Password Security System With phishing website detection- A Survey

Himanshu Chaudhari[1], Avinash Kumar[2], Akshay Bhandwale[3], Krishna Rathod[4], S.G.Pawar[5]

chaudharihk97@gmail.com,

avinashc9870@gmail.com,

akshaybhandwale@gmail.com,

kr7769014611@gmail.com ,

sgpawar_sits@sinhgad.edu

Abstract

Secure password storage is a very important aspect of systems based on password authentication. There are some security flaws in few authentication techniques. This password can still be cracked easily. Therefore, we are going to develop a system that will provide a way of storing a password by using cryptographic functions. The system will adopt a framework to prevent passwords in the data table.we propose a password authentication framework that is designed for secure password storage and could be easily integrated into existing authentication systems. The system consists of two phases: the registration phase and the authentication phase. In the registration phase, the user enters a username and password. The received password will be converted to hash value by using the Elliptic curve cryptography (ECC) security algorithm. This hash value will be then converted into a negative password using a negative password generation algorithm. The negative password will be then converted into Encrypted Negative Password by using selected symmetric-key algorithm i.e. Attribute-Based Encryption (ABE) and the encryption key will be a hash value of the plain password. This ENP will be then stored in authentication data table.

Keywords: Authentication, dictionary attack, look up table attack, negative database, secure password storage.

I. INTRODUCTION

Nowadays the computer, as well as information security, is the most significant challenge. Authorized users should access the system or information [1]. Authorization can't occur without authentication. For this authentication various techniques are available. Password ensures that computers or information can be accessed by those who have been granted the right to view or access them. The development of the Internet, a vast number of online services have emerged, in which password authentication is the most widely used authentication technique, for it is available at a low cost and easy to deploy. For instance, many users often select weak passwords they tend to reuse the same passwords in different systems they usually set their passwords using familiar vocabulary for its convenience to remember. Also, system problems may cause password compromises [1]. It is very difficult to obtain passwords from high-security systems. After obtaining authentication data tables from weak systems, adversaries can carry out offline attacks. Passwords in the authentication data table are usually in the form of hashed passwords. In this paper, a password protection scheme called Encrypted Negative Password (ENP) is proposed, which is based on the Negative Database (NDB) cryptographic hash function and symmetric encryption, The NDB is a new security technique that is

inspired by biological immune systems and has a wide range of applications. Symmetric encryption is usually deemed inappropriate for password protection. Because the secret key is usually shared by all encrypted passwords and stored together with the authentication data table, once the authentication data table is stolen, the shared Key may be stolen at the same time.if the Secret key is the hash value of the password of each user, so it is almost different and does not need to be specially generated and stored. Consequently, the ENP enables symmetric encryption to be used for password protection. As an implementation of key stretching muti-iteration symmetric encryption feature is introduced to further improve the strength of ENPs [2]. Compared with the salted password scheme and key Stretching, the ENP guarantees the diversity of passwords by itself without introducing extra elements (e.g., salt).
To the main contributions of this paper are as follows:
(1) We propose two implementations of the ENP: ENPI and ENPII, including their generation algorithms and verification algorithms. Furthermore, a password authentication framework based on the ENP is presented [2]
(2) We analyze and compare the attack complexity of hashed password, salted password, key stretching and the ENP. The results show that the ENP could resist lookup table attack without the need for extra elements and provide stronger password protection under dictionary attack.

II. RELATED WORK

A. Typical Password Protection Schemes

1)Hashed Password:
The simplest scheme to store passwords is to directly store plain passwords. However, this scheme presents a problem that once adversaries obtain the authentication data table, all passwords are immediately compromised. To safely store passwords, a common scheme is to hash passwords using a cryptographic hash function because it is infeasible to directly recover plain passwords from hashed passwords. The cryptographic hash function quickly maps data of arbitrary size to a fixed-size sequence of bits. In the authentication system using the hashed password scheme, only hashed passwords are stored. However, hashed passwords cannot resist look-up table attack. Furthermore, a rainbow table attack is more practical for its space-time trade-off [3].
Processor resources and storage resources are becoming richer, which makes the precomputed tables used in the above two attacks sufficiently large so that adversaries could obtain a higher success rate of cracking hashed passwords.
2)Salted Password:
To resist precomputation attacks, the most common scheme is a salted password. In this scheme, the concatenation of a plain password and a random data (called salt) is hashed through a cryptographic hash function. The salt is usually generated at random, which ensures that the hash values of the same plain passwords are almost always different. The greater the size of the salt is, the higher the password security is. However, under the dictionary attack, salted passwords are still weak. Note that compared with a salted password, the ENP proposed in this paper guarantees the diversity of passwords without the need for extra elements (e.g., salt)[3].
3)Key Stretching:
To resist a dictionary attack, key stretching, which converts weak passwords to enhanced passwords, was proposed. Key stretching could increase the time cost required to every password attempt so that the power of defending against a dictionary attack is increased. In the ENP proposed in this paper, like

key stretching, multi-iteration encryption is used to further improve password security under dictionary attack and compared with key stretching; the ENP does not introduce extra elements (e.g., Salt)[3].

B. Negative Database:

Some concepts of NDB are given below. Every entry in an '*'. The symbol '0' only matches the bit 0, and the symbol '1' only matches the bit 1; The symbol '*' can match either the bit 0 or 1. Every entry in an NDB consists of two kinds of positions: specified positions and unspecified positions. Positions, where the symbols are '0' or '1', are called specified positions, while positions, where the symbols are '*', are called unspecified positions. Accordingly, both '0' and '1' are specified symbols, and the '*' is the unspecified symbol. A sequence of bits is covered by one entry in an NDB; that is to say, the bits of the sequence are matched by the symbols of the entry at the specified positions. If a sequence of bits is covered by one entry in an NDB, we say that the sequence is covered by the NDB. If an NDB covers every entry in the (U-DB), we say that the NDB is complete; otherwise, it is incomplete. The NDB converted from a DB with only one entry is called the single NDB; otherwise, it is called the multiple NDB [4].

Paper Title	**Year**	**Objective**	**Limitation**
Intelligent Phishing Website Detection using Random Forest Classifier	2017	In this paper[1], an intelligent system to detect phishing attacks is presented. We used different data mining techniques to decide the categories of websites: legitimate or phishing. Different classifiers were used to construct an accurate intelligent system for phishing website detection.	The limitation of this approach is that the extracted features are only for the detection of phishing WebPages. They achieved 97.5% classification accuracy.
Visual Similarity-based Phishing Detection Scheme using Image and CSS with Target Website Finder	2017	The detection of phishing websites and identifying their target are imperative. In this paper[5], we propose a visual similarity-based phishing detection scheme using image and CSS with a target website finder. To remedy the first shortcoming, we focus on the fact that legitimate websites are often linked by other websites and regard such website as legitimate and store the screenshot and CSS in the database.	As future works, we will blush up the proposed scheme. In the current state, although the websites linked by at least one website are registered in the white list, we consider it is necessary to set an optimal threshold or some conditions. And then, we will try to reduce False Positive by selecting proper CSS.
The Shoulder Surfing Resistant Graphical Password Authentication Technique	2016	A proposed system provides strong security against brute force and guessing attacks as it has a good combination of two types of graphical passwords.[2]	Some limitations of graphical password techniques and the major limitation observed are that it is vulnerable to shoulder

			surfing attacks as images are used as a password.
Detection of Phishing Websites Using C4.5 Data Mining Algorithm	2017	The main aim of phishing is to steal sensitive information of the user such as password, username, PIN, etc.	One of the data mining algorithm C4.5 has been analyzed and the accuracy obtained for the classifier generated by this algorithm is 82.6%. As a data mining algorithm can determine phishing websites in real-time so this accuracy rate of C4.5 is quite significant.

Motivation: We are developing this system for providing more security to sensitive data. We analyze and compare the attack complexity of our scheme with that of typical password storage schemes (i.e., Hashed password, Salted password) under look up table attack and dictionary attack. To provide a way to prevent data tables to keep the user's sensitive information safe from various attacks. We are going to use a combination of various techniques to convert plain text information into encrypted information [4].

We are developing this system for providing more security to sensitive data. We analyze and compare the attack complexity of our scheme with that of typical password storage schemes (i.e., Hashed password, Salted password) under look-up table attack and dictionary attack. To provide a way to prevent data tables to keep the user's sensitive information safe from various attacks. We are going to use a combination of various techniques to convert plain text information into encrypted information [4] [5].

System Architecture:

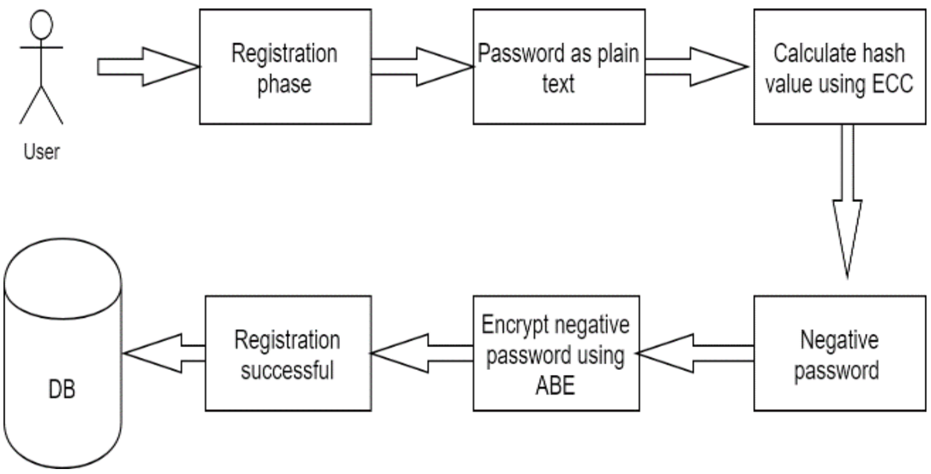

Fig 1. System Overview

The Fig(1) shows the flow of our System. The system consists of two phases: The registration phase and the authentication phase. In the registration phase, the user enters the username and password in plain text. The received password will be converted to hash value by using the ECC algorithm. This hash value will be then converted into a negative password using a negative password generation algorithm. The negative password will be then converted into Encrypted Negative Password (ENP) by using selected symmetric-key algorithm i.e. Attribute-Based Encryption (ABE).and the encryption key will be a hash value of the plain password. This ENP will be stored in the authentication data table. If the user password is matching with Authentication System then Registration will be Successful [5][6].

Implementation of Project

- **We have picked this algorithm from the paper[1],**

1) ECC Algorithm Flow:-

- **Input:** Plain Password.
- **Output:** True or False

Step 1: I/p as a plain password.

Step 2: Convert input plain password into Byte format

```
byte[] I/p1 = I/p.getBytes();
```

Step 3 : Convert input I/p1 into Hex to String

```
pass2=Hex.toHexString(msgHash);
```

Step 4 : Convert input pass2 into 16 byte long

Pseudo Algorithm Flow:-

- **Input:** a hashed password hash;

a negative password np

- **Output**: true or false

```
1: m    LENGTH(hash)
2: for I 1 to m with a step size of 1 do
3: if NUMBEROFSP(npi) 6= i then
4: return false
5: end if
6: end for
7: for i    1 to m with a stepsize of 1 do
```

```
8: if NUMBEROFSP(npi) 6= 1 then
9: return false
10: end if
11: k    INDEXOFSP(npi)
12: x[k]    :TOBIT(npi[k])
13: for j    i + 1 to m with a stepsize of 1 do
14: if npj [k] 6= TOSYMBOL(x[k]) then
15: return false
16: end if
17: npj [k]   '*'
18: end for
19: end for
20: if x = hashP then
21: return true
22: else
23: return false
24: end if
```

- **We have picked this algorithm from the paper[2],**

2) ABE Algorithm Flow :-

- **Input :** Negative Password.
- **Output :** True or False

Step 1 : I/p of Pseudo password.

Step 2 : Convert input of NP password into cipher

keyCph = Bswabe.*enc*(**I/p**, *policy*);

Step 3 :   Convert input I/p1 into Hex to String

cphBuf = SerializeUtils.*bswabeCphSerialize*(keyCph);

Step 3 :   Passed cphBuf   **to AES Coder**

aesBuf = AESCoder.*encrypt*(m.toBytes(), cphBuf);

Mathematical Model:-

ISSN: 2233-7857 IJFGCN

Mathematical model set theory S= {s, e, X, Y, Φ}

s= Start of the program

1. Register/Login into the system

2. Provide Plain Password.

e= End of the program

Identify the Encrypted Password

X= input of the program= {P, R, Q}

P = Plain Password

R= Hash Value

Q = Using ABE Encrypt the hash value

Y = Output of program = Fully Encrypted Password.

First, users provide the specific Plain Password.

Let A be the set of categories

Overall plain password is converted into hash value then encrypted it into By ABE

Y= E1+E2+... +Em / m

Where M is number of overall Encrypted Password.

Conclusion:- In this paper, we propose a secure password authentication framework that is designed for secure password storage and could easily integrate into existing authentication systems. We are also going to propose a password protection scheme called Encrypt Negative Password (i.e., ENP) and records in the authentication data table that are Encrypted Negative Passwords. At the end, we have analyzed and compared the attack complexities of the hashed password, salted password, key stretching, and the ENP. The results show that the ENP technique could resist look up table attacks and provide stronger password protection under the dictionary attack. It is worth mentioning that the ENP does not need extra elements (e.g., salt) while resisting a look up table attack. In the future, other NDB generation algorithms will be studied and will be used to the ENP to further improve password security.

REFERENCES

[1] (J. Bonneau, 2015), "Passwords and the evolution of imperfect authentication," Communications of theACM, vol. 58, no. 7, pp. 78–87, Jun. 2015.
[2] (Waghmare, 2016.) (J. Ma, May 2014) (Herley, 2007)"The shoulder surfing resistant graphical password authentication technique," Procedia Computer
Science, vol. 79, pp. 490–498, 2016.
[3] J. Ma, W. Yang, M. Luo, and N. Li, "A study of probabilistic password models," in Proceedings of 2014 IEEE Symposium on Security and Privacy, May 2014, pp. 689–704.

[4] D. Wang, D. He, H. Cheng, and P. Wang, “fuzzyPSM: A new password strength meter using fuzzy probabilistic context-free grammars,” in Proceedings of 2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, Jun. 2016, pp. 595–606.
[5] Y. Li, H. Wang, and K. Sun, “Personal information in passwords and its security implications,” IEEE Transactions on Information Forensics and Security, vol. 12, no. 10, pp. 2320–2333, Oct. 2017.
[6] D. Florencio and C. Herley, “A large-scale study of web password habits,” in Proceedings of the 16th International Conference on World Wide Web. ACM, 2007, pp. 657–666.