Secure Password Sharing and Storage using Encryption and Key-Exchange

Joseph Okwedo Mwamba
School of Computing and Informatics
University of Nairobi, Kenya

Andrew Mwaura Kahonge
School of Computing and Informatics
University of Nairobi, Kenya

ABSTRACT
Based on security best practices for passwords, the credential is a confidential pin for authenticating system users. Still, 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.Information security experts recommend using password management applications to mitigate security vulnerabilities resulting from the transmission of passwords from one person to another. 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. Also, 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 system development and focus group discussion. The result was a prototype that facilitated the secure sharing of passwords over the internet using asymmetric cryptography.

General Terms
Cryptography, Asymmetric encryption, password management, RSA, Key exchange.

Keywords
RSA, key exchange, encryption

1. INTRODUCTION
The current popular modes of password sharing, such as emails, SMS, and instant messaging, are not secure. In addition, the report by [1] 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.

The fundamental principle in password utilization is that users in any particular system should own their credentials, which should be kept private [2]. In a study to investigate security considerations in scenarios where credentials are team-based, [2] discovered that the principle of non-sharing might be rendered impractical. The policy is unusable due toaccounts for simple systems, which do not allow multiple credentials. In addition, some systems have local administrative accounts used to bypass central authentication servers—lastly, limited authorization on users' accounts necessitating borrowing credentials from colleagues [3].

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 compromisemore sensitive systems.

In this light, a research study was proposed to implement a system specifically meant for enhancing the security of passwords shared between individuals over the internet or via any other forms of communication. Furthermore, the system ensures the confidentiality and integrity of the stored and transmitted data over a communication network using asymmetric cryptography.

The specific objectives of the research were:To identify and study the vulnerabilities presented by current password management applications such as LastPass, Dashlane & Keepass; to assess the various data-centric approaches for protecting data at rest, in-use, and in transit using public-private-key encryption; to research and come up with a process model that mitigates the vulnerabilities presented by the current password management applications; and lastly, to design, develop, and test a prototype that validates the proposed process model.

2. RELATED WORK

2.1 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 and then eventually decrypted again into a readable format[4].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 generate keys, certificates, encryption, and data decryption

2.1.1 Public Key Cryptography Using RSA

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 be used to validate who sent the message. Validation is achieved when the sender encrypts the message using their private key, and then the receiver decrypts it using the sender's public key[4].

One of the most popular implementations of public-key cryptography is the 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)[4].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. Then, 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:

$$C = M^w \bmod z$$

The decryption function M is as below:

$$M = C^d \bmod z$$

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 $M = 13^3 \bmod 15 = 2197 \bmod 15 = 7$.

2.2 Current Password Sharing and Storage Tools

2.2.1 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[1].The security premise of the password manager revolves around a master password, which is created during the registration process. Then, the credential is used to authenticate 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 [1] indicated that an option to remember the MasterPassword isoffered on web browser extension and the mobile application versions of LastPass. This 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 feature eliminates the need to hold 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. [1]Also reported, LastPass cannot decrypt users' data due to a lack of access to the master password.

LastPass uses the master password and the username as the salt during account creation to generate the encryption. One

hundred thousand one hundred rounds of PBKDF2-SHA256 are used to create the encryption keys on the clientside. 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 login 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 [1] 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. Furthermore, to aggravate the situation, the retrieved credentials gained access from anywhere since it bypassed counter-measures such as multi-factor authentication.Another vulnerability reported by [1] is the acquisition of the user master password during an active login session on the local computer. The vulnerability 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-site scripting (CSS) and cross-site 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.

2.2.2 Dashlane password manager & security architecture

The password management application has a security premise similar to that of LastPass. 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. Therefore, the transmitted passwords are vulnerable if intercepted by an intruder. Furthermore, according to Dashlane, the passwords are deleted on the server-side once the process is completed [1].

2.2.3 KeePass password manager & security architecture

Keepass is a password management application that offers no server or cloud component for managing passwords[2]. 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[2]. 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[1].

2.3 Other Password Sharing Techniques

Besides the password management tools such as LastPass, a study by [1].has identified the following as the standard modes of password sharing**.**

2.3.1 Use of a 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. However, on the downside, the process of keeping, managing, and tracking records is very tedious[1].

2.3.2 Storing passwords on unencrypted files in 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.

2.3.3 Storing passwords Using Browser Software

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.

3. CONCEPTUAL MODEL

Figure 1 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 key 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 stored in the database, while the private key is used for decryption during data access.
ii. ***Central requests logging server*** - The server consists of a MySQL database. The server is used to cache requests for shared resources temporarily. The server also holds users' public keys to facilitate sharing during the transmission process. Sensitive

user data such as credentials and passwords are only stored locally at the endpoints. The server holds the shared resource's sender and the receiver's identity during the data share process.

iii. ***OpenSSL RSA*** encryption is used to generate keys & signatures.

iv. ***A secure socket layer (SSL)*** will secure data in transit between the sender and the receiver web service API.

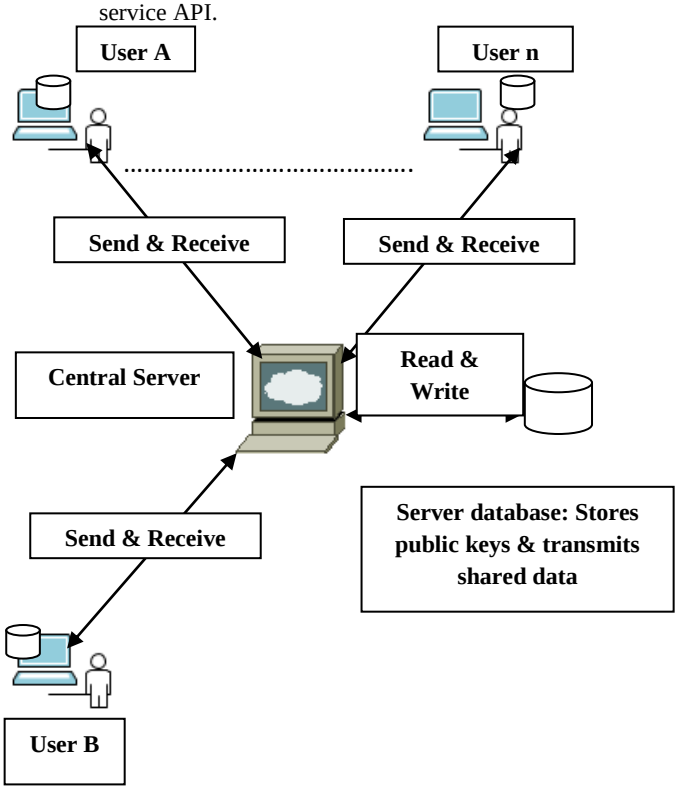

Figure 1 Shows the proposed conceptual model

4. METHODOLOGY

4.1 Research Design

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 the use of password management applications, thus acquiring insight into it to formulate a more specific research problem [5].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 artefact or a product [6].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. In addition, it was used to represent the outcome of the research study in the non-numerical form. In contrast, the quantitative approach was used in the numerical analysis of the data.

The study's methodology was guided by specific research objectives, such as designing and implementing a system that uses asymmetric encryption algorithms to validate access to shared sensitive data (password).To cater to 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.

4.2 Phase 1: System Design and Development

The prototype development used the waterfallmodel 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. The steps involved are: Requirement gathering and analysis, system design, implementation, integration and testing, deployment of the system, and maintenance

4.2.1 System requirements definition and analysis

The following requirements were established for the system prototype.

i. **Oracle Virtual Box**. It was used as the hypervisor for virtualizing and simulating the communicating clients, servers, and communication network.

ii. **WIN-10-VM-01** and **WIN-10-VM-02**. The virtual machines were running Microsoft Windows 10 Home Edition to simulate client computers, in this case, is called USER A and USER B**.**

iii. **WIN-10-VM-03**. The virtual machine was running Microsoft Windows 10 Home Edition to simulate the central **SERVER** that facilitates sharing of data between **USER A** and **USER B**.

iv. **Xampp (Apache, MySQL & Mercury) application**. It provided web server functionality for the PHP programing language, which the prototype was built on. It also offered the MySQL database client at the server end. In addition, the Mercury application was used to provide mail service to the users during sharing.

v. **SQlite3 database software**. It provided local database storage at the client end (WIN-10-VM-01 and WIN-10-VM-02).

vi. **OpenSSL**. It was used as the encryption libraryto generate the keys at the clients' end and conduct encryptions.

4.2.2 System Design or architecture

The system components, that is, the ***SERVER (WIN-10-VM-03),USER A (WIN-10-VM-01)***, ***USER B(WIN-10-VM-02)***, and the communication network,were implemented on a virtual environment. First, the Virtual Box hypervisor was installed on the host computer, and then the guest machines mentioned above were installed. Finally, 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***). In addition, 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***).

4.3 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.

4.3.1 Sources of data, population, and sample size

Online focus groups assisted in selecting or handpicking 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. Therefore, the target sample size was set at ten respondents. A total of 9 individuals participated in the discussion and provided feedback via an interview and an online questionnaire.

4.3.2 Data collection

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 incorporated a link to the system prototype through which the focus group accessed the system for demonstration. 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.

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. Instead, 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.

Literary information security materials from previous studies were used to collect information on various data protection techniques for protecting different states of data.

5. RESULTS AND DISCUSSIONS

5.1 Techniques for Protecting Data at Rest, in Motion, and Use

5.1.1 Techniques for protecting data at rest

Disk encryption. Refers to the encryption of the disk inside of the computer[7]. The protection is limited once the system is accessed, thus not applicable in the developed prototype.

File-level encryption. Symmetric or asymmetric encryption can be used to encrypt individual files. However, once the owner or receiver decrypts the document, it can be stored in unencrypted form, thus not applicable in the developed prototype[7].

Database Encryption. Protects data stored in the database using symmetric and asymmetric encryption[7]. Using asymmetric encryption, keys secretly owned by the owner are used to encrypt & decrypt the data. The technique was not applied in the developed prototype.

Data level encryption. The data or information stored can be encrypted using symmetric and asymmetric encryption to ensure confidentiality & integrity are maintained[7]. 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. The technique was applied in the developed prototype

5.1.2 Techniques for protecting data in motion

Data encryption. It refers to end-to-end encryption of data before its transmission over the network.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. The technique was applied in the developed prototype.

Tunnel / Channel encryption. Refers to the technique of creating a secure encrypted channel over the network through which the data can be transmitted[8]. It provides robust protection during transmission only. The technique was not implemented in the proposed prototype

SSL. SSL encryption is used to encrypt communications between a web server & a client communication via browser[8]. It Provides robust session encryption between clients and browser communications. It was applied in the developed system to protect API communications between clients & the distribution server

5.1.3 Techniques for protecting data in use

Authentication – IAM. It's a technique used to prove the identity of the principal accessing the data[9]. For example, digital certificates & signatures can be used to verify the identity of principals. The technique was applied in the prototype.

Authorization – Conditional / (RBAC) tools. Allow access to data based on the user's privilege or role in the system[9]. Access to private keys is allowed to owners only. Access to public keys is allowed to senders who have an agreement with the receiver. The technique was applied in the prototype.

5.2 Redefined Process Model for Storing and Sharing Passwords

The research resulted in the process model for implementing password management applications, as shown in **Figure 2**. The new process model is aimed at facilitating secure password sharing over the internet using symmetric cryptography.

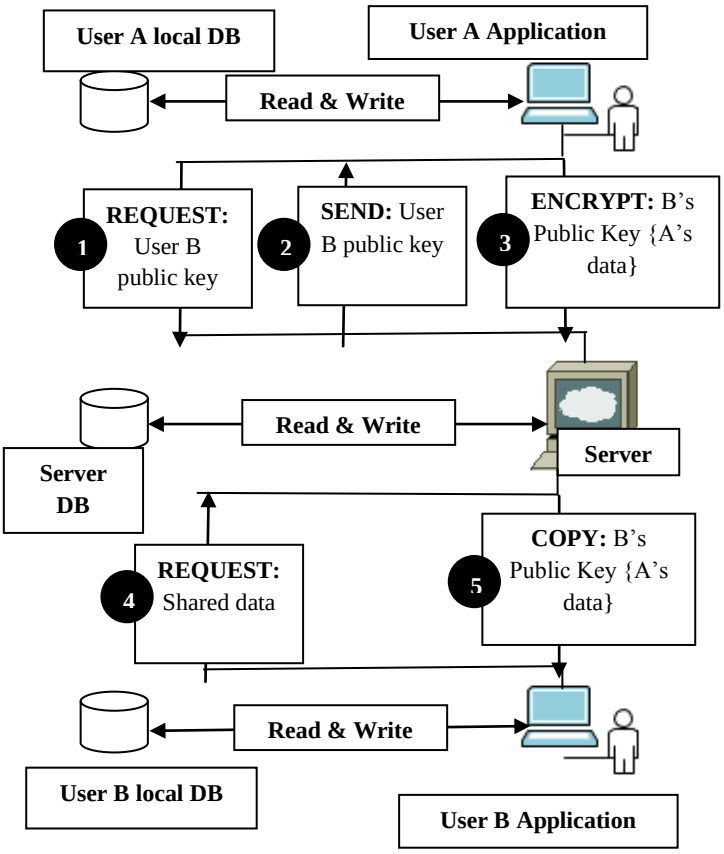

Figure 2 shows the redefined process model for password management applications

5.3 Developed System

5.3.1 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 generation was achieved using the OpenSSL library. RSA asymmetric encryption was used to create the public-private pairs. The generated string was based on the SHA-512 hashing algorithm, which provides a more robust encryption string than SHA-256 used in the LastPass password management tool.

5.3.2 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. First, 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).*** This helps in accessing the key of the receiver for encryption of the data to be sent during the share request. There is no harm if the central server is attacked since the attacker will only get access to the public key which is used for encryption. To decrypt the shared resource, the attacker will have to attack the local database of users, which are highly distributed, and without the direct identity of their storage address across the internet.

5.3.3 Data protection: In local storage

During user access to the vault (located at ***WIN-10-VM-01 and WIN-10-VM-02*** virtual machines), the user is prompted to 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 is used to embed 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 will signal 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.

5.3.4 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.To facilitate sharing,**USER A** will invoke an API request to the server. 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

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. Below is the format of the synchronization request.

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

The response to the REST API response is as shown below.

{

"1":{

"resourceid":"17",

"name":"AKmaoeldjufehnjdmdd,,d,e,dpdep,kfjdjsksllskskksks/AAmdkdmdkdd22msm3oeo93u22222-umdd,d,d",

"data":"Kddheyeeuemm,dldl,dmdndk9383ndmdy3u3i2922o2oi3mmfuu43i3oo3o3oo3i3u333y3893933003383883838"

}

}

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.

5.4 Results from Focus Group Discussion

After the prototype demonstration, users were presented with 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. However, 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.

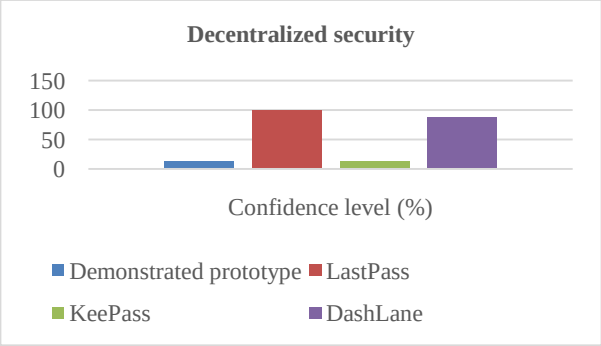

Figure 3 compares the decentralized security of the developed prototype against other password management applications

Regarding the encryption process model, 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 level in other applications was due to ease of access to the decryption keys.

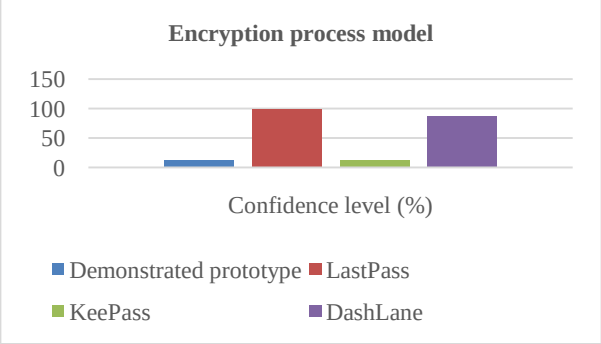

Figure 4 compares the encryption process model of the developed prototype against other password management applications

Regarding the process model of transmission, 100% of respondents expressed confidence in the developed prototype. 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.

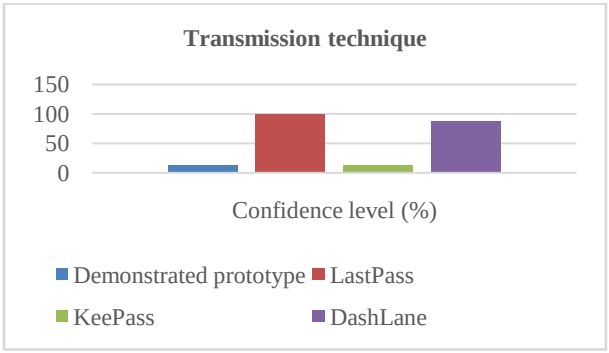

Figure 5 compares the data transmission technique of the developed prototype against other password management applications

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.

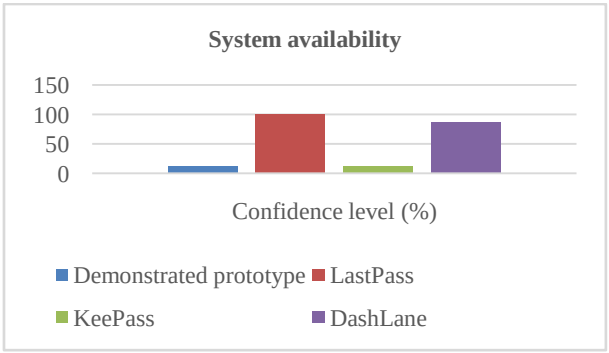

Figure 6 compares the availability of the developed prototype against other password management applications

5.5 Discussions

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[7].

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 marched against their digital signatures. On successful authentication, the prototype's user is 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 user data and keys in the developed prototype limits user's 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.

6. 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.

6.1 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.

6.2 Recommendation for Further 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.

7. REFERENCES

[1] S. Standridge, "Password Management Applications and Practices," SANS Institute Information Security Reading Room, 2016.

[2] M. Schumacher, "Security Considerations for Team Based Password Managers," SANS Institute Information security reading room, 2018.

[3] H. Ayal, K. Tzfania, S.-H. Rony, T. Elena, U. Florina, A. Shahar and A. Dan, "Prevalence of Sharing Access Credentials in Electronic Medical Records," Healthcare Informatics Research, 2017.

[4] Kessler, "An Overview of Cryptography," 2021. [Online]. Available: https://www.garykessler.net/library/crypto.html. [Accessed 4 June 2020].

[5] J. O. B, Researching Information Systems and Computing, London: SAGE Publications Ltd, 2006.

[6] R. Pello, "Design science research — a short summary," Medium.com, 2018. [Online]. Available: https://medium.com/@pello/design-science-research-a-summary-bb538a40f669. [Accessed 14 January 2021].

[7] "Data Encryption in Transit: What Your Business Needs to Know," BrightlineIT, 2021. [Online]. Available: https://brightlineit.com/data-encryption-transit-business-needs-know/. [Accessed 24 June 2021].

[8] Oracle, "Protecting Data in a Network Environment," Oracle, 2021. [Online]. Available: https://docs.oracle.com/cd/B12037_01/network.101/b10777/protnet.htm. [Accessed 13 June 2021].

[9] SealPath, "The three states of data guide: Description and how to secure them," SealPath, 23 June 2020. [Online]. Available: https://www.sealpath.com/blog/protecting-the-three-states-of-data/. [Accessed 22 May 2021].

[10] I. Iuli, R. Rob and C. Sunny, "“...no one can hack my mind”: Comparing Expert and Non-Expert Security Practices," Symposium on Usable Privacy and Security, 2015.

[11] Z. Zhu, D. Zhongqi and W. Yongge, "Security analysis of a password-based authentication protocol proposed to IEEE 1363," Theoretical Computer Science, 2005.

[12] A. Michel and . P. David, "Simple Password-Based Encrypted Key Exchange Protocols," 2005.

[13] R. W. F. Lai, C. Egger, M. Reinert, S. S. M. Chow, M. Maffei and D. Schr¨oder, "Simple Password-Hardened Encryption Services," USENIX Association, 2018.