Ric Nisley
Project Design Document

Contents

Section 1 – Product Description

I'm working with Ryan Harding. We are building a command-line-interface password manager. A user will be able to create an account, save a username - password – application set. Request a username and password for a specific application. It will include several GoLang files and a SQLite database. This will support a single user. The passwords will be encrypted within the database and each password will only be decrypted in the event that the user with the correct master password requests it.

Section 2 – Overview

2.1 Purpose

All users need secure passwords for a wide variety of applications, but human memory is iffy at best resulting in insufficient passwords and/or passwords being reused on multiple platforms. Our password manager provides a place where passwords can be stored securely and accessed easily so that user can user more secure passwords with greater variation across platforms.

2.2 Scope
This document explains the design of the software components of our password manager.

2.3 Concepts
The ability to store passwords for a myriad of accounts is core to the design. The idea is to allow easy access to passwords easily and securely so that the user can use complex and secure passwords without having to commit them all to memory. Passwords will all be encrypted within the database individually meaning that there is never a time that the database can be stolen and read. Passwords will be individually pulled from the database, decrypted in memory, and presented to the user. Any attempt to pull a password will be logged. Users will be warned upon a successful login of failed access attempts.

2.4 Requirements
Database is reasonably stored and accessible with authorization. Interface should be easy to use so it's worth the effort to the user. Passwords can be pulled using application or website name ignoring capitalization differences.

An important goal of the software is to store passwords securely so the passwords will be encrypted within the database to protect against leaks if the database file is stolen.

2.5 Non-Goals
Password manager is a project for class so no fancy UI is needed.

Multi user support is out of scope. It is intended that each user use a separate installation so that passwords for multiple users are not stored in the same place.

Database backup is out of scope, though it is recommended that password manager's database be backed up at regular intervals to reduce risk of data loss in the event of file corruption or deletion.

2.6 Outstanding Issues
Details of user authentication, password encryption, and logging are yet to be specified.

2.7 Alternative Design
The final design chosen is based on keeping only one user's information in the database. An alternative design was considered that would allow multiple users to use the same installation of the software. This was not pursued in an effort to make the software easy to purge in the event that a user fears security risk or chooses to stop using the software for some other reason. In this event the user may securely delete the database and know that their passwords are no longer on the machine without harming the functionality of any other user.

Section 3 – Use Cases
Users require multiple secure passwords for a myriad of platforms which ideally should be different from one another and difficult to guess, which can make them hard to remember. This will allow a user to store all their passwords in one place so that they can look them up easily while keeping them secure behind one master password for ease of use. This software is designed so that it can be installed multiple times on the same machine to allow multi-user machines to utilize the software without merging multiple users passwords into the same database.

Section 4 – System Architecture

Instances can run on separate machines, or multiple instances can be run on the same machine. Databases will be stored on the local machine and will not be accessible from other machines unless steps are taken by network administrators to make them available (this behaviour is out of scope and outside of the design of the application and may result in security and stability issues).

Section 5 – Data Design

Data is collected directly from the user and only returned at the request of the user with the submission of the master password. The database will only be able to store one username-password pair per web address or application description (ignoring capital letters). For example "Netflix", "netflix", and "NETFLIX" will be recognized all as the same entry and therefore only one username-password pair can be stored, however "netflix" and "netflix.com" will be recognized as separate entries with unique username-password pairs allowed.

Section 6 – API

Password manager accepts the following procedure calls:

New

This command initializes the software the first time and allows user creation. It will not work a second time.

Add

This command will allow users to add a new app name/website – username - password set to the database.

Update

This command will allow users to update the username and password for a specific app or website.

Get

This command will allow users to get the username and password for a specific app or website.

Help

This command will remind users of the proper format of procedure calls.

All procedure calls will be logged including whether they were successful or not. All procedures listed after "New" will require username and master password authentication. Users will be notified of failed attempts to run the application upon the next successful command execution.

Section 7 – User Interface Design

The user interface to the password manager is a command line window. Specific behaviors are described in other parts of this document.

Section 8 – Technical Design

The interface will allow read-write privileges to the database. The passwords will be salted encrypted with SHA256 when stored in the database and never decrypted within the database. They'll only be decrypted in system memory and presented to the user and then purged from memory.

Section 9 – Configuration

User is allowed to pull passwords and usernames when they provide the master password and the app/website name.

Section 10 – References

The following documents are useful for understanding this document.

- GoLang Documentation
- SQLite Documentation

END OF DOCUMEN T