Generating Honeywords From Real Passwords With Decoy Mechanism

[1]Ms. Komal Naik, [2]Prof. Varsha Bhosale, [3]Prof.Vinayak D.Shinde
[1]M.E. Student, Department of Computer, SLRTCE-Mumbai University, Maharashtra, India.
[2]Associate Professor, Department of Information Technology, VIT-Mumbai University, Maharashtra, India.
[3]Assistant professor, Department of Computer, SLRTCE-Mumbai University, Maharashtra, India.

Abstract - As society and business is more dependent on digital data, the threat continues to rapidly increase. Every year new mechanism against cyber security threats are introduced. But simultaneously the cybercriminals also create new techniques those overcome these efforts. Hence considering security and data protection as a priority new developments are needed. One of the important security issue is with disclosure of password file. To tune up this issue the concept of honeywords i.e. false password is introduced. If an attacker steals the password file it will include the original password and honeywords. For an adversary it will be difficult to distinguish between the genuine and decoy passwords. If he enters the honeyword for login it will trigger an alarm notifying the administrator about a password file breach. In this system, if the number of attempts exceeds the count of three or enter the password other than honeyword then the access will be issued but the files available will be decoy files. Thus, decoy mechanism secures the data of the legitimate user. System keeps the data of tracked IP's with them and use them to take appropriate action against the malicious users.

Keywords -- Authentication, Decoy, Honeywords, Login, Password, Security.

I. Introduction

Information security has become a most prominent requirement in this era which is secured using some authentication method. Many different methods for authentication exists (e.g. PINs, Patterns, Passwords etc.). Now-a-days most generally used method for authentication is passwords. Security of password is an important aspect. A password is a secret word, which a user must input during a login, only after that it is possible to get access. In password based systems, application developers must take care that passwords should not be stored in databases in plaintext or with unsalted hash values. In past, many hacking attempts had made possible for attackers to gain unauthorized access to the sensitive data as well as user passwords stored in database. Password protection helps us to protect information from unauthorized users. Revealing of password files is a serious security problem that has affected many users and companies like Yahoo, LinkedIn, eHarmony and Adobe since revealed passwords cause many possible cyber-attacks. These leaks have proved that many large companies are using weak hashing techniques that make hackers easy to crack the user passwords. The passwords in the eHarmony system were stored using MD5 hashes without salt and also the LinkedIn passwords were also stored with unsalted hash values by using SHA-1 algorithm [1]. To control these security problems, following two issues should be considered while developing a secure system. First, passwords must be stored in database using appropriate and strong hashing mechanism so that it will be difficult for attacker to reverse the hashes. Second, password file breach should be detected to take appropriate actions.

Many researchers have already worked for password security. Earlier, to protect online banking accounts from brute-force attacks, Herley and Florencio [2] proposed a new approach to detect the malicious behaviour on every incorrect or unauthorized login. For every single user false login attempts with few passwords will generate honeypot accounts (fake accounts) so that malign behaviour is caught. Recently, Juels and Rivest have presented the honeyword mechanism to detect an adversary who attempts to login with cracked passwords [3]. The concept is that for each username they build a set of sweetwords in which one word is the real password and the others are honeywords (false passwords). This approach was proposed considering the second issue about detecting the malicious entry. When an adversary tries to get access using any of the honeyword an alarm is triggered which notifies the administrator about the password file breach.

Recently, it's identified that for security of the system the key requirement is the honeyword generation algorithm such that generated honeywords should not be easily distinguished from the actual or correct passwords. The study includes honeyword based authentication approach in which it's sure that the attacker will be detected. The aim of this study is to validate whether data access is authorized or not when abnormal information access is detected. Confusing the attacker with decoy data protects against the misuse of the user's real data. Thus, decoy mechanism secures the data of the legitimate user. Our system keeps the data of tracked IP's with them and use them to block access on their network. This approach not only provide detection but also helps for avoidance.

II. LITERATURE SURVEY

PC users and businesses use computers for numerous functions, including transferring and storing digital data. Users may wish to keep this information restricted for a number of reasons. User authentication can be used for this purpose. User authentication is a means of verifying, the identity of an individual requesting access to certain information. There are several ways user authentication can be verified. The user can confirm his identity via secret information the requester uniquely knows, such as a password. So, Password Security is an important aspect. Compared with traditional password based authentication methods, Honeyword based authentication method has shown better results.

1. The Dangers of Weak Hashes

K. Brown in 2012 discussed that by implementing few best practices the damage caused by password leak can be minimized. It was found that there had been several high publicity password leaks in June 2012 including LinkedIn, Yahoo, and eHarmony. Secure system should not have any vulnerabilities that will allow hackers to get access to password files and should make sure that if the password hashes are been hacked it should not be easy to generate passwords from the hashes. Thus these leaks have proved that many companies are following weak hashing mechanisms. In this paper the author has discussed about basics of password hashing and best practices that should be followed while password storage [1].

2. Honeywords: Making Password-Cracking Detectable.

Juels and Rivest in 2013 proposed a method for improving the security of hashed passwords.To improve the security of the hashed password, honeywords (false passwords) needs to be generated for each user account. An attacker who steals a file of hashed passwords and inverts the hash function cannot tell if he has found the password or a honeyword. If the attacker attempts to login with the use of honeyword the auxiliary server will set off an alarm. When honeywords are used a successful brute-force password break does not give the adversary confidence that he can log in successfully without detection. Proposed system includes honeyword generation algorithms and comparison with different factors[3].

3. Guess again (and again and again): Measuring password strength by simulating password-cracking.

In this study in 2012 authors understood the effects of password composition methods on guessing ability of passwords. Text based passwords is the leading authentication method used in computer systems, in-spite of rapid advancement in attackers capabilities to perform password cracking. Considering this threat, password composition methods are becoming complex day-by-day. But, there is less research work defining the metrics to outline password strength. In this paper a new, efficient technique for calculating the password strength which can be implemented for a variety of password-guessing algorithms and tuned using a variety of training sets to gain insight into the comparative guess resistance of different sets of passwords is introduced [4].

4. Examination of a New Defense Mechanism: Honeywords.

It has become much easier to crack a password hash with the advancements in the graphical processing unit (GPU) technology. Once the password has been recovered no server can detect any illegitimate user authentication (if there is no extra mechanism used). They propose an approach for user authentication, in which some false passwords, i.e., "honeywords" are added into a password file, in order to detect impersonation. The authors in propose an interesting defense mechanism under a very common attack scenario where an adversary steals the file of password hashes and inverts most or many of the hashes. The honeyword system is powerful defense mechanism in this scenario. Namely, even if the adversary has broken all the hashes in the password file, he cannot login to the system without a high risk of being detected. Hacking the honeychecker has also no benefit to the adversary since there is no information about a user's password or honeyword in the honeychecker [5].

5. Achieving Flatness: Selecting the Honeywords from Existing User Passwords.

The system proposed in this paper works on the issue to overcome the security problems. A new honeyword generation algorithm which shows better results with respect to flatness, DOS resistance and storage is proposed. New generation algorithm uses a different method that selects the honeywords from existing user passwords in the system which provides realistic honeywords and provides a new honeyword generation method and also reduces storage cost of the honeyword scheme [6].

III. HONEYWORDS

In this section, we summarize the existing honeyword based model, various honeyword generation techniques and its security issues.

A. Existing Honeyword Model

Honeywords are false (decoy) passwords. For each user account, the legitimate password is stored with several honeywords in order to sense impersonation. The file of hashed passwords include the legitimate as well as decoy passwords. After inverting the hashes if the honeywords are selected properly, an attacker who steals a file is not sure if it is the real password or a honeyword for any account. If he enters the honeyword for login, it will trigger an alarm notifying the administrator about a password file breach. Various existing honeyword generation algorithms are used to generate honeywords. Thus, this approach puts the attacker at risk of being detected with every attempted login. Using brute force attack also it is sure that the attacker will be detected. Hence, honeywords can provide a very useful layer of defense.

B. Honeyword Generation Methods

Honeyword generation algorithms can be categorized into two groups.

Legacy-UI procedures: The password-change UI is unchanged i.e. it takes the same password entered by the user for honeyword generation [6]. Chaffing by tweaking concept is used to generate the honeywords. In this method, the user password is given as an input to the generator algorithm, it tweaks the selected positions of the correct password to generate the honeywords. Each character of the selected positions are replaced by randomly chosen character and a set of honeywords are formed. Consider a method Gen(m; d) in generator algorithm wherein d is the number of characters to be replaced. For example if tweaking last 2 positions from the real password, the d = 2. If user password is abc213 and d = 2, then honeywords abc289, abc245 may be generated [6].The various algorithms under this procedures are mentioned below in figure. Above method is weak in security standards but provides good usability standards. Generation algorithm should be such that the real password should not be easily distinguishable from honeywords.

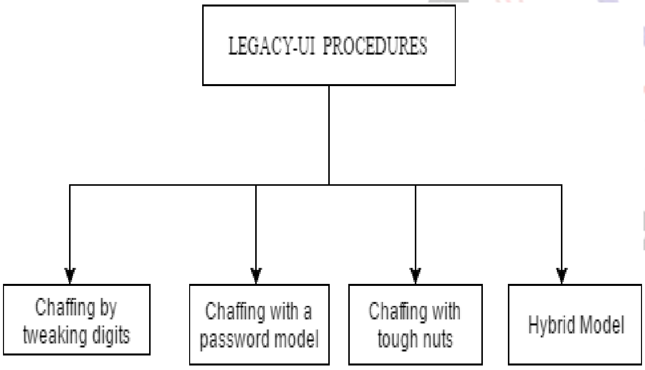


Figure 1: Types of Legacy-UI procedures algorithm[6].

Modified-UI procedures: The password-change UI is modified to allow better honey-word generation. The user's actual password is modified to end with a randomly chosen value to form a new user password. Take-a-tail method is an example of this category. For example, abcd23 is user's entered password then system generates '@12' as a tail. So now user's new password becomes abcd23@12. Above method has highest security standard but is very poor in usability standard. It is very difficult for user to remember the system generated information for his different accounts [6].

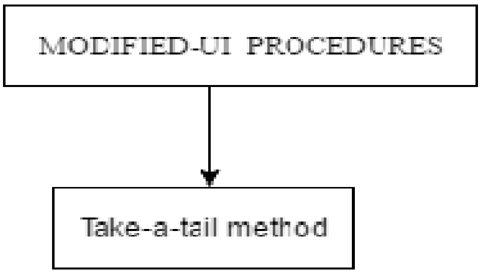


Figure 2: Type of Modified-UI procedure algorithm[6].

IV. PROPOSED MODEL

We use honeywords mechanism to launch disinformation attacks against unauthorized insiders, preventing them from distinguishing the original sensitive customer data from fake worthless data. In this research we will use an already well-established method of honeyword generation and have used a logic of ASCII to generate honeywords and SHA-256 algorithm for hashing. The attempted use of a honeyword for login will set off an alarm to the administrator and the unauthorized user will be given access to decoy files. System will also keep track of IP. Using IP tracking we can avoid unwanted request from a single system thus reducing the unnecessary computation.

A. Working Principle

In this section, the brief explanation of the working principle of the various sections which will be used in the system is mentioned.

Registration: An authenticate person who has an authorized access to the system is said to be a user. Here, User is going to register into system. While registration, for the given password by the user the system generates honeywords using honeyword generation technique. Our modified honeyword generation technique will be based on Legacy UI procedures concept but also ASCII logic will be added to meet more security standards. By using strong and secure hashing techniques the hashes for honeywords as well as actual password are generated and stored into the table in database. Along with Hash Values the original password hash is also stored at specific random position. While registration a valid e-mail id need to be provided. Our system meets security standards like key loggers, visible passwords, guessing attack etc. to keep the details safe from the hacker.

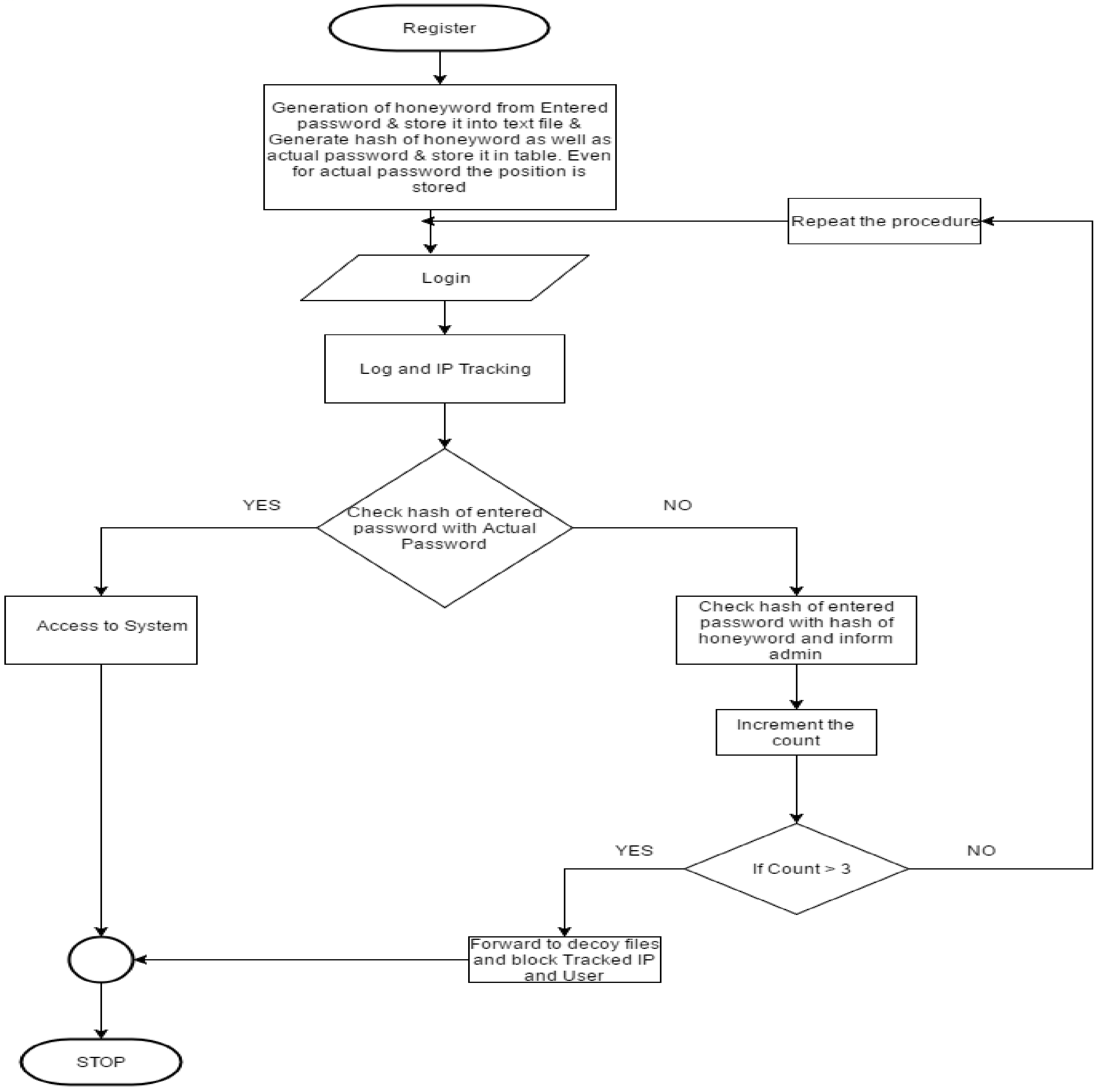

Register
Generation of honeyword from Entered password & store it into text file & Generate hash of honeyword as well as actual password & store it in table. Even for actual password the position is stored
Repeat the procedure
Login
Log and IP Tracking
YES
Check hash of entered password with Actual Password
NO
Access to System
Check hash of entered password with hash of honeyword and inform admin
Increment the count
YES
If Count > 3
NO
Forward to decoy files and block Tracked IP and User
STOP

Figure 3: Proposed System.

Login: User login into the system using his unique email id and password. If password matches with the hash of the original password then user gets access to the system. There are chances that user may also make typo errors while entering the password but in secure system it will be considered as an invalid login attempt. User will be given 4 chances, but if the user exceeds the count he will be also blocked and a new password which should be used for login will be provided to his mail id. User can upload as well as view the files after a successful login to the system.

Hacker: Hacker tries to login into the system. If he enters any honeyword then the alert is given to the Actual user and the admin through an email. And if suppose he try combination of password or any honeyword and it goes more than three attempt then he get access but to the decoy files. After cracking the password file he tries for the login, at the same time system track the IP address and is stored into the database.

Admin Login: Admin can login into system. He can manage the users (add, delete, etc.) of the system. Uploading the decoy files which are used to protect the actual files from the hacker. Receives an alarm notifying a password file breach in case of unauthorized access to the system or incorrect login. Keeps the track of IP as well as number of attempts the login to the system was done. This track helps to take necessary action against malicious user.

Log and IP Tracking: Log creation is done for each user action to the system and which is store into the database. After every registration the details about the date and time are stored in database for each user id. After valid user login, the system will track the valid user operations and track IP Address and data size of resources downloaded by each user per session. For every login, whether the attempt is valid or invalid the IP and the location is tracked. Log of number of attempts is also maintained in the database for every user id which will be help us to take necessary action against that adversary if he exceeds the count of greater than three.

The proposed system uses Chaffing by tweaking method to generate honeywords. The traditional method of chaffing by tweaking digits is modified to generate more confusing honeywords. In this method the generation algorithm uses ASCII values to replace or tweak the characters of the original password to form the honeywords. Generation algorithm should be such that the real password should not be easily distinguishable from honeywords. In this generation algorithm we have tried to improve the security standards so that the original password cannot be easily identified. All Legacy UI procedures are best in usability standards as compared to modified UI procedures. In our system, add on modules includes decoy data and IP tracking. Decoy data mechanism will keep the actual or sensitive data safe from being hacked and IP tracking will help the admin to take necessary action against the unauthorized user.

B. *HONEYWORD GENERATION ALGORITHM*

The proposed system uses Chaffing by tweaking method to generate honeywords. The traditional method of chaffing by tweaking digits is modified to generate more confusing honeywords. In this method the generation algorithm uses ASCII values to replace or tweak the characters of the original password to form the honeywords.

Step 1: User registers with username and password.

Step2: Original password is assigned a random position.

Step3: The last 3 digits of the password are replaced those are the number of characters to be tweaked.

Step4: (a) A random number between 33 to 126 is assigned to the last character of the password.

(b) The original character is been replaced by the ASCII value of the randomly generated number.

Step5. Repeat step 4(a) and (b) for last three characters of the password.

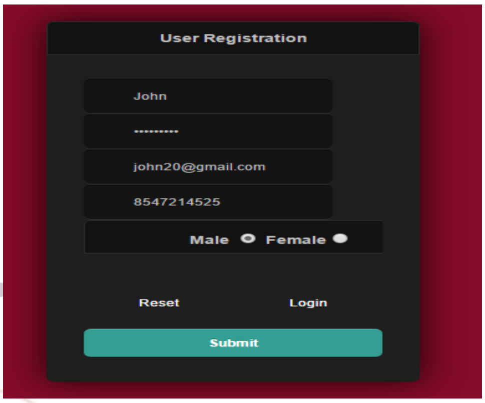
Step6: Repeat steps 3 to5 for n number of honeywords.

C. *Minimum software and hardware requirements for proposed model*

The proposed system discussed above will be required MySQL, JSP on Windows XP/7, 1.1 GHz, 2 GB RAM.

V. RESULTS AND DISCUSSION

The results include the user registration page where he needs to enter the valid email-id and the password. Once the user password is entered the modified chaffing by tweaking algorithm is applied on the original password to form n number of honeywords. Later hashing operation is performed on the sweetwords using SHA-256 algorithm and the hashes are stored into the database.



User Registration

John

----

john20@gmail.com

8547214525

Figure 4: User Registration

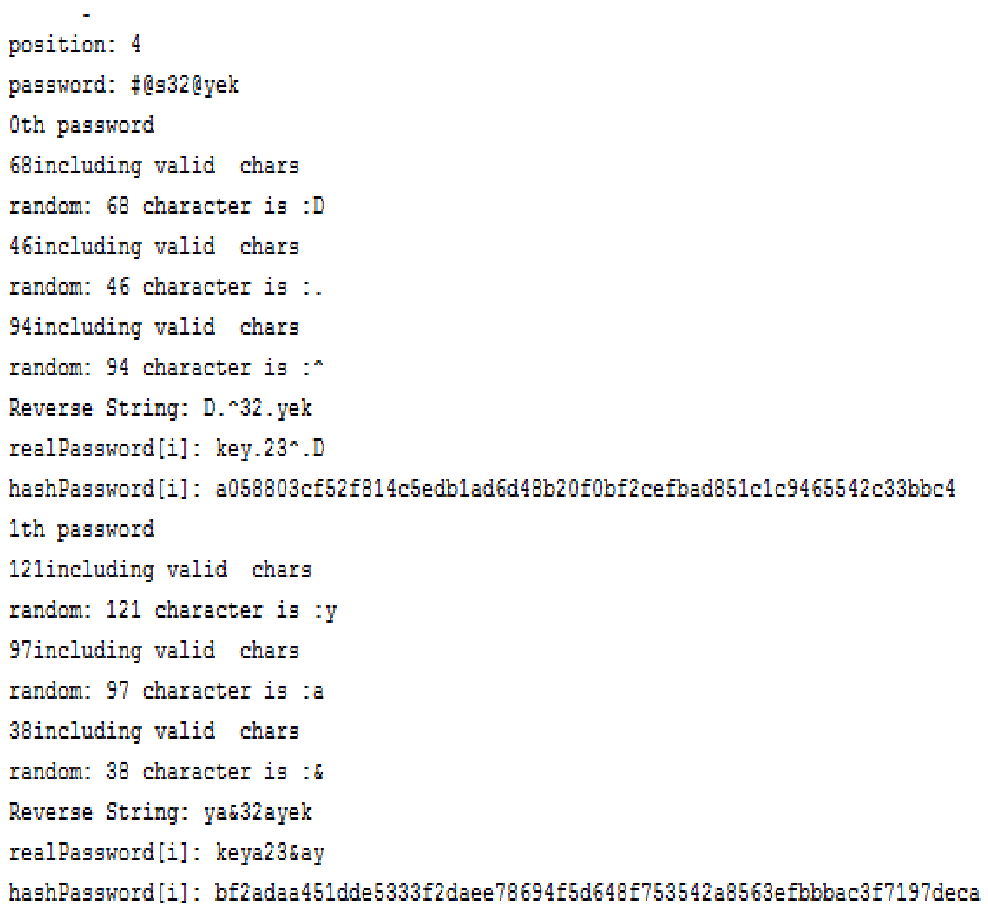
After registration the user login into the system and the generated honeywords and the hashes are as follows. Initially the random position is assigned to original password. This position will be between 3 to number of honeywords generated(n) – 1, because if the attempts are greater than 3 than unauthorized user will be given access to decoy files. Later for last 3 characters the random numbers between 33 to 126 are selected randomly and the original characters are replaced with ASCII values to generate honeywords. The screenshot shows the details about the modified chaffing by tweaking algorithm working and SHA-256 hashing.

For example consider the user input password as key@23s@#. In our study, the generation algorithm tweaks last 3 digits and generates 9 honeywords. The list of sweetwords (i.e. honeywords and actual password) is shown below.

keyn23&n4 keyR23UR6 keyj23(j+
key@23s@# keyu23Ju< keyw23]wY
key(23E(\ keyu236u8 keym23[mL

List of honeywords

Generation algorithm should be such that the real password should not be easily distinguishable from honeywords. In this generation algorithm we have tried to improve the security standards so that the original password cannot be easily identified. All Legacy UI procedures are best in usability standards as compared to modified UI procedures.



```
       -
position: 4
password: #@s32@yek
0th password
68including valid  chars
random: 68 character is :D
46including valid  chars
random: 46 character is :.
94including valid  chars
random: 94 character is :^
Reverse String: D.^32.yek
realPassword[i]: key.23^.D
hashPassword[i]: a058803cf52f814c5edb1ad6d48b20f0bf2cefbad851c1c9465542c33bbc4
1th password
121including valid  chars
random: 121 character is :y
97including valid  chars
random: 97 character is :a
38including valid  chars
random: 38 character is :&
Reverse String: ya&32ayek
realPassword[i]: keya23&ay
hashPassword[i]: bf2adaa451dde5333f2daee78694f5d648f753542a8563efbbbac3f7197deca
```

Figure 5: Honeyword Generation and Hashing

VI. CONCLUSION

Password security has always been a domain of active research. Honeyword based authentication have proved better results in this domain. The big difference between the traditional methods and when honeywords are used is that a successful brute-force password attack does not gives the attacker confidence that he can log in into system successfully without being detected. Research on better honeyword generation techniques has already been proposed with respect to security, usability, flatness, DOS resistance and storage. The use of decoy data mechanism will secure the confidential data of the authorized users from the hacker. In honeyword based authentication approach, it is sure that the attacker will be detected. The main aim of project is to validate whether data access is authorized or not when abnormal information access is detected. Confusing the attacker with decoy data protects from the misuse of the user's real data. The admin keeps the data of the tracked IP's with them and use them to block access on their network. Use of honeywords is very useful and works for every user account.

REFERENCES

[1] Brown and Kelly, "The dangers of weak hashes," SANS Institute InfoSec Reading Room, November 2013.

[2] H. C and F. D, "Protecting financial institutions from brute-force attacks," in IFIP International Information Security Conference, Springer US, September 2008.

[3] J. A and L. R. R, "Honeywords: Making Password cracking Detectable," in ACM SIGSAC conference on Computer & communications security, November 2013.

[4] Kelley, Patrick Gage and Michelle L. Mazurek, "Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms," IEEE Symposium on Security and Privacy, pp. 523-537, May- 2012.

[5] Genc, Ziya Alper and Mehmet Sabir Kiraz, "Examination of a New Defense Mechanism: Honeywords," IACR Cryptology ePrint Archive, p. 696, 2013.

[6] Imran Erguler, "Achieving Flatness: Selecting the Honeywords from Existing User Passwords," IEEE Transactions on Dependable and Secure Computing, vol. 13, no. 2, pp. 284 - 295, February 2015.

[7] M. Dell'Amico, P. Michiardi and Y. Roudier, "Password Strength: An Empirical Analysis," INFOCOM'10: Proceedings of the 29th Conference on Information Communications, vol. 10, pp. 983-991, 2010.

[8] Mirante, Dennis and Justin Cappos, "Understanding Password Database Compromises," Dept. of Computer Science and Engineering Polytechnic Inst. of NYU, 2013.

[9] Bonneau and Joseph, "The science of guessing: Analyzing an anonymized corpus of 70 million passwords," IEEE Symposium on Security and Privacy, pp. 538-552, May 2012.

[10] M. Bercovitch, M. Renford and L. Hasson, "HoneyGen: An automated honeytokens generator," in IEEE International Conference on Intelligence and Security Informatics, July 2011.

[11] ttp://www.rsaconference.com/writable/presentations/file_upload/dsp-w02-honeywords-a-new-tool-for-protection-from-password-database-breach_final.pdf.

[12] R. Morris and K. Thompson, "Password Security: A Case History," Communications of the ACM, vol. 22, no. 11, pp. 594-597, 1979.

[13] M. Raza , I. Muhammad, S. Muhammad and H. Waqas, "A Survey of Password Attacks and Comparative Analysis on Methods for Secure Authentication," World Applied Sciences Journal @ IDOSI Publications, vol. 19, no. 4, pp. 439-444, 2012.