# Introduction

CAPTCHA (*Completely Automated Public Turing test to tell Computers and Humans Apart*) or HIP (*Human Interactive Proof*) is an automatic security mechanism to distinguish whether the user is a human or a computer program. It creates and scores tests that can be solved by humans but are beyond the capabilities of present computer programs. It has evolved into the most generally utilized standard security measure for preventing automated computer program attacks. With the growth of Web services, Denial of Service (DoS) attacks by malicious automated programs have become a severe issue, and the Turing test has become a crucial approach for distinguishing people from dangerous automated programs. A human judge is authorized to pose a series of questions to two players, one of which was a computer and the other a human, and tell them apart in the original Turing test. CAPTCHA, like the Turing test, distinguishes humans from computers, but the judge is now a machine. In general, CAPTCHA is a cryptographic protocol [1] whose underlying hardness assumption is based on an AI problem. CAPTCHA implies a win-win situation: either the captcha is not broken and there is a way to differentiate humans from computers, or the captcha is broken, and a hard AI problem is solved. CAPTCHA is usually a simple visual test or puzzle that a human can complete without much difficulty, but an automated program cannot understand a complicated background to make them hard to be read by Optical Character Recognition (OCR) software. CAPTCHA has a wide variety of applications on the web and other applications such as Worms and Spam, Online Polls, Free Email Services, Preventing Dictionary Attacks and also plays a significant role in limiting usage rate.

HAI (*Human Artificial Intelligence*) researches the interactions between humans and computers, as well as the major phenomena that surround them. It denotes the usability characteristics that are firmly linked to the user interface and human factors. Hence, it is deeply involved with computer science, artificial intelligence, and cognitive psychology. The main concept in HAI is usability. From this perspective, puzzles like CAPTCHA, which humans can easily solve but computers find difficult, are an example of HAI. In this study, we provided an overview analysis of HAI under the security aspects of open concerns, difficulties, and opportunities of current CAPTCHA schemes. The remainder of this paper is organized as follows: Section II provides the taxonomy of CAPTCHA attacks. Section III describes CAPTCHA problem analysis. As a result, suggestions and recommendations are provided to build a good CAPTCHA in Section IV. Finally, Section V concludes the paper.

* 1. **CAPTCHA evolution**

The first person, Moni Naor [2], suggested theoretical approaches for distinguishing computers from humans. In 1997, the AltaVista web search engine was credited with being the first to use a CAPTCHA on the Internet [3]. Text-based CAPTCHAs were the leading technique in the early 2000s. A set of attacks were developed using image processing, pattern recognition, and machine learning (ML) algorithms to break popular text-based schemes [4]. Furthermore, anti-recognition and anti- segmentation algorithms were employed in an attempt to improve the security of existing text-based CAPTCHAs. In 2014, Google revealed that developments in AI technology could resolve distorted text variants with 99.8% [5]. Since 2004, computer vision (CV) problems, including image classification and recognition, were regarded as more difficult AI challenges than text recognition. Following that, many image-based CAPTCHA schemes with drag and drop, image selection, or sliding appeared in order to distinguish humans from computers. However, advanced CV and ML solutions aided in the defeat of the most important image- based CAPTCHA schemes between the years 2013 and 2018. Several image-based CAPTCHA schemes, such as reCAPTCHA V2 scheme, were attacked by ML [6]. Furthermore, approaches such as distortion, background noise mixing, and the use of adversarial instances were proposed as countermeasures against deep learning models. Adversarial examples by Szegedy et al. [7] and others have been suggested to enhance its security against ML-based attacks [8–10]. However, Na et al. [11] suggested a CAPTCHA solver that uses incremental learning on a limited dataset to defeat adversarial CAPTCHAs. To deal with visually impaired users, researchers proposed audio-based CAPTCHAs in addition to text-based and image-based CAPTCHAs. However, language barriers and poor usability limit the effectiveness of these schemes. Furthermore, supervised learning and automated speech recognition (ASR) [12] show how these schemes might be exploited. Researchers began developing behavioral-based CAPTCHA schemes in the 2010s to create difficulties based on behavioral features. The first behavioral-based CAPTCHA was launched by Geetest in 2012, while Google released No CAPTCHA reCAPTCHA in 2014 and invisible CAPTCHA in 2015 and 2017. Bot attacks mimicking the user’s behavioral pattern have been demonstrated to be vulnerable to these schemes [6]. Because of the serious privacy concerns, Cloudflare recently decided to discontinue the use of reCAPTCHA [13]. Finally, recent research directions use sensor data to create challenges that are difficult for automated bots to replicate. However, we must wait a sufficient amount of time before we can fully evaluate sensor-based CAPTCHAs.

* 1. **CAPTCHA codes**

CAPTCHA schemes vary and are constantly improved as a result of advancements in advanced technology, AI, and hacking techniques. Main CAPTCHA codes, shown in Fig. 1, are currently classified as cognitive/behavioral- based, video-based, audio-based, image-based, text- based, and others.

* + 1. ***Text‑based CAPTCHAs***

These CAPTCHAs became increasingly applied over the years. In these methods, the text is warped and shown to a user as an image and the user must enter this text accurately before passing this test. The AI hardness assumption is that humans can easily read the warped text, but bots using optical character recognition (OCR) techniques find it difficult. The different renderings of the challenge’s text can be classified into three subcategories: 2D, 3D, and animation. In Table 1, we list a detailed taxonomy of the typical text-based CAPTCHAs.

*2D text‑based CAPTCHAs* Andrei Broder with his team at the DEC Systems Research Center invented the 2D text-based CAPTCHA scheme in 1997. A similar method was used by the AltaVista website to prevent bots from influencing the rank of sites on the search engine [24]. Von Ahn and Blum created Gimpy CAPTCHA and EZ-Gimpy [14] in collaboration with Yahoo in 2000 to prevent bots from creating malicious advertisements and free accounts. Gimpy CAPTCHA requires you to correctly type at least three of seven random words in a dictionary. EZ-Gimpy is a condensed version of Gimpy only showing one word randomly in a dictionary. Gener- ated word images use a variety of fonts, gradients, noise, and other effects to make them difficult for bots to recog- nize. Monica Chew and Henry Baird suggested BaffleText [15] in 2003, a text-based CAPTCHA using pronounce- able pseudo-random words with masking algorithms to prevent recognition by OCR software. Megaupload.com created a segmentation-resistant CAPTCHA scheme in 2010. This method employs overlapping characters as well as the “Gestalt Perception” principle. According to the Gestalt perception principle, people can men- tally reconstruct individual characters, whereas com- puters still struggle with this task. The first version of ReCAPTCHA [16] was designed to protect websites from computer attacks. If a user types correctly the known words from old books’ two distorted words, they will pass the challenge. Chow et al. [25] proposed the concept of text-based clickable CAPTCHA. Their approach requests constructing a grid of clickable CAPTCHAs from multi- ple textual CAPTCHA challenges. The user must select the grid elements that correspond to the challenge requirement. Instead of using machine-printed text, the authors of [26, 27] proposed Handwritten CAPTCHAs to prevent recognition by OCR software.

*3D text‑based CAPTCHAs* These CAPTCHA schemes take advantage of sequences of 3D character recogni- tion by humans, but bots cannot, making them superior to 2D text-based CAPTCHAs. OCR Research Team [17] developed Teabag3D, a highly secure CAPTCHA. This CAPTCHA is composed of an image mixing textual characters with a 3D pattern. Super CAPTCHA [18] and 3DCAPTCHA [19] are text-based CAPTCHA schemes using the same assumptions as Teabag3D. Since 2013, Super CAPTCHA has been available as a WordPress. org plug-in. Imsamai and Phimoltares [28] developed the 3D CAPTCHA scheme, which involves showing 3D alphanumeric sequences and mixing many effects such as overlapping, rotation, noise, font variation, scaling, and other effects, to fool recognition of automated bots. Suzi et al. [20] recently suggested DotCHA, a 3D text-based CAPTCHA. 3D letters are made of small spheres in each challenge. Each letter is readable at a different twisted rotation angle around a horizontal axis. As a result, 3D text models need to be rotated several times to identify their letters.

*Animated text‑based CAPTCHAs* These CAPTCHAs add a time dimension to text-based schemes. In detail, the textual content is animated in a short clip for each challenge, making the extraction more difficult for auto- mated bots. In 2006, Fischer and Herfet [29] proposed one of the first animated CAPTCHA proposals. The concept of this CAPTCHA is to project text onto an ani- mated deforming surface. Naumann et al. [30] developed an animated CAPTCHA with the idea of the human ocular system perception in 2009. Only when the letters move, users can tell the difference between the text and the background. With the same concept, Cui et al. [31] introduced an animated CAPTCHA that only recog- nizes correct characters on moving. The “zero-knowledge per frame” principle is applied to ensure no informa- tion leaks in each frame. In 2010, Creo Group released the animated HelloCAPTCHA [21]. For each challenge, a sequence of six characters is presented in a GIF image with some effects: random positions, various orienta- tions, and others. The information is aligned to spread over multiple frames to prevent recognition over a sin- gle frame. The challenge in NuCaptcha [22] begins with a video of moving white font text, followed by three red characters in a dynamic background. To pass the chal- lenge, the red characters must be typed correctly by the user. In Dracon CAPTCHAs [23], five characters are displayed in fixed locations that have been randomly changed with effects of fade, blur, and noise.

* + 1. ***Image‑based CAPTCHAs***

Due to the recent failure of almost text-based CAPTCHAs, there is growing worry about their protection strength and accessibility. Lately, more designs are focusing on image-based instead of character recognition with the assumption of the general vision challenges being harder than text recognition. Table 2 contains a detailed categorization of the most commonly used image-based CAPTCHAs.

*Interactive‑based CAPTCHAs* These CAPTCHAs are based on the user’s interaction, such as swiping gestures or mouse movement, to reveal hidden points in an image. Conti et al. [32] suggested CAPTCHaStar in which the ability of humans to recognize shapes in a cluttered envi- ronment is used. The CAPTCHaStar challenge is made up of white pixels called stars that are randomly mixed together. The position of these stars changes depend- ing on where the cursor is. Users must drag the cursor so that the stars form an understandable shape before clicking the left mouse button to pass the CAPTCHA test. Okada et al. [33] created Noise CAPTCHA with the same concept. This CAPTCHA is made up of two different-sized and noisy images, as well as a hidden object or message in one of the images. Users must drag the small noisy image to identify the hidden object in the large image before clicking the “submit” button to pass the CAPTCHA challenge. Cursor CAPTCHA, proposed by Thomas et al. [34], displays five cursors randomly in a generated image. To pass the challenge, users must over- lap the mouse pointer onto a specific cursor.

*Selection‑based CAPTCHAs* These CAPTCHAs require users to choose candidate images from a set of images. Only text or text with a sample image can be used to describe this task. Asirra [35] is a typical CAPTCHA of this scheme, in which users are asked to select all cats from a set of 12 images of dogs and cats. In HumanAuth CAPTCHA [36], users are required to pick up all images that contain natural content among natural content images (such as a tree or a river) with artificial content images (such as a car or a watch). SEMAGE (SEmanti- cally MAtching imaGEs) CAPTCHA [37] differs from Asirra and HumanAuth CAPTCHA in that it requires users to select semantic images from an image set. As a result, the user must first recognize each image content and then identify the semantic relationship among them. Google released the “No captcha reCAPTCHA” [38] in 2014. Analyzing the browser environment (such as cook- ies and browser history), the system determines whether it is encountering a bot or not. The page will display only a checkbox or a selection-based CAPTCHA based on the risk level. The selection-based CAPTCHA chal- lenge renders nine candidate images and a sample image describing the image’s required content. In order to pass the challenge, the user must choose images that are simi- lar to the sample. Facebook’s image of CAPTCHA is similar to reCAPTCHA in its approach. To complete the challenge, users must choose images matching the hint description from a set of twelve images with varying con- tent. Avatar CAPTCHA [39] asks users to select avatar faces from a set of 12 grayscale images that include both human and avatar faces. FR-CAPTCHA [41] and FaceD- CAPTCHA [40] are two more face image CAPTCHAs. FR-CAPTCHA requires users to pick up the same per- son’s two face images in a complex background. On the other hand, in FaceDCAPTCHA, users are required to choose between visually warped human face images and non-human face images.

*Click‑based CAPTCHAs* These schemes display text and an image addressing where the user should click in order to pass the challenge. The main limitation of this type is that the challenge needs human intervention in order to generate a new instance. Implicit CAPTCHA

[42] is a common example which requires users to click on an identical location of an image. Tang et al. [51] pio- neered the use of SACaptcha in which the CAPCHA’s some regions linking an explained specific shape must be clicked by users to pass the challenge.

*Draw‑based CAPTCHAs* In 2006, Shirali-Shahreza, the first person, developed Drawing CAPTCHA [43], a drawing-based CAPTCHA. Diamond-shaped dots are connected by a user’s drawing lines. The most difficult aspect is that users must identify these dots against a noisy background. VAPTCHA (Variation Analysis-Based Public Turing Test to Tell Computers and Humans Apart) [44] consists of an image with a randomly generated tra- jectory in a challenge. To complete the challenge, users must draw a matching trajectory against this trajectory. In MotionCAPTCHA [45], similarly, users are also asked to draw a similar shape to the one rendered in the chal- lenge box.

*Slide‑based CAPTCHAs* In these CAPTCHAs, in order to solve a challenge, users must use a slider, such as drag- ging an image fragment to a correct location, rotating an image orientation or selecting a correct image form. WHAT’s Up CAPTCHA [46] displays three rotated images randomly, and users must rotate the images to their correct position. Minteye’s Slide-to-Fit CAPTCHA

[47] displays a swirled image, and users must move the provided slider until they see the undistorted image ver- sion. Tencent CAPTCHA requires users to move the slider to match two puzzle pieces.

*Drag and drop‑based CAPTCHAs* In these CAPTCHAs, users are required to align image pieces to form a com- plete image by dragging and dropping them. Garb CAPTCHA [48] displays four randomly shuffled pieces of an image. Users are required to reorder these image pieces to get the complete image to pass the CAPTCHA test. Hamid Ali et al. [52] pioneered the use of a puzzle- based CAPTCHA. Four image pieces of an image are required to be dragged and dropped into an empty four- cell grid to complete the challenge. Gao et al. [53] sug- gested a Jigsaw puzzle-based image-based CAPTCHA. In this CAPTCHA, an image is divided into many pieces (i.e., 9, 16, or 25) with only two wrongly positioned pieces. Users are required to swap the incorrect pieces to solve the challenge. Capy CAPTCHA [49] requires users to move a puzzle piece into a missing place in a challenge. This missing place is filled with a random image fraction. KeyCAPTCHA [50] displays three puzzle pieces and an incomplete image. Users are required to assemble these pieces to match the reference image. Once the cursor stays in the frame, the reference image will disappear. To pass the CAPTCHA challenge, users must move these pieces into the correct places.

* + 1. ***Audio‑based CAPTCHAs***

For people with visual impairments, a suggested alternative to visual CAPTCHA schemes was audio-based CAPTCHA schemes. They must type what they have heard to pass the test. At Carnegie Mellon University, the researchers introduced audio reCAPTCHA, acquired by Google later. To solve the challenge, users are required to identify eight digits spoken in human noise and only accept one incorrect digit in these digits. The eBay Audio CAPTCHA is made up of six digits in various spoken noisy voices. Microsoft CAPTCHAs are made up of ten digits in different spoken voices mixing the noise of some conversations. Yahoo CAPTCHA requires users to enter seven digits after three child-spoken beeps with background noise. The 2013 version of Audio reCAPTCHA requires users to recognize all of the digits divided into three clusters in the challenge. Three or four overlapping digits are found in each cluster. The new version of reCAPTCHA in 2017 included ten spoken digits and background noise. In Table 3, we list the most popular audio-based CAPTCHAs.

* + 1. ***Video‑based CAPTCHAs***

In the challenge, a short video is created, reflecting a certain content, users are required to understand and describe it by text. In Table 4, we list some typical video-based CAPTCHAs. Kluever et al. [54] suggested a CAPTCHA in which with a short video, users are required to watch and then type three words to describe it. Shirali-Shahreza et al. proposed Motion CAPTCHA [55] which requires users to describe the motion of the person in their watching video by choosing one of the sentences.

* + 1. ***Cognitive‑based CAPTCHAs***

CAPTCHA methods based on cognitive abilities that provide increased security have largely replaced tradi- tional Captcha methods. Cognitive abilities are brain- based skills that are the result of a distinct combination of neurobiological and psychological techniques. Knowl- edge, concentration, memory, judgment and assess- ment, reasoning and computation, problem-solving, and decision making are all aspects of human cognition and behavior. To distinguish between humans and bots, these CAPTCHA methods use biometric (something you are), physical (something you have), and knowledge- based (something you know) factors with or without the support of sensors like gyroscope or accelerometer [56, 57]. In Table 5, we list the most common cognitive- based CAPTCHAs. In 2020, Acien et al. [58] suggested BeCAPTCHA-Mouse that distinguishes humans from bots by analyzing mouse trajectories during the chal- lenge. Gametrics [59] differentiates between humans and bots by collecting and analyzing the user’s mouse move- ments during the operations of drag and drop to solve a dynamic cognitive game. GEETest and Netease [6], like Tencent CAPTCHA, require users to complete a sliding image-based CAPTCHA by moving the slider until two puzzle pieces are matched. If users complete the chal- lenge and their sliding behavior is not suspicious, they are considered to have passed the challenge. Siripitak- chai et al. [60] proposed EYE-CAPTCHA in which users are required to solve a math-based CAPTCHA by mov- ing their eyes. To complete the challenge, the user must identify the correct answer and use his eyes to move the answer to the center of the screen. In 2014, Google launched “No CAPTCHA reCAPTCHA” (reCAPTCHA V2). All that is required is to check the “I’m not a robot”box. However, user behaviors (such as click, mouse moving, and other behaviors) along with other infor- mation (browser, cookies, history etc.) are collected and analyzed in the background. If users are suspected of being bots, they need to complete a second image- based reCAPTCHA. In 2017, Invisible reCAPTCHA, an upgraded version of reCAPTCHA V2 was released. The evaluation process is initiated in the background by triggering a JavaScript API call or by users clicking on an existing button. Invisible reCAPTCHA, like the “No CAPTCHA reCAPTCHA” approach, requires a second image-based reCAPTCHA challenge if users are sus- pected of being bots. In 2015, Guerar et al. [61], the first person, introduced the physical CAPTCHA for mobile devices, called CAPPCHA (Completely Automated Pub- lic Physical test to tell Computers and Humans Apart). Users must tilt the device to a specific degree, which is difficult for bots to do. Hupperich et al. [62] introduced Sensor CAPTCHA in 2016, in which users are required to perform a complex gesture (such as fishing, hammer- ing, drinking) with their mobile devices. The authors of [63] proposed Pedometric CAPTCHA, in which humans are required to walk at least five steps. When the user walks, an acceleration is generated in the mobile device, making it difficult for bots. Mantri et al. [64] suggested a CAPTCHA scheme in which users must meet the requirement of moving the device in accordance with a specific guide showing on the device. Frank et al. [65] instructed users to perform a detectable gesture and rec- ognized by the gyroscope (such as rotating, tilting, or drawing), on moving the device. Guerar et al. [66] devel- oped Invisible CAPPCHA, which is similar to CAPP- CHA in that the challenge is invisible to users. Reading sensors detect user taps as opposed to touchscreen events, which bots can easily mimic. Furthermore, this CAPTCHA protects the user’s privacy by not sending sensitive data to the server. AccCAPTCHA [67] requires a user to play the rolling ball game. To complete the game, the user must control the ball using the device’s motion sensors. GISCHA, a mobile device game-based CAPTCHA, was proposed by Yang et al. [68]. To pass the challenge, a user must move the ball to the cor- rect hole. Ababtain et al. [69] suggested the CAPTCHA which requires users to pass a simple game using sen- sors. They proposed five games, each with several static and one moving object. Users must move the moving object to hit the correct target static objects in order to pass the challenge. SenCAPTCHA was proposed by Feng et al. [70] for locating an animal facial key point. Users are shown a small red ball and an animal image. Then, they must control the red ball into the animal’s eye center by tilting their devices. The authors [71] pro- posed BrightPass, a mobile authentication CAPTCHA to protect PIN/password. Their proposed mechanism uses screen brightness, which automated bots cannot detect, to determine when users should enter a correct digit or a deceptive digit. In the form of physical CAPTCHA, the authors [72, 73] proposed a PIN-based authentication CAPTCHA used for smartwatches. This mechanism is based on the same concept as CAPPCHA [74]. To enter the password, the bezel must be physically rotated to a specific degree. Similarly, the authors [75] use the digital crown rotation in smartwatches to protect the PIN code.

* + 1. ***Other types***

Stefan Popoveniuc [76] proposed the SpeakUP authentication method for remote unsupervised vot- ing in 2010. Voice biometrics is enhanced with text- based CAPTCHA. Voters must read out loud a voted candidate’s characteristics, rendered by 2D text-based CAPTCHA. Furthermore, voters’ voice biometric char- acteristics are identified through a challenge. The author also suggested recording the voter’s video of solving challenges. For protecting systems of facial authentica- tion, Uzun et al. [77] suggested rtCaptcha, a Real-Time CAPTCHA. Users must record their out loud pronuncia- tion of the presented 2D text CAPTCHA.

# CAPTCHA attack analysis

CAPTCHA has developed into the most popular utilized standard security measure for preventing automated computer program attacks. In recent years, many attack methods, developed by hackers or researchers, have effectively cracked all common conventional schemes. Some methods, including Invisible reCAPTCHA, have not yet been broken. However, with the introduction of fourth-generation bots accurately mimicking human behavior, a secure CAPTCHA would be hardly designed without additional special devices. Specially, almost all cognitive-based CAPTCHAs with sensor support have not yet been vulnerable to automated attacks. However, they are still compromised to human-assisted relay attacks due to having a limited number of challenges and can be only solved using trusted devices. Table 6 lists various recent CAPTCHA attack techniques, with DNN/ CNN and ML attack techniques dominating the list.

* 1. **Attack against text‑based CAPTCHA**

Text-based CAPTCHAs were the first CAPTCHA scheme and still remain the most popular. Mori and Malik [78] introduced an attack method of shape match- ing in 2003 to pass Gimpy and EZ-Gimpy CAPTCHAs with an accuracy of 33% and 92%, respectively. The pro- posed method [93] used a correlation algorithm and a direct distortion estimation algorithm to success- fully break EZ-Gimpy with a success rate of 99%. Chel- lapilla et al. [94, 95] created a highly secure CAPTCHA of anti-segmentation in 2005 after passing various text- based CAPTCHAs with machine learning. In 2008, sev- eral anti-segmentation CAPTCHAs, used by Google, Microsoft, and Yahoo, were demonstrated to be able to be cracked by El Ahmad and Yan [96, 97]. Later, other researchers attempted to pass these CAPTCHAs with higher success rates [98, 99]. El Ahmad and Yan [79] also broke Megaupload CAPTCHA with 78% of suc- cess. Google researchers [80] used neural networks to break the hardest category of ReCAPTCHA in 2014, with an accuracy of 99.8%. The authors [19] suggested 3D CAPTCHA attack methods without OCR software. In several 3D-based CAPTCHAs, such as 3DCAPTCHA, Teabag 3D, and Super CAPTCHA, they extracted pixels from the characters for automated challenge recognition. Using such a technique, the authors were able to break 3DCAPTCHA, Teabag 3D, and Super CAPTCHA with success rates of 58%, 31%, and 27%, respectively. Further- more, the same authors [100] were able to pass Teabag 3D by using the 3D textual objects’ side surface informa- tion. In the animated-based CAPTCHAs, Nguyen et al. [81] demonstrated how to easily extract information across multiple animated frames by using CL (Catch- ing Line) or PDM (Pixel Delay Map). These methods successfully defeated animated CAPTCHAs such as KillBot Professional, iCAPTCHA, Dracon CAPTCHA, and Atlantis. Due to their vulnerability to segmentation attacks, the same methods were used in [81] to defeat HelloCAPTCHA variants with a success rate ranging from 16 to 100%. NuCaptcha is a segmentation-resistant animated CAPTCHA that works by overlapping and cramming together to counter PDM or CL attack meth- ods. Elie Bursztein [82] separated objects in each frame with a success rate of 90% using an interest points (SIFT algorithm) density evaluation and bounding box shape analysis.

* 1. **Attack against image‑based CAPTCHA**

Golle [83] was successful in breaking the Asirra scheme. To accomplish this, SVM (support vector machine) was used to classify cats and dogs with a success rate of 82.7%. Hernandez-Castro et al. [84] suggested a side-channel attack breaking HumanAuth with an accuracy rate of 92%. Facebook image-based CAPTCHA and Google image-based CAPTCHA were bypassed by Sivakorn et al. [85] with success rates of 83.5% and 70.78%, respectively. The authors [6] achieved success rates of 79 and 88% with the new and old variations of reCAPTCHA V2. They also defeated China Railway CAPTCHA and Facebook image CAPTCHA with success rates of 90% and 86%, respectively. Besides, these authors broke different image-based CAPTCHA schemes, including the Tencent CAPTCHA with a success rate of 100%. Convolutional Neural Networks (CNN) [19] was applied to successfully break Avatar CAPTCHA, with a success rate of 99%. Both FaceDCAPTCHA and FR-CAPTCHA were defeated by Gao et al. [86] with success rates of 48% and 23%, respectively. Minteye CAPTCHA was defeated in [87] by utilizing the length of the image’s edges and Sobel operators. The attack method chooses the image with the smallest sum of edges based on the fact that a swirled image takes the longer edges. Hernandez-Castro et al. [88] suggested a low-cost attack using JPEG to measure image continuity. Using this side-channel attack, they successfully broke Capy CAPTCHA, Garb CAPTCHA, and KeyCAPTCHA with success rates of 65.1%, 98.1%, and 20%, respectively. Gougeon and Lacharme [89] were recently able to defeat CAPTCHAaStar with a success rate of 96%. They also demonstrated that the parameter tuning does not prevent this CAPTCHA from their attack on pixel concentration (stars) during image formation.

* 1. **Attack against Audio‑based CAPTCHA**

Tam et al. [101] experimented with an SVM-based approach to defeat audio reCAPTCHA with a success rate of 45% for the exact matching solution and a success rate of 58% for a “one mistake” passing condition. Decaptcha by Burzstein and Bethard [102] demonstrated a success rate of 75% in bypassing eBay’s audio CAPTCHAs. Their method analyzes the wave file using a Discrete Fourier Transform (DFT) and then clusters the energy spikes. Then, to recognize speech patterns, a supervised learning algorithm is employed to train audio data. The authors [103] introduced a CAPTCHA breaker with a non-continuous speech that broke Yahoo and Microsoft audio CAPTCHAs with success rates of 45% and 49%, respectively. The classification stage in this solver was supervised, whereas the automated segmentation stage was unsupervised. Amazon Mechanical Turk was used to label them, and the scraped CAPTCHAs were classified using the regularized least-squares classification (RLSC) algorithm. Due to the presence of semantic vocal noise, their system could only solve reCAPTCHA with a success rate of 1.5%. Sano et al. [90] suggested a CAPTCHA breaker for continuous speech to defeat anti-segmentation CAPTCHAs that overlap target voices. For speech recognition, Hidden Markov Models (HMMs) were employed and tested on the 2013 version of audio reCAPTCHA with a success rate of 52%. Bock et al. [91] presented unCaptcha that can bypass the 2017 version of audio reCAPTCHA with a success rate of 85.15% by utilizing free online services of speech-to- text and performing a minimal phonetic mapping for accuracy improvement.

* 1. **Attack against cognitive‑based CAPTCHA**

Using four simulation functions (Softmax, Sigmoid, Tanh, and ReLu) to mimic human behaviors, Zhao et al.

[6] successfully bypassed sliding-based CAPTCHA such as GeeTest and Netease CAPTCHA with success rates of 96 and 98%, respectively. By creating a tracking cookie for automated bots, Sivakorn et al. [85] were able to fool Google’s risk analysis system. As a result, after 9 days of automated bots browsing various Google services, the solver can check the box of “I’m not a robot.” Besides, the authors suggested a simple attack with a success rate of 70.78% for defeating the second reCAPTCHA V2 challenge. To break No CAPTCHA reCAPTCHA, the authors [92] applied the “divide and conquer” strategy. They were successful 97.4% of the time on a 100 × 100 grid and 96.7% of the time on a 1000 × 1000 screen resolution.

* 1. **Attack against Other CAPTCHAs**

Kluever et al. [54] developed a tag frequency-based approach to attack their proposed video-based CAPTCHA with a success rate of 13%. Hernandez-Castro et al. [104] were successful in breaking QRBGS CAPTCHA by the side-channel attack with a success rate of 44.54%. Mohamed et al. [105] demonstrated that dictionary-based attacks are able to defeat DCG CAPTCHAs. Moreover, developers [106, 107] proposed a solver that automatically bypasses SweetCAPTCHA, various slider CAPTCHAs (Taobao scheme) by developing a simple JavaScript code and puppeteer.

* 1. **Other attacks**
     1. ***Side‑channel attack***

Side-channel attacks are processes that attempt to solve an issue that is considerably easier than the original. The intended solution is built around a difficult challenge (AI-hard problem), whereas the actual solution is built around any design or implementation issues to avoid the more difficult approach. These attacks rely on random- ness deviations, missing uniform randomness, to identify a link between the challenges and their responses. In this case, the challenge provides (unintentionally, “leaked” or “side-channel”) knowledge on the answer. ASIRRA’s side- channel attacks are briefly described in this section [108]. ASIRRA is made up of over 25.000 photos, half of which are classed as cats or dogs. These photographs were pro- cessed by a classifier that, without utilizing any image recognition techniques, was able to discriminate between cat and dog pictures with about an accuracy of 60%. HumanAuth’s authors opted to mix a PNG image with a random JPG image picked from the library to prevent easy image library indexing. Choosing a new watermark that has a greater impact on the original image would come at the expense of human usability.

* + 1. ***Feature‑based attack***

In 2009, Philippe Golle [109] introduced the effective attacks on ASIRRA based on analyzing the CAPTCHA’s features, such as font, shape, texture, and color. By employing image processing, this approach divides the photographs into a cell grid of texture and color (gray- scale), which is then fed into support-vector machine (SVM) classifiers with a classification success of 83%.

* + 1. ***Database‑based attack***

If a CAPTCHA is based on a public knowledge database (i.e., labeled photos), there are numerous potential attacks against that database:

* + - * Database indexing attacks: tfle database can be downloaded (at least partially) to obtain tfle information needed to solve tfle CAPTCHA.
      * Database poisoning attacks: witfl an open and unprotected CAPTCHA database, our information can be uploaded to flelp us solve tfle CAPTCHA witfl tflis information.
    1. ***Human solving attack***

CAPTCHAs are intended to be completed by humans, but there exist markets for labor services solving CAPTCHAs [110] (usually in cheap labor regions) and relay attacks, which transmit CAPTCHA challenges to humans who benefit from solving them [111].

# CAPTCHA problem analysis

* 1. **Attack threats**

With the evolution of automated attacks, the differences in solving CAPTCHAs between humans and automated bots may become irrelevant: Should a human who is browsing another website or is presented with another program’s GUI be ineligible to solve our CAPTCHAs? Is a computer program that has been human-assisted still an automatic attack? Because it is difficult to distinguish between humans and bots, CAPTCHA schemes require additional mechanisms to improve their security:

* Measure a “fluman” quality, ability, or beflavior to distinguisfl between flumans and computers.
* Differentiate between flumans and fluman-assisted algoritflms to prevent magnifying or fluman-assisted attacks.
* Prevent relay attacks by differentiating between flumans wflo see tfle CAPTCHA on tfle original CAPTCHA site and tflose wflo see it on anotfler site/ interface [111].
* Prevent fluman farm attacks by employing metflods to tflwart or make more difficult tfle use of farms of solvers in solving tfle CAPTCHA.
  1. **AI hardness not transmitted**

The majority of CAPTCHAs have been vulnerable as a result of one of the following issues:

1. They are based on a mucfl more specific and weaker underlying problem tflan tfle original one intended.
2. Flaws from design or implementation make tflem mucfl easier to be bypassed by employing procedures analyzing tfleir cflallenges. As a result, tflese procedures are known as side-cflannel attacks because tfley attempt to solve a mucfl easier problem tflan tfle one intended by tfle CAPTCHA designers [104, 108].
3. The difficulty of an AI-unsolved problem is flard to convey to a CAPTCHA design. We do not know flow to categorize or deeply understand an AI flardness, so a CAPTCHA cflallenge of tflis AI flardness may be not difficult enougfl for automated bots.
   1. **Design flaws**
      1. ***Biased answer distribution***

One common mistake is to select a non-uniformly distrib- uted subset of possible answers. QRBGS (MathCAPTCHA) is one such example, with its designers employing one-digit figures in their arithmetic operations. As a result, the answers are likely to be small integers. Megaupload CAPTCHA is another example, which avoids using the values O, I, J, and 0. Worse, it always employs the three-letter-then-a-digit scheme, which makes it more user-friendly while also mak- ing it significantly less powerful. Teabag’s challenges [112] use only three-character lengths and avoid characters that are hard to distinguish in 3D projections. Characters “S,” “Z,” “3,” “P,” “b,” “w,” “M,” “t,” and “d” appeared more than 3% in a sample of 100 challenges, while a major set of other 34 char- acters, including “1” and “0,” did not appear (possibly to avoid coincidence with “I” and “O”).

* + 1. ***Biased challenge distribution***

Any biased idea in CAPTCHA design that is not based on randomness can allow challenge analysis, leading to side-channel attacks or challenge categorization analysis. Because the distribution of letter sizes in Teabag is not uniform, the frontal borders of the characters can be chosen based on their area size. There is also pixel correlation, which allows for back- border detection. Simple algorithms, such as pixel continuity, can detect growing background areas. In some challenges, the non-character image portion can be removed completely or nearly completely [112]. Another example is the Megaupload CAPTCHA, which always prints the letters and digits in the same font style, Antique Olive (as identified by Identifont). Characters are rotated at specific angles, clockwise or counter-clockwise, with the first letter clockwise and the second counter-clockwise. It also prevents the overlap of more than two characters [113].

* + 1. ***Correlation between challenge and answer***

The challenge may provide (unintentionally, “leaked” or “side-channel”) information based on the answer content. Side-channel attacks can be used to bypass the challenges by leveraging the leaked information.

* + 1. ***Evaluation of the answer***

It is not always necessary to make it easy for a CAPTCHA to determine whether or not their answers are correct. Avoid knowing whether an answer to a challenge is correct or incorrect, or any other way of knowing if it is close to being correct, if at all possible. We can communicate this information to the user via an intermediary communication mechanism (such as email accounts, which must also be controlled to limit emailing times) or we can transfer it to the user such that it is hard to be distinguished from automated bots.

* + 1. ***User dependence***

In general, making CAPTCHA dependent on the challenger is a bad idea, and it is even worse if this dependence can be known or guessed. ASIRRA, for example, displays pets in Petfinder that are near the challenger’s position in order to increase the chances of adoption for the pets displayed in the CAPTCHA (using IP geolocation). This flaw is critical because it facilitates many types of attacks, including database poisoning and database indexing.

* 1. **Implementation flaws**

Some CAPTCHA systems can be completely bypassed by leveraging the session ID of a previously used CAPTCHA [114]. That is due to poor implementation, but it was not unusual a few years ago. Some developers still encode the answer to the challenge in the URL or a form field. Using this mistake, many challenges can be requested with the same answer. As a result, a mean attack [115] can be launched by calculating the median values of those challenges. Another mistake in implementation is sending the client a hash of the answer, such as an MD5 hash, as a key. If the number of answers is limited or not distributed uniformly, the hashes of these answers can be easily learned enough to solve the challenges. Besides, using small fixed pools of challenges is one of the common implementation flaws. HumanAuth, for example, uses fewer than a hundred images, even masking them with logos, that are easily characterized or indexed [108]. Furthermore, HumanAuth only generates challenge answers with values 0 or a small integer. This allows another type of attack: if the answer 0 fails, we will answer with a series of integers beginning with the smallest absolute values. Another common mistake is that QRBGS challenges, as an example, are not created on demand, but rather are repeated [104]. Furthermore, some systems employ an extremely risky communication method with the CAPTCHA server, which is easily exploitable [116].

* 1. **Preserving users’ privacy**

In contrast to traditional CAPTCHA schemes, new sensor and behavioral-based CAPTCHA schemes have been shown to raise privacy concerns such as user behavioral data, cookies, and sensor data sent to remote servers. Some researchers proposed sending only the test results to the server, rather than the sensor data, as a solution. However, trusted hardware is required to prevent client-side hacking. As a result, the privacy of users should be strongly considered during the design phase of new CAPTCHA schemes.

* 1. **Device compatibility**

A robust and usable CAPTCHA is obviously expected to be compatible with a wide range of devices. The most promising CAPTCHA schemes, on the other hand, rely heavily on a single device. For example, CAPTCHA schemes based on touch- and-tap dynamics or mouse dynamics require device specialization. Sensor-based CAPTCHA schemes, which require sensors found only in smartwatches, tablets, or smartphones, are difficult to implement on the majority of users’ devices.

# How to design a good CAPTCHA

* 1. **Good properties**

Any new CAPTCHA design should be put into production in a test site, without other protections (to focus on the CAPTCHA’s hardness), for a long enough period of time to allow research. These new CAPTCHAs should include the following features to improve security against automated bots:

1. In all parameters, tflere sflould be randomness and a uniform distribution. For example, for a text CAPTCHA: uniform number of areas, lines, pixels witfl random properties (color, group, group size, etc.), variable number of cflaracters, various typefaces, image size, etc.
2. There sflould be no simpler CAPTCHA cflallenges: subtypes or alternatives sflould flave tfle same level of difficulty (sucfl as visual and audio CAPTCHAs).
3. The cflallenge sflould be as close to tfle original AI problem as possible.
4. The design sflould include features tflat detect automatic bypass or prevent relay attacks.
5. Cflallenges sflould be distributed uniformly and independent of users and answers. Furtflermore, tfle answers sflould be distributed randomly and uniformly. There sflould be no statistical relationsflip between tfle cflallenges and tfle answers.
6. Make it difficult for automated bots to determine wfletfler or not tfleir answers are correct by using adversarial samples, response mecflanisms, or communication metflods witfl CAPTCHA servers.
   1. **Security assurance**
7. Answer repetition: if an attacker is able to collect a finite quantity of cflallenges witfl tfle same answers, it must be confirmed tflat tflis attacker will not be able to create a better answer tflan a random answer. It means tflat tflere is no better attack tflan trial and error.
8. Cflallenge repetition: If our CAPTCHA flas only a finite set of different cflallenges and we do not know flow to solve tflem, tflere sflould be no bet- ter strategy tflan trial and error, witfl a low suc- cess rate
9. Non-categorization: If our CAPTCHA is made up of different types of cflallenges, tflere sflould be no way to tell tflem apart automatically or to classify tfle dif- ficulty of various cflallenges.
   1. **Security test**

For this test, we propose to create a large enough set of elements (T = test, A = answer) of tests. We look for non-uniformities in this distribution using general randomness and statistical analysis tools [108]:

* Inconsistencies in tfle distribution of A (potential blind attack).
* Inconsistencies in tfle distribution of T (type-of- cflallenge categorization and cflallenge analysis).
* Correlations among T and A (potential side-cflannel attack).

These tests can be performed for some simple properties of T, such as color histograms, area sizes, histograms, distances between similar areas, maximum and minimum for a block of bytes, and bit correlation with given vectors. This can be used to estimate the security parameters of any CAPTCHA proposal, avoiding pitfalls such as irrelevant parameter values that cause leakage of information [104, 108, 117].

# Conclusion

CAPTCHA is a competition between humans and computers. Computers attempt to mimic everything humans can do. On the contrary, Humans rely on AI’s hardness and cognition capability to challenge computers. Obviously, with the rapid and continuous development of technology, computers outfitted with the most robust and cutting-edge software and hardware are capable of solving AI’s most difficult problems at any time. In this paper, we have provided an overview analysis of HAI interactions between computer users and computers under current CAPTCHA schemes’ the security aspects of open concerns, difficulties, and opportunities in CAPTCHA design. We expect that this work will serve as a good starting point for new CAPTCHA designers in order to avoid some common design and implementation flaws, as well as for the development of new security assessment and assurance level evaluation methodologies.