Voice Presentation Attack Detection through Text-Converted Voice Command Analysis

Il-Youp Kwak
Samsung Research
Seoul, South Korea
ilyoup.kwak@samsung.com

Jun Ho Huh
Samsung Research
Seoul, South Korea
junho.huh@samsung.com

Seung Taek Han
Samsung Research
Seoul, South Korea
s.t.han@samsung.com

Iljoo Kim
Samsung Research
Seoul, South Korea
ij00.kim@samsung.com

Jiwon Yoon
Korea University
Seoul, South Korea
jiwon_yoon@korea.ac.kr

ABSTRACT

Voice assistants are quickly being upgraded to support advanced, security-critical commands such as unlocking devices, checking emails, and making payments. In this paper, we explore the feasibility of using users' text-converted voice command utterances as classification features to help identify users' genuine commands, and detect suspicious commands. To maintain high detection accuracy, our approach starts with a globally trained attack detection model (immediately available for new users), and gradually switches to a user-specific model tailored to the utterance patterns of a target user. To evaluate accuracy, we used a real-world voice assistant dataset consisting of about 34.6 million voice commands collected from 2.6 million users. Our evaluation results show that this approach is capable of achieving about 3.4% equal error rate (EER), detecting 95.7% of attacks when an optimal threshold value is used. As for those who frequently use security-critical (attack-like) commands, we still achieve EER below 5%.

CCS CONCEPTS

• **Security and privacy → Usability in security and privacy**; **Intrusion/anomaly detection and malware mitigation**.

KEYWORDS

Voice Command Analysis; Attack Detection; Voice Assistant Security

ACM Reference Format:
Il-Youp Kwak, Jun Ho Huh, Seung Taek Han, Iljoo Kim, and Jiwon Yoon. 2019. Voice Presentation Attack Detection through Text-Converted Voice Command Analysis. In *CHI Conference on Human Factors in Computing Systems Proceedings (CHI 2019), May 4–9, 2019, Glasgow, Scotland Uk.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3290605.3300828

1 INTRODUCTION

Voice assistant vendors (e.g., Apple's Siri, Amazon's Alexa, and Samsung's Bixby) have started to upgrade their solutions to support more advanced and useful commands – examples include sending messages, checking emails, making payments, and performing banking transactions – some of which may also have security and privacy implications. Such advanced commands make voice assistants an attractive target for attackers to exploit, and try to steal users' private information or perform unauthorized banking transactions.

To mitigate those threats, some voice assistants force users to first unlock their devices (e.g., using patterns or fingerprints) before submitting security-sensitive commands. However, this mandatory device unlock requirement sits uneasily with usability of voice assistants as users have to physically engage with their devices at least once in order to use voice assistants. Further, some devices like smart speakers do not have any physical input space for users to authenticate themselves.

As a more usable authentication method, voice biometric-based authentication methods have been proposed to *implicitly* check users' voices using trained (known) voice biometric features. Hence, users do not have to physically authenticate themselves. However, voice biometric based authentication methods achieve about 80–90% accuracy when there are background noises present [3, 17, 29, 31, 33, 37], and are often used with threshold values that reduce false rejection rates – compromising security as a result. Human mimicry attacks can be effective against them [14, 19, 26]. Such methods are also weak against voice synthesis attacks where attackers use deep learning techniques to train victims' voice biometric models using recorded voice samples, and generate new malicious voice commands [11]. To detect those voice presentation attacks, signal-processing based voice liveness detection solutions have been discussed recently [18]; but such solutions would also suffer from accuracy losses when there are environmental changes, and cannot guarantee detection performance against unseen conditions.

In this paper, we propose a novel way to identify users' genuine commands and detect suspicious commands based on "Text-conVerted VoICE command analysis" (Twice), and evaluate its feasibility using a large real-world voice assistant dataset collected over a two month period through a large IT company – comprising of about 34.6 million voice commands. We used text-converted voice command utterances (bag of words) and matched applications as the main classification features. We experimented with lightweight classification algorithms, and measured the average equal error rates (EER), detection time, and training time. To the best of our knowledge, we are the first group to consider analyzing voice command text utterances to detect voice presentation attacks.

Our key contributions are summarized below:

- Real-world voice assistant command analyses, showing that about 87.48% of the users are occasional users using less than 20 commands over a month.
- Voice presentation attack detection system design that initially works with a globally trained model (available immediately), and gradually switches to a more accurate user-tailored model with increase in the use of voice assistants.
- Evaluation of the feasibility of using text-converted utterances as features to detect anomalous use of security-critical commands, showing that the average EER is about 3.4%, and the detection accuracy measured with a month-period unseen data is 95.7%.
- Identification of security-critical voice commands that are being used on real-world voice assistants, including commands that can be used to access credit card information, and change security settings.

2 BACKGROUND AND THREAT MODEL

In this section we describe existing (commercialized) voice authentication solutions, and attack scenarios that can circumvent them. In doing so, we motivate the need for our text utterance analysis based attack detection solution.

Voice Assistant Authentication

The most widely adopted solution is voice biometric based authentication that uses raw waveforms, complex spectra features, and log-mel features to train and classify users [9, 21]. It does not require users to perform additional tasks or remember additional information to authenticate themselves. It is a continuous solution that can be used to verify users' every command. Google Assistant and Samsung Bixby are equipped with voice biometric based authentication solutions. However, existing algorithms [3, 17, 24, 37] achieve about 80–90% accuracy when there are ambient background noises. Hence, to maintain voice assistant usability, those solutions are used with threshold values that result in low false rejection rates with some compromises in false acceptance rates. This use of threshold values open doors for human mimicry attacks – where attackers use their own voices and try to mimic device owners' voices – to bypass authentication checks [19, 26]. Voice replay attacks (replaying device owners' recorded voices through a speaker) [8, 18, 32, 36] and voice synthesis attacks (new voice commands are generated using device owners' voice samples and trained models, and played through a speaker) [25, 27, 35] are also effective against voice biometric based authentication solutions.

Voice passwords are available on Samsung Bixby. Users are asked to choose and remember a secret word, and *say it* to authenticate themselves upon submitting sensitive or security-critical commands. If voice passwords are used in public places, however, people can easily hear and compromise users' secret words. An adversary could simply record users' voice passwords and replay them to bypass it. There are usability issues too, since users may be interrupted frequently to say their secret words.

Existing voice assistants often require users to unlock their mobile devices first (e.g., by entering their PINs or patterns) before processing and executing sensitive and privileged voice commands. To use Bixby on Galaxy phones, for example, most of the useful commands (e.g., calling someone or setting alarms) are currently restricted, requiring users to first unlock their devices – such security policy sits uneasily with future visions of how voice assistants should work and their usability benefits. In this paper, we assume that future mobile devices will be equipped with a voice biometric based authentication scheme, and allow users to seamlessly use voice assistants without having to physically touch their devices.

Our speech-to-text-converted utterance analysis based attack detection system, referred to as "Twice," will be more effective against voice synthesis attacks as it does not rely on voice biometric features. Voice password threats are not applicable since Twice is an *implicit* attack detection system.

Threat Model and Assumptions

The first attack scenario involves a piece of malware being installed on the victim's smart TV or PC at home, and subsequently exploiting a nearby smart speaker (e.g., Amazon Echo). Here, we assume that voice biometric authentication has not been activated on the smart speaker, or even if it is turned on, it is configured to minimize false rejection rates [3, 24], and would inevitably falsely accept some attack commands [1, 5]. This malware monitors background noise levels, and when it senses that there is no one at home, it plays a set of pre-recorded malicious voice commands to activate the smart speaker, and make purchases and request for money transfers (e.g., through applications like Venmo).

The second threat we consider in the same context is a hidden voice command attack [6, 7] that hardly alters the original sound (e.g., music) but still adds hidden malicious commands – such commands can be interpreted and executed by target devices. An attacker hides attack commands inside popular music files using the distortion techniques presented in [7], and uploads those malicious music files on an online streaming service such as Youtube. A victim, at home, plays those files through Youtube and listens to music. Those hidden malicious commands, inaudible to human ears, would be understood and executed by the smart speaker.

The third threat model we consider is a voice synthesis attack [16, 25, 27]. We assume that (1) voice biometric authentication system is activated on the victim's phone – this victim can use all voice assistant features on the phone without first unlocking the device, (2) an attacker knows the victim, and (3) the attacker has silently recorded sufficient length of the victim's voice samples. The attacker uses existing, easily accessible voice training tools like Google's Tacotron [30, 34] or Baidu's DeepVoice [2, 28], trains the victim's voice biometric models, and generates a series of new, malicious voice commands using that model.

The victim, while being acquainted with the attacker, leaves her phone unattended (but locked) on the table and goes to the toilet for a few minutes. The attacker exploits that moment to start playing the prepared attack voice script using the attacker's phone speaker to steal the victim's credit card information or request money transfers. Using this synthesis attack, the attacker has managed to execute severe commands in a very short period of time without having to brute-force the victim's screen lock pattern or PIN. We motivate the need for Twice based on those attacks, and demonstrate that Twice can be an effective *complementary* solution to make such attacks much more difficult to perform.

Latency and Resource Usage Requirements

Our conversations with several engineers at a large IT company revealed that there are latency and computational power usage requirements that must be considered upon deploying a machine learning-based service on voice assistant servers. This is because additional use of computational power and memory through continuous invocation of machine learning algorithms may incur unacceptable costs for businesses, and unwanted latency for processing voice commands. A single GPU may be expected to concurrently process 100 or more voice sessions (streaming commands), indicating that machine learning algorithms must be *light*, *simple*, and *fast*.

3 REAL-WORLD VOICE ASSISTANT COMMAND ANALYSIS

In this section, we describe the real-world dataset that was used to evaluate the Twice feasibility. We summarize key data characteristics that motivate the selection of classification features, and explain how the attack set was generated.

Dataset Description

The real-world dataset we used consists of English voice commands that were processed and logged by a large IT company's voice assistant service in November and December 2017, logging every English command submitted (and processed) from about 3 million users located in the United States. There are about 48 million voice commands in total, all submitted through the users' smartphones.

Each voice command data entry (record) consists of the following 6 attributes:

- *Device ID*: A unique ID associated with a smartphone used to submit a voice command;
- *Utterance*: Submitted voice command in textual (speech-to-text converted) format;
- *Time stamp*: Time of voice command received as recorded by the voice assistant server;
- *Command type*: Command type (e.g., open application or set alarm) categorized by the voice assistant after processing a voice command;
- *Matched application*: Target application that will be responsible for executing a response generated and returned after processing a voice command;
- *Processed state*: Boolean value indicating whether a submitted command was correctly interpreted and processed, and a response has been generated.

Device ID was used to select and group all commands belonging to the same user. Here, our assumption is that a smartphone (and the voice assistant application running on it) was used by a single user.

Data Pre-processing

First, we looked for noise in the dataset and removed (1) data entries that have no utterance information, and (2) data entries for which the processed state is "false" since such entries are not really supported by the voice assistant and are unlikely to be used again by users. After pre-processing, we were left with about 34.6 million voice commands from 2.6 million users. For the real-world voice command analysis (presented in this section), we only used the November 2017 dataset, consisting of about 17 million commands from 1.7 million users. The December dataset, consisting of about 17 million commands from 1.9 milliion users, was used as an unseen test set for accuracy evaluation.

Data Exploration

Voice Command Usage Frequencies. We counted the number of voice commands used by each user (grouped using Device ID), and ordered the users by the number of commands used in a decreasing order. Only 0.04% of the users (661) used 500 or more voice commands, 12.48% of the users (216,303) used between 20 to 499 voice commands, and 87.48% of the users (1,516,189) used less than 20 commands in a month. On average, about 50% of the users used four or less commands per month, indicating that most voice assistant users are occasional users, and voice assistants still seem to have limited impact in our lives. But we note that our analyses are based on just one voice assistant implementation, and any generalization of our results need to be performed with caution.

Popular Command Types. Table 1 shows the top 10 popularly used voice command types for occasional users who used less than 20 commands in a month, and for frequent users who used 500 or more commands. Among the occasional user group (87.48% of the users), the most popularly used command type was "Open [app-name]" (10.4%), which allows users to conveniently open an application without having to search and use touch screens. The second popularly used command type was "Call [person]" (9.0%), which allows users to quickly make phone calls without having to open and search contacts. "Set alarm" and "Take screen shot" were also popularly used.

Table 1: Top 10 voice command types used for occasional users and frequent users.

< 20 commands		≥ 500 commands	
Type	Percent	Type	Percent
Open [app-name]	10.4%	Open [app-name]	36.3%
Call [person]	9.0%	Close [app-name]	4.4%
Set alarm	5.9%	Close all apps	4.2%
Answer question	4.3%	Call [person]	3.2%
Take screen shot	3.9%	Go back	2.5%
Remind me [time]	3.1%	Answer question	2.4%
Turn on flash light	3.1%	Home screen	2.1%
Where is [location]	2.8%	Open message	2.0%
Turn off flash light	2.6%	Take screen shot	1.8%
Take a picture	1.7%	Search [keyword]	1.3%

We observed a slightly different trend for the more frequent users who used 500 or more commands. "Open [app-name]" was more dominantly used (36.3%), and commands for closing applications "Close [app-name]" (4.4%) and "Close all apps" (4.2%) were also popularly used. "Open message" (2.0%) and "Search [keyword]" (1.3%) were also popularly used among frequent users.

Popular Matched Applications. Next, we analyzed popularly used matched applications. The most frequently matched application (for executing voice assistant response) was "Action controller"[1] (28.97%) followed by "Contacts" (15.60%), "Clock" (9.91%) and "Messages" (5.25%).

To explore the possibility of using matched applications as classification features, we analyzed the most popularly used application combinations and their proportions for users who used 20 or more commands in a month. As shown in Table 2, the proportion of the most popularly used combinations (among all combinations detected) used were just 1.07% – there was no dramatic decrease in the proportions, indicating that the proportions are not significantly skewed toward small number of application combinations, and somewhat evenly distributed. Such characteristics would make applications another possible classification feature.

Table 2: Top 5 combinations of matched applications used by those who used 20 or more commands.

Matched application combinations	Percent
Action controller, Contacts	1.07%
Action controller, Contacts, Clock	1.05%
Action controller, Contacts, Clock, Settings	1.04%
Action controller, Contacts, Messages	0.83%
Action controller	0.81%

Attack Set Generation

To generate a global attack set to be used for attack detection model generation and accuracy evaluation, we first selected a set of security and privacy risks (related to smartphone privileges) from [12], which is a survey literature on smartphone users' concerns. There were 99 risks in total, including risks like "read your credit card in your wallet," "change your keylock/pattern/PIN," "posted to your Facebook wall," and "took a photo with your front-facing camera." We then identified keywords associated with each risk. For each keyword, we searched for synonyms, and tested them on the target voice assistant as a full command before including them; e.g., for the post-on-Facebook risk, we used the following search query: "[post `OR` update `OR` add `OR` share `OR` upload `OR` write `OR` compose `OR` print `OR` sign `OR` type `OR` note] `AND` facebook."

[1]Phone level control commands such as "Home screen,", "Open/Close [app-name]," "Scroll down," and "Go back"

We then searched for voice commands that match those keywords, using AND and OR conditions appropriately as shown above. We did not consider keyword sequences, and only considered full word matches. We removed all duplicates, and commands that would not lead to a particular threat (e.g., search commands that contain selected keywords), leaving us with a final set of 262,887 security-critical commands.

The largest risk category (counting attack commands) was "Set alarms" with 182,612 commands, the second largest was "Take screenshots" with 34,818 commands, followed by "Force quit all apps," and "Download data," and "Connect to a Bluetooth device" with 10,324, 7,972, and 5,665 commands, respectively. We show 5 example risk categories with high user concern levels (high severity) in Table 3, and three most popularly used commands for each category – we counted all command occurrences in November without removing duplicates. "Change voice password" was the most frequently used command from the "Change keylock/PIN/pattern" category, being used 1,686 times. From the "Read your credit card" category "Open credit card" and "Open Samsung Pay and show me the registered credit card" were popularly used. As for the "Mobile pay" category, "Open Samsung Pay" was the most popularly used command. Commands like "Pay with Bank of America card using iris" that would lead to actual payment transactions were also used by many users. From the "Post on Facebook" category, "Open Facebook and post a recent picture" was popularly used – this command could be used with the "Take a picture with the front camera" command ("Take a picture" category) to post a private photo of a user on Facebook. Status update commands like "Update my Facebook status to say *Where is the weed and send it*" were also popularly used. These real-world use of security-critical commands reinforce our motivations to improve voice assistant security, and demonstrate that attack scenarios described in Section "Threat Model and Assumptions" can be performed.

We do not claim that our attack set represents all possible attacks and risks associated with using smartphones or voice assistants (that is not the goal of our work); however, it does cover a sufficiently comprehensive set of attacks that users are concerned about, providing us with effectively large dataset (comprising of real-world, working commands) for evaluating the feasibility of our attack detection models.

Security-critical Command Definitions

A "security-critical command" is a command that could be used in a voice presentation attack to exploit one of the security threats mentioned in [12]. Our attack set (above) comprises of such security-critical commands selected from an existing set of real-world commands that are actually understood by the voice assistant. Here, our assumption is that all commands included in the attack set represent security-critical commands that are used by adversaries to achieve goals (lead to risks) mentioned in [12]. Our definition of an attack is an *anomalous use* of a security-critical command, and this is what Twice is trained to detect. By including the commands that were uttered by users in the attack set we also consider voice replay attacks (replaying victim's own utterances on victims' devices).

Table 3: Attack dataset comprising of 99 selected security risk categories. We selectively show 5 high-severity risk categories and top three most popularly used commands.

	Count
Category 1: Change keylock/PIN/pattern	5,201
Change voice password	1,686
Change password	441
Change unlock password	15
Category 2: Take a picture with front-facing camera	483
Take a picture with the front camera	118
Open front camera and take picture	45
Can you take a picture from the front camera	3
Category 3: Mobile pay	25,614
Open Samsung pay	14,259
Open Pay pal	1297
Open Venmo	497
Category 4: Read your credit card	962
Open credit card	56
Open Samsung pay and show me the registered credit card	13
Open Debit card	11
Category 5: Post on Facebook	3,646
Open Facebook and post a recent picture	97
Make a post on Facebook	58
Open Facebook and show post	35

Security-critical Command Usage Frequencies

Across all 1.7 million users, there were 415,831 users (23.99%) who have used at least one of the security-critical commands included in the November attack set (previous section). It is interesting to note that a noticeable proportion of current users submit commands that have potential security or privacy risks.

To study the proportions of security-critical commands being used, we selected users who have used at least one security-critical command and at least 20 commands in November, and computed the proportions (# security-critical commands / # all commands). There were just 29,979 users (1.73%) who fall under that category. 16,377 of such users (54.63%) used small proportions (0–10%) of security-critical commands. And 7,985 users (26.64%) were using noticeable proportions (more than 20%).

User Consents for Data Collection

We clarify that all the data we used for analyses were collected only with explicit consents from voice assistant users, acknowledging that their voice assistant interaction data and application usage data will be collected and may be used for data analytics and research purposes. We inquired and checked with a user privacy protection executive office, and received their confirmation that there are no legal and ethical issues in utilizing the collected data for research purposes so long as we do not present user identifiable information. We confirm that only the aggregated statistical results are presented in this paper, and it is not possible to identify users based on our results.

4 DESIGN

In this section we provide an overview of how Twice works, including the classification features and algorithms used.

System Overview

Twice is an AI-driven attack detection system that is integrated with existing voice assistant servers, utilizing their speech recognition and command processing services. Twice components are added to an existing voice assistant server architecture.

When a user says a command to a voice assistant application activated on her device, the voice assistant forwards that command and the device ID to the remote server through a secure channel. The voice assistant server uses the speech recognition service to generate a corresponding utterance, and feeds the text to the command processing engine, which in turn, adds the rest of the features described in Section "Dataset Description." It then processes the command, and generates a response to be sent back to the requesting device.

While it generates a response, it also sends the device ID and command text and extracted features to the Twice engine. The pre-trained attack detection classification model is selected based on the device ID – here, we assume one device is associated with one user (device owner). Based on the number of commands used cumulatively, the Twice engine chooses a more accurate classification (attack detection) model to use: the "global model" – trained with globally used commands – is *used by default* for new users; the "user-tailored model" – trained with users' own commands – is selected when the user has used sufficiently large number of commands (e.g., more than 500 commands), and the user-tailored model more accurately detects attacks. The selected model is used to compute the probability of the command being an attack. The Twice engine returns this attack probability result (e.g., 95% chance that the command is an attack, and has not originated from the user) to the command processing engine, which in turn, sends back the generated voice assistant response and attack probability result to the voice assistant application running on the user's device.

The voice assistant application compares the returned attack probability against a pre-configured authentication threshold value that is set based on the user's preferred voice assistant security level. If the attack probability value is equal or greater than the threshold value, the returned response is initially rejected. The user is then asked to explicitly enter her PIN to continue running the flagged, security-critical command. If the user enters the correct PIN, then the flagged command is marked as a genuine command, and is used to updated the user-tailored model.

Attack Detection Models

Feature Selection. Based on the data characteristics analyzed in Section "Data Exploration," we select the following features to be used in our classification models:

- *Bag of words (BoW) for users' commands and security-critical (attack-like) commands* [23]: Count of each word used in a set of voice commands previously used by a target user (user-tailored model), a global group of users (global model), and a global set of security-critical commands (attack set).
- *Matched application*: target applications are converted as "one hot encoding" features [15].

Binary Classification. We use a binary classification model to differentiate users own genuine commands from attack commands. Our classification feature matrix and binary classifier are generated through the following steps:

(1) From a global attack set generated, randomly sample a large number of voice commands to be used as an "attack dataset."
(2) Randomly sample a large number of users' voice commands to be used as a "global user dataset" for a global model (we selected 700 users), or choose a "target user dataset" for a user-tailored model.
(3) Convert utterance data into the BoW feature data (X_B). Use "attack dataset," and "global user dataset" for a global model and use "attack dataset," and "target user dataset" for a user-tailored model.
(4) Compute "one hot encoding" features [15] for selected matched applications (X_m).
(5) Define a ground-truth label vector, $\mathbf{y}$, where ith element of $\mathbf{y}$, y_i, is "1" if ith voice command is a security critical command, and "0" otherwise.
(6) Fit a binary classification model. Feature sets to be used are X_B and X_m, and the binary class ground-truth label variable is $\mathbf{y}$.

Training User Models. Twice is designed to create a tailored classification model for each user, and periodically update a model based on newly processed voice commands. Hence, to meet the latency requirements described in Section "Latency and Resource Usage Requirements," in addition to detection accuracy, we also need to consider computing resource usage overheads that would incur from continuously training and updating models. Being mindful of those requirements, we consider three lightweight classification algorithms, logistic regression, XGboost and SVM with appropriate regularization methods to prevent over-fitting problems.

Cost-sensitive Learning and Hyper Parameter Tuning. We used a cost-sensitive learning method [10] to train imbalanced datasets (e.g., for a user-tailored model, the majority of the train set commands may come from the attack dataset). We multiply the minority class (target user) with weight parameters, ensuring that the minority class also equally influences the classification model. The overall training latency will not change as it merely multiplies parameters in the loss function. We used the grid search method to select an optimal parameter set that would minimize the average EERs. The optimized set of hyper parameters include cost-sensitive learning parameters, regularization parameters, and classification algorithm specific parameters.

5 IMPLEMENTATION AND EVALUATION

This section presents the evaluation results, including the attack detection accuracy, detection time, and training time.

Software and Hardware Used

All of our classification models were implemented using Python on Ubuntu 16.04 (64-bit). The "sklearn" module was used to implement a logistic regression and SVM, and the "XGboost" module was used to implement XGboost. All of our experiments were executed on a single PC equipped with an Intel Core i7-6700 CPU (3.40GHz) with 4 cores, 8GB (2400MHz DDR4) memory, and 1TB (64MB Cache) 7200 RPM SATA 6Gb/s hard drive.

Setup

To evaluate the performance of the global model and user-tailored models with respect to the number of voice commands used, we divided the users into 6 groups as shown in Table 4. The first group includes the users who used 20–29 voice commands until the 26th day in the November dataset, the second group 30–49 commands, the third group 50–99 commands, and so on. Note, the November dataset was used for training models and computing EERs. The December dataset was used as "unseen test data" for measuring the attack detection accuracy using optimal threshold values (derived from EERs).

As for the attack set, the time and order (sequence) in which the attacks are performed is irrelevant; i.e., attack

Table 4: 6 user groups categorized by the number of commands used until the 26th day in November.

# commands	20–29	30–49	50–99	100–499	500–999	≥1,000
# users	71,813	53,296	29,265	9,679	339	195

commands can be used at any time in any order. Hence, We performed 5-fold cross validation (CV) to reduce any bias that might be introduced through randomly splitting training and testing sets. The number of attack commands selected for each of the 99 risk categories described in Section "Attack Set Generation" are vastly different. To maintain size balance across risk categories, we randomly selected 1,000 attack commands from each category and included them in the training/testing sets. For those categories that have less than 1,000 commands, we included all commands.

As for users' genuine (own) commands, we trained user-tailored classification models using the data from the first 26 days, and evaluated the detection accuracy and overheads using the commands submitted between the 27th and 30th (last) day. While creating these evaluation sets, we only selected "target users" (those to be evaluated) who used at least 6 commands in the four-day evaluation period (27–30th). As for the first four user groups who used less than 500 commands, we randomly selected 1,000 target users from each group for evaluation. As for the last two groups who used 500 or more commands, we used all users' data as target users to be included in the evaluation sets. To train the global (default) model, we randomly selected 700 users who used 20 or more commands in November, and used the commands they used during the first 26 days as the training set. To test the global model, we used the same target user groups described above.

To create user-tailored binary classification models, we combined each target user train set with 5 different attack train sets (from 5-fold CV) – effectively training 5 different models per target user (target user train set commands remained the same), and computed average EERs from the 5 cross validation attempts. As for creating the global binary classification model, we used the train set of 700 users, and combined their commands with 5 different attack sets to generate 5 different global models. We then tested 5 global models on each target user and computed the average EERs. In these experiments (see next section), the target user sets and attack sets – both the training and test sets – did not have any duplicating utterance entries.

To evaluate the performance of Twice against those who have used security-critical commands, we selected all users who have used 20 or more commands and at least one security-critical command in November. There were 415,831 users (23.99%) who belong to that category. We used the same techniques described above to train and evaluate the classification models. In these experiments, there were some duplicating utterance entries in the target user sets and attack sets, taking replay attack possibilities into consideration (see Section "Security-critical Command Users").

Attack Detection Accuracy and Overheads

The accuracy goal of Twice is to correctly identify a target user's own commands, and reject attack commands. To evaluate this for the three classification algorithms (logistic regression, XGboost, and SVM), we use EERs. The mis-classifying error rate for target user commands is referred to as "false rejection rate" (FRR), and the mis-classifying error rate for attack commands is referred to as "false acceptance rate" (FAR). Our binary classifiers return a probability score that represents the likelihood of a command being an attack, computing probability scores for every command in a given evaluation set. A value at which the FAR is equal to the FRR for a given command set determines the EER for a target user.

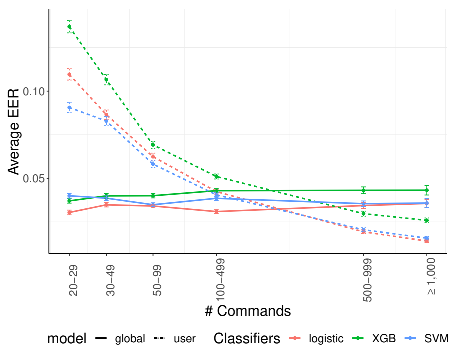

Figure 1: Average EER values across the 6 user groups tested with the four-day evaluation set.

Equal Error Rates. The average EER values are shown in Figure 1. As for the global model evaluation, logistic regression outperformed XGboost across all user groups with statistical significance (all $p < 0.01$, Wilcoxon signed-rank test with Bonferroni correction) but showed statistically significant superiority for just two user groups (those who used between 20–29 commands and 100–499 commands) against SVM (all $p < 0.01$, corrected Wilcoxon signed-rank test).

With the user-tailored models, logistic regression and SVM outperformed XGboost for all user groups (all $p < 0.01$, corrected Wilcoxon signed-rank test). Using Pearson's correlation, we found a negative correlation between the number of voice commands used and average EERs of all three user-tailored models (ρ = -0.23, $p < 0.01$) – demonstrating that the detection accuracy of the user-tailored models would improve with the increase in the use of the voice assistant.

Overall, the best performing global model (logistic regression) outperformed the user-tailored models for those who used less than 500 commands. However, all three user-tailored models outperformed the global model for those who used more than 500 commands (all $p < 0.01$, corrected Wilcoxon signed-rank test). To maintain consistently low EERs, Twice should start with the global model for new users (available immediately), and switch to the user-tailored models when users use about 500 commands.

Training Overheads and Detection Time. Table 5 shows the average time taken and average memory used for training each user-tailored model and the global model. It also shows the average attack detection (classification) time for classifying each command. To demonstrate the *worst case* overheads for the user-tailored models, we only considered the models generated for the 379 users who used 500 or more commands in November.

As for the user-tailored models, logistic regression and XGboost took 0.5 and 1.0 seconds on average for training, respectively. SVM was the slowest, taking 2.9 seconds on average. The training time, memory usage, and detection time data all passed Shapiro-Wilk normality test (all $p < 0.05$), and pairwise t-tests show statistically significant differences between the three algorithms with respect to training times (all $p < 0.01$, paired t-test with Bonferroni correction). We observed similar results for memory usage: SVM used about 15 MB more memory than the other two algorithms (all $p < 0.01$, corrected t-test). As for average detection times, we measured the time it took for a single command to be classified through a trained model. Logistic regression was the clear winner with about 0.11 milliseconds (all $p < 0.01$, corrected t-test). We observed similar trends for the global model.

Considering those EER and overhead results, logistic regression seems to be the obvious choice for training both the global model and user-tailored models. All our later analyses focus on logistic regression implementations.

Table 5: Average training time (seconds) and memory used (megabytes) to train models for those who used 500 or more commands. Average detection time (milliseconds) is also shown. Standard deviations are shown after each average value.

	Global model			User-tailored model		
	Logistic	XGboost	SVM	Logistic	XGboost	SVM
Training time (sec)	3.4 (0.13)	5.3 (0.07)	36.2 (22.35)	0.5 (0.01)	1.0 (0.01)	2.9 (1.66)
Memory usage (MB)	961.43 (0.05)	969.63 (0.62)	1071.13 (83.00)	821.67 (0.01)	823.89 (0.08)	837.03 (5.08)
Detection time (ms)	0.14 (0.04)	1.51 (0.18)	0.64 (0.06)	0.11 (0.02)	1.55 (0.18)	0.64 (0.06)

Accuracy Against Unseen Data. The binary classifiers generated from the previous evaluation, and the optimal threshold values determined from the EERs were used to measure the attack detection accuracy against the December dataset. We generated an "unseen test set," applying the same technique presented in Section "Attack Set Generation" to select 260,406 attack commands. We used the sampling technique described in Section "Setup" to maintain size balance across risk categories, selecting 17,092 commands to be included in the final test set. We used the same set of target users selected previously, and used all of their December commands sequentially to measure FRRs during a month period. Table 6 shows the FAR across 6 user groups. Similar to the EER results, the global model achieves low average FAR at 4.3% and performs better than the user-tailored models for the first three user groups; however, for those who have used 100 or more commands, the user-tailored models achieve lower FARs between 1.7–4.3%.

Table 6: Average FARs measured using the December attack set.

	Global model	User-tailored model					
# Commands		20–30	30–50	50–100	100–500	500–1,000	>1,000
FAR	4.3% (0.03)	10.1% (0.10)	8.34% (0.08)	5.9% (0.06)	4.3% (0.04)	2.3% (0.02)	1.7% (0.02)

To demonstrate FRR consistency over time, we show FRRs from day 1 to 31 in Figure 2. As for the occasional users, the user-tailored models showed inconsistent, increasing FRRs – this may be due to those occasional users using new commands that they have not used before. The average FRRs were much smaller and more stable for those who used 500 or more commands though. On the other hand, the global model showed more consistent FRR for all user groups, including those light users. Those results confirm that the global model should be used initially to maintain consistent FRRs below 10%, and eventually the user-tailored models would have to be used to achieve FRRs below 5%.

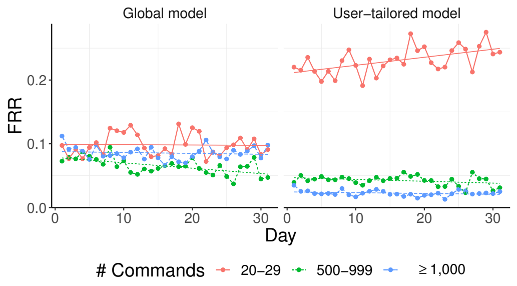

Figure 2: Average FRRs measured using the December target user commands, and plotted over a month period.

Security-critical Command Users

As described in Section "Setup," we experimented with 29,979 users who have previously used security-critical commands – dividing them into three groups based on the proportions of security-critical commands being used: "less than 10%," "10–20%," and "more than 20%" (see Section "Security-critical Command Usage Frequencies").

We measured the average EERs following the same settings used above. The results are shown in Figure 3. For all three graphs, the global model (with 5% or less EER) outperformed the user-tailored models for the first three user groups (all $p < 0.01$, corrected Wilcoxon signed-rank test). The user-tailored models started outperforming the global model for those who used more than 500 commands: from the first graph, we found statistically significant difference in the EERs (between the global model and user-tailored models) for the groups who used 500–999 commands and 1,000 or more commands (all $p < 0.01$, corrected Wilcoxon signed-rank test); from the second graph, only the 500–999 user group showed statistically significant difference – this is because the other heavy user groups have small sample sizes. By using the two models interchangeably, Twice will maintain average EERs below 5% even for those who frequently use security-critical commands.

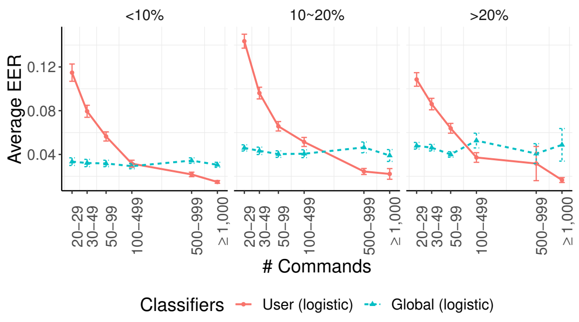

Figure 3: Average attack detection EERs for the three groups who used different proportions of security-critical commands.

6 DISCUSSIONS

Threat Model Mitigation

Results from Section "Accuracy Against Unseen Data" demonstrate that Twice is capable of detecting voice synthesis or hidden voice command attack scenarios (anomalous use of security-critical commands) with about 95% accuracy (TRR).

Integration with Existing Solutions

Twice uses classification features that are completely different to existing detection solutions that rely on signal-processing techniques [18, 27]. Hence, we surmise that Twice can be used as an effective complementary technology to further enhance user attack detection accuracy. As part of future work, we plan to evaluate the effectiveness of combining Twice features with features used in existing signal-processing based techniques.

User Experience Implications

With existing voice assistants, e.g., Bixby, users are required to say their voice passwords or first unlock their devices to submit any useful command. With the deployment of Twice, such use of explicit authentication could be minimized – requesting explicit authentication only when potentially malicious commands with security risks are detected. Further, Twice can be deployed as a complementary solution to improve the overall accuracy of existing voice security solutions, including voice biometric based authentication and signal-processing based voice liveness detection solutions (see Section "Related Work") that may suffer from high FRRs when used under varying environments. Deployment of Twice would reduce the overall FRR (less annoying for users), and improve the UX of voice security. Twice can be activated immediately for new users, and protect them from suspicious uses. FRRs would be consistent around 10% initially but would gradually fall with the use of more commands (see Figure 2). Once Twice switches to user-tailored models, FRRs would fall below 5%. Even for those who frequently use security-critical commands, we expect EERs below 5%. To further reduce user frustration, user-tailored models can be updated periodically whenever a user is asked to enter her password to authenticate a security-critical command – this would help Twice correctly classify the similar security-critical commands if they are frequently used.

Limitations

Twice cannot detect replay attacks that simply replay recorded commands. Hence, Twice would have to be deployed as a complementary solution to existing replay attack detection solutions. Having said that, we argue that such naive replay attacks are likely to have limited security impact: dedicated attackers would try to combine replay attacks with voice synthesis attacks to increase the attack severity (e.g., change a user's recorded banking command to send money to a rogue recipient). We surmise that Twice would be moderately effective in detecting this type of hybrid attack as the words assigned with high importance weightings in the attack features would likely remain (to maintain original attack intention), and other words with high importance are likely to be added to increase attack severity.

Another limitation is that our attack sets have been generated synthetically and only cover risks discussed in [12] – real-world attack commands may have different characteristics. If real-world attack sets become available in the future, it would be necessary to evaluate Twice against such sets.

7 RELATED WORK

Several techniques have been proposed to detect voice replay or synthesis attacks replayed through loudspeakers. Feng et al. [13] and Liu et al. [22] propose the use of wearable devices, such as eyeglasses, earbuds, or necklaces, to detect voice liveness. Their approaches check for the presence of users' physical body and muscle movements to verify that devices owners have indeed submitted voice commands being processed. They achieved about 97% accuracy in detecting voice liveness but rely on the use of earbuds, which are additional hardware that users would have to buy, carry, and use. Zhang et al. [36] monitor unique articulatory gestures using sound wave reflection techniques to check voice liveness. They achieved about 99% accuracy but rely on users physically holding their devices near their ears.

Recently, an extensive study was conducted to analyze the performances of machine learning-based human liveness detection techniques proposed as part of the "2017 ASVspoof competition" [18]. According to the study findings, the EERs varied from 6.73% to 45.55%. The best-performing approach combined three deep learning models and one SVM model to achieve high accuracy [20]. Logan et al. [4] use sub-bass over excitation and low frequency signal features to detect electronic speakers, achieving 100% TAR and 1.72% FRR in quiet locations. Just like any other voice biometric based technologies, however, these approaches would all face significant accuracy losses when there are background noise variations and environmental changes. Further, combining multiple complex models and features require heavy computational resources that sit uneasily with the latency requirements described in Section "Latency and Resource Usage Requirements." Being mindful of such constraints, Twice was designed to use small number of lightweight features, and guarantee fast training and classification times as well as model simplicity. Twice is the only solution known to date that analyzes the text utterances to detect voice presentation attacks.

8 CONCLUSIONS

The combined use of the global model and user-tailored models are integral in maintaining low and consistent EERs at around 3.4% for all users. To maximize detection accuracy and minimize false rejections, we recommend a switch (from global to user-tailored) when users have used up to about 500 commands. Even for those users who frequently use security-critical commands, Twice is capable of achieving EERs below 5%.

Training a user-tailored model only takes about 0.5 seconds in the worst case scenario, and attack detection decisions can be made within 0.11 milliseconds. Those results indicate that Twice is a highly efficient and accurate solution that could be used as a complementary solution to significantly improve voice attack detection accuracy, and improve user experiences in using voice assistants.

REFERENCES

[1] Muzhir Shaban Al-Ani, Thabit Sultan Mohammed, and Karim M. Aljebory. 2007. Speaker identification: A hybrid approach using neural networks and wavelet transform. *Journal of Computer Science* 3 (2007), 304–309.

[2] Sercan Arik, Gregory Diamos, Andrew Gibiansky, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou. 2017. Deep Voice 2: Multi-speaker neural text-to-speech. In *Advances in Neural Information Processing Systems 30*. NIPS, California.

[3] Bulent Ayhan and Chiman Kwan. 2018. Robust speaker identification algorithms and results in noisy environments. In *International Symposium on Neural Networks 2018*. ISNN, Minsk, 443–450.

[4] Logan Blue, Luis Vargas, and Patrick Traynor. 2018. Hello, Is It Me You're Looking For?: Differentiating Between Human and Electronic Speakers for Voice Interface Security. In *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. ACM, Stockholm, 123–133.

[5] Jean-Francois Bonastre, Driss Matrouf, and Corinne Fredouille. 2007. Artificial impostor voice transformation effects on false acceptance rates. In *Proc. Interspeech 2007*. ISCA, Antwerp, 2053–2056.

[6] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. 2016. Hidden voice commands. In *25th USENIX Security Symposium*. USENIX, Texas.

[7] Nicholas Carlini and David Wagner. 2018. Audio adversarial examples: Targeted attacks on Speech-to-Text. In *1st Deep Learning and Security Workshop*. IEEE, California.

[8] Si Chen, Kui Ren, Sixu Piao, Cong Wang, Qian Wang, Jian Weng, Lu Su, and Aziz Mohaisen. 2017. You can hear but you cannot steal: Defending against voice impersonation attacks on smartphones. In *Proceedings of the 37th IEEE International Conference on Distributed Computing Systems*. IEEE, Georgia, 183—195.

[9] Chung-Cheng Chiu, Tara N. Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J. Weiss, Kanishka Rao, Ekaterina Gonina, Navdeep Jaitly, Bo Li, Jan Chorowski, and Michiel Bacchiani. 2018. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, Alberta.

[10] Charles Elkan. 2001. The foundations of cost-sensitive learning. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI'01)*. IJCAI, Washington, 973–978.

[11] Serife Kucur Ergunay, Elie Khoury, Alexandros Lazaridis, and Sebastien Marcel. 2015. On the vulnerability of speaker verification to realistic voice spoofing. In *2015 IEEE 7th International Conference on Biometrics Theory, Applications and Systems (BTAS)*. IEEE, Virginia, 1–8.

[12] Adrienne Porter Felt, Serge Egelman, and David Wagner. 2012. I'Ve Got 99 Problems, but Vibration Ain'T One: A Survey of Smartphone Users' Concerns. In *Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM '12)*. ACM, New York, NY, USA, 33–44. https://doi.org/10.1145/2381934.2381943

[13] Huan Feng, Kassem Fawaz, and Kang G. Shin. 2017. Continuous authentication for voice assistants. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*. ACM, Utah, 343–355.

[14] Kuruvachan K. George, C. Santhosh Kumar, Ashish Panda, K. I. Ramachandran, K. Arun Das, and S. Veni. 2015. Minimizing the false alarm probability of speaker verification systems for mimicked speech. In *2015 Intl. Conference on Computing and Network Communications*. IEEE, Trivandrum, 703–709.

[15] David Money Harris and Sarah L. Harris. 2013. *Digital Design and Computer Architecture (2nd Ed)*. Morgan Kaufmann, San Francisco. 129–131 pages.

[16] Qin Jin, Arthur R. Toth, Alan W. Black, and Tanja Schultz. 2008. Is voice transformation a threat to speaker identification?. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, Nevada, 4845–4848.

[17] Martin Karu and Tanel Alumae. 2018. Weakly supervised training of speaker identification models. In *Odyssey 2018 The Speaker and Language Recognition Workshop*. ISCA, Les Sables dâĂŹOlonne, 24–30.

[18] Tomi Kinnunen, Md Sahidullah, Hector Delgado, Massimiliano Todisco, Nicholas Evans, Junichi Yamagishi, and Kong Aik Lee. 2017. The ASVspoof 2017 challenge: Assessing the limits of replay spoofing attack detection. In *Proc. Interspeech 2017*. ISCA, Stockholm, 2–6.

[19] Yee Wah Lau, M. Wagner, and D. Tran. 2004. Vulnerability of speaker verification to voice mimicking. In *Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing*. IEEE, Hong Kong, 145–148.

[20] Galina Lavrentyeva, Sergey Novoselov, Egor Malykh, Alexander Kozlov, Oleg Kudashev, and Vadim Shchemelinin. 2017. Audio replay attack detection with deep learning frameworks. In *Proc. Interspeech 2017*. ISCA, Stockholm, 82–86.

[21] Bo Li, Tara N. Sainath, Arun Narayanan, Joe Caroselli, Michiel Bacchiani, Ananya Misra, Izhak Shafran, Hasim Sak, Golan Punduk, Kean Chin, Khe Chai Sim, Ron J. Weiss, Kevin Wilson, Ehsan Variani, Chanwoo Kim, Olivier Siohan, Mitchel Weintraub, Erik McDermott, Rick Rose, and Matt Shannon. 2017. Acoustic modeling for Google Home. In *Proc. Interspeech 2017*. ISCA, Stockholm, 399–403.

[22] Rui Liu, Cory Cornelius, Reza Rawassizadeh, Ronald Peterson, and David Kotz. 2018. Vocal Resonance: Using Internal Body Voice for Wearable Authentication. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2 (2018), 1–23.

[23] Michael McTear, Zoraida Callejas, and David Griol. 2016. *The Conversational Interface: Talking to Smart Devices*. Springer, Switzerland. 166–167 pages.

[24] Shihono Mochizuki, Sayaka Shiota, and Hitoshi Kiya. 2018. Voice liveness detection using phoneme-based pop-noise detector for speaker verification. In *Odyssey 2018 The Speaker and Language Recognition Workshop*. ISCA, Les Sables dâĂŹOlonne, 233–239.

[25] Monisankha Pal, Dipjyoti Paul, and Goutam Saha. 2018. Synthesis speech detection using fundamental frequency variation and spectral features. *Computer Speech and Language* 48 (2018), 31–50.

[26] Saurabh Panjwani and Achintya Prakash. 2014. Crowdsourcing attacks on biometric systems. In *Proceedings of the Tenth Symposium On Usable Privacy and Security (SOUPS 2014)*. USENIX, California, 257–269.

[27] Tanvina B. Patel and Hemant A. Patil. 2015. Combining evidences from mel cepstral, cochlear filter cepstral and instantaneous frequency features for detection of natural vs. spoofed speech. In *Proc. Interspeech 2015*. ISCA, Dresden, 2062–2066.

[28] Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O. Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. 2018. Deep Voice 3: Scaling text-to-speech with convolutional sequence learning. In *6th International Conference on Learning Representations*. ICLR, Vancouver.

[29] Douglas A. Reynolds and Richard C. Rose. 1995. Robust text-independent speaker identification using Gaussian mixture speaker models. *1995 IEEE Transactions on Speech and Audio Processing* 3 (1995), 72–83.

[30] Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, RJ Skerry-Ryan, Rif A. Saurous, Yannis Agiomyrgiannakis, and Yonghui Wu. 2018. Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, Alberta.

[31] Roberto Togneri and Daniel Pullella. 2011. An overview of speaker identification: Accuracy and Robustness Issues. *IEEE Circuits and Systems Magazine, 09 June 2011* 11 (2011), 23–61.

[32] Francis Tom, Mohit Jain, and Prasenjit Dey. 2018. End-To-End Audio Replay Attack Detection Using Deep Convolutional Networks with Attention. In *Proc. Interspeech 2018*. ISCA, Hyderabad, 681–685.

[33] Jesus Villalba, Antonio Miguel, Alfonso Ortega, and Eduardo Lleida. 2015. Spoofing detection with DNN and one-class SVM for the ASVspoof 2015 challenge. In *Proc. Interspeech 2015*. ISCA, Dresden, 2067–2071.

[34] Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonhui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc Le, Yannis Agiomyrgiannakis, Rob Clark, and Rif A. Saurou. 2017. Tacotron: Towards end-to-end speech synthesis. In *Proc. Interspeech 2017*. ISCA, Stockholm, 4006–4010.

[35] Zhizheng Wu, Tomi Kinnunen, Nicholas Evans, Junichi Yamagishi, Cemal Hanilci, Md. Sahidullah, and Aleksandr Sizov. 2015. ASVspoof 2015: The first automatic speaker verification spoofing and countermeasures challenge. In *Proc. Interspeech 2015*. ISCA, Dresden, 2037–2041.

[36] Linghan Zhang, Sheng Tan, and Jie Yang. 2017. Hearing your voice is not enough: An articulatory gesture based liveness detection for voice authenticatio. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, Texas, 57–71.

[37] Xiaojia Zhao, Yuxuan Wang, and DeLiang Wang. 2014. Robust speaker identification in noisy and reverberant conditions. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)* 22 (2014), 836–845.