Institute of Computer Science

Bachelor’s Thesis

Investigations on Methods for Secure Key Management
And
Access to Cryptographic Libraries

Berker Vardarsuyu

Bachelor’s and Master’s Theses
of the Center for Computational Sciences
at the Georg-August-Universität Göttingen
Georg-August-Universität Göttingen

Institute of Computer Science

Goldschmidtstraße 7

37077 Göttingen

Germany

First Supervisor: Dr. Lena Wiese

Second Supervisor: Dr. Mehmet Gueltas
I hereby declare that I have written this thesis independently without any help from others and without the use of documents or aids other than those stated. I have mentioned all used sources and cited them correctly according to established academic citation rules.

Göttingen, 2016
Abstract

Information has been one of the most valuable things from the beginning of mankind and it is often a private thing but also supposed to be shared. But there has always been one problem; people have a desire to steal valuable things. It makes the sharing process more complicated. As a result, some security preventions have to occur in this process.

Cryptography is a mathematical method to secure information. There is a deep and complicated history about cryptography. Since community and communication birthed, cryptography took place in the cycle of human life and it has evolved in direct proportion with the evolution of communication methods.

Using cryptography was not common in society. For decades, only people who played an important role for the community needed to use cryptography. But the evolution of communication together with the increasing extent of information sharing created a big demand on cryptography. Therefore, nowadays many people need to secure information transfer methods to save their valuable information from being stolen.

People live in a digital age; conventional communication methods are not as common as digital ones. While the complexity of cryptographic methods and the demand for secure information transfer are growing, the requirement of users for solid software with a user friendly interface is also increasing.
Content

1. Introduction .................................................................................................................. 2
   1.1. Modern Information Exchanging ............................................................................ 2
   1.2. Story of Electronic Mail ........................................................................................... 2
   1.3. Thesis ...................................................................................................................... 4
2. Foundation ....................................................................................................................... 5
   2.1. Cryptography .......................................................................................................... 5
   2.2. PGP-Pretty Good Privacy and OpenPGP ................................................................. 6
   2.3. Key Storage ............................................................................................................ 7
   2.4. Mailvelope ............................................................................................................. 10
      a. Project Structure of Mailvelope ............................................................................... 10
      b. Cryptographic Background of Mailvelope ............................................................. 13
      c. Key Storage on Mailvelope .................................................................................... 15
      d. GUI of Mailvelope ................................................................................................ 16
3. Methods .......................................................................................................................... 21
   3.1. PHP and Java Methods ........................................................................................... 22
   3.2. JavaScript Method .................................................................................................. 27
   3.3. Java Method .......................................................................................................... 28
4. Result .................................................................................................................................. 30
   4.1. PHP and Java .......................................................................................................... 31
   4.2. JavaScript ............................................................................................................... 32
5. Conclusion and Discussion ............................................................................................. 32
List of Figures ....................................................................................................................... 35
List of Acronyms .................................................................................................................. 36
Bibliography ......................................................................................................................... 38
1. Introduction

1.1. Modern Information Exchanging

The internet ushered in a new era. After the discovery of the internet, universalization of information exchanging evolved faster than anything. In the end, everything began with the goal of data exchange.

Nowadays, over 3 Billion people, which are more than forty percent of the world population, use the internet. [1] All of them in a way exchange data. Some of them pulling data from databases of web pages, others of them share their information in private conversations.

Electronic mail is the first common private communication method for internet. As a report from the end of 2015 shows, almost 2.6 billion people use electronic mail. It means 80% of internet users use electronic mail for conversation. Also the users have mostly more than one account. The number of e-mail accounts in the whole world is about 4.5 billion. [2]

The numbers are huge and with direct proportion to that the threat of online privacy being violated is also enormous. Every growth creates a demand behind that. Security preventions are prior and important demands. In shadow of these demands, cryptographic algorithms are developed and integrated into security software.

1.2. Story of Electronic Mail

Communication has a deep history. With every step of humanity communication also got an evolution. Most common of traditional communication ways were telegraph and letters. Nowadays, we still use these methods to communicate. But in 1972, the electronic-mail was created during a military experiment and opened a new path for communication. The first electronic-mail communication was performed in 1972 between two machines by Ray Tomlinsin. That was the birth of electronic-mail. But it was not yet adapted for end users. Over the years, electronic-mail has merged together with other softwares and has been improved. In 1993, internet and e-mail systems came together. The first electronic mail standard was “Simple Message Transfer Protocol (SMTP)” .SMTP is a protocol for transport which transfers mails from the sender to receiver. [3]
There are some other protocols:

- “Multipurpose Internet Mail Extensions (MIME)”
- “Version 3 of Post Office Protocol (POP3)”

POP3 and IMAP are protocols for fetching. These protocols are functionary with transporting mails from remote server to local. POP3 offers to the users their own virtual post offices at their home; users can log into their accounts; then they can reach their mailbox. Users have all action permissions to their emails. After log in to the system all emails are downloaded to the client. Therefore, POP3 servers do not need huge amount storages like IMAP’s. IMAP is a way different protocol than POP3. Instead of the POP3 protocol, users can always reach their emails online; every email will be stored on server. [5]

The comparison between them explained as well as possible in one sentence:

“One is simple and robust; the other is powerful and flexible.” [5]

MIME is an extension which controls the multimedia attachment formats. It makes possible to send other kinds of files like images, audio, video and some others. [6]

At present time internet electronic-mails are a part of our life and it is one of the most common communication methods. Thenceforward, security problems cropped up. These security problems are information theft, fake e-mails, transfer of harmful files via e-mails etc.

**Security for E-Mails**

Communication offers to people exchange of information, under this definition, the most important security problem for e-mail is information theft. Because in e-mail is an information exchange between two or more than two people. Therefore, end-to-end encryption became the first and most important prevention for security of e-mail communication. It was also vulnerable; many threats from intruders, viruses and hackers are forcing the development of more secure standards. After that, there are many softwares cropped up. The most robust one of these softwares is Mailvelope.
1.3. Thesis

In this Thesis, the basics of cryptography and influence of it on mail security will briefly be examined.

This thesis occurs from four parts:

**Cryptographic algorithms and usage of them with e-mail technology.**

There are many cryptographic algorithms but the effectiveness of them is not automatically given for e-mail communication.

**Examination of Mailvelope.**

Mailvelope is one of the softwares which were created for e-mail security.

**Analyze of implementation methods over Mailvelope.**

After all, this thesis’s goal is the implementation of Mailvelope for data exchange security software. Therefore, technologies which are convenient for implementation should be analyzed in the following.

Mailvelope was developed with JavaScript. It makes it possible to use that as a browser extension. And it offers an aesthetic GUI on a dynamic HTML page. There are also some cons like pros in the use of JavaScript. Implementation possibilities with JavaScript and Java will be examined in the following.

Cryptography is always a sensible subject; a small error can be fatal. Therefore solid software is important. Mailvelope is the software which meets these criteria.

Under the point Mailvelope, it will be examined step by step how Mailvelope works and what kind of background it has. The goal of the thesis, the development and the possibilities of evolution of a project which is similar to Mailvelope will be examined. The main target of this thesis is also about encryption like Mailvelope but it will not be about encrypting any e-mail or other messages. Its main function will be data pulling from database and giving it to the user as encrypted. But there are some achievements in that:

- Robust: Secure against outside attackers.
- Local storage like Mailvelope: Data storage on local computer.
- Secure Key Storage: Encrypted local key storage.
- Different encryption algorithms

2. Foundation

In this thesis, many software and methods are used. Under foundation these methods and software will be explained. Before the explanation of software, there is a short introduction about cryptography; because at the end the goal is mostly connected with cryptography.

2.1. Cryptography

Cryptography is a concept which enables security to be achieved with mathematics. Basically, cryptography makes a secure transmission of a plaintext over a network possible.

There are 2 basic steps for that:
- Encryption of plain text to cipher text.
- Decryption of cipher text to plain text.

When security meets mathematics, then algorithms reveal. Every encryption and decryption process needs an algorithm. Algorithms are the guides of the cryptography which are formed by mathematical formulas. As a definition, algorithm is a procedure which offers a list of steps of operations that solve a problem. Essentially encryption and decryption use the same algorithm, but decryption uses the reverse of this algorithm.

Cryptography is a deep problem which needs algorithm every time. Before the algorithm can be applied it needs encryption methods. There are two encryption methods:
- Public-Key
- Symmetric-Key

As is understood from the name public-key, it is a shared key for everyone and private-key is just known by the user. After all, cryptographic systems still need algorithms which are divided into two types:
- Symmetric-Key Algorithms
- Public-Key Algorithms (Asymmetric-Key Algorithms)
- **Symmetric-Key Algorithm**

It has that name, because it uses the same keys or similar keys that are convertible to each other, for encryption and decryption. In this method, keys have to be protected. There are just one or two keys which should just be known by the parties involved. That is why keys have to be accessible just for both parties. If anyone else can access these keys, it will be defenseless. Alternative names for it are Private-Key or Conventional encryption algorithm.

- **Public-Key Algorithm**

Public-key algorithm uses different keys for encryption and decryption. It makes it more secure than symmetric-key algorithm. But if you compare the two keys, some disadvantages may be revealed. In this method, each side has a key pair; one of them is for encryption and the other one is for decryption. And this pairs consist of public and private-key. [7]

### 2.2. PGP-Pretty Good Privacy and OpenPGP

PGP is software which uses encryption for electronic mails and therewith makes it possible to protect privacy. PGP can also be usable for proving digital signatures.

PGP means “Pretty Good Privacy. The name of “Pretty Good Privacy” came from abununilities of PGP which are explained below.

- PGP encrypt the files or messages. For this encryption it uses IDEA.

  Elongation of IDEA is International Data Encryption Algorithm. IDEA is a private key encryption algorithm which is for OpenPGP standard. Encrypted items can be decrypted just by someone who knows the pass of encryption.

- The users have to create their own private and public keys. These keys enable that the user encrypts and signs the message which would be sent and decrypt the message which was received. Via PGP the user can encrypt and decrypt e-mails.

- PGP allows also keys management for the user. User can create a database which contains public keys and list of people who are in communication with them.
- PGP allows to the user to use digital signatures. It makes it possible to sign all documents and verify the people with the help of their unique signatures.

Public and symmetric-key methods are fundamental for PGP and OpenPGP. OpenPGP is an open-source version of PGP which is created in JavaScript and has algorithms without license problems.

PGP is not anymore an algorithm or method even so it is a cryptography system. That’s why it is more complicated than basic methods; it uses both of cryptography methods which are public-key and symmetric-key. Background reasons of that are disadvantages and advantages of them.

PGP content cannot oneself with one of them. If the PGP will operate just with asymmetric-key method, users would need a secure network that they can use for transmission of their key. Cause, receiver and sender have to share this key, which has to stay unknown for others, with each other. It makes this method still vulnerable. As a solution they can meet together but it is not an optimal solution for global communication.

Public-key method is more secure than symmetric-key, but a comparison of speed between them will show the big difference. Public-Key is a turtle and symmetric-key is a rabbit in this comparison.

2.3. **Key Storage**

Key storage is bounded with cryptography but at the same time they are two disparate concepts. The locksmiths can create a key but they cannot keep them safe; if you lose the key there is nothing to do from locksmith. Cryptography has the same logic; just mathematicians subrogate the locksmiths. A plain-text, which has important information, can be lost and this loss can create a big problem. But if the key is lost, there will be irrecoverable damage to security. Everything will be vulnerable and the cipher texts which are already encrypted will also be defenseless.

Defense or storage of cryptographic keys is important and there are some storage and security softwares for that. Beyond the softwares there are also some conventional methods for securing storage.
The conventional methods are:

- Storage in Password Protected File
- The Keystore Engine
- Encryption of Key

**Storage in Password Protected File**

Every OS (Operating-System) has a file system in it. And in this file system, there are permissions for read and write. These permissions can be protected with password. The file, which contains the keys, can be protected with password for read permission.

**The KeyStore Engine**

The Keystore engine is a class in Java which helps to manage and store private-keys, key pairs and certificates. It is an engine; therefore it does not create methods, instead it implements the method as an instance. The JKS format is the most used and standard format for KeyStore engine implementation. For implementation of JKS (Java Key Store), the code below can be used:

```
"KeyStore mystore = KeyStore.getInstance("JKS")"
```

This definition will give a management on keys in JKS format. There are also some other keystore formats which are:

- Jceks
- Pkcs12

Actually it is possible to remove the “Pkcs12” type because it allows to the user just a limited support which is just read-only support. Therefore it is not suitable to use for every implementation. Hence, there remain just two types which are “Jceks” and “JKS”. “JKS” is the first and default format from JCA’s SUN Provider and it is older than “Jceks”. “Jceks” is simply new in comparison to “Jks” and provider of that is JCE’s SunJCE.

“JKS” is “Java Key Store” and “Jceks” is “JCE Key Store” or “Java Cryptography Extension Key Store”. The first important point: if JCE is used in implementation then “Jceks” has to be used as a key store; only this way it will support the storage of keys which are created with JCE. On the other hand, opposed to “Jks”, “Jceks” offers better security on keys and further,
it is also possible to store private-keys which come with symmetric cryptography. The better protection of “Jckes” comes from use of a PBE with Triple-DES algorithm. After all, there are key tool and some other possibilities that allow management of key store from command line.

**Encryption of Key**

Encryption of key is not a common key storage technique like the keystore engine. But there are many possibilities of key storage with cryptography without other key management utilities. Encryption of key is also one of them.

Mailvelope is examined and it comes up that there is no usage of the keystore engine. As it has been examined, Mailvelope uses also key encryption for key storage. It is also mentioned under the “Key Storage of Mailvelope” topic that users need to assign a password in key generation phase. This password is the key of another encryption in this method. After the key generation, it will be necessary to store the key pair. Public key is not important for security; it will also be shared with third parties. Even so it has to be also stored but not in encrypted form. It is possible to store both of keys in text file but the private-key cannot be plain text like public key for security. In this phase, the password which users give at the beginning will be used for encryption of the private-key. And private-key will be stored as a cipher text which can only be decrypted with a unique user password. Mailvelope uses a local storage file instead of a text file.

Encryption of a key is a solution for secure storage, but it is not certain whether it is actually secure enough or not. If it is compared with “Java Key Store”, it will fall short. “Java Key Store” offers a more stable, more secure and well organized storage chance.

**Keytool**

Keytool is a tool which allows managing the key store and keys in it. It is also possible to manage certificates. Keytool works just on command line and there are defined keytool commands. Keytool makes it possible to do everything which would be also doable with Keystore engine. In opposition to Keystore engine, there is a simpler interface and commands; thus the management of keys is easier.

This command creates a new keystore which uses JCKES keystore type.

There are some other commands for Keytool:

- genkeypair: This code allows to generate a key pair.
- keyalg: Choosing the algorithm.
- dname: It has to be a remarkable name which is also associated with alias.
- keypass: It is a password but different then storepass. It helps to protect private-key.
- genseckey: It creates new private-key and a new key store for it.

[9] [10] [11]

2.4. Mailvelope

In this thesis, Mailvelope will be explored and explained in detail. This thesis is beyond cryptographic softwares more based on Mailvelope. Mailvelope is an origin and all its features, security achievements and weaknesses and the technologies for development and GUI will be explained.

Mailvelope is security software for email communication which offers end-to-end encryption. Every web mail providers has also some cryptographic protocols which are mostly SSL and TLS. But these protocols are sometimes not sufficient for privacy; hence Mailvelope is a beneficial project for web mail privacy.

a. Project Structure of Mailvelope

Mailvelope is beneficial software which provides us with more security and privacy on mail conversations, still this beneficial software needs many other projects. These are open source projects which are based on Mailvelope:

- OpenPGP.js
- Email.js
- DOMpurify
- Bootstrap
- JQuery
- OxygenIcons
The Use of these 6 open-source projects helped in the creation of Mailvelope; each of them has another task in this project. These tasks will be defined under project structure of Mailvelope.

There are some projects for encryption of emails which are FireGPG, Enigmail, GPGMail etc. but Mailvelope is created for web mail, Chrome and Firefox. [12] OpenPGP is already explained under PGP. That is why there is some explanation about other open-source projects which are used in Mailvelope.

**Email.js**

Official definition of Email.js is: “A toolbox for writing email applications in JavaScript. “

It is an open source project which includes components that are really necessary to use for creating e-mail applications in JavaScript.

These components include: [7] [13]

- **IMAP Client**
  
  “IMAP client library written with ES2015 (ES6).” [14]

- **SMTP Client**
  
  “SMTP Client allows you to connect to and stream data to a SMTP server in the browser.” [15]

- **MIME Builder**
  
  “emailjs-mime-builder is a low level rfc2822 message composer.” [16]

- **MIME Parser**
  It is library for parsing mine streams. [17]

- **Adress Parser**
  
  “Allows you to parse mime formatted e-mail address lists.” [18]

- **TCP Socket**
  
  “Enable apps to use the same codebase in Firefox OS, Chrome OS, and on the server.” [19]
• PGP Builder
  “Builds PGP/MIME” [20]

**DOMpurify**

DOMpurify is sanitation software for HTML, MathML and SVG. It is written in JavaScript and compatible with all browsers. It sanitizes the HTML; it reduces the inessential strings and constructs it as a new clear HTML. Therewithal, it prevents from XSS attacks. If a project about security build on HTML, it will be better to use DOMpurify. It is an open source software that helps to fix the security flaws that come with XSS attacks. Besides, it will help to make the website faster by cleaning unnecessary code lines. [21]

**Bootstrap**

Bootstrap is for HTML, CSS, and JavaScript front-end framework. It makes it easier to design web sites and makes these designs compatible to mobile devices. It contains many HTML, CSS and JavaScript templates, so the users can use all of them for own designs.

It is also another open source project and every day continues to evolve. It offers prepared CSS:

• Button Groups, button drop-downs
• Drop-downs
• Progress bars
• Labels, etc.

Alongside with CSS, it offers also many already finished JQuerys.

Bootstrap makes it possible to design a basic web page with a good appearance and functions via queries. [22] [23]

**JQuery**

JQuery is the most popular cross-platform JavaScript library in use today which is created for HTML. Common using of JQuery is mostly for animations and effects for sites on Web.
In 2006, JQuery was written by John Resig. Until today it became widespread and became a real alternative to Flash. Due to the cause of security problems on Flash it became more common than Flash. [24] [25]

**Oxygen Icons**

Oxygen Icons is a project which includes themes, icons, cursors and sounds and created for KDE(K Desktop Environment). [26]

b. **Cryptographic Background of Mailvelope**

As cryptographic software, Mailvelope needs a good cryptographic support. This support comes from an open-source development PGP. Instead of PGP, OpenPGP is used on Mailvelope.

- **Key Generation in Mailvelope and OpenPGP**

Key generation is the first step of OpenPGP. The process cannot start without key. OpenPGP offers some cryptosystems and algorithms for key generation which are:

- RSA and RSA (default)
- DSA and Elgamal
- DSA
- RSA

In Mailvelope, there is just one algorithm which is RSA. But it gives an option with that the user can choose how long the bits of key will be. The choices of bits are 1024, 2048 or 4096 bits long. Of course longer keys are more secure, but they need longer generation time. [27]

The relationship between OpenPGP and Mailvelope is really strong; Mailvelope needs OpenPGP almost in every process. OpenPGP is an open-source project version of PGP which is coded by programmers in JavaScript. And key generation in Mailvelope uses OpenPGP.

On “Generate Key” page, after the user has filled in all necessary fields, the user can click to submit button. After this action the user has to wait between 10-30 seconds. While this waiting period, Mailvelope software generates a key pair with the help of OpenPGP. After generation of the key the user can reach this key from the key list which is on “Display Keys” page.

It will be explained also with help of a figure in the GUI part of this thesis. [28]
• **RSA Key Generation**

RSA is a cryptosystem which uses public-key method. It is named RSA because designers of RSA are Ron Rivest, Adi Shamir and Leonard Adleman and the initials of their surnames are RSA. It is referred that OpenPGP uses RSA. The process of RSA key generation is complicated and occurs from many steps.

RSA Algorithm consists of four steps. These are key generation, key distribution, encryption and decryption. The process of key generation will be shown step by step below:

**Step 1:** RSA needs two different prime numbers which are selected randomly. They can be “p” and “q” as variables.

**Step 2:** Calculation of “n=p*q”.

That “n” value will be modulus for private and shared keys.

**Step 3:** Using of Euler’s totient function (\(\varphi\)).

Euler’s totient function should be used on “n” value.

\[ \varphi(n)= \varphi(p \times q) \]

and Euler’s totient function is a multiplicative function that is why the next step can be that:

\[ \varphi (p \times q)= \varphi (p) \varphi (q) \]

\[ = (p-1)(q-1) \]

\[ = n-(p+q-1) \]

**Step 4:** An integer will be needed “e: 1<e< \varphi (n)” and “e” and “\(\varphi(n)\)” should be coprime.

This e will be the Public-Key exponent.

**Step 5:** A variable (“d”) has to be determined as “d \equiv e-1 \pmod {\varphi(n)}”.

**Step 6:** This “d” will be the Private-Key exponent. d*e= 1 mod \(\varphi(n)\).

Every value should keep secret.

**Step 7:** The Public-Key consists of modulus n and the Public-Key exponent e.

**Step 8:** The Private-Key consists of modulus n and the Private-Key exponent d.

**Cipher Text:** “c”, **Plain Text:** “m”.

Then the Public-Key will be used for encryption: “C= m^e \pmod{n}”.  

14
And the Private-Key will be used for decryption: \( m = c^d \mod n \).

[29] [30]

c. Key Storage on Mailvelope

Key generation was the first step; after generation of keys the software needs to work on storage of these keys. In OpenPGP, it is possible to store these keys in local storage. On the other hand, if the user wants to publish them, there are many key servers onto which the user can load the keys. Key servers are basically key repositories. Mailvelope also offers an exportation of keys. At the beginning, after key generation, keys are stored in local storage of the browser. Users can reach these keys via Mailvelope extension on the browser. Keys can be displayed on browser, further they can be exported. If the user wants to export a key, it is possible to export them as a “.asc” file. With exportation it is easier to publish them on key servers.

Mailvelope is extension software for browsers (Firefox and Google Chrome). That is why it uses local storage of browser. If the user wants to reach one of the keys manually, he or she has to follow this step:

- Open the C: directory. And go to this address:

  “C:/Users/(Username of the user’s PC)/AppData/Local/Google/Chrome/User Data/Default/Local Storage/”

There are many local storage files in this folder. But one needs to look into the files which start with chrome-extension as a name.

Google Chrome has not just one extension; so the folder name has to be related with Mailvelope.

It will appear in this address:

“C:/Users/Username of your PC/AppData/Local/Google/Chrome/User Data/Default/Extensions/”

There will be many meaningless folder names. It is possible to find Mailvelope with the date of the folder or it is also possible to go in and search the files to prove Mailvelope, but it will take a long time.

In the PC which is used for this project it was found under this address:

“C:\Users\pc\AppData\Local\Google\Chrome\User Data\Default\Extensions\kajibbejlbobhfggdiogboambeijhkke”
This folder represents the Mailvelope extension. After specification of the folder name, this is “kajibbejlbohfaggdiogboambcijhkke”; it is easy to find the key files under:

“C:/Users/Username of your PC/AppData/Local/Google/Chrome/User Data/Default/Local Storage/”

“kajibbejlbohfaggdiogboambcijhkke” is the default folder name of Mailvelope. This folder name is already given under installation instructions on web page of Mailvelope which is “https://github.com/mailvelope/mailvelope”.

There it is necessary to search the local storage files which include “kajibbejlbohfaggdiogboambcijhkke” in their name. After finding the file, it can be opened in the text editor and it is possible to reach to the key. For public keys security is not that important because they will also be shared with others. But for private-key it is really important and the storage of private-key in text editor as a plain form is not secure for a cryptographic software. Thenceforward, the private-keys are stored in encrypted form. Every user has to give a password in the key generation process; this password is used for the encryption of private-key. So nobody can use the private-key without that password combination; this protection makes the storage secure.

**d. GUI of Mailvelope**

Graphical User Interface (GUI) is always very important step for software. Mailvelope is not complicated software but still GUI is really important. GUI’s first goal is offering to the user a simple and easy use of software. A simple GUI and easy using of software will attempt more users and will educate about software.

Mailvelope is a browser extension; the user has to download and install it first. After installation of Mailvelope on Google Chrome, the user will get a shortcut button on Chrome as it is shown in the following image.

![Figure 3.1](image)  

This shortcut button ensures to the user fast access and simple features.
After a click to this shortcut button, it will appear a simple interface like in figure 3.2.

![Mailvelope interface](image)

Figure 3.2

This interface ensures just simple features and not all of them. Still it is possible to do many things on this interface.

There are some buttons with features:

- **Activated**: If this button is activated, the user can use external editor. Also without activation of this button it is not possible to use “Add” button and “Reload” button which are on this interface. Actually if it will not be checked, the user cannot encrypt or decrypt any mail. They can just manage the keys in storage.

- **Help button**: will direct the user to Mailvelope’s web-page which includes documentation about Help. [https://www.mailvelope.com/en/help](https://www.mailvelope.com/en/help)

- **Add button**: is to add the active Web-Page address as an e-mail provider.

- **Reload button**: will re-start all Mailvelope tools.

- **Last activities**: The user can track the past activities.

Options will redirect the user to real GUI of Mailvelope, which is created with HTML, Bootstrap and JavaScript.
As it is illustrated in figure 3.3, there will appear three main pages here which are “Key Management”, “File Encryption” and “Options”.

Under “Key Management”, the user can reach to “Display Keys”, “Import Keys”, “Generate Key” and “Setup”.

- Generate Key: The page shown in figure 3.4. The user needs to fill blanks, which are supposed to contain name, e-mail address and password. In “Advanced” tab, it is possible to choose key algorithm and key size. But in Mailvelope, RSA is the only algorithm and default. Key size can be selected as 1024, 2048 or 4096 bits. Instead of key generation users can import keys.
- **Import Keys**: There are three ways for key importation to local key list for users which are “Key Search”, “Import key from file” and “Import key as text” as shown in figure 3.5.

- **Key Search**: Users can search the key from server as a name, e-mail address or key id. They can just search the public key because key servers are for public keys.

- **Import key from file**: It is available to import a file which is text or “.asc” type. It will be converted and imported to the key list.

- **Import key as text**: It is most conventional way of key importation. Users can write or paste the key’s text on that text field.

![Figure 3.5](image-url)

After importation or key generation, all keys can be listed under “Display Keys” page.

In this page, there is one table which occurs from “Name”, “Email”, “Key ID” and “Created”.

*Name*: Name of owner the email address.

*Email*: Email address.

*Key ID*: A unique key identification. It will be created automatically for each key.
The second main page is the file “File Encryption”. On this page, users have two features: one of them is usable as a sender and the other one as a receiver. As a sender, users can go to tab “Encryption” and add a file from a hard drive. After that, the user has to choose receivers which exist with their public key on the key list. Then it is allowed to download the encrypted files to hard drive. These files could be decrypted just with private-keys which match with the public key that encrypted the file.

“Decryption” tab will display almost the same page like “Encryption”. Encrypted files can be chosen and added to here from hard drive. And they can be decrypted with private-keys which match with public key, but just if these keys exist on the key list.
The third main page is “Options” that consists of five parts which are “General”, “List of Mail Providers”, “Security Log” and “Key Server”.

- **Security**: In this page which is shown in Figure 3.7, the user can setup how long the passwords are supposed to be kept, that are already used for decryption, in memory. Furthermore, it gives the users the opportunity to create special designs for the encrypted mail’s background. Receivers will see the Mailvelope background picture instead of a text, when they open encrypted mail. There is also an option which offers the users to choose whether they want to use pop-up or the same page to display decrypted Mails. Users can use it also for encryption that do they want to use Mailvelope’s text editor or mail provider’s.

- **List of Mail Providers**: It shown list of all mail providers. There are many of them which come automatically with Mailvelope. But it is also possible to add a new provider manually.

- **Key Server**: User can give the URL address of key server. [31]

### 3. Methods

Technology is a widespread ocean. There are many methods for software development. Every one of them has positive and negative points. Analyzing the advantages and disadvantages are an important process in order to find the optimal technology. Therefore, the compatible methods will be analyzed under this topic.

In Mailvelope project structure, all the softwares and methods are mentioned which are required by Mailvelope. It needs JavaScript in almost all phases. JavaScript will be examined as a method but except JavaScript PHP and Java methods will be also examined.
3.1. PHP and Java Methods

![Flowchart Diagram](image)

PHP and Java cooperation offer many features and easiness. Instead of Mailvelope, in these methods, there will not appear alone JavaScript code. There is above a workflow of PHP method in Figure 3.1.1.

The software and methods list of those that are used in this implementation;

- Java
- JCE (Java Cryptography Extension)
- HTML
- PHP
- KeyTool
The void of JavaScript will be replaced by Java with Java Cryptography Extension (JCE). Java cryptography extensions library is a crucial component for this implementation; therefore access on Java will also be essential. With this library, there are more potential algorithms to use for cryptography. Otherwise the implementation encounter with deficient algorithms. Other than this there will be another software which are MySQL; for the database connection, there are HTML; for GUI and PHP for the boundary between GUI, Java and databases. For more description in detail; it will have a login page for users which are already registered to the system. And for registration there will be another registration page which is separate from the login page. Every user will get a unique key pair after registration. After registration users can log into the system from the login page. Users can pull data from database according to their permission. But at this point, these data which are pulled by users will be encrypted with unique public password from the user. And it is only possible to be decrypted by the right combination of login password and private-key of user.

**Registration:** People who would like to be users have to register.

**Login:** Users, which are already registered to the system, will head straight to the login page instead to the registration page.

**Sending Data as a User:** The logged in user can send a text or text file from their page. The data which will be send will be encrypted from the key which belongs to user, who is already logged in.

The connection between HTML, PHP and Java with JCE makes this method more charming. On the one hand, it is possible to use JCE just on a Java platform that is why the project has to be created with Java definitely. On the other hand, the project needs to be a browser extension, so it needs to use a script language for web programming. At this point, there are some options which are JSP (*JavaServer Pages*), JSF (*JavaServer Faces*) or PHP (*Hypertext Preprocessor*). Because of the common use of PHP and many features of it, PHP was chosen to be used for this implementation. So a bridge code is needed between PHP and Java because the software needs to run the java.jar, which includes JCE, on the GUI. That is why the code is used that can be seen below.
<?php
exec("java -jar C:\Users\pc\Documents\NetBeansProjects\keyGen\dist\keyGen.jar");
?>

Structure of the Java and PHP Method

In the beginning will appear a registration page. In this implementation, there is a user system; not registered users cannot use any feature of the software. So it will be better to take a look at the registration page which is “register.php”.

There is a HTML form with a submit button and four text area for variables, which are name, surname, user name, e-mail and password. Users can fill these text areas then click the submit button; after this step the software will save these values to database, if every values are true.

-Key Generation (keyGen.jar)

The keyGen.jar is a small application for key generation; it has to work together with the registration page, because every new user needs a unique key pair for encryption and decryption features. With execution of keyGen.jar, it will be created a unique key pair and it will be saved to key store by key tool; but it is just available at the first registration. Users cannot create a key pair after registration. The code below will show the JCE codes for key pair creation in keyGen.java with explanation. But these codes will not be used for generation; instead of this, key tool commands will be performed.

Key pair generator is a method which comes from Java; Security library. This library is a feature that JCE to users offers. In generation, RSA algorithm was used. [32] [33]

Key will transform into a key pair which formed by private and public. It is possible store these keys also in sql database in User’s information; it will be a solution with JCE for the project. But, it will make the project vulnerable because the storage of keys on database will not be secure enough. Therefore the Key-tool commands will be performed instead of that.

In the key generation phase, the command of Keytool which is used is the following:

```
-keytool -genkey -alias "username" -keyalg RSA -keysize 2048 -keystore “Location of Keystore” -storepass “a Key Store Password” -keypass “password” -dname CN="name"
```

This command means;
-genkey: Generate Key

-alias: The alias will be taken from database like user name of users who are already in the registration process.

-keyalg: It is possible to choose key algorithms; in the command above, RSA was chosen.

-keysize: is the definition of key size of the algorithm.

-keystore: need address of a key store where the keys will be stored.

-storepass: is a password for whole key store.

-keypass: is a password for key pair. It will be also taken from database as password of user.

Under this command, it creates a key store as if it would be used for the first time; otherwise it will just add the new key pair to the key store. In this project, this command has to be used under Runtime code in Java. Key-tool commands are just performable on CMD.

“CMD is the default operating system shell for DOS operating systems and the default command line interpreter.”

That’s why it needs to be run under a runtime process.

“Runtime rt=Runtime.getRuntime();”

Java runtime code makes it possible to run external software and here it runs CMD of Windows. After running of CMD, the commands will be executed. Key-tool exists in Java, that’s why key-tool command has to be executed in directory that is “Java\jre.(Version Number)\bin\”.

After all, it will be a user created with information in database and a unique key pair in key store. [11]

-Login

As mentioned before, a registered user has information in database and key pair in key store. In login process, the e-mail address and password, which are on login page submitted, have to match with the address and password in database. If it will match, the login will be
successfully performed. Otherwise, it will give an error alert which says “False Password or E-Mail” and will not redirect the user to another page.

If the login process would be successful user will be redirected to the home page. Home page offers to the user some features that are:

- Sending Encrypted Data
- Pulling Encrypted Data as Decrypted

Both of these features need a connection with key store and database.

At the beginning, the key store has to be connected to pull key pair.

In pulling key, it will not use any Key Tool command like key generation. Instead of that, JCE library will be used for connection between Java and key store.

- Getting Keys (KeyGet.jar)

The first step of “keyGet.jar” is access to the key store. For this purpose, it will be used some basic “FileInputStream”, “KeyStore” and “Keystore.ProtectionParameter” codes.

There is also a simple explanation about KeyStore.ProtectionParameter:

“A marker interface for keystore protection parameters.

The information stored in a ProtectionParameter object protects the contents of a keystore. For example, protection parameters may be used to check the integrity of keystore data, or to protect the confidentiality of sensitive keystore data (such as a PrivateKey).” [34]

At the beginning for loading key store there is just a location of key store needed. After access to the key store, it will be necessary to pull out of private and public key. These keys have to belong to the user who is already logged in. After encryption, the encrypted data will be stored in database. It also makes the system safer because if an intruder has obtained the key pair of a user, the intruder still needs to get encrypted data from database and there will also be a request for the personal password. This system is just for securing data storage for people. There will not be a feature that enables the users to send data between them.
3.2. **JavaScript Method**

JavaScript and Java are totally different languages. Both of them have also different workflows as seen in Figure 3.1.1 and Figure 3.2.1. One of them works with interpretation Principe and the other with compilation. JavaScript can be called a script language and Java a programming language.

“*JavaScript is an interpreted programming language with object-oriented capabilities.*” [35]

Besides, JavaScript is a programming language which allows to interactive script for users in web browser. In JavaScript all of these scripts can work as embedded with HTML. Therefore,
JavaScript offers dynamic Web-Based GUI. Like in the examination of Mailvelope, it becomes apparent that it is also developed with JavaScript.

In JavaScript implementation, there will be two different pages on GUI. One of them is for user creation and another one is for sending data with the encryption process.

-User Creation: It will be like a register page in PHP. But instead of database, the information will be stored in a local text file. The user has to fill these fields:
  - User Name
  - Name
  - E-Mail
  - Password

After this process, a user row will be created in the local storage.

-Sending Data: Sending data page will show the user a text area for the text that the user wants to send as encrypted, a text field for user name and a password field for password.
  - User Name
  - Password
  - Text

The key pair will be generated in the same process and it also will be stored in local storage like user information.

JavaScript implementation has no login phase. That is why in this step, the implementation needs a password for encryption. Passwords will be used for decryption of private key like they are used in Mailvelope. After decryption, the plain key can encrypt the plain text which is inputted in the text area.

3.3. Java Method

Java method is combination of the other two methods. It has same logical workflow with JavaScript which shown in Figure 3.2.1.
In this method, there are two pages which are user creation and sending data pages. These pages have exactly same logic and system which are under JavaScript method explained. That is why at the beginning user has to give user name and password in user creation page.

![Image of user creation page](image1)

**Figure 3.4 1**

After user creation, a unique key pair, which is protected in key store with user password, will be produced. There will be no database for user information; instead of that all information will be stored in key store. Users, who have already a key pair, can use the second page. And the second page offers to user encryption or decryption of text.

![Image of encryption/decryption page](image2)

**Figure 3.4 2**

User can write his plain or encrypted text to the first text area and after filling the name and password it is possible to do encryption or decryption.

At the beginning in result, Java will be not compared with other methods because in Java method, it is not possible to make it an extension.
4. Result

<table>
<thead>
<tr>
<th>PHP and Java</th>
<th>JavaScript</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ Database Storage for User Information</td>
<td>+ A simple and dynamic GUI</td>
</tr>
<tr>
<td>+ Two layered password security (One password for login and one for key storage)</td>
<td>+ Easy implementation as an browser extension</td>
</tr>
<tr>
<td>+ Many kinds of algorithms with JCE</td>
<td>+ Local Storage</td>
</tr>
<tr>
<td>+ Programmable GUI</td>
<td>+ Password Encrypted Private Key</td>
</tr>
<tr>
<td>+ Java Key Store usage</td>
<td></td>
</tr>
</tbody>
</table>

PHP, Java and JavaScript technologies are experienced for implementation methods. Both of them offer unique features and bring along difficulties and deficiencies. There is no perfect choice, but there is always a best matching choice. For this reason the comparison of two methods will provide the users with information for the most suitable choice.

Negative points are more decisive than positive points. It is better to start with the comparison of positive points. There is a small list about that lines out which method offers what.

Positive points are important but still they are not enough to single out one of them. At first sight, JavaScript goes ahead with better GUI possibility and local storage, which is more secure for storage of cryptographic keys. Still JavaScript needs support of other open-source projects for cryptographic algorithms. At the other hand, in Java everything is compact and ready to process; with JCE there is no need to other projects for algorithms. It is also same for key storage; there are JKS and Key-tool which are compact with Java.

As it said positive points are not decisive. That is why, negative points need to be considered carefully to be ordered for they priority. Because a negative point concerning security will be more important than others. Actually there are no negative points, instead that there are not suitable points for the project.
PHP and Java

- Cause PHP, it is dependent on Server usage. (Vulnerable)

- Remote Database

JavaScript

- There is not JCE and that is why lacking of algorithms.

- There is not Java Key Store and Key Tool.

The negative points give more concrete consequences about the choice. But still both of them are like apple and pear. Java and PHP gives more freedom and many stable options without any extra exertion. In JavaScript, everything should be written from zero or should be taken from other open source projects.

4.1. PHP and Java

PHP was created by Rasmus Lerdorf. The first features of PHP were really limited; it was created in 1994 for common Web-site functions like hit-counter. Until 1995 it was evolved to Personal Home Page tools (PHP), which is more than just utilities. This evolution did not end in 1995 but rather continued until today and it is still evolving.

“PHP is a server-side, cross-platform, HTML embedded script language.” [36] This is said by creators of PHP. But one can hardly call PHP a simple script language. It is also a programming language which provides full features.

PHP can offer to users and developers dynamic interface and dynamic content which is really important for this implementation. Still, this implementation has to be an extension after all. Use of PHP on HTML simplifies the step for making this extension.

PHP forms are examined above and there is a registration obligation for users just like with Mailvelope. It gives more control on behalf of the user management because with it there is a remote database which is created for `User` table. Nevertheless, it is not obligatory even though it is an atony against attacks. All the steps of Mailvelope’s are realized on a local platform and it makes the software safer.

After user registration, the software will produce a confrontation with key storage. Key management and user management will be separated for security. Instead of user database, the key storage is local.
The most important problem of PHP implementation is that there is no possibility for fully local storage and local process. Therefore it is not appropriate implementation. [36]

4.2. JavaScript

JavaScript makes it possible to use all features in a local area. It creates secure software usability for users, which is really important for encryption software. But in the end, there are not so many cryptography algorithm varieties.

JavaScript does not allow to access on Java; therefore it is not feasible to use of JCE.

In PHP implementation, coexistence between PHP and Java could be created with a batch execution. But JavaScript does not allow any batch execution, because of the amount of security threats. If it would allow that, a web page could execute a harmful file on a user’s PC and it can be destructive.

JavaScript’s cryptographic background is dependent on other softwares. In Mailvelope that was OpenPGP. OpenPGP has just RSA, DSA and Elgamal algorithms for encryption. The other algorithms are not available on OpenPGP and it makes that software limited.

Key storage is also an important process; Java Key Store with Key-tool is a secure option. But it is just manageable via commands which are applicable on CMD. For this usage would also need an external execution, which is not possible.

As a result, JavaScript implementation offers secure local storage with text or local storage files but not a steady and well structured storage like Key Store. Lacking of algorithms will affect further development of the software.

5. Conclusion and Discussion

In this last part of the thesis the outcome of a software implementation, which is based on Mailvelope’s logic, and the important points as what this software is able to do and what not, are explained. Mailvelope’s achievements and deficiencies are compared with implementation and the goals of implementation with their success status will be explained.
The Goal

The goal is the creation of the software which is a bit similar to Mailvelope. It is just similar because of the local key storage, the protection of key’s, the encryption process protection, which is protected by user password and user friendly GUI, which is also as a dynamic extension available.

The main feature of the software is storage of encrypted texts, files, etc. in local system. It is also really different from Mailvelope. They are just working with the same logic. The encryption process will be also protected with user password and the keys help to the encryption and decryption like Mailvelope.

The Result and Discussion

The goal and the outcome should be the same; but they have always some differences. In this outcome, there are also some differences which ideally should be avoided but could not.

At the end, after detailed comparison, JavaScript is the best match between them. JavaScript is the best match, because it does not need any remote server; it is available to process in local server. It is the key point for implementation; everything is considerable except non-local action. JavaScript wins against PHP and Java implementation; but it is still not the perfect option for this implementation. That is why Java will be the best option but this time without PHP. The reasons and benefits of this decision will be explained in the conclusion. Java and PHP implementation was examined but Java is not. Java implementation is almost the same as Java and PHP implementation. It is just cleared from PHP codes, remote server and database. This clearance brought some shortcomings about the features. This clearance was cause of the security and made also positive effects about it.

The most important things about the goal were secure key storage, local storage, user friendly GUI, extension module and password protection. All of them are not available in Java implementation; there is a list above which shows all features one by one.

- **Secure Key Storage:** Java allows using of JKS; it offers secure key storage under Java guarantee. It is also more secure than what Mailvelope offers.

- **Local Storage:** Instead of a remote database, the encrypted files or texts will be stored in local computer. This local storage offer more reliable storage for files.

- **Password Protection:** As it was examined, users can assign a user password in key generation process. This password allows protecting private key and starting of encryption
process without permission. In this implementation, the key pair is stored in JKS and in JKS, there are two password fields which are key password and store password. Store password secures the whole key store. But the key password secures just one entry. That is why it will be assigned by user password. It means the user password will protect the keys of the user.

-User Friendly GUI: Java offers also user friendly GUI like the other platforms. Instead of HTML objects, Java swings will be used.

-Extension Module: Mailvelope was a browser extension and that was also one of the goals for this implementation. But it is not possible to do that just with Java. It is only possible with JavaScript and PHP implementation, but these are not stable in security. In this implementation, it has to be abdicated from a feature and it is an extension module.

The Outcome

In this thesis, the basic cryptography is examined. Additionally, it also includes an examination about mail security and effects of cryptography on mail security. In the light of these two basic investigations, the main investigation head on to Mailvelope, which is the software belonging to cryptography and mail security.

All of these examinations and robust structure of Mailvelope have created some questions about implementation possibility of a different type of cryptographic software which is not interested in secure mail transfer, but rather interested in a secure file storage system.

The implementation is fully separated from Mailvelope and based on other tools. It is also programmed with another programming language. Still, the implementation is mostly inspired by Mailvelope.

This research and implementation intend to take the initiative about creating secure storage software which offers to organizations security and user privacy.
List of Figures

Figure 3.1 ................................................................................................................. 16
Figure 3.2 ................................................................................................................. 17
Figure 3.3 ................................................................................................................. 18
Figure 3.4 ................................................................................................................. 18
Figure 3.5 ................................................................................................................. 19
Figure 3.6 ................................................................................................................. 20
Figure 3.7 ................................................................................................................. 20

Figure 3.1 1 .............................................................................................................. 22
Figure 3.2 1 .............................................................................................................. 27
Figure 3.4 1 .............................................................................................................. 29
Figure 3.4 2 .............................................................................................................. 29
List of Acronyms

SSL  Secure Sockets Layer

TSL  Transport Layer Security

SMTP  Simple Message Transfer Protocol

RFC  Request for Comments

MIME  Multipurpose Internet Mail Extensions

POP3  Version 3 of Post Office Protocol

IMAP  Internet Message Access Protocol

PGP  Pretty Good Privacy

IDEA  International Data Encryption Algorithm

TCP  Transmission Control Protocol

OS  Operating System

MathML  Mathematical Markup Language

HTML  Hyper Text Markup Language

SVG  Scalable Vector Graphics

XSS  Cross-site Scripting

CSS  Cascade Styling Sheets
KDE  K Desktop Environment
RSA  Rivest, Shamir and Adleman
DSA  Digital Signature Algorithm
GUI  Graphical User Interface
URL  Uniform Resource Identifier
PHP  Hypertext Preprocessor
JCE  Java Cryptography Extension
JSP  Java Server Pages
CMD  Command
DOS  Disk Operating System
JKS  Java Key Store
Bibliography


3. History of E-Mail. [Online] [Cited: ]

4. Internet E-Mail Standard. [Online] [Cited: July 16, 2016.]


11. KeyTool. [Online] [Cited: 23 August, 2016.]
http://docs.oracle.com/javase/7/docs/technotes/tools/solaris/keytool.html.


29. Ari Juels, Jorge Guajardo. RSA Key Generation with Verifiable Randomness. 05 February 2002.


32. SecureRandom. [Online] [Cited: August 15, 2016.]
https://docs.oracle.com/cd/E17802_01/j2se/j2se/1.5.0/jcp/beta1/apidiffs/java/security/SecureRandom.html.

33. JCE Oracle. [Online] [Cited: August 13, 2016.]
http://docs.oracle.com/javase/1.5.0/docs/guide/security/jce/JCERefGuide.html.

34. Protection Parameter. [Online] [Cited: August 25, 2016.]
https://docs.oracle.com/javase/7/docs/api/java/security/KeyStore.ProtectionParameter.html.


http://docs.oracle.com/javase/6/docs/api/java/security/KeyStore.html.