As per UGC guidelines an electronic bar code is provided to seure your paper

International Journal for Modern Trends in Science and Technology, 8(08): 212-216, 2022

ISSN: 2455-3778 online
DOI: https://doi.org/10.46501/IJMTST0808030
Available online at: http://www.ijmtst.com/vol8issue08.html

Preventing Password False Detection by Providing Security, using Reliable Honey Words

Dangeti Surya Vamsi, V.Bhaskara Murthy

Dept Of MCA, B.V.Raju College, Bhimavaram
Associate Professor , Dept Of MCA, B.V.Raju College, Bhimavaram

To Cite this Article
Dangeti Surya Vamsi and V.Bhaskara Murthy. Preventing Password False Detection by Providing Security, using Reliable Honey Words. International Journal for Modern Trends in Science and Technology 2022, 8(08), pp. 212-216. https://doi.org/10.46501/IJMTST0808030

Article Info
Received: 22 July 2022; Accepted: 15 August 2022; Published: 21 August 2022.

ABSTRACT

Breach in password databases has been a frequent phenomena in the software industry. Often these breaches go undetected for years. Sometimes, even the companies involved are not aware of the breach. Even after they are detected, publicizing such attacks might not always be in the best interest of the companies. This calls for a strong breach detection mechanism. Juels et al. (in ACM-CCS 2013) suggest a method called 'Honeywords', for detecting password database breaches. Their idea is to generate multiple fake passwords, called honeywords and store them along with the real password. Any login attempt with honeywords isidentified as a compromise of the password database, since legitimate users are not expected to know the honeywords corresponding to their passwords. The key components of their idea are (i) generation of honeywords, (ii) typo-safety measures for preventing false alarms, (iii) alarm policy upon detection, and (iv) testing robustness of the system against various attacks.In this work, we analyze the limitations of existing honeyword generation techniques. We propose a new attack model called 'Multiple System Intersection attack considering Input'. We show that the 'Paired Distance Protocol' proposed by Chakraborty et al., is not secure in this attack model. We also propose new and more practical honeyword generation techniques and call them the 'evolving-password model', the 'user-profile model', and the 'append-secret model'. These techniques achieve 'approximate flatness', implying that the honeywords generated using these techniques are indistinguishable from passwords with high probability. Our proposed techniques overcome most of the risks and limitations associated with existing techniques. We prove flatness of our 'evolving-password model' technique through experimental analysis. We provide a comparison of our proposed models with the existing ones under various attack models to justify our claims.

Breach in password databases has been a frequent phenomena in the software industry. Often these breaches go undetected for years. Sometimes, even the companies involved are not aware of the breach. Even after they are detected, publicizing such attacks might not always be in the best interest of the companies. This calls for a strong breach detection mechanism. Juels et al. (in ACM-CCS 2013) suggest a method called 'Honeywords', for detecting password database breaches. Their idea is to generate multiple fake passwords, called honeywords and store them along with the real password. Any login attempt with honeywords isidentified as a compromise of the password database, since legitimate users are not expected to know the honeywords corresponding to their passwords. The key components of their idea are (i) generation of honeywords, (ii) typo-safety measures for preventing false alarms, (iii) alarm policy upon detection, and (iv) testing robustness of the system against various attacks.In this work, we analyze the limitations of existing honeyword generation techniques. We propose a new attack model called 'Multiple System Intersection attack considering Input'. We show that the 'Paired Distance Protocol' proposed by Chakraborty

et al., is not secure in this attack model. We also propose new and more practical honeyword generation techniques and call them the 'evolving-password model', the 'user-profile model', and the 'append-secret model'. These techniques achieve 'approximate flatness', implying that the honeywords generated using these techniques are indistinguishable from passwords with high probability. Our proposed techniques overcome most of the risks and limitations associated with existing techniques. We prove flatness of our 'evolving-password model' technique through experimental analysis. We provide a comparison of our proposed models with the existing ones under various attack models to justify our claims.

1. INTRODUCTION

Password based authentication is the most widely accepted and cost effective authentication technique. In general practice, passwords are never stored in clear text to ensure confidentiality. Instead they are hashed and then stored along with other user related information. The process of performing a one-way transformation on the password and to obtain another string called the 'hashed' password is known as 'password hashing'. There are several ways to prevent an attacker from performing a dictionary attack by increasing the complexity of this attack manifolds. Making the password hashing algorithm more resource consuming is one way to prevent the adversary from pre-computing the dictionary. This was the main objective behind the Password Hashing Competition (PHC) that ran from 2013-2015. To further improve the security, use of cryptographic module for password hashing is explained in . Another approach is to introduce confusion by adding a list of fake passwords along with the correct password. This would discourage the adversary to mount dictionary attack even after compromising the database. In this technique, the server generates multiple fake passwords called honey words for each user, and stores them along with the actual password chosen by the user. Even if an attacker gets access to the password database, she would not be able to distinguish the actual password from honey words. Therefore with a very high probability, she is expected to enter a honey word to carry out the attack. If a honey word is entered instead of the password, the system raises an alarm, thus detecting the compromise of password database. The efficiency of this system basically depends on the ability of the honey word generation scheme to generate honey words that are indistinguishable from the real password. The authors in, provide some heuristic honey word generation along with detailed analysis of the system implementing the honey words technique. Continuing along the same line of research, we provide an experimental method for quantifying the flatness of honey word generation schemes. We also implement a distancemeasure between password and honey word using 'Levenshteindistance'to avoid false detection when a legitimate user makes a typing error and enters a honey word.

2. EXISTING SYSTEM

There are several ways to prevent an attacker from performing a dictionary attack by increasing the complexity of this attack manifolds. Making the password hashing algorithm more resource consuming is one way to prevent the adversary from precomputing the dictionary. This was the main objective behind the Password Hashing Competition (PHC). To further improve the security, use of cryptographic module for password hashing. Another approach is to introduce confusion by adding a list of fake passwords along with the correct password. This would discourage the adversary to mount dictionary attack even after compromising the database. This approach, of using fake passwords can help in detecting password database breaches. Specifically, any login attempt with one of the fake passwords detects the breach. The idea was influenced from some other existing techniques mentioned below. The honeypot technique, introduced in early 90's, is a system or component which influences the adversary to attack the wrong targets, namely honey pot accounts.

DISADVANTAGES

- Honey pot accounts are fake accounts created by the system administrator to detect password database breaches. Honey token is a honey pot that contains fake entries like social security or credit card numbers to identify malicious activity. Is a theft-resistant password manager that creates multiple decoy password lists along with the correct password list. Frequent cases of password database breaches(like that of LinkedIn in 2012 , Adobe in 2013 , eBay in 2014 , Ashley Madison in 2015 etc.,) are indicative of security issues in the current password based authentication systems which can fail to ensure user privacy. No efficient solution to detect such database breaches had been reported

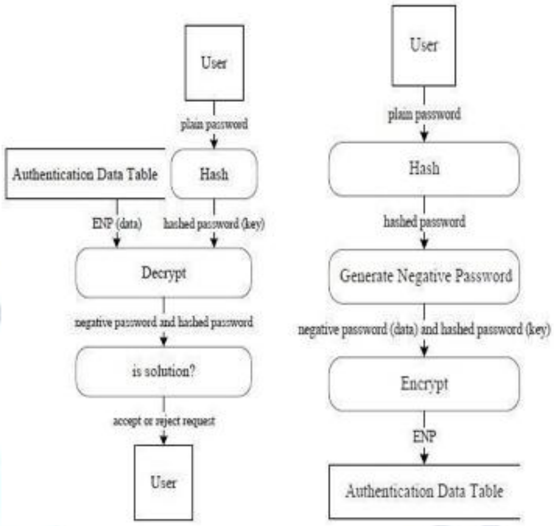
3. PROPOSED SYSTEM

The Honey words technique is a significant contribution towards detecting breaches of the password database. In this technique, the server generates multiple fake passwords called honey words for each user, and stores them along with the actual password chosen by the user. Even if an attacker gets access to the password database, she would not be able to distinguish the actual password from honey words. Therefore with a very high probability, she is expected to enter a honey word to carry out the attack. If a honey word is entered instead of the password, the system raises an alarm, thus detecting the compromise of password database. The efficiency of this system basically depends on the ability of the honey word generation scheme to generate honey words that are indistinguishable from the real password. The authors provide some heuristic honey word generation techniques, along with detailed analysis of the system implementing the honey words technique. Continuing along the same line of research, we provide an experimental method for quantifying the flatness of honey word generation schemes. We also implement a distance-measure between password and honey word using

ADVANTAGES

- By using honeyword, it helps to protect the critical/important personal data of the Govt population Data/Banking data.
- . It provides more security than existing system.
- It protects con dential data from in-sider as well as outsider.
- This honeyword will save million dol-lars of the IT organisation by protect-ing the con dential data from attacker or unauthorized users.

4. ARCHITECTURE DIAGRAM



5. IMPLEMENTATION

Initialization

Firstly, T fake user accounts (honeypots) are created with their passwords. Also an index value between [1;N], but not used previously is assigned to each honeypot randomly. Then k � 1 numbers are randomly selected from the index list and for each account a honey index set is built like Xi = (xi;1; xi;2; : : : ; xi;k); one of the elements in Xi is the correct index (sugarindex) as ci. Now, we use two password _les as F1 and F2 in the main server: F1 stores username and honeyin- dex set, <hui;Xi> pairs as shown in Table 2, where hui denotes a honeypot accounts. On the other hand F2 keeps index number and corresponding hash of password, <ci;H(pi) >, as depicted in Table 3. Let SI denote index column and SH represent the corresponding password hash column of F2. Then the function f(ci) that gives password hash value in SH for the index value ci can be de_ned as: f(ci) = fH(pi) 2 SH :<ci;H(pi) > stored pair of ui and ci 2 SIg.

Registration

After the initialization process, system is ready for user registration. In this phase, a legacy-UI is preferred, i.e. a username and password are required from the user as ui; pi to register the system. We use the honeyindex generator algo-rithmGen(k; SI) ! ci;Xi, which outputs ci as the correct index for ui and the honeyindexes Xi = (xi;1; xi;2; : : : ; xi;k). Note that Gen(k; SI) produces Xi by randomly selecting k �1 numbers from SI and also randomly picking a number ci =2 SI . So ci becomes one of the elements of Xi. One can see that the generator algorithm Gen(k; SI) is di_erent from the procedure described in [9], since it outputs an array of integers rather than a group of honeywords. Note, however, that the index array Xi is indeed represents which honeywords are assigned for ui.

Honeychecker

In our approach, the auxiliary service honeychecker is employed to store correct indexes for each account and we assume that it communicates with the main server through a secure channel in an authenticated manner. Indeed, it can be assumed that security enhancements for honeychecker and the main server presented in [16] are applied, but it is out scope of this study. The role and primary processes of the honeychecker are the same as described in the original study [9], except that <i; ci > pair is replaced with <ui; ci > pair in our case. The honeychecker executes two commands sent by the main server. The honeychecker only knows the correct index for a username, but not the password or hash of the password.

Login Process

System firstly checks whether entered password, g, is correct for the correspond- ing username ui. To do this, the hash values stored in F2 _le for the respective indices in Xi are compared with H(g) to _nd a match. If a match is not obtained, then it means that g is neither the correct password nor one of the honeywords, i.e. login fails. On the other hand, if H(g) is found in the list, then the main server checks whether the account is a honeypot. If it is a honeypot, then it follows a prede_ned security policy against the password disclosure scenario. Notice that for a honeypot account there is no importance of the entered password is genuine or a honeyword, so it directly manages the event without communicating with the honeychecker. If, however, H(g) is in the list and it is not a honeypot, the corresponding j 2 Xi is delivered to honeychecker with username as <ui; j > to verify it is the correct index. Honeychecker controls whether j = ci and returns result to the main server. At the same time if it is not equal then it assured that the proffered password is a honeyword and adequate actions should be taken depending on the policy.

6. CONCLUSION

In this paper, we proposed a password protection scheme called ENP, and presented a password authentication framework based on the ENP. In our framework, the entries in the authentication data table are ENPs. In the end, we analyzed and compared the attack complexity of hashed password, salted password, key stretching and the ENP. The results show that the ENP could resist lookup table attack and provide stronger password protection under dictionary attack. It is worth mentioning that the ENP does not need extra elements (e.g., salt) while resisting lookup table attack.

Conflict of interest statement

Authors declare that they do not have any conflict of interest.

REFERENCES

[1] Sharing in MULTICS. In Proceedings of the Fourth Symposium on Operating System Principles, SOSP 1973, Thomas J. Watson, Research Center, Yorktown Heights, New York, USA, October 15-17, 1973.

[2] Robert Morris and Ken Thompson. Password Security: A Case History, 1979. http://cswww.cs.yale.edu/homes/arvind/cs422/doc/unix-sec.pdf.

[3] S. Philippe Oechslin. Making a faster cryptanalytic time-memory trade-off. In Dan Boneh, editor, Advances in Cryptology – CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings, volume 2729 of Lecture Notes in Computer Science, pages 617–630. Springer, 2003.

[4] Password Hashing Competition (PHC),2014.https://passwordhashing.net/index.html.

[5] Donghoon Chang, Arpan Jati, Sweta Mishra, and Somitra Kumar Sanadhya. Rig: A simple, secure and flexible design for password hashing. In Dongdai Lin, Moti Yung, and Jianying Zhou, editors, Information Security and Cryptology - 10th International Conference, Inscrypt 2014, Beijing, China, December 13-15, 2014, Revised Selected Papers, volume 8957 of Lecture Notes in Computer Science, pages 361–381. Springer, 2014.

[6] Ari Juels and Ronald L. Rivest. Honeywords: making passwordcracking detectable. In 2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4- 8, 2013, 2013.

[7] Fred Cohen. The Use of Deception Techniques: Honeypots and Decoys. http://all.net/journal/deception/Deception Techniques .pdf.

[8] Lance Spitzner. Honeytokens: The Other Honeypot, 2003. http://www.symantec.com/connect/articles/ honeytokens-other-honeypot..

[9] HristoBojinov, Elie Bursztein, Xavier Boyen, and Dan Boneh. Kamouflage: Lossresistant password management. In Computer Security - ESORICS 2010, 15th European Symposium on Research

in Computer Security, Athens, Greece, September 20-22, 2010. Proceedings, pages 286–302, 2010.
[10] Wikipedia contributors. 2012 LinkedIn hack. Wikipedia, The Free Encyclopedia, Date retrieved: 29 May 2016.Availableat:https://en.wikipedia.org/w/index.php?title=2012 LinkedIn hack&oldid=722095159.
[11] Bruce Schneier. Cryptographic Blunders Revealed by Adobe's Password Leak. Schneier on Security, 2013. Available at: https://www.schneier.com/blog/archives/2013/11/ cryptographic b.html.
[12] Swati Khandelwal. Hacking any eBay Account in just 1 minute, 2014. Available at: http://thehackernews.com/2014/09/ hacking-ebay-accounts.html.
[13] Wikipedia contributors. Ashley Madison data breach. Wikipedia, The Free Encyclopedia, Date retrieved: 29 May 2016. Available at: https://en.wikipedia.org/w/index.php?title= Ashley Madison data breach&oldid=721001290.
[14] Troy Hunt. Observations and thoughts on the LinkedIn data breach, 2015. Available at: https://www.troyhunt.com/ observations-and-thoughts-on-the-linkedin-data-breach/.
[15] Michael Gilleland. Levenshtein Distance, in Three Flavors. Available at: http://people.cs.pitt.edu/_kirk/cs1501/assignments/editdistance/Levenshtein%20Distanc e.html.