

A SURVEY AND ANALYSIS OF CURRENT CAPTCHA APPROACHES

NARGES ROSHANBIN and JAMES MILLER

University of Alberta
roshanbi@ualberta.ca, jm@ece.ualberta.ca

Received January 9, 2012
Revised December 7, 2012

Computer programs are misusing Internet services designed for humans. A CAPTCHA, Completely Automated Public Turing test to tell Computers and Humans Apart, is a standard security mechanism to defend against such attacks. Two fundamental issues with CAPTCHAs are usability and robustness. It is important for a CAPTCHA to be both legible for humans and strong against malicious computer programs. Recently, computer vision and pattern recognition algorithms have broken many well-known CAPTCHAs. Lack of security and usability in CAPTCHAs designed to protect popular websites such as Gmail and Yahoo mail, with almost 500 million users in July 2011, would cause huge problems. Therefore, security researchers have become motivated to discover techniques to improve CAPTCHAs. Exploiting the gap in the recognition abilities between humans and computers is a key point to design a CAPTCHA that is hard-to-break for machines but easy-to-solve for humans.

In this paper, we introduce current CAPTCHAs and attacks against them; we investigate the robustness and usability of current CAPTCHAs and discuss ideas to develop more robust and usable CAPTCHAs.

Keywords: Web information systems, Security, CAPTCHAs

Communicated by: G.-J. Houben & E. Mendes

1 Introduction

A CAPTCHA, which stands for "Completely Automated Public Turing Test to Tell Computers and Humans Apart", is a test that can distinguish human users from computers/robots. In other words, it is a test, which most humans can pass, but computer programs cannot. Such tests are usually based on hard, open artificial intelligence problems such as the recognition of distorted text. The idea of a CAPTCHA comes from "Turing test", but it is sometimes described as a "reverse Turing test". The reason is that differing from the original Turing test, which is administered by a human and targeted to a machine; CAPTCHA challenges are automatically generated and graded by a computer. Moreover, the goal of designing a CAPTCHA is to distinguish, rather than to fail to distinguish, which is the main purpose of Turing tests. Another difference between a CAPTCHA and a Turing test is that the former was designed to act as a measure of progress for AI; however, the latter is a security mechanism.

The P for Public means that the code and the data used by a CAPTCHA should be publicly available. Thus a program that can generate and grade tests that distinguish humans from computers, but whose code or data are private, is not a CAPTCHA [1]. It is important for the challenges to be substantially different most of the times; otherwise, they might be recorded, solved by humans, and then used to answer future challenges. Thus, they should be generated pseudo randomly from a very large space of distinct challenges [2].

2 A Survey and Analysis of Current CAPTCHA Approaches

Using a CAPTCHA as a security mechanism is very important because it can prevent malicious programs from signing up for thousands of accounts, posting hundreds of comments in weblogs and so on. The most common bot programs include [3]:

- Voting bots could cast thousands of votes as masquerading humans in online polls.
- Email account registration bots, which could sign up for a lot of email accounts with free email service providers;
- Email spam bots, which could automatically send out thousands of spam messages;
- Weblog bots, which could post comments in weblogs pointing both readers and search engines to irrelevant sites;
- Search engine bots, which could automatically register web pages to raise their rankings in a search engine;
- Chat room bots [4].

The remainder of this article is organized as follows. Section 2 provides an overview of several CAPTCHAs which have appeared in the literature. Sections 3 and 4 discuss the robustness and the usability of these CAPTCHAs, respectively. Section 5 describes mechanisms or methods to break the existing CAPTCHAs. Finally, in section 6, based on the discussions in previous sections, existing CAPTCHA types, attack types and CAPTCHA vulnerabilities are summarized. We conclude by proposing a new CAPTCHA scheme to address these vulnerabilities and explore future research directions.

2 Current CAPTCHAs

In general, CAPTCHA methods can be divided into five groups:

- Text-based CAPTCHAs,
- Image-based CAPTCHAs,
- Audio-based CAPTCHAs,
- Motion-based CAPTCHAs, and
- Hybrid CAPTCHAs.

In text-based systems, distorted versions of characters of a word rendered as an image and are presented to the user. Then, the users are asked to type the answer that requires identifying all characters in the correct order. Because the image contains visual effects, it is difficult for a computer to recognize the words. Text-based CAPTCHAs have the weakness of being deciphered by OCR software. In order to overcome this weak point, other types of CAPTCHAs have been introduced. Image-based CAPTCHAs usually use the superiority of humans over computer vision systems in identifying the type of an object in an image. Although it is more convenient for the human to solve image-based CAPTCHAs rather than text-based ones, image-based CAPTCHAs have the difficulty of needing a large storage space.

An audio-based CAPTCHA picks a string, renders it to a sound clip and presents it to the users who are asked to recognize the contents of the audio clip. According to a large scale study on the usability of CAPTCHAs, audio-based CAPTCHAs are more problematic than other types [5]. Another category is motion-based CAPTCHAs in which a movie or animation is presented to the users and they are asked to recognize an action, animated word or image in the movie. This CAPTCHA is convenient for users. In addition, since the required processing time in this CAPTCHA is relatively high, it is more secure. However, the high loading time can be a disadvantage from a usability viewpoint. Another disadvantage is requiring a large database of animations. Finally, the term "hybrid CAPTCHA" has been selected for a CAPTCHA that is a combination of different types or designed for special purposes.

The remainder of this section introduces almost all of the CAPTCHAs that appear in the literature. The order of their introduction is not the same as the chronological order of their development. Introducing a CAPTCHA in a specific group does not mean that it does not have any features in common with other groups. Sometimes a CAPTCHA can belong to two or more different categories.

Some of the introduced CAPTCHAs are highly developed and work in the real world; others are just ideas brief sketched out by their authors. Regardless of their maturity, we present the basic idea behind each CAPTCHA in an effort to illustrate the wide variety of ideas and directions researchers are actively pursuing. Later sections will address the contribution, usability and robustness of these systems. Clearly, CAPTCHAs, which are presented in the literature as just brief ideas, will receive less attention in these later sections than their full-formed cousins.

2.1 Text-based CAPTCHAs

A text-based CAPTCHA is a distorted image of a sequence of characters on which different types of degradations, background clutters and color mixtures are applied to make it more challenging for attackers. We will introduce current text-based CAPTCHAs in six sub-groups:

- CAPTCHAs with “English words” as their CAPTCHA text,
- CAPTCHAs with “random strings” as their CAPTCHA text,
- CAPTCHAs based on handwritten text,
- CAPTCHAs based on linguistic knowledge,
- CAPTCHAs that necessitate more physical interaction with users,
- Non-English CAPTCHAs.

CAPTCHAs with “English words” as their CAPTCHA text: In some CAPTCHA systems, such as Gimpy, EZ-Gimpy, CaptchaService.org, PessimialPrint and reCAPTCHA, the CAPTCHA image contains English word(s).

Gimpy: Gimpy is one of the most famous CAPTCHAs which are primarily based on distorted text (Figure 1). This CAPTCHA was developed in collaboration with Yahoo with the aim of protecting chat rooms from spammers to make them unable to post classified ads and write scripts to generate free e-mail addresses. Gimpy picks seven words from a dictionary; then renders a distorted image containing those words. It finally presents them to its users and asks them to type three of the words of the image to gain entry to the service [6].

EZ-Gimpy (CMU): In this CAPTCHA, from Carnegie Mellon University, at first, a word is chosen from a dictionary. In the next step, the word is rendered to an image using various fonts; and different types of distortions such as black or white lines, background grids and gradients, blurring and pixel noise are added (Figure 2). Then, the user is asked to type the word [7].

Captchaservice.org: In this CAPTCHA (Figure 3), each challenge is a six-letter English word chosen from a set of 6000 words. The distortion method used is random shearing [8].

PessimialPrint: PessimialPrint (Figure 4) concentrates on degradations, such as adding noise to or blurring the images to defeat OCR techniques; the designers of this CAPTCHA argue that under the conditions of inferior image quality, there is a significant gap in pattern recognition ability between humans and machines [2]. This CAPTCHA works as follows. First, a word is pseudo-randomly selected from a fixed list containing 5-to-8-letter English words. Then, it is rendered with a typeface (from a fixed list of 5 fonts) and a fixed font size (size=8). Finally, a set of image degradations including x-scaling, y-scaling, skewness, blurriness and adding noise are applied to the image.



Figure 1: Gimpy [6]



Figure 2: EZ-Gimpy[7]



Figure 3: CaptchaService[8]



Figure 4: PessimialPrint[2]

reCAPTCHA: This CAPTCHA [9] selects its words from old printed material or scanned text that cannot be recognized by OCR programs. This strategy not only increases the security of the CAPTCHA; but also the solutions provided by human users can be used for deciphering non-digital text. This CAPTCHA shows two words to the user; the one whose answer is unknown and another ‘control’ word with a known answer. If the user enters the control word correctly, she is assumed to be a human and her answer to the other word is considered as a correct answer. If a specific number of users’ answers to an unknown word match, that word becomes a control word. Figure 5 represents an example of reCAPTCHA. This type of CAPTCHA in which there is no specific answer to the question asked from the user is referred to as ‘collaborative filtering CAPTCHA’ [10].



Figure 5: reCAPTCHA [11]

CAPTCHAs with “random strings” as their CAPTCHA text: using English words in some current CAPTCHAs makes them vulnerable to dictionary attacks. The solution for this issue is exploiting random strings instead of words. This technique is utilized by MSN CAPTCHA, Yahoo, Ticketmaster, Google, etc.:

Hotmail or MSN CAPTCHAs [12]: This CAPTCHA (Figure 6), used in the Hotmail email service registration, selects eight English characters (upper case letters and digits); then, after applying local and global warping, renders the characters with dark blue color on a light gray background. In the next step, three types of arcs are randomly added to make segmentation difficult. The arcs include: “Very thick arcs” (the same as the characters) of foreground color that do not intersect characters, “Thick arcs” of foreground color that intersect characters, and “Thin arcs” of background color that cut characters and remove some of their pixels.

Yahoo! CAPTCHA (Yahoo version2): Starting in August 2004, Yahoo! introduced its second generation CAPTCHA. Its characteristics include using a string of characters instead of English words, containing only black and white colors, using both letters and digits, and having connected lines and arcs as clutter. Two examples of this CAPTCHA are shown in Figure 7 [13].

Ticketmaster: www.ticketmaster.com, which is a well-known ticket sales and distribution website, uses the CAPTCHA of Figure 8. This CAPTCHA is characterized by crisscrossing lines at random angles [13].

Google/Gmail: The specifications of this CAPTCHA, used by Gmail.com, include: using only image warping for character distortion, having only two colors (one for foreground and the other

for background), locating characters close to each other and following a curved baseline. Examples of this CAPTCHA have been displayed in Figure 9 [13].



Figure 6: MSN CAPTCHA [12] Figure 7: Yahoo CAPTCHA [13] Figure 8: TicketMaster [13] Figure 9: Google [13]

BaffleText [14]: This CAPTCHA, Xerox PARC’s version of the Gimpy test, discusses that while computers are not good at recognizing images occluded or interfered by random shapes, humans can distinguish an entire shape or image regardless of its incomplete information according to the Gestalt theory [15].

The principle of Gestalt laws of perception is that the human brain interprets images in their entirety before perceiving their individual parts. Based on this theory, what a human sees when looking at an image is an effect of the whole image, which is more than the sum of its parts. Gestalt principles include proximity, similarity, symmetry, continuity, closure, figure and ground [15]. Proximity refers to how elements located close together tend to be perceived as a group (Figure 10-a). Similarity occurs when similar objects are grouped together (Figure 10-b). Symmetry refers to the tendency to group objects according to their symmetry and meaning (Figure 10-c). Continuity occurs by eye’s moving through an object and continuing to another (Figure 10-d). Closure occurs when parts of information of a shape are missing, but it can be perceived by human brain’s ability in filling in the missing information (Figure 10-e). Figure & ground refers to how the human eye differentiates an object from its surroundings (Figure 10-f). The object is perceived as *figure*; and the surrounding area is perceived as *ground*.

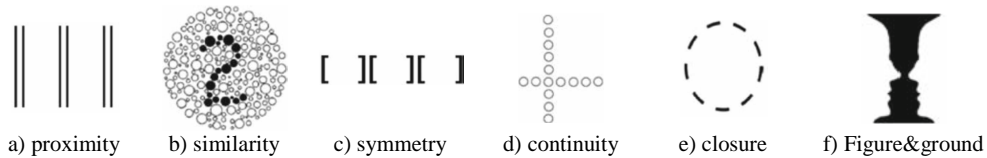


Figure 10: Gestalt theory principles [16]

Based on the closure principle of the Gestalt theory, if some portions of characters in a CAPTCHA test are removed, humans are good at inferring the whole picture from only partial information, while machines are not. **BaffleText** takes advantage of this human brain ability and uses random masks to obliterate parts of test characters to make a stronger CAPTCHA.

This CAPTCHA chooses non-English pronounceable character strings (instead of English words), selects a font type, produces a mask image (by selecting a mask shape and radius) and a random mask operation (addition, subtraction and difference) and combines this mask with the character-string image. Figure 11 represents examples of this CAPTCHA [14].

ScatterType: ScatterType CAPTCHA selects its challenge word from a set of 15,000 English-like nonsense words. Then the algorithm applies a font (from 100 different font types) to it. The image of each character is fragmented using horizontal and vertical cuts and fragments are forced to drift apart until it is difficult to resemble them into characters. In order to achieve the human

legibility, some characters with highest confusability ('q','c','l','o','u') are removed [17]. Examples of ScatterType CAPTCHA challenge images can be seen in Figure 12.

Other examples of text-based CAPTCHAs with random string as their CAPTCHA text include eBay (Figure 13), PHP-class (Figure 14) and MegaUpload (Figure 15) CAPTCHAs.



Figure 11:
BaffleText[14]

Figure 12:
ScatterType[17]

Figure 13:
eBay[18]

Figure 14:
PHP-class[19]

Figure 15:
MegaUpload [20]

Sequenced tagged CAPTCHA [21]: In this CAPTCHA, all characters are tagged with numbers and the user is asked to enter the characters based on the logical ordering of their tags. This strategy adds a new layer of security to the CAPTCHA. Variants of this CAPTCHA include characters as base text tagged by numbers (Figure 16-a), numbers as base text tagged by letters (Figure 16-b) and a hybrid scheme (Figure 16-c) [22].

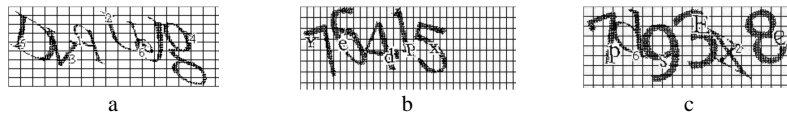


Figure 16: Sequenced tagged CAPTCHA [22]

CAPTCHAs based on handwritten text: While most current text-based CAPTCHAs use machine-printed text, which makes them vulnerable to pattern recognition attacks, there are CAPTCHAs that use handwritten text in their challenges. An example is Handwritten CAPTCHA.

Handwritten CAPTCHA [16]: This CAPTCHA is based on distorted handwritten text. The authors of this article discussed that according to Gestalt laws of perception and the Geon theory of pattern recognition, the interpretation of distorted handwritten text is easy and reliable for humans but difficult for automated programs. Gestalt theory, argues that humans are able to infer the whole picture from partial information [15] which helps them to recognize distorted images. According to Geon theory, humans recognize objects by separating them into geons (simple forms such as cylinders, cones and wedges). In human perception of occluded images of words, decomposing a word into known and unknown visual elements allows users to recognize characters by using rules such as: “words contain specific visual elements”, “combination of letters follows specific rules” and “words convey meaning” to recognize the entire word [16].

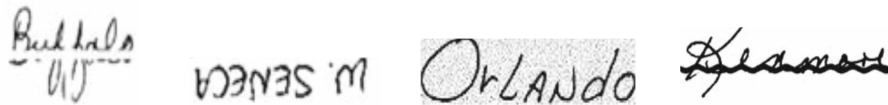


Figure 17: Various transformations applied to the handwritten CAPTCHA [16]

The designers of this CAPTCHA investigated the effects of different transformations including overlapping, adding occlusions, splitting the image in parts and displacing the parts, changing word orientation, etc. on the usability and security of their CAPTCHA (Figure 17). They

designed an algorithm to generate cursive English handwritten text synthetically. They used existing character images and performed auto-scaling, automatic baseline determination, ligature parameterization, ligature joining, skeleton perturbation and skeleton thickening to generate synthetic handwritten words [23].

CAPTCHAs based on linguistic knowledge: Some current CAPTCHA systems combine an OCR problem with linguistic knowledge in order to strengthen their tests. Examples of such CAPTCHAs include semCAPTCHA, odd-words-out, number-puzzle-text CAPTCHA, SS-CAPTCHA and text-domain CAPTCHA:

SemCAPTCHA [24]: In this CAPTCHA (Figure 18), three words, which are names of animals, are displayed to the user; and the user is asked to click on the one which belongs to a different category from the other two (mammals, reptiles, etc.). The three words are graphically similar, but semantically dissimilar.

‘Odd-words-out’ and ‘Number-puzzle-text’ CAPTCHAs: These two CAPTCHAs have been proposed by Captchaservice.org. “odd-words-out CAPTCHA” presents a list of words to the user who is asked to select the words that are not related to the general category of the list (Figure 19). “Number-puzzle-text CAPTCHA” provides the user with a textual description of a number and the user is asked to enter the number (Figure 20) [8].



Please indicate the word(s) that do not belong to the implicit category of the list.

banana	tangerine	pear
oxygen	bassoon	horse
apple	orange	

the number of biological mothers a person usually has, plus one thousand

Figure 18: semCAPTCHA [24]

Figure 19: odd-words-out [8]

Figure 20: number-puzzle-text CAPTCHA [8]

‘Strangeness in sentences’ CAPTCHA: This CAPTCHA [25] displays a list of sentences to the user including natural and wrong sentences. The human user is able to detect Natural sentences. Natural sentences are collected from newspapers, magazines and books; they are checked not to be available online. Wrong sentences are created by getting a machine translator to translate a natural sentence from a mother-tongue language into a non-mother-tongue language and retranslate the result to the mother-tongue language (e.g. Japanese → English → Japanese). This translation strategy usually produces wrong sentences, which are strange for human users. An example is displayed in Figure 21.

Text-domain CAPTCHA: In this CAPTCHA, a list of sentences is shown to a user who is required to find the sentences that are meaningful replacements of each other [26] (Figure 22). Many of the answers might be semantically correct, but contextually incorrect. Deciding which sentences are meaningful alternates based on the context is much easier for a human than it is for a computer.

Select a natural sentence!

- I am disgusted at rain every day.
- The curry was so hot that his tongue was burned and he had to go to the hospital
- I saw the movie of fascinating of which I of my early childhood was reminded.
- Be to eat rice when I return to Japan

Pick the sentences that are meaningful replacements of each other:

- The speech has to move through several more drafts.
- The speech has to run through several more drafts.
- The speech has to go through several more drafts.
- The speech has to impress through several more drafts.
- The speech has to strike through several more drafts.

Figure 21: SS-CAPTCHA [25]

Figure 22: text-domain CAPTCHA [26]

CAPTCHAs that necessitate more physical interaction with users: Requiring more users' physical interaction with the computer while solving a CAPTCHA can improve the robustness of the CAPTCHA. In this type of CAPTCHA, a user is required to enter the answers using the mouse or other physical devices instead of keyboard. Drag-n-Drop CAPTCHA, Physical CAPTCHA and iCAPTCHA are examples of this type of CAPTCHA:

Drag-and-drop CAPTCHA [27]: This CAPTCHA uses mouse actions to differentiate between computers and humans. In this CAPTCHA, the test is shown to the users and they are asked to drag and drop character blocks into their respective blank blocks sequentially (Figure 23).

Physical CAPTCHA: Golle and Ducheneaut [28] proposed a physical hardware-based CAPTCHA test to differentiate between human game players from computers based on the fact that only humans are able to interact with the physical world (e.g. pressing a button, tapping on a touchscreen, or moving a joystick). Since different games receive their inputs from various hardware (e.g. keyboard, joystick, etc.) and there is not a general input device, the designers of this CAPTCHA decided to develop a new cheaper generic hardware only for human verification which is called "CAPTCHA token" (Figure 24). This token, which comes in combination with a game, consists of a keypad, a screen and a CPU.

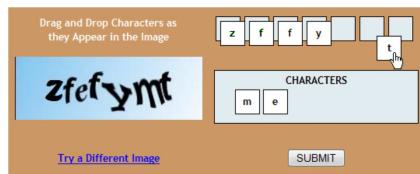


Figure 23: drag and drop CAPTCHA [27]



Figure 24: physical CAPTCHA [28]

iCAPTCHA [29]: This CAPTCHA is an interactive CAPTCHA designed to resist third party human attacks. In this CAPTCHA, the user is asked to click on each CAPTCHA character in each iteration. This iterative back and forth traffic between client and server increases the timing difference between a third party human solver and a real user which helps to detect attacks. An example of this CAPTCHA is displayed in Figure 25.

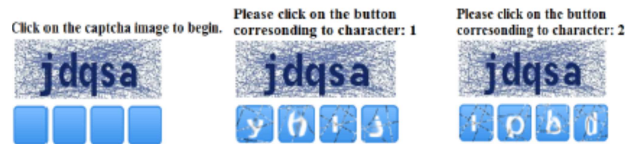


Figure 25: iCAPTCHA [29]

Non-English CAPTCHAs: Besides English CAPTCHAs, some CAPTCHAs have been developed in other languages. One reason for producing localized CAPTCHAs is attacks against current English CAPTCHAs. Another reason is that people are more comfortable with solving tests in their own languages. An example is Arabic CAPTCHA:

Arabic CAPTCHA: Khan et al. [30] proposed an Arabic CAPTCHA to be employed in 21 countries that use the Arabic alphabet. This CAPTCHA exploits the weakness of Arabic OCR systems in segmentation, which results from special characteristics of the Arabic language. These characteristics include dependency of the shapes of the letters on their positions in the word, the existence of a variety of diacritics in Arabic and different number of dots for letters as well as

different positions of dots (many Arabic characters have the same shapes but different number of dots in different locations). This CAPTCHA picks 4 to 9 Arabic letters, randomly selects a font, and adds noise and background clutter which might be confused with dots and diacritics. Other techniques used to strengthen this CAPTCHA system include adding a shadow of the word, character overlapping and changing the position of the word in the image. Examples of this CAPTCHA are displayed in Figure 26. Shirali et al. [31] has designed a similar Persian-Arabic CAPTCHA to detect spam SMS [31].



Figure 26: Arabic CAPTCHA [30]

2.2 Image-based CAPTCHAs

In this type of CAPTCHA, an image is displayed to the users and they are asked a question about the contents of the image. A variety of current image-based CAPTCHAs are discussed in this section in 6 sub-groups:

- CAPTCHAs based on detecting a certain object between other objects,
- CAPTCHAs based on detecting the common characteristic or the subject of all images,
- CAPTCHAs based on detecting a specific part of the test image,
- CAPTCHAs based on swapping the misplaced parts of the test image,
- Orientation-based CAPTCHAs,
- 3D CAPTCHAs.

CAPTCHAs based on detecting a certain object between other objects: in this group of image-based CAPTCHAs, the user is asked to distinguish a certain object between n objects. Examples of this group include Collage CAPTCHA, Asirra and Imagination.

Collage CAPTCHA: Collage CAPTCHA [32] selects pictures of six different objects, applies distortion effects such as rotating to the images and then merges them to create a single image. This image is presented to the users; and they are asked to click on a certain picture. For example, in Figure 27, an image containing an airplane, a car, an apple, an orange, a pineapple and a ball are displayed, and the user is asked to click on the image of the car. A similar CAPTCHA is proposed in [33] with the only difference that [33] provides a multilingual user interface.

Asirra: This CAPTCHA [34] asks users to identify cats in a set of 12 images of cats and dogs. An example of this CAPTCHA is illustrated in Figure 28.



Figure 27: Collage CAPTCHA [32]



Figure 28: Asirra CAPTCHA [34]

Imagination [35]: Imagination is a more advanced image-based CAPTCHA in which the test contains 8 different images located in 8 random orthogonal partitions of a rectangular area. Partitioning and dithering techniques are used to cut the original image tiling and create many false boundaries, which make segmentation difficult for computers (Figure 29). In the click step, a user is asked to click on the center of one of the eight images. In annotate step, the chosen image is distorted and displayed to the user to be annotated. The designers of this CAPTCHA have studied the effect of applying a variety of distortions (such as changing luminance, quantization level, dithering levels) on human and machine recognition [36].



Figure 29: IMAGINATION CAPTCHA [35]

CAPTCHAs based on detecting the common characteristic or the subject of all images: In another group of image-based CAPTCHAs including PIX, Bongo, and activity recognition CAPTCHA, a user is asked to distinguish a particular characteristic which is common between all shown objects:

PIX: In this type of CAPTCHA, a library of pictures with different subjects is prepared. A number of these pictures from within a single category or subject are selected, and presented to the user; and the user is asked to select a phrase expressing the common subject of these pictures. For example if the pictures presented are a globe, volleyball, planet and baseball, the common subject would be “ball” [12].

Bongo: As Figure 30 shows, this CAPTCHA uses two sets of images; each set has some particular characteristics. For example in this figure, one set is boldface, while the other is not. The system then presents a single image to the users and asks them to specify the set to which the image belongs. Because the number of possible solutions is small, this CAPTCHA is not highly robust to brute-force guessing. However, strategies such as cascaded multiple Bongo CAPTCHAs can improve the security of this CAPTCHA [1].

Activity recognition CAPTCHA: Vimina et al. [37] designed a CAPTCHA in which three distorted images of a common activity are displayed to the user. The user is required to detect the activity and annotate it from a list of activities (Figure 31).

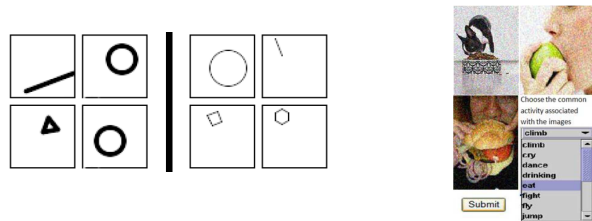


Figure 30: Bongo CAPTCHA [1] Figure 31: Activity recognition CAPTCHA [37]

CAPTCHAs based on detecting a specific part of the test image: another type of CAPTCHA asks users to detect and click on a specific part of the test image. Implicit CAPTCHA, drawing CAPTCHA, Line CAPTCHA and Artificial are of this type:

Implicit CAPTCHA [38]: This CAPTCHA asks a user to click on a specific part of the image. For example in Figure 32, the user is required to click on the climber’s glasses.

Drawing CAPTCHA: Drawing CAPTCHA [39] (Figure 33) displays numerous dots on a screen to the users and asks them to connect dots with a certain shape to each other. The designers of this CAPTCHA argue that computers have difficulty in recognizing the targets from the noise; however, it is easy for a human user to identify the special dots and connect them to each other. An advantage of this CAPTCHA is that it does not require any special knowledge or ability.

Line CAPTCHA: In this system [40], a blurred or randomly segmented line is presented to the user who is asked to drag the mouse along the line. Figure 34 represents this CAPTCHA.

Artificial: This CAPTCHA (Figure 35) is based on human face recognition [41]. In this CAPTCHA, a distorted face embedded in a background that includes face-like clutters is shown to the user. The user must detect the face and click on the eye corners and mouth corners.



Figure 32:Implicit CAPTCHA [38]

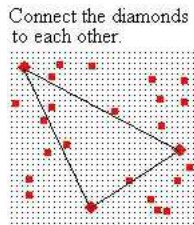


Figure 33: Drawing CAPTCHA [39]

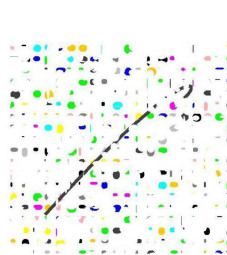


Figure 34: Line CAPTCHAs [40]

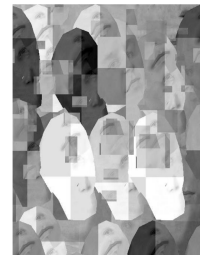


Figure 35: Artificial [41]

CAPTCHAs based on swapping the misplaced parts of the test image: Some CAPTCHAs ask users to exchange misplaced blocks of the image to make the original image. ‘Exchanging image blocks CAPTCHA’ and Jigsaw puzzle CAPTCHA are examples of this group:

Exchanging-image-blocks CAPTCHA: Liao [42] proposed a CAPTCHA based on swapping the contents of two misplaced non-overlapping regions in an image (Figure 36).

Jigsaw puzzle CAPTCHA: Gao [43] designed a CAPTCHA using jigsaw puzzles in which the user is required to solve a puzzle by swapping the two misplaced pieces (Figure 37).

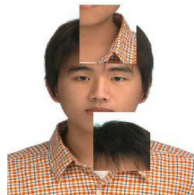


Figure 36: ‘exchanging image blocks’ CAPTCHA [42]



Figure 37: Jigsaw puzzle CAPTCHA [43]

Orientation-based CAPTCHAs: In this group of CAPTCHAs, such as Image Flip CAPTCHA, What's up CAPTCHA and Sketcha, users are required to detect the correct orientation of the objects in the test image:

Image Flip CAPTCHA [44]: This CAPTCHA displays an image composed of several sub-images to the user. The user has to detect all non-flipped images and click on them. An example is shown in Figure 38 in which three non-flipped images exist.

Whats up CAPTCHA: This CAPTCHA, proposed by Gossweiler et al. [45] asks users to detect the orientation of the randomly-rotated test image and then, use a slider to rotate the image to its upright position (Figure 39).

Sketcha [46]: Sketcha is another orientation-based CAPTCHA that shows the users a set of images which are line drawings of 3D models (Figure 40). The user must detect the upright orientation of each image. Detecting orientation in some images including symmetric images, and images that are typically oriented upside down is difficult (or impossible) for human users. Hence, the designers of this CAPTCHA applied a filter to remove these groups of images from the database of the CAPTCHA. The filtering process is as follows: in a user study, each user is shown some test images. If a certain number of users answer an image inconsistently, that image is considered as a difficult image and removed from the database.



Figure 38: Image Flip CAPTCHA [44] Figure 39: What's up CAPTCHA [45] Figure 40: Sketcha images [46]

Sub-image orientation CAPTCHA: Kim et al. [47] proposed a CAPTCHA based on the orientation of sub-images. The designers of this CAPTCHA argue that while a whole-size photo contains semantic cues such as sky, grass and dark brown ground that help a machine to detect image orientation, random sub-images do not include meaningful objects that a computer program can easily recognize. In this CAPTCHA, a number of random blocks of an image are cropped, rotated and shown to the user who is asked to find their correct orientations.

3D CAPTCHAs: Working with 3D images instead of 2D ones can reduce the probability of pattern recognition and database attacks. This strategy is used in "2D CAPTCHA from 3D models", Spmfizzle and 3D CAPTCHA:

2D CAPTCHA from 3D models: This CAPTCHA, designed by Hoque et al. [48], has a database which is populated with several 3D models. To create a new test, this CAPTCHA selects a 3D model, applies different distortions and lighting effects on it and transforms it to a 2D image. This image is displayed to the user to be identified. Since infinite number of 2D images can be gained from a 3D model, the image database for this CAPTCHA is very large.

Spamfizzle [49]: In this CAPTCHA, 3D objects are designed manually and their attributes and behaviors are described and recorded. To produce a new test, the CAPTCHA generation algorithm creates a 3D image by combining different objects and labelling the attributes of each object. The image is displayed to the user who is asked to enter the labels that correspond to a list of attributes. Figure 41 shows an example of this CAPTCHA.

3D CAPTCHA: Imsamai and Phimoltares [50] propose a 3D CAPTCHA based on the idea that recognizing a sequence of 3D characters is easy for the human, but difficult for computers. This CAPTCHA adds rotation, overlapping, noise, scaling, font variation and background texture and special characters to the image to make it stronger. Figure 42 represents examples of CAPTCHA.

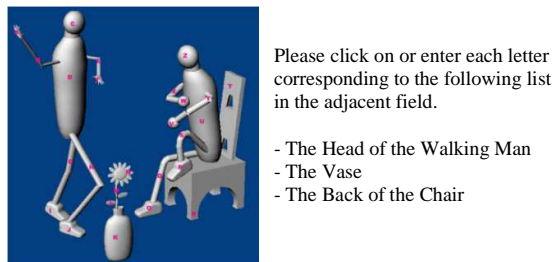


Figure 41: Spamfizzle CAPTCHA [49]



Figure 42: 3D CAPTCHA [50]

2.3 Audio-based CAPTCHAs

Audio-based CAPTCHAs are usually used as a complement for text-based CAPTCHAs. Many popular websites such as eBay, yahoo and Microsoft use both visual and audio CAPTCHAs [51]. An audio CAPTCHA generally picks a random sequence of letters or numbers; renders them into a sound clip; includes some level of distortion; and then presents the recording to the users. The user is asked to type the contents of the recording [1]. In one type of audio CAPTCHAs, known as spoken CAPTCHA, the users are required to repeat the test instead of typing it. This feature makes this CAPTCHA also suitable for blind users [52].

Including noise in audio-based CAPTCHAs can improve their security. Chan [53] discusses that adding background noise to sound CAPTCHAs decreases the accuracy rate of a speech recognizer more than that of a human. Sauer et al. [54] adds silence to the sound in order to discourage checksum/signature based attacks. Kochanski et al. [55] uses 18 different sets of distortion to sound in order to make the problem difficult for machines. However, According to Bursztein et al. [51], among various types of noise that can be incorporated in an audio-based CAPTCHA, semantic noise, i.e. a noise with similar characteristics of a spoken signal, is the most effective noise in improving security.

2.4 Motion-based CAPTCHAs

In this type of CAPTCHA, a movie or animation is shown to the users and they are asked a question about the contents of the clip. Answering the question usually requires recognizing an action, animated string or an image in the clip. Examples of motion-based CAPTCHAs include NUCAPTCHA, HelloCAPTCHA, animation CAPTCHA, and 3D animation CAPTCHA.

NUCAPTCHA: In this CAPTCHA [56], a string is animated from right to left and the user is asked to type the last word (Figure 43).

HelloCAPTCHA: A test in HelloCAPTCHA [57] consists of six letters or digits displayed in an animated GIF image and the user is required to type the characters (Figure 44).

Animation CAPTCHA: In this CAPTCHA [58], a few animated objects are shown to the user who is required to detect and click on one of the objects. Moving objects on a random path, rather than having a static test image, makes a CAPTCHA more secure against random guessing or segmentation attacks (Figure 45).

3D animation CAPTCHA: This CAPTCHA [59] selects three English letters or digits, renders the string to an image in which character area is covered by '1's and background area is covered by '0's. For example, Figure 46 shows the presentation of letter 'x' in this system. In the next step, the algorithm adds a third dimension to the image using a sin function and applies waves to this 3D animation. This CAPTCHA also changes colors randomly during the presentation of a test.

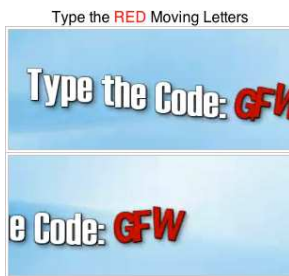


Figure 43: Frames of a test in NUCAPTCHA [56]



Figure 44: Frames of a HelloCAPTCHA test [57]



Figure 45: Frames of an Animation CAPTCHA test [58]

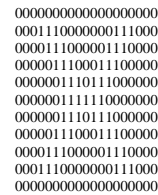


Figure 46: A character in 3D animation CAPTCHA [59]

2.5 Hybrid CAPTCHAs

This section provides examples of CAPTCHAs that combine text and multimedia in their systems. Moreover, CAPTCHAs designed for special purposes such as smartphones or for people with disability are discussed in this section.

Dynamic CAPTCHAs: While most available CAPTCHA systems use one or two pre-defined types of CAPTCHA, e.g. text-based, image-based, etc., a dynamic CAPTCHA selects a type of CAPTCHA among different available CAPTCHA types based on the information the user entered in the first steps of registration or the information provided by the user's web browser. An example of this group of CAPTCHAs is Dynamic CAPTCHA.

Dynamic CAPTCHA [60]: This CAPTCHA selects a type of CAPTCHA based on the limitations of a user's device (in entering characters or numbers) or a user's restrictions and disabilities (such as blindness), the native language of the user, etc.

CAPTCHAs with multiple challenges: Asking users to pass multiple challenges, probably of different types, instead of a single test improves robustness. ‘Multiple challenge-response system’ [61] is an example of this type of CAPTCHA.

Multiple challenge-response system: Longe et al. [61] designed a CAPTCHA in which the user has to solve multiple challenges instead of a single test. The challenges in this CAPTCHA include: a mathematical test and an image CAPTCHA. The mathematical test consists of alphanumeric characters and symbols; the user is expected to distinguish numbers and add them together (Figure 47). The image CAPTCHA is a distorted image of a random string of characters and digits.

Multi-type CAPTCHAs: A CAPTCHA can be a mixture of different types of CAPTCHA; for example a combination of text and image based CAPTCHAs. Question-based CAPTCHA and tree-based handwritten CAPTCHA are examples of this CAPTCHA group.

Question-based CAPTCHA: This CAPTCHA is a hybrid CAPTCHA proposed by Shirali-Shahreza [62]. In this CAPTCHA, the user is shown a test which is a mathematical question created by a combination of text and images; then she is asked to enter the answer (Figure 48).

$$\begin{array}{l} \boxed{6d4+if} = \square \\ \boxed{6zs9e+} = \square \end{array}$$

Figure 47: The mathematical challenge [61]

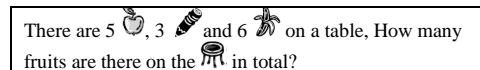


Figure 48: Question-based CAPTCHA [62]

Tree-based handwritten CAPTCHA: This CAPTCHA, which is a combination of text and graphics, uses the human’s ability to read handwritten text and understand tree structure to create a strong CAPTCHA. In this CAPTCHA, a synthetic handwritten tool creates the words, some random transformations are applied to them, and then, they are combined with a randomly generated tree structure and random test questions. An example is displayed in Figure 49. As it can be seen in the figure, the distorted handwritten words constitute the nodes of the tree and the random symbols are located in the middle of the branches. The random question “which word is connected to center by a line marked with a circle?” is asked in this example. It is easy for human but difficult for computers to answer this question [63].

CAPTCHAs designed for smartphones or other mobile devices: Increasing the popularity of smartphones and using them for browsing the internet highlights the need to design new CAPTCHA systems that consider their special characteristics. These characteristics include touch screen and difficulty in using the keyboard, etc. CAPTCHAs developed to be used on smartphones are usually not required to be as robust as those designed for PCs because of the lack of powerful processors and small memories. Examples of this group of CAPTCHA include CAPTCHA zoo and Highlighting CAPTCHA.

CAPTCHA Zoo: This CAPTCHA [64], which is designed for mobile devices, is based on the fact that humans are superior to machines in recognizing similar objects in images. In this CAPTCHA, the challenge image contains two visually similar types of animals: target animals and noise animals. Different colors, lighting, rotation and overlapping are applied to the image. Then the user is asked to recognize the target animals (Figure 50).

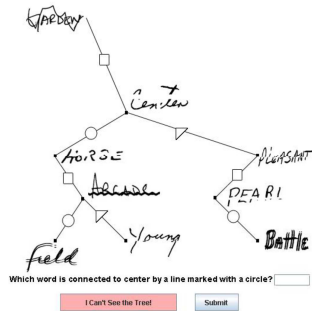


Figure 49: an example of the tree-based handwritten CAPTCHA [63]



Figure 50: CAPTCHA zoo [64]

Highlighting CAPTCHA: In this system, a random string on a noisy background is shown on the screen of the mobile device and the user is asked to highlight the characters with stylus [12].

CAPTCHAs designed for people with disability: Examples of this group of CAPTCHA include ‘Localized CAPTCHA for linguistically-challenged individuals’, ‘image/audio CAPTCHA’ and ‘a CAPTCHA for deaf people’.

Localized CAPTCHA for linguistically-challenged individuals: This CAPTCHA makes a bank of names of different objects; then chooses six of them and searches and finds yahoo images for these six names; shows the user one image for each name. Afterwards, this CAPTCHA selects one name from the six alternatives, says it (using speech) to the user and requests the user to click on the related image [65]. The designers argue that this CAPTCHA is easy for a human because it is appropriate for all ages, there is no need for typing abilities, it is in the user’s native language; hence, no English language knowledge is required. Figure 51 depicts this CAPTCHA.

Image/audio CAPTCHA [66]: This CAPTCHA combines pictures of objects with sounds associated with those objects (Figure 52). Since the audio is related to the image, this type of CAPTCHA can be useful for people possessing vision or hearing impairments. Another CAPTCHA, designed for blind people, contains a simple mathematical problem which is created and converted to speech using a text-to-speech system. The user is asked to listen to the sound and answer the question [67].

CAPTCHA for deaf people [68]: This CAPTCHA presents a word to the user using sign language and asks the user to recognize the word and choose its name from a list of words (Figure 53).



Figure 51: CAPTCHA for illiterate people [65]



Figure 52: Image/audio CAPTCHA [66]



Figure 53: CAPTCHA for deaf people [68]

3 Breaking CAPTCHA

In this section, we will explore attempts of compromising the CAPTCHAs presented in Section 2. However, many of the presented CAPTCHAs are nothing more than an idea and hence do not provide sufficient details with regard to their realization to allow an analysis of their robustness. This limitation also applies to sections 4 and 5 where we discuss the usability and robustness of CAPTCHAs. Hence, the remaining of this paper will focus on those systems that have actually been deployed and hence present the researcher with a “target” to analyze.

3.1 Algorithmic approaches to break CAPTCHAs

The general process in attacking CAPTCHAs is a segmentation step followed by an object recognition step. In the first step, the location of each object is found and in the second one, each object is recognized. Research suggests that segmentation is more difficult than recognition for machines. For example, in text-based CAPTCHAs, computers are very good at recognizing single characters, even if the characters are highly distorted [69]. Therefore, a CAPTCHA, which is designed to be segmentation-resistant, is less vulnerable to attacks.

In this section, at first, general segmentation and recognition attacks are discussed and then, examples of specific attacks against current CAPTCHAs are explained. As mentioned before, the majority of available CAPTCHAs are text-based CAPTCHAs; and most attacks have been designed against this type of CAPTCHA. Hence, most strategies and attacks discussed in this section are related to text-based CAPTCHAs.

3.1.1 Segmentation attacks

Well-known segmentation attacks such as vertical histogram, color-filling and snake segmentation along with pre-processing steps are used to segment current CAPTCHAs. Generally, a combination of these methods is utilized to break a CAPTCHA.

Pre-processing: This step is used to remove background patterns, separate foreground from the background and eliminate noise as much as possible. Examples of strategies used in pre-processing step explained in this subsection.

Binarization: Binarization converts the image into a black and white image using thresholding. As a result of binarization, some characters might be broken. In order to fix the broken characters, all pixels of background color that have neighbors of foreground color are converted to foreground color ([69] and [70]).

Background mesh removal [70]:

- Black mesh: after thresholding, the vertical and horizontal line pixels that do not have neighboring pixels are considered as background mesh pixels and removed.
- White mesh: after thresholding, white line pixels that have neighboring pixels are considered as background white mesh pixels and converted to the foreground color.

Arc Removal [69]:

- Thin arcs: This type of arcs can be removed using erosion/dilation operations.
- Thick arcs: These arcs can be identified by their pixel-count, shape and location. It is very likely that objects with a small pixel count, objects lacking circles or located near the image boundaries are arcs.

A strategy in removing any type of arcs is using histogram. If in x-histogram of an image, consecutive columns have only a limited number of foreground pixels; it is very likely that

those columns belong to an arc. The same concept is true for a y-histogram. This trick can make segmentation easier (Figure 54).

Simple segmentation: If the number of characters in a test and their widths are constant, the whole test image can be divided into parts of the same width, each part being a character. This strategy can also be used in combination with other segmentation techniques (for example, when histogram is not able to completely segment the characters and instead, returns “chunks of characters”) [69].

Vertical histogram Segmentation: This method segments the image into characters using a vertical histogram (Figure 55) ([69], [71] and [72]).

Color filling segmentation: This approach segments all the objects by finding connected components. In order to find a connected component, the algorithm at first detects a foreground pixel, and then traces all of its foreground neighbors until all pixels in this connected component are traversed [69].

Snake segmentation: In this strategy, lines separate the letters of a test. The lines are programmed to move such as snakes trying not to collide with the characters (Figure 56) [71].

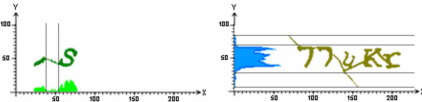


Figure 54: using histogram to remove arcs [69]



Figure 55: vertical histogram segmentation [69]

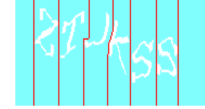


Figure 56: snake segmentation [71]

3.1.2 Recognition attacks

After segmenting objects of a test, a recognition step is required to detect each object. This type of attacks including object recognition attacks, pixel-count, dictionary and database attacks are discussed in this subsection.

Object recognition attack: A wide range of artificial intelligence object recognition algorithms including pattern matching methods or OCR systems can be used to recognize objects [73], [74].

Pixel-count attack: This attack is applicable to CAPTCHA systems in which each character has a constant pixel count and the pixel count of each character is different from that of other characters. In this attack, after segmentation, the number of foreground pixels of each segment is counted and used to look up Table 1 to identify the letter in the segment [72]. For letters with the same pixel counts (such as “P” and “V”), the algorithm uses a simple geometric analysis to differentiate them. For example, to distinguish between a “P” and a “V”, the algorithm draws a vertical line in the middle of the normalized letter. If the line cuts through the letter in only one point, the character is a “V”; otherwise, it is a “P”. The same process is performed for other similar cases.

Dictionary attack: Using only the words of a specific dictionary in a CAPTCHA limits the number of possible strings. An attack against such a CAPTCHA is designed to search the whole dictionary to answer the test. This attack is called a dictionary attack. Dictionary attacks can also be used in combination with other attacks. For example, when the vertical histogram is not able to segment all of the characters of a test because of overlaps between them, the dictionary is searched for all possible candidates, the word with the same pixel-count as the challenge word is selected as result (Figure 57) [72].

Database attack: Database attack is a type of attack in which the entire database is gradually revealed. Every time a challenge is displayed, a portion of the database is uncovered and by solving enough challenges, the attacker would have access to the whole database [34].

Table 1: A letter-pixel count lookup table [72]

Letter	Pixel Count	Letter	Pixel Count
A	183	N	239
B	217	O	178
C	159	P	162
D	192	Q	229
E	163	R	208
F	133	S	194
G	190	T	175
H	186	U	164
I	121	V	162
J	111	W	234
K	178	X	181
L	111	Y	153
M	233	Z	193

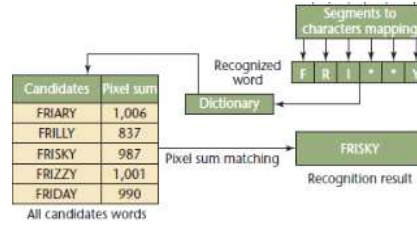


Figure 57: Dictionary attack [72]

3.1.3 Examples of attacks on current CAPTCHAs

In this section, examples of attacks against existing CAPTCHA systems are explored.

Attacks on current text-based CAPTCHAs

Attacks on EZ-Gimpy CAPTCHA: EZ-Gimpy CAPTCHA, which consists of an English word on a noisy background, has been vulnerable to several attacks. Chellapilla and Simard [70] discussed that the background of this CAPTCHA, which can be categorized into three different types of ‘no mesh’, ‘black mesh’ and ‘white mesh’, can be removed by simple pre-processing algorithms (Figure 58). One of the weaknesses of this CAPTCHA, which is using a dictionary of words, has made this CAPTCHA vulnerable to an attack designed by Mori and Malik. This attack uses shape context matching to identify the entire test word (rather than individual characters) with a success rate of 92% [75]. Their attack’s lack of reliance on detecting individual letters of a test reduces the complexity of the attack. Another attack on EZ-Gimpy – based on detecting the whole word instead of its characters – has been proposed by Moy et al. [76]. The designers of this attack took advantage of the small set of template images used in this CAPTCHA to break it with a success rate of 99%. In this attack, the attackers collected the small set of template images; and solved tests by comparing the challenge image with each of the templates to find the most correlated image.



Figure 58: An attack on EZ-Gimpy CAPTCHA [70]

An attack against Ticketmaster CAPTCHA: The general process of the attack is the same as EZ-Gimpy [70]; however, because of the criss-crossing lines used in the Ticketmaster CAPTCHA, a dilution and erosion step is added in order to remove every thin line in the image (Figure 59). This attack can break Ticketmaster CAPTCHA with a success rate of 4.9%.



Figure 59: An attack on Ticketmaster CAPTCHA [70]

An attack on MSN CAPTCHA: Steps of a segmentation attack on this CAPTCHA designed by Yan and Ahmad [69] are summarized in Figure 60. In this CAPTCHA, the few number of overlap or connection between characters, if any, allows vertical histogram and color-filling segmentation attacks to detect connected components (two first steps in Figure 60). In the third step, the different shape, location and pixel-count of arcs compared to those of characters helps remove the arcs. Finally, knowing the fact that each MSN CAPTCHA contains exactly 8 characters and that the width of each test is approximately constant and the direction of the test is always horizontal, each chunk of connected characters, which is the result of previous steps, can be divided into pieces of the same width to separate characters. The success rate of this attack is 60%.

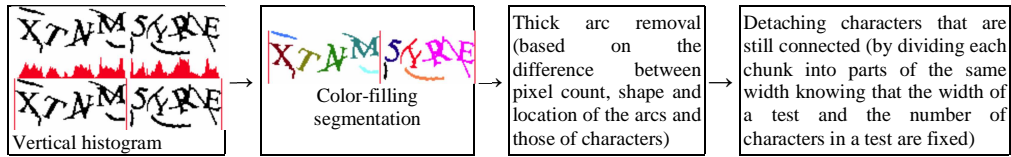


Figure 60: Steps of an attack on MSN CAPTCHA [69]

An attack on captchaservice.org: The steps of an attack against captchaservice.org designed by Yan and Ahmad [71] are displayed in Figure 61. To segment the characters, the designers of this attack used two flaws of this CAPTCHA, i.e. using distinct colors for foreground and background and having non-connected characters. After segmentation, the fact that each character in this CAPTCHA had a specific pixel-count helped attackers recognize characters by pixel-count attack. Finally, in cases that two characters of a test had the same pixel-count, and could not be distinguished by pixel-count attack, a dictionary attack was used to solve the test. The success rate of this attack was 94%.

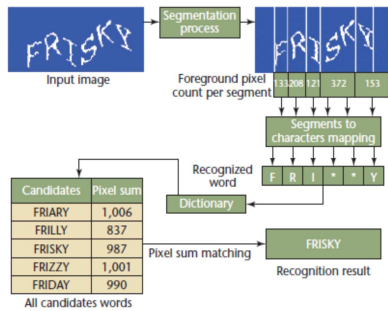


Figure 61: An attack against captchaservice.org [71]

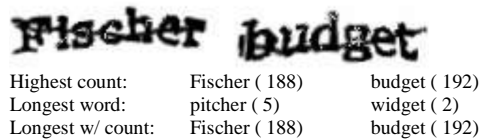


Figure 62: An attack on reCAPTCHA [77]

Attacks against reCAPTCHA: While reCAPTCHA challenges are selected from old books and newspaper text that cannot be recognized by OCR programs, they might be vulnerable to attacks after a pre-processing step (Figure 62). Two main weaknesses of reCAPTCHA are using English words and allowing one error in answering tests. Beede [78] proposed an algorithm to remove the reCAPTCHA’s curvature and sharpening the characters. After applying this pre-processing step, OCR can solve this CAPTCHA by a success rate of 23.6%. [77] applied a variety of erode/dilate matrices on every CAPTCHA word before OCRing; collects the results given by OCR and selects the longest result with a non-trivial count as the correct answer (Figure 62). The success rate of

this attack was 17.5%. The success rate increased to 23% after the designers of reCAPTCHA improved its usability by eliminating the horizontal line running through the challenge word.

Attacks on current image-based CAPTCHAs

Examples of attacked image-based CAPTCHAs include Drawing CAPTCHA, ArtiFacial, IMAGINATION; and Asirra (Figure 63). Most segmentation attacks on image-based CAPTCHAs use the vulnerability that the added noise is not similar to the foreground objects. For example, in *Drawing CAPTCHA*, the difference between the size of diamond-shape target dots and that of clutter dots was the cause of an attack designed by Lin et al. [64] with a success rate of 75%. Another example is an attack on *ArtiFacial* in which removing some parts of noise that did not have the same features as human face made segmentation easier [79] (success rate=18%). Another flaw of a CAPTCHA that makes segmentation less costly is having minimal solution requirements. For example, *Imagination* only requires a user to click on the center of one of several pictures provided by the test. Finding the boundaries of only one object, even in a noisy image with a lot of false boundaries is not very difficult for an attacker. Based on this weakness, Zhu et al. attacked *Imagination* with the success rate of 4.95% [79].

Most recognition attacks on image-based CAPTCHAs use image classifiers to recognize images. For example, Golle [73] used a combination of color and texture features to distinguish dog and cat images in *Asirra* CAPTCHA. This attack had a success rate of 10.3%.

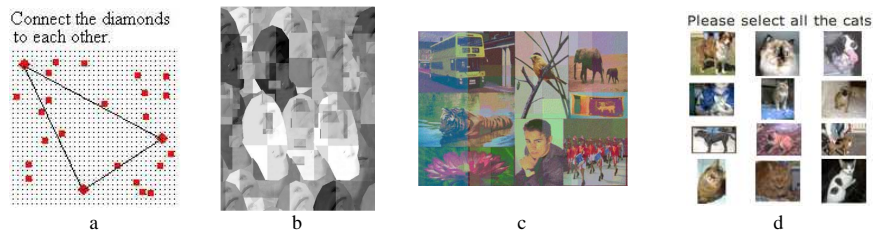


Figure 63: Examples of attacked image-based CAPTCHAs; a) Drawing CAPTCHA [39], b) ArtiFacial [41], c) IMAGINATION [35], and d) Asirra [34]

Attacks on current audio-based CAPTCHAs

Audio CAPTCHAs are relatively weaker than visual CAPTCHAs. The reason is that human visual system constitutes a larger portion of the brain in compared with human audio processing system. Bursztein et al. [51] proposed an attack on eBay, authorize, yahoo and Microsoft Audio CAPTCHAs with a success rate of 82%, 89%, 45.5% and 49% respectively. This attack segments the digits using a low-pass RMS filter that eliminates regular and constant background noises and leaves peaks corresponding to the digits. After segmentation, classifiers have been used to recognize digits. The most important weakness of these CAPTCHAs was the difference between background noise and foreground signals. reCAPTCHA that includes semantic noise, a noise with similar characteristics of a spoken digit, was more resistant against this attack.

Attacks on current motion-based CAPTCHAs

Attacking motion CAPTCHAs can be both harder and easier than image-based or text-based CAPTCHAs. It can be harder because in motion CAPTCHAs, segmentation and recognition, performed by the processing of the frames and by motion tracking, is more complex. It can be easier because segmentation phase can be more accurate by having multiple copies (frames) of the same CAPTCHA. The basis of most attacks on motion CAPTCHAs is to extract important

information from the animation frames and reduce the animation to a traditional text-based CAPTCHA (Figure 64). Based on this strategy, Bursztein [80] and Nguyen et al. [81] designed attacks on NUCAPTCHA and HelloCAPTCHA respectively. The success rate of the first attack was 83% and for the second one, the success rate was between 16% and 100%. The vulnerabilities of these CAPTCHAs, including having discriminative features between text and the noise, the fixed number and position of the characters, inappropriate use of colors, the constant number and the delay of the frames, guided the attacks. These attacks show that segmentation resistant in video CAPTCHAs, if not well designed, can be equivalent or less than text-based CAPTCHAs. One strategy to improve the security of motion CAPTCHAs suggested by Bursztein [80] is to use a confusing moving background.



Figure 64: Examples of attacks on motion CAPTCHAs: extracting information from animation frames and convert the clip into a single image; a) An attack on NUCAPTCHA [80], b) an attack on HelloCAPTCHA[81]

3.2 *Random guessing attacks*

In random guessing attack, also referred to as blind guessing or no-effort attack, an attacker tries to break a CAPTCHA by guessing the answer. In text-based CAPTCHAs, given the character set size, c , the probability of solving an n -character CAPTCHA challenge by blind guessing is $1/c^n$ [82]. In an image-based CAPTCHA that asks a user to detect an object between n candidate objects, the likelihood of a correct guess is $1/n$. Weaknesses of a CAPTCHA that can make it vulnerable to this attack include using a small input space, having a small number of candidate objects in a test, and imposing no limits on the number of attempts to solve a test.

3.3 *Using social engineering to break CAPTCHAs*

Using cheap 3rd party humans is a way to break CAPTCHAs. Kang et al. [83] designed a CAPTCHA phishing attack that is a form of social engineering attack. They deployed a CAPTCHA phishing interface on a webpage (called CAPTCHA carrier) and selected some high-traffic website as phishing areas to publish their phishing messages. Phishing carrier and phishing area can be two different webpages. Alternatively, phishing carrier can be integrated into the phishing area using Adobe Flash [84]. Since people are less willing to click an unknown hyperlink, the second approach has been more successful.

Truong et al. [29] designed another attack called “Instant Messenger CAPTCHA attack”. The major components of this attack are an attack script and an IM connector. The first component scrapes CAPTCHA images and uses the IM connector, to send them to the 3rd party human who solves the tests. Since instant messengers allow a real-time communication between participants, the attack cannot be detected using timeout values.

4 Robustness of CAPTCHAs

A good CAPTCHA must be both easy-to-solve for humans and strong enough to resist attacks. Designing CAPTCHAs that fulfil both usability and robustness criteria is difficult. The key point to design such a CAPTCHA is to exploit the gap in the recognition abilities between humans and computers (Figure 65).

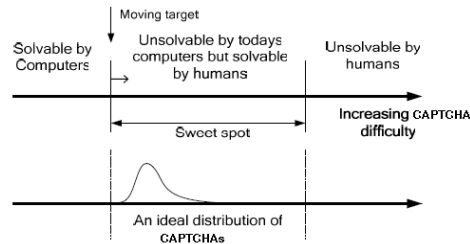


Figure 65: CAPTCHA difficulty for humans and computers [85].

A CAPTCHA method is called strong if no automated computer program can solve its challenges with a success rate of higher than 0.01% [85]. The security of a particular CAPTCHA test can be analysed by investigating its resistance to attacks that possibly may be used to break it. We divided this study into five subsections:

- Segmentation resistance,
- Recognition Resistance,
- Random guessing resistance,
- Security against 3rd party human solver attack; and
- Other security measures.

Most techniques are related to text-based CAPTCHAs because this type of CAPTCHA has been studied more than others. Many of the strategies might be mentioned in one category, but be applicable to other categories as well.

4.1 Segmentation resistance

A major determinant of CAPTCHA robustness is its resistance to segmentation attempts. Prior studies show that machine learning algorithms are better at solving recognition problems than segmentation problems [70]. Strategies to improve the segmentation-resistance of a CAPTCHA are discussed in this section.

Applying degradations: The first strategy to increase the security of a CAPTCHA is applying degradations to the test image (Figure 66) including:

Character fragmentation: Making horizontal and vertical fragments in characters or using thinned images to break characters into isolated portions makes a CAPTCHA more resistant to segmentation [17]. It can also reduce the success rate of a pixel-count attack.

Masking degradations: Using masks to degrade a CAPTCHA test can improve security [14].

Crowd letters together: Different ways to apply this strategy is using condensed fonts with narrower aspect ratios than usual or Italic fonts whose rectilinear bounding boxes overlap their neighbors or thickened images, so that characters merge together. Another strategy is to juxtapose characters in any direction (instead of just x direction) [69].

Additional arcs: These arcs may do or do not intersect with the original characters. If they are intersected, they can make segmentation difficult; and if they are not, the arcs can still baffle attacking algorithms since they might be confused with characters [12].

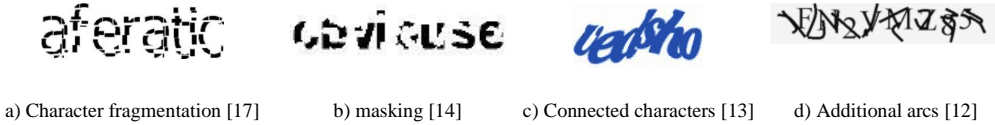


Figure 66: Degradations applied to the text

Text distortions: Another strategy to increase the CAPTCHA’s resistance to segmentation is applying text distortions [13] (Figure 67) including:

- *Global warping* (character-level elastic deformations applied to all of the characters of a challenge together and can foil template-matching algorithms for character recognition),
- *local warping* (small ripples, waves, and elastic deformations applied to each character independently and can foil feature-based algorithms for character recognition),
- *Scaling* (stretching or compressing the text in the x or y direction),
- *Translation* (moving the text either up or down and left or right), and
- *Rotation*.

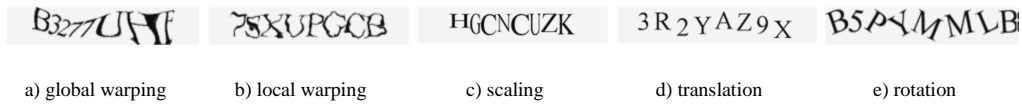


Figure 67: Various text distortions [13]

Background clutters: Clutters used to increase the security include blurring, gridlines, mesh, patterns and arcs for visual CAPTCHAs and background noise for audio-based CAPTCHAs. Examples of the background/foreground clutters can be seen in Figure 68. Applying a variety of background clutters in different tests makes it more difficult to extract foreground from the background using pre-processing algorithms. However, many types of degradations, if not applied properly, might reduce usability while having no positive effect on robustness [86]. For example, the effects of blurring, thresholding and noise can be removed by almost all current OCR systems [14]. Any noise that does not resemble challenge text/image/audio is not a good clutter [51]. A good idea is to use shapes similar to test images as background clutter.

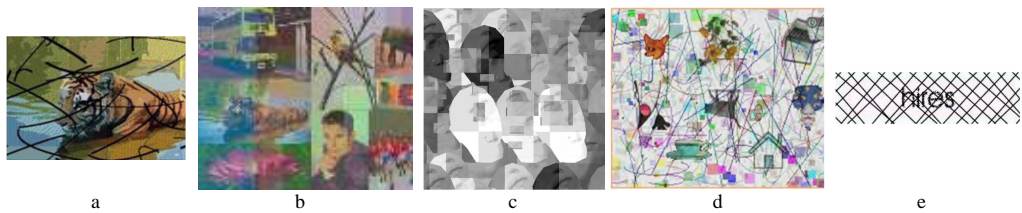


Figure 68: Examples of background clutters. a,b: IMAGINATION [35], c:ArtiFacial [41], Image Flip CAPTCHA[44] and e: TicketMaster [13] CAPTCHA

Making false boundaries: In an image-based CAPTCHA that contains a number of objects, an attacker can use a boundary detector to segment the objects. Creating false boundaries between objects when designing a CAPTCHA can solve this problem (Figure 68-b) [35].

Appropriate use of colors: Using multiple colors for the background and foreground, and including some foreground colors into the background and vice versa is another technique to make segmentation more difficult. Using two colors, one for background and the other for foreground or using different colors for adjacent characters helps attackers to separate them easily [87].

Randomness in the presentation of CAPTCHA test: Using limited patterns in displaying CAPTCHAs allows automated systems to find “where each character is” and facilitates their attempts in solving the challenge. The solution for this problem is to introduce more types of patterns and have them occur randomly. The ideal situation is to remove every pattern and use randomness in the presentation of CAPTCHA as far as possible to confuse the machine. For a text-based CAPTCHA, random positioning of the characters, different fonts and font sizes, using random features of other languages such as different writing directions, cursive features and diacritics; and random number of characters can improve robustness [82]. For image-based CAPTCHAs, random positioning of the images, different-shaped boxes and random background dimensions and for animated CAPTCHAs, random number of frames and random frame delay can reduce the probability of segmentation attacks [81].

Dynamic presentation of the CAPTCHA: Dynamism in the presentation of a CAPTCHA, for example animating the images by changing their positions on a random path with a random frame delay, reduces the probability of segmentation and random guessing attacks [58].

In summary, the strategies to make segmentation harder include:

- Applying degradations (character fragmentation, character overlapping, masking operations, additional arcs connected to the letters with the same width as letters),
- Background clutter (arcs, images similar to foreground objects, use a variety of clutters for different tests),
- Different colors,
- False boundaries,
- Randomness (position, direction, size and dimensions of the test image, text length, frame count, frame delay),
- Dynamic presentation of the CAPTCHA.

4.2 Recognition Resistance

Although the most important step in breaking a CAPTCHA is segmentation, strategies should be considered to improve CAPTCHA security against recognition attacks. Such techniques are discussed in this section:

Using random strings instead of words: Using non-English character strings instead of English words in a text-based CAPTCHA can reduce the probability of Dictionary attack. Inserting special characters will also confuse OCR systems [88].

Make it impossible to distinguish a character by counting its pixels: In text-based CAPTCHAs, making all characters have approximately the same pixel count or making a character have a very different pixel counts in different challenges definitely defeats pixel-count attack [72].

Using different fonts: Using different fonts can defeat pattern recognition attacks. Another suggestion is using handwritten text since humans are superior to machines in recognizing handwritten text [16].

Selecting the words, sentences or images that cannot be recognized by current recognition systems: In a text-based CAPTCHA, before selecting an image as a test image, make sure that current popular OCR

systems cannot solve it (reCAPTCHA [9] implements this strategy). In CAPTCHAs that use sentences instead of words, the source of sentences is usually old books or newspapers; make sure that sentences are not available on the web [25].

Using a large database: This technique improves CAPTCHA security by reducing the probability of random guessing and database attacks. There are different ways to increase the size of database of an image-based CAPTCHA; For example, using image collections of other picture providers [34] or enticing people to label images while playing a game [89].

Using visually similar but semantically dissimilar characters/images/audio: Similarity can confuse pattern recognition programs. When two objects are visually similar, it is much easier for human to recognize the slight difference between them than it is for a computer program [64].

Removing or deforming features to defeat pattern-matching programs: many image/text detection tools use pattern matching to recognize objects. Removing or deforming objects' features can confuse those programs. For example, sky, grass, the direction of a 'text' or that of a 'head' can be clues showing image direction. To remove such clues, one can use a sub-image that consists of fewer clues than the whole image [47]. Another approach is to use global transformations and feature deformations [16].

In summary, the strategies make recognition harder include:

- Using random strings instead of words,
- Making characters have the same pixel count,
- Using different fonts,
- Selecting words or images that cannot be recognized by current recognition systems,
- Using a large database,
- Using visually-similar but semantically-dissimilar characters/images/audio in tests, and
- Removing or deforming objects' features.

4.3 *Random guessing resistance*

In text-based CAPTCHAs, having a large character set will reduce the chance of blind guessing. In many current image-based CAPTCHAs, a test has only a few potential answers; which makes random guessing attacks easier for robots. Some security considerations to defeat this attack include:

Using mouse actions to select the correct answer: For example, if the test utilizes a 200x200 image and displays 6 potential objects, then the user needs to select a number between 1 and 6. Therefore, the probability of blind guessing will be 1/6 (or 0.17); however, if she is required to click near the center of the correct answer, e.g. Inside a radius of four pixels from the center, the success rate of blind guessing will be $\frac{8\pi 4^2}{200 \times 200}$ (or 0.0075) [35].

Multiple challenge-response system: Repetition of the test reduces the success rate of blind guessing attacks [61], [34].

Increasing the image size, increasing the number of potential answers [79], dynamic presentation of the CAPTCHA (animated answers) [58] and limiting the number of attempts to solve a test [77] are other strategies to make a CAPTCHA stronger against random guessing.

4.4 *Security against 3rd party human solver attack*

Techniques to defeat 3rd party attacks include:

- Limiting the number of attempts to solve a CAPTCHA test [77],

- Limiting the time of the validity of a CAPTCHA [77],
- Detecting IP addresses that give successive incorrect answers [90],
- Measuring solution time which is different for a legitimate user and a 3rd party human [29].
- Making the CAPTCHA image meaningful only on the protected web site [29].

4.5 Other security measures

Other strategies to improve the robustness of CAPTCHA systems are discussed in this section. Most of them try to highlight the gap between human and machine abilities in solving problems and recognizing objects. These strategies include:

- Using sentences as the source of the CAPTCHA instead of characters, words or images: this technique uses human ability in recognizing natural sentences and detecting machine-translated mistakes [25].
- Asking users to answer a logical question that requires human thinking [62].
- Using 3D images or characters based on the fact that humans are better than machines in recognizing 3D objects [50].
- Using structures that humans recognize better than machines; such as trees [63].
- Combining text and graphics in the tests: most recognition tools are domain specific; they work exclusively with graphics or text. This strategy can confuse those tools [62].
- Requiring more human interaction, e.g. using mouse click, dropdown list or drag-n-drop for answering CAPTCHA tests will reduce the risk of blind guessing attacks [91].

5 Usability of CAPTCHAs

In the development of a CAPTCHA, achieving a balance between usability and security is very important. This section discusses usability issues that should be considered in the design of CAPTCHAs. Some of these considerations are generic and can help every CAPTCHA to be more usable, some of them are specific strategies depending on the type of the CAPTCHA.

In this section, the usability issues of CAPTCHAs will be discussed under three dimensions:

- Distortion,
- Content,
- Presentation.

5.1 Distortion

Distortion method and level should be selected very carefully since many types of distortions not only make CAPTCHA less usable, but also reduce security control because system would have to ignore some users' mistakes or allow multiple attempts for failed tests [87]. Examples of clutter that confuse humans and do not improve security in text-based CAPTCHAs include:

- Every background noise that is not similar to foreground objects (Figure 69-a),
- Arcs thinner than the characters (Figure 69-b),
- Arcs with the same size as the characters (Figure 69-c: the first vertical arc might be confused with letter 'I')

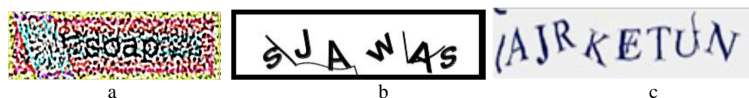


Figure 69: examples of degradations that confuse humans and do not improve security; a) EZ-Gimpy[7], b) Yahoo v.2.0[13]; and c) MSN [87]

In contrast with examples of Figure 69, Figure 70 shows a CAPTCHA [64] in which clutters are similar to the test objects. This CAPTCHA confuses machines, but it is still usable.

Some CAPTCHAs are more distortion-tolerant which means that a higher amount of distortion has a few effects on their usability. For example, in the CAPTCHA of Figure 71, the user does not need to recognize the objects completely; they only need to detect the orientation of the image [44].

Sounds in Audio-based CAPTCHAs are distorted by background noise. It can affect usability in such a way that characters might be confused with each other. For example the user might not be able to tell 'v' and 'b' apart [92].



Figure 70: An examples of degradations that do not confuse humans and improve security: CAPTCHA zoo [64]

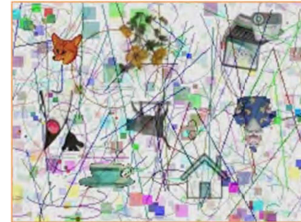


Figure 71: An example of a distortion-tolerant CAPTCHA: Image Flip CAPTCHA [44]

5.2 *Content*

One of the most important considerations in selecting the contents of a CAPTCHA should be its usability. Important points about the contents of a text-based CAPTCHA include:

- The size of the character set: although a large character set improves security, it might have negative effects on usability since a bigger character set implies a higher number of unknown characters and visually similar characters [93].
- Using random strings or words in CAPTCHAs: Using random strings makes the CAPTCHA more difficult for the human, while it increases robustness [87].
- The length of the string: While long strings increase robustness by reducing the success rate of random guessing, they decrease usability [87].
- Using ambiguous characters such as 'cl' and 'd' might confuse users [87].

Generally, the designer of a CAPTCHA can consider the following points about the contents of the CAPTCHA to make it more usable:

- Being not offensive [87],
- Being independent of a certain language, age or knowledge level [39] (dependency to a certain language is one of the characteristics of audio-based CAPTCHAs),
- Being suitable for all people including those with disabilities [60],
- Being usable not only on PCs, but also on smartphones [64],
- Bringing other social benefits such as helping to read and digitize old scripts (reCAPTCHA [9]), finding home for pets (Asirra [34]), etc.
- Selecting a type of CAPTCHA among different available CAPTCHA systems based on the user's information [60].

5.3 Presentation

The presentation of a CAPTCHA and its user-interface should be designed to enhance usability.

In text-based CAPTCHAs, font type, font size and image size affect CAPTCHA usability. In some fonts, some characters might be similar and not easily distinguishable by humans. Large font sizes are usually more convenient for human users since they improve visibility [87].

In image-based CAPTCHAs, the size of the CAPTCHA image is a factor of usability. Small images reduce server-processing time, and accelerate download process and occupy less space in webpage [94]. On the other hand, large images are more visible and thus easier for humans to response; they also reduce the probability of blind guessing and improve security. The size of the image should be decided to make a balance between usability and robustness.

In audio-based CAPTCHAs, presentation of the CAPTCHA does not generally affect the usability. However, Bigham and Cavender [92] discussed that most current audio CAPTCHAs are frustrating for blind users who use screen readers to access the CAPTCHA. The reason is that using navigation elements to listen to and answer the CAPTCHA distracts them and forces them to miss the beginning of the CAPTCHA. Improving the CAPTCHA interface to solve this issue can increase the usability of audio-based CAPTCHAs for blind users.

In motion-based CAPTCHAs, the load time can be changed according to the users' network limitations to enhance usability. It can be performed by the animation quality and dimensions; and by converting it to grayscale [95].

Other strategies to improve usability of CAPTCHAs under presentation dimension include:

- Appropriate use of colors: color can facilitate human recognition and confuse OCR systems. However, colors should be used properly in order not to cause negative impacts on security or usability [96]. For example, fancy colorful schemes or color patterns should not preferably be exploited because they usually confuse human, and also fail to resist attacks. An example is displayed in Figure 72.



Figure 72: a colourful background that can be easily extracted by OCR programs [87]

- CAPTCHAs' user interface: In different CAPTCHA systems, the users are asked to enter their answers by different methods such as typing the answer, selecting from a dropdown list, clicking the answer; or dragging and dropping answers to boxes. While the most common method is typing, mouse interaction methods are more usable since they simplify and accelerate the answering process. These methods improve robustness as well [91].
- Partial credit algorithm [34]: Another strategy to enhance usability is to give users partial credit which means if they solve a test almost correctly (e.g. 7/8), they are considered as possible 'human users' and are shown another test. By passing the second test almost correct, the user is identified as a human and gets access to the protected resource. It is a two-step algorithm that means two almost-correct answers are required. Converting it to a one-step algorithm, which requires only one almost-correct answer (as can be seen in reCAPTCHA [9]), would cause security problems.

6 Discussion and Conclusions

In the development of most human-interaction centric security mechanisms, a trade-off between security and usability is required. A strategy that increases the security, in many cases reduces the usability and vice versa. In the generation of CAPTCHAs, the weaknesses and limitations of many text-based CAPTCHAs have made them vulnerable to attacks. On the other hand, attempts to increase their security have often made them very difficult for humans. Hence, CAPTCHA developers have been trying to explore alternative models to design more usable and secure CAPTCHAs. Image-based, audio-based, motion-based and hybrid CAPTCHAs were proposed a long time ago to overcome the restrictions of text-based CAPTCHAs. The drawback of these types of CAPTCHA is that preparing a substantially large database of images, audio files or animation clips that does not require human intervention for categorization or labelling is close to impossible. These limitations make many of them vulnerable to attacks. To reduce the effects of these limitations, new generation CAPTCHAs, interactive CAPTCHAs, have been developed to increase the complexity of traditional CAPTCHAs for attackers. Interactive CAPTCHAs involve more human intervention in their tests. They use mouse actions such as clicking or drag-n-drop to induce a form of response that is easier to produce for humans and more difficult for robots. They offer a more enjoyable user-friendly human verification system. In addition, they improve the security by adding new layers of complexity required to attack them. Based on this argument, interactive CAPTCHAs seem to be the most promising CAPTCHA type.

In this article, we have introduced different types of CAPTCHAs (Table 2) and attacks against them. Many attacks, especially on non-text-based CAPTCHAs, are CAPTCHA specific. However, some of the most well-known strategies in attacking CAPTCHAs are summarized in Table 3. We have also investigated the weaknesses of current CAPTCHAs that make them vulnerable to these attacks. Table 4 illustrates these flaws.

Table 2: Different types of CAPTCHAs

Text-based	“English words” CAPTCHAs
	“Random strings” CAPTCHAs
	Handwritten text CAPTCHAs
	Linguistic knowledge CAPTCHAs
	Interactive Text-based CAPTCHAs
	Non-English CAPTCHAs
Image-based	CAPTCHAs based on detecting a certain object among other objects
	CAPTCHAs based on detecting the common characteristic of objects
	CAPTCHAs based on detecting a specific part of the test image
	CAPTCHAs based on swapping the misplaced parts of the test image
	Orientation-based CAPTCHAs
	3D CAPTCHAs
	Audio-based
Motion-based	
Hybrid	Dynamic CAPTCHAs
	CAPTCHAs with multiple challenges
	Multi-type CAPTCHAs
	CAPTCHAs for smartphones
	CAPTCHAs for people with disability

Table 3: Attacks on CAPTCHAs

Segmentation attacks	Pre-processing
	Simple segmentation
	Vertical histogram segmentation
	Color-filling segmentation
	Snake segmentation
Recognition attacks	Object recognition
	Pixel-count attack
	Dictionary attack
	Database attack
Random guessing	
Social engineering attacks	

Table 4: Major vulnerabilities of current CAPTCHAs

<i>Vulnerability source</i>	<i>Vulnerability</i>	<i>Description</i>	<i>Possible attacks</i>	<i>Examples of CAPTCHAs with this vulnerability</i>	<i>Examples of CAPTCHAs addressing this vulnerability</i>
Input space	Small input space	E.g. Latin alphabet	Recognition attacks, Random guessing	Google [13], BaffleText[14]	Asirra[34], sketcha [46]
	Using limited variations of each object.	E.g. limited number of fonts; or limited voices in audio CAPTCHAs	Object recognition, pixel-count attacks	Authorize audio [51]	ScatterType[17] , eBay Audio [51]
Similar objects	Only dissimilar objects in a test	Dissimilar in terms of shape, size, etc.	Object recognition, pixel-count	Drawing CAPTCHA [39], Google [13]	reCAPTCHA audio [51], CAPTCHA zoo [64]
	different pixel count of objects	E.g. each character has a unique pixel-count	pixel-count attacks	captchaservice [8]	Handwritten CAPTCHA [16]
Keyboard	Using physical keyboard as the input device	Causes all the limitations of a small input space	Recognition attacks, Random guessing	Most text-based CAPTCHAs	Drag-n-Drop [27]
Randomness	Constant number of objects in a test	The number of objects is known for attackers	Segmentation, random guessing	Collage [32]	Image Flip [44]
	Constant size of each object	Each object's size or width/length ratio is known	Segmentation, Random guessing	captchaservice[8]	Imagination [35]
	Constant position of the objects	E.g. characters located on a horizontal line, in the middle of the image	Segmentation, Random guessing	Collage [32]	Tree-based handwritten [63]
	Constant direction of the objects	E.g. horizontal, from left to right.	Simple segmentation, Vertical histogram	Most text-based CAPTCHAs	Non text-based CAPTCHAs
	Non-random strings	Using words instead of random strings	Dictionary attacks	Gimpy [6]	Google [13]
Color	Inappropriate use of colors	Distinct colors in foreground and background, or giving information by color.	Segmentation-preprocessing	MSN [12], Ticketmaster[13]	Imagination [35]
Noise and distortions	Not having noise	Background noise, mesh, arcs, ...	Segmentation attacks	captchaservice[8]	Ticketmaster[13], yahoo audio [51]
	Noise dissimilar to test objects	Dissimilar in terms of color, shape, location, ...	Pre-processing, Segmentation attacks	Ticketmaster[13], yahoo audio [51]	reCAPTCHA audio [51], CAPTCHA zoo [64]
	Having no object distortion	text/image/audio distortions	Object recognition	Collage[32]	captchaservice[8], MSN[12]
	Non-fragmented objects	Non-fragmented characters, images without false boundaries	Segmentation attacks	captchaservice[8]	ScatterType [17], BaffleText[14]
	Not-connected objects	Non-connected characters, images with obvious boundaries	Segmentation attacks, Object recognition attack	Ticketmaster[13]	Google [13]
Design	Having minimal solution requirements	e.g. allowing errors in answering tests; or requiring selection of just one (of N) object voluntarily.	Segmentation, Object recognition, Random guessing	Imagination [35]	Google [13]
	No restrictions on 'download limit' or 'error limit'	No limit on the number of CAPTCHA can be downloaded, or submitted with an incorrect response	Social Engineering attacks	eBay audio [90]	Asirra [34]

Based on this study, major vulnerabilities of current CAPTCHAs include:

- **Small-size input set:** Many current CAPTCHAs have small input spaces that make object recognition or blind guessing easy for attackers.
- **Dissimilar objects:** Most existing CAPTCHAs do not use similar elements in their tests. It is more straightforward for an attacker to distinguish between “dissimilar” characters.
- **Receiving input from a physical keyboard:** Using a standard keyboard implies having a limited set of characters which is a drawback for security.
- **Non-randomness in the presentation:** The existence of fixed and predictable patterns in the layout of elements of a CAPTCHA makes them vulnerable to segmentation attacks.
- **Inappropriate use of colors:** In many current CAPTCHA schemes, inappropriate use of colors negatively affects security and usability.
- **Lack of proper degradations:** The level and type of the distortions have not been decided properly in many current CAPTCHAs.
- **Lack of limits on the number of attempts to solve tests:** Allowing users, and apparently attackers, to attempt several times to pass a challenge has a negative impact on security.

Considering the discussed vulnerabilities of current CAPTCHAs, we propose a new interactive CAPTCHA that incorporates our suggested solutions to the major flaws in existing CAPTCHAs. Figure 73 illustrates a possible implementation of this idea. This CAPTCHA will be composed of a test (left rectangle) and a keyboard (right rectangle).

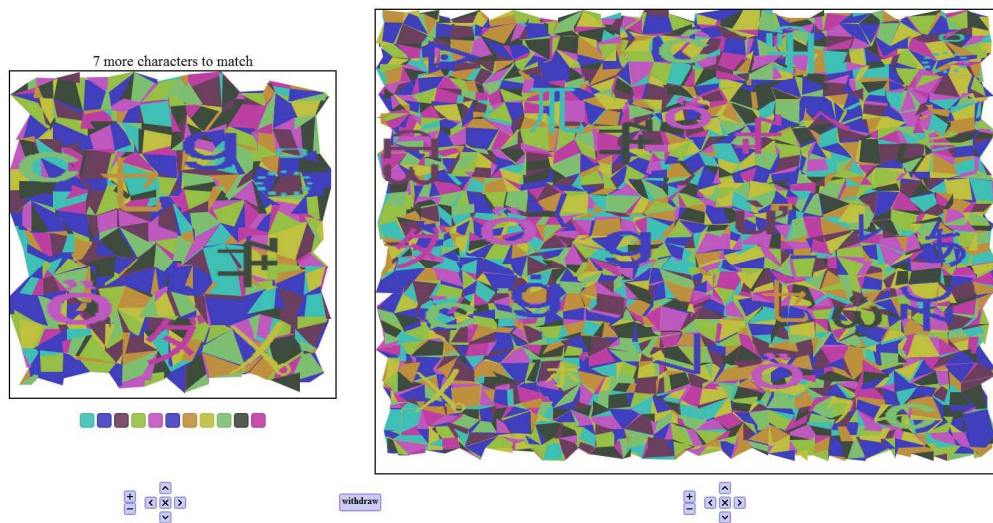


Figure 73: The proposed CAPTCHA

In this CAPTCHA, the user will be asked to find the correct match for each test Unicode character in the keyboard. The user can use zooming facilities or the color palette to reduce the complexity of the noisy background and detect the characters more easily. Strategies that will be applied in designing this CAPTCHA that seek to reduce the mentioned vulnerabilities include: using the large Unicode as the input space, including similar objects in the keyboard, using a virtual keyboard instead of a physical keyboard, incorporating randomness in the design including random size, number and position of the test and keyboard characters; and, a color representation that takes into account security and usability considerations.

These approaches are discussed in the remainder of this section.

Input Space: The small input space of most current CAPTCHAs, including text-based, image-based, audio-based and motion-based CAPTCHAs, reduces their security by allowing easier database attacks, automatic object recognition or blind guessing attacks. Almost no text-based CAPTCHA has considered using a large input space. This unwillingness to incorporate larger character sets in CAPTCHAs might stem from two groups of complications: a) CAPTCHA design implications b) user experience implications. A standard keyboard which is the main input device for the majority of text-based CAPTCHAs does not readily support a large number of characters. It only supports Latin characters – or only a few selected character sets at any time. Using a larger character space would require CAPTCHA designers to design virtual keyboards that would cause additional difficulties for them. Another major concern in adopting these extensions lies in designers’ prediction of negative user reactions. Using unfamiliar input spaces would lead to higher cognitive loads on CAPTCHA solvers which could cause dissatisfaction or discomfort for them. CAPTCHA users experience lower cognitive loads in solving tests whose elements include or resemble their known character sets [93].



On the other hand, the supporters of image-based CAPTCHAs believe that in contrast to text-based CAPTCHAs that have a small input space, image-based CAPTCHAs can enjoy a large amount of images coming from public or private databases [34] through the Internet. However, populating the database is a major issue with all image-based CAPTCHAs. An important problem with using images is that the CAPTCHA producer algorithm does not know the meaning of each image unless it is separately available. Consequently, unlike text CAPTCHAs which can use any random combination of characters in their challenges, descriptive information for images in image CAPTCHAs are required to be provided by a human. Labelling the name or category of each image by a human produces extra initial costs in terms of both time and money. Moreover, there may be legal issues in using the crawled images directly.

According to the above arguments, the lack of a large input space that can be produced automatically is still an issue. Unicode, with a potential capacity of over 1 million characters and approximately 110,000 encoded characters, is our solution for this problem. Using a variety of fonts and font variations (size, style, weight, width, etc.) to represent Unicode characters makes the input space much larger. For example, if 10 fonts and 4 font variations can display each character, the size of the input space will be 4,400,000. In this large input space, every element is referable and retrievable without requiring human intervention.

From a security viewpoint, using such a spacious input set will highly reduce the likelihood of the challenge being solved by either pattern recognition algorithms or random guessing. From a usability viewpoint, people are more comfortable with familiar characters and the existence of unfamiliar characters in this CAPTCHA could potentially cause some complications for users. However, in the proposed CAPTCHA, the users will not have to recognize a character and its linguistic meaning; they are only required to match some shapes.

Using similar objects in tests: One of the weaknesses of current CAPTCHAs is that their tests include objects that are dissimilar. Dissimilar items can be distinguished by attackers more easily since they have features that are more different. To improve the security, a designer can employ “similar” objects. Similarity can confuse pattern recognition programs. When two images are visually similar, it is much easier for humans to recognize the slight difference between them than it is for a computer program [64]. Currently, only a few CAPTCHAs have tried to use this strategy

in their tests (e.g. [64], [51]). The problem with these CAPTCHAs is that, due to the size restrictions of their input dataset, the number of similar objects is limited, and often the similarity of objects required to be decided manually.

Unicode, containing a large number of similar characters solves this problem for us. In fact, Unicode not only gives us an automatically-generated large input space, but also provides us with a lot of similar objects. An example of similar objects in the Unicode is  and  which are two different Sinhala characters. Although including similar objects in tests leads to stronger CAPTCHAs, solving such tests is also more challenging for humans. Hence, only a limited number of similar objects should be included in a test.

Virtual keyboard: Physical keyboards have been the main character input device since the advent of modern computers. Despite the general ease of use that standard keyboards afford computer users, their adoption for solving CAPTCHAs has introduced a major design constraint, namely, the limitation of the input space to the one offered by the keyboard. It is conceivable that removing this constraint from the CAPTCHA design process opens new avenues for developing more secure CAPTCHAs.

Using virtual keyboards instead of a physical keyboard in a CAPTCHA adds extra burden on any algorithm trying to break it. A virtual keyboard is an image which is a (potentially varying) part of the screen; a high computational complexity is required to process a robust virtual keyboard and segment the characters by attackers who capture the screen and try to analyse the screenshot in order to break the keyboard. From a usability viewpoint, a virtual keyboard does not put extra load on the user. In fact, it is even easier for some users to use the mouse to input information by a virtual keyboard rather than typing on a standard keyboard.

In the proposed CAPTCHA, resorting to Unicode as the character set of the CAPTCHA implies the need for a virtual keyboard. We will also employ strategies that can make the virtual keyboard more secure. These strategies include randomizing the number of keyboard characters, their sizes and their positions and having a noisy background to impede segmentation for machines.

Randomness in the presentation of CAPTCHA test: What makes a robot powerful in segmenting current CAPTCHA images is invariance and using limited patterns in their presentations. Increasing the amount of randomness will confuse machines and fail them to determine the correct number of segments in a test image. In text-based CAPTCHAs, instead of regular rectangular box that most current CAPTCHAs use, other shapes such as a circular box can be exploited. Characters can be juxtaposed in any direction such as right to left or vertical; or they can be positioned randomly to make segmentation harder. The number of characters could also vary from one challenge to another [82]. For image-based CAPTCHAs, random positioning of the images, different-shaped boxes and random background dimensions and for animated CAPTCHAs, random number of frames and random frame delay [95] can reduce the probability of segmentation attacks.

In the presentation of the proposed CAPTCHA, a lot of Randomness will be employed including random number, size, location and color of the test and keyboard characters.

Appropriate use of colors: Applying color to CAPTCHA tests can improve security since both OCR programs and segmentation algorithms perform poorly while dealing with color images. In the proposed CAPTCHA, multiple colors will be used in both background and foreground. The same colors exist in background and foreground to make the distinction between foreground and background difficult for an attacker. To increase randomness, the number of the colors and their

hue will be selected randomly. Moreover, Instead of solid colors, color gradients will be employed to prevent attackers from using ‘color’ as a clue for segmentation.

Applying degradations properly: One strategy to produce robust CAPTCHAs is to apply appropriate degradations to the test. Various distortions proposed to degrade text-based CAPTCHAs including blurring, adding random noise, interference by random-shape masks [14], background clutters such as grids and gradients [13]; random shearing, adding intersecting or non-intersecting arcs; and crowding letters together to remove white spaces between them [13]. Similarly, for image-based CAPTCHAs, degradations such as overlapping different images of a test to hide their boundaries, creating false boundaries [35] and inserting background noise, which is similar to test objects, have been proposed. For audio-based CAPTCHAs, adding noise such as white noise, sine waves, cracks, and voices similar to foreground sound is suggested [51]. However, it should be noted that adding distortions directly affects the usability of a CAPTCHA.

Recognizing objects in an over-distorted test is difficult or impossible for humans. Hence, it is very important to determine what distortion method should be applied to the test. Sometimes distortions not only reduce human recognition, but they also do not improve the robustness of the system. Such distortions can be removed by simple image processing methods [87].

In the proposed CAPTCHA, the availability of background noise which can be similar to some of the Unicode characters in terms of shape, color or location can potentially baffle attackers. In order to improve the usability and to help human users to recognize target objects in the noisy background, they will be provided with zooming and color-filtering tools.

Limiting the number of attempts to solve a test: A strategy to make CAPTCHAs stronger is imposing restrictions on the number of CAPTCHAs a user can try, the maximum number of attempts to solve a test, and the duration of the validity of a test [90]. If an IP address tries to download a large number of CAPTCHAs, it might be a robot trying to collect the CAPTCHA database. If an IP address performs several attempts on a test or spends a lot of time solving a test, it is probably a computer program trying to solve the test by random guessing or image processing attacks. Such IPs can be deprived of the service after a maximum number of attempts or at a given time [77].

References

1. L. Von Ahn, M. Blum, and J. Langford, "Telling humans and computers apart automatically," *Communications of the ACM*, vol. 47, pp. 56-60, 2004.
2. A. L. Coates, H. S. Baird, and R. J. Fateman, "PessimPrint: a reverse Turing test," *International Journal on Document Analysis and Recognition*, vol. 5, pp. 158-163, 2003.
3. J. Yan, "Bot, cyborg and automated turing test," in *Security Protocols Workshop*, 2006, pp. 190-197.
4. H. Baird and K. Papat, "Human interactive proofs and document image analysis," presented at the The 5th IAPR International Workshop on Document Analysis Systems (DAS 2002), 2002.
5. E. Bursztein, S. Bethard, C. Fabry, J. C. Mitchell, and D. Jurafsky, "How good are humans at solving CAPTCHAs? a large scale evaluation," in *2010 IEEE Symposium on Security and Privacy (SP)*, 2010, pp. 399-413.
6. C. Pope and K. Kaur, "Is it human or computer? Defending e-commerce with CAPTCHAs," *IT professional*, vol. 7, pp. 43-49, 2005.
7. M. Blum, L. Von Ahn, J. Langford, and N. Hopper, "The CAPTCHA project, "Completely automatic public turing test to tell computers and humans apart,"" *Dept. of Computer Science, Carnegie-Mellon University, www.captcha.net*, 2000.
8. T. Converse, "CAPTCHA generation as a web service," *Human Interactive Proofs*, vol. 3517, pp. 82-96, 2005.
9. L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum, "reCAPTCHA: Human-based character recognition via web security measures," *Science*, vol. 321, pp. 1465-1468, 2008.
10. M. Chew and J. Tygar, "Collaborative filtering CAPTCHAs," *The 2nd International Conference on Human Interactive Proofs (HIP 2005)*, pp. 66-81, May 2005.
11. (2008, October 8, 2012). *reCAPTCHA*. Available: <http://www.google.com/recaptcha>
12. M. Shirali-Shahreza, "Highlighting CAPTCHA," in *2008 Conference on Human System Interactions*, 2008, pp. 247-250.
13. K. Chellapilla, K. Larson, P. Simard, and M. Czerwinski, "Designing human friendly human interaction proofs (HIPs)," in *ACM Conference on Human Factors in Computing Systems (CHI 05)*, 2005, pp. 711-720.
14. M. Chew and H. S. Baird, "BaffleText: A human interactive proof," presented at the 10th Document Recognition & Retrieval Conference (SPIE), 2003.
15. G. Kepes, "Language of vision.[Chicago], P," ed: Theobald, 1944.
16. A. Rusu, A. Thomas, and V. Govindaraju, "Generation and use of handwritten CAPTCHAs," *International journal on document analysis and recognition*, vol. 13, pp. 49-64, 2010.
17. H. S. Baird, M. A. Moll, and S. Y. Wang, "ScatterType: A legible but hard-to-segment CAPTCHA," in *8th International Conference on Document Analysis and Recognition*, 2005, pp. 935-939.
18. (2012, Oct. 8). *ebay*. Available: www.ebay.ca
19. (2012, Oct. 8). *PHP Class CAPTCHA*. Available: <http://www.nogajski.de/priv/php/captcha/>
20. (2012, Jan. 8). *MegaUpload*. Available: www.megaupload.com
21. A. Gupta, A. Jain, A. Raj, and A. Jain, "sequenced tagged CAPTCHA: generation and its analysis," in *IEEE International Advance Computing Conference 2009 (IACC 2009)*, 2009, pp. 1286-1291.
22. A. Raj, A. Jain, T. Pahwa, and A. Jain, "Analysis of tagging variants of Sequenced Tagged CAPTCHA (STC)," in *IEEE Toronto International Conference on Science and Technology for Humanity (TIC-STH 2009)*, 2009, pp. 427-432.
23. A. O. Thomas, A. Rusu, and V. Govindaraju, "Synthetic handwritten CAPTCHAs," *Pattern Recognition*, vol. 42, pp. 3365-3373, 2009.
24. P. Lupkowski and M. Urbanski, "SemCAPTCHA—user-friendly alternative for OCR-based CAPTCHA systems," in *International Multiconference on Computer Science and Information Technology (IMCSIT 2008)*, 2008, pp. 325-329.

25. T. Yamamoto, J. Tygar, and M. Nishigaki, "CAPTCHA using strangeness in machine translation," in *The 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, 2010, pp. 430-437.
26. R. Bergmair and S. Katzenbeisser, "Towards human interactive proofs in the text-domain: Using the problem of sense-ambiguity for security," presented at the The 7th International Information Security Conference (ISC 2004), 2004.
27. A. Desai and P. Patadia, "Drag and Drop: A Better Approach to CAPTCHA," in *Annual IEEE India Conference (INDICON)*, 2009, pp. 1-4.
28. P. Golle and N. Ducheneaut, "Keeping bots out of online games," in *The 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology (ACE '05)*, 2005, pp. 262-265.
29. H. D. Truong, C. F. Turner, and C. C. Zou, "iCAPTCHA: the next generation of CAPTCHA designed to defend against 3rd party human attacks," in *IEEE International Conference on Communications (ICC)*, 2011, pp. 1-6.
30. B. Khan, K. Alghathbar, M. Khan, A. AlKelabi, and A. AlAjaji, "Using Arabic CAPTCHA for Cyber Security," in *Security Technology, Disaster Recovery and Business Continuity*. vol. 122, ed: Springer Berlin Heidelberg, 2010, pp. 8-17.
31. M. S. Shahreza, "Verifying Spam SMS by Arabic CAPTCHA," in *2nd IEEE International Conference on Information and Communication Technologies (ICTTA '06)*, 2006, pp. 78-83.
32. M. Shirali-Shahreza and S. Shirali-Shahreza, "Collage CAPTCHA," in *9th International Symposium on Signal Processing and Its Applications (ISSPA 2007)*, 2007, pp. 1-4.
33. M. H. Shirali-Shahreza and M. Shirali-Shahreza, "Multilingual CAPTCHA," in *5th IEEE International Conference on Computational Cybernetics (ICCC 2007)*, 2007, pp. 135-139.
34. J. Elson, J. R. Douceur, J. Howell, and J. Saul, "Asirra: a CAPTCHA that exploits interest-aligned manual image categorization," *14th ACM conference on Computer and Communications Security (CCS 2007)*, pp. 366-374, Oct.-Nov. 2007.
35. R. Datta, J. Li, and J. Z. Wang, "IMAGINATION: a robust image-based CAPTCHA generation system," in *13th ACM International Conference on Multimedia (Multimedia 05)*, 2005, pp. 331-334.
36. R. Datta, J. Li, and J. Z. Wang, "Exploiting the Human-Machine Gap in Image Recognition for Designing CAPTCHAs," *IEEE Transactions on Information Forensics and Security*, vol. 4, pp. 504-518, Sep 2009.
37. E. Vimina and A. U. Areekal, "Telling computers and humans apart automatically using activity recognition," in *IEEE International Conference on Systems, Man and Cybernetics (SMC 2009)*, 2009, pp. 4906-4909.
38. H. S. Baird and J. L. Bentley, "Implicit CAPTCHAs," in *SPIE-IS&T Electronic Imaging, Document Recognition and Retrieval*, 2005, pp. 191-196.
39. M. Shirali-Shahreza and S. Shirali-Shahreza, "Drawing CAPTCHA," in *28th International Conference on Information Technology Interfaces (ITI 2006)*, Cavtat, Dubrovnik, Croatia, 2006, pp. 475-480.
40. A. Karunathilake, B. Balasuriya, and R. Ragel, "User friendly line CAPTCHAs," in *International Conference on Industrial and Information Systems (ICIIS 2009)*, 2009, pp. 210-215.
41. Y. Rui and Z. Liu, "ARTiFACIAL: automated reverse turing test using FACIAL features," *Multimedia Systems*, vol. 9, pp. 493-502, 2004.
42. W. H. Liao, "A CAPTCHA mechanism by exchange image blocks," in *18th International Conference on Pattern Recognition (ICPR 2006)*, 2006, pp. 1179-1183.
43. H. Gao, D. Yao, H. Liu, X. Liu, and L. Wang, "A Novel Image Based CAPTCHA Using Jigsaw Puzzle," in *13th IEEE International Conference on Computational Science and Engineering (CSE)*, 2010, pp. 351-356.
44. M. Bandy and N. Shah, "Image flip CAPTCHA," *ISC International Journal of Information Security (ISeCure)*, vol. 1, pp. 105-123, 2009.

45. R. Gossweiler, M. Kamvar, and S. Baluja, "What's up CAPTCHA?: a CAPTCHA based on image orientation," in *18th International Conference on World Wide Web 2009*, pp. 841-850.
46. S. A. Ross, J. A. Halderman, and A. Finkelstein, "Sketcha: a CAPTCHA based on line drawings of 3D models," in *19th International Conference on World Wide Web*, 2010, pp. 821-830.
47. J. W. Kim, W. K. Chung, and H. G. Cho, "A new image-based CAPTCHA using the orientation of the polygonally cropped sub-images," *The Visual Computer*, vol. 26, pp. 1135-1143, 2010.
48. M. E. Hoque, D. J. Russomanno, and M. Yeasin, "2D CAPTCHAs from 3D models," in *IEEE SoutheastCon 2006*, 2005, pp. 165-170.
49. (Jan. 1, 2012). *Spamfizzle CAPTCHA*. Available: <http://spamfizzle.com/CAPTCHA.aspx>
50. M. Imsamai and S. Phimoltares, "3D CAPTCHA: A next generation of the CAPTCHA," in *International Conference on Information Science and Applications (ICISA)*, 2010, pp. 1-8.
51. E. Bursztein, R. Beauxis, H. Paskov, D. Perito, C. Fabry, and J. Mitchell, "The Failure of Noise-Based Non-continuous Audio CAPTCHAs," in *2011 IEEE Symposium on Security and Privacy (SP)*, 2011, pp. 19-31.
52. S. Shirali-Shahreza, H. Abolhassani, H. Sameti, and M. H. Shirali-Shahreza, "Spoken CAPTCHA: A CAPTCHA system for blind users," in *ISECS International Colloquium on Computing, Communication, Control, and Management (CCCM 2009)*, 2009, pp. 221-224.
53. T. Y. Chan, "Using a test-to-speech synthesizer to generate a reverse Turing test," in *IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2003)*, 2003, pp. 226-232.
54. G. Sauer, H. Hochheiser, J. Feng, and J. Lazar, "Towards a universally usable CAPTCHA," in *4th Symposium On Usable Privacy and Security (SOUPS '08)*, Pittsburgh, 2008.
55. G. Kochanski, D. Lopresti, and C. Shih, "A reverse turing test using speech," in *7th International Conference on Spoken Language Processing*, 2002, pp. 1357-1360.
56. (2010, October 8, 2012). *NUCAPTCHA*. Available: <http://www.nucaptcha.com/>
57. (October 8, 2012). *HelloCAPTCHA*. Available: <http://www.hellocaptcha.com/>
58. E. Athanasopoulos and S. Antonatos, "Enhanced CAPTCHAs: Using animation to tell humans and computers apart," in *10th International Conference on Communications and Multimedia Security (CMS 2006)*, 2006, pp. 97-108.
59. J. S. Cui, J. T. Mei, W. Z. Zhang, X. Wang, and D. Zhang, "A CAPTCHA implementation based on moving objects recognition problem," in *International Conference on E-Business and E-Government (ICEE)*, 2010, pp. 1277-1280.
60. M. Shirali-Shahreza and S. Shirali-Shahreza, "Dynamic CAPTCHA," in *International Symposium on Communications and Information Technologies (ISCIT)*, 2008, pp. 436-440.
61. O. Longe, A. Robert, and U. Onwudebelu, "Checking Internet masquerading using multiple CAPTCHA challenge-response systems," in *The 2nd International Conference on Adaptive Science & Technology (ICAST 2009)*, 2009, pp. 244-249.
62. M. Shirali-Shahreza and S. Shirali-Shahreza, "Question-based CAPTCHA," in *International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007)*, 2007, pp. 54-58.
63. A. Rusu, R. Docimo, and A. Rusu, "Leveraging cognitive factors in securing WWW with CAPTCHA," in *The 2010 USENIX conference on Web application development (WebApps'10)*, 2010.
64. R. Lin, S. Y. Huang, G. B. Bell, and Y. K. Lee, "A new CAPTCHA interface design for mobile devices," in *Australasian User Interface Conference, Australasian Computer Science Week (ACSW2011)*, 2011.
65. M. H. Shirali-Shahreza and M. Shirali-Shahreza, "Localized CAPTCHA for illiterate people," in *International Conference on Intelligent and Advanced Systems (ICIAS)*, 2007, pp. 675-679.

66. J. Holman, J. Lazar, J. H. Feng, and J. D'Arcy, "Developing usable CAPTCHAs for blind users," in *9th international ACM SIGACCESS conference on Computers and accessibility*, 2007, pp. 245-246.
67. M. Shirali-Shahreza and S. Shirali-Shahreza, "CAPTCHA for blind people," in *7th IEEE International Symposium on Signal Processing and Information Technology (ISSPIT 2007)*, 2007, pp. 995-998.
68. S. Shirali-Shahreza and M. Shirali-Shahreza, "A new human interactive proofs system for deaf persons," in *5th International Conference on Information Technology: New Generations (ITNG 2008)*, 2008, pp. 807-810.
69. J. Yan and A. S. El Ahmad, "A Low-cost Attack on a Microsoft CAPTCHA," in *15th ACM Conference on Computer and Communications Security (CCS 08)*, 2008, pp. 543-554.
70. K. Chellapilla, Simard, P., "Using machine learning to break visual human interaction proofs (HIPs)," *Advances in Neural Information Processing Systems*, vol. 17, pp. 265-272, 2004.
71. J. Yan and A. S. El Ahmad, "Breaking visual CAPTCHAs with naive pattern recognition algorithms," in *The 23rd Annual Computer Security Applications Conference (ACSAC 07)*, 2007, pp. 279-291.
72. J. Yan and A. S. El Ahmad, "CAPTCHA Security A Case Study," *Ieee Security and Privacy*, vol. 7, pp. 22-28, Jul-Aug 2009.
73. P. Golle, "Machine learning attacks against the Asirra CAPTCHA," in *The 15th ACM conference on Computer and communications security (CCS 2008)*, 2008, pp. 535-542.
74. C. W. Lin, Y. H. Chen, and L. G. Chen, "Bio-Inspired Unified Model of Visual Segmentation System for Captcha Character Recognition," *2008 Ieee Workshop on Signal Processing Systems: Sips 2008, Proceedings*, pp. 158-163, 2008.
75. G. Mori and J. Malik, "Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2003, pp. 134-141.
76. G. Moy, N. Jones, C. Harkless, and R. Potter, "Distortion estimation techniques in solving visual CAPTCHAs," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2004, pp. 23-28.
77. J. Wilkins. (2009, Oct. 8, 2012). *Strong CAPTCHA guidelines*. Available: <http://bitland.net/captcha.pdf>
78. R. Beede, "Analysis of reCAPTCHA effectiveness," University of Colorado at BoulderDec. 2010.
79. B. B. Zhu, J. Yan, Q. Li, C. Yang, J. Liu, N. Xu, *et al.*, "Attacks and design of image recognition CAPTCHAs," in *The 17th ACM conference on Computer and communications security (CCS '10)*, 2010, pp. 187-200.
80. E. Bursztein. (2012, October 8). *How we broke the NuCaptcha video scheme and what we propose to fix it*. Available: <http://elie.im/blog/security/how-we-broke-the-nucaptcha-video-scheme-and-what-we-propose-to-fix-it/#.T-tDK7VfGIA>
81. V. Nguyen, Y. W. Chow, and W. Susilo, "Breaking an Animated CAPTCHA Scheme," in *The 10th International Conference on Applied Cryptography and Network Security (ACNS'12)*, 2012, pp. 12-29.
82. J. Yan and A. S. El Ahmad, "CAPTCHA Robustness: A Security Engineering Perspective," *Computer*, vol. 44, pp. 54-60, Feb 2011.
83. L. Kang and J. Xiang, "CAPTCHA phishing: a practical attack on human interaction proofing," in *The 5th International Conference on Information security and cryptology (Inscrypt)*, 2011, pp. 411-425.
84. (October 8, 2012). *Adobe Flash*. Available: <http://get.adobe.com/flashplayer/>
85. K. Chellapilla, K. Larson, P. Simard, and M. Czerwinski, "Building segmentation based human-friendly human interaction proofs (HIPs)," presented at the The 2nd International Workshop on Human Interactive Proofs (HIP 2005), 2005.

86. K. Chellapilla, K. Larson, P. Simard, and M. Czerwinski, "Computers beat humans at single character recognition in reading based human interaction proofs (HIPs)," in *The 2nd Conference on Email and Anti-Spam*, 2005.
87. J. Yan and A. S. El Ahmad, "Usability of CAPTCHAs or usability issues in CAPTCHA design," in *The 4th symposium on Usable privacy and security (SOUPS)*, 2008, pp. 44-52.
88. J. Bentley and C. Mallows, "CAPTCHA challenge strings: Problems and improvements," in *The 18th SPIE-IS&T Electronic Imaging, Document Recognition and Retrieval*, 2006.
89. L. Von Ahn and L. Dabbish, "Labeling images with a computer game," in *The SIGCHI Conference on Human Factors in Computing Systems (CHI '04)*, 2004, pp. 319-326.
90. E. Bursztein and S. Bethard, "Decaptcha: breaking 75% of eBay audio CAPTCHAs," in *The 3rd USENIX conference on Offensive technologies (WOOT'09)*, 2009.
91. S. K. Chaudhari, A. R. Deshpande, S. B. Bendale, and R. V. Kotian, "3D drag-n-drop CAPTCHA enhanced security through CAPTCHA," in *The International Conference and Workshop on Emerging Trends in Technology*, Mumbai, Maharashtra, India, 2011, pp. 598-601.
92. J. P. Bigham and A. C. Cavender, "Evaluating existing audio CAPTCHAs and an interface optimized for non-visual use," in *The SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*, 2009, pp. 1829-1838.
93. B. R. Chiswick and P. W. Miller, "Linguistic distance: A quantitative measure of the distance between English and other languages," *Journal of Multilingual and Multicultural Development*, vol. 26, pp. 1-11, 2005.
94. M. Tariq Banday and N. Shah, "A Study of CAPTCHAs for Securing Web Services," *IJSDIA International Journal of Secure Digital Information Age*, vol. 1, pp. 66-74, December 2009.
95. M. Shirali-Shahreza and S. Shirali-Shahreza, "Motion CAPTCHA," in *Conference on Human System Interactions*, 2008, pp. 1042-1044.
96. A. Kolupaev and J. Ogijenko, "CAPTCHAs: Humans vs. bots," *IEEE Security & Privacy*, vol. 6, pp. 68-70, 2008.