Dewey Mitchem

Devin Dulay

April 29th 2017

CMSC 413 – Introduction to Cybersecurity

Password Manager

Contents:

* Project Description
* User Guide

# Project Description

The following project was written in C# using WinForms in order to create a lightweight, executable file that could be deployed on any windows machine. The system was also designed so that if the user deleted the application, the database would persist and could be accessed from another version of the application. The databases that are used for this application are an encrypted variant of a SQLite Database with the database and passwords encrypted for twice the encryption.

The WinForms contains three main interactive forms: *Login*, *Create Database*, and *Password Manager*, respectively. In the background are two helper classes: SQLiteDatabase and Crypt. SQLiteDatabase is the SQLite helping class that handles the connection between the file and the program. It also provides methods that allow the user to make queries against the database. Crypt is a helper class based off the Cryptography DLL file built by Microsoft. Using this class, we are able to encrypt the plaintext versions of passwords provided by the user as well as decrypt them at runtime.

## Program Flow

On launch, the user will be shown the *Login* form prompting the user to either create a new database or use an existing file paired with its associated password. If the user chooses to create a new database, the *Create Database* form will be launched to allow for generation of a new file wherever specified. It will then continue to the *Password Manager* form.

If the user decides to use an existing database previously generated paired with a correct password, the connection will be established and the *Password Manager* form will be launched. In the *Password Manager*, all the contents from the table will be generated on the left side for the user to select. Once a user double-clicks a node, it will fill in the appropriate contents on the right for the user. The user can also add, delete, or modify an entry if they choose to do so.

When creating a new password, the user must specify what the title of the entry should be, For Example: “Devins Password”. They should include a username as well. Finally, the user should fill out the password to complete the form.

To delete an expired username/password combination, simply clicking on a node and pressing the delete button will remove it from the database and the currently displayed tree. Modifying one is done in a similar fashion by clicking on a node and selecting the Modify button at the bottom.

## Database(s)

The backend database is a SQLite variant that when created by our program contains only one table of data. The database itself is encrypted to prevent malicious users or programs from accessing the database without permission using third-party applications. As a secondary measure in the event that the database is broken into, the passwords themselves are encrypted inside the database. So, in the event a malicious actor gets in, the only information that can be retrieved is the user and encrypted password.

The design decision to use an encrypted SQLite database has to do with the lightweight, portable functionality that it provides. The ideology behind this was to allow a user to copy their encrypted database to any system that they desire without the worry of exposing data or the hassle of installing clients to handle a database.

## Encryption/Decryption

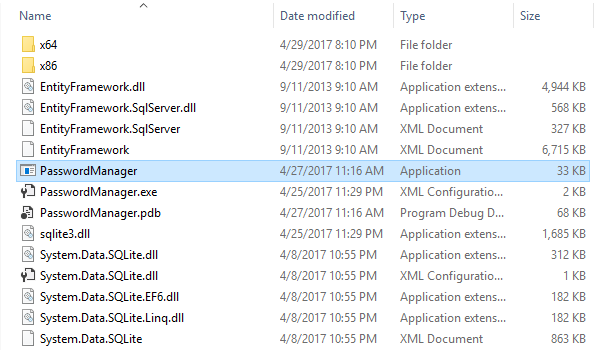
The system is protected from unwanted adversaries on two fronts: the database itself and its contents. The database is encrypted by setting a password on the SQLite system when the database is initially created using the SQLite DLL. With this password in place, the database will refuse to return any query statements without previously being unlocked and simply states that the file is encrypted.

Once the database has been unlocked or infiltrated by some other means, the user and password for each of the entries in the table are initially encrypted using a AES/SHA-256 cryptosystem and are only decrypted at runtime of the program using the specified password. This protects against potential brute force attempts on the database to obtain data leaving the adversary with an encrypted string of gibberish for both the user and password. The cryptosystem was imported by using the Cryptography DLL provided inside of c#.

# User Guide:

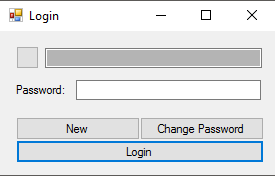
## Install/Open Step:

To begin, open the “PasswordManager.exe” executable file located where it was saved like the following screenshot below has indicated.



## Step 1 – Login Screen:

Once opened, the following “*Login*” screen below should appear. If you currently do not have a database made with this application, choose the **New** Database button to create a new database. If you currently have a database generated from this application, which will end with extension “.db3”, then continue to Step 3 for reading data into a table.



Login Password Textbox

“Change Password” Database Button

“Login” Database Button

“New” Database Button

Database Path Textbox

File Selection Button

**File Selection Button –** When this button is clicked, it will trigger a traversal system for the computer’s directory in order to locate a file. Choosing a file will fill out the **Database Path Textbox** with the path of the file for the user to see. Note: This system only supports “.db3” files generated by this application.

**“New” Database Button** – This button is used in order to create a new database to be logged with new passwords that the user wants to manage. Clicking this will take you to the “*Create Database*” as shown in Step 2.

**Database Path Textbox –** This textbox will be filled out with the path of a chosen file as a result of the **File Selection Button**.

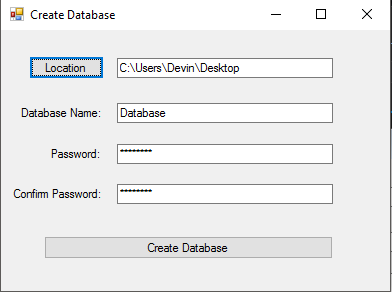
**Login Password Textbox –** This textbox serves as the line for a user to enter the password that correlates with the database file chosen. All characters as the type are masqueraded with “\*” for security reasons.

**“Login” Database Button –** If clicked, the database file and password will be used to attempt a connection to the underlying database. If successful, the “*Password Manager”* Form should appear with all the data currently saved into the database.

**“Change Password” Database Button –** Provided the user knows the password to the database, clicking this form will open a prompt to allow a user to change the current database’s password. Doing so will also re-encrypt all the contents of the database in the background to reflect these changes.

## Step 2 – Creating a New Database:

When the **New** button is clicked on the “*Login”* Form, the “*Create Database*” Form will appear to prompt you to create a new Database to save passwords to. To start, click the **Location** button and choose a location to save the Database to. Next, decide what to name the database by filling out the line next to the label “Database Name”. When the application generates a database for you, it will use the name given with the extension “.db3”. Like the example below, if my database name is “Database”, then the file generated will become “Database.db3”.

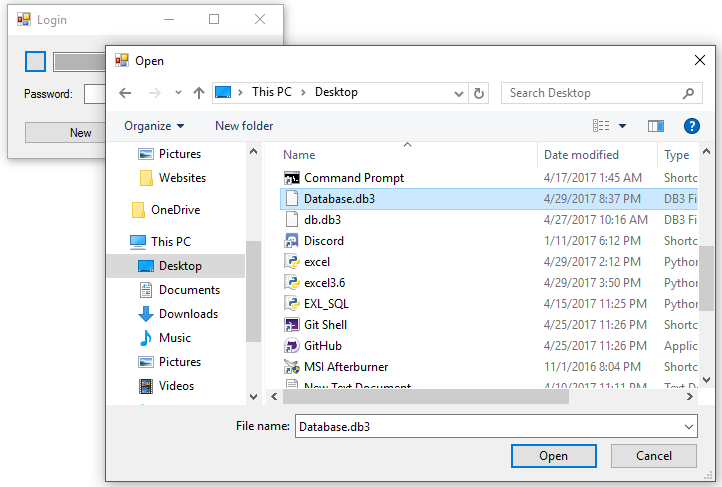


Once you have decided on a location for the file and the name of the file, a password to encrypt it with must be specified. Passwords must be between 8 and 32 characters in length for a baseline encryption of the file. For consistency, fill out the same password that you would like to encrypt the database with next on the lines next to **Password** and **Confirm Password** respectively.

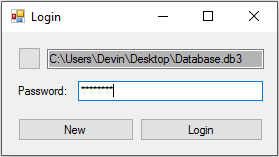
Once these steps have been completed successfully, click the **Create Database** button to create the file and to move onto Step 3 to add new passwords to the manager.

## Step 3 – Accessing/Reading from the Database:

As specified in Step 1, if you currently have a database for this system, click the gray **File Selection Button**  to choose the file. Note that only “.db3” files generated by this system can be used. When you have located the desired file that hosts the passwords, click the **Open** button to complete the selection.



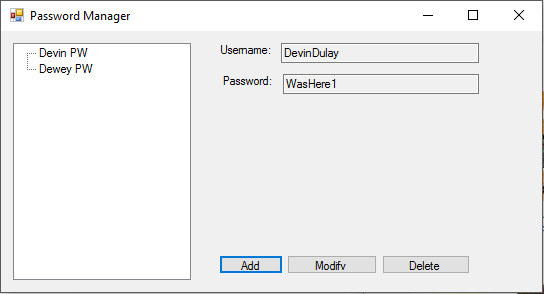
Once having chosen a file, enter the correct password in the password textbox. The password entered will be masqueraded with “\*” for security purposes.



Clicking **Login** will attempt to access the database with the given password. If successful, it will open the *“Password Manager*” Form. If the password is incorrect, or the file type is wrong, the user will be notified that it was incorrect.

Password Entries/Nodes

Selected Node Username



Modify Button

Delete Button

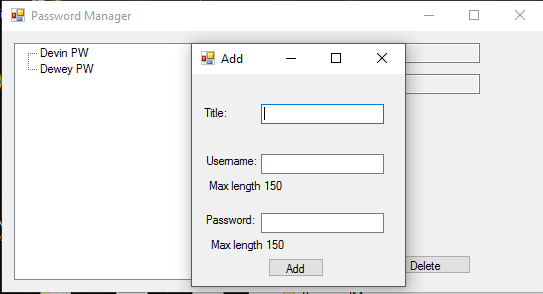
Selected Node Password

Add Button

To view the contents of all the stored passwords, simply double-click on one of the nodes that appear on the left side. When one of the nodes is selected, the user name and password will fill out their respective textboxes on the right. Note that these textboxes are read-only, so they cannot be altered from here and can only be copied by the user.

### Creating a New Entry:

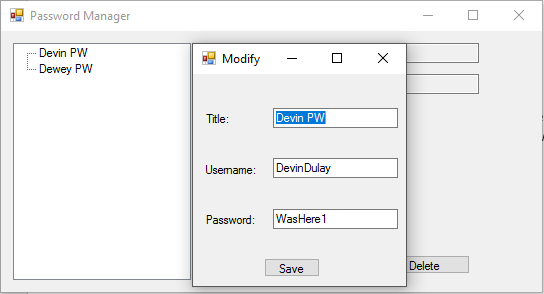
To begin, click the **Add** button on the bottom of the *Password Manager* form. A new form, *Add*, will appear prompting you to specify a required title, username, and password. The title will be what is displayed inside the program for the user to discern which account is which. Usernames and passwords are both capped at a maximum of 150 characters.



Once the appropriate fields have been filled out, click **Add** to confirm that the new entry can be added to the database. It will immediately show up on the left side to be accessed.

### Modifying an Entry:

In order to modify a current entry, highlight the desired node from the list on the left side of the form. Then click the **Modify** button on the bottom of the current form. This will open up another form called *Modify* to modify the current entry’s contents.



Once the desired corrections have been made, click the **Save** button to complete the transaction.

### Deleting an Entry:

If a node no longer has become obsolete, an option exists to delete specific nodes. To do this, first select the node to be deleted from the list on the left-sided list. Next click the **Delete** button at the bottom of the current *Password Manager* Form. This will immediately remove the node from the list, so it is imperative that the user makes sure they know what they are deleting.