Automatic Generation and Evaluation of Usable and Secure Audio reCAPTCHA

Mohit Jain[*+§], **Rohun Tripathi**[†+§], **Ishita Bhansali, Pratyush Kumar**[‡§]
[*]Microsoft Research India, mohja@microsoft.com
[†]Amazon Go, USA, rohun.tripathi.5@gmail.com
[‡]Indian Institute of Technology Madras, pratyush@iitm.ac.in

ABSTRACT
CAPTCHAs are challenge-response tests to differentiate humans from automated agents, with tasks that are easy for humans but difficult for computers. The most common CAPTCHAs require humans to decipher characters from an image and are unsuitable for visually impaired people. As an alternative, audio CAPTCHA was proposed, which require deciphering spoken digits/letters. However, current audio CAPTCHAs suffer from low usability and are insecure against Automatic Speech Recognition (ASR) attacks. In this work, we propose *reCAPGen*, a system that uses ASR for generating secure CAPTCHAs. We evaluated four audio CAPTCHA schemes with 60 sighted and 19 visually impaired participants. We found that our proposed *Last Two Words* scheme was the most usable with success rate of >78.2% and low response time of <14.5s. Furthermore, solving our audio CAPTCHAs can transcribe unknown words with >82% accuracy.

Author Keywords
CAPTCHA; evaluation; MTurk; blind; visually impaired.

CCS Concepts
•**Human-centered computing** → **Accessibility;**

INTRODUCTION
CAPTCHAs (Completely Automated Public Turing test to tell Computers and Humans Apart) [1] are challenge-response tests that determine whether a user is human or a computer by asking the user to perform a task that computers cannot yet perform. They are widely used as security measures to prevent automated abuse of online services. A key challenge with developing a CAPTCHA scheme is the inherent trade-off between *usability* and *security*, as the task must be difficult for a computer to solve so that it is secure, while also being usable, *i.e.*, it can be completed by a human with low effort.

[+]Both have contributed equally to this work.
[§]This work was done when they were working at IBM Research.

ASSETS '19, October 28–30, 2019, Pittsburgh, PA, USA.

ACM ISBN 978-1-4503-6676-2/19/10 ...$15.00.
http://dx.doi.org/10.1145/3308561.3353777

Visual CAPTCHAs are widely used, asking users to decipher distorted characters in an image. They rely on keen visual perception of humans to discern foreground text from the noisy background. This limits their usability in certain scenarios, such as for people with visual impairments and on devices without a computer screen. Hence, an alternative called audio CAPTCHA was introduced, wherein users identify the digits or letters spoken in a garbled audio. Nearly 1% of all CAPTCHAs are delivered as audio rather than images [8]. However, audio CAPTCHAs have been found to be very difficult and time-consuming with a success rate of below 52% [4, 8, 30]. Current audio CAPTCHAs are also not secure, as automated programs have been able to successfully break more than 70% of audio CAPTCHAs [7, 23, 36]. In order to improve Internet's accessibility to visually impaired users, there is a need to develop usable and secure audio CAPTCHAs.

In addition to accessibility, an important potential use case for audio CAPTCHAs is adding a layer of security on devices with speech as the primary mode of interaction. The last decade has seen rapid growth in speech-based interfaces on a variety of mobile and ubiquitous platforms, including smart speakers, VR/AR devices, and smartwatches. These devices have seen rapid adoption within the visually impaired community as well, mainly for accessing the web and navigating unfamiliar environments [13]. However, voice assistants are vulnerable to potential security threats [9, 20] including simple replay attack and attacks by a text-to-speech systems [11]. For instance, in the case of a suspicious transaction ("*Alexa, transfer a hundred dollars to Ryan's account*"), Alexa can ask the user to repeat the words in an audio CAPTCHA clip. Such a system would block automated agents, *e.g.*, a malicious agent on a phone app that impersonates a human by replaying a previously recorded human voice command (replay attack) or by generating a command with automated text-to-speech. Note: We claim that *reCAPGen* generated audio CAPTCHAs are secure against a few common attacks, however we do not claim that audio CAPTCHAs can ensure complete security of speech-based interfaces, as several other attacks are still feasible on them. As audio CAPTCHAs can enhance the security of speech-based interfaces for all users, we evaluated its usability with both sighted and visually-impaired users.

Solving a CAPTCHA requires human effort, which can be channeled to achieve a useful task [39]. Such CAPTCHA schemes are called reCAPTCHA. Visual reCAPTCHA pop-

ulates visual CAPTCHAs with text from old printed books that optical character recognition (OCR) have failed to recognize; this helps in the digitization of books [39]. Similarly, audio reCAPTCHA was proposed for the transcription of audio clips from old radio programs that ASR systems have failed to transcribe [18, 37]. Though secure, word-based audio reCAPTCHA was found to be unusable [18]. The audio reCAPTCHA project is no longer active, and details about the project are not well-documented. Hence, state-of-the-art audio CAPTCHAs comprise of random digits in a garbled audio, and do not include the reCAPTCHA component.

In this work, we propose *reCAPGen* system to generate audio reCAPTCHAS with the following goals:

1. **Usability**: Usability is measured using two metrics – success rate and time taken to solve the CAPTCHA. For visual CAPTCHAs, the average time taken is 9.8s and success rate is 87.3% [8].

2. **Security**: The generated CAPTCHAs need to be secure against the previously known common attacks.

3. **reCAPTCHA**: The human effort in solving a CAPTCHA should be channeled to some meaningful purpose. Solving audio CAPTCHA can generate audio transcriptions.

4. **Auto-generation**: For a secure and usable reCAPTCHA to work at scale, it must be equipped to auto-generate millions of unique audio CAPTCHAs without any human intervention. Moreover, this large database of CAPTCHAs makes it more robust against security attacks [4, 36].

reCAPGen selects audio clips from old radio programs, podcasts, and YouTube lectures, parts of which ASR systems fail to transcribe. *reCAPGen* then adds a calibrated amount of noise that would minimally affect the human ability to solve the generated audio CAPTCHAs, but further compromises an ASR system's ability to do so. This ensures that the generated audio CAPTCHAs are secure against ASR solving it. Most of the previous works use ASR attacks to break audio CAPTCHAs [29, 5, 25] (Table 2). Moreover, as our CAPTCHA involves arbitrary words (not taken from a fixed set), it is secure against multi-class classifiers, also known as supervised learning techniques. In contrast, current audio CAPTCHAs taken from a fixed set of letters/digits are solvable by supervised learning with a high success rate of >70% [7, 6, 36]. To increase the usability of audio CAPTCHAs, we enable users to speak their response, instead of having to type it. With this, our system can be used to enhance the security of devices with speech as the dominant mode of interaction.

To evaluate the usability of these CAPTCHAs, we conducted a user study with 60 sighted people on the Amazon Mechanical Turk platform and with 19 visually impaired people in-person, comparing four audio CAPTCHA schemes: (1) Random Digits (*RD*): the state-of-the-art audio CAPTCHA, (2) Two Words (*TW*): two consecutive words from an audio clip which need to be transcribed (similar to [33]), (3) Last Two Words (*LTW*): 5-7 words from an audio clip, where only the last two words need to be transcribed, and (4) Full Phrase (*FP*): 5-7 words from an audio clip, where the full phrase needs to be transcribed (similar to [18, 37]). We propose the novel LTW scheme. In LTW, we combine the TW and FP scheme in a way such that it reduces the mental demand on the user to memorize and repeat a long phrase, while providing context to the last two words that needs to be repeated by the user to solve the CAPTCHA. Data collected from the user study showed that LTW achieved the highest success rate (78.2% with sighted participants and 81.3% with visually impaired participants) with comparable response time (9.6s with sighted participants and 14.5s with visually impaired participants), and high usability ratings by the participants. Moreover, we found that solving our audio CAPTCHAs led to generation of $>$82% accurate audio transcription. These accurate transcriptions can help making media accessible to people with hearing impairments.

To summarize, the major contributions of our work are: (1) A fully automatic system, *reCAPGen*, to generate usable and secure audio reCAPTCHAs (available as an API). (2) A novel audio CAPTCHA scheme, *Last Two Words*, combining the benefits of prior proposed audio CAPTCHAs. (3) A user evaluation of generated audio reCAPTCHAs, proving their usability for both visually impaired and sighted users, and identifying suitability of variations to different use case scenarios.

RELATED WORK

In this section we discuss the prior work on audio CAPTCHA. We begin by discussing the following metrics: usability and security (Table 1). We then provide an overview of audio reCAPTCHA, followed by discussing sound-based content and different user input modalities. Throughout the section, we discuss the accessibility aspect of these audio CAPTCHAs.

Usability of Audio CAPTCHA

The most prevalent audio CAPTCHA scheme consists of several speakers saying digits at randomly spaced intervals with added background noise. The noise not only challenges automated agents, but also makes the CAPTCHA more difficult for humans to solve [37]. In a large-scale evaluation with 318,000 CAPTCHAs from 21 CAPTCHA schemes (13 visual and 8 audio schemes), Bursztein *et al.* [8] found that sighted users achieved an average success rate of 87% with visual CAPTCHA, but only 52% with audio CAPTCHA. Moreover, users took an average time of 9.8s to solve a visual CAPTCHA, compared to 28.4s for audio CAPTCHA. Similarly, Bigham and Cavender [4] found that both sighted and blind users had a low success rate of 39% and 43%, respectively, with audio CAPTCHA. These evaluations clearly demonstrate that state-of-the-art audio CAPTCHAs are unusable for humans.

Security of Audio CAPTCHA

The two most common approaches to break audio CAPTCHAs are supervised learning approach and ASR (Table 2). Tam *et al.* [36] analyzed the security of audio CAPTCHAs based on random digits and letters, and showed that an SVM-based approach was able to correctly solve 71% of them. With only 10 digits and 26 letters, supervised techniques (such as SVM, RLSC) use training data to classify each digit/letter to a particular class. DeCAPTCHA [7] broke 75% of eBay's audio CAPTCHAs using an undisclosed supervised technique. More recently, random digits audio CAPTCHAs have been

Audio CAPTCHA	Description	Success	Time	Secure	reCAPTCHA	AutoGen	User Input
Google, eBay, *etc.* [4, 7, 8, 36]	Random digits	52%	28.4s	No	No	Yes	Type
Meutzner *et al.* [21]	Random 4-6 words	~58%	>20s	Yes	No	Yes	Type
Old Radio [18, 37]	6-10 words phrase	46%	35.75s	Yes	Yes	Yes	Type
HIPUU [31, 32]	Image & sound	90%	65.64s	No	No	No	Type/Click
SoundsRight [19]	Sound	92%	45s	Maybe	No	No	Press a key
Telephone/VOIP [28, 35]	Random digits	13.7%	80.25s	No	No	Yes	Press keys
HearSay [33]	Two words	75%	~13s	No	No	No	Speak
reCAPGen	(Last) Two words	78.2%	9.6s	Yes	Yes	Yes	Speak

Table 1: Prior work related to Audio CAPTCHA (Note: Success Rate and Time Taken are measures of Usability)

Work	Type	Method	Accuracy
2008 [36]	Google RD, Digg RD&L, RD	SVM*	67, 71, 45%
2009 [7]	eBay RD	–*	75%
2011 [6]	Authorize RD&L, Digg RD&L, eBay RD, MS RD, Yahoo RD	RLSC*	89, 41, 82, 49, 45%
2013 [29]	Google RD	ASR	52%
2017 [5]	Google RD	ASR's	85.15%
2017 [25]	Google RD	ASR	>90%

Table 2: Breaking Audio CAPTCHA work. RD&L: Random Digits and Letters. *SVM, RLSC are different multiclass classifiers, using supervised learning approach.

broken using state-of-the-art ASR's [29, 5, 25]. Interestingly, Google CAPTCHAs are solvable using Google ASR with a high success rate of >90% [25]. This shows that current CAPTCHAs are not secure. Meutzner *et al.* [21, 22] proposed audio CAPTCHAs consisting of 4-6 English words, with noise sounds inserted at random positions in the audio. However, the English words were taken from 100 high frequency words; a CAPTCHA scheme should not select questions (or words, in this case) from a fixed set, as it may be easily solvable by computers using supervised learning techniques [4, 36].

Audio reCAPTCHA

Ahn *et al.* proposed visual reCAPTCHA [39] consisting of two words: a *control word* that is correctly recognized by OCR, and a *suspicious word* that is not successfully interpreted by OCR. To solve the visual reCAPTCHA, the user must solve the control word correctly since its answer is already known to the system. By answering the suspicious word, the user helps digitize books. To add the reCAPTCHA component to audio CAPTCHA, researchers proposed using 6-10 word-long audio clips from old radio programs, which ASR systems failed to transcribe [18, 37]. It consisted of a few words that validated a user to be human, while the other words helped in the audio transcription [39]. The rationale behind using long radio clips was that the human mind will use the contextual clues in the phrase to decipher the distorted audio, thus making this task easier for humans, as compared to recognizing random digits. However, when evaluated with 10 blind participants, Lazar *et al.* found poor usability with success rate of 46% and average time of 35.75s [18]. This may be because remembering and typing 6-10 words of a phrase is difficult and time-consuming.

Sound-based Audio CAPTCHA

Apart from numbers, letters, and words for audio CAPTCHA, sound-based approaches have also been proposed. HIPUU (Human Interaction Proof, Universally Usable) [14, 30, 31, 32] presents an image with a corresponding sound-based audio clip, such as a bird image with bird chirping audio, to make the CAPTCHAs accessible to blind users. The user can view the image and/or listen to the sound to identify the content and solve the CAPTCHA. SoundsRight audio CAPTCHA [19] asks the user to identify a specific sound (*e.g.*, sound of a bell) in an audio and press *space* button each time it occurs. Although both HIPUU and SoundsRight usability evaluation results were promising with above 90% success rate for blind users, there are security concerns as the images can be recognized by automated tools, and the number of human recognizable sounds that can be used in SoundsRight is very limited, which can jeopardize its security [4, 36].

Modalities for Answering Audio CAPTCHAs

Apart from typing, other modalities for answering audio CAPTCHAs have been proposed, including pressing buttons [19, 28, 35] and speaking [33]. To use digit-based audio CAPTCHAs for telephony systems [28, 35], instead of typing the response, the user presses the numeric buttons on the phone [35]. In an evaluation with 90 sighted participants, the average success rate was found to be 13.7% with average time of 80.25s [28] . Shahreza *et al.* [33] proposed HearSay, in which users say the solution to a two-word audio CAPTCHA aloud. The two words were randomly selected from 100 most frequent words, making it susceptible to supervised learning attacks. Sighted participants in their study achieved a success rate of 75% (compared to 61% with typing). For audio transcription (not related to security/CAPTCHA), Vashistha *et al.* proposed Respeak [38], a voice-based, crowd-powered system that uses ASR to transcribe audio files. The Respeak and HearSay evaluations [33, 38] showed that using speaking skills, rather than typing skills, helps to reduce error rate and response time.

In this work, we use speaking to solve CAPTCHAs generated from audio on which ASR have failed, thus achieving **auto-generated usable and secure audio reCAPTCHA**.

reCAPGen SYSTEM DESIGN

In this section, we step through our system to automatically generate audio CAPTCHAs from input audio files (Figure 1).

Audio Source and its Chunking

We used audio from old radio programs [2], podcasts [3], and YouTube lectures (in history, psychology, and computer science) as input. We avoided songs and movie dialogues as

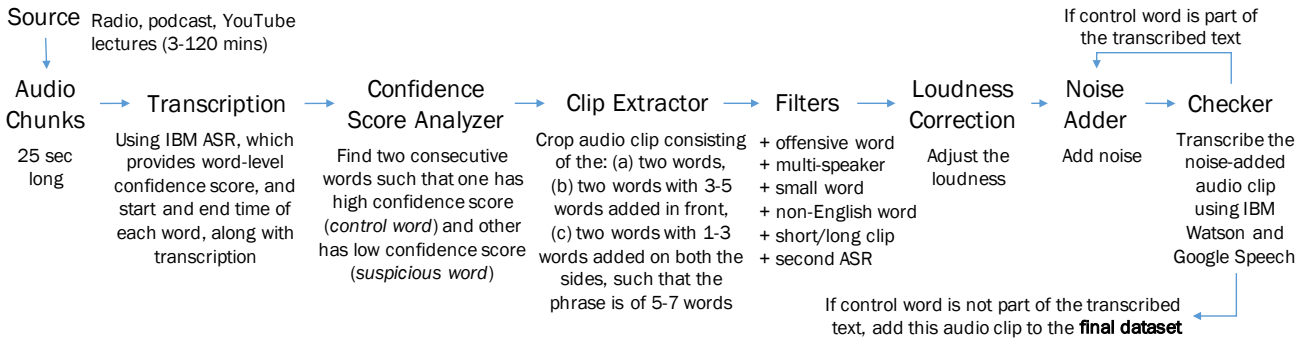

Figure 1: Audio reCAPTCHA automatic generation system, *reCAPGen*.

audio sources (unlike [38]), because song lyrics and movie subtitles are easily available on the Internet and hence are susceptible to automated attacks. The audio source length ranged from 3 minutes to 2 hours. As transcribing large-sized audio file usually takes a long time, *reCAPGen* splits the source audio file into 25s chunks and transcribes each chunk separately. To generate the audio transcription, *reCAPGen* used IBM Watson Speech-To-Text (STT) API service [15], which is an ASR system. We selected IBM ASR because: (1) it provides a word-level confidence score for the predicted transcription, and (2) it outputs the start and end time of each word in the audio clip. Most other ASR services, including Google Speech [10], only outputs the phrase-level confidence score along with the transcription, and do not provide the start and end time of each word.

Analyze Transcription to Get Candidate CAPTCHAs

reCAPGen analyzes the obtained transcription of each audio chunk to find two consecutive words, such that one word has a high transcription confidence score (of above 0.85), while the other has a low transcription confidence score (of below 0.5); these form the *control word* and the *suspicious word*, respectively. The control word is used for authenticating the human user, while the suspicious word is used for audio transcription. We chose the threshold values (of 0.85 and 0.5) after experimenting with several values. In our audio chunks, 93.1% of the identified words had a confidence score of above 0.75. Using manually generated ground truth transcription data, we found that 97.5% of the words with a confidence score of above 0.85 were correctly transcribed, and 89% of the words with a confidence score of below 0.5 were incorrectly transcribed.

reCAPGen then extracts an audio clip of these two words using the start and end time of the words, to form *candidate audio CAPTCHA* clips. Wherever possible, for the audio clip extraction, *reCAPGen* uses the end time of the word before the first word, and the start time of the word after the second word. This increases the probability that the start of the first word and the end of the second word is not cut abruptly.

Audio Clip Filters

Each candidate clip passes through several filters. Filters were added based on results and feedback obtained from two lab-based pilot studies with 4 participants each. In each study, we asked participants to listen to 30 different audio clips and verbally transcribe it. After that, we obtained subjective feedback related to the audio clips.

1. *reCAPGen* excludes audio clips wherein the control word is rude or offensive, as per IBM ASR's profanity filter.
2. Using IBM ASR's speaker detection, *reCAPGen* removes audio clips that have more than one speaker for the two words. The sudden change of the speaker between the two words was found to be disturbing for our pilot users.
3. *reCAPGen* checks the number of characters in the control word and discards audio clips with a small word as the control word. The first pilot study found that small words (3 or fewer characters) are usually difficult for the users to identify correctly (with 18.3% success rate).
4. The control word should be present in the English dictionary. If the control word is neither found in the Python Natural Language Toolkit (NLTK) [26] nor in the WordNet dictionary [24], *reCAPGen* discards the audio clip.
5. *reCAPGen* checks that the length of the audio clip containing only two words is between 0.5-1.5s. We found that longer audio clips usually have a long pause, laughter sound, or musical gaps, which can be distracting for users. Also, very short audio clips may have words spoken very quickly, which can be hard for the users to understand.
6. *reCAPGen* transcribes the audio clip using additional ASRs (*e.g.*, Google Speech [10]), and if the control word is not found in the transcription, the audio clip is removed. This step filters out audio clips for which the control word is incorrectly transcribed by IBM ASR, even with a confidence score of above 0.85.

Applying the filters removes a large number of candidate clips. Yet there are sufficient generated candidate clips for scalable automatic generation. For a 10-minute audio source, 24 audio chunks are created. Of these, on average 10.3 (sd=3.1) audio clips have consecutive words with high and low confidence scores. All the filters together discard 78.1% of the candidate clips, thus for a 10-minute source audio clip, on average 2.3 (sd=1.8) CAPTCHA clips are generated.

Audio Loudness Correction

In the first pilot study, participants found a few clips to be too loud such that it was "*jarring*" for them, or too soft in volume that it was hard to hear. *reCAPGen* increases or decreases the mean sound level of the clip to -15.93 dBFS. We use this

dBFS value because it is the mean for a large dataset of audio files from different sources [2, 3] (m=-15.93 dBFS, sd=4.5).

Background Noise Addition

To ensure that an ASR system would not be able to solve the generated candidate audio CAPTCHA by correctly identifying the control word, *reCAPGen* adds random background noise to the audio clip. For each clip, *reCAPGen* calculates the minimum noise that needs to be added such that both IBM and Google ASR fails to identify the control word correctly. More ASR's can be added to this stage. *reCAPGen* uses a variation of binary search algorithm to calculate the minimum noise required, thereby ensuring that the human usability is not affected more than is required for security considerations. As ASRs with improved accuracy are designed in future, this step ensures that *reCAPGen* can dynamically adapt to make the right trade-off between usability and security.

In the second pilot study, we compared the transcription of audio clips with the noise added only over the control word, versus addition of noise over the complete audio clip. We found that sudden changes in the background noise distracted participants and reduced their success rate. Thus, *reCAPGen* adds noise to the complete audio clip. Clips that are correctly solved by either IBM or Google ASR, even after adding the upper threshold of noise, were discarded. We chose the noise threshold to be 70% of the mean sound level of the audio clip. The average noise level added among all the audio clips was -10.8$\pm$2.9 dBFS.

AUDIO RECAPTCHA EVALUATION

To evaluate the usability of generated audio reCAPTCHAs, we conducted two studies: with sighted users on Amazon Mechanical Turk [16], and with visually impaired users in-person. We chose to study the two populations, because audio CAPTCHAs have been traditionally used as an alternative to visual CAPTCHAs for people with visual impairments, while recently with the growth of speech-based interfaces, audio CAPTCHAs will be relevant for sighted users as well.

Types of audio CAPTCHAs Evaluated

We evaluated these – one CAPTCHA (RD) and three re-CAPTCHA (TW, LTW and FP) – schemes:

(1) **Random Digits** (*RD*): For RD, we used the state-of-the-art audio CAPTCHA from Google. Each CAPTCHA audio file consisted of 10 random digits in a sequence, spoken by different speakers over different lengths of time with different background noises. For the validation, we manually generated the correct solution to the downloaded CAPTCHAs.

(2) **Two Words** (*TW*): In TW, the audio clip only consisted of the control word and the suspicious word. This is similar to HearSay [33], though in their system, the two words were randomly selected from the 100 most frequent words, and hence those CAPTCHAs were not secure.

(3) **Last Two Words** (*LTW*): LTW used an audio containing a 5-7 word long phrase, with only the last two words need to be transcribed. Compared to TW, we included extra words at the beginning to provide context to the user, in order to aid them in solving the CAPTCHA. To generate LTW CAPTCHAs, the *Clip Extractor* module (Figure 1) included 3-5 words before the two words to form the audio clip. The average number of words per LTW file was 5.9$\pm$1.3. This extra context in LTW requires more background noise (1.78 dBFS) to be added, compared to TW.

(4) **Full Phrase** (*FP*): FP used 5-7 word long phrase and the user must transcribe the whole phrase. The FP scheme is similar to the 6-10 word long audio CAPTCHAs generated from old radio programs [18, 37]. We selected 5-7 words because it has been found that humans can transcribe up to 7 words at a time [21, 38]. The number of words per FP file was 6.0$\pm$0.9. To generate FP CAPTCHAs, *reCAPGen* randomly added 1-3 words of audio before and after the audio clip generated for TW, ensuring a total of 5-7 words. Similar to LTW, the FP audio clip also provided context to the user. Additionally, the system ensured that the audio clip length was not more than 4s long (based on previous results [38]), for both LTW and FP. In the future, FP audio clips can be generated such that the control and suspicious word are maximally separated by 5 words between them. (Note: Consecutive control and suspicious word is a requirement only for TW and LTW.)

For a fair comparison between the last three reCAPTCHA schemes, we use the audio reCAPTCHA dataset generated for TW, to generate audio CAPTCHAs for LTW by prepending 3-5 words, and for FP by prepending and appending 1-3 words. For instance, TW: "*have taken*", LTW: "*and some biographers have taken*", and FP: "*some biographers have taken the position*". We chose these four CAPTCHA schemes, mainly because RD is the state-of-the-art system, and TW and FP have been previously proposed with high usability score and are similar to our proposed novel LTW scheme.

Participant Demographics

Sighted MTurkers

Sixty participants (32 male, 28 female), with a mean age of 36.8 years (sd=9.1, 18-62 years) participated in the study. We actually collected data from 65 people; however, five of them used offensive language in their responses and achieved a success rate of below 30%, hence their data was excluded from the analysis. The majority of the participants were from the United States (45); the remaining were from India (12), Canada (2), and Jamaica (1). Twenty-six participants had a high school education, thirty had a Bachelor's degree, and four had a Master's degree. All participants reported that they were able to understand and speak English fluently. Forty-eight participants stated that English was their first language, while the remaining twelve reported Hindi (8) and Tamil (4) being their first language. All participants had previous experience with visual CAPTCHAs, and 38 of them rated them to be frustrating. With respect to audio CAPTCHAs, 40 participants knew about them, but only three had experience using them. None of the participants reported any hearing or speech impairments.

Visually Impaired people

Nineteen participants (13 male, 6 female), with a mean age of 29.2 years (sd=4.4, 21-42 years) participated in the study. As

it is hard to recruit visually impaired participants on crowdsourcing platforms, our participants were recruited from three vocational training centers for blind people, located in India. Ten of the participants were completely blind (*i.e.*, 100% visually impaired); on an average, they self-reported being 90% visually impaired (sd=11.2%, min=70%). None of them can read any text on the computer screen visually, and relied on audio-based screen readers. All of them were proficient in using computers with the help of JAWS screen reader [17]. Eleven participants stated that Kannada was their first language, the remaining reported Telugu (3), Marathi (3) and Urdu (2) as their first language. All of them reported that they were able to understand and speak English fluently. Three participants were computer teachers at these centers, while remaining were students. Thirteen participants had a Bachelor's degree, three had a Master's degree, and three had a high school education. All participants had previous experience with audio CAPTCHAs, and all of them rated them to be frustrating. None of them reported any hearing or speech impairments.

Procedure

To evaluate the four audio CAPTCHA schemes, we performed a comparative evaluation. On MTurk (during Aug 2017), after reading a short description of the audio CAPTCHA task, anyone can opt to participate in the study by clicking on the user study link. On the other hand, to recruit visually impaired participants, two researchers visited three vocational training centers for blind people (in Nov 2017), and recruited people who were available to participate.

We instructed participants to listen to the short audio clips (0.5-4 seconds long) and speak its transcription aloud into the microphone. We gained explicit permissions from the participants to record and store their responses. We instructed participants to use a headphone with a microphone for the study in order to record high quality audio. We used the Google Webkit Speech Recognition API [34] to transcribe participants' responses. As this API is only available on the Chrome browser, we required participants to use the Chrome browser on a computer or laptop. To ensure that the participant was not suffering from any hearing or speech impairment, and that the participant could understand and speak English, only individuals who could correctly transcribe two out of the five trial CAPTCHAs were eligible to participate. After the screening page, the demographic questionnaire was administered. The language code for Google Webkit API was set based on the country selected in the demographic questionnaire. *E.g.*, for Indian participants, the code en-IN was used; whereas for Americans, en-US was used.

After demography, the audio CAPTCHA task began. The order of the four audio CAPTCHA schemes was random to counter for any order effects. In each of the four schemes, participants must first practice on two audio CAPTCHAs. Participants were allowed to skip any number of CAPTCHAs, but were required to attempt fifteen CAPTCHAs per scheme. Participants could attempt twice to solve each CAPTCHA question. Though existing real-world CAPTCHAs do not allow a second attempt, we allowed it in the user study to gain

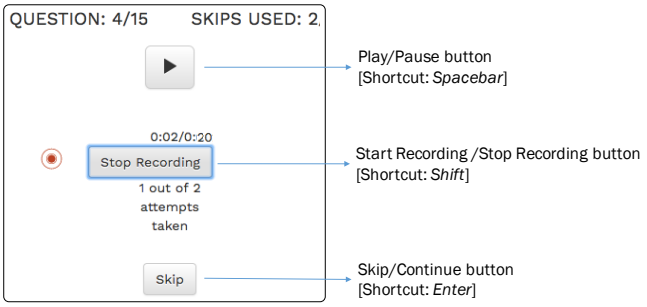

Figure 2: User Interface used for the audio CAPTCHA study

further insights (similar to [4]). The study design ensured that a particular audio CAPTCHA question consisting of a control-suspicious word pair was not repeated across the three schemes (TW, LTW and FP). At the end of each scheme, we asked participants to rate that scheme on a 5-point Likert scale with respect to *mental demand*, *frustration*, *effort required*, and *perceived success* from NASA-TLX scale [12], along with providing subjective feedback. At the end of the study, participants ranked the four audio CAPTCHA schemes, and provided subjective feedback. All participants' quotes in the following sections, are taken from this written feedback. Note: For the visually impaired participants, one of the study facilitators filled the demographic, rating and ranking questionnaire based on the participants' inputs.

Participants took ~45 minutes to complete the study and were paid $4 USD. In a real-world scenario, users have an incentive (*e.g.*, buy a ticket, register for a website, etc.) to solve CAPTCHAs. Similarly, we tried to mirror that in the crowdsourcing setting with an additional performance-based bonus component, to motivate the participants to answer the CAPTCHA questions as quickly and as accurately as possible. If the participant was among the top-third of the participants in terms of accuracy, he/she received a bonus of $2; participants in the middle-third received a bonus of $1, and the bottom-third earned no bonus. Similarly, the fastest third of the participants received $2 as bonus, the middle-third received $1, and the slowest-third earned no bonus. For calculating the bonus, MTurkers and visually impaired participants were evaluated separately. This study protocol has been approved by the ethics review board of an academic institute.

User Interface of the User Study

We used *reCAPGen* offline to generate 250 audio clips in each of the three reCAPTCHA conditions. We developed a front-end browser-based user interface (UI) connected to a web server to conduct the study. We kept the UI of the CAPTCHA question page to a minimal design with only three buttons: a Play/Pause button, a Start Recording/Stop Recording button, and a Skip/Continue button (Figure 2). Participants could listen to the audio file using the *Play/Pause* button. Participants may listen to the audio clip any number of times. To provide the answer, the participant needs to click the *Start Recording* button, speak the answer aloud after a beep sound, and then click the *Stop Recording* button. When there is a long pause of 1.5s or more, *Stop Recording* would get automatically triggered. We provide participants written and audio

feedback about whether the transcription recorded was the correct answer or not. A Levenshtein edit distance [27] of 1 or less between the control word and the participants' response, was taken as a correct answer (similar to [31]). If the participant's answer was correct, a *Continue* button appears; clicking this button would take the participant to the next question. In the case of a wrong answer, the participant could either attempt to transcribe it again or *Skip* the question. After two attempts (successful or not), participants can only *Continue* to the next question. The UI also supported keyboard shortcuts – '*Spacebar*' for Play/Pause, '*Shift*' for Start Recording/Stop Recording, and '*Enter*' for Skip/Continue (Figure 2) – based on prior work [4]. All user clicks and audio recordings are logged at the server.

RESULTS

We present the findings from the analysis of the logs, ratings and feedback provided. Note: $P_{1\text{-}60}$ refers to sighted MTurk participants, and $P_{61\text{-}79}$ refers to visually impaired participants. None of the participants failed the screening test.

In total, participants completed 4,740 audio CAPTCHAs (= 79 participants $\times$ 4 schemes $\times$ 15 audio CAPTCHAs/scheme). The average success rate with the three CAPTCHA schemes – TW, LTW and FP – was 75.7% for sighted users and 77.2% for visually impaired users, in the first attempt. Although participants were given up to two attempts to solve each CAPTCHA, we focus on evaluating their ability to solve a CAPTCHA on the first attempt, as most state-of-the-art CAPTCHAs only allow a single attempt. We conducted a mixed-model analysis of variance (ANOVA) – on the success rate of solving the audio CAPTCHA, the time taken to solve it, and the number of times the audio file was played – treating the four schemes as a fixed effect and participant as a random effect, for each of the two participants groups. We do not report comparative statistics between sighted and visually impaired participants, as that is not the aim of this paper.

Success Rate

Success rate is calculated as the ratio of the total number of CAPTCHAs solved correctly using control word validation by the total number of CAPTCHAs attempted (ignoring skipped CAPTCHAs). Considering only the CAPTCHAs solved in the first attempt, the ANOVA test showed a significant main effect of the type of audio CAPTCHA presented on the success rate ($F_{3,177}$=15.7, $p<0.001$ for the sighted participants and $F_{3,54}$=27.4, $p<0.001$ for the visually impaired participants) (Table 3). This prompted us to investigate pairwise differences. We employed Tukey's HSD procedure to address the increased risk of Type I error. We found that for sighted participants, the success rate of RD (89.0$\pm$7.7%) to be significantly higher than all the other three schemes ($p<0.001$). This result is in sharp contrast to previous works [8, 4], because our sighted participants noted the random digits using pen and paper before answering. On the other hand, for visually impaired participants, we found the success rate of RD (26.7$\pm$8.1%) to be significantly lower than all the other three schemes ($p<0.001$), as visually impaired users had to memorize the whole sequence of 10 random digits to answer.

Our results are not too surprising, as previous work HearSay [33] achieved 75% success rate with a two-word audio CAPTCHA scheme, with situationally impaired users. In our case, TW achieved 75.1% success rate with visually impaired users. With added context in LTW, visually impaired users performed the best with 81.3% success rate. Moreover, FP saw the highest improvement in success rate compared to prior work [37, 18], which can be attributed to two reasons: reduction of the phrase length to 5-7 words and using speech to solve the CAPTCHA instead of typing 6-10 words.

Taking the second attempt also into account, the results related to the main effect and pairwise comparisons remain the same, though the success rate across all three reCAPTCHA schemes increased to above 85% with both sighted participants and visually impaired participants.

As the three reCAPTCHA schemes used the same set of control-suspicious word pairs, we conducted further analyses to understand whether certain word pairs were more successful in one scheme than another. We found them to be not highly correlated – Pearson correlation values were 0.37 between TW and LTW, 0.17 between TW and FP, and 0.18 between LTW and FP. Moreover, we found that for each control-suspicious word pair, at least one scheme had a success rate of above 90% in the first attempt. This indicates that there was variability across the three reCAPTCHA schemes, and for a given audio file, one scheme may lead to a higher success rate than the other. Identifying which of the three schemes is best suited for a particular audio clip requires more data and is part of our future work.

Finally, using Google Webkit Speech Recognition API for transcribing user's speech input might have lead to additional errors. We performed a manual verification of 10% (474 randomly chosen recordings) of user's audio responses, and found Google Webkit's transcription error to be below 3%.

Time Taken

For each audio CAPTCHA question, *start-time* is the time when the *Play* button was pressed for the first time for that question, and *end-time* is the time when the *Stop Recording* button was pressed. *Time Taken* is calculated as the difference between the start-time and the end-time. For CAPTCHAs that were correctly solved by participants in their first attempt, the ANOVA test showed a significant main effect of the audio CAPTCHA scheme on the time taken ($F_{3,177}$=691.1, $p<0.0001$ for the sighted participants, and $F_{3,54}$=983.7, $p<0.0001$ for the visually impaired participants) (Table 3). Pairwise comparisons showed a significant difference between all four schemes, with $p<0.01$ (Table 3). This may be attributed to the fact that the audio file length in each of the four schemes varied – RD: 18.6$\pm$1.6s, TW: 1.6$\pm$0.4s, LTW: 3.4$\pm$0.5s, and FP: 3.6$\pm$0.3s. Considering both attempts, results related to the main effect and pairwise comparisons of time taken were the same.

Number of Plays

ANOVA showed a significant main effect of the CAPTCHA scheme on the number of times the *Play* button was pressed ($F_{3,177}$=26.3, $p<0.0001$ for the sighted participants, and $F_{3,54}$=35.2, $p<0.0001$ for the visually impaired participants).

Scheme	Success Rate (%)		Time Taken (s)		# Plays		# Skips	
Random Digits **RD**	89.0 7.7	26.7 8.1	33.3 35.4	73.6 41.1	1.1 0.3	3.8 3.3	0.2 0.6	0.1 0.1
Two words **TW**	74.2 9.2	75.1 7.7	7.4 5.9	12.6 4.2	1.5 1.4	2.0 2.2	0.9 1.1	0.2 0.1
Last Two Words **LTW**	78.2 10.1	81.3 8.3	9.6 6.2	14.5 3.9	1.2 0.7	1.3 0.6	0.7 1.0	0.1 0.1
Full Phrase **FP**	74.7 10.8	75.3 7.8	12.5 21.3	19.4 4.3	1.7 1.5	3.1 3.7	1.1 1.1	0.2 0.2

Table 3: Results in the format: mean std for MTurkers, mean std for visually impaired users.

Pairwise comparisons showed that for sighted participants, the RD and LTW schemes to be significantly better than the TW and FP schemes, and for visually impaired participants, the TW and LTW schemes to be significantly better than the RD and FP schemes, with $p<0.001$ (Table 3). In RD, sighted participants mentioned writing the numbers down using pen and paper, resulting in the fewest number of audio plays, while visually impaired participants tried to memorize the 10-digits sequence by listening and repeating it multiple times. For LTW, both sighted and visually impaired participants mentioned being "*attentive*" to listen for the last two words, requiring very few replays. The TW clips were the shortest, so participants listened to those clips again, "*just to be sure*" - P_{67} before submitting their response. In FP, participants stated that it was "*hard to remember long phrases*" - P_{79}, hence the audio was played multiple times.

Number of Skips

The number of skipped questions also showed main effect on the audio CAPTCHA scheme being used ($F_{3,177}=12.4$, $p<0.0001$), only with the sighted participants (Table 3). Visually impaired participants rarely skipped questions, which may be because three of them mentioned that skipping questions showed weakness: "*Why should I skip questions? I can do all of them.*" - P_{68}, or because skipping requires pressing more keys. In a similar manner, visually impaired participants rarely used the *Stop Recording* option. They preferred being silent after providing their response to auto-submit their answer. This reduces the total number of controls they need to use to two keys in the best-case scenario: *Spacebar* once to play the audio clip, and *Shift* key once to Start recording. Sighted participants skipped significantly fewer CAPTCHAs in RD (0.2 ± 0.6) compared to the other three schemes, with $p<0.01$. In TW, LTW, and FP, questions were skipped mostly because the first and/or the last word was "*cut off weirdly*" - P_{34}, *i.e.*, the audio "*started or ended with half words*" - P_{70}. For extracting audio clips, we used the word-specific start and end time obtained from the IBM ASR, hence this issue will become less severe in future as the ASR improves. In a real-world implementation, any audio CAPTCHA that is skipped or incorrectly answered by 3 or more users should be removed from the dataset (similar to [39]).

Transcription Accuracy - reCAPTCHA

One of the goals of our CAPTCHA scheme is reCAPTCHA, *i.e.*, leveraging the human effort of solving the CAPTCHA for a meaningful task. In our case, crowd-source the transcription of words which receive a low confidence score from ASR systems. Accurate audio transcriptions can highly benefit the hearing impaired community, by making the media accessible to them. In the case of visual reCAPTCHA [39], if 50% of the users have the matching response for a suspicious word, it is taken as the correct digitization of that text. In our study, as the three reCAPTCHA schemes used the same set of control-suspicious word pairs, we analyzed the transcriptions obtained across participants to find the similarity in the guessed suspicious word. We found that for 83.8% of the total 244 audio clips (occurring 3 or more times) across the three schemes with visually impaired participants, more than 50% of the responses for suspicious words were same. This can be interpreted as the accuracy of audio transcription for the suspicious word. The transcription accuracy for sighted users was 82.0% for 775 audio reCAPTCHAs. There was no significant difference between the three schemes.

Ratings and Preferences

Participants rated the four audio CAPTCHA schemes on a 5-point Likert scale (Figure 3), selected their two preferred CAPTCHA schemes, and wrote reasonings. Analyzing this data showed main effect for:

Mental Demand ($F_{3,177}=21.2$, $p<0.001$ for sighted, and $F_{3,54}=28.9$, $p<0.001$ for visually impaired),

Frustration ($F_{3,177}=7.8$, $p<0.001$ for sighted, and $F_{3,54}=15.7$, $p<0.001$ for visually impaired),

Effort Required ($F_{3,177}=10.7$, $p<0.001$ for sighted, and $F_{3,54}=15.2$, $p<0.001$ for visually impaired), and

Perceived Success ($F_{3,177}=7.5$, $p<0.01$ for sighted, and $F_{3,54}=18.0$, $p<0.001$ for visually impaired).

Note that a lower score is better for all metrics, except for *Perceived Success*. Ranking-wise, for sighted participants, TW was one of the two most preferred schemes for 51 participants, RD for 33 participants and LTW for 27 participants, while for visually impaired participants, LTW was one of the two most preferred schemes for 17 participants, TW for 14 participants and FP for 5 participants.

For sighted users, with respect to *Mental Demand*, FP was found to be significantly worse than all the other schemes ($p<0.01$). Sighted participants mentioned that FP was the most challenging (P_{22}), the most difficult (P_5) and the most frustrating (P_{43}). In contrast, visually impaired participants perceived RD to be the most mentally demanding ($p<0.01$), followed by FP ($p<0.01$). Only two visually impaired participant praised RD stating "*helps to concentrate... helps to increase my mental ability*" - P_{73}, while all others complained about the high mental demand of memorizing 10 random digits. Moreover, LTW was found to be significantly more mentally demanding than TW for sighted participants, with $p<0.01$, whereas visually impaired participants did not show any significant difference between them. Sighted participants complained that they were not sure about which words would be the last two words. Hence, they had to focus on the entire clip, resulting in high mental demand (P_{33}). On the other hand,

visually impaired participants appreciated LTW as it "*has less issues of chopped words* (compared to TW)"-P_{70}.

For *Frustration* and *Effort Required*, RD and TW were rated significantly better than FP by sighted users, and TW and LTW were rated significantly better than RD and FP by visually impaired users, with $p<0.001$ (Figure 3). Thirty sighted participants mentioned RD being easy, mainly because "*there was much less background noise*" - P_{18}, and "*could write the numbers down and speak them*" - P_{37} (similar to [30]). On the other hand, visually impaired participants found RD to be frustrating as "*there are lots of numbers*" - P_{72} and "*numbers are spoken very very slowly*" - P_{64}. Four of them asked for "*an option to increase the speed of the* (RD) *audio clip*" - P_{78}, which is one of the features in the JAWS screen reader. Forty-five sighted and 11 visually impaired participants appreciated TW as it was easy and fast: "*easy to remember only 2 words*" - P_{34}, "*least mentally demanding*" - P_{30}, and "*fastest... as quickly repeat two words*" - P_{58},P_{76}. Similar to TW, both sighted and visually impaired participants praised LTW as "*the two words had a context provided by the words preceding them*" - P_2. Only five sighted and two visually impaired participants praised FP as it was a full phrase with context: "*More words makes it easier to guess the sentence, even if 1-2 words were not clear they can be guessed with the help of others*" - P_9, "*I like them* (RD and FP) *because only these two used my mind, rest were too easy*" - P_{75}. Although clips in FP were restricted to 7 words, most participants complained that it had too many words, making it complicated (P_{10}) and hard to remember (P_{48},P_{61}).

Finally, sighted participants believed that they were significantly more successful with RD compared to FP, while visually impaired participants perceived to be more successful with TW and LTW compared to FP and RD, with $p<0.01$. However, sighted participants complained that RD was slow (P_{13},P_{20}), tedious (P_{33}) and "*annoyingly long*" - P_{43}. P_{36} mentioned: "*There were so many numbers I had to write them down... which seems like it might make things more difficult for the* (blind) *people who need to use audio captchas.*", which was exactly the case with the visually impaired participants.

Security Evaluations

The design of *reCAPGen* prioritizes security with an adversary-in-the-loop model. During the generation of CAPTCHAs, we use state-of-the-art ASRs from Google and IBM in the *Checker* module (Figure 1) to verify that the noise-augmented clip is not correctly transcribed. As ASRs evolve or new ones emerge, these can be added to the *Checker* module to enhance security.

Existing work on breaking CAPTCHAs (Table 2) either use ASRs [25, 5, 29] or use supervised learning attacks customized to a given CAPTCHA generating engine [6, 7, 36]. First, we used state-of-the-art ASRs from Microsoft and Amazon. Note that these ASRs are not used in the *Checker* module and thus expose the system to generalization vulnerabilities across ASRs. We found that not a single clip was correctly transcribed. We then added a pre-processing step of denoising the clip using time-frequency block thresholding as described in [40]. With this, we achieved a very low success rate of 0.7%. Note that previous audio CAPTCHAs have been broken using standard ASRs [25, 5]. This demonstrates that the CAPTCHAs generated by *reCAPGen* are secure against ASR attacks. Second, we attempted an attack with an SVM-based supervised learning model as demonstrated in [6, 7, 36]. It failed to break our TW CAPTCHAs as it had a success rate of below 0.1%. This is explained by the large variability in the control words: The words are randomly sampled from a large English vocabulary as used in a wide variety of videos from old radio programs, podcasts and YouTube lectures on variety of topics. In contrast, previous works generated CAPTCHAs from small sets, such as the set of 10 digits [8], 100 most frequent words [33, 21], or 20 sounds [19]. These made them vulnerable to machine learning attacks which could be successfully trained on the small corpus.

Another possible security concern is speaking aloud the response to the CAPTCHA. This does not pose any privacy or security risk as the content of the CAPTCHA contains no sensitive information. Also, a new CAPTCHA is generated on every try. Finally, more sophisticated attacks may be designed. For instance, if one or many devices listened to a large number of audio CAPTCHAs and their responses, could they train a deep neural network model to break *reCAPGen* CAPTCHAs? The feasibility and success of such an attack remains to be verified, and we plan to do that as part of future work.

DISCUSSION

To summarize, TW and LTW were the best performing schemes, achieving high success rate, low time taken, and high ratings from both the sighted and visually impaired participants. Visual CAPTCHAs success rate is 87% and takes 9.8s to solve [8]; TW and LTW achieved a comparable success rate and time taken to visual CAPTCHA. Thus they are usable and accessible enough to be considered for real-world CAPTCHA, especially for visually impaired users, and either of them can be chosen based on the application needs.

Finally, our proposed system, *reCAPGen*, has several parameters that can be tuned to further increase the usability and/or security of the generated CAPTCHAs.

Design Implications

CAPTCHAs Adaptable to Future Attacks: All CAPTCHAs, including visual and audio CAPTCHAs, must evolve in response to increasing strength of automated adversaries, specifically with advances in machine learning. Visual reCAPTCHA system [39] uses the images of words on which OCR failed. Similarly, in our proposed *reCAPGen* system, we added the *Checker* module (Figure 1) to filter out candidate audio clips that are solvable by state-of-the-art ASR systems after adding the calibrated amount of noise. More ASR systems can be added to that module, to further strengthen the 'adversary-in-the-loop' generation method of CAPTCHAs.

Say No to RD: RD is the most widely used state-of-the-art scheme and achieves high success rate with sighted users. However security is a major concern [7, 36]. The long time required by both sighted and visually impaired users to answer RD audio CAPTCHA, and the low success rate by visually impaired users, makes this approach unusable. Moreover RD

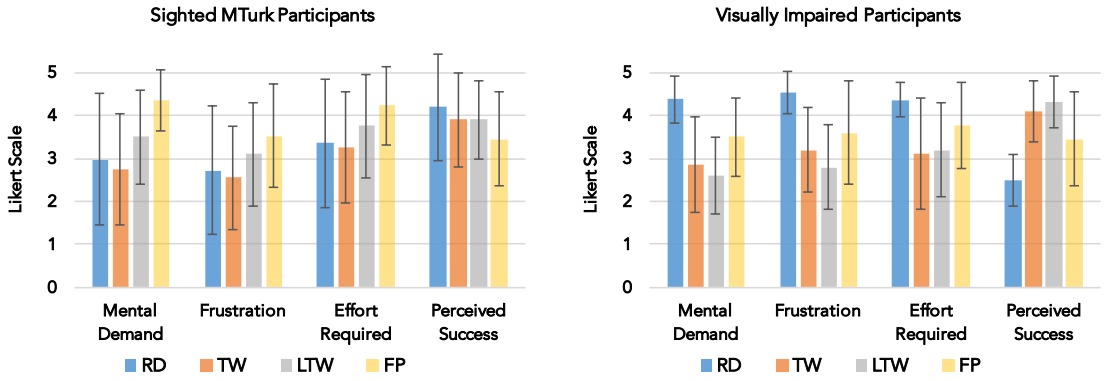

Figure 3: Likert-scale rating by the participants (with standard deviation shown as error bars)

does not provide the reCAPTCHA benefits. Therefore, we believe that RD should be avoided in any secure application.

Addition of Noise: A calibrated amount of noise needs to be added such that other ASR systems fail to break the generated audio CAPTCHAs by identifying the control word correctly. Due to the extra context in LTW and FP, they are prone to better recognition by ASRs, which results in more background noise being added to them compared to TW. This in certain CAPTCHAs can make LTW harder to solve than TW for humans. From a security perspective, adding noise only to the control word portion of the audio file suffices. However, we found that adding the noise to the whole audio clip increases its usability. Moreover, it must be ensured that the added noise is not greatly affecting the usability for humans to solve the CAPTCHA. If the success rate of solving CAPTCHAs by humans is low, the noise threshold can be reduced. This will result in generation of fewer, but more usable, audio CAPTCHAs from an input audio file.

Accessibility: In this work, we enabled users to speak their response, instead of having to type it. Visually impaired users have been found to prefer speech input over keyboard-based text entry [4]. This may be a reason for high acceptance. However, in real-world use, there can be situations (like public space) where text entry may be preferred over speech. For such situations, an optional text input mode must be present along with the speech mode. Moreover, from a usability perspective, LTW should be preferred to add CAPTCHAs to speech-based interfaces, as the number of times the audio file was played was close to 1. Replaying the audio would require a longer interaction with the device (for security purposes), which may not be suited for visually impaired users. Fewer replays with LTW was a result of the context provided by the words preceding the control and suspicious word.

Applicability of Different Schemes: We found that certain audio clips are more usable for humans in one CAPTCHA scheme, compared to others. Example-1: a few of the TW and LTW audio CAPTCHAs have the ending word being chopped off, as IBM ASR was not able to accurately find the end time of the last word. However for those audios, FP can work better as the control and suspicious word won't get chopped off. Example-2: for a few LTW CAPTCHAs, participants were unsure of the 'last two words'. For such audios, TW might outperform LTW as the participants just have to repeat the whole audio clip, instead of identifying the last few words. In the future, the system should be able to understand such behaviors, and identify the correct scheme for a particular audio CAPTCHA.

Limitations

First, the sample size was relatively small at 79 participants, compared to a few previous studies with more than 150 participants [4, 8]. However, due to our within-subject design, in total, a large number of audio CAPTCHAs (4740) were solved by our participants. Second, sighted and visually impaired participants were from different demographies. Amongst our 60 MTurkers, 45 were Americans, while all visually impaired participants were Indians. This may have influenced our results, as the source audio files were taken randomly. As native speakers are found to be more accurate than non-native speakers for transcription task [21], in the future, demographic specific audio CAPTCHAs can be presented to further increase its usability. Third, our security evaluation was limited to past known attacks. *reCAPGen* generated audio reCAPTCHAs resilience against new future attacks is unknown. Finally, we should have ideally compared our proposed system with the previous radio-clip based audio reCAPTCHA system [18, 37]. However, that audio reCAPTCHA project is no longer active, and the technical details about the project are not well-documented. Thus, we are unable to replicate their approach.

CONCLUSION

In this work, we propose *reCAPGen*, a system that automatically generates usable and secure audio reCAPTCHAs. It uses audio files that ASR systems fail to transcribe with high confidence. To evaluate the usability of generated CAPTCHAs, we conducted studies with sighted and visually impaired users, as audio CAPTCHAs were proposed to make the Internet accessible to visually impaired people and with the recent growth of speech-based interfaces, audio CAPTCHAs have potential usage for the general population as well. Our proposed LTW scheme exhibits a good trade-off between usability and security, achieving a success rate of >78% with sighted users and >81% with visually impaired users, and an average response time of <15s. Also, these CAPTCHAs exploit the human effort to generate transcriptions for audio files with a high accuracy. To conclude, we hope that others will use our proposed system to generate audio CAPTCHAs for several applications.

REFERENCES

[1] Luis Von Ahn, Manuel Blum, Nicholas J. Hopper, and John Langford. 2003. CAPTCHA: Using Hard AI Problems for Security. In *Proceedings of the 22Nd International Conference on Theory and Applications of Cryptographic Techniques (EUROCRYPT'03)*. Springer-Verlag, Berlin, Heidelberg, 294–311. http://dl.acm.org/citation.cfm?id=1766171.1766196

[2] Internet Archive. 2017a. Old Time Radio. (2017). Retrieved Jan 4, 2017 from https://archive.org/details/oldtimeradio

[3] Internet Archive. 2017b. Podcasts. (2017). Retrieved Jan 4, 2017 from https://archive.org/details/audio_podcast

[4] Jeffrey P. Bigham and Anna C. Cavender. 2009. Evaluating Existing Audio CAPTCHAs and an Interface Optimized for Non-visual Use. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 1829–1838. DOI: http://dx.doi.org/10.1145/1518701.1518983

[5] Kevin Bock, Daven Patel, George Hughey, and Dave Levin. 2017. unCaptcha: A Low-Resource Defeat of reCaptcha's Audio Challenge. In *11th USENIX Workshop on Offensive Technologies (WOOT 17)*. USENIX Association, Vancouver, BC. https://www.usenix.org/conference/woot17/workshop-program/presentation/bock

[6] E. Bursztein, R. Beauxis, H. Paskov, D. Perito, C. Fabry, and J. Mitchell. 2011. The Failure of Noise-Based Non-continuous Audio Captchas. In *2011 IEEE Symposium on Security and Privacy*. 19–31. DOI: http://dx.doi.org/10.1109/SP.2011.14

[7] Elie Bursztein and Steven Bethard. 2009. Decaptcha: Breaking 75% of eBay Audio CAPTCHAs. In *Proceedings of the 3rd USENIX Conference on Offensive Technologies (WOOT'09)*. USENIX Association, Berkeley, CA, USA, 8–8. http://dl.acm.org/citation.cfm?id=1855876.1855884

[8] Elie Bursztein, Steven Bethard, Celine Fabry, John C. Mitchell, and Dan Jurafsky. 2010. How Good Are Humans at Solving CAPTCHAs? A Large Scale Evaluation. In *Proceedings of the 2010 IEEE Symposium on Security and Privacy (SP '10)*. IEEE Computer Society, Washington, DC, USA, 399–413. DOI:http://dx.doi.org/10.1109/SP.2010.31

[9] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. 2016. Hidden Voice Commands. In *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association, Austin, TX, 513–530. https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/carlini

[10] Google Cloud. 2017. Google Cloud Speech API. (2017). Retrieved Jan 1, 2017 from https://cloud.google.com/speech/

[11] William Haack, Madeleine Severance, Michael Wallace, and Jeremy Wohlwend. 2017. Security Analysis of the Amazon Echo. (2017), 1–14.

[12] Sandra G Hart and Lowell E Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. *Advances in psychology* 52 (1988), 139–183.

[13] D K-h Ho. 2018. Voice-controlled virtual assistants for the older people with visual impairment. *Eye*, Article 32 (2018), 2 pages. DOI: http://dx.doi.org/10.1038/eye.2017.165

[14] Jonathan Holman, Jonathan Lazar, Jinjuan Heidi Feng, and John D'Arcy. 2007. Developing Usable CAPTCHAs for Blind Users. In *Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility (Assets '07)*. ACM, New York, NY, USA, 245–246. DOI: http://dx.doi.org/10.1145/1296843.1296894

[15] IBM. 2017. IBM Watson Speech to Text. (2017). Retrieved Jan 1, 2017 from https://www.ibm.com/watson/services/speech-to-text/

[16] Amazon Inc. 2017a. Amazon Mechanical Turk. (2017). Retrieved June 1, 2017 from https://www.mturk.com/mturk/welcome

[17] Freedom Scientific Inc. 2017b. Blindness Solutions: JAWS. (2017). Retrieved Nov 5, 2017 from http://www.freedomscientific.com/Products/Blindness/JAWS

[18] Jonathan Lazar, Jinjuan Feng, Olusegun Adelegan, Anna Giller, Andrew Hardsock, Ron Horney, Ryan Jacob, Edward Kosiba, Gregory Martin, Monica Misterka, Ashley O'Connor, Andrew Präck, Roland Roberts, Gabe Piunti, Robert Schober, Matt Weatherholtz, and Eric Weaver. 2010. POSTER: Assessing the Usability of the new Radio Clip-Based Human Interaction Proofs.

[19] Jonathan Lazar, Jinjuan Feng, Tim Brooks, Genna Melamed, Brian Wentz, Jon Holman, Abiodun Olalere, and Nnanna Ekedebe. 2012. The SoundsRight CAPTCHA: An Improved Approach to Audio Human Interaction Proofs for Blind Users. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 2267–2276. DOI: http://dx.doi.org/10.1145/2207676.2208385

[20] Xiangyu Liu, Zhe Zhou, Wenrui Diao, Zhou Li, and Kehuan Zhang. 2015. When Good Becomes Evil: Keystroke Inference with Smartwatch. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15)*. ACM, New York, NY, USA, 1273–1285. DOI: http://dx.doi.org/10.1145/2810103.2813668

[21] Hendrik Meutzner, Santosh Gupta, and Dorothea Kolossa. 2015. Constructing Secure Audio CAPTCHAs by Exploiting Differences Between Humans and Machines. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 2335–2338. DOI:http://dx.doi.org/10.1145/2702123.2702127

[22] Hendrik Meutzner, Santosh Gupta, Viet-Hung Nguyen, Thorsten Holz, and Dorothea Kolossa. 2016. Toward Improved Audio CAPTCHAs Based on Auditory Perception and Language Understanding. *ACM Trans. Priv. Secur.* 19, 4, Article 10 (Nov. 2016), 31 pages. DOI: http://dx.doi.org/10.1145/2856820

[23] Hendrik Meutzner, Viet-Hung Nguyen, Thorsten Holz, and Dorothea Kolossa. 2014. Using Automatic Speech Recognition for Attacking Acoustic CAPTCHAs: The Trade-off Between Usability and Security. In *Proceedings of the 30th Annual Computer Security Applications Conference (ACSAC '14)*. ACM, New York, NY, USA, 276–285. DOI: http://dx.doi.org/10.1145/2664243.2664262

[24] George A. Miller. 1995. WordNet: A Lexical Database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[25] Yair Mizrahi. 2017. ReBreakCaptcha: Breaking Google's ReCaptcha v2 using Google. (2017). Retrieved Mar 18, 2018 from https://east-ee.com/2017/02/28/rebreakcaptcha-breaking-googles-recaptcha-v2-using-google/

[26] NLTK Project. 2017. Natural Language Toolkit. (2017). Retrieved Jan 15, 2017 from http://www.nltk.org/

[27] E. S. Ristad and P. N. Yianilos. 1998. Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20, 5 (May 1998), 522–532. DOI:http://dx.doi.org/10.1109/34.682181

[28] Niharika Sachdeva, Nitesh Saxena, and Ponnurangam Kumaraguru. 2013. On the Viability of CAPTCHAs for Use in Telephony Systems: A Usability Field Study. In *Proceedings of the 11th Asia Pacific Conference on Computer Human Interaction (APCHI '13)*. ACM, New York, NY, USA, 178–182. DOI: http://dx.doi.org/10.1145/2525194.2525265

[29] Shotaro Sano, Takuma Otsuka, and Hiroshi G. Okuno. 2013. Solving Google's Continuous Audio CAPTCHA with HMM-Based Automatic Speech Recognition. In *Advances in Information and Computer Security*. Springer Berlin Heidelberg, Berlin, Heidelberg, 36–52.

[30] Graig Sauer, Harry Hochheiser, Jinjuan Feng, and Jonathan Lazar. 2008. Towards a Universally Usable CAPTCHA. In *Proceedings of the 4th Symposium on Usability, Privacy and Security (SOUPS'08)*. ACM, New York, NY, USA, 2.

[31] Graig Sauer, Jonathan Holman, Jonathan Lazar, Harry Hochheiser, and Jinjuan Feng. 2010a. Accessible Privacy and Security: A Universally Usable Human-interaction Proof Tool. *Univers. Access Inf. Soc.* 9, 3 (Aug. 2010), 239–248. DOI: http://dx.doi.org/10.1007/s10209-009-0171-2

[32] Graig Sauer, Jonathan Lazar, Harry Hochheiser, and Jinjuan Feng. 2010b. Towards A Universally Usable Human Interaction Proof: Evaluation of Task Completion Strategies. *ACM Trans. Access. Comput.* 2, 4, Article 15 (June 2010), 32 pages. DOI: http://dx.doi.org/10.1145/1786774.1786776

[33] Sajad Shirali-Shahreza, Gerald Penn, Ravin Balakrishnan, and Yashar Ganjali. 2013. SeeSay and HearSay CAPTCHA for Mobile Interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 2147–2156. DOI: http://dx.doi.org/10.1145/2470654.2481295

[34] Glen Shires. 2017. Voice Driven Web Apps: Introduction to the Web Speech API. (2017). Retrieved Jan 15, 2017 from https://developers.google.com/web/updates/2013/01/Voice-Driven-Web-Apps-Introduction-to-the-Web-Speech-API

[35] Yannis Soupionis and Dimitris Gritzalis. 2010. Audio CAPTCHA: Existing Solutions Assessment and a New Implementation for VoIP Telephony. *Comput. Secur.* 29, 5 (July 2010), 603–618. DOI: http://dx.doi.org/10.1016/j.cose.2009.12.003

[36] Jennifer Tam, Sean Hyde, Jiri Simsa, and Luis Von Ahn. 2008a. Breaking Audio CAPTCHAs. In *Proceedings of the 21st International Conference on Neural Information Processing Systems (NIPS'08)*. Curran Associates Inc., USA, 1625–1632. http://dl.acm.org/citation.cfm?id=2981780.2981983

[37] Jennifer Tam, Jiri Simsa, David Huggins-Daines, Luis Von Ahn, and Manuel Blum. 2008b. Improving Audio CAPTCHAs. In *Proceedings of the 4th Symposium on Usability, Privacy and Security (SOUPS'08)*. ACM, New York, NY, USA, 2.

[38] Aditya Vashistha, Pooja Sethi, and Richard Anderson. 2017. Respeak: A Voice-based, Crowd-powered Speech Transcription System. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 1855–1866. DOI:http://dx.doi.org/10.1145/3025453.3025640

[39] Luis von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. 2008. reCAPTCHA: Human-Based Character Recognition via Web Security Measures. *Science* 321, 5895 (2008), 1465–1468. DOI: http://dx.doi.org/10.1126/science.1160379

[40] Guoshen Yu, Stephane Mallat, and Emmanuel Bacry. 2008. Audio Denoising by Time-Frequency Block Thresholding. *IEEE Transactions on Signal Processing* 56, 5 (May 2008), 1830–1839. DOI: http://dx.doi.org/10.1109/TSP.2007.912893