ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

EPSCM: An Energetic Password Strategy Creation Mechanism

[1] V.Nirmala, [2]Dr P.Chittibabu, [3]B.Sreenivasulu
[1] Assistant professor, [2] Principal, [3]Student
[1][3] MCA Department, [1][2][3] APGCCS,Rajampet

Abstract: Text-based passwords have been used widely in both online and offline applications, the passwords very easily carried, secure and personally use for all web services in. The mainly password user will create simple and common week and strong passwords they cannot replace the password in real time future that why raises the great security for the data base passwords through by using the offline cracking attacks. Another, we implement the password strength measure (password checker) based on password will create at the time the user identify password (I,e weak, strong, good) that can stored on the database it can provide the security to the password users. In password checker, by using the attack based model to crack the passwords. To address this type of passwords it will introduce the Energetic Password Strategy Creation Mechanism i ,e, (EPSCM). In this to keep safe the password database through password is distributed space and generated the dynamic structures to the users for create the alternative passwords. We further introduced the password diverse metric that evaluates the password database in terms of password distribution and space.

Index Terms—password, password checker, password diversity.

I. INTRODUCTION

Text-password based authentication schemes are a popular means of authenticating since last decades, for its easiness, cost effectiveness, directness and freedom to all users. The text based passwords are containing alphanumeric (digits, characters) and special keyboard characters and it was used as a shared secret leak incidents have happened recently and frequently. The most keep safe of information is the password strength measure is known as password checker. By using this password strength checker to create the passwords for websites and applications for now days. Which is evaluating the strength of passwords like good, strong, week during at the time of users registration. The users mostly will choose to create the strong passwords. In registration at the time user will observe the lack of accuracy and consistency. By using the attack based model it will easily find the password strength measure to crack the passwords of the database. Using an attack-based model, we show that the password

Checkers are effective for attackers to facilitate password cracking. With a certain amount of computational time, the attacker will attack the strength based passwords can compromise more passwords with a specific rating with the help of the strength checkers. This implies that the static policies and scoring mechanisms used by password strength checkers exert bias on the password characteristics distribution. Passwords with the same rating follow an obvious pattern which can be exploited by the attacker to refine the training data. By using the attack based model to observe the selective and non-selective strength based passwords after then it will attack the passwords in the database By defining a set of password creation policies and showing users password strength scores, password checkers can exert a strong bias on password characteristics, especially when the policies and scoring mechanisms remain static. The passwords registered to a database are largely similar to the specific password patterns enforced by the associated checker. When password checkers very among websites and applications, they is certain to happen rely on similar rules that focus on specific password properties (e.g., length, number of digits and special characters). When rules are relatively relaxed, password users may create simple passwords following a common distribution. When rules are relatively demanding, the password distribution is closely correlated to the scoring metrics and can be inferred. The password is related to scoring rating of the passwords will create using the attack based model it will crack the passwords in the database. To measure this weakness to provide the Energetic password strategy Creation Mechanism (EPSCM). EPSCM is providing the high security to the users and also password database. In EPSCM the password is distributed to password space and distribution because of to protect the password database. And also further we invent the password diverse

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

metric that is comparison of passwords. In this metric it will check the password policy history if is there exists or not it will find out based on to provide the security to password database. In EPSCM is which generates dynamic password policies for users. Each new user obtains a different password policy to follow, which is generated in real-time from the server based on but not reflecting the current password distribution. Since the policies users receive are dynamic, and unpredictable, an adversary cannot use them to infer the characteristics of the password database or select password training data.

II METHODOLOGICAL WORK PROCESS

(i) commercial password checkers

Traditional password policies have become less popular as the more user-friendly password strength checkers become widely adopted by major websites and software. The main reason is that good password policies can easily be too stringent to use, while password strength checkers push users to create "strong" passwords. Furthermore, due to the exposure of the policies and scoring mechanisms careful attackers can utilize the password checkers to mount more powerful attacks on passwords with high strength ratings.

A)Datasets, Checkers, and Crackers

Table I lists the 5 datasets that add up to around 81 million passwords. The datasets are leaked from several incidents , where attackers acquire passwords by online attacking techniques. Although the password data were leaked illegally, it has been once made publicly available and used widely password research for benevolent purposes. To obtain a collection of usable password strength checkers and cracking algorithms. Due to the space limitation, we only present two checkers listed in below table.

TABLE1
DATASETS

Name	Size	Language	Site	Type
Renren	4.7M	Chinese	renren.com/	social networks professional networks
LinkedIn	5.4M	English	linkedin.com/	
Tianya	31M	Chinese	tianya.cn/	Internet forum
Rockyou	32.6M	English	rockyou.com/	Game
Gamigo	6.3M	German	en.gamigo.com/	Game

TABLE2
PERCENTAGE OF "STRONG" PASSWORDS

checker	Gamigo	Rockyou	Tianya	Renren	LinkedIn
Bloomberg-Train	0.05%	6.30%	0.31%	0.72%	
Bloomberg-Test	0.05%	6.27%	0.31%	0.72%	
QQ-Train	12.44%	22.20%	1.75%	2.56%	
QQ-Test	12.44%	22.12%	1.74%	2.56%	

Bloomberg is a popular English business and news forum and QQ is a well-known Chinese portal providing numerous web services. They provide relatively accurate and consistent feedback to users. There are 4 levels of password strength in both password checkers to make them comparable, and the highest rating is "strong" in common. We use three state-of-the-art password cracking algorithms, JtR (John the Ripper-Markov), OMEN (Ordered Markov ENumerator), and PCFG (Probabilistic Context-free Grammar), which have relatively optimal performance in password cracking.

B. Threat Model: Take Your Checker, Crack Your Passwords

commercial password checkers can be used to enhance offline password attacks. In our threat model, we assume an attacker aims to crack a target set of password hashes leaked from a website which uses a password strength checker. This means that the hashed passwords can have different strength ratings. We also assume the attacker has access to the checker and obtained another dataset of plain text passwords leaked from other websites as prior knowledge, which is used to train the password crackers. However, the target passwords might have been created mostly by users who trust the strength feedback from the checker and create passwords only if they are labelled as "strong". Then, the target passwords are reasonably dissimilar from the training passwords which come from other sources. Therefore, to compromise such biased target passwords, the attacker will likely have better cracking results if the training passwords are also "strong".

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

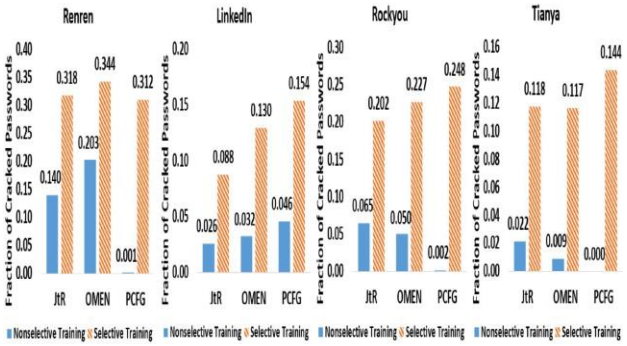

Fig 1. BLOOMBERG

Figure 1 summarizes the evaluation process. First, we randomly select 50% of the passwords from a dataset in to be the Nonselective Training dataset. Then, we apply a password strength checker in Table II to score each password in the Nonselective Training dataset, and select only those passwords label as "strong" to make up the Selective Training dataset. From the other 50% of the passwords, we apply the same checker selection method to build the Testing dataset. Finally, we use Nonselective Training and Selective Training datasets separately, as input to JtR, OMEN, and PCFG, to crack the Testing dataset.

In Table II, we show the percentages of selected passwords from the datasets, e.g., Bloomberg-Train and Bloomberg-Test indicate the percentages of "strong" passwords marked by Bloomberg's checker in the datasets from which we sample training and testing data, respectively. we perform passwords cracking in both Intra-site and Cross site scenarios. In Intra-site cracking, the training data and target data come from the same original dataset and in Cross site cracking, the training data is from a different dataset.

III.ENERGETIC PASSWORD GENERATOR

EPSCM is a diversity-based and database-aware application that generates password creation policies dynamically for the users. Instead of purely focusing on the complexity of candidate passwords, EPSCM enforces a baseline complexity on the passwords (e.g., more than 6 characters long) to protect them from simple attacks, e.g., dictionary, brute-forcing.

Type	Description
Length	use a range of password length
Composition	use a number of different character types
Alternation	use a number of character type transitions
Good Chars	include specific characters
Bad Chars	exclude specific characters
Structure	use a specific structure

Table 3 Password strategy Requirement Types

However, more focus is put on protecting the password distribution within a database by preventing aggregation of similar passwords that form a characteristically biased distribution. As long as a candidate password meets the policy, it is accepted and no additional strength feedback is provided. The policies are generated to search for candidate passwords that balance the password characteristics distribution.

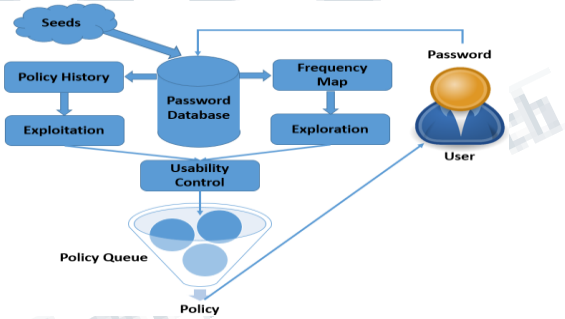

Fig2 :System Architecture

we show how EPSCM works. Initially, system administrators can place complex or random passwords as seeds in the password database. The seeds can form a white list to inject certain desired password characteristics, e.g., structures, n-grams. Based on the seeds, EPSCM can start to generate password policies to users. Since dynamically generating policies requires necessary computational time depending on the number of existing passwords, to avoid delay in responding to users' requests, a policy queue is used to store policies as a buffer each time when a batch of policies are created. When the size of the policy queue reduces below a threshold, e.g., 25%, EPSCM is signaled to generate new policies.

a) Two Modes: Explore and Exploit

There are two modes for EPSCM to expand the usable password space and balance the password distribution. The exploration mode mainly aims to expand the password space by actively introducing new characteristics based on the global characteristics frequency map. Before an incoming password is hashed,

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

EPSCM extracts its characteristics and stores the metadata in the frequency map, which keeps tracks of the overall distribution of password attributes e.g., frequency of structures, characters, and denotes the current password space. In exploration mode, EPSCM creates policies that require users to be more "creative" in making a password e,g., using the character "(" which is not usual even in special characters. In this way, the passwords can cover a larger textual search space than the regular human linguistic patterns. Initially when there are not many passwords, a random mechanism is adjusted to launch the exploration mode more often to aggressively enlarge the password space. When the password characteristics distribution is relatively uniform as observed from the exploitation mode, the exploration mode is also evoked to introduce new characteristics. Since purely expanding the password space is equivalent to making random passwords, EPSCM also relies on another major component. The exploitation mode aims to enhance password diversity and balance the current password distribution, with the help of the password policy history. Since passwords are hashed in the database, EPSCM stores previously generated password policies to approximate current password distributions and analyze the password diversity. EPSCM then identifies password characteristics that exist in the database with low appearance frequencies, and generate policies that require such characteristics. Therefore, EPSCM creates policies that are usable and balance the password distribution by temporarily increasing the frequencies of less common password attributes. Based on the two modes, EPSCM determines the critical characteristics requirements, but only renders the final policies after passing them to the usability control module.
Include the character(s): 'v', 'Z'
Avoid the character(s): a, s, e
Use the structure: LLLLLUUS
Number of characters: 8 to 12 (inclusively)
Number of character types: 4
Number of alternations: 3 to 4 (inclusively)
Include the character(s): '?', 'U', ')'.

b) Exploration

In exploration mode, EPSCM creates policies that require users to be more creative and innovative in making a password e,g., using the character, alternative characters, special numbers, upper case letters. In this way, the passwords can cover a larger textual search space than the regular human linguistic patterns. Initially when there are not many passwords, a random mechanism is adjusted to launch the exploration mode more often to aggressively enlarge the password space. When the password characteristics distribution is relatively uniform as observed from the exploitation mode, the exploration mode is also evoked to introduce new characteristics. In exploration is maintained by the administrator will create the passwords based on the policy structure. The policy structure and length should be the same. The policy structure based will divide the password distribution. After distribute the password policy structure it will inserted into the password database. In exploration mode update is available, if any updatation is their in the structure it will modify them.

c) Exploitation

In exploitation mode, it is used for to compare the new created policy structure into policy history. In policy history, all the stored policies database will given to them. If enter newly created the policy structure after it will check the policy history if there exists the structure it will displays the message if the policy structure is already exists in the database to enter the another new structure. Given below figure it will displays:

Fig 3: Exploitation

In exploitation mode, if the enter the policy structure is not exists, it will displays the message use this policy structure for creation.

d) Seed

The seeds can form a all the database stored password policies in the seed. password characteristics, e.g., structures, password length. Based on the seeds, EPSCM can start to generate password policies to users. Since dynamically generating policies requires necessary computational power depending on the number of existing passwords, to avoid delay in responding to users' requests, a policy queue is used to store policies as a memory each time when a batch of policies are created.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

e)Usability Analysis

The usability of EPSCM can translate into the ability for usersto follow the policies and maintain the passwords they create.In this section, we test EPSCM on real users and collect passwordsfor further analysis. We aim to show the comparison of usability between commercial password checkers and EPSCM. Recruitment. After our protocol was approved by the Institutional Review Board (IRB), we conducted a usability test of EPSCM on Amazon Mechanical Turk [19]. We restricted the participants to a qualification type that requires at least 95% of approval rate, and 500 approved tasks. We excluded minors and only included English speakers. Protocol. Our approach is to test if users who create their passwords by EPSCM policies can successfully remember them for a reasonable time period. We also require the same participants to create passwords using QQ's password strength checker as the control group. The participants are not informed of the purpose of our study or anything introduced in this paper. Each participant who accepts our human intelligence task (HIT) on Amazon Mechanical Turk is asked to access our web server with registration and login services. Participants are asked to complete 4 sessions of experiments which are separated by time intervals of 24 hours, 48 hours, and 72 hours to finish the entire study. In the first session, the participant is directed to visit two artificial websites to register two accounts with usernames and passwords, following a EPSCM policy and using QQ's checker, respectively. Then the participant simply concludes the session by logging into the accounts with the credentials they just created. For the rest of the sessions, participants simply return to our web interface during the time specified at the end of each previous session and logged into the accounts with their credentials. All participants are informed in the beginning of to remember the passwords up to a week. Of the 7 participants who forgot their passwords in at least one session, 4, 2, and 1 of them had to re-create their passwords in exactly 1, 2, and 3 sessions, respectively. Most of the participants who had to re-create passwords in one session did it in session 2 and if the participant successfully logs in in session 2, it is almost certain for them to pass the rest of the sessions. QQ's checker as the control group shows very similar statistics but is slightly better. This demonstrates that policies generated dynamically from EPSCM are not less usable, at least compared to existing password checkers, in terms of the ability of users to maintain the passwords. Furthermore, the passwords from EPSCM and QQ's checker share 54 out of 90 passwords in common. Since password policies from EPSCM are dynamic, unpredictable and in various templates, a more likely explanation for this phenomenon is that about half of the users reused passwords created with EPSCM for the QQ' checker. Although reusing passwords is a bad practice, this implies that users either reuse the passwords to get strong strength feedback in QQ's checker, or to better maintain the passwords.

In a final survey, we further obtain subjective feedback on the usability of EPSCM and QQ's checker. When asked about the ease in following the policies or strength feedback, 63.75% and 73.75% of participants thought EPSCM and QQ's checker, respectively were above average. In addition, 65.43% and 61.73% thought EPSCM and QQ's checker, respectively.

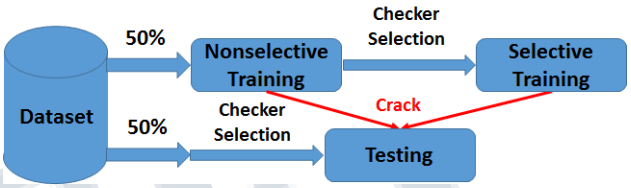

Fig 4 : Attack Based Model

f) Evaluation of the Diversity Measure

To evaluate the robustness of our metric, we look at both the effectiveness in its protecting passwords from cracking.

EPSCM is the first password policy generator that can generate password policies dynamically according to the password distribution. Since there is no password strength feedback is returned and the policies generated by dynamic and unpredictable, the attacker will find it extremely difficult, if not possible. The policies themselves are in different formats and only contain information that is ideally contrary to the distribution in the database, because EPSCM always tries to balance the current distribution and expanding password space. The performance of the EPSCM is the provide the high security to password database and also to provide the user no one can't be attack the password. When the user registration at the time the password database is provide the password policy structure to the user differently because of attacker cannot find out the structure.

IV CONCLUSION

In this paper, we study the password space and distribution to understand password dataset security better. Due to the limitation of existing strength measuring mechanisms, we propose a new and usable alternative based on an effective diversity metric to better protect passwords from offline cracking attacks. We start by identifying issues with the existing commercial password strength checkers and evaluate them from the

adversarial perspective. While previous work has analyzed the consistency and accuracy of the checkers, much effort has not been spent on their limitations of biasing and leaking password distributions to the adversary. Through our evaluation, we find that password strength checkers are effective in helping attackers mount more powerful attacks. The reason is that password strength checkers rely on static scoring policies that exert bias on the password distribution. The checkers can be leveraged by the attackers easily to select training data that are similar to the target passwords. To propose an effective alternative that addresses the limitations of password strength checkers, we implement DPPG to generate dynamic policies for users, which is based on a password diversity metric and the current password distribution. To the best of our knowledge, DPPG is the first dynamic password policy generator that provides unpredictable dynamic policies and enforces usability control. Through exploration and exploitation modes, DPPG can expand the password space and balance password characteristics distribution which increase the overall security of the password dataset. Through a usability study, we test DPPG in practice and collect passwords for further analysis. Experiments are also conducted to show that the collected passwords are more diverse in their attributes and have good security. To study the password distribution and its security impact, we define the concept of password diversity. To the best of our knowledge, this is the first attempt to define and quantify password diversity considering a vector of password attributes.

In future work, It can be quantification is extensible and can be adjusted with different weight values to shift the focus of measurement. To provide a way to analyze the password diversity of a dataset, we propose the diversity-based password security metric which is a key component for DPPG to generate effective policies. We also evaluate the metric from an adversarial perspective using it to sample data for an attack-based evaluation. Through cracking experiments in different setups, we conclude that the metric is effective in evaluating the security of password datasets and thus can serve as an effective start to evaluate password dataset security with regards to password diversity.

REFERENCES

[1] C. Herley, P. C. Oorschot, and A. S. Patrick, “Passwords: If we’re so smart, why are we still using them?” FC, 2009.

[2] D. Florˆencio and C. Herley, “A large-scale study of web password habits,” WWW, 2007.

[3] D. Florˆencio and C. Herley, “Where do security policies come from,” SOUPS, 2010.

[4] M. Weir, S. Aggarwal, M. Collins, and H. Stern, “Testing metrics for password creation policies by attacking large sets of revealed passwords,” CCS, 2010.

[5] X. C. Carnavalet and M. Mannan, “From very weak to very strong: Analyzing password-strength meters,” NDSS, 2014.

[6] S. Ji, S. Yang, and R. Beyah, “Pars: A uniform and open-source password analysis and research system,” ACSAC, 2015.

[7] J. Yan, A. Blackwell, and R. Anderson, “Password memorability and security: Empirical results,” S&P, 2004.

[8] S. Ji., S. Yang, X. Hu, W. Han, Z. Li, and R. Beyah, “Zero-sum password cracking game: A large-scale empirical study on the crackability, correlation, and security of passwords,” Dependable and Secure Computing, IEEE Transactions on, 2015.

[9] A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang, “The tangled web of password reuse,” NDSS, 2014.

[10] J. Ma, W. Yang, M. Luo, and N. Li, “A study of probilistic password models,” S&P, 2014