted before storage, and deletion is achieved by destroying the encryption key. The process works as follows: The data owner encrypts the file with a random key. A third-party manager encrypts this key with a control key. The control key expires after a set time or upon deletion request, making the data unrecoverable. A secure deletion system must guarantee honesty, permanence, and efficiency. The cloud provider should transparently handle deleted data, ensuring it cannot be recovered. Deletion must be permanent and not merely hidden, and the process should be efficient without excessive resource consumption. Secure deletion is essential for protecting sensitive data and preventing breaches in cloud storage.

[9] Mohamed Hamdy Eldefrawy, Khaled Alghathbar1 and Muhammad Khurram Khan proposed One-Time Password (OTP) system which is designed to enhance user authentication by generating a unique password for each login session or transaction, thereby mitigating the risk of replay attacks. This system utilizes a nested hashing approach, allowing for forward and infinite OTP generation without the need for a synchronized server clock or costly SMS delivery methods. By employing two different hash functions, the OTP generation process ensures that even if a seed is compromised, users can manually reregister with new seeds, maintaining the integrity and security of the authentication process.The main advantage of using a mobile phone in the two-factor authentication technique is that most users already possess mobile phones, eliminating the need for additional hardware tokens that require purchase, deployment, and support. This integration simplifies the authentication process while enhancing security by combining something the user knows (a static password) with something they have (an OTP sent to their mobile device).

[10] Hemalatha S suggested that fingerprint recognition is a biometric authentication system that uses a person’s fingerprints to identify them. Fingerprint recognition systems typically involve capturing a fingerprint image, preprocessing the image, extracting features from the image, and matching the features to a database of fingerprints. Fingerprint recognition systems are used in a variety of applications, including security systems, access control systems, and point-of-sale systems. Here are the important concepts of fingerprint recognition system: Fingerprint image preprocessing: This step involves improving the quality of the fingerprint image by removing noise and enhancing the fingerprint ridges. Fingerprint feature extraction: This step involves extracting features from the fingerprint image, such as minutiae (ridge endings and bifurcations). Fingerprint matching: This step involves comparing the features extracted from the fingerprint image to the features stored in a database of fingerprints. Fingerprint recognition systems are becoming increasingly popular due to their accuracy and reliability. However, fingerprint recognition systems are also vulnerable to spoofing attacks, where a fake fingerprint is used to fool the system.

Reference	Description	Advantages	Disadvantages
[1]	- Evaluates security of mobile messaging apps. - Focuses on confidentiality, integrity, and availability. - Uses multi-factor authentication (OTP, CBA) and encryption.	- Strong encryption and authentication. - Ensures user privacy and data security.	- Complex implementation. - Metadata collection risks.
[2]	- Cloud-based messaging app with real-time chat. - Includes Firebase storage for file uploads/downloads. - Uses Google account integration for registration/login.	- Easy to use and integrates with Google. - Saves local storage space.	- Requires constant internet. - Limited focus on encryption.
[3]	- Decentralized system with multiple servers. - Ensures fault tolerance and user data control. - Uses Signal Protocol for end-to-end encryption.	- No single point of failure. - Enhanced privacy and scalability.	- Complex server management. - Manual server switching required.
[4]	- Secure messaging protocol with forward and future secrecy. - Uses double ratchet mechanism for key updates. - Ensures message confidentiality and integrity.	- High security and forward secrecy. - Widely used in apps like Signal and WhatsApp.	- Complex to implement. - Performance overhead.
[5]	- Ensures messages are encrypted on sender's device. - Uses Signal Protocol for secure key exchange. - Protects messages from interception.	- Enhances user privacy. - Protects against hackers and governments.	- Not universally adopted. - Complex key management.
[6]	- Messages automatically delete after a set time. - Uses temporary encryption keys for security. - Prevents unauthorized access and data persistence.	- Reduces data breach risks. - Enhances confidentiality.	- Complex key management. - Not suitable for persistent data.
[7]	- Ensures data is permanently deleted in cloud storage. - Uses encryption key destruction for secure deletion. - Provides verifiable deletion mechanisms.	- Protects sensitive data. - Ensures compliance with regulations.	- High computational cost. - Complex key management.
[8]	- Uses encryption to ensure permanent data deletion. - Destroys encryption keys to make data unrecoverable. - Ensures transparency and efficiency in deletion.	- Reliable and irreversible deletion. - Protects against breaches.	- Relies on third-party key managers. - Lack of user control.
[9]	- Generates unique passwords for each login session. - Uses nested hashing for OTP generation. - Enhances security with mobile-based 2FA.	- Prevents replay attacks. - No need for hardware tokens.	- Vulnerable to mobile device compromise. - Delays in OTP delivery.
[10]	- Biometric system using fingerprints for identification. - Involves image preprocessing, feature extraction, and matching. - Used in security and access control systems.	- High accuracy and reliability. - No need for passwords.	- Vulnerable to spoofing attacks. - Privacy concerns with biometric data.

III. CONCLUSION

As digital communication continues to evolve, so do the threats that compromise user privacy, data security, and freedom of expression. From mass surveillance programs exposed by Edward Snowden to data breaches affecting millions of users worldwide, the risks associated with centralized messaging platforms have become evident. Governments, corporations, and cybercriminals alike have exploited weaknesses in traditional systems, leading to privacy invasions, censorship, and unauthorized data harvesting. Against this backdrop, the need for a decentralized, privacy-preserving messaging solution has never been more urgent. This research introduces a fault-tolerant, encrypted, and decentralized instant messaging system that addresses these critical concerns. By eliminating reliance on a single central authority and implementing a distributed server model, the proposed system enhances resilience against server failures, cyberattacks, and data misuse. Furthermore, by leveraging end-to-end encryption (E2EE) through the Signal Protocol, it ensures that only intended recipients can access message content, thereby fortifying communication against eavesdropping and unauthorized interception. In addition to technical advantages, the proposed system has far-reaching societal and geopolitical implications. Secure communication is essential for journalists, activists, whistleblowers, and individuals living under repressive regimes who rely on encrypted messaging to avoid persecution. Furthermore, businesses handling confidential intellectual property, legal firms managing sensitive case files, and financial institutions processing high-stakes transactions all require secure communication channels to prevent corporate espionage and data leaks. However, while the decentralized approach enhances security and user autonomy, it also introduces challenges such as scalability, interoperability, and ease of use for non-technical individuals. Future research should focus on optimizing automated server failover mechanisms, improving user-friendly key management solutions, and exploring cross-platform compatibility to increase adoption. Additionally, resistance from governments advocating for encryption backdoors poses a legal challenge that must be addressed through ongoing discussions on digital rights, policy frameworks, and cybersecurity regulations. Despite these challenges, this research lays a strong foundation for rethinking the future of secure messaging. By combining decentralization, cryptographic advancements, and privacy-centric architecture, the proposed system offers a robust alternative to traditional messaging platforms, fostering a more secure digital environment. As cyber threats and regulatory battles continue to shape the landscape of digital communication, solutions like this will play a pivotal role in protecting user privacy, enhancing digital sovereignty, and ensuring secure, censorship-resistant communication for generations to come.

REFERENCES

[1] Dashtinejad, Pejman. ”Security System for Mobile Messaging Applications, Jan. 8, 2015.” KTH University, web edition: 1-31.

[2] Anurag Mhatre, Nitesh Marchande, Riddhesh Khedekar, Amruta Sankhe .”Android Based Instant Messaging Application Using Firebase”,International Research Journal of Engineering and Technology (IRJET), Volume:09, Issue:04, Apr 2022

[3] D. Mashru, G. M. Mangipudi, H. Swamy, S. Halangali and S. E, ”A Decentralised Instant Messaging Application with End-to-End Encryption,” 2023 20th Learning and Technology Conference (L and T), Jeddah, Saudi Arabia, 2023, pp. 48-53.

[4] N. Kobeissi, K. Bhargavan and B. Blanchet, ”Automated Verification for Secure Messaging Protocols and Their Implementations: A Symbolic and Computational Approach,” 2017 IEEE European Symposium on Security and Privacy (EuroS and P), Paris, France, 2017, pp. 435-450.

[5] Bhuse, Vijay. ”Review of End-to-End Encryption for Social Media.” In International Conference on Cyber Warfare and Security, vol. 18, no. 1, pp. 35-37. 2023.

[6] T. -Y. Tung, L. Lin and D. T. Lee, ”Pandora Messaging: An Enhanced Self-Message-Destructing Secure Instant Messaging Architecture for Mobile Devices,” 2012 26th International Conference on Advanced Information Networking and Applications Workshops, Fukuoka, Japan, 2012, pp. 720-725.

[7] D. Zheng, L. Xue, C. Yu, Y. Li and Y. Yu, ”Toward Assured Data Deletion in Cloud Storage,” in IEEE Network, vol. 34, no. 3, pp. 101-107, May/June 2020.

[8] Luo, Yuchuan, Ming Xu, Shaojing Fu, and Dongsheng Wang. ”Enabling assured deletion in the cloud storage by overwriting.” In Proceedings of the 4th ACM international workshop on security in cloud computing, pp. 17-23. 2016.

[9] M. H. Eldefrawy, K. Alghathbar and M. K. Khan, ”OTP-Based Two-Factor Authentication Using Mobile Phones,” 2011 Eighth International Conference on Information Technology: New Generations, Las Vegas, NV, USA, 2011, pp. 327-331

[10] Hemalatha, S. ”A systematic review on Fingerprint based Biometric Authentication System.” In 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), pp. 1-4. IEEE, 2020.