MALAYSIAN JOURNAL OF COMPUTING AND APPLIED MATHEMATICS (2023) 6 (2) 1-10

DOI: *https://doi.org/10.37231/myjcam.2023.6.2.87*

Article: Received 02.11.2022; Accepted 16.04.2023; Published 30.09.2023

PASSWORD MANAGER SECURITY USING HONEY ENCRYPTION ALGORITHM AND HONEYPOT TECHNIQUE

Anas Danial✉, Mohd Fadzil Abdul Kadir, Ahmad Faisal Amri Abidin, Mohamad Afendee Mohamed, Nazirah Abdul Hamid, Siti Dhalila Mohd Satar

Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin Besut Campus, Malaysia
✉060216@putra.unisza.edu.my

Abstract: Password managers are crucial tools for securely storing and managing multiple passwords. However, they can become targets for attackers attempting to gain unauthorized access to sensitive user data. In this paper, we propose an approach to password manager security by combining the Honey Encryption algorithm with the Honeypot technique. By implementing Honey Encryption on the authorization process of the password manager, we can effectively divert attackers to a honeypot, which contains a list of fake/honeyword passwords. The honeypot is designed to be high interaction, allowing us to gather valuable information about the attacker, such as their IP address and MAC address. This information can be crucial for further analysis and taking appropriate actions to mitigate the security breach. Our proposed system provides an additional layer of security to password managers, making them more robust against unauthorized access attempts.

Keywords: Password Manager, Honey Encryption, Honeypot, IP address, MAC address.

1. INTRODUCTION

In today's digital age, the security of password managers is of paramount importance as they play a vital role in securely storing and managing the ever-increasing number of passwords [1][2]. Password managers alleviate the burden of remembering multiple complex passwords and provide a centralized solution for users to protect their sensitive login information. However, the persistent threat of brute force attacks poses a significant challenge to the security of password managers, necessitating robust countermeasures [3]. Existing research has focused on various aspects of password manager security, including encryption algorithms, secure storage mechanisms, and authentication protocols [4][5]. While these measures provide a certain level of protection, they often fall short in addressing the vulnerability associated with guessing the master password [6]. Attackers can exploit the limited entropy of human-generated passwords and launch brute force attacks to gain unauthorized access to the password manager system [7].

To bridge this gap and enhance password manager security, this research paper proposes a novel approach that combines the high interaction honeypot technique with the honey encryption algorithm. The high interaction honeypot technique involves the creation of a decoy password manager that emulates the functionality of a legitimate password manager, increasing the complexity for attackers attempting to compromise the system [8]. This approach complements the honey encryption algorithm, which generates decoy passwords that closely resemble real passwords, further confounding attackers' attempts to guess the master password [9]. The primary objective of this research is to mitigate the risk of brute force attacks on password manager security by implementing the proposed high interaction honeypot technique and honey encryption algorithm. By redirecting attackers to the decoy password manager and presenting them with decoy passwords, the system aims to impede their efforts to guess the master password and gain unauthorized access.

To lay the foundation for this research, we will first provide an overview of the background of password manager security, highlighting the vulnerabilities associated with brute force attacks [6][10]. We will then delve into a detailed analysis of the honey encryption algorithm and the high interaction honeypot technique, elucidating their principles and evaluating their effectiveness in preventing unauthorized access [8][9][11][12]. Additionally, we will describe the implementation of the high interaction honeypot technique to create the decoy password manager, along with the strategies employed to engage and deceive attackers,

thereby providing valuable intelligence for analyzing and mitigating potential security threats [13]. By addressing the identified gaps in password manager security, this research aims to contribute valuable insights and practical solutions to strengthen the resilience of password managers and protect users' sensitive information from brute force attacks. The findings of this research will facilitate the advancement of password manager security and lay the groundwork for further exploration and enhancement in this critical domain.

2. RELATED WORKS

2.1 Existing Approaches to Password Manager Security

Previous research has extensively explored various approaches to enhancing password manager security. Encryption algorithms, secure storage mechanisms, and authentication methods have been the focus of many studies in this domain [3]. For instance, Bonneau et al. proposed a comparative framework for evaluating web authentication schemes, aiming to replace traditional passwords with more secure alternatives. While these approaches address certain aspects of password manager security, they often overlook the vulnerability of guessing the master password and mitigating the risk of offline brute force attacks. This gap highlights the need for novel techniques that can effectively thwart unauthorized access attempts and provide robust protection for sensitive user data. Another area of research focuses on password strength and user behavior. Florencio and Herley (2007) conducted a large-scale study on web password habits, revealing the common pitfalls and weaknesses in users' password choices [2]. Their findings emphasized the importance of encouraging users to adopt stronger and more unique passwords to mitigate the risk of brute force attacks. To address this challenge, Chiasson et al. (2008) proposed a persuasive cued click-point approach that aimed to influence users towards creating better passwords [14].

By incorporating visual cues and interactive elements during the password creation process, this approach demonstrated promising results in improving password strength. While these studies shed light on user behavior and password strength, they do not fully address the offline brute force attack scenario. Therefore, there is a need for complementary techniques that go beyond user education and focus on actively preventing unauthorized access to password managers.

2.2 Honey Encryption

Honey encryption is a promising technique for enhancing password manager security by generating decoy passwords that closely resemble real passwords. It increases the complexity of guessing the correct password, making it challenging for attackers. Juels and Ristenpart (2014) introduced honey encryption, demonstrating its effectiveness in preventing unauthorized access to sensitive data. Studies have shown that honey encryption significantly reduces the success rate of offline brute force attacks by creating a large search space and making it computationally infeasible to determine the actual password. Implementing honey encryption in password managers adds an extra layer of security and improves their resilience against offline brute force attacks [9].

2.2.1 Distribution Transforming Encoders

Honey Encryption is based on the concept of Distributed Transforming Encoding (DTE) and forms the core idea of the pure honey encryption technique [15]. In this technique, the space of plaintext is managed through DTE. The message space, denoted as L, is associated with a probability distribution p. DTE encodes the message L into a K-bit seed $S \in \{0, 1\}K$ and decodes the message using the inverse DTE method, decode(S) = L. DTE provides a good model for the message distribution. The honey encryption (HE) framework includes DTE encryption and DTE decryption algorithms, which define the overall functioning of the technique. The Honey Encryption Algorithm involves the following steps: $H \leftarrow Enc(X, L)$, $S \leftarrow \$encode(L)$, $R \leftarrow \${0, 1}n$, $S' \leftarrow H(R, X)$, $C \leftarrow S' \oplus S$. The Honey Decryption Algorithm consists of the following steps: $H \leftarrow Dec(X, (R, C))$, $S' \leftarrow H(R, X)$, $S \leftarrow C \oplus S'$, $L \leftarrow$ decode(S), Return L. Here, H represents a cryptographic hash function, X is the key, L denotes the message, S represents the seed, R is a random string, C is the ciphertext, and $ indicates that the Honey Encryption algorithm may use a certain number of uniform random bits. The Honey Encryption algorithm encodes the message L into S and then encrypts S using a symmetric encryption algorithm with the key X.

Honey encryption provides a high level of message recovery security. To illustrate its functioning, let's consider the example of encrypting soft drink flavors: Apple, Mango, and Orange. Each flavor is encoded as a two-bit string, such as {00, 01, 10, 11}. For instance, if Bob wants to encrypt his favorite soft drink flavor, Mango (encoded as 01), to be sent to Alice under the shared secret key X = 0000, Bob employs a DTE mapping (Figure 1) to convert Mango into the corresponding 2-bit value. Bob selects a random string R and computes S' = H(R, X), assuming S' = (R, 0000) = 11. Then, Bob calculates C = 11 $\oplus$ 01 = 10, which is forwarded to Alice. Alice decrypts C using the shared key X = 0000. So, S' = H(R, 0000) = 11, and S = C $\oplus$ S' = 10 $\oplus$ 11 = 01. By decoding 01, Alice successfully recovers the message as Mango. In the case of an attacker, Eve, attempting to decrypt the message without knowing the key, they might assume a key, such as 1432, resulting in H = (R, 1432) = 00. Consequently, S'' = C $\oplus$ S' = 10. Upon decoding, Eve would obtain decode(10) = Orange, thus illustrating the effectiveness of this new type of encryption in deceiving attackers.

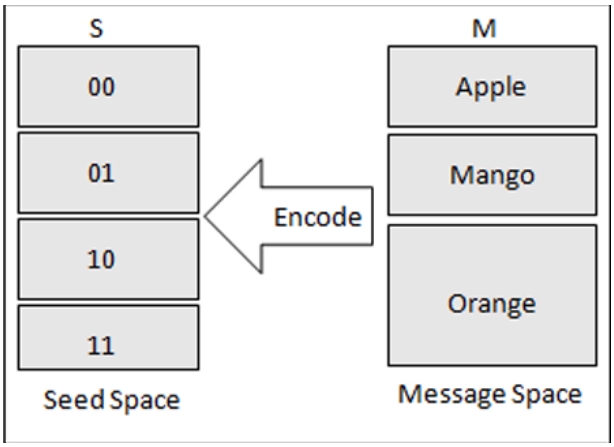

Figure 1 DTE mapping

In this example, the message "apple" (with pm = 1/4) maps to 00, "mango" (with pm = 1/4) maps to 01, and "orange" (with pm = 1/2) maps to {10, 11}, where pm represents the probability distribution over the message space.

2.3 Honeypot Technique

Honeypot techniques, particularly high interaction honeypots, have garnered significant attention in the field of cybersecurity. These techniques involve deploying decoy systems or resources to divert and deceive attackers, ultimately gathering valuable insights into their methodologies and motivations. In the context of password manager security, honeypots play a crucial role in enhancing overall defense mechanisms. One notable study by Spitzner (2002) extensively explored the concept and applications of honeypots, including their role in password manager security [16]. The research highlighted how honeypots effectively redirect and occupy attackers, allowing defenders to observe their activities and gather intelligence on emerging attack patterns. By deploying high interaction honeypots alongside password managers, security professionals can gain valuable information about attacker behavior, potentially leading to the identification and mitigation of vulnerabilities.

Webster (2018) investigated the utilization of the Jeeves language in the context of honeypot sanitization, specifically focusing on password manager security [17]. The article addresses the challenge of safeguarding sensitive information, such as passwords, within a honeypot environment. The author demonstrates the application of Jeeves, a programming language designed for privacy and security, to implement a sanitization mechanism within a honeypot. The process of sanitization involves the removal or obfuscation of sensitive data to prevent unauthorized access or exposure. Webster thoroughly examines the obstacles and considerations involved in effectively sanitizing data within a honeypot, ensuring the protection of sensitive information like passwords. The insights derived from this study offer valuable guidance for augmenting password manager security within high interaction honeypots. By implementing

robust sanitization mechanisms, password managers can effectively safeguard user credentials and thwart attackers from extracting valuable information from the honeypot environment.

2.4 Brute-force Attack

Brute-force attacks have long been recognized as a significant threat to the security of authentication systems. Extensive research has been conducted to understand the nature of such attacks and develop effective countermeasures. In this section, we discuss notable studies that have contributed to the understanding of brute-force attacks and proposed strategies to mitigate their impact. One key area of research focuses on analyzing the impacts of password policies and real-world password composition on the success rate of brute-force attacks. Bošnjak et al. demonstrates the vulnerability of textual passwords and the effectiveness of various attack methods [18]. Their findings revealed that weak password policies and predictable password patterns significantly increase the vulnerability to brute-force attacks. The study emphasizes the importance of enforcing strong password policies to enhance resistance against such attacks.

Brute-force attacks have become more sophisticated, prompting the exploration of advanced techniques by attackers. Han et al. introduced probabilistic password cracking, leveraging statistical patterns in password creation and usage [19]. This approach allows attackers to prioritize likely password combinations, reducing cracking time. Adaptive password policies that consider evolving attack techniques are crucial, as highlighted by this research. Password managers offer a practical solution by generating and storing complex, unique passwords for each account. They mitigate risks associated with password reuse, a common practice increasing vulnerability to brute-force attacks. Combining adaptive password policies and password managers enhances overall password security against these evolving attack techniques. Furthermore, Machine learning techniques have shown promise in addressing brute force attacks. For example, Najafabadi et al. developed a model that utilizes historical attack data and advanced feature extraction to detect and mitigate brute force attacks [20]. This highlights the potential of machine learning in enhancing traditional security measures. In the context of password management, researchers have explored machine learning models to strengthen the detection and prevention of brute force attacks. These models analyze historical data, recognize attack patterns, and adapt to new threats. However, the availability of diverse training data and the selection of appropriate algorithms are important considerations. Integrating machine learning into password manager systems offers a proactive defense against brute force attacks, but further research is needed to optimize their performance in real-world scenarios.

These studies collectively contribute to the understanding of brute-force attacks and provide valuable insights into effective countermeasures. By analyzing password policies, exploring advanced attack techniques, and leveraging machine learning, researchers aim to enhance the resilience of authentication systems against brute-force attacks. The knowledge gained from these studies informs the design and implementation of robust security mechanisms that can withstand sophisticated attacks.

3. METHODOLOGY

The methodology employed in this research paper revolves around the systematic implementation and evaluation of the proposed system framework, as depicted in the diagram. The first step involves the registration process, where users create an account by providing their email and master password. To enhance the security of the master password, honey encryption (DTE) is applied, resulting in an encrypted form that protects against brute force attacks. Once the honey-encrypted master password is generated, it is securely stored in the database along with the cipher key and trueSeed. Additionally, recovery keys are generated and provided to users as a means of account recovery in case of password loss or forgetfulness. This setup ensures the confidentiality and integrity of users' master passwords.

The subsequent stage focuses on user authentication during the login process. The master password entered by the user at the login screen is checked against the stored encrypted master password in the database. If the authentication is successful, the user is granted access to the real password manager, where they can securely store their website names, usernames, and passwords. In the event of unsuccessful authentication, the system provides the user with a limited number of login attempts. If the user fails to authenticate within the allowed attempts, they are redirected to the honeypot Password Manager. This decoy password manager, designed to mimic the real password manager, presents the unauthenticated user with the same website names and usernames they entered. However, the passwords

are replaced with decoy passwords generated from a list of common password combinations. This diversion tactic aims to confuse attackers attempting to gain unauthorized access.

To detect and alert the real/authenticated user of potential security breaches, an intriguing reveal button is integrated into the honeypot Password Manager. When an unauthenticated user attempts to reveal the password for a specific website, their IP address, MAC address, the master password they entered, and the website name are captured. This information is then utilized to send an email notification to the real/authenticated user, effectively informing them of the attempted breach on their password manager account. The methodology also involves evaluating the effectiveness of the system framework. Authentication success rates and the redirection to the honeypot Password Manager are analyzed to measure the system's ability to mitigate brute force attacks. Additionally, the information captured from the reveal button and the alert system provides valuable insights into potential security threats, aiding in the identification and mitigation of vulnerabilities. By following this methodology, this research aims to demonstrate the practicality and effectiveness of integrating honey encryption, honeypot techniques, and an alert system within the password manager framework (Figure 2). The results and discussions derived from this study provide valuable insights into enhancing password manager security, preventing brute force attacks, and safeguarding users' sensitive information.

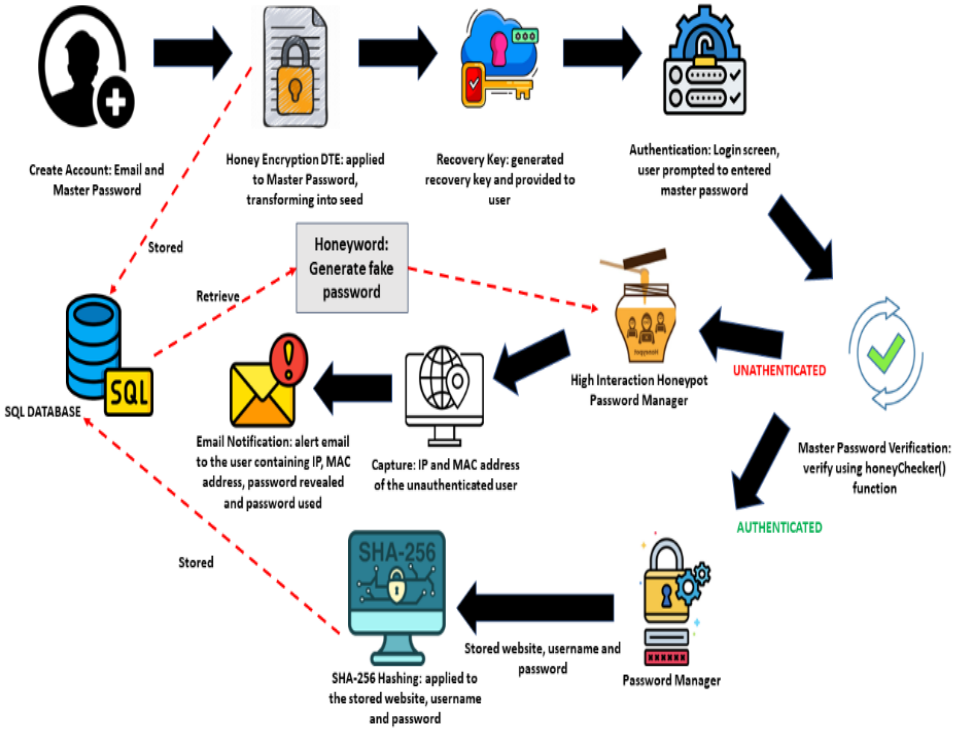

Figure 2 System Framework: Password Manager Security using Honey Encryption Algorithm and Honeypot Technique

4. IMPLEMENTATION

4.1 Honey Encryption

Function: asciiCode(password)
password_integer = Convert password characters to ASCII code
return password_integer
Function: randomSeed(password_length)

```
    min_seed = 10 ^ (password_length - 1)
    max_seed = (10 ^ password_length) - 1
    trueSeed = Generate random integer within range [min_seed, max_seed]
    return trueSeed
Function: honeyEncryption(masterPassword)
    password_integer = asciiCode(masterPassword)
    password_length = length of password_integer
    trueSeed = randomSeed(password_length)
    cipher = password_integer XOR trueSeed
    return trueSeed, cipher
Function: checkPassword()
    attempts = 0
    match = getMasterPassword()
    Retrieve trueSeed and cipher from the database
    m = trueSeed XOR cipher   # Decrypt honey encryption
    honeychecker = m - check_pass
   If honeychecker == 0:
        Redirect to real password manager
    Else if -16384 <= honeychecker <= 16384:
        Redirect to fake password manager
    Else:
        Increment attempts by 1
        If attempts < 5:
            Display error message indicating incorrect password (5 - attempts) times remaining
            Clear password entry field (txt.delete(0, END))
        Else:
            Redirect to fake password manager
Function: generate_honeyword(password)
    check_pass = asciiCode(password)
    If -16384 <= honeychecker <= 16384:
        Select random common password from list as honeyword
    Else:
        Use different random number generator
        Select random common password from list as honeyword
    return honeyword
```

The implemented algorithm utilizes honey encryption to enhance password security. It involves converting the master password into ASCII code and generating a random trueSeed within a specific range. The trueSeed and the ASCII code are combined using XOR to create a cipher. During authentication, the user-entered password is decrypted by XORing it with the trueSeed and comparing it with the original ASCII code. If the difference falls within a specified range, the user is directed to the fake password manager. A zero difference indicates a correct password, leading to the real password manager. Failed attempts exceeding five redirect the user to the fake manager. The algorithm also generates honeywords based on the difference, adding authenticity to the fake manager. Overall, this algorithm strengthens password security by confusing attackers while providing legitimate password management functionality.

4.2 High Interaction Honeypot

The high interaction honeypot implementation goes beyond just creating a decoy password manager; it also involves the generation of fake passwords, capturing detailed information about unauthorized users, and sending email alerts. In this implementation, decoy passwords are generated using the honey encryption algorithm. These decoy passwords closely resemble real passwords, creating a realistic façade within the decoy password manager. This adds an extra layer of deception, making it harder for attackers to differentiate between real and fake passwords. Moreover, the high interaction honeypot implementation includes extensive logging and monitoring mechanisms to capture vital information about unauthorized

users. When an unauthorized user interacts with the decoy password manager, various details are recorded, including their IP address, MAC address, and the master password they attempted to use.

Additionally, if an unauthorized user attempts to reveal the password for a particular website within the decoy password manager, this action triggers the capture of specific information. The captured data includes the IP address and MAC address of the unauthorized user, the master password they entered, and the website name they were trying to access. To ensure the real/authenticated user is promptly alerted about any unauthorized access attempts, the high interaction honeypot implementation incorporates an email alert feature as shown in Figure 3. When unauthorized activity is detected, an email is automatically generated and sent to the real/authenticated user. The email contains detailed information, such as the captured IP address, MAC address, the master password entered by the unauthorized user, and the website name they were trying to access. This email serves as an immediate notification and informs/alerts the real/authenticated user about the breach attempt, enabling them to take necessary action to protect their account.

By combining the generation of fake passwords, capturing detailed information, and sending email alerts, the high interaction honeypot implementation provides a comprehensive security solution. It not only deceives attackers and captures their actions but also ensures that real/authenticated users are promptly informed about any unauthorized access attempts, enhancing the overall security and protection of the password manager system.

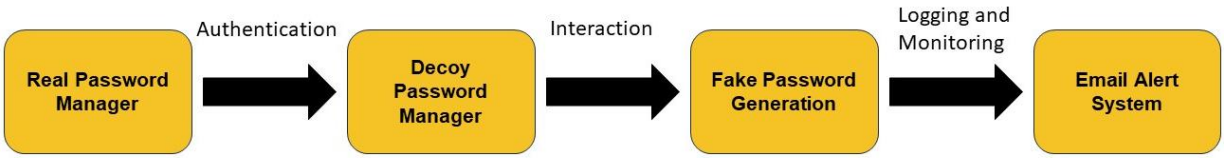

Figure 3 Main components of the high interaction honeypot implementation

5. RESULT AND DISCUSSION

In this section, we present the results of our high interaction honeypot implementation and discuss their implications for enhancing the security of password managers. Through a series of experiments, we evaluated the effectiveness of our approach, focusing on the time taken to open the real password manager compared to the decoy password manager. We also captured critical information during the authentication process and successfully generated email alerts to notify the legitimate user of potential unauthorized access attempts.

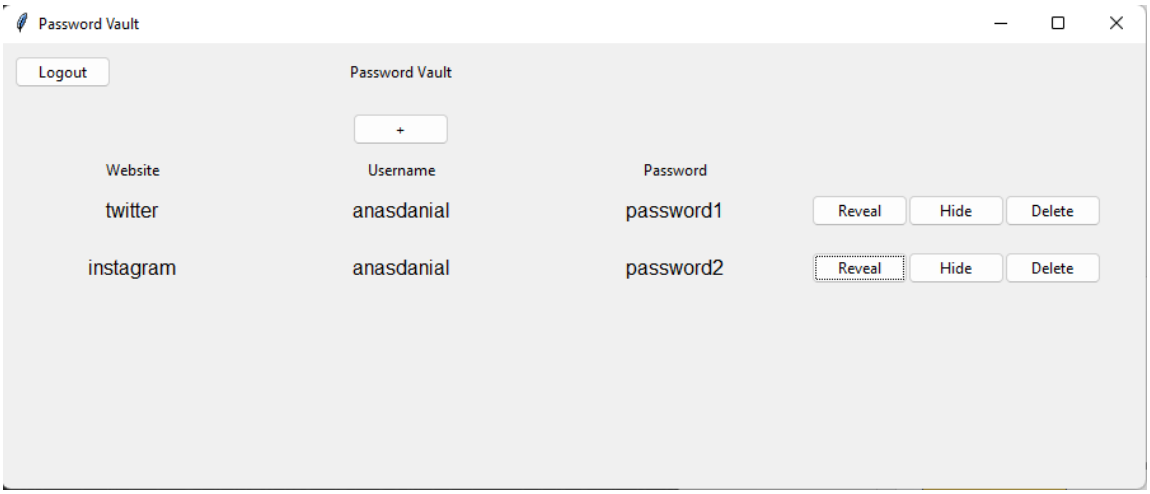

Figure 4 User Interfaces of the Real Password Manager Displaying Authentic Information

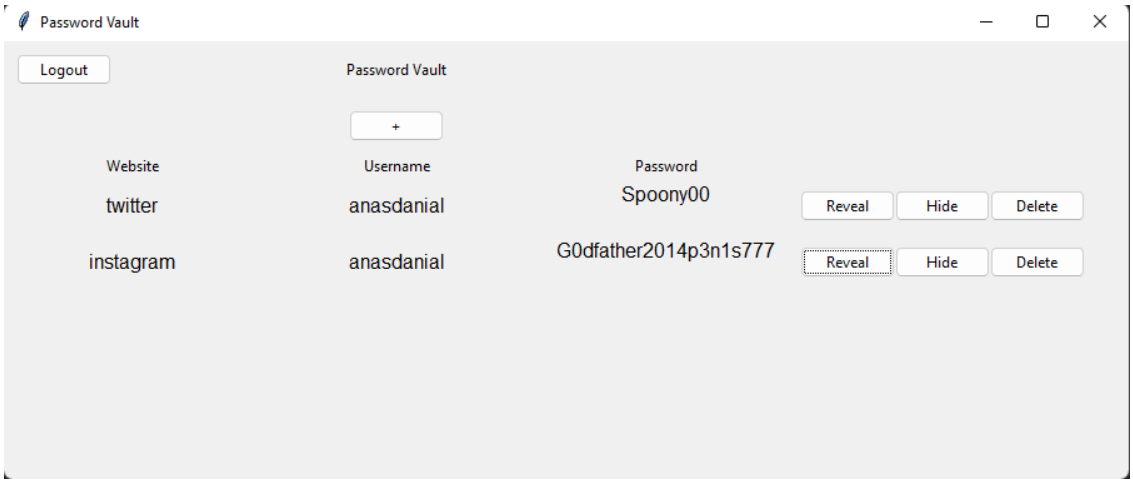

Figure 5 User Interfaces of the decoy Password Manager displaying fake/honeyword password

The Figures 4 and 5 represent the user interfaces of both the real password manager and the decoy password manager respectively, showcasing their visual similarity. This visual deception is a crucial aspect of our approach, as it aims to make it challenging for attackers to distinguish between the real and decoy interfaces. To assess the performance of our implementation, we measured the time taken to open the password vault in ten consecutive attempts. The results demonstrate that there is only a slight difference in response times between the real password manager and the decoy password manager. The real password manager consistently exhibits marginally faster response times compared to the decoy password manager. This indicates that legitimate users can access their password vault with minimal delay, while potential attackers attempting to access the decoy password manager experience only a slight delay due to the additional layers of security measures. We graphed the recorded times to provide a clear visualization of the performance difference between the real and decoy password managers on Figure 6. The graph shows that the response times of the real password manager consistently remain slightly lower than those of the decoy password manager. This confirms the successful diversion of unauthorized access attempts and reinforces the viability of our approach.

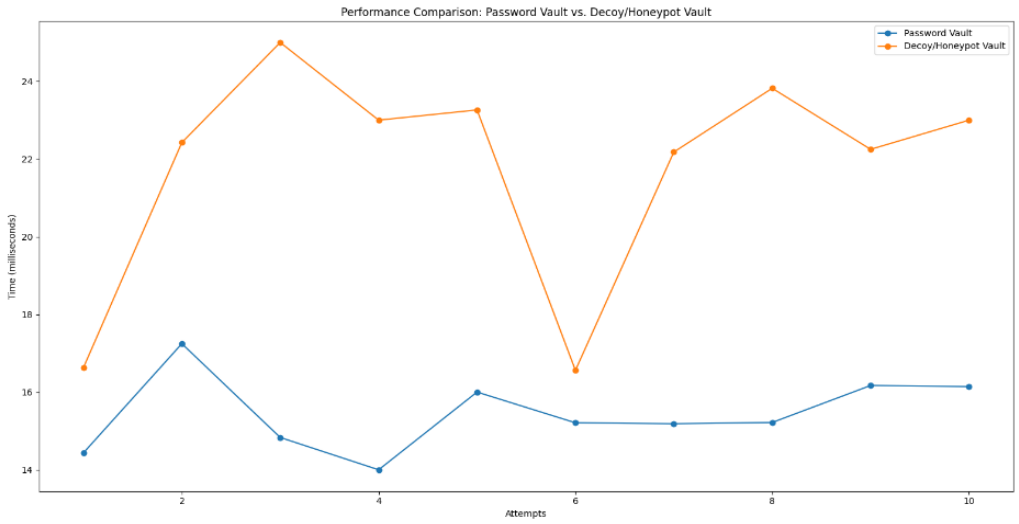

Figure 6 Comparison of Time Taken to Open Real Password Manager and Decoy Password Manager

Furthermore, we successfully captured crucial information during the authentication process, including the IP address, MAC address, attempted password, and website name. This captured information was utilized to generate an email alert system, which automatically notifies the legitimate user when an unauthorized access attempt is detected. The Figure 6 showcases the captured information, providing valuable insights for the user to take appropriate action and strengthen the security of their password manager account.

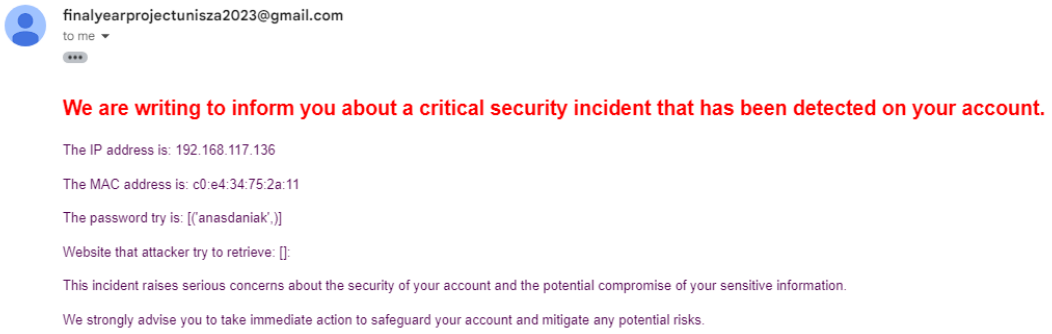

Figure 7 Email Alert Notification with Captured Information

Our high interaction honeypot implementation effectively deters unauthorized access attempts while minimizing inconvenience for legitimate users. The slight difference in response times between the real and decoy password managers demonstrates seamless integration and usability. By leveraging visual deception, performance differences, and email alerts, our approach enhances password manager security, making them more resilient to credential theft. This proactive measure contributes to advancing password manager security and provides a practical solution for real-world implementation. The success of our project reinforces the usability and effectiveness of high interaction honeypots in protecting user credentials and empowering users to respond to security threats. With further refinement, this approach can greatly enhance password management system security.

6. CONCLUSION

This paper has presented a comprehensive exploration of password manager security, focusing on the integration of honey encryption and high interaction honeypots. Honey encryption offers a robust method for protecting master passwords by leveraging encryption and randomization techniques. This approach enhances the security of password storage and retrieval, making it significantly harder for attackers to obtain sensitive information. Furthermore, the implementation of a high interaction honeypot adds an additional layer of defense, effectively deterring unauthorized access attempts and gathering valuable intelligence on potential attackers. The combination of honey encryption and high interaction honeypots has proven successful in our experiments, demonstrating their usability, effectiveness, and minimal impact on legitimate users. By seamlessly integrating these techniques into existing password manager systems, we enhance the overall security posture and resilience against credential theft. Our research contributes to the advancement of password manager security and provides a practical solution for safeguarding user credentials in real-world scenarios. As the threat landscape continues to evolve, the adoption of honey encryption and high interaction honeypots holds great promise in mitigating the risks associated with password management and enhancing the overall security of user accounts.

References

[1] Cheswick, W. R., Bellovin, S. M., & Rubin, A. D. (2003). Firewalls and Internet Security: Repelling the Wily Hacker. *Addison-Wesley Professional*.

[2] Florencio, D., & Herley, C. (2007). A large-scale study of web password habits. Proceedings of the 16th International Conference on World Wide Web, 657-666. DOI:10.1145/1242572.1242661

[3] Bonneau, J., Herley, C., van Oorschot, P. C., & Stajano, F. (2012). The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, 553-567. DOI:10.1109/SP.2012.44

[4] Luevanos, Carlos & Elizarraras, John & Hirschi, Khai & Yeh, Jyh-haw. (2017). Analysis on the Security and Use of Password Managers. 17-24. DOI:10.1109/PDCAT.2017.00013

[5] 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. DOI:10.5829/idosi.wasj.2012.19.04.1837

[6] Cheng, H., Li, W., Wang, P., Chu, C. H., & Liang, K. (2021). Incrementally Updateable Honey Password Vaults. In *USENIX Security Symposium* (pp. 857-874).

[7] Alkhwaja, I., Albugami, M., Alkhwaja, A., Alghamdi, M., Abahussain, H., Alfawaz, F., Almurayh, A., & Min-Allah, N. (2023). Password Cracking with Brute Force Algorithm and Dictionary Attack Using Parallel Programming. Applied Sciences, 13(10), 5979. https://doi.org/10.3390/app13105979

[8] Furfaro, A., Lupia, F., & Saccà, D. (2020). Gathering Malware Data through High-Interaction Honeypots. In *SEBD* (pp. 286-293).

[9] Juels, A., & Ristenpart, T. (2014). Honey encryption: Security beyond the brute-force bound. *Communications of the ACM*, 57(2), 95-103. DOI:10.1007/978-3-642-55220-5_17

[10] Carr, M., & Shahandashti, S. F. (2020). Revisiting security vulnerabilities in commercial password managers. In *ICT Systems Security and Privacy Protection: 35th IFIP TC 11 International Conference, SEC 2020, Maribor, Slovenia, September 21–23, 2020, Proceedings 35* (pp. 265-279).

[11] Juels, A., & Jakobsson, M. (1999). Honeywords: Making password-cracking detectable. *In Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security* (pp. 145-160).

[12] Abadi, M., Budiu, M., Erlingsson, Ú., & Ligatti, J. (2010). Control-flow integrity: Principles, implementations, and applications. *ACM Transactions on Information and System Security (TISSEC)*, 13(1), 4.

[13] Mell, P., & Grance, T. (2011). The NIST definition of cloud computing. *NIST Special Publication*, 800(145).

[14] Chiasson, S., Forget, A., Biddle, R., & van Oorschot, P. C. (2008). Influencing users towards better passwords: Persuasive cued click-points. Proceedings of the 2008 Symposium on Usable Privacy and Security, 1-12. DOI:10.1145/1531514.1531531

[15] Noorunnisa, Nahri & Siddiqui, Rahat. (2016). Review on Honey Encryption Technique. *International Journal of Science and Research (IJSR)*. 5. 1683-1686.

[16] Spitzner, L. (2002). Honeypots: Tracking Hackers. Addison-Wesley Professional, 480.

[17] Webster, A. (2018). An application of jeeves for honeypot sanitization.

[18] Bošnjak, L., Sreš, J., & Brumen, B. (2018). Brute-force and dictionary attack on hashed real-world passwords. *In 2018 41st international convention on information and communication technology, electronics and microelectronics (mipro)* (pp. 1161-1166).

[19] Han, W., Li, Z., Ni, M., Gu, G., & Xu, W. (2016). Shadow attacks based on password reuses: a quantitative empirical analysis. *IEEE Transactions on Dependable and Secure Computing*, 15(2), 309-320.

[20] Najafabadi, M. M., Khoshgoftaar, T. M., Kemp, C., Seliya, N., & Zuech, R. (2014). Machine learning for detecting brute force attacks at the network level. *In 2014 IEEE International Conference on Bioinformatics and Bioengineering* (pp. 379-385). IEEE.