# Password Manager Enhancement: Better Security

Pin-Chun Chen
CSIE
National Taiwan University
Taipei Taiwan
b07902113@ntu.csie.edu.tw

En-Ting Lin
CSIE
National Taiwan University
Taipei Taiwan
b06902023@ntu.csie.edu.tw

Guan-Hong Liu
CSIE
National Taiwan University
Taipei Taiwan
b07902005@ntu.csie.edu.tw

Yi-Chun Lo
CSIE
National Taiwan University
Taipei Taiwan
b07902127@ntu.edu.tw

## ABSTRACT

Password managers nowadays provides access to all of the passwords stored once the user authentication is passed (usually by master password). However, the master password itself isn't guaranteed secure and the storage of such manager may be breached. Since the manager stores sensitive data, risks reduction is much in need.

We aim to provide a scheme for passwords requiring more security to the current password manager, which security doesn't bind with master password. And even if password manager's database is compromised the password won't be directly leaked.

## 1. INTRODUCTION

Password manager is a software application that can help manage and store user's passwords. It helps user maintain a large number of passwords and accounts information, generating strong passwords to make sure user's passwords have good security properties. Most of the password managers stores passwords under encryption to decrease risk of database leakage. For example, 1Password use AES encryption to encrypt the strong passwords.[7] User can access strong passwords via passing through the authentication. There are different ways to authenticate, the major way is using Master password with other factors, which are called "multiple authentication". It is designed for the purpose of having the authentication become harder to prevent user's password manager from being easily compromised. There are two ways password managers work: cloud & local. The most significance difference between them is that cloud password manager stores all the encrypted passwords and authentication information in cloud database, while local password manager stores data in local device, and it needs a third trusted party to achieve synchronization. Some of the famous cloud password managers like 1Password, LastPassword, Norton, etc. They use cloud database to synchronize user's devices, which means user can access password manager conveniently in any device they want to authenticate. In contrast, some of the well-known local password manager like KeePass, Enpass, Pass, etc. They have some tradeoff between safety and convenience. Since encrypted data stored in local device will make attacker harder and less interested in launching an attack due to physical limitation and the small scale, local password managers have better security assurance in terms of database compared to cloud. Nowadays, more and more people claim that password manager is the safetest way [3][4] to protect your passwords and convince everyone to trust and start using it. To know how the password manager works, we are going to introduce the scheme.

### 1.1 Original scheme

The original scheme of password manager can be described as follows: we denote password manager as **PM**, strong password as **SP**, master password as **MP**, key derive function as **H**, key generated by H as **k**, keyed encryption as **E**, and cipher encrypted by E as **c**. The password manager has three states: offline, online but not authenticated, online and authenticated. When password manager is offline, it stores the strong passwords encrypted by credential. We assume that there exists a secure channel between PM and user to exchange MP and SP. To access the SP, user has to give Password manager MP and other factors. After PM verified H(credential), which are implemented differently depending on companies, the state of PM will become online and authenticated. User then give a request for one of SP, and PM use MP to decrypt the SP store in PM. (Figure 1)

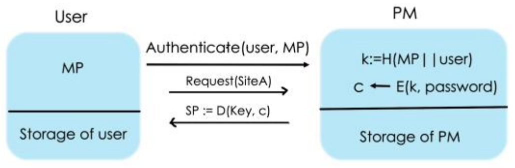

**Figure 1: password manager original scheme.**

### 1.2 Motivation & Research problem

It seems like password manager has several benefits: you don't have to memorize all your password except MP, it can auto-generate highly secure passwords for you, it can alert you to a phishing site, it helps protect your identity, it saves your time, etc.[5] With the claim of security and convenience, there are more and more people start using password manager to manage their passwords instead of using the conventional way to memorize them. However, having a manager who controls all of your passwords also implies great risk if the manager itself were to be breached. Imagine the attacker somehow gets the master password, he may have the ability to access every account credential stored in the password manager. Even if the user enables 2-factor authentication, which makes the attacker unable to pass authentication even if he gains the master password. It doesn't imply safety and there are ways to bypass 2FA [8]. Some of the attack doesn't even need to pass authentication, they simply make use of users within authenticated mode. Therefore, we concern that putting all eggs in one basket may after all be a bad idea. In fact, in *Your botnet is my botnet: analysis of a botnet take*over: "*It is also interesting to observe that 38% of the credentials stolen by Torpig (a malware botnet) were obtained from the password manager of browsers, rather than by intercepting an actual login session."* [13] Stated the exact concern of us towards password manager's security. We are going to list several already-known attacks for password manager and later discuss how we can encounter the vulnerabilities we are facing.

## 1.3 Known Attack Methods

Since password manager helps manage your passwords, we have to consider whether it is a problem of putting all eggs in one basket. Storage of all the encrypted passwords in a password manager is to have more security benefits or more issues. Once the attacker gets the master password, he may have the ability to access every password and account stored in the password manager. Many password managers said that they have 2-factor authentication, which can make sure that the attacker cannot access passwords even if he gains the master password. However, 2-factor authentication doesn't imply 100% safety [8][9]. Some of the attacks pretend users to turn off the setting of 2-factor, making it easier to pass through the authentication process. The followings are some known attack methods, we will explain it more detailly in section 4.3.

*1.3.1 storage attack.* As introduced, password manager has three states: offline, online but not authenticated, online and authenticated. There is research found that as long as your password manager is online, you have to beware of the password manager putting yourself to risk [2]. For instance, 1Password had once been found that under certain conditions, there exists specific operations which lead your master password plaintext stored in the memory [10]. Attackers are willing to take effort attacking cloud password manager's database, as long as it has bugs, they will take privilege and have chance to get all of your passwords.

*1.3.2 Authenticated State Exploitation.* There are attacks that do not have to pass through the authentication. Instead, they take advantage of the clients who are authenticated. For example, attacker may launch a sweep attack. After finding websites which have security holes, attacker can inject some malicious code in order to steal victim's passwords. As long as victim allow password manager to auto-fill the password, the content would be sent to the attacker. In fact, to find a website which contains security problems is not that difficult. For example, wi-fi hot spots in public often redirect user to a login website, and the website may have already been compromised. What's more, websites using http are not safe, as packets intercepted by attacker are plaintext. Attacker may use phishing to get user's password plaintext. XSS injection is an example of code injection attack which is effective and simple. By methods mentioned above, attacker does not have to be authenticated but just exploit the authenticated state of client.

*1.3.3 Malware.* Malware can not only work in storage attack, but also have the ability to steal passwords from other inputs. Key logger is an infamous one, attacker can get passwords directly without any effort. Though some password managers suggest that user enter the password via the screen key board [11]; nevertheless, there are malwares that have the ability to record your screen input. If attacker get your master password, even passwords that are not be key logged might be leaked.

*1.3.4 Implementation error.* There are several password managers which are found existing implementation errors, bringing to leakage of user's credentials for arbitrary websites [12].

## 1.4 Our Goal

In conclusion, password manager provides some good qualities. However, it does not follow the rule of "No single point of failure", and the security aspect faces great challenge. Therefore, we hope to provide a scheme built on top of current password manager scheme to inherit its good qualities while enhancing its security at the same time.

## 2. RELATED WORK

Many researches have made great efforts to enhance users' password, and there are various password managers nowadays to handle memory problems. Also, many researches work on discovering the vulnerabilities in current password managers, which is also crucial to the improvement of password manager's security.

However, some of them such as 1Password were found vulnerable under certain circumstances by ED TARGETT [2]. He noted that under certain user actions, the master password can be left in memory in cleartext even while locked. Moreover, Charlie Osborne shown that some well-known password managers are also vulnerable [6]. Therefore, we thought that enhancing existing password managers is necessary.

1Password enhance its safety by using device generated "secret key" so that only master password itself isn't enough for decrypting stored passwords and makes brute-force attack when database leaked unfeasible. This greatly decreases the risk of database storage. We think our work has similarity in keeping piece of information away from the password manager. Yet their method still doesn't defend against authenticated state attack and the key pair (master password and "secret key") is still single point of failure for all passwords. Nonetheless, their scheme requires no extra memory for users thus still provides great improvement in password manager security.

## 3. Our scheme

We aim to provide a scheme which allows user to make a tradeoff between convenience and security. Which can be flexibly applied on passwords user deemed as valuable. We will first make the assumptions in section 3.1, then state the attacker model in section 3.2. After, we address the security properties in 3.3, then finally, we present the details of our scheme in section 3.4.

### 3.1 Assumption

We only focus on the vulnerabilities which a password manager can help secure. For that purpose, we make several assumptions to make sure our research stays on track.

*3.1.1 Master Password.* The user does not use old passwords as master password nor reuses master password as any other password. Which means master password provides certain level of complexity against dictionary attack. However, we don't assume the complexity of user's master password to be as strong as random generated passwords, since it is still user generated and the complexity may differ widely upon different users.

*3.1.2 Cryptographic primitives.* The cryptographic primitives themselves are not this research's focus. Therefore, we do not discuss any exploitation towards the cryptographic primitives used: To reverse a Hash or decrypt without key, the only way is to perform brute-force attack.

*3.1.3 Communication channel.* The communication channel between the client and the password manager, the client and SiteA is secure. There is nothing a password manager can do if there is no safe channel to authenticate the client or the passwords sent from client is intercepted.

*3.1.4 Online brute-force.* If the online brute-force attack cannot bypass the password manager, the password manager will act when an account is judged as under attack. (account lockout, flag warnings, etc.) Furthermore, since our scheme causes more overhead, we expect the websites of the accounts our scheme applies on to be more secure demanding, so the websites will also act if it senses an attack. As a result, the force of online brute-force attacks will be limited, even more so if it cannot bypass the password manager.

### 3.2 Threat model

As we aim to reduce the risk of using current password managers, our attacker model enables attacker to perform all attacks mentioned previously in chapter 2 except for implementation error attacks, as implementation errors are uncontrollable and could directly disable the scheme. We further discuss the situation where the attacker straight up gains sensitive information to demonstrate our scheme's resistance against data leakage.

*3.2.1 Computational power.* Attacker is able to offline brute-force WP, but not SP. As for MP, stated in 3.1.1, the complexity depends on user thus we don't rule out the possibility of MP being broken.

*3.2.2 Network control.* Attacker controls part of the internet and may have the target interact with his domains. He can also monitor the traffic regarding his target.

*3.2.3 Malware.* Attacker may infect the target's device with malware. We mainly discuss keylogger malware, as the effect of storage probing malware will be covered in the next section and phishing malware's function can be done by *Network control* ability.

*3.2.4 Extra information.* We also discuss the possibility that the attacker obtains the master password of a user or directly compromises the PM's database.

### 3.3 Security property

Under the assumptions, our scheme provides the following security properties against attackers in the threat model:

1. Leak of master password doesn't result in direct leakage of any strong password, and the attacker can only perform online brute-force attack against weak password with the help of password manager.
2. In case of database compromised, though master password is subject to offline attack, in order to attack any weak password, the master password has to be breached first.
3. Even if both master password leaked and database compromised, the attacker is still unable to offline attack weak password.
4. Under malware infection, only the passwords of accounts where user completes the login session within the infected time interval are directly leaked.

### 3.4 Scheme design

*The concept of our scheme is to not allow password manager to possess all information, so we can lower the level of trust towards the password manager: from trusting it will keep the passwords safe, to trust it will help us keep the password safe.*

*3.4.1 Password generation.* The strong password of siteA will now be H( $Secret_{siteA}||WP_{siteA}$ ). Where $Secret_{siteA}$ is password manager generated, having entropy at least as original strong password, stored in exact way as original strong password. $WP_{siteA}$ is user generated, and only possessed by User. By doing so, we break the material to generate strong password in two pieces, one for password manager to ensure strength of password, the other for user to make sure password manager itself is not capable of yielding strong password itself.

*3.4.2 Password request.* For user to request a strong password after success authentication, the user now doesn't directly gain the strong password on demand, instead he gains power to request H($Secret_{siteA}||WP_{siteA}$), where $WP_{siteA}$ is user provided. Note that the password manager does not and cannot verify the correctness of generated strong password, since if it possesses information to even just verify, it will become tool for attacker to perform offline attack. With that being said, the password manager can still sense if an account is being attacked, either detecting multiple tries in a row, or it cooperates with SiteA and expect to receive success login message whenever a strong password request is made by user.

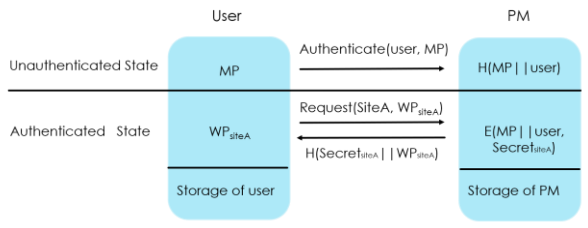

**Figure 2 : request password through our scheme**

## 4 Evaluation

In this part, we first assess the overhead our scheme causes. Then we explain the difference between our scheme and original scheme to show that only little modification is needed and how flexibility can be achieved. After, we compare various aspects of our scheme with original scheme to fully evaluate our scheme.

## 4.1 Overhead

For every SP generated under our scheme, we require user to memorize a WP. Though unideal, but if there is no more information a user needs to provide to access his passwords within authenticated state, an attacker successfully exploits authenticated state of a user gains access to every password of the user. Even if the user does not maintain authenticated state to avoid the problem above, this causes tremendous overhead as he needs to re-authenticate for any password requirement, and this merely moves the point of failure to the authentication process. Therefore, we consider the extra memory overhead necessary.

Fortunately, WP doesn't have to be very strong (In fact, if user is willing to memorize a SP, he might as well doesn't need the password manager): Since the only parties possessing any form of WP's information are User (WP) and SiteA (Enough to verify SP, possibly H( H($Secret_{siteA}||WP_{siteA}$)) ), the strength of WP is only under pressure when attacker is able to perform H( $Secret_{siteA}||WP_{guess}$) to query SiteA: implying either $Secret_{siteA}$ is leaked or authenticated state is exploited, both in the original PM case, SP is leaked (the storage method of secret is as in original scheme's SP). And even in those cases, the attacker cannot offline brute-force WP since the only party capable of verifying H( $Secret_{siteA}||WP_{guess}$) is SiteA (Unless SiteA's database is compromised too).
Furthermore, user only need to apply this scheme on passwords he thinks is valuable, so the overhead only tolls on a portion of his passwords.

## 4.2 Compatibility

The only modification of this scheme besides that the stored information is now secret not SP, is in the request stage. PM retrieves user message (WP) and sends back not SP but H($Secret_{siteA}$||WP). Since there isn't much change, we only need to add extra tag of which scheme it use in the stored data, then the PM can decide whether it should expect a WP from user and output accordingly.

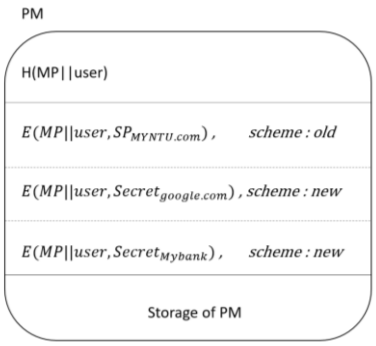
PM
H(MP||user)
$E(MP||user, SP_{MYNTU.com})$, scheme : old
$E(MP||user, Secret_{google.com})$, scheme : new
$E(MP||user, Secret_{Mybank})$, scheme : new
Storage of PM

**Figure 3: flexibly choose the scheme of passwords**

As our scheme can be built on top of current password manager's works, we think that Cloud PM would fit our scheme better, since Cloud PM has better convenience but less security compared to local PM, which our scheme compliments. Therefore, the following evaluation will be based on our scheme on top of Cloud PM.

## 4.3 Security

In order to show that our scheme provides more security than original password manager scheme. In this section, we list the previously mentioned attacks against password manager, and theoretically put our scheme under these attacks and compare the results with the original scheme.

*4.3.1 Storage attack.* Against storage attack, as in the original scheme, we lost our MP and Secrets (can be decrypted if MP leaked). However, as long as password manager does not store the WP input by user, there is no way that attacker gain information about WP, limiting his attack to online attack only. Note that there is incentive for password manager to temporarily store MP at running stage to maintain authenticated state and decrypt the SPs (otherwise user has to re-authenticate every time). But there is no reason other than implementation error to store WP which the sole purpose is to deny the password manager's ability to self-generate SP.

*4.3.2 Authenticated state.* The power of authenticated user is gaining access to query H($Secret_{siteA}||WP_{guess}$), even attacker make use of the authenticated state, he still can only perform online attack and even cannot bypass the password manager.

*4.3.3 Key logging.* Even if the MP has been logged, since WP remains unknown as long as user has not input it in the malware infected time interval, attacker only gains SPs for which user has completed the login session. Therefore, for user with security consciousness. The damage can be stopped after detection of malware.

## 4.4 Ours & Cloud & Local

We compare several aspects of Ours, Cloud and Local PM in this section, ours would be Cloud PM applying our scheme.

*4.4.1 Synchronization.* Since WPs aren't stored in the password manager, the synchronization effort of ours and cloud should be the same.

*4.4.2 Convenience.* Cloud has synchronization advantage over Local PM which contributes to convenience, meanwhile ours require extra user memory which could be considered expensive.

*4.4.3 Storage risk.* The cost of attacking Local PM storage is higher than cloud since attacks against cloud's database are at larger scale. However, though ours uses the cloud structure, the stored information is not complete, so the attacker gains limited information even if database compromised. Therefore, we judge our storage risk comparable to locals.

*4.4.4 Phishing risk.* Current password manager defends phishing at autofill mode [1], as ours could use the same technique to decide whether to provide H($Secret_{siteA}||WP_{guess}$) ability, so SP phishing against all schemes should be hard. As for phishing for MP, the difficulty should be the same. Yet in case of MP really phished, our scheme suffers less damage as it is not the only piece of information required to get SP.

*4.4.5 Authenticated user & Key logging risk.* As stated in previous section, our scheme provides certain level defense against this attack compared to original scheme. Cloud and local acts the same regarding both issues.

| | Sync | Convenience | Storage | Phish | Authenticated user | Key log |
|---|---|---|---|---|---|---|
| Cloud | ○ | ★ | △ | ○ | ✕ | ✕ |
| Local | △ | ○ | ○ | ○ | ✕ | ✕ |
| Ours | ○ | △ | ○ | ★ | ○ | △ |

Chart 1: comparison between Cloud, Local and Ours

## 5 Future work

As our scheme should be compatible with original PM scheme, we could implement it by modifying existing opensource Cloud PMs such as KeePass or Bitwarden. After implementation, it would be easier to truly evaluate the effectiveness of our scheme and make improvements based on the information gained by implementation.

Another possible work we can do is to combine our scheme with other security enhancing works, such as 1Password's key generating scheme mentioned in related work, could potentially further strengthen our data storage security.

## 6 Conclusion

We came up with a scheme which could enhance the security of desired password at the cost of extra user memory. And we can see in the evaluation that the WP indeed become the key ingredient that attacker cannot gain even successfully launched various attacks. That being said, there is still much work that can be done and hopefully our work can improve the security of password managers.

## ACKNOWLEDGMENTS

Special thanks for professor Hsu-Chun Hsiao and TA 毛偉倫, giving us many great advices and suggestions for our topic and research ways. Discussion with peers also has positive effect on our research, thanks 連崇安 and 黃申昌 for giving some advices and questions about our work.

## REFERENCES

[1] https://www.howtogeek.com/451177/how-a-password-manager-protects-you-from-phishing/
[2]https://www.cbronline.com/news/password-manager-security?fbclid=IwAR0CG6FC6k2dfX-ZiF8ElwAjvwZIU7_USAl2i0ZX_JcYkFKvAhP9Q12bAL0
[3] https://passwordbits.com/you-need-password-manager/
[4] https://www.techrepublic.com/article/5-reasons-why-you-should-use-a-password-manager/
[5] https://www.malwarebytes.com/what-is-password-manager/
[6] https://www.zdnet.com/article/critical-vulnerabilities-uncovered-in-popular-password-managers/
[7] https://www.quora.com/How-do-password-managers-encrypt-passwords
[8] https://www.pcmag.com/news/google-phishing-attacks-that-can-beat-two-factor-are-on-the-rise
[9] https://www.cnet.com/news/two-factor-authentication-what-you-need-to-know-faq/
[10] https://www.cbronline.com/news/password-manager-security
[11] https://helpdesk.lastpass.com/zh-tw/your-lastpass-icon/loggin-in/virtual-keyboard/
[12] https://crypto.stanford.edu/~dabo/pubs/papers/pwdmgrBrowser.pdf
[13] https://dl.acm.org/doi/pdf/10.1145/1653662.1653738