“It is just a phrase, it will pass”: A novel approach to generating memorable passphrases

Shreya Arun Naik
s4an@uwaterloo.ca
David R. Cheriton School of Computer Science
University of Waterloo

Amandeep Kaur
a3kaur@uwaterloo.ca
David R. Cheriton School of Computer Science
University of Waterloo

1 Abstract

Analyzing password strength has been the focus of recent research considering how passwords continue to be the dominant method for authentication. Strong passwords like random strings are challenging to remember, whereas memorable passwords are simple to crack. Although Password Managers (PMs) are an attractive alternative, they have alarming security issues and usability concerns of their own. In this work, we propose a novel approach to generate memorable passphrases for multilingual users. Our idea assumes that the user is at least bilingual, or the user has some means of querying romanizations of non-English words. We analyze the non-randomness and guessability of passphrases made up of romanized words using state-of-the-art guessability tools. Based on our analysis, we offer carefully constructed recommendations for users to generate passphrases. We then conduct a user study to determine how usable people find passphrases created using these recommendations and gauge their perception of security of this approach. The results of this work will help in evaluating the security and usability of our recommendations and can also be used to improve existing proactive password checkers, strength estimators and password-cracking tools for security auditing.

2 Introduction

Passwords, one of the oldest security tools of the internet, do not provide enough protection in this day that sees rising cybercrime and sophisticated cyber attacks. Attacks on passwords continue to remain in the top most sought-after attack vectors to compromise security of a system/application to steal information/credentials.

Attacks on passwords can be broadly classified into two types: online attacks and offline attacks. An online password attack is carried out at the login interface of a system/application, with the attacker trying a large number of username-password pairs in hopes of guessing the victim’s credentials. Online attacks are limited by the speed of the underlying network and more importantly, by the system/application’s limitations on the number of wrong attempts. Offline attacks, on the other hand, utilize a leaked hash file of the victim’s passwords to launch hash compute-then-compare attack.

Password guessing and cracking attacks are only getting more refined as technology advances. However, cognitive ability of users to remember passwords more or less remains the same. If password-based authentication has to stay secure, we need to make sure users of the system are aware of how to create strong passwords. To this end, we propose that multilingual users, while creating passphrases, which have been proven to be a better alternative to traditional passwords, use romanized words from their native non-English languages. In a recent study, Joshua Tan et al. [11] evaluated that any password that takes less than 10^6 is not safe against online attacks. The same study proposed that 10^{14} is the minimum threshold required to withstand an offline attack. We therefore use 10^{14} as our benchmark to determine if minimum guess numbers for our generated passphrases provide enough security against both online and offline attacks.

3 Methodology

The study was conducted in three phases. In the **Passphrase generation** phase, we generate passphrases for analysis using our own passphrase generator **RomIT**. In the **Passphrase Analysis** phase, we evaluate the quality of passphrases generated by RomIT using zxcvbn, Hashcat, and Shannon’s Entropy. Lastly, we run a **User study** to guage our participants’ perceptions of using our passphrase-creation recommendations. This pilot study was conducted from November 28th - December 2nd, 2022 with 33 participants from South Asian ethnicity.

3.1 Phase 1: Passphrase generation

In this phase, we used *Dakshina Dataset* by Brian et al.[7] to generate passphrases for analysis using our own passphrase generator RomIT. Dakshina dataset, compiled in March 2019, contains text from 12 South Asian languages written in both Latin and their native scripts. For each language, the dataset includes a large portion of native script Wikipedia text, a romanization dictionary of native script words with attested romanizations, and some full sentence parallel data in both the language’s native script and standard Latin alphabet. Four of the twelve languages are Dravidian, and the remaining eight are Indo-Aryan (kn, ml, ta, and te). While the other languages have text written in Brahmic scripts, two of the languages — sd and ur — have text written in Perso-Arabic scripts.

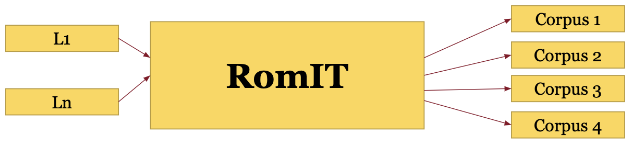

Figure 1. RomIT - passphrase generator

As seen in Figure 1, our password generator, RomIT can take as input, *n* number of languages and generate 4 corpora as discussed below.

- **Corpus 1**: English passphrases. This corpus will serve as our baseline. We create this dataset using the NLTK corpus wordlist. An example could be the passpharse: *horseoceanlifeteeth*. This is made up of 4 different words - horse, ocean, life, teeth.
- **Corpus 2**: Transliterated passphrases from non-English languages without separators. This corpus consists of transliterated phrases from non-English languages, without any separators. An example could be - *ghodasamundarzindagidaanth* , which is a romanized version of words in Hindi language.
- **Corpus 3**: Transliterated passphrases from non-English languages with separators. This corpus consists of transliterated phrases from non-English languages, with separators. An example could be - *ghoda@samundarzindagi!daanth* , which is a romanized version of the language Hindi, with separators.
- **Corpus 4**: Transliterated passphrases with l33t. This corpus consists of transliterated phrases, with some replacements through l33t (leet). An example could be - *gh0d@5@mund@rzind@gidaanth*

3.2 Phase 2: Passphrase Analysis

To evaluate the quality of passphrases generated by RomIT, we employ three methods pertaining to measuring randomness, guessability score and resilience to password-cracking attacks. Each of these methods and their results are discussed here in detail.

3.2.1 Estimation of randomness. U.S. Defense Department's Password Management Guideline which is popularly known as The Green Book [6] suggested evaluating password strength in terms of character space, modeled as X = A^M where, X is the maximum number of guesses needed to crack the password, A is the size of the character space and M is the length of the password. X is more commonly expressed in bits as M . $\log_2$ A, which is simply Shannon's entropy X, or H(X) of a distribution [8]. Entropy in bits provides a reasonable estimate of randomness of a password and can therefore be used to compare how random distributions of passphrases from RomIT are from their English-only counterparts. We

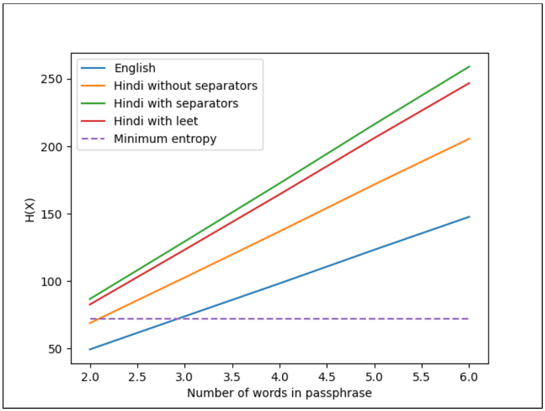

Figure 2. Comparison of Shannon's entropy of RomIT's passphrase corpora. X-axis represents the number of words making up the passphrase and Y-axis represents Shannon's entropy, H(X).

calculate Shannon's entropy of each passphrase in every corpus generated by RomIT and the results show that entropy of non-English passphrases with separators or l33t is more than the entropy of English passphrases as depicted in 2. The reason being separators and l33t replacements increase character space. Furthermore, including separators also increases length unlike the inclusion of l33t as l33t performs replacements. Consequently, RomIT's corpus of passphrases with separators have a greater average entropy than the corpus of passphrases with l33t. We consider this to be a naive metric as romanized non-English words use the same character space of [a-z] [A-Z] as English words. Length of words therefore becomes a feature with more weight. Any non-English language with word lengths typically less than English words will accordingly be penalised by Shannon's entropy if no replacements or separators are used.

3.2.2 Guess Numbers. We use guess numbers, which are an indication of the resistance of a passphrase to password guessing [5], to estimate the strength of passphrases. As such, short password and passphrase resistance to guessing was used to measure strength. Guess number allows for password policy and inter-cultural comparison by computing the percentage of passwords that can be cracked by an algorithm and computing the percentage of passwords that can be cracked at a given number of guesses.

We use *zxcvbn* [12] to evaluate how guessable RomIT's passphrases are. zxcvbn is a password strength estimator developed by Daniel Lowe Wheeler that uses heuristics to guess passwords, unlike Probabilistic Context-Free Grammar (PCFG) and Markov Chains that use probability [5]. It bases its password estimates on three stages: match, score and research. Match enumerates all the patterns in the passphrases it can detect by matching against several English dictionaries,

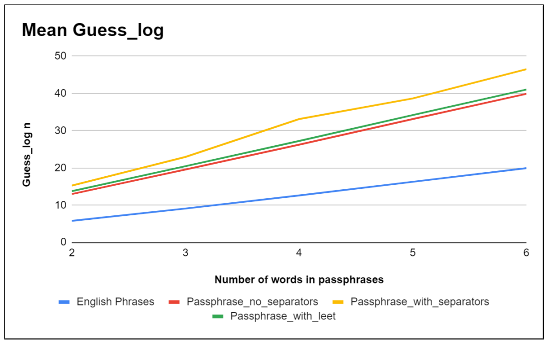

Figure 3. Comparison of zxcvbn's guess numbers of RomIT's passphrase corpora. X-axis represents the number of words making up the passphrase and Y-axis represents the average guess log.

spatial keyboard patterns, repetition of characters, l33t, uppercases, commonly used sequences (123, 987), years, dates (3-13-1997, 13.3.1997, 1331997). In the score stage, zxcvbn calculates entropy of each pattern detected in the match stage, independent of the rest of the passphrase. In the search stage, passphrases are broken down into the simplest non-overlapping patterns that constitute the passphrase. This helps infer the lowest bound of an attacker's attempts at cracking this password

zxcvbn then proceeds to calculate entropy of the passphrase by summing entropies of individual patterns. Although this results in underestimations, the author argues that these are acceptable underestimations as they insinuate greater security. These calculations of repeat, spatial, dictionary entropies are transformed to return guess numbers. Greater the guess number the better.

Figure 3 depicts comparison of guess numbers for RomIT's passphrase corpora with English passphrases. RomIT's passphrases perform better even for passphrases with as few as two and three individual words. Therefore, we infer that passphrases that constitute non-English words fare better than only-English phrases regardless of the length. Here, we duly note that these high guess numbers for non-English passphrases can be attributed to zxcvbn's limited dictionaries. Evaluating what scores this password strength estimator outputs for non-English passphrases by also considering dictionaries from other languages can make for an interesting future research question.

Furthermore, zxcvbn fails to correctly split non-English passphrases into individual segments in both the match and search stages possibly owing to its English-only dictionary. As an example, let's consider a passphrase from C2 (non-English passphrase with separators),

ghoda@samundar_zindagi#daanth

zxcvbn returns the following match sequence,

1. 'ghoda@samundar_' : pattern: bruteforce, guesses_log10: 15
2. 'zindagi:' pattern: dictionary, guesses_log10: 4.40605, dictionary_name: english_Wikipedia, rank: 25471, ...
3. '#daanth:' pattern: bruteforce, guesses_log10: 7

It incorrectly identifies *ghoda* and *samundar* to be one single sequence when they actually are two separate Hindi words. However, it rightly splits *zindagi* which is a comparatively common Hindi word used fairly often in English literature. This further bolsters our inference that zxcvbn's accuracy is hampered by its very limited set of dictionaries.

It is also pertinent to mention that zxcvbn gave RomIT's non-English passphrases an average strength score of 4 (out of 4) and estimated guess times for all, throttled online attacks, unthrottled online attacks, offline attacks with a fast hashing algorithm (like MD5) and offline attacks with a slow hashing algorithm (like bcrypt) to be *centuries*. As a result, we do not launch brute force attacks using Hashcat on RomIT-generated passphrases in the next phase of analysis.

3.2.3 Password cracking attack using Hashcat.

Hashcat is a free, open-source password recovery utility that is deemed to be the world's fastest and most advanced [10]. It facilitates distributed cracking by supporting CPUs, GPUs and other hardware accelerators on multiple platforms. Hashcat provides five unique modes of attacks to choose from and supports over 300 hashing algorithms [10]. As mentioned earlier, it is infeasible to launch a brute-force attack on RomIT's corpora owing to already exhibited high guess scores and time (in centuries). In this section, we therefore limit ourselves to dictionary attacks and combinator attacks on hashed passphrases. We used two devices with NVIDIA's latest Tesla T4 GPU to crack passphrases first, hashed with salted MD5, then with bcrypt over SHA512. We used the RockYou dataset [1] of leaked passwords and an English dictionary as wordlists. Additionally, we also used two password mangling rulesets provided by Julian Dunni and Anthony Weems, the first of which is the Hob064 set [3] that contains 64 of the most frequently used password patterns used to crack passwords, and the d3adhob0 set [3] which is much more extensive.

When tested with only 10 hashed passphrases from RomIT (both English and non-English), Hashcat failed to crack any. We attribute this failure to the usage of an only English dictionary and to non-existence of longer romanized passphrases (more than 4 words) in the RockYou dataset. Additionally, Hashcat allows a maximum of two wordlists to combine words from in the combinator attack, further limiting Hashcat's capability of cracking passphrases made up of more than two words, unless the attacker creates two new wordlists by concatenating singular words from the original dictionary. Creating new wordlists along these lines to further test Hashcat's efficacy at cracking passphrases greater than a certain length can be tried in the near future.

Taking into consideration the results from our analysis, we infer that passphrases made up of romanized words from non-English languages are comparatively more secure against password guessing/cracking attacks than English passphrases.

3.3 Phase 3: User study

Research Questions: Our user study aims to determine how usable people find passphrases created using proposed recommendations and gauge their perception of the security of this approach. We ask four main Research Questions (RQs):

- **RQ1**: Do users romanize the same word differently?
- **RQ2**: Do users find passphrases created based on our recommendations usable?
- **RQ3**: Do users find generated passphrases more secure than their previous passwords?
- **RQ4**: Do users prefer using our recommendations to create passphrases over their passwords in the future?

In RQ1, we tried to understand if our users romanized the same word differently. An example could be the Hindi transliteration of the word "balloon" which can be spelled as "Gubara", "Gubaara" or "Gubaaraa". In RQ2, we wanted to find the usability aspect of our passphrase recommendations. From the user's point of view, whether the passphrase generated using our proposed approach is memorable and usable at the same time. The RQ3 deals with the security aspect of passphrases. Through this question, we wanted to explore the participant's views on the security of using recommended passphrases as compared to their previous passwords. The last research question RQ4 deals with answering if users would prefer the proposed passphrase generation recommendations to create passphrases over their previously used methods.

3.3.1 Survey Structure.

Our survey was administered online through two google forms which lasted for around 5 minutes each. We conducted the user study from November 28th - December 2nd, 2022. The whole user study was initiated in 3 phases. Our survey structure is described below.

- **Pre-study survey**: The first phase consisted of participants filling up a pre-study survey form. Through this form, we intended to extract linguistic features of our participants. The survey included questions like the number and names of the languages they speak. In this survey form, we also collected some general password generation techniques that were followed by them. These included questions related to our proposed models, such as if they use non-English words in their passwords and similar questions. The survey form can be accessed here.[1]
- **Authentication Task**: For the second phase of our study, we set up a dummy signup/login webpage with minimal account registration and login functionalities like credential input and an option to change passwords if forgotten. We requested participants of our study to visit this login page at least once every day for a week to authenticate themselves here.[2]
  We duly note that our database associated with this login website did not store any passwords. Authentication tasks were implemented using Google Firebase's APIs that use brcypt to hash passwords. We used Heroku to host our website.
- **Post-study survey**: The third phase consisted of participants filling up a post-study survey form. Through this form, we intended to explore the usability and security aspect of our proposed approach. The usability aspect included questions like the number of times a participant forgot their passphrase, their most common method to remember passphrase. We also included questions that evaluated the security of our methodology, with questions like if they think the generated passphrase is more secure than their previous passwords, what is the rationale behind it and so forth. We also garnered feedback to further improve our study. [3]

3.3.2 Recruitment and Demographics .

Our survey was administered at the University of Waterloo, a public university in Canada. The authors distributed an official mail with study details to a sample of their friends, consisting of South Asian (Indian and Pakistani) ethnicity. The study was conducted from November 28th - December 2nd, 2022. The study was attempted by a total of 33 participants out of a pool of 53, providing a response rate of 62.2%. The email subject line clearly stated that it was an invitation to participate in an "Empirical Usability Study Of Transliterated Passphrases' conducted by students of the David R. Cheriton School of Computer Science, Faculty of Mathematics, University of Waterloo. The sample consisted of mainly younger (mean age was 23.4 years), male-identifying (63.6% males, and 36.4% females), with 81.1% participants already having an undergraduate degree. The participant pool consisted of technically savvy individuals.

4 Inferences from User Study

4.1 RQ1: Difference in Romanization

In response to the RQ1, do users romanize the same word differently, we asked users to type romanizations of certain Hindi words in English. The English words chosen for transliteration were "balloon", "with", "sun", "eyes" and "happy". The results for words "balloon" and "sun" are shown in Figure 4. As can be seen, we received 12 variations of the transliterated word "balloon". This study helped us infer how

[1]https://forms.gle/bjqeDFhk9U8gfW9MA

[2]https://romit-2.herokuapp.com/

[3]https://forms.gle/hcotDxJdbpcYb1mU8

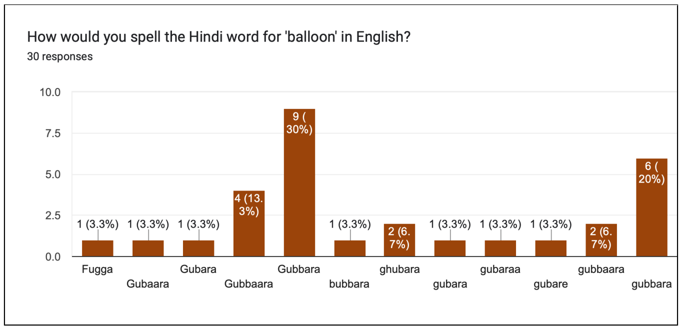

Figure 4. Romanized variations of the word "Balloon". X-axis denotes the received responses, with the Y axis denoting the frequency

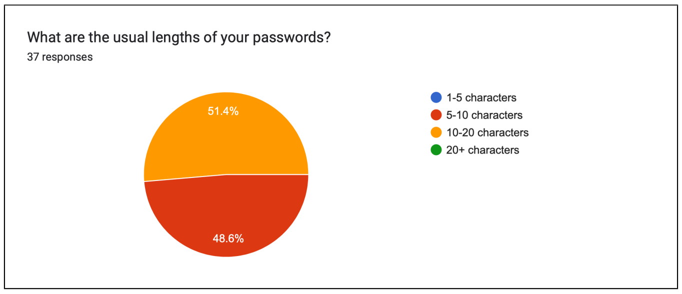

Figure 5. Graph depicting the responses for the usual length of passwords used by the participants of the study

detrimental a breach containing passphrases generated using our recommendations could be to the general security of password authentication. For cracking even a common word like “balloon” the attacker needs to think and try different variations of the spelling, which would lead to an increase in time and effort, thus making our passphrase even hard to guess/crack.

We also enquired about the participants’ general password-generation strategies. We were mainly concerned with finding answers to 3 aspects. The first question was related to the usual length of their passwords. From our study results in Figure 5, it was found that 51.4% of participants used 10-20 characters for their passwords. The second question was related to the re-use of passwords over multiple websites. It is surprising to see the results in Figure 6. wherein 83.8% of users mentioned that they re-use their passwords. This is a concerning result because if such a password is leaked, the user’s multiple websites could be compromised. The last question we asked attempted to explore if the user reset their passwords frequently so as to improve security and prevent password compromisation. An alarming 78.4% of participants denied doing as seen in Figure 7. This would be even more detrimental if the users re-use their passwords and do not re-set them frequently.

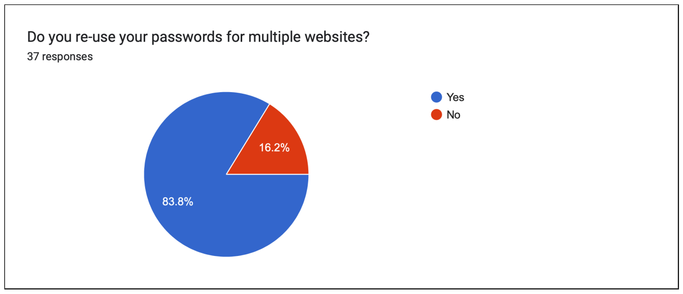

Figure 6. Graph depicting the responses for the question Re-usage of password across multiple websites

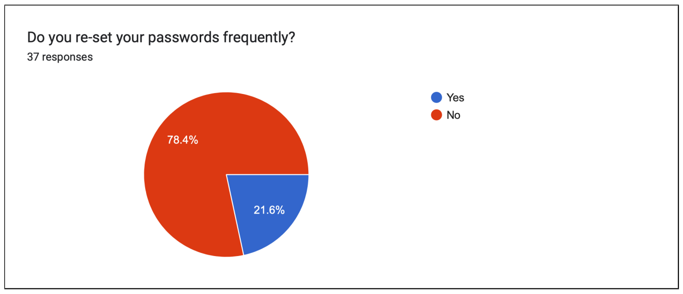

Figure 7. Graph depicting the responses for the question Re-setting of password across multiple websites

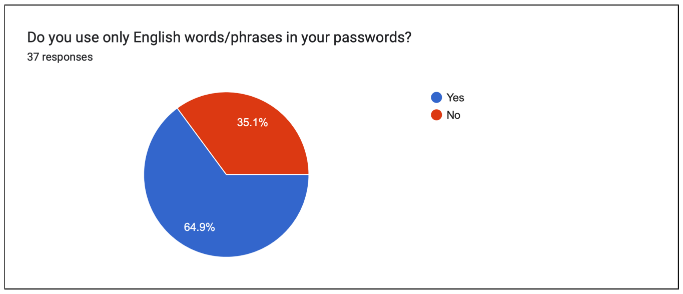

Figure 8. Graph depicting the responses for the usage of English words in passwords by the users

Next, we wanted to understand their password composition strategies. Our idea assumes that the user is at least bilingual, or the user has some means of querying romanizations of non-English words. Approximately 70% of users stated to use only English words/phrases in their passwords, Figure 8 . This result shows that the majority of users’ passwords could be compromised by an English dictionary attack. We asked a follow-up question related to the change of a password. Around 46% of the users started to make some modifications to their previous passwords. However, 54% of users generated a new password every time they were asked to change it as seen from Figure 9.

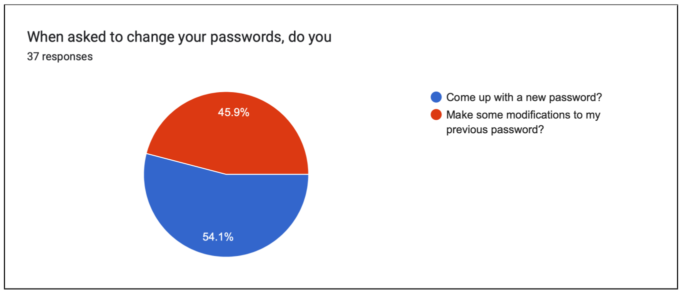

Figure 9. Graph depicting the responses for the question related to changing the password

4.2 RQ2: Usability

In this section, we answer RQ2, which explores the usability aspect of our proposed methodology. We begin by asking the count of words used by users for making passphrases. Almost 51% of the users used 4-5 words in their passphrase. The previous study [13] states that using 5 words to form a passphrase improves security. From a usability viewpoint, Approximately 63.3% of participants found remembering phrases from other languages easy, with only 15% finding it difficult. 21% responded to be neutral.

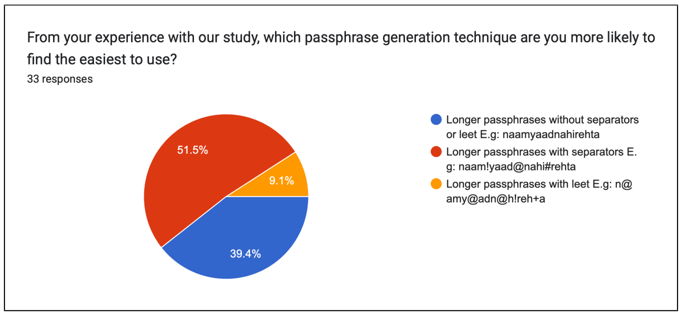

Figure 10. Graph depicting the responses for the question related to preference of passphrase generation technique used by the participants

We generated three passphrase generation techniques. The first technique used longer passphrases without separators or l33t E.g: naamyaadnahirehta. The second technique proposed was to use longer passphrases with separators. E.g: naam!yaad@nahi-rehta. The last technique used longer pass with l33t E.g:n@amy@adn@h!reh+a. In terms of usability, the users were asked about their preference for the passphrase generation technique. Almost 51% of the users found the variation 'Longer passphrase with separators' easiest to use. The least preferred was 'Long passphrase with l33t'. This could be attributed to the fact that l33t adds to the complexity of remembering, hence the results, as seen in Figure 10. The participants were also asked if the length of the passwords made it hard in terms of using and remembering the passphrases. Approximately 40% users agreed to the statement while 27% disagreed, as shown in Figure 11.

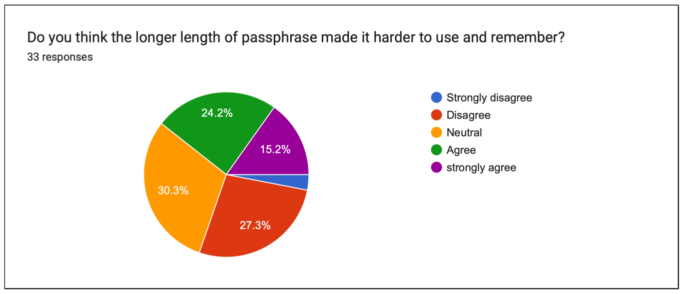

Figure 11. Comparison of the length of the passwords with the usability aspect

Only 9.1% users had to reset their passphrases in the study. Almost 51% of users never forgot their passphrase. This is an encouraging result from a usability point of view.

4.3 RQ3: Security

We now discuss RQ3: 84.8% of the participants found the generated passphrases more secure than their previously used passwords, as shown in figure 12. By using our methodology, participants were able to create more secure passphrases, which is an encouraging result for our study. We requested our participants to provide a rationale behind their decision (optional). The majority of the users stated the longer length of the passphrase to be the most prominent reason for increased security, eg. *"It is longer, less predictable", "Longer strings are harder to crack by brute force, irrespective of l33t or other tricks. This is why companies have been moving away from suggesting complex passwords and instead focusing on length more (a password like 'MyElephantDumboIsHuge' is harder to crack and easier to remember than MrR0b0T@3142"*. Some of the users attributed the security to the inclusion of multiple languages, eg. *"Yes, I believe the new passphrase system suggested feels more secure since it combines different languages. Furthermore, it is easy to remember these passwords*

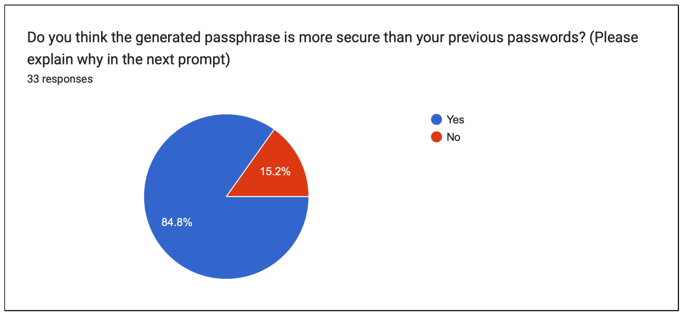

Figure 12. Graph depicting responses for the question related to the Security Evaluation

as they are in our regional language.",“Its a random bunch of words, and the language factor makes it more complex, hence its difficult to guess", and “It used multiple languages, so it's difficult for a hacker to run a brute force code to generate hashes for passwords with multiple languages.". A few of the users also pointed out to the variations in romanization, which could lead to added security, eg. *“It is hard to guess how you can spell the romanized words. Even if you know the password, you might not be able to correctly input it due to variation on romanized spellings.", “Non-english passwords seem harder to be performed a brute force attack on. Also the transcribing of a regional language is up to me. For example, I can say "nana" or "naanna" or "naanna" and they can all translate to "dad" in English. I think that makes it harder to guess my password even if an attacker knows my regional language."*

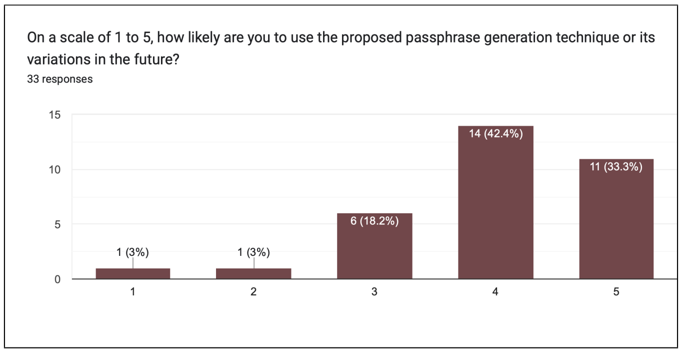

Figure 13. Graph depicting the question related to future use of the proposed methodology

4.4 RQ4: Future Prospects

Finally, we concluded the survey by asking the users if they would like to use our proposed methodology in the future. This survey form was anonymized. As seen from Figure 13, almost 75% of users stated that they will use it in future. This is an encouraging result for our analysis.

5 Limitations and Future work

RomIT is a truly-random generator of passphrases. However, studies have confirmed that the distribution of characters that people use to create their passwords is skewed [2] [4] [9]. For instance, users tend to use some alphabets more than the others, capitalize the first letter, use predictable suffixes like 123, ! and like. We believe that assuming users choose some words more frequently than others is a conservative assumption at best. Therefore, our results should be considered as reasonable upper bounds of used metrics. However, this upper bound also applies to the English passphrases that RomIT generates i.e RomIT is truly-random while generating only-English passphrases too. Therefore, we hypothesize that the delta between our baseline values and comparative metrics should not vary too much even with a non-random passphrase generator. We can attempt to prove this hypothesis by creating rulesets for RomIT denoting which words tend to be more commonly used and with what probability. We observed a mild case of this discrepancy in zxcvbn's evaluation of *ghoda@samundar_zindagi#daanth*. The estimator rightly identified *zindagi* owing to its high usage in Wikipedia articles.

Although RomIT is fully capable of mixing words from more than one non-English language to form passphrases, our study did not evaluate strength, randomness and resilience of such passphrases. We intend to carry out this portion of work in the near future.

We deem our analysis of resilience of passphrases to Hashcat's password cracking attacks limited by the shortcomings of the tool itself. We therefore plan to abstract password generation and hashing functionalities of Hashcat's open-source project to compare with our own. We speculate this comparison can give way to more fine-grained and interesting results that are not bound by the tool's restraints.

6 Conclusion

Passwords continue to be the dominant method for authentication. There is always a trade-off between strong passwords and usable passwords. In this work, we proposed an innovative approach for generating memorable as well as secure passphrases for multilingual users. We introduced our password generator *RomIT* which can produce 4 different corpora with romanized multilingual language passphrases based on the rules provided to it. We then evaluated our passphrases using statistical tools like zxcvbn, hashcat, and Shannon entropy. Our statistical analysis proves that using multilingual words increases the security of a password.

While results from this study highlight the strength and usability benefits of non-English passphrases, they also expose the limitations of password strength estimators and password cracking tools. While some of their limitations can be ameliorated by including words from non-English languages in their dictionaries, this might not be too feasible for estimators like zxcvbn which are designed to be lightweight and fast.

We finally conducted a pilot user study to evaluate the usability of our approach. From our pilot study, we conclude that users found our proposed approach more usable and secure than their previously used password-generation methods. An important inference from the user study is that participants found the “Longer passphrase with separators", using 4-5 individual words, most usable. This corpus also scored the best in our statistical analysis. Nevertheless, all other corpora of non-English passphrases scored better than only-English passphrases in terms of security. Based on these results, we recommend users to include words from non-English languages/regional languages to form passphrases of at least 4 words, with separators in between them (if that helps them remember it better).

References

[1] William J. Burns. 2019. Common password list (rockyou.txt). *Kaggle* (Jan 2019). https://www.kaggle.com/datasets/wjburns/common-password-list-rockyoutxt

[2] Matteo Dell'Amico, Pietro Michiardi, and Yves Roudier. 2010. Password strength: An empirical analysis. In *2010 Proceedings IEEE INFOCOM*. IEEE, 1–9.

[3] Julian Dunning and Anthony Weems. 2019. Password cracking rules for Hashcat based on statistics and Industry Patterns. *Github* (Jan 2019). https://github.com/praetorian-inc/Hob0Rules/blob/master/d3adhob0.rule

[4] Zhigong Li, Weili Han, and Wenyuan Xu. 2014. A {Large-Scale} Empirical Analysis of Chinese Web Passwords. In *23rd USENIX Security Symposium (USENIX Security 14)*. 559–574.

[5] Pardon Blessings Maoneke, Stephen Flowerday, and Naomi Isabirye. 2018. The influence of native language on password composition and security: A socioculture theoretical view. In *IFIP International Conference on ICT Systems Security and Privacy Protection*. Springer, 33–46.

[6] U.S. Department of Defense. May 1985. Password Management Guideline. *CSC-STD-002-85* (May 1985).

[7] Brian Roark, Lawrence Wolf-Sonkin, Christo Kirov, Sabrina J. Mielke, Cibu Johny, Işin Demirşahin, and Keith Hall. 2020. Processing South Asian Languages Written in the Latin Script: the Dakshina Dataset. In *Proceedings of The 12th Language Resources and Evaluation Conference (LREC)*. 2413–2423. https://www.aclweb.org/anthology/2020.lrec-1.294

[8] Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal* 27, 3 (1948), 379–423.

[9] Richard Shay, Saranga Komanduri, Patrick Gage Kelley, Pedro Giovanni Leon, Michelle L Mazurek, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. 2010. Encountering stronger password requirements: user attitudes and behaviors. In *Proceedings of the sixth symposium on usable privacy and security*. 1–20.

[10] Jens Steube and Gabriele Gristina. 2015. Hashcat advanced password recovery. *hashcat* (2015). https://hashcat.net/hashcat/

[11] Joshua Tan, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. 2020. Practical recommendations for stronger, more usable passwords combining minimum-strength, minimum-length, and blocklist requirements. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 1407–1426.

[12] Daniel Lowe Wheeler. 2016. zxcvbn:{Low-Budget} Password Strength Estimation. In *25th USENIX Security Symposium (USENIX Security 16)*. 157–173.

[13] Xiaoyuan Wu, Collins W Munyendo, Eddie Cosic, Genevieve A Flynn, Olivia Legault, and Adam J Aviv. 2022. User Perceptions of Five-Word Passwords. In *Annual Computer Security Applications Conference*. 605–618.