International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 05, May-2015

Password Protection Enhancement using Fuzzy Membership and PRNG

Anupam Singh
Department of Computer Science & Engineering
Echelon Institute of Technology,
Faridabad, Haryana, India

Dr. Neeta Wadhwa
Department of Computer Science & Engineering
Echelon Institute of Technology,
Faridabad, Haryana, India

***Abstract*—Password security of user accounts is the major concern in today's fast moving and advanced computing world. Today the traditional methods for providing security are not much sufficient to protect the data or its authentication as today computing processors like i7 runs with 300,000 MIPS approximately which is now the plus point for the hackers to attack. This paper describes the password protection method for advanced security using fuzzy functions and pseudorandom number generator. It removes the restriction of the limited length password or the password with specific protocols like using only alphabet, alphanumeric or limited symbols. The described method is fit for all the ASCII values from 0-255 and user can enter password with his own comfort.**

Keywords—Fuzzy membership, password protection, random number generation, hash function, ASCII code

I. INTRODUCTION

The password in any domain whether it's offline or online is the best way to protect the data and used for the identification of the authorized user. If the user entity provided the valid and true password value it will considered as the real or the authenticated user otherwise the password protected system denial the retrieval or the access of the data or protected application. But as now the most of the businesses, the online banking transactions, and the profiles on the different social networking sites are not safe from the reach of attackers. Attackers use different techniques to break the security of the passwords like using Trojan programs that share files via instant messenger, Phishing, Information Brokers, Internet Public Records, Trojan Horses, Wormhole Attack[2] brute force attack, guessing attack, statistical attack and many more.

Despite of having various attacks there are also different measures to counter these attacks like storing of the message digest i.e. hash function for password instead of the original password on the server side and there are lots of techniques like MD5 and SHA1, where now the traditional ones are now not that safe from the dictionary attack.

To prevent the security of the password traditionally there is way in which we put one guard which gives exponential complexity to break the password using brute force attack [3][4]but if we increase the guard from one to two and further it increase the complexity of password breaking in multiple exponential like one says to waste the computing cycles in style of useful password hash [5] can be fruitful for the password security.

The present work proposed the fuzzy membership [1] function and the random number generator to increase extra layer of security and this new method is out of any limitations of password length or password validation.

II. PSEUDO RANDOM NUMBER GENERATORS

As pseudo refers false or fake the pseudo random number generators (PRNG) are the deterministic programs or algorithm to generate the arbitrary numbers. The PRNG are not the complete random generator as any of the time it will regenerates the same sequence of values. The reason of having PRNG comes into existence because the True Random number generators (TRNG) are almost impossible to implement in computer world and if we required some number or sequence to repeat we can't generate it using the TRNG , still there are various methods which can generate the true random numbers like flipping of coins , through recording the noise and etc. and the PRNG are generates with the self-feedback loop with different level of entropy which determined the randomness of the values generated.

As the PRNG can also be reproduced it's also have the big significance in the field of cryptography and especially in hash function generation, which is applied in the described method.

III. RELATED WORK

Fuzzy logic is the vastsubject to discuss and it has many applications like in artificial intelligence, robotics, and cryptography for example hash generation [1] for the password protection. Fuzzy systems are suitable for assistance in complex levels of system control ('decision making') [6], and not only as an alternative to controllers on lower levels, such as speed and position controllers [7].

Further it has been found from the researches that if we bound the user to create passwords for the their respective accounts with using some pre-defined constraints like at-least one symbol, one capital letter and one number it becomes difficult for the user to create one lengthy password which

results in the limited length password for the ease of remembrance [8] and if user create some long passwords than those are quite easy to break as they are some phrases, liners [9] or in many cases numbers in series and mobile numbers.

By getting some underlined points from the research's that it's not the waste of resources and the time if we iterated the hash functions [5] and different computing methods forthe sake of complexity incrementto harden the crypt analyser work to break the password.

In this regard the following paper describes the method with mixing of fuzzy membership, PRNG and the iteration of the above two techniques to enhance the password security and with no password length bound.

IV. PROPOSED METHOD

In this work first there is theoperation on individual symbols from the password string where these symbols are converted it into the ASCIIvalue (considering them as integer). Than in second part the values are passed from thefuzzy member function and then from the PRNG function and lastly iterates the cycle for the predefined numbers of times, after getting random hashvaluewhich is one way generated variable sized values, we concatenated all in sequence by iteration function and last converted it into fixed hash value using SHA-512[10].

Normally in general process in the password hashing methods [1] when the password is generated, it get digest and stored into the password records on the server side, when the user sends the user name and the respective password, the system compares it with the message digest value with the recorded one in the server database and if there is a match access granted otherwise the request for grant is dropped as mentioned in the below fig. 1.

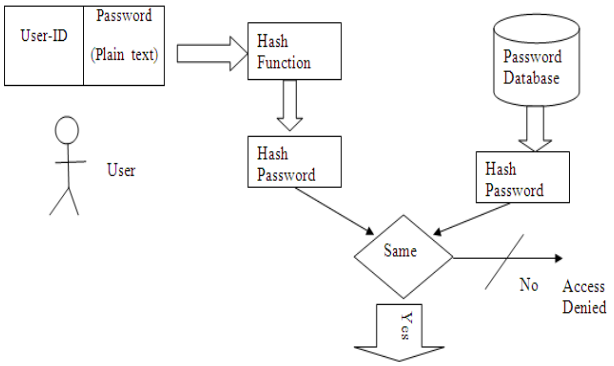

Fig. 1. General architecture for hashing of the password [1]

The following above structure had some vulnerability as hash value prevent the eve to get access to the account besides of having the password records [1] but if the password length is small like 5 digit value and eve can easily apply all the combinations called the brute force attack and matches all self-generated hash values with the recorded ones offline and where it gets the match and finds its corresponding user ID and finally the ID is compromised.

Wherein the proposed architecture as mentioned in the fig.2 below password first converted into ASCII code and then get processed through described method and lastly before getting saved on the server side get converted in to the hash function i.e. SHA 512.

By applying this method we can easily fool the eve as if eve get length of the string for password and try all possible values and converted them to hash function and tries to match them with the recorded one the server side will not get the true possible value as the hash actually was of the processed value and not of the actual password string.

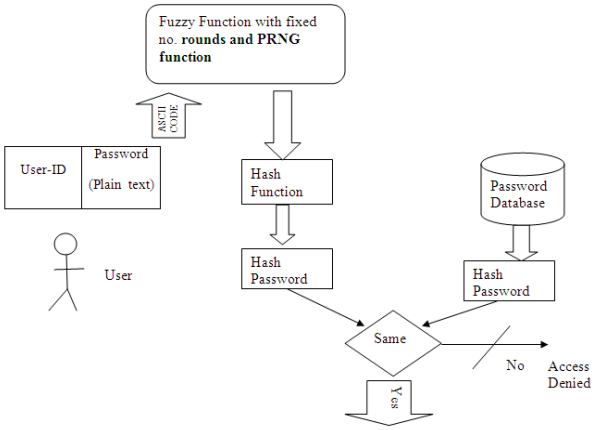

Fig. 2. Proposed architecture of the password hashing

V. METHDOLOGY

In this part there is a design of fuzzy function and its corresponding PRNG. The fuzzy function and the respective PRNG are designed in java.

A. Proposed function

For the implementation the following algorithm is applied as described below

Algorithm First

Step 1: get the String (i.e. password) from user.

Step 2: get the length of the string i.e. len=Length of the string.

Step 3: store all characters in the character array list.

Step 4: for each character for i=1 to len do// start of loop

Step 5: convert char value into ASCII value.

Step 6: put the ASCII value in fuzzy member decision function (refer second algorithm below)

Step 7: operate the fuzzy value with random value (refer third algorithm below) i.e. fuzzy value= fuzzy value+ (i * random value)

Step 8: take the mode of fuzzy value to 256 (key board has total 256 ascii values only)
Fuzzy value= (fuzzy value) mod 256

Step 9: convert fuzzy value to character // end of loop

Step 10: append the characters into String/word

Step 11: append the result string into last string

Step 12: repeat step 2 to step 11 for fixed no. of rounds (in implementation we used 2 for ease of explanation)
// end of fuzzy function

B. Fuzzy Function

This is the part where the value enters into the fuzzy member function and and get one of the fuzzy value from the function. The algorithm is as follows:

Algorithm Second

Step 1: get the ascii value i.e. arg=ascii value.

Step 2: put the value in fuzzy membership block (see Fig 3.3)

Step 3: Compare the value arg

```
if(arg>=0&&arg<32)
{
x1=arg*m;
x2=(31-arg)*m;
}
else{
if(arg>=32&&arg<64)
{
x1=(arg-32)*m;
x2=(63-arg)*m;
}
else{
if(arg>=64&&arg<96)
{
 x1=(arg-64)*m;
 x2=(95-arg)*m;
 }
else{
if(arg>=96&&arg<128)
{
 x1=(arg-96)*m;
x2=(127-arg)*m;
}
 else{
 if(arg>=128&&arg<160)
{
x1=(arg-128)*m;
x2=(159-arg)*m;
 }
 else{
 if(arg>=160&&arg<192)
 {
 x1=(arg-160)*m;
 x2=(191-arg)*m;
    }
else{
if(arg>=192&&arg<224)
 {
  x1=(arg-192)*m;
  x2=(223-arg)*m;
  }
  else{
```

```
if(arg>=224&&arg<256)
 {
  x1=(arg-224)*m;
 x2=(255-arg)*m;
   }
```

Step 4: in this step return the value as x=(x1*x2)mod256

Here m = 3125 i.e. the interval in the membership function
That is 32 and when x1 or the x2 is multiplied by the 1/32 that is the value 0.03125. Here as considered the value taken as whole number and ignored the decimal symbol.
The following figure 3.is the representation of membership function:

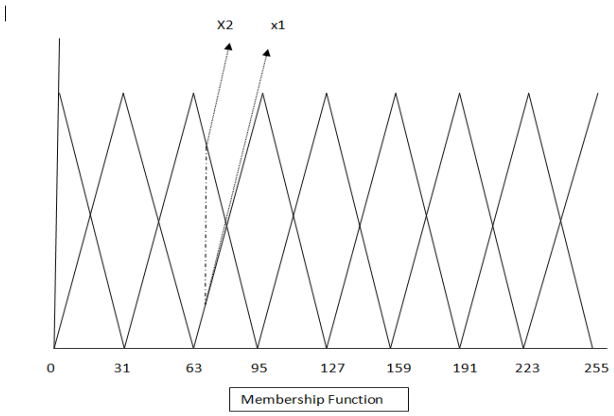

Fig 3. Membership function

C. *PRNG*

For the PRNG we used the concept of centered Hexagonal number because in this we always get the prime values (almost). Now the steps to use properties in the implemented PRNG are as follows:

Algorithm Third

//For random Value generation

Step 1: get the length of the string len=String length

Step 2: value=$len^3 - (len - 1)^3$

Step 3: value=value -1;

Step 4: return value
// here the properties of the factorization problem one way function comes in to play that's why the number converted into even number

Finally In the next step we used the SHA-512[10] function to get the fixed length hash value which gives 128 hexadecimal value fixed output value which is still prevent from the attacker.

VI. RESULT SETS

The implementation was successful and the method is implemented in java language in net beans IDE. Where the mentioned fig.4 below is the snapshot for the value entering of user ID and the password

Fig4. Screen shot for the user login page

The following snap shot is the output part where we display the hash value generated by our method and the final SHA-512 generated fixed length value.

Fig 5. Screen shots for the hash display of fixed SHA-512 bits and variable length

VII. CONCLUSION

This method is the advancement in the password protection but we knows the none of the methods are fully protected this will also secured for the current time and for short future and if eve equipped with the more resources the following method will also become vulnerable to the eve attack, by this method we shows that the iteration methods and the any suitable PRNG or any mathematical one way formula can be fruitful for the security of password as it gives headache to the eve to crack the password in less than the brute force attack time and if he get succeed on one gate he has to pass the second gate which will comes into extra cost for the eve to break the password , further it frees the barrier

of fixed length password or restriction on the creation of the password as it doesn't guarantees that the long passwords are safe enough[8], and here we also gives freedom to the user to use any of the ASCII value to create his password and to become more secure.

REFERENCES

[1] Seyed Hasan Mortazavi Zarch, Hussein, and MadiheSadat"Enhance the security of password by fuzzy controller", publication : 978-1-4799-3351-8/14/$31.00 ©2014 IEEE.

[2] Minakshi Bhardwaj and G.P. Singh "Types of Hacking Attack and their Counter Measure" ,International Journal of Educational Planning & Administration.Volume 1, Number 1 (2011), pp. 43-53, © Research India Publications,http://www.ripublication.com/ijepa.htm

[3] Behrouz A. Forouzan,"Cryptography and Network Security", Edition, 2.Publisher, McGraw-Hill Education (India) Pvt Limited, 2011

[4] William Stallings, "Cryptography and Network Security Principles and Practice 5 Edition", ISBN13:978-0-13- 609704-4, 2011.

[5] Markus Dürmuth "Useful Password Hashing: How to Waste Computing Cycles with Style" *Publication*: Proceedings of the 2013 Workshop on New Security Paradigms Workshop, NSPW '13 - *Date*: September 2013

[6] Zadeh Lotfi A, "Fuzzy Logic issues contentions and perspectives",IEEE International Conference on Acoustics, Speech, and Signal Processing,1994.

[7] http://www.ro.feri.unimb.si/predmeti/int_reg/Predavanja/Eng/4.Fuzzy%20logic/_26.html.

[8] Richard Shay, Saranga Komanduri, Adam L. Durity, Phillip (Seyoung) Huh, Michelle L. Mazurek, Sean M. Segreti "Can Long Passwords Be Secure and Usable?".*Publication*: Proceedings of the 2014 SIGCHI Conference on Human Factors in Computing Systems, CHI '14 - *Date*: April 2014

[9] Vance, Ashlee. "If Your Password Is 123456, Just Make It HackMe". The New York Times (2010-01-10).

[10] FIPS PUB 180-4FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION Secure Hash Standard (SHS) CATEGORY: COMPUTER SECURITY SUBCATEGORY: CRYPTOGRAPHY Information Technology Laboratory ,National Institute of Standards and Technology ,Gaithersburg, MD 20899-8900 March 2012