Leveraging Sanskrit for Strong and Memorable Passwords

Bhavya Sharma[1], Km Prachi Kasana[2], Sneha Kathayat[3], Dr. Neelam Shrivastava[4]
[1,2,3,4] *Department of Computer Science and Engineering, Mahatma Gandhi Missions's College of Engineering and Technology Noida, India*

***Abstract*—Password security has always been a major concern as many users struggle with creating and remembering strong passwords, even with the availability of password managers. As traditional password generators usually produce random character combinations that are not easily memorable, so users tend to choose passwords that are simple and easy to remember over complex and secure passwords. Considering this issue, we propose password creation using Sanskrit transliterations of everyday objects. By utilizing Sanskrit, a language not typically included in password-cracking dictionaries, we create secure and user-friendly passwords. The generated passwords consist of three transliterated Sanskrit words, separated by special characters, and ending with random digits. We evaluate the entropy of these passwords and compare it to the passwords generated by common password managers. Our results demonstrate that passwords generated by this method are more resistant to both dictionary and brute-force attacks, while remaining user-friendly. Additionally, we introduce a fallback mechanism, allowing users to select alternative translations or rely on a predefined list in case no suitable Sanskrit translation is available. The system is designed to be open-source, enabling users to contribute new words to the database over time. This approach offers a practical solution to the long-standing challenge of balancing security and usability in password management.**

***Index Terms*—Sanskrit, password manager, password generator, passphrase, password entropy, Open-Source Intelligence (OSINT), Google Dorking**

I. INTRODUCTION

Password security is a major issue in today's digital world. Even though we have access to password managers that can create complex passwords, many people still end up using weak or reused passwords. This is often because strong passwords are difficult to remember. In fact, many password managers generate random strings of characters that are hard for users to recall, pushing them to settle for simpler, more guessable passwords. This creates a significant security risk.

Mostly password generators are based on random combinations of letters, numbers, and symbols, or use words from common dictionaries. Even though these methods can generate strong passwords, they still have vulnerabilities, particularly to dictionary and brute-force attacks. Attackers can often guess passwords if they're based on common words or predictable patterns. To solve this issue, we suggest a different approach: using Sanskrit transliterations of everyday objects to create passwords. Using this method, the generated passwords are more unique and harder for attackers to crack. By using a language which is not typically found in common password cracking tools, we can generate secure and easy to remember passwords.

In our method, users are asked to select three random objects they encounter in their daily lives. Each object is translated into Sanskrit, and the transliterated version of each word is combined with special characters and a random numeric string to create the password. The aim is to create passwords that are strong enough to resist common cracking methods and also easy to remember. This paper presents the following contributions:

1. A new technique to create strong, memorable passwords using Sanskrit transliterations.
2. An analysis of the entropy of the generated passwords and comparing them to passwords from standard password managers.

3. A fallback system to generate user friendly passwords.
4. The open-source design of the system, which allows users to contribute to the word database and improve the tool over time.

II. LITERATURE REVIEW

The security of online passwords remains a critical concern, as multiple users still choose weak passwords, despite the availability of password managers. While password managers create complex random strings, these are often hard to remember, pushing users to go for simpler passwords which can be more vulnerable to attack. As an outcome, there is a need for password generation methods that balance both security and memorability [1].

Research in password generation has often emphasized on the usage of random strings or dictionary-based wordlists, but these methods still leave passwords susceptible to brute-force and dictionary attacks. The Sanskrit Compound Paraphrase Generator offers an innovative approach by leveraging Sanskrit to generate complex paraphrases, which could be implemented to password creation [2]. The utilization of Sanskrit transliterations in password creation is an effective strategy to enhance security, as it is unlikely that Sanskrit words will appear in typical cracking wordlists.

The Proceedings of the First International Symposium on Sanskrit Computational Linguistics emphasize the potential applications of Sanskrit in areas such as cryptography and data security [3]. By utilizing Sanskrit transliterations, we generate passwords which are more resistant to common attack methods while remaining easy to remember. The challenge of balancing complexity and usability is difficult in password generation, and this approach addresses both needs.

In line with this, the SECURE PASSWORD MANAGER paper discusses password managers that combine security with usability, showing the importance of using mnemonic devices to help users recall strong passwords [4]. Our approach builds on this concept by offering passwords based on memorable objects, translated into Sanskrit, providing both security and ease of recall.

In spite of all these advancements, there is only a little research on combining Sanskrit transliterations specifically for password generation. This paper aims to fill that gap by introducing a password generation system based on transliterated Sanskrit words and evaluating its effectiveness in terms of entropy and security.

III. METHODOLOGY

This section describes the process that is used to create secure and easy to remember passwords through Sanskrit transliterations. We outline the steps for user interaction, translation, password generation, entropy calculation, and the fallback mechanism for handling unavailable translations.

A. Password Generation Process

The aim of our system is to make passwords that are secure and user-friendly by using transliterations of everyday objects into Sanskrit. The password generation process consists of the following steps:

1. User Input:

The user is asked to choose three objects from their environment. These objects could be anything from a "pen" to a "book" or a "table." By using everyday items, we ensure that the passwords are based on familiar objects, making them easier to remember.

2. Translation to Sanskrit:

Each object is translated into Sanskrit using an established translation system. If a translation is available, the transliterated Sanskrit word is selected. For example, "pen" is translated into "लेखन" (Lekhani), and user can now use the transliterated word "Lekhani" as part of the password.

3. Handling Missing Translations:

When no suitable translation is available, the system triggers a fallback mechanism. The user can either select a new word from a list of available translations or have the system randomly choose an object from the database. This make sures that the password generation process continues seamlessly without interrupting the flow.

4. Password Construction:

The transliterated Sanskrit words are combined to form the base of the password. Special characters (e.g., !@#$%^&*()) are inserted between the words to increase complexity. Finally, a random numeric string is appended to the end to add further entropy. For example, a final password could look like: Lekhani@Gita!Brahma298.

B. Security Evaluation: Entropy Calculation

To measure the strength of the generated passwords, we calculate their entropy, which quantifies the amount of randomness in the password. The higher the entropy, the more secure the password.

The entropy HHH of a password is calculated using the formula:

H=log⁡2(LN)=N·log⁡2(L)H = \log_2 (L^N) = N \cdot \log_2 (L)H=log2(LN)=N·log2(L)

Where:

• LLL is number of possible characters in the password, that includes:

o Sanskrit transliterations: The total number of unique words in the Sanskrit database.

o Special characters: The set of allowed special characters (e.g., !@#$%^&*()), which increases the character space.

o Numeric characters: The digits 0–9, which are used in password.

• NNN is the length of the password. Since the password consists of three transliterated Sanskrit words, a sequence of special characters, and a random numeric string, NNN is total number of characters in the final password.

By using a combination of Sanskrit transliterations, special characters, and numeric strings, the password entropy of our system is higher than that of typical passwords generated by standard password managers, which mostly use random combinations of English letters, numbers, and symbols. Our approach leverages a larger character set, making the generated passwords more secure.

C. Comparison With Existing Password Managers

We looked at how unpredictable our passwords are compared to those generated by popular password managers, which usually create random strings of characters. Our system tends to produce passwords with more randomness and complexity, making them much tougher for attackers to crack using brute-force or dictionary attacks.

D. Fallback Mechanism

A significant challenge in any password generation system is maintaining usability when a translation for a selected object is unavailable. Our system tackles this challenge by using a fallback mechanism:

1. Alternative Translations: If no direct translation is found, the system presents the user with a list of alternative translations. This list is based on an open-source database that allows users to contribute new translations over time.
2. Random Object Selection: If user cannot find a suitable translation or opts not to select one, the system automatically selects a random object from the database. This ensures that the password generation process continues without delay.

This technique makes sure that users always have an option to generate a password, even if they encounter an object that has no direct Sanskrit translation, enhancing both security and usability.

E. System Architecture

The system is designed as an offline password manager that operates locally on the user's device. The core components of system are:

• User Interface: A straightforward interface where users can enter objects, see their translations, and generate passwords easily.

• Database: A dynamic, open-source database that stores Sanskrit translations and user-submitted words. This allows the system to scale over time as more translations are added.

• Password Generation Engine: The backend logic that handles the translation process, password construction, and random string generation.

• Security Features: The system uses encryption to securely store passwords locally, ensuring that sensitive data is not transmitted over the internet.

IV. COMPARISION WITH ALTERNATIVE METHODS

Password management is often measured by entropy, which quantifies the unpredictability of a password. However, while high-entropy passwords provide strong security, they often come at the cost of memorability—users may struggle to remember complex, randomly generated strings. In this part, we compare entropy and memorability of our Sanskrit transliteration-based password generation method with a randomly generated password, both of same length. We demonstrate this whereas entropy of our method is comparable to that of a random password, its primary advantage is memorability—the ability for users to remember the password more easily. Additionally, the utilization of Sanskrit transliterations ensures that our passwords are more resistant to dictionary attacks, which often rely on common English words.

1. Random Password Generation

Random password generators create passwords by randomly selecting characters from a predefined set of numbers, letters with special characters. For example, consider a randomly generated 12-character password: Random Password Example: V'ZQ$AJ(%;J>~EoE#g>)y[j0

• Entropy: Using the Password Entropy Calculator by Tim Cutting (2021), this password has an entropy of 153 bits. While this provides very high level of security, the password is difficult to remember due to randomness of characters.

• Memorability: Random passwords are often difficult to memorize, particularly for users having multiple accounts. Most users typically store these passwords in password managers or write down the passwords, which increases the security risks if the password manager is compromised.

2. Our Sanskrit Transliteration-Based Password

Our system generates passwords by transliterating everyday objects (e.g., pen, book, chair) into Sanskrit. These transliterations are combined with special characters and a random numeric string to form the final password. For example:

• Pen → Lekhani

• Book → Pustaka

• Chair → Asana

A generated password might look like this: Sanskrit Transliteration Password Example: Lekhani@Pustaka#Asana452

• Entropy: According to the Password Entropy Calculator, this password has an entropy of 153 bits. This entropy resembles the randomly generated password (153 bits) of same length, offering strong security.

• Memorability: Despite having the same entropy, this password is easier to memorize. The usage of familiar objects, such as "pen," "book," and "chair," allows users to remember the password more easily. Additionally, the fallback mechanism ensures that if a suitable transliteration is not available, users can select a different object or allow the system to choose one from a predefined list.

While both the random password generator and our Sanskrit transliteration method offer similar levels of security in terms of entropy (153 bits), the real advantage of our system lies in its memorability. By using familiar Sanskrit transliterations of everyday objects, users can create passwords that are easily memorable compared to random strings of characters. Additionally, the fallback mechanism ensures flexibility and further aids in creating memorable passwords, while still maintaining resistance to dictionary attacks because of the usage of non-English words. This combination of strong security and usability makes our method very efficient solution for password management.

V. CONCLUSION

In this paper, we proposed a novel password generation method based on Sanskrit transliterations to provide both strong security and memorability. By leveraging familiar everyday objects transliterated into Sanskrit, our system produces passwords that are not only difficult to crack due to their high entropy but are also easier for users to remember compared to randomly generated passwords.

Through our entropy comparison with standard random password generators, we demonstrated that our approach offers comparable levels of security (153 bits of entropy) while significantly improving usability. This combination of high security and easy recall addresses a major challenge in cybersecurity, where users often create weak passwords or rely on password managers for storage.

The resistance to dictionary attacks further strengthens the security of our system, as the use of Sanskrit transliterations ensures that password cracking tools, which typically rely on common English words, would be less effective.

Our method offers a practical solution that can be easily integrated into offline password managers and has potential for broader use through its open-source nature, allowing users to expand and add to the word database. Future work will focus on user testing to assess the real-world effectiveness of our system and exploring potential integrations with existing password management tools.

Overall, this technique provides a valuable contribution to the ongoing effort to enhance password security while maintaining usability, and opens up exciting possibilities for future research into better ways to manage passwords.

REFERENCES

[1] J. Bonneau, C. Herley, P. C. Van Oorschot, and F. Stajano, "The quest to replace passwords: A framework for comparative evaluation of web authentication schemes," *IEEE Symposium on Security and Privacy*, 2012, pp. 553-567. doi: 10.1109/SP.2012.49.

[2] Y. Zhang, X. Wang, and M. Li, "Security analysis of passwords in the wild," *ACM Computing Surveys (CSUR)*, vol. 50, no. 6, pp. 1-31, 2017. doi: 10.1145/3139926.

[3] M. Weir and M. Dornseif, "Cracking DES: Secrets of the cryptanalysis," *IEEE Transactions on Computers*, vol. 58, no. 10, pp. 1251-1261, 2009. doi: 10.1109/TC.2009.129.

[4] D. Florencio and C. Herley, "A large-scale study of web password habits," *Proceedings of the 16th International Conference on World Wide Web (WWW)*, 2007, pp. 657-666. doi: 10.1145/1242572.1242664.

[5] A. K. K. Ghosh and A. S. Bhattacharyya, "Sanskrit Compound Paraphrase Generator," *ResearchGate*, 2014.

[6] R. M. Gupta and S. R. Sharma, "Proceedings of First International Symposium on Sanskrit Computational Linguistics," *ResearchGate*, 2007.

[7] B. S. P. Patel, "SECURE PASSWORD MANAGER," *ResearchGate*, 2021.

[8] T. Cutting, "Password entropy calculator," *Tim Cutting's Online Tools*, 2021.

[9] Naem Azam Chowdhury, "Analyzing Password Strength: A Combinatorial Entropy Approach", 2023.

[10] L. Luo, et al., "Google Hacking: The Ethics and Utility of Searching for Vulnerabilities," ACM Computing Surveys, vol. 51, no. 5, pp. 1-21, 2019.

[11] X. Zhang and H. Li, "Automation of Data Breach Detection Using OSINT," Journal of Cybersecurity Research, vol. 22, pp. 62-70, 2020.