Network and Complex Systems
ISSN 2224-610X (Paper) ISSN 2225-0603 (Online)
Vol.12, 2021


# Authentication Database Leakage Detection

Asst. Teach. ISRAA S.AHMED
Computer Department, Informatics Institute for Postgraduate Studies(IIPS)/ Iraqi Commission for Computers and Informatics(ICCI)

**Abstract**
Authentication is the first step for any important applications. Password verification is widely employed in the Internet. Some advanced mechanisms including bio features, such as fingerprints or retina. However, digital password authentication credential is commonly used in the public. In other words, traditional password verification is still an important authentication credential mechanism for today's online services. Password database may be compromised. Once if the database is stolen, it is dangerous and critical for any services. Consequently, password database leakage becomes an important issue. Detection and countermeasure are essential for such a disaster. Our research proposing a scheme based on a authentication database storage mapping method to detect possible leakage of authentication database. By multiple mapping mechanism, if the authentication database is stolen, an attacker could not know any user's correct password. This method can reduce additional space of storing passwords with ability to detect security event for stolen authentication databases.
**Keywords:**Application Security; System Security; Leakage Detection; Authentication database, Password
**DOI:** 10.7176/NCS/12-01
**Publication date:** January 31st 2021

## 1 INTRODUCTION

Over the past decades, with the popularity of personal computers and mobile devices, the convenience of the Internet, and the development of social networks, more and more people rely on the Internet to deal with or share things on life, such as online payment, online shopping, etc. The school also provides a variety of online services, such as course enrollment system, online courses, etc., to provide teachers and students a convenient campus life. All of the above are need to use the user authentication to achieve identity confirmation to use the appropriate application.

Password is still the most important credential for today's account certification. Therefore, it plays a very important role in human life. The evolution of the password so far has developed a variety of forms to achieve certification. For example, fingerprints, sound waves, retina and so on. However, the traditional digital password authentication credential is still widely accepted by the public, but in the past few decades, the content of this certification mechanism hasn't been much change.

In recent years, information security issues have gradually been taken seriously, system security has become one of the considerations in the use of services by users. In particular, the password is the first line of defense, but the file leaks are endless. These common password leak events, causing the user's personal information to be stolen. Not only the occurrence of identity theft, but also cause serious property damage. According to Gemalto's 2016 report [1], the most serious is the account access, followed by identity theft.

## 2 RELATED WORK

The concept of Honeyword was first introduced by Juels and Rivest [3]. Their main concept was to design a defense mechanism to detect whether a password file was leaked. In the paper, the author assumes that the attacker can obtain the password file and the ability to convert the password into plaintext. The principle of Honeyword stores the user's real password with the (k-1) fake password generated by the Honeyword generator, so the password file is stored in a format with a user with multiple possible passwords. As a result, even if the password file leaked, the attacker cannot know which one is the user's real password. If he wants to log in as a fake identity, there is a great possibility of entering a fake password, so the system will find the password file may be compromised. At this point, the system can trigger the alarm based on the safety policy set by the administrator, and notifying the administrator to make the corresponding deal. The password file structure is shown in Figure 1.

 

| User Name | Password List |
|---|---|
| $u_1$ | $H(W_{1,1}), H(W_{1,2}), H(W_{1,3})\ldots, H(W_{1,k})$ |
| $u_2$ | $H(W_{2,1}), H(W_{2,2}), H(W_{2,3})\ldots, H(W_{2,k})$ |
| $u_3$ | $H(W_{3,1}), H(W_{3,2}), H(W_{3,3})\ldots, H(W_{3,k})$ |
| … | … |
| $u_i$ | $H(W_{i,1}), H(W_{i,2}), H(W_{i,3})\ldots, H(W_{i,k})$ |

**Fig.1 The password file structure**

The principle of Honeyword is as follows: when a new user registers, he will submit a set of account ui and password pi to the system, then the Honeyword generator Gen (k) will produce a set of fake passwords Wi = ( wi, 1, wi, 2, ..., wi, k,), which contains the correct password for the new user wi, ci = pi. And further for security reasons, the index table c will be stored on a third-party server, independent of the login server. The encrypted password list will be stored as Hi = (vi, 1, vi, 2, ... , vi, k), where j-th will be the real user's password wi, j. In the Honeyword system, a very important part is Honeyword generator. In [3] this paper, the author describes the three ways to generate the password, respectively, as follows:

**Chaffing by tweaking**

A way to replace a selected character position, which must be replaced by the same type of character, such as letter-to-letter, symbol-to-symbol, digit-to-digit, and so on. One of the common practices is "chaffing-by- tail-tweaking", which is a way to change the last few digits to generate Honeywords. For example: password123, if you choose to change the last two digits and generate three Honeywords, the Honeyword generator may generate a password list of W ={password135, password148, password123, password107}.

**Chaffing-with-a-password-model**

This method is using the same syntax module to do the replacement, that is, the group as a unit, such as: letter group, digital group. To the above example: password123, will be divided into W8 | D3, that is, 8 letters

3 digital form. Assuming that you want to generate two Honeywords, it is possible to generate a password list of W = {chaffing345, keyboard987, password123}. This will be more effective than "Chaffing by tweaking" because it makes Honeyword more realistic and closer to the real password, so that the attacker cannot tell the difference between the real password and Honeywords, increase the possibility of entering the fake password.

### 2.3 Chaffing with “tough nuts”

This kind of Honeyword is a way to confuse the attacker, this password will increase the length of the characters or the complexity of characters composition, it can improve the difficulty of password cracking, thereby increasing the attacker's attack time.

There is also a way that Honeyword is produced in a mixed way, such as "chaffing- by- tweaking-digits" with "chaffing-with-a- password-model". In this way, you can first generate a group of models, and then generate b passwords by replacing digits, so that will form k = a x b passwords. The strength of this approach will be stronger than using only one way to generate Honeywords, improving the security of the system.

## 3 RESEARCH METHODS

With the importance of identity authentication to modern networks, the security and reliability of cryptographic authentication is becoming increasingly important. However, the user and the password they use are still the weakest part of the security mechanism since the appearance of password authentication [4]. Therefore, to enhance the security of certification, we propose a password-mapping system in accordance with the concept of Honeyword system [3]. We put the user's password in the system in random order, so that the user password of the mapping account is not the original user set on the surface. This not only retains the detection mechanism of the password file leakage, but also reduces the system storage cost.

When a real user wants to use the system service, they will establish their own account, provide their own user name and password to the system. The system will be based on this name and password to verify the login, and then respond the privilege to the login. As described in Introduction, it is possible to know that an attacker can steal a user's password in different ways, such as sniffing, sending malware, and so on at different layers in the network architecture, and then they can crack the password by brute force, dictionary attack and different password crack tool. So in this case, we assume that the attacker:

 

- Has the ability to obtain the real user- related account password file in the system, and
- Has the ability to get the password in plain text and wants to log in the system.

Our system architecture is mainly composed of three parts, including the user, administrator and database, the system architecture is shown in Figure 2.

The manager section mainly manages the registration of the user and the login authentication, and makes appropriate processing when the login abnormal. To make the password authentication system more secure, the administrator needs to limit the password setting according to the management policy, for example, the minimum characters of the password length, the minimum types of the password characters and so on. The last is the database, which mainly contains two tables. A data table to store the user's account, the hash function after the password, another data table is the password checklist, store the user ID and user's true password index.

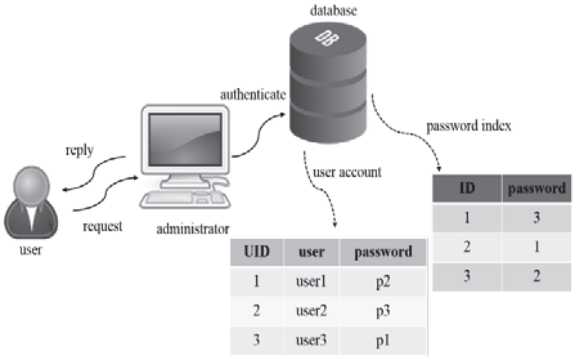


**Fig.2 System Architecture**

## 4 IMPACT EVALUATION

We compare the differences between the traditional system, the Honeyword system, and the system we proposed. As you can see in Table 1, our passwords are stored in the same way as traditional systems, and the Honeyword system stores multiple passwords for a single user. While our system requires additional ID and index values compared to traditional system storage costs, but we reduce the cost of storing additional t passwords for each user compared to the Honeyword system, greatly reducing the system's storage cost.

In the probability of attack success part, assuming that the attacker has obtained the password file, and successfully cracked into a plaintext password, then the probability of attack success is 100% in the traditional system. That is because in the traditional way of password storage, the username maps the real user password. Honeyword system is based on the number of fake passwords and the flatness of fake passwords to determine the probability of attack success.

**Table 1. System Comparison.**

| | Traditional | Honeyword | Mapping-password |
|---|---|---|---|
| *Storage Method* | one to one | one to multiple | one to one |
| *Storage Costs* | $h(p) \times N$ | $(t \times h(p) + c + nL) \times N$ | $(h(p) + 2\times c) \times N$ |
| *Probability of Attack Success* | 100% | 1/t | 1/N |
| *Detection Rate* | 0% | (t-1)/t % | 100% |

The more the fake password, the smaller the probability that each user will be successfully attacked, but it needs extra storage. As for our system, as the number of users rise, the probability of attack success becomes smaller, and the scheme does not need additional storage.

## 5 CONCLUSION

Based on the Honeyword system [3], we propose a password-mapping system to detect leakage of authentication database. Since uses may use repeatedly the same password in different applications. Through experimental results, our proposed scheme shows similar performance to the original one. The scheme can detect the leakage at an acceptable level of overheads. Due to low complexity, our proposed system can effectively improve the security scheme of password authentication, and detect any account leakage events. Furthermore, our proposed scheme can achieve the scalability for its simplicity.

## REFERENCES

[1] Gemalto, 2016 Mining for Database Gold - Findings from the 2016 Breach Level Index, (Gemalto, 2016).

 

[2] The OpenLDAP Foundation, OpenLDAP Software 2.4 Administrator's Guide,( The OpenLDAP Project, 2017).
[3] A. Juels and R. L. Rivest, Honeywords: Making password-cracking detectable, in Proc. 2013 ACM SIGSAC (Berlin, Germany, 2013), pp. 145–160.
[4] S. Ahmed Laskar, "High Capacity data hiding using LSB Steganography and Encryption," *International Journal of Database Management Systems*, vol. 4, pp. 57-68, 2012.
[5] G. Kaur and A. Kochhar, "A steganography implementation based on LSB & DCT," *International Journal for Science and Emerging Technologies with Latest Trends*, vol. 4, pp. 35-41, 2012.
[6] G. Viji and J. Balamurugan, "LSB Steganography in Color and Grayscale Images without using the Transformation," *Bonfring International Journal of Advances in Image Processing*, vol. 1, p. 11, 2011.
[7] G. Satyavathy and M. Punithavalli, "LSB, 3D-DCT and Huffman Encoding based Steganography in