



UNIVERSITY OF NAIROBI
SCHOOL OF COMPUTING AND INFORMATICS

SECURE PASSWORD SHARING AND STORAGE USING ENCRYPTION AND KEY-
EXCHANGE

By

Joseph Okwedo Mwamba

P53/11562/2018

Supervisor:

Dr. Andrew Mwaura Kahonge

A report submitted to the School of Computing and Informatics in partial fulfillment of the requirements
for the award of the degree of Masters of Science in Distributed Computing Technology

2021

Declaration

I hereby declare that the work submitted is my work and has not been presented at this or any other institution.

Joseph Okwedo Mwamba

Signature:



Date: 06/08/2021

This proposal has been submitted for examination with my approval as University Supervisor:

Dr. Andrew Mwaura Kahonge

Signature:



.....
Date: 26-Aug-2021

Dedication

I dedicate the study to my family members for their constant support and encouragement.

Acknowledgment

I would like to sincerely express my gratitude to my supervisor, Dr. Andrew Mwaura, for the time and intellectual guidance he offered me during the study. He was genuinely devoted and professional.

To all lecturers and students who facilitated my learning experience and for the time we spent together, not forgetting the members of the public who made this research study successful by providing me with the required information when needed.

To all family members, friends, workmates at Net & Com Ethiopia and SportPesa Kenya for the support and encouragement they offered me during the study.

Abstract

Based on security best practices for passwords, the credential is a confidential pin for authenticating system users, but there are instances where users share a common password for resources. Credentials sharing necessitates the passing of sensitive private information between individuals, thus creating a litter of sensitive data across email boxes and other forms of communication. To mitigate security vulnerabilities resulting from the transmission of passwords from one person to another, information security experts recommend using password management applications. On the contrary, there have been researched studies revealing vulnerabilities in password management applications.

The main objective of the research was to develop a process model for password sharing using asymmetric cryptography. In addition, part of the objectives was to build and test a prototype that facilitates the secure sharing of passwords over the internet using the redefined process model. The research was an exploratory study consisting of two phases. The first phase involved the design and implementation of a system prototype. The prototype was used for managing shared credentials, whereby passwords were stored in an encrypted database. A pair of public and private keys were used to encrypt and decrypt the data during transmission and owner's access for usage. The second phase of the study included a focus group discussion, which was used to evaluate the developed prototype.

The result of the study was a prototype of a system that facilitated the secure sharing and storage of passwords over the internet using asymmetric cryptography. The prototype was also used to model how the vulnerabilities exhibited in the current password management applications can be avoided.

Table of Contents

Declaration.....	i
Dedication	ii
Acknowledgment.....	iii
Abstract.....	iv
1 Chapter One: Introduction.....	1
1.1 Background	1
1.2 Problem statement	2
1.3 The research objectives	2
1.3.1 General objective.....	3
1.3.2 Specific objectives.....	3
1.4 Research questions.....	3
1.5 Scope and limitation of the study.....	3
1.6 Significance of the study	3
2 Chapter Two: Literature Review.....	4
2.1 Best security practices for passwords.....	4
2.2 The state of password usage as a daily activity	5
2.3 Overview of Cryptography	8
2.3.1 Public key cryptography.....	9
2.3.2 Hash Functions.....	10
2.3.3 Digital Signatures.....	10
2.4 Related systems - Current password storage and sharing tools	11
2.4.1 LastPass password manager & security architecture	12
2.4.2 Dashlane password manager & security architecture	14
2.4.3 KeePass password manager & security architecture	14
2.5 Other password sharing techniques and tools.....	15
2.5.1 Use of notebook or paper	15
2.5.2 Storing passwords unencrypted in a file on a connected device.....	15
2.5.3 Storing passwords using browsers.....	15
2.6 Conceptual model.....	15
2.6.1 Authentication process.....	16
2.6.2 Encryption and decryption process.....	17
2.6.3 Data sharing process.....	17

3	Chapter Three: Methodology	19
3.1	Research design.....	19
3.2	Phase 1: System design and development	19
3.2.1	<i>System requirements definition and analysis</i>	19
3.2.2	<i>System Design or architecture</i>	21
3.2.3	<i>Implementation phase – GUI screens</i>	23
3.2.4	<i>Integration and testing</i>	26
3.2.5	<i>Deployment and system maintenance</i>	29
3.3	Phase 2: Focus group discussion.....	30
3.3.1	<i>Sources of data, population, and sample size</i>	30
3.3.2	<i>Data collection</i>	30
3.4	Data analysis.....	30
4	Chapter Four: Results and discussions	31
4.1	Techniques for protecting data at rest, in motion, and use	31
4.2	Redefined process model for storing and sharing passwords	33
4.3	Developed system	34
4.3.1	<i>Keys generation</i>	34
4.3.2	<i>Keys encryption and storage</i>	35
4.3.3	<i>Data protection: In local storage</i>	35
4.3.4	<i>Data protection: During transmission</i>	36
4.3.5	<i>Email notification payload</i>	37
4.3.6	<i>Account recovery mechanism</i>	37
4.4	Results from the focus group	37
4.5	Discussion from the results.....	39
5	Chapter Five: Conclusion and recommendations	41
5.1	Conclusion	41
5.2	Limitations.....	42
5.3	Future work.....	42
	References	42
	Appendix	44
	Appendix A: Project schedule.....	44
	Appendix B: Survey questionnaire – System evaluation.....	44
	Appendix C: List of tables and figures.....	45
	<i>List of figures</i>	45

<i>List of tables</i>	45
<i>Appendix D: System installation instructions</i>	46

1 Chapter One: Introduction

1.1 Background

Despite well-defined ICT policies about sensitive information sharing, credentials sharing is one of the main scourge that waters down the effectiveness of security policies in organizations. It starts with simple implementations, such as a shared password for wireless network access to credentials for sensitive resources, such as data center servers. Usually, employees infringe on simple policies, such as not writing down the credentials to sending sensitive data to their colleagues through email, short messages, and other forms of communication. For example, a study to determine how prevalent medical practitioners use their colleagues' private accounts to access patients' electronic medical records (Hassidim, et al., 2017) established that at least 57% of their respondents had accessed sensitive health records using colleague's accounts. They cited reasons, such as the colleague wanting to act while away, technical malfunction, and limited user account privileges.

One of the critical aspects of authentication is the ability of the system to uniquely identify an entity or a principal accessing or making requests within the system. Authentication can be achieved with a token, certificate, or biometric-based authentication. Password credential is one of the most popular token-based authentication mechanisms. Over the years, the world has witnessed increased password users due to utilities such as bank accounts, email accounts, credit card accounts, social media, and many other forms of information system platforms. These users vary in terms of age and information security awareness. The increase in password users has also elevated the authentication mechanism as one of the most targeted security vulnerabilities in information systems (Standridge, 2019). (Higgins, 2014) reported that two out of three cybersecurity breaches at Verizon (an American telecommunications company) involved password attacks through stolen or misused credentials. The report indicates that most internet and other information systems users do not follow or are unaware of safe password management practices (Rubenking, 2015).

ICT security best practices stipulate that passwords must be complex by containing upper and lower case characters. It also requires passwords to include numbers and symbols. The policy also recommends that passwords should not be a word in any language, slang, dialect, and jargon. To further disassociate the credential from the owner, it is recommended that the credential should not be based on user personal information. Lastly, it should never be written down. On the contrary, the password scheme must be efficient and practical. Due to these stringent specifications, research by (Brown, Bracken, Zoccoli, & Douglas, 2004) shows that users tend to circumvent the difficulty in learning and remembering secure passwords by adopting their inappropriate methods of generating and storing passwords. A survey study conducted by Google shows that at least fifty percent of people prefer using their memory to keep track of their multiple passwords and usage. The fact that information systems users can remember their multiple passwords and usage is a testimony to either they are not using best password practices, or they are reusing their passwords on multiple accounts, or both (Ion, Reeder, & Consolvo, 2015).

Sharing information is one of the fundamental aspects of organizations; hence, security systems and policies should be designed with such preferences. To curb the practice of insecure password sharing, most information security experts recommend using password management applications (Standridge, 2019). Other than assisting in generating complex passwords that are hard to crack, existing password management applications, such as LastPass, is used by security experts to securely store, keep track and share passwords in cases where joint accounts are in use. Despite such applications, a study by (Ion, Reeder, & Consolvo, 2015) shows that most non-security experts do not make use of password management applications. The survey showed that seventy-three percent of security experts use password management applications

compared to twenty-four percent of non-experts, even though both groups agree on recommended password best practices.

1.2 Problem statement

The current popular modes of password sharing, such as emails, SMS, and instant messaging, are not secure. In addition, the report by (Standridge, 2019) has revealed several vulnerabilities related to the model of communication or information sharing in the current password management applications such as LastPass, KeePass, and Dashlane.

Password best practices are critical to any information system. A fundamental principle in password utilization is that users in any particular system should own their credentials, which should be kept private (Schumacher, 2019). In a study to investigate security considerations in scenarios where credentials are team-based, (Schumacher, 2019) discovered that the principle of non-sharing might be rendered impractical due to the following reasons or situations:

The first scenario is that the account credentials might be for a simple system, which does not allow multiple accounts. A perfect example of single-user systems is phones, simple network devices with a non-centralized authentication system, and a thermostat.

As illustrated by (Schumacher, 2019), a second scenario is a local administrative account used to bypass the centralized authentication database. There are also cases where individual users are experiencing communication failures with the centralized authentication system, thus requiring a local account to avoid the central authentication server. Such credentials are usually shared among members of the team managing the subject system or device.

As mentioned earlier, a different study by (Hassidim, et al., 2017) indicated that at least 57% of corporate employees working in the health sector admitted having used colleagues' credentials to access information from the system. They cited reasons, such as colleagues wanting to work remotely, unavailability of a personal account, and limited authorization on their accounts to perform their assigned duties effectively.

Complete adherence to password best practices always results in cryptic passwords, which are hard to remember. Due to this factor, password users are usually forced to write down their passwords for remembrance. Unfortunately, the written passwords are traditionally stored in unencrypted text files or a piece of paper, thus creating a security vulnerability. For instance, unauthorized individuals can access the text files or the notes used to store the passwords, thus resulting in password theft, which is later used to compromise more sensitive systems.

In this light, I proposed to carry out a study and implement a system specifically meant to enhance the security of passwords shared between individuals over the network or internet. The proposed prototype enhances the security of transmitted data by re-writing the process model for sharing sensitive credentials, thus curbing the vulnerabilities presented by the current password management applications. Furthermore, the system ensures the confidentiality and integrity of the stored and transmitted data over a communication network using asymmetric cryptography.

1.3 The research objectives

The objectives have been broken down into specific and general goals, with the particular objectives drawn from the general objectives.

1.3.1 General objective

The primary objective of the research was to create a process model and prototype of a system that facilitates secure password sharing over the internet.

1.3.2 Specific objectives

- i. To identify and study the vulnerabilities presented by current password management applications such as LastPass, Dashlane & Keepass.
- ii. To assess the various data-centric approaches for protecting data at rest, in-use, and in transit using public-private-key encryption.
- iii. To research and develop a process model that mitigates the vulnerabilities presented by the current password management applications.
- iv. To design, develop, and test a prototype that validates the proposed process model.

1.4 Research questions

The following research questions were developed from the research objectives.

- i. Which data-centric techniques are used to protect data at rest, in-use, and in transit?
- ii. Which vulnerabilities are presented in current password management applications?
- iii. How can asymmetric encryption be used to redefine the process model of password sharing over the internet?
- iv. How can one apply an asymmetric encryption algorithm for generating public and private keys for encrypting and decrypting data?

1.5 Scope and limitation of the study

The study considered implementing a web-based portal, which served as a prototype for a system that facilitated secure password storage and sharing over a communication network. In addition, a study into secure asymmetric encryption assisted in implementing the prototype.

Several password management tools exist. They assist users in trying to achieve the entire suite of security best practices for passwords. This study was limited to redefining the process model of sharing sensitive data across the network, as implemented in the current password management applications.

1.6 Significance of the study

The existence of best security practices for passwords does not mean password users will adhere to them. (Bellovin & Merritt, 1992) Echoed that people will always use a simple password, forget them or even write them down. Due to these reasons, password misuse is one of the most extensively used forms of system breach (Keith, Steinbart, & P.J., 2007).

Studies have shown that encryption algorithms can be used to protect passwords, even if it is a bad one (Bellovin & Merritt, 1992). Thus, the study contributes to information security by ensuring passwords are stored in encrypted form within the users' selected storage location and championing the secure transmission of passwords, whether in encrypted or plain-text form. The research study is significant since it assists organizational employees and individuals operating joint accounts with a secure means of transmitting sensitive information securely over the internet.

2 Chapter Two: Literature Review

Strong passwords are essential in ensuring the security of data (O'Toole, Feeney, Heard, & Nainpally, 2018). They are also an integral part of the information system as the direct line of defense in protecting most components of information systems such as networks, devices, storage, databases, and files (Standridge, 2019). In summary, the chapter reviews literary materials detailing recommended security practices for passwords, an overview of cryptography and current password storage and sharing tools, presenting the benefits and the security risks of using them. Lastly, a conceptual model for the proposed prototype is provided to illustrate the vision of how to overcome the weakness presented in the reviewed systems.

2.1 Best security practices for passwords

To secure sensitive systems, a report by (O'Toole, Feeney, Heard, & Nainpally, 2018) defined the following standards as the best practices that organizations and individuals should employ.

The first standard defines that passwords should be at least eight characters in length. Length space allows the user to create a complex password consisting of a variant of characters. Also, the specified minimum size is considered a practical number of characters to remember without writing the sensitive information down.

The standard also specifies that passwords should not be made up of any readable word from any specific language or slang. The specification assists in preventing password breaches through attacks such as dictionary attacks. According to (O'Toole, Feeney, Heard, & Nainpally, 2018), the principle also applies to words with multiple capitalization schemes, constituting or symbols, numbers, and letters substituted for one another.

It is also specified that passwords should not be extracted from the name of an object or any particular phenomena related to the user, service, or its intended purpose. An example of a prohibited point of extraction for passwords is user's names, email addresses, phone numbers, date of birth, pets, etc. The rule also applies to the inverted forms of passwords extracted from phenomena related to the owner.

Best practices for passwords also entail avoiding the application of repetitive or sequential letters or numbers. The standard helps protect credentials against vulnerabilities such as brute-force and dictionary attacks where the characters' build-up of the credential can be generated quickly, thus exposing the system.

Users should avoid passwords and user ID reuse from one system to another. The key objective of the principle is to minimize the vulnerability in case one of the systems is breached. The other systems used by the user will be protected since there is no duplication of credentials, thus limiting the scope of access and damage by the intruder.

Another principle and a key to this study stipulate that a password or any credential for that matter should be kept private. Any system that requires administration or access from more than one principal should have the capability to self-authenticate multiple user accounts or use centralized authentication servers and protocols such as RADIUS, TACACS, and LDAP.

The last principle, which is also part of this study, states that passwords should be stored securely, preferably through ingrainings into users' or owner's memory. Notably, due to the above policies and standards, the user always ends up with passwords, which are hard to remember, thus requiring a means of storing and remembering them. Furthermore, sometimes users result in keeping the passwords in unencrypted text files. Also, the authenticating systems are required to store the passwords in an unreadable format.

2.2 The state of password usage as a daily activity

To draft practical ICT guidelines relating to password usage, organizations and individuals must obtain accurate information regarding how users manage passwords in their daily lives, taking into account their behavioral and managerial capabilities in a context where multiple passwords are in use (Grawemeyer & Johnson, 2009).

In line with security best practices for passwords, (Grawemeyer & Johnson, 2009) carried out an empirical study to investigate how individuals from various backgrounds utilize the password authentication mechanism in their daily life. The study encompassed 22 participants who had to create a daily diary of their password usage activities. The focus of the study was to investigate whether the participants made use of multiple passwords, the type or complexity of passwords created by users, the relationship between the password and the sensitive services they represent, password reuse, how the passwords are stored, and other strategies users employ to manage their credentials.

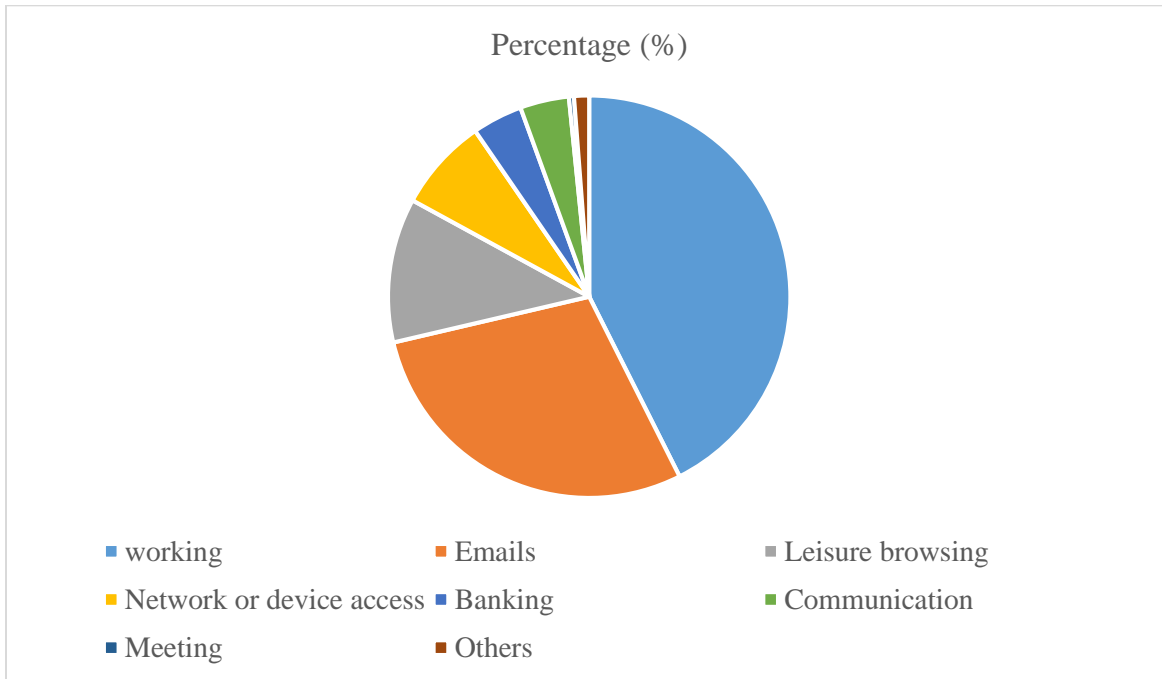
Within seven days, a total of 991 authentication requests which required users to use passwords were generated by the participants, thus enforcing the notion of passwords being the most commonly used authentication mechanism.

(Grawemeyer & Johnson, 2009) echoed that 43% of the participants required a password for work purposes, while email access and leisure internet browsing represented 29% and 12% respectively of total password usage of the participants. **Table 1** and **Figure 1** below illustrate the activities that contributed to password usage by the participants.

Table 1 showing the activities that require password authentication

Activity	Instances generated	Percentage (%)
working	422	42.58
Emails	285	28.76
Leisure browsing	115	11.60
Network or device access	74	7.47
Banking	40	4.04
Communication	39	3.94
Meeting	4	0.40
Others	12	1.21
Total instances	991	

Figure 1 showing the daily activities that required a password authentication mechanism



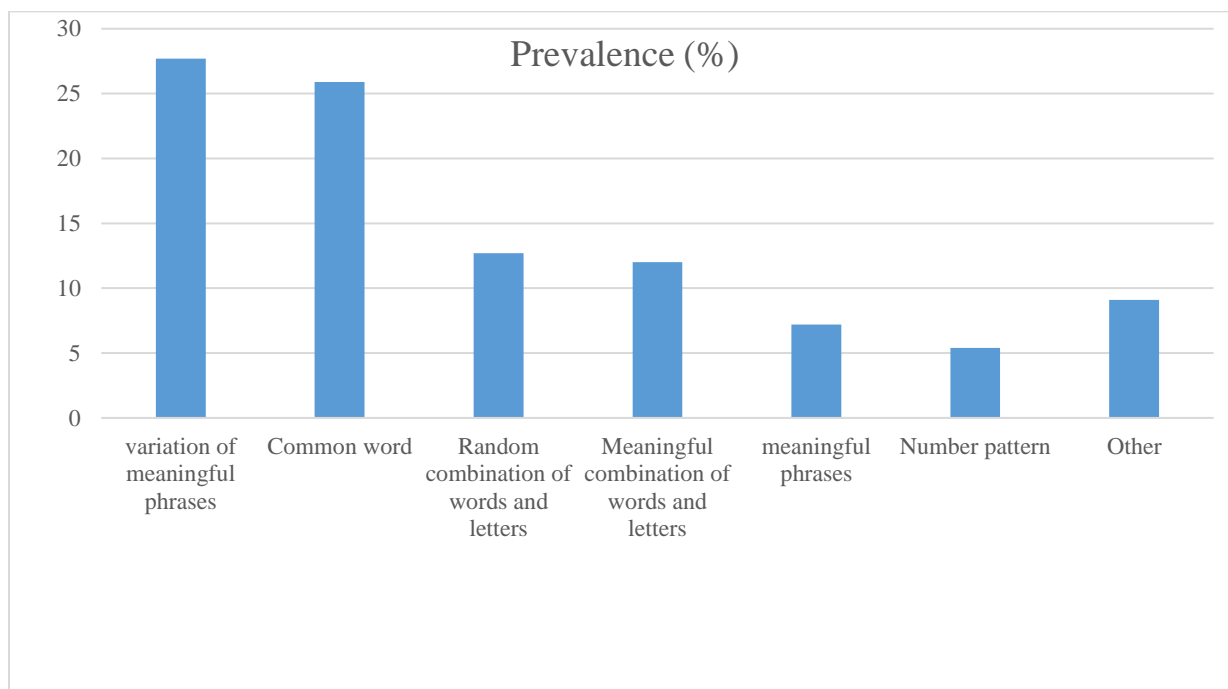
Security policies specify that users should employ the use of multiple passwords for different access accounts. The study by (Grawemeyer & Johnson, 2009) revealed that each user possessed an average of 8 passwords cutting across the various services they use.

(Charoen, 2014) recommends that the type of password should be complex, consisting of alphanumeric characters, fifteen characters in length, not be a word in any language, dialect, or slang, and not be based on any information that may relate to the user, device accessed, or the services in use. On the contrary, the study by (Grawemeyer & Johnson, 2009) revealed that the participants based their passwords on common words or names, meaningful phrases, simple sentences, and personal information, such as birthdays. Per the standard, some participants employed the use of alphanumeric passwords. Generally, the complexity of the password depended on the sensitivity of its purposes, such as banking and leisure. **Table 2** and **Figure 2** below are a representation of how users generated their types of passwords.

Table 2 showing the frequency of password types created by users

Password type	Prevalence (%)
variation of meaningful phrases	27.7
Common word	25.9
A random combination of words and letters	12.7
Meaningful combination of words and letters	12
meaningful phrases	7.2
Number pattern	5.4
Other	9.1

Figure 2 types of passwords generated by users



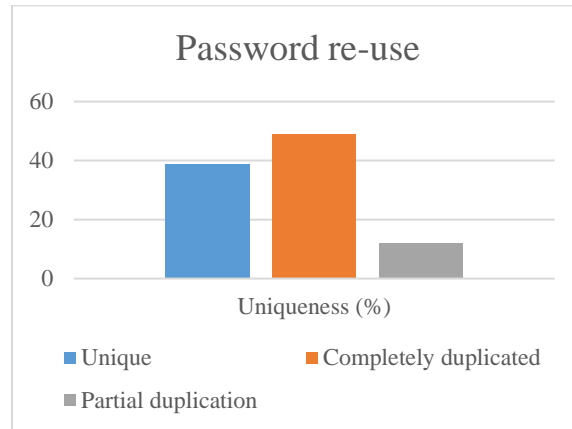
Security standards require that passwords should be changed regularly (Standridge, 2019). The rule applies whether the type of password is complex or straightforward. Out of 175 passwords, the study by (Grawemeyer & Johnson, 2009) revealed that the participants never changed 79% of passwords. 0.1% were changed more than once per year while 0.9 were changed once per year. Complex passwords are used to enhance password security. Still, in this instance, the participants opted not to review or change the password they consider as complex, thus infringing on the password policy.

The extent of password reuse can be implemented by duplicating the whole password from one service or platform to another and implementing part of the original passwords within the newly generated ones. The study by (Grawemeyer & Johnson, 2009) revealed that only 39% of the passwords were unique. The study also revealed that the frequency of password reuse was subject to the complexity of the password and the sensitivity of the system or the service in use. If a password is compromised, it can always be correlated to credentials used on other systems and services that are possible using the same ID and password, thus increasing the extent of the vulnerability (Haber, 2016). **Table 3** and **Figure 3** below show how the unique passwords were generated.

Table 3 showing the prevalence of password reuse

Password	Uniqueness (%)
Unique	39
Completely duplicated	49
Partial duplication	12

Figure 3 showing password reuse prevalence



Passwords should not be noted down on text files or a piece of paper for remembrance (Charoen, 2014). A good password policy should enable users to create practical and complex passwords that are easy to remember (Burnett , 2002). Despite the no-writing policy, it is notable that copying passwords onto text files and pieces of paper for storage and sharing is rampant, especially in the corporate world with stringent security password policies. The study by (Grawemeyer & Johnson, 2009) pointed out that at least 0.06% of passwords by the participants were written down. The statistic is relatively low, but (Grawemeyer & Johnson, 2009) attribute it to the prevalence and extent of password reuse, the type of the password, and the frequency in which the password is used.

2.3 Overview of Cryptography

Refers to the art of generating secrets to enhance privacy, confidentiality, authentication, integrity, non-repudiation, and to facilitate key exchange. The process begins with a readable message format known as plain text, which is then encrypted into an unreadable format known as the ciphertext, then eventually decrypted into a readable format (Kessler, 2020).

Below is an example of cryptography encryption and decryption functions:

$$C = E_k(P)$$
$$P = D_k(C)$$

Where **P** = plain-text, **C** = cipher text, **E** = the encryption method, **D** = the decryption method, and **k** = the key.

Cryptography algorithms are classified into three classes, namely

Symmetrical cryptography algorithms. They are also known as secret-key cryptography. It is a type of cryptography where a single key is used to perform data encryption and decryption. It involves algorithms such as Advanced Encryption Standard, Data Encryption Standard, and Blowfish.

Asymmetrical cryptography. Also known as public-key cryptography. It consists of the use of two keys. The public key for encryption and the private key for decryption. It involves encryption algorithms such as RSA and Diffie-Hellman algorithm

Hash Functions: Refers to irreversible mathematical transformations used to facilitate authentication or assess shared information integrity. It involves encryption algorithms such as SHA.

For the study, the focus domain was public-key cryptography using RSA since its application was used to facilitate the generation of keys, certificates, and encryption, and data decryption.

2.3.1 Public key cryptography

It involves using two derived keys, a public key advertised to everyone and a private key held privately for decryption. For example, assume there exists two individuals, Agnes and Bill. Agnes wants to send Bill a message. Agnes will encrypt the information she wants to send to Bill using Bill's public key. In return, Bill will decrypt the message using his private key to read the message. Other than encrypting and decrypting the message, the technique can also validate who sent the message. Validation is achieved when the sender encrypts the message using his or her private key, and then the receiver decrypts it using the sender's public key. One of the most popular implementations of public-key cryptography is the RSA algorithm. Below is the analysis of the algorithm.

2.3.1.1 Public key cryptography using RSA Algorithm

RSA is an encryption algorithm developed by Rivest, Shamir, and Adelman. The main functionalities of the RSA are to offer a means for facilitating key exchanges, generating digital signatures, and providing encryption to small blocks of data. Its implementation focuses on encrypting the session key used for secret key encryption (message integrity) or the encrypted message cipher (digital signature) (Kessler, 2020). It is considered one of the most secured asymmetric algorithms because its algorithms involve large numbers and the difficulty of generating prime factors for those large numbers. The key generation process is as follows:

1. Select two prime numbers, x , and y . Generate the modulus z , where $z=xy$.
2. Select number w , which is relatively prime (does not divide evenly into) to x and y . Where $w=(x-1)(y-1)$. Select the third number, e , that is relatively prime to (i.e., it does not divide evenly into) the product $(p-1)(q-1)$. The number e is the public exponent.
3. Compute private integer d , where d is the private exponent from the quotient $(wd-1)/[(p-1)(q-1)]$.

The public key is represented by the pair (z,w) . This information is made publicly available since it is not mathematically feasible to regenerate d from z and w if x and y are large enough.

The encryption function C is as below:

$$C = M^w \text{ mod } z$$

The decryption function M is as below:

$$M = C^d \text{ mod } z$$

Mathematically, the algorithm takes in a lot of computing power when the selected prime numbers are too large. Below is a mathematical example using purposefully selected small prime numbers.

1. Let $x=3$ and $y=5$.
2. Compute modulus z where $z = xy = 15$.
3. Select w which must be relatively prime to x and y . $(x-1)(y-1) = (2)(4) = 8$. Select $w=11$.

4. Compute d such that $(wd-1)/[(x-1)(y-1)]$ is an integer. Thus, the value $(11d-1)/[(2)(4)] = (11d-1)/8$. $d=3$ which is an integer.
5. Assume the message we want to send has a value of 7 (i.e., $M=7$).
6. The sender will use the public encryption function (M) using the public key pair $(w,z)=(11,15)$. Then, the ciphertext (C) is computed with the formula $C = 7^{11} \bmod 15 = 1977326743 \bmod 15 = 13$.
7. The receiver decrypts using private key pair $(d,z)=(3,15)$ and generates the plain-text with the formula $M = 13^3 \bmod 15 = 2197 \bmod 15 = 7$.

2.3.2 Hash Functions

This refers to one-way encryption algorithms that compute a fixed-value hash length on a plain-text, thus making it impossible for the regeneration of the previous content or the size encrypted.

It is mainly used to authenticate the contents of a file, thus validating its integrity. Another popular usage of hash functions is the encryption of passwords within the file system. It is also used in public-key algorithms to facilitate password logins, encryption key management, and generation of digital signatures. **Table 4** below is a summary of the various versions of the Secured Hashing Algorithms (SHA)

Table 4 showing various versions of the SHA algorithm

	SHA-1	SHA-224	SHA-256	SHA-384	SHA-512
Message digest	160	224	256	384	512
Message size	$< 2^{64}$	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Block size	512	512	512	1024	1024
Word size	32	32	64	64	64
Number of steps	80	64	64	80	80

2.3.3 Digital Signatures

Refers to electronic signatures used for authenticating the identity or origin of a document and to validate the integrity of the received document. It involves asymmetric cryptography consisting of two functions: signing using the private key and verifying using the public key. The signature creation is always performed by the signer using a unique hash generated using a private key. The authenticity of a signature is assured since there is no possibility of a similar signature generation using a different pair of messages and private keys. The signature verification is accomplished using the public key concerning the message sent. The proposed prototype intends to incorporate digital signatures to confirm the source of requests before sensitive information is dispatched.

2.4 Related systems - Current password storage and sharing tools

Last Pass, Keepass, and Dashlane are some of the most popular password management tools recommended by most Information Technology security experts (Standridge, 2019). These tools have provided a drastic shift from the traditional password management and storage practices such as storing credentials on a piece of paper or notebook, storing passwords on unencrypted files, and storing credentials on the browsers for easier remembrance.

According to a study by (Rubenking, 2015), which was also reiterated by (Standridge, 2019), Last Pass, Keepass, and Dashlane provide the following features at a minimum to enhance the security of passwords.

i. Credentials sharing among users

This feature is the primary focus of the study. The tools, as mentioned earlier, have provided their users with a resource-sharing feature to secure how these credentials are protected during transmission. The passwords are secured during transmission using TLS. Each user is attributed with a public key for encrypting data and a private key for decrypting data during account creation. These keys are then secured within a vault using built-in encryption.

ii. Creation of unique and strong passwords

The tools assist users in generating passwords in line with the security best practices. Factoring the length, format, and characters set, the passwords generated for the users are complex enough to sustain attempts such as dictionary attacks, password cracking, and other forms of password attacks.

iii. Safely store passwords

To ensure that the information stored is secured or less critical for use in case of a breach, these password management tools offer encryption for protecting the passwords stored within the vault. A master password, which the portal owner holds, is used as the decryption mechanism for accessing the contents stored by the tools.

iv. Bookmarks and Auto-login to websites

The tools provide a feature for storing websites URL and credentials for accessing the portals hosted on the URL. This information is later used to give auto-fill to login pages of the bookmarked URLs.

v. Password synchronization across multiple devices

This feature allows users to install the password management tools on multiple devices, then configure the master password across them for access. It helps in distributing the information stored within the vault across multiple endpoints for access anywhere and at any time.

vi. Provide access to your passwords from a public device

When the application has not been installed on the devices, these tools allow users to access the vault through a public website. The synchronization between the public website and the local portal is executed via scripts embedded within the local browser (Standridge, 2019).

vii. Multi-factor authentication

It refers to an authentication mechanism using what users' have and what users know.

2.4.1 LastPass password manager & security architecture

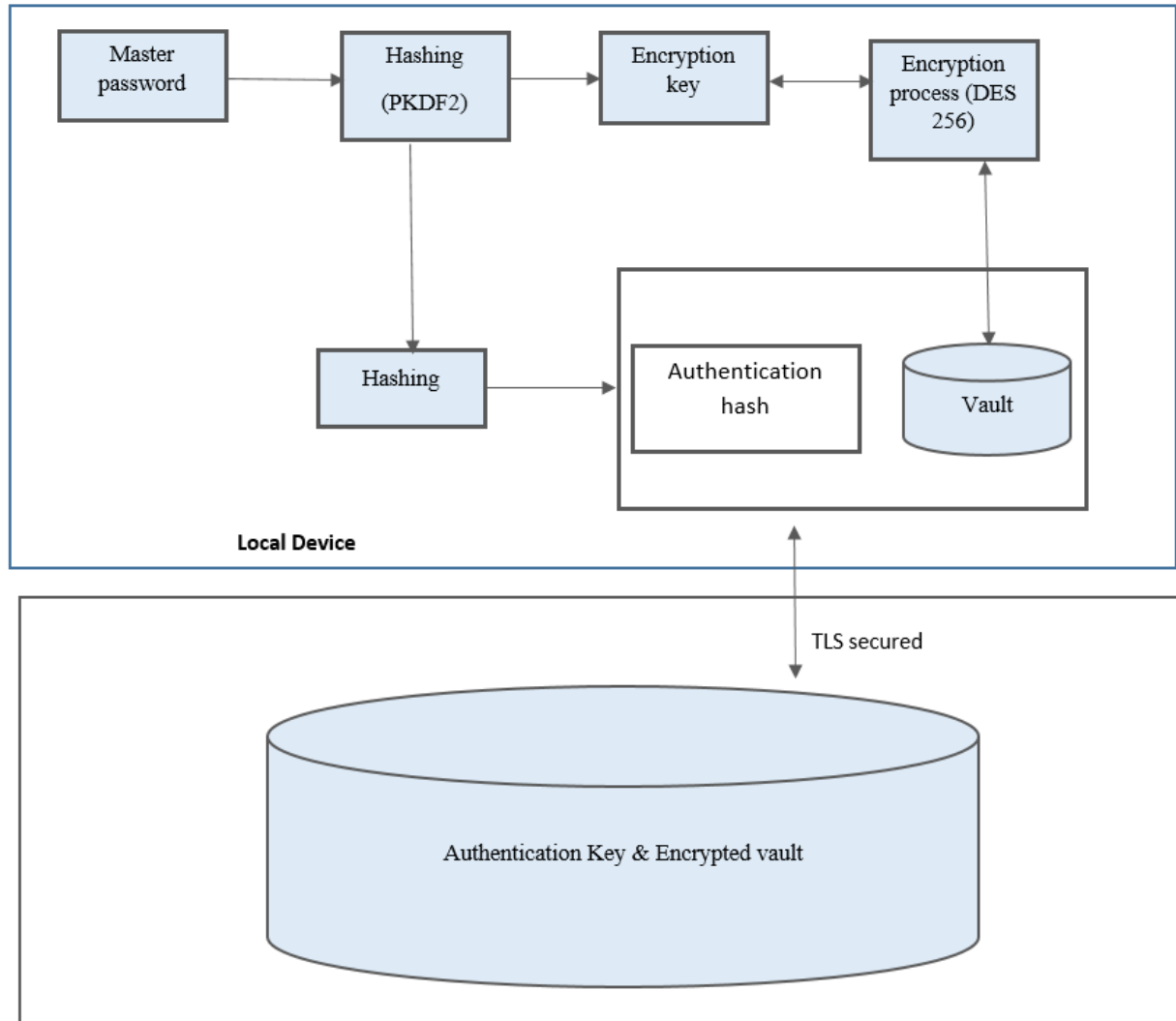
LastPass is a password management application that helps users enhance their productivity and minimizes the chances of password-related breaches during creation, storage, usage, and sharing (Standridge, 2019). The security premise of the password manager revolves around a master password, which is created during the registration process. Then, the credential authenticates into the portal via the browser extension or by the remote web portal.

LastPass portal or vault is used to manage credentials and identities or links (URL) for other resources, thus bringing out its functionality as a password manager. The user can conduct operations such as adding, viewing, and managing credentials and other resources saved within the LastPass vault. Principal access to the vault is facilitated using the correct username and Master Password.

A report by (Standridge, 2019) indicated that remembering the Master Password is offered on web browser extension and the mobile application versions of LastPass. The functionality reduces the effort on the user's end to keep remembering the master password. However, on the contrary, it exposes the platform to a security vulnerability since the password can be acquired in un-encrypted form from the application, thus providing an opportunity to decrypt the owners' vault by an intruder.

The application security design follows the 'local-only encryption' model. It means that the user data stored in the vault can only be encrypted and accessed from the user's local computer. The distribution of users' data to local storage eliminates the need to store the master password on LastPass remote servers. Once user data is encrypted using the unique key stored locally, the data is synchronized to LastPass remote secure storage. (Standridge, 2019) Also reported, LastPass cannot decrypt user's data due to lack of access to the master password.

Figure 4 shows the security architecture of the LastPass password manager



Regarding **Figure 4**, LastPass uses the master password and the username as the salt to generate the encryption during account creation. One hundred thousand one hundred rounds of PBKDF2-SHA256 are used to create the encryption keys on the client-side. In addition, another round of hashing is performed on the master password, after which it is sent to the remote server for storage. The remotely stored credential helps during the log-in process on the LastPass domain and for the addition of extra devices for local storage.

Despite all of the above security measures, the report by (Standridge, 2019) also indicated that researchers were able to retrieve one-time recovery passwords (ROTP) from LastPass local storage, then use them to gain access to the victim's local storage. To aggravate the situation, the retrieved credentials gained access from anywhere since it bypassed counter-measures such as multi-factor authentication.

Another vulnerability (Standridge, 2019) reported is acquiring the user's master password during an active login session on the local computer. The exposure revealed that the key to the encrypted local database was stored within the local computer. Thus, access to the key gave way to the entire database contents. The techniques used to access the keys were cross-side scripting (CSS) and cross-side request forgery (CSRF).

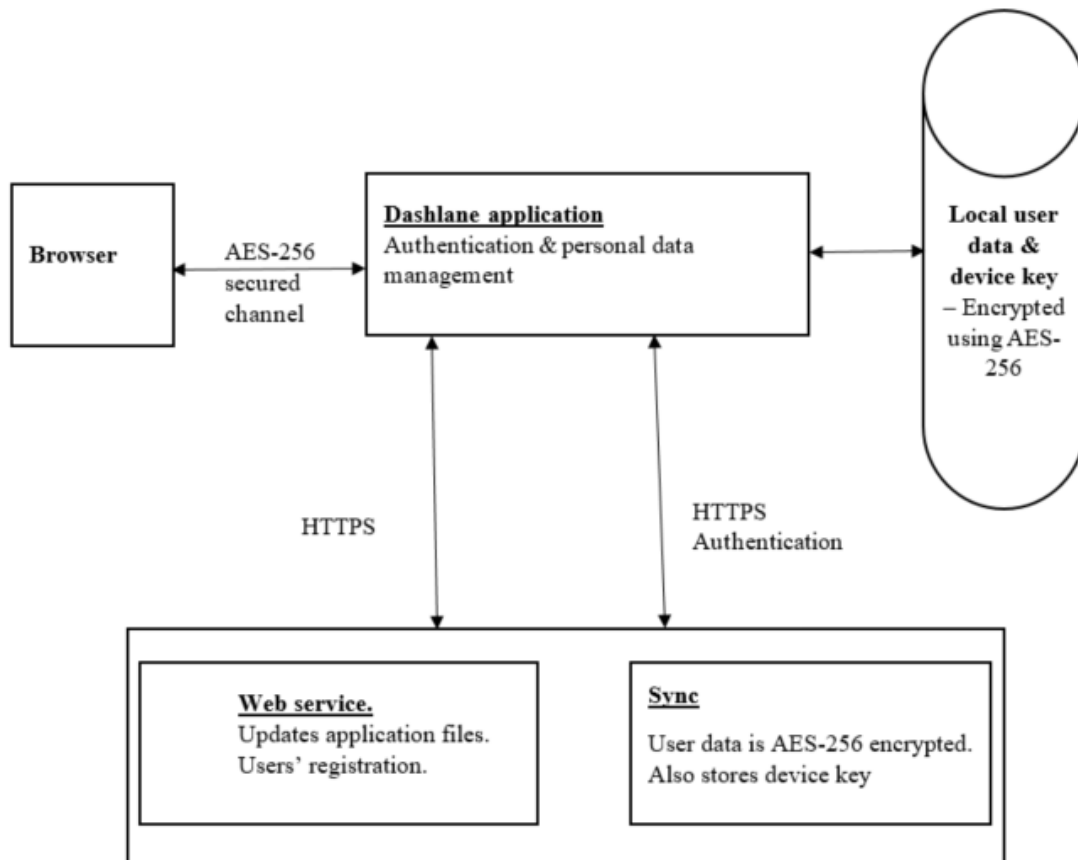
Partial encryption of the user’s vault was also presented as a security vulnerability. The URL or the identity of the resources whose corresponding passwords were stored in the vault were maintained in plain text passwords.

2.4.2 Dashlane password manager & security architecture

The password management application has a security premise similar to that of LastPass. In **Figure 5**, the master password is used to generate a key, which is used to encrypt the password vault. Ten thousand rounds of PBKDF and SHA-256 are used to create the encryption key on the client machine only. The differentiating factor to the LastPass application is that the master password is never stored nor sent.

The weakness presented by Dashlane is the optional password-changing module implanted on the server-side. The passwords are sent between the server and the client machine without encryption using the previously generated key during this action. However, the platform relies on SSL for protection. This functionality renders the transmitted passwords vulnerable if intercepted by an intruder. Furthermore, according to Dashlane, the passwords are deleted on the server-side once the process is completed (Fowler, 2014).

Figure 5 shows the security architecture of the Dashlane password manager



2.4.3 KeePass password manager & security architecture

KeePass is a password management application that offers no server or cloud component for managing passwords (Standridge, 2019). Instead, the entire application service is implemented locally on users’

devices, such as computers or a universal serial bus (USB). The application offers the following choices as modes of password protection.

- i. A key file is stored locally on the computer or USB drive.
- ii. Master password with an option to use it with the key file.
- iii. Operating system username and password for use on Windows operating system.

SHA-256 is used to generate an encryption key from a long list of values such as date, time, cursor position, and performance counters. By default, 6000 rounds of encryption are used to generate the key. However, the number of rounds is customizable. The formula below is used if the key file is used together with the master password:

SHA-256 (SHA- 256 (password), key file contents).

In the year 2015, a vulnerability was revealed within the export function (Standridge, 2019). Upon the discovery of the exposure, a tool known as KeeFarce was released to conduct exploitations. The tool was used to copy the contents of an active KeePass database into a CSV file. The resulting file contained the user names, passwords, notes, and URLs stored in the database in plain text. The only limitation to the vulnerability is that an adversary needs access to the local device with administrative privileges before the malware or the tool could be deployed and executed. The vulnerability is also reported to exploit other password-management applications (Standridge, 2019).

2.5 Other password sharing techniques and tools

Besides the password management tools such as LastPass, a study by (Standridge, 2019) has identified the following as the standard modes of password sharing.

2.5.1 Use of notebook or paper

Refers to a scenario where the passwords are written down as notes on a piece of paper. It's the most popular manual method for storing passwords. One of the advantages that come with the techniques is that resources are kept offline. On the downside, the process of keeping, managing, and tracking records is very tedious (Standridge, 2019).

2.5.2 Storing passwords unencrypted in a file on a connected device

Refers to a case where passwords are stored or typed in a file, then stored within the device from which they will be used.

2.5.3 Storing passwords using browsers

Modern browsers, such as Google Chrome, Microsoft Edge, and Mozilla Firefox allow users to store their passwords for more effortless future authentication to resources accessed via the browser. In addition, chrome and Microsoft Edge offer storage in an unencrypted format, while Mozilla Firefox provides encryption and decryption functionalities.

2.6 Conceptual model

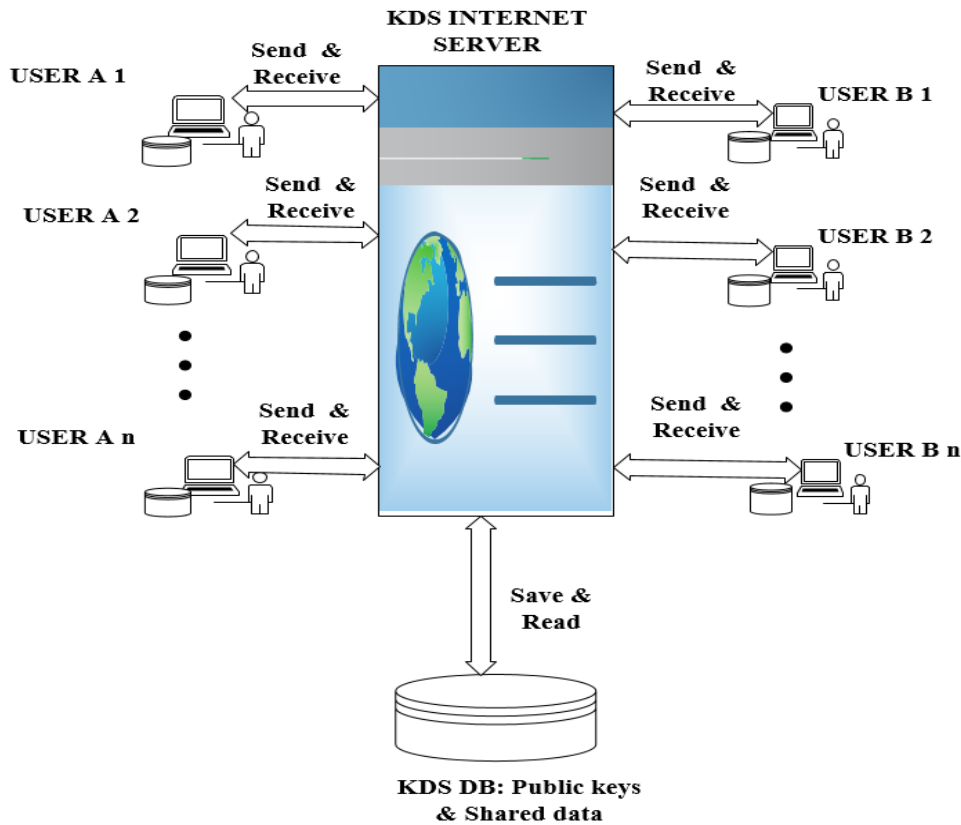
Figure 6 illustrates the proposed system model, which is a web-based portal consisting of the following:

- i. *Local client computers or endpoints* – Hosts local encrypted lightweight SQLite database. The local storage holds owner registration details (names, passwords, device id, and network address), public-private keys pairs, and sensitive data (passwords). The keys are generated during registration

on the local computer using the OpenSSL RSA algorithm. The pair's public key is used to encrypt data as they are stored in the database, while the private key is used for decryption during data access.

- ii. **Key distribution & requests logging server** - The server is used to store public keys and the shared data temporarily.
- iii. **OpenSSL RSA** encryption is used to generate public-private keys pairs at users' endpoints.
- iv. **A secure socket layer (SSL)** is used to secure data in transit between the clients (sender and receiver) and the central server REST API.

Figure 6 representing the conceptual model of the proposed system



2.6.1 Authentication process

The principal first name, last name, email, device MAC address, and the device IP address on the network are captured during the registration process. In addition, a pair of public and private keys are generated for the user on the local system. The pair's public key is then used to encrypt the user-captured data before storage into the local database.

During user access to the portal, a pair of the email, password, and device MAC address is used to authenticate against the principal. This authentication limits access to the stored contents to the registered device only. During the process, the system will also retrieve the private key, encrypted principal email, encrypted password, and encrypted device MAC address stored in the local database.

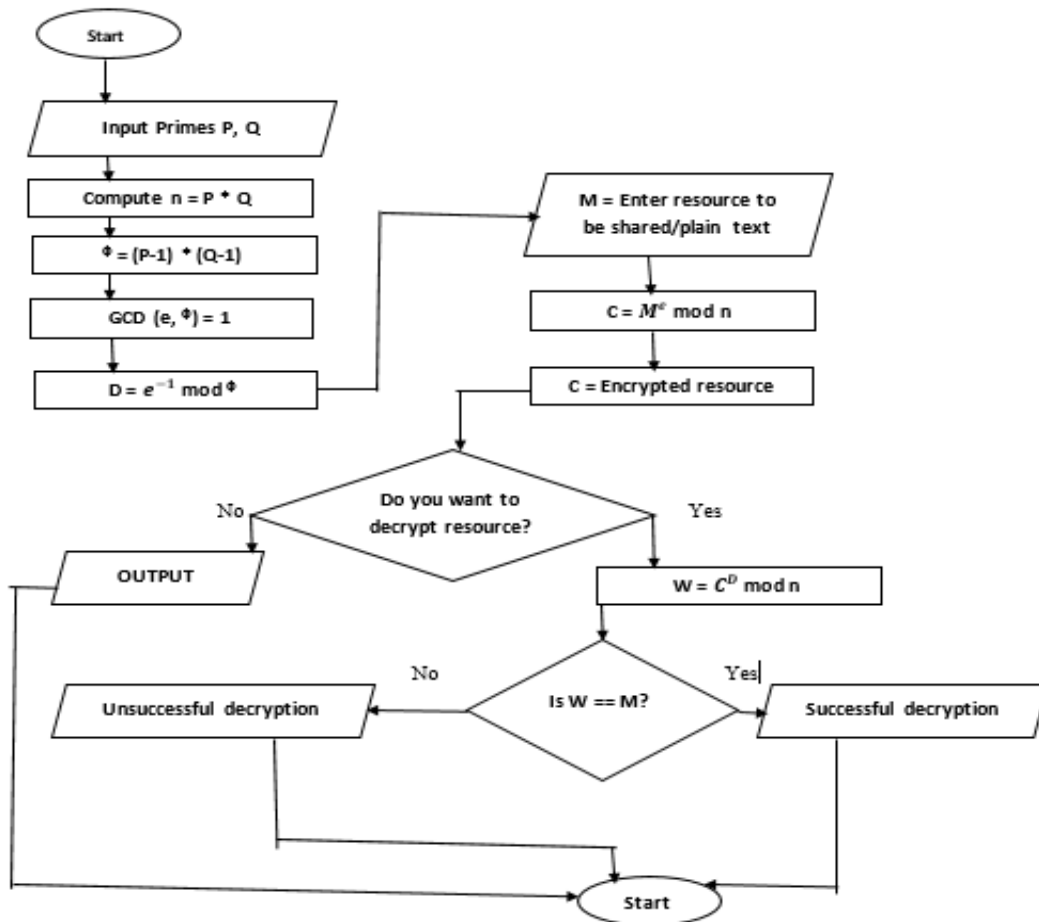
The private key is used to decrypt the retrieved principal's details. The decrypted tokens are then compared to the ones submitted by the principal. A match for each item marks a successful

authentication process. Public and the private key session is created for future encryptions and decryptions as the principal is granted access to the portal. There is no synchronization of the sensitive data (both the principal identity and the stored resources) to a central server, thus eliminating the single point of attack where all users' keys and data can be found.

2.6.2 Encryption and decryption process

Data and keys storage is done on the principal's local computer. **Figure 7** shows how the OpenSSL RSA algorithm generates the public and the private key during the registration process. The generated keys are used to encrypt and decrypt all subsequent data stored into the local database; this includes the principal's identity credential

Figure 7 showing the encryption and decryption process using RSA



2.6.3 Data sharing process

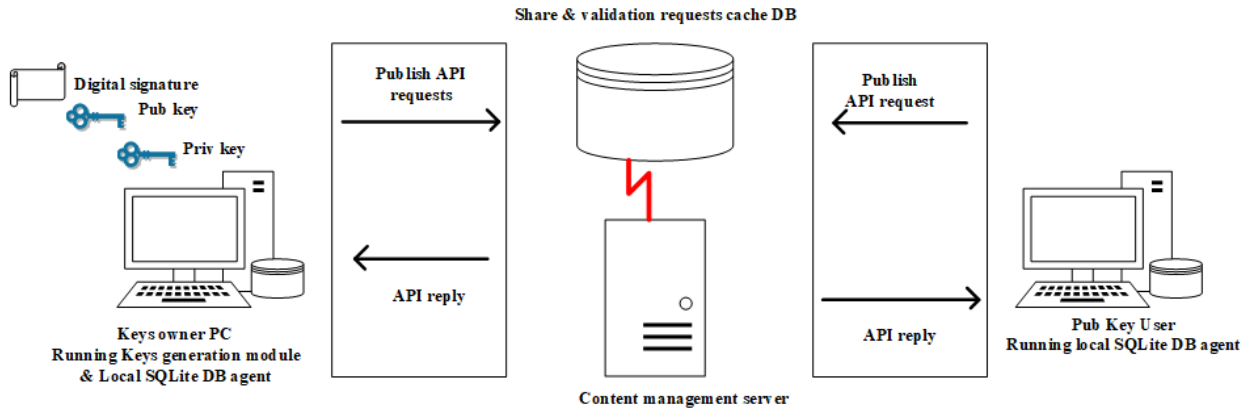
A central server is implemented to serve as a temporary cache repository of the shared resource and to act as the public keys distribution center. For instance, a data share request has a data structure, as shown in **Table 5** below.

Table 5 representing the structure of the ‘data share’ request format

Sender email	Recipient email	Shared resource ID	Shared information
sender@example.com	receiver@example.com	11	Data

Figure 8 shows how the recipient uses data components of the request to invoke an API process to the central server. The recipients of the shared data will receive a notification of the share request via email address. The messages carry no payload of the transmitted data but act as an alert to prompt the recipients to access their vault. The shared resource will be automatically synchronized to the local database through the API.

Figure 8 representing how data is shared in the proposed system



3 Chapter Three: Methodology

3.1 Research design

The research design refers to the strategy employed to conduct the study (Oates, 2006). The exploratory research design method was adopted to obtain and analyze the necessary data using quantitative and qualitative approaches. Exploratory research was the preferred strategy since it assisted in providing answers and gaining familiarity with an existing phenomenon such as password users' awareness on security best practices, thus acquiring insight into it to formulate a more specific research problem (B, 2020).

Design science research was used since the study aimed to develop valid and reliable knowledge for designing solutions, utilize the gained knowledge to solve problems, and create change or improve existing solutions. The result of the research was an artifact or a product (Pello, 2018). The research process involved problem identification, the definition of objectives for the solution, design & development, demonstration, evaluation & communication.

The qualitative approach method was incorporated into the research to provide reason and description to the answers generated to the research question. It was used to represent the outcome of the research study in non-numerical form. The quantitative approach was used in conducting a numerical analysis of the data generated during the research process.

The study's methodology was guided by specific research objectives, such as to design and implement a system that uses asymmetric encryption algorithms to validate access to shared sensitive data (password). To cater for the specified requirements, the research study involved two phases. The first phase entailed system analysis, design, and implementation. The second phase involved a focus group discussion with selected members of the public in evaluating the developed system.

3.2 Phase 1: System design and development

The primary goal of this phase was to develop a prototype of a system that will assist in enhancing secured password sharing. The prototype development used the waterfall model for the software development life cycle (SDLC). The waterfall model refers to a sequential software development approach in which one phase or component must be completed before proceeding to the next phase. Below are the stages of the development cycle.

3.2.1 System requirements definition and analysis

The study involved gathering all the requirements necessary for implementing the model for password sharing. The requirements were used to provide a guideline of the functionalities the system had to incorporate and define the system's components.

3.2.1.1 Software and hardware requirements

Varieties of software and hardware tools were used to develop and simulate the system prototype. **Table 6** shows the list of client endpoints, server endpoints, and communication network requirements needed to demonstrate password sharing over a communication network. Also, an encryption channel or algorithm was required to facilitate encryption during data transmission.

Table 6 shows the software and hardware requirements for the system prototype

Software / Hardware	Version	Purpose
Oracle Virtual Box	6.1.22 r44080	To act as the hypervisor for the virtualization and simulation of

		the communicating clients, servers, and the communication network
WIN-10-VM-01	A virtual machine running Microsoft Windows 10 Home Edition	To simulate one of the client computers, in this case, called USER A
WIN-10-VM-02	A virtual machine running Microsoft Windows 10 Home Edition	To simulate one of the client computers, in this case, called USER B
WIN-10-VM-03	A virtual machine running Microsoft Windows 10 Home Edition	To simulate the central SERVER that facilitates sharing of data between USER A and USER B .
Microsoft Windows OS	10 Home Edition	To simulate the operating system at the client's and server's end.
Xampp (Apache, MySQL & Mercury) application	3.2.4	To provide web server functionality for PHP programming language, which the prototype is built on. It also provides the MySQL database client at the server end. The Mercury application is used to deliver mail service to the users during sharing.
SQLite3	3.35.4	To provide local database storage at the client end (WIN-10-VM-01 and WIN-10-VM-02).
OpenSSL	1.1.1k	To act as the encryption scheme for generating the keys at the clients' end and conducting encryptions.

3.2.1.2 Functional requirements

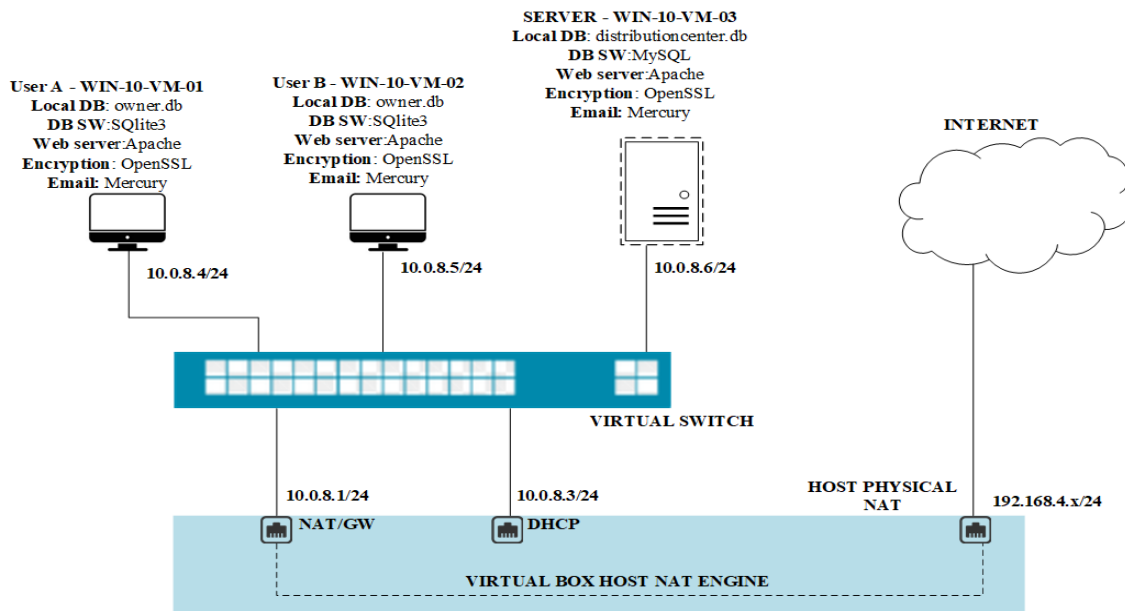
The system had the following proposed functionalities at a minimum.

- i. A web-based interface to facilitate registration, login, and access to the platform.
- ii. The system required an asymmetrical encryption mechanism for generating public, private, and digital keys for the users. The encryption mechanism was used to encrypt the stored contents. In this case, the OpenSSL library was used to generate the public, private and digital signatures.
- iii. A communication channel, such as email, to facilitate the passing of certificates and the shared resource. In this instance, the Mercury Xampp application was used to provide email service for communications dispatch.
- iv. Identity or digital signatures verification mechanism between users
- v. Persistent structured storage for storing keys and resources shared between users. The SQLite3 was used to facilitate persistent storage on the client's end, while the MySQL database enabled request caching and users' data storage on the server end.

3.2.2 System Design or architecture

On completion of requirements specifications, the organization and interrelationship of the system components are integrated, as shown in **Figure 9**. The architecture was used as a guideline during the implementation phase. The system components consisted of **SERVER (WIN-10-VM-03)**, **USER A (WIN-10-VM-01)**, **USER B (WIN-10-VM-02)**, and a communication network. The Virtual Box hypervisor was installed on the host computer, and then the guest machines mentioned earlier were installed. The NAT Network mode was enabled on the hypervisor to ensure the guest machines can communicate with each other and access the internet simultaneously. The SQLite3 database, OpenSSL encryption scheme, Mercury mail service, and the Apache web servers were installed on the client computers (**WIN-10-VM-01 and WIN-10-VM-02**). Also, a database called ‘*owner.db*’ was created at the client end to facilitate storage of the generated keys and the encrypted sensitive data. The MySQL database, Apache web server, and the Mercury mail server were installed on the **SERVER (WIN-10-VM-03)**.

Figure 9 shows the architecture of the system



The following database schemas were created for the ‘*owner.db*’ located at the client computers (**WIN-10-VM-01 and WIN-10-VM-02**). **Table 7** was used to store user’s data and keys during the registration process. **Table 8** was implemented to store passwords, which the user will share later, while **Table 9** was used to keep the recipient's identity and the resource being shared.

Table 7 showing the users’ account table at the local database

Myaccount table/relation			
Field name	Data type	Primary Key	Foreign Key
Id	Integer	Yes	
Email	Text		
Password	Text		
Pubkey	Text		

Privkey	Text		
Signature	Text		
Device	Text		
Firstname	Text		
Lastname	Text		
Date	Text		
address	Text		

Table 8 showing the resources' table at the users' local database

Resources table/relation			
Field name	Data type	Primary key	Foreign key
Resource_id	Integer	Yes	
Name	Text		
Data	Text		
Share_status	Text		
Date	Text		

Table 9 showing the shared resources table at the users' local database

Share_center table/relation			
Field name	Data type	Primary key	Foreign key
Share_id	Integer	Yes	
Recipient_email	Text		
Resource_shared	Integer		Yes
Transmission_status	Text		

'Dictributioncenter' database was created in the *SERVER (WIN-10-VM-03)*. Table 10 stores users' public data such as email and public key synchronized to the central server. Table 11 temporarily stores the shared data until the recipient synchronizes it to his/her local database.

Table 10 shows the registered users table at the central distribution server

Users table/relation			
Field name	Data type	Primary key	Foreign key
User_id	Integer	Yes	
Email	Text		
Firstname	Text		
Lastname	Text		
Password	Text		
Pubkey	Text		
Primary_device	Text		
Address	Text		
data	Text		

Table 11 shows the shared resources cache or synchronization table

Share_center table/relation			
-----------------------------	--	--	--

Field name	Data type	Primary key	Foreign key
Share_id	Integer	Yes	
Sender_email	Text		
Receiver_email	Text		
Resource_id	Integer		Yes
Resource_name	Text		
Resource_data	Text		
Location	Text		
Transmission_status	Text		
Date	Text		

3.2.3 Implementation phase – GUI screens

Each program component (encrypted database, password management portal, and communication module) was developed and tested independently. The method of testing the features is referred to as unit testing. Later on, the components were integrated to form one system. The implemented system components are shown in the screenshots below.

Figure 10 shows the system's registration component screen, which was used to fetch users' names, email, and passwords. These details formed the identity of the users in the system. On form submission, a pair of public and private keys were generated for the users. The user data were encrypted with the public key before storage in the local database. The screen also provided a login form for users to access the local portal.

Figure 10 shows the login and registration screen for the local portal

The screenshot displays the 'Password & Key Management Portal' interface. It is divided into two main sections: 'Already have an account?' and 'Create an account'.

The 'Already have an account?' section features a login form with a 'Login' button. It includes a 'forgot your password?' link and a lock icon representing encryption.

The 'Create an account' section features a registration form with a 'Register' button. It includes fields for 'Your first name', 'Your last name', 'Your email address', 'Your password', and 'Password confirmation', along with a plus icon representing account creation.

Figure 11 shows the local portal or vault screen after the user's successful authentication via the login form. The screen provides the user with a button to save a password(s) into the system's local database. The user is also presented with an edit button for future modification of the stored passwords. In addition to the edit button, the user is also provided with a button to delete the stored password(s). The screen also shows the sharing button, which enabled the portal user to share the stored password with another person. Lastly, the lock icon on the screen was used to symbolize that the stored password is encrypted.

Figure 11 shows the dashboard of the implemented prototype

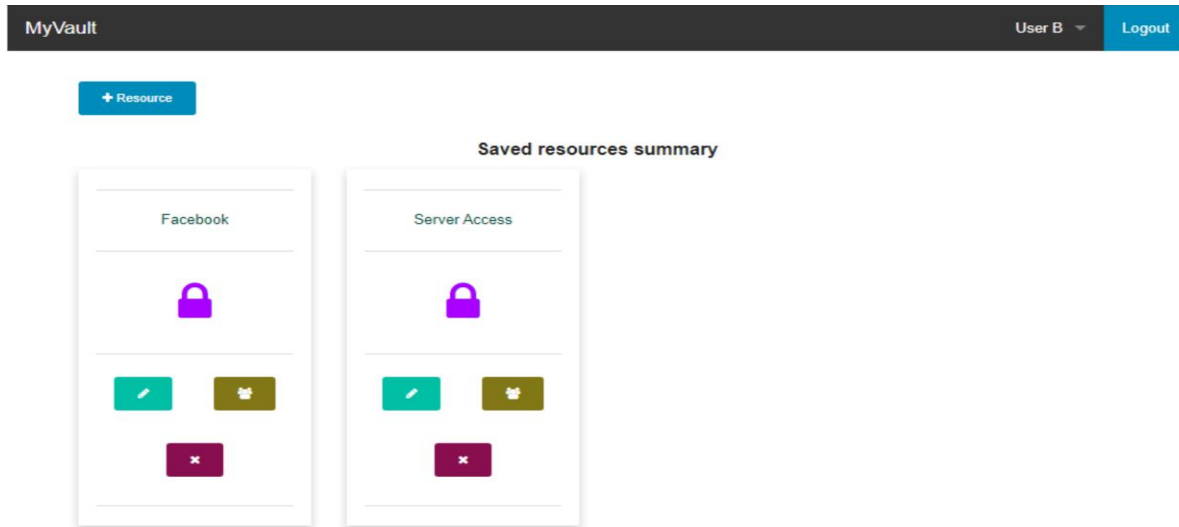


Figure 12 shows the form presented to the user when they click on the add resource button. The form captures the password details before it is encrypted, then stored into the local database. The details captured are the password and a brief description of it.

Figure 12 shows the screen for creating or saving a resource in the portal

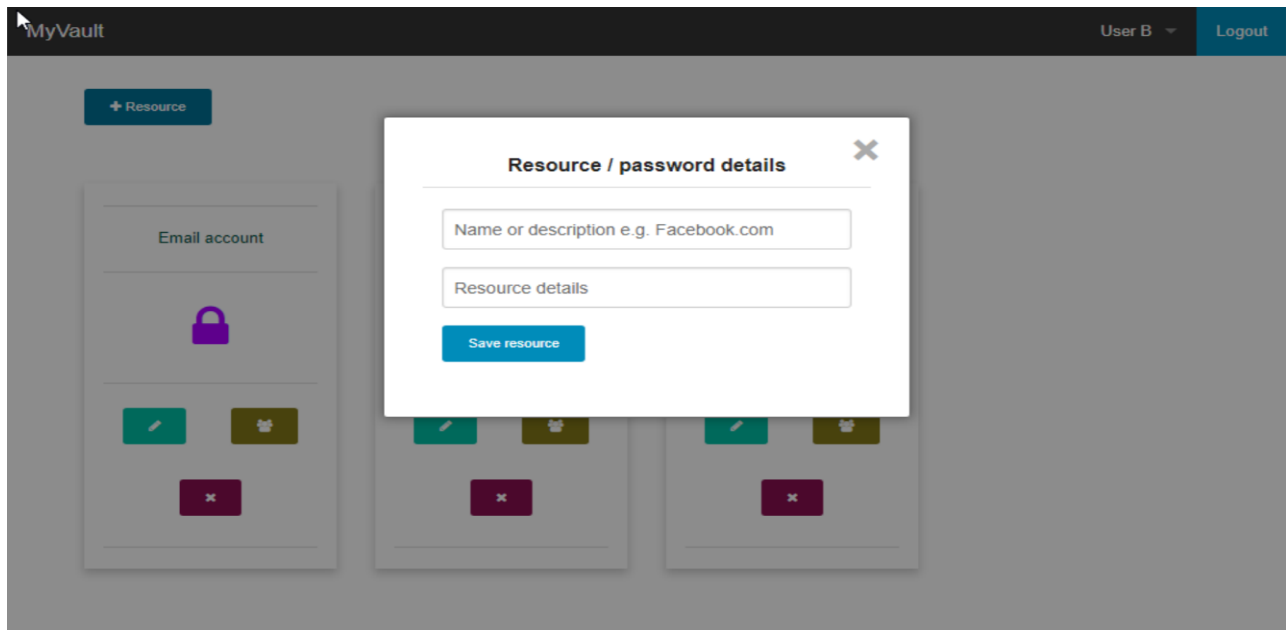


Figure 13 shows the screen or the form presented to the user when they click the shared button on the saved resource. The form was used to prompt the user to provide the email address of the receiver of the resource. On form submission, the application will request the server for the user's public key represented by the

provided email address. The system will then use the public key to encrypt the shared password before sending it to the central server for temporary storage as it awaits the recipient to pull the data to his/her local database.

Figure 13 shows the screen for sharing resource or password

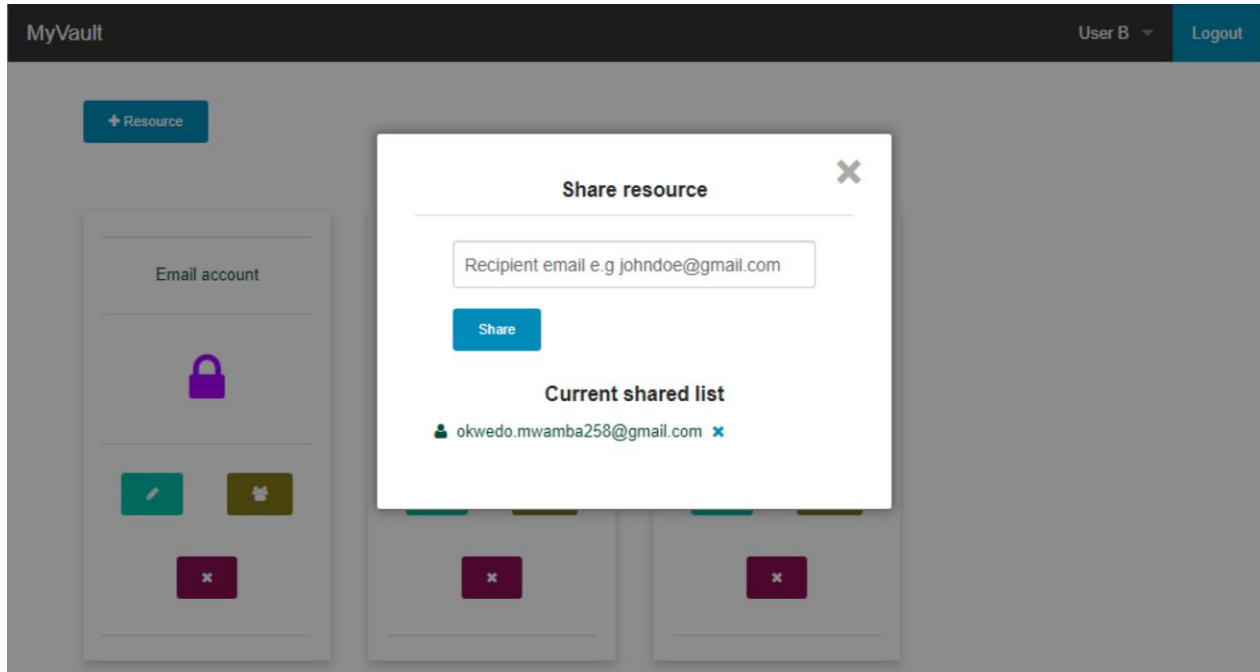


Figure 14 shows a screen from the central server portal. The screen displays a list of users whose details such as names, the public key, and the registered device have been synchronized to the central key distribution server. The screen is used for monitoring purposes only. User registration and data storage processes are all decentralized to the user's local application and database.

Figure 14 showing a list of registered users from the server administration screen

User id	User names	Email	Ip Address	Device MAC	Date
4	User B	jmwamba35@outlook.com	10.0.8.4	08-00-27-7D-83-31	2021-May-Sun
3	Joseph Mwamba	okwedo.mwamba258@gmail.com	10.0.8.5	08-00-27-E8-F8-C9	2021-May-Sat

Figure 15 shows the list of the temporarily stored shared data. The server only stores the id of the password, the email of the sender and the receiver, and lastly, the status of the transmission. First, the transmission

status is marked as completed, and then the shared data is deleted from the server once the recipient has to pull the transmitted data to his or her local database.

Figure 15 shows a list of resource share requests or logs from the server administration screen

The screenshot shows the MyVault interface with a navigation bar containing 'MyVault', 'Users', 'Shared resource', and 'Logout'. Below the navigation bar is a table titled 'Shared resource' with the following data:

Share id	Sender email	Receiver email	Resource id	Transmission status	Date
21	jmwamba35@outlook.com	okwedo.mwamba258@gmail.com	18	Complete	2021-Jun-Tue
20	jmwamba35@outlook.com	okwedo.mwamba258@gmail.com	17	Complete	2021-May-Sun
19	okwedo.mwamba258@gmail.com	jmwamba35@outlook.com	17	Complete	2021-May-Sat

3.2.4 Integration and testing

The phase involved bringing together all the components of the password-sharing system into one fully functional system. The modules went through unit testing to ensure seamless compatibility and functionality in relation to one another. Below were the test cases for the integrated system.

3.2.4.1 Registration module testing

a) Sanitization and verification of the form data

The objective of performing this test was to ensure that the user submits sufficient information required to access the system and generate the necessary keys, as shown in **Figure 16** and **Figure 17**. Therefore, the test case was defined as *no empty field was allowed on the form* to display the error message in case of one. If all the required fields are provided, the keys are generated, the public key is used to encrypt the user details, then they are all populated into the local database.

Figure 16 showing form validation test error message

The screenshot shows a registration form with the following fields and a validation error message:

Please provide your first name

Your first name

Your last name

Your email address

Your password

Password confirmation

Register

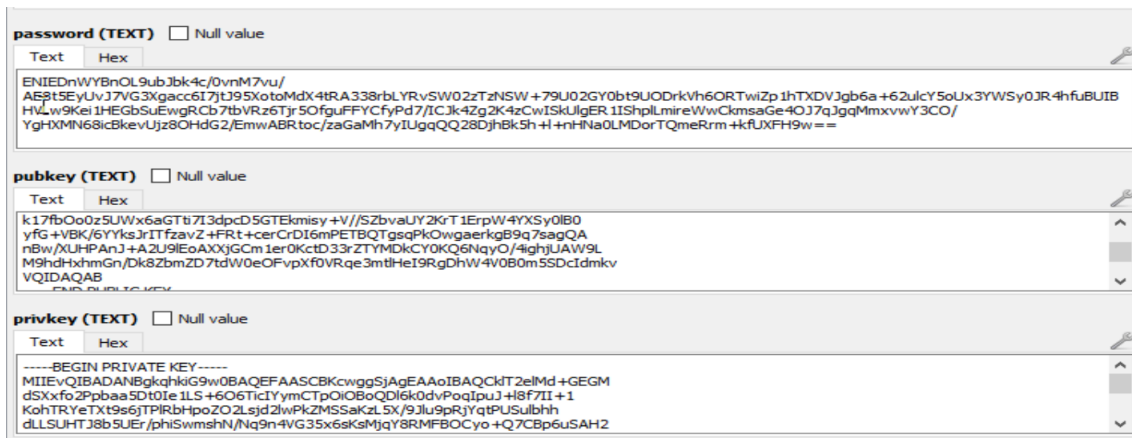
Figure 17 showing an alert message on successful registration



b) Ability to generate a public key, private key, and digital certificate

The test case ensured that the registered user has public and private key pairs assigned, as shown in Figure 18. The public and the private keys were generated using the OpenSSL library.

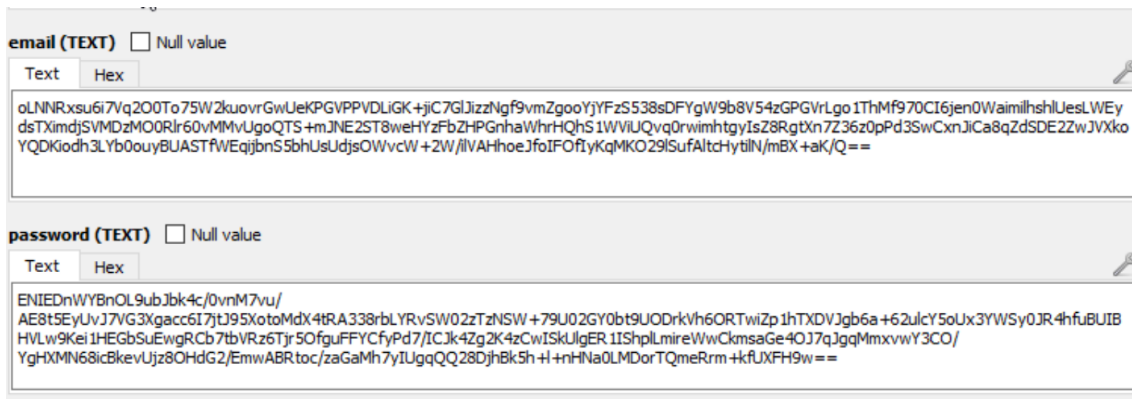
Figure 18 shows the keys and the signature assigned after user registration



c) Encrypt the submitted data using the public key

The test case ensured that the users' credentials stored in the database are encrypted using the public key. Figure 19 shows the encrypted format of the data in the database after the public key was applied.

Figure 19 showing user data encrypted in the database

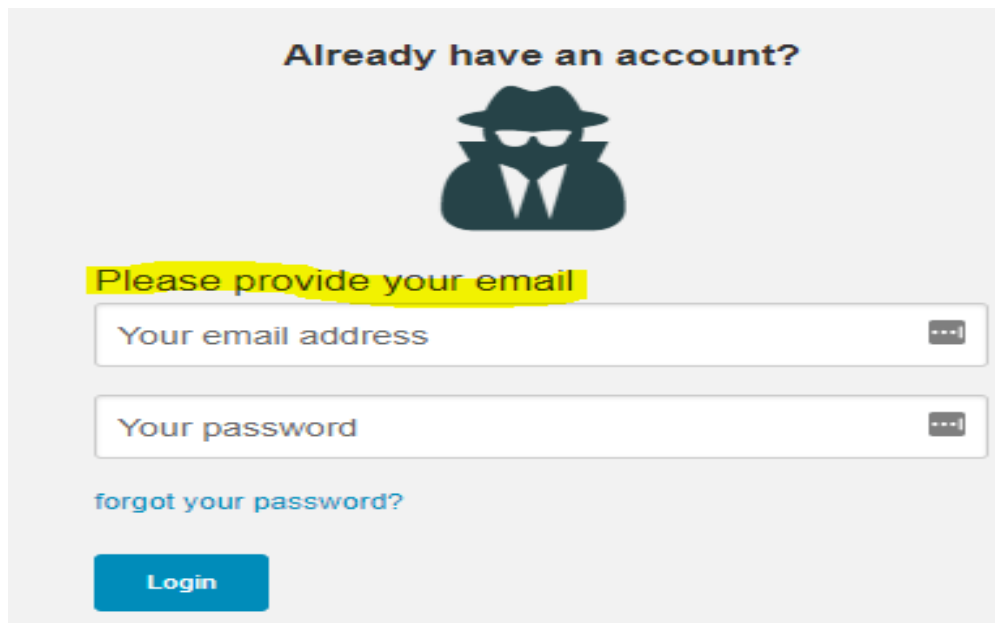


3.2.4.2 Login module testing

a) Validate form data

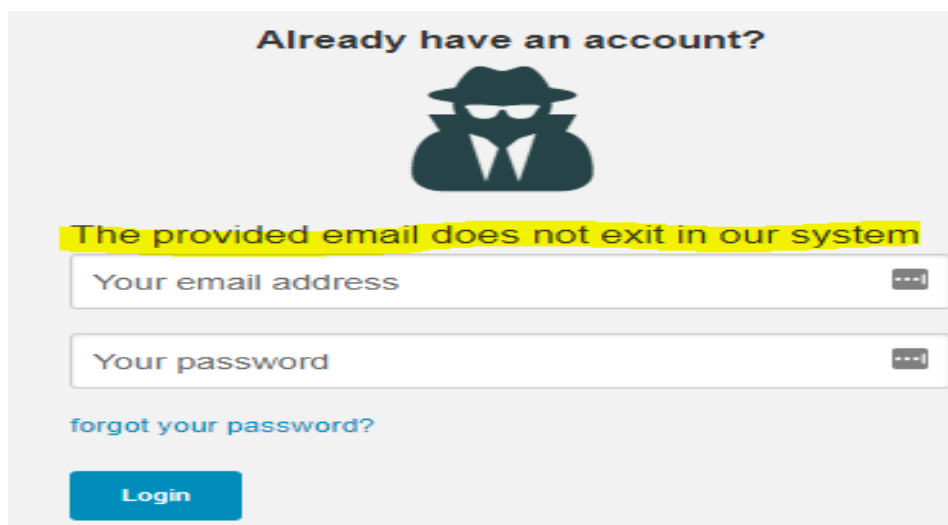
Figure 20 and Figure 21 are from a test case that ensured users provided all the details required for authentication. The test case also confirmed that the users use the credentials that are already registered on the system. Once the form passed the above tests, the user was redirected to the sharing center portal.

Figure 20 showing the error message for validating the login form



The screenshot shows a login form titled "Already have an account?" with a silhouette of a person in a hat and sunglasses. Below the title, the text "Please provide your email" is highlighted in yellow. The form contains two input fields: "Your email address" and "Your password", both with "..." icons on the right. Below the fields is a link "forgot your password?" and a blue "Login" button.

Figure 21 showing validation of user existence in the database



The screenshot shows a login form titled "Already have an account?" with a silhouette of a person in a hat and sunglasses. Below the title, the text "The provided email does not exist in our system" is highlighted in yellow. The form contains two input fields: "Your email address" and "Your password", both with "..." icons on the right. Below the fields is a link "forgot your password?" and a blue "Login" button.

- b) **Validate user signature, decrypt user data from the database, and redirect the user to the sharing center.**

The test case ensured that the user was validated against the database. The test result for the case was user access to the sharing center, which marked a pass to every test case specification.

3.2.4.3 Sharing center module testing

- a) **Generation of share request**

Figure 22 and Figure 23 show the outputs of the test case. The test case ensured that the user submits a request to another user registered in the portal. The case also required the user to complete the form. If all the details are provided, the test case submits the request to the central server. An error is displayed if a user submits an empty or a partially filled form.



Figure 22 showing unsuccessful request submission



Figure 23 showing successful request submission

- b) **Ability to decrypt received information using the local private key**

The test case ensured that the receiver of the shared password could decrypt the information sent to him or her within the portal using his/her respective private key. Figure 24 shows the decrypted view of the sent information within the portal.

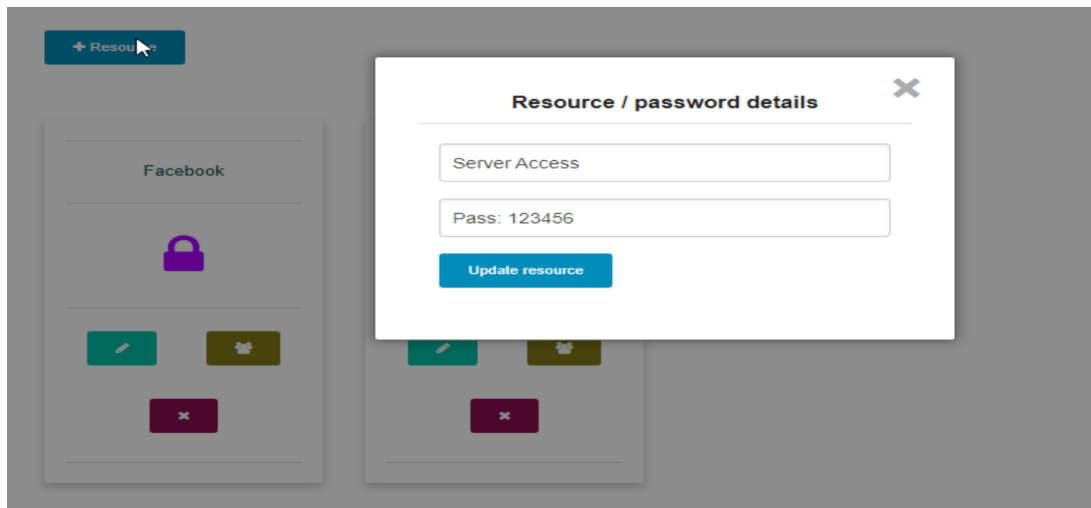


Figure 24 shows successful decryption of shared resources on the portal

3.2.5 Deployment and system maintenance

The phase involved doing a practical demonstration of the system with the selected individuals specified in the population and sampling section. In addition, the stage provided an opportunity for the targeted

population to work with the system and provide feedback for evaluation purposes. Finally, the phase involved supporting the system and applying recommendations as suggested by the targeted population.

3.3 Phase 2: Focus group discussion

The focus group discussion was conducted after the development phase of the prototype to gather inputs from participants who had the opportunity to use or view the demonstration of the prototype.

3.3.1 Sources of data, population, and sample size

The online focus group assisted in selecting or hand-picking a group of participants with a high probability of generating valuable data. The group consisted of individuals who make use of password management applications as part of their daily activity. The target sample size was set at ten respondents. A total of nine selected individuals were selected to experience and experiment with the system, then provide feedback using a questionnaire found at APPENDIX B.

3.3.2 Data collection

Primary and secondary sources of data were considered. For primary sources, questionnaires and interviews were administered during the focus group discussion. Literary materials from the previous studies on password security were considered for secondary sources when assessing techniques for protecting different states of data.

3.3.2.1 Questionnaires

Questionnaires were used as a tool for collecting primary data from the individuals who had the opportunity to interact with the prototype. An online form, Google form, was used to draft the questions and input fields through which respondents submitted their feedback. The form was distributed to the participants via email and instant messaging applications such as WhatsApp. The online form was convenient since data formatting and interpretation are automated.

3.3.2.2 Interviews

A structured and formal interview was used as the primary tool for data collection. The nature of the questions was predetermined, and it focused on the objectives of the study. No new questions were provided during the interview. The questions were sent in advance to the respondents to facilitate preparations. The data obtained from the interview was recorded on forms, and optionally, a digital form of it will be recorded on the respondent's approval.

3.3.2.3 Documentations

Literary information security materials from previous studies relating to password management applications and techniques for protecting different data states were considered the secondary sources for data collection.

3.4 Data analysis

The data acquired from the focus group participants were analyzed to provide insight into the defined research questions. The data from the questionnaires was summarized into figures or tables. The information from the secondary sources was used for literary analysis, thus providing comparison criteria for previous and current research in the research area.

4 Chapter Four: Results and discussions

4.1 Techniques for protecting data at rest, in motion, and use

The following are considered as the three states of information or data (SealPath, 2020).

Data at rest: This refers to data stored in a physical medium and is not in an active status of use. It ranges from documents of files stores in a file server to database file systems. **Table 12** shows the techniques for protecting data at rest.

Data in transit: This refers to data in motion or information being shared via channels such as email, web, instant messaging, and collaboration tools such as Skype. **Table 13** shows the techniques for protecting data in motion or transit.

Data in use: This refers to a state where the stored data is opened by one or more applications to facilitate access or consumption by the user. **Table 14** shows the techniques for protecting data in use.

Using desktop research, the report by (SealPath, 2020), (brightlineIT, 2021), and (Oracle, 2021) illustrated various ways of protecting data in different states. **Table 12**, **Table 13**, and **Table 14** below provide an assessment of different techniques used to protect different states of data with their application or relevance to the implemented system.

Table 12 shows the assessment of techniques for protecting data at rest

Protecting data at rest				
Technique	Description	Evaluation		
		Remark	Application	
			Yes	No
Disk encryption	Refers to the encryption of computers' disks.	The protection is limited once the system is accessed.		
File-level encryption	Symmetric or asymmetric encryption can be used to encrypt individual files.	Once the owner or receiver decrypts the document, it can be stored in unencrypted form.		
Database Encryption	Protects data stored in the database using symmetric and asymmetric encryption.	Using asymmetric encryption, keys secretly owned by the owner are used to encrypt & decrypt the data.		
Data level encryption	Using symmetric and asymmetric encryption, the data or information stored can be encrypted to ensure confidentiality & integrity in maintained.	The users' data stored in the local database will be encrypted using the public key using asymmetric encryption. A private key will be used to decrypt the data.		

Table 13 shows the assessment of techniques for protecting data in motion

Protecting data in motion				
Technique	Description	Evaluation		
		Remark	Application	
			Yes	No
Data encryption	Refers to end-to-end encryption of data before its transmission over the network. It involves secret shared keys or ownership of private keys for decryption once data reaches its destination.	The data can be stored in its encrypted form once the transmission is complete. The receiver shall use the secret keys to decrypt the data when access is required.		
Tunnel / Channel encryption	Refers to the technique of creating a secure encrypted channel over the network through which the data can be transmitted.	The technique provides robust protection during transmission only.		
SSL	SSL encryption is used to encrypt communications between web server & a client communication via browser	Provides robust session encryption between clients and browser communications. It is applicable in the developed system to protect API communications between clients & the distribution server		

Table 14 shows the assessment of protecting data in use

Protecting data in use				
Technique	Description	Evaluation		
		Remark	Application	
			Yes	No
Authentication – IAM (Identity Access Management)	It's a technique used to prove the identity of the principal accessing the data.	Digital certificates & signatures can be used to verify the identity of principals.		
Authorization - Conditional Access or Role-Based Access Control (RBAC) tools	Allow access to data based on user's privilege or role in the system	Access to private keys is allowed to owners only. Access to public keys is allowed to senders who have an agreement with the receiver.		

4.2 Redefined process model for storing and sharing passwords

One of the research outputs was a redefined process model for implementing password sharing over the internet using asymmetric cryptography, as shown in **Figure 25**. In addition, the process model mitigates the vulnerabilities in current password-sharing tools.

Figure 25 illustrates the detailed process model for password management applications

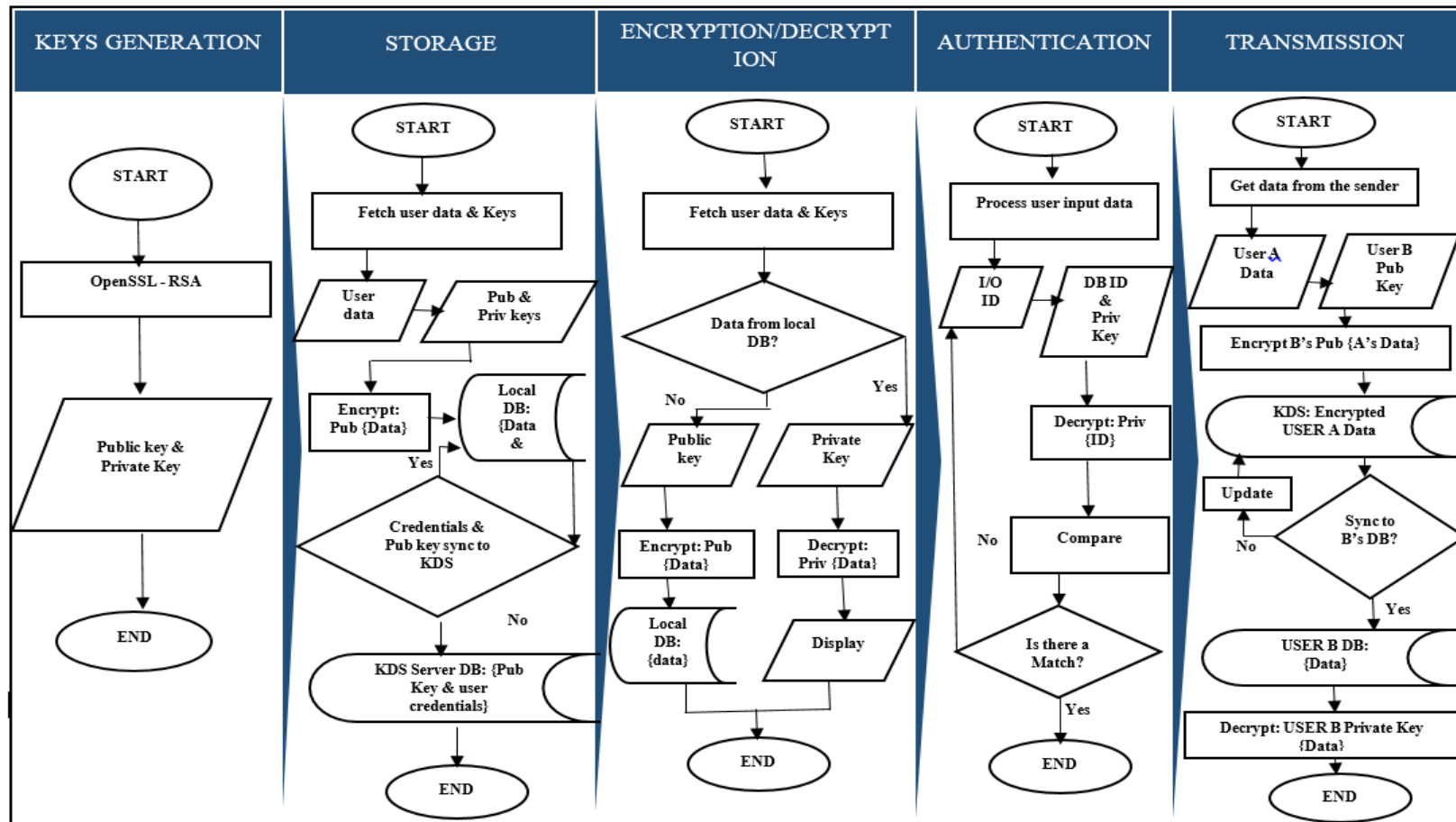
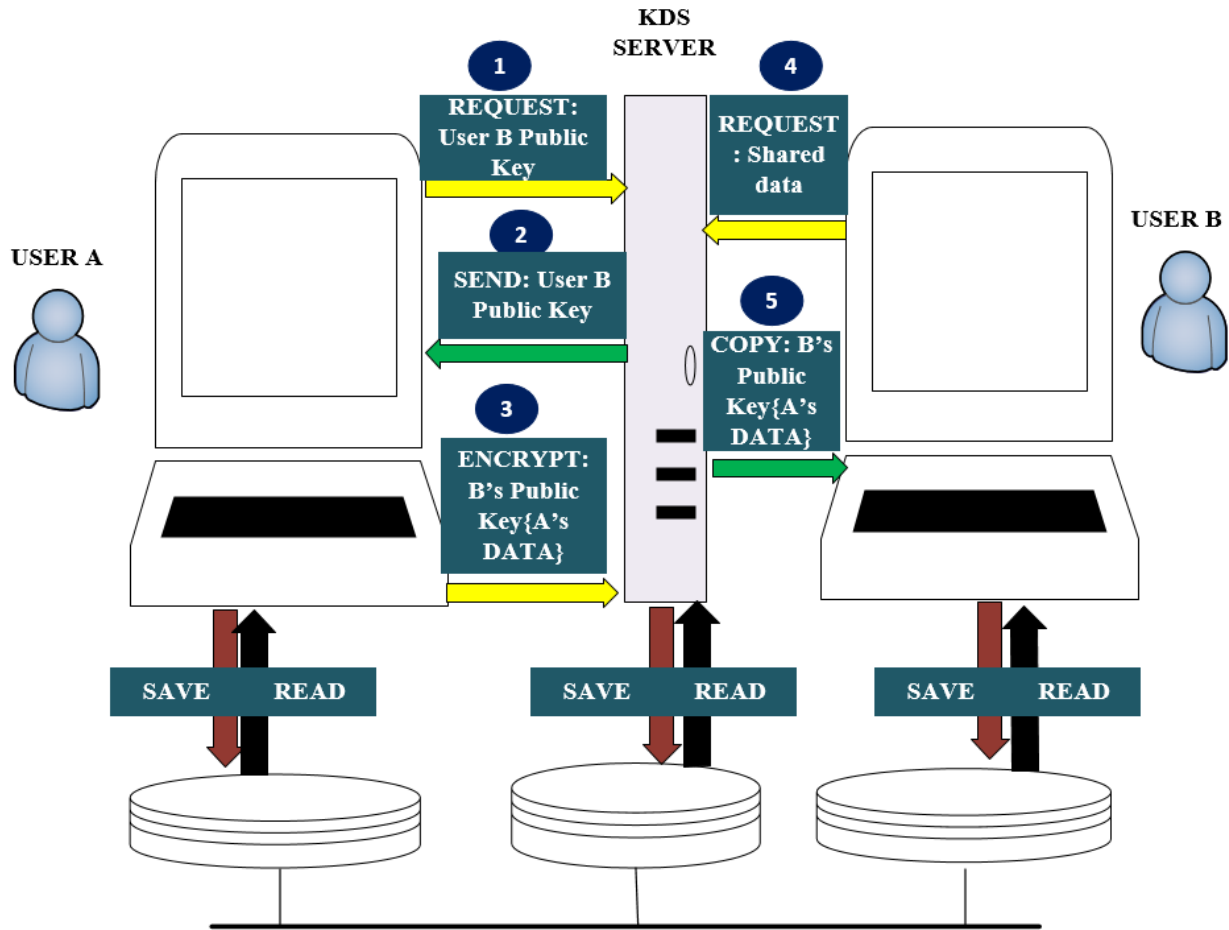


Figure 26 showing an abstract process model



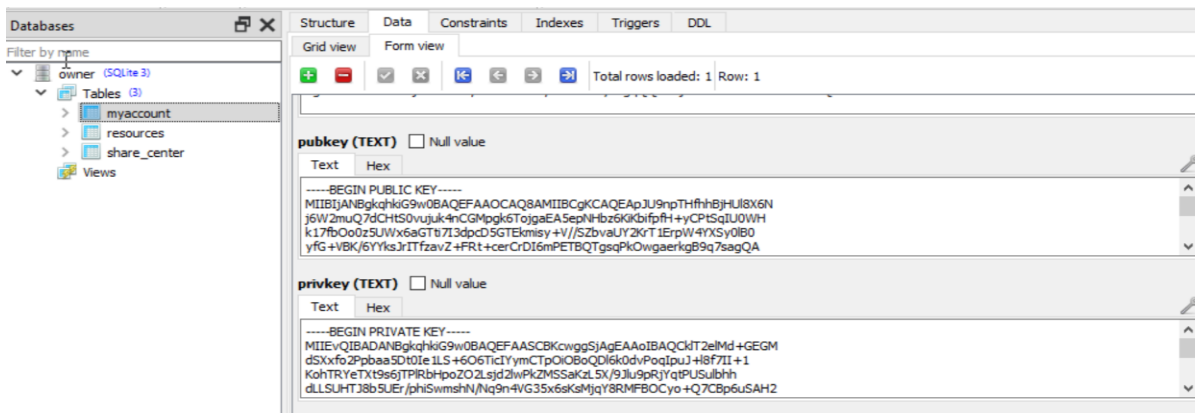
As illustrated in **Figure 25** and **Figure 26**, the sharing process begins when the sender's application initiates a request to the key distribution server for the receiver's public key. Once the server grants the application access to the receiver public key, the application would use the key to encrypt the password before dispatch. Then, the encrypted password is sent to the server for temporary storage. The receiver's application always prompts the server to confirm data shared to its identity (email address). If positive, the receiver's application pulls the transmitted data from the server and stores it in its local database. The cached data at the central server is deleted once the receiver has it in his or her local database, thus, marking the completion of the sharing process. All subsequent access to the shared data is facilitated locally at the receiver's end using his or her locally stored private key.

4.3 Developed system

4.3.1 Keys generation

During user registration, the public and the private keys were generated on the local computers *WIN-10-VM-01* and *WIN-10-VM-02*. Keys creation was achieved through the use of the OpenSSL library. RSA asymmetric encryption was used to generate the public-private pairs. The generated strings were based on the SHA-512 hashing algorithm, which provides a more robust encryption string than SHA-256 used in the LastPass password management tool. **Figure 27** shows the public-private key pairs generated by the system.

Figure 27 shows public and private keys stored in the local database



4.3.2 Keys encryption and storage

The public and the private keys are stored in the local SQLite3 database (owner.db) located at **WIN-10-VM-01 and WIN-10-VM-02**. To protect the integrity of the keys stored locally, an extra layer of encryption is provided to the keys to render them unusable if an intruder accesses the local database file. Each of the generated keys (public and private) is split into two segments. Then, an MD5 string is generated using a combination of the user email address, password, and registered device MAC address. The generated MD5 hash is then inserted between the two split segments of the keys, and then the keys are stored in the database. This simple encryption layer invalidates the use of the keys directly unless the intruder knows the identity of the embedded string.

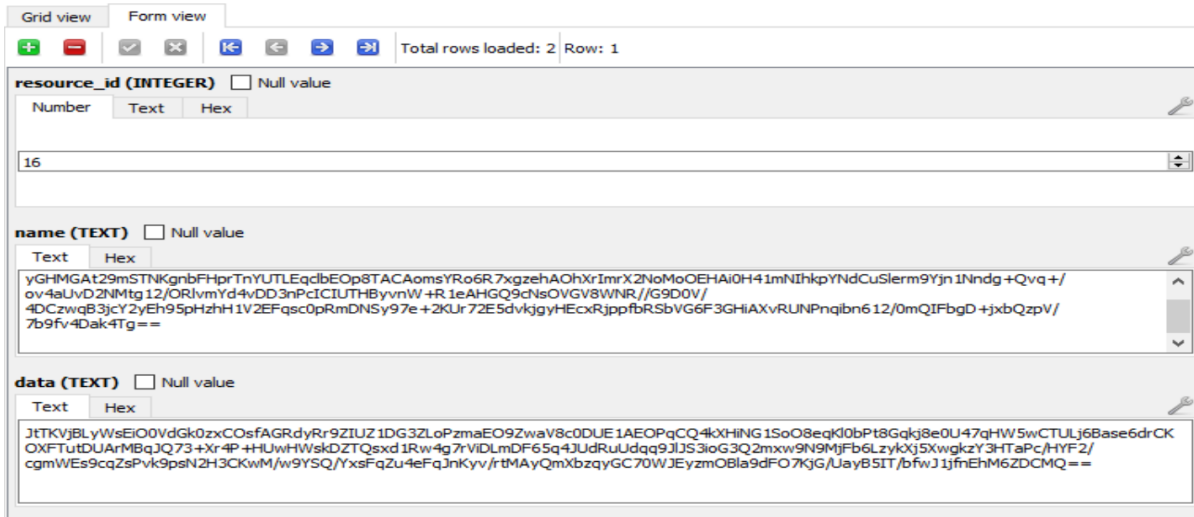
In addition to local storage, the public key is also pushed to the central database located at the central **SERVER (WIN-10-VM-03)**. Synchronization facilitates access to the receiver's key for data encryption to be sent during the share request. Therefore, there is no harm if the central server is attacked since the attacker will only get access to the public key used for encryption. To decrypt the shared resource, the attacker will have to attack the local database of users, which are highly distributed, without the direct identity of their storage address across the internet.

4.3.3 Data protection: In local storage

During user access to the vault, he or she is prompted for an email and password tokens. The MAC address of the device accessing the vault is extracted automatically. These tokens are used to generate the MD5 hash, which embeds an extra layer of encryption to the keys. In addition, the only pairs of keys stored in the local database are retrieved, and each is split into three segments. The central segment is compared to the principal authentication tokens. A match signals successful authentication. The first and the third segments of each key are merged again, and a session of each combination is created to facilitate future encryptions and decryptions in the vault.

The public key is used to encrypt all subsequent data stored in the local database, while the private key is used to carry out decryptions during access. **Figure 28** illustrates the data encrypted in the 'owner.db' located at **WIN-10-VM-01 and WIN-10-VM-02** virtual machines.

Figure 28 shows data encrypted using the public key, then stored in the local database



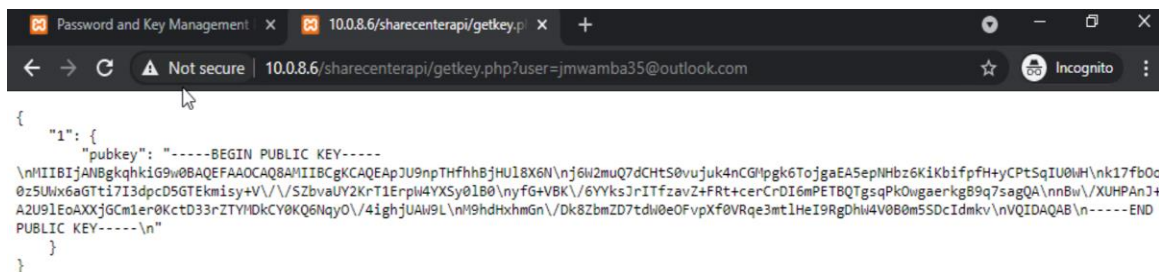
4.3.4 Data protection: During transmission

To initiate sharing of data, the sender and the receiver must first agree on the email addresses for communication. In addition, both users (*WIN-10-VM-01* and *WIN-10-VM-02*) must have the applications installed on their devices. Finally, the email address should be the active registered identity in the local system.

USER A will invoke an API request to the server to facilitate sharing, as shown in **Figure 29**. The API will grant **USER A** access to the **USER B** public key. The key will be used to encrypt the data before it leaves the local database of **USER A** to the central **SERVER** (*WIN-10-VM-03*) for caching until **USER B** synchronizes it to his or her local database. The URL below shows the format of the REST API request.

https://www.10.0.8.6/sharecenterapi/getkey.php?user=user_email_address

Figure 29 shows API request for the receiver public key

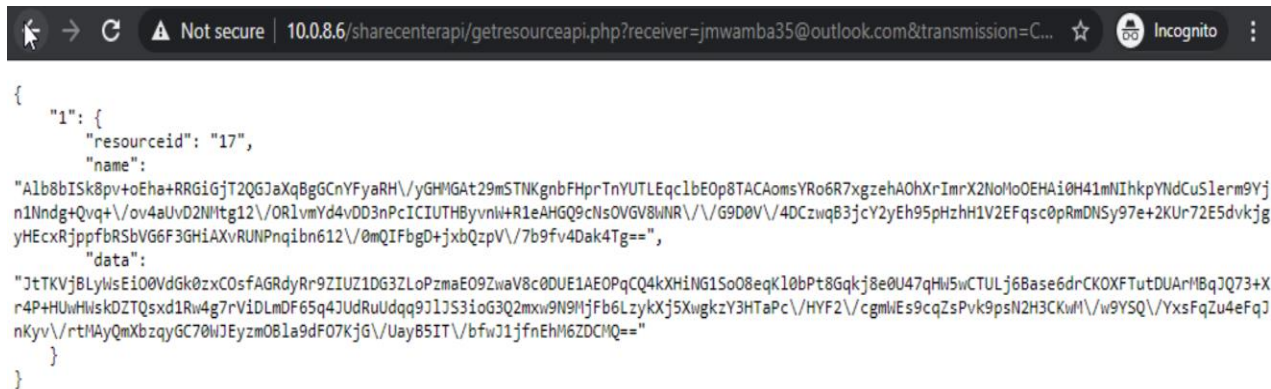


For **USER B** to receive the shared data, his or her portal will automatically invoke an API request to the server to determine if there is a pending resource transmission to his/her email address. If yes, the portal will invoke a second request to the central **SERVER** (*WIN-10-VM-03*) to synchronize the shared resource

to the local database. The response of the server is as shown in **Figure 30**. Below is the format of the synchronization request.

https://www.10.0.8.6/sharecenterapi/getresourceapi.php?receiver=user_email_address&transmission=Incomplete

Figure 30 Shows the API response to the receiver for the shared data



Once **USER B** Gets a response from the server, he/she stores the received data into the local databases without the need for additional encryption. This is because the payload was initially encrypted using his/her public key before dispatch. Future decryptions of the received data will be performed using his/her locally stored private key.

4.3.5 Email notification payload

An email notification is generated when the sender synchronizes the shared data to the central server. The receiver will get the alert using conventional email services such as GMAIL, YAHOO, OUTLOOK, etc. The only payload in the email is a URL, which redirects the receiver to his /her local vault, then synchronization of data will happen automatically.

4.3.6 Account recovery mechanism

The account recovery process is invoked when the user forgets his/her password of the local vault. When **'forgot password'** is clicked, an email dispatch with an OTP is invoked from the local vault to the registered email. The user will use the OTP in the next window to confirm his/her identity. Once there is a match with the temporal OTP stored locally, the vault will grant access to the private key, which is used to encrypt the new vault password.

4.4 Results from the focus group

After the demonstration of the prototype, users were presented with questions in the form of questionnaires and interviews to evaluate the developed system against the existing password management applications. Out of nine participants, eight respondents provided feedback to express their experience. The system evaluation criteria were categorized into decentralized system security, encryption, transmission technique, and availability.

In terms of decentralized security, 100% of the respondents expressed that the prototype enhances the confidentiality of their stored data, as illustrated in **Figure 31**. The confidence level for LastPass, KeePass, and Dashlane was 12.5%, 75%, and 25%, respectively. Low confidence levels in LastPass and Dashlane

were due to a centralized database system of user data, public and private keys. Exposure of the central database might grant intruders with the decryption keys and possibly the encryption functions, thus compromising the confidentiality and integrity of the stored data.

Figure 31 compares the confidence level of decentralized security of the prototype to that of other password management applications

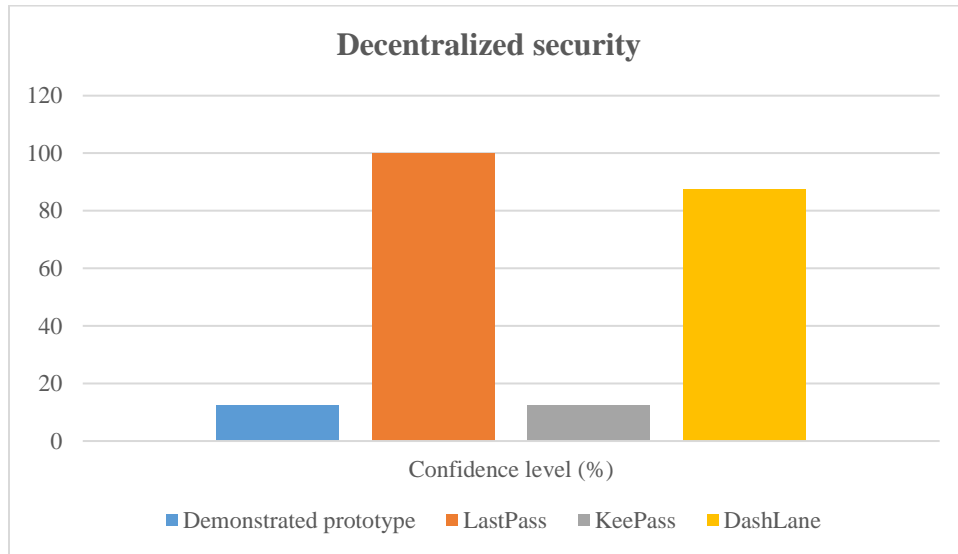
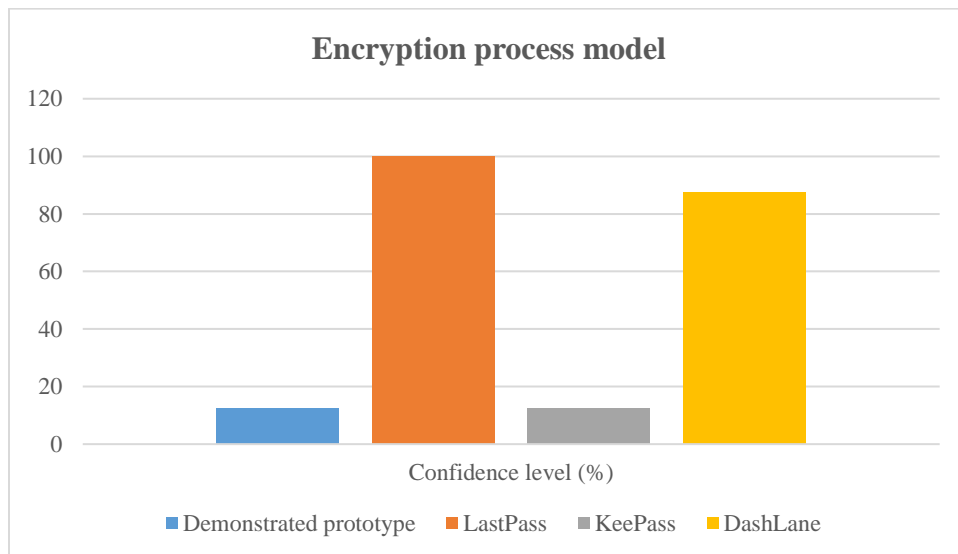


Figure 32 shows the confidence level concerning the encryption process model. Again, 100% of respondents expressed their confidence in the developed prototype. However, less than 38% expressed confidence in the other password management applications. The low confidence levels in other applications were due to ease of access to the decryption keys.

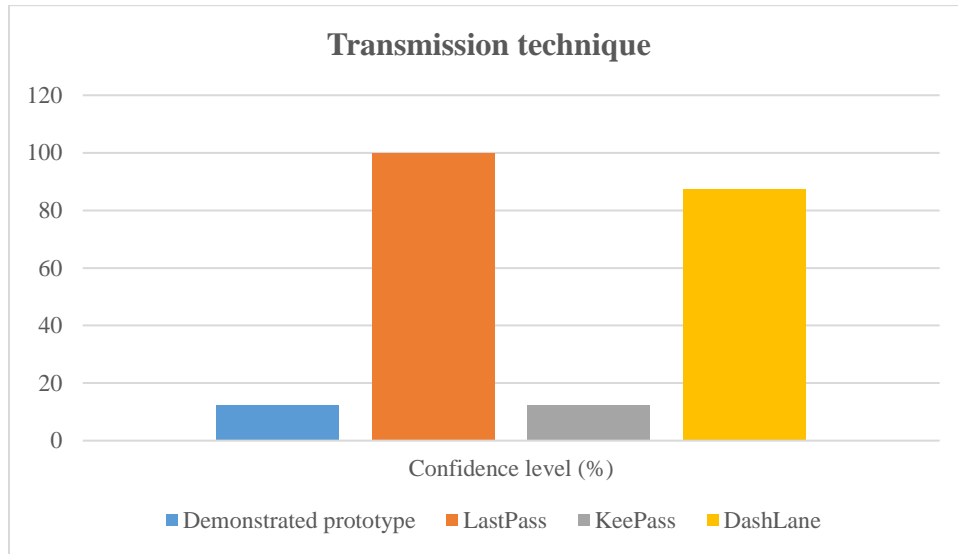
Figure 32 compares the confidence level of the encryption process model of the prototype to that of other password management applications



Regarding the process model of transmission, 100% of respondents expressed confidence in the developed prototype, as shown in **Figure 33**. LastPass and DashLane had a confidence level of 87%. On

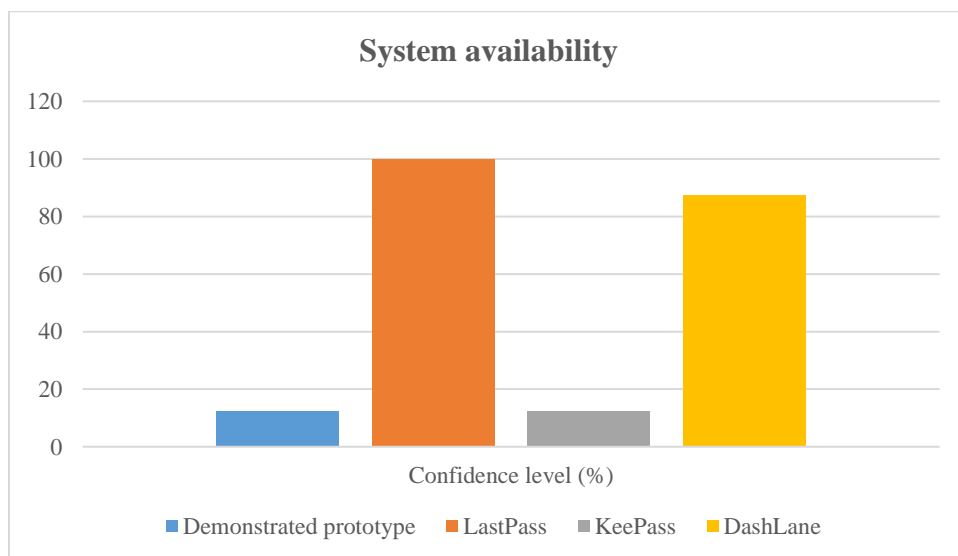
the other hand, KeePass had a low confidence level of 50%, which was attributed to the cumbersome technique of data sharing using thumb drives.

Figure 33 compares the confidence level of the transmission technique of the prototype to that of other password management applications



In terms of availability, respondents expressed their low (13%) confidence level in the developed prototype. The decentralized model does not provide means of recovery when the local database is lost or deleted. LastPass, KeePass, and DashLane had a confidence level of 100%, 12.5%, and 87.5%, respectively. The illustration of the outcome is as shown in **Figure 34**.

Figure 34 compares the confidence level of system availability of the prototype to that of other password management applications



4.5 Discussion from the results

The principles of cybersecurity are confidentiality, integrity, and availability. Together, they form the basis of any aspect of information security; hence considered as the main objectives for any developed

information security program or solution. Whenever there is a breach, it is always certain that one of the pillars (CIA) has been compromised. Confidentiality ensures the privacy of an asset or data is limited to the authorized principals only; integrity ensures the asset or information is not tampered with; hence, it can be trusted. Lastly, availability ensures the resources are continuously running and accessible (Walkowski, 2019).

Based on the results from the developed prototype, the integrity of the data is ensured via encryption using the public key. The encryption of the information is performed while data is at rest and when it is in transit. The keys for accessing data for reading or modification are protected; hence the data can be trusted. Reviewed traditional means of sharing and storing passwords such as notebooks, emails, and text messages have no encryption; thus, the data's integrity in these techniques is compromised since information is stored and transmitted in plain text. Password management applications such as LastPass have encryption applied to the data. Still, the master password, which is used to decrypt user's data, is stored loosely in the browser application, thus giving away user's data to an intruder who might compromise its integrity. In addition, the data in LastPass is susceptible to the service provider since the decryption keys, decryption function, and the data are centralized.

The confidentiality of passwords shared using emails, text messages, and written notes is vulnerable to unauthorized access since anyone with access to the medium of sharing can read and acquire the information. However, the identity of the owner and the receiver in the developed prototype is ensured by forcing users to provide their authentication tokens, which are matched against their digital signatures. On successful authentication, the prototype's users are authorized to access their respective private keys to access their data. In LastPass, centralization and easily accessible master password lowers the level of confidentiality of data.

In terms of availability, extreme decentralization of users' data and keys in the developed prototype limits users' access to their data in situations where their primary storage device is not present. On the other hand, in LastPass, the data is readily available from any of the registered locations and devices due to the centralization of data.

5 Chapter Five: Conclusion and recommendations

5.1 Conclusion

Data-centric techniques such as data-level encryption, SSL encryption, authentication, and authorization were assessed and used in the developed prototype to protect data at rest, in-use, and transit. Data level encryption using a public key ensured the integrity of data during transmission and storage. SSL encryption ensured the integrity and confidentiality of data transmission between the clients and the central servers. Lastly, authentication and authorization were used to secure users' identities and permission to access data in the prototype.

Vulnerabilities such as centralized user's data and keys, easily accessible master passwords, and easily exportable user's data were some of the vulnerabilities reported in current password sharing applications. In comparison to the concept of the master password employed in the LastPass, the prototype was able to store the vault decryption keys in an encrypted format. This helped to mitigate the vulnerability of the unencrypted master password stored in the web browser plugin. In addition, the prototype completely decentralized users' data and keys from the central server, thus eliminating the single point of attack as presented in LastPass. This is important, especially when the attacker has access to or knowledge of the decryption function. Decentralization limits the attack to the locally compromised device since the endpoints are evenly distributed over the internet, and their identity is opaque. In addition, if the central server is compromised, the data and the decryption keys are missing. KeePass had a reported vulnerability whereby the data stored in the USB could be exported into CSV in a plain-text format. The system has managed to overcome the vulnerability by ensuring the keys to manage decryptions have an extra layer of encryption, thus invalidating data decryptions unless the attacker has exclusive access to the keys decryption function.

Using asymmetric cryptography and the techniques for protecting different states of data, a new process model consisting of keys generation, storage, encryption, decryption, authentication, and transmission was created, thus enhancing the security of data shared by multiple users during storage, transmission, and usage.

A prototype was developed using the combination of the redefined process model for password sharing applications and the assessed techniques for protecting different states of data, thus helping in ensuring the security of passwords as they are transmitted over the internet and in mitigating the vulnerabilities presented by some of the current password management applications.

The prototype has demonstrated the ability to implement an asymmetric algorithm that facilitates the generation of keys for encryption and decryption of data. In addition, the prototype has also been able to facilitate identity validation using a digital signature, thus satisfying objective number 3 (To apply an asymmetric encryption algorithm for generating public and private keys for encrypting and decrypting data).

Based on the system test and evaluation, the shared resources between different users were successfully encrypted at the local storage and during transmission. This represents the achievement of the primary objective, which was to create a prototype of a system that facilitates secure password sharing over the internet.

5.2 Limitations

The resulting prototype from the study only covers how secure sharing of data can be facilitated online, using a web-based portal only. The email communication channel was the only medium of sensitive information sharing covered by the prototype.

5.3 Future work

Some of the main features of systems incorporated with keys management systems are the ability to Jointly manage keys with partners and recover keys in case the system users need to make a change or feel insecure with the old keys and signatures. I recommend future studies to improve the system with a similar goal to provide flexibility and enhance the security of the keys.

References

- B, F. (2020). *Exploratory Research: What are its Method & Examples?* Retrieved from Formplus: <https://www.formpl.us/blog/exploratory-research>
- Bellovin, S. M., & Merritt, M. (1992). *Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks*. Oakland: IEEE Symposium on Research in Security and Privacy.
- brightlineIT. (2021). *Data Encryption in Transit: What Your Business Needs to Know*. Retrieved from brightlineIT.com: <https://brightlineit.com/data-encryption-transit-business-needs-know/>
- Brown, A. S., Bracken, E., Zoccoli, S., & Douglas, K. (2004). Generating and remembering passwords. In A. S. Brown, E. Bracken, S. Zoccoli, & K. Douglas, *Applied Cognitive Psychology* (pp. 641–651). John Wiley & Sons, Ltd.
- Burnett, M. (2002). *Ten Windows Password Myths*. Retrieved from Symantec: <https://www.symantec.com/connect/articles/ten-windows-password-myths>
- Charoen, D. (2014). *Password Security*. Bangkok: National Institute of Development Administration.
- Grawemeyer, B., & Johnson, H. (2009). *Using and managing multiple passwords: A week to a view*. United Kingdom: Human Computer Interaction Group, Department of Computer Science, University of Bath BA2 7AY.
- Haber, M. J. (2016). *PASSWORD REUSE - OVERCOME THE VULNERABILITY*. Retrieved from Beyond Trust: <https://www.beyondtrust.com/blog/entry/password-reuse-overcome-vulnerability>
- Hassidim, A., Korach, T., Shreberk-Hassidim, R., Thomaidou, E., Uzefovsky, F., Ayal, S., & Ariely, D. (2017). *Prevalence of Sharing Access Credentials in Electronic Medical Records*. Health Informatics Research.
- Higgins. (2014). *Stolen Passwords Used In Most Data Breaches*. Retrieved from Dark Reading: <https://www.darkreading.com/stolen-passwords-used-inmost-data-breaches/d/d-id/1204615>
- Ion, I., Reeder, R., & Consolvo, S. (2015). *"no one can hack my mind": Comparing Expert and Non-Expert Security Practices*. USENIX Association.
- Kaspersky. (2020, May). *Cryptography Definition*. Retrieved from Kaspersk: <https://usa.kaspersky.com/resource-center/definitions/what-is-cryptography>

- Keith, M., Steinbart, & P.J. (2007). *The usability of passphrases for authentication: an empirical field study*. International Journal of Human–Computer Studies.
- Kessler, G. C. (2020). *An Overview of Cryptography*.
- Kumar, V. A., Kumar, A. A., Mounica, K., & Hitesh, S. (2019). *An Efficient and Secured Secret Password Sharing Technique using Block Chain*. International Journal of Recent Technology and Engineering (IJRTE).
- O’Toole, E., Feeney, L., Heard, K., & Naimpally, R. (2018). *DATA SECURITY PROCEDURES FOR RESEARCHERS*. J-PAL North America.
- Oates, B. J. (2006). *Researching Information Systems and Computing*. London: SAGE Publications Ltd.
- Oracle. (2021). *Protecting Data in a Network Environment*. Retrieved from Oracle.com: https://docs.oracle.com/cd/B12037_01/network.101/b10777/protnet.htm
- Pello, R. (2018). *Design science research — a short summary*. Retrieved from medium: <https://medium.com/@pello/design-science-research-a-summary-bb538a40f669>
- Rubenking. (2015). *Survey: Hardly Anybody Uses a Password Manager*. Retrieved from PC Magazine: <http://securitywatch.pcmag.com/securitysoftware/332517-survey-hardly-anybody-uses-a-password-manager>
- Schumacher, M. (2019). *Security Considerations for Team Based Password Managers*. SANS Institute Information security reading room.
- SealPath. (2020). *The three states of data gude: Description and how to secure them*. Retrieved from www.sealpath.com: <https://www.sealpath.com/protecting-the-three-states-of-data/>
- Standridge. (2019). *Password Management Applications and Practices*. SANS Institute Information Security Reading Room.
- Walkowski, D. (2019). *What is the CIA Triad*. Retrieved from f5.com: <https://www.f5.com/labs/articles/education/what-is-the-cia-triad>

Appendix

Appendix A: Project schedule

Activity	Duration (Week Numbers)															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	16	16
Planning & requirements analysis																
Defining requirements																
Designing product architecture																
Developing the product																
Testing the product																
Deployment																
Performing interviews																
Distributing questionnaires																
Reviewing documents																
Data cleaning																
Data analysis																

Week	Date Range
1	2 nd – 6 th February
2	9 th – 13 th February
3	16 th – 20 th February
4	23 rd – 27 th February
5	28 th March – 4 th April
6	7 th – 11 th April
7	14 th – 18 th April
8	21 st – 25 th April
9	28 th April – 1 st May
10	4 th – 8 th May
11	11 th – 15 th May
12	18 th – 22 nd May
13	25 th – 29 th May
14	2 nd – 6 th June
15	9 th – 13 th June
16	16 th – 20 th June

Table 15 showing dates for the project schedule

Appendix B: Survey questionnaire – System evaluation

The online form is accessible via a [Focus group discussion feedback questionnaire](#)

Appendix C: List of tables and figures

List of figures

FIGURE 1 SHOWING THE DAILY ACTIVITIES THAT REQUIRED A PASSWORD AUTHENTICATION MECHANISM	6
FIGURE 2 TYPES OF PASSWORDS GENERATED BY USERS	7
FIGURE 3 SHOWING PASSWORD REUSE PREVALENCE	8
FIGURE 4 SHOWS THE SECURITY ARCHITECTURE OF THE LASTPASS PASSWORD MANAGER.....	13
FIGURE 5 SHOWS THE SECURITY ARCHITECTURE OF THE DASHLANE PASSWORD MANAGER	14
FIGURE 6 REPRESENTING THE CONCEPTUAL MODEL OF THE PROPOSED SYSTEM.....	16
FIGURE 7 SHOWING THE ENCRYPTION AND DECRYPTION PROCESS USING RSA	17
FIGURE 8 REPRESENTING HOW DATA IS SHARED IN THE PROPOSED SYSTEM	18
FIGURE 9 SHOWS THE ARCHITECTURE OF THE SYSTEM	21
FIGURE 10 SHOWS THE LOGIN AND REGISTRATION SCREEN FOR THE LOCAL PORTAL	23
FIGURE 11 SHOWS THE DASHBOARD OF THE IMPLEMENTED PROTOTYPE	24
FIGURE 12 SHOWS THE SCREEN FOR CREATING OR SAVING A RESOURCE IN THE PORTAL	24
FIGURE 13 SHOWS THE SCREEN FOR SHARING RESOURCE OR PASSWORD	25
FIGURE 14 SHOWING A LIST OF REGISTERED USERS FROM THE SERVER ADMINISTRATION SCREEN.....	25
FIGURE 15 SHOWS A LIST OF RESOURCE SHARE REQUESTS OR LOGS FROM THE SERVER ADMINISTRATION SCREEN	26
FIGURE 16 SHOWING FORM VALIDATION TEST ERROR MESSAGE	26
FIGURE 17 SHOWING AN ALERT MESSAGE ON SUCCESSFUL REGISTRATION.....	27
FIGURE 18 SHOWS THE KEYS AND THE SIGNATURE ASSIGNED AFTER USER REGISTRATION.....	27
FIGURE 19 SHOWING USER DATA ENCRYPTED IN THE DATABASE	27
FIGURE 20 SHOWING THE ERROR MESSAGE FOR VALIDATING THE LOGIN FORM	28
FIGURE 21 SHOWING VALIDATION OF USER EXISTENCE IN THE DATABASE.....	28
FIGURE 22 SHOWING UNSUCCESSFUL REQUEST SUBMISSION	29
FIGURE 23 SHOWING SUCCESSFUL REQUEST SUBMISSION.....	29
FIGURE 24 SHOWS SUCCESSFUL DECRYPTION OF SHARED RESOURCES ON THE PORTAL	29
FIGURE 25 ILLUSTRATES THE DETAILED PROCESS MODEL FOR PASSWORD MANAGEMENT APPLICATIONS	33
FIGURE 26 SHOWING AN ABSTRACT PROCESS MODEL	34
FIGURE 35 SHOWS PUBLIC AND PRIVATE KEYS STORED IN THE LOCAL DATABASE	35
FIGURE 28 SHOWS DATA ENCRYPTED USING THE PUBLIC KEY, THEN STORED IN THE LOCAL DATABASE	36
FIGURE 29 SHOWS API REQUEST FOR THE RECEIVER PUBLIC KEY	36
FIGURE 30 SHOWS THE API RESPONSE TO THE RECEIVER FOR THE SHARED DATA.....	37
FIGURE 31 COMPARES THE CONFIDENCE LEVEL OF DECENTRALIZED SECURITY OF THE PROTOTYPE TO THAT OF OTHER PASSWORD MANAGEMENT APPLICATIONS	38
FIGURE 32 COMPARES THE CONFIDENCE LEVEL OF THE ENCRYPTION PROCESS MODEL OF THE PROTOTYPE TO THAT OF OTHER PASSWORD MANAGEMENT APPLICATIONS	38
FIGURE 33 COMPARES THE CONFIDENCE LEVEL OF THE TRANSMISSION TECHNIQUE OF THE PROTOTYPE TO THAT OF OTHER PASSWORD MANAGEMENT APPLICATIONS	39
FIGURE 34 COMPARES THE CONFIDENCE LEVEL OF SYSTEM AVAILABILITY OF THE PROTOTYPE TO THAT OF OTHER PASSWORD MANAGEMENT APPLICATIONS	39

List of tables

TABLE 1 SHOWING THE ACTIVITIES THAT REQUIRE PASSWORD AUTHENTICATION.....	5
TABLE 2 SHOWING THE FREQUENCY OF PASSWORD TYPES CREATED BY USERS	6
TABLE 3 SHOWING THE PREVALENCE OF PASSWORD REUSE.....	7
TABLE 4 SHOWING VARIOUS VERSIONS OF THE SHA ALGORITHM.....	10

TABLE 5 REPRESENTING THE STRUCTURE OF THE 'DATA SHARE' REQUEST FORMAT	18
TABLE 6 SHOWS THE SOFTWARE AND HARDWARE REQUIREMENTS FOR THE SYSTEM PROTOTYPE	19
TABLE 7 SHOWING THE USERS' ACCOUNT TABLE AT THE LOCAL DATABASE	21
TABLE 8 SHOWING THE RESOURCES' TABLE AT THE USERS' LOCAL DATABASE.....	22
TABLE 9 SHOWING THE SHARED RESOURCES TABLE AT THE USERS' LOCAL DATABASE.....	22
TABLE 10 SHOWS THE REGISTERED USERS TABLE AT THE CENTRAL DISTRIBUTION SERVER.....	22
TABLE 11 SHOWS THE SHARED RESOURCES CACHE OR SYNCHRONIZATION TABLE.....	22
TABLE 12 SHOWS THE ASSESSMENT OF TECHNIQUES FOR PROTECTING DATA AT REST	31
TABLE 13 SHOWS THE ASSESSMENT OF TECHNIQUES FOR PROTECTING DATA IN MOTION	32
TABLE 14 SHOWS THE ASSESSMENT OF PROTECTING DATA IN USE.....	32
TABLE 15 SHOWING DATES FOR THE PROJECT SCHEDULE	44

Appendix D: System installation instructions

Virtual machine exports have been provided to facilitate the implementation of the proof of concept of the system; WIN-10-VM-01, WIN-10-VM-02, and WIN-10-VM-03. The virtual machines instances consist of the dependencies required to run the systems and the system itself. Below are the steps to set up and run the prototype:

1. Import the provided virtual machine exports into the virtual box hypervisor.
2. Once the import is complete, run or start the virtual machines.
3. Once logged in to the virtual machines, ensure the MySQL and the Apache webserver are running using the Xampp software.
4. WIN-10-VM-01 and WIN-10-VM-02 will serve as the client endpoints while WIN-10-VM-03 will serve as the central server endpoint.
5. On clients, access the application using the URL below on the browser.
<http://localhost/vault>
6. On the server, you can access the admin view using the URL below on the browser.
<http://localhost/admin>