Table 1: Revision History

Date	**Developer(s)**	**Change**
October 4, 2017	Phil Pavlich	Rev.0 of Document
November 9, 2017	Thomas Mullen	Number Requirements
...	...	...

3XA3 Requirements Document

Group 20 (2020Vision)
Mullen, Thomas - mullentc - 001406837
Pavlich, Phillip - pavlicpm - 001414960
Bauer, Ivan - bauerim - 001418765
L02

1 Introduction

Password security is a problem that most people face multiple times a day. Breaches can result in personal data being lost, stolen or corrupted. With modern technology and specialized techniques, attackers are able to crack weaker passwords. A secure practice is to have different password for all your applications. This can help secure your devices and online services but the amount of different passwords can be difficult to remember. That is where password managers can become handy. They allow a user to store all your passwords in one location. That one location can only be accessed by inputting the master password.

2 Project Drivers

2.1 Purpose

The purpose of this project is to create a secure method of generating and managing passwords that is not vulnerable to existing brute-force attacks.

2.2 Stakeholders

In the 21st century, security is a major topic that is applicable to most people. It appears with computer systems, cell phones, banking, online services and many more personal services. Password protection allows a user to log in to a service and access personal information. If password protection is not secure, attackers would be able to to get that personal data by accessing the service. Our stakeholders consist of anyone that requires a password for some online or physical service.

2.3 Constraints

There will be difficulties testing the software to the full extent that attackers are capable of, due to the fact that it uses an unconventional cryptographic technique. Time will also be a constraining factor, as the deadline for completion is approximately two months from the writing of this document. Although an open source project will be used to guide development, this project may present other technical challenges that prove difficult to overcome.

2.4 Naming Conventions

For naming conventions, variables and functions will be written using camelCase. This means the first character is a lower case letter and future words begin with a capital. Modules and classes will begin with a capital letter so that they can be distinguished from a function or variable. At the top of every module, a description in the form of a comment will outline the purpose of the module and modifications that have been made to the module throughout development. The project will follow Standard JavaScript coding style.

2.5 Terminology

The "user file" consists of any data files that are stored locally on the system, not including any source code. This includes preferences. The "master password" is the main password that is required to access to all other passwords. A "generated password" is any password that is created and managed by the service, specifically not the master password.

3 Scope

HashPass is the open source Github project that the software is based on. Currently, this Chrome extension detects what domain you are on and combines that with a phrase that you enter as a key. Once those two strings are combined, it uses a hashing algorithm to provide the user with a password for the given domain. Our goal is to make improvements on this project. The scope of our project will involve making this app more secure and harder to attack. We will be adding more components into the hashing algorithm. It will still be done as a Chrome extension so this is only applicable for web users. It will be available for every web domain that exists.

4 Project Issues

The software is still vulnerable to an attacker that can observe key presses on the system, as well as access the user file. There will be no multi-factor authentication, so if the user's master key and user file are compromised they run the risk of all generated passwords being exposed.

Installation through the Chrome Web Store requires the user to trust Google's infrastructure. Although this is a suitable compromise for most users, it may be seen as a flaw for others.

4.0.1 Off-The-Shelf Solutions

There are currently several high quality password management systems, and all have slightly different features that make them unique.

The software will be based on a specific open source project, HashPass, in order to assist the team in designing some of the components that are planned for implementation.

HashPass uses a hashing function to combine the the domain name of an online service with the master password in a way that is difficult to reverse. This creates a unique password for each service. Although difficult to reverse, it is not impossible, and brute force attacks are possible if the service is compromised.

A more traditional system that will not be used as a reference is 1Password. This solution generates random passwords and encrypts them using the master password. This method is vulnerable to brute force attacks is the local device is compromised.

5 Functional Requirements

- R1: The software shall generate unique passwords from the master password and user file in a way that is difficult to reverse.
- R2: Passwords will only be generated when the user enters the master password into a form and presses submit.
- R3: The user will access this program through an extension on Google Chrome.

6 Non-Functional Requirements

6.1 Usability Requirements

- R4: The software must be easy to use without any prior knowledge of password managers.
- R5: Only limited experience with computers in general should be required (installing a Chrome extension, web browsing, typing).
- R6: The software should be accessible to users with impaired vision and other disabilities according to the W3C Web Content Accessibility Guidelines 2.0.

6.2 Safety Requirements

- R7: The software should not cause the host machine or any connected machines to operate in an unsafe way.
- R8: The software should be isolated from other software and not read/write any shared memory without the user's intentional action (clipboard memory, web page content, unrelated local files).

6.3 Performance Requirements

- R9: The software should be able to generate a password in less that 10 milliseconds. This time interval is small enough that the delay will be unnoticeable to the user.
- R10: The software should occupy no more than 5 megabytes on the system, including the compiled source code and any additional data.

6.4 Installability Requirements

- R11: The software should be installable from the Chrome Web Store by clicking "Install". No additional installation steps should be required for first time users.

- R12: If the user is migrating to a new system, the user can provide the "user file" to restore passwords and preferences. No additional installation steps should be required.

6.5 Operational Requirements

- R13: The expected environment is the Chrome 61 Javascript engine, specifically the extension environment. It will be assumed that Chrome operates identically across operating systems.

6.6 Security Requirements

- R14: The software should not store any data that can be used to directly determine the master password.
- R15: The master password and generated passwords must not remain in memory.
- R16: Both the user file, master password, and some publicly available name for the service must be required to generate a password for that service.

6.6.1 File Integrity Requirements

The integrity of system files is out of scope for this project. This is the responsibility of the host system.

6.6.2 Audit Requirements

Two classmates who were not involved with development will examine source code, unit tests and the deployed software in order to ensure that our software meets the appropriate specifications and standards.

6.7 Cultural and Political Requirements

- R17: The software must not use any words or phrases that are deemed offensive by more than 15 percent of a test group.

6.8 Legal Requirements

- R18: Personal information will be implemented so as to adhere to the Personal Information Protection and Electronic Documents Act.
- R19: The software's defaults will comply with the UCSC Password Strength and Security Standards.