BluePass: A Mobile Device Assisted Password Manager

Yue Li[1,*], Haining Wang[2], Kun Sun[3]

[1]Department of Computer Science, College of William&Mary, yli@cs.wm.edu
[2]Department of Electrical and Computer Engineering, University of Delaware, hnw@udel.edu
[3]Department of Information Sciences and Technology George Mason University, ksun3@gmu.edu

Abstract

With the growing number of online accounts a user possesses, managing passwords has been unprecedentedly challenging. Password managers have emerged to help users managing their passwords. However, state-of-the-art cloud based password managers are vulnerable to data breach and a master password becomes a single point of failure. To address these security vulnerabilities, we propose BluePass, a password manager that stores the password vault (i.e., the set of all the encrypted site passwords of a user) locally in a mobile device and a decryption key to the vault in the user computer. BluePass partially inherits the security characteristics of two-factor authentication by requiring both a mobile device and a master password to retrieve and decrypt the site passwords. BluePass leverages short-range nature of Bluetooth to automatically retrieve site passwords and fill the login fields, providing a hand-free user experience.

Received on 20 November 2018; accepted on 04 December 2018; published on 21 December 2018

Keywords: Authentication, Password, Password Manager, Two-factor Authentication

doi:10.4108/eai.10-1-2019.156244

1. Introduction

Text-based password still dominates online authentication despite that it has long been plagued by a well-known and long-standing problem: the wide use of weak password. Due to limited human memory, users tend to choose weak passwords as they cannot remember strong passwords. However, weak passwords are easy to guess and thus are vulnerable to various password cracking attacks [4, 29, 32, 42, 44]. Though numerous alternatives have been proposed to replace text-based password authentication, none of them is able to surpass text-based password in every aspect [5]. Facing the dilemma that passwords cannot be easily replaced in the foreseeable future, system administrators have attempted to enforce users to select more secure passwords such as longer passwords with different types of characters, as well as suggest users to choose a unique password for each of their accounts and frequently change them [6].

As these password enforcement policies create serious memorability hassle for users, they seldom follow the most secure practice [3, 6]. For example, on average users may reuse one password for as many as 3.9 accounts [15]. Therefore, instead of impractically expecting users to select a strong password for each online account, password managers are developed as built-in or standalone tools to help users manage their passwords. A password manager includes a vault that stores all the user's passwords, and the user only needs to remember one master password to access all the passwords in the vault. To support user authentication on different devices, password managers usually synchronize the vaults to their own servers and provide a downloading service to their users. However, a password manager has its own security and usability problem. For example, password managers usually synchronize the local vault to the remote server, which makes data breach possible [24]. Furthermore, to enhance usability, many browser built-in password managers do not necessarily need a master password, which makes it vulnerable to unauthorized use and sacrifices portability. Even being used, a

*Corresponding author. Email: yli@cs.wm.edu master password becomes a single point of failure. The user interface of a password manager should also be carefully designed to keep it simple and user-friendly. Otherwise, the usability issues of a password manager may lead to reduced security, stemming from incomplete user mental models [9].

For critical online services, users may desire more secure authentication than password-only authentication. Therefore, two-factor authentication (2FA) is proposed to include another layer of protection to user accounts. Nowadays many leading service providers such as Google and Microsoft, have integrated 2FA into their online systems. However, 2FA requires extra efforts from a user such as checking its smartphone or email to retrieve a one-time code as a secondary authenticator. In order to improve usability, transparent 2FA has been proposed [10, 33] by leveraging additional devices (mainly user smartphones) to automatically complete the enhanced authentication procedure without user involvement. Therefore, users can increase the security by using 2FA and maintain the same level of usability as password-only authentication. However, these approaches are harder to deploy because the modifications at both the web server side and the client side are imperative.

In this paper, we propose BluePass, an enhanced password manager that partially inherits the security benefit of 2FA to improve the security and usability of existing password managers. One of the key features of BluePass is to isolate the storage of the password vault from that of the decryption key. Here the password vault is the set of all the encrypted site passwords of a user. Specifically, the password vault is stored locally in a mobile device (e.g., a user's smartphone) and the decryption key is stored in the BluePass server, which can be accessed and downloaded only once to a computer after authentication through a master password. The mobile device communicates with the computer using Bluetooth in a user transparent manner. When a user needs to log in a website, the computer will automatically request the site password from the mobile device. The encrypted site password will then be delivered from the mobile device through Bluetooth. Afterwards, the computer is able to decrypt the site password using the local decryption key and auto-fill the web forms for the user.

BluePass is secure since it does not store password vaults on a server and is not vulnerable to massive password breach. Furthermore, as a password vault and its decryption key are stored separately, losing one of them will not practically leak any password. While BluePass itself uses 2FA, it does not require any modifications on the website servers. Thus, the underlying password framework remains unaltered, i.e., logging into a website still only needs one site password. BluePass is also usable since it demands little effort to configure on the computer and no extra effort from a user to authenticate afterwards. The data transmission of BluePass is fully automatic through Bluetooth after the initial configuration stage. Users are motivated to choose much more secure site passwords as they do not need to remember them, making their passwords much less prone to password cracking such as brute-force and dictionary attacks.

We implement a BluePass prototype in Android and Google Chrome platforms and evaluate its efficacy in terms of security, overhead, and usability. First, we conduct a comprehensive security analysis to demonstrate that BluePass can defend against various attacks. Then we evaluate the auto-fill latency of BluePass by recording the time between login forms being detected and the forms being automatically filled. We also run a series of experiments, in which we retrieve passwords under different frequencies, to measure the energy overhead of BluePass. Based on our experimental results, BluePass is energy efficient while automatically filling in the login forms with user-unperceived latency. Afterwards, we conduct a user study including 31 volunteers to examine the usability of BluePass. The results show that the test subjects regard BluePass as both secure and usable. Most test subjects are reported to be motivated to choose more secure passwords and to less likely re-use existing passwords by BluePass. Moreover, the majority of testers report that they are willing to use BluePass to manage their passwords.

The remainder of the paper is organized as follows. Section 2 elaborates the system overview and threat model. Section 3 details the system architecture of BluePass and Section 4 conducts security analysis on BluePass. Section 5 illustrates the prototype implementation of BluePass. We evaluate BluePass in Section 6 and present a user study in Section 7. Section 8 discusses BluePass-related issues and its limitation. Section 9 surveys related work and finally, Section 10 concludes this paper.

2. System Overview and Threat Model

Before presenting the BluePass system, we first introduce important BluePass notations for clarification purposes.

- *BluePass server*: a server that is mainly responsible for registering users and distributing keys to user computers.
- *Key pair* (K_1, K_2): a pair of RSA keys that are used by the mobile device and the computer to encrypt/decrypt site passwords. K_1 is only stored in the mobile phone while the K_2 is stored in the BluePass server for distribution. We manage to use only one pair of keys to protect bi-directional

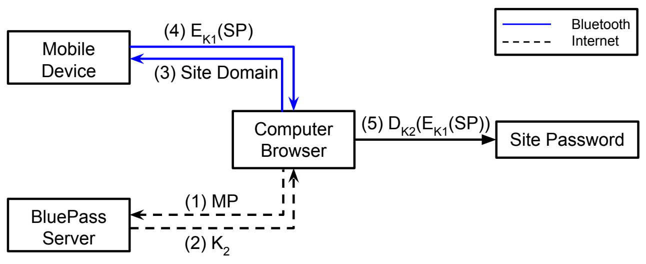


Figure 1. BluePass Authentication

Solid lines indicate that the process needs to be run whenever a site password is requested. Dashed lines indicate that it does not necessarily run when a site password is requested. In a *trusted computer*, it runs only once for a long term while in a *untrusted computer*, it runs once in each browser instance.

communication, and the details can be found in 8.1.

- *Master password (MP)*: a user uses its master password to authenticate itself to the BluePass server and retrieve its own decryption key K_2. A master password is the only password a user needs to remember.
- *Site password (SP)*: passwords to access online services, which will be encrypted by K_1 and then stored in the BluePass mobile application .
- *Trusted computer*: a computer that the user trusts, such as the user's personal computer. It stores the decryption key K_2 for a long term.
- *Untrusted computer*: a computer that the user does not trust, such as a library computer. The decryption key K_2 must be retrieved from the BluePass server every time a browser is opened in the untrusted computer. K_2 is only temporarily stored in a browser instance, and is removed when the browser instance is terminated.
- *Client-side (computer/browser) application*: the user installs it on the computer, which is in charge of detecting and auto-filling login forms, communicating with the mobile device, and decrypting the received site passwords.
- *Mobile application*: the user installs the app on its mobile device. The app stores the encrypted site passwords and delivers the encrypted site passwords to the user computer through Bluetooth.

2.1. System Overview

BluePass works with two premises. First, a site password can only be recovered by having both the encrypted site password $E_{K_1}(SP)$ that is only stored in the mobile device and the corresponding decryption key K_2 that is distributed through the BluePass server. Second, the encrypted site password $E_{K_1}(SP)$ can only be retrieved from the mobile device to the user computer through Bluetooth, which requires the proximity of the two devices. The flow chart of BluePass password authentication is shown in Figure 1.

The working mechanism of BluePass mainly includes three phases, which are detailed as follows.

Phase 1: Registration is a once-in-a-lifecycle operation, in which a user needs to register for the BluePass service. The user installs the BluePass mobile app on its mobile device and uses its master password to log into the BluePass account. The mobile device is then initialized with an empty password vault.

Phase 2: Configuration is to install and configure the user devices. First, a client-side application needs to be installed on the user computer. Then, the user will log into the BluePass server and download the decryption key K_2 into the computer. The user will store the key either for a long term or temporarily, depending on whether the computer is trusted or untrusted. Note that the installation of the client-side application on a computer is also a one-time operation. The retrieval of K_2 from the BluePass server is needed each time only when the user opens a browser from an untrusted computer.

Phase 3: Authentication is almost transparent to the user. In a trusted device, the user only needs to carry the registered mobile phone and see the passwords being automatically filled. In an untrusted device, the user needs to re-enter the master password every time a new browser instance is opened since the key K_2 is deleted when a browser instance is closed.

2.2. Threat Model

Attackers aim at stealing one or (preferably) all site passwords in the password vault. In the design of BluePass, all the site passwords of a user are encrypted and stored in the user's mobile device. We assume that the attacker cannot access the encrypted site passwords in the mobile device and knows the decryption key from the computer at the same time.

All attacks can be classified into two categories: *co-located attacks* and *remote attacks*. A co-located attack can only happen within the Bluetooth communication range of the user mobile device, while a remote attack can be launched from anywhere. In a co-located attack, since the attacker could access the encrypted site passwords, we must prevent the decryption key from falling into the hand of the attacker. Therefore, both the BluePass server and the master password cannot be compromised. Moreover, the communications for key distribution must be protected. By contrast, since

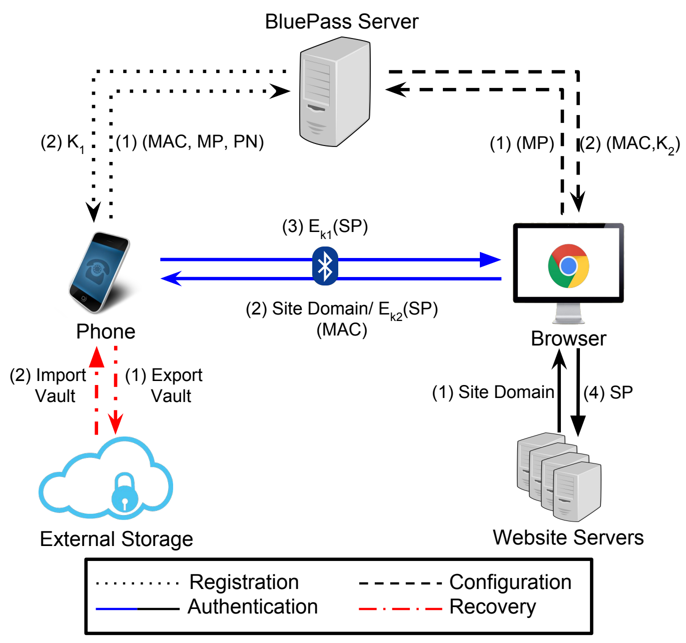


Figure 2. BluePass Architecture

the attacker cannot access the mobile device in a remote attack, either the BluePass server or the master password could be compromised. Also, no secure communication is required for key distribution. As the Bluetooth reachability is very limited (33 feet for class 2 Bluetooth devices), a co-located attack is much more difficult to launch than a remote attack.

3. System Architecture

3.1. Core Functions

As mentioned in Section 2.1, BluePass mainly consists of three phases. The first two phases, registration and configuration of BluePass, are mostly one-time effort; however, the third phase, authentication, will be triggered each time a user needs to log in a website. Figure 2 illustrates BluePass architecture and the data flow of these three phases.

Registration. The black dotted lines in Figure 2 show the registration process. To register a BluePass service, the user only needs to download a BluePass application to the mobile phone and create a master account on the BluePass server. The creation of the master account is similar to the creation of an account in any website (i.e., choosing a username and a password). However, given the critical security property of BluePass, a user should register using its phone number as another factor when the user forgets the master password or change the associated mobile device. Upon logging into the master account on the mobile app, the user can choose to associate online authentication with this mobile device. Then BluePass follows a traditional 2FA mechanism. It sends a code to the user's phone number. If the code is correctly input and verified, the device is then bound to the user's BluePass account. The device information, specifically the MAC address of the device Bluetooth, will be uploaded to the BluePass server. The MAC address is used for the client-side application to automatically locate the associated mobile device without user involvement. For a newly associated device, the BluePass Server generates a pair of asymmetric keys (K_1, K_2). It then distributes K_1 to the mobile phone and keeps only K_2 on the server side. We list the database of the BluePass server in Table 1 and that of the mobile device in Table 2 populated with made-up data. The registration should only be done once on the mobile device. After registration, the mobile device is initialized as a password vault.

Configuration. After registering for a BluePass service, the user has stored a password vault in its own mobile device. Then the computer needs to be configured to run BluePass, which is shown in the dashed black lines in Figure 2. The user installs and runs a client-side application, and then logs into the BluePass server to fetch the Bluetooth MAC address of the mobile device and K_2 generated during the registration. At this point, the user can choose whether the computer is trusted or not. If the computer is trusted, the Bluetooth MAC address and K_2 will be stored in the browser for a long term. Otherwise, they will be deleted after the user closes the current browser instance. Note that pairing the two devices in Bluetooth can also be automatic by using RFCOMM insecure mode, in which the Bluetooth data is broadcasted and the target MAC address is specified in the data.

Authentication. The solid lines in Figure 2 show the data flow of BluePass authentication process. First, the user directs the browser to a website it wants to login and the website server responds to the user with the page requested. The BluePass client-side application will examine the Document Object Model (DOM, which is a tree structure representing the webpages) of the page and check the existence of a login form. If a login form is present, the extension requests the corresponding credentials from the mobile phone using Bluetooth. After receiving the request, the mobile application returns the encrypted credentials. If no related credential exists, BluePass will instead respond with a "NO_PASSWORD" flag. We realize that auto-filling in a non-HTTPS environment is vulnerable to JavaScript injection attacks [40]. So, we only do auto-filling for websites that are based on HTTPS. For other websites, BluePass will pop up a window for a user's consent before filling the login form. Note that none of the above steps require any user interactions. This fully automated authentication enables users to login a website in a hand-free manner. When there exists more than one account for a specific website, since there is no

Table 1. Server Side Data

Username	Salt	$H(MP + Salt)$	K_2	Device MAC address	Phone number
Alice	ifu92@fb	$a4f3b3c9e61b838f8cda07\ldots$	$a2513fa584029e\ldots$	BC:F5:AC:9D:9A:57	12345678
Bob	01dm.a<w	$daa4a403bfec911a3ef199\ldots$	$231e927f8506d3\ldots$	BC:F5:AC:9D:9A:58	87654321

Table 2. Mobile Device Data

Domain	username	$E_{K_1}(Password)$
.yahoo.com/	aliceweb1	$Encrypted\_Password_1$
.yahoo.com/	aliceweb2	$Encrypted\_Password_2$
.google.com/	aliceweb2	$Encrypted\_Password_3$

way to predict which one the user will use, the browser will let the user choose an account to be decrypted and automatically filled in the forms.

3.2. Account Management

Account management is essential to a password manager. Users should be able to add, edit, or delete the credentials in BluePass. These functions must be correctly designed to guarantee the security of BluePass.

The addition of an online account into BluePass can be done when a user has manually inputted the login credentials into a new website. BluePass adopts a similar approach just as current browser built-in password managers. Recall that the browser will request the credentials when the login form is present in a web page. If the "NO_PASSWORD" flag is sent back, the browser knows that no login credentials are associated to this particular website. If the user manually inputs the credentials, the browser will capture the value in the form before submission and prompt a non-intrusive dialog window, asking whether the user wishes to store the login information into BluePass. Specifically, there are three options: "yes", "not this time", and "never". If "yes" is chosen, the browser will encrypt the credentials using key K_2 and send it to the mobile device (see Figure 2). The mobile device will decrypt the information using K_1 and encrypt it again using K_1. Mathematically the process is denoted as $E_{K_1}(D_{K_1}(C))$, in which $C = E_{K_2}(SP)$. Then the encrypted credentials are stored in the BluePass database.

The edition of an online account is similar to the addition process. The browser monitors if the user has modified the value in the login form when being submitted. If the password is changed, the browser will prompt a dialog that asks for user permission to update the login credentials in BluePass. Upon user consent, the browser will send the updated values in an encryption and decryption procedure similar to that of adding a new account. Note that the chosen option of "never" should also be recorded in the password vault, which prevents the dialog from prompting repeatedly. In this case, the password vault records the domain name and the username without storing a password. When an empty password is passed back, the application is acknowledged that the user does not wish to store the login credentials. The revocation of the "never" status can be done in the administration page in the mobile applications.

The deletion of login credentials can also be done on the mobile application's administration page. The mobile application shows a list of websites whose site passwords are stored in the mobile phone. The user can choose to delete one of the websites' login credentials. However, the user needs to manually input the website's URL and login credentials. Before the deletion is granted, the user must input the correct master password. This will prevent an attacker from manipulating the user's online accounts.

3.3. Recovery

When using a centralized password manager, users can back up their password vaults on the server side. However, BluePass does not rely on a centralized server to back up and synchronize all the passwords to prevent single point of failure. Instead, it stores local copies on mobile devices. Though users usually do not lose their mobile devices quite often, it is essential for BluePass to back up and recover the password vault when the mobile devices are lost, which is illustrated with red lines in Figure 2.

Users can choose to back up their vaults to an external storage including a portable hard disk, a USB, or a cloud storage. If a user loses the mobile device, it can recover the vault from the external storage. Backing up the vault to a user-owned physical device may require the user to periodically back up and synchronize the password vault to the external storage device. Alternatively, BluePass allows users to synchronize their password vaults to a cloud drive provider. Nowadays many large drive providers, such as Google Drive or Dropbox, have published APIs to facilitate data synchronization. Note it still ensures the 2FA design of BluePass. Specifically, an attacker needs to breach both the BluePass Server and the cloud provider server to collect the two necessary pieces of secret. A more detailed discussion on cloud synchronization is provided in Section 8.4. On the other hand, security sensitive users may still choose to manually synchronize their vaults to a USB or a hard drive that is solely controlled by themselves.

4. Security Analysis

BluePass is secure in a sense that as long as a user does not lose two factors at the same time, the user's login information is safe. We conduct a security analysis on BluePass to verify the robustness of BluePass against various attack vectors.

4.1. Two-Factor Security

We have introduced that BluePass relies on the premise that two factors need to be possessed to derive a site password. The two factors are user mobile device and a master password. Now we discuss the security of BluePass when one of the factors is compromised.

Master password. An attacker may be able to compromise the master password of a user, which can be done through different ways such as guessing, phishing, shoulder-surfing, etc. The compromisation of a trusted computer is also equivalent to losing the master password because the attacker can directly extract K_2 from the trusted computer. In such scenarios, the attacker is able to obtain key K_2. However, if the attacker does not have the password vault of the user, K_2 is merely a meaningless long string and the security of BluePass holds. Besides, the user is able to change the master password and re-generate a new key pair.

Mobile device. If an attacker gains access to the mobile device by either compromising the device or stealing the device, it may be able to access the encrypted password vault and the encryption key K_1. If backing up on cloud is enabled, an attacker may also access the encrypted password vault by breaching the cloud provider. However, without the decryption key K_2, the attacker cannot decrypt the site passwords from the encrypted password vault. Moreover, the mobile phone itself may have its own protection, such as an unlock code or fingerprint verification. Nowadays It is also not uncommon that the mobile phone is associated with an online account, through which the user can de-activate the mobile phone and delete the stored data.

4.2. Data Breach and Brute-force Attacks

A serious threat to a password manager is data breach. There have been many reported extensive data breach incidents, resulting in significant financial losses and private information leakage. Currently there is no guarantee that an online service would be immune to data breach. Therefore, any online service could be a victim. When a data breach happens in BluePass, the attacker may steal the database of the BluePass server. Under this scenario, the attacker may be able to mount a brute force attack against the master password of a user or directly disclose the decryption key. Note that we recommend a user to choose a very strong master password against offline brute force attacks. Moreover, the BluePass server can leverage multiple techniques, such as key stretching or honey encryption [8], to make a brute force attack much more difficult to succeed. Even though, we do not assume that a master password is safe since it is reported that many web systems are not or wrongly following the best practice [6].

In a normal password manager such as LastPass or 1Password, the loss of a master password also means the loss of an entire password vault, especially when the data is breached. When an attacker successfully mounts a brute force attack against the master password, it can also retrieve all the passwords from the password vault. Unlike these password managers, BluePass does not centralize the password vault storage. Instead, the password vault of a user is stored locally in its own mobile device. A server data breach would at most leak the user master passwords and then further leak the decryption keys. However, as the password vault of each user is not stored at the BluePass server, a data breach at the BluePass server cannot break BluePass.

Assuming that a password vault is lost from a user's mobile device, we believe that brute-force cracking such an encrypted password vault is impractical given the current computing power. We emphasize that the password vault is protected by K_1, which is 2048-bit long randomly generated RSA key. Cracking K_1 is much harder than cracking a master password, which is generated by a human user within limited and predictable password space.

4.3. Broken HTTPS or Bluetooth

If an attacker compromises the HTTPS communication, it will be able to steal the encryption/decryption key pair (K_1, K_2) of a user. However, K_1 and K_2 are only transmitted through the web when a user installs BluePass on its mobile device (K_1) or when the user log on BluePass from a new computer (K_2), which makes the attack strictly time sensitive. Even though, having the key pair does not help the attacker to identify any of the user's site password, unless the attacker can also eavesdrop on the Bluetooth connection (i.e., co-located attack) to capture the encrypted password in transmission. On the other hand, eavesdropping Bluetooth alone does not compromise BluePass either, since the content is encrypted.

To succeed, the attacker needs to compromise both HTTPS and Bluetooth communications to steal site passwords from users. However, such a successful attack is very difficult to launch, due to time (to steal the keys) and location (to eavesdrop the Bluetooth) constraints. Furthermore, a large scale attack is infeasible since Bluetooth signals can only be sniffed within a short range.

5. Implementation

BluePass consists of three major components that cooperate with each other on user authentication, namely, a BluePass server for user registration and key distribution, a BluePass client application on the laptop for detecting and auto-filling the website login forms, and a BluePass app on the mobile phone serving as the password vault and administration console. We build the BluePass client application in a Macbook Air running OS X 10.10.4 and Chrome 46.0.2490.80. We implement the BluePass app on a Nexus 5 running Android version 4.4.2.

5.1. BluePass Server

We implement a BluePass server using Cherrypy[1], a python web framework. We use self-signed certificate in https to protect communication. The key pair (K_1, K_2) is generated using Pycrypto[2] on the server side. We use sqlite database to store user data (see Table 1 for detail). When registering to the service, we do not use the standard 2FA to verify the phone number since it is not necessary for evaluation and user study. After registration, the user needs to log in BluePass on both mobile application to upload mobile phone Bluetooth MAC address and download K_1 and client-side application to download K_2 and Mobile phone Bluetooth MAC address.

5.2. BluePass Client-side Application

We build the BluePass client-side application on Chrome platform, which consists of 2 modules: one Chrome application for Bluetooth communication and one Chrome extension for password auto-filling. We use two modules because currently Chrome extension does not support Bluetooth API while Chrome application does. However, only Chrome extensions allow reading and modifying the DOM of web pages, which unavoidably makes us separate client-side application functionality into 2 modules. Chrome application is more like a native application, but it is built on Chrome platform to deliver content in HTML, CSS and Javascript (e.g., Google Doc, Google Drive). It uses the *chrome.Bluetooth* API to connect to the Bluetooth device and then communicate with the smart phone through Bluetooth. The Chrome extension is responsible for detecting the authentication form and automatically fill the form after decrypting the site password from the mobile application.

The communication between the Chrome application and the chrome extension is implemented through Chrome External Messaging[3]. Specifically, this extension specifies the Application ID, which is a unique identifier for the Chrome application. After Chrome extension delivers the data to the application that is binded to the ID and has a pre-added listener, the listener can extract the data. The communication from Chrome application to Chrome extension works similarly.

Our prototype implements the BluePass client on the Chrome platform to simplify the communications among different modules; however, the framework of BluePass can be widely deployed on more platforms as long as both the computer and the mobile device have Bluetooth support and the browser extension is able to communicate with local applications on the computer. First, Bluetooth has become a standard device on modern computers and smartphones. Second, communication between browser extensions and native applications has been supported by most modern browsers, including Internet Explorer, Chrome, Firefox, Safari, Opera, etc.

5.3. BluePass Mobile Application

The BluePass mobile application starts a BluePass service, which runs in the background of Android and has a dedicated thread to listen to the incoming Bluetooth connection, which helps transparently authenticate a user to a registered website. The BluePass service inherits from the *Service* class in Android and keeps running until the user explicitly stops the service.

BluePass mobile application has a simple and clear user interface, which shows the status of the background BluePass service, either “running” or “suspended”. The user can easily change the service status by clicking “Start BluePass Service" or “Stop BluePass Service" buttons. When the service status is running, the Bluetooth listener starts listening and remains active even the mobile device turns off the screen and goes to sleep. Whenever users would like to stop the service, they just need to open the application and click the “Stop BluePass Service" button.

We use RFCOMM Bluetooth protocol to establish communication between the mobile phone and the computer, since RFCOMM is widely supported and provides public APIs in most modern operating systems. Android supports two modes of RFCOMM connections, *secure mode* and *insecure mode*. The secure mode requires successful pairing before any RFCOMM channel can be established while the insecure mode allows connection without pairing two devices. Secure mode RFCOMM adds another layer of encryption. However, as BluePass communication is secured by

[1] http://www.cherrypy.org/
[2] https://www.dlitz.net/software/pycrypto/
[3] https://developer.chrome.com/extensions/messaging

(K_1, K_2) so that it does not rely on Bluetooth security. While insecure mode may fit better since it saves a pairing step from the user, Chrome application does not support insecure RFCOMM communication due to security concern. Therefore, we use the secure RFCOMM connection mode. Consequently, in the registration phase, the user also needs to pair the mobile phone and the computer first if they have never been paird before. Note that pairing only needs to be done once in a computer unless the user manually deletes paird devices on the mobile phone or computer.

6. Evaluation

We first evaluate BluePass as an authentication scheme using the framework developed by Bonneau et al. [5] and compare it with other related schemes. Then we measure the time latency introduced by BluePass in the authentication process. Moreover, we analyze the power consumption of BluePass, especially on the mobile device. And finally we discuss the reachability of BluePass.

6.1. Comparative Evaluation Framework

We use the comparative authentication scheme evaluation framework [5] to compare BluePass with other related authentication schemes. The results are summarized in Table 3. We can see that BluePass is *physically-effortless* since the entire authentication process is transparent to the user and *Quasi-Nothing-to-Carry* since users still need to carry their mobile phones though they carry them anyway. BluePass is accessible since it does not require the cellphone to have signal or cellular data. BluePass is *Quasi-Resilient-to-Throttled-Guessing* and *Quasi-Resilient-to-Unthrottled-Guessing*. Although BluePass itself does not enhance the security of the underlying password mechanisms, it can help defend throttled and unthrottled guessing by generating long random passwords for users and motivating users to use more secure passwords since they do not need to remember the passwords.

Bonneau et al. [5] points out that the framework does not describe all possible properties of an authentication scheme. Besides these factors, BluePass also keeps a simple and clean user mental model, which is highly suggested since wrong mental models easily make user passwords weaker [9]. Furthermore, BluePass strengthen usability by not requiring users to delete their password traces after use on a untrusted computer as other password manager (e.g., log out master account or delete local password vault).

6.2. Password Auto-fill Latency

For a usable password manager, the time required to fill the password field should be short. We record the delay between the time that the password input form is detected and the time that the form is automatically filled (denoted as T_{bp}). Since the delays on different websites may be different due to the specific website design, we choose 20 major providers from Alexa Top 100 website[4]. For each site, we make up a username/password pair and test the pair of credentials for at least 50 times. The password of each site is a randomly generated 16 byte string composed of all 4 characters types (Uppercase character, lower case character, digit, and special character). Note that the username and password do not need to be legitimate login information for the website since it does not impact the experiment result of the transmission latency.

Besides the Bluetooth communication latency, we also measure the loading time (denoted as T_{load}) for a website since page loading and Bluetooth communication tasks are running in parallel, indicating that the actual latency a user is experiencing is at most $T_{bp} - T_{load}$, which is the time difference between BluePass running time and page loading time. T_{load} is measured by injecting a piece of javascript code as follows. *DomContentLoadedEventEnd* measures the time that all javascripts that need to run immediately are being executed, and *navigationStart* returns the time that the browser is ready to send the HTTP request[5].

```
<script>
var Tload = window.performance.timing.\
domContentLoadedEventEnd
- window.performance.timing.navigationStart
</script>
```

The results for all 20 sites are shown in Figure 3. Figure 3 does not show the T_{load} results for two web services, Tumblr and mail.ru, that have much higher T_{load} (averaged 2,700-2,800 *ms*). Generally the BluePass delay time (T_{bp}) is slightly higher than the page loading time (T_{load}). To illustrate the extent of the time gap, we show statistical analysis in Table 4. In the last two rows, we do not include Tumblr and mail.ru in our analysis since they have significantly higher T_{load} that are not representative for normal cases. With the two sites excluded, the average T_{bp} is 814.6 *ms*, which is short enough to be acceptable by most users. Furthermore, the actual delay that a user experiences is $T_{bp} - T_{load}$, which is only 181.1 *ms* in average. The standard deviation for T_{load} is higher than T_{bp}. The loading time T_{load} could be different under various factors, such as network condition, website implementation, etc. Since T_{load} highly depends on the website implementation, heavy javascript use in a site could largely contribute to a high T_{load}.

[4]http://www.alexa.com/topsites

[5]https://developer.mozilla.org/en-US/docs/Web/API/PerformanceTiming

Table 3. BluePass Scheme Evaluation

Scheme	Usability								Deployability						Security										
	Memorywise-Effortless	Scalable-for-Users	Nothing-to-Carry	Physically-Effortless	Easy-to-Learn	Efficient-to-Use	Infrequent-Errors	Easy-Recovery-from-Loss	Accessible	Negligible-Cost-per-User	Server-Compatible	Browser-Compatible	Mature	Non-Proprietary	Resilient-to-Physical-Observation	Resilient-to-Targeted-Impersonation	Resilient-to-Throttled-Guessing	Resilient-to-Unthrottled-Guessing	Resilient-to-Internal-Observation	Resilient-to-Leaks-from-Other-Verifiers	Resilient-to-Phishing	Resilient-to-Theft	No-Trusted-Third-Party	Requiring-Explicit-Consent	Unlinkable
Password			●		●	●	❍	●	●	●	●	●	●	●		❍						●	●	●	●
Firefox (with MP)	❍	●	❍	❍	●	●	●		●	●	●		●	●	❍	❍					●	●	●	●	●
LastPass	❍	●	❍	❍	●	●	●	❍	●	❍	●		●		❍	❍	❍	❍		❍	●	●		●	●
Tapas	●	●	❍	❍	●	❍	●		❍	●	●			●	●	❍					●	●	●	●	●
BluePass	❍	●	❍	●	●	●	●	❍	●	●	●			●	●	❍	❍	❍		❍	●	●	●	●	●
Sound-Proof			❍		●	●	❍	❍	●	●		●		●	❍		●	●		●	●	●	●	●	

● indicates that the scheme fully carry the characteristic and ❍ indicates that the scheme partially carry the characteristic (the Quasi prefix). We take rows 1-3 from [5], row 4 from [30], and row 6 from [33].

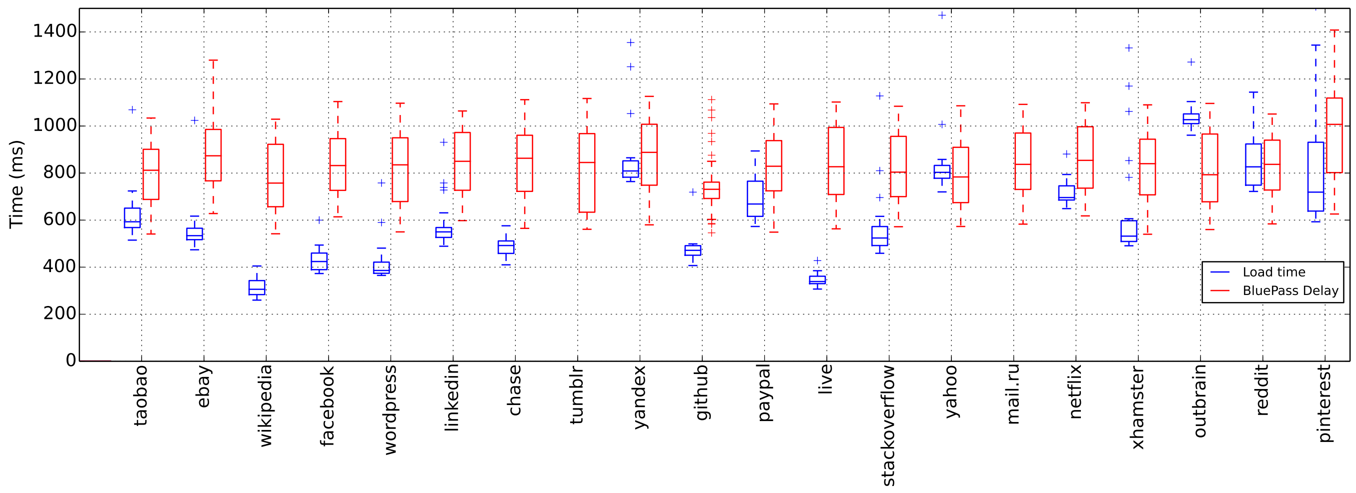


Figure 3. BluePass Latency

Table 4. Delay Statistics

	Median	Mean	SD	Skewness
T_{bp}	778.0	814.6	158.3	1.6
T_{load}	599.5	837.6	691.3	2.6
T_{bp} (removed)	775.0	812.5	155.2	1.6
T_{load} (removed)	570.0	631.4	259.8	2.6

On contrast, T_{bp} is relatively stable since Bluetooth communication and mobile device computing are almost the same in each login attempt. Since the delay caused by BluePass is bounded by $T_{bp} - T_{load}$, BluePass imposes a very low latency on the password auto-filling process. According to our user study, users can hardly notice the latency.

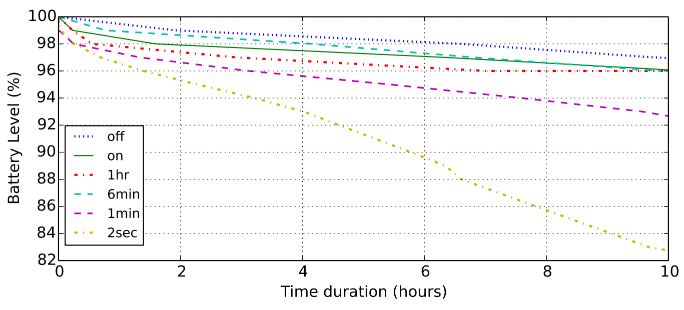


Figure 4. BluePass Power Consumption

6.3. Power Consumption

One major concern of BluePass usage is the power consumption overhead on the mobile device, since BluePass requires the mobile device serve as a Bluetooth server that keeps listening to incoming connections. We measure the extra power consumption imposed by BluePass through monitoring the power levels of the mobile device when running BluePass password retrieval process in different frequencies. Furthermore, though it is commonly believed that Bluetooth does not consume much battery when it is not being used to transmit data, we record the power level of the device when Bluetooth is turned off (we call it a clean state) to show a complete power consumption analysis.

To monitor the current battery level of the mobile device, we register a broadcast receiver in a simple battery monitoring application on the mobile device to listen to battery level changing event, upon which the current battery level and the timestamp are recorded. We also tune the login frequency in the browser side (by refreshing a webpage in different frequency) to evaluate different use cases. As the password retrieval process is almost the same when logging into all websites (since the password is encrypted using RSA, the data size is the same for all passwords), we only need to test BluePass on one website, which we choose Facebook in our experiments.

Except for the login frequency and BluePass on/off status, we keep all other settings exactly the same, such as installed and running application on the device as well as the network status (e.g., Wifi connection is turned off). We use a Nexus 5 mobile phone for evaluation, which has 2100 mAh battery capacity. As it takes a long time to use up the battery that has been fully charged, we run each experiment for 10 hours before charing the phone and running the next experiment. Though the granularity of battery usage broadcasting is in percentage level that may not be highly accurate, it is sufficient to evaluate the power efficiency of BluePass in a 10-hour test period.

Figure 4 illustrates the battery level dynamics through time under different experiment setups when BluePass is turned on or off. The reasonable frequency of login attempted by a normal user should not exceed 100 times a day, which means that the login frequency should lies around 0-10 times per daily hour. Logging 0 time per hour, which indicates that the Bluetooth only keeps listening, consumes 1% more power than a clean state. Furthermore, With 10 logins per hour, the power consumption is still only 1% more than a clean state. We believe it is an unnoticeable overhead for users, given that almost 90% of users charge their phone more frequently than once per 2 days[6]. Figure 4 also shows the power consumptions when the user tries to login every 2 seconds, 1 minute, and 1 hour, respectively. We can see that login very frequently (every 2 seconds) consumes a noticeable 17% power during the 10 hours, which is a significant overhead. However, we argue that normal users would not try logging in at such a high frequency.

In our experiments, the mobile phone is in a state that does not receive cellular or wifi signal, so the battery drains very slowly. When the mobile phone is in normal daily usage, the battery usage becomes much higher. However, the BluePass power consumption remains the level of 1% of total power with 10 hours use.

6.4. Reachability

The communication range between the mobile phone and the computer should not be either too long or too short. A long communication range makes BluePass less secure due to easier unauthorized access of the computer and larger packet sniffing area. On the other hand, a short communication range degrades the usability since users need to attentively place their mobile phones close to the computer for BluePass to work. With Bluetooth communication, BluePass functions well within up to 33 feet range (may be shorter in practice), which enables a user to leave the mobile phone in living room while using BluePass in bedroom. Furthermore, the range is sufficiently short to significantly mitigate unauthorized use of device attacks and packet sniffing attacks.

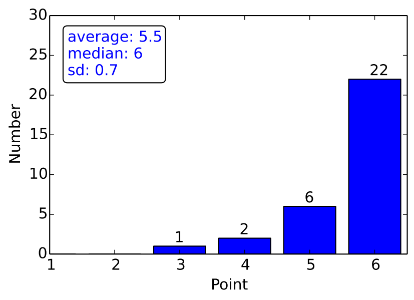
7. User Study

To verify how real users rate the security and usability of BluePass, we conduct a user study to gather feedback and comments from normal users. Upon approval of IRB of our institution, we recruit 31 volunteers to use and comment on BluePass. We ask each of the volunteer to complete a series of tasks. Afterwards, they are asked to finish a short questionnaire, surveying their perspective towards BluePass. Faulkner [13] suggests that a user study of over 20 people should be able to identify the usability problem with a high percentage certainty. Our user study should show representative and comprehensive user feedbacks.

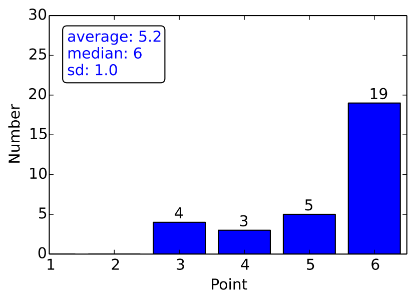
7.1. Methodology

Other than the BluePass website, we deploy a test site for our testers to register and log in. The test site is an imitation of other web services such as Facebook and LinkedIn. The site has only three main functions: registering a new account, logging in/out, and changing current password. We use our own deployed test site for ethical concerns because we do not want our testers to create dummy accounts in some real web services. We download and install BluePass applications on both the computer and the Android phone before asking the test subjects to use them. We do not require test subjects to

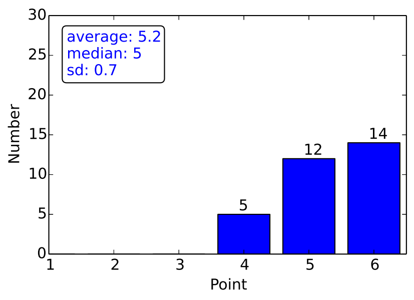
[6]http://lifehacker.com/how-often-do-you-need-to-charge-your-smartphone-1441051270



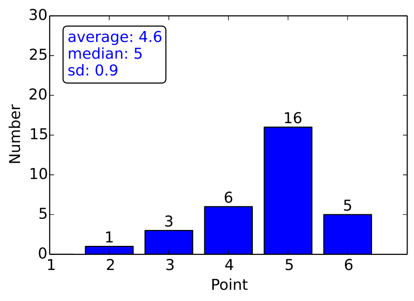
(a) Understandable



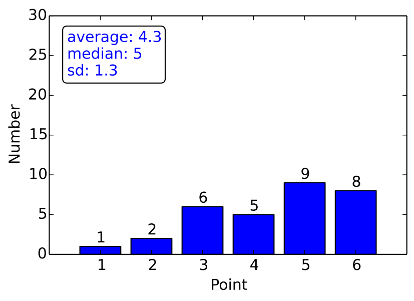
(b) Easy to set up



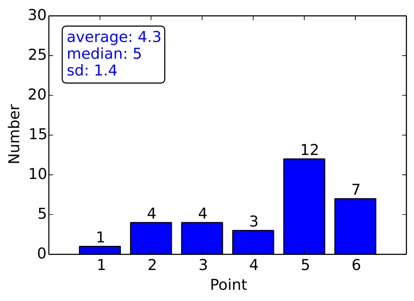
(c) More secure than other PM



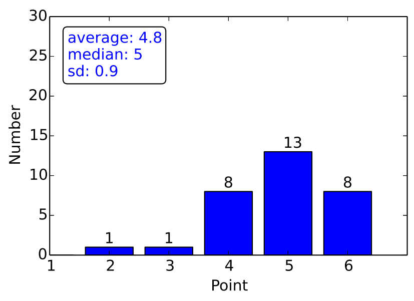
(d) More usable than other PM



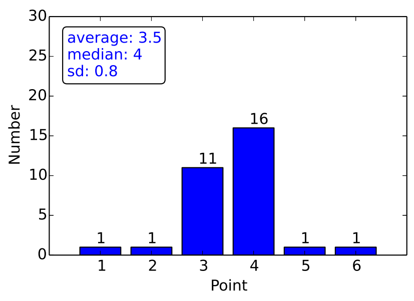
(e) Motivate more secure password



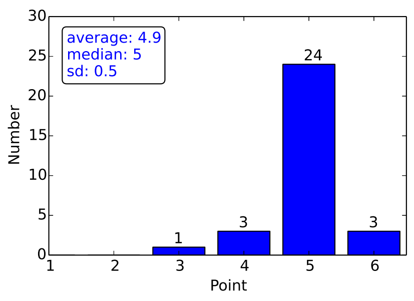
(f) Motivate less re-use password



(g) Willingness to use



(h) Rate other PMs



(i) Rate BluePass

Figure 5. Survey Results

complete this task since it is a routine step that many other password managers also require. Both the browser and the mobile phone are under factory settings except for having BluePass installed.

The computer and the Android phone are in the same room. It is well known that a class 2 Bluetooth (which is commonly found on mobile device) can reach 33 feet distance; though the reachability of Bluetooth could be affected by environmental factors, it can still reach over 20 feet [14, 45], which is sufficient for normal use cases.

First we present the volunteers a one-page explanation on what BluePass is and how it works. Then we ask them to complete a list of tasks, which are briefly summarized as follows.

1. Register to BluePass server and configure BluePass.
2. Create a new account in our self-deployed test site.
3. Log in the test site (Migrate password).
4. Try using BluePass to log in again (Log in from a primary computer).
5. Change the current password and try using BluePass to log in (Change Password).
6. Configure BluePass in another computer and log in (Log in from another computer).
7. Turn off BluePass and try logging in.
8. Turn on BluePass and try logging in.

During the study, we closely monitor how the testers use BluePass and try to identify potential usability issues. We aim to test BluePass usability in four categories as suggested in [9], namely, migrating accounts, logging in from a primary computer, changing passwords, and logging in from other computers. We also present the testers a security taste of BluePass by letting them stop BluePass service on the mobile phone and try logging in. The testers are then asked to take a short questionnaire, answering 10 6-point scale questions to present their opinions on BluePass. Each user study takes approximately 30 minutes to complete.

7.2. Demographics

This user study constitutes 31 volunteers, which include 16 males and 15 females. As the study is only in a school scale, most of the volunteers age 20 - 30 years old. Besides, most of them have or are pursuing a bachelor or master degree. Three volunteers have Ph.D degrees. In order to spread our study to people of different computer expertise, we recruit volunteers from a broad range of background, including law, education, history, statistics, biology, economics, physics, mathematics, music, and computer science.

7.3. Results

After the study, all testers have a relatively accurate understanding of how BluePass works. Four volunteers mention that they do not understand how asymmetric encryption/decryption works. However, they successfully grasp the idea that all their passwords are stored in their mobile phones and the key to their password vaults is downloaded and stored in the browser. We believe that such a mental model is good enough for them to behave securely. Since the built-in password mangers are available in browsers, all testers understand and have used such password managers before. They all successfully finish the assigned tasks. We notice that users sometimes misuse BluePass such as trying to store passwords in BluePass without starting the BluePass service in their mobile phones. However, they quickly figure out the reason and fix the problem. We do not observe any user action that is potentially dangerous. Thus, using BluePass does not cause a potential security threat, which is important to a password manager [9].

After finishing the tasks, the testers take a post-study questionnaire. The results are shown in Figure 5. As shown in Figure 5a and Figure 5b, testers generally think the concept of BluePass is understandable and it is fairly easy to set up. All testers think BluePass is more secure than password managers they have used (as in Figure 5c). 87% of testers (27 out of 31) agree that BluePass is more usable (as in Figure 5d) than other password managers. For instance, *"Sometimes I feel my passwords are saved everywhere using those password managers. Knowing that all my passwords exist only in my mobile phone is much better."*. However, four testers hold the opposite opinion, and one of them mention that *"I think all password manager do auto-filling, I only use a password manager in my personal computer. It is convenient."*. We realize that browser built-in password managers may be more usable when the users do not take actions to ensure security or they only need part of the functionality (such as only use it on a personal computer). Six testers show concern on password recovery when the mobile phone is lost. Although five of them still think BluePass is more usable than other password managers, having to back up the password vaults by themselves lowers their ratings.

Figures 5e and 5f illustrate that BluePass motivates the testers to increase password security. More than 70% (22 out of 31) of the testers are motivated to choose more secure passwords and less likely re-use existing passwords, thus making their passwords much stronger. However, though the testers report they are motivated to use more secure passwords, we notice that only nine testers have tried using random passwords generated by BluePass to create/change their passwords. We believe the reasons are multi-fold. First, when asked why not use passwords generated by BluePass, five testers mention that they feel "unsafe" when they cannot remember their passwords. Second, although being explicitly instructed to treat the account as a real account, some testers may still think they are just creating a test account. As a result, they feel reluctant to create a secure password. Third, five testers state that they are creating more secure passwords and they are just not using passwords generated by BluePass.

The majority of testers (94%) are willing to use BluePass to manager their passwords (see Figure 5g). We also ask the testers to compare BluePass to other favorite password managers they have used (see Figures 5h and 5i), and testers show large preference to BluePass over existing password managers. BluePass is rated averagely 4.9 points while the average rating of existing password managers is only 3.5 points. To summarize, BluePass is generally considered more secure and usable than existing password managers by the testers. Most of them show preference to BluePass and willingness to use it. Thus, it is evident that BluePass does help users secure their passwords.

8. Discussion and Limitations

We first discuss the rationale behind using only one pair of RSA keys for bi-directional communication. Next, we discuss the login scenarios when the mobile phone or Bluetooth are unavailable. Furthermore, we stress the importance of maintaining 2-factor authentication in BluePass supporting functions. Finally we point out some limitations of BluePass.

8.1. RSA Key Pair

BluePass can use only one RSA key pair (K_1, K_2) to achieve bi-directional communication between the mobile phone and the computer. We must guarantee that the compromise of K_1 will not lead to the compromise of K_2, and vice versa. We know that all public key cryptography algorithms ensure that it is hard to derive the private key from the public key, but not vice versa. For instance, given an ECC private key, it is easy to derive the ECC public key, since $public\_key = private\_key * G$. However, for RSA, in theory, it is hard to derive either e or d from knowing the other one. Therefore, we can use only one pair of RSA keys with careful parameter settings.

There are two minor things to notice in the detailed RSA implementation. First, in practice e is usually chosen a small/fixed number, but this should be avoided. Second, RSA private keys are often stored in their “Chinese Remainder Theorem" form, which includes the two secret numbers often denoted p and q, from which the totient is computed. With totient and the private exponent, the public exponent is quickly computed. Therefore, BluePass cannot use the Chinese Reminder Theorem to speed up the calculation.

8.2. Login on Mobile Device

As users may also use their mobile devices to log in online services, we introduce how BluePass could be used on mobile devices only. Recall that to use BluePass, 2 factors must be present and the mobile device is the secondary factor. However, in this scenario, only one factor remains and the mobile device also plays the role of the computer. The user only needs to input the master password in the mobile device to download the decryption key K_2. After the decryption, the BluePass mobile app can retrieve any site password for user to input and then immediately discard K_2. Note that browsers in mobile device usually cannot install any extension/application, making the auto-filling user experience unavailable. In case the computer does not support Bluetooth communications, we could also apply the same procedure to manually retrieve the site password from the mobile device and type it into the computer.

8.3. Extended Functionalities

BluePass is designed to avoid single point of failure, so some maintenance functions must be modified or extended to prevent them from being misused by the attackers. For instance, an attacker should not be able to change the master password of a victim user when the current master password is compromised. Toward this end, BluePass requires at least two factors to successfully use these functions. It is not difficult to strengthen the security of these functions by adding another authentication factor. For instance, in cases that the user lose access to one factor such as forgetting the master password, the phone number could be used as another factor. Recall that the user register its phone number in the BluePass server, and the phone can be used to receive one-time passcode as a second factor. However, losing the mobile device usually indicates temporally losing the passcode. To solve the problem, such combination has to wait a gap time for the user to react to device loss. Note that we assume even losing the mobile device, the user still possesses the phone number, which is usually the case.

8.4. Password Vault Synchronization

When a cloud provider, such as Google Drive, is used for password vault synchronization, it can become a captivating target for attackers. Its compromisation makes massive password vault leakage possible. However, even if password vaults are leaked, brute-forcing the long decryption key K_2 is still infeasible. Thus, unless BluePass server is compromised at the same time, user site passwords are still protected. We emphasize that a data breach is unlikely to happen simultaneously at both the cloud and BluePass server sides in practice. Whenever either side is compromised (i.e., one factor is lost), BluePass server is able to enforce users to re-generate a new pair of keys and naturally results in a totally different password vault, which recovers the two factor security guarantee and secures passwords.

8.5. BluePass Limitations

BluePass has several limitations. First, a user has to carry a powered-on mobile phone to make BluePass work; otherwise, BluePass falls back to conventional ways that users remember and input passwords. However, if the user chooses a random password generated by BluePass, the chance is minimal for the user to still remember the random password. In other words, BluePass does not completely resolve the dilemma between password strength and password memorability. Second, BluePass cannot work well when the mobile device or the computer does not support Bluetooth communication. In those cases, the hand-free benefit cannot be guaranteed by BluePass. Instead, the users have to use their phones to display their site passwords after inputting their master passwords.

9. Related Work

Password Study. Password is criticized to be insecure along its survival. Nearly four decades ago, Morris and Thompson [31] announced that password is vulnerable to dictionary attacks, and it remains vulnerable to various guessing attacks [25, 32, 41, 43]. It is generally believed that there exists a general trade-off between security and memorability [44]. Because of the predictable feature of a user password, researchers tried to unveil the characteristics of passwords [4, 7, 27, 38, 42] and their relation to human habit and psychology [11, 15, 19]. [26] conducts a study on password recovery.

Whereas numerous evidences show that "easy" passwords are insecure, users generally do not follow advices from security experts and are inclined to choose weak passwords or reuse passwords [1, 11, 15]. On the other hand, it is shown that users have limited compliance budget following security advice and the budget value varies significantly by the service nature of websites [1, 3, 6]. Given a plethora of attack vectors, following security advice that specifically aims to defend against just one or few types of attacks becomes unrealistic for users. Therefore, it is crucial for a website to carefully manage its security policies, even allowing slight security sacrifice. Florêncio and Herley [16] demonstrated that many leading web services generally allow weak passwords in fear of losing users.

Due to various drawbacks of password authentication, many alternative schemes has been proposed to replace passwords. For example, graphic-based passwords [12, 21] and biometrics-based passwords [20]. However, Bonneau et al. [5] evaluated all mainstream alternative schemes and concludes that none of them is able to replace the dominating status of password authentication. By contrast, many web services simply accepted the fact that passwords are weak and strive to survive with imperfect but practically effective techniques [6].

Password Managers. Facing the dilemma of not being able to replace passwords, many works focus on helping users manage and remember their passwords, which indirectly enhance password strength due to decreased memorability requirement. In consequence, password manager earns its prosperity. Despite ubiquitous "memorize and fetch" type of password managers such as browser built-in password managers or LastPass, researchers also proposed password managers that can enhance password security in addition to usability. PwdHash [36] attempts to hash user entered passwords and the domain URL to produce strong passwords. Similarly, Password Multiplier [18] also hashes user data to generate strong passwords.

Password manager significantly reduce the memory burden on users. However, it has its own usability and security problems [28]. PwdHash and Password Multiplier are criticized to have severe usability problems due to the fact that users failed to capture the correct mental model, which may even cause security flaw [9]. Karole et al. [23] investigated user perceptions and usability towards three types of managers. Silver et al. [40] demonstrated that careless auto-filling policy on non-https websites could make passwords be extracted directly from the web form by an attacker. Furthermore, A well-known vulnerability of password manager is single point of failure – the master password. To reduce the threat, Tapas [30] separates the storage of the password wallet from the key.

Two-Factor Authentication. Two-factor authentication (2FA) is a general concept that authentication needs two factors to complete. Besides traditional password, the second factor ranges from security questions [35, 37] to user-owned devices [2] and biometrics [22]. Although 2FA enhances security in the authentication process, it introduces usability drawback since users have to spend considerably more time to complete a login, which results in low adoption rate. To reduce such extra effort from users, many works focus on making the second factor transparent to users [10, 33, 39]. With a transparent 2FA scheme, users can achieve considerable usability improvement while trading off some other system resources such as power consumption for recording ambient sound in [33]. Besides, these schemes require more than light modifications on server or client side, making them hard to deploy and the additional usability benefit stays only in theory. BluePass itself is a variation of 2FA at the client side, which requires both a mobile device and a master password to retrieve site passwords. However, at the online server side, it remains as one-factor authentication as BluePass does not change the underlying password authentication framework. Thus, no modification at the online server side is required, making BluePass easily deployable.

Phone-assisted Authentication. In Tapas [30], researchers first proposed to isolate the password vault and the key in a smartphone and a computer, respectively. However, Tapas relies on a Rendezvous Server as a third-party to establish connection between the phone and the computer, which introduces a big usability problem – the user has to signal consent on both the browser and the cellphone in order to retrieve each password. Besides, Tapas implementation allows only one pair of computer and phone, which makes portability difficult – users cannot retrieve passwords on another computer. Furthermore, it does not address many real-world problems such as losing the phone. BluePass significantly improves usability while achieving the same level of security as previous works.

In most studies, mobile phones are used as a second-factor in a 2FA. Despite of traditional 2FA such as Google 2-step verification [17], mobile phones could also be used to fulfill other purpose or enhance usability. Phoolproof [34] uses smartphone to prevent phishing attacks, in which the phone and browser communicate via Bluetooth. PhoneAuth [10] uses the phone as a transparent second factor to authenticate through bluetooth. SoundProof [33] correlates ambient sound recorded on both the mobile phone and computer to ensure that the two devices are co-located. Shirvanian et al. [39] proposed to use mix-bandwidth devices to enhance server, defending against even offline attacks.

10. Conclusion

This paper introduces a hand-free password manager called BluePass for achieving both strong security and high usability. BluePass attains the security level of 2-factor authentication by storing password vaults in a mobile device and the decryption key in the user computer separately. Exploiting the automatic bluetooth communication between the mobile device and the computer, BluePass enables a hand-free password retrieval process for users. BluePass also places the decryption keys to remote servers to support password portability while de-centralizing the storage of password vaults to prevent a single point of failure. We implement a BluePass prototype on Android and Google Chrome platforms. Through system evaluation, we show that the password retrieval latency a user experiences is less than 200 milliseconds on average, and BluePass only consumes a negligible 1% battery power with 10 hours normal use on a mobile device. Through a user study comprising of 31 testers, we demonstrate that BluePass does motivate users to choose stronger passwords and less likely to reuse existing passwords. In addition, users find BluePass easy to comprehend and use, implying that human factors (such as misbehavior or wrong user mental model) generally do not weaken BluePass security.

References

[1] Adams, A. and Sasse, M.A. (1999) Users are not the enemy. *Communications of the ACM* .

[2] Aloul, F., Zahidi, S. and El-Hajj, W. (2009) Two factor authentication using mobile phones. In *AICCSA 2009* (IEEE).

[3] Beautement, A., Sasse, M.A. and Wonham, M. (2008) The compliance budget: managing security behaviour in organisations. In *NSPW* (ACM).

[4] Bonneau, J. (2012) The science of guessing: analyzing an anonymized corpus of 70 million passwords. In *IEEE Security & Privacy*.

[5] Bonneau, J., Herley, C., Van Oorschot, P.C. and Stajano, F. (2012) The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *IEEE Security & Privacy*.

[6] Bonneau, J., Herley, C., van Oorschot, P.C. and Stajano, F. (2015) Passwords and the evolution of imperfect authentication. *Communications of the ACM* .

[7] Bonneau, J., Preibusch, S. and Anderson, R. (2012) A birthday present every eleven wallets? the security of customer-chosen banking pins. In *Financial Cryptography and Data Security* (Springer).

[8] Chatterjee, R., Bonneau, J., Juels, A. and Ristenpart, T. (2015) Cracking-resistant password vaults using natural language encoders. In *IEEE Security & Privacy*.

[9] Chiasson, S., van Oorschot, P.C. and Biddle, R. (2006) A usability study and critique of two password managers. In *Usenix Security*.

[10] Czeskis, A., Dietz, M., Kohno, T., Wallach, D. and Balfanz, D. (2012) Strengthening user authentication through opportunistic cryptographic identity assertions. In *ACM CCS*.

[11] Das, A., Bonneau, J., Caesar, M., Borisov, N. and Wang, X. (2014) The tangled web of password reuse. In *NDSS*.

[12] Davis, D., Monrose, F. and Reiter, M.K. (2004) On user choice in graphical password schemes. In *USENIX Security*.

[13] Faulkner, L. (2003) Beyond the five-user assumption: Benefits of increased sample sizes in usability testing. *Behavior Research Methods, Instruments, & Computers* .

[14] Feldmann, S., Kyamakya, K., Zapater, A. and Lue, Z. (2003) An indoor bluetooth-based positioning system: Concept, implementation and experimental evaluation. In *International Conference on Wireless Networks*.

[15] Florencio, D. and Herley, C. (2007) A large-scale study of web password habits. In *ACM WWW*.

[16] Florêncio, D. and Herley, C. (2010) Where do security policies come from? In *SOUPS* (ACM).

[17] Google (2016) Google 2-step verification. URL https://www.google.com/landing/2step/.

[18] Halderman, J.A., Waters, B. and Felten, E.W. (2005) A convenient method for securely managing passwords. In *WWW* (ACM).

[19] Howe, A.E., Ray, I., Roberts, M., Urbanska, M. and Byrne, Z. (2012) The psychology of security for the home computer user. In *IEEE Security & Privacy*.

[20] Jain, A.K., Ross, A. and Pankanti, S. (2006) Biometrics: a tool for information security. *IEEE Transactions on Information Forensics and Security* .

[21] Jermyn, I., Mayer, A.J., Monrose, F., Reiter, M.K., Rubin, A.D. *et al.* (1999) The design and analysis of graphical passwords. In *Usenix Security*.

[22] Jin, A.T.B., Ling, D.N.C. and Goh, A. (2004) Biohashing: two factor authentication featuring fingerprint data and tokenised random number. *Pattern recognition* .

[23] Karole, A., Saxena, N. and Christin, N. (2010) A comparative usability evaluation of traditional password managers. In *ICISC 2010* (Springer).

[24] Lewis, D. (2015) Lastpass suffers data breach again. URL http://www.csoonline.com/article/2936105/data-breach/lastpass-suffers-data-breach-again.html.

[25] Li, Y., Wang, H. and Sun, K. (2016) A study of personal information in human-chosen passwords and its security implications. In *INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, IEEE* (IEEE): 1–9.

[26] Li, Y., Wang, H. and Sun, K. (2018) Email as a master key: Analyzing account recovery in the wild. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications* (IEEE): 1646–1654.

[27] Li, Z., Han, W. and Xu, W. (2014) A large-scale empirical analysis of chinese web passwords. In *Proc. USENIX*

Security.
[28] Li, Z., He, W., Akhawe, D. and Song, D. (2014) The emperor? s new password manager: Security analysis of web-based password managers. In *USENIX Security*.
[29] Malone, D. and Maher, K. (2012) Investigating the distribution of password choices. In *ACM WWW*.
[30] McCarney, D., Barrera, D., Clark, J., Chiasson, S. and van Oorschot, P.C. (2012) Tapas: design, implementation, and usability evaluation of a password manager. In *ACSAC* (ACM).
[31] Morris, R. and Thompson, K. (1979) Password security: A case history. *Communications of the ACM* .
[32] Narayanan, A. and Shmatikov, V. (2005) Fast dictionary attacks on passwords using time-space tradeoff. In *ACM CCS*.
[33] Nikolaos Karapanos, Claudio Marforio, C.S. and Capkun, S. (2015) Sound-proof: Usable two-factor authentication based on ambient sound. In *Proc. USENIX Security*.
[34] Parno, B., Kuo, C. and Perrig, A. (2006) *Phoolproof phishing prevention* (Springer).
[35] Pinkas, B. and Sander, T. (2002) Securing passwords against dictionary attacks. In *ACM CCS*.
[36] Ross, B., Jackson, C., Miyake, N., Boneh, D. and Mitchell, J.C. (2005) Stronger password authentication using browser extensions. In *Usenix security*.
[37] Schechter, S., Brush, A.B. and Egelman, S. (2009) It's no secret. measuring the security and reliability of authentication via "secret" questions. In *IEEE Security & Privacy*.
[38] Schweitzer, D., Boleng, J., Hughes, C. and Murphy, L. (2009) Visualizing keyboard pattern passwords. In *IEEE VizSec*.
[39] Shirvanian, M., Jarecki, S., Saxena, N. and Nathan, N. (2014) Two-factor authentication resilient to server compromise using mix-bandwidth devices. In *NDSS*.
[40] Silver, D., Jana, S., Chen, E., Jackson, C. and Boneh, D. (2014) Password managers: Attacks and defenses. In *Usenix Security*.
[41] Veras, R., Collins, C. and Thorpe, J. (2014) On the semantic patterns of passwords and their security impact. In *NDSS*.
[42] Veras, R., Thorpe, J. and Collins, C. (2012) Visualizing semantics in passwords: The role of dates. In *IEEE VizSec*.
[43] Weir, M., Aggarwal, S., De Medeiros, B. and Glodek, B. (2009) Password cracking using probabilistic context-free grammars. In *IEEE Security & Privacy*.
[44] Yan, J., Blackwell, A., Anderson, R. and Grant, A. (2004) Password memorability and security: Empirical results. *IEEE Security & Privacy Magazine* .
[45] Zhou, S. and Pollard, J.K. (2006) Position measurement using bluetooth. *IEEE Transactions on Consumer Electronics* .