IMPROVING SECURITY AND ACHIEVING FLATNESS TO THE USER-PASSWORD BY USING HONEYWORDS

Lalit Hajarimal Suthar[1], Vimal Chetan Singh[2], Mahesh Arun Umale[3], Hemalata Manohar Khandagle[4]

[1] Student, Department of Information Technology,
SKN Sinhgad Institute of Science & Technology, Lonavala, Maharashtra, India
[2] Student, Department of Information Technology,
SKN Sinhgad Institute of Science & Technology, Lonavala, Maharashtra, India
[3] Student, Department of Information Technology,
SKN Sinhgad Institute of Science & Technology, Lonavala, Maharashtra, India
[4] Student, Department of Information Technology,
SKN Sinhgad Institute of Science & Technology, Lonavala, Maharashtra, India

ABSTRACT

Authentication and Authorization of user is important to get access into any secured system for which username and password is necessary. Disclosure of password files is a serious security Issue, for which Juels and Rivest proposed honeywords (decoy passwords) to detect attacks against Hash password databases. To protect hash password databases from third party, we implement Honeywords which converts a valid password into a new hash passwords. New passwords is the combination of existing password called honeywords, which are nothing but the Fake passwords. If proper honeywords are chosen, a cyber-attacker who steals a file of hashed passwords would be confused, whether it is a real password or honeyword for user Account. Login with honeywords will trigger an Alarm, which will notify the administrator about a password file Infraction. To identify or detect occurrence of password database infraction honeypot concept is used. So we implement an easy and efficient solution which will protect and detect exposure events of passwords file.

Keyword: - *Authentication, Authorization, Infraction, honeywords, login, honeypot.*

1. INTRODUCTION

Leakage of password files is a severe security problem that has affected millions of users Accounts and companies like Yahoo, LinkedIn and Adobe, since disclosed passwords make the users target of many possible cyber-attacks. These recent activities have demonstrated that the weak password storage methods are currently in place on many web-servers. For example, the LinkedIn passwords encryption were using the SHA-1 algorithm without a salt and similarly the passwords in the eHarmony system were also stored using unsalted MD5 hashes. Indeed, once a password file is stolen, by using the password cracking techniques like the algorithm of Weir et al. it is easy to capture most of the plaintext passwords. In this respect, there are two problems that should be considered to solve these security problems: First, passwords must be protected by taking right precautions and storing with their hash values computed through salting or other complex mechanisms. Hence, for an attacker it must be hard to invert hashes to get plaintext passwords. The second point is that a secure system should detect whether a password file leak incident happened or not to take appropriate actions. In this study, we focus on the latter issues and deal

with fake passwords or accounts as a simple and cost effective solution to detect compromise of passwords. Honeypot is one of the best solution to identify occurrence of a password database breach. In this method, the administrator purposely creates fake user accounts to lure adversaries and detects a password leakage, if any one of the honeypot passwords get used. This idea has been modified by Herley and Florencio to protect online banking accounts from password bruteforce attacks. According to the study, for each user incorrect login try with some passwords lead to honeypot accounts, i.e., malicious behavior is recognized. For instance, there are 108 possibilities for an eight-digit password and let system links 10,000 wrong password to honeypot accounts, so the adversary performing the bruteforce attack 10,000 times more likely to access a honeypot account than the genuine account. Use of decoys for building theft-resistant was introduced by Bojinov et al. in called as Kamouflage. In this model, the fake password sets are stored with the real user password set to mask the real passwords, thereby forcing an adversary to carry out a considerable amount of work before getting the Right information. Recently, Juels and Rivest have presented the honeyword mechanism to detect an adversary who attempts to login with cracked passwords. Basically, for each username a set of sweet-words is generated such that only one element is the correct password and the others are honeywords (decoy passwords). Hence, when an adversary tries to enter into the system with a honeyword, an alarm is triggered to notify the administrator about a password leakage.
In this study, we analyze the honeyword approach and give some remarks about the security of the system. Furthermore, we point out that the key term for this method is the generation algorithm of the honeywords such that they shall be identical from the correct passwords.

2. TECHNICAL DESCRIPTION

2.1 Literature Survey

2.1.1 Examination of a New Defence Mechanism: Honeywords

Authors: Ziya Alper Genc, Suleyman Kardas, Mehmet Sabir Kiraz
Description: It has become much easier to crack a password hash with the improvement in the graphical processing unit (GPU) technology. An attacker can recover a user's password using brute-force attack on password hash. Once the password has been recovered no server can detect any invalid user authentication. In this context, Juels and Rivest published a paper for improving the security of hashed passwords. They propose a method for user authentication, in which some fake passwords, i.e., "honeywords" are added into a password file, in order to detect enactment their solution includes a secure server called "honeychecker" which can differentiate a user's real password among her honeywords and quickly sets off an alarm whenever a honeyword is used. In this paper, we analyse the security of the proposal, provide some possible improvements which are easy to implement and introduce an improved model as a solution to an open problem.

2.1.2 Investigating the Distribution of Password Choices

Authors: David Malone, Kevin Maher NUI Maynooth
Description: In this paper we will look at the distribution with which passwords are selected. Zipf's Law is commonly observed in lists of selected words. Using password lists from four different online sources, we will scrutinize if Zipf's law is a good candidate for describing the frequency with which passwords are selected. We look at a number of standard statistics, used to estimate the security of password distributions, and see if modelling the data using Zipf's Law produces good evaluation of these statistics. We then look at the analogy of the password distributions from each of our sources, using geuss as a metric. This shows that these distributions provide adequate tools for cracking passwords. Finally, we will show how to frame the distribution of passwords in use, by asking users to choose a different password.

2.1.3 Improving Security Using Deception

Authors: Mohammed Alme shekah, Eugene H. Spafford, Mikhail J. Atallah
Description: As the merge between our physical and digital worlds continues at a rapid step, much of our information is getting available online. In this paper, they developed a novel taxonomy of methods and techniques that can be used to protect digital information. They discuss how information has been secured and show how we can structure our methods to achieve better output. They explore complex relationships among security techniques ranging from denial and isolation, to degradation and confusion, through negative information and deception, ending with attacker attribution and counter-operations. They present analysis of these relationships and discuss how they

can be applied at different place within organizations. They also found some of the areas that are worth further investigation. We map these security techniques against the cyber kill-chain model and discuss some findings.

They identify the use of ambiguous information as a useful security method that can significantly improve the security of systems. They posit how the well-known Kerckhoffs's principle has been misread to drive the security community away from deception based mechanisms. They examine advantages of these techniques can have when securing our information in addition to traditional methods of hiding. They show that by intelligently introducing ambiguous information in information systems, we not only lead attackers awry, but also give organizations the ability to detect leakage of data; create doubt and uncertainty in any leaked data; add risk at the adversaries' side to using the leaked information; and significantly enhance our abilities to attribute attackers. They discuss how to overcome some of the challenges that abstract the adoption of deception-based techniques

2.1.4 Honeywords: Making Password-Cracking Detectable

Authors: Ari Juels, Ronald L. Rivest
Description: They suggested a simple method for enhancing the security of hashed passwords: the maintenance of other honeywords" (decoy passwords) related with each user's account. An attacker who steals a file of hashed passwords and inverts the hash cannot tell if he has found the password. The attempted use of a honeyword for login sets an alarm. An auxiliary server (the honeychecker") can differentiate the user password from honeywords for the login, and will set off an alarm if a honeyword is submitted.

2.1.5 Password Cracking Using Probabilistic Context-Free Grammars

Authors: Matt Weir, Sudhir Aggarwal, Breno de Medeiros, Bill Glodek.
Description: Selecting the most effective word-distorting rules to use when performing a dictionary-based password cracking attack can be a tough task. They discuss a new method that generates password structures in highest probability order. They first automatically create a probabilistic context free grammar depend on a training set of previously disclosed passwords. This grammar then allows us to create word-mangling rules, and from them, password guesses to be used in password cracking. They will also show that this approach seems to provide a more effective way to crack passwords as compared to traditional techniques by testing our tools and techniques on real password sets.

2.2 Problem Statement

Suggest an alternative approach that selects the honeywords from existing user passwords in order to provide realistic honeyword – a perfectly flat honeyword generation method.

2.3 Proposed System

In this study, we focus on the security issue and deal with decoy passwords or accounts as a simple and cost effective solution to detect compromise of passwords. Honeypot is one of the methods to identify occurrence of a password database breach. In this approach, the administrator purposely creates fake user accounts to lure adversaries and

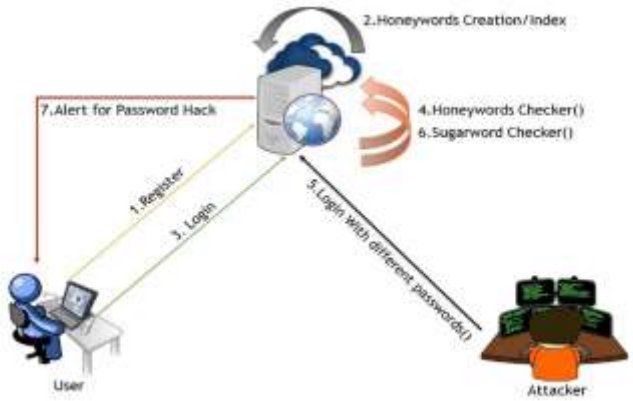

Fig -1: Working of System

Detects a password disclosure, if any one of the honeypot passwords get used. In this paper we have proposed a novel honeyword generation approach which decreases the storage overhead and also it addresses majority of the drawbacks of existing honeyword generation techniques. Proposed model is based on use of honey words to detect password-cracking. We propose to use indexes that map to valid passwords in the system. The contribution of our approach is two-fold. First, this method requires less storage compared to the original study. Within our approach passwords of other users are used as the decoy passwords, so prediction of which password is fake and which is correct becomes more complicated for an adversary.

2.4 Implementation

2.4.1 Introduction

In this project, we separate the honeyword approach and give some notice about the security of the system. We point out that the key for this method is the generation algorithm of the honeywords such that they shall be in-distinguishable from the correct passwords. Therefore, we propose a new method that created the Honeywords using the existing user passwords combination in hash format.

2.4.2 Implementation Details

Algorithm:
Inputs:

1. T fake user accounts (honey pots)
2. index value between [1;N]
3. index list ,which is not previously assign to user

Procedure:
Step 1: Honey pots creation: fake user account

a. For each account honey index set is created like Xi =>(xi;1; xi;2; : : : ; xi;k); one of the elements in Xi is the correct index (sugar index) as Ci
b. Create two password file file f1 and file f2. F1 Store username and honey-index set <hui,xi> Where hui is honey pot account and F2 keeps the index number and the corresponding hash of the password <Ci;H(pi)>

Step 2: Generation of honey-index set

In Step 1 we insert honey index set in file F1 but don't know how to create that We use honey index generator algorithm Generator(k; SI) →Ci;Xi

Generate Xi.

a. select xi randomly selecting k-1 numbers from SI and also randomly picking a number ci SI .

b. Ui; ci pair is delivered to the honey checker and F1, F2 files are updated.

Step 3: Honey checker

Set: Ci, Ui

Sets real password index Ci for the user Ui

Check: Ui, j

Checks whether ci for Ui is equal to given j. Returns the output and if equality does not hold, notifies system a honey word situation.

Step 4: Encryption

- We have a user message (password) space M which contains all possible messages. We outline these messages to a seed space S by using a DTE (distribution-transforming encoder).
- The seed space is the space of all n-bit binary strings for some predetermined n. Each message in m 2 M is mapped to a seed range in S.
- The size of the seed range of m is directly related to how most likely m is in the message space M. We require some knowledge about the message space M in order for the DTE to map messages to seed ranges, specifically the DTE requires the CDF (cumulative distribution function) of M and some information on the ordering of messages.
- Additionally, the seed space S must be large so that even the message with smallest probability in the message space S is assigned at least one seed. With this information, we can find the cumulative probability range corresponding the message m and relate it to the same percentile seed range in Space S.

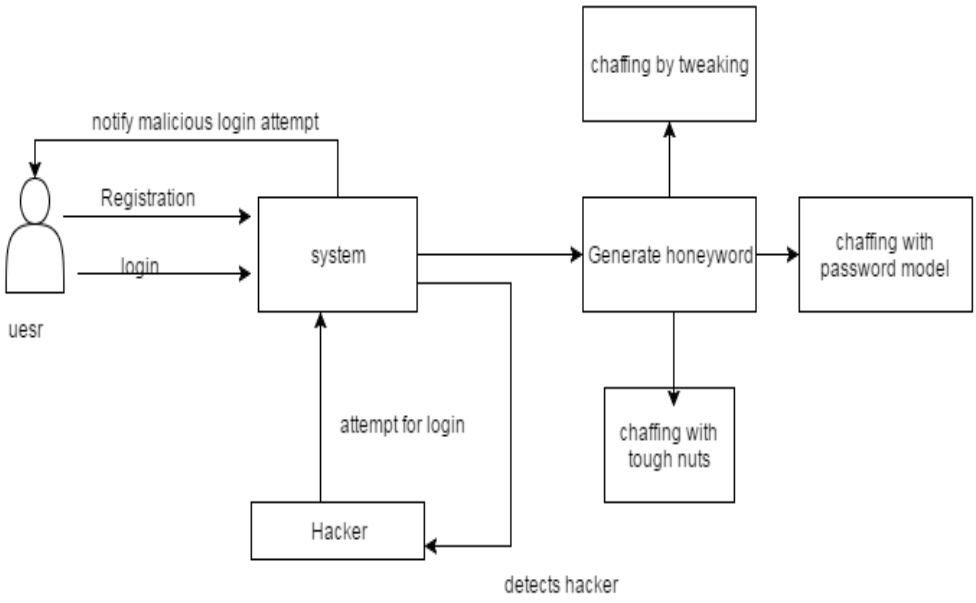

Fig -2: System Architecture

3. Honeyword Generation Methods

3.1 *Chaffing-by-Tweaking*

In this method, the generator algorithm which twist selected character positions of the real Password to produce the honeywords. Each character of a user password in pre-determined positions is replaced by a randomly selected character of the same type: digits by digits, letters are replaced by letters, and special characters by special characters. Number of positions to be twisted, denoted as t should depend on sys policy. As
An example t ¼ 3 and tweaking last t characters may be a method for the generator algorithm Gen(k,t). Another Approach named in the study as "chaffing-by-tweaking-digits" is executed by tweaking the last t positions that contain digits. For example, by using the last technique for the password 42hungry and t ¼ 2, the honeywords 12hungry and 58hungry may be generated.

3.2 *Chaffing-with-a-Password-Model*

In this approach, the generator algorithm takes the password from the user and relying on a probabilistic model of correct passwords it generates the honeywords. In this model, the password is spitted into character sets. For instance, mice3blind is decomposed as four-letters + one-digit + five-letters) L4 þ D1 þ L5 and replaced with the same composition like gold5rings.

3.3 *Chaffing with "Tough Nuts"*

In this method, the system intentionally injects some special honeywords, named as tough nuts, such that inverting hash values of those words is computationally inefficient, e.g. fixed length random bit strings should be set as the hash value of a honeyword. An example for a tough nut is given in as '9,50PEe [KV.0? RIOtcL-:IJ"b+Wol !*]!NWT/pb'. It is stated that the number and positions of tough nuts are selected randomly. By means of this, it is expected that the attacker cannot seize whole sweet-word set and some sweet-words will be blank for her, thereby deterring the adversary to realize her attack. It is discussed that in such a situation the adversary may pause before attempting login with cracked passwords.

3.4 Comparison of Honeyword Generation Models

Method	**DoS Resistance**	**Flatness**	**Storage Cost**
Tweaking	*Weak*	*Weak*	hN*
Password-model	*Strong*	*Strong+*	khN
Our model	*Strong*	*Strong+*	4kN+hN+4N

Table -1: comparison of Honeyword generation Method

4. CONCLUSIONS

We have study carefully the security factors of the honeyword system and introduce a number of defect that need to be fitted with before successful realization of the scheme. In this respect, we have pointed out that the strong point of the honeyword system directly depends on the generation algorithm finally, we have presented a new approach to make the generation algorithm as close as to human nature by generating honeywords with randomly selecting passwords that belong to other users in the system. We present a standard approach to securing personal and business data in the system. We propose monitoring data access patterns by marking user behaviour to determine if and when a malicious insider illegally accesses someone's documents in a system service. Decoy documents stored in the system along with the user's real data also serve as sensors to detect illegitimate access. Once unauthorized data access or exposure is suspected, and later verified, with challenge questions for instance, we barrage the malicious insider with fake information in order to dilute or divert the user' s real data. Such preventive attacks that rely on dis-information technology could provide unusual levels of security in the system and in social networks model. In the future, we would like to refine our model by involving hybrid generation algorithms to also make the total hash inversion process difficult for an adversary in getting the passwords in plaintext form a leaked password hash file. Hence, by developing such methods both of two security goal – increasing the total effort in recovering plaintext passwords from the hashed lists and detecting the password disclosure – can be provided at the same time.

5. ACKNOWLEDGEMENT

The authors would like to thanks Prof. Anand Bone who, guided us through the paper work and such a great Motivation. The authors would also like to thanks Prof. Ganesh Kadam who provided all related help while working on this task.

6. REFERENCES

[1]. D. Mirante & C. Justin, "Understanding passwords databases compromises," Dept. of Computer. Sci. Eng. Polytechnic Inst. of NYU, New York, NY, USA: Tech. Rep. TR-CSE-2013-02, 2013.
[2]. D. Mirante & C. Justin, "Understanding passwords databases compromises," Dept. of Computer. Sci. Eng. Polytechnic Inst. of NYU, New York, NY, USA: Tech. Rep. TR-CSE-2013-02, 2013.
[3]. K. Brown, "The dangers of weak hashes," SANS Institute InfoSec Reading Room, Maryland US, pp. 1–22, Nov. 2013,[Online]. Available: http://www.sans.org/reading-room/ whitepapers/authentication/dangers-weak-hashes-34412
[4]. M. Weir, S. Aggarwal, B. de Medeiros, and B. Glodek, "Password cracking using probabilistic context-free grammars," in 30th IEEE Symp. Security Privacy, 2009, pp. 391–405.
[5]. F. Cohen, "The use of deception techniques: Honeypots and decoys," Handbook Inform. Security, vol. 3, pp. 646–655, 2006.
[6]. M. H. Almeshekah, E. H. Spafford, and M. J. Atallah, "Improving security using deception," Center for Education and Research Information Assurance and Security, Purdue Univ., West Lafayette, IN, USA: Tech. Rep. CERIAS Tech. Rep. 2013-13, 2013.

[7]. C. Herley and D. Florencio, "Protecting financial institutions from brute-force attacks," in Proc. 23rd Int. Inform. Security Conf., 2008, pp. 681–685.
[8]. A. Juels and R. L. Rivest, "Honeywords: Making password cracking detectable," in Proc. ACM SIGSAC Conf. Comput. Commun. Security, 2013, pp. 145–160.
[9]. J. Bonneau, "The science of guessing: Analyzing an anonymized corpus of 70 million passwords," in Proc. IEEE Symp. Security Privacy, 2012, pp. 538–552.
[10]. D. Malone and K. Maher Investigating the distribution of password choices. in Proc. 21st Int. Conf. World Wide Web, 2012,pp. 301–310.
[11]. M. Burnett. The pathetic reality of adobe password hints. [Online]. Available: https://xato.net/windows-security/adobe-passwordhints, 2013.
[12]. H. Bojinov, E. Bursztein, X. Boyen, and D. Boneh, "Kamouflage: Loss-resistant password management," in Proc. 15th Eur. Conf. Res. Comput. Security, 2010, pp. 286–302.

BIOGRAPHIES

	Lalit Hajarimal Suthar slalit360@gmail.com Bachelor of Engineering In Information Technology SKN Sinhgad Institute of Science & Technology, Lonavala, 410401
	Vimal Chetan Singh *vimalsinghkashyapp@gmail.com* Bachelor of Engineering In Information Technology SKN Sinhgad Institute of Science & Technology, Lonavala, 410401
	Mahesh Arun Umale *mahesharunumale@gmail.com* Bachelor of Engineering In Information Technology SKN Sinhgad Institute of Science & Technology, Lonavala, 410401
	Hemalata Manohar Khandagle *hemlata7509@email.com* Bachelor of Engineering In Information Technology SKN Sinhgad Institute of Science & Technology, Lonavala, 410401