New Text-Based User Authentication Scheme Using CAPTCHA

Atiya Usmani, Amrah Maryam, M. Sarosh Umar and Muneeb Hasan Khan

Abstract In recent times, there is a pressing needs to develop more robust and efficient user authentication techniques in order to protect the confidentiality and integrity of crucial information. Thus, many new techniques are being developed like graphical passwords, two-step verification, and others. In this domain, the conventional text-based passwords are often criticized and considered less secure but surprisingly they are still the most common authentication technique. This popularity is due to their ease of implementation and use. In this paper, we have proposed new text-based user authentication technique using CAPTCHA. This technique overcomes most of the challenges faced by both conventional and contemporary authentication methods and offers a higher level of security than them. The password is user-friendly, quick and easy to compute, and more secure.

Keywords One-time password (OTP) · Authentication · Security Attacks · CAPTCHA

1 Introduction

Nowadays, user authentication and cyber security are a grave concern among computer scientists and engineers. There is an increase in reported cyber crimes such as data theft, cyber extortion, hacking, and various other scams and frauds.

A. Usmani (✉) · A. Maryam · M. Sarosh Umar · M. H. Khan
Department of Computer Engineering, Zakir Hussain College of Engineering and Technology, Aligarh Muslim University, Aligarh 202002, Uttar Pradesh, India
e-mail: atiya.usmani@zhcet.ac.in

A. Maryam
e-mail: almas36912@zhcet.ac.in

M. Sarosh Umar
e-mail: sarosh.umar@zhcet.ac.in

M. H. Khan
e-mail: muneebhkhan@zhcet.ac.in

S. Fong et al. (eds.), *Information and Communication Technology for Competitive Strategies*, Lecture Notes in Networks and Systems 40,
https://doi.org/10.1007/978-981-13-0586-3_32

Most of the contemporary software and Web sites used text-based password. These passwords have their demerits and limitations such as: They are susceptible to security attacks like phishing attacks, dictionary attacks, and brute force attack. The user often has to write down his password which the intruder can easily get hold off. Thus, we can conclude that text-based password is not the ideal user authentication systems, has many limitations of their own, and is not truly secure.

This means that there is an opportunity to develop new techniques [1] and methods for user authentication. These can be entirely different techniques like token-based authentication, biometrics-based authentication, we can also create a slightly modified text-based technique, or it can be a hybrid of any two technologies mentioned above. But there are certain requirements that this newly developed user authentication technique should satisfy [2]. These are listed below:

1. It should overcome the security drawbacks of contemporary password that is it should not be prone to shoulder surfing, dictionary attacks, brute force attack, etc.
2. It should be quick, easy to compute, and user-friendly.
3. Moreover, if it is a one-time password and fairly random, then it can be classified as a good user authentication technique.

In this paper, we have proposed a new text-based user authentication technique that satisfies all the above requirements. Then, we discuss its implementation and results.

2 Background

2.1 Authentication Methods

The main methods of user authentication are:

1. **Token-based authentication** requires token, for example, an ATM card, id card, RFID card. The user is authenticated based on the token. It is also referred as "what you have" type of authentication [3].
2. **Biometrics-based authentication** is also referred as "what you are" type of technique. Every human being has some unique features, biometric uses these features for authentication like the iris scan, fingerprint, facial recognition. These signs are highly secure but extremely expensive [3].
3. **Knowledge-based authentication** technique is also referred to as "what you know" type of technique. It includes both text-based and picture-based authentication [4, 5].

2.2 Possible Attacks on Text-Based Passwords

1. **Brute Force Attack**: Brute force attack is a trial and error method used by the attacker to decode encrypted data such as password through exhaustive effort [6].
2. **Dictionary Attacks**: In dictionary attacks, the attacker attempts to gain illicit access to a computer system by using a very large set of words, generally words from a dictionary to generate potential passwords.
3. **Spyware Attacks**: Spyware is a software that gathers information about the user without his knowledge and may send it to another entity without the user's consent, for example, key listening spyware or keyloggers.
4. **Shoulder Surfing**: Shoulder surfing refers to using direct observation techniques, such as looking over someone's shoulder to get information. This can also be done by using a video camera and analyzing the video footage later.
5. **Eavesdropping**: In eavesdropping, the attacker intercepts the communication between the user and the server and obtains crucial information of the user.

3 Proposed Scheme

We have proposed a new text-based authentication scheme which generates a new password each time using CAPTCHA. This technique has been devised in order to achieve a higher level of security and overcome the challenges faced by conventional text-based passwords.

3.1 Method

User authentication system provides the following functions.

1. **Registration**: During registration, the user has to enter his email and an eight character password (i.e., his real password). This real password serves as a private key for the user. He will never have to enter it again. The user also has to enter a location digit. This digit will be used later at the time of login. It gives the location of the user's answer based on the CAPTCHA challenge in the user's one-time login password. The rest of the characters in the login password are dummy characters.
2. **Login**: When the user logins every time, he sees a CAPTCHA of 15 characters. It is a random character string but it will have any one character from the user's real password or key. Let's call it C'. Using this key and the location where his password character appears in the CAPTCHA, he has to calculate new character *NC* by (1).

$$NC = Key\,[loc] \tag{1}$$

Here, *NC* is the character in the key at location *Loc*. *Loc* is given by (2).

$$Loc = loc1 + loc2 \tag{2}$$

In (2), *Loc1* is the location of C' in user's real password or key, and *Loc2* is given by (3). If the real password string contains same characters at two different indexes, the first index is taken as *Loc1*.

$$Loc2 = integer\ value\ of\ ((Loc_{Cp})/10) + ((Loc_{cp})\ modulus(10)) \tag{3}$$

Loc_{Cp} is the location of C' in CAPTCHA, where C' is the first character which is both in the random string CAPTCHA and also in the real password. If the location is a two-digit number, then the two-digit will be added.

This new character *NC* has to be entered at the correct location given by the location digit specified during registration. The other characters will be some dummy characters in the password. The password should altogether be 15 characters long. The user just need to identify the common character C' and note down the location. Then, all he needs to do is count the characters from C' based on the location in the real password.

Example 1:

Real Password: Alics%91

Location Digit: 4

CAPTCHA (Fig. 1):

C' = A, First common character in CAPTCHA and real password.
Loc2 = 3, location of C' in CAPTCHA.
Since it is a two-digit no. (12) both the digits are added (1 + 2).
Loc1 = 1, location of C' in real password.
New Character NC = c.

In simple terms, the user identifies the common character "A" that comes in his real password (Alics%91) and the CAPTCHA and notes down its location, i.e., 12. Since the location is a two-digit number, he adds the two digits, 1 + 2 = 3. So he counts three characters more from "A" in Alice%91, which will be "c". Thus, the new character "sc" has to be entered at the correct location in the login password, indicated by the location digit 4. The other characters are dummy.

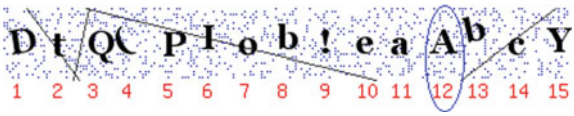


Fig. 1 CAPTCHA for example 2

Password:

A2pc9Mxq81nls*

Successfully logged in.

Thus, the password is completely random each time the user logs in it may not even have the same characters.

4 Implementation

The system is developed by using MySQL and Apache server. PHP has been used to develop the front-end software [7], and MySQL has been used to store and retrieve the data from the database. The user interface is made using HTML/CSS.

4.1 Registration Page

During registration, the user has to provide necessary information which will be saved in the database. These include his email, the result location, and a password. The result location is between 1 and 15. It indicates where the user wants to enter the result in the 15 character password at the time of login. The password should be eight characters long and must contain a special character and a numeral. The user is advised to keep a password that is easy to remember (Fig. 2).

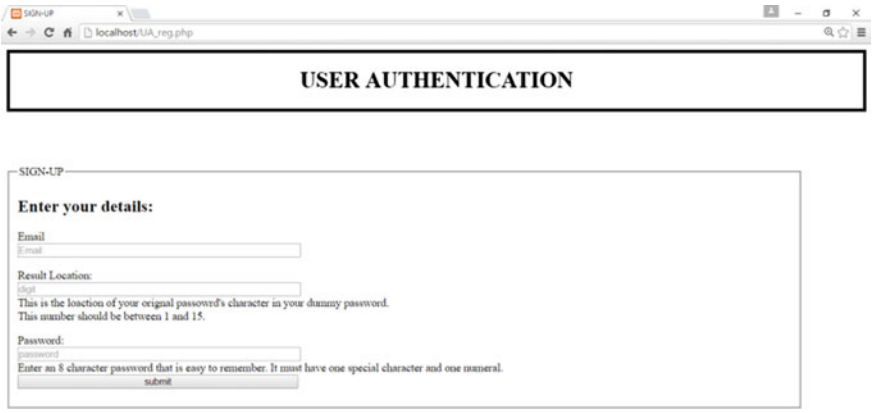


Fig. 2 Registration page

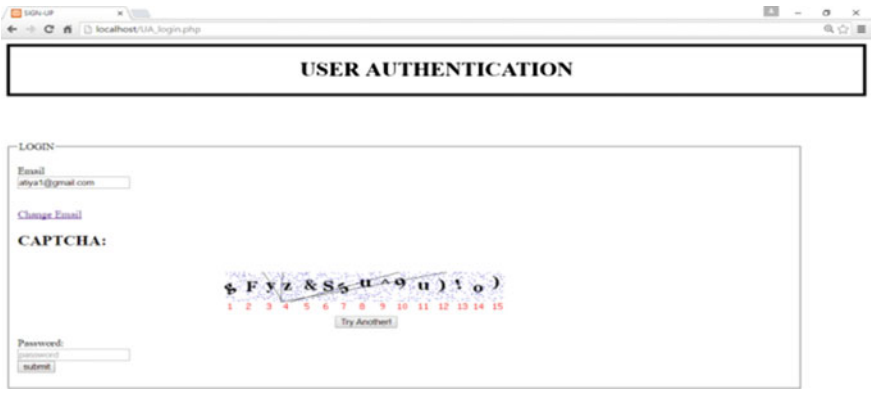
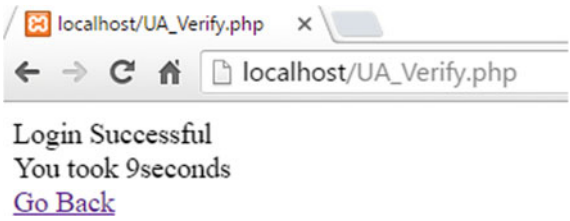


Fig. 3 Login page

Fig. 4 After login



4.2 Login

The user is given a challenge based on the information he has provided during registration. The user can login only if he successfully overcomes that challenge (Fig. 3).

The time taken by the user is automatically recorded in the database for each login attempt (Fig. 4).

5 User Study and Results

We asked different people to create an account in our authentication system. Altogether ten people volunteered. Then, we asked each user to log into his account three times and the time taken by the user was automatically calculated and stored in the database. We have assessed the quality of our software based on three main factors: time taken to log in, user-friendliness, and security.

username	time
amrah4@gmail.com	14
amrah4@gmail.com	10
amrah4@gmail.com	14
amrah4@gmail.com	8
amrah4@gmail.com	14
amrah4@gmail.com	11
amrah4@gmail.com	9
amrah4@gmail.com	14
amrah4@gmail.com	9
amrah4@gmail.com	12
amrah4@gmail.com	12
amrah4@gmail.com	7
amrah4@gmail.com	12
amrah4@gmail.com	15
atiya1@gmail.com	9
atiya1@gmail.com	13
atiya1@gmail.com	13
atiya1@gmail.com	7
atiya1@gmail.com	17
atiya1@gmail.com	9
amrah4@gmail.com	9
amrah4@gmail.com	13
amrah4@gmail.com	14
amrah4@gmail.com	51
nadiasid@gmail.com	36

username	time
nadiasid@gmail.com	23
nadiasid@gmail.com	17
ramsha.fatima@zhcet.ac.in	19
ramsha.fatima@zhcet.ac.in	24
ramsha.fatima@zhcet.ac.in	19
zeba.khanam@zhcet.ac.in	12
zeba.khanam@zhcet.ac.in	52
zeba.khanam@zhcet.ac.in	32
zeba.khanam@zhcet.ac.in	25
zeba.khanam@zhcet.ac.in	23
shivangi@gmail.com	46
shivangi@gmail.com	28
shivangi@gmail.com	29
bushrakha9198@gmail.com	28
bushrakha9198@gmail.com	26
bushrakha9198@gmail.com	19
shira@gmail.com	39
shira@gmail.com	22
shira@gmail.com	22
shira@gmail.com	16
bob@gmail.com	31
bob@gmail.com	45
bob@gmail.com	21

Fig. 5 Time taken by each candidate login table

Fig. 6 Average time query

```
MariaDB [ua]> SELECT AVG(time) FROM time_table AS Average_Time;
+-----------+
| AVG(time) |
+-----------+
|   18.6203 |
+-----------+
1 row in set (0.07 sec)
```

5.1 Time Taken to Login

We have maintained a database which stores the time taken to log in by each of our ten volunteers. An instance of the database is shown in Fig. 5.

We calculated the average time taken by each volunteer for the entire database and the result was 18.6203 s (Fig. 6).

When we first explained the scheme to a new user, he took significant amount of time to log in. But by the third attempt, the time to log in decreased dramatically. Some experienced users could login in just 6 s. Also, the time depended on the result location chosen by the user. Locations 1 and 15 give the fastest result. The average time take by each user in 3–5 attempts is shown (Fig. 7).

Fig. 7 Average time for each user

username	AVG(time)
amrah4@gmail.com	14.9756
atiya1@gmail.com	10.5000
bob@gmail.com	26.2000
bushrakha9198@gmail.com	24.3333
nadiasid@gmail.com	25.3333
ramsha.fatima@zhcet.ac.in	20.6667
shira@gmail.com	24.7500
shivangi@gmail.com	34.3333
zaonab.zaheer@zhcet.ac.in	21.2500
zeba.khanam@zhcet.ac.in	28.8000

5.2 User-Friendliness

The software is easy to use and does not require mathematical calculations. The given scheme is easy enough to be used by children's as well. The user has to remember only his/her real password. Additional memorization is not needed.

5.3 Security

The real password is never entered or shared. It acts as a key.

Sample space [8] for the real password (key) is given by Eq. (1)

$$S = A^N \tag{4}$$

where

N Length of the password,
A Total keyboard characters (usually 94).

If we calculate the key space using (4) we get $(94)^8$.

5.3.1 Password Space

For every instance of the random CAPTCHA string, the attacker must get the correct character at the correct location. This can be represented mathematically as follows:

$^{94}C_1 * {}^{15}C_1 = 1.4 \times 10^3$

The password space is comparatively less but the scheme is still very secure due to the following reasons:

1. As the proposed challenge is based on the CAPTCHA, a software or a robot cannot be employed to crack the user's password.
2. For each CAPTCHA challenge, the user has just one trial, for next attempt the CAPTCHA would be different. So even by manual brute force, the chances of cracking are nearly 0.001%. And this chance does not increase with subsequent attempts.
3. If by chance the intruder breaks into the user's account once which is very less likely to happen, he won't be able to do so again because he does not know the users key.
4. Each time the user's password is completely different and random.
5. The password is also immune to keyloggers, spyware, dictionary, brute force, shoulder surfing, guessing, eavesdropping, and other attacks which is discussed in the next section.

6 Comparison with Existing Work

- The user never has to enter his real password unlike other text-based schemes.
- Each time a new password is generated.
- Brute Force Attack: The attacker cannot go for exhaustive search to decode the password as we are using CAPTCHA, so he cannot use any computer software for this task. Moreover, the password is completely different each time.
- Shoulder Surfing, Spyware, and KeyLogger Attack: Unlike common text-based password our scheme is immune to these attacks because each time the password is completely different and random. It is impossible for the attacker to guess the correct password even after looking at previous login passwords.
- Dictionary Attacks and Guessing: The passwords are completely random and meaningless which cannot be found in a dictionary.
- Eavesdropping: Even if the attacker gets the password by intercepting the communication between the user and the server it would be completely useless as for the next time the password would be different.
- Our proposed scheme is not as easy as the conventional text-based password scheme but it is really simple and does not involve any complex mathematical calculations.

7 Conclusion

Based on the results, we conclude that the performance of the proposed method is good enough to be implemented. It can be used to protect confidential information like personal data, records, files, accounts. And can also be combined with other softwares and applications for user authentication. We can also incorporate two-step verification method in the scheme. If the user takes longer than usual to log in, we can send an one-time password (OTP) on his phone. The user can get through only if he enters his OTP correctly. That would significantly increase our password space.

References

1. Khandelwal, A., Singh, S., Satnalika, N.: User authentication by secured graphical password implementation. Int. J. Comput. Appl. (0975-8887) **1**(25), 22–24 (2008)
2. Bhanushaki, A., Mange, B., Vyas, H., Bhanushali, H., Bhogle, P.: Comparison of graphical authentication password technique. Int. J. Comput. Appl. **116**(1) (2015)
3. Kushwaha, B.K.: An approach for user authentication one time password (numeric and graphical) scheme. J. Global Res. Comput. Sci. **3**(11), 54–57 (2012)
4. Banue, S.S., Shedge, K.N.: CARP: CAPTCHA as a graphical password based authentication scheme. Int. J. Adv. Res. Comput. Commun. Eng. **5**(1), 14–18 (2016)
5. van Oorschot, P.C., Wan, T.: TwoStep: an authentication method combining text and graphical passwords. In: Babin, G., Kropf, P., Weiss, M. (eds.) E-Technologies: Innovation in an Open World. MCETECH 2009. Lecture Notes in Business Information Processing, vol. 26, pp. 233–239. Springer, Berlin, Heidelberg (2009)
6. http://searchsecurity.techtarget.com/definition/brute-force-cracking. Accessed 15 Nov 2017
7. https://www.sitepoint.com/simple-CAPTCHAs-php-gd/. Accessed 15 Nov 2017
8. https://en.wikipedia.org/wiki/Password_strength. Accessed 15 Nov 2017