Using Client-Side Substitution Ciphers to Enhance Password Strength

Research Manual

Student Name: Kevin Davitt
Supervisor: James Egan
C00215181@itcarlow.ie

Abstract

According to the CISSP. The three types of authentication are; something you know, something you have and something you are. Of these, something you know is the most common. In terms of something you know, passwords are used and typically consist of a string of characters that the user of a system must remember in order to authenticate. Users frequently create weak passwords, for example, among the top 10 passwords disclosed in breaches are “123456”, “abc123” and “password”. (Rob Picheta, 2019) Despite administrators enforcing password policies, users continually create weak passwords and continue to be one of the weakest links in the security chain. In addition to this phenomenon, due to advances in technology and the ubiquitous presence of smartphones, the average amount of passwords that an average person needs to create and subsequently recall has increased dramatically, according to digitalguardian.com, the average number of accounts associated with a single email address is 130 (Digital Guardian, 2019).

Considering the fact that users are encouraged to create long and complex passwords, encouraged to change them periodically, use non-dictionary words and that the number of accounts a user has is growing, this document suggests a method to ensure strong passwords across accounts while keeping them as memorable as possible for the users in question, the method utilizes an application that uses arbitrary keyboard layouts to map the keys a user presses to alternative characters.

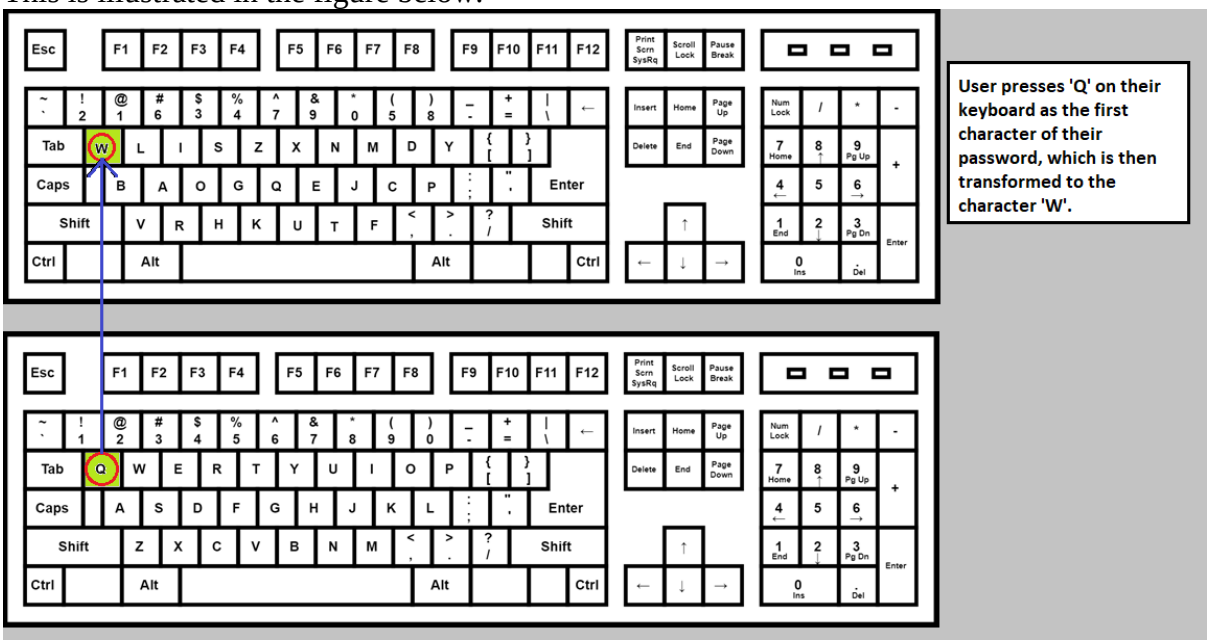
This document aims to justify the usefulness of such an application through research based around the state of the art in passwords and authentication in 2019, what is used now and what the vision of global leaders are, going forward. It also aims to compare the advantages and disadvantages of various implementations and infrastructure.

Table of Contents

Introduction

In 2019, passwords continue to be the one of the most popular methods of authentication, this, coupled with the amount of accounts that the average person has, creates a problem. Users of accounts are told to make every password unique, long, complex and to change it often. The amount of passwords a user has to remember is far greater than the typical user's memory capacity, it is often joked that users spend 50% of their time clicking the forgot password button.

Using dictionary words as your password is not advised, as attackers can craft dictionaries and use them in brute force dictionary attacks. This paper focuses on a way that users could continue using dictionary words without making their passwords weaker. The suggested method for this is by having each key of a user's keyboard mapping to a different character, physically, the user would be typing in a password they can easily remember, but logically, each key will map to a different character, meaning that the "actual" password used for a service, will not be a dictionary word. This is illustrated in the figure below.



Using this method for passwords across accounts has several advantages;

- Allows users to use any password, provided it is long enough
    - The user's password string is transformed according to a unique map, meaning the string received by the service is different to the string typed by the user
- Enhances the randomness of the password server-side

    - This improves the strength of the password by transforming dictionary word to non-dictionary words.
- Resistant to brute force and dictionary attacks
    - Attackers that are using dictionary attacks from common password lists will not find the password used by the user, due to the mapping. Brute forcing hashes will have to try every possible combination, as using a dictionary attacks is no longer viable.
- Resistant to shoulder surfing
    - If an attacker sees a user typing the password on the keyboard, they still do not have the mapping. The mapping is unique for each user of the application, meaning the password the attacker saw is useless.
- Resistant to hardware keyloggers
    - Some implementations of the proposed mapping software will be resistant to keyloggers as the keys pressed by the user are changed by software on the machine. This will depend on the sophistication of the keylogger.

Topics Researched

Password Mapper Overview

The basic idea behind the password mapper is to help diversify the character set used in passwords while allowing users to have easy to remember passwords as users typically attempt to create passwords they will remember and consequently create weak passwords (Kävrestad, Eriksson and Nohlberg, 2019). By mapping the keys on the keyboard to different characters users can create easy to remember passwords that are also secure. There are three ways to look at this;

- Each character in the password is being substituted for another character or multiple characters i.e. Q -> W, W -> s39
- The keyboard layout no longer reflects what character is inputted when a key is pressed (Figure 1)
- The password is now a pattern on the keyboard, with the user using a string to remember the pattern

Each of these concepts ultimately amount to the same result, the user types in a string from their perspective, and that string is transformed before it is received by the authentication service.

Mechanism Category

This idea vaguely falls into the category of Password Reformation as defined by an Article in the World Applied Sciences Journal. (Raza, Mudassar & Iqbal, Muhammad & Sharif, Muhammad & Haider, Waqas. 2012) The main difference being that the transformation is static, and will be the same every time as the transformation is being done on the client side, before being received by the server. One type of Password reformation as defined by this paper is the S3PAS system (Zhao, Huanyu & Li, Xiaolin. 2007), the solution suggested here requires extra server technologies and education of the user as to how the system works. This system involves users identifying their "session passwords" from a graphic on screen. Due to its requirements from the server, it is different to the proposed solution, as the password mapping solution occurs at the client side only of the client-serer model, and the input mechanism for the final transformed password remains the same.

The ideas are similar in that the secret used to authenticate is transformed, but it is where that transformation takes place that is different. In addition, the inputs from the user logging in are dynamic in the case of the S3PAS system, whereas with the proposed mapping solution, the users inputs to authenticate will be the same each time.

The below table, from the World Applied Sciences Journal, illustrates some types of password authentication mechanisms and their resistance to common attacks on passwords.

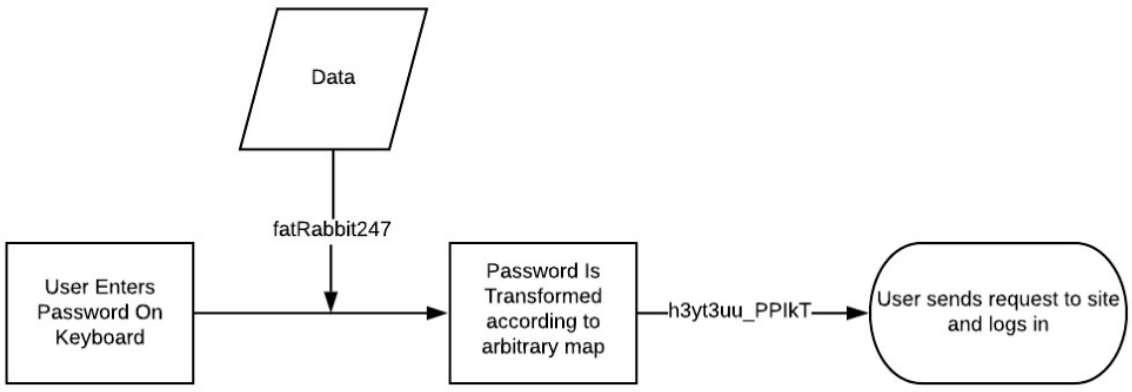
Table 1: Analysis of authentication methods

Method	Resistance to attacks	Additional Hardware Requirement	Cost	Mental attitude effects	Protection level	Processing Time
Conventio-nal password scheme	No		Normal		Low	Fast
Key stroke dynamics	Shoulder surfing, pishing, key loggers	No	Normal	Yes	Medium	Medium
Click patterns	Shoulder surfing, pishing, key loggers	No	Normal	Yes	Medium	Medium
Graphical passwords	Shoulder surfing	Yes	High	Yes	Medium	Slow
Biometrics	Shoulder surfing, pishing, key loggers etc	Yes	High	No	High	Slow
Authentication Panel	Video recording, shouldering	No	Normal	Yes	High	medium
Reformation Based	Brute force, video recording, shoudering and dictionary attacks	No	Normal	No	Medium	Fast
Moving Balls Based	Dictionary attacks, shouldering	No	Normal	Yes	High	medium
Expression Based	Brute force, video recording, shoudering and dictionary attacks	No	Normal	Yes	High	Fast
Virtual Passwords	Pishing, key loggers and all other online attacks	May be	May be high	No	Medium	Fast
Time Signature	Shoulder surfing, dictionary attacks, replay attacks, key loggers etc	No	Normal	Yes	High	Slow

Figure 2

Placing the proposed method into the Reformation Based category here shows it's resistance to attacks and enhanced protection for the user. For example, using this method makes passwords more resistant to brute forcing hashing attacks, in the event a database containing usernames and hashes were compromised. This is significant as according to the report by Verizon in 2018, 81% of successful Cyberattacks are due to compromised usernames and passwords. (Enterprise.verizon.com, 2019)

Password Mapper Authentication Flow Diagram



The above illustrates how a user would physically press keys that are in their usual layout on their keyboard, once the keys are entered, each character is simply substituted for an alternative, according to a unique randomly generated character mapping scheme. This is a substitution cipher. The diagram below illustrates a simple substitution cipher.

Incomplete Sample Map

key	char
f	h
a	3
t	y
R	t
a	3
b	u
b	u
i	_
t	P
t	P
2	l
4	k
7	T

Suggested Method

For the proposed solution, a user will download the mapping application and create an account. This application will generate a unique character map as illustrated above, at its heart, it is essentially a simple substitution cipher. It is important to stress that the goal of the cipher here is not to encrypt the password, just to transform it, so once transformed, it is more "random" than a user-chosen password. The map will then be stored on the server side, so it may be downloaded to other devices that the user has. Initially, the focus will be on developing an application that works on PCs only, with the view to design a mobile version in the future.

This solution may also offer features like classic password managers, where passwords can be saved for multiple accounts. It is recommended that this feature is used only for low value accounts, with medium and high value accounts using the mapping function.

Each user's unique map should be kept secret to ensure the strength of the mapping application in securing passwords, although there is some degree of resistance even if maps are compromised.

Increased Password Strength

Password strength is determined through two factors, through the total possible amount of passwords needed to be guessed in order for it to be successfully brute force and how easy it is to guess. For example, using the password "password" is predictable and likely to be guessed by an attacked.

Given that an attacker knows the character set used and the password length, the mathematical formula to calculate the total amount of passwords needed to be guessed under normal circumstances is C^n, where C is the size of the character set and n is the password length.

If individual characters typed by a user are mapped to multiple characters before being submitted to a service, the formula to calculate the total possible passwords becomes C^{n*m}, where m is the number of characters that an individual character is mapped to. It is also worth noting that due to the characters being mapped, the password also becomes far less predictable and less likely to be guessed by an attacker.

An example using real numbers is shown below, for simplicity, it is assumed that the attacker knows the size of the character set and the length of the original password that the user would remember. It is also assumed that every character will map to the same number of characters as one another. There are some other factors at play such as unique characters in the original password but calculating their minor impact on the following calculation is trivial and has been omitted.

Example: *Assuming an alphanumeric character set, a password of length 8 and each character from the original password mapping to a random 3 character string, where C is the size of the character set, n is the length of the original password and m is the number of keys a character is mapped to.*

Formula without mapper: $\mathrm{C^n}$
Formula with mapper: $\mathrm{C^{n*m}}$

Permutations without mapper: $62^8 = 218340110000000$
Permutations with mapper: $62^{8\times3} = 104087970000000000000000000000000000000000000$

Using the laws of indices we can show that the password generated using the mapper application would be $\frac{C^{n\times m}}{C^n}$ times more secure than not using it. In this example, the password using the mapper software would be $\frac{62^{8\times3}}{62^8}$ more secure, that is 62^{16} or 47672402000000000000000000000 times more secure.
This proves how using an application such as this would drastically increase the security through increasing the total number of passwords an attacker would have to guess and through eliminating predictable passwords.

Similar Applications

Similar applications to the proposed Password Mapper include common Password Managers, such as LastPass and 1Password. These services save passwords to an encrypted vault, and the vault can be accessed and used to login to different services using the master password for the account. The main problem with Password Managers is the single point of compromise, i.e. the master password to the account. The proposed method in this manual suggests a single password to access the mapping function, this means the passwords themselves are not stored. The trade-off here is that the user is expected to remember passwords from a physical standpoint, such as the keys on the keyboard used to type “fatRabbit”, while the application will map the password to the password that will be used on sites. This eliminates the single point of compromise, as if someone accesses the maps, they still do not have the password.

The proposed method could involve a classic Password Manager style application for low value accounts, and then involve using the mapping function along with a password for higher value accounts.

Security – Justifying the need for the proposed solution

Human Memory and Password Recollection

Relevance to Project

The proposed password solution is designed with the end user in mind, with the view that users have too many passwords to remember and given the complexity and length requirements, remembering complex passwords for multiple accounts is out of the questions for the average user.

Discovery

In a study completed in 2012, 84% of participants reported having trouble remembering passwords when they had to remember between 7 and 9. (Pilar et al., 2012)

Combined with the fact that on average, an email address is linked to approximately 130 accounts, each likely requiring a password, it simply becomes irrational to expect users to remember every password while also keeping them long and complex. (Digital Guardian, 2019)

According to a study at Carnegie Mellon University, only 35% of people memorize their passwords, further accentuating the fact that users struggle to recall passwords. (Ur et al., 2019)

How did this help?

These studies accentuate the need to aid users in remembering passwords, the proposed method will allow users to use weaker and easier to remember phrases for their passwords while, they are actually using strong passwords. All these studies show that users find it difficult to recall passwords, these studies prompted research of whether the typical user seems capable of generating a strong password when required to do so.

Password Strength and User Generation

Relevance

According to studies, users tend to generate weak passwords even when consciously attempting to generate strong passwords, this further highlights the need for a password mapper in order to mitigate brute force attempts at cracking weak password hashes in a backend database.

Discovery

According to a study done at Carnegie Mellon University, the average user may not know how to create a strong password, in the study, 46% of passwords created by participants who generally wished to create strong passwords, were vulnerable to attacks. There were three different password policies used to create three different passwords, the table below illustrates the vulnerability of created passwords.

Policy	**#Unique Created**	**#Number Vulnerable**	**% Vulnerable**
Low	37	21	56%
Medium	47	24	51%
High	47	21	45%

This study also revealed some common misconceptions that users seem to have about strong passwords, such as adding a single digit such as “1”, or a symbol such as “!” to the end of a password will automatically make it strong. This study also suggests the reason why users may think that simply complying with a password policy guarantees the security of their password. (Ur et al., 2019)

In addition to this, it has been observed in 2007 that users often try to use the weakest password they can get away with, while not necessarily knowing it is weak. (Florencio and Herley, 2019)
Similar behaviour was observed in 1979, where users chose the shortest possible passwords and used simple combinations, such as a name with a single number on the end. (Morris and Thompson, 1979)

How does this help?

These studies highlight the fact that many users find it difficult to create truly strong passwords even when actively trying to, this highlights the weaknesses in user generated passwords. Password strength is dictated by the total possible passwords that an attacker would have to guess in order to brute force the system, if users continue to use common architecture when it comes to passwords i.e. capitalised letter, rest of string, number, special symbol, the amount of possible passwords for an attacker to guess is drastically reduced, mathematically speaking.

Security of Existing Solutions

Relevance to Project

Proving the need for a secure password management solution, keeping the strong points of existing password managers without the weak points.

Discovery

Guide to Password Manager Vulnerabilities, identified in all 3 tested Password Managers.
User Interface, Classic Web Based, Bookmarklet and Authorization were the categories of vulnerabilities identified.
This study shows that the development of a Password Manager, especially a web based one, requires a defence in depth approach and having security in mind from the very first stage of development. (Li et al., 2019)

In 2019, a vulnerability in LastPass allowed malicious adversaries to potentially access credentials used by users, if they could get the user to execute a malicious piece of JavaScript. This is not as complex as it may sound, as a user simply visiting a malicious website could be enough. (Cimpanu, 2019)

Password managers in the past have proved a huge target for cyber criminals, for example, a case in which 2578 unit of Ethereum cryptocurrency was compromised seemed to involve the compromise of the password manager 1Password (Password Managers: Under the Hood of Secrets Management - Independent Security Evaluators, 2020). While the proposed solution would also represent a target for cyber criminals, it does would not contain user’s passwords but instead vast amounts of substitution ciphers, potentially over 100 per user. This decreases the value of such information to an attacker.

How does this help?

This study again highlights the need for a secure password manager that does not have critical vulnerabilities, the proposed mapping application presents an opportunity to improve on existing password manager concepts while reducing the value of compromise for cyber criminals.

Passwordless Authentication

Relevance to Project

It is important to review the current state of Passwordless authentication mechanisms when researching for a Password Solution.

Discovery

In 2004, Microsoft predicted users would be using passwords less and less. (Kotadia, 2019)
Despite this prediction 15 years ago, passwords are still heavily used. Arguably, due to the proliferation of online services, users have begun to rely on passwords more. One thing that has changed is the availability of MFA, though this typically encompasses passwords as the "something you know" factor.

Microsoft also launched Windows Hello in 2015, this is facial recognition system where the users face is used to authenticate. However, in December 2017, German security company SySS found a way to bypass the system and authenticate successfully in many cases using a photograph of the user. (Grasmueck, 2019)

A cryptosystem should be secure even if everything about the system, except the key, is public knowledge. This is known as Kerckhoff's principle. Facial Recognition for authentication in systems arguably does not conform to this, if you consider the user's face to be the key.
You can forget your password, but that does not mean somebody else is more likely to know it. However, if you are using "something you have" as Passwordless authentication, losing it means it may fall or has fallen into the hands of an adversary.

In 2013, the Fast Identity Online Alliance (FIDO) was formed with the view to create authentication standards that reduce over reliance on passwords. While they have been moderately successful, these standards require implementation on the part of each service. This leaves the user vulnerable on web application that are not FIDO compliant. (FIDO Alliance - Open Authentication Standards More Secure than Passwords, 2020)

How does this help?

This research demonstrates that although technology leaders want to move away from passwords, the reality is that for the short term they are here to stay, particularly in the context of MFA. This further emphasizes the point that there is still a need for users to be using strong passwords on the client side and a requirement on services to store passwords on the server side.

Researched Technology Stack

MySQL

MySQL will be used as the backend database to store the map for each user, MySQL was chosen as it is easy simulate using a tool like XAMPP. The database will store the unique maps for each user along with their login credentials for secure authentication.

PHP
A secure login mechanism will be required on the webserver to only allow authenticated users to access the map belonging to them. The webpages and security will be implemented using PHP due to previous experience using PHP for previous college projects.
Using PHP will also illustrate a true understanding of common web vulnerabilities and how to protect against them.

PowerShell
PowerShell may be used on the client side to communicate with the web service and retrieve the maps from the server, using PowerShell will allow the copying of the transformed password to the clipboard for pasting into the login field for the users accounts. It also supports the automatic clearing of the clipboard after X seconds, reducing the likelihood of the password from being leaked via the clipboard.

JavaScript
JavaScript will be used to perform the mapping of the password in a Browser Extension implementation. JavaScript is supported by all modern Web Browsers and will allow the installation of the extension on any desktop OS. JavaScript can also perform ajax web requires and so the extension will be able to communicate with the backend web server for the core mapping functionality, and of course the necessary functionality such as log in and map creation.

Infrastructure
Amazon Web Services
Amazon Web Services (AWS) global computing, data storage, analytics, application and deployment services platform. Know as the leader in Infrastructure as a service (Iaas). Many large organisations such as Netflix, Workday and Samsung use AWS to enable their digital flexibility and make their digital transformation efforts a reality (Computing, Computing and Computing, 2020). An Elastic Compute 2 (EC2) instance will be used to run the webserver and database. These instances enable services to be deployed very rapidly and more resources can be allocated to the machine easily if demand increases.
AWS will be used to host the web application component of this project for several reasons. The application will be developed in a Linux environment, which AWS is more suited to deploying. The free-tier of AWS services will enable this project to be up and running for smaller cost. AWS is trusted by several high-profile customers. AWS was recommended by a lecturer at IT Carlow.

Microsoft Azure
Microsoft Azure is a rival of AWS and offers many similar services. The pricing is comparable but Azure is more expensive for this project, the key difference for this project is that there is less flexibility when it comes to non-windows infrastructure and that would be a hindrance in the development of this project.

Summary and Conclusion

Users of services that require authentication have many passwords they must try to remember. The research shows that users struggle to remember passwords once they have 7 or more unique passwords, on average, there are 130 accounts associated with an email address, each account likely requiring a password. This presents a huge memory issue for users.
One of the ways users can cope with this problem is using a traditional password manager. One of the drawbacks to this is that if an attacker gets the master password for the password manager, they will get access to all a user's accounts. It is also not ideal for a third-party service to be storing your passwords on the web even while encrypted.

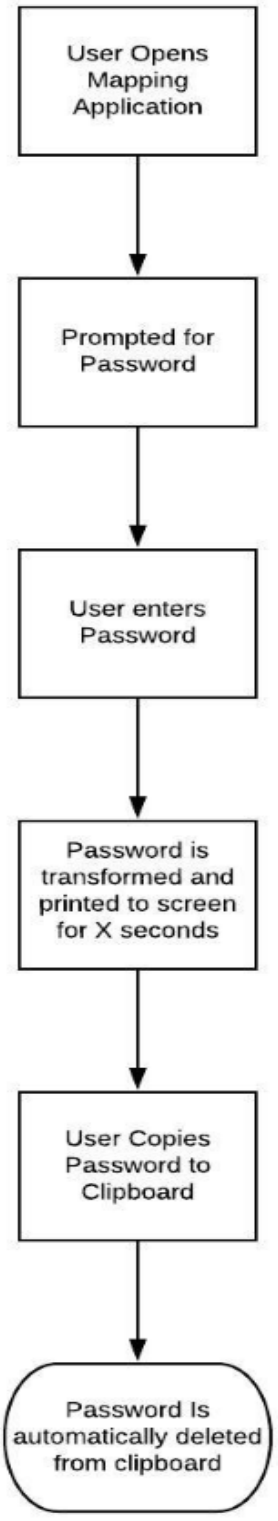
The research also shows that users struggle to create strong passwords even when actively trying to, as they often follow common patterns or end up recycling passwords or using substrings of passwords across sites. This can be addressed not only through user education but also through using the proposed keyboard mapping solution which effectively protects against dictionary attacks, were the service to be compromised. It has the added benefit of allowing users to use passwords that are easier to remember because in fact, the user is just remembering a keystroke pattern with the images on the keyboard not necessarily mapping to the key they represent. E.g. User presses the "Q" key but "W" is inputted.

Technology leaders in the industry in general want to go Passwordless, at present this is not a practical solution as there are flaws with solely using other methods of authentication. Passwords are widely used and understood and relatively inexpensive for services to implement. While the use of "something you have" and "something you are" means of authentication are best practices when it comes to MFA, it is my opinion that the world is not ready to go completely Passwordless, as this implies no longer using the "something you know" factor of authentication.

In conclusion, the proposed solution is for making passwords easier for users to remember, while making them stronger and resistant to common attacks. In contrast the classic password manager, this application does not save the users password anywhere, as the user must remember their simple phrase and it will be transformed by the application.
I believe this to be a more viable solution than the classic password manager, especially for high-value accounts. Two suggested proof of concept implementations are below.

Implementations

PowerShell

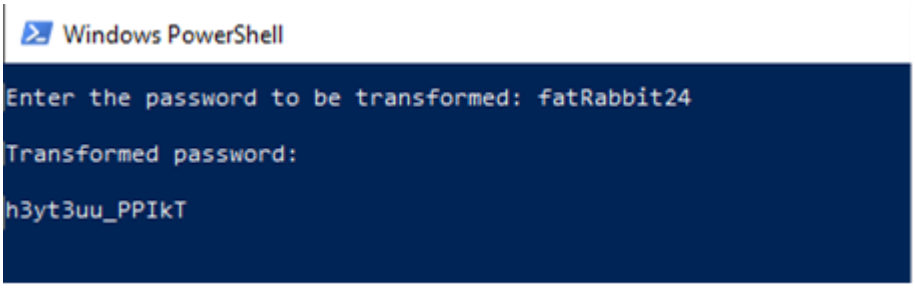


User opens a console application that maps strings, the keyboard mapper. User authenticates with the service.

The mapping application makes a call to the webserver if the map is not saved locally, downloading the relevant map.

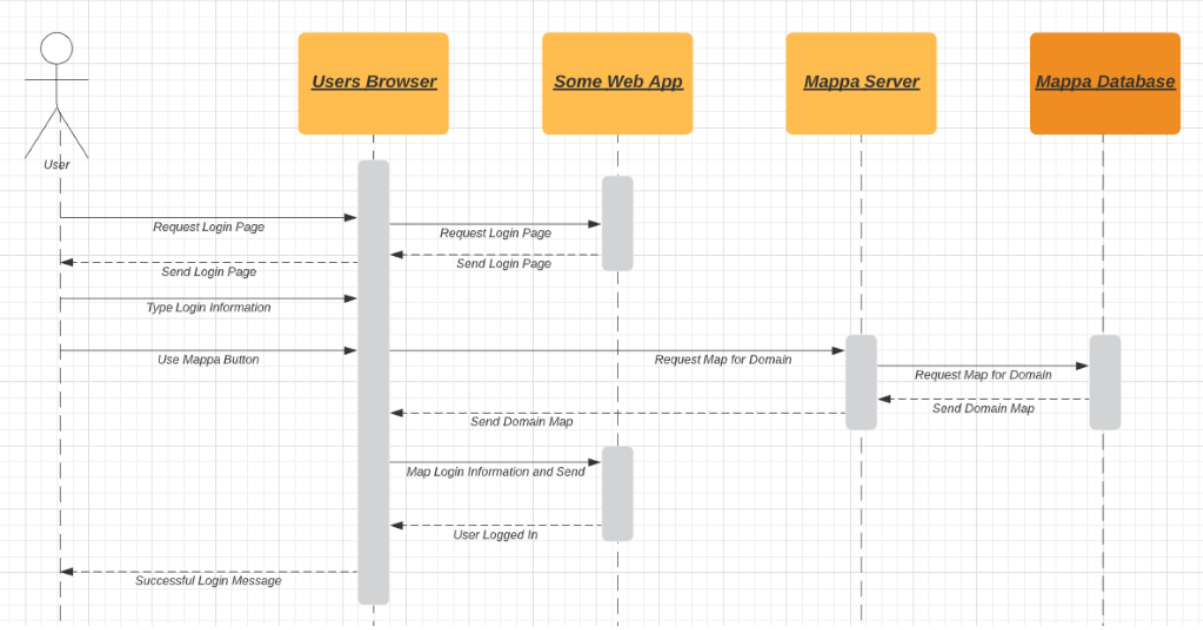
User enters the password from their perspective.

The transformed password is automatically copied to the clipboard and ready to paste into the relevant application.



Clipboard deleted using command: *"cmd.exe /c echo off | clip"*
This is done automatically by the application. The user can now paste the transformed password into the login field of the relevant service.

Web Browser Extension



The diagram above illustrates the use of the Password Mapping application as a Web Browser Extension, the process is ultimately the same for the user as implementation #1 except for that the mapped password will be input into the 3rd party site automatically and the passwords as more secure as they are never on the clipboard.

While this implementation reduces the type of application the Password Mapper can be used on to Web Browser based services, it increases the OS's that it can be run on, as a modern browser running Windows/Mac/Linux will equally support the JavaScript based extension.

Glossary

Hash

A one-way mathematical function called a hash function outputs a message digest, message digests uniquely identify a piece of data. Message digests are also sometimes called hashes.

Keylogger

A piece of malicious computer code that records every keystroke made by the user, typically in order to acquire usernames and passwords.

Shoulder Surfing

Observing what a user is doing from behind, over their shoulders.

Brute Force Attack

Trial and error method use to obtain credentials, where every possible combination is tried. This is the digital equivalent to trying every key on the keyring.

Dictionary Attack

Similar to a brute force attack, though instead of every possible combination, a large set of more likely words (keys) and generated and used.

CISSP

Short for Certified Information Systems Security Professional, a cyber security certification from (ISC)².

Cipher

An algorithm for performing encryption or decryption.

MFA

Multifactor authentication.

Cryptosystem

A suite of cryptographic algorithms needed to implement a particular security service.

Bibliography

Rob Picheta, C. (2019). The most commonly hacked passwords, revealed. [online] CNN. Available at: https://edition.cnn.com/2019/04/22/uk/most-common-passwords-scli-gbr-intl/index.html [Accessed 29 Oct. 2019].

Digital Guardian. (2019). Uncovering Password Habits: Are Users' Password Security Habits Improving? (Infographic). [online] Available at: https://digitalguardian.com/blog/uncovering-password-habits-are-users-password-security-habits-improving-infographic [Accessed 29 Oct. 2019].

Raza, Mudassar & Iqbal, Muhammad & Sharif, Muhammad & Haider, Waqas. (2012). A Survey of Password Attacks and Comparative Analysis on Methods for Secure Authentication. World Applied Sciences Journal. 19. 439-444. 10.5829/idosi.wasj.2012.19.04.1837.

Zhao, Huanyu & Li, Xiaolin. (2007). S3PAS: A Scalable Shoulder-Surfing Resistant Textual-Graphical Password Authentication Scheme. 467 - 472. 10.1109/AINAW.2007.317.

Enterprise.verizon.com. (2019). [online] Available at: https://enterprise.verizon.com/resources/reports/DBIR_2018_Report.pdf [Accessed 29 Oct. 2019].

Cimpanu, C. (2019). LastPass bug leaks credentials from previous site | ZDNet. [online] ZDNet. Available at: https://www.zdnet.com/article/lastpass-bug-leaks-credentials-from-previous-site/ [Accessed 31 Oct. 2019].

Pilar, D., Jaeger, A., Gomes, C. and Stein, L. (2012). Passwords Usage and Human Memory Limitations: A Survey across Age and Educational Background. PLoS ONE, 7(12), p.e51067.

Digital Guardian. (2019). Uncovering Password Habits: Are Users' Password Security Habits Improving? (Infographic). [online] Available at: https://digitalguardian.com/blog/uncovering-password-habits-are-users-password-security-habits-improving-infographic [Accessed 31 Oct. 2019].

Ur, B., Noma, F., Bees, J., Segreti, S., Shay, R., Bauer, L., Christin, N. and Cranor, L. (2019). "I Added '!' at the End to Make It Secure": Observing Password Creation in the Lab. [online] Usenix.org. Available at: https://www.usenix.org/conference/soups2015/proceedings/presentation/ur [Accessed 31 Oct. 2019].

Kävrestad, J., Eriksson, F. and Nohlberg, M. (2019). Understanding passwords – a taxonomy of password creation strategies. Information & Computer Security, 27(3), pp.453-467.

Florencio, D. and Herley, C. (2019). A Large Scale Study of Web Password Habits. [online] Microsoft Research. Available at: https://www.microsoft.com/en-us/research/publication/a-large-scale-study-of-web-password-habits/ [Accessed 31 Oct. 2019].

Morris, R. and Thompson, K. (1979). Password security: a case history. Communications of the ACM, 22(11), pp.594-597.

Li, Z., He, W., Akhawe, D. and Song, D. (2019). The Emperor's New Password Manager: Security Analysis of Web-based Password Managers. [online] Usenix.org. Available at: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/li_zhiwei [Accessed 31 Oct. 2019].

Kotadia, M. (2019). Gates predicts death of the password. [online] CNET. Available at: https://www.cnet.com/news/gates-predicts-death-of-the-password/ [Accessed 31 Oct. 2019].

Grasmueck, D. (2019). SYSS-2017-027: Biometricks: Bypassing an Enterprise-Grade Biometric Face Authentication System. [online] Syss.de. Available at: https://www.syss.de/pentest-blog/2017/syss-2017-027-biometricks-bypassing-an-enterprise-grade-biometric-face-authentication-system/ [Accessed 31 Oct. 2019].

Computing, A., Computing, A. and Computing, A., 2020. *Advantages And Disadvantages Of Cloud Computing - Intellipaat*. [online] Intellipaat Blog. Available at: <https://intellipaat.com/blog/tutorial/amazon-web-services-aws-tutorial/advantages-and-disadvantages-of-cloud-computing/> [Accessed 2 April 2020].

Independent Security Evaluators. 2020. *Password Managers: Under The Hood Of Secrets Management - Independent Security Evaluators*. [online] Available at: <https://www.ise.io/casestudies/password-manager-hacking/?fbclid=IwAR1OB0ztP5ChYlqRrmXegKgwcNhpXWlKzFucN3XEH2JWP4NfM9vk6B_EMdY> [Accessed 14 April 2020].

FIDO Alliance. 2020. *FIDO Alliance - Open Authentication Standards More Secure Than Passwords*. [online] Available at: <https://fidoalliance.org/> [Accessed 14 April 2020].