Journal of Engineering and Applied Sciences 13 (Special Issue 4): 3708-3714, 2018
ISSN: 1816-949X

Generation of Random Number in Time Dependent Color CAPTCHA for Enabling Web Security Access

[1]S. Pradeep Kumar and [2]R. Ramachandaran
[1]Department of CSE, Sathyabama University, Chennai, India
[2]Department of ECE, Dhanalakshmi College of Engineering, Chennai, India

Abstract: CAPTCHA is used as a standard security mechanism to prevent the bots to enter into the commercial websites like e-Governance, inventory, educational sector and so on. Nowadays, hackers wrote malicious program and enter into the website to destroy their resources. To prevent such kind of activities in our proposed approach introduces a Random Ordered Time Dependent Color CAPTCHA (ROTDCC) for enhancing the web security protection. In such a case, the user has to enter all the characters based on the random number generation model for ensuring the security on web access. Based on the final user entry, a system can note the time and allots the random number for each allocate CAPTCHA character in the screening test. There are permutable choice to allocate various ordered random numbers for the CAPTCHAs character using Modulo Division Nine (MDN) algorithm. All the characters used in the CAPTCHA word are to be represented in different colors. This makes the same character can have the chance to represent different colors which in turn the pixel count and intensity value vary. Some graphical operation and cracks are added in the CAPTCHA word to perform a CAPTCHA entry test by the user. Consequently, this method is unbreakable by any robot using machine learning technique and in turn the human based threats to be reduced. This proposed system to display the CAPTCHA on the web entry form provide with additional security in major application areas such as banking sectors, online shopping and defense services.

Key words: ROTDCC, Random Ordered Time Dependent Color CAPTCHA, CAPTCHA Completely Automated Public Turing test to tell Computer and Human Apart, web security, defense services, online shopping

INTRODUCTION

In the modern era, the requirement of online services in the internet for providing e-Mail, purchasing, search engine with a available resources. In such a case, denial of services made by a malicious program has become a serious problem and hackers create a false account to destroy the resources. In order to avoid such a problem from the malicious problem, the CAPTCHA has been introduced to distinguish between a Robot and a user (Alkhawlani *et al*., 2015). Most of the web application requires the user to do registration and then only to enter and access this information for the daily needs at any point of time. In generally humans are superior than machine to enter into the website. The server of a particular website can give a large number to recognize the Image or text based CAPTCHA. But nowadays to make the CAPTCHA face two defects such that not able to recognize the character by the user and hackers wrote a malicious program to extract the character from the CAPTCHA set. To clear the above two aspects in this study introduces a random order mouse click based CAPTCHA provides convenience to the user and not able to detect the CAPTCHA character at any time for the hackers. It combines the features of text based and image based CAPTCHA (Singh and Pal, 2014).

Major of the website uses the own style of text based CAPTCHA representation (Alsuhibany, 2016). Generally, used CAPTCHA characters are 26 alphabets and 10 digits to represent in a programming language like java script, PHP, Phyton. Some problem faced in the generally used website as shown in Fig. 1. There are some rules to be followed for design the good CAPTCHA to a human user only:

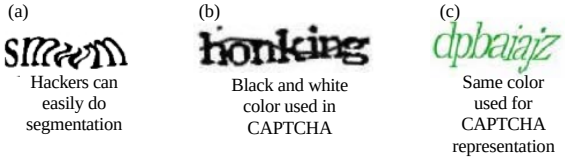

Fig. 1: Problems faced in Comonly used web sites: a) Wikipedia; b) Facebook and c) Google

Corresponding Author: S. Pradeep Kumar, Department of CSE, Sathyabama University, Chennai, India

- Computer program can able to generate the CAPTCHA character then and there
- User can able to recognize these CAPTCHA characters with a reasonable amount of time
- Hackers wrote malicious program using sophisticated software cannot detect CAPTCHA character at any time

Literature review: Gimphy introduced a CAPTCHA method at Carnage University in order to distinguish between the user and a machine. Most of the word in the CAPTCHA set were found in the English dictionary and it is easily broken by robot (Mori and Malik, 2003). Hackers break a Google CAPTCHA with a success rate of 46.2. This attack impacts a success rate of segmented CAPTCHA character has reached 95% (Ahmad *et al*., 2014). Yan has successfully broke a visual CAPTCHA from the CAPTCHASERVICE.ORG. The mechanism has simply count the number of pixel in each segmented CAPTCHA character. Hence, the hackers has easily predict the CAPTCHA characters thru a fixed pixel count and the segmentation mechanism has easily resistant to the best software available on the market (Yan and El-Ahmad, 2007). The security of existing CAPTCHA by adding noise and distortion to the text. It create harder to the user to recognize the CAPTCHA character with high rate (Azad and Jain, 2013).

MATERIALS AND METHODS

The proposed research has introduced a new evolutionary trend in a random order mouse clicked based CAPTCHA. This CAPTCHA test provides the combination of text and mouse based interaction scheme. The user fills all the sufficient details in the web application form then only the CAPTCHA screening test will appear. The separate window for the CAPTCHA screening test is to be displayed under the same web application form. Each and every CAPTCHA character are to be represented in different colors. The same character can have the chance to represent different colors which in turn size and pixel count vary. Inside the right side of the CAPTCHA screening test, a set of alphabets and numerical are displayed under a separate grid layout. The total number of characters set are displayed in the screening text is to be differed in various ways by using the number of CAPTCHA character or same CAPTCHA representation. Based on the final user entry by the user in the web application form and the system can allocate a random number in the CAPTCHA character set using modulo division nine algorithm. The User clicks the character one by one based on the sorted random number generated. The random number displayed is differed for the same number of CAPTCHA's representations. This proposed CAPTCHA is a convenient graphical user interface to the human user and it is free from OCR attacks. So, the hackers cannot have the chance break a CAPTCHA design test.

Generation of dynamic random number in mouse clicked CAPTCHA: The web based applications need the security by the CAPTCHA representation. Nearly ten colors are used to represent the CAPTCHA characters. All the individual CAPTCHA character are represented in different colors in a screen test. This makes the pixel count and intensity value can vary with the same character used. Table 1 shows the CAPTCHA character 'S' represent in different color with pixel count vary.

Till now all the web application uses the same color CAPTCHA letter with a fixed pixel count. This lead to the hackers may have the chance to break the CAPTCHA screening test. This shows the CAPTCHA has effectively resultant to segmentation affect by extracting individual character. The Yahoo CAPTCHA has successful achieve a success rate of 48% and baidu was 34% (Gao *et al*., 2014).

The degree of distortion and cracks make a strongly strengthen to the CAPTCHA and also may lead to the failure recognition by the user (Shanker *et al*., 2013). The random number is generated under each CAPTCHA character. Depends upon the final entry (system time) by the user, the random number generation differ by the same number of CAPTCHA representation. In addition to that some graphical operation and cracks are added in the CAPTCHA screening text. Meanwhile it won't disturb the human user to recognize the CAPTCHA character (Zhu *et al*., 2014). In the CAPTCHA screening test, a separate window is allotted for displaying the CAPTCHA characters along with dynamic random numbers as shown in the Fig. 2. In addition to that, four labels (shift, clear, go and cancel) are used below inside the CAPTCHA screening test.

User entry: User filled all the information in the available web form and then allowed to attempt for the CAPTCHA

Table 1: Look up table entries

Letter	Color used	Pixel count	CAPTCHA character
S	Violet	169	
S	Blue	175	
S	Green	184	

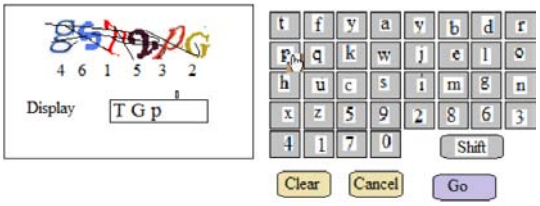

Fig 2: Mouse clicked based CAPTCHA; Shift: it is helpful to display upper character; Clear: it delete the entire user entry characters; Cancel: it delete a latest typed character; Go: go and proceed for the CAPTCHA screening test

Table 2: Look up table entries

Label	CAPTCHA character	Recognized word	No. of characters used
1		Ke5j8aE	7
2		uAdNS3e	7
3		dsOTU8	6

screening test. Each and every CAPTCHA character corresponding random number is displayed. Based on the sorted random order, CAPTCHA test ask the user to click the characters (with the help of mouse) available in the challenging area of a given screen (Kulkarni and Fadewar, 2015). If the user not able to click the sorted CAPTCHA characters correctly, once gain the a new CAPTCHA screening test will appear in the same given web form. The test is allotted for the maximum of four times in order to restrict the bots to access the information from the web.

Working principles: The CAPTCHA word used in the database that are appearing in the CAPTCHA screening test using random algorithm. The random numbers are displayed in the CAPTCHA entry test based on the Modulo Division Nine algorithm (MDN). The random number is allotted under each CAPTCHA character can vary with every second for the same CAPTCHA word used in the screening test.

Random number generation in time dependent CAPTCHA: The CAPTCHA characters used are {(a, b, ..., y, z), (A, B, ..., Y, Z), (0, 1, ..., 8, 9)}. Table 2 shows the set of CAPTCHA character used (sample) along with number of characters and a recognized word. In the database nearly 2000 sets of CAPTCHA words are maintained. The number of CAPTCHA characters is used to frame a word may vary from 6-8. The sample database shows the collectivity of sample CAPTCHA character sets.

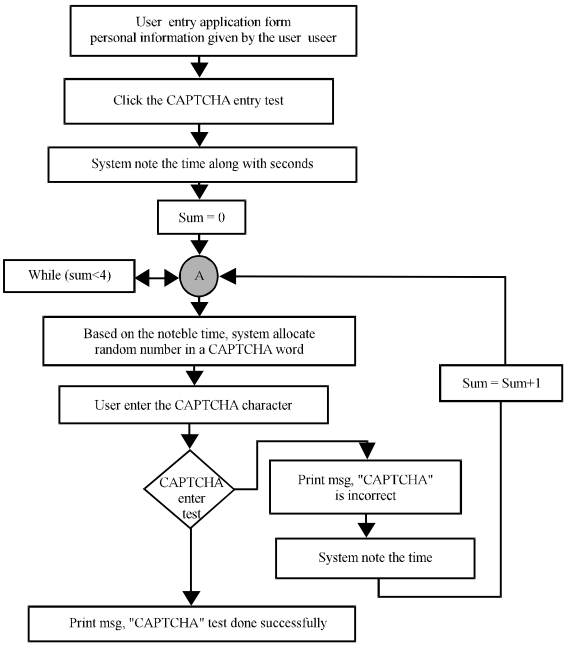

Fig. 3: Working flow of a time dependent color CAPTCHA

All the CAPTCHA's characters are used in the set are to be represented in different colors (Table 2). Hence, the same character lead to a different pixel count and the intensity value for the same CAPTCHA. It gives the prediction and security to the owner of the website for maintaining the CAPTCHA character set allotted for the screening test. The number of random numbers is used in the database set is equivalent to the number of CAPTCHA character used (Table 2). The given flow chart Fig. 3 shows the generation of random number time dependent color CAPTCHA.

System time generate random number in time dependent color CAPTCHA: The user enters all the sufficient details correctly in the web application form then only the CAPTCHA screening test will appear. System time was notable at the time of CAPTCHA entry test (Yadava *et al*., 2011). If the "user entry information" is valid, CAPTCHA word along with a equivalent number of characters is generated in the screening test according to the notable system time (Table 3).

As per the notable system time, CAPTCHA word is displayed on the screening test. Permutable choice as to be made to allocate a random number in different order using modulo division nine algorithm (Table 3). Each and every second, the allotted random number in the CAPTCHA word to represent in different order using

Table 3: Look up table entries

Cases	Notable system time	Using MDN algorithm, remainder value	Allotted random numbers/comments
No. of characters used: 7			
1	11:35:44 AM/PM (1+1+3+5+4+2) = 18% 9 = 0	0	1 3 5 7 2 4 6; User click the odd position entry CAPTCHA letter and then click the even position CAPTCHA letter
2	11:35:45	1	2 4 6 1 3 5 7; User click the even position CAPTCHA letter and then click odd position CAPTCHA letter
3	11:35:46	2	1 2 3 4 5 6 7; User click the same CAPTCHA character as same as it is
4	11:35:47	3	7 5 3 1 6 4 2; User click the odd position CAPTCHA letter in reverse order and then click even position CAPTCHA letter in reverse order
5	11:35:48	4	6 4 2 7 5 3 1; User click the even position CAPTCHA letter in reverse order and then click odd position CAPTCHA letter in reverse order
6	11:35:49	5	1 3 5 7 2 4 6; User click the odd position entry CAPTCHA letter and then click the even position CAPTCHA letter in reverse order
7	11:35:50	6	2 4 6 7 5 3 1; User click the even position CAPTCHA letter and then click odd position CAPTCHA letter in reverse order
8	11:35:51	7	7 5 3 1 2 4 6; User click the odd position CAPTCHA character letter in reverse order and then click the even position CAPTCHA letter
9	11:35:52	8	6 4 2 1 3 5 7; User click the even position CAPTCHA character in reverse order and then click the odd position CAPTCHA letter

MDN. Using MDN algorithm remainder value can have 0, 1, 2, …, 8. There are nine cases arises from the system notable time. It can be listed as shown in Table 3.

Representation of CAPTCHA characters and usage characters: All the CAPTCHA characters are readable only by the user. Some noise such as line, dots and curves are added in the screening test. User never get confused at at any time. Each CAPTCHA characters are colliding in the middle of neighboring characters. Hence, the hackers have a negative chance to break a single CAPTCHA character alone (Von Ahn *et al*., 2004).

User entered all the characters thru means of mouse in a available grid layout. Each characters hold a separate grid key in the screening test. The usage characters are {(a, b ,..., z), (0, 1, ..., 9)}. There are various combination to display all the 36 usage characters in the separate grid layout structure (Fig. 2).

Graphical operation: The displaying the CAPTCHA character in the screening text leads to the graphical operations are:

- Scaling: some of the CAPTCHA character may shrink or enlarge in either in 'X' or 'Y' axis. It can be held in difference in size of 20%
- Rotation: some of the CAPTCHAs character that may rotate in between 20-20°C
- Translation: using the matrix transformation to translate a particular CAPTCHA character in a available text box
- Sliding: some CAPTCHA character may optimally slide either in left or right relative to row above/below

All the usage characters are displayed inside the CAPTCHA screening test of a user typed web application form. Each character hold a separate grid layout. so there are 36 separate {(a, b, ..., z,), (0, 1, ..., 9)} grid layout for the characters used. Each and every CAPTCHA test, all the arrangement of usage characters are to differ. There are maintaining a ten different order to confuse the bots for predict the correct characters.

Advantages:

- Bots using sophisticated software cannot have the chance to predict the CAPTCHA character at any time
- Restrict to many attacks such as On line guessing attacks, OCR attacks. Dictionary attacks

Implementation: The representation of CAPTCHA must be a user friendly to the user and strongly restrict the bots to enter into the commercial websites. In the database maintains nearly a 600 CAPTCHA character set. Using the random number generator collect the CAPTCHA character set from the database and allotted in the screening test. The working algorithm hold the working process of a CAPTCHA screening test.

Algorithm; Mouse clicked CAPTCHA:

```
Var count: Number of times a user appears the CAPTCHA screening test
           maximum_attempt: limit for the maximum number of
           times to the user appear in the screening test
Begin
  Step 1: User entered all the entries in the web application Form
  Step 2: Count = 0
  Step 3: If user entry information are "wrong" or "empty"
          Display a Message ,"Invalid data"
          Else
          While(count≤maximum_attempt)
            Begin
              Step 4: Display a message ,"Go and proceed for the
                      CAPTCHA screening test"
```

```
                Step 5: count = count+1
                    Step 6: User clicked the CAPTCHA character based
                        on sorted order
                    Step 7: If the user entry CAPTCHA character matched
                        with the data base set
                                Display a message, "CAPTCHA DONE
SUCESSFULLY"
                            Else
                                Count = count+1
                        Goto Step 1
                    End
            End
```

Generation of random ordered mouse clicked CAPTCHA: The various statistical analysis can be carried out to generate a CAPTCHA character set. It uses cryptographic hash function can be used to generate a random ordered mouse clicked CAPTCHA (ROMCC) in a high securable manner (Banne and Shedge, 2016). Random number generator uses a various combination of all available characters {(a, b, ..., z), (A, B, ..., Z), (0.1, ..9)}. Using the MDN algorithm value, the system can allocate a random number during the CAPTCHA screening test. Meanwhile, the system can also create a recognize word using the sorted random number and maintain in the database itself.

The Modulo division Nine (MDN) algorithm is used in the time dependent CAPTCHA character algorithm. Depending upon the final user entry time, a set of random n umber is allotted under a CAPTCHA character set. These CAPTCHA words uses the combination of all available alphanumeric characters.

Algorithm; Random number modulo division nine:

```
    Var : hr-> User entry: Notable hour time
          mi->User entry: Notable minute time
          sec->User entry: Notable seconds time
          total-> a sum of the individual digits of a time
  Begin
    initially total: = 0
      while(hr>0)
        Begin
          total = total+(hr%10); hr = hr/10
        End
      while(mi>0)
        Begin
          total-total+(mi%10); mi = mi/10
        end
      while(sec>0)
        begin
          Total = total+(sec%10); sec = sec/10;
        End
       K = sum%9
        //k can value 0,1,2,3 & 4
        // allot a case 'K+1' value
        Case; (0,1,2,3, ..., 8)
          Depending upon the case, a set of Random
   number are displayed in a screening test
End
```

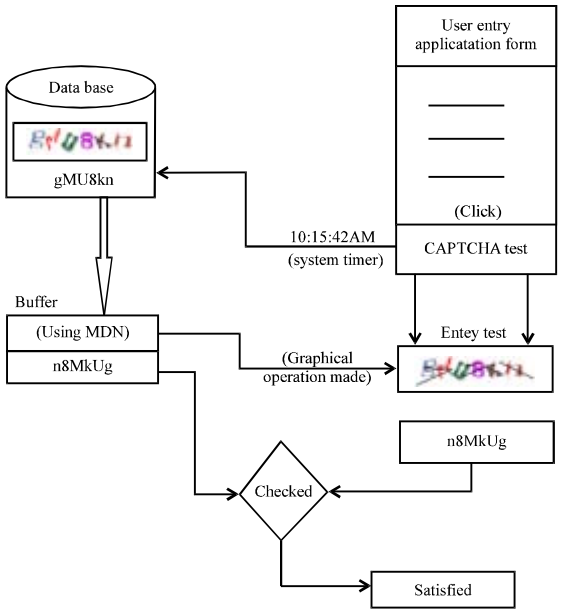

Fig. 4: Validation of a user entry test

Validation: User filled all the necessity details in the currently used web application form and then allowed to appear for the CAPTCHA screening test.

The number of CAPTCHA characters allotted in the screening test can vary from 6-8. Below each CAPTCHA character a corresponding number is displayed (Fig. 4). Based on the sorted random values, the user clicked all the characters and machine check the user entry character along with a recognized word in the database. The database allocates an equivalent sorted recognized word in the buffer during the screening test. If the user not able to clicked all the characters based on the sorted order, a new set of CAPTCHA characters will generate after some time period.

RESULTS AND DISCUSSION

We explore the CAPTCHA screening test in the same user entry web form. Figure 5 shows the snapshot for the successful completion of a user entry validation process. The effectiveness of a ROMCC is measured by evaluating its accuracy (Ahmed *et al*., 2016). An accuracy is defined as the fraction of CAPTCHA that were correctly answered by the human user.

$$\text{Accuracy} = \text{Sum}/m$$

Where:

Sum = Number of CAPTCHA test successfully performed by the user

m = Number of CAPTCHA test conducted

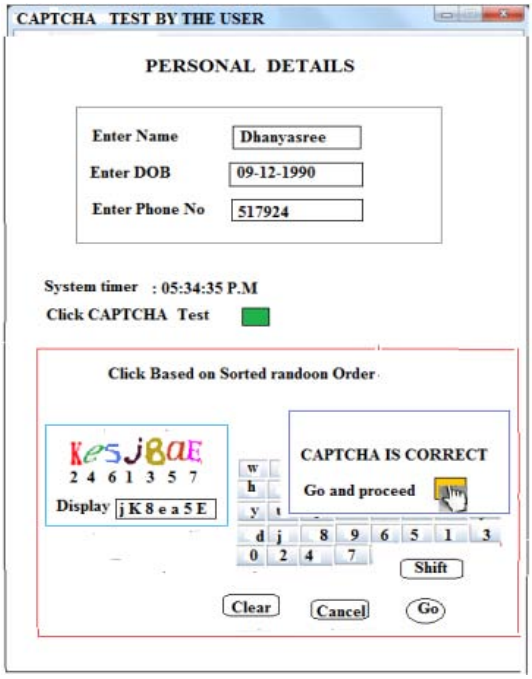

Fig. 5: CAPTCHA screening test by the user

The accuracy is calculated as follows:

$$\text{Accuracy->}g(x)=\sum_{x=1}^{x=m}*\left[\sum_{y=1}^{y=n}a[y]+\sum_{z=1}^{z=n}b[z]\right]s[x],y,z$$

Where:

y = Iteration→human user recognize 'n' number of CAPTCHA's character

z = Iteration→human user predicts the Random Number Generator (RNG), i.e., based on sorted RNG, user enters the CAPTCHA's character

x = Iteration→m times, number of sample test conducted

a[y] = 1 {if the user predicts all CAPTCHA's character correctly, otherwise a[y] = 0}

b[z] = 1 {user entered CAPTCHA's character correctly based on recognition, otherwise a[z] = 0}

s[x] = 1 { CAPTCHA test has correct}

$$g(x)=\sum_{x=1}^{x=m}*\left[\sum_{y=1}^{y=n}a[y]+\sum_{z=1}^{z=n}b[z]\right]s[x]$$

$$g(x)=\sum_{x=1}^{x=m}*\left[\sum_{y=1}^{y=n}(1)+\sum_{z=1}^{z=n}(1)\right]s[x]$$

$$g(x)=\sum_{x=1}^{x=m}*[(n-1+1)+(n-1+1)](1)$$

Table 4: ROMCC performed by the user

CAPTCHA test			Predict all the ROMCC CAPTCHA character/human recognition		
No. of character used	Attacks	Total samples	Scenario-1	Scenario-2	Scenario-3
6-8	Online, Dictionary, OCR	100	95%	96%	96%

Finally, the $g(x) = [(m-1+1)(2n)] = \theta(2nm) = \theta(nm)$. Thus, the time complexity of accuracy_CAPTCHA_entry algorithm is $\theta(nm)$.

An accuracy has reached for a 97% in all the possible combination of CAPTCHAs. For 100 number of test conducted (m = 100) using various number of CAPTCHAs and Table 3 shows that the proposed method has achieved good results rather than other approaches (Table 4).

CONCLUSION

The proposed method displays a CAPTCHA screening test only after the completion of the user registration process. Each and every character appear in different colors. The representation of CAPTCHA makes easier for the user, even though the characters collide with others and have some cracks. In order to create the time dependent color CAPTCHA in a more securable way, the CAPTCHA screening test is to be displayed within a limited time period. If the user wants to write the CAPTCHA test within the given period and then only sign and enter into the required web area. If delay occurs during the registration process, a new set of CAPTCHA will be generated.

REFERENCES

Ahmad, A.E. J. Yan and M. Tayara, 2014. Robustness of Google CAPTCHAs. Master Thesis, Newcastle University, Newcastle Upon Tyne City, England, UK.

Ahmed, T., K.A. Tushar, S.I. Nova and M.M. Rahman, 2016. Simple, robust and user friendly CAPTCHA InstaCap for web security. Intl. J. Hybrid Inf. Technol., 9: 163-182.

Alkhawlani, M., M. Elmogy and E.H. Bakry, 2015. Text-based, content-based and semantic-based image retrievals: A survey. Intl. J. Comput. Inf. Technol., 4: 58-66.

Alsuhibany, S.A., 2016. Evaluating the usability of optimizing text-based CAPTCHA generation. Methods, 7: 164-169.

Azad, S. and K. Jain, 2013. CAPTCHA: Attacks and weaknesses against OCR technology. Global J. Comput. Sci. Technol., 13: 1-5.

Banne, S.S. and K.V. Shedge, 2016. CARP: Captcha as graphical password based authentication scheme. Intl. J. Adv. Res. Comput. Commun. Engg., 5: 14-18.

Gao, H., W. Wang, Y. Fan, J. Qi and X. Liu, 2014. The robustness of connecting characters together CAPTCHAs. J. Inf. Sci. Eng., 30: 347-369.

Kulkarni, S. and H.S. Fadewar, 2015. Mouse dynamic based CAPTCHA: Brief review. Intl. J. Adv. Res. Comput. Sci. Software Eng., 5: 383-385.

Mori, G. and J. Malik, 2003. Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA. Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Vol. 1, June 18-20, 2003, IEEE, Madison, Wisconsin, USA., ISBN:0-7695-1900-8, pp: 134-141.

Shanker, D., P. Gupta and A. Jaiswal, 2013. Hybrid collage CAPTCHA. Intl. J. Sci. Eng. Res., 4: 1-7.

Singh, V.P. and P. Pal, 2014. Survey of different types of CAPTCHA. Intl. J. Comput. Sci. Inf. Technol., 5: 2242-2245.

Von Ahn, L., M. Blum and J. Langford, 2004. Telling humans and computers apart automatically. Commun. ACM, 47: 56-60.

Yadava, P., C. Sahu and S. Shukla, 2011. Time-variant Captcha: Generating strong Captcha security by reducing time to automated computer programs. J. Emerging Trends Comput. Inf. Sci., 2: 701-704.

Yan, J. and A.S. El-Ahmad, 2007. Breaking visual CAPTCHAs with naive pattern recognition algorithms. Proceedings of the 23rd Annual Conference on Computer Security Applications, December 10-14, 2007, Miami Beach, FL, USA., pp: 279-291.

Zhu, B.B., J. Yan, G. Bao, M. Yang and N. Xu, 2014. Captcha as graphical passwords-A new security primitive based on hard AI problems. IEEE. Trans. Inf. Forensics Secur., 9: 891-904.