Key Exchange Outside of TCP/IPasdfasdfasdf

Design Document

Group: SDMAY20-52

Client/Faculty Advisor: Julie Rursch

Jacob Moody
Jack Potter
Andre Chickering
Jordan Svoboda
Joel Wacker
Logan Woolery

December 3rd, 2019

Executive Summary

Development Standards and Practices Used

Summary of Requirements

This project has fairly minimal hardware and software requirements necessary to produce our final product. The hardware requirements for this project consist of computers on which we will perform development work, a server that will host the backend infrastructure for our project, and several cell phones on which the application we develop can be deployed for testing purposes. The software requirements for the development of our project can be broken down into the integrated development environments (IDEs) necessary to build and deploy our application as well as the infrastructure software required for deploying and testing our backend code including a hypervisor and firewall.

The computers necessary for the development of our project will primarily consist of our own personal laptops and desktops, which will be largely be sufficient. However, we have found the need to source an additional Apple laptop/desktop of any kind, as there are numerous issues preventing the development of iOS applications on anything other than a computer running OSX. The server hardware necessary for hosting our project's backend will also likely consist of our own server(s). While we initially approached the university to request a virtual private server or a world-routable IP address that could forward traffic to a server we would provide, we encountered difficulties that rendered pursuing the issue untenable. Instead, we will simply use our own servers which already have hypervisors installed that will make the deployment of a new virtual server very straightforward. Finally, between the members of the group, we have a selection of various iPhone and Android cell phones, which will allow testing across numerous platforms.

All of the software required for the development of this project is free and/or open source, eliminating software costs as a constraint on our project. The Android Studio and XCode IDEs will be used for the development and deployment of the phone applications, and team members will be free to utilize their own editors and workflows for the rest of the project.

Applicable Courses from ISU Curriculum

We will draw from our experiences in the following courses throughout the completion of this project:

- Computer Science 309
  - Software development best practices
    - Version control using Git
    - Continuous integration using Git/Jenkins
    - Test driven development with unit testing
  - Android application development
    - Use of Android Studio IDE
    - Integration of external Java libraries
  - Client/server communications
    - Restful APIs
    - WebSockets
    - Client login and authentication
- Computer Engineering 308
  - Low-level coding practices
    - Multithreaded application development
    - Thread-safe programming
    - Understanding of Unix system calls
    - Network communications at low levels
- Computer Science 228
  - Data Structures
    - Storing messages, conversations, users efficiently
    - Ensure messages and conversations can be easily searched
    - Store key hashes efficiently
  - Algorithms
    - Searching conversations/messages
    - Connecting users for conversations
    - Generating/sharing keys quickly

- Computer Engineering 234X
    - Ethical Considerations
        - Ensuring we have no access to user messages/keys
    - Legal Considerations
        - Exporting cryptography suites internationally
        - Responses to law enforcement requests
- Computer Engineering 430/530
    - Network protocols and security
        - Network stacks
        - Protocol weaknesses and securities

New Skills/Knowledge acquired that was not taught in courses

The two primary technologies that we will use that have not been taught in a class before are Golang and Flutter. Golang is an extremely versatile language developed by Google that will make up the majority of our back-end code. We chose to use Golang for several reasons. First of all, it is very extensible and makes it very easy to implement open source libraries for cryptography and secure network communications that have been thoroughly audited to ensure efficient and secure performance. Second, Golang makes developing test suites in real time very straightforward, and it contains a number of useful utilities to ensure complete coverage of all code by the test suites. Finally, Golang can easily export executable builds of our project that can be run on nearly any architecture or operating system very easily.

We will be using Flutter for the development of our client applications. Flutter is a Google project that allows mobile applications to be developed once in the Dart language and then built into native executables for both Android and iOS. We chose Flutter as it will ensure that all versions of our client application will feel and work similarly, and while we may need to do some manual tweaking for each platform, it will generally increase the efficiency of our development and ensure all features are consistent across both applications.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Acknowledgement

We are grateful for the opportunity to work on a project envisioned, researched and designed by our group. As we are not professional engineers yet, it is a gift to be able to work with other students who have a similar desire to learn about and implement a topic that we share a passion for. Thank you to Iowa State University for allowing us to pursue this project and providing us the support necessary to complete the project. Thank you also to Dr. Rursch for her mentorship throughout this process.

1.2 Problem and Project Statement

1.3 Operation Environment

Our messaging app is intended to operate on any iOS or Android platform. This means that it could be used in any part of the world or any time of day, so we need to keep it uniform across all devices. The fact that it could be used in any part of the world means that the security of our app must be as strong as possible so that no one (from a single hacker sitting in a coffee shop to a hostile foreign government) is able to crack the encryption and read the messages that are being sent. In relation to time of day, we need to ensure that the app is up and running whenever a user wishes to send or receive a message.

1.4 Requirements

1.5 Intended Users and Uses

1.6 Assumptions and Limitations

1.7 Expected End Product and Deliverables

2 Specifications and Analysis

2.1 Proposed Design

2.2 Design Analysis

2.3 Development Process

2.4 Design Plan

3 Statement of Work

3.1 Previous Work and Literature

3.2 Technology Considerations

3.3 Task Decomposition

3.4 Possible Risks and Risk Management

3.5 Project Proposed Milestones and Evaluation Criteria

3.6 Project Tracking Procedures

3.7 Expected Results and Validation

4 Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline

4.2 Feasibility Assessment

4.3 Personnel Effort Requirements

Table 1: Task/Time Breakdown

Task	**Time**
Requirements	36 Hours
Specifications	24 Hours
Prototyping	80 hrs
Development	180 hrs
Stabilization	30 hrs
Final Release	30 hrs

4.4 Other Resource Requirements

This project will require development and testing for our clients and servers. The iOS and Android clients will be tested on an iOS and an Android device respectively, and we have the devices necessary to develop and test our project. We have independently sourced a MacBook to assist in the building and deploying of the iOS client code.

4.5 Financial Requirements

As this entire project will be developed and completed with free and/or open source software run on hardware that is already owned by our group, we do not anticipate any financial costs at this time. Group members will use their own computers to develop the project, and we have borrowed a MacBook to assist in building and deploying iOS code. The project will be deployed on server hardware and phones owned by various group members.

5 Testing and Implementation

5.1 Interface Specification

5.2 Hardware and Software

5.3 Functional Testing

5.4 Non-Functional Testing

5.5 Process

5.6 Results

6 Closing Material

6.1 Conclusion

6.2 References

6.3 Appendices