International Journal of Electronics, Communication & Soft Computing Science and Engineering
ISSN: 2277-9477, Volume 2, Issue 1

Integrating Random Character Generation to Enhance CAPTCHA using Artificial Intelligence

Manisha B. Thombare **Parigha M. Derle** **Arpita K. Wandre**

***Abstract* — Completely Automated Public Turing Tests to Tell Computers and Humans Apart (CAPTCHAs) are now an almost standard security mechanisms for defending against undesirable and malicious bot programs on the Internet (especially those bots that can sign up for thousands of accounts a minute with free email service providers, send out thousands of spam messages in an instant, or post numerous comments in blogs pointing both readers and search engines to irrelevant sites). CAPTCHAs generate and grade tests that most humans can pass but current computer programs can't. Such tests—often called CAPTCHA challenges—are based on hard, open artificial intelligence problems.**

***Key Words* — CAPTCHA, Internet security, Usability.**

I. INTRODUCTION

CAPTCHAs are widely used these days to prevent any script that is automated to perform an activity that is meant for a human. They are basically used to prevent bots which use these scripts to make false accounts at the websites. CAPTCHAs can be categorized into two types mainly: -

A. Visual CAPTCHA and
B. Audio CAPTCHA.

Visual CAPTCHAs have characters with noise applied in background. Audio CAPTCHAs have characters written on a complex background like Visual CAPTCHAs but they also read out the characters which is helpful for a physically disabled person or in case some character cannot be recognized.

Since the early days of the Internet, users have wanted to make text illegible to computers. The first such people were hackers, posting about sensitive topics to online forums they thought were being automatically monitored for keywords. To circumvent such filters, they would replace a word with look-alike characters. Numerous other variants, such that a filter could not possibly detect all of them. This later became known as leetspeak.

The first discussion of automated tests which distinguish humans from computers for the purpose of controlling access to web services appears in a 1996 manuscript of Moni Naor from the Weizmann Institute of Science, entitled "Verification of a human in the loop, or Identification via the Turing Test". Primitive CAPTCHAs seem to have been later developed in 1997 at AltaVista by Andrei Broder and his colleagues in order to prevent bots from adding URLs to their search engine.

Looking for a way to make their images resistant to OCR attack, the team looked at the manual to their Brother scanner, which had recommendations for improving OCR's results (similar typefaces, plain backgrounds, etc.). The team created puzzles by attempting to simulate what the manual claimed would cause bad OCR.

In 2000, von Ahn and Blum developed and publicized the notion of a CAPTCHA, which included any program that can distinguish humans from computers. They invented multiple examples of CAPTCHAs, including the first CAPTCHAs to be widely used (at Yahoo!)[1].

A. Previous System

CAPTCHAs are used to prevent bots from using various types of computing services. Applications include preventing bots from taking part in online polls, registering for free email accounts (which may then be used to send spam), and, more recently, preventing bot-generated spam by requiring that the (unrecognized) sender pass a CAPTCHA test before the email message is delivered. They have also been used to prevent people from using bots to assist with massive downloading of content from multimedia websites prevent bots from posting spam links as a comment or message. In previous CAPTCHA building process a meaningful random word is generated by NLP algorithm. This generated word is given as input to image processing algorithm such as histogram which is known.

To break the CAPTCHA hackers use following method: The CAPTCHA image is given as an input to the program. The image should be in .bmp format. The CAPTCHA image is then converted into gray scale image. It is converted into gray scale because CAPTCHA image contains many colors and to work on each of them is very difficult so converting it into gray scale helps only to work on 256 intensity values. Algorithm to Convert 24bit to 8bit Gray Scale Image.

a. Get BMP File As Input File.
b. Extract Header Information.
c. If input is 24 bit, Create Output Header and Palette.
d. Read three pixels from input file and calculate average.
e. Write Average in Output File.
f .Repeat step d and e till end of file.

II. PROPOSED SYSTEM

There are a number of important characteristics that a CAPTCHA can exhibit. These include the difficulty to be solved by OCR and any attack programs, readable common distortions, resisting malicious attacks, carrying many bits of information, the capability of coexisting with other CAPTCHAs, and little cognitive computation requirement by the user. The relative importance of these characteristics depends on the CAPTCHA type. The principles behind CAPTCHA are as follows:

- The user is presented with a garbled image on which some text is displayed. This image is generated by the server using random text.
- The user must enter the same letters in the text into a text field that is displayed on the form to protect.
- When the form is submitted, the server checks if the text entered by the user matches the initial generated text. If it does, the transaction continues. Otherwise, an error message is displayed and the user has to enter a new code.
- Exploits observation that humans are still much better than computers at many pattern recognition tasks.
- Paradigm is variant of well known Turing Test
- Is test effective at keeping out machines?
- Is test tolerable to humans?

A. Working of system

Word character generation algorithm generates random word using random character generation algorithm. NLP processor processes the word. Then random character generation algorithm works on that word then CAPTCHA building algorithm 1,2 and 3 works on that word.

After process if image processing algorithm 1 and 2 the CAPTCHA is displayed on the user side to re-enter.

Thus, the user re-enters the CAPTCHA word in reverse order.

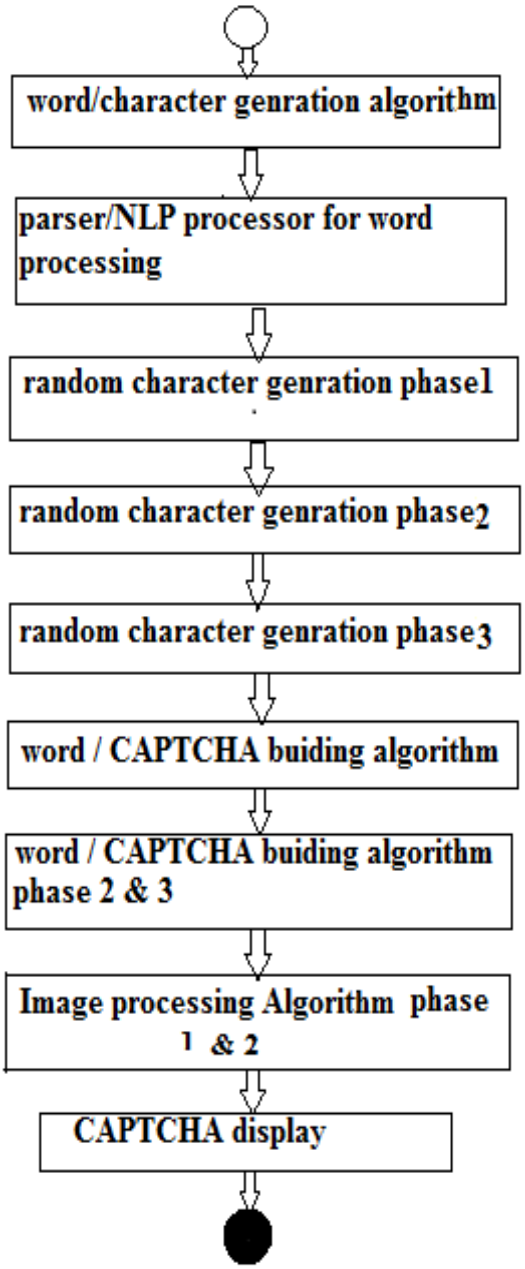

Fig. 1. Working of system

III. IMPLEMENTATION ASPECTS

A. Algorithm:

In this paper, a new method has been developed for differing human users and computer programs from each other by mainly splitting CAPTCHA image into several parts with rotation and drawing a great deal of lines and circles randomly to the background. Additionally, a grid effect has been added to the background. Lines and circles have been randomly drawn in the color of text so that OCR program confuse while distinguishing which one is character

or not. In our method, CAPTCHA text consists of the characters and number in a range of "ABDE FHKL MNPRST UVWXZ abdefgikm nopqrstuvwxyz023456789". The text is composed of five characters, and each character has its own bending and size value. Characters are split into several parts and each part is given randomly a rotation value in a certain angle domain interval such as: [-1°, 1°], [-3°, 3°], [- 5°, 5°]. Image parts are also split individually with random width and height values which provide an extra difficulty for OCR programs while finding the start and end of the images. Rotation in character parts provides confusion in recognizing the exact one. The text shown in Figure 2 below is indeed 'W9XZq'. This text is easily recognizable by the human but not OCR program. This CAPTCHA image is split into 8 parts as (4 X 2) matrix shape and each split has a random rotation angle value between -3 and 3 degree. Splits have random width and height values. Background and CAPTCHA text are in similar colors.

Fig. 2. Drawing CAPTCHA image

There is a grid in black color at the background. Lines and circles are also drawn in black color such as the CAPTCHA text. When you look at the first character, it is not easy to recognize the letter exactly due to rotation and splitting of character image. It seems like 'V' or 'I' or 'W'. In fact, it is 'W' but it is not recognizable for OCR program because character 'W' is split into two different parts as 'V' and 'I'. The other letters in CAPTCHA image have same difficulty for OCR program. The programming steps of the algorithm that we developed to generate CAPTCHA images are given with pseudo code and runtime output screenshots as in follows;

Step 1. Start the session.

Step 2. Generate *n* letters random string from the string "ABDEFHKLMNPRSTUVWXZabdefgikmnopqrstuvwxyz0 23456789".//Take out some easy letters not to be confused by the user; C/G I/l Q/O h/b. //Users of this algorithm may choose other languages such as Arabic and Korean by modifying the string as they wish.

Step 3. Create the hash for the random text and put it into the session.

Step 4. Create transparent CAPTCHA image with *w* by *h* image size and add CAPTCHA text over it. Transparent CAPTCHA image with text can be created by specific PHP (Personal Home Page) built-in function: imagettftext().

Step 5. Set the initial X-position and Y-position of captcha image to 0.

Step 6. Split the captcha into *k* by *l* Matrix shape by dividing the captcha width into *k* parts and the height into *l*.

Step7. Start a loop from 0 to k*1 // After completing the first row in order to split into k parts, then pass to next row. If (i+1) Mod k+1 = 0 Then Set initial X-Position to 0 and initial Y-Position to Split Height (Image Height / l) End If.

Step 7.1. Create an array to put the split parts and put the split images into array.

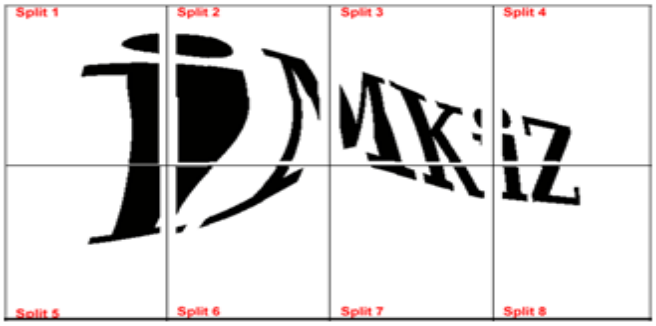

Fig. 3. Transparent split CAPTCHA image without rotation in Step 8

Step 7.2. Randomize integer between *-d* and *d* to give random rotation to the splits. Rotate the splits with randomized variable that is random between *-d* and *d*.

Fig. 4. Transparent split CAPTCHA image with rotation between -1° and 1°

Step 7.3. To pass to another split in one row, increase the initial X-Position by Split Width (Image Width / k) in each loop step.

Step7.4. End Loop.

Step 8. Combine the splits to create new CAPTCHA with split and rotation.

Step 9. Add background to transparent new Captcha image object with randomly drawn lines and special effects (Number of lines=250, line color is black and add grid effect).

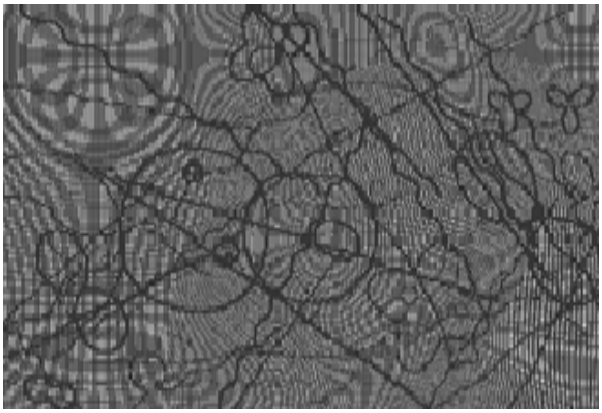

Fig. 5. Background with randomly drawn lines and special effects in Step 9

Step 12. Destroy the final CAPTCHA image object to be refreshed in each session.

B. Working of a system

- Registration: Ask user to enter registration form.
- Send Question: Server sends question to user.
- Polling answer: User sends answer to question asked by server.
- Generate CAPTCHA keyword: Server creates a CAPTCHA keyword.
- Initial image processing on CAPTCHA: Image processing algorithm generates an image of CAPTCHA keyword.
- Multiple image processing on CAPTCHA: Various image processing algorithm processes on CAPTCHA image.
- View CAPTCHA to User: The CAPTCHA generated by server is displayed on user system.

Fig. 6. CAPTCHA image displayed on user system

- Ask user to re-enter CAPTCHA: User enters the CAPTCHA in reverse order as given in a following diagram:

Fig. 7. Re-entered CAPTCHA by user

- If entry is correct then accept poll consider it for calculation: If User entered CAPTCHA is matches with reverse string of CAPTCHA then it is correct and consider that poll in computation.

IV. MATHEMATICAL MODEL

A. Probability

-Event:

A set of possible outcomes is called “event”.

-Sample Space:

Set of all possible outcomes is called “sample space”.

-Probability of event:

Let A be an event which is finite or countable infinite that is A={x1,x2,….xn} or A={x1,x2,..xn,…}

Then probability of A is defined as: P (A) =∑P (xi)

-Probability of our Project Modules:

1. Android Places:

We have two possibilities for getting of location of particular place i.e. whether get proper location or not. (Success or Fail)

P (Success) =1/2

P (Fail) =1/2

Hence P(get location) =P(Success)+P(Fail)

=1/2 + 1//2

= 1

2. Tracking Friends and Family:

We have two possibilities for tracking the location using GPS i.e. whether success access or not. (Success or Fail)

P(Success) =1/2

P(Fail) =1/2

Hence,

P(location of friends and family) = P(Success) + P(Fail)

= 1/2 +1/2

=1

3. Disaster Management:

We have two possibilities for accessing the location using GPS i.e. whether success access or not. (Success or Fail)

P(Success) =1/2

P(Fail)=1/2

Hence P(agent location) =P(Success) + P(Fail)
=1/2 + 1/2
= 1

V. CONCLUSION

In our system we are developing CAPTCHA building algorithms in which the CAPTCHA will be entered in a reverse order so it will be easy to check the user is there or not.

In this way, this system will enhance the security of system.

REFERENCES

[1] DESIGNING CAPTCHA ALGORITHM: SPLITTING AND ROTATING THE IMAGES AGAINST OCRS.
[2] Jeff Yan and Ahmad Salah El Ahmad Newcastle University, England.
[3] Blum, M., 2000, The CAPCTHA Project, Completely Automatic Public Turing Test to Tell Computers and Humans Apart", Dept. of Computer Science, Carnegie-Mellon University.
[4] Von Ahn, L., Blum, M., Nicholas, J.H., Langford, J., " CAPTCHA: Using Hard AI Problems For Security", *In* Proceedings of Eurocrypt, pp.294-311, 2003
[5] CAPTCHA design: colour, usability and security.
[6] Captcha Robustness: A Security Engineering Perspective.
[7] CAPTCHA: Using Hard AI Problems For Security.
[8] Enhancing E-mail Security by CAPTCHA based Image Grid Master Password.
[9] L. Ahn, M. Blum, N.J. Hopper and J. Langford, CAPTCHA: Using Hard AI Problems for Security, Proceedings of International Conference on the Theory and Applications of Cryptographic Techniques, pp. 294, 2003.
[10] Algorithm To Break Visual CAPTCHA.
[11] A FRAMEWORK FOR DEVANAGARI SCRIPT-BASED CAPTCHA.
[12] Enhancing E-mail Security by CAPTCHA based Image Grid Master Password.
[13] ALGORITHM FOR SECURED ONLINE AUTHENTICATION USING CAPTCHA.

AUTHOR'S PROFILE

Manisha B. Thombare
B.E. Computer (Appear), SITRC, Nashik.

Parigha M. Derle
B.E. Computer (Appear), SITRC, Nashik.

Arpita K. Wandre
B.E. Computer (Appear), SITRC, Nashik.