Requirements

Jackie Law (jjl284) *jjl284@cornell.edu*
Nishad Mathur (nm594) *nm594@cornell.edu*
Ning Wang (nw265) *nw265@cornell.edu*
Zhan Zhao (zz444) *zz444@cornell.edu*

March 9, 2017

1 Personnel

- Jackie Law (jjl284) *jjl284@cornell.edu*
- Nishad Mathur (nm594) *nm594@cornell.edu*
- Ning Wang (nw265) *nw265@cornell.edu*
- Zhan Zhao (zz444) *zz444@cornell.edu*

2 System Backlog

In this document we will use the following terminology: **accounts** contain information that authenticate users and **keychains** are collections of passwords and their metadata. Sharing and encryption are done at a keychain level.

2.1 Completed

User type	Assets	Importance	User story
customer	password	M	As a customer, I can add passwords to a keychain.
customer	password	M	As a customer, I can create keychains to store a group of passwords so that I can organize my passwords.
customer	keychain	M	As a customer, I can use a master password to access my keychains so that I dont need to remember a different password for each keychain..
customer	metadata	M	As a customer, I can store metadata associated with the password so that I can remember what the password is for.
customer	password	M	As a customer, I can edit passwords stored in a keychain that I created.
customer	password	M	As a customer, I can view the password stored in a keychain that I created.
customer	metadata	M	As a customer, I can view metadata stored in a keychain that I created.
customer	metadata	M	As a customer, I can edit metadata stored in a keychain I have permissions to edit so that I can keep my metadata up to date.

2.2 Backlog

User type	Assets	Importance	User story

customer	account	M	As a customer, I can create a new account to use the password manager.
customer	keychain	W	As a customer, I can use different master passwords for different keychains so that I can share some keychains but not others.
owner	password	S	As a password owner, I can grant editing permission of a keychain to another customer so that we can share passwords.
owner	password	S	As a password owner, I can revoke editing permission of a keychain to another customer so that we can unshare passwords.
owner	password	S	As a password owner, I can grant viewing permission of a keychain to another customer so that we can share passwords.
owner	password	S	As a password owner, I can revoke viewing permission of a keychain to another customer so that we can un-share passwords.
owner	password	S	As a password owner, I can remove a password from the system, and thereby also removing it from the accounts that have access to it (after synchronization).
owner	password	S	As a password owner, I can review logs of modifications made to a keychain so that I can see who edited, viewed or deleted a password.
editor	password	M	As a password editor, I can modify a password and its metadata.
viewer	password	M	As a password viewer, I can see a password and its related information.
customer	password	S	As a customer, I can have the system generate a random password for me.
customer	password	S	As a customer, I can specify the length of a generated password so that I can use it where there may be specific requirements.
customer	password	S	As a customer, I can specify the character sets used to generate a password so that I can use it where there may be specific requirements (example: characters it must or must not have).
customer	password	S	As a customer, I can save a password generation scheme to save me re-entering the details.
customer	password	S	As a customer, I can load a saved password generation scheme to save me re-entering the details.
customer	password	C	As a customer, I can have the strength of my passwords evaluated so I can see if I need to change them.
customer	password	W	As a customer, I can receive suggestions as to how to improve the quality of my passwords so I can improve them.
customer	password	W	As a customer, I can receive reminders to change my login details after a time period so that I can change them regularly.
customer	keychain	M	As a customer, I can synchronize keychains across multiple devices for convenience.
customer	keychain	S	As a customer, I can make changes to a keychain and have it synchronize across multiple devices.
customer	password	S	As a customer, I can access keychains while offline for convenience.
customer	password	C	As a customer, I can edit keychains while offline.

customer	password	S	As a customer, I can manually resolve editing conflicts from changes made while online.
customer	password	C	As a customer, I can manually resolve editing conflicts from changes made while offline.
customer	password	W	As a customer, I can automatically resolve editing conflicts from changes made while offline.
customer	password	W	As a customer, I can automatically resolve editing conflicts from changes made while online.
administrator	account	S	As a system administrator, I can remove unused accounts to improve efficiency.
customer	account	S	As a customer I should be able to edit an accounts details if they change to keep my account up to date .
customer	account	S	As a customer I should be able to login to my account to enable synchronization.
customer	application	W	As a customer, my copy of the application should always prompt me about updates to ensure I am running the latest version with the latest security fixes.
customer	application	W	As a customer, my copy of the application should be able to download updates for my convenience.
customer	application	W	As a customer, my copy of the application should verify the update's authenticity to ensure I only get safe versions.
customer	password	W	As a customer, I should be able to access my passwords through my native password management interface (if one exists).
customer	non-password secret	C	As a customer, I should be able to add non password secrets to a keychain (for example private key) as to synchronize and protect this data.
customer	password	W	As a customer, I should be able to import passwords and data from other password management tools.
customer	password	W	As a customer, I should be able to access my passwords from the web.
customer	password	W	As a customer, I should be able to edit my passwords from the web.
customer	password	W	As a customer, I should be able to access my passwords from my mobile device.
customer	password	W	As a customer, I should be able to edit my passwords from my mobile device.

3 System Purpose

Our goal is to build a password management system where the user can setup the system by first creating a 'keychain' in which they can store their passwords securely and secure this keychain with a master password; the user should then be able to synchronize these secured keychains between devices which the user controls. These keychains should be accessible offline and (depending on implementation) be immutable when offline or can reconcile changes made when offline with the master copy on the server.

The user should then be able to enter their password (and associated metadata, such as use case (e.g. Netflix), descriptions, etc) into the keychain and these should then be synchronized between client devices. If they so choose they should then be able to generate passwords according to a set of criteria and have these inserted into the keychain automatically.

The system should offer a service which allows the user to evaluate the strength of their stored passwords and (for example) detect password reuse across passwords in a keychain. A useful stretch

goal may be to interface with a service which provides notification of known hacks and alert the users if any of their passwords have been compromised, improving user security.

The last major feature is to allow the user to opt to share individual keychains with other users, for example, allowing work-flows such as shared household Netflix accounts. At any point, a user who has shared a password can decide to unshare their keychain. At this point, any future changes made to the keychain will not be visible to those that the password was shared with.

4 Threat Model

4.1 Threat Concern

An important consideration when considering the threats which may present themselves to a system are the intended class of users for the system. The intent is to design a system that home users, small business, small households and other groups should be able to securely store and share these users' passwords. As a consequence the scope of attacks whom may attempt to penetrate this system can be considered more closely and can meaningfully inform how best to allocate resources when building the project. As a consequence the intent of the initial versions of this tool are not to protect the tool from every possible threat, as the scope would likely be far too large and many attacks are infeasible to defend from when using Java without strong control of the hardware and OS, but to focus on the threats that these groups may face.

An example of this would be an acquaintance attempting to access the users login details for some website (without permission) or an attacker with limited exploitation capabilities attempting to access a users passwords. It may also involve targeted attackers who can use existing exploits to attack a system but cannot discover new unknown exploits. In general, attackers who do not have access to the memory of the program (data or executable) should not be able to access users' passwords. A potential example of this is that a Dolev-Yao attacker who has fully compromised all network channels connecting to a user should not be able to access the plain-text passwords.

If we decide to implement this stretch goal and a user uses multiple master passwords for separate keychains, then they should be treated as if they were created by separate users for the purposes of this section. The compromise of one should not lead to the compromise of others.

Finally, the primary goal is to prevent disclosure of stored passwords. Any other concerns are secondary (although still important and relevant). This is due to the primary goal of attackers likely being to access the user's password details to access/forge the account associated with the password details.

The following is a list of attacks that our system will protect against:

- Compromise of the server - Whether by rouge employees, law enforcement or other attackers. There should be no way of accessing the user's passwords without the master password.
- Compromise of the network - Whether by the network operator, law enforcement or otherwise there should be no way to accessing the user's passwords without the master password.
- Malware with user level privileges - Malware which can read the file directly should not be able to access the passwords.
- Users without passwords - They should not be able to access the passwords without the master password.
- Users with the password for one keychain - They should not be able to compromise other keychains.
- Users who have previously had a keychain shared with them - They shouldn't be able to access future changes to the keychain.
- Compromised master passwords causing other users' passwords to be compromised - If a user's password is compromised then it should not be possible to access keychains which they do not have permissions to access. For example if User A has shared keychain B with user B and the attacker has compromised user B's password then the attacker should not be able to access other keychains created by user A.

- Attackers with keychain - They should not be able to decrypt the keychain without the master password or encryption key. For example if an attacker penetrated the synchronization server they should not be able to access the stored passwords.

4.2 Non-threats

The following is a list of threats which we consider out of scope when considering allocating resources for development. An effort may be made to mitigate or counter these hazards, but no guarantees are offered. The general aim is to trust that the client doesn't face any invasive malware, and the server and network cannot be trusted, as well as understanding that humans are a weak-link in the security chain.

- An attacker with access to client memory - There would likely be a result of either root access or other exploits allowing memory access.
- An attacker with root access to the client machine - This could likely encompass a variety of issues, for example system wide key-loggers accessing the password as it is typed in.
- An attacker with the ability to alter the application executable - As all operations are performed client side altering the client application and bypassing any verification methods which may be in place to prevent this means that the attacker could directly intercept any operations (such as accessing the master password) which can allow any number of problematic issues. It is difficult to implement without specific OS and/or hardware support, which is out of scope for this project.
- Users leaking passwords which they have been authorized to access - Fairly self evidently, by making a password accessible to a user, it then becomes impossible to prevent the user from doing with these passwords what they want. Auditing may help discourage this form of attack, but it impossible to prevent entirely.
- Users from leaking passwords after their access to a shared keychain has been revoked - This specifically is problematic due to features such as client side processing, offline support and compromised clients making any attempt to forcefully delete the shared keychain from the user effectively impossible.
- Social Engineering or Phishing attacks against the user - These attacks can lead to any of the above issues, memory access, root access, leaking of the master password and as explained above it can become problematic to protect against these threats. Education can help, but it is clearly not a perfect counter.

5 Security Goals

5.1 Security Goals

- The system shall prevent unauthorized users from viewing the passwords (for both shared and unshared passwords). This protects the **confidentiality** of the passwords.
- The system shall prevent unauthorized users from altering stored passwords (for both shared and unshared passwords). This protects the **integrity** of the passwords.
- The system shall detect unauthorized access attempts to an account and notify the user of the account. This warns against possible breaches of **confidentiality** or **integrity** of the passwords and the **integrity** or **availability** of the account depending on what changes the attacker makes to the account.
- The system shall prevent users from changing the permissions of keychains which they don't own. This protects the **confidentiality** of the passwords and **integrity** of the keychains.
- The system shall store a backup of the keychains so that there is another copy of keychains if a file. This protects the **availability** of the passwords.

6 Essential Security Elements

Authentication

Users will need their username and master password to login to the system. Currently, only a master password is required for access, but we may include more depth of protection as the design of the system updates. The master password will be required to have high strength: at least 8 characters long, has at least one upper case letter, a lower case letter, a number, and a symbol. The actual login key is generated using the master password, which ensures server side security since the application does not store user password in any form.

Similarly, a user will be authenticated by entering their master password in order to modify the master password. We may include multiple factor authentication for this step such as answering personal questions to add extra security to protect the access to a user's account.

Lastly, no one shall be able to gain access to an account without the master password. If a user forgets their master password, then unfortunately they would lose access to their account and stored passwords. This is a risk that the system has in exchange for confidentiality of user information.

Authorization

Users will only be able to access the list of passwords that they are the owner of or that were shared with them.

For each password stored, there are three levels of access privilege: owner, editor, and viewer. Viewers have access to see the passwords within a keychain and their associated metadata, but cannot modify any information. Editors also have view access, and in addition can also modify passwords in the keychain and their information. Owner has the highest privilege, and each keychain can have only one owner, but we may consider the possibility of multiple ownership in a group setting. In addition to editing and viewing, the owner can also share and unshare a keychain, granting or revoking other users editor or viewer access. They can also remove password or keychain. A removed password will be removed from the password lists of every user who has access to the keychain that the password was stored in. The owner can also transfer ownership, giving another user owner access and change their own access level to either editor, viewer, or none.

Audit

The system will log all user activities. For user login, the system will keep track of login status, locking up the system after a certain number (likely five) of consecutive unsuccessful logins within a five minute interval. A locked system will require additional user information to unlock. Moreover, the account holder will be notified if there is unusual behavior detected on their account (e.g. multiple failled login attempts, login from a new device).

The system also keeps track of password related activities such as creation, modification, and sharing, and the log will be available to the owner to detect unwanted changes such as modification of password by an ill-willed editor and accidental sharing of the password to a wrong user. Note that the actual values of passwords won't be kept in logs for security reason.

Confidentiality

The system protects the confidentiality of user information by enforcing security goals that prevent any user from accessing another user's account. We use authentication to ensure that logged in user is indeed the account holder.

We enforce confidentiality of passwords by authorization and separation of privilege. The system makes sure that a password is visible to only those who have been granted permission to see the password.

To protect the confidentiality of messages among server and clients over the networks that transports the passwords, the system will use encryption systems such as hybrid encryption scheme.

Integrity

The system protects integrity of passwords by the separation of privilege. Owners of passwords are trusted to grant editing permission to those who they trust to uphold the integrity of the

information. The system also logs modifications so that users can detect violations of information integrity.

Similar to confidentiality, the system will protect integrity of network messages between server and clients. The system will use signing to ensure that 1) the sender is indeed who the receiver is expecting and 2) missing or modification of packets will be detected and senders will be required to re-send the messages.