*rt*Captcha: A Real-Time CAPTCHA Based Liveness Detection System

Erkam Uzun, Simon Pak Ho Chung, Irfan Essa and Wenke Lee
Georgia Institute of Technology
euzun@gatech.edu, pchung34@mail.gatech.edu, irfan@gatech.edu and wenke@cc.gatech.edu

***Abstract*—Facial/voice-based authentication is becoming increasingly popular (e.g., already adopted by MasterCard and AliPay), because it is easy to use. In particular, users can now authenticate themselves to online services by using their mobile phone to show themselves performing simple tasks like blinking or smiling in front of its built-in camera. Our study shows that many of the publicly available facial/voice recognition services (e.g. Microsoft Cognitive Services or Amazon Rekognition) are vulnerable to even the most primitive attacks. Furthermore, recent work on modeling a person's face/voice (e.g. Face2Face [1]) allows an adversary to create very authentic video/audio of any target victim to impersonate that target. All it takes to launch such attacks are a few pictures and voice samples of a victim, which can all be obtained by either abusing the camera and microphone of the victim's phone, or through the victim's social media account. In this work, we propose the Real Time Captcha (*rt*Captcha) system, which stops/slows down such an attack by turning the adversary's task from creating authentic video/audio of the target victim performing *known authentication tasks (e.g., smile, blink)* to *figuring out what is the authentication task, which is encoded as a Captcha*. Specifically, when a user tries to authenticate using *rt*Captcha, they will be presented a Captcha and will be asked to take a "selfie" video while announcing the answer to the Captcha. As such, the security guarantee of our system comes from the strength of Captcha, and not how well we can distinguish real faces/voices from synthesized ones. To demonstrate the usability and security of *rt*Captcha, we conducted a user study to measure human response times to the most popular Captcha schemes. Our experiments show that, thanks to the humans' speed of solving Captchas, adversaries will have to solve Captchas in less than 2 seconds in order to appear live/human and defeat *rt*Captcha, which is not possible for the best settings on the attack side.**

I. Introduction

With automatic facial/voice recognition becoming more accurate[1] and available[2], online services are increasingly allowing their users to authenticate using their face/voice. In such authentication schemes, all one needs to do to authenticate is to perform simple tasks like smile, blink or nod in front of his/her mobile phone, while the phone's camera will record the video of the user performing such a task and send it to the service provider. The authentication is successful if the service provider determines that the received video is indeed that of the expected user performing the required task.

While these new generation facial-recognition-based authentication systems offer superior usability and are robust in benign settings, existing works already have shown that they are quite vulnerable to several kinds of attacks. Depending on how the malicious video/media is fed into the authentication system in order to impersonate a user, existing attacks can be classified as *presentation attacks* and *compromising attacks* (see Fig. 1 for an illustration).

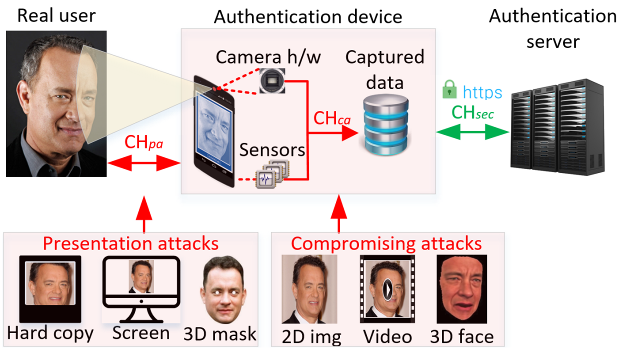


Fig. 1. Attack channels specified by ISO/IEC 3017-1 standard and possible spoofing media types deployed via these channels. CH$_{pa}$ and CH$_{ca}$ represent presentation and compromising attack channels respectively.

As we can see in Fig. 1, presentation attacks physically "present" the impersonating media (mostly static photos or masks) to the sensors used by the authentication system, while compromising attacks involve tempering/fabricating the digital output of the sensors.

An observation we can make here is that compromising attacks are potentially much more scalable than presentation attacks, because compromising attacks can happen entirely in the digital world, while presentation attacks are restricted to physically presenting something to the target system. Recent advances in face modeling (e.g. Face2Face [1]) further make it easier to automate compromising attacks. In particular, all an attacker needs to generate authentic looking video of the

[1]Facebook [2] and Google [3] have respectively achieved recognition accuracies of 97.35% and 99.63% on faces under different illumination, pose and facial expressions

[2]Through cloud services like Microsoft Cognitive Services or Amazon Rekognition

Network and Distributed Systems Security (NDSS) Symposium 2018
18-21 February 2018, San Diego, CA, USA
ISBN 1-891562-49-5
http://dx.doi.org/10.14722/ndss.2018.23253
www.ndss-symposium.org

target victim performing the task necessary for authentication are pictures or videos of the victim. Such materials can be obtained either through abusing the camera of the authentication device, or through accessing publicly available data from the victim's social media accounts– all very doable once the attacker has a foothold on the authentication device (e.g. have their malicious app installed and granted the right permissions). Once such "authentication video" is generated, the attacker will need a way to feed it into the authentication system to launch a compromising attack. This final step can be achieved in a number of ways, e.g. through compromising the authentication device's OS (and then modifying the output buffer of the camera) or through reverse engineering the authentication protocol and directly talking to the server.

Based on the above observation that compromising attacks are much easier to launch in a large scale than presentation attacks, we will focus on the former in this paper. However, as we will note in Section §VIII, our solution is also expected to make state-of-the-art presentation attacks more detectable.

Regarding existing work in defense against both presentation and compromising attacks, we can argue that *most defenses focus on making it harder for the adversary to generate sufficiently good video of the victim performing the "authentication task."* An obvious example is to perform better analysis of the received video to determine if it came from a real human or was somehow synthesized. A less obvious example is liveness detection, which is mainly used to defeat "replay" attacks that employ static pictures of the victim; and this simply pushes the attackers to create animatable 3D models, instead of continuing to use static pictures. *What remains constant is that the "authentication task" is predefined, fixed and known to the adversary, and this allows the adversary to develop new technologies to create authentic videos of the victims performing the right task.*

In this work, we propose a different, orthogonal approach; instead of making it harder for the adversary to generate video of the victim performing the known task to fool our system, *we make it harder for the adversary to know what is the required task the user must perform in order to successfully complete the authentication*. In particular, at each authentication attempt, instead of asking the user to perform simple actions such as blink or smile in front of the camera on the authentication device, our server will send the user a Captcha and have the authentication device take a video of the user answering that Captcha. *For a normal user, this will be easy, but for an automated attack, this will mean automatically solving the Captcha before feeding the answer into the algorithm for generating fake video of the user answering the Captcha.* As in all Captcha schemes, we can assume that the time it takes for a real user to solve the Captcha is significantly shorter than it is for the adversary (even if some human intervention is involved, see Section §VII for our evaluation of this claim), and only authenticate in case the correct response is received within some threshold time.

To summarize, the high level idea of our system is to turn the attacker's task from generating good quality video of the victim to one of breaking Captcha– a task that is well understood and studied by the security community. *A point worth noting is that while the security of our system depends on the security of the Captcha scheme we used, we consider the security of Captcha an orthogonal problem. We believe Captcha (i.e. being able to tell if a user of an online service is a human) is so important, not only to us, but to the whole online ecosystem, that the security of Captcha will continue to improve, and any improvement in the security/robustness of Captcha can be easily adopted to our system.*

We have implemented our high level idea in a prototype system called *rt*Captcha. Our user study shows that normal human response time to the Captcha presented at authentication time is less than 1 second even for the most complex scheme. We also conducted experiments on the same challenges with existing Captcha solving services and state-of-the art techniques which has 34.38% average recognizing accuracy and 6.22 seconds average execution time [4]. In other words, there is a very large safety margin between the response time of a human solving a Captcha and a machine trying to break one.

In summary, the contributions of this work are:

1) We perform an empirical spoofing analysis on current cloud based audio/visual recognition and verification systems that use state-of-the art data-driven deep learning architectures.
2) We propose a practical and usable liveness detection scheme by using security infrastructure of Captchas to defeat even the most scalable and automated attacks.
3) We perform analysis on existing automated and man-powered Captcha-breaking services and state-of-the art Captcha-solving algorithms by using the most popular Captcha schemes in the market.
4) We have implemented a prototype Android application and conducted a user study.
5) Our evaluations show that audio response of a normal human being to a Captcha challenge is much shorter than automated attacks which have state-of-the art synthesizers and Captcha-breaking methods.

In the rest of the paper, we will introduce our threat model in Section §II, present existing attacks and defense mechanisms in Section §III, present our experiment on launching compromising attacks against existing, publicly available systems in Section §IV, describe design and details of *rt*Captcha in Section §V, evaluate the human performance and existing Captcha breaking tools on solving Captcha challenges along with usability and user acceptance of *rt*Captcha in Section §VI, provide the security analysis of *rt*Captcha against our threat model in Section §VII, discuss about limitations and future works in Section §VIII, and conclude in Section §IX.

II. Threat Model

In this work, we focus on defending against *powerful, automated* compromising attacks. We assume the following threat model:

- the authentication device is a mobile phone with a camera and a microphone,
- the authentication server is *not* compromised,
- the kernel of the authentication device can be compromised,
- there is no form of attestation mechanism on the authentication device, since software attestation is theoretically security by obscurity and hardware attestation is not yet available for phones,
- the protocol between the client app running on the authentication device and the server is known to the attacker, thus the attacker can run malicious version of the client app on the authentication device that will completely control the camera and microphone input to the authentication server,
- the attacker can abuse the camera and microphone on the authentication device to collect samples of the face and voice of the victim; the collected samples can then be used to generate models of the victim's voice and face, which can then be used to synthesize videos and audios for impersonating the victim during a future authentication session,
- the attack needs to be completely automated, and it needs to happen on the victim's authentication device, otherwise we believe the attack cannot scale.

Out-of-scope: Based on above assumptions, we consider the following attacks out of scope for this work. 1) Presentation attacks which involve showing 2/3D printed and wearable masks, hard copy photos or device screens displaying the target's face and other rudimentary manipulations, (since this means the attack must happen on an authentication device physically controlled by the attacker, thus violating the last assumption in the threat model). 2) Facial mimicry manipulations through face reenactment [1], [5], where facial expressions of an imposter solving the Captcha are captured and applied in real time to a 3D model of the victim to synthesize the victim's face responding to our liveness detection challenge[3] (this requires a source actor to perform the act before the software can map it to the target subject, which is not a practical and scalable attack scenario for automated tasks).

III. Related Work

In this section, we summarize existing liveness detection methods against both presentation and compromising attacks.

A. Presentation Attacks and Defenses

The requirement of liveness detection systems against face spoofing attacks was first introduced by researchers who showed that existing face authentication applications for both desktop and mobile platforms are vulnerable to single image spoofing [6], [7]. As a defense mechanism against this attack, researchers proposed challenge-response based liveness detection mechanisms that involve user interaction such as smile, blink, lip or head movement, etc. [8], [9]. However, frame switching or video-based attacks proved how easy it was to bypass smile or blink detection since they have arbitrary facial frames creating a motion to fulfill desired challenges [10]. Both image and video-based attacks are deployed as presentation attacks, but, they also are suitable for a compromising attack scenario. However, the latter attacks and corresponding defense mechanisms have been sophisticated for either presentation or compromising attacks.

Against presentation attacks, researchers mainly focused on discriminating 3D structure, texture or reflectance of a human face from a planar surface. To this end, 3D shape inferring features such as optical flow and focal length analysis, color and micro texture analysis, or features extracting reflectance details (such as visual rhythm analysis) have been proposed against presentation attacks [11]–[16]. On the other hand, researchers proposed a wearable 3D mask for presentation attacks to defeat all of these anti-spoofing methods. However, reflectance and texture analysis-based defense mechanisms have also been proposed against 3D mask attacks [17]–[20]. It is worth noting that many different approaches and design choices have been proposed at the competitions on the countermeasures to presentation attacks [21]–[23].

B. Compromising Attacks and Defenses

Recent advances in 3D face model creation (using a couple of images) have been employed to launch compromising attacks [24]. In order to capture enough raw material for model generation, the victim's face/voice could be captured through a user interface (UI) redressing attack caused by a malicious app that allows particular permissions (e.g. draw-on-top on Android device [25]) without his/her notice. To generate a 3D face model from captured image/video, the most suitable approach in existing literature is to use pre-built 3D Morphable Models (3DMMs) [26]–[28]. 3DMMs are the statistical 3D representations built on facial textures and shapes of many different subjects (e.g. 10,000 faces in [27]) by incorporating with their facial expressions and physical attributes at the same time. Once built, a 3DMM is ready for reconstruction according to facial attributes of a victim's face. The details of building a 3D face model could be found in [27], but the overall pipeline is as follows. First, facial landmarks which express pose, shape and expression are extracted from the victim's face. Then, the 3DMM is reconstructed to match the landmarks from both the 3D model and the face. Hence, pose, shape and expression of the face are transferred to the 3DMM. After reshaping the 3DMM, texture of the victim's face is conveyed to the 3D model. Since a 2D face photo/frame does not contain full representation of its 3D correspondence, a photo-realistic facial texture is generated from the visible face area in the photo/frame for missing parts in the 3D representation [29]. Then, this 3D face is transferred into a VR environment to fulfill requested challenge tasks (e.g., smile, blink, rotate head, etc.).

On the defense against compromising attacks, even though some inertial sensor-assisted methods increase the security

[3]In [1], this can be achieved with a 20ms latency.

of facial authentication systems [30], such a compromised environment with given permissions can allow attackers to use additional sensor data to manipulate the motion of 3D face model in a VR environment. Another defense mechanism against these attacks, especially against VR based ones, could be analyzing the authentication media by using forensic techniques to detect forged audio/video [31], [32]. However, since 3D face models are created from scratch with high fidelity texture data, these methods could not detect any forgery on spoofing media. On the other hand, new approaches (such as discrepancy analysis on color filter array of camera sensor noise or multi-fractal and regression analysis on discriminating natural and computer generated images) could be used as countermeasures against 3D-face-model-based attacks [33], [34]. However, attackers can extract genuine noise patterns or features from existing or captured images to embed them into generated video in a compromised device, thus, these defense mechanisms also fail against our threat model [35]. Hence, defense mechanisms against compromised attacks should not rely on additional device data as suggested in previous works.

User authentication through audio response to text challenges is first proposed by Gao et al. [36]. However, their goal is mainly to distinguish between natural and synthesized voice. Their results show that human responses can pass the system with 97% accuracy in 7.8 seconds average time while a very basic text-to-speech (TTS) tool (Microsoft SDK 5.1[4]) can pass the system with 4% success rate. In contrast to *rt*Captcha, the work in [36] use plain-text challenges and thus allows the attacker to easily learn what is the task involved in the liveness detection challenge, and thus can be easily defeated by more sophisticated real-time synthesis of the victim's voice (e.g. [37], [38]). Shirali et al. [39] proposed a scheme that involves audio Captchas. In their system, challenges are sent to users in audio formats and users give audio responses back to the system. They use audio features such as Mel-Frequency Cepstral Spectrum (MFCC) to correlate challenge and response audios at the decision side. They achieved 80% of authentication accuracy on average. However, since breaking audio Captchas are as easy as breaking plain-text challenge by using a speech-to-text application, this work also does not provide good defense against compromising attacks. To the best of our knowledge, *rt*Captcha is the first approach that binds a text-based Captcha challenge response with user's biometric data in the realm of audio/visual liveness detection.

IV. Evaluating the Security of Existing Systems

In this section, we study how vulnerable the most popular facial and voice-authentication systems are to the compromising attacks which motivate our work in *rt*Captcha. In particular, we tested all studied systems against compromising attacks of various levels of sophistication in terms of how they create the impersonating video/audio of the victims, using open source spoofing datasets.

[4]www.microsoft.com/en-us/download/details.aspx?id=10121

Facial Authentication Systems Studied

We tested most popular cloud-based facial recognition services that are provided or funded by major companies such as Microsoft[5], Amazon[6], AliPay(Face++)[7] and Kairos[8].

Database: We tested each studied system against videos showing real/fake faces. We used subjects from the open source CASIA Face Anti-Spoofing Database [40]. In particular, we took the genuine videos from the CASIA Face Anti-Spoofing Database and: 1) used them as positive samples to test each studied system, and 2) used them as samples for generating synthesized videos, and used them as negative samples against each tested system. For our experiment, we used the first 10 subjects from the CASIA database.

Synthesizing methods: We tested each studied system against videos synthesized using methods with various levels of sophistication. Fig. 2 presents a complete set of synthesized video for a user in the database (5th subject in training set). We can summarize the synthesizing techniques employed starting from the most complex to the simplest as followed:

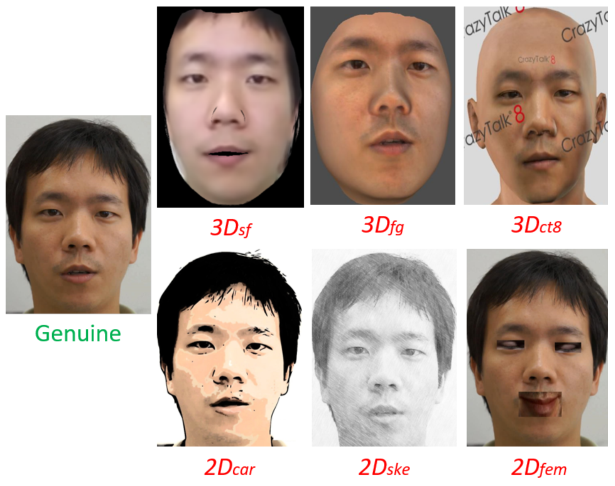


Fig. 2. A full media set including genuine and fake versions of it for a subject in our face authentication database.

1) 3D Face Model: This is the state-of-the art method for generating fake face video for the purpose of compromising attacks [24]. For our experiments, we generated 3D face models from genuine videos of each subject in our dataset by using three different tools: i) Surrey Face Model (labeled as $3D_{sf}$ in Fig. 2), a multi-resolution 3DMM and accompanying open-source tool [28]; ii) FaceGen[9] ($3D_{fg}$), and iii) demo version of CrazyTalk8[10] ($3D_{ct8}$) commercial tools used for 3D printing or rendering 3D animation and game characters. Although the demo tool puts a brand mark on 3D models, they don't seem to have any effect on the effectiveness of the attack.

[5]azure.microsoft.com/en-us/services/cognitive-services/
[6]aws.amazon.com/rekognition/
[7]www.faceplusplus.com
[8]kairos.com
[9]https://facegen.com
[10]https://www.reallusion.com/crazytalk

TABLE I
BASELINE AND SPOOFING RESULTS OF CLOUD BASED FACE AUTHENTICATION SYSTEMS

	Baseline / Overall Conf. (%)		Number of Verified Faces Over 10 / Overall Confidence Rate (%)					
Cognitive Service	**TP**	**TN**	$3D_{sf}$	$3D_{fg}$	$3D_{ct8}$	$2D_{car}$	$2D_{ske}$	$2D_{fem}$
Face API MS Azure	**10 / 78**	**10 / 65**	10 / 70	10 / 75	10 / 70	10 / 82	10 / 84	10 / 86
Kairos Human Analytics	**10 / 80**	**8 / 58**	10 / 75	10 / 78	10 / 73	10 / 91	10 / 83	10 / 80
Cognitive Services Face++	**10 / 87**	**10 / 83**	10 / 86	10 / 71	10 / 72	9 / 77	7 / 80	7 / 75
Amazon Rekognition	**10 / 97**	**10 / 82**	10 / 89	8 / 77	9 / 67	7 / 84	6 / 84	9 / 89

2) Cartoonized and Sketch Photos: To detect whether the face authentication systems check the texture information or not, we convert randomly grabbed frames from the genuine videos to cartoonized and sketch forms[11]. We express these manipulations with $2D_{car}$ and $2D_{ske}$, respectively.
3) Fake Eyes/Mouth Photo: Finally, we replaced eyes and mouth regions of the stationary photos with fake ones which are cropped from an animation character. We conduct this attack method to prove that facial authentication and verification systems only focus on the location of facial attributes. To create appropriate fake eyes and mouth, we first extract the facial landmarks to get their regions. Afterwards, we reshape our fake eyes and mouth templates to exactly fit their corresponding regions. This manipulation is represented by $2D_{fem}$ in the evaluation results.

Methodology: We experimented with each studied service as followed: we enrolled each subject with his genuine face sample. After the enrollment, we established the baseline performance of each service by presenting one genuine face from the enrolled user (thus measuring its true positive, TP) and one genuine face from a different user (thus measuring its true negative, TN). To test the robustness of each service against attacks, we presented each service with all of our synthesized videos. To make the experiment more realistic, we generated the synthesized videos using samples different from those used for registration. The success rate of each synthesis technique and its overall similarity rates (which is the tested service's measure of how close the presented video is to the one from registration) are in Table-I. Since most of the services accept 50% of similarity rate for correct verification, we also consider this threshold in our experiments.

Findings: First of all, under benign conditions, we find the analyzed services have an overall baseline true positive (TP) of 100% and an overall baseline true negative (TN) of 95%, with 85.5% and 75.7% overall confidence rates, respectively. Our results also show that Amazon Rekognition service performs best among all tested service, since its confidence rates on both TP and TN are highest. Unfortunately, we also find that all of the analyzed services are vulnerable against almost all of the tested synthesis techniques. Results show that 92.5% of the spoofed faces are detected as genuine copies with an average similarity rate of 79%. More specifically, Cartoonized and Sketch photo attacks showed that the texture information is not considered in the authentication process at these systems. When we made detailed analysis to understand the reason for a lower matching rate in the Sketch photo attack, we conclude that it is because the tested services cannot detect facial region on those samples. The success of Cartoonized and Sketch photo attacks highlights that attackers can succeed without putting much effort in building a high fidelity facial texture; it would add to the latency in generating the synthesized video to answer the liveness detection challenge presented. Moreover, results of fake eyes/mouth spoofing amusingly proved that all of these systems are only using the landmark locations as the facial feature set on their face authentication protocol. 3D facial model spoofing results also support these outcomes since we used non-sophisticated tools to create 3D models and facial textures. Even though the demo software puts some brand marks over the generated face, we still get very high similarity rates with these 3D models. Hence, faces created by one of the latest 3D facial model generation software (e.g. [29], [41]) are very unlikely to be detected as fake by these services. As a result, we can make an inference that even if a facial authentication scheme uses a challenge-response based liveness detection mechanism (such as smile/blink detection) accompanying one of these services, it will be very easy to spoof such a scheme even by conducting a rough switching frame manipulation (e.g. when asked to blink, go from a frame with open eyes to one with closed eyes for a short time) or using a demo application to create 3D face model and manipulate the model to answer the challenge. For instance, Fig. 3 shows how easy it is to get a high smiling probability from Microsoft Cognitive Service even with a rough manipulation on a genuine face while preserving similarity rate around 78.52%. Assuming a security mechanism that uses smile-detection as a liveness clue and MS Face API to authenticate user face (as Uber does for driver authentication), then a crude attack as in the figure can defeat this mechanism without using any sophisticated tool or algorithm.

Voice Authentication Systems

In this section, we show that automatic speaker verification (ASV) systems also have similar vulnerabilities to compromising attacks as do their facial recognition counterparts. To make a clear demonstration, we systematically conducted attacks with synthesized voices from the Microsoft Speaker

[11]http://www.cartoonize.net

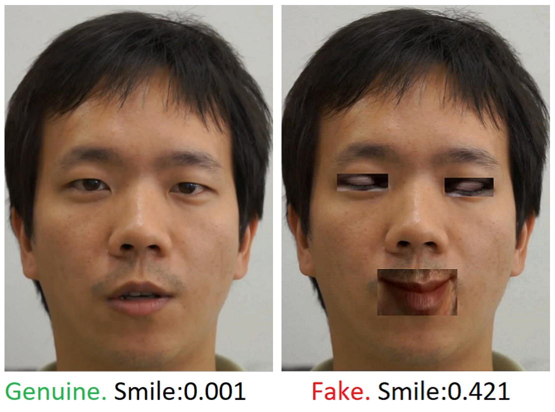


Fig. 3. Smiling probabilities of genuine and fake samples while their similarity rate is 78.52%.

Identification (SI) service by using open-sourced synthesized speech data sets.

Database: In our experiments, we used two different datasets, from ASV Spoofing Challenge [42] (V_{asv}) and DNN based speaker adaptation work by Wu et al. [38] (V_{dnn}). The first dataset, V_{asv}, contains both genuine and synthesized voice samples for a total of 106 male and female users. ASV Spoofing Challenge is organized for Interspeech 2015 Conference to determine best technique detecting spoofing methods against automatic speaker verification systems. Hence, organizers published V_{asv} dataset containing synthesized versions of genuine data which are generated by 7 voice conversion (VC) and 3 speech synthesizing (SS) techniques. We denoted these samples from V^1_{asv} to V^{10}_{asv}. Some of the synthesized data have published before the submissions evaluated, which are called *known attacks*, and some of them are used only for evaluation which are called *unknown attacks*. Overall spoofing detection accuracy of the submissions is around 97% for *known attacks* and 91% for *unknown attacks*. It is worth noting that almost all submitted countermeasure methods perform worst on the last SS based samples (V^{10}_{asv}).

The second dataset, V_{dnn}, contains both genuine and synthesized samples for one female and one male speaker, where the synthesized speech samples were generated by using 7 different settings of their DNN framework. These samples are denoted as V^{1-7}_{dnn}. Objective and subjective experiments prove that all DNN techniques in the paper have better adaptation performance than the hidden Markov model baseline in terms of naturalness and speaker similarity. Hence, we use this dataset as the state-of-the art spoofing attacks.

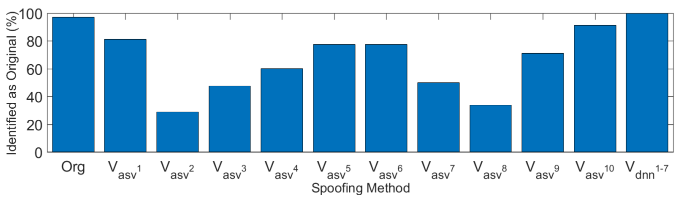


Fig. 4. Success rate of speaker spoofing attacks to Microsoft SI service.

Methodology: We first enrolled 10 users using their genuine samples from the two datasets, (2 users from V_{dnn} and 8 randomly selected users from V_{asv}), each with a total of 30 seconds of speech samples. We then tested the targeted service against 10 genuine samples from the enrolled user, as well as 7 (for V_{dnn}) or 10 (for V_{asv}) synthesized samples generated for the enrolled user by each tested technique, and evaluated if each tested sample was successfully identified as the enrolled user.

Findings: In Fig. 4, we present the genuine identification results for the genuine samples (Org), synthesized samples generated by 10 different methods in the V_{asv} dataset (V^1_{asv} to V^{10}_{asv}) and 7 different DNN methods in the V_{dnn} dataset from left to right. V^{1-7}_{dnn} average result is given for 7 DNN based synthesizers in the V_{dnn} dataset. First of all, we note that 97% of the genuine samples were identified correctly. Hence, it shows that the cloud service is working accurately for the recognition tasks. On the other hand, samples synthesized by various tested SS and VC methods have an average success rate of 64.6%. More specifically, even with the worst performing VC tool, there are still 28.75% of the synthesized samples identified to be from the real enrolled user. Additionally, samples from open sourced TTS synthesizers (10^{th} method of V_{asv}) can have a 90% chance of being considered legitimate. Finally, if an adversary generate synthesized voice of a victim by using a DNN based approach, the SI service identify the forged speakers as a genuine one 100% of time (this is true for all methods/settings in V_{dnn}). The results also prove that the parameter space to synthesize is much more bigger than those which are used by verification methods. That is why, even the simplest VC approach can tune the voice characteristics of the victim to the level of a verification system's requirements.

V. Our Approach

The workflow of *rt*Captcha is summarized in Figure 5. The workflow starts when an authentication device (i.e. mobile phone) needs to start an authentication/registration session; to proceed, it will establish a secure connection with our authentication server. Upon receiving requests over the secure channel, our server will generate and send a Captcha challenge to the authentication device and measure the time until the authentication device responds. The session will time out if no response is received after a predefined period.

On the authentication device, once it receives the Captcha challenge, it will display the challenge to the user and start recording the user's audio response. The client app running on the authentication device will also take a number of snapshots of the user at random time while he/she is responding to the challenge, using the front camera on the phone. We use the phone's voice recognition system to determine when the user has finished responding to the Captcha challenge; the captured voice and face samples will then be sent to the server using the established secure channel. To avoid unnecessarily utilizing the more expensive voice/facial recognition service, our server will perform initial sanity check of the response by transcribing the audio response received using a standard speech-to-text

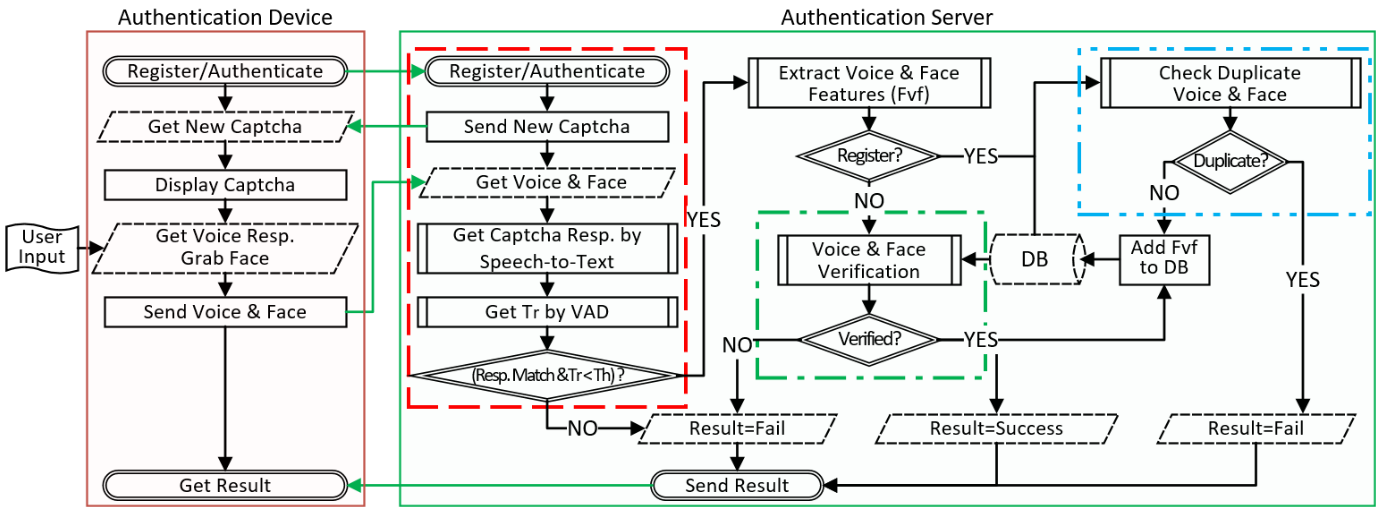


Fig. 5. Process flow diagram of the Real Time Captcha. (T_r, T_h and F_{vf} refer to user response time, human response time threshold and face & voice feature vector, respectively.)

(STT) algorithm to determine if the response corresponds to the Captcha solution to the challenge we sent. We will also determine how much time it takes for the user to start responding to the challenge by measuring when did the first speech activity happen in the received response. If the user took too long to start responding, we will consider the liveness test a failure and reject the authentication/registration request. If the received response passes the preliminary checks, we'll perform the more expensive analysis to determine if the validity of the received voice and face samples (the exact process will depend on whether the request is for authentication or registration, and we will detail the process for each case below).

Registration: Our analysis for registration is very simple and mostly involves sanity check of the received face and voice sample to make sure they came from a real human being to further avoid bot registration and avoid wasting resources to establish accounts for non-existent/non-human users. If necessary, we can also match the received samples against that of all existing users to detect attempts to register multiple accounts for the same person. If the face and voice samples pass all our tests, we will proceed to create the new user account and tie the received face and voice sample to that user and use them for future authentication sessions.

Authentication: For authentication requests, if the user is trying to authenticate as user X, we will compare the received facial and voice samples against the samples received at the establishment of account X. If the samples are verified as coming from user X, we can confirm the liveness and authenticity of the request; liveness is confirmed since the Captcha challenge is correctly answered, authenticity is confirmed through the matching face and voice sample. Thus, we will report to the user that the authentication is successful and let him/her log in as user X. Upon successful authentication of a user, we can also add the received face and voice sample for this authentication attempt to the user's record to improve his/her face and voice profile for future authentication.

Using this framework, we can prevent the adversary from launching automatic, large-scale user impersonation using compromised phones. In the following, we will provide implementation details of the *rt*Captcha system.

A. *Captcha Challenge*

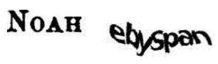
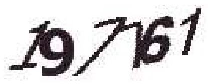
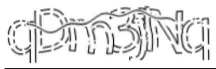
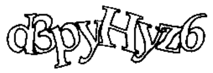
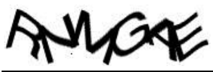
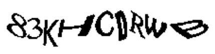
For our implementation of *rt*Captcha, we've employed a number of commonly used Captcha generation tools so we can experiment with and fine tune the difficulty level of our liveness detection. In the following we will give a brief description of the Captcha schemes we've experimented with.

In the literature, text-based Captchas are classified into three different categories according to font styles and positional relationships between adjacent characters. The three categories are, namely, character isolated (CI) schemes, hollow character schemes, and crowding characters together (CCT) schemes [4]. Some Captcha providers also use variable character sizes and rotations or different kinds of distortions and background noises to make their Captcha harder to break. For our experiments, we obtained Captcha samples used by Gao et al. [4] (which conducted generic Captcha breaking attacks on them). We also have modified the *Cool PHP Captcha*[12] framework to create variable size Captchas of short phrases or numbers to include random lines on background. Table II summarizes different Captcha schemes we have experimented with in our user study and evaluations.

Regarding the hardness of these Captcha schemes, Brodic et al. [43] shows that an average Internet user can solve text and numeric Captchas in hollow and CCT (reCaptcha) schemes at around 20 seconds on average (3 secs. min.). They also show that Captcha solving time is correlated with education and age. However, previous findings focus on the scenario where the user has to *type* in the answer to the Captcha, while in our case, *they only have to speak out the answer*, which should

[12]Cool PHP Captcha is used in the reCaptcha scheme, and is available at https://github.com/josecl/cool-php-captcha

TABLE II
MOST POPULAR CAPTCHA SCHEMES

Sample	Scheme	Websites
	reCAPTCHA (CCT scheme)	linkedin, facebook google,youtube,twitter blogspot, wordpress...
	Ebay (CCT scheme)	ebay.com
	Yandex (Hollow scheme)	yandex.com
	Yahoo! (Hollow scheme)	yahoo.com
	Amazon (CCT scheme)	amazon.com
	Microsoft (CI scheme)	live.com bing.com

be faster and easier than typing. Thus, we have performed our own user study to determine how long it will take users to complete the liveness challenge in our settings. Our findings are reported in Sect. VI.

B. Transcribing Captcha Responses

As a first step in our validation of the face and voice samples received for the liveness test under *rt*Captcha, we transcribe the voice sample using a speech-to-text (STT) algorithm to see if it's a correct answer to the Captcha we sent. In our framework, we used a Hidden Markov Model (HMM)-based approach with a pre-trained dictionary. We used the open-sourced CMU Pocketsphinx library, Carnegie Mellon University's Sphinx speech recognition system [44], in our user study app since it provides a lightweight library working on mobile devices. CMU Sphinx is the state-of-the art solution among HMM based approaches. There also are many sophisticated alternatives for this step. For example, recently Baidu's open source framework Deep Speech 2 exceeds the accuracy of human beings on several benchmarks [45]. They trained a deep neural network (DNN) system with 11,940 hours of English speech samples. Cloud-based cognitive services such as Microsoft Bing Speech API[13] or IBM Watson Speech to Text[14] also could be used as STT algorithm for this step. However, network latency caused by audio sample transmission could be a drawback in our framework.

C. Audio Response Validation

Given a verified audio response of the Captcha challenge, the next verification process tests user response time to the challenge. Unlike typing-based responses, our further analysis will show that giving audible response is much faster. Furthermore, the attacker's time window for breaking Captcha challenges and synthesizing a victim's face and challenge announcing voice is smaller than even the duration of audible response. As depicted in the top of the Fig. 6, adversarial action time is limited with the beginning of the speech activity.

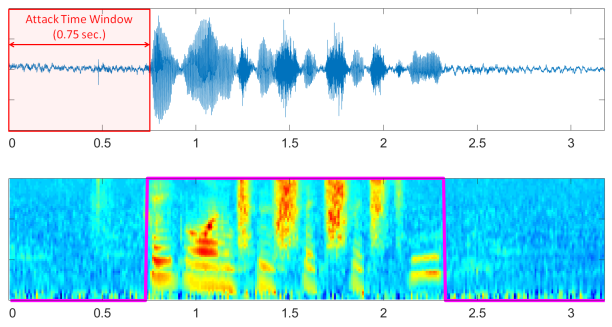


Fig. 6. Time window for adversarial actions.

Speech activity detection, also referred to as voice activity detection (VAD), is a ubiquitous method that has been studied and discussed in different contexts such as audio coding, content analysis and information retrieval, speech transmission, automatic segmentation and speech recognition, especially in the noisy environments [46], [47]. In our framework, we used a hybrid model that follows a data-driven approach by exploiting different speech-related characteristics such as spectral shape, spectro-temporal modulations, periodicity structure and long-term spectral variability profiles [48]. After getting different streams representing each of these profiles, the information from the streams is applied to the input layer of a Multilayer Perceptron classifier. The overall equal error rate of this approach is around 2% when a classifier is built with 30 hours data and tested on 300 hours data. Since our audio responses will be a few seconds, the error rate will be a few milliseconds. On the bottom of Fig. 6, we presented speech activity detection on the spectrogram of a sample Captcha response audio from our experiments.

Once user response time has been extracted, if it is within an expected human response time and not-longer than the breaking time of the corresponding Captcha scheme, we verify the challenge response as a genuine attempt. The reference response time window could be adapted for each user and Captcha scheme with his/her response times from the successful attempts since Captcha reading behavior could vary for each person and scheme.

D. Facial and Voice Verification

After getting a correct Captcha response within a real human response time, we verify the user's facial and voice data by using data from the registration phase. If the attempt is a new user registration, we again make facial and speaker recognition to check that the new user is not a duplicate one. Facial and speaker recognition and verification literature could be investigated under two different categories; first one is feature or descriptor-based (old fashioned) and the second one is a data-driven DNN-based (modern) approach.

[13]https://azure.microsoft.com/en-us/services/cognitive-services/speech/
[14]https://www.ibm.com/watson/services/speech-to-text/

In our experiments, we used cognitive services of Face++ and Microsoft to verify a user's face and voice, respectively.

VI. Evaluation

In this section, we present the results of our evaluation on *rt*Captcha to show that it provides a strong, yet usable, liveness detection to protect facial/voice-based authentication systems against compromising attacks. In particular, we have performed a user study to measure:

- the time difference between a real user solving the Captcha presented by *rt*Captcha versus the time it takes for an algorithm to break it
- the usability and user acceptance of *rt*Captcha

Note that our user study has been approved by the Institutional Reviews Board of our institution.

A. User Study Procedure and Data Collection

We implemented an Android app to experiment with five different challenge response-based liveness detections, where the user either has to read numbers or text presented on the screen, or perform an action in front of the screen. All text-based challenges will have the user read a number of phrases comprised of two to three simple words, and numeric challenges of 6-digit numbers.

It is worth noting that users pronounced all of the numeric or phrase challenges (in plain text or Captcha forms) out loud in our experiments.

To be more specific, our five tested liveness detection based upon the following challenges: ❶ two text phrase and one numeric challenges as plaintext; ❷ three numeric challenges as Captcha images with reCaptcha, Ebay and Yandex schemes; ❸ three text phrase challenges in an animated Captcha images with reCaptcha scheme. In this task, we display challenge words individually by animating (e.g., sliding from left to right) them sequentially with small time delays. The idea behind this approach is to prevent the attacker from extracting the challenge from one single frame, and instead force him/her to extract the Captcha as moving targets. On the other hand, we believe understanding an animated Captcha should be not too much more difficult than solving one at a fixed location for a human being. For this part of our experiment, we used Captcha samples collected by Gao et al. [4] for Ebay and Yandex schemes. To obtain reCaptcha samples that are either purely numerical or purely text (which are not included in the dataset from [4]), we generated them using *Cool PHP Captcha* tool which creates custom word Captchas in reCaptcha scheme; ❹ challenge to blink, and ❺ challenge to smile.

To improve the usability of our liveness detection, for challenges ❶ to ❸, our app will only present one challenge at a time, and we used CMU Pocketsphinx library for real-time speech recognition on mobile devices to know when the user has finished attempting the current challenge (by noticing the stop of utterance), show them whether they're successful before moving on to the next challenge phrase or number. Similarly, for challenges ❹ and ❺, we used Google's Mobile Vision API to obtain smiling and blinking probability to determine when the user has answered our challenge. Fig. 7 shows sample screen shots from our Android app while conducting Captcha challenges and blink detection.

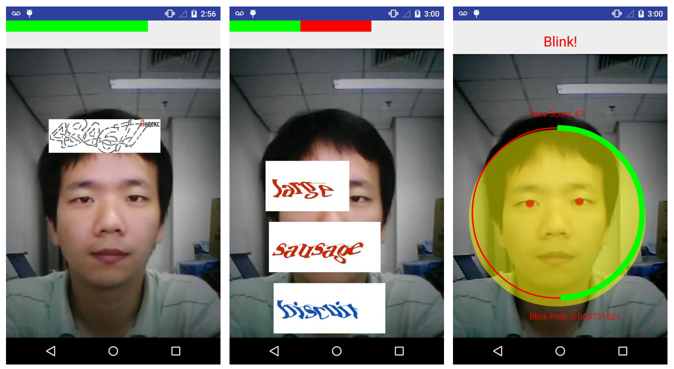


Fig. 7. Screen shots from Android app for user study. From left to right; tasks ❷, ❸ and ❹, respectively.

We recruited 31 volunteers for our experiments and had them use our Android app, which has installed on a LG Nexus 4 device we own. At the beginning of our experiment with each participant, we explained the purpose of our experiment and showed them an introductory video about how to use the Android app to complete the tasks. Then we asked each participant to answer 3 rounds of challenges for each of the 5 different kinds of challenges listed above (i.e., 15 challenges in total). For each challenge, we set a timeout of 10 seconds and considered it a failure and moved on to the next if the participant did not answer the challenge in that time. For the first three types of challenges, we captured the user's audio responses and some facial frames while answering the challenges (like we did in *rt*Captcha), as well as how long it took to answer the challenge and whether the answer was correct. We also compared the facial and voice data from different challenges to determine if it's the face and voice of the same user. For the fourth and fifth challenge types, we only measured and saved blink and smile detection time along with their probability without capturing any video/audio data.

B. Findings

Before delving into the details of the results from the aforementioned experiment, it is worth noting that participants correctly announced the Captcha challenges with an **89.2%** overall accuracy and **0.93 seconds** overall response time. The accuracy is much higher and the response time is excessively smaller than state-of-the art Captcha breaking algorithms (detailed in further sections). Moreover, 100% of participants' faces and voices are verified correctly with 93.8% (by Face++) and High (by Microsoft) overall confidence values, respectively.

Figure 8 presents the response time distributions of the participants. While response (and detection) time to any type of challenge that involves the user reading something are below two seconds, the minimum time to give a smile or blink response is higher than the largest measured response

TABLE III
RESPONSE TIMES AND SUCCESSFUL RECOGNITIONS OF THE CHALLENGES WITH OUR APPROACH (HUMAN_{aud}), MEN-POWERED CAPTCHA SOLVING SERVICE (ATTACK_{typ}), OCR BASED (ATTACK_{ocr}) AND STATE-OF-THE ART CAPTCHA BREAKING ALGORITHMS (ATTACK_{best}) [4].

Task #	Captcha Scheme	Time (secs)				Recognition Accuracy (%)			
		Human_{aud}	Attack_{typ}	Attack_{ocr}	Attack_{best}	Human_{aud}	Attack_{typ}	Attack_{ocr}	Attack_{best}
1	Plaintext	**0.77**	N/A	N/A	N/A	**91.9**	N/A	N/A	N/A
2	reCaptcha_{num}	**0.90**	22.11	2.98	10.27	**87.1**	96.7	0	77.2
2	Ebay_{num}	**0.73**	12.33	2.79	05.98	**94.1**	100	0	58.8
2	Yandex_{num}	**0.89**	15.05	3.30	15.50	**87.7**	96.7	0	02.2
3	$\text{reCaptcha}_{phrase}$	**1.02**	20.88	3.03	N/A	**88.0**	91.5	0	N/A

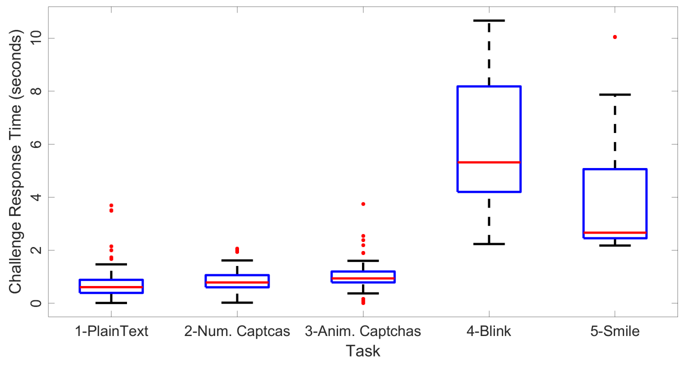


Fig. 8. Distribution of challenge response times for each tasks.

time to any of the Captcha challenges (task 2 and 3). The slow detection time for blink and smile may be due to the limitation of our implementation, but we believe they generally require every frame to be analyzed and thus can be more difficult than detecting the utterance of the answer to a text or numerical Captcha challenge. In other words, our experimental results show that Captcha based liveness detection challenges are not going to increase the end-to-end time to authenticate a user over existing smile or blink based challenges.

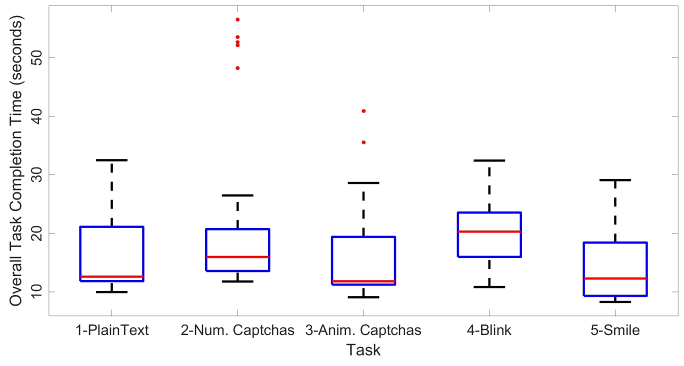


Fig. 9. Distribution of overall completion times for each tasks.

Finally, Fig. 9 shows the overall time to answer all 15 challenges, and we see that there's no significant difference between participants.

In Table III, the left most columns (Human_{aud}) give the average response times and recognition accuracies of our participants for each Captcha scheme in challenge type 1 to 3. Also, Fig. 10 presents distribution of them for each challenge in tasks 1 to 3. Our results show that participants' response times remain mostly constant over the different types of Captcha schemes tested, and are not affected by the difficulty level of the Captcha. Similarly, recognition accuracies from plain-text and Ebay Captcha challenges to reCaptcha and Yandex Captchas vary only slightly. Moreover, while numeric Captchas have consistently better accuracies than English phrase-based Captchas, the difference is below 5%.

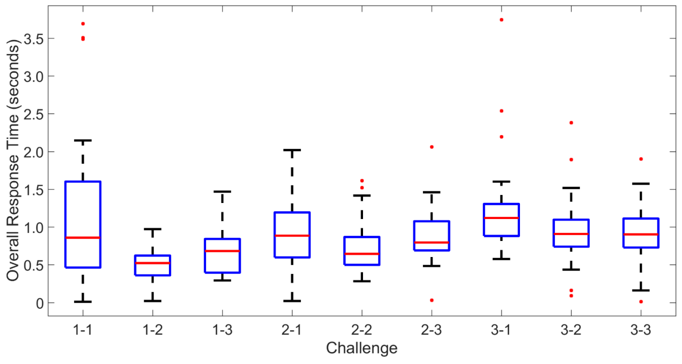


Fig. 10. Distribution of response times for each challenges in each tasks. (1_1 and 1_2: Plaintext Phrases, 1_3: Plaintext numbers, 2_1: reCaptcha numbers, 2_2: Ebay numbers 2_3: Yandex numbers, 3_1, 3_2 and 3_3: Animated reCaptcha phrases)

Fig. 10 also shows that there is a slight warm-up effect at the beginning of each tasks; the response times for the first challenge of each task is longer than that for the others. Moreover, when we change the challenge type (e.g. from phrase to numeric at task 1) or the Captcha scheme (e.g. from Ebay to Yandex at task 2), we also observe a slight warm-up effect. On the other hand, since we did not change challenge type or Captcha scheme on task 3, response times decreased in each trial at this task. In any case, even if we have a warm-up effect, the maximum response time to the Captcha challenge is 3.74 seconds, which is still below the execution time of current Captcha-breaking algorithms.

Finally, when a user fails to correctly answer any kind of liveness detection challenge, he/she will be asked to try again. So, in Table IV, we measure how many times our participant had to re-try before a successful authentication could be completed under the different types of challenges. Our results show that in almost all cases, participants needed to

TABLE IV
AUTHENTICATION ACCURACIES BY NUMBER OF TRIALS(%)

Trial #	Task-1	Task-2	Task-3	Task-4	Task-5
1	90.3	87.1	90.3	80.7	90.3
2/3	100	100	96.8	100	100

try at most two times to successfully respond to any challenge. The only exception happened for one participant under the animated Captcha challenge. When we manually inspected his response, we realized that the problem was caused by the speech recognition algorithm.

C. Usability Evaluation

We measured the usability and user acceptance rate by asking subjective questions at the end of our user study. Each participant faced two to four questions depending on their answers. We asked the following questions with the stated multiple choices:

1) Have you ever interacted with any kind of facial authentication systems? (Y/N)
2) If yes, what was the challenge it asked? (Blink/Smile/Other)
3) Would you consider using Real Time Captcha in the future? (Y/N)
4) If no, why?
   - I don't like Captchas.
   - I don't like voice recognition systems.
   - I prefer using password protection.

Overall, 87.1% of the participants never have used any kind of facial authentication system and 81.5% of them lean toward an authentication system that offers the proposed liveness detection scheme. The rest do not want to use our framework because of the voice recognition component or Captcha scheme. Finally, 12.9% of the participants have used smile or blink detection-based facial authentication systems, and they stated that they prefer *rt*Captcha. Even though an 84% favorable response from our participants shows promise, we consider it only a preliminary result. As future work, we plan to perform another user study to establish the usability of our system in the general population. This is because: 1) our current user study has a small number of participants, and 2) the participants are from limited diversity of age and background (mostly university students who are familiar with Captchas).

VII. SECURITY ANALYSIS

In this section, we will first present our analysis to determine how likely it is for an attacker to successfully evade *rt*Captcha and impersonate the user. As mentioned in the threat model, we assume the attacker can compromise the victim phone's kernel, and can have his/her malicious version of the client app used for authenticating with *rt*Captcha. Furthermore, the attacker can also use the victim's phone camera and microphone to collect face and voice samples of the victim, and use available techniques to build accurate models for the victim's face and sound. Thus, when *rt*Captcha presents the attacker with a Captcha, his/her main obstacle in achieving successful authentication is to *solve the Captcha before the authentication session times out*. Once the Captcha is solved, the created facial/voice model of the victim can be used to create video/audio of the victim saying the answer to the Captcha. This fabricated answer can be sent to our authentication server either by injecting it into the system as outputs from the camera and the microphone (through a compromised kernel) or directly into a malicious version of the client app.

Since our system measures the time between when the Captcha is first presented to the time when the user starts to speak, one possible attack against our system is for the attacker to produce an arbitrary "filler" sound (e.g., "errrr") while trying to automatically solve the challenge and then inject the synthesized video. This attack can lead to one of the following scenarios: 1) the "errrr" part is detected by the speech-to-text library, and results in the attacker giving the wrong response, or 2) the "errrr" part is ignored by the speech to text (in this case, we can modify our system to ignore the beginning of an utterance, but instead the beginning of speech recognized by the speech-to-text). Another potential attack against use of the start of speech is for the attacker to focus their effort in identifying/solving the first part of the Captcha. However, we will argue that a main challenge in solving Captcha is to break up the different characters and digits, and thus this attack may not buy the attacker much time.

One key to considering the attacker's chance of success is the threshold for session time out; let's call it Th_{legit}. To put it another way, the strength of *rt*Captcha depends on the difference between a Th_{legit} threshold that's long enough for legitimate human users to have good success rate at authentication, versus a Th_{legit} threshold that allows for accurately breaking Captcha using a Captcha-breaking algorithm. Thus, in the following, we will refer back to our experiments in Sect. VI.

A. Setting $Th_{legit} = 5sec$

Participants in our user study responded to 98.57% of the challenges in less than 3 seconds. Furthermore, our results in Sect. VI also show the studied users have an overall accuracy of 87.1% for all tested Captcha schemes, and there seems to be no correlation between their response time and their success rate. In other words, we will not see any significant improvement in the user's rate of successfully answering the Captcha even if we set Th_{legit} significantly higher. Thus, for the rest of our discussion, we'll assume Th_{legit} to be 5 seconds.

B. Automated Attacks under $Th_{legit} = 5sec$

Now let's consider what is the attacker's chance of breaking our Captcha and successfully generating the video/audio of a victim answering the Captcha with a session time out of 5 seconds. We will base our discussion on different kinds of Captcha breaking methods with different levels of sophistication.

The most primitive Captcha breaking method we have tested is Optical Character Recognition (OCR) based. In particular, we tested the Captcha used in our study against one of the OCR-based Captcha solving websites[15]. As presented in the Attack$_{ocr}$ columns of Table III, the tested site could not solve any of our Captcha challenges. Later on, we investigated if it can successfully decode anything but plain-text lookalike Captcha images without background noise or distortions.

We've also experimented with state-of-the-art Captcha breaking schemes from Gao et al. [4] and Bursztein et al. [49], which are based on character segmentation and Reinforcement learning (RL) respectively. Table-VI summarizes their best decoding accuracy and solving times for various schemes on commodity laptops. We consider the work in [4] to be state-of-the-art because it proposes the most generic solution and is the only published work that defeats the Yandex scheme. In Table III we referred to their system as Attack$_{best}$. While the results in Table-VI show that some Captcha schemes can be broken in approximately 3 seconds, the overall recognition accuracies can be very low (while the corresponding accuracies for harder schemes in our user study remain above 85%). Thus, we believe that setting Th_{legit} at 5 seconds gives us a very good safety margin against compromising attacks that could employ even the most advanced Captcha-breaking scheme.

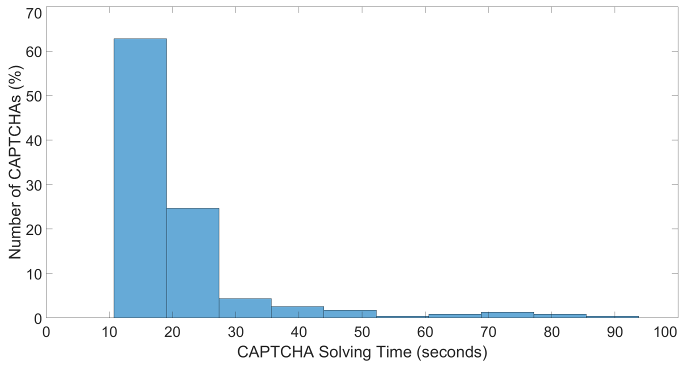


Fig. 11. Distribution of response times of *2captcha* service for our Captcha database.

C. Semi-Automated Attacks

Although in our threat model we stated that any attack that requires human intervention is not going to scale and thus is out of scope, we still will consider the possibility of breaking *rt*Captcha using cloud-based, manual Captcha solving services, since this is a commonly used attack method against other Captcha schemes. In particular, attackers may use the authentication device as a proxy and ship Captcha solving tasks to the real human workers. There are many man-powered Captcha-solving services that report high recognition rates, as presented in Table-V. We obtained recognition times and accuracies, as advertised on the official website of each service in Table-V. While the advertised times are much higher than our 5 seconds Th_{legit} threshold, we also used *2captcha.com* to break the Captcha dataset we used in the user study to obtain real numbers for a fair comparison. The average response times and decoding accuracies of this service under each scheme are presented in Table-III under the Attack$_{typ}$ columns, while the distribution of response times are presented in Fig.11. The average solving time is 19.17 seconds (with 10.75 seconds at minimum) with 96.2% overall solving rate. As such, once again, an attacker trying to launch compromising attacks based on these services will not be able to beat the 5 second threshold, and that is true even if we do not consider other time overheads caused by synthesizer, which has T_{tts}=1.1 seconds (TTS delay time) at best [37], etc.

[15]http://www.captchatronix.com

TABLE V
REPORTED AVERAGE DECODING ACCURACY AND TIME OF TYPING BASED HUMAN RESPONSES TO CAPTCHA CHALLENGES

Service	Acc(%)	Time(s)	Service	Acc(%)	Time(s)
anti-captcha	99.0	7	2captcha	96.6	10
captchaboss	99.9	8	imagetyperz	99.0	12
deathbycaptcha	95.8	10	9kw.eu	N/A	30

TABLE VI
BEST DECODING ACCURACY AND TIME OF GENERIC ATTACKS

	Gao et al. [4]		Bursztein et al. [49]	
Scheme	Acc(%)	Time(s)	Acc(%)	Time(s)
reCAPTCHA(Old)	7.8	8.06	21.74	7.16
reCAPTCHA	77.2	10.27	19.22	4.59
Yahoo!	5	28.56	3.67	7.95
Baidu	44.2	2.81	54.38	1.9
Wikipedia	23.8	3.74	28.29	N/A
QQ	56	4.95	N/A	N/A
Microsoft	16.2	12.59	N/A	N/A
Amazon	25.8	13.18	N/A	N/A
Taobao	23.4	4.64	N/A	N/A
Sina	9.4	4.83	N/A	N/A
Ebay	58.8	5.98	47.92	2.31
Yandex	2.2	15.5	N/A	N/A

D. Other Security Benefits

While the main strength of *rt*Captcha lies in presenting the attacker with a challenge that is difficult to answer automatically, and thus nullifying the advantage they have in being able to generate authentic-looking/sounding video/voice of the victim and inject it into the authentication process at will, *rt*Captcha also comes with a surprising benefit over other liveness detection challenges like blinking and smiling. That is, it is very difficult to capture the user giving out the answer to the right Captcha "just by accident." In particular, liveness challenges that are based on blinking and smiling are very vulnerable to attacks like UI redressing attacks [25], or more advanced attacks like those described in [50]. Under both scenarios, the attacker can drive the legitimate authentication app to a state where it's presenting the user with its liveness detection (either by using Intent, which is harder to control for more than one UI, or using the accessibility service), while covering up the phone's display with an overlay (so the user doesn't know he/she is being attacked). With liveness challenge based on blinking or smiling, this attack is likely to be successful because people naturally blink and smile occasionally, and thus they will provide the answer to the underlying challenge and help the attacker to authenticate them unknowingly. With *rt*Captcha, such an overlay-based attack is unlikely to be successful because it is very unlikely the victim will spell out the answer to the right Captcha by accident while the overlay is obscuring the screen and the underlying app is waiting for a response.

VIII. Discussion and Future Works

Reaching a big diversity in our user study was our limitation as in the previous studies [30]. Since we mainly recruited people around the university, these users were usually more familiar with the underlying technology with *rt*Captcha. Hence, human performance may vary among other populations, especially among those who rarely use mobile devices or are unfamiliar with Captchas, as stated by Brodic et al. [43].

One of the main security infrastructures in our framework relies on speech recognition since we capture audio response to the Captcha challenges. Hence, the STT algorithm must be robust enough to minimize the false negatives for legitimate user responses. The collected data in our user study involves ambient office, restaurant and outside environments with A/C sound, hums and buzzes, crowd and light traffic sounds. However, our data still have limited background noise variations to test the robustness of used STT method in our experiments. Having said that, we can always use other powerful STT approaches such as Deep Speech 2 by Baidu [45] or cloud-based solutions (instead of CMU Pocketsphinx library) for noisy environments. Moreover, recent advances in lip reading (e.g. LipNet [51]) provide around 95.2% of sentence-level speech recognition accuracy by only using visual content. Combining such an approach with STT would probably give very accurate results to legitimate challenge responses. Moreover, using lip reading-based speech recognition also will increase the usability of the system in a silent environment.

Our future work includes implementing a lip reading method on the Captcha response recognition. Furthermore, future research is required to analyze more data collected from different populations and environments with varying noise types and levels. It also could be required to analyze varying illumination and pose to measure face recognition and verification performance in a real life scenario. However, most of these limitations are related with all audio/visual authentication systems.

Even though we consider presentation attacks out of scope for this work, we believe that by requiring the user to actually say something in order to authenticate, we will create extra challenges for even state-of-the-art presentation attacks. In particular, no matter whether the presentation attack employs static pictures or wearable masks, *the attacker will have difficulty in using those materials to present genuine muscle movement*. This is obviously true for static pictures, but even for wearable masks, where it is doubtful that one can move lips (without exposing the lips of the person wearing the mask underneath) well enough to appear to be saying the answer to our Captcha challenge. Thus, on top of incorporating lip reading into our system, in the future we also plan to evaluate how this will stop attacks from wearable masks (which should be the state-of-the-art for presentation attacks).

Finally, one can argue that the recently announced Face ID[16] by Apple already provides a robust security mechanism against our threat model, and also, wearable masks. However, our approach still can be applicable through the existing smart phones (estimated around 2.3 billion by the end of 2017[17]) without the requirement of any additional hardware (e.g., depth camera, infrared sensors, etc.).

IX. Conclusions

Our work outlines several aspects of an audio/visual authentication system and presents a novel and practical approach, called *rt*Captcha to straighten the flaws of existing liveness detection systems. First, our exhaustive analysis on major cloud-based cognitive services (which have a market size of \$15 billion[18]) clearly reveals that an applicable and spoof-resistant liveness detection approach is an urgent need. On the other hand, Captcha-based human authentication has been used successfully on the web for more than a decade. Therefore, *rt*Captcha is a suitable fit for this urgent and growing need. Additionally, our user study and comparative threat analysis with its results proves that our scheme constitutes a strong basis against even the most scalable attacks involving the latest audio/visual synthesizers and state-of-the art Captcha-breaking algorithms.

Acknowledgment

The authors would like to thank the anonymous reviewers for their constructive comments and feedback, as well as Tara La Bouff for her proofreading efforts. This research was supported by the ONR under grants N0001409-1-1042, N00014-15-1-2162 and N00014-17-1-2895, and by the DARPA Transparent Computing program under contract DARPA-15-15-TC-FP-006. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the ONR or DARPA.

References

[1] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2387–2395.

[2] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1701–1708.

[16]https://www.apple.com/iphone-x/

[17]https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

[18]http://www.biometricupdate.com/research

[3] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 815–823.

[4] H. Gao, J. Yan, F. Cao, Z. Zhang, L. Lei, M. Tang, P. Zhang, X. Zhou, X. Wang, and J. Li, "A simple generic attack on text captchas," in *NDSS*, 2016.

[5] K. Li, F. Xu, J. Wang, Q. Dai, and Y. Liu, "A data-driven approach for facial expression synthesis in video," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 57–64.

[6] N. M. Duc and B. Q. Minh, "Your face is not your password face authentication bypassing lenovo–asus–toshiba," *Black Hat Briefings*, 2009.

[7] Y. Li, K. Xu, Q. Yan, Y. Li, and R. H. Deng, "Understanding osn-based facial disclosure against face authentication systems," in *Proceedings of the 9th ACM symposium on Information, computer and communications security*. ACM, 2014, pp. 413–424.

[8] G. Chetty and M. Wagner, "Multi-level liveness verification for face-voice biometric authentication," in *Biometric Consortium Conference, 2006 Biometrics Symposium: Special Session on Research at the*. IEEE, 2006, pp. 1–6.

[9] G. Pan, L. Sun, Z. Wu, and S. Lao, "Eyeblink-based anti-spoofing in face recognition from a generic webcamera," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE, 2007, pp. 1–8.

[10] I. Muslukhov. (2017) Breaking liveness detection on Android (4.1.1). [Online]. Available: https://www.youtube.com/watch?v=zYxphDK6s3I

[11] J. Bai, T.-T. Ng, X. Gao, and Y.-Q. Shi, "Is physics-based liveness detection truly possible with a single image?" in *Circuits and systems (ISCAS), Proceedings of 2010 IEEE international symposium on*. IEEE, 2010, pp. 3425–3428.

[12] X. Tan, Y. Li, J. Liu, and L. Jiang, "Face liveness detection from a single image with sparse low rank bilinear discriminative model," *Computer Vision–ECCV 2010*, pp. 504–517, 2010.

[13] J. Määttä, A. Hadid, and M. Pietikäinen, "Face spoofing detection from single images using micro-texture analysis," in *Biometrics (IJCB), 2011 international joint conference on*. IEEE, 2011, pp. 1–7.

[14] A. da Silva Pinto, H. Pedrini, W. Schwartz, and A. Rocha, "Video-based face spoofing detection through visual rhythm analysis," in *Graphics, Patterns and Images (SIBGRAPI), 2012 25th SIBGRAPI Conference on*. IEEE, 2012, pp. 221–228.

[15] S. Kim, S. Yu, K. Kim, Y. Ban, and S. Lee, "Face liveness detection using variable focusing," in *Biometrics (ICB), 2013 International Conference on*. IEEE, 2013, pp. 1–6.

[16] Z. Boulkenafet, J. Komulainen, and A. Hadid, "Face spoofing detection using colour texture analysis," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 8, pp. 1818–1830, 2016.

[17] N. Erdogmus and S. Marcel, "Spoofing in 2d face recognition with 3d masks and anti-spoofing with kinect," in *Biometrics: Theory, Applications and Systems (BTAS), 2013 IEEE Sixth International Conference on*. IEEE, 2013, pp. 1–6.

[18] S. Liu, B. Yang, P. C. Yuen, and G. Zhao, "A 3d mask face anti-spoofing database with real world variations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 100–106.

[19] N. Kose and J.-L. Dugelay, "Reflectance analysis based countermeasure technique to detect face mask attacks," in *Digital Signal Processing (DSP), 2013 18th International Conference on*. IEEE, 2013, pp. 1–6.

[20] I. Manjani, S. Tariyal, M. Vatsa, R. Singh, and A. Majumdar, "Detecting silicone mask based presentation attack via deep dictionary learning," *IEEE Transactions on Information Forensics and Security*, 2017.

[21] M. M. Chakka, A. Anjos, S. Marcel, R. Tronci, D. Muntoni, G. Fadda, M. Pili, N. Sirena, G. Murgia, M. Ristori *et al.*, "Competition on counter measures to 2-d facial spoofing attacks," in *Biometrics (IJCB), 2011 International Joint Conference on*. IEEE, 2011, pp. 1–6.

[22] I. Chingovska, J. Yang, Z. Lei, D. Yi, S. Z. Li, O. Kahm, C. Glaser, N. Damer, A. Kuijper, A. Nouak *et al.*, "The 2nd competition on counter measures to 2d face spoofing attacks," in *Biometrics (ICB), 2013 International Conference on*. IEEE, 2013, pp. 1–6.

[23] I. J. C. on Biometrics. (2017) Competition on generalized face presentation attack detection in mobile authentication scenarios. [Online]. Available: https://sites.google.com/site/faceantispoofing/

[24] Y. Xu, T. Price, J.-M. Frahm, and F. Monrose, "Virtual u: Defeating face liveness detection by building virtual models from your public photos," in *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association, 2016, pp. 497–512.

[25] M. Niemietz and J. Schwenk, "Ui redressing attacks on android devices," *Black Hat Abu Dhabi*, 2012.

[26] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3d faces," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 1999, pp. 187–194.

[27] J. Booth, A. Roussos, S. Zafeiriou, A. Ponniah, and D. Dunaway, "A 3d morphable model learnt from 10,000 faces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5543–5552.

[28] P. Huber, G. Hu, R. Tena, P. Mortazavian, P. Koppen, W. Christmas, M. Ratsch, and J. Kittler, "A multiresolution 3d morphable face model and fitting framework," in *Proceedings of the 11th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2016.

[29] S. Saito, L. Wei, L. Hu, K. Nagano, and H. Li, "Photorealistic facial texture inference using deep neural networks," *arXiv preprint arXiv:1612.00523*, 2016.

[30] Y. Li, Y. Li, Q. Yan, H. Kong, and R. H. Deng, "Seeing your face is not enough: An inertial sensor-based liveness detection for face authentication," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 1558–1569.

[31] M. Kobayashi, T. Okabe, and Y. Sato, "Detecting forgery from static-scene video based on inconsistency in noise level functions," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 883–892, 2010.

[32] Y. Rao and J. Ni, "A deep learning approach to detection of splicing and copy-move forgeries in images," in *Information Forensics and Security (WIFS), 2016 IEEE International Workshop on*. IEEE, 2016, pp. 1–6.

[33] F. Peng and D.-l. Zhou, "Discriminating natural images and computer generated graphics based on the impact of cfa interpolation on the correlation of prnu," *Digital Investigation*, vol. 11, no. 2, pp. 111–119, 2014.

[34] F. Peng, D.-l. Zhou, M. Long, and X.-m. Sun, "Discrimination of natural images and computer generated graphics based on multi-fractal and regression analysis," *AEU-International Journal of Electronics and Communications*, vol. 71, pp. 72–81, 2017.

[35] R. Caldelli, I. Amerini, and A. Novi, "An analysis on attacker actions in fingerprint-copy attack in source camera identification," in *Information Forensics and Security (WIFS), 2011 IEEE International Workshop on*. IEEE, 2011, pp. 1–6.

[36] H. Gao, H. Liu, D. Yao, X. Liu, and U. Aickelin, "An audio captcha to distinguish humans from computers," in *Electronic Commerce and Security (ISECS), 2010 Third International Symposium on*. IEEE, 2010, pp. 265–269.

[37] Z. Wu and S. King, "Investigating gated recurrent networks for speech synthesis," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 5140–5144.

[38] Z. Wu, P. Swietojanski, C. Veaux, S. Renals, and S. King, "A study of speaker adaptation for dnn-based speech synthesis." in *INTERSPEECH*, 2015, pp. 879–883.

[39] S. Shirali-Shahreza, Y. Ganjali, and R. Balakrishnan, "Verifying human users in speech-based interactions." in *Interspeech*, 2011, pp. 1585–1588.

[40] Z. Zhang, J. Yan, S. Liu, Z. Lei, D. Yi, and S. Z. Li, "A face antispoofing database with diverse attacks," in *Biometrics (ICB), 2012 5th IAPR international conference on*. IEEE, 2012, pp. 26–31.

[41] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, "What makes tom hanks look like tom hanks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3952–3960.

[42] Z. Wu, T. Kinnunen, N. Evans, J. Yamagishi, C. Hanilçi, M. Sahidullah, and A. Sizov, "Asvspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge," *Training*, vol. 10, no. 15, p. 3750, 2015.

[43] D. Brodić, A. Amelio, and I. R. Draganov, "Response time analysis of text-based captcha by association rules," in *International Conference on Artificial Intelligence: Methodology, Systems, and Applications*. Springer, 2016, pp. 78–88.

[44] D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky, "Pocketsphinx: A free, real-time continuous speech

recognition system for hand-held devices," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 1. IEEE, 2006, pp. I–I.

[45] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," *arXiv preprint arXiv:1512.02595*, 2015.

[46] S. Mousazadeh and I. Cohen, "Voice activity detection in presence of transient noise using spectral clustering," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 6, pp. 1261–1271, 2013.

[47] E. Uzun and H. T. Sencar, "A preliminary examination technique for audio evidence to distinguish speech from non-speech using objective speech quality measures," *Speech Communication*, vol. 61, pp. 1–16, 2014.

[48] M. Van Segbroeck, A. Tsiartas, and S. Narayanan, "A robust frontend for vad: exploiting contextual, discriminative and spectral cues of human voice." in *INTERSPEECH*, 2013, pp. 704–708.

[49] E. Bursztein, J. Aigrain, A. Moscicki, and J. C. Mitchell, "The end is nigh: Generic solving of text-based captchas." in *WOOT*, 2014.

[50] Y. Fratantonio, C. Qian, S. P. Chung, and W. Lee, "Cloak and dagger: From two permissions to complete control of the ui feedback loop," in *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 2017, pp. 1041–1057.

[51] Y. M. Assael, B. Shillingford, S. Whiteson, and N. de Freitas, "Lipnet: Sentence-level lipreading," *arXiv preprint arXiv:1611.01599*, 2016.