PassUSB

Team Number 10: Sonali Benni, Rey Jairus Marasigan, Chidiebere Otuonye, Kaiya Roberts, and Gentman Tan

Florida International University

CEN4010 U02: Software Engineering I

Kianoosh G. Boroojeni

March 11, 2022

Abstract

PassUSB

Introduction

With the prevalence of information technologies, there exists an ever-increasing need for individuals to secure one's own access to online accounts. The typical method of doing so requires the user to create a secret passphrase that they would then be responsible for memorizing in order to access a given system. However, several factors make such a task difficult and unsafe; first, the exponential rise in computational power has led to the feasibility of "brute force attacks", in turn forcing IT administrators to enforce increased password length and complexity. Another effect that the increase in password length has is making the memorization of multiple different passwords difficult, thereby incentivizing individuals to unsafely reuse their own passwords. With these shortcomings in mind, we propose a system that would solve all of these issues in a single package.

Purpose of system

Solve the security issues facing PC end users in the realm of password authentication

Scope of the system

In scope: Multi-platform mobile application, password management, multi-plaform USB keyboard emulation, mitigations against man-in-the-middle, replay and spoofing attacks Out of scope: Application data security, side-channel attacks

Objectives and success criteria of the project

- Provide a mobile app for users to create and store passwords
- Provide a USB hardware dongle that can be paired with the mobile app which can type in passwords in lieu of keyboard input

Definitions, acronyms, and abbreviations

- PassUSB: the project's USB dongle solution that emulates a USB keyboard
- Password manager: a computer program that generates, stores and retrieves passswords for its users
- HID (Human Interface Device): a computer device that facilitates communications between a computer user and a computer
- App: a computer application
- Pairing: the process of recognition and acknowledgement between the mobile device and USB dongle

Overview of document

The project will consist of two types of coding assignments, one for frontend development i.e. mobile app development, and the other for backend development i.e. microcontroller programming. The mobile app will prompt the user to create a new password database, in which he/she will then enter a master password that is to be used to secure the database. The user will be given an option to pair the PassUSB with the app. Should the user choose to or not choose to pair the PassUSB, the user is then able to utilize the app's password generation, management and storage features.

Requirements Elicitation

Use Case ID:	UC-232
Created By: Gentman Tan	Last Updated By: Gentman Tan
Actors:	• User • MobileDevice • PassUSB
Description:	Initialize a secure connection between the Owner's MobileDevice to the PassUSB for future device communication

Preconditions:	• The User has created a database • The User has opened the app and unlocked their database • The User has selected the menu option to pair a new device
Normal Flow:	• User is prompted to enable Bluetooth discovery mode • Once found, the Bluetooth dongle's name and serial number is displayed • User is prompted to either allow or decline the pairing process with the displayed bluetooth dongle • When the "Allow" option is chosen, the MobileDevice and PassUSB is subsequently paired

Use Case ID:	UC-1-AA
Use Case Name:	Add account
Created By: Rey Jairus Marasigan	Last Updated By: Rey Jairus Marasigan
Date Created: February 21, 2022	Last Revision Date: February 21, 2022
Actors:	• User
Description:	This use case is used for specifying the different elements of our system that the DeviceUser goes through to add an entry to a list of account entries within our app. The ideal outcome of this use case is that the account is added to the database without any adverse effects.
Preconditions:	• The User has created a database • The User has opened the app and unlocked their database • The User is in the correct directory to be able to see and touch the plus icon.
Postconditions:	• At minimum: the system retains consistency and integrity in case of an error or deviation from the flow of events i.e. the user can "add account" again without repercussions from previous try. • Everything in harmony and works without error: Account is added to the database. • The account added can be accessed after being saved to the database.

Normal Flow:	(assumes the user wants to generate password when adding the account) 1. The DeviceUser presses the plus icon on the top right 2. The view switches to a display that shows 3 text fields: a website field, a username and a password field. The first two will be required to be filled using the pop-up keyboard. The password field can either be manually filled or automatically generated using the associated button located to its left. <generate password use case> 3. The user presses a button that opens to another view that shows multiple fields, boxes, and sliders to be filled, checked, and adjusted, respectively, to meet a certain criteria associated with the password. 4. The user enters the field with a pop-up keyboard on their phone. 5. Below the fields, the users checks various boxes or switches that configures the password generated by the apps such as: • contains special characters- an option that includes special characters in the password • contains numbers- an option that includes number • etc. 6. Below the switches, there exists a slider that controls how long the password will be. When it is adjusted towards the right, it increases the length. When it is adjusted towards the left, it decreases the length. The maximum length of the slider occupies the entire width of the screen with some deadzone. The minimum length is 8 characters and the maximum is 32 characters. Initially the slider is set to 12. 7. As the user changes the length of the desired password, a text box that shows the password itself changes. The password is shown in its pure form i.e. the actual password to be used. Here, the password is changeable by pressing on the box and using the pop-up keyboard. Initially, the password generated here is from the combination of the slider and checkboxes. 8. The user then presses the "save password" button below to generate the password. <generate password use case> 9. The user finally then presses the “add account” button below to add the account to the list. 10. A notification pops up that lets the user know that the account has been added. 11. The state of the app then returns to its default view.

Alternative Flows:	2b. In step 2 of the normal flow, if the user chooses to manually enter the password rather than generate a new password, 1. The entire normal flow can skip to step 9 7b. In step 7 of the normal flow, if the user changes the generated password and then slides the password slider, 1. An entirely different password is generated. 2. The new password is instantly displayed within the text box. 3. The normal use case continues thereafter on step 7 where the user can edit the newly generated password. 9b. In step 9 of the normal flow, if the website field and username field is not filled 1. A notification pops up to let the user know about the required fields 2. The app is scrolled the field needed to be filled 3. The fields are highlighted red to indicate requirement. 4. The keyboard pops up automatically 5. Once the fields are filled, the use case enters step 8 of the normal flow.
Exceptions:	Exceptions to adding account to list 9b. In step 9 of the normal flow, if the user does not enter a website or username.
Includes:	editAccount- steps 3 to 7 would be required to edit an entry within the list and can be separated in its own use case
Frequency of Use:	On-demand
Special Requirements:	• Users should not wait >20 seconds for an action to be performed • Animations should last <250ms • Colors should be made to be accessible to colorblind • Application should be made available on all major mobile platforms
Assumptions:	• The Customer has installed and is using the application • The application is readable to the Customer • The client has the ability to use the keyboard and interact with the screen.

Notes and Issues:	• Is the minimum and maximum possible length of the password generator feasible for the users? • Should there be a limit to the amount of accounts added to the list? • What are the different switches that we can include in the forum? • should the app request for the master password (the password required to enter the app)or other biometrics for the ability to add an entry to the list? • Should we use other characters outside of the printable ascii characters?