Using Automatic Speech Recognition for Attacking Acoustic CAPTCHAs: The Trade-off between Usability and Security

Hendrik Meutzner, Viet-Hung Nguyen, Thorsten Holz, Dorothea Kolossa
Horst Görtz Institute for IT-Security (HGI), Ruhr-University Bochum
{hendrik.meutzner, viet.nguyen-c7k, thorsten.holz, dorothea.kolossa}@rub.de

ABSTRACT

A common method to prevent automated abuses of Internet services is utilizing challenge-response tests that distinguish human users from machines. These tests are known as CAPTCHAs (*Completely Automated Public Turing Tests to Tell Computers and Humans Apart*) and should represent a task that is easy to solve for humans, but difficult for fraudulent programs. To enable access for visually impaired people, an acoustic CAPTCHA is typically provided in addition to the better-known visual CAPTCHAs. Recent security studies show that most acoustic CAPTCHAs, albeit difficult to solve for humans, can be broken via machine learning.

In this work, we suggest using speech recognition rather than generic classification methods for better analyzing the security of acoustic CAPTCHAs. We show that our attack based on an automatic speech recognition system can successfully defeat reCAPTCHA with a significantly higher success rate than reported in previous studies.

A major difficulty in designing CAPTCHAs arises from the trade-off between human usability and robustness against automated attacks. We present and analyze an alternative CAPTCHA design that exploits specific capabilities of the human auditory system, i.e., auditory streaming and tolerance to reverberation. Since state-of-the-art speech recognition technology still does not provide these capabilities, the resulting CAPTCHA is hard to solve automatically. A detailed analysis of the proposed CAPTCHA shows a far better trade-off between usability and security than the current quasi-standard approach of reCAPTCHA.

1. INTRODUCTION

In order to limit or even prevent automated abuse of online services (e.g., automated account creation or crawling), it is necessary to distinguish human users from programs. A common approach is to rely on a so-called *Completely Automated Public Turing Test to Tell Computers and Humans Apart* (CAPTCHA) that should be easy to solve by humans, but difficult to break by machines [5, 29, 30]. Such challenge-response tests are predominantly based on some kind of obscured graphic such as a highly distorted text that has to be recognized by the user. Some tests also require the user to solve simple mathematical equations [7] or to identify specific objects within an image [20]. The security properties of such visual CAPTCHAs have been discussed several times within the last decade [14, 31, 32] and various security improvements have been proposed [6, 10, 24].

To enable access for visually impaired users and to be applicable for non-graphical devices, most Internet services also provide an acoustic verification scheme in addition to visual CAPTCHAs. Up to now, acoustic CAPTCHAs have received less scientific attention compared to the visual ones and existing studies have demonstrated that most acoustic CAPTCHAs can be easily broken [3, 4, 22, 26]. Thus, a major problem arises in practice from the fact that many popular Internet services make use of acoustic CAPTCHAs that are possibly vulnerable, rendering any supplementary image-based CAPTCHA irrelevant.

We consider an acoustic CAPTCHA to be broken if an algorithm is able to find the correct sequence of words in the audio signal, which we will refer to as *transcription* later on. Therefore, the crucial measure for the security strength of a CAPTCHA is given by the success rate for transcribing the signals in an automated manner. Here, the definition of a plausible value for the maximum allowed success rate is highly controversial, as one has to consider a trade-off between human usability and robustness against automated attacks. Some authors require the success rate of an automatic solver to be less than 0.01 % in order to render the CAPTCHA secure [5], whereas other studies refer to a maximum allowed success rate of 1 % [3] or 5 % [26]. However, the definition highly depends on the underlying threat model that arises from several properties and resources of the attacker. Here, the degree of theoretical knowledge about signal processing and machine learning as well as the number of available IP addresses will influence the success of an automated attack significantly. For our purposes, we assume that all potential vulnerabilities are induced by the CAPTCHA itself (i.e., insecure implementations are not within the scope of this work).

Preliminaries and Prior Work.

Acoustic CAPTCHAs are commonly based on a sequence of words where the underlying lexicon is often limited only to digits between zero and nine, which can be beneficial for non-native speakers or just to keep the user instructions sim-

ACSAC '14 New Orleans, Louisiana, USA

http://dx.doi.org/10.1145/2664243.2664262

ple (e.g., "please type in every digit you hear"). However, the security aspect of such small-vocabulary-based CAPTCHAs becomes highly questionable as state-of-the-art machine learning methods are fairly advanced, especially when the number of individual words to be recognized is low.

Previous studies have performed a security analysis of available acoustic CAPTCHAs by using automated solvers. Bursztein et al. [3] and Tam et al. [26] utilize a two-stage approach by first segmenting the CAPTCHA into individual word segments and then finding the respective word transcriptions by using a generic classification algorithm. With such an approach, Bursztein et al. evaluated the strength of six acoustic CAPTCHAs and found that they can break all but one scheme: the success rate was between 41 % and 89 % for five schemes, only reCAPTCHA [12] (as of 2010) withstood their attack since the success rate was only 1.5 %. Sano et al. [22] demonstrate that a newer version of reCAPTCHA (as of April 2013) can be broken with a success rate of 52 % by using *automatic speech recognition* (ASR).

Up to now, only a limited number of studies explored secure design strategies for acoustic CAPTCHAs. Recent studies analyze the effectiveness of artificial signal distortion [1, 16] or suggest to use speech signals that exhibit a poor overall quality like old radio broadcast recordings [25]. Schlaikjer [23] proposed a CAPTCHA where the task is to recognize open-domain speech[1].

As discussed above, attacking CAPTCHAs in an automated fashion generally represents a machine learning problem that can be tackled by various methods. Hence, it is not only practical but also indispensable to exploit a broad range of existing weaknesses in current machine learning techniques when designing secure acoustic CAPTCHAs. Obviously, the introduction of artificial distortions into a speech signal can lower the performance of an automated solver. The difficulty lies in distorting the signals such that the speech intelligibility is preserved.

Our Approach.

We consider automated attacks on acoustic CAPTCHAs as a speech recognition task and demonstrate how state-of-the-art ASR techniques can be utilized to accomplish it. A major benefit of using ASR arises from the fact that the temporal characteristics of speech are inherently considered and incorporated by the employed algorithms (cf. Sec. 2) as compared to a generic classification algorithm.

Bursztein et al. reported [4] that the ASR-based approach performed worse than the classification-based approach for attacking the eBay CAPTCHA. We assume that this finding is based on the authors using pre-trained acoustic models—freely available on the Internet—without adapting the models to the CAPTCHA signals.

In contrast to previous work [3, 26, 22], our approach is not based on a pre-segmentation of the CAPTCHA into individual parts. This is advantageous as we do not require any task-specific adaption of the segmentation stage, rendering the CAPTCHA solver more general.

The results indicate that our method clearly outperforms previous approaches based on segmentation and word recognition: we achieve a success rate of 19.19 % for the older version of reCAPTCHA analyzed by Bursztein et al. [3] (13× improvement). We can also break the current reCAPTCHA with a success rate of 62.8 % which is 11 % higher than the success rate reported for reCAPTCHA by Sano et al. [22]. In addition, we show that the human success rate for the current reCAPTCHA scheme is 38.4 % below that of our automated CAPTCHA solver, rendering the trade-off between usability and security fairly disadvantageous.

Furthermore, we systematically explore how CAPTCHAs can be improved upon to attain the original goal of being much more intelligible to humans than to machines. More specifically, we have generated CAPTCHAs by exploiting the auditory perception of humans in the design process. Our proposed CAPTCHA achieves a far better trade-off between usability and security as compared to the current reCAPTCHA scheme.

Our Contributions.

In summary, our contributions in this work are as follows:

- We demonstrate that speech recognition methods can yield considerable performance improvements for solving acoustic CAPTCHAs in comparison to a classification-based approach as it is used by Bursztein et al. [3]. To allow for a fair comparison, our analysis is based on a previous reCAPTCHA scheme from 2010.

- We analyze the current reCAPTCHA scheme and present the respective human recognition scores that were obtained from a listening test. In this context we discuss the usability of the CAPTCHA by comparing the performance of humans vs. machines.

- We propose alternative design strategies for creating more secure acoustic CAPTCHAs. These design strategies are motivated by the auditory perception of humans, where we exploit human abilities that current machine learning algorithms are lacking to a certain degree. We also conduct a security analysis of the proposed design strategies by using an automated solver and present the results of a listening experiment to examine human usability.

2. TECHNICAL BACKGROUND

State-of-the-art speech recognizers are usually based on *hidden Markov models* (HMMs) or *artificial neural networks* (ANNs) [11, 15, 18], which is advantageous compared to a generic classification-based approach. This is due to the fact that speech represents a slowly time-varying process, so it is reasonable to incorporate algorithms that can handle time series by nature.

Our attacks are conducted by using HMM-based speech recognition. For convenience, we make use of the hidden Markov model toolkit (HTK) [33], which represents a standard toolkit in the research community. For our approach, all model topologies, i.e., the number of states and mixtures and the allowed state transitions have been manually adjusted to be well suitable for the task and the ensuing model training has been done from scratch.

2.1 HMM-based Speech Recognition

In order to perform speech recognition, the time-domain representation (i.e., the waveform), of the input signal is first segmented into a series of overlapping frames. In practice,

[1]Open-domain speech covers complex scenarios (e.g., lectures and talks) and it is not limited to a specific task.

typical frame lengths are in the range of 20–25 ms as speech can be regarded to be short-time stationary within this interval. The overlap between consecutive frames is preferably 50–75 % to capture the temporal evolution of the waveform continuously. Then, a feature vector is computed for each frame of the input signal. Two prominent types of features for ASR are *Mel frequency cepstral coefficients* (MFCC) [19, 21] or *perceptual linear prediction* (PLP) coefficients [13]. These features are motivated by several properties of the auditory system and enable a compact representation of the speech frames, where PLP features are more suitable for suppressing speaker-dependent details. In order to incorporate signal dynamics, it is valuable to also include the 1st and 2nd-order derivatives of the feature coefficients into the final feature set. After feature extraction, the next step is to learn suitable segment models for representing the speech using the features and the label information of the training material. The segment models should represent the basic units of speech, e.g., phonemes[2], in large-vocabulary applications, but for small-vocabulary tasks, it is feasible and ultimately leads to higher accuracy, when entire word models are trained. Each segment model is represented by an HMM that consists of a customizable number of states and transitions, and a common topology for speech recognition is a left-to-right model [21].

Each state q of the model possesses an individual output probability:

$$\mathrm{b}(\mathbf{o}_m) = \mathrm{P}(\mathbf{o}_m | q_m = i) \quad i = 1, 2, \ldots, Q \;, \tag{1}$$

where Q denotes the number of states and $\mathbf{o}_m$ is the feature vector—the so-called *observation*—at frame m. For modeling speech, the output probabilities are usually represented by a *Gaussian mixture model* (GMM) comprised of K components:

$$\mathrm{b}_q(\mathbf{o}) = \sum_{\kappa=1}^{K} \gamma_{\kappa,q} \mathcal{N}(\mathbf{o} | \boldsymbol{\mu}_{\kappa,q}, \underline{\boldsymbol{\Sigma}}_{\kappa,q}). \tag{2}$$

Here, $\mathcal{N}(\mathbf{o} | \boldsymbol{\mu}_{\kappa,q}, \underline{\boldsymbol{\Sigma}}_{\kappa,q})$ is the multivariate normal distribution parameterized by its mean vector $\boldsymbol{\mu}_{\kappa,q}$ and its covariance matrix $\underline{\boldsymbol{\Sigma}}_{\kappa,q}$. Each Gaussian component is scaled with the mixture weight $\gamma_{\kappa,q}$, where

$$\sum_{\kappa=1}^{K} \gamma_{\kappa,q} = 1 \quad \forall q. \tag{3}$$

The complete HMM is defined by the parameter set:

$$\lambda = (\underline{\mathbf{A}}, \mathcal{B}, \boldsymbol{\Pi}), \tag{4}$$

where $\underline{\mathbf{A}}$ denotes a matrix of state transition probabilities and $\mathcal{B}$ is a set containing the parameters of the output probability distributions of all states. The vector $\boldsymbol{\Pi}$ contains the initial probabilities that can be given, for example, by the word probabilities or by a fixed value. The model parameters λ can then be estimated by optimizing a *maximum likelihood* (ML) criterion:

$$\lambda_{\mathrm{ML}} = \arg\max_{\lambda} \mathrm{P}(\mathbf{O} | \lambda), \tag{5}$$

where $\mathbf{O} = \mathbf{o}_1 \ldots \mathbf{o}_T$ represents a sequence of T observations. A solution for Eq. 5 can be found by using the Baum-Welch algorithm [21]. Before we can solve Eq. 5, the mean vectors $\boldsymbol{\mu}_{\kappa,q}$ and covariance matrices $\underline{\boldsymbol{\Sigma}}_{\kappa,q}$ of the respective state output distributions need to be initialized properly. Since ML estimation can only yield a local optimum for a finite number of data points, the initialization of the model parameters can influence the ASR performance considerably later on. A simple approach for initialization is to use the global mean and variance of the entire training data, which is often referred to as *flat-start* initialization. When the temporal occurrence of words in the training material is known, better results are achieved by initializing each word model individually with the corresponding statistics, which is regarded as a *bootstrap* initialization.

After model training (i.e., after the final parameters of Eq. 5 have been obtained), the individual segment models are combined into a compound HMM that incorporates a user-defined grammar of the speech recognition task. For recognizing a speech utterance, the sequence of observed features $\mathbf{o}_1 \ldots \mathbf{o}_T$ can be decoded by searching for the best state sequence $q_1^* \ldots q_T^*$ through the compound HMM:

$$[q_1 \ldots q_T]^* = \arg\max_{q_1' \ldots q_T'} \mathrm{P}(q_1' \ldots q_T' | \mathbf{o}_1 \ldots \mathbf{o}_T, \lambda). \tag{6}$$

Equation 6 can be solved by means of a Viterbi decoder, which implements a dynamic programming approach for finding the most likely sequence of states.

2.2 Performance Assessment

In order to assess the ASR performance, it is necessary to compare the speech recognition output with a corresponding reference transcription. Here, one problem arises from the fact that the recognition output can differ from the reference transcription in its length if the underlying grammar is not restricted to a fixed number of words. However, the comparison of two text strings of different lengths can be achieved by using a dynamic programming procedure. This procedure aligns the recognizer output with the reference transcription and then counts the word deletion (W_D), insertion (W_I) and substitution (W_S) errors. The standard metric for assessing ASR performance is given by the word and sentence accuracy, which are computed by:

$$\text{Word Acc.} = 100 \cdot \frac{W - W_D - W_I - W_S}{W}, \tag{7}$$

$$\text{Sent. Acc.} = 100 \cdot \frac{S_C}{S}, \tag{8}$$

where W, S denote the total number of words and sentences in the reference transcription, respectively, and S_C is the number of correctly recognized sentences. In the following, we regard the transcription of a CAPTCHA as a sentence.

3. ATTACKING CAPTCHAS USING ASR

In this section, we conduct a security analysis on a previous version of reCAPTCHA [12] and compare our results with those of Bursztein et al. [3]. We show that state-of-the-art speech recognition is more suitable for evaluating the security strength of a CAPTCHA in that it leads to higher success rates than a generic classification-based approach.

Data Description.

We use data of a previous reCAPTCHA scheme comprising 198 labeled signals where the labels include the respective temporal information about the exact digit positions.

[2]A phoneme is the smallest linguistic unit in a certain language.

The audio files were downloaded from the reCAPTCHA website [12] in 2010 and hand-labeled at our institute. Each CAPTCHA is constructed from a sequence of 8 digits between zero and nine. The CAPTCHAs consist of real speech and the digits in each CAPTCHA are spoken by different male and female speakers. All CAPTCHAs are distorted with additive background noise that represents a time reversed speech signal as it is reported by Tam et al. [26].

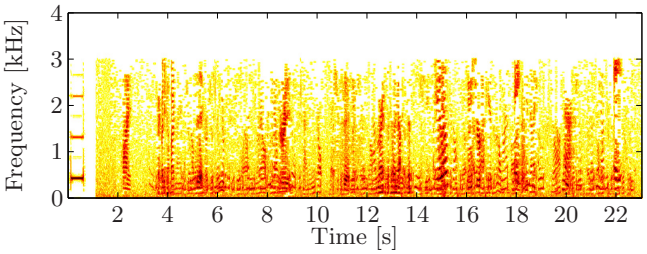

Figure 1: Example of reCAPTCHA 2010 showing the digit sequence "5-6-3-6-0-8-9-8".

The signals are sampled at 8 kHz and the spectral content is band-limited to 3 kHz. Each signal starts with a short tonal sound (beep) which is then followed by eight isolated digits that are clearly separated in time. The signal parts between the digits contain only the interfering background noise and will thus be referred to as *noise segments* later on. Figure 1 provides an example of the CAPTCHA by showing the spectrogram, which is derived from the squared magnitude of the short-time Fourier transform. Since all CAPTCHAs exhibit the same principal structure, it is straightforward to derive a suitable grammar that can be incorporated into the decoder to improve the recognition performance. Due to the limited number of signals, we make use of a K-fold cross-validation. For this purpose, the corpus is partitioned into $K = 6$ complementary subsets, where we train an individual recognizer for each subset by using 60 % of the data and the remaining data is used for assessing the recognition performance. The overall recognition performance, i.e., the word and the sentence accuracies, are obtained by averaging the decoding results of all cross-validation sets.

ASR Training.

Each word is modeled with an HMM that has 3 states per phoneme and 8 Gaussian components per state. Two additional models are utilized for representing the starting tone and the noise-only segments, respectively. We provide our analysis results for two different feature types that are either derived from 13 MFCCs or 13 PLPs. In each case, the features are augmented with their respective first and second order temporal derivatives, yielding a 39-dimensional feature vector. To train the recognition system, it is first necessary to initialize the model parameters λ as defined by Eq. 4. We have found that a flat-start initialization of the model parameters leads to poor recognition rates. As we have the temporal label information at hand, we derive the initial model parameters by using a bootstrap approach. Here, we vary the amount of the bootstrap data and compare the results in Sec. 3.1. After model initialization, several iterations of Baum-Welch training are performed to estimate the model parameters.

Table 1: Average cross-validation results in percent for reCAPTCHA 2010 shown for a varying amount of bootstrap data (Init [%]). The best results are highlighted.

Feat.	Init [%]	Sent Acc.	Word. Acc
MFCC	25	11.11	74.24
	50	10.10	73.42
	75	**14.14**	**74.87**
	100	12.12	73.86
PLP	25	16.67	78.03
	50	**19.19**	**78.98**
	75	13.64	78.34
	100	18.18	77.84

Signal Decoding.

Due to the consistent structure of the CAPTCHA, we utilize a task-specific grammar for decoding:

$$\mathrm{S} \rightarrow \mathrm{T\,N\,D\,N\,D\,N\,D\,N\,D\,N\,D\,N\,D\,N\,D\,N\,D\,N\,D\,N}\,, \qquad (9)$$

with S, T, N denoting the start symbol, the beep tone and the noise segment, respectively. The symbol D represents the individual digits, i.e.,:

$$\mathrm{D} \rightarrow \mathrm{zero|one|two|}\cdots\mathrm{|eight|nine}\,. \qquad (10)$$

3.1 Evaluation

Table 1 shows the average cross-validation results of reCAPTCHA 2010 for a varying amount of bootstrap data. The PLP features outperform the MFCC features by approximately 5 % for the sentence accuracy, which might be due to the occurrence of different speakers in the CAPTCHA, as the PLP features are more suitable for speaker-independent recognition. The highest sentence accuracy for the PLP features is achieved for initializing the models with 50 % of the available data in the respective cross-validation set. Thus, we can see that an increased amount of bootstrap data does not necessarily result in a higher recognition performance, which might be due to overfitting effects that lead to a reduced generalizability of the models.

When comparing our results with those of Bursztein et al. [3], we can see that our ASR-based approach leads to a considerably increased success rate. The maximum achieved sentence accuracy for reCAPTCHA reported in [3] is 1.52 %, a factor of 13 below the best score of Tab. 1. Hence, it is advisable to use speech recognition methods for evaluating an acoustic CAPTCHA as this provides a more realistic indication of the level of robustness against automated attacks.

4. ANALYSIS OF RECAPTCHA 2014

We now provide a security and usability study of the current reCAPTCHA scheme. For this purpose, we have collected approximately 2,500 CAPTCHAS from the reCAPTCHA website [12] between November and December 2013 where we utilize only a fractional amount of the data for training our system. The respective transcriptions were obtained in a listening experiment that is outlined in detail in Sec. 4.2. We selected reCAPTCHA since it is widely used and previous versions of the scheme have proven to be relatively robust against automated attacks in comparison to other CAPTCHAs [3, 26].

4.1 Overview of the CAPTCHA Scheme

The current version of reCAPTCHA is similar to the version of 2010 in that it is only constructed from a sequence of digits between zero and nine. However, the total number of spoken words is not fixed but varied between 6 and 12. Furthermore, the digits are spoken in a block-wise manner, which means that some digits are immediately followed by other digits without any noticeable speech pause in between and some of the digits are even overlapping in time. However, all blocks of those contiguously spoken digits are well separated in time by a short speaking pause and the majority of CAPTCHAs consists of three blocks. The digits are spoken by both a male and a female voice at a very slow speaking rate. The speech appears synthetic and the overall voice quality is comparatively low. All signals exhibit the same stationary background noise and the energy of the background noise is inversely proportional to frequency. The sampling frequency is 16 kHz and the signals are bandlimited to approximately 5.8 kHz. An example of the CAPTCHA is given in Fig. 2 by showing the spectrogram.

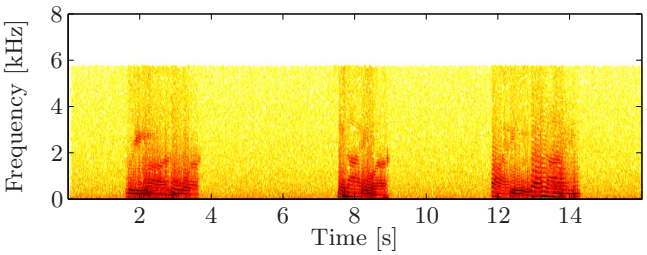

Figure 2: Example of reCAPTCHA 2014 showing the digit sequence "314-694-5279".

4.2 Obtaining Transcriptions

To obtain the CAPTCHA transcriptions that are essential for training the ASR system, we conducted a listening experiment at our institute. The listening experiment took place in a controlled environment, i.e., an audiometric room, in which the acoustic conditions were kept constant throughout. All audio signals were presented via professional studio headphones and the audio volume could be adjusted to the level of comfort by the participants.

A group of 12 human listeners participated in the experiment and each participant was asked to label a set of 50 CAPTCHA signals. The participants were briefed that the signals consist of a varying number of digits that are separated in time by distinct speech pauses. The task was to provide a numeric transcription using only the letters {"0", "1", "2", ..., "9"} for each digit as well as a marker symbol "-" for each speech pause, e.g., a complete transcription may read "894-770-136". The marker symbol is used to enable an identification of individual digit blocks later on. We created a graphical user interface in MATLAB that enables the audio playback as well as the labeling for a set of successively presented CAPTCHA signals. The setup allows the user to playback each audio signal multiple times until a transcription has been saved by the user. It was not allowed to play or edit a previously transcribed CAPTCHA. We collected 4 transcriptions for each CAPTCHA to enable an identification of reliable transcriptions later on. Some of the listeners participated in the experiment multiple times where we ensured that no CAPTCHA was labeled more than once by the same person. In this way, we collected 1000 transcriptions, corresponding to 250 individual CAPTCHAs.

Table 2: Inter-labeler agreement in percent for reCAPTCHA 2014. The results are shown for the individual digit block transcriptions and for the full transcription (i.e., an agreement on all blocks). The scores are based on the listening test results of 250 different CAPTCHAs.

	# Agreements			
	1 (No)	2	3	4 (All)
Block transcription	11.20	29.73	31.87	27.20
Full transcription	49.20	36.00	10.00	4.80

Ethical Considerations.

Before performing the listening test, all participants were informed that they were to take part in a scientific study and that their data was used to analyze the properties of a CAPTCHA. All data collected during our study was used in an anonymized way so that there is no link between collected data and individual participants. Our institute does not fall under the jurisdiction of an IRB or similar ethics committee.

4.3 Usability

After the listening experiment, we received a feedback from some of the participants, stating that most signals are very difficult to understand. This can be confirmed by the fact that there is only a relatively small number of CAPTCHAs where the transcriptions of four different participants match with each other. Table 2 shows the percentage of CAPTCHAs and digit blocks on which there was no agreement (1), and on which 2, 3 or 4 participants agreed. Since each CAPTCHA consists of a fixed number of digit blocks, an agreement on all digit blocks is equivalent to an agreement on the full transcription of the CAPTCHA.

We can see from Tab. 2 that the overall transcription agreement between the participants is comparatively low, as there is only a full agreement on the transcriptions for 4.8 % of the signals between all participants and 49.2 % of the signals received diverging transcriptions, i.e., no pair of participants agreed on the same transcription. This indicates that the spoken digit sequences are perceived differently among the individual participants and that most CAPTCHAs are presumably only partially solvable. Nevertheless, the majority of digit blocks, i.e., 59.07 % (31.87 % + 27.20 %), received an identical transcription by at least 3 participants, which we exploit for training the ASR system in the next section.

4.4 Security Analysis

We assess the security of reCAPTCHA by first training our automated solver offline and then attacking the CAPTCHA signals on the reCAPTCHA website [12]. The employed ASR system uses the type of models described in Sec. 3, since the principal recognition task remains unchanged.

Training the ASR System.

Due to a high discrepancy between the CAPTCHA transcriptions, we are not able to train the ASR system on the full transcriptions in a reliable manner. In order to cope with this issue, we identify those signal parts, i.e., digit blocks, with a high inter-labeler agreement. We extract all digit

blocks that were labeled with at least 3 identical transcriptions. The extraction of individual digit blocks from the CAPTCHAs is based on a *voice activity detector* (VAD), which identifies the temporal locations of digit blocks and noise segments. The VAD represents the same ASR system that is used for the attack with the difference that it uses only two different models, i.e., one universal model for speech and one model for noise. The VAD was trained by using 5 randomly chosen CAPTCHAs that were manually labeled. Using the VAD yields an overall number of 443 individual digit blocks—corresponding to 1500 single digits—that are used for training the attacking ASR system.

Performing the Attack.

The attack on the reCAPTCHA scheme was conducted on February 12th and 13th 2014 by solving the acoustic CAPTCHAs on the reCAPTCHA website [12] via an automated script. This script successively downloads the audio signals and submits the respective transcriptions that have been produced by the ASR decoder. Due to a limited number of possible CAPTCHA downloads within a certain period of time from a single IP address, we used TOR [8] to increase the number of available IP addresses and enable a larger number of CAPTCHA requests for our experiment.

When ignoring the Internet connection speed, the solving time per CAPTCHA depends mainly on the Viterbi decoder of HTK. On average, we solve 2 CAPTCHAs per second on a standard desktop computer, which is 32× faster than a user would need to listen to the signals.

For speech recognition, we exploit our prior knowledge about the CAPTCHA scheme and adjust the employed grammar as follows:

$$\begin{aligned} \mathrm{S} &\rightarrow \mathrm{N\,B\,N\,B\,N\,B\,N}\,, \\ \mathrm{B} &\rightarrow \mathrm{D\,D\,[D]\,[D]}\,, \end{aligned} \tag{11}$$

where S, B, and N denote the start symbol, a digit block and a noise segment, respectively. The symbol D represents the individual digits as defined by Eq. 10 and [D] specifies an optional digit.

We solved 10,864 CAPTCHAs in total, receiving a correct response from the website for 6,827 attempts, which corresponds to an average success rate of 62.8 %.

Comparing this result with the success rate of 19 % for the older version of reCAPTCHA (cf. 3) we can see that the current scheme appears to be easier to for the ASR system. We assume that this is due to the fact that the current version of reCAPTCHA uses synthetic speech rather than real speech recordings as well as a relatively simple background noise signal[3].

Usability Revisited.

After the attack, we possess a reference transcription for 6,827 CAPTCHA signals, which enables us to assess the human recognition scores for the reCAPTCHA scheme. We have conducted a subsequent listening experiment, involving 10 participants, where each participant was asked to provide a transcription for 50 randomly chosen CAPTCHAs. Again, the participants were briefed that they were to take part in a scientific study. With the help of this second listening test, we have collected the human recognition scores for 500 different CAPTCHA signals. Table 3 depicts the human recognition scores for the current reCAPTCHA scheme.

[3]The difference in the complexity of the background noise signals is clearly noticeable when comparing Fig. 1 and 2.

Table 3: Human recognition scores in percent for reCAPTCHA 2014. The scores are based on the listening test results of 500 different CAPTCHAs (sentences), corresponding to 5,140 digits (words).

	Sent Acc.	Word Acc.
Mean	24.40	84.69
Std. dev.	17.35	6.55

We can see from Tab. 3 that the mean sentence accuracy is quite low at only 24.40 %, rendering the intelligibility of the CAPTCHA fairly questionable. The word accuracy is given by 84.69 %, which represents an acceptable score regarding the intelligibility of individual digits.

5. PERCEPTUALLY MOTIVATED DESIGN

In order to render the training of an automated CAPTCHA solver more difficult, it is beneficial to introduce as much variability into the speech signals as possible. In principle, this can be achieved by expanding the lexicon, i.e., by using a higher number of words for the CAPTCHA. Further, it is advantageous to introduce several speakers into the signals that exhibit different speaking characteristics like fundamental frequency, speaking rate and, particularly, dialect. Thus increasing the *complexity* of the speech signal will typically compel an attacker to acquire more labeled CAPTCHA signals for training a robust automated solver.

Another important design aspect is to prevent certain attack strategies. Segmentation-based attacks (cf. [3, 26]) for instance can be hindered or even prevented when designing a CAPTCHA by filling the speech pauses with supplementary signals that have roughly the same energy as the speech, but do not carry any semantic information. This procedure can also confuse the attacking system, especially when the supplementary signals exhibit similar spectral characteristics as the speech signal itself. Suitable signal types are, for instance, speech shaped noise or multi-talker babble noise.

5.1 Exploiting Auditory Perception

There is a variety of complex acoustic scenarios that humans with normal hearing can easily cope with, whereas they represent a challenging task for ASR. One major problem for ASR is the simultaneous occurrence of multiple sound sources that overlap both in time and in frequency. However, the human auditory system possesses the ability of isolating acoustic streams from a mixture of sound events, which is known in the literature as the phenomenon of *auditory streaming* [2, 9]. Hence, for designing robust acoustic CAPTCHAs, we can attempt to exploit the human ability of sound separation while creating a more challenging ASR task. We can construct the audio signal such that the individual words are somewhat superimposed on each other. This renders word isolation in time virtually impossible as there is no separation between adjacent words. Of course, the superposition of words has to be done in a way that allows a human listener to identify the designated order of words. Thus, there has to be a certain time delay between the beginning of two adjacent words.

Another difficulty for ASR is the recognition of speech in the presence of reverberation, as this causes a tempo-

ral smearing or dispersion of the audio signal. Despite this temporal smearing, speech intelligibility in reverberant environments remains relatively high for human listeners [28].

Based on these insights, we generate CAPTCHAs by exploiting both of the above mentioned concepts, i.e., auditory streaming and reverberation, as discussed in the following.

5.2 Generation of Signals

In order to produce CAPTCHAs that are comparable to the reCAPTCHA schemes, we also restrict the underlying vocabulary to digits between zero and nine. For signal generation, we use the TIDIGITS speech corpus [17] which originally consists of 326 different speakers (111 men, 114 women, 50 boys and 51 girls) each pronouncing 77 digit sequences. Here, we created a subset of the corpus by incorporating only the single-digit recordings of 25 male and 25 female speakers, yielding a total number of 1000 individual digits that represent the signal repository for our CAPTCHA generation approach.

Creation of Digit Blocks.

Random digit signals are chosen from the database, alternating male and female speakers. The signals are concatenated in time such that two consecutive digits are superimposed to some extent and we refer to the outcome as a digit block in the following. The number of digit blocks is varied between 4 and 5, which results in a maximum number of 8 and 10 digits per CAPTCHA, respectively. At first, all digit blocks are separated by a silence segment with its length chosen randomly and uniformly from the interval $[0.75\text{ s}, 2.5\text{ s}]$. The superposition of signals is based on the short-time *root mean square* (RMS) power of the isolated digit sequences and is controlled by a small set of parameters. As a first step, the RMS power of the k-th digit signal $x_k(t)$ is computed in a frame-wise manner:

$$p_k(m) = 10\,\log_{10}\left(\sqrt{\frac{1}{L}\sum_{i=0}^{L-1} x_k(mR+i)^2}\right). \tag{12}$$

In order to perform signal concatenation, we derive an individual pivot element for each RMS power curve by searching for the location of the maximum:

$$\hat{m}_k = \arg\max_m \left(p_k(m)\right). \tag{13}$$

Next, we determine two power minima that enclose $\hat{m}_k$ and lie within a certain power range. To achieve this, we first compute a power floor for each digit:

$$\check{p}_k = \hat{p}_k - \beta\overline{p_k(m)}, \tag{14}$$

where $\hat{p}_k$ represents the value of the power maximum that is:

$$\hat{p}_k = p_k(\hat{m}_k), \tag{15}$$

with β as a threshold constant and $\overline{p_k(m)}$ denoting the sample mean of $p_k(m)$. The location of the power minimum on the left hand side of $\hat{m}_k$ is then given by:

$$\check{m}_k^{\mathrm{L}} = \arg\min_{0\ldots\hat{m}_k} \left(|p_k(m) - \check{p}_k|\right) \tag{16}$$

and the location of the power minimum on the right hand side of $\hat{m}_k$ reads:

$$\check{m}_k^{\mathrm{R}} = \arg\min_{\hat{m}_k\ldots M} \left(|p_k(m) - \check{p}_k|\right), \tag{17}$$

where M is the number of available frames. Using these maximum and minimum values, we compute two anchor points for each digit. The left anchor point of the k-th digit is midway between $\hat{m}_k$ and $\check{m}_k^{\mathrm{L}}$:

$$\acute{m}_k = \frac{\hat{m}_k - \check{m}_k^{\mathrm{L}}}{2} + \check{m}_k^{\mathrm{L}} \tag{18}$$

and the right anchor point of the k-th digit is centered between $\hat{m}_k$ and $\check{m}_k^{\mathrm{R}}$:

$$\grave{m}_k = \frac{\check{m}_k^{\mathrm{R}} - \hat{m}_k}{2} + \hat{m}_k. \tag{19}$$

A digit block is then created by aligning two digits at their respective left and right anchor points. Therefore, we superimpose two consecutive digit signals in time such that:

$$\acute{m}_{2l} \stackrel{!}{=} \grave{m}_{2l+1} \quad \forall\, l = 0, 1, 2, \ldots \tag{20}$$

is fulfilled and we denote the signal that contains all digit blocks by $y(t)$. Figure 3 shows an example of the block creation method by means of the RMS power curves for two consecutive digits.

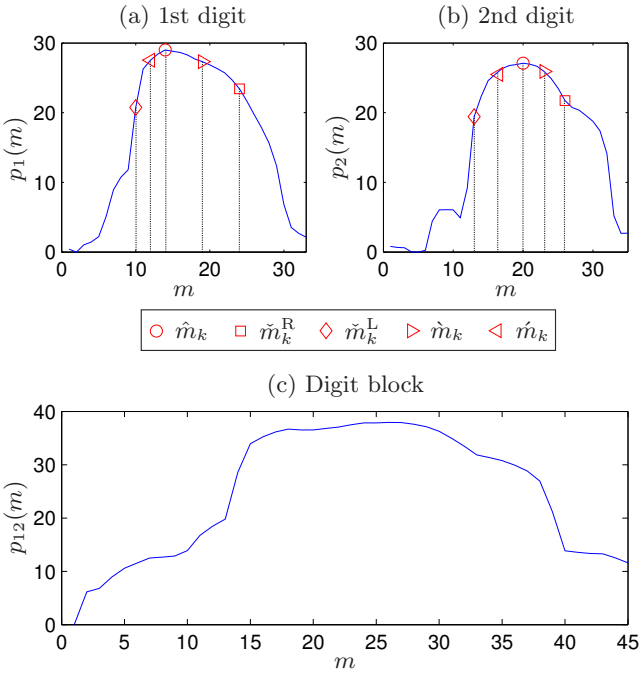

Figure 3: Example of the block creation method. (a) shows the short-time RMS power of the first digit and (b) shows the short-time RMS power of the second digit. The short-time RMS power of the digit block that results from signal concatenation is shown in (c).

Incorporating Background Noise.

After creating the sequence of digits blocks $y(t)$, all silent parts within the signal are superimposed by multi-talker babble noise from the NOISEX-92 database [27]. The NOISEX babble noise is 235 s long and we extract random noise segments of suitable length for creating the noise parts of

the CAPTCHA. The noise segments are scaled such that the RMS power curve of $y(t)$ exhibits no conspicuous peaks and remains fairly stationary. We denote the noisy waveform of digit blocks by $y'(t)$.

It should be stressed at this point that the utilized corpora, i.e., TIDIGITS and NOISEX, were chosen for convenience to provide a proof of concept. We are aware of the fact that the data is not suitable for a practical application of the CAPTCHA as these corpora are publicly available, which would facilitate the training of automated CAPTCHA solvers.

Reverberating Signals.

The final CAPTCHA signal $z(t)$ is obtained by a convolution of the noisy digit sequence $y'(t)$ with an artificially generated room impulse response $w(t)e^{-t/\tau}$:

$$z(t) = y'(t) * \left(w(t)e^{-t/\tau}\right), \tag{21}$$

where $w(t)$ represents white Gaussian noise with zero mean and unit variance and τ is a predefined decay time.

For creating CAPTCHA signals, two different reverberation times are compared. A common way for characterizing reverberation is the time constant T_{60}, which is the time it takes for the room impulse response power to drop by 60 dB. We create CAPTCHA signals for $T_{60} = 100$ ms and for $T_{60} = 300$ ms and compare the human performance as well as the robustness against ASR-based attacks in Sec. 5.3 and Sec. 5.4, respectively. The spectrogram of the CAPTCHA is shown in Fig. 4 for a reverberation time of 100 ms.

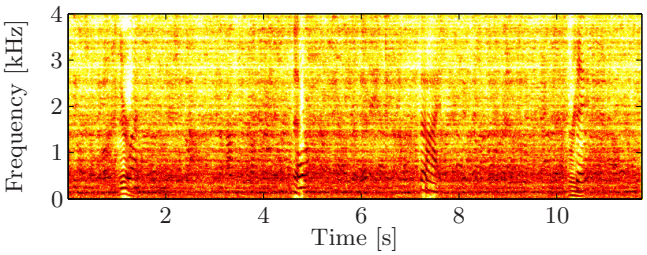

Figure 4: Example of the proposed CAPTCHA design for $T_{60} = 100$ ms showing the digit sequence "01-64-75-36".

5.3 Usability

We assess the usability of the proposed CAPTCHA, i.e., the human success rate, by means of the same listening experiment as it is outlined in Sec. 4.2. This environment does, of course, constitute an ideal situation, which has been chosen for reproducibility and consistency of results.

Two different variants of the proposed CAPTCHA are compared, one with a reverberation time of 100 ms and the other with a reverberation time of 300 ms. In the experiment, we find the results for 200 randomly selected signals at each reverberation time.

The overall number of individual participants was 20 and each participant was asked to transcribe a set of 50 audio signals per experiment. In order to derive the transcription agreements as in Sec. 4.2, each signal is labeled by four different listeners. Some participants attended multiple times where we again ensured that no CAPTCHA was labeled more than once by the same listener.

Table 4: Inter-labeler agreement in percent for the proposed CAPTCHA. The results are shown for the individual digit block transcriptions and for the full transcription (i.e., an agreement on all blocks). The scores are based on the listening test results of 200 different CAPTCHAs for each reverberation time.

	# Agreements			
	1 (No)	2	3	4 (All)
Block transcription	10.43	11.26	24.69	53.63
Full transcription	19.50	28.00	34.00	18.50

(a) $T_{60} = 100$ ms

	# Agreements			
	1 (No)	2	3	4 (All)
Block transcription	9.55	31.42	58.67	0.37
Full transcription	57.50	29.00	13.50	0.00

(b) $T_{60} = 300$ ms

Table 5: Human recognition scores in percent for the proposed CAPTCHA. The scores are based on the listening test results of 800 CAPTCHAs (sentences), corresponding to 7,280 digits (words).

	Sent Acc.	Word Acc.
Mean	56.38	91.74
Std. dev.	21.47	7.18

(a) $T_{60} = 100$ ms

	Sent Acc.	Word Acc.
Mean	37.81	86.88
Std. dev.	17.65	7.90

(b) $T_{60} = 300$ ms

The results for the transcription agreement between the participants are shown in Tab. 4a and Tab. 4b, for the reverberation time $T_{60} = 100$ ms and for $T_{60} = 300$ ms, respectively. The actual human recognition scores, computed by using the reference transcription, are presented in Tab. 5.

Considering first the reverberation time $T_{60} = 100$ ms, we can see from Tab. 4a that the majority of digit blocks (53.63 %) received a transcription agreement from all participants, whereas the number of agreements on the full transcription is comparatively low (18.50 %). This effect is also reflected in the human recognition scores (cf. Tab. 5a), where the word accuracy is satisfactorily high (91.74 %) but the sentence accuracy is only 56.38 %. For the reverberation time $T_{60} = 300$ ms, the number of transcription agreements is significantly lower than with $T_{60} = 100$ ms. Here, the majority of CAPTCHA signals (i.e., 57.50 %) has no agreement on the full transcription. Furthermore, only 0.37 % of the digit blocks show an agreement between all participants and no CAPTCHA has a full transcription agreement between all participants. Regarding the human recognition scores for $T_{60} = 300$ ms, we can see from Tab. 5b that only 37.81 % of the CAPTCHAs are transcribed correctly.

5.4 Security Analysis

The security analysis of the proposed CAPTCHA is based on the same ASR system that is used in the previous sections, to allow for a fair comparison.

System Setup.

Since we have generated all CAPTCHAs ourselves, we have the exact knowledge about the temporal positions of digits within the audio signals. Thus, we utilize 200 signals for bootstrapping the HMM parameters, which represents a high-quality initialization procedure compared to Sec. 3 and Sec. 4, since the transcriptions are perfectly time-aligned with the spoken digits.

After initializing the HMMs, the model parameters are reestimated using a varying number of CAPTCHA signals to analyze the influence of additional training material on the recognition performance. Here, we vary the number of training examples that are used for model reestimation between 200 and 1600 CAPTCHAs. In addition, we compare the results for three different reverberation times, i.e., 0 ms, 100 ms, and, 300 ms, where the former represents the case of a non-reverberant signal. In each case, the recognition results are based on 10,000 randomly chosen CAPTCHAs. We apply a task-specific grammar for signal decoding:

$$\begin{aligned} &\mathrm{S} \rightarrow \mathrm{N\,B\,N\,B\,N\,B\,N\,B\,N\,[B\,N]}\,, \\ &\mathrm{B} \rightarrow \mathrm{D\,D}\,, \end{aligned} \tag{22}$$

where the symbols S, B, D and N are as defined by Eq. 11.

Results.

Table 6 shows the ASR recognition results for a varying number of training examples and different reverberation times. It should be emphasized that this security analysis of the proposed CAPTCHA is much more rigorous than the one applied to reCAPTCHA (cf. Sec. 4.4) as we make use of the ideal reference transcriptions for training the ASR system. Furthermore, the number of digits that were utilized for training is 1.2–9.6 × (1800–14400 digits) higher than for reCAPTCHA (1500 digits).

We can see from Tab. 6 that the sentence accuracy for the non-reverberant case, i.e., $T_{60} = 0$ ms, and the lowest amount of training examples, i.e., 200 CAPTCHAs, is comparatively high at 15.86 %. Additionally, the word and sentence accuracies are further improved by increasing the number of training examples. Here, it appears that the relationship between the number of training examples and the resulting accuracies is virtually linear. Thus, an ASR-based CAPTCHA solver can easily cope with the overlapping digit sequences as well as with the babble noise segments to some degree, rendering these signals non-robust against automated attacks.

For $T_{60} = 100$ ms, the word and sentence accuracy drops by approximately 10–20 %, depending on the number of training examples. This demonstrates that already moderate reverberation effects are already problematic for speech recognition. However, the ASR system is able to find the correct transcription for approximately 6 % of the attacked CAPTCHA signals, averaged over the number of utilized training examples.

As one would expect, the lowest ASR scores are given for $T_{60} = 300$ ms. Here, the ASR system only identifies approximately 3 % of the CAPTCHAs correctly.

We can conclude that the best trade-off between usability and security for the proposed scheme is given for $T_{60} = 100$ ms. A side-by-side comparison between the proposed CAPTCHA and the current reCAPTCHA scheme is provided by Tab. 7. Here, the success rate of the attack is 5.33 % when using 1800 digits for training the ASR system. This is a factor 12 below the success rate found for the reCAPTCHA scheme (62.8 %) by using 1500 digits for system training. Furthermore, the human success rate for the proposed scheme was found to be 56.38 % that is twice as high as for the current reCAPTCHA scheme.

Table 6: ASR results in percent for the proposed scheme. The scores are based on 10,000 different CAPTCHAs (sentences), corresponding to 90,140 digits (words).

# Train	T_{60} [ms]	Sent Acc.	Word Acc.
200	0	15.86	77.03
200	100	5.33	64.49
200	300	1.25	56.11
400	0	17.42	78.38
400	100	5.06	65.32
400	300	2.34	60.20
800	0	20.38	79.71
800	100	6.87	67.21
800	300	3.14	62.88
1600	0	26.43	82.43
1600	100	6.26	67.37
1600	300	4.11	64.66

Table 7: Summary of recognition scores in percent for ASR and human listeners, shown for the proposed CAPTCHA ($T_{60} = 100$ ms) and the current reCAPTCHA scheme.

	ASR	Human
Proposed CAPTCHA	5.33	56.38
reCAPTCHA 03/2014	62.8	24.40

6. CONCLUSION AND FUTURE WORK

In this paper, we show that it is advisable to utilize state-of-the-art speech recognition techniques for evaluating the security of acoustic CAPTCHAs, as this yields significantly higher success rates for CAPTCHA solving than previously suggested classification methods and is therefore more indicative of the security risk of a CAPTCHA. Using the well-known reCAPTCHA as an example, we can see that the general characteristics of the scheme have not changed notably within the last four years, which leads to comparatively high success rates for automated attacks. In addition, the degree of signal distortion has been increased for this scheme at the cost of reducing human accuracy, rendering the trade-off between usability and security problematic.

We propose an experimental CAPTCHA scheme that exploits both the auditory capabilities of humans and weaknesses in current ASR techniques. In this scheme, we render the ASR-based attack more difficult by introducing overlapping speakers and artificial room reverberation effects. The outcome of our experiments is that the proposed CAPTCHA exhibits a far better trade-off between human usability and security as the current reCAPTCHA scheme.

Our results confirm the findings of previous research in that we show that a very conservative CAPTCHA, even if it is only barely intelligible for humans, can potentially be learned by an automated system at a relatively low cost. We can hence assume that it is infeasible to create a small-vocabulary-based CAPTCHA with a very high human success rate ($\geq 90\,\%$) while remaining robust against automatic solvers, i.e., yielding only insignificantly low success rates ($\leq 1\,\%$ or even $\leq 0.01\,\%$). Thus, it becomes necessary to investigate into new directions for CAPTCHA design, such as the utilization of a larger vocabulary or the incorporation of context-based questions that require human intelligence.

For the time being, though, our acoustic CAPTCHA did succeed in significantly improving human pass rates—from 24 % to 56 %—while reducing the attack success rate of an ASR solver from 63 % to 5 %, when comparing with the current, state-of-the-art reCAPTCHA scheme. This lends credibility to the general approach of explicitly considering human auditory processing capabilities in CAPTCHA designs, and should serve as a building block for new systems that offer a satisfactory trade-off between usability and security.

Acknowledgments

The authors would like to thank all participants of the listening experiments. The research was supported by the DFG Research Training Group GRK 1817/1. The CAPTCHA examples of our proposed scheme are available upon request.

7. REFERENCES

[1] S. Bohr, A. Shome, and J. Z. Simon. Improving Auditory CAPTCHA Security. Technical report, ISR, A. James Clark School of Engineering, 2008.

[2] A. S. Bregman. *Auditory scene analysis: The perceptual organization of sound*. MIT press, 1994.

[3] E. Bursztein, R. Bauxis, H. Paskov, D. Perito, C. Fabry, and J. C. Mitchell. The Failure of Noise-Based Non-Continuous Audio Captchas. In *Proc. IEEE Symposium on Security and Privacy*, 2011.

[4] E. Bursztein and S. Bethard. Decaptcha Breaking 75% of eBay Audio CAPTCHAs. In *Proc. WOOT*, 2009.

[5] K. Chellapilla, K. Larson, P. Y. Simard, and M. Czerwinski. Building Segmentation Based Human-Friendly Human Interaction Proofs (HIPs). In *Human Interactive Proofs*. Springer, 2005.

[6] R. Datta, J. Li, and J. Z. Wang. IMAGINATION: A Robust Image-based CAPTCHA Generation System. In *Proc. ACM MM*, 2005.

[7] dfactory. Math Captcha, 2014. `http://wordpress.org/plugins/wp-math-captcha/`.

[8] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-generation Onion Router. In *Proc. USENIX Security Symposium*, 2004.

[9] P. Divenyi. *Speech separation by humans and machines*. Springer, 2005.

[10] I. Fischer and T. Herfet. Visual CAPTCHAs for document authentication. In *Proc. MMSP*, 2006.

[11] M. Gales and S. Young. The Application of Hidden Markov Models in Speech Recognition. *Foundations and Trends in Signal Processing*, 2007.

[12] Google. reCAPTCHA. `http://www.recaptcha.net`.

[13] H. Hermansky, B. Hanson, and H. Wakita. Perceptually based linear predictive analysis of speech. In *Proc. ICASSP*, 1985.

[14] A. Hindle, M. W. Godfrey, and R. C. Holt. Reverse Engineering CAPTCHAs. In *Proc. WCRE*, 2008.

[15] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 2012.

[16] G. Kochanski, D. P. Lopresti, and C. Shih. A reverse turing test using speech. In *Proc. INTERSPEECH*, 2002.

[17] R. G. Leonard and G. Doddington. TIDIGITS Linguistic Data Consortium, 1993. Linguistic Data Consortium.

[18] R. P. Lippmann. Review of neural networks for speech recognition. *Neural computation*, 1989.

[19] N. Morgan, H. Bourlard, and H. Hermansky. Automatic Speech Recognition: An Auditory Perspective. In *Speech Processing in the Auditory System*. Springer, 2004.

[20] picatcha.com, 2014. `http://picatcha.com`.

[21] L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, 1993.

[22] S. Sano, T. Otsuka, and H. Okuno. Solving Google's Continuous Audio CAPTCHA with HMM-Based Automatic Speech Recognition. In *Advances in Information and Computer Security*, 2013.

[23] A. Schlaikjer. A dual-use speech CAPTCHA: Aiding visually impaired web users while providing transcriptions of Audio Streams. *LTI-CMU Technical Report*, 2007.

[24] R. Soni and D. Tiwari. Improved captcha method. *IJCA*, 2010.

[25] J. Tam, J. Simsa, D. Huggins-Daines, L. von Ahn, and M. Blum. Improving Audio CAPTCHAs. In *Proc. SOUPS*, 2008.

[26] J. Tam, J. Simsa, S. Hyde, and L. von Ahn. Breaking Audio CAPTCHAs. In *Proc. NIPS*, 2008.

[27] A. Varga and H. J. M. Steeneken. Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems. *Speech Communication*, 1993.

[28] T. Virtanen, R. Singh, and B. Raj. *Techniques for Noise Robustness in Automatic Speech Recognition*. Wiley, 2012.

[29] L. von Ahn, M. Blum, N. J. Hopper, and J. Langford. CAPTCHA: Using Hard AI Problems for Security. In *EUROCRYPT*. Springer, 2003.

[30] L. von Ahn, M. Blum, and J. Langford. Telling Humans and Computers Apart Automatically. *CACM*, 2004.

[31] J. Yan and A. S. El Ahmad. Breaking visual captchas with naive pattern recognition algorithms. In *Proc. ACSAC*, 2007.

[32] J. Yan and A. S. El Ahmad. A Low-cost Attack on a Microsoft CAPTCHA. In *Proc. ACM CCS*, 2008.

[33] S. Young. The HTK Hidden Markov Model Toolkit: Design and Philosophy. *Entropic Cambridge Research Laboratory, Ltd*, 1994.