JCSE International Journal of Computer Sciences and Engineering Open Access

Research Paper Volume-3, Issue-5 E-ISSN: 2347-2693

Automatic crack of Yahoo CAPTCHA

Niket Choudhary[1*], and Sudarshan Deshmukh[2]

[1*,2] *Department of Computer Engineering, Pimpri Chinchwad College of Engineering,*
Savitribai Phule Pune University, India,

www.ijcseonline.org

Received: Apr/22/2015 Revised: May/02//2015 Accepted: May/16/2015 Published: May/30/ 2015

Abstract— CAPTCHA is a security mechanism used to prevent bots from using the services of the website intended for humans. Till date, a number of CAPTCHA schemes have been successfully broken which made the design of CAPTCHAs an interesting area of research. Schemes of CAPTCHAs can be categorized as text based, image based, animation based, natural language based, option based and audio based. This paper explains some of the strengths and weaknesses of the CAPTCHA currently used by Yahoo and steps to crack it automatically. The CAPTCHA is an animation based text CAPTCHA. It is cracked by first removing the noise in the background and finally applying our own developed Optical Character Recognition (OCR) program which is specialized for reading characters in Yahoo CAPTCHA only. The automatic crack program has a successful rate of 63%.

Keywords— Breaking Yahoo CAPTCHA; animation based; text based; OCR;

I. Introduction

A CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) is a challenge given to its user to identify whether the user is a human or a program. The CAPTCHA was introduced by [1] in 2003. Since then CAPTCHAs are used as a standard security mechanism by online service providers to prevent bots from using their services intended for humans. A bot can use an online service by creating number of email accounts and sending thousands of spam mails every minute [2].

Over the years, many popular online service providers such as Yahoo, Google, Microsoft, Facebook etc. have been constantly using CAPTCHA to provide security against online abuse. From the variety of CAPTCHAs text based CAPTCHAs are most popular because of their low implementation and maintenance cost [3]. Text based CAPTCHAs consists of a random sequence of characters, either alphabets or digits or both, presented on an image. The image may contain noise in the background or text may be distorted to make the challenge difficult to be cracked by a program.

In spite of all these many CAPTCHAs got successfully cracked automatically in past using the concepts of machine learning, computer vision, pattern recognition or other methods [2, 4, 5, 6]. Other than this, OCR programs are also becoming very efficient in recognizing text within an image.

In the next section we have described the automatic cracking techniques applied over some of the most popular and difficult CAPTCHAs developed in past. Then, in section 3 we have explained our proposed work which is examination of robustness of Yahoo CAPTCHA and development of the program to crack it automatically. Section 4 contains results and discussion and finally section 5 concludes the paper.

II. Literature Survey

Repeated development of automatic cracks of CAPTCHAs made design of a secured and robust CAPTCHA an interesting area of research. Many researchers studying the robustness of existing CAPTCHA designs have shown that how they are vulnerable to automatic attacks. The weaknesses in these CAPTCHAs are explained next.

The CAPTCHAs developed at Carnegie Mellon University known as Gimpy family of CAPTCHAs, also used by Yahoo, were successfully attacked by [7] with successful rate of 92% for EZ-Gimpy CAPTCHA and 33% for Gimpy CAPTCHA. Their automatic cracking technique was based on matching shape contexts of characters and using a database of known objects. The CAPTCHA was presenting an English word from the collection of pre-defined 600 English words. This made cracking the CAPTCHA easier and they were able to crack the text all at once instead of cracking it character by character. Reference [4] developed a low-cost attack for Microsoft CAPTCHA used during 2007. This CAPTCHA was automatically cracked but gave an important principle of segment resistant. The principle of segment resistant states that a CAPTCHA whose characters if could not be separated then it would be difficult to crack it automatically. In spite of cracking the CAPTCHA [4] did not negate the value of the principle- segment resistant. They showed that fewer CAPTCHAs which are called to follow the segment resistant principle do not actually follow it and characters can be separated after a close examination of it. Many CAPTCHAs still in use on internet can be cracked easily by a simple approach of pixel count. Each character in the CAPTCHA is made up of different count of pixels and so can be easily recognized [2, 8, 9]. They also suggested that segment resistant principle is not strong enough alone. It must be accompanied by some local or global warping for distortion of characters. Through the approach of [10] to automatically crack EZ-Gimpy and Gimpy CAPTCHA [8] showed the importance of one new principle for designing CAPTCHAs- hard to separate text from background. It is the first stage that any automatic attack goes through. Thus if text is difficult to locate in the image then no further work can be done to crack the challenge. Reference [11] used image processing techniques, k-means clustering algorithm, digital image in-painting, recognition of characters based on cross-correlation etc. to automatically crack many e-Banking CAPTCHAs. The popular Google reCAPTCHA is successfully cracked by [12] using the concept of shape contexts of entire words and by [13] using the segmentation and recognition of characters. Reference [14] showed how simple attacks can break a number of hollow CAPTCHA designs. So far we discussed about automatic crack of text based CAPTCHAs but crack of other schemes are also available. For example a number of automatic cracks of image based CAPTCHAs can be seen in the work of [15]. Reference [16] demonstrated attack on MathCAPTCHA. Reference [17, 18] broke audio based CAPTCHAs.

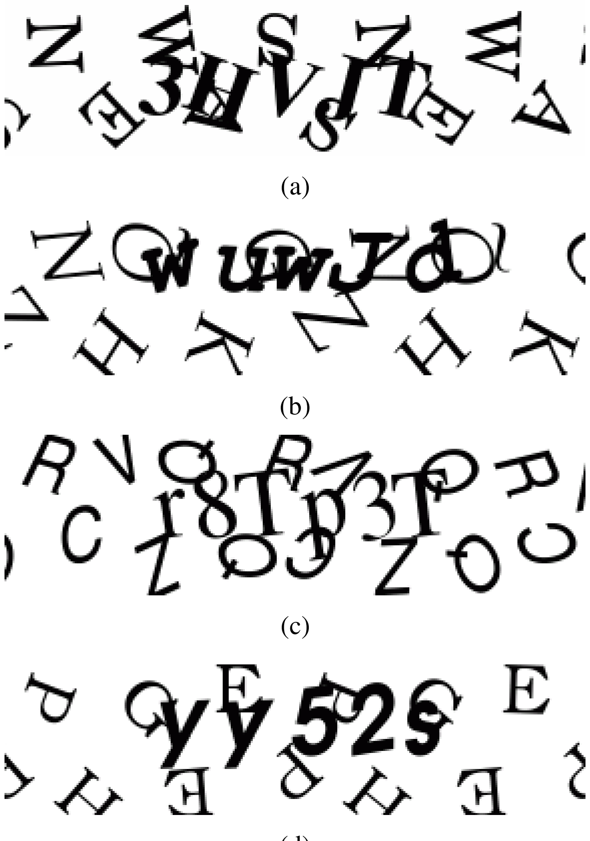
III. Proposed Work

In our proposed work we have examined the robustness of Yahoo CAPTCHA and developed a program to break it. Fig. 1(a)-(d) show four examples of Yahoo CAPTCHA.

Our observations on the CAPTCHA are as follows:

1. It is a collection of 40 images played with the time gap of 0.1 second.
2. Dimension of each image is 290X80 pixels.
3. All images are black and white.
4. The animation shows the impression of vertical or diagonal movement of text either clockwise or anticlockwise.
5. The background noise consists of capital letters written with different font than the font of the required text. There are two rows of noise and movement is from left to right.
6. While in motion the characters of text get slightly inclined towards left hand side or right hand side. Inclination of each character is different from others.
7. Fonts of text and noise are not fixed and never same.
8. Characters used are A, F, G, H, J, L, M, T, V, b, c, d, e, n, p, r, s, u, w, y, z, 2, 3, 4, 5, 6, 7 and 8.
9. Number of characters in text is 5, 6 or 7.

After the close examination of the CAPTCHA we developed a program to break it.



(a)

(b)

(c)

(d)

Fig. 1. Examples of Yahoo CAPTCHA (a) Example 1 (b) Example 2 (c) Example 3 (d) Example 4

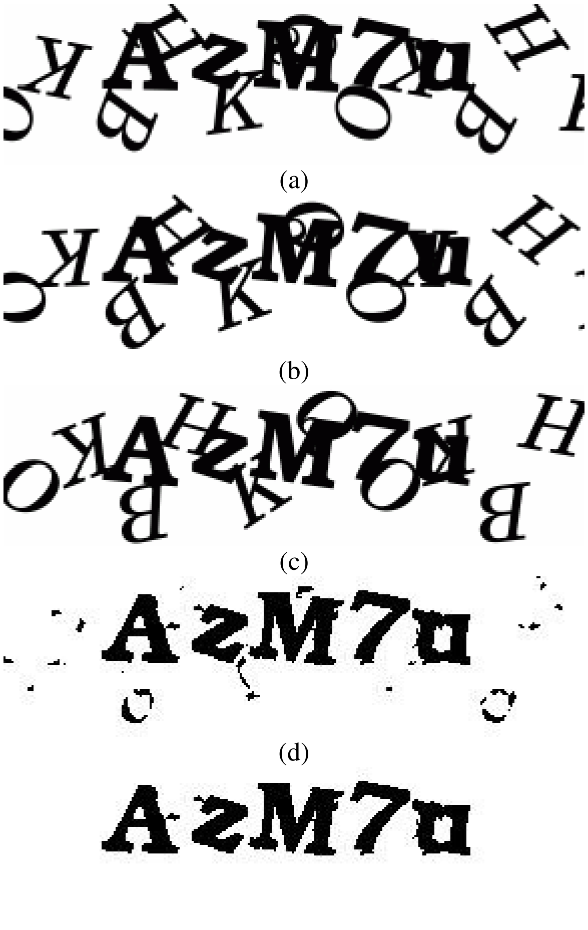
Steps of the program are as follows.

1. Separate the images of the animation.
2. Select image number 10, 11 and 13. Store the image contents in binary matrix where 1 representing black pixels and 0 representing white pixels.
3. Superimpose them in such a manner so that row number of image 11 is 2 more than row number of image 10 and row number of image 13 is 6 more than row number of image 10. Store the result in matrix 1. (Note: If text moves vertically downwards then this step makes texts in the three images overlap.)
4. Again superimpose image number 10, 11 and 13 in such a manner so that row number of image 11 is 2 less than row number of image 10 and row number of image 13 is 6 less than row number of image 10. Store the result in matrix 2. (Note: If text moves vertically upwards then this step makes texts in the three images overlap.)
5. Again superimpose image number 10, 11 and 13 in such a manner so that row number of image 11 is 2 more than row number of image 10, column number of image 11 is 4 more than column number of image 10, row number of image 13 is 6 more than row number of image 10 and column number of image 13 is 12 more than column number of image 10. Store the result in matrix 3. (Note:

If text moves diagonally downwards from left to right then this step makes texts in the three images overlap.)

6. Again superimpose image number 10, 11 and 13 in such a manner so that row number of image 11 is 2 less than row number of image 10, column number of image 11 is 4 less than column number of image 10, row number of image 13 is 6 less than row number of image 10 and column number of image 13 is 12 less than column number of image 10. Store the result in matrix 4. (Note: If text moves diagonally upwards from right to left then this step makes texts in the three images overlap.)
7. Again superimpose image number 10, 11 and 13 in such a manner so that row number of image 11 is 2 more than row number of image 10, column number of image 11 is 4 less than column number of image 10, row number of image 13 is 6 more than row number of image 10 and column number of image 13 is 12 less than column number of image 10. Store the result in matrix 5. (Note: If text moves diagonally downwards from right to left then this step makes texts in the three images overlap.)
8. Again superimpose image number 10, 11 and 13 in such a manner so that row number of image 11 is 2 less than row number of image 10, column number of image 11 is 4 more than column number of image 10, row number of image 13 is 6 less than row number of image 10 and column number of image 13 is 12 more than column number of image 10. Store the result in matrix 6. (Note: If text moves diagonally upwards from left to right then this step makes texts in the three images overlap.)
9. Count the number of 1s in matrix 1, 2, 3, 4, 5 and 6. Select the matrix having highest count of 1 for further processing. Consider matrix 1 is selected.
10. Identify every cluster of 1s and represent them with a unique number.
11. Get the count of each unique number in the matrix and if count is found less than 60 then convert it to 0. (Note: Due to step 10 and 11 most of the noise present in the matrix is removed as their pixel count is less than 60. The characters of text always contain pixel count more than 60 so they sustain the steps 10 and 11.)
12. If any object has height greater than 35 then check for any pixels present in the row of the top and bottom of the object.
13. If two alternate rows of either top or bottom of the object contain no pixel then remove the pixels of that object up to 5 rows from top (if no other pixels are detected at the top) or up to 5 rows from bottom (if no other pixels are detected at the bottom). (Note: Steps 12 and 13 are used to remove noise which is attached to the character at the top or bottom. Steps 12 and 13 convert the value of that noise to 0.)
14. Finally apply OCR on the resultant matrix.

The OCR, developed by us, is based on the limited set of characters and their inclinations. Traditional OCR programs need clean image of texts but as our final image consists of characters with little noise attached with it and with different inclinations we developed OCR with samples containing straight characters, left and right inclined characters. Cleaning process of one sample of Yahoo CAPTCHA is shown in Fig. 2 (a)-(e).


(a)

(b)

(c)

(d)

(e)

Fig. 2. Steps to crack Yahoo CAPTCHA (a) Take image number 10 (b) Take image number 11 (c) Take image number 13 (d) Overlap image number 10, 11 and 13 (e) Remove noise still remaining using pixel count

IV. Results and Discussion

We tested our program over 100 Yahoo CAPTCHAs and out of them for 63 challenges we got the correct results. The program could not crack the CAPTCHAs in which texts in image number 10, 11 or 13 are at the extreme top or bottom of the image. When reaching at the top or bottom of the image the inclination of characters from their previous inclinations differed a lot and thus superimposition of the three images did not produce a clear picture. Our program also failed in some cases where the font of the text was very thin.

Making the motion of text random in the animation will fail the program. Because of following a pattern it is easier to locate the text in the next constituent image and minor inclinations of characters are also not creating any problem. So it is easy to get the text after superimposing three consecutive images. Even if making the motion of text random, density of black pixels needs to be uniformed throughout the image otherwise based on observation of density of black pixels in constituent images it will be easy to locate the text in the image.

V. Conclusion

This paper investigated the robustness of Yahoo CAPTCHA. The CAPTCHA appears stronger because of being animation based; not having a particular font for the text; keeping the text in motion; change of inclination of characters of text and creation of background noise with alphabets only. But due to following a particular pattern in motion of the text our developed program was able to produce a cleaner image by superimposing three images i.e. image number 10, 11 and 13. Finally we applied our own developed OCR program containing samples of limited characters with different inclinations and achieved a successful rate of 63% in cracking the CAPTCHA automatically. Our program will fail if motion of text is made random.

References

[1] L.von Ahn, M. Blum, N. J. Hopper and J. Langford, "CAPTCHA: using hard AI problems for security", Springer, vol. 2656, **2003,** pp. **294-311**.

[2] J.Yan and A. S. E. Ahmad. "Breaking visual CAPTCHAs with naive pattern recognition algorithms", IEEE Computer Society, **2007**, pp. **279-91**.

[3] K.Chellapilla, K. Larson, P. Y. Simard and M. Czerwinski, "Designing human friendly human interaction proofs (HIPs)", ACM, **2005,** pp. **711-720**.

[4] J.Yan and A. S. E. Ahmad, "A low-cost attack on a Microsoft CAPTCHA", ACM conference on computer and communications security,ACM, **2008,** pp. **543-554**.

[5] A.S.E. Ahmad, J. Yan and W. Y. Ng, "CAPTCHA design: color, usability, and security", IEEE Internet Computing, **2012**, pp. **44-51**.

[6] Y.Nakaguro, M. N. Dailey, S. Marukatat and S. S. Makhanov, "Defeating line-noise CAPTCHAs with multiple quadratic snakes", Computers & Security, Elsevier, **2013,** pp. **91-110**.

[7] G.Mori and J. Malik, "Recognizing objects in adversarial clutter: breaking a visual CAPTCHA", IEEE Computer Society, **2003,** pp. **134-144**.

[8] J.Yan and A. S. E. Ahmad, "CAPTCHA security: a case study", IEEE Security and Privacy, **2009,** pp. **22-28**.

[9] J.Yan and A. S. E. Ahmad, "CAPTCHA robustness: a security engineering perspective", IEEE Computer Society, **2011,** pp. **54-60**.

[10] G.Moy, N. Jones, C. Harkless and R. Potter, "Distortion estimation techniques in solving visual CAPTCHAs", IEEE Computer Society conference on Computer Vision and Pattern Recognition, **2004,** pp. **23-28**.

[11] S.Li, S. A. H. Shah, M. A. U. Khan, S. A. Khayam, A. R. Sadeghi, R. Schmitz, "Breaking e-banking CAPTCHAs", Proceedings of the 26th Annual Computer Security Applications Conference, **2010,** pp. **171-180**.

[12] P.Baecher, N. Buscher, M. Fischlin and B. Milde, "Breaking reCAPTCHA: a holistic approach via shape recognition", IFIP advances in information and communication technology, **2011,** pp. **56-67**.

[13] C.Cruz-Perez, O. Starostenko, F. Uceda-Ponga, V. A. Aquino and L. Reyes-Cabrera, "Breaking reCAPTCHAs with unpredictable collapse: heuristic character segmentation and recognition", Springer, **2012,** pp. **155-165**.

[14] H.Gao, W. Wang, J. Qi, X. Wang, X. Liu, J. Yan, "The robustness of hollow CAPTCHAs", ACM conference on computer and communications, **2013,** pp. **1075-1086**.

[15] B.B. Zhu, J. Yan, Q. Li, C. Yang, J. Liu, N. Xu, M. Yi and K. Cai, "Attacks and design of image recognition CAPTCHAs", ACM conference on computer and communications security, **2010,** pp. **187-200**.

[16] C.J. Hernandez-Castro, A. Ribagorda, "Pitfalls in CAPTCHA design and implementation: the math CAPTCHA, a case study", Computers and Security, Elsevier Advanced Technology, **2010,** pp. **141-157**.

[17] J.Tam, J. Simsa, S. Hyde, L. von Ahn, "Breaking audio CAPTCHAs", Advances in Neural Information Processing Systems 21, **2008,** pp. **1625-1632**.

[18] E.Bursztein, R. Beauxis, H. Paskov, D. Perito, C. Fabry, J. C. Mitchell, "The failure of noise-based non-continuous audio CAPTCHAs", IEEE symposium on security and privacy, **2011,** pp. **19-31**.

AUTHORS PROFILE

Niket Kumar Choudhary is pursuing M.E. in Computer Engineering from Pimpri Chinchwad College of Engineering. He has completed B.E. in Information Technology in 2013. His area of interest is Network Security.

Prof. Sudarshan Deshmukh is a senior faculty at Pimpri Chinchwad College of Engineering. He has completed M.E. in Computer Engineering and B.E. in Information Technology. He has teaching experience of 11+ years. His area of interest is Distributed system and Parallel Architectures.