Cyber Security Using Arabic CAPTCHA Scheme

Bilal Khan[1], Khaled Alghathbar[1, 2], Muhammad Khurram Khan[1], Abdullah Alkelabi[2], and Abdulaziz Alajaji[2]
[1]Center of Excellence in Information Assurance, King Saud University, Saudi Arabia
[2]Department of Information Systems, King Saud University, Saudi Arabia

Abstract: *Bots are programs that crawl through the web site and make auto registrations. CAPTCHAs, using Latin script, are widely used to prevent automated bots from abusing online services on the World Wide Web. However, many of the existing English based CAPTCHAs have some inherent problems and cannot assure the security of these websites. This paper proposes a method that focuses on the use of Arabic script in the generation of CAPTCHA. The proposed scheme uses specific Arabic font types in CAPTCHA generation. Such CAPTCHA exploits the limitations of Arabic OCRs in reading Arabic text. The proposed scheme is beneficial in Arabic speaking countries and is very useful in protecting internet resources. A survey has been conducted to find the usability of the scheme, which was satisfactory. In addition, experiments were carried out to find the robustness of the scheme against OCR. The results were encouraging. Moreover, a comparative study of our CAPTCHA and Persian CAPTCHA scheme shows its advancement over Persian CAPTCHA.*

Keywords: *CAPTCHA, Arabic, automated-bots, cyber security, spam.*

Received July 11, 2010; accepted March 1, 2011

1. Introduction

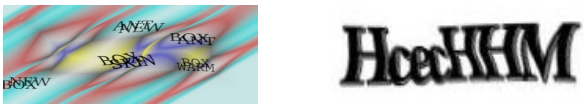
Internet is used by many people to avail different online services, e.g., email services, online shopping, blogs and other online memberships etc. In all of these services the user has to register itself by filling an online form. However, bots, poses as a human, sign up for these free services automatically. Preventing online services from these automated-bots is a challenge. is used to differentiate between human and bots. CAPTCHA stands for Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA). It is a form of HIP, which is used to prevent automated bots from abusing online services on the World Wide Web [17]. Yahoo, Google and many other websites uses different types of CAPTCHAs to secure their services from automated bots Figure 1. These CAPTCHAS are hard for the Optical Character Recognition (OCR), software used to read characters from the image, to be broken.

Some well-known types of CAPTCHAs are Gimpy and EZ-Gimpy [2]. Gimpy uses an image of seven words from the dictionary and generates a distorted image of the words to make it hard for the OCR to recognize Figure 2-a [11]. The image is then presented to the user. In order to be successful, the user has to read three words from the image and enter them.

a) Google b) yahoo

Figure 1. CAPTCHA used.

In contrast to Gimpy, EZ-Gimpy CAPTCHA is an image consisting of a single word [11]. The image is distorted with different techniques Figure 2-b to confuse OCR.



a) Gimpy CAPTCHA b) EZ-Gimpy CAPTCHA

Figure 2. An example.

In this paper, a novel approach is adopted towards securing web from automated bots, i.e., Arabic CAPTCHA is proposed. Due to the natural complexity of Arabic text, it is more resistant to optical character recognition than other languages, e.g., Latin [3, 4]. It is expected that the proposed CAPTCHA will be very useful in protecting a large amount of internet resources, being wasted by spam, in Arabic speaking countries where the trend of using internet is growing fast. According to the World Internet Stats, there are more than twenty countries where Arabic is spoken as a first language [9]. These countries are listed in Table 1, including Iran where Persian is spoken a language that uses Arabic script. Table 1 shows the usage of internet in these countries in 2003 and 2010 [9]. It is evident from Table 1 that there is an increase in the number of internet users from 2003 to 2010. As shown in Figure 3, Iran is on the top where the number of internet users exceeds 32.20million in 2010, followed by Egypt and Morocco where these Figures are reaching to 12.50million and 10.30million, respectively.

Table 1. Internet usage in Arabic speaking countries.

Serial #	Country	Internet Usage, in Dec/2000	Internet Usage, Latest Data (2009)
1	Iran	250,000	32,200,000
2	Egypt	450,000	12,568,900
3	Morocco	100,000	10,300,000
4	Saudi Arabia	200,000	7,761,800
5	Sudan	30,000	4,200,000
6	Algeria	50,000	4,100,000
7	Syria	30,000	3,565,000
8	UAE	735,000	3,558,000
9	Tunisia	100,000	2,800,000
10	Jordan	127,300	1,595,200
11	Kuwait	150,000	1,000,000
12	Lebanon	300,000	945,000
13	Oman	90,000	557,000
14	Qatar	30,000	436,000
15	Bahrain	40,000	402,900
16	Yemen	15,000	370,000
17	Palestine (West Bk.)	35,000	355,500
18	Libya	10,000	323,000
19	Iraq	12,500	300,000
20	Eritrea	5,000	200,000
21	Mauritania	5,000	60,000

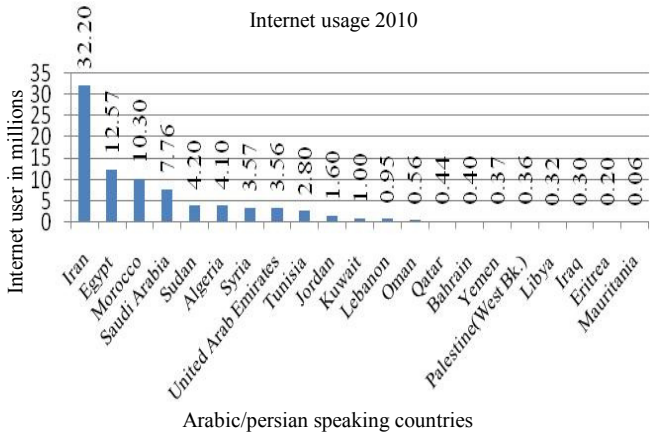


Figure 3. Graph showing the internet usage in 2010 in Arabic/Persian speaking countries.

Apart from Arabic speaking regions, Arabic script is used in other languages, spoken by people in different parts of the world. These languages are Persian, Urdu, Pashto, Sindhi, Punjabi, which are spoken in different parts of Asia. A majority of these people are the users of the internet.

In addition to Yahoo, Google, MSN and Instant Messengers, there are thousands of websites that provides services in Arabic language to the Arabic speaking users. These include websites from educational institutions, government organizations, online shopping, non/semi government organizations and social networking websites. In many of these websites, the user has to register itself before she/he uses the services provided by these websites. Many online service providers will take interest in providing online services with our proposed security mechanism, hence optimizing the potential of internet in these countries. The proposed scheme not only prevents automated-bots but also facilitate the user while interacting the authentication process.

The rest of the paper is organized as follows: section 2 is related work. Section 3 describes the proposed methodology and explains the characteristics of Arabic script, weaknesses of Arabic OCRs, implementation, security and usability of the proposed Arabic CAPTCHA. Finally, section 4 concludes the paper.

2. Related Works

The idea of CAPTCHA was first practically implemented by AltaVista to prevent automated-bots from automatically registering the web sites [10]. The mechanism behind the idea was to generate slightly distorted characters and to present it to the user.

Gupta *et al.* [7], proposed the method of embedding numbers in text CAPTCHAs. They state that the tagged numbers in the text confuses the OCR and make it unable to segment the characters.

Elson *et al.* [6], proposed a method in which the user has to select the image of a cat from the collection of images of cats and dogs. For the image collection they used the database of more than 3 million photos from a website used to find homes for homeless pets, i.e., petfinder.com. Ahn gave the idea of automated Turing test that is not based on the difficulty of the OCR, rather on hard artificial intelligence problems [1]. Their proposed method uses hard AI problems to construct CAPTCHAs in order to differentiate humans from computers.

Although, securing web from automated-bots has been given a strong attention in the research community, however, the emphasis is mostly on the CAPTCHAs using Latin script which has many inherent weaknesses in it. CAPTCHA implementation that uses other language has been ignored by the research community except [14, 15] that use Persian language. Therefore, we consider our work a significance progress in the field of web security using CAPTCHAs based on Arabic script. This paper presents a CAPTCHA scheme based on the Arabic text. The Arabic text is presented as image and distorted with different techniques so that it is unbreakable by OCRs, but still easily readable by humans.

3. Proposed Methodology

In this section, some of the features of the Arabic script are described briefly followed by the limitations of Arabic OCR. Then keeping in view the features of Arabic script and limitations of Arabic OCR, the proposed scheme is explained.

3.1. Characteristics of Arabic Script

Arabic script has some special characteristics which do not exist in others, e.g., Latin, Chinese, and Japanese. Arabic writing is someway like handwritten Latin [18]

and it has 28 letters. It is written from right to left shown in Figure 4, whereas Arabic numbers are written from left to right. A word consists of different number of characters. Some characters are not connectable from the left with the succeeding character. For example, 'ر' in Figure 4 is not connectable.

جدّي يعمل في التجارة

Figure 4. A sample of Arabic text.

The Arabic letters have different shapes depending on its position in the word, i.e., initial, middle, final or isolated. In contrast to Latin characters, Arabic characters have different sizes [12]. Moreover, several characters have different number of dots as well as different positions of dots, i.e., above, below and in the middle of the character shown in Figure 5. In addition, diacritics, e.g., shadda, maddah, hamza etc., are used in Arabic text [18].

ز ج ب ش

Figure 5. Characters having different positions and number of dots.

3.2. Weaknesses of Arabic OCR

In contrast to Latin script, the recognition of Arabic characters is relatively more difficult due to its inherent characteristics. The crucial step in recognizing Arabic characters is the segmentation of word into separate characters [18]. Varying sizes of the characters is one of the reasons that make the segmentation and recognition hard. Baseline detection is also a problem that makes the segmentation step difficult in machine printed as well as handwritten Arabic text [5].

Arabic OCRs are font dependent. They have been developed for a few standard Arabic font types and unable to recognize text which is written in different font types. Especially, the font types in which the characters or sub words are overlapped, poses a serious problem in the segmentation stage. There are many Arabic characters that have similar shapes e.g., ب ت, ض ص, غ ع, خ ح ج, ظ ط, ذ ر ز و, ق ف. The differences between these pairs of letters are the number and position of dots. These similar characters and dots confuse OCR to recognize characters correctly [13]. The proposed scheme is developed keeping in view the weaknesses of the state of the art Arabic OCRs.

3.3. CAPTCHA Implementation

This section explains the implementation and algorithmic details of our CAPTCHA scheme. Several methods have been developed in our program to make it secure. Figure 6 shows the different phases of implementation in the form of a flowchart. Basically our CAPTCHA scheme consists of two types:

1. Arabic letters and Arabic font types Table 2.
2. The one which is randomly selected from a dictionary of 271000 words.

Table 2. Arabic font types used in the proposed scheme.

Serial #	Font Name	Serial #	Font Name
1	Al-Hadith1	27	Hesham Free
2	Al-Homam	28	Hesham Ghorn Italic
3	Al-Kharashi 1	29	Hesham Gornata
4	Al-Kharashi	30	Hesham Kashkool
5	Al-Kharashi	31	MCS Hashimy S_I normal
6	Al-Kharashi	32	MCS Hashimy S_U normal
7	Al-Kharashi	33	MCS Hor 1 S_U Normal 2000
8	Al-Kharashi	34	MCS Hor 1 S_U Snail 2000
9	Al-Kharashi Diwani	35	MCS Hor 4 S_U Bite 2000
10	Al-Mujahed Al-Anbobi	36	MCS Madina E_I 3D
11	Al-Mujahed Free	37	MCS Madina E_U 3D
12	ALmusam_free	38	MCS Madinah E_U normal
13	DecoType Naskh Special	39	MCS Madinah S_I normal
14	DecoType Naskh Swashes	40	MCS Rika S_I normal
15	DecoType Naskh Variants	41	MCS Rikaa E_U 3D
16	DecoType Thuluth	42	MCS Taybah S_U rose
17	FreeHand	43	MCS TOPAZ
18	FS_Graphic	44	MCS Wadiy E_U normal
19	FS_India	45	MCS Wadiy S_U normal
20	FS_Shehab_Points	46	Monotype Koufi
21	FS_Shehab_Shatter	47	PT Bold Broken
22	FS_Strip	48	PT Bold Dusky
23	Hesham AlSharq	49	PT Bold Mirror
24	Hesham Bold	50	PT Bold Stars
25	Hesham Fostat	51	PT Simple Bold Ruled
26	Simple Indust Shaded	52	Sidon

In addition to types of CAPTCHA, the proposed program supports different levels of CAPTCHA complexity. There are three levels of CAPTCHA complexity; they are easy, medium and hard. These levels differ in the number of Arabic characters appearing in the CAPTCHA, types of Arabic characters, font types and background noise. In easy level CAPTCHA, the number of characters in a word are ranging from 4 to 5, easy font types which are more easily read by the human and less distortion are employed. In medium level, a CAPTCHA has 5 to 6 numbers of Arabic characters, a little more background distortion in the form of dots and lines.

In contrast to easy and medium level CAPTCHAs, hard level CAPTCHA employs the number of characters ranging from 7 to 9, more complex Arabic characters and high intensity of distortion in the form of arcs, lines and dots. The purpose of these levels of CAPTCHA complexity is to protect the system from bots. Initially the user is presented with easy level CAPTCHA. In case the requests are generated from bots, program gradually increases the CAPTCHA complexity level. Algorithms 1 and 2 illustrate the

implementation of different levels of CAPTCHA complexity.

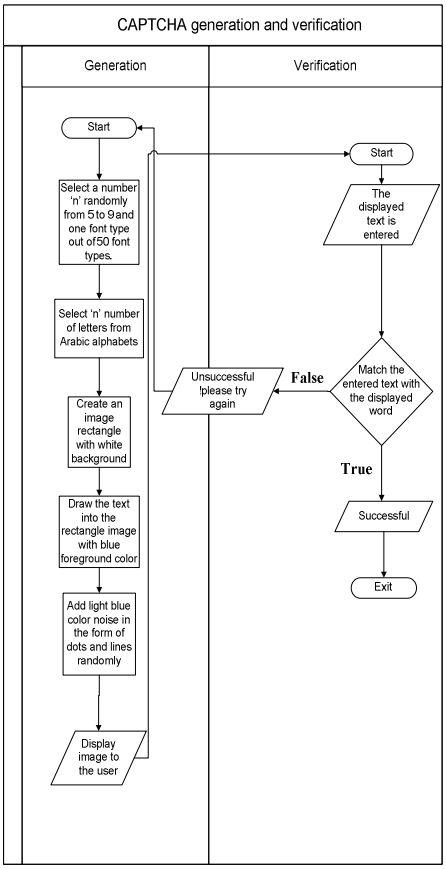


Figure 6. Flowchart for the CAPTCHA generation and verification.

Algorithm 1: Text generation
Input: level of CAPTCHA complexity, Type of CAPTCHA
Output: text generation for CAPTCHA (String)
1. *Procedure*
2. *Text_length=select a number randomly in the range of 4 to 9.*
3. *If text_length=4 or 5 then*
4. *Select CAPTCHA level as easy*
5. *Else if text_length=6 or 7 then*
6. *Select CAPTCHA level as medium*
7. *Else if text_length=8 or 9 then*
8. *Select CAPTCHA level has hard*
9. *End if*
10. *If CAPTCHA level=easy then*
11. *For a=1 to text_length do*
12. *Select letters from a collection of already specified easy letters*
13. *End for*
14. *Else If CAPTCHA level=medium then*
15. *For a=1 to text_length do*
16. *Select letters from the collection of already specified letters*
17. *End for*
18. *Else If CAPTCHA level=hard then*
19. *For a=1 to text_length do*
20. *Select letters from 28 Arabic alphabets in addition to hard letters like ء, ئ, ؤ, ! and ~ etc.*
21. *End for*
22. *End if*
23. *End procedure*

Algorithm 2: Image generation with white background and blue foreground text.
Input: text string
Output: image generation
1. *Procedure*
2. *A fixed size rectangle is selected*
3. *An image is generated within the rectangle*
4. *Drawobject.backgroundcolor=white*
5. *Randomly generate four points (P1, P2, P3, P4) with random horizontal and vertical coordinates in the image.*
6. *Select one font type out of 50 Arabic font types depending on the CAPTCHA complexity level*
7. *Draw the input text within four selected points (coordinates)*
8. *If CAPTCHA complexity level=easy then*
9. *Size of text image=60-70% of the whole image else if CAPTCHA complexity level=medium then size of text image=50-59% of the whole image else if CAPTCHA complexity level=hard then size of text image=40-49% of the whole image size*
10. *End if*
11. *drawobject.foregoudcolor=blue*
12. *end procedure*

Algorithm 3: Deforming the CAPTCHA image.
Input: Level of CAPTCHA complexity, image with white background and blue foreground text
Output: deformed image with dots and lines
1. *Procedure*
2. *IF(Level of CAPTCHA complexity is easy) then*
3. *L=10*
4. *For a=1 to L do*
5. *Drawobject.color=light blue*
6. *Select two points with different horizontal and vertical axis*
7. *Drawobject.line*
8. *End for*
9. *m=randomly select a number between 1200 to 1300*
10. *For a=1 to m do*
11. *Drawobject.color=light blue.*
12. *Drawobject.noise=dots*
13. *End for*
14. *IF level of CAPTCHA complexity=Medium then*
15. *n=10*
16. *For a=1 to n do*
17. *Drawobject.color=light blue*
18. *Select two points with different horizontal and vertical axis*
19. *Drawobject.line*
20. *Drawobject.Arc*
21. *End for*
22. *m=randomly select a number between 1300 to 1400*
23. *For a=1 to m do*
24. *Drawobject.color=light blue.*
25. *Drawobject.noise=dots*
26. *End for*
27. *IF level of CAPTCHA complexity=hard then*
28. *n=15*
29. *For a=1 to n do*
30. *Drawobject.color=light blue*
31. *Select two points with different horizontal and vertical axis*
32. *Drawobject.line*
33. *Drawobject.Arc*

34. End for
35. m=randomly select a number between 1400 to 1500
36. For a=1 to m do
37. Drawobject.color=light blue
38. Drawobject.noise=dots
39. End for
40. End procedure

Algorithm 4: Provide the User with a New CAPTCHA
Input: level of CAPTCHA complexity, type of CAPTCHA
Output: CAPTCHA image display
1. *Procedure*
2. *Text=Call Text generation(level, type)*
3. *Image=Call Image generation(text)*
4. *DeformedImage=Call deformedimage(Image)*
5. *TimeStamp=System Time*
6. *ChID=Generates new Challenge ID and Store it in DB along with the TimeStamp as one record*
7. *ChallengeBlock=Encrypt(Text, User IP, TimeStamp, ChID)*
8. *Prints Deformed Image, ChallengeBlock*
9. *End procedure*

Algorithm 5: Validating CAPTCHA
Input: User Input, Challenge, User IP
Output:
1. *Procedure:*
2. *Validate Inputs and Check if all inputs is given*
3. *Block=Decrypt the ChallengeBlock*
4. *CurrentTime=System Time*
5. *If CurrentTime-Block. TimeStamp>5Minutes Or record (Block. ChID, Block. TimeStamp) is not Found in record Or UserIP≠Block.UserIP*
6. *Or Block.Text≠User Input then*
7. *Return false*
8. *Else delete record(Block. ChID, Block. TimeStamp) and*
9. *Return True*
10. *End If*
11. *End procedure*

3.3.1. Mechanism Against Brute Force and Replay Attack

The algorithm has the ability to protect the system from the brute force attack. The system keeps record of the IP addresses from where the CAPTCHA requests generate. If the requests are more than 100per minute then it increases the complexity level of CAPTCHA gradually from easy to medium and from medium to hard. If the requests are still coming and they are more than 1000 per minute then the requests are blocked from that IP address for next 24hours.

In addition to the brute force attack, the system also provides defense against the replay attack. The CAPTCHA server sends the challenge to the user along with the timestamp. The server expects the response from the user before the timestamp expires. If response is received before timestamp expires, the challenge ID is deleted from log, so that it is not used again Algorithm 5. This way the system is protected from the replay attack. The following steps are involved for the secure communication between end user, CAPTCHA server, and application server Figure 7.

1. User loads web page and request service from webapp.
2. Webapp provides a form to be filled and API key.
3. User request CAPTCHA from API, providing API key.
4. API server provides, CAPTCHA text in the form of image, in addition to a challenge, i.e., timestamp, CAPTCHA text, IP, API key, domain, challenge ID in encrypted with 256bit key using AES encryption method.
5. User submits the filled form, encrypted challenge and response to the challenge to the webapp.
6. Webapp further forwards it to the API server, where the API server decrypts the challenge and compare it with the information it already has and return the result to the webapp server.
7. If the response is true, the user is granted access to use the requested service, i.e., registering for email service, wikis, blogs etc., else the user can request the service again.

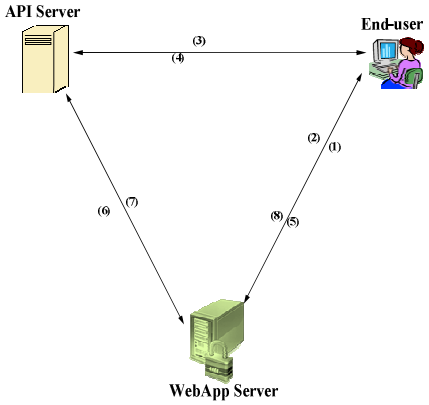


Figure 7. Secure communication between end user, API server and application server.

3.4. Robustness of the Proposed Arabic CAPTCHA Scheme

This section explains the features of our proposed and developed CAPTCHA scheme. It shows how the CAPTCHA scheme has been designed to make it hard for the OCR to read. The proposed CAPTCHA is designed in such a way that it makes the different stages of CAPTCHA solving hard, namely preprocessing, segmentation and character recognition. Figure 8 shows the sample of an image generated by our program. The salt and pepper noise is added in the image. Such background clutter increases problems for the CAPTCHA solver during preprocessing, as diacritics, e.g., 'ﻋ', dots etc., will be removed in the noise removal process. Noise is added to the image to hinder the segmentation and character recognition process.

The background color is white with light blue color noise whereas foreground color is blue. Such color combination looks good and is hard for the OCR to separate noise from the image.

Figure 8. A sample CAPTCHA with diacritics.

The text image shown in Figure 9 uses font PT Bold Mirror, one of the font types used in our scheme that complicates the job for OCR. Because Figure shows that the original text in the image has its duplicate copy in the form of shadow. As opposed to human user, the OCR will detect two words in the image instead of one. Such feature confuses OCR in recognizing the characters of the word.

Figure 9. A sample of CAPTCHA using PT bold mirror font.

Character overlapping in text is a good feature which makes the segmentation step hard for the OCR. The program generates words with overlapping features. For example in Figure 10, the letter Zai 'ز' is overlapped with the letter Lam 'ل'. In such situation, the OCR is unable to separate the two overlapped words and hence is unable to recognize the character in the image.

Figure 10. Overlapping encircled in our generated CAPTCHA.

Baseline detection helps the CAPTCHA solver to segment the word. Some techniques have been used in our CAPTCHA generation scheme to make the baseline detection difficult. For example in Figure 11, the irregular positions of letters make it hard to detect the baseline. Moreover, the scheme generates a unique CAPTCHA each time, which avoids the possibility of making database of the images by spammers.

Figure 11. A sample CAPTCHA with baseline detection problem.

In addition to font type, size and noise, another factor is the position of the word that varies from image to image. Such randomly placed words in CAPTCHAs make the segmentation and recognition hard [8].

3.5. Performance Evaluation

In this section we evaluate the robustness of our CAPTCHA scheme. To find out whether OCR recognizes the text in image correctly, the image has to be preprocessed to remove the background noise. Suppose the total number of characters in CAPTCHA text 'T' are 'm'. Within these m characters, I is the set of characters carrying at least one dot. So the set of characters carrying dots can be represented as a subset of {1, 2, 3….m}, i.e., for any $I \subseteq \{1, 2 \ldots m\}$. And the total number of dots in text is n, where $n \geq 1$.

Let E_I be the event that I is the set of characters with dots, i.e., $1 \leq I \geq m$. Since CAPTCHA image has background noise, in order to recognize the text correctly, it has to be removed through preprocessing. Images have been preprocessed in matlab Table 3. It is observed that in addition to noise, genuine dots have been removed during the preprocessing stage. In order for the OCR to recognize the letters correctly it has to place dots on the characters.

Let A_d be the event that OCR places these dots on the characters and the number of these dots is 'k'. Thus the event A_d corresponds to a false positive. Now we want to evaluate $\rho(A_d)$ by using conditional probabilities:

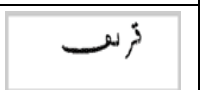
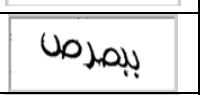
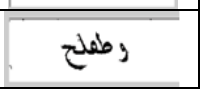
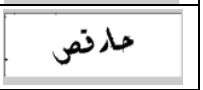
$$\rho(A_d) = \sum_{I \subseteq m} \rho(A_d \mid E_I).\rho(E_I)$$

Where

$$\rho(A_d \mid E_I) = \frac{\rho(A_d \cap E_I)}{\rho(A_d)}$$

The false positive here is the false indication by OCR that the dot has been placed correctly, which is in fact incorrect. In this case, CAPTCHA system needs 100% accurate response from the user, therefore the threshold value is 0. In case the OCR submits response to the CAPTCHA challenge and it has a dot placed on the wrong character or in the wrong position considering it a right place, then it will be false positive and will be rejected by the CAPTCHA system.

Table 3. CAPTCHA words before and after preprocessing.

Original Word	After Preprocessing
	
	
	
	
	

3.6. Usability of The Proposed CAPTCHA Scheme

The proposed algorithm, which is discussed in detail in section 3.3, has been implemented in VB.Net. The

efficiency of any CAPTCHA scheme can be analyzed in two ways, i.e., readability and robustness. If a CAPTCHA is robust enough that OCR cannot break it but it is not human readable then it is not useful. Therefore, for the CAPTCHA to be useful, it should be human readable. It is observed that the increasing complexity in a CAPTCHA to make it secure reduces its readability. In previous section, the robustness of the proposed CAPTCHA was analyzed against the abilities of Arabic OCR. This section describes the results of a survey that was conducted mainly to analyze the readability of our proposed CAPTCHA [16]. Over one hundred and fifty individuals participated in the survey from different professional backgrounds, both male and female, and their ages ranging from 17 to over 50 years.

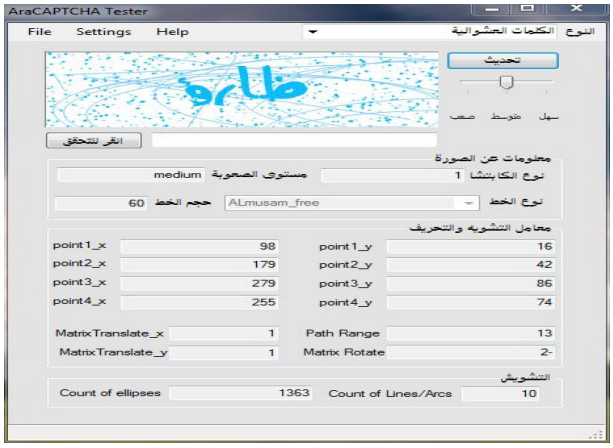


Figure 12. Arabic CAPTCHA generation with various parameters.

Each user was good in speaking, reading and writing Arabic language who belonged to Arabic speaking countries as well as South Asian countries. Fifty five different CAPTCHA samples were generated using our developed program shown in Figure 12, each having different font types, sizes and background clutter as well as with different levels of complexity. Each participant attempted to solve all 55 CAPTCHA samples on www.araCAPTCHA.com/survey [16].

1. The results of the survey were analyzed in two ways: with respect to font types.
2. With respect to characters.

3.6.1. Analysis with Respect to Font Types

From the analysis of the survey, it was found that the users were comfortable in reading the text images in 50 out of 52 font types, even with background noise, obfuscation, deformation and distortion in the image. Some images in specific font types were very easy for the users to read. The rate of reading for images in those font types was 100%. Table 4 lists these font types. These font types are easy for the individuals to read but hard for the OCR to solve. For example, image in Figure 13 is easy for the human to solve, but hard for the OCR to read.

Table 4. Most readable and less readable font types.

Most Readable Font Types	Less Readable Font Types
MCS Madina E_U 3D	Hesham Alsharq
PT Bold Dusky	MCS Hashimy S_I Normal
DecoType Naskh Special	n/a
MCS Wadiy E_U normal	n/a
Al-Mujahed Al-Anbobi	n/a
PT Simple Bold Ruled	n/a
Monotype Koufi	n/a
FS_Graphic	n/a
MCS Rika S_I normal.	n/a
FS_Shehab_Points	n/a

Figure 13. Font type PT bold mirror, human readable but hard for the OCR to solve.

There are only two font types for which the human readability rate was very low ranging from 0 to 30 %. Although reading CAPTCHA images in these font types is hard for the OCR, but it is also reduces human readability rate shown in Figure 14. However, the number of those font types is only 2 and they are Hesham Alsharq and MCS Hashimy S_I Normal Table 4.

Figure 14. Hard for the OCR to solve but human unreadable.

Despite the fact that these two font types were hard to read, the overall readability rate of the 52 font types remained high, i.e., 99.6%.

3.6.2. Analysis with Respect to Characters

The survey results were also analyzed with respect to characters. The regular 28 Arabic alphabet characters in addition to other characters were easy to read for all participants irrespective of the font type in which they were used. Some of the characters were hard to read, and they were replaced by the readers with similar looking regular characters while solving the CAPTCHA. Table 5 shows the genuine characters which were replaced with the similar looking characters. The results found from the survey are satisfactory as compared to English based CAPTCHAs, in which the image is obfuscated to make it unbreakable by OCR but makes it hard for the human to read.

Table 5. Characters those were confusing to the reader.

Original Letter	Replaced by
إ	ا
ك	خ
د	ر
غ	ف
د	ر
ش	ن
ص	م
ض	ح
ح	ج
ي	ب
ؤ	و
ق	ت
ن	ز
ض	ح
ظ	ط

3.7. Comparative Study of Our CAPTCHA (Arabic) and Persian CAPTCHA Schemes

Table 6 compares the features of our CAPTCHA scheme with the Persian CAPTCHA [14] and shows its advancement over the Persian CAPTCHA. In contrast to our CAPTCHA scheme, Persian CAPTCHA has some security weaknesses. For example, it uses only one font type, i.e., Naskh, which is vulnerable to Persian OCR whereas our CAPTCHA scheme uses 52 font types randomly. In their scheme, noise is added in the form of lines which is good but not enough. In addition, the background and foreground color is same which reduces the human readability rate. Moreover, the position of CAPTCHA text is static, i.e., center, whereas in our CAPTCHA it varies randomly.

Table 6. Comparison of features of Arabic schemes (ours) and persian CAPTCHA scheme.

Features	Arabic Scheme	Persian Scheme [14, 15]
Number of Font Types	50	1
Variation of Font Types	Randomly	n/a
Variation of Font Size	Randomly	n/a
Lines (as Background Noise)	Randomly	Randomly
Dots (as Background Noise)	Randomly	n/a
Use of Dictionary Words	n/a	n/a
Number of Letters	5 to 9	3 to 9
Baseline Detection by OCR	Not Possible	Possible
Text Coordinates	Varies randomly	Static (center)
Background and Foreground Color	Different	Same

4. Conclusions and Future Work

In this paper, a novel CAPTCHA scheme is proposed. The proposed scheme uses Arabic script to generate an image. The image is distorted by adding various types of noises in the background in the form of dots, lines and arcs. The background and foreground colors are selected so that the overall CAPTCHA image is attractive for the user. The varying number of characters, font types and font sizes make it extremely hard for the OCR to read our CAPTCHA. The algorithm is efficient and the user does not have any problem while interacting with the system. To evaluate the readability rate of CAPTCHA images, a survey was conducted consisted of over one hundred and fifty individuals. Those survey participants were from Arabic speaking countries as well as non-Arabic South Asian countries who can understand the Arabic script. It was found from the survey that the overall readability rate of the images was high. So our proposed CAPTCHA scheme can be used in non-Arabic speaking countries where languages use Arabic script such as Urdu, Pashto and Persian etc., Some experiments were conducted to find out the robustness of the CAPTCHA that were encouraging.

References

[1] Ahn L., Blum M., Hopper N., and Langford J., "CAPTCHA: Using Hard AI Problems for Security," *in proceedings of the 22nd international conference on Theory and applications of cryptographic techniques (Eurocrypt)*, Heidelberg, vol. 2656, pp. 294-311, 2003.

[2] Ahn L., "Telling Humans and Computers Apart or How Lazy Cryptographers Do AI," *Communications of the ACM*, vol. 47, no. 2, pp. 57-60, 2004.

[3] Al-muhtaseb H., Mahmoud S., and Qahwajib R., "Recognition of Off-Line Printed Arabic Text Using Hidden Markov Models," *Journal of Signal Processing*, vol. 88, no. 12, pp. 2902-2912, 2008.

[4] Abdulla S., Al-nassiri A., and Salam R., "Off-Line Arabic Handwritten Word Segmentation Using Rotational Invariant Segments Features," *International Arab Journal of Information Technology*, vol. 5, no. 2, pp. 200-208, 2008.

[5] Al-shatnavi A. and Omar K., "A Comparative Study Between Methods of Arabic Baseline Detection," *in proceedings of International Conference on Electrical Engineering and Informatics*, Malaysia, pp.73-77, 2009.

[6] Elson J., Douceur J., and Saul J., "Asirra: A CAPTCHA that Exploits Interest-Aligned Manual Image Categorization," *in Proceedings of the 14th ACM Conference on Computer and Communications Security*, Virginia USA, pp. 366-374, 2007.

[7] Gupta A., Jain A., Raj A., and Jain A., "Sequenced Tagged CAPTCHA: Generation and its Analysis," *in Proceedings of International Advance Computing Conference*, India, pp. 1286-1291, 2009.

[8] Hindle A., Godfrey M., and Holt R., "Reverse Engineering CAPTCHAs," *in Proceedings of the 15th Working Conference on Reverse Engineering*, USA, pp. 59-68, 2008.

[9] Internet Usage in the Middle East, Source Online, available at: http://internetworldstats.com/stats5.htm, last visited 2010.
[10] Lillibridge M., Abadi M., Bharat K., and Broder A., "Method for Selectively Restricting Access to Computer Systems," United States Patent 6195698. Applied 1998 and Approved 2001.
[11] Mori G. and Malik J., "Recognizing Objects in Adversarial Clutter: Breaking A Visual CAPTCHA," *in Proceedings of 2003 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 134-141, 2003.
[12] Moussaa S., Zahourb A., Abdelhafid A., and Alimi A., "New Features Using Fractal Multi-Dimensions for Generalized Arabic Font Recognition," *Journal of Pattern Recognition Letters*, vol. 31, no. 5, pp. 361-371, 2010.
[13] Sattar S., Haque S., Pathan M., and Gee Q., "Implementation Challenges for Nastaliq Character Recognition," *in Proceedings of International Multi Topic Conference (IMTIC)*, Pakistan, pp. 279-285, 2008.
[14] Shirali-shahreza M. and Shirali-Shahreza M., "Persian/Arabic Baffle Text CAPTCHA," *Journal of Universal Computer Science*, vol. 12, no. 12, pp. 1783-1796, 2006.
[15] Shirali-shahreza M. and Shirali-Shahreza M., "Advanced Nastaliq CAPTCHA," *in Proceedings of 7th IEEE International Conference on Cybernetic Intelligent Systems*, UK, pp. 1-3, 2008.
[16] Survey Website for Arabic CAPTCHA, available at: http://www.aracaptcha.com/survey, last visited 2010.
[17] Thomas A., Rusu A., and Govindaraju V., "Synthetic Handwritten CAPTCHAs," *Journal of New Frontiers on Handwriting Recognition*, vol. 42, no. 12, pp. 3365-3373, 2009.
[18] Zheng L., Hassin A., and Tang X., "A new Algorithm for Machine Printed Arabic Character Segmentation," *Journal of Pattern Recognition Letter*, vol. 25, no. 15, pp.1723-1729, 2004.

Bilal Khan is working as a researcher at Center of Excellence in Information Assurance, King Saud University, Saudi Arabia. Received his MSc in Internet, Computer and system security from University of Bradford, UK. He has several journal and conference papers. His research interests include cyber security and information security management.

Khaled Alghathbar, PhD, CISSP, CISM, PMP, BS7799 lead auditor, is an associate professor and the director of the Centre of Excellence in Information Assurance in King Saud University, Saudi Arabia. He is a security advisor for several government agencies. His main research interest is in information security management, policies, biometrics and design. He received his PhD in Information Technology from George Mason University, USA.

Muhammad Khurram Khan is currently working as an associate professor and R & D Manager at Center of Excellence in Information Assurance, King Saud University, Saudi Arabia. He is the founding editor of Bahria University Journal of Information and Communication Technology. He is the editorial board of several international journals. He also plays role of guest editor of several international journals of Springer-Verlag and Elsevier Science. He is an active reviewer of many international journals. He has been included in the Marquis Who's Who in the World 2010 edition. He was recently awarded a certificate of appreciation for outstanding contributions in Biometrics and Information Security Research, AIT Conference. He has also secured an outstanding leadership award at IEEE international conference on Networks and Systems Security 2009, Australia. He has published more than 90 research papers. His areas of interest are biometrics, multimedia security, digital data hiding, and authentication protocols.

Abdullah Alkelabi is the IT risk assessment officer at Alinma Bank, Saudi Arabia. He received his BSc degree in computer information systems from King Saud University, Saudi Arabia.

Abdulaziz Alajaji is working as a teaching assistant in College of Computer Sciences and Information Systems, King Saud University, Saudi Arabia. He received his BSc in Information Systems from King Saud University, Saudi Arabia. His main research interest is in information security technologies.