Self-Destruct Messenger

By:

Muhammad Aatif Khan
38587
Hamza Zawari Khalid
35772
Syed Abdurrehman
31980

Supervised by:
Dr. Jawad Iqbal

Faculty of Computing
Riphah International University, Islamabad
Spring/Fall 2021

Submitted To

Faculty of Computing,

Riphah International University, Islamabad

As a Partial Fulfillment of the Requirement for the Award of the Degree

of

Bachelors of Science in Cyber Security

Faculty of Computing
Riphah International University, Islamabad

Date: [date of final presentation]

Final Approval

This is to certify that we have read the report submitted by Muhammad Aatif Khan (38587), Hamza Zawari Khalid (35772), Syed AbdurRehman(31980), for the partial fulfillment of the requirements for the degree of the Bachelor of Science in Cyber Security (BS CYS). It is our judgment that this report is of sufficient standard to warrant its acceptance by Riphah International University, Islamabad for the degree of Bachelor of Science in Cyber Security (BS CYS).

Committee:

1

_____

Dr. Jawaid Iqbal
(Supervisor)

2

_____

Mr .Yawar Abbas
(In-charge Cyber Security)

3

_____

Dr. Musharraf Ahmed
(Head of Department)

Declaration

We hereby declare that this document "**Self-Destruct Messenger**" neither as a whole nor as a part has been copied out from any source. It is further declared that we have done this project with the accompanied report entirely on the basis of our personal efforts, under the proficient guidance of our teachers especially our supervisor **DR. Jawad Iqbal**. If any part of the system is proved to be copied out from any source or found to be reproduction of any project from anywhere else, we shall stand by the consequences.

Muhammad Aatif Khan
38587

Hamza Zawari Khalid
35772

Syed Abdurrehman
31980

Dedication

This project is dedicated to our friends, mentors, and educators, whose unwavering support and encouragement have been a constant source of strength throughout this journey. Their belief in our capabilities has fueled our determination to succeed. It is for this reason that we must express gratitude to our colleagues and the academic community who have provided us with advice and knowledge as we have evolved in our respective careers. Their faith in us has motivated us to become better than what we thought we could achieve. So thank you for being there for us.

Acknowledgement

We would like to express our sincere gratitude to the following individuals whose guidance and support were instrumental in the successful completion of this project:

- **Project Supervisor:** We are particularly grateful to our project supervisor, **Dr. Javed Iqbal**, for his invaluable expertise, insightful feedback, and unwavering encouragement throughout the entire project development process. His dedication and support played a pivotal role in shaping the direction and success of our work.
- **Faculty Advisors:** We extend our sincere thanks to our faculty advisors, **Dr. Muhammad Mansoor Alam,** whose awareness for different messaging system help us to pursue this project, **and Mr Hammayun,** whose knowledge and guidance were invaluable throughout the research phase.
- **Technical Support Staff:** We appreciate the assistance provided **by Mr. Osama Raza, Mr. Awais Nawaz, Mr. Tabbasum Javed**, and **Mr. Mueed Mirza.** Their technical expertise and willingness to help were crucial in overcoming technical challenges.
- **Project Convener:** We acknowledge the leadership of **Mr. Yawar Abbas** as the project convener, whose guidance ensured the project's smooth execution within the academic framework.

Muhammad Aatif khan

38587

Hamza Zawari Khalid

35772

Syed Abdurrehman

31980

Abstract

In the contemporary world which is highly interconnected by virtue of the internet, secure communication is important both personal and corporate perspective. This project introduces the Self-Destruct Messenger as a confidential and secure messaging application that is embedded and designed with automated data deletion, encryption, and real-time chat restrictions features in order to protect private messaging. This approach incorporates innovative protection mechanisms that limit the chances of unwanted penetration and ensure secrecy of information. At the same time Self-Destruct Messenger provides a special set of features for secure communication. Conversations are protected by using different keys with encryption that takes place during the session and all the messages will evaporate into thin air shortly after they are received. Messages are not stored continuously since real time communication restrictions allow both users to see the message only when they are online to prevent over exposure of data. Excess data on devices or servers is also eliminated as the sent messages are automatically erased after the control over the session is erased. Optional watermarks and anti-screenshot features help to safeguard the message content from being captured. Self-Destruct Messenger has a straightforward user interface that was developed for desk pier and mobile devices aimed at functionality and ease of use. Thanking to its extensive security, clients are able self-explanatory mentor access information Self-Destruct Messenger is ready to allow consumers and businesses to communicate in private, secure ways at a time where cyber security risks are ever evolving. Users of Self Destruct Messenger will remain in constant control over their communications even with potential threats of a data breach.

Keywords: Secure Communication, Real-Time Encryption, Self-Destruct Messenger, Data Deforestation, Anti-Screenshot, Privacy Protection, Cybersecurity.

Table of Contents

CHAPTER 1: 1

INTRODUCTION 1

1.1 Introduction 2
1.2 Opportunity & Stakeholders 2
1.2.1 Opportunities 2
1.2.2 Stakeholders 3
1.3 Motivations and Challenges 3
1.3.1 Motivations 3
1.3.2 Challenges 4
1.4 Significance of Study 5
1.5 Goals and Objectives 6
1.6 Scope of Project 6
1.7 Chapter Summary 7

CHAPTER 2: 9

LITERATURE / 9

MARKET SURVEY 9

2.1 Introduction 10
2.2 Literature Review/Technologies 10
2.3 Comparative Analysis 12
Dynamic Encryption: 12
Real-Time Communication Requirement: 13
Anti-Screenshot Measures: 13
Automatic Session-Based Deletion: 13
2.4 Research Gaps 13
Static Encryption Keys: 14
Vulnerability for Offline Messages: 14
Screenshot Prevention: 14
Persistent Data Retention: 14
2.5 Problem Statement 14
2.6 Chapter Summary 15

CHAPTER 3: 16

REQUIREMENTS AND SYSTEM DESIGN 16

3.1 Introduction 17
3.2 System Architecture 18
3.3 Functional Requirements 18
3.4 Non Functional Requirements 19
3.5 Design Diagrams 20
3.5.1 Use Case: 20
3.5.2 Dataflow: 21
3.6 Hardware and Software Requirements 22
3.6.1 Hardware Requirements 22

DEVELOPMENT MACHINE REQUIREMENTS 22

Device Requirements for Testing 23

End-User Device Requirements *23*
Security Features (Optional) *23*
3.6.2 Software Requirements: *24*
3.7 THREAT SCENARIOS 25
3.8 THREAT MODELING TECHNIQUES 26
STRIDE: *26*
DREAD: *26*
LINDDUN: *26*
Attack Trees: *26*
3.9 THREAT RESISTANCE MODEL 26
3.10 CHAPTER SUMMARY 27
REFERENCE AND BIBLIOGRAPHY **28**
GITHUB **29**

List of Figures

Figure 2.1: Survey/ Literature Review 14
Figure 3.1: System Architecture 19
Figure 3.2: Use Case 21
Figure 3.3: Dataflow 22
Figure 3.4: Threat Resistance Model 27

List of Tables

Table 2.1: Survey/ Literature Review…… 22

Table 2.2: Existing Systems…… 24

Chapter 1: Introduction

Chapter 1: Introduction

1.1 Introduction

Now that everything is becoming digital, security and privacy have become crucial issues. There has been an exponential rise in data breaches, hacks and illegal access therefore there are serious concerns on how to secure private information while communicating. Applications like WhatsApp, Telegram and Signal have end-to-end encryption technologies, however, some shortcomings do exist including the limited amount of encryption and static encryption keys, and screenshotted malicious information. Right now, there are issues of privacy in communicating through phone applications due to stored communications.

The Self-Destruct Messenger project aims to provide a communication platform that surpasses the privacy limitations of existing systems. The project intends to provide end-to-end encrypted secure messaging network that allows for real time conversations and uses multiple keys to encrypt endpoints for individual channels. Automatic message deletion would be enabled at the end of the exchange so that messages are not permanently saved on either the servers or the devices. On top of that, anti-screenshot mechanisms would also be included to go a step further preventing any extreme sensitive information from being captured or shared. This platform is extremely beneficial to individuals requiring the utmost privacy particularly in the legal, medical, governmental, and corporate industries.

1.2 Opportunity & Stakeholders

This platform is extremely beneficial to individuals requiring the utmost privacy particularly in the legal, medical, governmental, and corporate industries

1.2.1 Opportunities

- **Protection of Concerns Regarding Sensitive Information:** The focus of this project is to provide a secure means of communication to users who need privacy such as the legal, medical or business fields.
- **Need for a Privacy Focused Tool In The Communication Market:** The increasing awareness concerning data protection and privacy gives rise to a huge gap in the market for a communication instrument that ensures safety and security of user data.

- **Advances in Secure Communication:** The project also enables the involvement in the development of new technologies in the field of secure communications using software systems that provide autonomous criminalization of messages regardless of format including video of the user in addition to self-deleting texts and screenshotted images.

1.2.2 Stakeholders

- **The Government and Law Enforcement Agencies:** Such external parties require secure communications facilities for the transmission of sensitive information pertaining to national security, investigations, and law enforcement procedures. It can provide secure communications and sharing of sensitive information within organizations.
- **Compliance Officers:** Vulnerability and threat intelligence discussing with client's data security concerns the need for adequate communication tools for Cybersecurity practitioners and consultants without being tapped or breached.
- **Businesses and Corporations**: Even such services are required for a multitude of industries, including but not limited to: Banking services, financial services, healthcare, and even law firms, require a secure service that will protect crucial communications as well as data privacy laws.
- **End Users**: Individual users who prioritize privacy and security in their personal or professional communications, including legal professionals, healthcare providers, and executives.

1.3 Motivations and Challenges

1.3.1 Motivations

- **Increasing Data Breaches and Privacy Concerns**

  The protection of confidential information has grown into a critical concern among individuals, on one hand, and private and state institutions, on the other. The necessity for privacy calls for an automated solution like Self Destruct Messenger which goes ahead to make communication more private.

- **Limitations of Current Messaging Apps**

However, certain secure messaging applications, such as Signal or WhatsApp, can be given end-to-end encryption, but they still have data vulnerabilities. An app's message history or static encryption keys are two of the oversights that expose these apps to significant vulnerability. Such facts indicate the need for a better system that advances the current situation regarding safety and security of data beyond encryption.

- **Demand for Privacy-Centric Solutions**

Considering the workplace of professionals involved in reserves like Legal, Government and Healthcare, they require secure communication with little or no breaches. All these specialists require a means of communication that fully secures information especially when the said information is very confidential. The proposed approach is to make the communication secure by integrating necessary features with security requirements of the users.

- **Need for Ephemeral Messaging**

There are numerous apps that allow users to retain messages permanently which can compromise the safety and security of the users if the messages are accessed and mismanaged. The urgency that drove the development of this project was the need for all communication to be temporary and disappear completely after a short time.

- **User Demand for Control Over Data**

As it stands, more users are looking to control their data, since they deem privacy as a right. This need is further supported by the idea to grow the project, as features such as automatic messaging deletion and anti-screenshot warnings allow users to better handle their information shared.

1.3.2 Challenges

- **Implementing Dynamic Encryption**

With the aim of building a self-destruct messenger application, this project intends to alternate encryption dynamics for each session unlike traditional messaging applications which rely solely on static encryption keys. This presents a problem because it requires a degree of encryption complexity which would compromise the speed of instantaneous communication.

- **Anti-Screenshot Protection**

Allowing users to not take screenshots during a self-destruct mode is one of the noticeable features of a Self-Destruct Messenger. To configure such functionality certain platform APIs

must be used, however, such capabilities might be limited by the operating system which makes this part of the project more complicated.

- **Real-Time Communication Requirement**

  To improve security on the platform, communication between users is limited to only when they are both online. Even though this does improve security of the platform, it may frustrate users who are accustomed to sending messages offline.

- **Automatic Message Deletion**

  To ensure that messages get deleted on both the client and the server, a proper implementation is required at the backend. Creating secure deletion techniques to ensure that data cannot and will not be able performed retrieval. presents a challenge, particularly in real-time settings.

- **User Authentication and Access Control**

  Allowing only trusted users to take part in any private conversation and enforcing a safe user login is an obstacle to overcome. There is a requirement for a strong restriction that helps avoid unauthorized access without creating a barrier to user experience.

1.4 Significance of Study

- **Enhanced Privacy and Security for Sensitive Communications**

  This research is of great significance since its aim is to facilitate the development of a communication instrument that is protected from interception. In sectors such as healthcare, attorneys, or even the government, secure communication is vital. Self-Destruct Messenger would ensure that such professionals engage in meaningful conversations without any risk of amassing or gaining access to sensitive information.

- **Addressing Gaps in Existing Messaging Platforms**

  However, it is vital to understand that current ones like WhatsApp, Signal or Telegram provide encryption but do not offer dynamic encryption or anti-screenshot. Thus, by bridging these gaps the Messenger That Self Destructs would set up new standards in secure communication and usher in new strides in the field of messaging.

- **Meeting the Market Demand for Privacy-Centric Solutions**

  With the growing awareness of privacy protection, there is a growing need for data-centric solutions that prioritize security and control over the information. This study is important because it creates a device that would be appealing to the numerable users thought to optimize their

privacy, thereby meeting some of the market demand for more secure functions and taking a corner in the secure communications market.

- **Supporting Data Protection and Compliance in Professional Fields**

  To comply with various privacy-related regulations such as GDPR or HIPAA, many businesses seek appropriate protection systems for their data. The Self-Destruct Messenger can help individuals and businesses to achieve these objectives by providing a communications tool which destroys messages as soon as they are sent thus preventing any further exposure risk and properly facilitating compliance

1.5 Goals and Objectives

- **Implement Real-Time Communication**

  To avoid offline message vulnerabilities, make sure that messages can only be sent and received when bothusers are online.

- **Dynamic Encryption**

  Employ dynamic encryption which makes it much more difficult to breach the session key by creating and using a new session encryption key for every session.

- **Automatic Message Deletion**

  All communications will be permanently erased from the client and server at the end of the session, leaving no trace.

- **Mechanisms Against Screenshots**

  Stop users from taking screenshots of confidential data by using platform-specific APIs and other security measures.

- **User Authentication and Authorization**

  Make sure that strong authenticationprocedures are in place so that only authorized users can start or join a communication session.

1.6 Scope of Project

- **User Authentication**

  Ensures the system is only accessed by the specified users.

- **Secure Key Exchange**

  Safely transfers encryption keys between communicatingparties.
- **Dynamic Encryption**

  Generates unique encryption keys for each session tosecure communication.
- **Real-Time Communication**

  Enables live, secure exchanges between users inreal time.
- **Automatic Message Deletion**

  Delete messages once a session ends or usersdisconnect.
- **Anti-Screenshot Measures**

  Prevents users from capturing screenshots ofconversations.
- **User Interface & Experience**

  Focuses on creating an intuitive, user-friendlyinterface.
- **Chat Interface**

  Provides a simple and secure space for text-based real-timecommunication.
- **Session Status Indicators**

  Displays real-time information about session activityor user presence.
- **System Design & Architecture**

  Blueprint for integrating all components withscalability and security.
- **Security Testing**

  Assesses system vulnerabilities through rigorous testingmethodologies.
- **Deployment & Monitoring**

  Ensure system functionality and security post-launch with continuous monitoring.

1.7 Chapter Summary

The goal of the Self-Destruct Messenger project is to create a protected, private messaging tool allowing individuals and organizations who are holding private discussions to communicate.

As theft and unauthorized access tend to increase, other messaging applications like WhatsApp and Signal still pose serious threats to privacy because encryption keys are static, and messages are always stored. The Self-Destruct Messenger can avoid these issues with tools such as dynamic encryption, after a message is opened it will delete itself, and protection against screenshots.

- **Key Opportunities and Stakeholders:**

  Institutions that process confidential and sensitive records such as the government, medical, legal, and corporate sectors have a bright outlook for the platform. The main stakeholders consist of government institutions, security experts, businesses, and individual users who seek privacy.

- **Motivation and problem:**

  The position is fueled by the increased need of users for control over their existing application privacy and the rising incidents of breaches. The rest of the problem stem from dynamic encryption implementations, screenshot protection, real-time messaging, secure message deletion, and user authentication.

- **Significance of the Study:**

  This project addresses the deficiencies left by existing platforms, meets the needs of the market with respect to privacy, and protects privacy by supporting data protection measures as such industries where data privacy is adhered to find it crucial.

- **Goals and Objectives:**

  The specific objectives of the project are Real Time Messaging, implementing dynamic encryption, real time auto delete of the messages, screenshot prevention measures and enhancing user authentication mechanisms.

- **Project Areas:**

  The scope includes end user authentication, key exchange, encrypted messages in real time, delete messages automatically, protect snapshots, user friendly design, system design, security evaluation, and surveillance after release.

Chapter 2:
Literature / Market Survey

Chapter 2: Introduction

2.1 Introduction

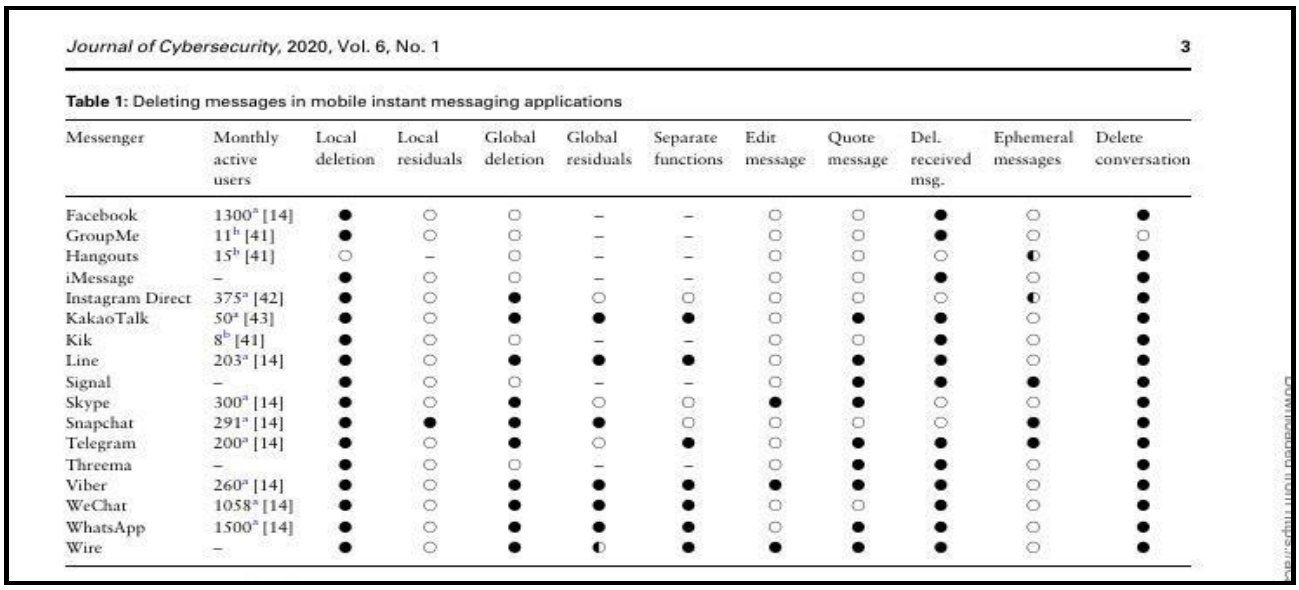
Receiving and sending encrypted messages is becoming ever more important in today's information age. Strong measures collapsed the need for easy privacy protection when conversations were done in private. In order to enhance security, messaging applications like Signal, Telegram, and WhatsApp use self-destruct messages together with end-to-end encryption. However, some of the issues that these systems still seem to encounter include static encryption keys, offline storage of messages, and lack of effective anti-screenshot measures. In this section, we discuss these systems and illustrate their weaknesses in a bid to provide a clear scope for the intended improvements in this project. On the other hand, the Self- Destruct Messenger project looks to resolve such issues by employing advanced security principles and architecture. These features include, but are not limited to, anti-screen capture measures such as session level encryption, real-time communication, and message destroying feature after the session termination.

2.2 Literature Review/Technologies

This section reviews prominent technologies and products that focus on secure messaging. The overview includes encryption, ephemeral messaging, deletion and anti-screenshot features in the existing system.

- Ephemeral Messaging and Deletion Features The essence of ephemeral messaging can also be described by self-destructing messages that are sent but only for a period, after which they vanish or disappear upon the user's choice.
- Encryption Standards Standards for Encryption Secure messaging requires encryption, and the most widely used method is end-to-end encryption (E2EE). E2EE makes sure that the content of the communication is only visible to the sender and the recipient. Nonetheless, there are significant differences even within E2EE frameworks:
- Static Encryption Keys: A lot of systems (like Telegram) only employ one key at a time. This static technique is secure, but it exposes the entire session if the key is compromised.
- Dynamic Encryption Keys: As an alternative, dynamic encryption isolates each communication instance and creates distinct keys for each session or message. Although rather uncommon in commonplace applications, this method greatly improves security. By creating new keys for each

session, the Self-Destruct Messenger will integrate dynamic encryption, lowering the danger of exposure.



Journal of Cybersecurity, 2020, Vol. 6, No. 1 3

Table 1: Deleting messages in mobile instant messaging applications

Messenger	Monthly active users	Local deletion	Local residuals	Global deletion	Global residuals	Separate functions	Edit message	Quote message	Del. received msg.	Ephemeral messages	Delete conversation
Facebook	1300[a] [14]	●	○	○	–	–	○	○	●	○	●
GroupMe	11[b] [41]	●	○	○	–	–	○	○	●	○	○
Hangouts	15[b] [41]	○	–	○	–	–	○	○	○	◐	●
iMessage	–	●	○	○	–	–	○	○	●	○	●
Instagram Direct	375[a] [42]	●	○	●	○	○	○	○	○	◐	●
KakaoTalk	50[a] [43]	●	○	●	●	●	○	●	●	○	●
Kik	8[b] [41]	●	○	○	–	–	○	○	●	○	●
Line	203[a] [14]	●	○	●	●	●	○	●	●	○	●
Signal	–	●	○	○	–	–	○	●	●	●	●
Skype	300[a] [14]	●	○	●	○	○	●	●	○	○	●
Snapchat	291[a] [14]	●	●	●	●	○	○	○	○	●	●
Telegram	200[a] [14]	●	○	●	○	●	○	●	●	●	●
Threema	–	●	○	○	–	–	○	●	●	○	●
Viber	260[a] [14]	●	○	●	●	●	●	●	●	○	●
WeChat	1058[a] [14]	●	○	●	●	●	○	○	●	○	●
WhatsApp	1500[a] [14]	●	○	●	●	●	○	●	●	○	●
Wire	–	●	○	●	◐	●	●	●	●	○	●

- Messenger: Enumerates the many messaging applications under comparison.
- Monthly active users: Gives an app's approximate monthly active user count.
- Local deletion: Indicates whether users can erase messages locally, meaning on their own device, using the app.
- Local deletion residuals: Indicates whether messages that have been removed still exist on the user's device.
- Global deletion: Indicates whether users can remove messages from all devices on which they are stored using the app.
- Global deletion residuals: Indicates whether messages that have been erased still exist on other devices.
- Separate functions: Indicates whether the program includes distinct features for erasing sent and received messages.
- Edit message: Shows whether the program lets users edit messages that have been sent.
- Quote message: Indicates whether users can respond to or quote passages within a message using the app.
- Received message deletion: Indicates whether the program lets users remove messages they've received.
- Ephemeral messaging: Shows whether the application allows messages to vanish or self-destruct.
- Delete conversation: Comments on whether users are allowed to erase the whole chat box from the application.

2.3 Comparative Analysis

This section addresses some of the major objectives and strategies of the Self Destruct Messenger and compares them to other systems and points out how this system helps to improve upon the existing systems to protect privacy concerns of the users.

Existing Systems:

Features	Dynamic Encryption	Automatic Message Deletion	Real-Time Communication Requirement	Anti-Screenshot Measures	Google/Email Authentication	Backup on Server
Signal	✓	!	✗	✗	✗	✓
Telegram	✓	!	✗	✗	✗	✓
WhatsApp	✓	✗	✗	✗	✗	✓
Botim	✓	✗	✗	✗	✓	✓
Snapchat	✓	!	✗	✓	✓	✓
Our system	✓	✓	✓	✓	✓	✗

There are a number of messaging applications that offer encryption but the deficiency of full confidentiality and protection of data remain.:

- Signal: Provides its users with self-deleting messages and end to end encryption but uses static encryption keys instead which can be easy targets once the session has started.
- WhatsApp: Mostly popular due to its end to end encryption but does not have any form of dynamic encryption and also keeps records of all messages sent and received even after the sessions thus posing a risk of unintended access to the still stored data.
- Telegram: Features "Secret Chats" with end-to-end encryption and self-destructing messages. However, it still allows screenshots, and like many others, does not enforce real-time communication.

Dynamic Encryption:

In every interaction, users have a unique encryption key. This protects users within one session, rather than

across many sessions in which other encrypting platforms, also considering the encrypting device used entails a low risk factor. The Commando Project achieves a much lower risk by ensuring that even if a key is compromised, only data from one session is available.

Real-Time Communication Requirement:
Unlike present applications where messages can be sent and stored for subsequent viewing, the features of Self-Destruct Messenger require both users to be online before they can engage in communicating. Such a live interaction is consistent with the need for provision of high secrecy as such unnecessary risk of information being accessed while stored is minimized.

Anti-Screenshot Measures:
As thanks to the current systems having poor measures against screenshots, users are able to take screenshots and save sensitive information even with the use of encrypted communication. Within the context of combating this vice, Self Destruct Messenger will embed platform API's such as iOS UI Screen Captured Did Change Notification as well Android FLAG_SECURE to counter the phenomena of screenshots and enhance security around information that is exchanged.

Automatic Session-Based Deletion:
For any distinguishing feature of Self Destruct Messenger, all conversations will be deleted at the end of every session because it is an automated function meaning everything is being forgotten, be it on the client side or server side, no traces of any messages remain. This is done in order to further minimize the risk of data theft and further from interception.

2.4 Research Gaps

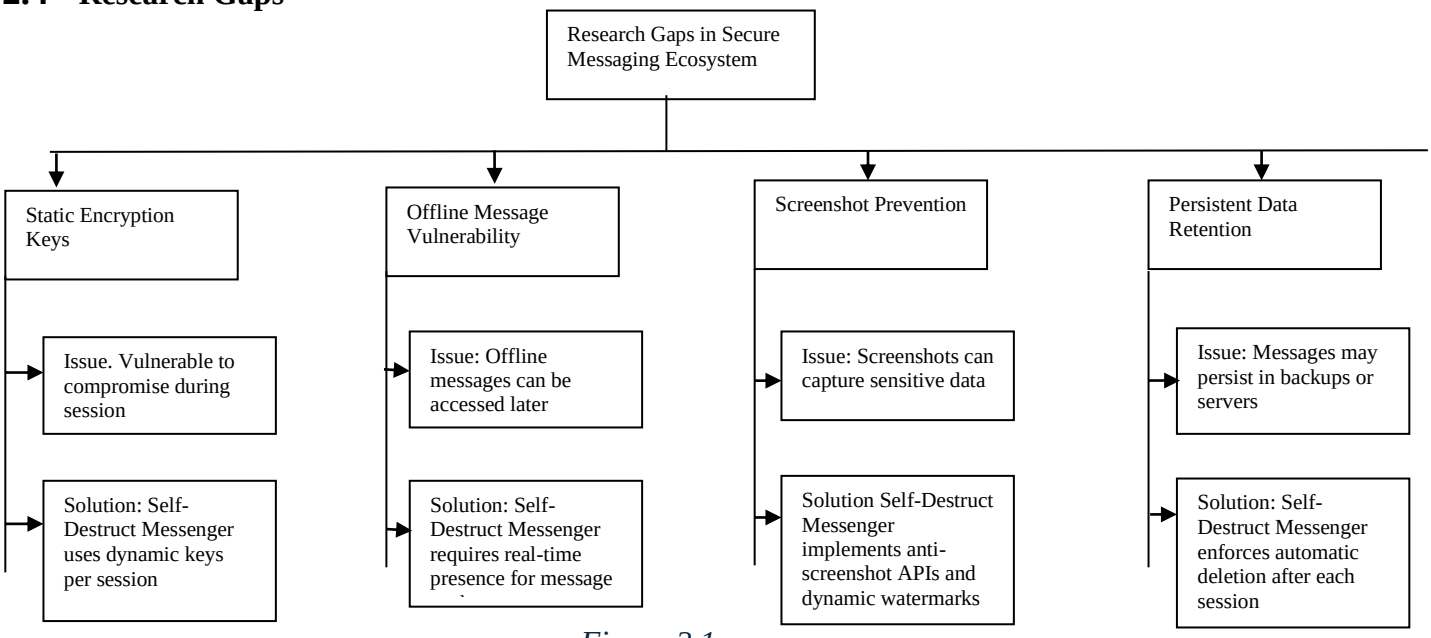


Figure 2.1

Static Encryption Keys:
Across all platforms, the use of static encryption keys within sessions is still vulnerable. These keys have the potential to reveal entire conversations if they are exploited. This is lessened by Self-Destruct Messenger, which isolates the security risk of each chat by creating a unique key for every session.

Vulnerability for Offline Messages:
A lot of platforms allow offline messaging, so users can access messages even when one of them is not online. This function is hazardous because cached messages could be seen or intercepted without the user's awareness, despite its convenience. This risk is reduced by Self-Destruct Messenger's real-time requirement, which makes sure that messages are only sent while both users are online.

Screenshot Prevention:
There is a serious flaw in the few systems that now prohibit screenshots. Sensitive messages that are not encrypted can be revealed via screenshots. Self-Destruct Messenger can avoid screenshots on web and mobile applications by using dynamic watermarks and anti-screenshot APIs.

Persistent Data Retention:
Existing platforms either keep communications on servers until they are specifically erased or permit them to persist in backups. If server data or backup systems are compromised, this poses a privacy concern. In order to solve this, the Self-Destruct Messenger system requires that messages be automatically deleted at the end of each session, leaving no data on servers or devices.

2.5 Problem Statement

An example of this enhanced privacy intrusion can be addressed with the assistance of end to end encryption that has been built into many platforms As apps having this kind of setting are Incomplete due to a number of issues including:

- Static Encryption Keys: Most platforms use a single encryption key for the duration of the discussion, known as "static encryption keys," which puts all data at danger if it is compromised.
- Persistent Messages: If unapproved parties or data breaches gain access to stored messages, there is a serious risk.

- Real-Time Communication Requirement: Offline support for any number of platforms is one of those support, Off course the other person has to be involved in the conversation, otherwise, they are at risk of being compromised.
- Screenshot Vulnerability: Sensitive messages can easily be recorded via screenshots, providing a considerable risk for misuse or illegal release.

2.6 Chapter Summary

This chapter offered an evaluation of the currently available secure messaging systems, identifying the gaps that exist in various aspects including real-time interactions, dynamic messaging communication, anti-screenshot and delete features. It has also attempted to discuss how some of the problems could be resolved. According to the description provided, a Self-Destruct Messenger risks trying to build an irritating messaging platform that has live restrictions on message threads, instant enlargement of sent messages, and automatic removal of those messages after a conversation with the same remove scope. The Self-Destruct Messenger seeks to fill these gaps and set a benchmark for securely communicating with users across multiple industries who would like to secure privacy.

Chapter 3:
Requirements and System Design

Chapter 3: Introduction

3.1 Introduction

Focuses on outlining the fundamental specifications and design tenets of the encrypted messaging app Self-Destruct Messenger, which places a high value on secrecy, security, and privacy. Users want to know that their interactions are safe from data leaks, illegal access, and other security threats as digital communication becomes more and more integrated into both personal and professional lives. This chapter describes the functional and non-functional requirements that influence the Self-Destruct Messenger's fundamental features and performance benchmarks in order to satisfy these expectations. The system's architecture, design schematics, hardware and software requirements, and particular attack scenarios that potentially compromise the system's security are then covered in detail. In order to guarantee thorough threat mitigation, the chapter also examines threat modeling approaches and presents a threat resistance model, outlining the safeguards put in place to protect user data and maintain trust in the system.

In order to meet the operational objectives and security issues particular to a self-destructing, privacy-focused messaging platform, this chapter lays the groundwork for the system's architecture. Most importantly, it captures in detail interactions between users of the system and system components as well as the different requirements in volumes that lead to the deployment of the system. The metrics provided in this chapter, therefore, serve the purpose of helping design a robust platform that helps not only meet the expectations of its users but also reduces several risks related to security and privacy.

3.2 System Architecture

Describe the Self-Destruct Messenger's architecture in general terms. And explain the function of each part, such as the server for message routing, the client for user interaction, and the encryption layer for data protection.

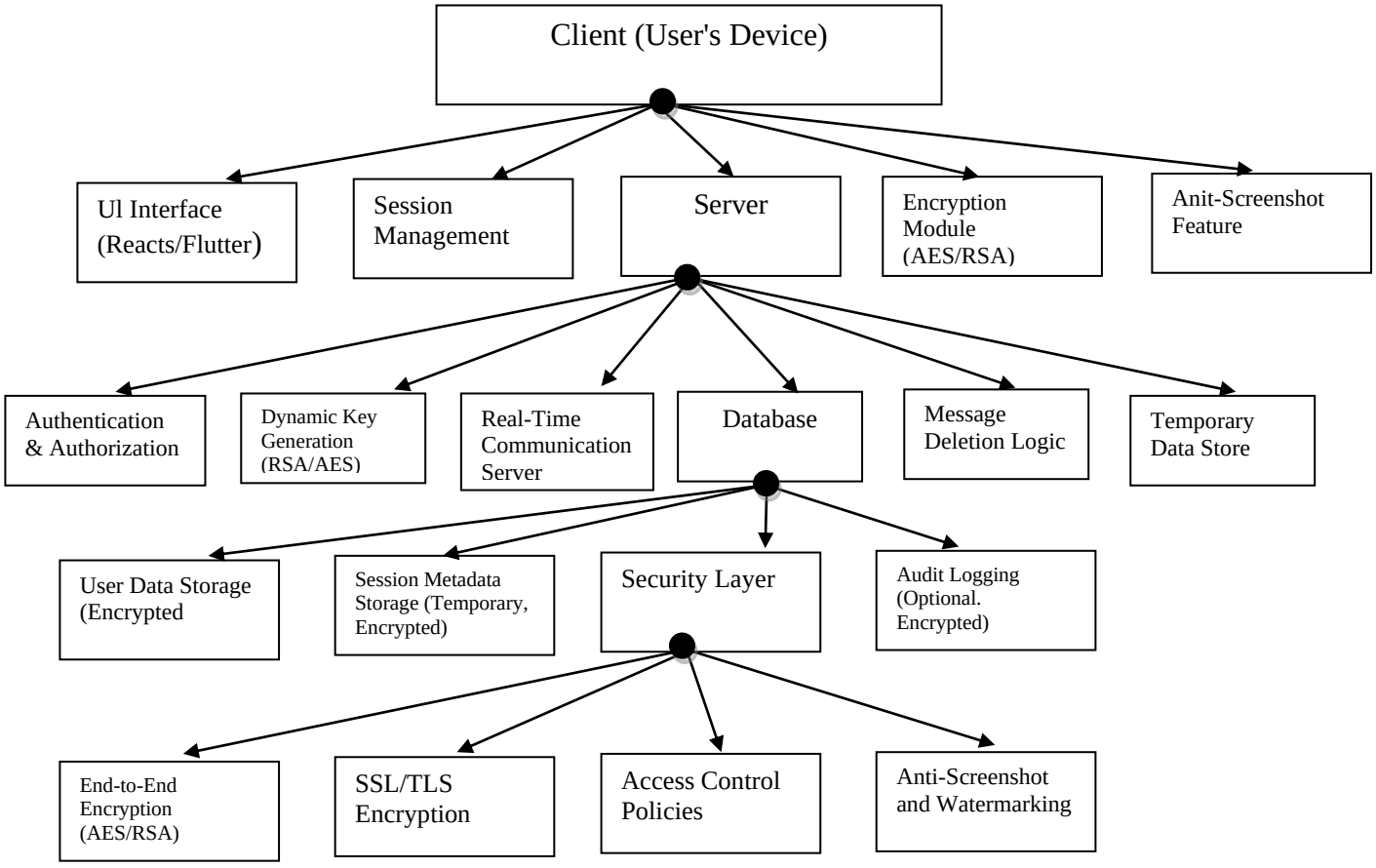


Figure 3.1

3.3 Functional Requirements

- **Goal:**

  Specify the functions that the system should have.

- **Conditions:**

  Users can register for and log in to their accounts.

- **End-to-end encryption**:

Only the devices of the sender and the recipient are able to decode and decrypt messages.

- **Self-Destruct Messages:**
  Messages are set to self-delete after a certain time frame automatically, that has been established by the firm.
- **Message Deletion com Configure:**
  Allow users to delete messages from both sides.
- **Online Presence Detection:**
  Messages may be sent back and forth only if both users are online.
- **Screenshot Prevention:**
  On compatible devices anti capture of images of the screen features.

3.4 Non-Functional Requirements

- **Security:**
  Simply put, compliant data privacy and encryption policies; effective data security measures, and authenticating limitations features.
- **Performance:**
  Users can expect a real time interaction followed by message transmission with a very low delay.
- **Scalability:**
  Such a system is capable of accommodating a continuous increase in the number of users.
- **Reliability:**
  Very high accessibility and up-time statistics.
- **Usability:**
  An easy-to-use interface for sending and receiving messages.

3.5 Design Diagrams

3.5.1 Use Case:

The user starts a secure chat conversation after authenticating to confirm their identity. To preserve anonymity, messages are encrypted and transmitted in real time during the session. To improve security, the system blocks pictures and automatically deletes communications after a predetermined amount of time or occurrence. Either the user or the system can terminate the chat session, guaranteeing that the exchange is safely stopped.

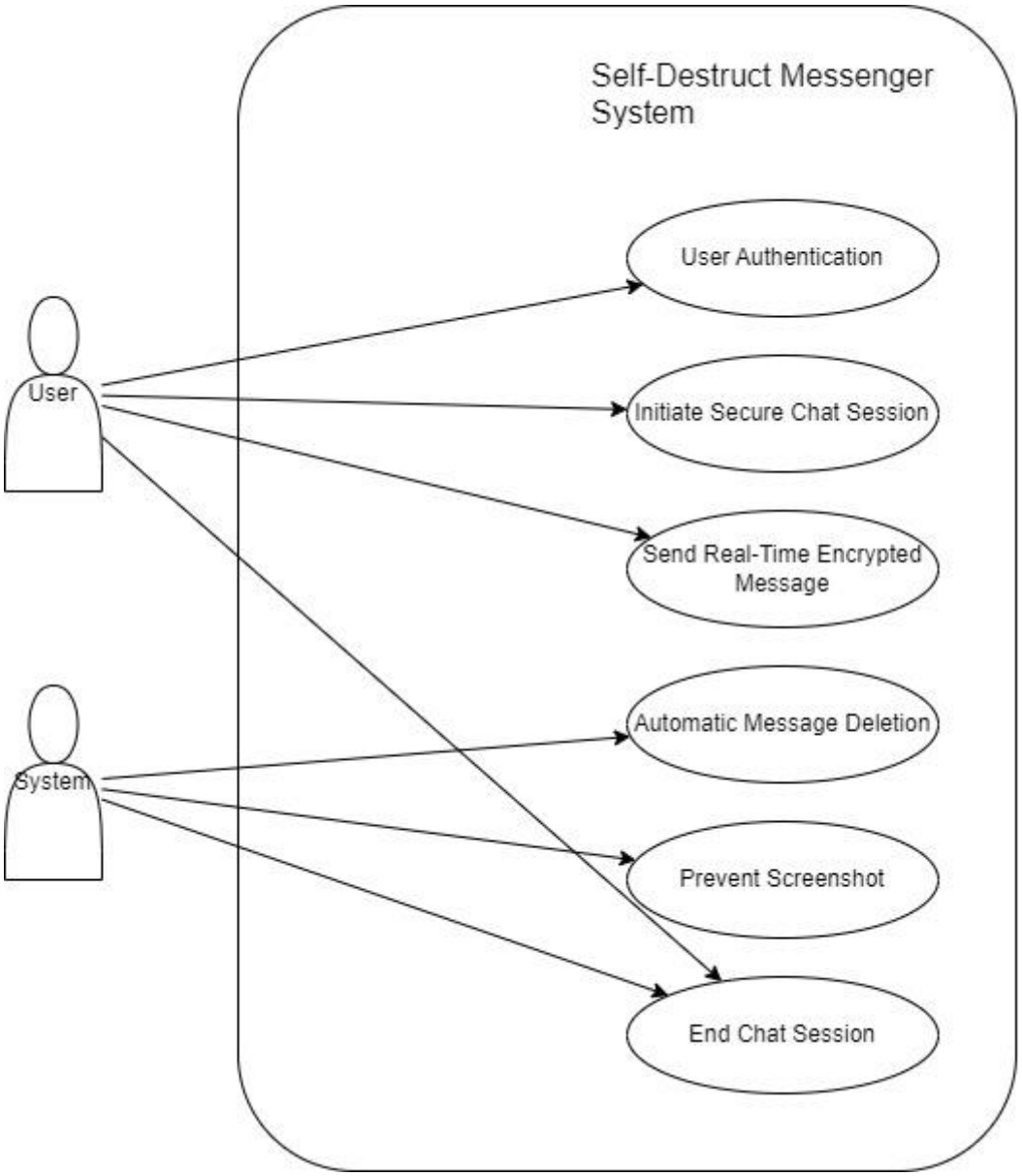


Figure 3.2

3.5.2 Dataflow:

User authentication is the first step in the procedure, during which the user's credentials are checked. The system starts a chat session and saves session details after authentication. After that, a session key is created to allow message encryption. Before being dispatched, the user's encrypted communications are momentarily kept in a message queue. The system automatically deletes messages after they are dispatched, recording the deletions for documentation purposes. With regulated message preservation and deletion, this flow guarantees safe, encrypted communication.

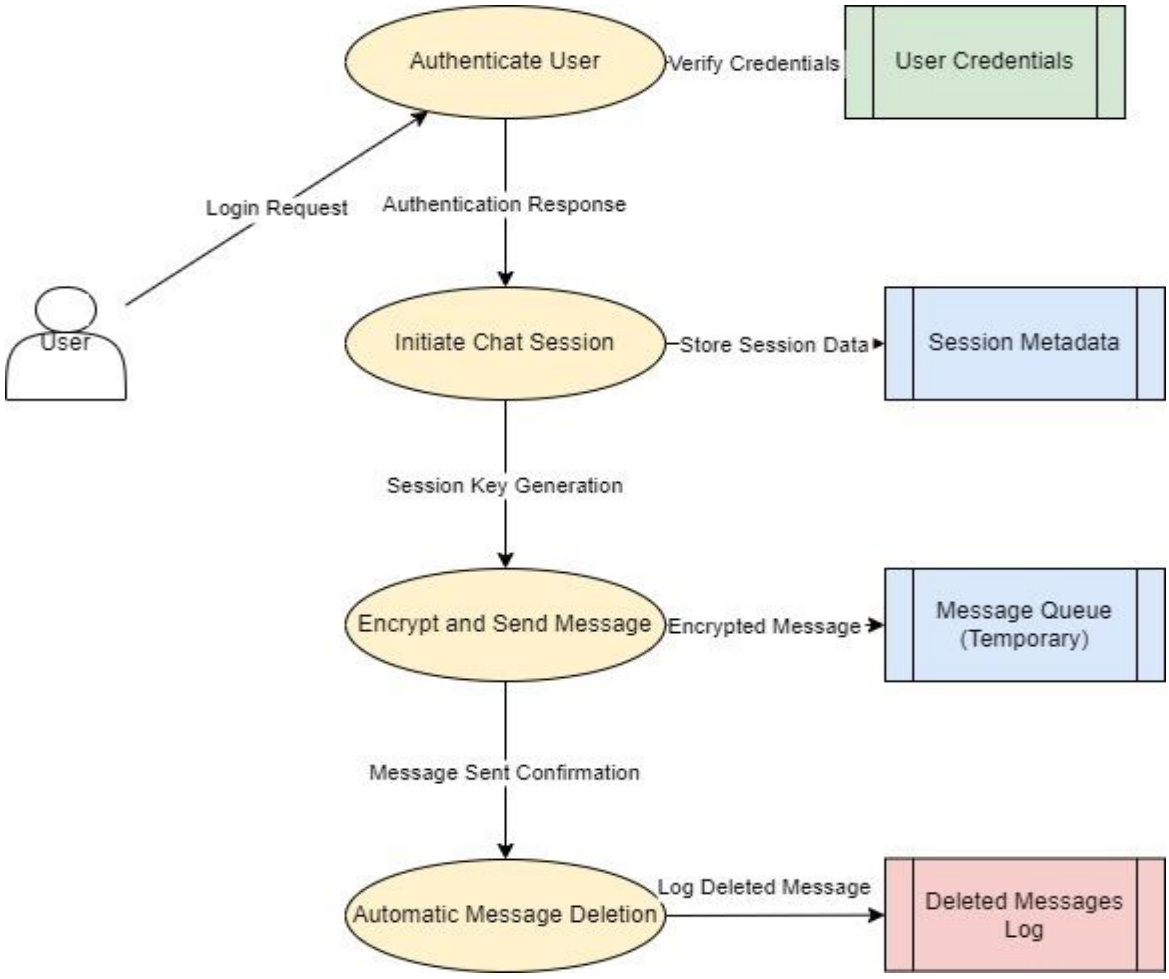


Figure 3.3

3.6 Hardware and Software Requirements

3.6.1 Hardware Requirements

Development Machine Requirements

For efficient app development and testing:

- Processor: Quad-core or higher, for smooth handling of IDEs and emulators.
- RAM: Minimum of 8 GB, recommended 16 GB for multitasking.
- Storage: At least 10 GB of free space for SDKs, libraries, IDEs, and project files.
- Operating System: Windows, macOS, or Linux. **macOS** is required for iOS development due to iOS build and simulator limitations.

Device Requirements for Testing

To ensure broad compatibility, the app should be tested on both physical devices and emulators:

- **Physical Devices**:
    - Android: Version 5.0 (Lollipop) or higher.
    - iOS: Version 11.0 or higher.
- **Emulators**:
    - Android Virtual Device (AVD) and iOS Simulator to replicate various environments for compatibility testing.

End-User Device Requirements

For optimal performance, user devices should meet the following specifications:

- **Operating System**:
    - Android: Version 8.0 (Oreo) or higher.
    - iOS: Version 11.0 or higher.
- **Hardware**:
    - Processor: Octa-core for Android, A10 chip or higher for iOS.
    - RAM: Minimum of 3 GB to handle encryption and multitasking.
    - Storage: 200 MB of free space for app data.
    - Display: 720p or higher for clear UI.
- **Connectivity**:
    - Network: Reliable internet connection (Wi-Fi or 4G/5G) for real-time messaging.
    - Bluetooth (optional): For additional device pairing if needed.

Security Features (Optional)

For enhanced security:

- Biometric Security**:** Fingerprint or Face ID for app access.

- Secure Enclave: Recommended for iOS, to safely store sensitive data.

3.6.2 Software Requirements:

Development Environment

- **Flutter SDK**:
    - The project requires the latest stable version of Flutter SDK for cross-platform development. By leveraging Flutter, a developer gets the same experience on both iOS and Android thus allowing rapid application development.
- **Dart SDK**:
    - Flutter is packaged with the Dart SDK that is important for building Flutter apps. The programming language Darts provides its features to write asynchronous programs which can be used for real-time chat.
- **IDE**:
    - Some popular Integrated Development Environments (IDEs) to consider would be: Visual Studio Code, Android Studio, or IntelliJ IDEA. These IDEs need to have plugins for Flutter and Dart for better coding, debugging, and testing.

Dependencies & Libraries

- **Encryption Libraries**:
    - Pointy Castle**:** This library is crucial for implementing RSA and AES encryption in Dart, securing messages through robust encryption algorithms.
    - flutter_secure_storage: This library is used to securely store keys on the device, enhancing data protection by ensuring that encryption keys are stored in a secure environment.
- **WebSocket Communication**:
    - web_socket_channel**:** A Dart library that facilitates WebSocket connections for real-time messaging. WebSocket channels allow efficient

data exchange between users and the server, enabling instant message delivery.

- **Backend Framework**:
    - Node.js with Express or Firebase. Although not essential, a backend architecture aids with immediate messaging and data management. This would allow for the use of Node.js with Express where a tailored backend is needed. Alternatively, Firebase will also offer an off the shelf backend which is easily expandable.

Testing and Debugging

- **Emulators and Simulators**:
    - Android Emulator: Used to test the Android version of the application.
    - iOS Simulator: Required for testing the iOS version of the application. This is particularly important for ensuring cross-platform compatibility.
- **Security Testing Tools**:
    - OWASP ZAP and Burp Suite: These tools can be used optionally for penetration testing, helping to identify potential security vulnerabilities in the application.

Operating System

- **Windows, macOS, or Linux**:
    - Flutter development is compatible with Windows, macOS, and Linux. However, macOS is required if the developer plans to build and test the application for iOS, due to Apple's restrictions on iOS app development.

3.7 Threat Scenarios

1. **Scenario 1:**
   MFA and strong password restrictions are used to prevent unauthorized access attempts.

2. **Scenario 2:**

   SSL/TLS encryption prevents a man-in-the-middle attack.
3. **Scenario 3:**

   Secure session storage and transient tokens prevent session hijacking.
4. **Scenario 4:**

   Using anti-screenshot tools to capture malware screenshots.

3.8 Threat Modeling Techniques

STRIDE:

Examine possible dangers in all areas (e.g., spoofing, tampering, and repudiation).

DREAD:

Sort hazards according to their potential for harm, reproducibility, exploitation, etc.

LINDDUN:

Take care of privacy issues (identifiability, linkability, etc.).

Attack Trees:

Showcase attack routes to identify weaknesses.

3.9 Threat Resistance Model

Key Threats:

List the main dangers, including MitM attacks, illegal access, screenshots, and more.

Reductions:

1. Use secure authentication and MFA as access controls.
2. End-to-end encryption should be used.
3. Data management: Implement deletion and self-destruct procedures.
4. Use real-time only messaging and anti-screenshot software as privacy precautions.

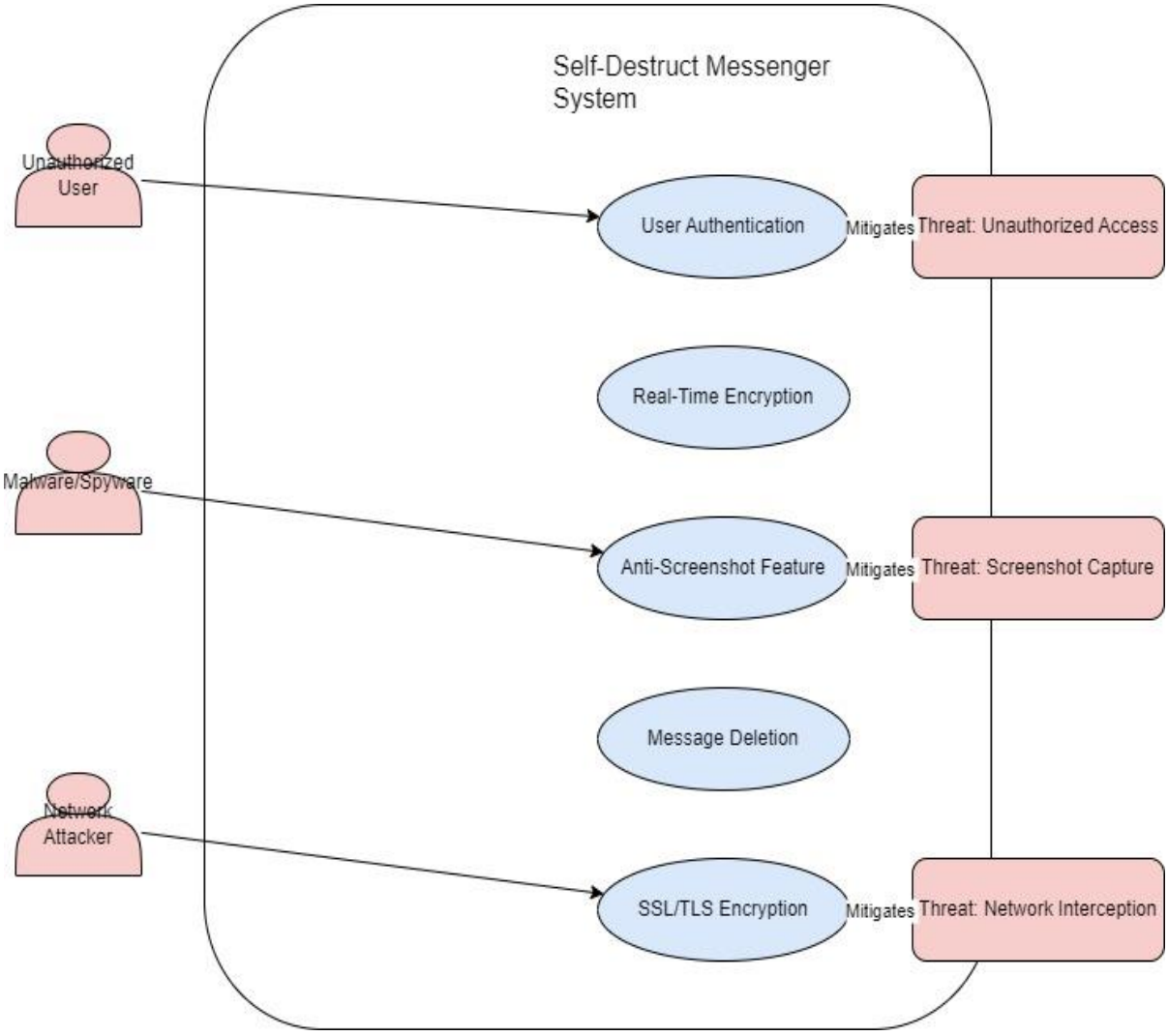


Figure 3.4

3.10 Chapter Summary

A detailed examination of the Self-Destruct Messenger's specifications and design was given in Chapter 3. This chapter provides a thorough framework for creating a safe, dependable, and user-friendly messaging application by addressing both functional and non-functional needs, building a secure architecture, and implementing strict threat analysis and mitigation procedures. This framework will direct future development stages

and guarantee that all features of the Self-Destruct Messenger meet the strict security and privacy requirements set out in the current digital communication environment.

Reference

[1] Schnitzler, Theodor, et al. "Exploring user perceptions of deletion in mobile instant messaging applications." *Journal of Cybersecurity* 6.1 (2020): tyz016.

[2] Heath, Howard, Áine MacDermott, and Alex Akinbi. "Forensic analysis of ephemeral messaging applications: Disappearing messages or evidential data?." *Forensic Science International: Digital Investigation* 46 (2023): 301585.

[3] Qiwei, Li, et al. "Feminist Interaction Techniques: Deterring Non-Consensual Screenshots with Interaction Techniques." *arXiv preprint arXiv:2404.18867* (2024).

[4] https://www.slashgear.com/769717/snapchat-is-not-private-nor-is-it-safe/

[5] https://www.whatsapp.com/legal/privacy-policy-eea

[6] https://telegram.org/privacy?setln

GitHub

https://github.com/HAMZAZAWARI2/selfdm

GitHub is a cloud operated repository that works using Git which is an SCM, for better assistance in working on software projects as a team. It enables software developers to host, control and monitor changes for their code repositories. Moreover, it is a user-friendly repository with built GIT making it appealing for almost any developer level.

How GitHub Helps in a Software Developer's Life

- **Version Control:** One other wonderful feature GitHub has to offer is the ability for developers to work on the same code and track the modifications made. This cuts down the chances of losing work as developers can switch back to older versions of the code with little stress.
- **Collaboration:** Several developers can engage in a single project in parallel without erasing other users work. GitHub allows the use of branches; therefore, if a developer wants to work on a new feature or fix a bug, they can create a new branch, work on it, then merge it to the main branch after completion.
- **Code Review:** Code reviewing is an important concept in managing projects that GitHub has streamlined by using pull requests. In this system, developers are allowed to make changes, and team members can review those changes, comment, and give feedback before those changes are merged in the main branch. This helps not only to enforce higher code quality but also encourages knowledge sharing among team members.
- **Issue Tracking:** GitHub can Issue Tracking that allows any developer to submit bugs, feature requests or even tasks. This helps teams in managing their workloads and helps them as well in getting a good understanding of the overall project.
- **Documentation:** Developers are able to create a Markdown file or a GitHub wiki page which allows them to generate a good level of documentation within their projects. This documentation includes but is not limited to installation, usage, and contribution procedures which aids new contributors easily join the project.

- **Continuous Integration/Continuous Deployment (CI/CD):** There are many CI/CD in built tools within GitHub that allow deploying the application automatically after the testing phase. This means that all updates made to the code and how the different aspects of the application interact with one another are verified post they were submitted.
- **Community and Open Source:** Community and Open Source: GitHub contains a lot of open-source projects that give opportunities to developers to work on projects that are already started, learn from other projects, and show their results to future employers.

Managing and Using GitHub Regularly During the Project

Throughout the project, the way in which GitHub was handled and applied was as follows:

- **Repository Creation:** The project "Self-Destruct Messenger" had a specific GitHub repository created that acted as the main source for all documents gearing towards the project.
- **Regular Commits:** Systematic documentation of the project history was catered for in the project by encouraging members of the team to commit changes regularly with proper descriptive messages. This enabled the ease of tracing any change made in the project.
- **Pull Requests:** The team submitted pull requests before making changes to the main branch. There were instances when one of the team members needed to assess the code, comment on it, and hold a meeting on how to make the code better, thereby improving the overall quality of the code.
- **Documentation Updates:** Adaptation of a project was reflected on the website, and relevant documentation also was included in the repository in a timely manner. This included documentation such as the steps performed during installation, how to use the product, and how the project was developed. In this way, all the team members and the future collaborators were provided with the relevant and recent updates.
- **Regular Meetings and Updates:** Regular meetings were held for the purpose of discussing the progress that has been made so far, reviewing pull requests, and solving problems that team members face or had faced at that time.