BYPASS: RECONSIDERING THE USABILITY OF PASSWORD MANAGERS

TINA SAFAIE

A THESIS

IN

THE DEPARTMENT OF

CONCORDIA INSTITUTE FOR INFORMATION SYSTEMS ENGINEERING

PRESENTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF MASTER OF APPLIED SCIENCE

IN INFORMATION SYSTEMS SECURITY

CONCORDIA UNIVERSITY

MONTRÉAL, QUÉBEC, CANADA

APRIL 2021

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Tina Safaie**

Entitled: **ByPass: Reconsidering the Usability of Password Managers**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science
in Information Systems Security

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

Dr. Walter Lucia _____ Chair

Dr. Mohammad Mannan _____ Supervisor

Dr. Amr Youssef _____ Supervisor

Dr. Elizabeth Stobert _____ Supervisor

Dr. Jermey Clark _____ Examiner

Dr. Walter Lucia _____ Examiner

Approved _____ *Dr. Mohammad Mannan, Graduate Program Director*

Chair of Department or Graduate Program Director

_____ *April 2021* _____ _____

Dr. Mourad Debbabi, Acting Dean

Gina Cody School of Engineering and Computer Science

ABSTRACT

ByPass: Reconsidering the Usability of

Password Managers

Tina Safaie

Since passwords are an unavoidable mechanism for authenticating to online services, experts often recommend using a password manager for better password security. However, adoption of password managers is low due to poor usability, the difficulty of migrating accounts to a manager, and users' sense that a manager will not add value. In this work, we present ByPass, a novel password manager that is placed between the user and the website for secure and direct communication between the manager and websites. This direct communication allows ByPass to minimize the users' actions needed to complete various password management tasks, including account registration, logins, and password changes. ByPass is designed to minimize errors and improve usability. Our goal is to create a space where security could be the users' primary task, and allow them to focus cleanly and consistently on account management tasks. The constancy of the ByPass interface is intended to allow users a greater sense of control over their passwords and accounts. By using the API to move account interactions into this space, we hope to create an interface where users

knew where to address security concerns, and access the controls to address those concerns. Current password managers hint at this functionality (and include innovative tools, such as security audits) but their placement outside the authentication interaction hampers the functionality they are able to support.

We conducted a usability evaluation of ByPass and found that this approach shows promising usability, and can help users to better manage their accounts in a secure manner.

We also conducted a security analysis of ByPass and showed the security improvements that can be achieved with the support of APIs for password managers. Our study shows that many known security vulnerabilities can be eradicated from the foundation of password managers, and significant usability can be gained with the inclusion of APIs support for password managers.

Acknowledgments

I would like to express my sincere thanks to my supervisors, Dr. Mohammad Mannan, Amr Youssef, and Elizabeth Stobert, for their continued support and guidance. It is my honour and pleasure to have the opportunity to work under their supervision. This work could not have been possible without their guidance, bright insights, and comments. I have learnt a lot from them and would like to express my gratitude for their patience, motivation, enthusiasm and immense knowledge.

I would like to thank my lab-mates from Madiba Security Research Group, especially Tousif, to share their experience and be by my side through this journey.

Last but not least, I want to thank my family for their endless love, support, and encouragement. I am incredibly lucky to have them in my life. I owe a special thanks to my supportive friends Mohammadreza and Ali. I am so grateful for their continues support and kindness.

I also dedicate this thesis to my mother, father, and my lovely sister. You are the reason I accomplish this today.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Password-based online services are ubiquitous, despite many known security and usability limitations of passwords [13, 31]. Password managers can alleviate some of these drawbacks by removing the need for people to memorize a large number of strong passwords, helping users cope with different password policies, creating unique passwords and providing protection features against some attacks (e.g., phishing). However, current adoption of password managers is low, and even among password manager users, many still do not use them effectively [51].

Previous studies of password managers [3, 5] show that usability issues can contribute to low adoption. Some other works [4, 52] attempted to improve the adoption of password managers, and some [17, 23] described many issues such as those caused by underestimating the risk of insecure behaviors and the user's mistaken or incomplete mental model

from the password manager. While several studies [8, 44] have investigated different ways to improve the usability of password managers, the basic wallet structure of the password manager has remained the same. In all of them, a password manager is regarded as client-side software that helps users store and fill passwords on a targeted website without the website recognizing whether it is the user who is filling the password field or a software program.

Considering the previous studies in addition to security and usability issues according to the current design of password management systems, in this work, we investigate how the design of password managers can be re-thought to facilitate adoption and minimize usability problems. In particular, we propose integrating websites with the password manager by utilizing an API as a novel method for completing account management functions through software. Through this method, we not only increase the security and mitigate some known attacks against them [14] but we also improve usability by minimizing the tasks given to the user. The idea of integrating websites with the password manager can also address adoption problems that hinder people from using password managers to improve their accounts' security.

Our idea is not to create yet another password manager. We want this new design, to act as a vehicle for investigating more fundamental questions about account management. Password managers, while seeming to be one of the most accessible solutions to the password problem, have not been widely adopted. This suggests that a new approach to password management is needed: users need to be empowered with tools to effectively access and manage the security of their accounts by relieving them of tasks more easily

completed by a computer.

In this work, to examine our idea by designing and developing ByPass, a new password manager that sits between the user and the website, reducing friction resulting from usability problems. In ByPass the idea is to ask numerous websites to provide their account management functions' APIs. ByPass uses an API for direct communication between the website and password manager, which allows the password manager to not only directly send credentials to the website, but also to query the website for information such as the password policy. In addition, this communication channel allows the introduction of innovative features, such as automated password changes, and account creation/deletion through the password manager. The primary goal of ByPass is to provide a more usable password management system that encourages users to behave securely. ByPass integrates nudges for secure behaviour, and is designed to make tasks such as account migration and password changes as simple as possible. As such, we design ByPass to have a user-friendly interface in order to work better for most people from different walks of life and encourage them to use our proposed password manager for completing account management functions.

In this work, we focused on ByPass's impact on end users. We evaluated the usability of our prototype implementation of ByPass by conducting a study with 20 participants. We found that participants were able to quickly and easily add new accounts to ByPass, migrate existing accounts into the manager, and change their passwords. Users were generally positive about the features in ByPass, but expressed concerns about the ways in which ByPass

moves the locus of control—the degree of peoples' believe on control of some outcome—proposed by Rotter [56]. It is worth noting that we evaluated account creation, account migration, password change, and account deletion of ByPass through the user study. After analyzing the results, we decided to provide a server-side for ByPass in order to address the mobility issue and support the password sharing feature. ByPass code is available on GitHub.[1]

1.2 Thesis Statement

Our research aims to create a secure and user-friendly environment that allows users to manage their online accounts by using a password manager. The development of a usable but secure system is the primary goal of this research. The following research questions are investigated as part of this goal:

Question 1. What are the key usability and security flaws in password managers? Could these flaws be caused by using password managers as a separate tool independent of the web service providers?

Question 2. Can introducing APIs and binding the password managers and web services together improve the password manager's adoption rate and encourage the users to use password managers more efficiently?

Question 3. Does allowing password managers to perform the third-party account management tasks improve usability? How about security?

[1] https://github.com/TinaSafaie/ByPass

1.3 Contributions

Our contributions can be summarized as follows:

1. We propose and develop ByPass, a password manager considering usability and user experience along with security enhancement. Even with strong evidence in favour of password managers and proven security advantages of using them, password managers are not accepted by the majority of the users as they suffer greatly in terms of usability. We investigate existing password managers to identify key usability and security flaws and developed, ByPass, yet not another password manager that solves these issues.

2. We propose the inclusion of APIs and direct end-to-end communication between the password managers and websites as a core component in the realm of password managers. Many of the usability and security flaws of password managers are introduced because of using them as a separate software for managing passwords and inclusion of API makes the password managers a native component of the authentication process, streamline password managers to cut through resistance due to usability, and solves many security flows from the foundation level of a password manager.

3. We design a password manager supporting both online and offline password storage considering users' privacy and security. Considering users' trust in a trusted third party, portability to share passwords across multiple devices, and reliable availability, we design ByPass to be configurable to support both storing passwords online and offline depending on users' needs and use cases.

4. We conduct a cognitive walkthrough to identify usability improvements in ByPass from the feedback of expert users and domain experts. Our experiments shade insightful sights on common early design mistakes that appears while designing a password manager.

5. We conduct a thorough user study with 20 participants to investigate the gap between user experience and design expectation. Through our study we verify the improvement in user experience in ByPass and also identify critical design refinement from users' feedback.

6. We investigate major security flaws in modern password managers and performed a security evaluation of ByPass and other password managers. Our experiment shows ByPass is resilient to many of these security vulnerabilities from the root of its foundation.

ByPass design, security, and all the features in the offline mode like account creation, account migration, password change, and account deletion as well as the user study has been discussed and published in [69].

1.4 Thesis Organization

The rest of the thesis is organized as follows. In Chapter 2, we present an overview of password managers and a literature review of three different aspects of them; security, usability, and the new proposals. Afterwards, in Chapter 3, we introduce our detailed

design of ByPass and its features. Next in Chapter 4, we discuss the security of password managers in addition to the security of ByPass and the attack mitigation. In the following chapter, we evaluate ByPass and present our results in different categories. Finally, in Chapters 6 and 7, we present our discussion, recommendations and conclusion.

Chapter 2

Background and Related Work

A significant amount of work has been done for the progression of password managers, yet to this day, password managers are not seeing any light at the end of the tunnel. We investigated existing password managers, their usability aspects, security issues, and new proposals. In this chapter, we present the background associated with password managers and the related work literature. We categorize the related works into three different parts and discuss each of these parts below.

Password managers are client-side software tools primarily designed to help users with password-based authentication protected by a primary password. Almost all managers are designed based on a *wallet* model, where the central function is to manage a list of passwords, protected by a primary password. The security and usability of password managers have been studied extensively. Modern password managers range in sophistication from browser-based managers that save and fill users' passwords, to more complex standalone

programs (e.g., 1Password[1], and Lastpass[2]) covering various features such as customizable password generation, security audits, family password sharing, and travel mode which we are going to explain more below.

Generating secure, random passwords can be counted as one of the password manager's primary functions. In some password generators, the user can customize the generated password by choosing the password's length, the combination of characters, numbers, uppercase and lowercase letters so that the generated password is more recognizable for them.

Auto-filling is a feature that is offered by almost all the password managers. It means the password manager fills the credential boxes as soon as the page is loaded. Auto-submitting is another feature in which the password manager not only auto-fills the credentials but also, submits them.

Password sharing can be an essential feature while you are in a family or business setting when more than one person needs access to the same account. By offering this feature, password managers are reducing the security risks associated with sending passwords over messages or emails.

There are plenty of works on password managers covering different aspects of them. We summarized prior related works into three categorizes: Security of password managers, usability of password managers, and new proposals on development of password managers.

[1] https://www.1password.com/
[2] https://www.lastpass.com/

2.1 Security of Password Managers

By saving all passwords in one place, password managers create a central point of failure in a user's security ecosystem. Although various proposals have been made for password managers that mitigate this problem [25, 55, 62, 66], the majority of commercially-available password managers are vulnerable in this way. The prevalence of such attacks is unclear, but it is compounded by the likelihood that users will choose insecure and easily-guessed primary passwords [71].

Recently, Carr and Shahandashti [14] revisited various known attacks on password managers, and analyzed whether password managers are still vulnerable to those attacks. They also identified four new vulnerabilities: phishing attacks, clipboard vulnerability, PIN brute force vulnerability (for applications), and brute force via an extension (evaluated against five popular commercial password managers). Their results showed that all of the tested password managers were vulnerable to at least one of the attacks they considered.

Oesch and Ruoti [50] provided the first evaluation that examined the full password manager cycle covering; password generation, storage, and auto-fill. Their results demonstrate that although password managers have improved since a couple of years ago, some of them are still saving unencrypted metadata or vulnerable to clickjacking attacks. They recommend that researchers evaluate password managers' progress in eliminating current password managers' common vulnerabilities and improving usability.

Silver et al. [60] studied auto-fill password policies among different types of password managers. Although the auto-fill function can increase the usability of password managers,

poor implementation of this function can lead to exposure of users' credentials. They also described the *sweep* attack, which requires the attacker to have control over the user's WiFi (e.g., in a public hotspot run/controlled by the attacker); the attacker can extract the user's credentials by injecting malicious JavaScript to the website without the user noticing.

Li et al. [38] analyzed five third-party web-based password managers, revealing four security vulnerabilities—including classical web application vulnerabilities such as Cross Site Scripting (XSS) and Cross Site Request Forgery (CSRF). They also analyzed authorization and user interface vulnerabilities, and proposed guidelines to mitigate these identified issues. In addition to the Li et al. [38] work, Stock et al. [70] investigated XSS attacks on password managers too. Although their work focused on built-in browser password managers only, they highlighted the risk of password auto-filling feature, which is not only in the built-in browser password managers. They also suggested an approach to protect auto-saved passwords from the XSS attacks. In their solution, passwords should be filled in a placeholder and replaced right before submitting them into the login form.

Gray et al. [28] proposed a technique to determine the risk that local password managers can cause. To test it, they analyzed three password managers and the results revealed that unencrypted password data could be discovered in temporary files even after closing the application.

Gasti et al. [24] analyzed different database formats that password managers use to store data. Through designing two security models of active and passive attackers, they identified the risks that arise when the attacker has physical access to the device.

Karlsson [33] detected a weakness in the auto-fill function and URL parsing code of

the LastPass browser extension. The vulnerability was in treating the wrong part of the URL as a domain. Thus, an attacker could deceive the extension into providing the user's credentials for any saved websites by modifying a URL.

Zhao et al. [76] performed a security analysis on two cloud-based password managers; LastPass [36] and RoboForm [54]. They focused on three types of attacks: brute force, local decryption, and request monitoring attacks, and their results indicate several vulnerabilities that an insider/outsider attacker can induce in the two considered password managers.

2.2 Usability of Password Managers

Chiasson et al. [17] conducted an early usability evaluation of two types of password manager, and found that users' poor mental models of password managers caused them to make dangerous errors. Karole, Saxena and Christin [34] compared a third-party password manager to a phone-based manager and a USB manager. They found that users preferred the control implicit in having passwords stored on a local device.

There are a variety of factors leading to the low adoption of password managers. Aurigemma et al. [5] found that insufficient time for installation, lack of immediacy, and users' feeling that using a password manager needs an additional effort that they don't want to spend contributed to password manager non-adoption. Maclean and Ophoff [43] found that trust, habit, and performance expectancy are the three factors that lead to the use of password managers. Pearman et al. [51] interviewed 30 participants, including people

who choose not to use a password manager, about their password manager use (and non-use). They found that factors such as lack of awareness and poor understanding of how password managers work prevent people from adopting password managers. They found that that users could be divided into two categories: people for whom convenience is a priority, and people for whom security is a priority.

Seiler-Hwang et al. [58] analyzed the usability of smartphone password managers and suggested security, user interaction, and integration with external applications as three key areas for improvement. They mentioned that only knowing about the password manager's existence is not enough to motivate an individual to use it. Users need to be encouraged to install and try password managers. Alkaldi and Renaud [3] identified three phases to user adoption of password managers: searching, trialling, and deciding. They later explored how self-determination theory can be used to encourage users to adopt password managers, and found that recommender tools can cause users to at least search for and try password managers [4].

Lyastani et al. [42] studied whether using a password manager can influence password strength and re-use. They used a browser extension to collect data on different password entry methods. Participants who used technical support for password creation had stronger passwords. They concluded that password generators improve security, but are under-adopted, highlighting the need to rethink password manager workflows.

2.3 Proposals for New Password Managers

A number of studies have designed new tools to address existing problems with password management. McCarney et al. [44] created *Tapas*, a dual possession, theft-resistant password manager in which an adversary needs to gain access to both devices in order to read the saved credentials. In their design, both devices – a mobile phone and a desktop computer – must be available in order to use the password manager, and the user does not need to have any primary password. Barbosa et al. [8] designed *UniPass*, a password manager for visually impaired users. *UniPass* allowed users with visual impairment to access their accounts and passwords more effectively and to use public computers. Although not positioned directly as a password manager, Ruoti and Seamons [57] propose a system in which passwords are stored in the operating system, and verified (using zero-knowledge proofs) by an independent password checking service. The system minimizes the interaction between users and passwords, as well as the attack surface for passwords.

Stobert et al. [67] designed *Versipass*, a password manager that generates passwords using visual cues by showing an image to the user. They used the graphical password to generate new passwords in order to address the memorability issues and improve the password manager's usability.

Chapter 3

ByPass: Design and Implementation

ByPass is designed to address usability problems that prevent users from adopting password managers while maintaining good security properties. In this section, we discuss the design of ByPass, features supported and introduced by it, and our prototype implementation.

3.1 Design Overview and Goals

ByPass is designed to reduce friction in password managers resulting from usability problems. Our insight is that many of the usability problems with password managers result from the password manager functioning as an intermediary between the user and the regular login page. Copy/paste errors, wrong fields are filled by the auto form fillers, leakage of passwords from the clipboard while copy pasting passwords, all result from placing the password manager as an external resource, accessible only by the user and not the website. Traditional password managers offer little conceptual advantage beyond a list of written

passwords, and in doing so, they miss an opportunity to genuinely address the usability problems with passwords and add value beyond other password storage mechanisms.

Chiasson et al. [18] offered four separated guidelines on the design of user interfaces. The design of functional interfaces is a matter of security because in security management systems, usability problems can lead to vulnerabilities in security. The authors proposed design areas can be summed up as: usable security, ecological interface design, social navigation, and persuasive technology. We followed their usable security strategy, which primarily indicate that the user should be aware of the security tasks they have to perform, be able to identify whether or not they have accomplished a particular task, avoid making dangerous errors, continue to use and communicate with the system, say whether the task has been accomplished and finally be able to define the current system status.

Below we explain how our design of ByPass follows and covers Chiasson et al. [18] proposed usable security strategies:

1. The user needs to be aware of the security tasks she has to perform: According to the Chiasson et al. [18] guidelines, selecting a secure primary password is the very first and vital task. We express the significance of the primary password by placing a message on top of the primary password field on the ByPass registration page. We use `zxcvbn` [73], which is a password strength estimator used in known sites such as Google and Dropbox [22], to guide the users to select a strong primary password. We measure the strength of the entered password and give a feedback to the user in order to help her pick a strong password. To make sure that the user is aware of the importance of primary password and chooses a secure one, we put a restriction on the selected password, so the user can only create an

account if she chooses a strong password according to the `zxcvbn` measurement.

2. The user should be able to perform the task successfully: We put a wizard on the very first page of ByPass to help the user understands the steps she will be completing afterwards, before going through them.

3. The user should be comfortable with the interface to keep working: By hovering the mouse on various buttons and fields, ByPass gives more details to make them more understandable. We also used numerous icons to make our password management system more user friendly and understandable.

4. The users should be prevented from making dangerous errors: We put an extra step of confirmation for functions such as account deletion. This additional move was to ensure the user is aware of what she is about to do.

5. The user should be able to tell if the task has been completed: A pop-up message shows up after each task completion to inform her that the task has been completed.

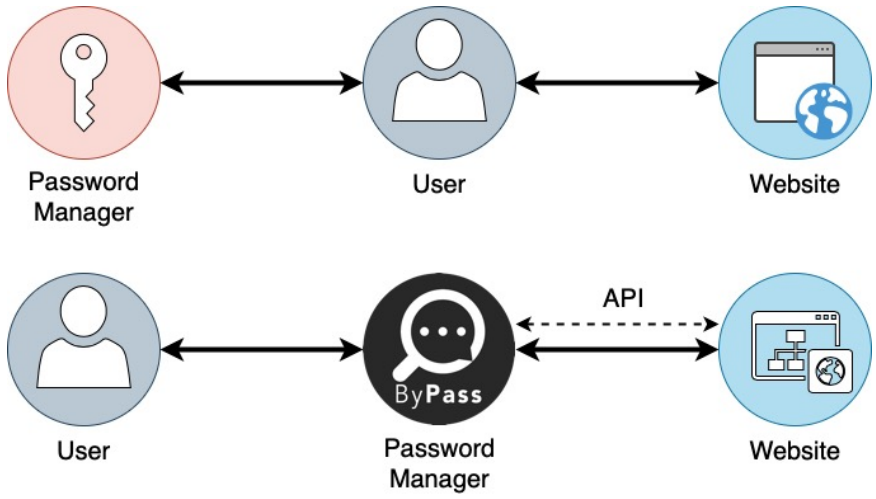


Figure 1: Current password managers architecture vs ByPass architecture

As shown in Figure 1, ByPass communicates directly with websites, thus avoiding

errors resulting from user manipulation of passwords and forms. It provides an API for websites to use so that it can pass credentials directly to these websites. This allows users to skip the manual login procedure when using ByPass, improving usability and avoiding errors. Our goal is to design a mechanism to reduce the number of clicks and actions required to complete account/login management tasks.

ByPass is designed to nudge users toward choosing secure options. For example, password generation features in password managers remain mostly unused [42, 51, 68]. ByPass creates secure randomly-generated passwords by default, and only allows users to select their own passwords by clicking through multiple steps. These generated passwords are secure and unique, and because users usually do not need to interact with these passwords directly, they are encouraged to take advantage of this functionality.

The usability advantages of ByPass are only realized if the website chooses to implement the ByPass API. This is a strong requirement, and we did not want to create a tool that was only usable with affiliated websites. Thus, we designed ByPass to extend the functionality of password managers without taking anything away. ByPass can be used as a repository for copying and pasting passwords in the same way that existing managers can, but adds functionality for websites that implement the needed APIs. Note that when we discuss usability and security of ByPass, we focus on the new design, not this backward-compatibility feature which shares the usual drawbacks of traditional password managers.

Moving the password manager to sit between the user and the website also creates an opportunity to consider other functions that password managers typically exclude, including account creation on third-party websites, account deletion, and automated password

changes. The security of ByPass is discussed in Chapter 4, but user authentication to the manager is handled exactly as it is by traditional password managers, i.e., with a username and primary password.

3.2 ByPass Features

The key features of ByPass include third-party website account creation, account migration, direct account logins, password changes, account deletion, and password sharing; more features can also be easily accommodated. Figure 2 shows a flow diagram of user interactions when setting ByPass up for a new account.

For the account creation, the user needs to enter a valid email address as well as a strong primary password.

To assess the strength of the user's chosen primary password and to help them improve it, we have integrated a password strength meter in this primary password selection field. Figure 3 shows how the password strength meter function gives real-time feedback on each of the entered characters in the password field. There is the confirm primary password page followed by the account registration page to make sure the user remembers their chosen password on the previous page.

Adding Accounts. ByPass provides two options for users when adding new accounts: to register a new account, or to enroll an existing one; both these tasks are abstracted into the same function: "Adding an account". We used a wizard to guide users through the process of choosing what kind of accounts they are adding to ByPass, and to ensure that they enter

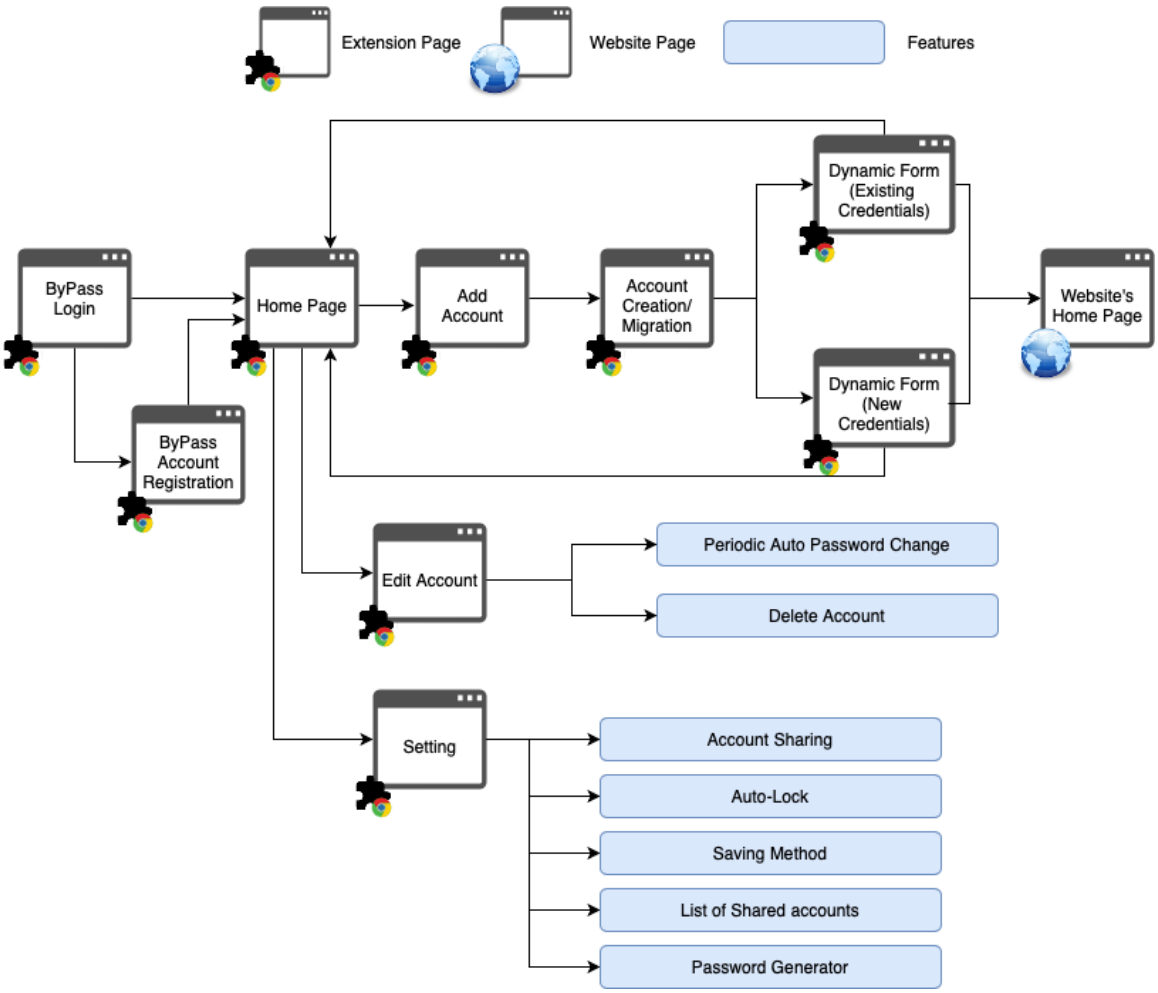


Figure 2: User flow diagram of ByPass.

the correct information (see Figure 4).

Because ByPass communicates directly with websites, all user interactions with accounts can be moved to ByPass, including account registration. When users want to create an account on a new website, they enter the website's name in ByPass, and ByPass queries the website for the registration fields they want. Users fill these fields, and ByPass automatically populates the password field with a random password conforming to the password policy (also sent by the website). Users can view and regenerate the random password, but must go through an additional confirmation to do so.

Figure 3: ByPass registration page

To add an existing account, after searching for the website name in ByPass, users simply enter their username and password, similar to standard password managers. If the user enters the name of a website that does not have the ByPass API installed, the password manager will act like a regular password manager, and store credentials for the user to manually enter.

ByPass displays a wizard page to show the user an overview of the following steps. The first thing is to enter the intended website's address. The website's address should be entered accurately for the websites having more than one login page on different sub-domains.

In the second step, as shown in the the user flow diagram depicted in Figure 2, the user should choose either to login to the website or create a new account. If the user already have an account on that website and they are only willing to migrate their account to ByPass, they should choose the login option. But if they want to create a new account

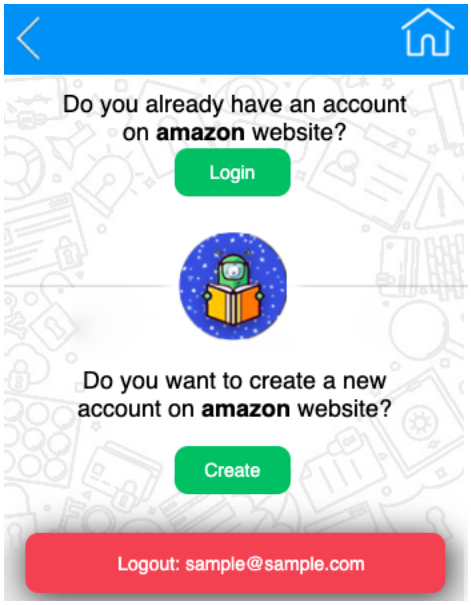


Figure 4: Login or create an account page

on the website, the create account option should be selected.

How the next page is going to look depends on which function and which website did the user choose in the previous pages. Meaning that, if the user enters the address of a website—e.g., amazon.ca—and chooses account creation, the next page will display all the required credentials for creating an account on that specific website. This page is dynamic, and each time by considering the website's name and create or login option, shows a page containing required credentials of that specific website for completing a function. Refer to the account creation function for the online store as an example in Figure 5.

Password Generation Policies. Having a good password policy is strongly recommended. Nowadays, most service providers have their own password policies—e.g., minimum eight characters, uppercase and lower case letters, use of special characters, etc.—for accepting passwords. Hence, these services may not straightforwardly accept randomly generated

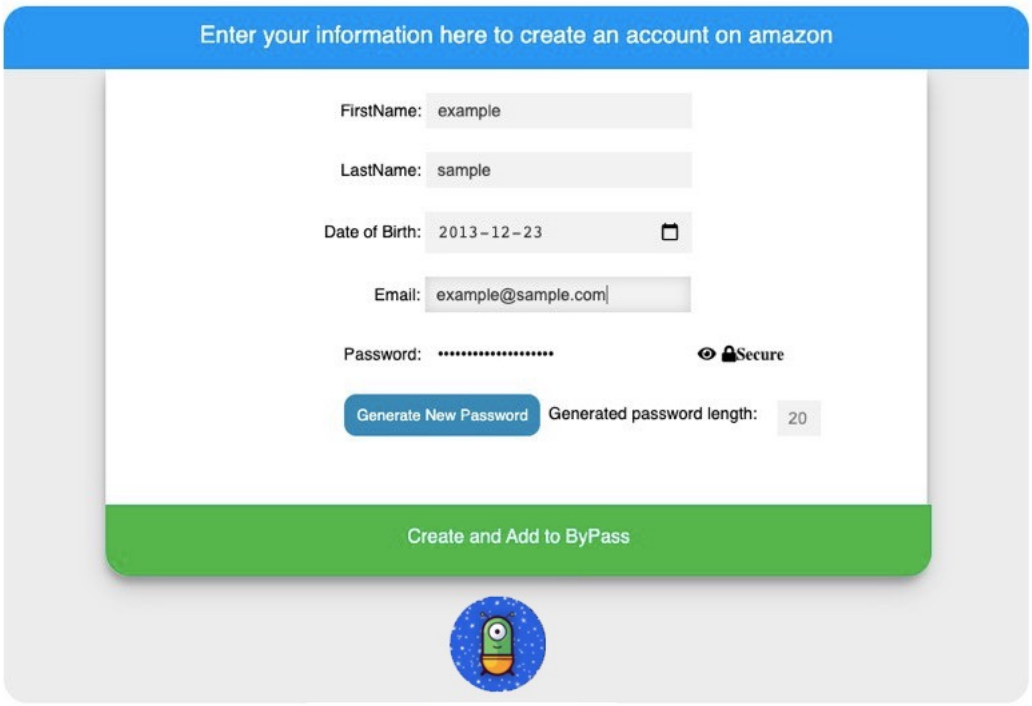


Figure 5: Account creation on a third-party website through ByPass

passwords. We have designed ByPass to accommodate the service provider-specific password policies. We allow the service providers to provide APIs to let ByPass know about their password policies, and based on that ByPass generates passwords that will satisfy those policies. As a result of using these APIs, in the account creation page, we automatically fill the password fields considering that website's password policies. Yet, the user is still able to change the password by using the generate password button or change the length of the password.

For the account migration, the user should select the login option (shown in Figure 4). After entering the username and the password, they can choose either to log in, which they will be taken on the home page of the website, or only to save the credentials and go back to the ByPass home page.

Account Login. After adding an account to ByPass, the user can log into the website by opening ByPass to the account page, and clicking on the "Go" button. ByPass sends the user's credentials to the website via secure API calls, and opens the site's main page in a browser, this is to avoid any direct interaction with login pages, e.g., navigating to intended URLs, and typing usernames and passwords.

Periodic Auto Password Change. Password changes are frequently avoided by users due to serious usability drawbacks [32, 68]. While password expiration policies are no longer recommended by NIST [48], they are frequently mandated by organizations. Password changes are particularly onerous to users of password managers, who typically must open the manager to copy the old password, navigate to the password change menu, generate a new password in the manager, ensure it is saved in the password manager account entry, and copied into the website. Because passwords are generally masked to avoid shoulder surfing attacks, it can be difficult to know which string is copied into which field, inviting errors. In ByPass, the direct communication between manager and website elides all of these steps, and makes password changes into a one step process for the user. On the edit account page, users can view their password, change it with the (re)generate password function, and set a timer for automatic password changes (see Figure 6).

Account Deletion. Although not typically considered as part of password management, another opportunity afforded by the ByPass architecture is to easily delete website accounts from the password manager. Account deletion is frequently made difficult by website designers [29], but privacy rights (such as the right to be forgotten [6]) indicate that account deletion should be an available straightforward process. In ByPass, account deletion is

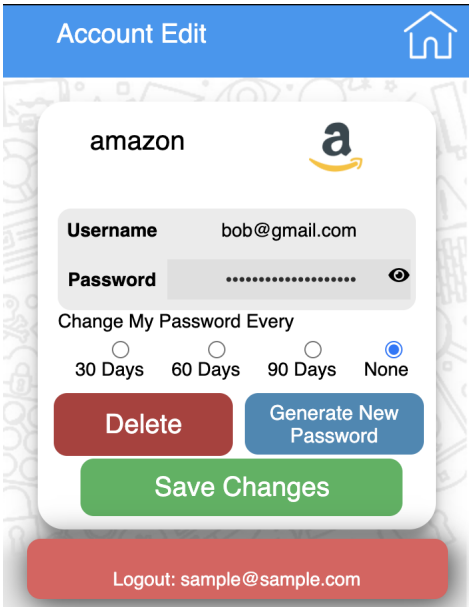


Figure 6: ByPass edit account page

accessed through the "Edit account" page, and includes a two-step confirmation process.

Password Generator. In the initial phase of ByPass, we focused on keeping the users away from manually typing in passwords. However, our experiments, interaction with the users, and the user study results showed us that users lacked the confidence of not having control over the generated passwords. Hence, we included a password generator that allows the users to see, put conditions, and gives the option to agree with the password that is generated. Therefore, in ByPass, we do not take anything away from what the existing password managers have and decide to add this feature (Figure 7).

Our password generator allows the user to configure and generate customized passwords by allowing to choose the length of the password or add uppercase/lowercase letters, numbers, or symbols while generating passwords.

Saving Method. ByPass allows the user to select a data storage system. The saving option

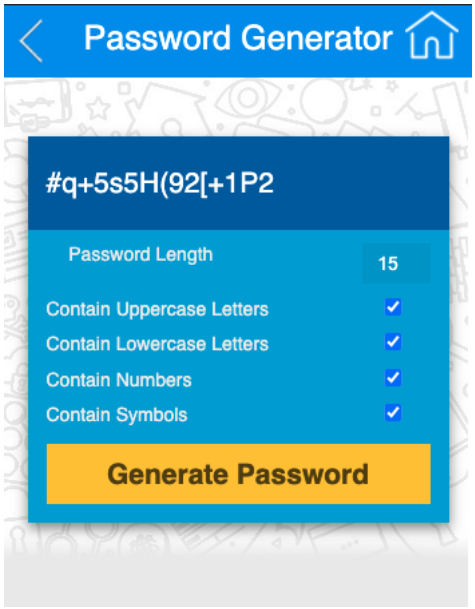


Figure 7: ByPass password generator function

can be either online (on the cloud) or offline on the user's device. We store all the data in the cloud by default unless the user decides otherwise. We remove all their data from the cloud as soon as they decide to have a local database.

Account Sharing. Account sharing is a new feature in password managers that can be divided into two categories; (1) accounts that the user has shared with someone and (2) accounts that are shared with a user.

ByPass allows two users to share their accounts with each other for a specific period of time. Both users can have access and manage their shared accounts, only by visiting the "List of shared accounts" page.

Let's assume that Bob has an account on the Amazon.ca website and decides to share his account with Alice for only 12 days. To do so, the only thing he should do is to go the account sharing page (See Figure 8) on ByPass, choose the name of account he wants to

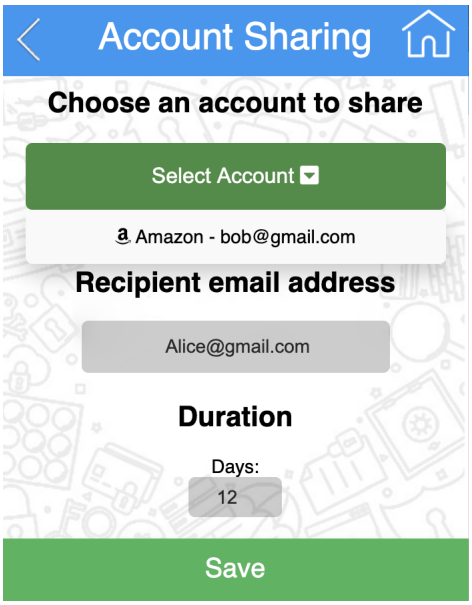


Figure 8: Sharing an account page

share, enter Alice's email address with which she has created her ByPass account, and in the end choose the duration of sharing (Here is 12 days).

As soon as Bob shares his account, Alice will receive an email from ByPass telling her a new account has been shared with her and she can see it in the "List of shared accounts" page. As shown in Figure 9, Alice can see which accounts has been shared with her, from whom, and for how long. By clicking on the "Go" button, Alice would be redirected to the home page of the Amazon.ca website. The "Days to expire" on the figure shows Alice how long she has access to Bob's account and if she chooses, she can stop using his account by clicking on the "Stop" button. In addition, ByPass also allows Bob to stop shearing his account with Alice any time before the expire date.

In terms of technical details, when Bob chooses the account he is willing to share (here we assume it's the Amazon website) and enters Alice's email address, the extension communicates with the ByPass server and sends Bob's username (the one that he created

the ByPass account with), Alice's username, the account's ID, and the sharing duration. It is worth noting that each account that the user adds to ByPass has a unique ID in the database table. When the ByPass server receives the request, it checks if the username and account ID are correct. If yes, it saves the record and gives Alice an email informing her that an account has been shared with her.

When Alice decides to log in to Bob's account, the extension first sends a request containing her username, Bob's username, and the account's ID (same as above) with a login request to the ByPass server. After checking if Alice has the right to access Bob's account or not, the server sends an API to Amazon containing Bob's credentials and a flag (called "shared"), which is set to true. For better security, we set this "shared" flag in the API to let the website know that this user is not the account holder and should not be able to complete the account management functions. Some actions, like changing the password/email address, should not be modified by anyone rather than the account holder.

Upon receiving the share request from ByPass, the website first checks the validity of credentials and then saves the state (that the account is shared). Afterwards, it generates a token which is valid only once and sends it back to the ByPass's server. The server sends the token to the extension, and finally, Alice logs in to Bob's account while the account setting feature is disabled for her.

Furthermore, Bob can revoke sharing his account with Alice whenever he decides. On the other hand, when Alice wants to get access to a shared account, ByPass confirms with the server-side that her access permission has not expired and she has a valid access. As soon as Bob clicks on the stop sharing button, sharing value of the Bob's account with

Figure 9: List of the shared accounts

Alice will be set to false. Thus, Alice will be prevented from accessing to Bob's account if she tries, because when the server receives a request from Alice, the server will check if the account sharing value for Bob's account is true or not. In the case where Bob has revoked sharing his account, the server will return an error to Alice. Although if Alice is already logged in to the website and Bob decides to revoke sharing at that time, the website wouldn't kick Alice out of the page.

If the user chooses to have an offline database, the password sharing feature will be disabled as for providing this feature, a trusted server is required.

Auto-Lock. This feature allows the users to choose a period to auto-lock the password manager in a case inactivity. This feature providers protection against scenarios where the user leaves their computer unattended and the password manager unlocked. The user is given the following options to choose from: 30 minutes, 1 hour, 1 day or never.

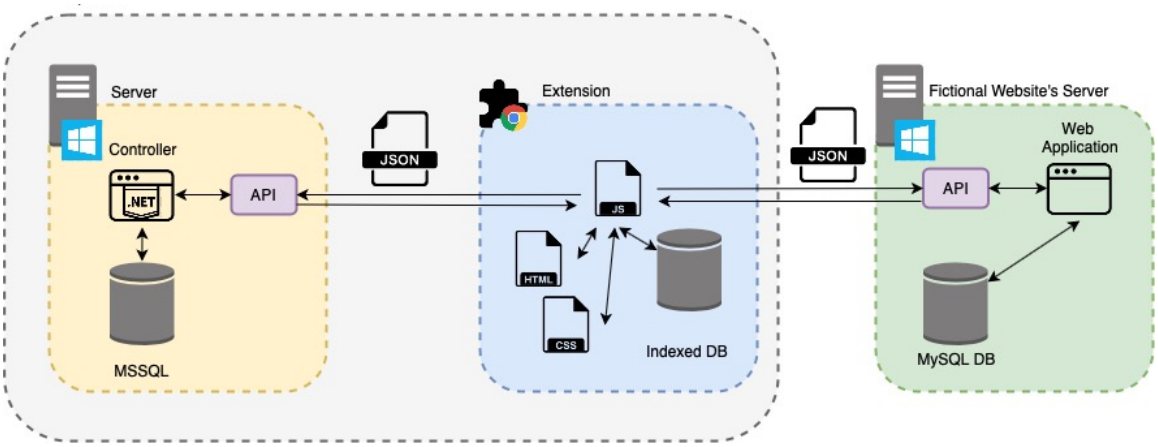


Figure 10: ByPass API interaction between server side, user browser, and fictional website.

3.3 Implementation Details

The ByPass implementation has two components: the password manager itself which is implemented as a Chrome browser extension and the server-side. We also developed two fictional websites to test and evaluate ByPass. In what follows, we provide more details on each of these components. Figure 10 shows the ByPass API interaction between the ByPass's server, browser extension, and the fictional website's back-end.

Extension. Chrome extensions [26] are pluggable programs for chrome browser built on top of the web technologies such as HTML, CSS, and JS. For navigation and data manipulation in them, we use JavaScript (JS). Each browser has certain storage methods such as cookie, local storage, session storage, web SQL, and IndexedDB which can be used for different purposes. We use IndexedDB [35] to store data, which is available for all the modern web browsers, including Chrome 4+ and Firefox 10+ [61]. This has the potential to save a large amount of data, so we can provide high-performance application search using API.

Web framework can provide us with the ability to automate some overhead related to

typical web development activities, such as accessing databases and managing sessions. However, as we want our password manager to be a stand-alone chrome extension without needing any desktop application, instead of using a web framework, we used pure JS, jQuery, and HTML to take care of the above mentioned features.

We use three main files: Manifest.json, HTML, CSS, and JS files. In Manifest.json, we add all the configuration and permissions of the chrome extension that needed to be satisfied by the browser.

Server-side. To bring mobility to ByPass, we developed the server-side to allow all data to be synchronized. The server is hosted on the Microsoft Azure cloud service, and we used `.Net Core`, a Microsoft-developed open-source platform, to make this project more accessible to research community and open-source development.

The core component of the server is the database which is secured by Azure defender and is responsible for vulnerability assessment and advance threat protection. We use transparent data encryption (TDE) which designs a secure system and encrypts the assets in addition to building a firewall around the database server. ByPass server consists of a controller which is developed by `Microsoft ASP .Net core.cors (2.2.0)` and in order to connect the controller to the database through the queries, we use `Microsoft Entity Framework core (3.1.8).`

The other framework is a JSON framework called `Newtonsoft.json(12.0.3)` capable of serialize and deserializing any .NET objects as well as sending and receiving JSON APIs between the server-side controller and the extension.

We decided to create a .NET core server-side that gives ByPass the ability to be dockerized [12] in case if any company decides to have an independent password manager, as ByPass is open source and companies can develop their own version of ByPass on their servers. The other benefits of using the .NET core are that it is open-source, and capable of running on any operating system.

API Handling. We use a REST API for communicating with website back-end as it improves the performance, has a simple interface, allows independent modification of the components, and requires low bandwidth [21]. API calls are made via HTTPS using JSON. ByPass enables various account management functions, and currently implements API calls for creating an account, logging into an account, sharing an account, editing, and deleting an account. Our prototype API is written in C# using the .NET framework and `System.Web.Http` library. The way we call those APIs is by creating an instance of `XMLHttpRequest()` and then opening a POST request between the extension and the website to send data using the exact URL for the given website's API. To have an asynchronous communication, we set a value to true, which indicates the asynchronous request. We create an asynchronous `XMLHttpRequest (XHR)` and let the browser continue working normally while the request is being handled [46].

For `Request.SetRequestHeader` in the `XMLHttpRequest` instance, we set some variables such as: content-type, authorization, and access-control-allow-origin values. These values might differ according to each website's configuration. In our case, we set content type for the webmail service provider to JSON and the online store to `x-www-form-urlencoded`. HTTP Authorization request header is a credential given from the

server to a user agent for the authentication. After that, in a request.send we send the user's credentials as well as our intended function value, which is a pre-agreed value between the server and the user agent. Now, the server can authenticate the user, verify and execute the request.

Table 1 lists the API endpoints that ByPass needs from websites to support its various features. Below we explain more about each of the API calls.

Table 1: The API calls that ByPass needs make to support features

Name	**API Endpoint**	**HTTP Method**
Login	/Login	POST
Account Registration	/Register	POST
Change Password	/ChangePassword	POST
Account Deletion	/DeleteAccount	DELETE
Account Sharing *	/SharedAccount	POST

(*) = This API call is only available on the online mode of ByPass

Login. This API includes the username and password of the user's account, which ByPass sends to the intended website.

Account Registration. We make this API call while creating a new account on a third-party website through ByPass. This API includes all the required information for creating a new account on a specific website.

Change Password. Changing the password of an existing third-party account is possible through ByPass when a website provides the ChangePassword API endpoint. Password change API contains the user's old password, the new password and username.

Delete Account. The websites supporting account deletion function can provide ByPass the DelecteAccount API endpoint to allow ByPass to delete accounts.

Shared Account. This API call can be made only by the ByPass server-side, not the

extension. SharedAccount API is the same as the login API call with the "shared" flag set to True.

Fictional Websites. We created two fictional websites as prototypes to communicate with ByPass and to handle user requests and communicate with the back-end database. One of these sites resembles an online store website (such as Amazon.ca) and the other one resembles an email service provider (Webmail). We refer to them as the e-commerce website and the webmail, respectively. We chose these two types of websites as most people interact with them more than the other websites.

In these websites, we used JSON Web Tokens [30] for authentication purposes. JWT [30] is a standard for authentication mechanism with a standard signature algorithm. The token contain three parts: header, body, and cryptographic signature. The header consists of a token and signing algorithm, the body contains the claim, which is the value that we want to secure, and the cryptographic signature is formed using the header and body. All of our data transmissions use the Transport Layer Security (TLS) protocol and they were set up through CloudFlare Keyless SSL [19, 65].

It should be noted that the implementation details described here are specific to our prototype, but ByPass is designed to work with a wide variety of setups. The API is packaged as a library that can be included by web developers, and the manager is database-agnostic.

Chapter 4

Security and Attack Mitigation

All the password managers encompass security-critical information, hence they need to use strong cartographic protocols and standards. ByPass is composed of different components– e.g., user-facing Chrome extension, the server-side, and API endpoints. Thus, we put our focus on security all of those components.

In this section, we discuss the security of ByPass in terms of its databases (both online and offline), compare it to some other commercial password managers, and discuss measures ingrained within ByPass to mitigate this vulnerability.

4.1 Password Manager Security Evaluation

Many tools, studies, and reports have exposed vulnerabilities of password managers. We selected some of the well-known attacks on password managers and performed those on the latest versions of the password managers available during our experiment. Our experiments

show that many of these vulnerabilities (including ones that were exposed more than 5 years ago) still exist on password managers' latest versions. Some of the vulnerabilities exist because of the basic design principle of current password managers. Hence, they cannot be fixed with a quick patch or software updates. On the other hand, the inclusion of APIs in our design makes ByPass inherently resilient to these attacks. In this section, we are going to delineate the details and results of these experiments.

In terms of security concerns of password managers, the study by Li et al. [38] and Silver et al. [60] has a significant impact on the security community. We followed their methodology, and picked some of the commonly disclosed vulnerabilities in password managers by reviewing the literature and testing some password managers against them. The results indicate that auto-fill and auto-submit features have been the researcher's interest for a long time. Below we explain each of the attacks that are mostly from Li et al. [38] and Silver et al. [60] works.

User Interface Vulnerability. This vulnerability was mentioned by Li et al. [38] for the web-based password managers. When the password manager itself is vulnerable to the phishing attack, it can reveal all the user's passwords. We used the LogMeOnce [41] password manager to test the user interface vulnerability. We assume that the attacker has control over the website, and that the user has stored their credentials on the password manager.

When the user visits that website's login page, the password manager pops up an iFrame for authentication to auto-fill the credentials. There is a button in an iFrame to redirect them to the password manager's login page. At this point, the attacker runs a script to delete the

actual iFrame and instead create a fake iFrame that looks precisely the same as the real one. Then the user clicks on the button and goes to the fake password manager's login page controlled by the attacker, and enters their username and primary password. Throughout this attack, an adversary can get the user's primary password and access all the stored passwords. This attack is applicable for all password managers vulnerable to phishing attacks and the one using an iFrame.

Interface Vulnerability attack is ineffective against ByPass. As ByPass uses APIs to authenticate users and when users try to login to some site they directly log in from ByPass without going into any other site, the option of embedding an iFrame is not available for an attacker. Hence, ByPass is resilient against interface vulnerability attack.

VaultBreaker. VaultBreaker [63] is a tool designed for attacking password managers using three attacking techniques: Proxying, Memory Parsing, and Clipboard Event Hooking. At the time of testing, VaultBreaker could only penetrate 1Password [1], BitWarden [10], and DashLane [20]. We could only use this tool's clipboard event hooking function as the two other functions did not seem to be functional. We tried this against all three password managers mentioned above. 1Password and Bitwarden were not vulnerable to the attack anymore, but in the case of Dashlane, we could get the username and primary password as they were copied to the clipboard.

ByPass authenticates users by using APIs, so the users never need to copy their passwords in the clipboard; hence the clipboard watcher of VaultBreaker fails to steal passwords from the clipboard. Furthermore, ByPass is not vulnerable to memory parsing and uses secure encryption (which makes ByPass resilient to exposing in-memory passwords) making

ByPass resilient to VaultBreaker.

Sweep Attacks. This attack was developed by Silver et al. [60] with a focus on the auto-fill and auto-submit features of the password managers. They introduced three different types of this attack; iFrame sweep attack, Window sweep attack, and Redirect sweep attack, assuming that the attacker has control over the Wi-Fi.

We launched the iFrame sweep attack against Limitlesslane [39] password manager. In this attack, the user is trying to connect to a public Wi-Fi controlled by an adversary. In order to connect, there is a terms and conditions web page to which the user should agree. While there is only the terms and conditions visible on that page, multiple invisible iFrames point to the various website's login pages. We chose Limitlesslane [39] password manager as it supports auto-submission too. Here, we first saved a website's credentials on the password manager. Then we put an invisible iFrame on the hotspot web page. As soon as the terms and conditions page was loaded, the password manager filled the credentials in the hidden iFrame and submitted it. As a result, an adversary could get access to the user's account. It is worth mentioning that supporting the auto-fill feature would be enough to launch this attack since the attacker can write a script to submit the credentials. Filling the credentials in an iFrame is a dangerous function that should not be supported by password managers anymore.

ByPass is resilient to User Sweep Attacks. ByPass never auto-fills the inputs or auto-submits the login page. It uses APIs and communicates directly with the service providers to authenticate the users and forwards the users directly to the landing page. Hence, these types of attacks become ineffective against ByPass.

4.2 ByPass Database Security

In this section, we discuss the security of ByPass database, both in the online and offline mode, and then explain some attacks that can be mitigated through our design.

Security of the Offline Database. As a result of using an offline database, it can be subject to offline attacks (e.g., via guessing the primary password). We designed two security components for securing our offline database. The first component of our design is to authenticate the primary password and generate the encryption key. We use 256-bit Advanced Encryption Standard (AES) which is a symmetric block cipher encryption and encrypt our password database offline. The key of AES-256 is derived via PBKDF2 [75] from a user-chosen primary password, a random salt, and 100,000 iterations. We choose the salt from the user's mouse movement using onmousemove event handler [64] while initializing ByPass. We store the SHA256 value of this key, along with the salt, in the encrypted database. The key is recreated from the user-supplied primary password and the user is authenticated by matching the hash of the re-generated key with the stored value. Existing measures (see e.g., [15, 16, 59]) can be adopted to enhance resistance against offline attacks.

The second component of our design is to secure the credentials for different sites stored in the offline database. After stabilizing a session, ByPass can access the 256-bit encryption key which is derived from the user's primary password. When storing any credentials for any website, ByPass encrypts the information using the encryption key and stores the encrypted data. Whenever ByPass needs to access any of the credentials, it gets the encrypted data of that website and decrypts the data as long as the user's session is

active. As soon as the user's session terminates, ByPass can no longer access the encryption key as session termination is equal to missing the primary password. Meaning that the decryption key is not available anymore.

Security of the Online Database. By default, our current ByPass implementation stores account information both locally (offline) and on the server (online) unless the user decides to use only the offline database. In that case, ByPass server deletes all the user's accounts and disables some features like account sharing and cross-device supporting. In terms of security, we use Microsoft Azure defender [45] to secure our online database. In addition to the Azure defender [45], we encrypt the password field of the database with AES-256 using PBKDF2 [75] from the user's primary password (same as what we mentioned for the offline database).

4.3 Attacks Mitigation

In terms of reducing attacks, since we perform all the communications between the password manager and websites through TLS, remote network attacks, and SSL stripping attacks are mitigated; a remote attacker cannot intercept and/or modify our communications.

Since ByPass communicates directly with web servers (instead of filling forms on client-facing pages), encouraging users to use ByPass for completing account management functions can mitigate HTTP(S) auto-fill vulnerabilities and sub-domain equivalence attacks [11, 14]. HTTP(S) vulnerabilities happen when the password manager fills the credentials on an HTTP version of the website while the credentials were saved on the HTTPS

version, and in this way the password manager makes an opportunity for the attacker to extract the user's credentials from non-HTTPS pages of the website. Avoiding auto-fill also helps us avoid the *sweep* attacks [60], where credentials are filled to invisible fields in a malicious webpage, and exfiltrated using JavaScript.

Copy to clipboard is a feature that most password managers utilize if they cannot auto-fill the credentials. If the password manager doesn't support enough protection for the credentials that were copied to the clipboard, it will lead to clipboard vulnerabilities, e.g., exposing passwords to other sites/processes; see e.g., [14]. In ByPass, passwords and other information are communicated directly to the website, and no password copying/pasting is used for login or other account-related tasks, which avoids leaking passwords from the paste buffer.

User interface-based password brute-force attacks are another vulnerability that affect extension-based password managers [14] if attackers gain access to the password manager user interface. To reduce this vulnerability, we add a delay based on the specific number of wrong primary passwords entered by the user (similar to some leading commercial password managers).

Chapter 5

Usability Evaluation

ByPass is designed to address usability issues that prevent end-users from adopting and using password managers. We conducted two usability evaluations of ByPass: an inspection-based evaluation of an early prototype, and a lab-based user study of the higher fidelity prototype. It is worth mentioning that at the time of the user study, ByPass was at the first version, which did not have a server-side. By analyzing the user study results, we decided to add a server-side to make ByPass online, address the syncing problem, and support the password sharing feature.

5.1 Cognitive Walkthrough

After creating the first prototype implementation of ByPass, we wanted feedback on the usability of the manager. At that point, the prototype included the main functions (adding

accounts, logging in, and a version of password changing), but was not sufficiently functional for a user study. We chose to conduct a cognitive walkthrough [72] because of its focus on learnability and because it allowed us to include the perspective of novice users.

We conducted a pluralistic walkthrough with five evaluators, including the project team leads, the developer, and two volunteers playing the part of a novice ByPass user. One of these volunteers had longtime experience using a password manager, and the other had no password manager experience. We walked through the process of creating a new user account on ByPass, migrating an existing account, registering for a new third-party account, changing a password, and logging into a website.

In general, our novice participants found interacting with the prototype confusing, and were unsure what steps to first take, and what information to enter where. This led us to redesign much of the user interface, and include stronger markers of flow between steps, e.g., the account-adding wizard and more prominent navigation buttons.

Another issue that arose in the walkthrough was that the inexperienced participant seemed to have few mental models for common password tasks, and in particular, had no language for describing them. The abstraction taking place in the password manager was confusing to them, and they had multiple problems entering the right (fictitious) credentials, or choosing the correct menu option.

One of our central questions was whether the features in ByPass made sense to users – would using ByPass seem jarring, given that it departs from the usual interaction with websites? We expected the volunteer with previous password manager experience to question how ByPass worked, but they did not comment until prompted by us. When queried, they

spoke to the intuitiveness of the feature set, and said that they did not question how those features were working *because they just worked.*

5.2 User Study

The second phase of our usability evaluation was to conduct an in-lab user study to evaluate the usability of the ByPass prototype with unbiased users. Following the cognitive walk-through, the ByPass prototype was completely redeveloped into a higher fidelity prototype (described in Section 3.3).

In evaluating ByPass, one difficulty we encountered was creating a realistic testing scenario. Ecological validity, or the realism of the study situation, is of the utmost importance in security studies. Tasks that seem easy or manageable as primary tasks (such as remembering a password) are not always manageable when happening in the context of another more important task. Simulating this for a password manager is difficult – the gold standard evaluation would seem to be a field study where participants use the manager for their own accounts. We would want to collect instrumented data from such an evaluation, and privacy could be a significant concern (not only for data collection, but potentially biasing participants' behaviour). For ByPass, a study of this type carried the additional challenge of needing websites to be implemented with the API, further restricting our ability to have users test the manager with their own accounts.

The goal of our user study was to gain insight into how easily users learned to use ByPass, their reactions to the functions, and assess how using an API can help the user to deal

with password-protected websites. We did not include a control condition in this evaluation, because we did not feel that existing managers provided a meaningful comparison to the novel features offered in ByPass. Our goal was to evaluate the usability of ByPass (including the problems that arose). We chose an in-lab study so that we could observe participants' interactions with ByPass, and ask follow-up questions about their experience.

Each study session lasted between 30 minutes and one hour, and was divided into three parts. Participants first completed a pre-test questionnaire (see Appendix E) asking about existing password habits and demographics. After that, we provided them with a short introduction to password managers and a brief overview of ByPass (see Appendix B). They were asked to complete six tasks (see Appendix D) using ByPass, and then to complete the post-test questionnaire (see Appendix E). Participants were paid 15 CAD. The study was approved by the Concordia and Carleton institutions ethics board (see Appendix G).

It is worth mentioning that we conducted the user study offline and in-person so that we could primarily focus on how ByPass works and figure out the improvements that ByPass can offer in the second phase. However, from the user's feedback, we further enhanced ByPass by adding online support and account sharing features on the following phases of development. These two features address findings from our user study. We gave the participants six tasks to complete in the way that each task was responsible for covering a specific feature of ByPass. At the end of the study, we tested ByPass account creation, adding a third-party website's account to ByPass, account creation through ByPass, password change, and account deletion functions of ByPass.

We provided the participants a task description sheet that can be found in Appendix D,

including a short description of each task. As some tasks like creating a new account needed credentials like an email address, password, etc., we provided the required credentials unique for each participant into the task description sheet. We also reminded participants not to use their own passwords during the study. In below, we give an overview of each task.

Task 1: ByPass account registration. The very first interaction of the user and a password manager is the account registration. Creating an account on ByPass requires an email address and a primary password. Thus, to complete this task, we gave each participant a unique email address, and for the primary password, we asked them to choose something on their own but not their personal ones.

Task 2: E-commerce website account creation. We asked participants to create an account on the e-commerce website with the credentials given to them. By identifying this task, we did not attempt to test ByPass itself, but we aimed to show participants that the websites we use are genuine and functional.

Task 3: Account migration and log in. Task 3 was to add the newly created e-commerce website's account to ByPass and then log in. This task can be counted as the first primary function for all password managers. Users should be able to add their existing credentials for a specific website to the password manager. Without considering the API's capabilities, most of the password managers can fill the username and password fields for the user. In some cases, there is an automatic auto-fill feature in the password managers that fills the credentials' fields as soon as the login page loads. In some other cases, the password manager can not only fill the fields but also automatically hit the login button (auto-submit).

The user can see the login page in this case, but there is no need for additional interaction. However, in ByPass, we have the benefit of using API in the way that, after adding the credentials of a specific website to ByPass for the first time, by clicking on one button, the user will be taken to the home page of that website without even seeing the login page.

Task 4: Webmail service account creation through ByPass. For this task, we asked participants to use ByPass in order to create an account on the webmail website with the provided credentials. For us, asking the user to create a third-party account using a password manager instead of searching in the browser is critical, as this feature is new, and it might look strange to the user. Providing a wizard page and adding some hints in different pages of ByPass aimed to help user understand this feature better. We added this task to examine how participants would go through this new feature and how they would like it.

Task 5: Password change. This task was to change the e-commerce website's password using ByPass. Changing passwords is another important function for password managers, as they are identified as a tool to help the user choose a secure, random, and unguessable password. Our goal was to make sure that this feature works well in ByPass, and that participants feel confident to use. Most of the password managers assist users by providing a password generator function. But the user cannot change the password of a website through the password manager because it does not have a direct communication with the intended website to do it on behalf of the user. Only in Dashlane [20] there is a feature called password changer which allows the user to automatically change their password through the passwords section in the manager. The way that Dashlane works is that it provides certain number of websites, and password auto-change feature works only for those websites.

LastPass [36] had that feature earlier, too, but it seems they no longer support it. In ByPass, for the websites that we have direct communication with them through the API, the user can easily change their password by only one click.

Task 6: Account deletion. The last task was to delete the e-commerce website account using ByPass. This, also, is a new feature we can provide with API support. Finding a way to delete an account has always been difficult; with this feature, the user can delete her account as quickly as possible.

We designed the study to emulate a plausible first-time experience with ByPass. To improve the ecological validity of the study, we created two fictional websites (discussed in Chapter 3) to implement the API and be used for study tasks. These websites were open-source versions of a webmail platform and an e-commerce website, and we gave participants a handout to use in the study with the website URLs and credentials to use in the study. We reminded participants never to use their own passwords in the study.

Participants. We recruited 20 participants (12 female) by posting posters around (our university campus). Participants ranged in age from 18 to 46 with a median age of 24. 18 participants were students, and 2 university employees. 15 participants (75%) reported having previous experience with a password manager, though only three participants reported that a password manager was their primary means of saving a password. Table 2 demonstrates more details on the participants demographics.

Throughout the pre-test questionnaire (see Appendix E) we got some information on the participant's password security attitude and habit, their experience in using a password manager, and their password saving's method. Table 3 indicates the results of the pre-test

Table 2: Participants demographics

Number of Participants	
	20
Gender	
Male	8
Female	12
Age group	
18 - 20	6
20 - 30	10
30 - 40	3
40 - 50	1
Fields of study	
Computer	7
Mechanical Engineering	3
Civil Engineering	3
Physics	1
Arts	2
Math	2
Social Science	1
Finance	1
Education level	
Graduate	7
Undergraduate	5
College	2
High School	6
Computer Skill	
Novice	6
Neutral	12
Expert	2

questionnaire.

Table 3: Participants password-related habit

Password-related habit	Number of participants
Number of password protected accounts	
0 to 10	8
11 to 50	8
51 to 100	1
More than 100	3
Prior password manager use experience	
Yes	15
No	5
Password security concern	
Not concerned at all	1
Neutral	5
Very concerned	14
Password change interval	
Monthly	0
Every 6 months	2
Once a year	1
I only change my password if I have to	17
Password saving method	
Re-use the same password	9
Write passwords down on a paper	4
Write passwords on the file on computer	9
I use a password manager	3
The browser saves passwords for me	7
Remember in my head	9

Table 4: Descriptive statistics for task completion time in seconds.

Tasks	Mean	SD	Median	Minimum	Maximum	Range
New ByPass account	150.0	125.5	112.5	53.0	578.0	525.0
New web account	188.1	95.4	166.0	73.0	535.0	462.0
Account migration	60.8	25.3	50.5	33.0	128.0	95.0
Login via ByPass	4.4	0.8	4.2	3.2	6.7	3.5
New account via ByPass	161.4	104.5	120.5	52.0	502.0	450.0
Password change	73.1	65.8	43.5	14.0	218.0	204.0
Account deletion	19.6	13.1	13.0	7.0	50.0	43.0

5.3 Results

We structured our usability evaluation around the ISO 9241 definition of usability [9], and evaluated ByPass using three measures: efficiency, effectiveness, and satisfaction. We recorded task completion times, success rates, and errors via instrumented data collection in the prototype, and measured satisfaction using Likert scale questions on the post-test questionnaire. We evaluated efficiency using the time spent on each task. For effectiveness, we assessed the number and types of errors that occurred during the study, and for satisfaction, we considered findings from the Likert scale questions. We also recorded observations related to task completion, participants' comments, problems, and recommendations.

5.3.1 Time

Table 4 shows descriptive statistics for the duration of each study task, and Figure 11 shows the distributions of completion time for all tasks. Times were recorded in the manager logs from the appearance of the first screen to successful completion of the task, and include the time participants spent making errors.

Although there were outliers, the median completion times were less than three minutes

for all tasks. Keeping in mind that all participants were completely new to ByPass and were completing all tasks for the very first time, these results are encouraging. Variance was relatively low for most tasks, particularly account migration and login, which we think could form the majority of the users' tasks when setting up the manager for the first time.

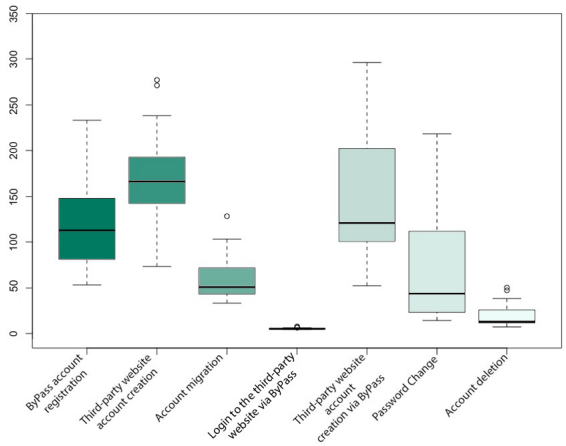


Figure 11: Boxplots showing the distributions of task completion time in seconds.

ByPass account registration involves the user entering their email address, and choosing a primary password. To enhance the ecological validity of the evaluation, we included this task in the study and used a strength meter to encourage participants to choose a strong primary password. As mentioned above, participants were warned not to use their real passwords. Ten participants had trouble picking a suitable primary password, and the median completion time was 113 seconds. As the ByPass registration process is no different than regular account creation, participants were not asked to recall this password during the study.

The second task had users register directly on one of our external websites, so that they would have an existing account to add to ByPass in a later step. This step also gave us a

baseline for the length of time needed to register on a "regular" website. This task had the highest median completion time at 166 seconds, and a few participants took much longer to complete it. It is also possible that network latency had an effect on the times for this task, but in any case, the studies were conducted in only two locations, so we would not expect significant differences in latency.

Later in the study, participants added that account to ByPass, and this process was considerably faster; the median time to add an account (through ByPass) was just 51 seconds. Following migration, participants were instructed to log into that account through ByPass and participants had no trouble doing this very quickly, in a median of just 4 seconds.

The process of creating a new website account via ByPass was new to all users, and the median completion time was 121 seconds. There was a large interquartile spread, and a few participants struggled considerably with this task, but we expect that times might decrease as users grew more familiar with the manager. In any case, an average of two minutes to register on a new website seems to be an acceptable time. According to our logs, 95% of the participants used the password automatically generated by ByPass, which means that our nudges were successful in encouraging users to use a secure password while keeping convenience.

Participants were asked to change the password on one of their accounts on ByPass, and the median completion time was 44 seconds. 75% of the participants used the password generation function to change their password and choose a new one. The variance for this task was very low, and participants generally did not encounter any problems. One thing that may have inflated times was that participants clicked on the re-generate password

button an average of 4.7 times. It was unclear exactly why participants were doing this, but for some, they seemed to want to demonstrate to themselves that the password was actually changing. Others were looking for a password that appealed to them.

The password change page also included a periodic password auto-change feature, allowing the user to pick a period of 30, 60, and 90 days, so the password manager will automatically change the password as the period ends. We did not include this feature in a task, but most participants expressed interest in this feature, and discussed it in relation to the annoyance of regular password changes. Time spent on these conversations may have also artificially increased the time spent on the password change tasks.

The final study task was to delete an account. At this point, participants were relatively familiar with ByPass, and the median completion time for this task was only 13 seconds.

5.3.2 Errors

We were particularly interested in the kinds of errors that participants made while using ByPass, as dangerous errors in computer security can be difficult to undo and can open the user to serious vulnerabilities [74]. Most participants did not make any errors: the median number of errors per participant was zero, and the maximum was four. We examined errors on two dimensions: the type of error committed, and the incidence of those errors.

Errors were logged by the ByPass software, and then grouped thematically for this analysis. Some errors may have been excluded during this process; e.g., repeated regeneration of a password was not counted as an error, though it is not the intended behaviour.

Table 5 shows the total number of errors by task. By far the most error-prone task was

Table 5: Total number of errors committed by usability study task.

Task	Number of errors
ByPass account registration	34
Third-party website account creation	4
Third-party website account creation via ByPass	2
Account migration	1
Password change	0
Account deletion	0
Login to third-party website via ByPass	0

the ByPass account registration, incurring a total of 34 errors.

The majority of these errors were password mismatches (i.e., the confirmed password not matching the created one), difficulties in choosing a sufficiently strong primary password, and filling in all fields appropriately. The irony that this task had the most errors is not lost on us: we included this step in the evaluation only for the purposes of realism, but we cannot ignore the fact that authenticating to the password manager itself was the most problematic part of the process. The next most error-prone task was the account creation on a third-party website, where participants had a few problems with password mismatches.

Encouragingly, there were few errors in the tasks that actually involved using ByPass. Below is a description of each error type, with the total number of times it occurred in parentheses: Figure 12 provides a bar plot of the number of errors for each type.

Password Mismatch on the Registration Page (21). This error was unique to the ByPass registration page, and it occurred when the entered primary password and the confirm primary password were not matched. This error was replicated 21 times in the study with a median of 1 time per participant. The maximum number of errors made by one individual was 3.

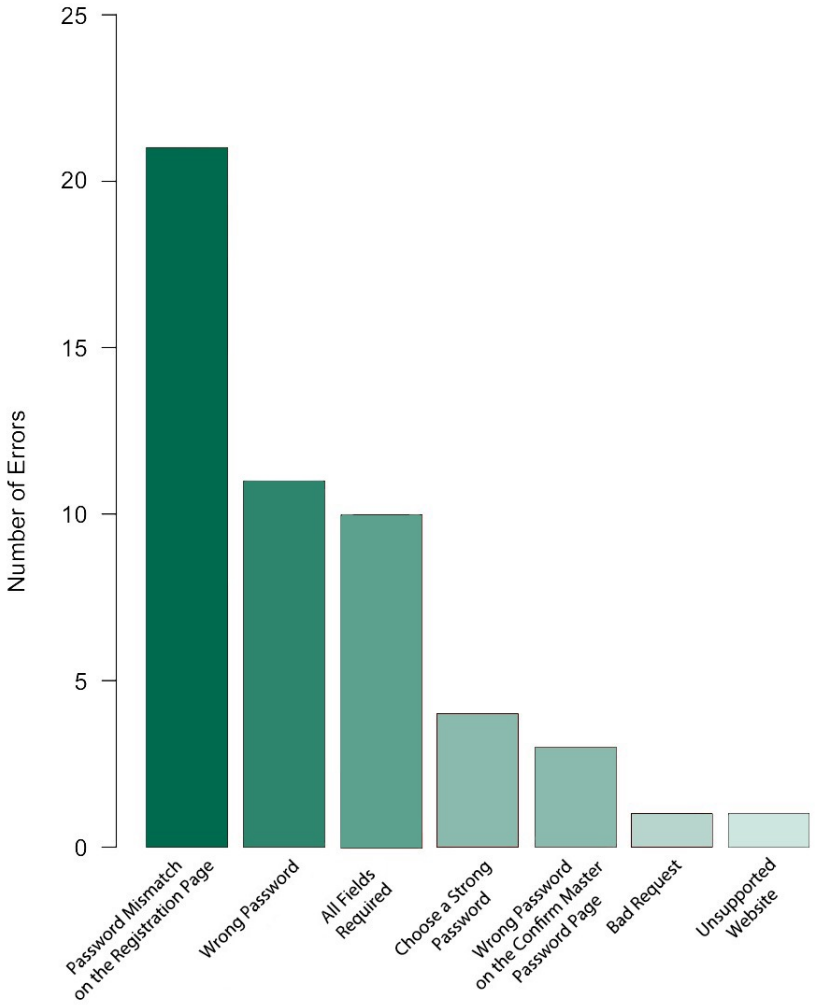


Figure 12: Bar plot showing occurred errors by type

We told participants not to use their own passwords, and on the other hand, as for the primary password they were asked to choose the strongest password they could, difficulty of remembering the newly chosen password can be one of reasons causing this error.

Wrong Password (11). We did not expect participants to perform anything other than the tasks described in the study, but some participants took steps to confirm whether or not the tasks they completed in ByPass were really accomplished on the websites. In doing so, these participants made some incorrect password errors when logging into the

websites managed through ByPass. The magnitude of this error was evaluated from both e-commerce website and webmail activity records.

There were a total of eleven incorrect password entries. Participants particularly wanted to check the password change task by visiting the website and testing the new and old passwords. Since they used the auto-password generation function for changing the password, they sometimes entered their previous password mistakenly, leading to this error.

All Fields Required (10). Although this validation was included on all the pages containing different fields, all of these errors happened on the ByPass registration page, and these errors were caused by our design of the login page. Our extension's window range was not wide enough to accommodate all the fields on the registration page when the user was typing into the primary password field. As the user typed, the password strength meter gave more feedback and expanded in size, shifting the password confirmation box below the page break. Although we included a hint encouraging users to scroll down, several participants missed it.

Choose a Strong Password (4). For study realism, we included a password strength meter (based on `zxcvbn` [73]) on the ByPass account creation page, and required participants to choose a primary password with a "strong" rating. Four users attempted to use a password that did not fill this requirement, and these problems stemmed from not paying attention to the instruction reminding them to pick a strong password, from a lack of understanding of what forms a good password, and from ignoring the feedback from the password strength meter.

Incorrect primary Password (3). Following creation of their primary password, participants were asked to use it to log into ByPass. Three participants made mistakes in entering their password during login.

Bad Request (1). This error category encompasses several different errors: when ByPass sends packets containing user information to websites, it waits for an HTTP response status of 200 (OK) as an approval. Incorrect email addresses, incorrect passwords, and request timeouts belong to this type of error. In our study, this happened only once when a participant mistyped an email address.

Unsupported Website (1). Features like creating an account, changing passwords, or deleting an account, can be done only for websites for which we have their API. This error occurs if the user tries to do any of these functions on an unsupported website. One participant had a typo while they were entering the name of the webmail service for the account creation function on ByPass, and because we did not support that name, this error appeared.

5.3.3 Usability Perceptions

In our case, user perception [27] is classified as the mechanism by which individuals select, manage, and recall their sensory impressions in order to perceive the environment of the password manager. These impressions are affected by the suitable and necessary design of the usability properties. As it can be observed from the findings when a password manager is well designed, user perception leads to improved incentive for using it which increases the probability of user satisfaction and helping them to have a better experience

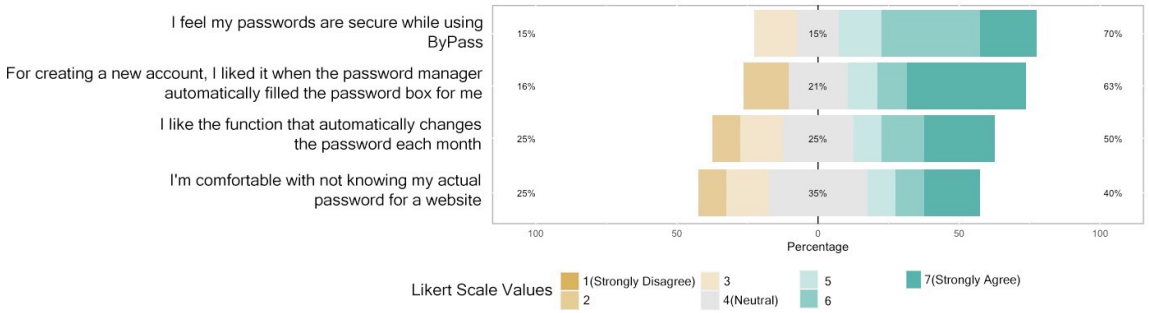


Figure 13: Responses to Likert scale questions asking about participants' interest in ByPass features.

on managing their passwords.

We were interested in how our participants perceived the usability and security of ByPass. We asked participants for their responses to 12 Likert-scale questions, asking about the ease of use, perceived security, and desire to use ByPass in future. Participants were asked to rate their agreement on a 7 point scale where 7 was most positive. Figure 13 shows the distribution of responses for selected questions.

Participants were universally positive about the ease of use of the password change process ($med = 7$) and website login processes ($med = 7$) in ByPass. They were also positive about the ease of creating new accounts through ByPass ($med = 7$) and adding existing accounts ($med = 7$), as well as the ease of deleting accounts ($med = 7$). Participants also found it easy to migrate a third-party account to ByPass ($med = 7$).

The median agreement score for the perceived security of ByPass was 6, indicating general satisfaction with the security of ByPass, though participants expressed more frustration with the process of choosing a primary password ($med = 5$). In the discussion,

some participants mentioned that they would trust ByPass more if it were a software application from a well-recognized organization like Google. A few of the more technical participants commented that they liked the fact that there is no server-side component to the manager.

Participants were most negative about features that related to user control, see Figure 13. They were not positive about the concept of not knowing the actual password ($med = 4.5$), and displayed mixed responses about the password generation and auto-change features. 31% of the participants wanted to use their own password, instead of relying on a password generator to create one for them.

The likeliness of using the auto-password change function seemed to be less wanted. Since they were not asked to use this function during the study, lack of awareness about this feature as well as not having a clear perception could be the explanations for these low ratings.

5.3.4 Qualitative Observations

We documented feedback and difficulties encountered by participants during the qualitative analysis. Below we are going to show some of the participant's expressions on ByPass.

For the password auto-change function, P1 said: *"As the password for the website is generated by the password manager, I feel it's secure, and there is no need to change it each month,"* but on the other hand P2, and P8 liked this feature: *"I liked the idea of password auto-change function"* and *"I liked that ByPass can change my password automatically but giving me a notification before changing the password would be nice."*

These comments helped us to conclude, users, in general, are comfortable with the idea of automatic password generation. Still, at the same time, they need some control over those generated passwords. Comments similar to these helped us to design our password generator considering usability.

We received quite good feedback on the third-party account creation function through ByPass. P6 had an experience on using LastPass and he mentioned: *"I always had a problem with copying the password from the password manager and pasting it into the website;"* P10 after creating the account expressed: *"That was easy and very convenient that I didn't have to redo the login process."* Three out of the 20 participants decided to verify whether by ByPass they could create a webmail account. They expressed their feelings after testing as P7 mentioned: *"I liked that ByPass automatically opened the home page of the Webmail website for me;"* P11: *"It's a good idea and hope that it really becomes successful;"* P16: *"ByPass is really different from other password managers and it makes everything much easier;"* P20: *"I liked this feature because it's easy and fast."* The account deletion function had positive comments overall. P4: *"I like the delete function because it's always hard to find the delete button in the websites."*, P9: *"I'm a big fan of your delete account function"*, P10: *"It was straightforward to delete an account and ByPass is really easy to use as it has lots of colors and icons that helped me find the way"*. We had also not so positive comments as well; P1 told us: *"It feels quite bad since ByPass does not show the official website's page"*, P6 mentioned: *"I think some functions like syncing and saving notes are missing in ByPass."*, and P8 added: *"For account deletion I would like to be more steps to prevent me from unintentional clicks"*.Responses on the account management features of

ByPass allowed us to validate the usability improvements that ByPass offers.

Participants gave us some comments on the design as P7 said: *"Because of the icons you used it was clear for me how to find my way;"* and according to the wizard page P13 mentioned: *"I liked that you put a page for showing the following the steps, I could complete the tasks without even reading the task description!"* Our very last question in the post-test questionnaire was *"How would you rate ByPass?"* Both the rating score and qualitative analysis show that most of the participants liked ByPass although it has some downsides too.

To conclude, ByPass re-architects the password manager to adjust the locus of control for the user. The user is given more control over some aspects of their password management tasks (password changes, account deletion), but less control over passwords themselves. In ByPass, passwords are generated randomly by default, and obscured to the user.

Chapter 6

Discussion & Recommendations

Although widely recommended as a simple step that users can take towards improving their password security, few users adopt password managers. Adoption problems and the extra work of using a manager are thought to be part of the reason that users do not turn to these tools [3, 4]. In this paper, we designed and evaluated ByPass, a password manager that rethinks the users' interactions with the password manager, thereby encouraging adoption and encouraging users to make use of security features. ByPass uses an API for secure communication between the password manager and websites, freeing the user from creating and avoiding errors resulting from copying and pasting. ByPass is built to extend traditional password manager features and supports new functionality such as automated password changes and account deletion. ByPass nudges users toward using secure randomly generated passwords.

The downside of the ByPass approach to password management is that it requires buy-in from websites, who must implement the API. We hope that the promise of increased

security compliance from users might motivate websites to include the ByPass API, and that in turn, this might encourage users to adopt ByPass over other password managers. We acknowledge the uphill nature of this process, while leaving it somewhat out of scope for this work; we think it is worthwhile to investigate how an architecture such as that of ByPass impacts users even without knowing how uptake might look. We specifically designed ByPass to also be backwards compatible with websites that do not implement the API, and designed the API so that it can be included as a library by website developers. In future work, we plan to further study the feasibility of ByPass's implementation, as well as the implications for web developers, and how they can be supported in implementing ByPass.

We conducted two early evaluations of ByPass's usability, and found that users were able to understand and use the features in ByPass. Most participants did not encounter major problems, though there is undoubtedly still room for improvement in the ByPass user interface. The results of this study will be used to improve the prototype development. However, through the process of designing, implementing and evaluating ByPass, we made a number of observations that affect the design of not only ByPass, but password management tools in general.

6.1 An Abstraction Layer for Accounts

ByPass adds a layer of abstraction between the user and the website, where all account-management related interactions take place within the password manager. Adding this

abstraction layer brings up various design questions: Where should the password manager sit in the physical space of the web browser? How to instrument the browser extension to "correctly" interrupt interactions with websites? How to train users to go to the password manager first for account management-related tasks? What kind of language should be used to correctly convey password tasks when they are de-situated from their website contexts? Our evaluation suggests that while most participants were able to interpret what was happening, some had great problems.

Our goal was to create a space where security could be the users' primary task, and allow them to focus cleanly and consistently on account management tasks. The constancy of the ByPass interface is intended to allow users a greater sense of control over their passwords and accounts. By using the API to move account interactions into this space, we hoped to create an interface where users knew where to address security concerns, and access the controls to address those concerns. Current password managers hint at this functionality (and include innovative tools, such as security audits) but their placement outside the authentication interaction hampers the functionality they are able to support.

6.2 Control vs. Automation

ByPass re-architects the password manager to adjust the locus of control for the user. The user is given more control over some aspects of their password management tasks (password changes, account deletion), but less control over passwords themselves. In ByPass, passwords are generated randomly by default, and obscured to the user.

Our contention in ByPass is that the user does not need to know their passwords. With the addition of the API, this is functionally true, but it does not seem to fulfill users' sense of security self-efficacy [53]. In our study, users expressed unhappiness about not knowing their passwords, both in comments and in the post-test questionnaire. We also observed them engaging in "epistemic actions" [40] – actions that serve only to understand a situation rather than to advance a goal, such as repeatedly regenerating passwords, or double-checking on the website that a password had actually changed. Some of these reactions may be due to unfamiliarity, but they echo the findings of previous studies where users have expressed both a desire for control [4, 51] and a corresponding sense of responsibility for security [51].

Automating security is often tricky. Solutions such as TLS certificates automate nearly all of the security interaction, turning to the user only when a certificate is not validated, but demonstrate failures when users are unprepared to cope with these situations. Conversely, passwords leave nearly all of the control in the hands of the users, expecting them to exert individual responsibility for all aspects of the password management task. ByPass attempts to find a middle ground for users, removing tasks that needlessly involve users (e.g., reading the password policy, choosing a password that conforms to it). In designing ByPass, we became aware of the myriad corner cases in which the user might need to exert control, and we attempted to leave accessible controls for situations such as assigned passwords, and other unusual contexts.

6.3 Testing a Password Manager

In evaluating ByPass, one difficulty we encountered was creating a realistic testing scenario. Ecological validity, or the realism of the study situation, as is of the utmost importance in security studies. Tasks that seem easy or manageable as primary tasks (such as remembering a password) are not always manageable when happening in the context of another more important task. Simulating this for a password manager is difficult – the gold standard evaluation would seem to be a field study where participants use the manager for their own accounts. We would want to collect instrumented data from such an evaluation, and privacy could be a significant concern (not only for data collection, but potentially biasing participants' behaviour). For ByPass, a study of this type carried the additional challenge of needing websites to be implemented with the API, further restricting our ability to have users test the manager with their own accounts.

Demand characteristics are the other reason for making ecological validity more challenging. As proposed by Nichols et al. [47], it is likely that participants respond in ways they feel the researcher needed or intended them to instead of acting spontaneously. In ByPass, although we made the best effort to reduce the demand, but as we were paying the participants and the researcher was explaining the tasks to them as well as observing them during the study, there would be a chance that their responses may have been affected.

6.4 Offline vs. Online Password Storage

In our prototype of ByPass, we allow both offline and online password storage and let the user decides which option meets their needs. The inclusion of online password storage allows portability and gain access to password sharing feature. However, this option introduces a trusted third party requirement, and in an ideal security design perspective, a third party cannot be trusted. Furthermore, a trusted third party can be an obstacle in the user's and service providers' trust in ByPass. On the other hand, having the password in an offline database removes the trusted third party from the design, but ByPass severally suffers from portability and cross-device support. Furthermore, many requirements for the use cases of a modern password manager cannot be satisfied with offline password storage. Hence, in our current prototype, we offer both online and offline password storage.

Chapter 7

Conclusion

No work so far has focused on using APIs within the password manager development. Our key contribution is developing a password manager with new capabilities resulting from different API calls from the websites. Although this approach requires buy-in from websites, it results in increased password manager usability resulting from fewer actions required by the user. Assuming the integration of password-based online services, in ByPass, we shift the focus to comprehensive *account management* instead of current approaches to *password management*, primarily limited to create/save/fill passwords. We address critical security limitations of current password managers, and at the same time, improve ease of use and control over numerous accounts that a typical user needs to manage.

The usability of password managers is a key issue, since there is no benefit in developing a secure password manager when users cannot make use of it. In this thesis, we designed and implemented ByPass, a password management software offering new features to users by reducing the number of required actions for specific task completion. The

key idea of our proposed password manager is that API-enabled secure communication between the password manager and websites allows various password management tasks to be streamlined for the end user. ByPass is not only a password manager but can also be referred to as an account management system as it supports third-party account creation, password change,password sharing, and account deletion directly through the password manager.

We constructed a prototype implementation of ByPass and evaluated it in a user study with 20 participants. The results show that the participants found ByPass easy to use, and our concept is effective both in terms of usability and security. ByPass successfully nudged participants towards using automatically generated passwords, and most of the participants were able to learn how to use ByPass efficiently, while making few errors.

In future work, we plan to integrate the findings of this usability evaluation into the ByPass user interface, integrate new features to support users, and further test ByPass in more ecologically valid scenarios. We also plan to address elements of password manager design that were left out of scope in this early prototype. One such fragment of our current design is password storage. We plan to further investigate and improve our design to have online password storage without the need for a trusted third party with the inclusion of a decentralized architecture.

ByPass raises important questions about where security controls should be placed for end-users. Users desire control, but this may be at odds with good usability. The abstraction of password tasks in password manager creates an extra management step for users, and managers must be carefully designed so that users are supported in understanding this

abstraction. However, we think that redesigning the password manager could be key to seeing its wide adoption.

Bibliography

[1] 1Password. Password Manager for Families, Businesses, Teams, 2021. URL https://www.1password.com/.

[2] 1Password. If You Forgot Your Master Password or You Can't Unlock 1Password, 2021. URL https://support.1password.com/forgot-master-password/.

[3] Nora Alkaldi and Karen Renaud. Why Do People Adopt, or Reject, Smartphone Password Managers? In *International Symposium on Human Aspects of Information Security & Assurance*, Darmstadt, Germany, 2016.

[4] Nora Alkaldi, Karen Renaud, and Lewis Mackenzie. Encouraging Password Manager Adoption by Meeting Adopter Self-Determination Needs. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*, Maui, Hawaii, USA, 2019.

[5] Salvatore Aurigemma, Thomas Mattson, and Lori Leonard. So Much Promise, so Little Use: What is Stopping Home End-Users from Using Password Manager Applications? *50th Hawaii International Conference on System Sciences*, 2017.

[6] Jef Ausloos. The 'Right to be Forgotten' – Worth remembering? *Computer Law & Security Review*, 28(2):143–152, 2012.

[7] Avira Operations GmbH & Co. Avira Password Manager - Secure and Easy, 2021. URL https://www.avira.com/en/password-manager/.

[8] Natã M Barbosa, Jordan Hayes, and Yang Wang. UniPass: Design and Evaluation of a Smart Device-Based Password Manager for Visually Impaired Users. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 49–60, Heidelberg, Germany, 2016.

[9] Nigel Bevan, James Carter, and Susan Harker. ISO 9241-11 revised: What Have We Learnt About Usability Since 1998? In *International Conference on Human-Computer Interaction*, pages 143–151, 2015.

[10] Bitwarden Inc. Bitwarden Open Source Password Manager, 2021. URL https://bitwarden.com/.

[11] M Blanchou and P Youn. Password Managers: Exposing Passwords Everywhere. *Whitepaper, iSEC partners*, 2013.

[12] Carl Boettiger. An Introduction to Docker for Reproducible Research. *SIGOPS Oper. Syst. Rev.*, pages 71–79, 2015.

[13] Joseph Bonneau, Cormac Herley, Paul C Van Oorschot, and Frank Stajano. The Quest

to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. In *IEEE Symposium on Security and Privacy*, pages 553–567, San Francisco, CA, USA, 2012.

[14] Michael Carr and Siamak F. Shahandashti. Revisiting Security Vulnerabilities in Commercial Password Managers. In *IFIP International Conference on ICT Systems Security and Privacy Protection*, pages 265–279, Maribor, Slovenia, 2020.

[15] R. Chatterjee, J. Bonneau, A. Juels, and T. Ristenpart. Cracking-Resistant Password Vaults using Natural Language Encoders. In *IEEE Symposium on Security and Privacy*, pages 481–498, San Jose, CA, USA, 2015.

[16] Haibo Cheng, Zhixiong Zheng, Wenting Li, Ping Wang, and Chao-Hsien Chu. Probability Model Transforming Encoders Against Encoding Attacks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 1573–1590, Santa Clara, CA, USA, 2019.

[17] Sonia Chiasson, Paul C van Oorschot, and Robert Biddle. A Usability Study and Critique of Two Password Managers. In *15th USENIX Security Symposium (USENIX Security 06)*, pages 1–16, Vancouver, BC, Canada, 2006.

[18] Sonia Chiasson, PC van Oorschot, and Robert Biddle. Even Experts Deserve Usable Security: Design Guidelines for Security Management Systems. In *SOUPS Workshop on Usable IT Security Management (USM)*, pages 1–4, Pittsburgh, PA, USA, 2007.

[19] CoudFlare Inc. The Web Performance & Security Company, 2021. URL https://www.cloudflare.com/en-ca/.

[20] Dashlane Inc. Password Manager App for Home, Mobile, Business, 2021. URL https://www.dashlane.com/.

[21] Fernando Doglio. *Pro REST API Development with Node.js*. Apress, 2015.

[22] Dropbox. Dropbox is the World's First Smart Workspace, 2021. URL https://www.dropbox.com/.

[23] Michael Fagan and Mohammad Maifi Hasan Khan. Why Do They Do What They Do?: A Study of What Motivates Users to (Not) Follow Computer Security Advice. In *Twelfth Symposium on Usable Privacy and Security (SOUPS 2016)*, pages 59–75, Denver, CO, USA, 2016. URL https://www.usenix.org/conference/soups2016/technical-sessions/presentation/fagan.

[24] Paolo Gasti and Kasper B. Rasmussen. On the Security of Password Manager Database Formats. In *Proceedings of the 13th International Conference on Information Security and Cryptology*, pages 233–251, Berlin, Heidelberg, 2012. Springer-Verlag.

[25] Maximilian Golla, Benedict Beuscher, and Markus Dürmuth. On the Security of Cracking-Resistant Password Vaults. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1230–1241, Vienna, Austria, 2016. Association for Computing Machinery.

[26] Google Developers. Google Chrome Extension, 2021. URL https://developer.chrome.com/docs/extensions/.

[27] Douglas J Gould, Mark A Terrell, and Jo Fleming. A Usability Study of Users' Perceptions Toward a Multimedia Computer-Assisted Learning Tool for Neuroanatomy. 4(1):175–183, 2008.

[28] Joshua Gray, Virginia NL Franqueira, and Yijun Yu. Forensically-sound analysis of security risks of using local password managers. In *2016 IEEE 24th International Requirements Engineering Conference Workshops (REW)*, pages 114–121, Beijing, China, 2016.

[29] Hana Habib, Sarah Pearman, Jiamin Wang, Yixin Zou, Alessandro Acquisti, Lorrie Faith Cranor, Norman Sadeh, and Florian Schaub. "It's a Scavenger Hunt": Usability of websites' opt-out and data deletion choices. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12, New York, NY, USA, 2020. Association for Computing Machinery.

[30] Muhamad Haekal and Eliyani. Token-Based Authentication using JSON Web Token on SIKASIR RESTful Web Service. In *International Conference on Informatics and Computing (ICIC)*, pages 175–179, Mataram, Indonesia, 2016.

[31] Cormac Herley and Paul Van Oorschot. A Research Agenda Acknowledging the Persistence of Passwords. *IEEE Security & Privacy*, 10(1):28–36, 2012.

[32] Philip G Inglesant and M. Angela Sasse. The True Cost of Unusable Password Policies: Password Use in the Wild. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 383–392, New York, NY, USA, 2010.

[33] Mathias Karlsson. How I made LastPass Give Me All Your Passwords, 2016. URL https://labs.detectify.com/2016/07/27/how-i-made-lastpass-give-me-all-your-passwords/.

[34] Ambarish Karole, Nitesh Saxena, and Nicolas Christin. A Comparative Usability Evaluation of Traditional Password Managers. In *International Conference on Information Security and Cryptology*, Seoul, Korea, 2010. Springer-Verlag.

[35] Stefan Kimak and Jeremy Ellman. HTML5 IndexedDB Encryption: Prevention Against Potential Attacks. *International Journal of Intelligent Computing Research*, 6:621–630, 2015.

[36] Lastpass. The Last Password You Have to Remember, 2021. URL https://www.lastpass.com/.

[37] Lastpass. Recover Your Lost Master Password for LastPass, 2021. URL https://support.logmeininc.com/lastpass/help/recover-your-lost-master-password-lp020010.

[38] Zhiwei Li, Warren He, Devdatta Akhawe, and Dawn Song. The Emperor's New Password Manager: Security Analysis of Web-Based Password Managers. In *23rd*

USENIX Security Symposium (USENIX Security 14), pages 465–479, San Diego, CA , USA, 2014.

[39] Limitlesslane. Online Password Manager - Auto Login - Auto Save, 2021. URL https://limitlesslane.com/.

[40] Zhicheng Liu, Nancy Nersessian, and John Stasko. Distributed Cognition as a Theoretical Framework for Information Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1173–1180, 2008.

[41] LogMeOnce. PassswordLess & Smarter Identity Management, 2021. URL https://www.logmeonce.com/.

[42] Sanam Ghorbani Lyastani, Michael Schilling, Sascha Fahl, Michael Backes, and Sven Bugiel. Better Managed than Memorized? Studying the Impact of Managers on Password Strength and Reuse. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 203–220, Baltimore, MD, USA, 2018.

[43] Raymond Maclean and Jacques Ophoff. Determining Key Factors that Lead to the Adoption of Password Managers. In *2018 International Conference on Intelligent and Innovative Computing Applications (ICONIC)*, pages 1–7, Mon Tresor, Mauritius, 2018.

[44] Daniel McCarney, David Barrera, Jeremy Clark, Sonia Chiasson, and Paul C

Van Oorschot. Tapas: Design, Implementation, and Usability Evaluation of a Password Manager. In *Proceedings of the 28th Annual Computer Security Applications Conference*, pages 89–98, Orlando, FL, USA, 2012.

[45] Microsoft Azure. Azure Defender, 2021. URL https://azure.microsoft.com/en-ca/services/azure-defender/.

[46] Mozilla and Individual Contributors. MDN Web Docs: Synchronous and Asynchronous Requests, 2021. URL https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/Synchronous_and_Asynchronous_Requests.

[47] Austin Lee Nichols and Jon K Maner. The good-subject effect: Investigating participant demand characteristics. *The Journal of general psychology*, 135(2):151–166, 2008.

[48] NIST. *NIST Special Publication 800-63B, Digital Identity Guidelines, SP-800-63B Section 5.1.1.2*. 2017.

[49] Norton Life Lock Inc. Norton Password Manager, 2021. URL https://my.norton.com/extspa/passwordmanager/.

[50] Sean Oesch and Scott Ruoti. That Was Then, This Is Now: A Security Evaluation of Password Generation, Storage, and Autofill in Browser-Based Password Managers. In *USENIX Security Symposium*, Anaheim, CA, USA, 2020.

[51] Sarah Pearman, Shikun Aerin Zhang, Lujo Bauer, Nicolas Christin, and Lorrie Faith

Cranor. Why People (don't) Use Password Managers Effectively. In *Fifteenth Symposium On Usable Privacy and Security (SOUPS 2019)*, pages 319–338, Santa Clara, CA, USA, 2019.

[52] Hirak Ray, Flynn Wolf, and Ravi Kuber. Why Older Adults (Don't) Use Password Managers. In *30th USENIX Security Symposium (USENIX Security 21)*, Vancouver, BC, Canada, 2021.

[53] Hyeun-Suk Rhee, Cheongtag Kim, and Young U Ryu. Self-efficacy in Information Security: Its Influence on End Users' Information Security Practice Behavior. *Computers & security*, 28(8):816–826, 2009.

[54] RoboForm. RoboForm Password Manager: Say Goodbye to Writing Down Passwords, 2021. URL https://www.roboform.com/.

[55] Blake Ross, Collin Jackson, Nick Miyake, Dan Boneh, and John C Mitchell. Stronger Password Authentication Using Browser Extensions. In *14th USENIX Security Symposium (USENIX Security 05)*, Baltimore, MD, USA, 2005.

[56] J. B. Rotter. Generalized Expectancies for Internal Versus External Control of Reinforcement. *Psychological monographs*, 80(1):1–28, 1966.

[57] Scott Ruoti and Kent Seamons. End-to-End Passwords. In *NSPW*, New Hampshire, USA, 2017.

[58] Sunyoung Seiler-Hwang, Patricia Arias-Cabarcos, Andrés Marín, Florina Almenares, Daniel Díaz-Sánchez, and Christian Becker. "I Don't See Why I Would Ever Want to

Use It" Analyzing the Usability of Popular Smartphone Password Managers. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1937–1953, London, United Kingdom, 2019.

[59] M. Shirvanian, S. Jareckiy, H. Krawczykz, and N. Saxena. SPHINX: A Password Store that Perfectly Hides Passwords from Itself. In *International Conference on Distributed Computing Systems (ICDCS'17)*, pages 1094–1104, Atlanta, GA, USA, 2017.

[60] David Silver, Suman Jana, Dan Boneh, Eric Chen, and Collin Jackson. Password Managers: Attacks and Defenses. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 449–464, San Diego, CA, USA, 2014.

[61] Azzam Sleit and Ala'a Al-Shaikh. Evaluating IndexedDB Performance on Web Browsers. In *8th International Conference on Information Technology (ICIT)*, pages 488–494, Jordan, 2017.

[62] Trevor Smith, Scott Ruoti, and Kent Seamons. Augmenting Centralized Password Management with Application-Specific Passwords. In *Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017)*, Santa Clara, CA, USA, 2017.

[63] DO SON. VaultBreaker: Attacks Against Common Password Managers, 2019. URL https://securityonline.info/vaultbreaker/.

[64] Emily Stark, Michael Hamburg, and Dan Boneh. Symmetric Cryptography in

Javascript. In *2009 Annual Computer Security Applications Conference*, pages 373–381, Honolulu, HI, USA, 2009.

[65] Douglas Stebila and Nick Sullivan. An Analysis of TLS Handshake Proxying. In *Proceedings of the 2015 IEEE Trustcom*, pages 279–286, Helsinki, Finland, 2015.

[66] Elizabeth Stobert and Robert Biddle. A Password Manager that Doesn't Remember Passwords. In *Proceedings of the 2014 New Security Paradigms Workshop*, pages 39–52, New York, NY, USA, 2014.

[67] Elizabeth Stobert and Robert Biddle. A Password Manager that Doesn't Remember Passwords. In *Proceedings of the 2014 New Security Paradigms Workshop*, pages 39–52, Victoria, BC, Canada, 2014.

[68] Elizabeth Stobert and Robert Biddle. The Password Life Cycle. *ACM Transactions on Privacy and Security (TOPS)*, 21(3):1–32, 2018.

[69] Elizabeth Stobert, Tina Safaie, Heather Molyneaux, Mohammad Mannan, and Amr Youssef. ByPass: Reconsidering the Usability of Password Managers. In *International Conference on Security and Privacy in Communication Systems*, pages 446–466, Washington, DC, USA, 2020.

[70] Ben Stock and Martin Johns. Protecting users against XSS-based password manager abuse. In *Proceedings of the 9th ACM symposium on Information, computer and communications security*, New York, NY, USA, 2014. Association for Computing Machinery.

[71] Ding Wang, Zijian Zhang, Ping Wang, Jeff Yan, and Xinyi Huang. Targeted Online Password Guessing: An Underestimated Threat. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1242–1254, New York, NY, USA, 2016. Association for Computing Machinery.

[72] Cathleen Wharton, Janice Bradford, Robin Jeffries, and Marita Franzke. Applying Cognitive Walkthroughs to More Complex User Interfaces: Experiences, Issues, and Recommendations. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 381–388, 1992.

[73] Daniel Lowe Wheeler. zxcvbn: Low-Budget Password Strength Estimation. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 157–173, Vancouver, BC, Canada, 2016.

[74] Alma Whitten and J Doug Tygar. Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *8th USENIX Security Symposium (USENIX Security 99)*, Washington, DC, 1999.

[75] Frances F Yao and Yiqun Lisa Yin. Design and Analysis of Password-Based Key Derivation Functions. pages 3292–3297, 2005.

[76] Rui Zhao, Chuan Yue, and Kun Sun. A Security Analysis of Two Commercial Browser and Cloud Based Password Managers. In *2013 International Conference on Social Computing*, pages 448–453, Alexandria, VA, USA, 2013.

Appendices

Appendix A

UDS Framework

We evaluate ByPass using the Usability-Deployability-Security (UDS) framework of Bonneau et al. [13] which is a framework for evaluating user authentication schemes. They proposed 25 different properties. We updated four of the properties to make a more detailed comparison between ByPass and other password managers. Towards choosing the password managers, we picked five of the most popular password manager extensions based on the Chrome web store's number of downloads (LastPass [36], Norton Password manager [49], Avira [7], Dashlane [20], and 1PasswordX [1]) as well as iCloud Keychain, Google Password Manager, and Firefox Lockwise.

Below, we explain some of the criteria which are more related to ByPass from the Bonneau et al. [13] UDS framework in addition to introducing four new criteria that were not listed in the original framework and we added them to make a better comparison of ByPass.

U1 – Effortless-Account-Management: The password manager supports third-party

website's account managing, which means that the users can manage their account like changing password through the password manager.

U2 – Ease-of-Account-Sharing: The password manager offers sharing an account's credentials so that the users can securely share their credentials for a specific account using the password manager.

U7 – Easy-Recovery-from-Loss: An individual can easily gain access over the account if he forgets the primary password. Recovery becomes convenient, for example by utilizing backups or secondary recovery schemes.

D1 – Offering-only-Offline-Database: The password manager offers the offline database for the users who are not comfortable using applications with third-party servers.

D3 – Server-Compatible: It is not required for the service provider to alter the existing setup to communicate with the password manager.

D5 – Mature: In addition to research, a password manager should be implemented and deployed in a large-scale environment for authentication purposes in order to achieve a full circle for this criteria.

S8 – Resilient-to-Sweep-Attack: The password manager resists the type of attacks caused by the auto-fill and auto-submission features. This attack was mentioned by Silver et al. [60] while there is an active man-in-the-middle network attacker who is able to extract passwords from the password manager.

S12 – Unlinkable: This is a criterion that involves privacy where two conspiring websites cannot determine if the same user is authenticating to each of them (disregarding IP address and the username).

Table 6: Evaluation of ByPass using UDS Framework

Password Managers	U1: Effortless-Account-Management	U2: Ease-of-Account-Sharing	U3: Memorywise-Effortless	U4: Easy-to-learn	U5: Efficient-to-Use	U6: Infrequent-Errors	U7: Easy-Recovery-from-Loss	D1: Offering-only-Offline-Database	D2: Negligible-Cost-per-User	D3: Server-Compatible	D4: Browser-Compatible	D5: Mature	D6: Non-Proprietary	S1: Resilient-to-Physical-Observation	S2: Resilient-to-Targeted-Impersonation	S3: Resilient-to-Throttled-Guessing	S4: Resilient-to-Unthrottled-Guessing	S5: Resilient-to-Internal-Observation	S6: Resilient-to-Leaks-from-Other-verifiers	S7: Resilient-to-Phishing	S8: Resilient-to-Sweep-Attack	S10: No-Trusted-Third-Party	S11: Requiring-Explicit-Consent	S12: Unlinkable
ByPass (Online)	●	●	◑	●	●	●			●		●	◑	●		●						●	●	◑	●	●
ByPass (Offline)	●		◑	●	●	●		●	●	●	●	◑	●		●						●	●	●	●	●
LastPass		◑	◑	●	◑	●	◑		◑	●	●	●			◑						◑		●	●	●
Norton Password Manager			◑	◑	◑	●			●	●	●	●			◑						◑		●	●	●
Avira			◑	●		●			◑	●	●	●			◑						◑		●	●	●
Dashlane		◑	◑		◑	●	◑		◑	●	●	●			◑						◑		●	●	●
1Password X		◑	◑	◑	◑	●	◑		◑	●	●	●			◑						◑		●	●	●
Firefox Lockwise			◑	●	◑	●	◑		●	●	●	●	●		◑						◑		●	●	●
iCloud Keychain			◑	●	◑	●	●		●	●	●	●			◑						◑		●	●	●
Google Password Manager (Android)			◑	●	◑	●	●		●	●	●	●			◑						◑		●	●	●

●= Offers the benefit; ◑= Almost offers the benefit; ○= Does not offer the benefit
U = Usability; D = Deployability; S = Security

As Table 6 illustrates, ByPass fully addresses 14 out of 24 criteria. In what follows, we are going to explain more on the result of the evaluation we did on the aforementioned password managers among the 24 criteria.

A.1 Usability

By taking advantage of APIs, ByPass addresses effortless account management. Via the password manager, the user can log in to their account and easily complete account management functions such as changing the password, deleting the account, or even creating a new account.

Account sharing is another criterion that improves the password manager's usability as well as security, in the way that the user doesn't need to copy the plain password and give it to another person by text or email. Among all password managers only ByPass

supports account sharing feature. As LastPass [36], Dashlane [20], and 1PasswordX [1] allow sharing passwords only hence they get half-circle.

For memory-wise effortless criteria, all the password managers listed here earn a half circle as the user needs to recall at least one thing, which is the primary password.

Measuring how convenient is to learn about the functionality of a password manager is a qualitative thing. Nevertheless, the approach we wanted to calculate the ease of use was to count the number of clicks and the time to perform a given task, such as registering an account and testing if the password manager provides the user with any tips while attempting to complete a task. We used this method for evaluating the use efficiency criteria too.

We didn't face any problem in terms of errors while we were trying to log in with the aforementioned password managers, and it isn't hard for a user to use them, thus all of them receive full circle for the Infrequent-Errors criteria.

In Easy-Recovery-from-Loss criteria, the user should be able to regain the ability to authenticate if they forgot their credentials. In ByPass, Norton Password Manager [49], and Avira [7], if the user forgets the primary password, they wouldn't be able to get access to their account again, so that they receive no circle. LastPass [36], Dashlane [20], 1PasswordX [1], and Firefox Lockwise receive half-circle as they provide some ways to recover the account in the case of forgetting the primary password. Using the family or team account in 1PasswordX [2] or setting an SMS account recovery in LastPass [37] enables them to receive half circle but not full circle since there is still a chance that the user has never set up these recovery approaches. The iCloud Keychain and Google Password

Manager provide various user friendly ways–e.g., using two devices, SMS, email, etc.–for the account recovery and therefore they are achieving full circles.

A.2 Deployability

ByPass is the only password manager that provides the support for fully Offline-Database in terms of deployability. It is also non-proprietary as it is open source like Firefox Lockwise. However, it is the only password manager which is not Server-Compatible as it requires slight changes in the website's servers for providing the APIs. ByPass is also not as mature as the other password managers; hence, it receives half circle for this criteria. ByPass, Norton Password Manager [49], Firefox Lockwise, iCloud Keychain, and Google Password Manager are fully free of charge, while the other managers have free trial version with reduced functionality and the user needs to pay for the full version.

A.3 Security

For such security criteria like Resilient-to-Physical observation, Throttled and Unthrottled guessing, and Resilient-to-Internal-Observation, password managers cannot achieve any circle according to the nature of them in relying on a password-based authentication system.

All the password managers we have evaluated achieve full circle for the Unlinkable criteria in reference to study of Bonneau et al. [13]. As for ByPass, randomly generated passwords are unlikable and when we disregard the IP address, similar to previous work [13], ByPass is unlikable as well even though it uses APIs for authentication.

Among all the password managers, ByPass is the only one that receives full circle for Resilient-to-Phishing and Resilient-to-Sweep-Attack [60] criterias. The reason is in using an API for skipping the landing page as ByPass serves the user on the home page of the website thus two goals will be achieved. One is all the APIs are through the TLS connection and an adversary would not be able to intercept it, and the second goal is preventing an attacker from modifying the landing page of the website. As we mentioned earlier, an attacker with the power of controlling the network can be able to inject some malicious scripts and different invisible login forms on the website's page so that the password managers with the auto-fill or auto-submit features would automatically fill in the credentials boxes [60].

Although ByPass online mode has a trusted third-party server, it allows the user to choose to deploy their own server. Therefore, it will achieve a half circle for No-Trusted-Third-Party criteria. Firefox Lockwise is also an open-source password manager that gives the user the ability to deploy their own version hence gets half-circle same as ByPass. On the other hand, the ByPass offline mode requires no trusted third-party hence ByPass offline mode achieves full circle.

The iCloud Keychain and Google Password Manager have almost the same behaviour-e.g. both come free with their software suite, come built-in with their browsers and devices, ability to recover from loss, etc. In contrast, Firefox Lockwise being also a browser first password manager has some distinct differences from these two. As a starter, it is not proprietary like the other two. Although it is open-source, in terms of deployability, it is very similar to iCloud Keychain and Google Password Manager.

In conclusion, ByPass acts better in terms of security compared to the other eight password managers. Although ByPass is not so mature and server compatible, it still can compete with popular password managers and encourages users to have a better and more secure experience using a manager.

Appendix B

User Study Script

- Greeting participants

- Providing the consent form: Providing the consent form: This is the consent form, please read it carefully. As it is mentioned in the form.

- *While they are reading the consent form*: This study will take around 1 hour and you will be given 15$ for your participation. During this time, we first give you a pre-test questionnaire which I'm going to explain more later, then you are asked to complete 6 tasks related to our password manager prototype (ByPass) and as the last step, you will be given a post-test questionnaire that mainly asks about your experience working with ByPass.

- Starting the tasks: We developed a password manager called "ByPass" and by presenting this user study we want to evaluate our prototype, not you! So, if you face any errors during the experiment don't feel bad, it's not your fault. It's a problem on our side. Please note that all the needed credentials to complete the tasks are provided for you and please do not use any of your own credentials because we can see them.

First, I start by giving a short description on what a password manager really is! Password manager is a software application that is used to store and manage passwords that the user has for different online accounts, e.g., Gmail, Facebook, Amazon, etc. Password managers can generate unique passwords for you and store them in a secure format.

Password managers may also include a “password generator”. This is a software tool that creates random or customized passwords for the user.

ByPass is a password manager that automatically generates passwords for the user. You can use ByPass to create accounts and log in directly to websites. ByPass communicates securely with websites so that you can avoid the login page.

We are testing our password manager on two websites. As you can see on your browser. One of them is called Amazon and the other one is web mail. These websites are operated by us, and for the purposes of this study, you can pretend they are real service providers that you have account on them. As you may know creating an account on each website requires different information such as your name, email address, phone number and etc. So, do our websites – but we will provide you.

By considering my description, please go through the tasks that are given to you.

Assume that you just downloaded and installed ByPass on your Chrome browser. First of all, you need to create an account for yourself on ByPass to get started. Please go through the task description paper and complete task 1.

As you just travelled to Canada, you want to create an account on Amazon.ca to start purchasing stuff for your new home. Follow the steps on description paper and create an account on our Amazon website.

Then you want to see how your newly installed password manager works. To do so, you start by adding your Amazon account credentials to ByPass and try to login again to Amazon but this time through ByPass.

Perfect, now you are sure that the password manager is perfectly working and you can trust it!

Recently your manager asked you to create an email address on webmail. You think this can be a good time to use your password manager functionality, so you start by creating an account on webmail through ByPass.

After some days your feel that the chosen password for your amazon is not secure enough so you decide to change your password through ByPass. You want to make sure to choose the strongest possible password this time.

You are recently feeling some suspicious activities on your Amazon account and you decide to delete your account from Amazon and to do so, you get help from ByPass because you find it really hard to figure out where "delete my account" button is in the Amazon website!

- Well done! You have completed all the tasks!

- Post-test questionnaire: Now I'm going to give you the post-test questionnaire which is the main part of this study. We want to know how did you feel while you were working with our password manager. Please be honest in your responses, which are anonymous.

- End of the experiment Feel free to ask any question!

Appendix C

Information and Consent Form

INFORMATION AND CONSENT FORM

Study Title: Improving Password Manager Usability
Researcher: Tina Safaie
Researcher's Contact Information: t_safaie@encs.concordia.ca
Faculty Supervisor: Mohammad Mannan, Amr Youssef, Elizabeth Stobert
Faculty Supervisor's Contact Information: m.mannan@concordia.ca - youssef@ciise.concordia.ca - ElizabethStobert@cunet.carleton.ca

Tina Safaie Principle Investigator- Master's student at the CIISE department Concordia University t_safaie@encs.concordia.ca EV10.173, 1515 Ste-Catherine Street West, Montreal, QC, Canada H3G 2W1	Mohammad Mannan Faculty Supervisor Concordia University 514-848-2424 ext. 8972 m.mannan@concordia.ca EV9.189, 1515 Ste-Catherine Street West, Montreal, QC, Canada H3G 2W1

Amr Youssef Faculty Co-Supervisor Concordia University 514-848-2424 ext. 5441 amr.youssef@concordia.ca EV9.181, 1515 Ste-Catherine Street West, Montreal, QC, Canada H3G 2W1	Elizabeth Stobert Faculty Co-Supervisor Carleton University 613-520-2600 ext. 2434 elizabeth.stobert@carleton.ca HP5127, 5302 Herzberg Laboratories, Carleton University, Ottawa, ON, Canada K1S 5B6

Source of funding for the study:

You are being invited to participate in the research study mentioned above. This form provides information about what participating would mean. Please read it carefully before deciding if you want to participate or not. If there is anything you do not understand, or if you want more information, please ask the researcher.

A. PURPOSE

The purpose of the research is to investigate and improve the usability of password managers. We are interested in how we can make a password manager usable so that more people tend to use it. We have designed and constructed a password manager prototype that integrates several new features intended to help usability, and in this study, we are evaluating those features.

B. PROCEDURES

During this study, we will ask you to complete tasks on the password manager prototype as if you were using it for your own accounts. These tasks are similar to typical account tasks you may regularly handle, such as creating new accounts, logging in, and changing passwords. You will not be asked to share any of your own credentials, or use any information relating to your own accounts.

You will be asked to fill in three questionnaires: a pre-test questionnaire that asks questions about your security behavior, a demographics questionnaire, and a post-test questionnaire that asks about your impressions of the password manager.

In total, participating in this study will take around 1 hour.

C. RISKS AND BENEFITS

This research is not intended to benefit you personally.

D. CONFIDENTIALITY

We will gather the following information as part of this research:

- Demographics data

- Your responses to the pre- and post-test questionnaires.
- Instrumented data recorded in the software that collects data on which buttons are pressed, which keys are typed, and the length of time it takes to complete tasks.

By participating in this study, you agree to let the researchers have access to information about your demographics and your responses to our questionnaires.

We will not allow anyone to access the information, except people who are directly involved in conducting the research. We will only use the information for the purposes of the research described in this form.

The information gathered will be coded. That means that the information will be identified by a code, and there will be no link between your name and the code identifying your data. All data will be saved with password protection. We will keep the coded data indefinitely.

We intend to publish the results of the research. However, it will not be possible to identify you in the published results. If you would like to see a copy of the study results, notify the researcher to be sent a copy of the publication when it is available.

F. CONDITIONS OF PARTICIPATION

You do not have to participate in this research. It is purely your decision. If you do participate, you can stop at any time. You can also ask that the information you provided not be used, and your choice will be respected. If you decide that you don't want us to use your information, you must tell the researcher before study session concludes.

There are no negative consequences for not participating, stopping in the middle, or asking us not to use your information. You may withdraw your data at any time during the study, but will not be able to withdraw your data after your participation in the study has finished.

The study will take at most 1 hour to complete, and you will be paid a $15 honourarium for your time. If you choose to withdraw from the study, you will still be paid the honourarium.

G. PARTICIPANT'S DECLARATION

I have read and understood this form. I have had the chance to ask questions and any questions have been answered. I agree to participate in this research under the conditions described.

NAME (please print)_____

SIGNATURE _____

DATE _____

Note that this study has the Carleton university clearance (CUREB-B Clearance #112295).

If you have questions about the scientific or scholarly aspects of this research, please contact the researcher. Their contact information is on page 1. You may also contact their faculty supervisor.

Should you have any ethical concerns with the study, please contact the REB Chair, Carleton University Research Ethics Board-B (by phone: 613-520-2600 ext. 4085 or by email: ethics@carleton.ca). For all other questions about the study, please contact the researcher. You can also, contact the Manager, Research Ethics, Concordia University, 514.848.2424 ex. 7481 or oor.ethics@concordia.ca.

Appendix D

User Study Tasks Description

Tasks Description

1. Create an account on ByPass

   Username: ben@gmail.com
   Password: “choose something on your own but not your personal password”

2. Go to our Amazon website tab which is already open on your browser and create an account with provided credentials:

   First name: Ben
   Last name: Saleh
   Date of Birth: 1990/Feb/02
   Email: ben@yahoo.com
   Password: Rainbow123!

3. Go to ByPass and add your Amazon account (the one that you just created) and then login.

   Enter “Amazon” for website address

4. Use ByPass to create an account on webmail.

   Enter “Webmail” for website address
   Email address: ben@webmail.com

5. Change your webmail accounts’ password through the ByPass.

6. Delete your Amazon account through ByPass.

Appendix E

Pre and Post-test Questionnaires

Section A: Pre-test Questionnaire

A1. **On a scale of 1 (novice) to 10 (expert), how would you rate yourself with respect to your computer skills?**

1 2 3 4 5 6 7 8 9 10

A2. **How often do you browse the web?**

- ☐ Daily
- ☐ Several times a week
- ☐ Once a week
- ☐ Less than once a week

A3. **Approximately how many password protected accounts do you have?**

Please answer with a number

A4. **Do you sometimes re-use the same password on different web sites?**

- ☐ Yes
- ☐ No

A5. **Have you ever used a password manager? If yes, which one? If no, why not?**

A6. **Do you save your passwords in the web browser (e.g. Chrome, Firefox, Safari, Internet Explorer)?**

A7. **What criteria do you use for choosing a password?**

Select more than one if appropriate

☐ It is easy for you to remember

Comment

☐ It is difficult for others to guess

Comment

☐ It is suggested by the system

Comment

☐ It is the same as another password you currently have

Comment

☐ Other (Please specify):

Comment

A8. **How concerned are you about the security of your passwords?**

(Not at all concerned) 1 ☐ 2 ☐ 3 ☐ 4 ☐ (Neutral)5 ☐ 6 ☐ 7 ☐ 8 ☐ 9 ☐ (Very concerned) 10 ☐

A9. **If you had to create a new password for your bank account using a normal password system, how would you go about choosing a new password?**

A10. How do you remember your passwords?

Select more than one if appropriate

- ☐ Re-use same password
- ☐ Write them down on a paper
- ☐ Write them on the file on computer
- ☐ I use password manager
- ☐ The browser saves them for me

A11. How often do you change your passwords?

- ☐ Monthly
- ☐ Every 6 months
- ☐ Once a year
- ☐ I only change my password if I have to

A12. Have you ever installed a browser extension?

- ☐ Yes
- ☐ No

A13. Which one are you more comfortable to use?

- ☐ Software as an extension
- ☐ Software as an application

Section B: Demographics question

This information will be held completely confidential. All questions are optional.

B1. How old are you?

B2. Sex:

Check one

- ☐ Male
- ☐ Female
- ☐ Non-binary/Other-identifying gender
- ☐ Prefer not to say

B3. **What is your highest level of study?**

- ☐ High School
- ☐ Undergraduate
- ☐ Graduate
- ☐ Professional degree
- ☐ College
- ☐ Others

B4. **What is your occupation?**

B5. **(If you are a student) In what academic program are you enrolled? / (If you are working) what was your major field of study?**

Section C: Post-test Questionnaire

C1. **It was easy to understand what each button does in ByPass.**

Strongly Disagree1	2	3	4	Neutral5	6	7	8	9	Strongly Agree5
☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

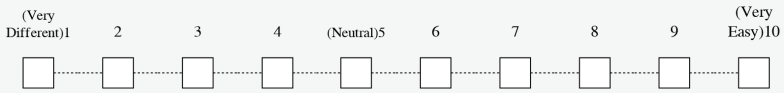
C2. **I needed to learn a lot of things before I could get going with ByPass.**

(Strongly Disagree)1	2	3	4	(Neutral)5	6	7	8	9	(Strongly Disagree)10
☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

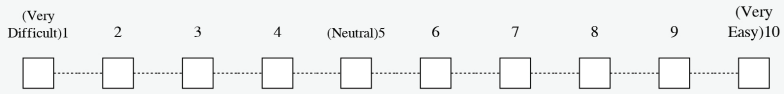
C3. **How easy was it for you to create an account on ByPass?**

(Very Difficult)1	2	3	4	(Neutral)5	6	7	8	9	(Very Easy)10
☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

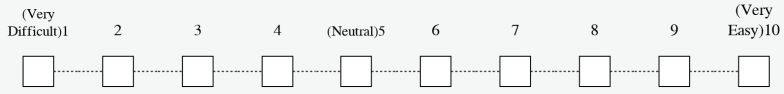
C4. **How difficult was it for you to choose a secure master password on ByPass?**



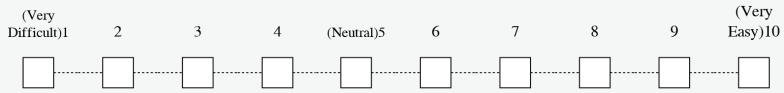
C5. **How easy was for you to create a new account on "Amazon" website through ByPass?**



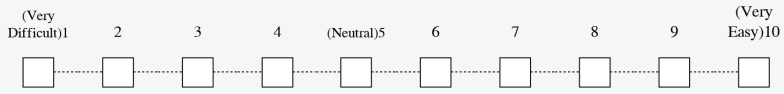
C6. **How easy was it for you to add your existing account to ByPass?**



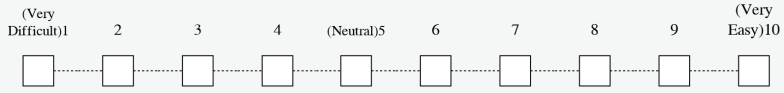
C7. **How difficult was it for you to change one of accounts' passwords using ByPass?**



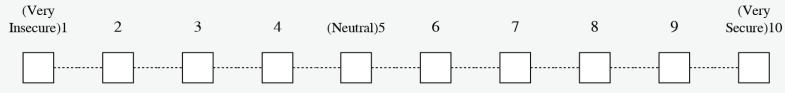
C8. **How easy was it for you to delete your account from "Amazon" website through ByPass?**



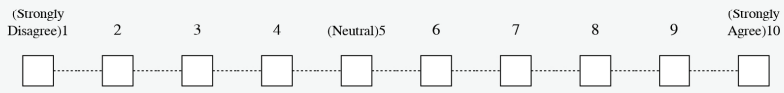
C9. **How easy was it to use the password manager as a browser extension?**



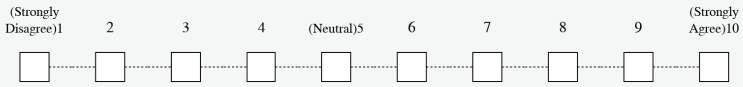
C10. **How secure did you feel ByPass is?**



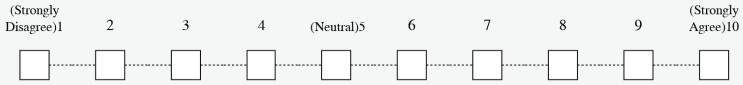
C11. **For creating a new account, I liked it when the password manager automatically filled the password box for me.**



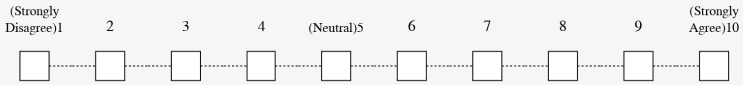
C12. **Creating an account through ByPass was easier than creating an account in the regular way. (e.g. by visiting the website directly)**



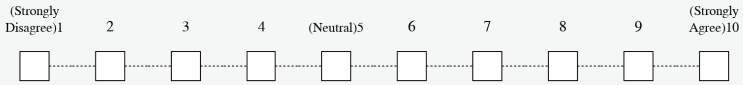
C13. **I'm comfortable with not knowing my actual password for a website.**



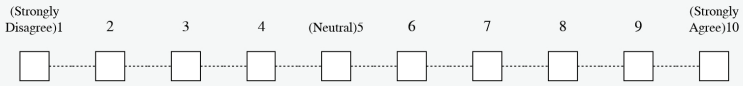
C14. **I like the function that automatically changes the password each month.**



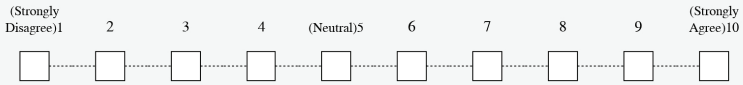
C15. **I feel my passwords are secure while using ByPass.**



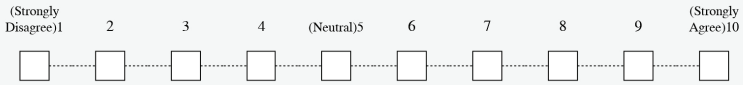
C16. **I would ByPass if it become available online.**



C17. **I think most people would learn how to use ByPass very quick.**



C18. **Overall, how would you rate ByPass?**



Appendix F

Recruitment Poster

Earn 15$ for FREE

Looking for a user study Participation

Carleton UNIVERSITY

Université Concordia University

Improving Password Manager Usability

We are conducting a user study of our prototype password manager. This is to examine how our software is working and how comfortable are the participants with using our password manager. The study lasts about 1 hour, and we pay $15.

Note This study has been approved by both Concordia university ethics committee (approval # 30012570) and Carleton University Research Ethics Board-B (CUREB-B Clearance #112295).

To participate in this study, you must:

- Be used to entering username and passwords to access websites
- Be aged 18 or over
- Speak English

Please contact Tina Safaie for more details at **t_safaie@encs.concordia.ca**

1.Open your camera

2.Scan the QR Code

Appendix G

Ethics Certificate

CERTIFICATION OF ETHICAL ACCEPTABILITY FOR RESEARCH INVOLVING HUMAN SUBJECTS

Name of Applicant:	Tina Safaie
Department:	Faculty of Engineering and Computer Science\ CIISE
Agency:	N/A
Title of Project:	Improving Password Manager Usability
Certification Number:	30012570

Valid From: January 31, 2020 To: January 30, 2021

The members of the University Human Research Ethics Committee have examined the application for a grant to support the above-named project, and consider the experimental procedures, as outlined by the applicant, to be acceptable on ethical grounds for research involving human subjects.

Richard DeMont

Dr. Richard DeMont, Chair, University Human Research Ethics Committee