SECURE USER AUTHENTICATION BY ENCRYPTED NEGATIVE PASSWORD

GARAGA BHAVANI [#1], K.RAMBABU [#2]

[#1] MCA Student, Master of Computer Applications,

D.N.R. College, P.G.Courses & Research Center, Bhimavaram, AP, India.

[#2] Head & Assistant Professor, Master of Computer Applications,

D.N.R. College, P.G.Courses & Research Center, Bhimavaram, AP, India.

ABSTRACT

Secure password storage is a one of the main aspect in each and every system during password authentication, which is not at all stored in current systems.Almost all the systems try to encrypt the data which is to be stored into the database but they are not bothered about the password field which is to be stored into the database for user authentication. Hence in this proposed application, we try to apply Encrypted Negative password (ENP) in order to encrypt the plain password and then store the password in a secure manner.The main scope or objective of designing this current application is to apply multi stage security for password authentication by encrypting the given password and also try to apply hash generation technique on that encrypted password and try to generate a negative password which make the intruder so confuse to break the pattern.

Keywords: Encrypted Negative Password, Password Authentication,Encryption, Decryption

1. INTRODUCTION

OWING to 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. Hence, password security always attracts great interest from academia and industry. Despite great research achievements on password security, passwords are still cracked since users' careless behaviors. For instance, many users often select weak passwords they tend to reuse same passwords in different systems. they usually set their passwords using familiar vocabulary for its convenience to remember. In addition, system problems may cause password compromises. It is very difficult to obtain passwords from high security systems.

On the one hand, stealing authentication data tables (containing usernames and passwords) in high security systems is difficult. On the other hand, when carrying out an online guessing attack, there is usually a limit to the number of login attempts. However, passwords may be leaked from weak systems. Vulnerabilities are constantly being discovered, and not all systems could be timely patched to resist an attack, which gives adversaries an opportunity to illegally access weak systems. In fact, some old systems are more vulnerable due to their lack of maintenance. Finally, since passwords are often reused, adversaries may log into high security systems through cracked passwords from systems of low security. 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. However, because processor resources and storage resources are becoming more and more abundant, hashed passwords cannot resist precomputation attacks, such as rainbow table attack and lookup table attack. Note that there is a trend of generalization of adversaries, because anyone could obtain access to information on vulnerabilities from vulnerability databases, such as the Open Source Vulnerability Database (OSVDB), National Vulnerability Database (NVD), and the Common Vulnerabilities and Exposures (CVE) [19], and then make use of these information to crack systems. Moreover, they could download and use attack tools without the need for very professional security knowledge. Some powerful attack tools, such as hashcat [20], RainbowCrack [21] and John the Ripper [22], provide a variety of functions, such as multiple hash algorithms, multiple attack models, multiple operating systems, and multiple platforms, which raises a higher demand for secure password storage.

In these situations, attacks are usually carried out as follows. First, adversaries precompute a lookup table, where the keys are the hash values of elements in a password list containing frequently-used passwords, and the records are the corresponding plain passwords in the password list. Next, they obtain an authentication data table from low security systems. Then, they search for the plain passwords in the lookup table by matching hashed passwords in the authentication data table and the keys in the lookup table. Finally, the adversaries log into higher security systems through cracked usernames and passwords, so that they could steal more sensitive information of users and obtain some other benefits. A considerable number of attacks are carried out in this way, so that adversaries could obtain passwords at a low cost, which is advantageous to their goals. One of the main reasons for the success of the above lookup table attack is that the corresponding hashed password is determined for a given plain password. Therefore, the lookup table could be quickly constructed, and the size of the lookup table could be sufficiently large, which results in a high success rate of cracking hashed passwords.

Typical password protection schemes include hashed pass-word, salted password and key stretching. Among these schemes, hashed password would be gradually eliminated for its vulnerability for precomputation attacks. Although salted password could resist precomputation attacks, it introduces an extra element (i.e., salt) and could not resist dictionary attack. In addition, salt tends to be implemented by mistake (such as salt reuse and short salt). Key stretching schemes, such as bcrypt [23], scrypt [24] and Argon2 [25] (the winner of Password Hashing Competition [26]), are used to defend against dictionary attack. Although key stretching schemes provide stronger password protection than salted password under dictionary attack, they impose an extra burden on programmers for configuring more parameters. In addition, they also use salt to resist precomputation attacks.

Besides these schemes, some other password protection schemes were proposed. In [17], a scheme based on MD5 was proposed. It is a variant of salted password, where the salt is two random strings. Although it could resist lookup table attack and make dictionary attack difficult, it introduces many parameters, which makes it complicated and inconvenient to use. In [27], dynamic salt generation and placement are used to improve password security. Essentially, this scheme is also a variant of salted password, where the salt is a random string that is dependent on the original password. Consequently, it could resist lookup table attack, however it could not defend against dictionary attack and also introduces an extra element (i.e., salt). In [28], improved dynamic Key-Hashed Message Authentication Code function (abbreviated as d-HMAC) was proposed for password storage. It is also a variant of salted password, where the salt is the user's public key, and it introduces a secret key, which makes it inconvenient to use. In summary, although some new password protection schemes were proposed, they are similar to typical password protection schemes essentially. Therefore, in Section VI, without loss of generality, we only compare the typical password schemes with our scheme.

. 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 [37]. Thus, these passwords are immediately compromised. However, in the ENP, the secret key is the hash value of the password of each user, so it is almost always 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 [38], multi-iteration encryption is introduced to further improve the strength of ENPs. 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).

2. LITERATURE SURVEY

Literature survey is the most important step in software development process. Before developing the tool, it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, ten next steps are to determine which operating system and language used for developing the tool. Once the programmers start building the tool, the programmers need lot of external support. This support obtained from senior programmers, from book or from websites. Before building the system the above consideration r taken into for developing the proposed system.

1. Cipher text-Policy Attribute-Based Encryption

AUTHORS: Taeho Jung1, Xiang-Yang Li12, Zhiguo Wan34, Meng Wan5

In several distributed systems a user should only be able to access data if a user posses a certain set of credentials or attributes. Currently, the only method for enforcing such policies is to employ a trusted server to store the data and mediate access control. However, if any server storing the data is compromised, then the confidentiality of the data will be compromised. In this paper we present a system for realizing complex access control on encrypted data that we call Ciphertext-Policy Attribute-Based Encryption. By using our techniques encrypted data can be kept confidential even if the storage server is untrusted; moreover, our methods are secure against collusion attacks. Previous AttributeBased Encryption systems used attributes to describe the encrypted data and built policies into user's keys; while in our system attributes are used to describe a user's credentials, and a party encrypting data determines a policy for who can decrypt. Thus, our methods are conceptually closer to traditional access control methods such as Role-Based Access Control (RBAC). In addition, we provide an implementation of our system and give performance measurements.

2. Multi-authority attribute based encryption with honest-but-curious central authority

AUTHORS: Vladimir Boˇzovi´c1 , Daniel Socek2? , Rainer Steinwandt1 , and Vikt´oria I. Vill´anyi

An attribute based encryption scheme capable of handling multiple authorities was recently proposed by Chase. The scheme is built upon a single-authority attribute based encryption scheme presented earlier by Sahai and Waters. Chase's construction uses a trusted central authority that is inherently capable of decrypting arbitrary ciphertexts created within the system. We present a multi-authority attribute based encryption scheme in which only the set of recipients defined by the encrypting party can decrypt a corresponding ciphertext. The central authority is viewed as "honest-but-curious": on the one hand it honestly follows the protocol, and on the other hand it is curious to decrypt arbitrary ciphertexts thus violating the intent of the encrypting party. The proposed scheme, which like its predecessors relies on the Bilinear DiffieHellman assumption, has a complexity comparable to that of Chase's scheme. We prove that our scheme is secure in the selective ID model and can tolerate an honest-but-curious central authority. Building on the proposal for multi-authority based attribute based encryption from [4], we constructed a scheme where the central authority is no longer capable of decrypting arbitrary ciphertexts created within the system. In addition to showing security in the selective ID model, we showed that the proposed system can tolerate an honest-but-curious central authority. Since both Chase's scheme and the proposed scheme rely on the same hardness assumption, and have a comparable complexity, the new scheme seems a viable alternative to Chase's construction. However, since only the proposed method is capable of handling a curious yet honest central authority, the proposed scheme is recommended in applications where security against such a central authority is required.

3. Decentralizing Attribute-Based Encryption

AUTHORS: Allison Lewko ∗ University of Texas at Austin alewko@cs.utexas.edu

We propose a Multi-Authority Attribute-Based Encryption (ABE) system. In our system, any party can become an authority and there is no requirement for any global coordination other than the creation of an initial set of common reference parameters. A party can simply act as an ABE authority by creating a public key and issuing private keys to different users that reflect their attributes. A user can encrypt data in terms of any boolean formula over attributes issued from any chosen set of authorities. Finally, our system does not require any central authority. In constructing our system, our largest technical hurdle is to make it collusion resistant. Prior Attribute-Based Encryption systems achieved collusion resistance when the ABE system authority "tied" together different components (representing different attributes) of a user's private key by randomizing the key. However, in our system each component will come from a potentially different authority, where we assume no coordination between such authorities. We create new techniques to tie key components together and prevent collusion attacks between users with different global identifiers. We prove our system secure using the recent dual system encryption methodology where the security proof works by first converting the challenge ciphertext and private keys to a semi-functional form and then arguing security. We follow a recent variant of the dual system proof technique due to Lewko and Waters and build our system using bilinear groups of composite order. We prove security under similar static assumptions to the LW paper in the random oracle model.

4. Accountable Authority Ciphertext-Policy Attribute-Based Encryption with White-Box Traceability and Public Auditing in the Cloud

AUTHORS: Jianting Ning1 , Xiaolei Dong2 , Zhenfu Cao2 and Lifei Wei3

As a sophisticated mechanism for secure fine-grained access control, ciphertext-policy attribute-based encryption (CP-ABE) is a highly promising solution for commercial applications such as cloud computing. However, there still exists one major issue awaiting to be solved, that is, the prevention of key abuse. Most of the existing CP-ABE systems missed this critical functionality, hindering the wide utilization and commercial application of CP-ABE systems to date. In this paper, we address two practical problems about the key abuse of CP-ABE: (1) The key escrow problem of the semi-trusted authority; and, (2) The malicious key delegation problem of the users. For the semi-trusted authority, its misbehavior (i.e., illegal key (re-)distribution) should be caught and prosecuted. And for a user, his/her malicious behavior (i.e., illegal key sharing) need be traced. We affirmatively solve these two key abuse problems by proposing the first accountable authority CP-ABE with whitebox traceability that supports policies expressed in any monotone access structures. Moreover, we provide an auditor to judge publicly whether a suspected user is guilty or is framed by the authority. In this work, we addressed two practical problems about the key abuse of CPABE in the cloud, and have presented an accountable authority CP-ABE system supporting white-box traceability and public auditing. Specifically, the proposed system could trace the malicious users for illegal key sharing. And for the semitrusted authority, its illegal key (re-)distributing misbehavior could be caught and prosecuted. Furthermore, we have provided an auditor to judge whether a malicious user is innocent or framed by the authority. As far as we known, this is the first CP-ABE system that simultaneously supports white-box traceability, accountable authority and public auditing. We have also proved that the new system is fully secure in the standard model. Note that there exists a stronger notion for traceability called black-box traceability. In black-box scenario, the malicious user could hide the decryption algorithm by tweaking it, as well as the decryption key. And in this case, the proposed system with white-box traceability in this paper will fail since both the decryption key and decryption algorithm are not well-formed. In our future work, we will focus on constructing an accountable authority CP-ABE system which is black-box traceability and public auditing.

3. EXISTING SYSTEM

In the existing system all the authentications used to be done on data by using several primitive cryptography techniques such as public key cryptography, private key cryptography and secret key cryptography. But no system is concentrating more on the password which is used as one primary factor for user authentication. In general the users choose remember able passwords for entering into the account and these passwords can be easily hacked by the intruders by using ethical hacking software's such as key loggers and so on.Hence the user authentication is becoming less secure..

LIMITATION OF EXISTING SYSTEM

The following are the limitation of existing system. They is as follows:

1. In the primitive password based authentication system,the user choose a valid username and password and these details are stored as it is in the database,so there is a chance of obtaining these credentails easily by the hackers.
2. There is no security for the password attribute which is stored as it is in plain text manner.
3. All the primitive mechanism try to store passwords under hidden manner in GUI, but they cant give more security against the key loggers who try to gain illegal access.

   In the existing system we try to provide security for the data in terms of encrypting the data using secure cryptography algorithms but there is no security technique to provide additional security for the password field.

4. PROPOSED SYSTEM

In this proposed work, we propose a password authentication framework that is designed for secure password storage and could be easily integrated into existing authentication systems. The current application is to apply multi stage security for password authentication by encrypting the given password and also try to apply hash generation technique on that encrypted password and try to generate a negative password which make the intruder so confuse to break the pattern. By applying negative password technique for the password field there is very less chance for the attacker to crack the password to enter illegally into others account.

ADVANTAGES OF THE PROPOSED SYSTEM

The following are the advantages of the proposed system:

1. The current system uses Negative password technique as a main source for authentication.
2. The current system tries to provide utmost security for the password credentials which is stored into the database.

The current system can be accessed only by the valid users and if any un-authorized user gains plain password and username, he cannot login into the account without having negative password value

5. SOFTWARE PROJECT MODULES

Implementation is the stage where the theoretical design is converted into programmatically manner. In this stage we will divide the application into a number of modules and then coded for deployment. We have implemented the proposed concept on Java programming language with JEE as the chosen language in order to show the performance this proposed protocol. The application is divided into 4 modules:

1. Registration Phase
2. Authentication Phase
3. Encrypted Negative Password (ENP)
4. NDB generation algorithm

Now let us discuss about each and every module in detail as follows:

5.1 REGISTRATION PHASE MODULE

1. On the client side, a user enters his/her username and password. Then, the username and plain password are transmitted to the server through a secure channel;

2. If the received username exists in the authentication data table, “The username already exists!” is returned, which means that the server has rejected the registration request, and the registration phase is terminated; otherwise, go to Step (3);
3. The received password is hashed using the selected cryptographic hash function;
4. The hashed password is converted into a negative password using an NDB generation algorithm (i.e., Algorithm A.1 or Algorithm A.2 in the Appendix);
5. The negative password is encrypted to an ENP using the selected symmetric-key algorithm, where the key is the hash value of the plain password. Here, as an additional option, multi-iteration encryption could be used to further enhance passwords;
6. The username and the resulting ENP are stored in the authentication data table and “Registration success” is returned, which means that the server has accepted the registration request.

5.2 AUTHENTICATION MODULE

- Authority will upload the file in cloud. And uploaded file will store in drive HQ in encrypted format.
- Authority will give secret key for all files when user request for any file and the secret key will be send to corresponding user mail Id.
- He is the one who mainly provide security of the data which is stored into the live cloud server.

5.3 TRUSTEE MODULE

- It acts as admin for cloud server.
- Trustee will give request for all files security response when user request for any file.
- He is one who is responsible for Granting Permissions for the Users during Login
- He is mainly responsible for user authentication.

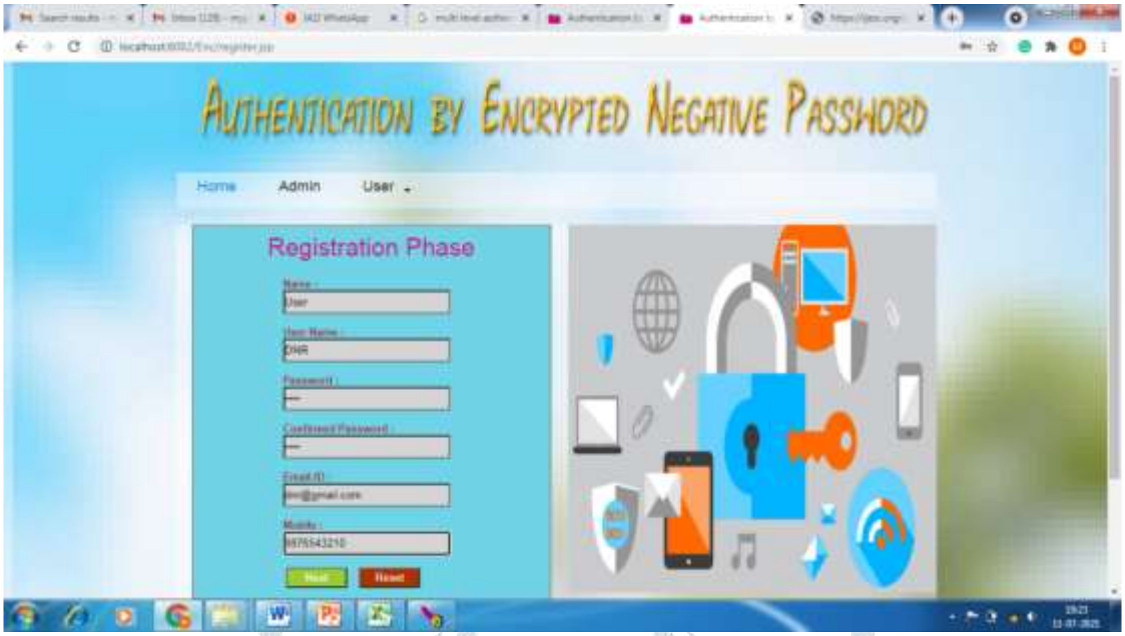
5.4 CLOUD SERVER MODULE

- Cloud view uploaded files in cloud.
- Cloud view Downloaded files by user in cloud.
- This is one which can accept all the data from authority and store the data in encrypted manner.

In this application we try to use this cloud as live cloud server and try to store the data under DRIVEHQ public cloud server

6. EXPERIMENTAL RESULTS

User Registration Page



AUTHENTICATION BY ENCRYPTED NEGATIVE PASSWORD

Home Admin User

Registration Phase

Name : User

User Name : DNR

Password : ----

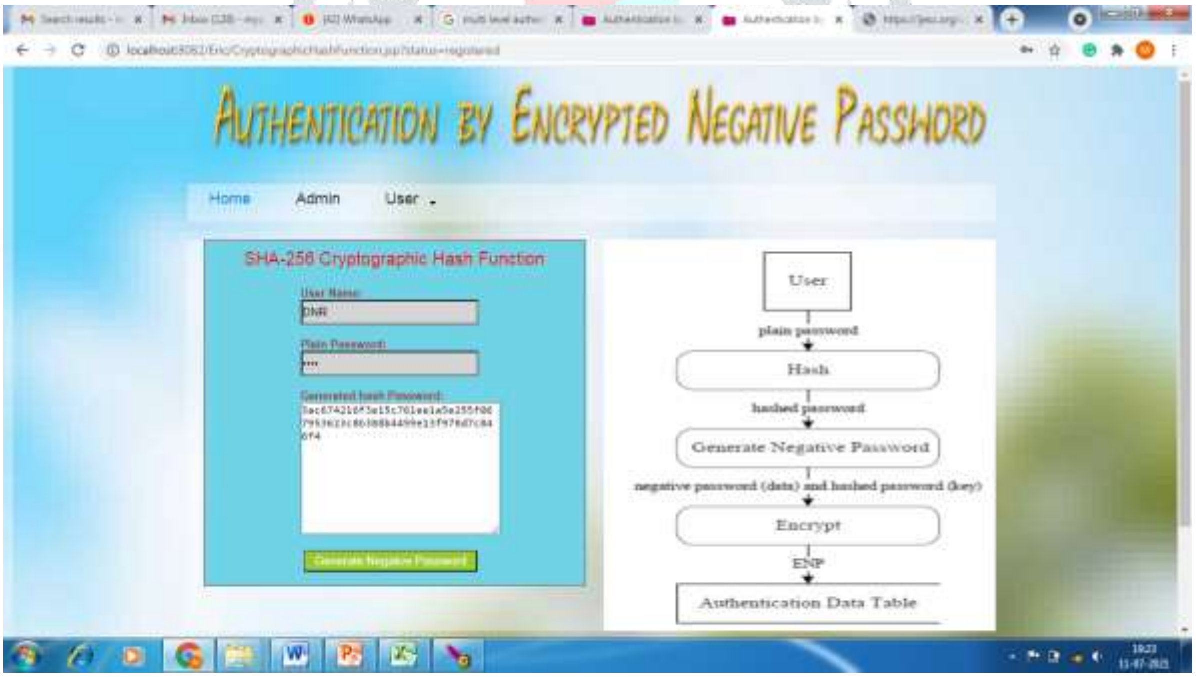
Confirmed Password : ----

Email ID : dnr@gmail.com

Mobile : 9876543210

Next Reset

Customer Registration



AUTHENTICATION BY ENCRYPTED NEGATIVE PASSWORD

Home Admin User

SHA-256 Cryptographic Hash Function

User Name: DNR

Plain Password: ****

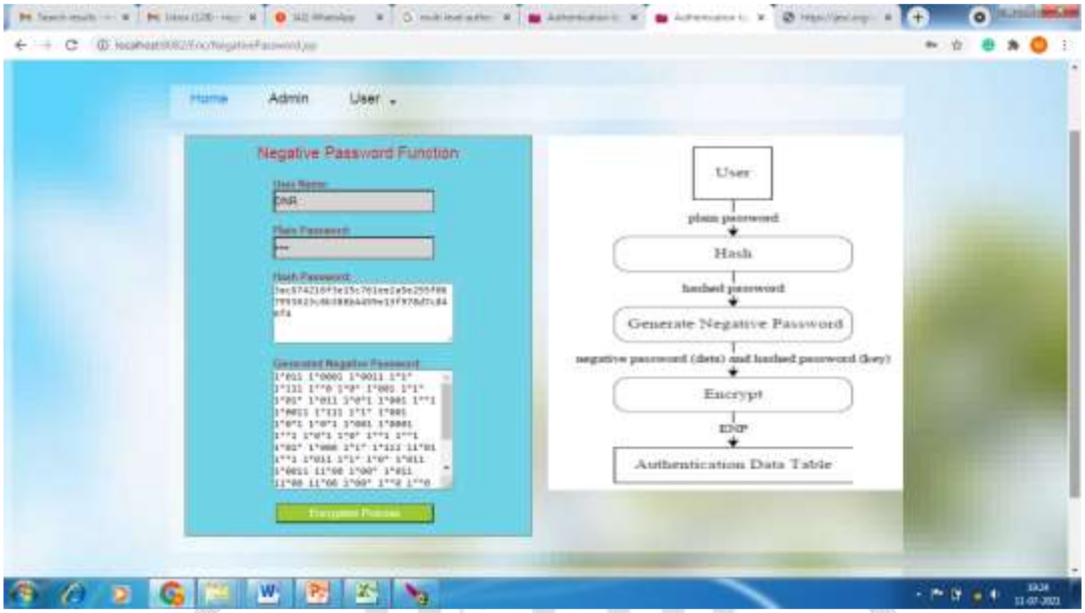
Generated hash Password:

```
3ac674216f3e15c761ee1a5e255f06
7953623c8b388b4459e13f978d7c84
6f4
```

Generate Negative Password

User
plain password
Hash
hashed password
Generate Negative Password
negative password (data) and hashed password (key)
Encrypt
ENP
Authentication Data Table

USER GENERATE NEGATIVE PASSWORD



Negative Password Function

User Name: DNR

Plain Password: ***

Hash Password: [illegible]

Generated Negative Password: [illegible]

User

plain password

Hash

hashed password

Generate Negative Password

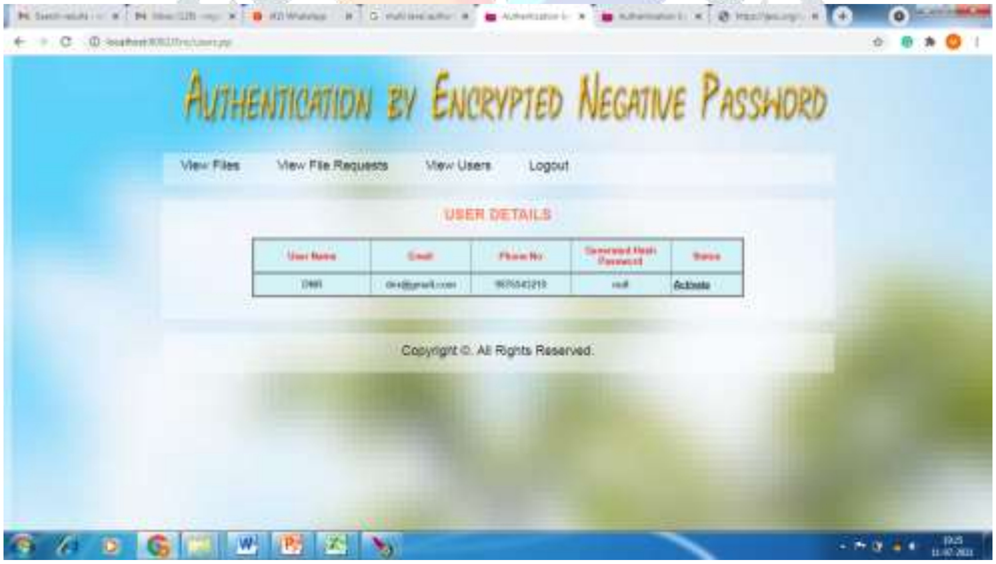
negative password (data) and hashed password (key)

Encrypt

ENP

Authentication Data Table

ADMIN VIEWS THE NEW USER



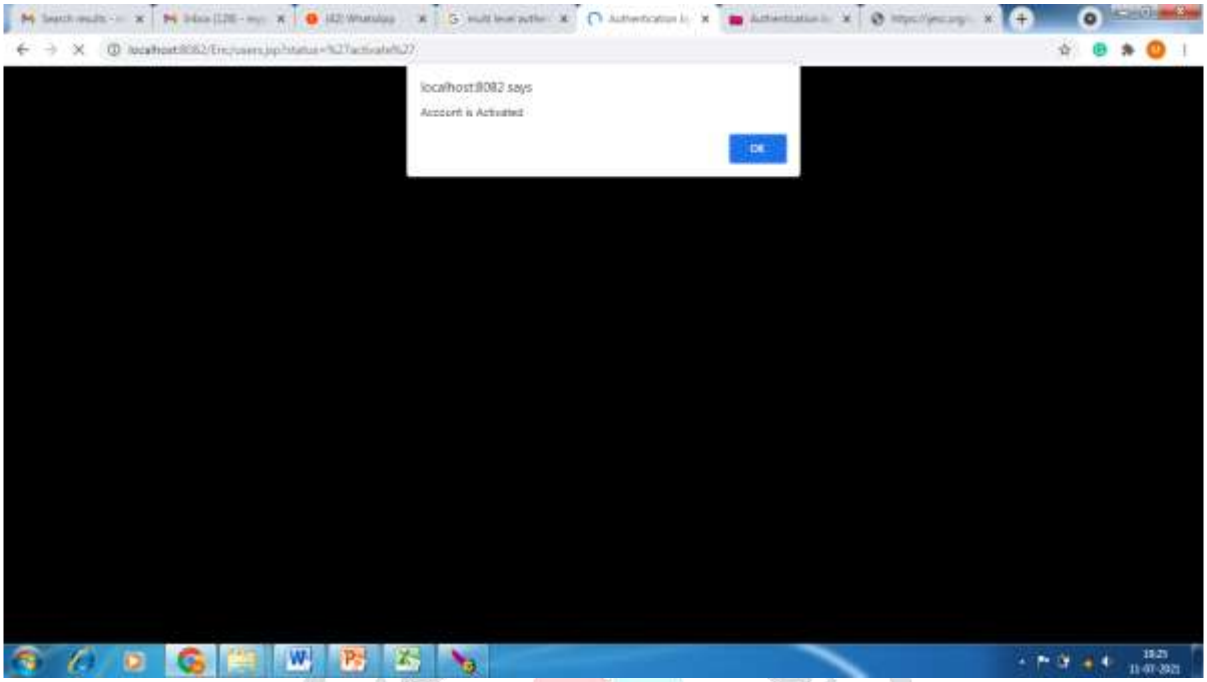
AUTHENTICATION BY ENCRYPTED NEGATIVE PASSWORD

USER DETAILS

User Name	Email	Phone No	Generated Hash Password	Status
DNR	dnr@gmail.com	[illegible]	null	Activate

Copyright ©. All Rights Reserved.

USER IS ACTIVATED BY ADMIN



localhost:8082/Erc/users.jsp?status=%27activate%27

localhost:8082 says

Account is Activated

7. CONCLUSION

In this PROJECT, 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.

8. REFERENCES

[1] J. Bonneau, C. Herley, P. C. van Oorschot, and F. Stajano, “Passwords and the evolution of imperfect authentication,” Communications of the ACM, vol. 58, no. 7, pp. 78–87, Jun. 2015.

[2] M. A. S. Gokhale and V. S. Waghmare, “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] A. Adams and M. A. Sasse, "Users are not the enemy," Communications of the ACM, vol. 42, no. 12, pp. 40–46, Dec. 1999.

[5] E. H. Spafford, "Opus: Preventing weak password choices," Computers & Security, vol. 11, no. 3, pp. 273–278, 1992.

[6] 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.

[7] 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.

[8] R. Shay, S. Komanduri, A. L. Durity, P. S. Huh, M. L. Mazurek, S. M. Segreti, B. Ur, L. Bauer, N. Christin, and L. F. Cranor, "Designing password policies for strength and usability," ACM Transactions on Information and System Security, vol. 18, no. 4, pp. 13:1–13:34, May 2016.

[9] 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.

[10] H. M. Sun, Y. H. Chen, and Y. H. Lin, "oPass: A user authentication protocol resistant to password stealing and password reuse attacks," IEEE Transactions on Information Forensics and Security, vol. 7, no. 2, pp. 651–663, Apr. 2012.
[11] M. Zviran and W. J. Haga, "Password security: An empirical study," Journal of Management Information Systems, vol. 15, no. 4, pp. 161– 185, 1999.

[12] P. Andriotis, T. Tryfonas, and G. Oikonomou, "Complexity metrics and user strength perceptions of the pattern-lock graphical authentication method," in Proceedings of Human Aspects of Information Security, Privacy, and Trust. Springer International Publishing, 2014, pp. 115– 126.

[13] D. P. Jablon, "Strong password-only authenticated key exchange," SIGCOMM Computer Communication Review, vol. 26, no. 5, pp. 5–26, Oct. 1996.

[14] J. Jose, T. T. Tomy, V. Karunakaran, A. K. V, A. Varkey, and N. C. A., "Securing passwords from dictionary attack with character-tree," in Proceedings of 2016 International Conference on Wireless Communications, Signal Processing and Networking, Mar. 2016, pp. 2301–2307.

[15] A. Arora, A. Nandkumar, and R. Telang, “Does information security attack frequency increase with vulnerability disclosure? an empirical analysis,” Information Systems Frontiers, vol. 8, no. 5, pp. 350–362, Dec. 2006.

[16] R. Song, “Advanced smart card based password authentication protocol,” Computer Standards & Interfaces, vol. 32, no. 5, pp. 321–325, 2010.

[17] M. C. Ah Kioon, Z. S. Wang, and S. Deb Das, “Security analysis of MD5 algorithm in password storage,” in Proceedings of Instruments, Measurement, Electronics and Information Engineering. Trans Tech Publications, Oct. 2013, pp. 2706–2711.