SeCrypt: A Cloud-Based Password Wallet and Manager

Prof. Shweta Joshi[1], Vaibhav Kunjir[2], Pranjal Mavale[3], Abhishek Godase[4]

Department of Computer Engineering, Anantrao Pawar College of Engineering and Research, Pune, India

---***---

1 Abstract— Although various solutions have been suggested for password management systems, most methods require the support of external physical hardware infrastructure. Due to the increase in complexity and cost of set-up of supporting hardware requirements, scalability will always be an issue with such systems. In this paper, we present the design of a Cloud based password management system. The proposed method uses Cryptography for better encryption and is supported by a web-based architecture, for easily managing passwords and providing an indoor cloud based password manager system .

Key Words: AWS, cryptography, Encryption, Cloud computing, AES, SeCRYPT, etc.

1. Introduction

The purpose with this project is to find out how todays password handling and security works. It will also check, with the help of a survey, how ordinary people handle their own passwords. How often they change them, their view on how secure the passwords are and if they use tools to help keep changing and handle their passwords. Many different tools are available to help users create and handle their passwords but how secure are theses "password management"- tools in reality? In this project, a few selected password management and generator programs will be tested to see how they work. Is it something that's easy to use and what advantages does it bring to everyday users, in regard to letting oneself keep the control of the passwords? The results of the survey, regarding user view on password security, will be compared with information regarding how one should reason to create secure passwords. One can find a lot of advice from both companies and experts on how a secure password should look like. e. Do people use these advices and how do the information reach the everyday user ?

2. Purposed System

The proposed system is a web application designed to address the needs and requirements outlined in the project description. It aims to provide a user-friendly and efficient solution for the target audience by leveraging modern technologies and best practices in web development. The system will be developed using a combination of front-end and back-end technologies to ensure a seamless user experience and robust functionality.

• **System Architecture**:
The web application will follow a client-server architecture, with the front-end and back-end components communicating through well-defined APIs. The front-end will be developed using popular web technologies such as HTML5, CSS3, and JavaScript, while the back-end will utilize a server-side programming language such as Python, PHP, or Node.js. The chosen technology stack will be determined based on the specific requirements of the project.

• **User Interface**:
The proposed system will feature an intuitive and visually appealing user interface. The user interface design will be created with a focus on usability and accessibility, ensuring that users can easily navigate through the application and perform tasks efficiently. The layout will be responsive, adapting to different screen sizes and devices to provide a consistent experience across platforms.

• **Functionality**:
The web application will encompass the following core functionalities as outlined in the project description:

1.**User Registration and Authentication**: Users will be able to create accounts, log in securely, and manage their profiles.

2.**Data Input and Management**: Users will have the ability to input and manage data relevant to the application's purpose, such as creating, updating, and deleting records.

3.**Search and Filtering**: The system will provide powerful search and filtering capabilities to allow users to retrieve specific information quickly.

4.**Reporting and Visualization**: The application will generate reports and provide data visualization options to present information in a meaningful and easily understandable manner.

5.**Notifications and Alerts:** Users will receive notifications and alerts related to important updates or events within the system.

6.**Security and Privacy:** The system will employ robust security measures to protect user data, including encryption, secure communication protocols, and role-based access control.

• Integration:
The proposed system will support integration with other relevant systems or APIs, if required, to enhance its functionality and provide seamless integration with existing workflows or services. This could include integration with

external databases, payment gateways, social media platforms, or other third-party services.

• Testing and Quality Assurance:
The development process will include rigorous testing and quality assurance to ensure the system performs as expected and meets the specified requirements. Various testing methodologies, such as unit testing, integration testing, and user acceptance testing, will be employed to identify and rectify any bugs or issues.

• Deployment and Maintenance:
Once development is complete, the web application will be deployed on a reliable hosting environment or cloud platform. Regular maintenance and updates will be performed to address any issues, implement new features, and ensure the system remains secure and up-to-date with evolving technologies.

Overall, the proposed system aims to deliver a robust, user-friendly, and scalable web application that meets the project requirements. Through careful planning, design, development, and testing, the system will provide an efficient and reliable solution to its intended users.

3. Mathematical Model

1 Variables and Functions:

1.1 Variables:

• express: Represents the Express.js module.
• router: Represents an Express router.
• db: Represents the database connection module.
• signupValidation: Function for validating user registration input.
• loginValidation: Function for validating user login input.
• validationResult: Function from the 'express-validator' module.
• bcrypt: Module for password hashing.
• jwt: Module for JSON Web Tokens.
• auth: Middleware function for user authentication.
• encrypt: Function for encryption.
• decrypt: Function for decryption.
• mysql: Module for MySQL database operations.
• Functions:
• loginHandler: Handles the '/login' route.
• registerHandler: Handles the '/register' route.
• addPasswordHandler: Handles the '/addpassword' route.
• showPasswordsHandler: Handles the '/showpasswords' route.
• decryptPasswordHandler: Handles the '/decryptpassword' route.

5. Route Handlers:

5.1 '/login' route handler:

• Input: loginValidation (Request)
• Output: Response
• Functionality:
Validate the login input.
Query the database for a user with a matching email or name.
If the user is not found, return an error response indicating incorrect email or password.
If the user is found, compare the provided password with the hashed password stored in the database.
If the passwords match, generate a JWT token and return a success response with the token and user information.
If the passwords don't match, return an error response indicating incorrect email or password.

1.2 '/register' route handler:

• Input: signupValidation (Request)
• Output: Response
• Functionality:
Validate the registration input.
Query the database to check if the email or name is already in use.
If the user already exists, return an error response indicating that the user is already in use.
If the user is new, hash the provided password.
Insert the user's name, email, and hashed password into the database.
If the insertion is successful, return a success response indicating the user has been registered.

1.3 '/addpassword' route handler:

• Input: auth (Request)
• Output: Response
• Functionality:
Extract the password and title from the request body.
Encrypt the password using the encrypt function.
Get the user ID from the authenticated request.
Create a MySQL database connection using the provided credentials.
Create a 'password' table if it doesn't exist in the database.
Insert the encrypted password, user ID, and title into the 'password' table.
If the insertion is successful, return a success response.

1.4 '/showpasswords' route handler:

• Input: auth (Request)
• Output: Response
• Functionality:
Get the user ID from the authenticated request.
Create a MySQL database connection using the provided

credentials.
Create a 'password' table if it doesn't exist in the database.
Query the 'password' table for passwords associated with the user ID.
If the query is successful, return a response with the retrieved passwords.

1.5 '/decryptpassword' route handler:

- Input: auth (Request)
- Output: Response
- Functionality:

Decrypt the encrypted password received in the request body using the decrypt function.
Return the decrypted password in the response

4. Database Description

User Information:
• User ID: A unique identifier for each user in the password manager system.
• Username The usemate chosen by the user for their account.
• Email Address: The email address associated with the user's account.
• Password: The encrypted password chosen by the user for their account.

Stored Passwords:
• •Website/Application: The name or URL of the website or application for which the password is stored.
• Username/Email: The username or email associated with the website or application.
• Password The encrypted password for the website or application
• Additional Information: Any additional details or notes provided by the user regarding the stored password (c.a. security questions.

Encryption and Security:

• Encryption keys: information related to encryption keys used to secure user
• Security Settings: User preferences and settings related to password security, such as password length requirements, complexity rules, and password expiration policies.

Access Logs:

• Timestamp: The date and time of access to the password manager account.
• IP Address: The IP address from which the account was accessed.
• Device Information: Details about the device used to access the account (eg, device type ,Operating System).

Account Activity:

• Login/Logout Events: Information about user login and logout events, including timestamps and IP addresses.
• Password Change Events: Details about password changes made by the user including timestamps and previous passwords.
• Account Recovery Events: Records of account recovery processes initiated by the user, including timestamps and associated recovery methods.

5. Vulnerability analysis

In this section, we first define the threat model and assumptions that we consider throughout this paper. We then use an analogy to justify the essential problem of existing BPMs. Finally, we provide a detailed vulnerability analysis regarding without and with a master password mechanism.

5.1 Threat model and assumptions:

"Where a threat intersects with a vulnerability, risk is."For Browser-based Password Managers (BPMs), the threat sources are attackers who want to steal the sensitive login information stored by BPMs.

5.2 Threat model

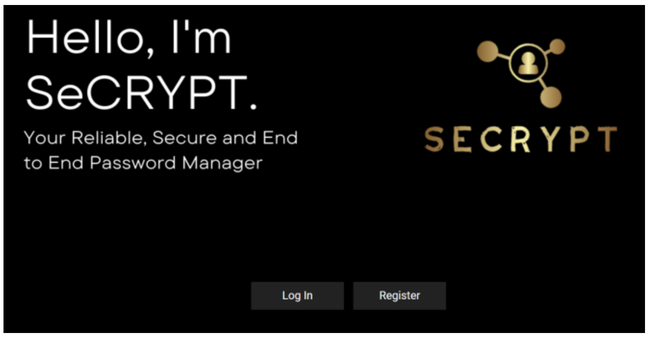
Our basic threat model is that attackers can temporarily install malware such as Trojan horses and bots on a user's computer using popular attacks such as drive-by downloads Moshchuk . The installed malware can then steal the login information stored by BPMs. For example,Stone-Gross et al. inferred that 38% of the credentials stolen by the Torpig bot were obtained from the password managers of browsers, rather than by intercepting an actual login session . Note that the malware can run at the system-level or at the application-level, and can evenbemalicious browser extensions . Indeed, if the occurrences of such threats are rare or do not have high im-pacts, BPMs would not bother to encrypt their stored pass-words in the first place. Therefore, our focus will be on investigating the vulnerabilities of BPMs that could be exploited by potential threat sources to easily decrypt the passwords stored by BPMs. International Journal of Engineering Research & Technolo Published by, www.ijert.org .

5.3 Assumptions

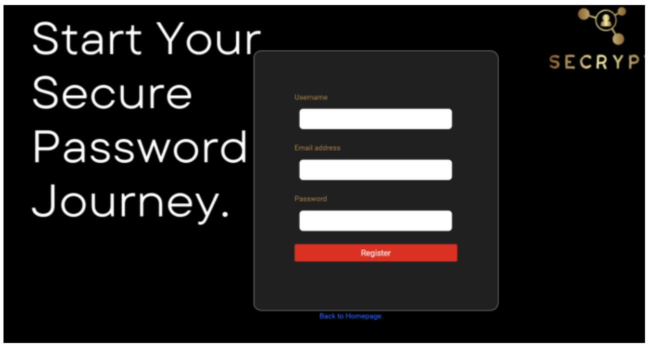
We assume that it is very difficult for the installed malware to further compromise the operating system to directly identify cryptographic keys from a computer's memory (Halderman et al. because this identification often requires elevated privilege and is prone to false positives. We assume that the installed malware can be removed from the system by security-conscious users in a timely manner, so that even though sensitive login information stored by BPMs can be stolen within a short period of time, it is very difficult for attackers to use tools such as keyloggers to further intercept users' passwords for a long period of time.

5.4 Without a master password mechanism:
Through source code analysis, binary file analysis, and experiments, we found that chrome uses the three-key TripleDES algorithm to encrypt a user's passwords for different websites. Chrome saves each encrypted username, encrypted password, and plaintext login webpage URL address into the login table of a SQLite (SQLite Home Page) database file name sign ons. sqlite. The Triple-DES keys are generated once by chrome and then saved into a binary file named key.db starting from the byte offset location 02F90. Although the keys generated on different computers are different, they are not bound to a particular computer or protected by other mechanisms.

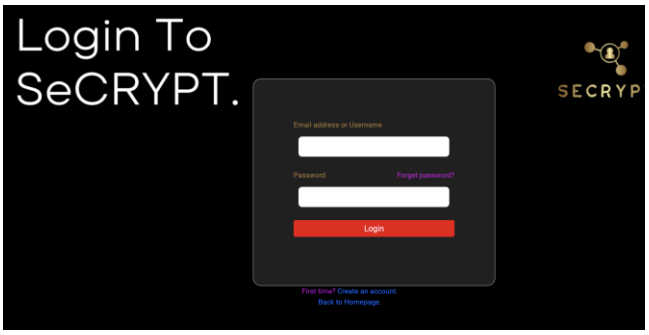
Landing Page



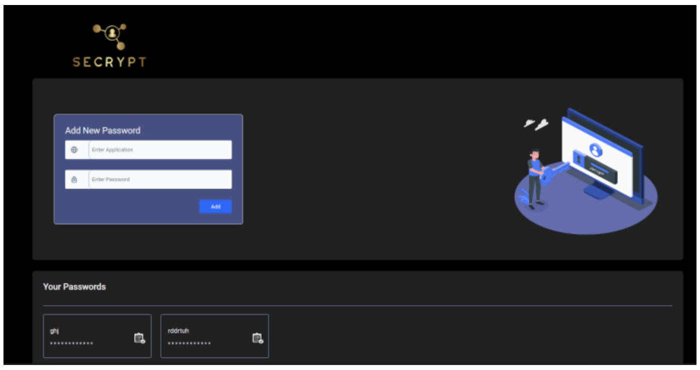
Register Page



Login Page

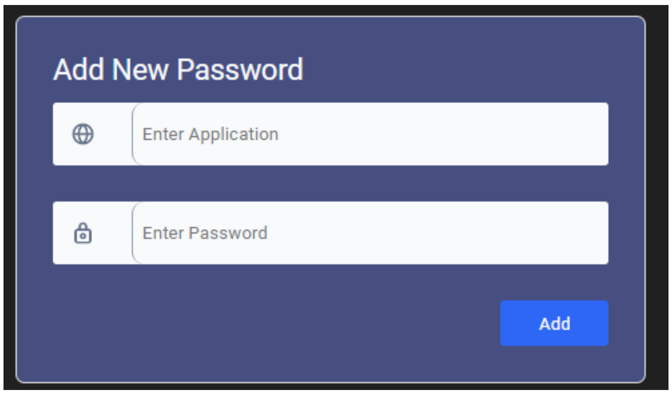


Home Page

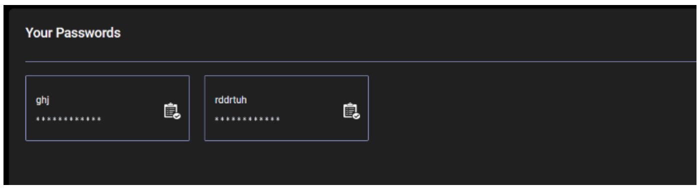


z

Add Password



Show Password



Feature scope

➤ Android/IOS App:In future, we will publish android and IOS app for SeCRYPT for easy access to mobile user.

➤ Password Auto-Fill: We will develop browser extensions so thatpassword can be autolled with minimal user remediation.

Conclusion

A password manager is software that allows us to create new passwords, save and manage our login information. Both not using and using a password manager have hazards, but the risk of not using one much surpasses the risk of using one for most people. If you do decide to use a password management, you should mitigate the danger of doing so by enabling two-factor authentication, planning your master password recovery procedure, and not saving a high-risk password in your password manager.

References

Cusumano Michael "Cloud computing and SaaS
A new computing platforms. Communications of the ACM 534 (2010).
2 Gasti, Paolo, and Kasper B. Rasmussen. "On the security of password manager database formats. European Symposium on Research in Computer Security. Springer,Berlin Heidelberg. 2012.

3. URL: https://www.malwarebytes.com/what-is-

4. Abdullah, Ako Muhamad. "Advanced encryption standard (AES) algorithm to encrypt and decrypt data. Cryptography and Network Security 16 (2017): 1-11.

5. Cramer, Ronald, and Victor Shoup. "Signature schemes based on the strong RSA assumption. ACM Transactions on Information and System Security (TISSEC) 33 (2000):

6. Cunningham, A. (2019, July 17). The Best Password Managers. Wirecutter.
https://thewirecutter.com/reviews/best-password-managers/

7. "Password Protection: How to Create Strong Passwords"
Griffith,http://www.pemag.com/article2/0.2817.2368 485.asp

8. Hoffman, C. (2019, November 14). Why You Should Use a Password Manager, and How to Get Started:
https://www.howtogeek.com/141500/why-you-should-use-a: password-

manager-and-how-to-get-started/

9. LastPass (n.d.) Plans and Pricing Retrieved February 29, 2020 from https://www.lastpass.com/pricingCloud Based password Manager.

10. Gallagher, E. A. (2019). Choosing the Right Password Manager. Serials Review 45(1/2). 84-87. https://doi-org.ezproxy.simmons.edu/10.1080/00987913.2019.16113 10

11. "An Analysis of Password Manager Applications" by C. Rossow, E. Andriesse, and H. Bos, published in the Proceedings of the Network and Distributed System SecuritySymposium in 2017.

12 A Comparative Study of Password Managers" by J. A. Arias-Cabarcos, F. J. González- Castaño, and L. Fernández-Sanz, published in the Journal of Computer and SystemSciences in 2018.

13. "A Survey of Password Security Practices and Behaviors" by J. Bonneau, published in the Communications of the ACM in 2012.

14. "A Survey of Two-Factor Authentication Methods" by A. Al-Frajat, M. Al-Bakri, and I. M.Aluaimi, published in the International Journal of Advanced Computer Science and Applications in 2019.

15. "Password Hashing Competition" by A. Bernstein, D. Bono, J. Lange, and L. Schwabe, published in the Proceedings of the USENIX Security Symposium in 2015.

16. "Password-based Authentication: A Systematic Literature Review" by T. M. Ahmed, M. M. Rahman, and A. Hasan, published in the Journal of Network and Computer Applications in 2017.

17. "A Study of the Usability and Security of Password Managers" by M. Ur, R. Segreti, L Bauer, N. Christin, and L. F. Cranor, published in the Proceedings of the USENIX Security Symposium in 2017.

18. "A Comparative Study of Password Storage Schemes for Web Applications" by A. Juels and R. L. Rivest, published in the Proceedings of the Network and Distributed System Security Symposium in 2014.