A Cloud-Based Password Manager for Multiple Transaction Accounts

Aborisade, D.O.
Department of Computer Science
Federal University of Agriculture Abeokuta (FUNAAB)
Abeokuta, Ogun State, Nigeria.
aborisadeda@funaab.edu.ng

Alowosile, O.Y., Odumosu, A.A. & Adedeji, A.A.
Department of Computer Science
Abraham Adesanya Polytechnic
Ijebu-Igbo, Ogun State, Nigeria.
alowosile78@gmail.com, adesoladapo2000@gmail.com, adedejibdm@gmail.com
+234-805-438-8100

Banjo, O.A.
Department of Computer Engineering
Abraham Adesanya Polytechnic
Ijebu-Igbo, Ogun State, Nigeria.
steadydoo70@gmail.com

*Corresponding Author: alowosile78@gmail.com

ABSTRACT

The emergence of Internet Technologies obviously has brought about several online transaction systems requiring authentication. These transaction systems require their users to supply access code (password) to authenticate them for any transaction to take place. Consequently, a number of password managers have been proposed and designed to assist users recollect their several passwords for transactions. Most users of online transaction systems keep many account passwords that must not be recorded and forgotten, but which they often find difficult. Hence, they are faced with the challenges of effectively managing their ever growing passwords for transacting on their systems. The existing password managers have been helpful mostly in the area of helping users recollect forgotten password on a particular account (platform) at a time, but not on multiple accounts (platforms). In this paper, an architectural framework for password manager on multiple accounts (platforms) was proposed and implemented. The architecture is composed of three modules namely login and account module, security module and the authentication module. The architectural design was implemented and tested with hundred networked systems. The results of implementation show that the proposed password system is effective and capable of assisting users easily recollect any of their multiple passwords for a particular transaction.

Keywords: Passcode management, Passcode security, Secured multiple accounts, Internet Technologies

Aims Research Journal Reference Format:

1. INTRODUCTION

A password is a string of characters used to login to a computer and other systems for files access, program access, and other resources (Melanie, 2012). They are used to ensure that people do not access any system unless they are authorized to do so (Melanie, 2012). Despite the argument by notable security researchers that the use of passwords as authentication method is vulnerable and characterized by security and usability draw back, it has continued to maintain its position as first line of system defense in every facet of our lives such as in home systems, office systems, to financial transactions systems. The reason for this is because of its incumbency, familiarity, and low cost.
The number of passwords users use on their transaction systems in recent times has been on the increase (McCarney et al., 2012). Such systems are often found in online banking, web mail sites, and social media sites e.t.c. For example, it has been generally noted that an average Nigerian keeps about seven (7) to ten (10) passwords for their different systems at home, offices and for their financial transactions. This means that at every point in time, they are confronted with the problem of memorizing and recollecting all their unique passwords for the systems on which they want to transact at a particular time. It has been observed that as the number of these passwords increase, users find it more difficult to recollect the appropriate password for a particular transaction at a particular time, hence, the need for this paper. Password managers are designed to relieve password fatigue and reduce login time (McCarney et al, 2012). They can also indirectly facilitate better password quality and a reduction in password reuse.

To use a password manager, existing accounts must be migrated into the manager and potentially replicated across multiple devices (McCarney et al., 2012). Major research efforts in this area have been found to include password managers, strengthening password quality and the use of alternative authentication mechanisms into passwords (i.e. graphical or object-based passwords). In this paper, we focus on developing a password manager for multiple accounts password. The remaining section is organized as follows: section 2 reviews a number of relevant literatures on password manager section 3 describes the methodology for the proposed system while section 4 and 5 describe the results and concludes the work respectively.

2. RELATED WORK

The proliferation and popularity of password as the commonest method of authenticating users to enhance security has attracted attentions of many researchers. A number of such relevant researches were reviewed in this paper. (Tam et al., 2010) examined five (5) password-management behaviours with the intention of investigating users knowledge of password quality, users motivation for password selection, and the effect of account type on password management behaviours. Their results showed that users understand the difference between good and bad passwords and their resulting consequences. They also found that the motives behind password selection and password behaviours are complex. Their research contribution provided a new way of looking at password management behaviour. (Tam et al., 2010) used Construal Level Theory to discover that trade-offs between security and convenience as an important determinant of password quality. They showed that this tradeoff could be positively influenced by imposing a time-frame factor. Their research efforts clearly provided a first step towards improving password management behavior, they however did not solve problem of managing multiple passwords that most users have passcodes.

Mannan and Van Oorschot (2008) introduced an object-based password scheme called ObPwd. ObPwd enables users to select a Security threats that such approaches must address, known attacks, and methodological issues related to empirical evaluation were also discussed. Although further research and improved methodology were also identified this did not include multiple password scenario. (Boyen, 2007) revisited the venerable question of "pure password"-based Key Derivation and Encryption, and expose security weaknesses in current implementations that stem from structural flaws in Key Derivation Functions (KDF). In this work, they advocated for a fresh redesign, named Halting KDF (HKDF), which was thoroughly motivated on the following grounds: (i) By letting password owners choose the hash iteration count, they gained operational flexibility and eliminate the rapid obsolescence faced by many existing schemes. (ii) By throwing a Halting-Problem wrench in the works of guessing that iteration count, they widened the security gap with any attacker to its theoretical optimum. (iii) By parallelizing the key derivation, they allowed legitimate users exploit all the computational power they can muster, which in turn further raises the bar for attackers. (Boyen, 2007) submitted that HKDFs are practical and universal because they work with any password, any hardware, and a minor change to the user interface.

As a demonstration (Boyen, 2007) offered real-world implementation for the a password generating object from their local collection or from the web, and then converts the password object (e.g. an image, a particular piece of music, excerpt from a book) to a (potentially) high-entropy text password that can be used for regular or secondary web authentication, or in local applications (e.g. encryption). Instead of requiring users to memorize an exact password, ObPwd only requires one to remember a hint or pointer to the password object used. They implemented a prototype, and solicit feedback from the research community in regard to using digital objects as passwords (Mannan and Van Oorschot, 2008). (Sreelatha and Shashi, 2011) proposed user authentication using native language passwords was proposed whereby a user is expected to select a character from his native language and also submits the shape of that character in a grid when creating the password.
This ensure that the is correctly authenticated using these information when there is need to login. This idea is based on their belief that users can remember their native language passwords better than passwords is any other language (Sreelatha and Shashi, 2011). The proposed scheme is found to be resistant to eavesdropping, brute-force attack, shoulder surfing and hidden camera. The research observation shows that although users are able to remember their character, they always find it difficult to remember the password shape. Another drawback observed is that password registration and login takes time in the proposed scheme. (Ross et al., 2005) described a simple browser extension called PwdHash, that transparently produces a different password for each site, improving web password security and defending against password phishing and other attacks. Since the browser extension applies a cryptographic hash function to a combination of the plaintext password entered by the user, data associated with the web site, and (optionally) a private salt stored on the client machine, theft of the password received at one site will not yield a password that is useful at another site. While the scheme requires no changes on the server side, implementing this password method securely and transparently in a web browser extension turns out to be quite difficult. They describe the challenges faced in implementing PwdHash and some techniques that may be useful to anyone facing similar security issues in a browser environment (Ross et al., 2005).

Yee and Sitaker (2006) described Passpet as a tool that improves both the convenience and security of website logins through a combination of techniques. They reported that Password hashing helps users manage multiple accounts by turning a single memorized password into a different password for each account. They proposed new improvements to these techniques, discussed how they are integrated into a single tool, and compare Passpet to other solutions for managing passwords and preventing phishing. (Thomas, 2012) introduced Enterprise Random Password Manager (ERPM) which was designed to deal with the problem of privileged password management in cross-platform enterprise environments. In ERPM, when a user needs to login, he or she logs on to the ERPM web console and requests a password for the account. Depending on how the EPRM is configured, the request might be approved automatically, or the user might need to wait for approval. Either way, when the request is approved, ERPM will issue the user a complex temporary password for the account. This password can be displayed on the screen, sent through email, or transmitted though a text message.

ERPM ensures that the password has been synchronized on the related system before issuing it to the user. Unlike typical administrator passwords, this password is valid for a limited time only, before it expires and the password is reset. Administrators also have the option of checking in a password, at which point the password will be reset ahead of schedule. (Bonneau et al., 2012) evaluated two decades of proposals to replace text passwords for general-purpose user authentication on the web using a broad set of twenty-five usability, deployability and security benefits that an ideal scheme might provide. The scope of proposals surveyed was also extensive, including password management software, federated login protocols, graphical password schemes, cognitive authentication schemes, one-time passwords, hardware tokens, phone-aided schemes and biometrics. Their comprehensive approach led to key insights about the difficulty of replacing passwords. They concluded that many academic proposals have failed to gain attraction because researchers rarely consider a sufficiently wide range of real-world constraints. Beyond the analysis of current schemes, their framework provided an evaluation methodology and benchmark for future web authentication proposals (Bonneau et al., 2012).

3. METHODOLOGY

3.1 Proposed Architectural Framework Design
The Architectural framework for the password manager proposed in this paper as described in Figure 1 is designed based on the text-based password authentication scheme. According to the figure, the architectural framework is divided into three major modules, namely; user’s accounts and login module, the security module and the authentication module.
User’s accounts and login module: This module depicts all user activities involving attempts to gain access to the password manager, supply of minimum number account owned by the user and culminating in the display of all the accounts owned by the user. In this module the user loads the password manager, create accounts and supply minimum information about the accounts owned by him/her. Then the system displays all the accounts owned by the user. Security module: This module is the second module in the architecture. It consists of the integrated database (Security Level 1) and the cloud database (Security Level 2). It is the engine house of the password manager because it houses all the repositories of various accounts passwords and passcode information and helps to secure these information. It is called the security module because it provides two levels of security for the passwords in a manner that all users accounts passwords are encrypted and stored in Level 1 and each time any of these passwords are changed, a copy of all these passwords is made in the cloud database to ensure their security. This is the reason why it is called the second level of security for the passwords. Authentication module: This is the module where a particular user account password meant for a particular transaction at a time is located, compared with the copy in the cloud database before it is authenticated for the transaction to take place.
3.2 Methodology for System Operational Design at Interface Level

With the proposed password manager system S running and invoked through the user login, the system demands for the password for the desired transaction accounts. Two assumptions are made at this point. First, it is assumed the user has forgotten his password for transacting on that account. It is also assumed that the user would be able to remember at least one or two of his/her several transaction account names. Therefore, the user is expected to supply at least one of his multiple transaction account names. When the account name is supplied, the proposed password manager system displays all the names of all the transaction accounts owned by the user. With this, the user is reminded of all his/her account names. The user would then be expected to select the name of the transaction account on which he intends to transact at that particular time. Once this is done, the password manager system automatically generates the password meant for that particular account chosen by the user for instant transaction to take place. In case the transaction account password is changed, the password manager system facilitates the transfer and replacement of the new password as encrypted data between the Level 1 security (local integrated database) and the Level 2 (cloud database).

3.3 Methodology for the proposed System at Operational Level

At operational level of the proposed password management system S, every user is made to have a name code and for every transaction account created by the user a passcode is given called acct code so that when a user is creating a new transaction account the system allows him to supply his name (name code) and a pin code (called the actual code) among other information like his/her saddest moment. Let the password structure be defined as $αβPwd$ where $α$ = name code (as unique and known to all transaction accounts (TA) in the system, $β$ = acct code is the unique code that is created and assigned to a particular transaction account (TA)), and $Pwd$ = is the actual code (password) created by the user for a particular transaction account such that

$$workingcode= αβiPwd_i$$ (1)

where

$n$= number of passwords owned by the user and there exists a symmetric relationship between passwords in the integrated database and the cloud database as defined by $(αβiPwd_iLevel1) \cap (αβiPwd_iLevel2)$ if $(αβiPwd_iLevel2, αβiPwd_iLevel1) \in \mathbb{R}$ for all $αβiPwd_iLevel2, αβiPwd_iLevel1 \in \text{Level1 and Level2 such that } f^{-1}((\text{exCi}αβiPwd_iLevel2)) = (\text{exCi}αβiPwd_iLevel1).$ and $\text{exC}$ stands for extra code for login.

The system is designed in such a way that when a user creates a transaction account, he is asked to supply a number of other information including information about the user “most unhappy moment”. To use the password manager system, the user is expected to supply the name of any of his transaction accounts he could recollect and information about his most unhappy moment as supplied when creating the account. At the time the transaction account is being created, the system generates a name code and acct code for the account name and account code respectively and attach to password (actual code) supplied by the user to form the working code. These are encrypted and stored in the integrated local database (Level 1 security). A copy of this is immediately made on cloud database (Level 2 security) associated with the local integrated database. In the same manner, the date value supplied by the user as his most unhappy moment is evaluated using a Hash function technique and copied on the cloud database. Whenever a user forgets his password for any of his transaction accounts, the only thing he has to do is to select the option for the transaction of choice at the time and supply correct date value for his most unhappy moment. Once these are correctly supplied, and because his name code is made unique and known to all his transaction accounts while his acct code is unique to every transaction account he keeps and these two code are embedded in the working code the account name supplied by the user is compared with the working code encrypted in the database and the hash values for the transaction account is reverted from the cloud database and also compared with the most unhappy value for a match, once they match, the user required password is auto-generated and authenticated for the transaction to take place.
To show that the proposed system S is capable of effectively generating the password for the desired transaction at a particular time, decision tree is used to model the system. To do this, we have to first establish that if the proposed system S with m-ary structure of height h has l TA, then 
\[ h \geq \left\lceil \log_m l \right\rceil \]
and also establish that if the system m-ary tree is full and balanced, then 
\[ h = \left\lceil \log_m l \right\rceil. \]

Taking logarithms to the base m shows that 
\[ \log_m l \leq h \] because h is an integer, we have 
\[ h \geq \left\lceil \log_m l \right\rceil. \]
Now suppose that the system is balanced. Then each TA is at level h or h-1, and because the height is h, there is at least one TA at level i. It follows that there must be more than mh-1 leaves (TA). Because l \leq mh, we have 
\[ mh-1 < l \leq mh. \]
Taking logarithms to the base m in this inequality gives 
\[ h-1 < \log_m l \leq h. \]
Hence, 
\[ h = \left\lceil \log_m l \right\rceil. \]

Suppose there are nine (9) transaction accounts with the same degree of probability for selection as a desired transaction account to make a total of ten transaction accounts.

To determine which of the transaction accounts is to be selected at a time, we need to determine the number of comparisons that are necessary using the proposed system. Since, there are ten transaction accounts of the same probability for selection. There are three (3) possibilities for each comparison of two passwords at a time namely; two passwords are right or the first password is right while the second password is wrong or the first password is wrong while the second password is right. Therefore, the proposed system S adequately models the decision tree for the sequence of comparisons is a 3-ary tree. Therefore, there are at least ten leaves (TA) in the in the system S because there are ten possible outcomes (because each of the ten TA’s can be selected and each possible outcome must be represented by at least one TA. The largest number of comparisons needed to determine the right password for transaction is the height of the decision tree (S) as represented in Figure 2. From Alberto et al. (2002), it follows that the height of the decision tree (ie. S) is at least 
\[ \left\lceil \log_3 10 \right\rceil = 3. \]
Hence, at least three (3) are needed to select the right transaction password in the proposed system S.
4. VALIDITY PROOF FOR THE PROPOSED SYSTEM

4.1 Mathematical Induction step of Proof
To prove the validity of the proposed system $S$, it is modeled with a rooted tree so that it can shown that it is balanced if all its transaction accounts $TA$ are at levels $h$ or $h-1$. Since a rooted $m$-ary tree of height $h$ is balanced if all it leaves are at levels $h$ or $h-1$ (Kenneth, 2007). To prove this, we need to prove that there at most $mh$ transaction accounts $TA$ in the proposed system $S$, using mathematical induction on height. First we consider a system $S$ of height 1. This system consists of a root with no more than $m$ children, each of which is a transaction. Hence, there are no more than $m1 = m$ transactions in a system $S$ of height 1. This is the basis step of the inductive argument. We then assume that the result is true for all systems ($m$-ary trees) of height less than $h$, this is the inductive hypothesis. Let $S$ be an $m$-ary tree of height $h$. The transaction accounts $TA$ of $S$ are the $TA$ of sub$S$ obtained by deleting the edges from the root to each of the vertices at Level 1 as shown in Figure 3. Each of these sub$S$ has height less than or equal $h-1$. So, by the inductive hypothesis, each of these rooted tree (system $S$) has at most $m(h-1)$ transaction accounts $TA$, because there are at most $m$ such subs, each with a maximum of $mh-1$ transaction accounts $TA$, there are at most $m.mh-1 = mh$ transaction accounts $TA$

![Figure 3: Steps for the Inductive Proof](image)

4.2 Evaluation Results
The proposed password manager system design is implemented with Microsoft Visual C#.NET with Microsoft SQL Server and MongoDB as the back-end. The password manager system is deployed on one hundred (100) networked systems for 100 different users to test at the same. Each user is allowed to use ten (10) different transaction accounts for the test. Response from the users indicate that the proposed system is capable of using the password manager at the same time on the same network. Some of the sample implementation snapshots are as indicated in figure 4,5,6,7 and 8.
Figure 4: Interface for password account in Facebook site

Figure 5: Interface for retrieving password information in the system
Figure 6: Interface for auto generate password for transaction

To evaluate the extent to which the proposed password manager system can be used by users to achieve specified ability attributes like Learnability, Efficiency, Memorability, Errors, and Satisfaction (Alberto et al., 2002) information are obtained from 100 users. The information obtained from the users are represented in the Tables 1-5 and analyzed in figures 7,8,9,10 and 11

Table 1: User’s measure of password managers in terms of their Efficiency

<table>
<thead>
<tr>
<th>Usability Attribute</th>
<th>KeePass</th>
<th>Lastpass</th>
<th>Browser-based</th>
<th>Our Proposed System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Efficiency</td>
<td>25</td>
<td>23</td>
<td>22</td>
<td>30</td>
</tr>
</tbody>
</table>

Figure 7: Graph Analysis of measure of systems efficiency
Table 2: User’s measure of password managers in terms of their Learnability

<table>
<thead>
<tr>
<th>Usability Attribute</th>
<th>KeePass</th>
<th>Lastpass</th>
<th>Browser-based</th>
<th>Our Proposed System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learnability</td>
<td>18</td>
<td>15</td>
<td>25</td>
<td>42</td>
</tr>
</tbody>
</table>

![Learnability Graph](image)

Figure 8: Graph Analysis of measure of systems Learnability

Table 3: User’s measure of password managers in terms of their Memorability

<table>
<thead>
<tr>
<th>Usability Attribute</th>
<th>KeePass</th>
<th>Lastpass</th>
<th>Browser-based</th>
<th>Our Proposed System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memorability</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>75</td>
</tr>
</tbody>
</table>

![Memorability Graph](image)

Figure 9: Graph Analysis of measure of systems Memorability
Table 4: User’s measure of password managers in terms of their Satisfaction

<table>
<thead>
<tr>
<th>Usability Attribute</th>
<th>KeePass</th>
<th>Lastpass</th>
<th>Firefox</th>
<th>Our Proposed System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Satisfaction</td>
<td>20</td>
<td>25</td>
<td>20</td>
<td>35</td>
</tr>
</tbody>
</table>

Figure 10: Graph Analysis of measure of systems Satisfaction

Table 5: User’s measure of password managers in terms of errors they contain

<table>
<thead>
<tr>
<th>Usability Attribute</th>
<th>KeePass</th>
<th>Lastpass</th>
<th>Firefox</th>
<th>Our Proposed System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Errors</td>
<td>25</td>
<td>60</td>
<td>10</td>
<td>5</td>
</tr>
</tbody>
</table>

Figure 11: Graphical Analysis of measure of systems errors
The foregoing data and analysis as obtained from 100 users during testing show that the proposed password manager system is highest in efficiency, learnability, memorability, and satisfaction but lowest in occurrence of error(s) when compared with other existing password managers.

5. CONCLUSION AND FUTURE WORK

The trend in every society of the World is fast moving towards a situation where the number of passwords that an average user has continue to increase on daily basis owing to their participation in transaction accounts ranging from online banking systems, web mail sites, to social media sites. Although a number of password managers already exist, the idea proposed in this paper is meant to bring to fore another unique perspective to the development of password manager. This paper presents a unique perspective to the development of password manager that affords a user to have automatic access to any of his passwords amongst several others across his multiple transaction accounts. The evaluation report of this research shows this proposed system promises to be the best password managers when implemented. Future research effort would be geared toward improving the system to handle up to about twenty (20) transaction accounts password.

REFERENCES