**Mini Project Report on**



**Notes and Password Manager**



**Submitted in partial fulfillment of the requirement for the award of the degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE & ENGINEERING**

**Submitted by:**

**Student Name**  **University Roll No.**

**Ayush Vyas 2018755**

***Under the Mentorship of***

**Ms. Tanusha Mittal**



**Department of Computer Science and Engineering**

**Graphic Era (Deemed to be University)**

**Dehradun, Uttarakhand**

**January 2023**



**CANDIDATE’S DECLARATION**

I hereby certify that the work which is being presented in the project report entitled **“Notes and Password Manager”** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineeringof the Graphic Era (Deemed to be University), Dehradun shall be carried out by the under the mentorship of  **Ms Tanusha Mittal** Department of Computer Science and Engineering, Graphic Era (Deemed to be University), Dehradun.

Name:  Ayush Vyas   University Roll No.: 2018755

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Description** | **Page No.** |
| Chapter 1 | Introduction  1.1 Brief information  1.2 Problem Statement  1.3Passwords's Historical Context  1.4 Ideal Password Manager  1.5 Types of Password Manager  1.6 Python Password Manager | 4  4-5  5-6  6  6-7  7 |
| Chapter 2 | Literature Survey  2.1 Related work | 8-9 |
| Chapter 3 | Methodology  3.1 Library used  3.2 Functions used  3.3 Method used | 10  10-11  11 |
| Chapter 4 | Result and Discussion | 12-14 |
| Chapter 5 | Conclusion and Future Work  5.1 Conclusion  5.2 Future Work | 15  16-17 |
|  |  |  |

References 18

**Chapter 1**

**Introduction**

**1.1 Brief information**

The Internet of Things (IoT) has gained enormous global popularity in the modern era. The vast majority of what are referred to as "smart devices" can connect to the Internet and access information located anywhere in the globe. And whether we like it or not, we need to go through some layers of authentication in order to access the services that these gadgets have to offer. Web services like UBa student accounts, administrative platforms, moodle.org, Gmail, Facebook, Yahoo mail, and others are examples of the services. Passwords are one of the most popular authentication mechanisms used today, requiring users to enter their login and password before using particular services. With the threat of cybercrime present in today's world.



**1.2 Problem Statement**

There are then primarily two issues. How to develop secure passwords for internet accounts comes first, followed by how to manage these passwords (remembering these passwords and keeping these passwords safe). We can all attest that the primary factor in a large number of cyberattacks is password compromise. Because so many people still do not employ secure password management procedures, research has shown that two out of every three breaches involved attackers utilising stolen or improperly used credentials. My research has led me to the conclusion that the majority of students, employees, and internet users generally rely solely on memory to remember their passwords. The mere fact that someone relies solely on memory is sufficient evidence that they are not using secure password procedures since if they can remember

**1.2.1 Solution to the Raised Problem**

Software programmes for managing passwords are one solution to the issue because they meet all the standards for secure password management. They assist us in generating strong passwords for all of our online accounts, whether they be for students, staff, e-commerce sites, etc. We can remember an infinite number of passwords thanks to them. They are often quick, effective, and simple to use, and the majority include extra features like auto-fill to reduce or completely do away with the need for data entry during an online transaction or account registration. We can see that password management software provides the convenience we need to keep track of all of our varied passwords. Each and every account must have a different password to ensure secure password management. Passwords must be lengthy, complex, and contain letters, numbers, and symbols.

**1.3 The Passwords's Historical Context**

Known for being a pioneer in the creation of time-sharing operating systems, Fernando J. Corbató is a well-known American computer scientist. (July 1st, 1926) born in Oakland, California 1950 saw Corbató graduate with a bachelor's degree from the California Institute of Technology, and 1956 saw him get a doctorate in physics at the Massachusetts Institute of Technology. After graduating, he started working at the Computation Center at MIT, and in 1965, he was promoted to professor. It is said that Corbató was the first to employ passwords to protect access to files on a big computer system. Since sophisticated hacker networks and password-cracking software were not yet available, the original passwords were straightforward and simple to store. But it was also simple to trick the system. Allan Scherr, a Ph.D. researcher who had access to CTSS in 1962, printed all of the passwords saved there.

**1.4 Ideal Password Manager**

A password manager is a piece of software made specifically for organising and storing passwords and other private information.

Despite the fact that password managers have several advantages, such as quicker authentication, secure data storage, and protection against brute force and phishing assaults, not many users have started using them [7]. Only 1% of the 836 employees who responded to a survey in a large firm said they use a password manager, per the study [3, 7].

According to the study by Fagan et al. [6], although users of password managers cited their value and convenience, others cited security concerns as the primary deterrent to using them.

The investigation of the emotional differences between "users" and "non-users" also finds that participants.

**1.5 Types of Password Manager**

We can divide password managers into multiple categories based on their implementation and provided functionality [8].

An ideal password manager should include: ∙

**1.5.1 Securely stored personal information** – The personal information written or inserted into a password manager should be stored in an encrypted form. The most common practice is to protect the vault of the password manager with a master key. The master key is essentially a password to enter the vault of passwords to keep the vault secure. Alternatively, a token can be used for the same purpose. ∙

**1.5.2 Password Generator** – Users can generate stronger and safer passwords via password meter or generator build inside a particular password manager. The study of web password habbits by Florencio et. al. [2] states that an average user has 6.5 passwords, shared across 3.9 different services and thus should have a unique password for each service. It ensures that if one account gets breached, other accounts will not be compromised. ∙

**1.5.3 Faster authentication** – Password managers offer various functionality tools including autofilling passwords. The manager 2 2. Ideal Password Manager fills in the information automatically without the involvement of the user. It can even execute login protocols and visit pre-set domains which prevents phishing attacks.

**1.5.4 Synchronization** – Users typically share passwords across multiple devices, so it is convenient to have shared access to the database. The synchronization can be managed automatically or manually. 2.1 Usability of the Ideal Password Manager

**1.6 Python Password Manager**

Python is a popular programming language for creating password managers because of its user-friendliness, substantial library, and vibrant developer community. Numerous articles may cover the usage of Python to implement different password manager functions, including secure password storage, password generation, and password sharing.

**1.6.1Security considerations**:

The degree to which a password manager is used determines how secure it is. As a result, numerous articles may cover the best techniques for ensuring the security of password managers created with Python, including the use of encryption and secure communication protocols, the use of salted hashes for password storage, and the significance of user authentication and access control.

**1.6.2 Comparison with other languages**

Python can be compared to other programming languages like C++, Java, and JavaScript in terms of how well suited it is for creating password managers. Some articles may compare Python's advantages and disadvantages to those of other programming languages and highlight Python's particular advantages for password management.

**1.6.3 Comparison with other ready-made password managers**:

Some articles may compare the features of various ready-made password managers and go over the advantages and disadvantages of creating a password manager from scratch in Python as opposed to utilising a ready-made programme.

**1.6.4 Examples and case studies**:

Some articles may include case studies and examples of password managers that have been created in the real world using Python, as well as discussions of their functionality, security issues, and lessons learned

**Chapter 2**

**Literature Survey**

**2.1 Related Work**

Password manager research has clear connections to applied cryptography and penetration testing.

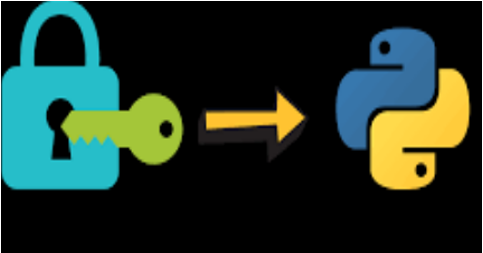
The body of our work will therefore return to citing a variety of academic papers on the security of more well-known password managers and penetration testing write-ups created by security auditing teams, though we will also cite some papers that demonstrate proof of concept or have otherwise made contributions to the field. Additionally, we did our best to include all audits conducted on the three password managers we looked into. The first research on password managers was done in 2003 by Luo and Henry, who showed a proof of concept and implemented a more efficient password manager than Microsoft Passport.

According to a recent study , 41% of respondents include at least one piece of personal information, such as a birthday, in their passwords, while others capitalise a letter to comply with password policy. Ciampa, a security researcher, examined 32 million leaked passwords and discovered that only 12% of passwords were 9 characters or longer . Password reuse is not secure, but it is memorable and 91% of participants reuse at least one of their passwords for multiple accounts. The MTurk study found that participants reused 71% of their passwords on average, confirming previous research that found widespread password reuse. Furthermore, 59% of participants reuse passwords for multiple accounts as a result of the

Writing down passwords in a secure location or using password managers can be a promising solution for password reuse. Password managers provide the advantages of strong passwords and uniqueness over other entry methods , while also improving usability by providing autofill login forms [. Furthermore, password managers generate, store, and encrypt passwords, while users only need to remember one master password ; Komanduri et al. state that when users use memory aids, they create stronger passwords, which can encourage them to use password managers . Stobert and Biddle discovered in a study on the use of passwords among experts and non-experts that the majority of non-experts use browser-saving features whereas the majority of experts use a dedicated password manager.

According to a recent interview study  users of browser password managers are motivated by convenience, whereas users of separate password managers are motivated by increased security. According to a study on the security practises of experts and non-experts  the usability drawbacks of password managers are more difficult to deal with for non-experts, implying that the low adoption rate of password managers may be due to an ingrained mental model. The results of a survey study on smartphone password managers  show a number of rejecting factors, including usability, lack of awareness and trust, security concerns, device memory, battery, and control. A password study  discovered that some participants do not use passwords.

To summarise, researchers  state that current password managers and browsers do not prevent password reuse, and that this should be researched further while maintaining a positive user experience with password managers. Similarly, researchers  wonder why users of password managers continue to use weak passwords and reuse passwords, and they recommend further research to better understand and address the issues that keep users from using password managers. One study  suggested that password managers be improved in terms of usability before recommending them to people, whereas other studies  state that current forms of password managers may not be complete solutions. To improve trust and visibility, Stobert and Biddle  recommend integrating password managers into browsers and operating systems.





**Chapter 3**

**Methodology**

**3.1 Libraries Used**

**3.1.1 Hashlib**

Hashlib is a python library that contains hash functions for generating cryptographic hashes such as SHA-1, SHA-256, and SHA-512. When a user attempts to log in, it can securely store passwords by hashing them and comparing the hash values.

**3.1.2 secrets**

secrets is a Python standard library module that allows you to generate cryptographically secure random numbers and strings. It can be used to generate one-of-a-kind salts for salting passwords before hashing them.

**3.1.3 csv**

csv is a Python built-in library that allows you to read and write CSV (Comma Separated Values) files. It can be used to store hashed passwords as well as associated information like site, username, salt, and hash.

**3.2 Functions Used**

The script contains several functions that perform various tasks:

**masterHashCompare(password):** compares the provided password to a salted hash stored in a master password file, returning True if the passwords match, False otherwise.

**createMasterHashFile(password):** if no salted hash master password file exists, it is created, and the passwords CSV file is initialised.

**hashPassword(password, length)**: uses SHA-512 to create a salted hash of the given password and returns the salt and the number of characters in the hash.

**writeCsv(site, username, salt, hash)**: adds a line to the passwords CSV file containing the specified site, username, salt, and hash.

**readCsv(site)**: reads the passwords CSV file and returns the line corresponding to the given site if it exists.

**deleteCsv(site):** deletes the line for the specified site.

**3.3 Method Used**

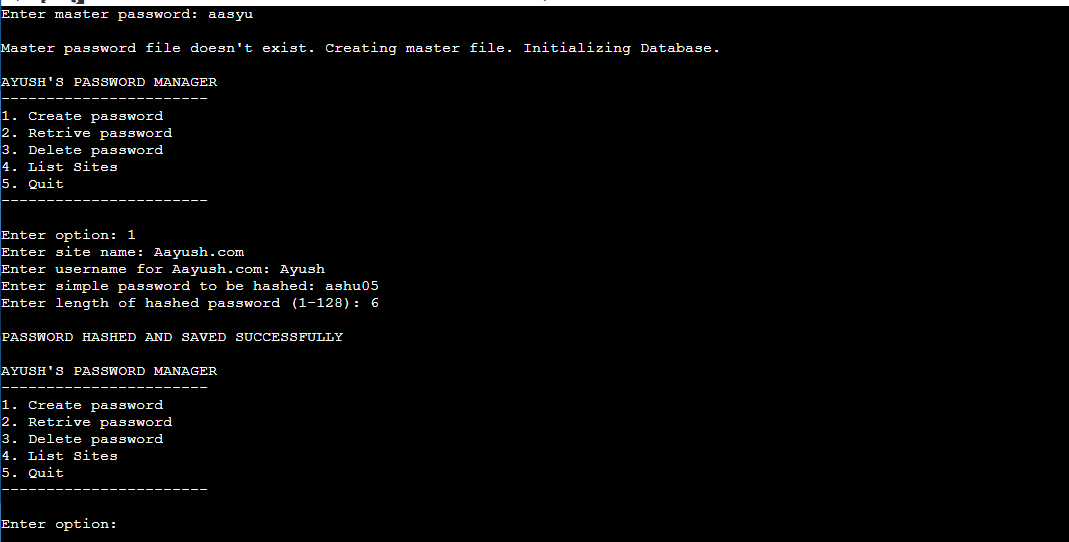
This python password manager hashes and salts passwords with SHA-512 and stores them in a simple .csv file

Passwords are hashed and salted using the hashlib library, while random salts are created using the secrets library and saved in a CSV file using the csv library. Users may create, retrieve, remove, and list password entries by site using the software. In order to use the password manager, the user must input a master password in the program's master password function. To achieve this, a salted hash of the master password entered is compared to the master password kept in a separate file.  
  
SHA-512 is widely used in a variety of applications, such as digital signatures, message authentication codes, and key derivation functions. It is used, for example, in the Digital Signature Algorithm (DSA), which is a digital signature standard in the United States. It is also used to ensure the integrity of data transmitted over the internet in various secure communication protocols such as SSL and TLS.

**Chapter 4**

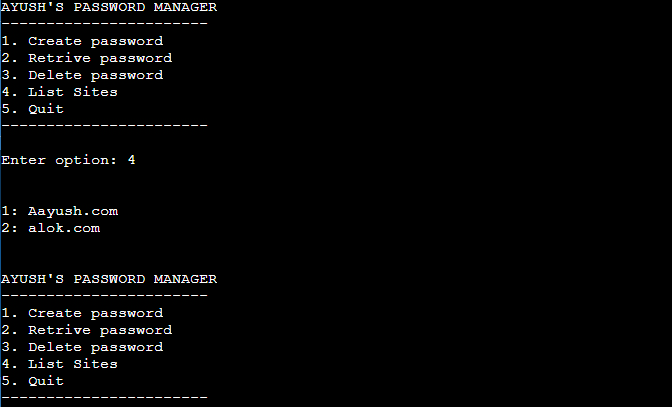
**Result and Discussion**

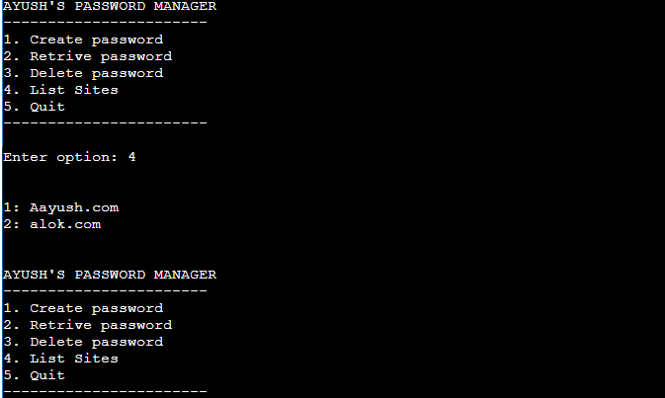
In fig 4.1 we have created a unique master password for our password manager . then we create a site and made a password for it



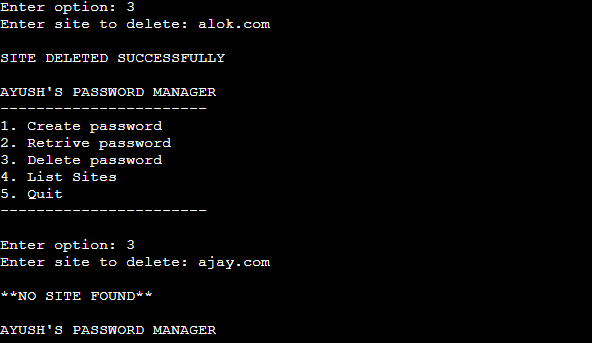
**Fig4.1 Creating a site and password**

In fig 4.2 we will make a new site and will try to retrive the password for it. Here the master password will be same.



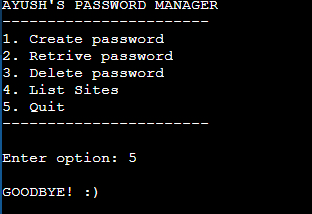
**Fig 4.2 Retrive Password**

In fig 4.4 we will se how to delete a site. To delete a site simply just press option 3 and desired site will be deleted successfully .



**Fig 4.4 deletion of site**

To exit from code just press 5 in menu and will you be exited with a GOODBYE!:) message.



**Fig 4.5 Quit**

**Chapter 5**

**Conclusion and Future Work**

**5.1 Conclusion**

Password-based authentication is known to be insecure these days.

Several websites, including LinkedIn and eHarmony, have already been compromised.

The data was stolen. A list of Gmail accounts was made public in 2014.One issue is that the user tries to choose passwords that are easy to remember. Another issue is that if he has a strong password, he uses it for all of the applications and websites where he has an account. These methods can increase the likelihood of successful attacks.  
Password managers are available to help with these issues. A password manager is a piece of software or hardware that stores all of your passwords. By using this password manager, the user only needs to remember one password to access all of the others. There are numerous password managers available, each with its own set of features such as encryption, two-factor authentication, password generation, and so on. LastPass, FinalKey, and other web-based password managers are available.  
These password managers guard against various attacks such as SQL injection, key logging, and man in the middle, but they are still vulnerable. David Silver and colleagues, as well as Karthikeyan Bhargavan and Antoine Delignat-Lavaud, demonstrated some attacks on existing password managers . Another issue is that LastPass, a well-known password manager, leaked some private information . As a result, this paper proposed a non-web-based solution. The strategy is to store all passwords encrypted with AES on 128 bits on the phone and to have a device that emulates a keyboard. As a result, the user can use the application on any computer. This approach will help to prevent internet-based attacks. The disadvantage is that the design  
Finally, we can say that we were successful in developing a prototype that can be used on a daily basis, has high security due to multiple layers, and is simple to use.

**5.2 Future Work**This chapter discusses some optional features that can be included in the project.

**5.2.1 Mobile application**

The phone app can be ported to a variety of operating systems. It can, for example, be written for iOS and Windows phones. Multiple users can be accommodated in this manner. This will not be difficult because the application is simple.

**5.2.2 graphical user interface application**

This application, like the phone app, can be ported to different operating systems, such as Windows and iOS. It would be simple to port the application because it is written in C++

**5.3 Prevent various attacks**

This section proposed various approaches to preventing certain attacks that cannot be defeated with the current implementation.

**5.3.1 Attacks through side channels**

The chapter on Attack explains how these attacks work. To prevent such attacks, the hardware should be enhanced with a cryptoprocessor, which can detect attempts to tamper with the device. If someone tries to read the memory, it can erase the memory, a process known as zeroization. It can also prevent timing attacks by using a cryptoprocessor, which attempt to measure the time required for various operations.

**5.3.2 Read memory**

These attacks can also be prevented in this case by using a cryptoprocessor. If the exterior layer is tampered with, the EEPROM memory is cleared, and the attacker is unable to read the memory.

**5.4.3 Malware**

If the computer is infected with malware, the attacker can read the password in plain text. There are some basic defences in this case, such as reinstalling the operating system on the computer to ensure that the computer is not infected. Another option is to change the encryption algorithm from AES to RSA. So, when the hardware device is connected to the computer or when the phone connects to the device

**References**

1. For hashing 512 https://www.geeksforgeeks.org/implement-secure-hashing-algorithm-512-sha-512-as-functional-programming-paradigm/
2. Python Libraries : https://www.geeksforgeeks.org/libraries-in-python/
3. <https://auth0.com/blog/hashing-passwords-one-way-road-to-security/>
4. The Evolution of Password Manager : https://www.bluespace.tech/blog/evolution-of-password-manager/