Software Engineering- CSC 4350

Spring 2017

2/5/2017

ADEPT

Amani Konduru

Benjamin Garber (**D**aniel)

Edward Bull

Paul David Utesch

Team

Table of Contents

I. Introduction

Topic: An encryption and decryption system for message communication

We will have two layers of encryption. The first will be basic SSL for encrypted communications. This is already implemented in Java and is as simple as creating an SSLSocket. The second layer will be message layer encryption, and this is unlocked with a user's password and a key. The messages stored on disk will always be encrypted until they are viewed by a user. This gives us a chance to implement some encryption algorithms ourselves rather than relying on a Java library.

There would be two components to the program. A server component, and a client component. The server would always have a listening socket. Any time a connection is initiated, this will spawn off a thread that handles the new connection, while the main thread will continue to listen for further connections. This way we can handle multiple connections at the same time. Once a message is received, the metadata will be used to store the encrypted message in a simple PostgreSQL database, with the owner as a primary key. We might be able to use the SMTPS protocol for server exchanges. The client would be a program that on run and periodically afterward would connect to a designated server and use either an IMAPS or POP3S protocol to request a sync for a particular user. The server needs to know how to handle both SMTPS and IMAPS or POP3S interactions. The client program should have some basic login functionality and maybe a SQLite database to store usernames and a hashed/salted password. If this isn't enough to satisfy the requirements, we would implement a simple GUI with spare work cycles.

For testing purposes, we would require a third program for the professor to run. This program would install the server component with its database, and the client component with its database, to a local folder and then run both. The client would be pre-configured to simply connect to local host as its server with whatever port the server is designated to listen on. The tester could then create two test accounts, send an e-mail from one to the other, and make sure it works.

II. "shall" statements

1.0
The Adept Mail Server shall store user emails in a database.

1.1
The Adept Mail Server shall move user emails between user-designated mailboxes upon an authenticated request from that user.

1.2
The Adept Mail Server shall delete user-designated emails from its database upon an authenticated request from that user.

1.3
The Adept Mail Server shall serve user data when authenticated requests are received from the Adept Mail Client via a minimally compliant IMAP protocol.

1.4
The Adept Mail Server shall send user emails from other Adept Mail Servers upon an authenticated request from that user via a minimally compliant SMTP protocol.

1.5
The Adept Mail Server shall receive user emails from other Adept Mail Servers via a minimally compliant SMTP protocol.

1.6
The Adept Mail Server shall encrypt all incoming and outgoing connections using the TLS 1.2 standard.

1.7
The Adept Mail Server shall support multiple concurrent connections.

1.8

The Adept Mail Client shall request user email data from the Adept Mail Server via a minimally compliant IMAP protocol.

1.9

The Adept Mail Client shall store user email data locally in a local database.

2.0

The Adept Mail Client shall send user emails to the Adept Mail Server via a minimally compliant SMTP protocol.

2.1

The Adept Mail Client shall provide a graphical user interface to allow users to generate requests and view their emails.

2.2

The Adept Mail Client shall require local authentication from any user before executing local requests.

2.3

The Adept Mail Client shall provide remote authentication to the Adept Mail Server prior to executing any requests.

2.4

The Adept Mail Client shall locally encrypt and decrypt the subject and body of every email it sends and receives, respectively, using symmetric-key block encryption based on a user provided password. Decryption shall occur only upon viewing within the Adept Mail Client itself, and decrypted documents shall not be stored.

III. Requirements Traceability Matrix (RTM)

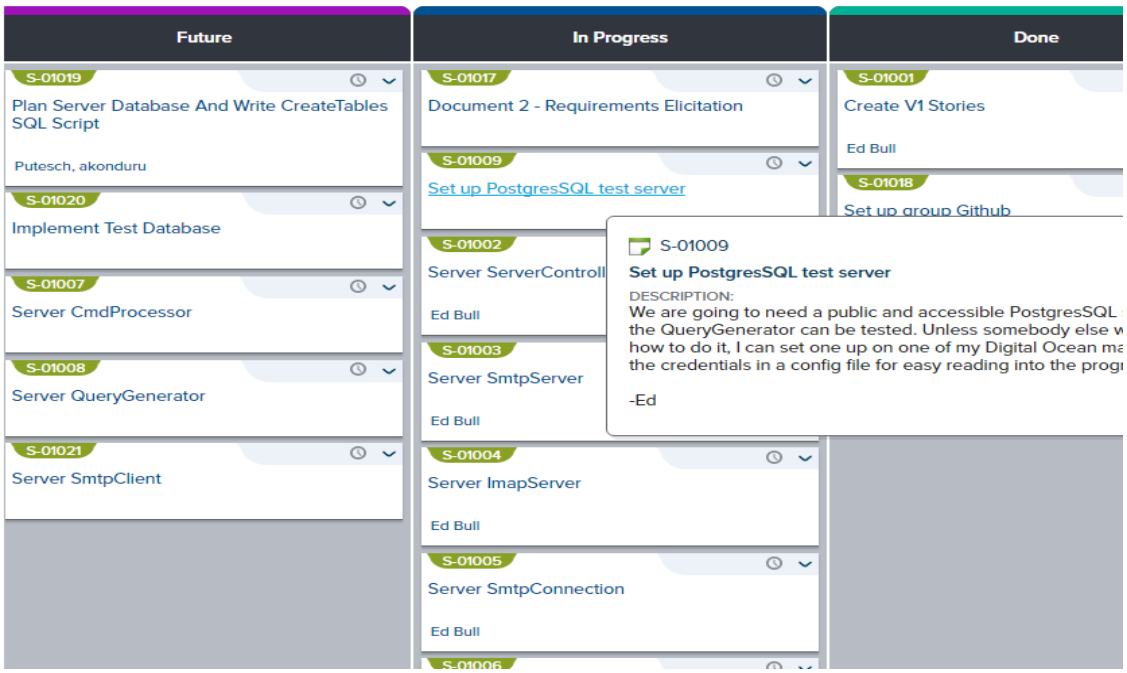
Entry #	Para-graph	System Specification text	Type	Build	Use Case Name	Category
1	1.0	The Adept Mail Server shall store user e-mails in a database.	SW	1	TBA - Document #3	Server
2	1.1	The Adept Mail Server shall move user e-mails between user-designated mailboxes upon an authenticated request from that user.	SW	2	TBA - Document #3	Server
4	1.2	The Adept Mail Server shall delete user-designated e-mails from its database upon an authenticated request from that user.	SW	2	TBA - Document #3	Server
5	1.3	The Adept Mail Server shall serve user data when authenticated requests are received from the Adept Mail Client via a minimally compliant IMAP protocol.	SW	1	TBA - Document #3	Server
6	1.4	The Adept Mail Server shall send user emails from other Adept Mail Servers upon an authenticated request from that user via a minimally compliant SMTP protocol.	SW	2	TBA - Document #3	Server
7	1.5	The Adept Mail Server shall receive user emails from other Adept Mail Servers via a minimally compliant SMTP protocol.	SW	2	TBA - Document #3	Server
8	1.6	The Adept Mail Server shall encrypt all incoming and outgoing connections using the TLS 1.2 standard.	SW	1	TBA - Document #3	Server
9	1.7	The Adept Mail Server shall support multiple concurrent connections.		1	TBA - Document #3	Server

10	1.8	The Adept Mail Client shall request user email data from the Adept Mail Server via a minimally compliant IMAP protocol.	SW	1	TBA - Document #3	Client
11	1.9	The Adept Mail Client shall store user email data locally in a local database.	SW	1	TBA - Document #3	Client
12	2.0	The Adept Mail Client shall send user emails to the Adept Mail Server via a minimally compliant SMTP protocol.	SW	1	TBA - Document #3	Client
13	2.1	The Adept Mail Client shall provide a graphical user interface to allow users to generate requests and view their emails.	SW	2	TBA - Document #3	Client
14	2.2	The Adept Mail Client shall require local authentication from any user before executing local requests.	SW	1	TBA - Document #3	Client
15	2.3	The Adept Mail Client shall provide remote authentication to the Adept Mail Server prior to executing any requests.	SW	1	TBA - Document #3	Client
16	2.4	The Adept Mail Client shall locally encrypt and decrypt the subject and body of every email it sends and receives, respectively, using symmetric-key block encryption based on a user provided password.	SW	2	TBA - Document #3	Client

IV. Updated WSD

Name	Role
Amani Konduru	Project manager and tester -Document handler -Java Programmer -Database Structure (PostgresSQL schemas)
Benjamin Garber (Daniel)	Developer and programmer -Servers -Java Programmer
Edward Bull	Programmer and tester -Java Programmer -GUI tester
Paul David Utesch	Developer and programmer -Java Programmer

V. Gantt Chart



VI. Glossary

1. **Encryption:** the process of converting data into a code, to prevent unauthorized access. Encryption is the process of transforming data into an unreadable, encrypted form. The transformation is done using one of several cryptographic algorithms that leverage computationally difficult mathematical problems to make reversing the transformation difficult if not effectively impossible

2. **Symmetric Encryption**: Symmetric Encryption uses a key or set of keys to both encrypt and decrypt data. If data is to be shared between two parties, they must both have the key or keys to decrypt or encrypt the data.

3. **Asymmetric Encryption:** Asymmetric Encryption, also known as Public Key Encryption, is a type of encryption where anyone in possession of a public key can encrypt a message. That message can then only be decrypted with a private key. This method is often used for identity authentication because it is computationally expensive. Once authentication is completed, communications will then often transition into symmetric encryption after generating a symmetric encryption key.

4. **End to End Encryptio**n: Only the communicating users can read the messages.

5. **SSL/TLS (Secure Sockets Layer / Transport Layer Protocol):** TLS and the now-deprecated SSL it is based on are network security protocols meant to secure client-server connections using both symmetric encryption for data transfer and asymmetric encryption for identity authentication. While there are many options that can be set in an SSL/TLS session, the foundation of the protocols lie in using encryption to authenticate the identities of the connected parties and to secure the privacy of the data transferred between them.

6. **Server:** a server program awaits and fulfills requests from client programs, which may be running in the same or different computers.

7. **Client:** requesting program or user.

8. **Socket:** Is one endpoint of a two-way communication link between two programs running on the network. A socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent to.

9. **SMTP protocol:** Simple Mail Transfer Protocol. It is an Internet standard for electronic mail (email) transmission. SMTP was first defined by RFC 821 and updated in RFC 5321.

10. **IMAP protocol:** Internet Message Access Protocol. Itis an Internet standard protocol used by e-mail clients to retrieve e-mail messages from a mail server over a TCP/IP connection. IMAP is defined by RFC 3501.

11. **TCP/IP :** IP (Internet Protocol) is the basic communication language or protocol of the Internet. It can also be used as a communications protocol in a private network (either an intranet or an extranet). TCP (Transmission Control Protocol) is layered on top of IP to provide certain network control and data validation features for many internet communications.

VII. Conclusion

Almost every large-scale modern program must, at some point or another, communicate over a network. Likewise, almost every large-scale modern program must manipulate data and typically does so using a database. And while there may have been some leeway on security in the history of computer science, secure network connections and data storage are today non-negotiable. Selecting a project like an e-mail server implementation allows us to familiarize ourselves with the server-client relationship, basic networking principles, and some of the basic security measures that all networked programs must implement to send, receive, and store data in a responsible manner. We will gain experience reading, understanding, and implementing common networking protocols like SMTP and IMAP. We will also gain some exposure to database modeling and how to programmatically interface with the databases we create, both PostgreSQL for the server to support concurrency, and SQLite for the client. We will also work on a graphical user interface for our client program and will have a chance to think about user experience and UX design. We will learn the basics of both local symmetric key encryption and the asymmetric encryption algorithms used in network communication like SSL/TLS.

Even though the scope of a one-semester project may not allow us to fully implement each protocol or allow us to fully secure our network communications, a minimally-compliant program such as the Adept Server and Client will allow us to gain a perspective on every aspect of a modern program and apply some concepts which in other classes might have remained only theoretical.