Combating Replay Attacks Against Voice Assistants

SWADHIN PRADHAN, UT Austin, USA
WEI SUN, UT Austin, USA
GHUFRAN BAIG, UT Austin, USA
LILI QIU, UT Austin, USA

Recently, there has been a surge in the popularity of voice-first devices, such as Amazon Echo, Google Home, etc. While these devices make our life more convenient, they are vulnerable to new attacks, such as voice replay. We develop an *end-to-end* system to detect replay attacks without requiring a user to wear any wearable device. Our system, called REVOLT, has several distinct features: (i) it intelligently exploits the inherent differences between the spectral characteristics of the original and replayed voice signals, (ii) it exploits both acoustic and WiFi channels in tandem, (iii) it utilizes unique breathing rate extracted from WiFi signal while speaking to test the liveness of human voice. After extensive evaluation, our voice component yields Equal Error Rate (EER) of 0.88% and 10.32% in our dataset and ASV2017 dataset, respectively; and WiFi based breathing detection achieves Breaths Per Minute (BPM) error of 1.8 up to 3m distance. We further combine WiFi and voice based detection and show the overall system offers low false positive and false negative when evaluated against a range of attacks.

CCS Concepts: • **Human-centered computing** → *Ubiquitous computing*; *Ambient intelligence*;

Additional Key Words and Phrases: Voice, Replay attack, Spectrogram, Breathing, Authentication, FFT.

ACM Reference Format:
Swadhin Pradhan, Wei Sun, Ghufran Baig, and Lili Qiu. 2019. Combating Replay Attacks Against Voice Assistants. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 3, Article 100 (September 2019), 26 pages. https://doi.org/10.1145/3351258

1 INTRODUCTION

Motivation: Voice is becoming a popular modality of user interface, thanks to the recent advances in speech processing [12]. Voice based interaction is attractive since it is intuitive, operates over longer range, and supports complex queries on a single go. Hence, the voice-based systems are gaining tremendous traction [1], as evidenced by a wide variety of products from several companies, such as Amazon, Google, Apple [39]. These devices enable us to control appliances at home, provide valuable information (*e.g.*, weather, news, recipe), and answer questions.

However, voice is an open channel and inherently insecure. Voice is prone to impersonation, synthesis, or replay attacks [62]. Note that, although the deep learning assisted voice-based user identification is quite main-stream [11], it is vulnerable under replay attacks (*i.e.*, when one plays pre-recorded or synthesized voice commands). Furthermore, with the advances in voice synthesis techniques [13, 21] and different ways to record user's actual voices [30], the *voice replay attack* has become increasingly harder to detect. Below are a few example scenarios that either happened or are likely to happen:

- Opening a smart-door when no one is at home, which results in burglary.

Authors' addresses: Swadhin Pradhan, UT Austin, USA, swadhin@cs.utexas.edu; Wei Sun, UT Austin, USA, wei@cs.utexas.edu; Ghufran Baig, UT Austin, USA, ghufran@cs.utexas.edu; Lili Qiu, UT Austin, USA, lili@cs.utexas.edu.

2474-9567/2019/9-ART100 $15.00
https://doi.org/10.1145/3351258

- Using a voice-first device to start a car without the owner's knowledge.
- Sending SMS or placing a call to premium numbers without one's knowledge.
- Intrusive advertisement by activating the voice-first device like Burger King did recently [6].

To avoid these kinds of situations, we should detect the *liveness* of the voice, *i.e.*, the command is actually uttered by the legitimate user at that very moment.

Challenges: The insecurity of voice-based systems against audible replay attack has been recognized. [62] proposes a wearable that uses voice-correlated accelerometer signature for the detection purpose. However, carrying a device for liveness detection can be cumbersome. [59] uses magnetometer signature of commercial speakers to prevent replay attack. Similarly, a few recent works have attempted to read the lip movement through audio or WiFi to match with the uttered sound [76, 79, 91, 92]. However, these approaches on *voice liveness detection* only work within 10cm even after a rigorous training and calibration, which is insufficient for practical usage scenarios. Camera-based detection can be effective but is vulnerable to lighting conditions or occlusion. Moreover, camera-based approach is much more intrusive: many users are comfortable using voice to interact with home assistants but have serious concerns with cameras taking pictures and videos of them [2]. In general, developing a *wearable-free long-range non-intrusive* system to combat against audible voice-replay attack poses a significant challenge.

Our Approach: In this paper, we aim to design a voice-liveness detection system, called *RE*play-resilient *VO*ice *L*egitimacy *T*ester (REVOLT), to prevent replay attack by maintaining the following design goals: (i) Users do not need to wear any devices, (ii) There are no cameras to preserve privacy, (iii) It should support up to a few meters for typical smart speaker usage scenarios. To achieve these goals, we use both voice and WiFi channels together to differentiate between live vs. replayed voice. Since voice and WiFi channels have different characteristics, they are complementary and can be used together to improve accuracy.

We first analyze replayed and original voice and find replayed voice introduces significant distortions to both low and high frequency components. Such distortions affect both spectral and phase features of the voice, and are present across a wide range of speakers, microphones, sampling rates [67], and environments. Based on these insights, we identify effective voice features to detect replay. Next, we use changes in the WiFi CSI traces to detect breathing to verify human presence, where WiFi CSI [63] captures the channel characteristics such as attenuation and delay across different propagation paths and is measured using known transmission symbols. To improve the detection range, we estimate the direction of arrival (DoA) of voice and move the WiFi directional antennas towards the user to strengthen breathing signals. We further leverage the observation that a user's breathing rate slows down noticeably during speech to check if the voice comes from a live user on site. Finally, we use the changes in wireless CSI spectrograms to develop a convolution neural network model (CNN) to further prevent sophisticated insider replay attack, which manipulates breathing by pretending to speak during the replay. We evaluate each of these stages using real experimental traces. Our results show that our system achieves high accuracy. This work is a first-step toward enhancing the security of voice-first devices in a longer range. Our contributions can be summarized as follow:

- We develop voice-based replay detection and give insights about why certain voice features are useful in distinguishing the original and replayed voice.
- We leverage the WiFi CSI to detect human presence and live speaking. To the best of our knowledge, we are the first that exploits the synchronized changes in voice and breathing to detect replay attacks.
- We implement a prototype of REVOLT on a commodity hardware, and conduct extensive evaluation to demonstrate its effectiveness.

Outline: The rest of the paper is organized as follow. We first describe attack models in Sec. 2. Then we give an overview of our REVOLT in Sec. 3. We describe and evaluate different modules with their respective operations within REVOLT in Sec. 4, Sec. 5, and Sec. 6. Next, we provide end-to-end evaluation of REVOLT in Sec. 7, while discussing different modes. We review related work in Sec. 8 in detail and discuss about the limitations in Sec. 9. We conclude with highlighting the importance of REVOLT in Sec. 10.

2 THREAT MODEL

In our threat model, an attacker can collect voice samples from the victim, or synthesize the voice [9, 13] commands. Then the attacker replays the voice commands through a speaker to bypass the voice-enabled systems. We classify the threats into the following three categories:

- *Remote voice replay*: An attacker has access to a speaker inside a home and can remotely replay the recorded or synthesized voice using the compromised device. Such attacks are feasible as shown in [88]. To make the attacks harder to detect, the attacker can further post-process the voice samples to mimic live voice.
- *Local voice replay*: The attacker is physically close to the smart speaker and can replay the pre-recorded voice *e.g.*, using the attacker's phone or speaker.
- *Sophisticated replay*: The attacker not only replays but also manipulates his physiological signals. The attacker could silently speak without actually emitting the sound during the replay to mimic the breathing pattern of a live speaker.

In this paper, we do not consider human-based voice impersonation attack, since this is widely studied in the related works (*e.g.*, [68, 78, 98]). These works generally detect the impersonation by exploiting the unique characteristics in an individual's voice. Replaying a pre-recorded voice poses a different and challenging issue since voice synthesis or stealthy recording can capture or mimic the victim's voice to evade the previous detection strategies. Therefore, our focus is mainly to prevent *voice replay* attacks.

3 OVERVIEW

In the following section, we give an overview of our system architecture and work flow.

3.1 System Overview

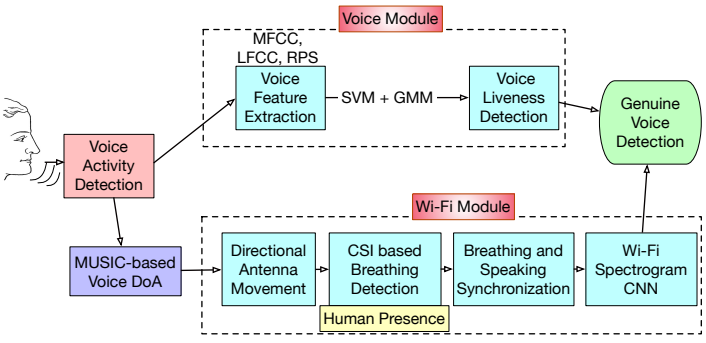


Fig. 1. Architecture of REVOLT.

REVOLT is a wearable-free long-range non-invasive solution to defend against voice replay attacks. Our key idea in this work is to exploit voice and WiFi signals together to prevent voice replay attacks.

As illustrated in Fig. 1, our system first detects the presence of incoming voice. Then it runs both *voice module* for voice analysis and *WiFi module* for breathing detection and analysis. These modules start recording the data simultaneously. At the first stage, to strengthen the received WiFi signal for breathing analysis, our system computes the Direction-of-Arrival (DoA) of the voice source based on a wake-up word like *Alexa* and directs the WiFi directional antenna in that direction. Then the *WiFi module* uses the WiFi CSI to detect the breathing signal for user presence identification. Simultaneously, *voice module* records the voice signal and extracts the voice features like LFCC, MFCC etc. from the spectrogram (described in detail in Sec. 4.1) and applies a pre-trained supervised machine learning algorithm to distinguish between the original and replayed voice (as shown in Fig. 1. Furthermore, the *WiFi Module* checks if the breathing rate changes whenever the voice is active. Finally, the *WiFi Module* uses a pre-trained convolution neural network model (CNN) to detect whether the breathing changes are natural or manipulated based on the voice and WiFi features (to prevent *sophisticated replay*).

Note that, our WiFi analysis requires the user is *quasi-static*. When the user moves, it automatically extracts the period when the user is static and uses the WiFi traces from the static period for detection. If the user continuously moves, it flags as a potential attacker and requires additional authentication.

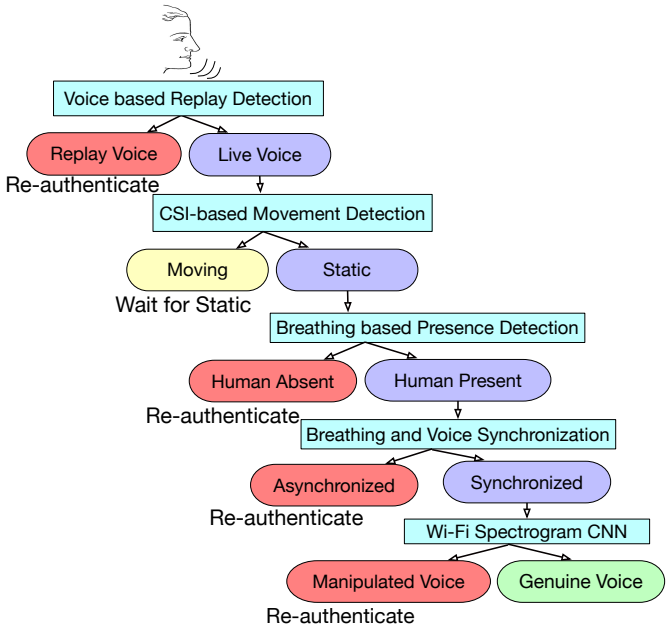


Fig. 2. Work-flow of REVOLT.

Fig. 2 shows the work-flow of our algorithm within the previously described modules. The *voice module* uses the pre-trained machine learning model to detect whether the voice command is replayed or not. If it passes this test, the *WiFi module* filters out the period in which the WiFi CSI amplitude changes substantially. Then it detects the presence of breathing signature to ascertain the human presence. If it passes this checking, we use the *voice module* and *WiFi module* together to check if the breathing pattern is synchronized with voice. In this way, we can detect if the voice is coming from the user whose breathing signal is being detected. At the final stage, we will invoke the CNN module to check the liveness of the user's voice to prevent against a sophisticated attacker controlling his breathing to synchronize with the replayed voice. If it fails at any stage,

the voice command is flagged as potential replayed voice, which include all red nodes in the tree; additional authentication is required as shown in Fig. 2. If it passes all stages, the voice command is considered genuine. Note that our system is designed to prevent different types of replay attacks. We can use the security policy as described below to configure the detection thresholds.

3.2 Security Policy

We observe that different commands require different levels of security. For example, playing music is not as mission critical as opening a door/window or starting a car. Therefore, we can allow users and product designers to group the commands into a few security classes, each associated with a certain security policy. For example, for mission critical commands, we should be more conservative and minimizing the false positive (*i.e.*, flagging replayed voice as live). For less critical commands, we can tolerate some false positive in return for reduced false negative (*i.e.*, flagging live voice as replayed and requiring unnecessary re-authentication). We select the appropriate parameters in REVOLT according to the security policy. To demonstrate this, in Sec. 7, we have described to operational settings, namely, *Critical* and *Casual*, which shows desirable false positive and false negative percentages.

4 VOICE-BASED REPLAY DETECTION

In this section, we introduce the *Voice Module* of REVOLT. We first describe the spectral and phase structure of live voice vs. replayed voice. Based on the insights, we identify new features for detecting replayed voice. Then we describe our evaluation methodology and results.

4.1 Traditional Voice Features

We extract several important spectral features from voice to detect replayed voice. They include Mel-Frequency Cepstral Coefficient (MFCC), Linear Frequency Cepstral Coefficient (LFCC), Relative Phase Shift (RPS), which will be described below. These features essentially aim to capture frequency and phase structure of the voice segment. We have also tried several other features, such as IMFCC, CQCC, PD etc. [99], but find that MFCC, LFCC, and RPS are the best features and also complement well with each other.

MFCC: Cepstral coefficients or cepstrum features are the derived time-frequency features. It essentially deconvolves the voice source and voice filter by applying frequency transformation twice. The most widely used cepstrum features are Mel-frequency cepstral coefficients (MFCC) [80]. It models the way in which a human perceives sound by using logarithmic filter banks. Therefore it has higher resolution in lower frequency spectrum.

LFCC: It is the same as MFCC but uses linear filter-banks across the entire frequency. Therefore, it has better resolution at the high frequency due to its use of linear filter banks instead of logarithmic filter banks in MFCC.

Relative Phase Shift (RPS): Relative Phase Shift (RPS) transforms instantaneous phases from two harmonic frequencies into a relative measure against a common reference phase [74]. This measure removes the linear phase component and makes the phase structure more clear. RPS assumes a harmonic model with N harmonics for voice segments of the form: $S(t) = \sum_{k=1}^{N} A_k(t)cos[\phi_k(t)]$, where A_k is the amplitude. ϕ_k is the instantaneous phase of the kth harmonic and computed as: $\phi_k = 2\pi k f_1 t + \theta_k = 2\pi f_k t + \theta_k$ where $f_k = k f_1$ is the kth harmonic. We denote the fundamental frequency as f_1 and the initial phase offset as θ_k. If we assume the fundamental frequency as our reference and for simplicity $\theta_1 = 0$, we can solve for θ_k (now called the RPS) as follow:

$$\frac{\phi_1(t_a)}{2\pi f_1} = \frac{\phi_k(t_a) - \theta_k}{2\pi f_k} \tag{1}$$

or

$$RPS, i.e., \theta_k = \phi_k(t_a) - k\phi_1(t_a) \tag{2}$$

Therefore, we get more smooth phase structure (Fig. 6(b)) in RPS than random instantaneous phase as shown in Fig. 6(a). To further parameterize the RPS described in Eq. 2, we perform mel-filtering, unwrapping, and DCT transformation.

4.2 Datasets Used

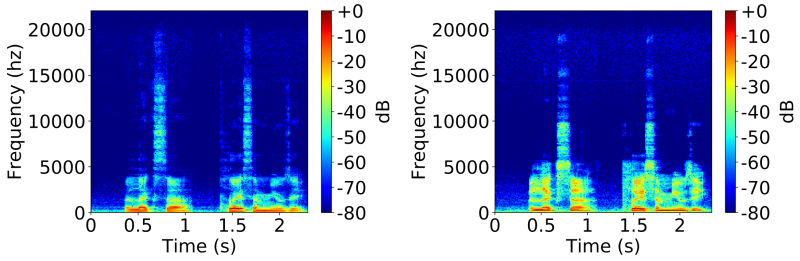
We evaluate our voice-based detection using both *ASV2017 Dataset* [4] and our dataset.

ASV2017 Dataset: The dataset is primarily based on the recent text-dependent RedDots corpus [70] and its replayed version using 42 speakers and 5 microphones under 123 environment configurations [3, 99]. Experimenting using such a large number of speakers and microphones in the dataset helps to develop general detection schemes that are robust against artifacts of specific speakers or microphones. The audio is captured at a sampling rate of 16KHz. The training set consists of 10 speakers and 3 microphones, and the testing set consists of 24 speakers and 110 replay configurations (*i.e.*, the number of combinations of speakers, microphones, and locations). The evaluation set contains around 50,000 audio files. Refer to [3, 4, 99] for more details.

Our Dataset: In addition, we collect data from 10 users (4 women and 6 men between 25 to 45 year old) using 9 speakers, 9 microphones, and across 5 different locations. The dataset contains 6 commands, which includes 3 short utterances: *Alexa*, *Siri*, *Cortana* and 3 most popular Alexa commands like *Alexa, what is the weather*, *Alexa, set a timer*, and *Alexa, play some music*. We record the audio at a sampling rate of 44.1 KHz with a varying distances from 10cm to 250cm. We use 9 different microphones: a standard USB microphone [41], macbook pro 2014 microphone, macbook pro 2017 microphone [46], Google Pixel 2 microphone, Motorolla G5 microphone, Xiaomi MI 8 microphone [48], Tonor karaoke microphone [47], Apple iPad Pro microphone [42], and an Samsung S7 microphone [33]). We also use 9 different speakers for replay: Mokcao style Bluetooth Speaker [24] (for its smaller size), Logitech Speaker System Z323 [20], Cyber Acoustics speaker [40], JBL Flip4 bluetooth speaker [43], macbook pro 2017 speaker, Google Pixel 2 speaker, Motorolla G5 speaker, Apple iPad Pro speaker [42], and the Samsung S7 speaker [33]. We use these speakers and microphones since they cover a wide range of types, sizes, and quality. In addition, we also perform our experiments in different rooms: a lab, a conference room, an open stair-case area, an apartment living room, and an apartment bed-room and vary the background noise level to evaluate the robustness of our system. For noise related experiments, we add white noise, musical instruments, conversations, singing, or whispering to the live or recorded voice samples.

4.3 Insights

Next, we analyze voice spectral and phase structure diagrams based on the above features. We use the datasets described in Sec. 4.2 for our analysis.



(a) Live Voice Segment. (b) Replayed Voice Segment.

Fig. 3. Difference in Linear Frequency Spectrogram.

(i) Frequency Characteristics: Fig. 3 plots the linear time-frequency spectral diagrams of live vs. replayed voice, where the x and y axes in the graph denote the time and frequency, respectively. In the replay attack, the attacker records the voice of a target speaker and later plays it back, as depicted in Fig. 4. Therefore, replayed voice goes through more conversion steps than live voice. As shown in Figure 3, the high frequency has significant differences. In particular, from 7KHz to 16KHz, there is more spectral energy present in the live voice than the replayed voice. This happens for the following two reasons: (i) non-flat frequency response of speakers and microphones especially at high frequency since these frequencies are rare in typical voice and speakers are not optimized for these bands, and (ii) replayed voice goes through a microphone twice and each time it incurs aliasing at the analog-to-digital converter (ADC) (Fig. 4), which introduces additional distortion due to quantization error.

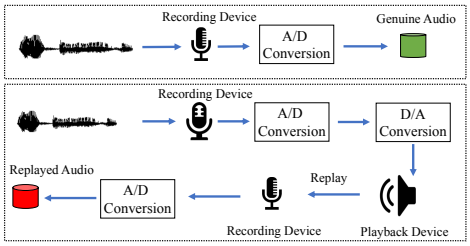
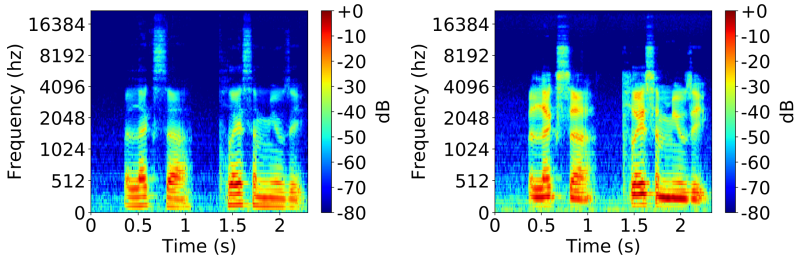


Fig. 4. Genuine and Replay File Generation.

In addition, we observe stronger replayed voice below 4 KHz because human ears are biased toward the frequency within 4 KHz according to psycho-acoustics study [26, 66] and speakers tend to strengthen signals at these frequencies. As mel-filtering zooms in this lower frequency region, we can observe in Fig. 5 that the replayed voice (Fig. 5(b)) has higher energy in these lower frequency bands (specifically 100 Hz to 400 Hz). In comparison, as shown Fig. 5(a), the energy decay more gradually with the frequency in the live voice. This is true across all types of speakers used in both our and ASV2017 datasets (Sec. 4.2). **(ii) Phase Characteristics:** We

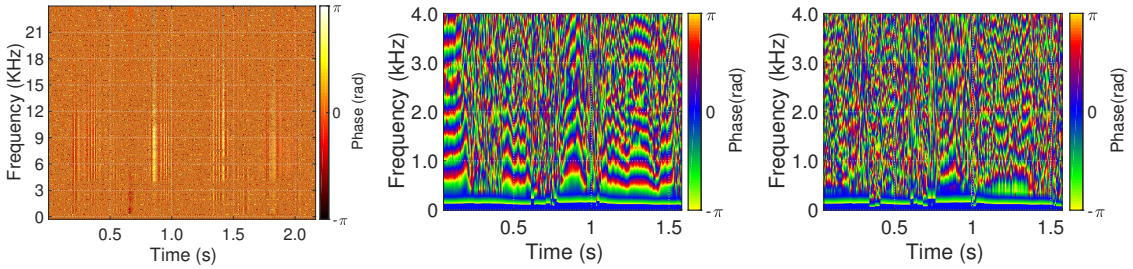


(a) Live Voice Segment. (b) Replayed Voice Segment.

Fig. 5. Difference in Mel-frequency Spectrogram.

also study the relative phase structure through RPS spectrogram of live vs. replayed voice. Fig. 6(b) and Fig. 6(c) compare the RPS spectral diagrams of the live vs. replayed voice, where the x and y axes in the graph denote time and frequency, respectively. As they show, there are clear stripes in the phase graph of live voice. In comparison, the stripes in the replayed voice are evident only in the low frequency. This holds true across all speakers and

users in our study ((Sec. 4.2)). The stripes in the phase graph of the original voice is due to vocal-fold vibration driven human speech production [14, 35, 94]. The absence of clear stripes of the replayed voice is also caused by speakers' distortion and ADC quantization error incurred by extra DAC and ADC operations in replay as shown in Fig. 4.



(a) Instant Phase of Voice Segment. (b) RPS of Live Voice Segment. (c) RPS of Replayed Voice Segment.

Fig. 6. Difference in Phase Structure Spectrogram.

4.4 Our Voice Features

(i) Combining MFCC-LFCC: Since we observe clear difference in frequencies above 10 KHz or below 8 KHz between the original and replayed voice, we seek a frequency spectrogram that provide more granularity at those frequencies. Specifically, we zoom into the frequencies above 10 KHz and below 8 KHz. We use linear filters at higher frequencies and logarithmic filters at lower frequency by extracting the 26 coefficients from LFCC (from 49 to 75), which corresponds to frequencies above 4 KHz and extracting the first 64 coefficients from MFCC, which corresponds to frequencies below 8 KHz. Here the coefficients mean the number of frequency bands in a filter bank. These combined features are new and will be shown more effective to detect replay.

(ii) Combining Frequency and Phase Features: We further combine frequency and phase based features to enhance the accuracy by combining the outputs from the two classifiers using these features.

4.5 Machine Learning for Replay Detection

We use the following machine learning models and their ensemble to detect replay: SVM is effective when we know which features are important and there are a small number of important features, and GMM can learn the important features from a larger number of features but takes more time to build a model. GMM is common in the speech community. Furthermore, the choice of these machine learning models are motivated by the chosen features depicting clear difference between the live vs. replayed voice.

SVM: Support vector machines (SVM) is a widely used supervised machine learning algorithm. It finds one or more hyper-planes to separate training data belonging to different classes. There are linear and non-linear SVM. We use a non-linear SVM with RBF kernel from libSVM library [18] after trying out different possibilities. We train the SVM with 5-fold cross validation in our dataset by splitting the data into different user groups. The training and testing data do not have the same user voice command. For the ASV2017 dataset, we train using the training and development data and test using the evaluation data.

GMM: Gaussian Mixture Models (GMM) is a popular probabilistic model to capture normally distributed sub-populations within an overall population. This technique assumes that all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. We train two Gaussian mixture

Table 1. EER of different features on our dataset. The last two rows in bold represent the results from our features. The numbers inside () represent the number of coefficients used in the filter-bank.

Feature	Classifier	MVN	EER
LFCC (90)	GMM	No	4.82
MFCC (90)	GMM	No	4.95
LFCC (90)	GMM	Yes	14.93
MFCC (90)	GMM	Yes	18.41
LFCC (90), Δ, Δ^2	GMM	No	8.01
MFCC (90), Δ, Δ^2	GMM	No	10.57
LFCC (90), MFCC (90)	GMM	No	**3.83**
RPS (64)	GMM	No	7.73
LFCC (90)	SVM	Yes	4.21
MFCC (90)	SVM	Yes	**1.49**
LFCC (90), Δ, Δ^2	SVM	Yes	4.20
MFCC (90), Δ, Δ^2	SVM	Yes	2.18
LFCC (90), MFCC (90)	SVM	Yes	**1.56**
LFCC-MFCC	**SVM**	**Yes**	**1.11**
LFCC-MFCC/RPS	**Fusion**	**Yes**	**0.88**

Table 2. EER of different Features on ASV2017 Dataset. The last two rows in bold represent the results from our features. The numbers inside () represent the number of coefficients used in the filter-bank.

Feature	Classifier	MVN	EER
LFCC (90)	GMM	No	29.39
MFCC (90)	GMM	No	30.32
LFCC (90)	GMM	Yes	18.92
MFCC (90)	GMM	Yes	17.66
LFCC (90), Δ, Δ^2	GMM	Yes	19.02
MFCC (90), Δ, Δ^2	GMM	Yes	17.49
LFCC (90), MFCC (90)	GMM	Yes	15.12
RPS (64)	GMM	No	22.05
LFCC (90)	SVM	Yes	16.63
MFCC (90)	SVM	Yes	**13.19**
LFCC (90), Δ, Δ^2	SVM	Yes	17.91
MFCC (90), Δ, Δ^2	SVM	Yes	15.51
LFCC (90), MFCC (90)	SVM	Yes	13.56
LFCC-MFCC	**SVM**	**Yes**	**12.13**
LFCC-MFCC/RPS	**Fusion**	**Yes**	**10.32**

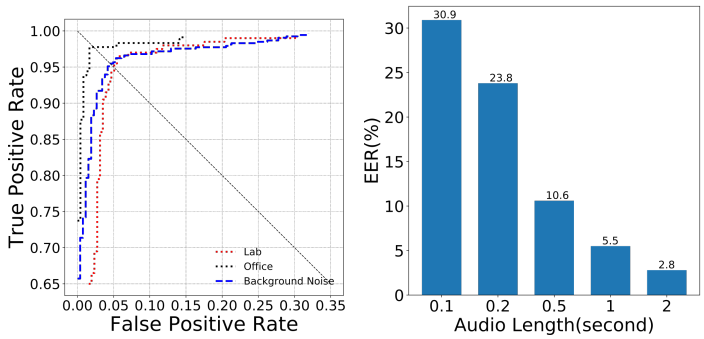
models using live and replayed voice, and compute the likelihood of new voice samples with respect to these models. We train GMM using RPS as RPS has a continuous sequence pattern and correlated data across different frames. We also test the impact of feature normalization through Mean-Variance Normalization (MVN) [84].

4.6 Implementation

As silent periods of voice segments has no apparent benefit in training/testing, we extract the active voice using a utility within *covarep* [8] package. We use *librosa* [17] package to get MFCC and design linear triangular filter to get LFCC. For the phase feature, we use *covarep* [8] package to extract the effective RPS pattern. We use *sklearn libsvm* to train SVM using the MFCC-LFCC feature and use *vlfeat* [38] package to train GMM using the RPS feature. We further combine the output from both models by learning the weights using bosaris toolkit [5].

4.7 Evaluation Results

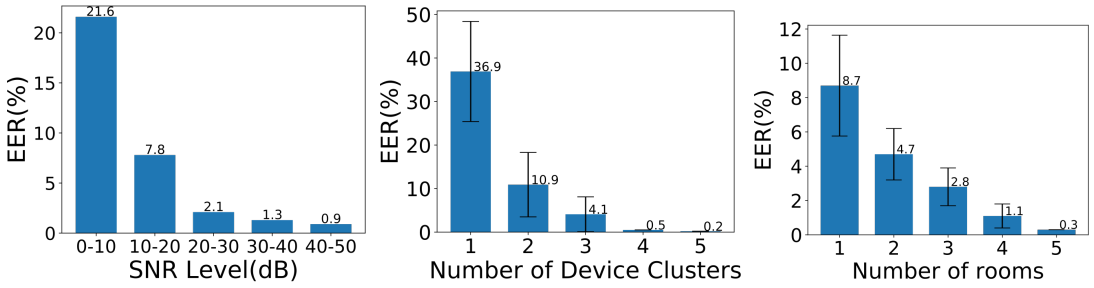
We quantify the accuracy using equal error rates (EER), which is the error rate when False Negative Rate (FNR) equals False Positive Rate (FPR). Following the definitions in [99], false negative denotes live voice considered as replayed and false positive denotes replayed voice considered as live. The lower the EER, the better the performance. Apart from the features described in Sec. 4.1, we also calculate delta (Δ) (*i.e.*, the difference between consecutive Cepstral Coefficients (CC)) and double delta (Δ^2) features (*i.e.*, the difference between Delta). Table 1 and Table 2 show the EER results of different features, machine learning algorithms, and feature normalization techniques in our and ASV2017 datasets, respectively. We make several observations. First, our choices of filter banks in LFCC and MFCC out-performs the standard LFCC and MFCC by focusing on the frequencies that matter the most. Using phase in addition to LFCC-MFCC further reduces the error from 1.11 to 0.88 in our dataset and from 12.13 to 10.32 in the ASV2017 dataset.



(a) Impact of Configuration on ROC. (b) Impact of Audio Length on EER.

Fig. 7. ROC Curve and EER Bar-graph.

Impact of Different Factors: To evaluate the robustness of our voice-based replay detection module, we also test in different conditions (*e.g.*, increasing the background noise, and changing the recording room environments). As illustrated in the ROC curve (Fig. 7(a)), the EER (dotted diagonal line) remains within 5% in all cases. We use house-hold sounds such as conversation, door opening, utensil sound from [32], human conversation, and musical sound, as background noise; and we initially use two different locations: a conference room and an apartment room for testing, and apply the model trained using the data collected from the lab. Later, to evaluate the generalization and robustness of our *voice module*, we include three more locations: an open stair-case area, an apartment living room, and an apartment bed-room, 6 more speaker-microphone pairs, different background noise levels, and different voice intensity levels of speech utterances, in our testing dataset. In the following, we show that the accuracy remains high even with more diverse locations.



(a) Impact of different Noise Levels. We add different amount of background noise to both live and replayed voice samples.

(b) Impact of Device Diversity. We train a model using a specified number of speaker microphone pairs and report the EER from randomly drawn samples.

(c) Impact of Location Diversity. We train a model using a specified number of rooms and report the EER from randomly drawn samples across all rooms.

Fig. 8. Impact of Diversity in Training in EER.

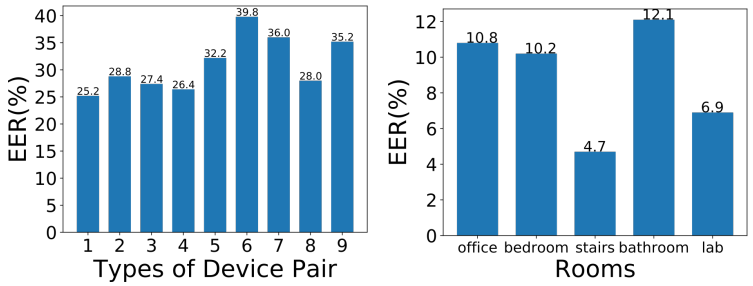
In experiments, the basic pattern of phase or frequency spectrogram does not change significantly. Therefore, the EER remains low. Background noise impacts both the live and replayed voice spectrograms equally and thus does not degrade the EER significantly. Next, we vary the audio length. In the Fig. 7(b), we can see that for a very small duration (0.1sec), the EER is very high (30%) due to the lack of the differentiating features in such a

small duration. However, when the audio duration lasts 1sec (*e.g.*, *Alexa* takes around 0.8sec to pronounce), the EER becomes only 5.5% since more distinguishable features in both frequency and phase space become available. We further add different amount of background noise to vary the SNR of both live and replayed voice. Fig. 8(a) shows that the EER degrades by within 7.8% even under low SNR (10 − 20 dB).

Next we evaluate the impact of the number of locations used for training. For all cases, we draw 1500 voice samples for training regardless of the number of locations used for training. In this way, we can ensure the accuracy difference comes from the diversity in the training locations instead of the increased number of samples. We generate the testing data by randomly drawing 8000 voice samples from the remaining data sets, which contain different locations and device pairs. Fig. 8(c) shows that training from one room yields a high EER (5.92-11.57) in the testing data; training from two different types of locations is sufficient to get reasonable EER (2.9%) in the testing data-set collected from more locations.

We further evaluate the impact of the number of device pairs. As before, we also limit the total training data to 1500 voice samples. As shown in Fig. 8(b), when there is only one device pair used for training, the EER is 25% to 29%. However, as soon as we include 3 device pairs, the EER reduces to below 3.5%.

To better understand the impact of data diversity on our voice model, we focus on the EER of a single room or a single device scenarios. Fig. 9(a) show the EER is generally high using the training from one device (25-35%) or one location (4-12%) (shown in Fig. 9(b)). These results show that device and environment diversities are both important but including only a small number of device pairs and environments is sufficient to get a general model that works for a wider range of scenarios.

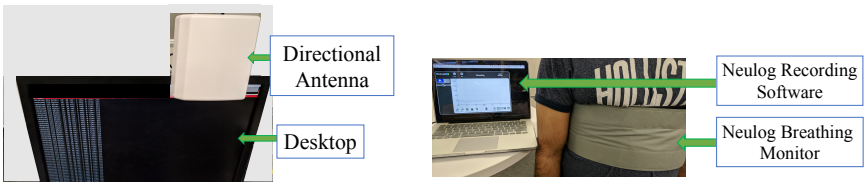


(a) EER of samples drawn from all device pairs as we vary the device pair used.

(b) EER of samples drawn from all rooms as we vary the room used for training.

Fig. 9. Impact of Single Device and Location Diversity.

Summary: We make the following observations from the results. (i) Among individual features, LFCC-MFCC performs the best for both our and ASV2017 datasets, which is at least 10% better than MFCC or LFCC alone. This is consistent with our intuition since they provide finer-granularity over the frequencies in which the live and replayed voice differ the most. (ii) Combining LFCC-MFCC with phase using the bosaris toolkit [5, 23] gives the lowest EER. This is as expected since both frequency and phase show distinct patterns between live and replayed voice. The impact of frequency feature is higher. (iii) ASV2017 has larger EER than our dataset for two reasons. First, their datasets have a lower sampling rate: 16 KHz in their dataset vs. 44.1 KHz in our dataset, which corresponds to measuring up to 8 KHz and 22.05 KHz frequencies, respectively. As shown in Section 4.3, the frequencies above 10 KHz see large difference between live versus replayed voice, the lower sampling rate used in ASV2017 significantly limits the detection accuracy. Second, their datasets have much more testing configurations and speakers (24 speakers and 110 replay configurations) than those in the training (10 speakers and 3 replay



(a) Initial static transmitter of REVOLT. (b) Breathing recorder of REVOLT.

Fig. 10. Breathing Experimental Setup.

configurations) [3]. Overall, our evaluation sheds light on which features are effective for voice-based detection and why. (v) Our detection algorithm is robust to background noise level and environments. In both cases, the EER remains within 5%. 1 second audio is sufficient for replay detection. (vi) Our system generalizes well with different variations like room, device, noise level, when the training data lacks diversity.

5 WIFI BASED PRESENCE DETECTION

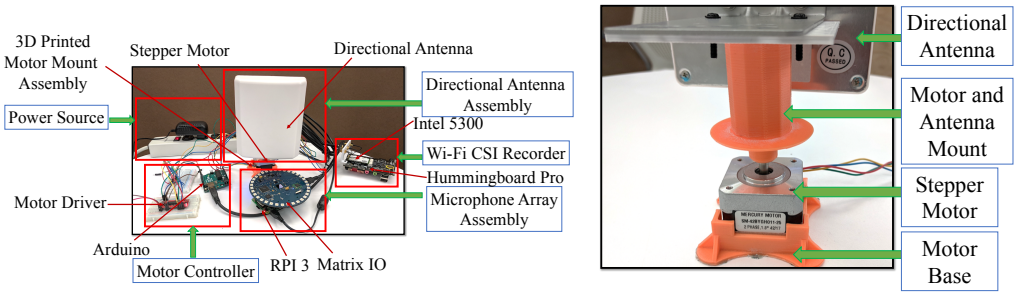
In the next stage, we detect the presence of breathing using WiFi CSI. This is an important vital sign which ascertains the user presence and avoids remote replay. This part is built on the existing works [49, 75, 89]. We introduce a few modifications to the signal processing pipeline and apply them to more reliably detect user presence. Below we provide the detail of our breathing based presence detection model, our measurement setup, and the evaluation results.

5.1 Breathing Rate Detection Using CSI

CSI Setup: We use the Intel WiFi 5300 (iwl5300) card to collect the Channel State Information (CSI). We use the modified driver and firmware from [19] that reports the channel-state matrices for 30 sub-carrier groups in a 20 MHz channel at 5.7GHz (channel 140). Each matrix specifies the amplitude and phase of the WiFi channel between a pair of transmit and receive antennas. We modify the logging utility of [19] to process the CSI data in real-time. We use [71] to sanitize the phase. Our customized C code computes and normalizes the CSI matrices. These matrices are then sent to a python program that uses *SciPy* library [28] to perform band pass filtering of the amplitude and phase of these CSI values. We then perform FFT on these filtered data and use *peakutils* library [27] to detect peaks in the FFT spectrum to get breathing events and rates.

Approach: [49, 75, 89] demonstrate the feasibility of using WiFi CSI amplitude to monitor the breathing rate. Built on these works, we first extract the CSI amplitude and phase. We use band-pass filtering (from 0.5Hz to 1.5Hz) to remove noise in the CSI data. We also remove the DC component by subtracting the average. Then we compute the variance of each sub-carrier group CSI, and select the sub-carrier groups whose variance exceeds 90%. We use the time series from the selected sub-carrier groups to form a matrix, and perform PCA on the matrix to select the first principle component. We compute FFT on the principle component and select the peak. We use the peak every 30 seconds to form a BPM (Breathes per minute) time-series, and perform outlier removal on the BPM time-series. Different from the existing works, we use both CSI phase and amplitude to detect breathing and estimate BPM. Fig. 14 shows using the phase along with the amplitude improves the BPM estimation by around 10% . While the exact BPM is not required for user presence detection, it is useful for detecting breathing rate change during speech in Sec. 6.2.

Experimental Setup: We use a 3.20GHz Intel(R) Pentium 4 CPU 2GB RAM desktop equipped with Intel 5300 NIC (Network Interface Controller) as the transmitter and use the interface injection mode [19]. The transmitter has a Tupavco TP542 directional antenna [37] with 13 dBi and 40 degree VPOL and operates in IEEE 802.11n AP mode



(a) Receiver or transmitter of REVOLT.

(b) 3D printed stepper motor assembly.

Fig. 11. Prototype of REVOLT.

at 5.7GHz on channel 140. The desktop runs Ubuntu 12.04 LTS with modified Intel driver and firmware [19]. The setup is shown in Fig. 10(a). During the measurement, the transmitter sends packets at a rate of 100 packets/second. The collected CSI are then stored and processed at the receiver. We capture the ground-truth of breathing in parallel using the Neulog Respiration monitor [31] through the USB module [25] as shown in Fig. 10(b). The ground truth data is only used to quantify the accuracy, and not required in our system.

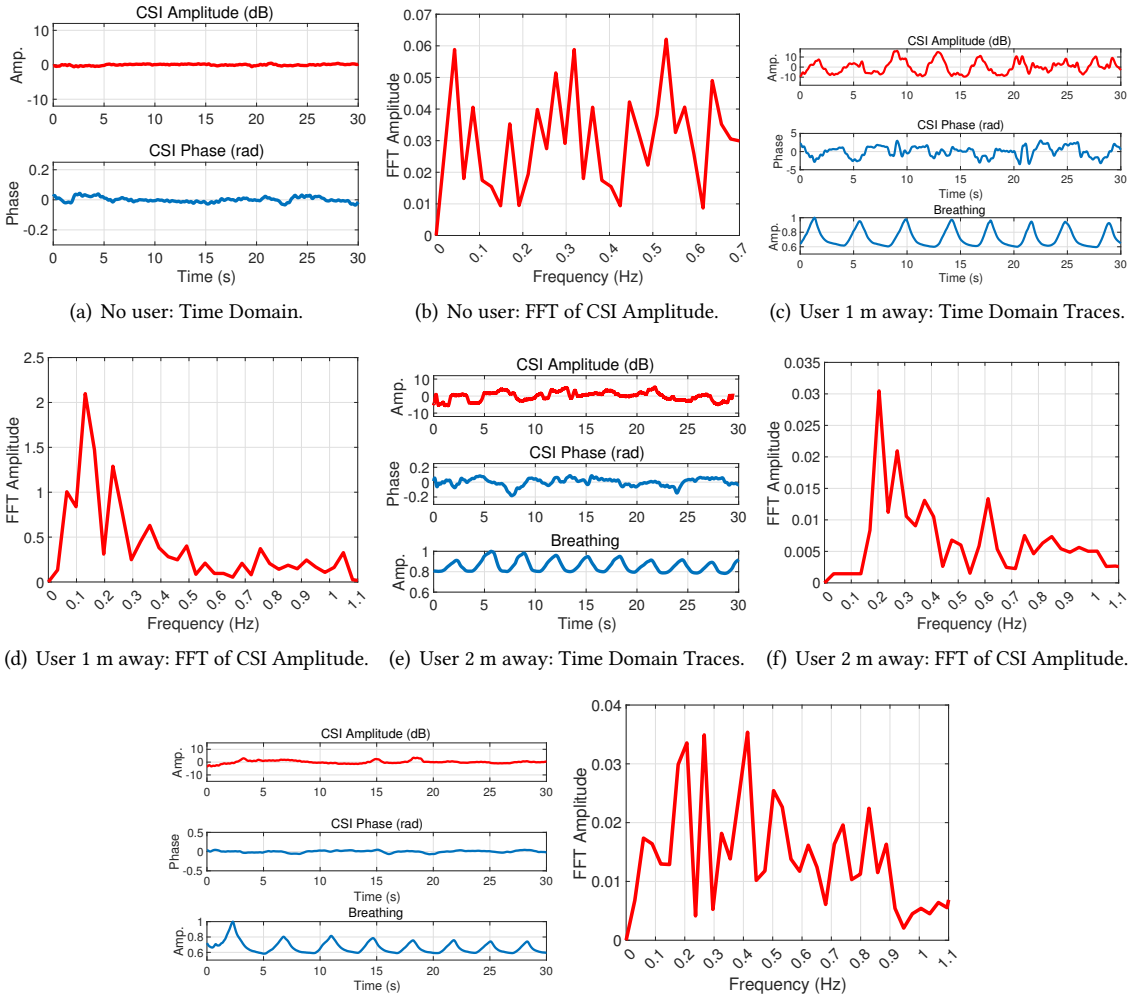
To improve the detection range, we build a voice activated movable antenna setup shown in Figure 11(a). It consists of (i) Matrix-IO board [22] to get audio samples, (ii) Intel 5300 card to get WiFI CSI, (iii) Raspberry Pi [29] to detect voice and perform audio processing, (iv) Hummingboard Pro [15] connected with Intel card to perform CSI processing, (v) directional antennas mounted on the Intel 5300 card, and (vi) arduino, stepper motor, motor controller, motor driver, and 3D printed mount shown in Fig. 11(b), to move the directional antenna towards the direction of incoming voice. NTP is used to synchronize all modules. We apply the MUSIC algorithm [53] implemented in Eigen3 library [10] on Raspberry Pi to determine DoA of the incoming voice, and then move the tx/rx antennas towards the voice using arduino.

We have recruited the same 10 users (8 male and 2 female) for our breathing detection experiments where the users stand at different locations. We ask them to speak different voice segments so that our voice-directed directional antenna can move and pick up breathing pattern based on the CSI trend. We separately evaluate each of the components in REVOLT. We perform our experiments in three locations: the lab, conference room, and bedroom. We keep the normal furniture set-up in the respective rooms.

Impact of CSI Phase: Previous works use the CSI amplitude to detect breathing. Fig. 13 shows that there are peaks in the FFT spectrum of both amplitude and phase. Sometimes phase spectrum has a peak and sometimes amplitude spectrum has a peak. This occur more often for longer distances. If both of the peaks are present, we calculate the BPM from each and take the average from them. As shown in Figure 14(a), if we detect breathing based on peaks in either phase or amplitude spectrum, we improve the breathing detection from 83% to 96%. In addition, Fig. 14(b) shows that using both CSI phase and amplitude improves BPM error from 1 to 0.82.

5.2 Detecting Presence Using Breathing

Visualizing Breathing Patterns: Fig. 12 shows the time domain and frequency domain of the CSI amplitude and phase. As we can see, when there is no user, the signals are almost flat and there are no dominant peaks. When there is a user nearby, we plot CSI amplitude, phase, and ground truth breathing in Fig. 12(c)(e)(g). When the user is within 2m, we can see periodic changes in the time-domain CSI and dominant peaks in their FFT plots. The time-domain changes are more prominent at 1 m than at 2 m. As the user moves out of range (*e.g.*, 5 m away),



(a) No user: Time Domain. (b) No user: FFT of CSI Amplitude. (c) User 1 m away: Time Domain Traces.

(d) User 1 m away: FFT of CSI Amplitude. (e) User 2 m away: Time Domain Traces. (f) User 2 m away: FFT of CSI Amplitude.

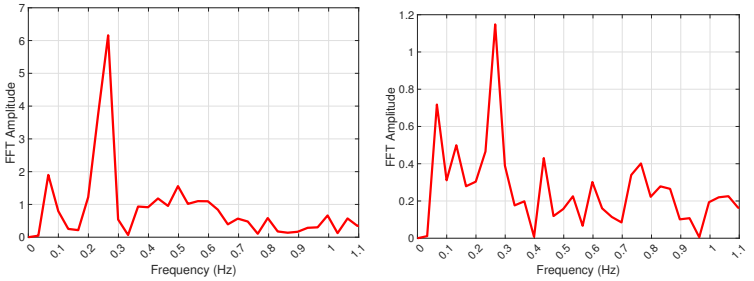
(g) User 5 m away: Time Domain Traces. (h) User 5 m away: FFT of CSI Amplitude.

Fig. 12. Trace Patterns due to Breathing of a User.

the signals become flat and there are no sharp peaks in the FFT plot. These results show the feasibility of using WiFI to detect breathing within 2 m.

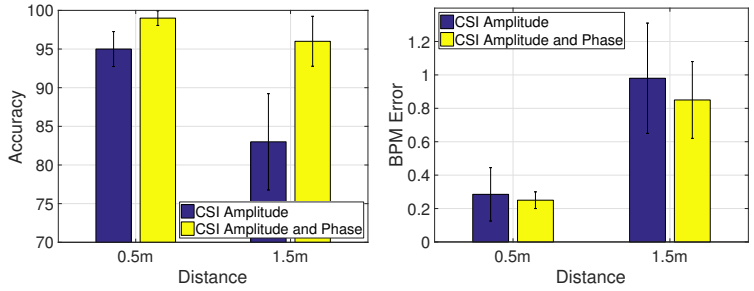
Impact of CSI Phase: Previous works use the CSI amplitude to detect breathing. Fig. 13 shows that there are peaks in the FFT spectrum of amplitude and phase. Sometimes phase spectrum has a peak and sometimes amplitude spectrum has a peak. This occur more often for longer distances. If both of the peaks are present, we calculate the BPM from each and take the average from them. As shown in Figure 14(a), if we detect breathing based on peaks in either phase or amplitude spectrum, we improve the presence of breathing detection from 83% to 96%. In addition, Fig. 14(b) shows that using both CSI phase and amplitude improves BPM error from 1 to 0.82.

Accuracy of Detecting User Presence: We use the setup in Fig. 15(a) to evaluate the accuracy of detecting user presence. As illustrated in Fig. 15(b), we can detect the user presence over 90% in a $2m \times 3m$ area based on



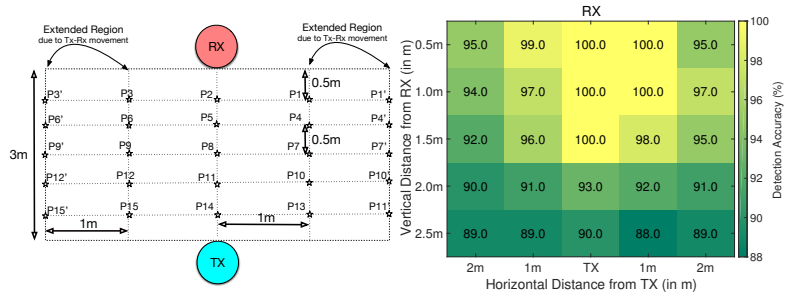
(a) FFT of CSI Amplitude. (b) FFT of CSI Phase.

Fig. 13. Impact of Phase in Breathing.



(a) Improvement in Breathing Detection. (b) Improvement in BPM Error.

Fig. 14. Use of CSI Phase and Amplitude. Error bar denotes standard deviation unless otherwise specified.

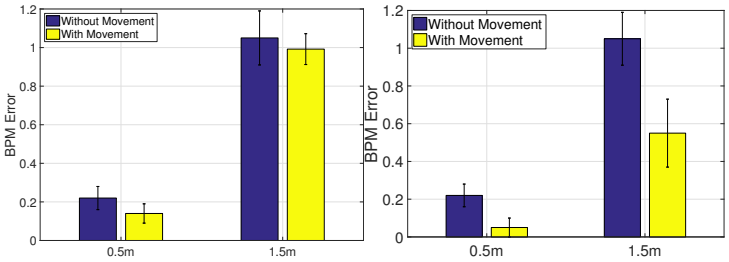


(a) Experimental Scenarios. (b) Breathing Detection Accuracy Heat-map.

Fig. 15. Breathing Experiment Setup.

breathing frequency detection. By steering the transmitter and receiver antennas, we can increase the area to $3m \times 3m$. To get a wider range of breathing detection, we need to move the directional antennas at both the Tx and Rx in our setup (Fig. 11(a)). Fig. 16 shows that moving Rx reduces BPM error from 0.2 to 0.17 at 0.5 m and from 1.1 to 1 at 1.5m, and moving both Tx and Rx further reduces the BPM error to 0.08 at 0.5m and 0.57 at 1.5m.

To move the directional antenna, we use the MUSIC algorithm [53] to calculate the direction-of-arrival of voice. As shown in Fig. 17, the DoA median error is 1.1 degree at 1 m and 5 degree at 5 m using a normal



(a) Moving Rx. (b) Moving Rx and Tx.

Fig. 16. Breathing Error due to Tx/Rx movement.

voice level. While we use directional antennas for our breathing detection to extend the range since several smart speakers [44, 45] have movement capability, other approaches to extend range can also be used such as beam-forming, increasing antenna gain, and/or using customized hardware.

Impact of Movement on Breathing Detection: To test the robustness of our breathing detection system, we ask 2 users to have small body movements (*e.g.*, swaying) or large body movements (*e.g.*, walking) at the test locations while speaking. We find that under large body movement either our breathing detection module fails or the predicted BPM is wrong over 90% cases. However, as shown in Fig. 18(a), we can detect the large movements based on the CSI amplitude change (similar to [100]) and discard the traces during that period. In comparison, the small movement at that locations only increases the breathing detection error by only 5% on average as shown in Fig. 18(b).

We use the CSI traces under movement combined with static traces to study the feasibility of movement detection, which can help in correctly selecting the traces for breathing detection. We find out that if we select absolute CSI amplitude change as 10, we can get up to 95% accuracy of detecting breathing activity.

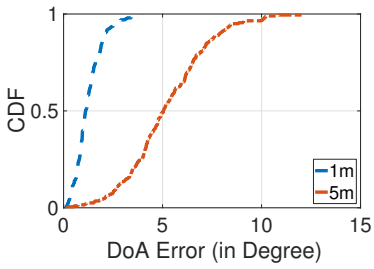


Fig. 17. CDF of MUSIC-based voice DoA error.

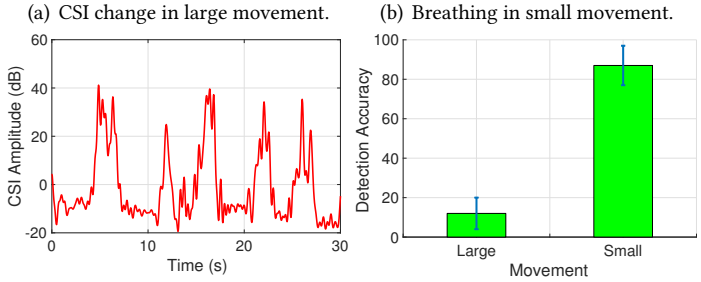
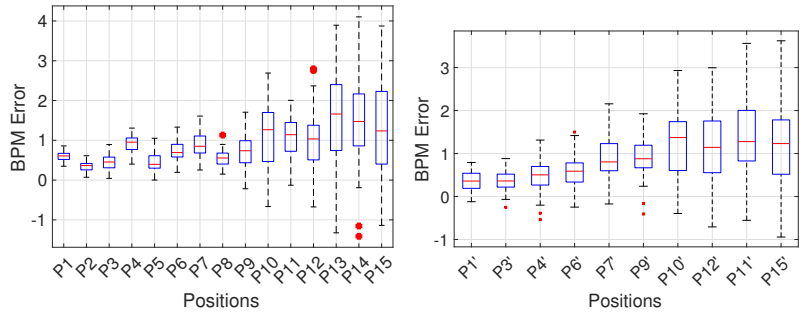


Fig. 18. Impact of Movement.

Impact of Movement on Breathing Detection: To test the robustness of our breathing detection system, we ask 2 users to have small body movements (*e.g.*, swaying) or large body movements (*e.g.*, walking) at the test locations while speaking. We find that under large body movement either our breathing detection module fails or the predicted BPM is wrong over 90% cases. However, as shown in Fig. 18(a), we can detect the large movements by thresholding the CSI amplitude change (similar to [100]) and discard the traces during that period. In comparison, the small movement at that locations only increases the breathing detection error by only 5% on average as shown in Fig. 18(b).



(a) Static Tx and Rx. (b) Moving Tx and Rx.

Fig. 19. Breathes per minute (BPM) Error at Different Positions.

We use the CSI traces under movement along with the static traces to study the feasibility of movement detection, which can help in correctly selecting the traces for breathing detection. We find out that we achieve up to 95% accuracy to detect breathing activity if we select the absolute CSI amplitude change as 10. [

6 BREATHING PATTERN DETECTION

In the next stage of REVOLT, we leverage the breathing change during speaking to determine if breathing comes from a *live speaker*. To the best of our knowledge, ours is the first that uses the breathing change during speech for live voice detection. We use WiFi CSI traces to estimate the breathing rate continuously. Combining WiFi based detection with voice based detection improves the accuracy and range of voice replay detection.

6.1 Breathing Rate Estimation

We use the setup in Fig. 15(a) to evaluate the accuracy of breathing rate estimation. Fig. 19 shows the BPM error without moving the transmitter or receiver antennas for all points except the left-most and right-most points in Fig. 15(a). Without steering the transmitter or receiver, we cannot detect breathing for the left-most and right-most points. After steering the transmitter or receiver antennas towards the person's voice, we can not only detect breathing for the left-most and right-most points, but also achieve lower BPM error as shown in Fig. 19.

We make a few observations based on the above experiments. First, the BPM error is low in all cases and can be utilized to detect if the breathing and voice are synchronized. Second, it is clear that steering both antennas yields a larger reduction in BPM error than moving only one antenna. Third, the error is lower when the user is closer to the receiver antenna. This is because when the user is closer to the receiver antenna, it affects more paths; in comparison, when the user is farther from the receiver antenna (but closer to the transmitter antenna), some of the signals reflected by the user do not reach the receiver.

6.2 Combining Breathing and Speech Pattern

The previous breathing detection checks for physical human presence to rule out remote replay. In some cases, an attacker may be physically present when replaying (*i.e.*, local replay). To protect against this kind of replay, we observe that the user that is replaying but not speaking has different breathing patterns from the user who is speaking. In particular, when a user is speaking, his breathing rate slows down [7]. As an example, Fig. 20 shows a user's voice (Fig. 20(b)), and his ground truth breathing based on the breathing monitor (Fig. 20(d)), and corresponding changes in CSI amplitude (Fig. 20(a)) and phase (Fig. 20(c)). As we can see, there is synchronized

change in all of them when the user is actually speaking. In comparison, when the user simply replays without speaking, his breathing rate and CSI remain stable as in Fig. 12(c).

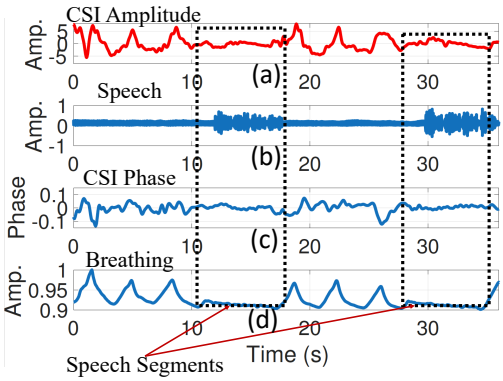


Fig. 20. Breathing Pattern Change when Speaking at 1 m.

Approach: Based on the above observation, we develop the following scheme to detect local replay.

- We segment the time based on whether voice is on or off.
- We compute the BPM for each segment using the corresponding CSI traces.
- When transitioning from the silent segment to the speaking segment, we check if the breathing rate decreases by at least a threshold (δ). When transitioning from a speaking segment to a silent segment, we also check if the breathing rate increases by at least the same threshold (δ). We determine δ based on analysis of around 1000 voice segments and breathing segments from our 8 users. Note that REVOLT can learn this threshold over time from the user during the device setup.
- We compute the number of times (M) the speech and non-speech segments match with our threshold conditioning (*i.e.*, the number of correct transitioning segments). Next, we calculate the ratio ($\frac{M}{N}$) of this correct transitioning segments (M) and total number of segments (N). If the calculated ratio is above 85%, we deem the user is actually speaking. In Table 3, we see different false positive and false negative ratios while changing the correlation ratios. It shows that too strict or too loose correlation ratio results in lower accuracy.

Evaluation: We have 10 users perform live speaking, remote replay, physical replay three times each at 5 grid positions (P2, P7, P8, P9, and P14 in Fig. 15(a)). We quantify the detection accuracy as the fraction of times that we correctly classify the user as live vs. replayed voice. Fig. 21 shows the detection accuracy as we vary the BPM change threshold δ. As we can see, the detection accuracy is highest around 90% when δ is around 8.

6.3 WiFi Spectrogram based CNN

Although the previous stage prevents the local replay but fails against a sophisticated user who also controls breathing while replaying. To prevent this replay attack, we further analyze the breathing patterns. Fig. 22(a) shows that there is subtle difference in the breathing patterns during actual speaking vs. controlled breathing. Controlled breathing is more flat than normal breathing pattern during speech. Interestingly, there is also difference in the breathing pattern across users as illustrated in Fig. 22(b) (User 1 has a decreasing pattern during speech compared to User 2's wavy pattern). Furthermore, there is a subtle variation in facial movement in the

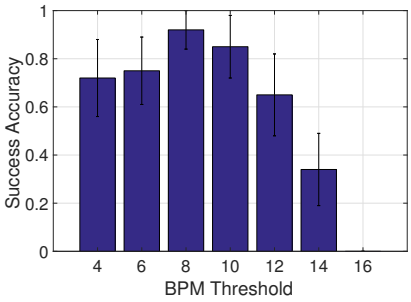
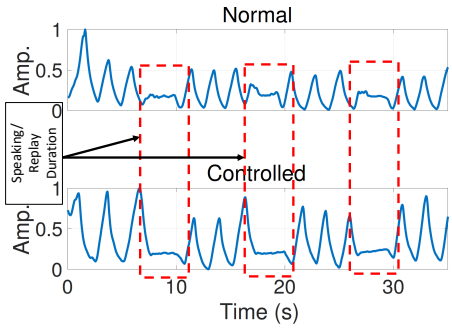


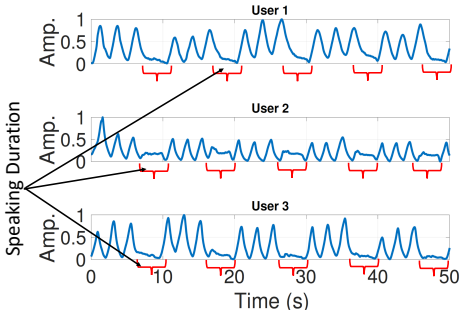
Fig. 21. Breathing-Speaking Synchronization.

Table 3. Impact of Synchronization Correlation Ratio on the Performance of REVOLT.

Ratio	FP (%)	FN (%)
0.60	**8.70**	**9.83**
0.65	**7.20**	**11.57**
0.70	**6.45**	**12.58**
0.75	**5.35**	**13.40**
0.80	**3.38**	**13.85**
0.85	**1.90**	**14.48**
0.90	**1.85**	**17.93**



(a) Live vs Controlled Breathing Pattern.



(b) Different users' Breathing Pattern during Speech.

Fig. 22. Unique Breathing Patterns during Speech and Non-speech Segments.

same utterances from different users [52]; different body sizes also affect the wireless signals [100, 103]. All these variations affect the wireless CSI traces, which makes it hard to model. Instead, we train a convolution neural network (CNN) using the phase and amplitude spectrogram of the first component of CSI PCA trace to automatically differentiate between actual speaking vs. controlled breathing as follow.

CNN Architecture Setup: We adopt a standard CNN architecture comprising of a three consecutive convolution layers with ReLu activations (of 64 block size) and followed by max-pooling layers connected to a two fully-connected feed-forward networks and ending with a single unit and a sigmoid activation. We employ 3×3 filters in each convolution operation. We use *keras* and *tensorflow* to implement this while inputting 200×200 spectrogram images of both phase and amplitude. The CSI traces are 30 second long. We use *rmsprop* optimizer with *crossentropy* loss function, 0.5 drop-out, and no regularization for this binary-classifier CNN (live vs. replayed). We train the CNN after collecting data from 10 users. Each user performs live speaking and controlled replay, and we use 100 samples from each case. We get multiple phase and amplitude spectrograms from each of the sample and train our CNN with 2100 spectrograms. While testing, we use a 5-fold cross validation of on the samples 10 users using *scikit-learn*, *keras*, and *tensorflow*. Furthermore, we later create a separate testing set from 2 users to test the effectiveness of our model.

Results: We apply our CNN to our collected data from 10 users in a 5-fold cross validation. Interestingly, even using this simple CNN, we achieved the accuracy of 75.85%. This means that this spectrogram based model can prevent at least two thirds of this sophisticated replay attack. Next, we test our model using the data from

Table 4. CNN Accuracy Trade-offs with Cost Function Change.

Cost Function (Recall;Specificity)	FP (%)	FN (%)
Type 1 (0.1;0.9)	**4.64**	**22.03**
Type 2 (0.3;0.7)	**7.96**	**18.49**
Type 3 (0.7;0.3)	**17.59**	**8.78**
Type 4 (0.9;0.1)	**21.34**	**6.45**

the two users that are not present in the training data. The accuracy is between 68.01 and 74.47. Next, we use a customized loss function, which provides different weights to recall. In this way, we can smoothly trade off between the false positive vs. the false negative. Table 4 shows the trade-offs. If we provide more weight to specificity (0.9), we can achieve false positive rate of 4.64 even under this sophisticated replay. As part of the future work, we plan to conduct more extensive training and evaluation.

7 EVALUATING END-TO-END REVOLT

So far, we have evaluated each component in REVOLT separately to differentiate between live vs. replayed voice. In this section, we examine how our overall system performs. In this *end-to-end* evaluation, we have selected two sets of parameters to support two application requirements: *Mission Critical* vs. *Casual*, where the former favors a low false positive and the latter favors a low false negative.

7.1 Evaluation Results

As mentioned in Section 3, we can associate different commands with different protection levels, which map to different thresholds used in voice and WiFi based detection. Following [99], the false positive is defined as treating replayed as live and accepting commands from an attacker, and the false negative is defined as treating live as replayed and requiring unnecessary additional authentication. It is evident that false positive is more damaging for security sensitive applications like car starting, house opening etc., whereas false negative is irritating for low-risk interactions such as querying weather query, playing music, etc.. Therefore, we select parameters in our voice and WiFi detection schemes according to different application requirements.

In our evaluation, we recruit 10 users (8 male and 2 female) between 25 and 45 year old to test our system. They interact with REVOLT by speaking a few commands (*e.g.*, *Alexa, what is the weather?*, *Alexa, can you open the house for me?*, *Alexa, can you start the car in the garage?*) while breathing normally at two different positions (P2, P7 in Fig. 15(a)). At each position, a user performs 3 runs each for live voice, local replay without controlled breathing, and sophisticated replay with controlled breathing. Altogether, we have $10 \times 2 \times 9 = 180$ runs and a total of 1800 voice segments combined with WiFi traces. To mimic remote replay, we put a Samsung S7 smartphone at the grid positions to replay the recorded audio without the user (180 runs).

Table 5. Combined Performance of REVOLT (*Critical*).

Components	FP (%)	FN (%)
After Speech based [Voice]	**2.96**	**2.94**
After Breathing Detection [WiFi]	**2.25**	**4.57**
After Breath Synch. [Voice and WiFi]	**1.05**	**7.63**
After Spectrogram CNN [WiFi] (Final)	**0.04**	**9.95**

Table 6. Combined Performance of REVOLT (*Casual*).

Components	FP (%)	FN (%)
After Speech based [Voice]	**2.96**	**2.94**
After Breathing Detection [WiFi]	**5.38**	**3.17**
After Breath Synch. [Voice and WiFi]	**7.05**	**4.63**
After Spectrogram CNN [WiFi] (Final)	**6.38**	**4.83**

Table 5 and Table 6 report the average false positive and false negative rates across users using only voice, only WiFi, and the combined method. We use the same speech model to detect replay attacks based on the voice

samples. It has a false positive rate around 2.96%. As we would expect, the voice based detection degrades as the voice SNR decreases. Most of the error comes from one user who speaks softly. The false negative rate is 2.94% as it detects a few live voice samples as replay due to background noise.

If the speech module considers voice as live, further tests will be applied. In particular, the next stage detects breathing using WiFi CSI as shown in Fig. 2. In the *Critical* setting, breathing detection reduces the overall false positive rate to 2.25% at the cost of increasing the false negative rate to 4.57% since both stages may incorrectly flag live voice as replayed. The *Casual* setting has false positive and false negative rates of 5.38% and 3.17%, respectively, by choosing a longer band-pass filter from 0.2Hz to 2.0Hz and a smaller peak detection threshold to pick up smaller breathing activity.

If both stages consider voice as live, REVOLT further analyzes voice and WiFi traces to determine if they are synchronized. This stage further removes the replayed voices. In the *Critical* setting, it reduces the overall false positive rate to 1.05% but increases the overall false negative rate to 7.63% as shown in Table 5. At this stage, most of the remote and local replay cases are flagged correctly. The error mainly comes from (i) wrongfully detecting breathing from some low frequency noise component and (ii) failing to pick up the breathing signal, which results in an increase of false negative of up to 7.63%. In the *Casual* setting, as we change the synchronization threshold from 0.85 to 0.70, we can limit the false negative rate below 4.63%. However, in this setting, the false positive percentage is 7.05% as shown in Table 6.

The final stage, which uses CNN to detect sophisticated replay attack, removes the *replay* cases further. In the *Critical* setting, REVOLT reduces the false positive rate to only 0.04% but increases the false negative rate to 9.95%. However, due to a higher recall weight (0.68) in cost function of CNN, the false negative rate in the *Casual* setting reduces to 4.83% but the false positive rate remains around 6.38%.

Table 7. CSI Processing Timing on Hummingboard.

Operation	**Size**	**Time (ms)**
FFT	500	0.16
FFT	1000	0.27
FFT	2000	0.55
FFT + Peak +Predict	500	3.62
FFT + Peak + Predict	1000	5.29
FFT + Peak + Predict	2000	8.06

Table 8. Audio Feature Extraction Timing on RPI3.

Operation	**Size (44.1K)**	**Time (s)**
Feature Extraction	1 sec	0.3
Feature Extraction	5 sec	0.9
Feature Extraction	10 sec	1.8
Extraction + Predict	1 sec	0.5
Extraction + Predict	5 sec	1.1
Extraction + Predict	10 sec	2.1

7.2 Running Time of REVOLT

We implement our system on the setup shown in Fig. 11(a). Specifically, the audio module runs on Raspberry Pi 3 model B [29], which has a 1GHz single core with 512 MB RAM, and the WiFi module runs on Hummingboard pro [15], which has 1.2 GHZ quad-core with 1GB RAM. Both audio and WiFi modules are implemented in python using scipy [28] for FFT and peak detection, librosa [17] for audio feature extraction, sklearn [34] (SVM) packages, and keras [16] with tensorflow back-end [36]. To accelerate the process of feature extraction and inference, we use the pre-compiled SVM model from our previous dataset (only 136 KB) and extract the cepstrum based features for detection. For the CNN based detection at the final stage, we use the pre-built CNN model of 19MB size on the Hummingboard.

We run each experiment 50 times, and report the average of WiFi and audio processing. As Table 7 shows, the entire processing of the 30 second CSI data with a rate of 100 samples per second using FFT size of 2000 takes only around 8.06 ms including the band-pass filtering, BPM rate estimation, and CNN based inference. Table 8 shows that the voice feature extraction and inference take within 2 second for 10 second of audio data. In general,

feature extraction and inference takes around 20% of the actual audio data duration. Since audio processing is done in parallel with the recording and does not increase the perceived delay. In addition, the MUSIC DoA estimation takes 0.3 seconds on average, and motor movement takes 0.5 seconds averaged over 20 runs.

7.3 Impact of Intelligent Audio Attack on REVOLT

We further analyze the cases where the attacker tries to mimic live voice by adding more energy to the high frequency part of the voice. We evaluate three pre-processing strategies: (i) Adding the average energy of the live voice in the 10-18 KHz frequency banks, (ii) Adding the average difference between the live vs. replayed voice from our traces in 10-18 KHz, and (iii) Adding energy drawn from the empirical distribution of the differences between the live vs. replayed voice in 10-18 KHz. We select 300 replayed voice data from our collected dataset and pre-process them using the above strategies. (i) does not increase the detection error, while the latter two strategies increase the detection error from 1.31% to 9.83% and 10.24%, respectively. The accuracy is still acceptable since there are subtle difference in both lower frequency and phase characteristics of the live vs. replayed voice, which are harder to re-produce.

8 RELATED WORK

Voice assistants are becoming ubiquitous due to its myriad use-cases [54, 82] through different voice interfaces [83]. The authors in [54, 85] show that people are using up to 30 commands a day with these voice-first gadgets as these devices are becoming more personal [85], which is a clear departure from just a few years back [55, 77]. However, the inherent open nature of voice has raised privacy and security concerns [61, 65]. To address these concerns, there are various defense approaches proposed to enhance the security of these devices.

Audio only approaches: Voice-enabled systems are vulnerable to replay attacks [57]. Recent advances in voice synthesis techniques [9, 13] make it easier to synthesize voice commands. Several machine-learning based approaches exploit spectral features [60, 67] to differentiate between live and replayed voice. Our work advances these works by (i) improving the spectral features through zooming into the frequencies where the live and replayed voice differ the most, (ii) using both frequency and phase to detect replay, and (iii) shedding light upon systematic combination of features to address time and accuracy trade-off. Furthermore, all these previous works utilize massive number of features with ensemble or neural network based techniques which are expensive to run in embedded platforms. Interestingly, the observation regarding high frequency spectral characteristics difference reported by us matches with [97]; however, they do not jointly exploit the phase structure of the audio or the wireless modality in replay prediction. Recently, a few works [88, 102] modulate voice commands to inaudible frequency that can be decoded by microphones. However, these attacks are orthogonal to our approach, which aims to prevent more prevalent and easier to launch audible channel attack. Moreover, there are a few works [51, 73, 81, 87] that focus on cepstral feature-based neural network-driven speaker recognition, which do not explicitly address the replay attack problem. As these works concentrate on finding the inherent patterns in voice spectrogram of a specific user, they do not guard against voice replay.

Wi-Fi only Approaches: A few works use Wi-Fi CSI changes to authenticate users based on activity and gait etc. [64, 100]. Their Wi-Fi based user identification is complementary but different from our work since they focus on movement based detection. [50] uses a customized multi-GHz FMCW software radio, which can detect both breathing and heart rates. [86] uses USRP platform at 2.4GHz signal to detect breathing rate. Recently, researchers have used commodity Wi-Fi to measure breathing and heart rate [75], minute walking or activity pattern [96] etc. We borrow from these works, specifically [75, 86], to detect the breathing rate with some modifications. Our innovation lies in (i) using both amplitude and phase for breathing detection and applying it to voice liveness test, and (ii) further utilizing unique breathing pattern during speaking to detect live voice.

Other Approaches: A few recent works [104, 105] look at the phoneme location or the articulatory gesture for live voice identification. However, these works require extensive training and work for only a few cm range. Another work [62] uses correlated accelerometer signature of a wearable for the authentication purpose. However, the overhead of carrying a device can be significant. Furthermore, a recent work [59] uses magnetometer signature to prevent loud-speaker based replay attack, but it only works within 6cm. [79] uses WiFi CSI traces to identify if the mouth movement correlates with voice commands to prevent remote replay attack. However, it works only within 10cm and does not prevent sophisticated replay attack. A few recent works [58, 106] also leverage breathing sound to differentiate users, but only for a very short range (*e.g.*, 3cm). Furthermore, researchers have also come up with non-voice textual command semantics based approaches [56, 72], which uses the textual patterns of commands to prevent attacks. Complimentary to these works, there is a series of silent speech recognition techniques [69, 90, 93], either using new equipments or additional information from the voice commands, to prevent voice information leakage. Our system significantly increases the range to support room-scale detection and prevent the sophisticated replay attack by intelligently leveraging both voice and WiFi modalities.

9 DISCUSSION

Our performance evaluation demonstrates the effectiveness of REVOLT on detecting voice replay attacks. There are a few avenues to further improve the robustness and generalization of our system.

First, our system can be generalized to work for homes with multiple users or pets. We can apply the approach in [50] to detect breathing from each of the users and determine if any user has breathing that is synchronized with the voice. Second, our current system works up to 2m. To further increase the range, we can use higher gain antennas or customized hardware (*e.g.*, [95]). Third, we can further diversify the datasets used for training to generalize the voice and WiFi models. Fourth, we can incorporate other physiological signals like heart rate [75] or soft biometric like gait [101] into replay detection.

10 CONCLUSION

In this paper, we develop and evaluate a voice replay detection system. Its unique advantages are wearable-free, privacy preserving, and supporting room-scale detection. It leverages the inherent differences between live vs. replayed voice and human breathing pattern during speech detected through WiFi. We show that leveraging both voice and WiFi features can detect different types of replay attacks. Our evaluation shows our system is a promising step towards securing voice-first devices.

REFERENCES

[1] 2018. 24.5M voice-first devices expected to ship this year, but apps struggle to retain users. https://techcrunch.com/2017/01/24/24-5m-voice-first-devices-expected-to-ship-this-year-but-apps-struggle-to-retain-users/.

[2] 2018. Amazon's camera-equipped Echo Look raises new questions about smart home privacy. https://techcrunch.com/2017/04/26/amazons-camera-equipped-echo-look-raises-new-questions-about-smart-home-privacy/. (2018).

[3] 2018. The ASVspoof 2017 Challenge: Assessing the Limits of Replay Spoofing Attack Detection. http://bit.ly/2Gh9AJH.

[4] 2018. Automatic Speaker Verification Spoofing and Countermeasures Challenge. http://www.asvspoof.org.

[5] 2018. Bosaris Toolkit. https://sites.google.com/site/bosaristoolkit/.

[6] 2018. Burger King's new ad forces Google Home to advertise the Whopper. http://bit.ly/2FE9jiD.

[7] 2018. CHANGES IN SPEECH AND BREATHING RATE WHILE SPEAKING AND BIKING. https://www.phonetik.uni-muenchen.de/ reichelu/publications/ICPHS1005.pdf. [Online] Last Accessed : 10/30/2018.

[8] 2018. A Cooperative Voice Analysis Repository for Speech Technologies. https://github.com/covarep/covarep.

[9] 2018. Deep Voice 3: 2000-Speaker Neural Text-to-Speech. http://research.baidu.com/deep-voice-3-2000-speaker-neural-text-speech/.

[10] 2018. Eigen3 Library. http://bit.ly/LXBsEr. [Online] Last Accessed : 10/30/2018.

[11] 2018. Google Home can now tell users apart just by their voice. https://arstechnica.com/gadgets/2017/04/google-home-gets-support-for-multiple-users/.

[12] 2018. Google's speech recognition technology now has a 4.9% word error rate. https://venturebeat.com/2017/05/17/googles-speech-recognition-technology-now-has-a-4-9-word-error-rate/.
[13] 2018. High-fidelity speech synthesis with WaveNet. https://deepmind.com/blog/high-fidelity-speech-synthesis-wavenet/.
[14] 2018. Human Voice Frequency Range. http://www.seaindia.in/blog/human-voice-frequency-range/.
[15] 2018. Hummingboard Pro. https://www.solid-run.com/product-tag/hummingboard-pro/. [Online] Last Accessed : 10/30/2018.
[16] 2018. Keras: The Python Deep Learning Library. https://keras.io.
[17] 2018. Librosa: Audio and Music Processing in Python. https://librosa.github.io.
[18] 2018. libSVM Library. https://www.csie.ntu.edu.tw/ cjlin/libsvm/.
[19] 2018. Linux 802.11n CSI Tool. https://dhalperi.github.io/linux-80211n-csitool/. [Online] Last Accessed : 10/30/2018.
[20] 2018. Logitech Speaker System with Z323. https://www.amazon.com/Logitech-Speaker-System-Z323-Subwoofer/dp/B002FU5QM0.
[21] 2018. Lyrebird allows you to create a digital voice that sounds like you with only one minute of audio. https://lyrebird.ai.
[22] 2018. Matrix Creator IO Board. https://www.matrix.one/products/creator. [Online] Last Accessed : 10/30/2018.
[23] 2018. Measuring, refining and calibrating speaker and language information extracted from speech. http://bit.ly/2DhmhRD.
[24] 2018. MOKCAO STYLE Bluetooth Speaker. https://www.amazon.com/Bluetooth-Speakers-Waterproof-Christmas-Gift-Black/dp/B072BHWYVJ.
[25] 2018. Neulog USB Module. https://neulog.com/usb/. [Online] Last Accessed : 10/30/2018.
[26] 2018. Psychoacoustics. https://en.wikipedia.org/wiki/Psychoacoustics. [Online] Last Accessed : 10/30/2018.
[27] 2018. Python PeakUtils. https://pypi.python.org/pypi/PeakUtils. [Online] Last Accessed : 10/30/2018.
[28] 2018. Python Scipy Package. https://www.scipy.org. [Online] Last Accessed : 10/30/2018.
[29] 2018. Raspberry PI3 Model B. https://www.raspberrypi.org/products/raspberry-pi-3-model-b/. [Online] Last Accessed : 10/30/2018.
[30] 2018. Research Finds Automated Voice Imitation Can Fool Humans and Machines. https://cacm.acm.org/careers/192367-research-finds-automated-voice-imitation-can-fool-humans-and-machines/fulltext.
[31] 2018. Respiration Monitor Belt logger sensor NUL-236. https://neulog.com/respiration-monitor-belt/. [Online] Last Accessed : 10/30/2018.
[32] 2018. Room Sound Example from House-hold. https://freesound.org/people/klankbeeld/sounds/363216/.
[33] 2018. Samsung Galaxy S7. https://www.amazon.com/Samsung-Galaxy-S7-SM-G930F-Smartphone/dp/B01CJSF8IO.
[34] 2018. SKLearn libSVM. http://bit.ly/2p5fFRU.
[35] 2018. Source-Filter Model. http://bit.ly/2HZkDYt.
[36] 2018. Tensorflow: An open source machine learning framework for everyone. https://www.tensorflow.org.
[37] 2018. Tupavco TP542 Panel WiFi Antenna. www.amazon.com/Tupavco-TP542-Panel-WiFi-Antenna/dp/B015QF7EMK/. [Online] Last Accessed : 10/30/2018.
[38] 2018. VLFeat. http://www.vlfeat.org.
[39] 2018. Voice-First Devices Are The Next Big Thing – Here's Why. www.forbes.com/sites/ilkerkoksal/2018/02/01/voice-first-devices-are-the-next-big-thing-heres-why/.
[40] 2019. Cyber Acoustics CA-3602FFP 2.1 Speaker Sound System with Subwoofer. https://www.amazon.com/gp/product/B00BXF5HQ8/.
[41] 2019. Fifine USB Microphone. https://www.amazon.com/Microphone-Condenser-Recording-Streaming-669B/dp/B06XCKGLTP/.
[42] 2019. iPad Pro. https://www.apple.com/shop/buy-ipad/ipad-pro/12.9-inch-display-256gb-space-gray-wifi.
[43] 2019. JBL Flip4 Bluetooth Speaker. https://www.amazon.com/JBL-Portable-Bluetooth-Wireless-Protective/dp/B074WGD1JL/.
[44] 2019. JIBO Social Assistant Robot. https://ideaing.com/product/jibo-robot.
[45] 2019. LG CLOI Home. https://www.lg.com/global/lg-thinq-appliances/cloi.
[46] 2019. MacBook Pro (15-inch, 2017). https://apple.co/2He4xvs.
[47] 2019. Tonor Dynamic Karaoke Microphone. https://www.amazon.com/gp/product/B01ISNU3X4/.
[48] 2019. Xiaomi Mi 8. https://www.gadgetsnow.com/mobile-phones/Xiaomi-Mi-8.
[49] Heba Abdelnasser, Khaled A. Harras, and Moustafa Youssef. 2015. UbiBreathe: A Ubiquitous non-Invasive WiFi-based Breathing Estimator. In *MobiHoc '15*. 277–286.
[50] Fadel Adib, Hongzi Mao, Zach Kabelac, Dina Katabi, and Robert C. Miller. 2015. Smart Homes That Monitor Breathing and Heart Rate. In *Proc. of CHI*.
[51] K. S. Ahmad, A. S. Thosar, J. H. Nirmal, and V. S. Pande. 2015. A unique approach in text independent speaker recognition using MFCC feature sets and probabilistic neural network. In *2015 Eighth International Conference on Advances in Pattern Recognition (ICAPR)*. 1–6.
[52] Amr Alanwar, Bharathan Balaji, Yuan Tian, Shuo Yang, and Mani Srivastava. 2017. EchoSafe: Sonar-based Verifiable Interaction with Intelligent Digital Agents. In *SafeThings'17*. ACM, New York, NY, USA, 38–43.
[53] M. A. Alrmah, S. Weiss, and S. Lambotharan. 2011. An extension of the MUSIC algorithm to broadband scenarios using a polynomial eigenvalue decomposition. In *2011 19th European Signal Processing Conference*. 629–633.
[54] Frank Bentley, Chris Luvogt, Max Silverman, Rushani Wirasinghe, Brooke White, and Danielle Lottridge. 2018. Understanding the Long-Term Use of Smart Speaker Assistants. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3 (2018), 91:1–91:24.

[55] A.J. Bernheim Brush, Bongshin Lee, Ratul Mahajan, Sharad Agarwal, Stefan Saroiu, and Colin Dixon. 2011. Home Automation in the Wild: Challenges and Opportunities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, 2115–2124.

[56] Giovanni Campagna, Silei Xu, Rakesh Ramesh, Michael Fischer, and Monica S. Lam. 2018. Controlling Fine-Grain Sharing in Natural Language with a Virtual Assistant. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3 (2018), 95:1–95:28.

[57] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. 2016. Hidden voice commands. In *25th USENIX Security Symposium (USENIX Security 16), Austin, TX*.

[58] Jagmohan Chauhan, Yining Hu, Suranga Seneviratne, Archan Misra, Aruna Seneviratne, and Youngki Lee. 2017. BreathPrint: Breathing Acoustics-based User Authentication. In *ACM MobiSys '17*. 278–291.

[59] S. Chen, K. Ren, S. Piao, C. Wang, Q. Wang, J. Weng, L. Su, and A. Mohaisen. 2017. You Can Hear But You Cannot Steal: Defending Against Voice Impersonation Attacks on Smartphones. In *ICDCS 2017*. 183–195.

[60] Zhuxin Chen, Zhifeng Xie, Weibin Zhang, and Xiangmin Xu. 2017. ResNet and Model Fusion for Automatic Spoofing Detection. In *Interspeech 2017*. ISCA, 102–106.

[61] Eugene Cho. 2019. Hey Google, Can I Ask You Something in Private?. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, 258:1–258:9.

[62] Huan Feng, Kassem Fawaz, and Kang G. Shin. 2017. Continuous Authentication for Voice Assistants. In *ACM MobiCom '17*. ACM, New York, NY, USA, 343–355. https://doi.org/10.1145/3117811.3117823

[63] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. 2011. Tool release: gathering 802.11n traces with channel state information. *Computer Communication Review* 41 (2011), 53.

[64] Feng Hong, Xiang Wang, Yanni Yang, Yuan Zong, Yuliang Zhang, and Zhongwen Guo. 2016. WFID: Passive Device-free Human Identification Using WiFi Signal. In *MOBIQUITOUS 2016*. 47–56.

[65] Jason I. Hong and James A. Landay. 2004. An Architecture for Privacy-sensitive Ubiquitous Computing. In *Proceedings of the 2Nd International Conference on Mobile Systems, Applications, and Services (MobiSys '04)*. 177–189.

[66] David M. Howard and James Angus. 2000. *Acoustics and Psychoacoustics* (2nd ed.). Butterworth-Heinemann, Newton, MA, USA.

[67] Zhe Ji, Zhi-Yi Li, Peng Li, MaoBo An, Shengxiang Gao, Dan Wu, and Faru Zhao. 2017. Ensemble Learning for Countermeasure of Audio Replay Spoofing Attack in ASVspoof2017. In *Interspeech 2017*. 87–91.

[68] R. C. Johnson, T. E. Boult, and W. J. Scheirer. 2013. Voice authentication using short phrases: Examining accuracy, security and privacy issues. In *2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*. 1–8.

[69] Naoki Kimura, Michinari Kono, and Jun Rekimoto. 2019. SottoVoce: An Ultrasound Imaging-Based Silent Speech Interaction Using Deep Neural Networks. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, 146:1–146:11.

[70] T. Kinnunen, M. Sahidullah, M. Falcone, L. Costantini, R. G. Hautamäki, D. Thomsen, A. Sarkar, Z. H. Tan, H. Delgado, M. Todisco, N. Evans, V. Hautamäki, and K. A. Lee. 2017. RedDots replayed: A new replay spoofing attack corpus for text-dependent speaker verification research. In *ICASSP 2017*. 5395–5399.

[71] Manikanta Kotaru, Kiran Joshi, Dinesh Bharadia, and Sachin Katti. 2015. Spotfi: Decimeter level localization using WiFi. In *ACM SIGCOMM Computer Communication Review*, Vol. 45(4). ACM, 269–282.

[72] 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 *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, 598:1–598:12.

[73] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren. 2014. A novel scheme for speaker recognition using a phonetically-aware deep neural network. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 1695–1699. https://doi.org/10.1109/ICASSP.2014.6853887

[74] P. L. De Leon, I. Hernaez, I. Saratxaga, M. Pucher, and J. Yamagishi. 2011. Detection of synthetic speech for the problem of imposture. In *ICASSP 2011*. 4844–4847.

[75] Jian Liu, Yan Wang, Yingying Chen, Jie Yang, Xu Chen, and Jerry Cheng. 2015. Tracking Vital Signs During Sleep Leveraging Off-the-shelf WiFi. In *MobiHoc '15*. ACM, New York, NY, USA, 267–276.

[76] Li Lu, Jiadi Yux, Yingying Cheny, Hongbo Liuz, Yanmin Zhu, Yunfei Liu, and Minglu Li. [n. d.]. LipPass: Lip Reading-based User Authentication on Smartphones Leveraging Acoustic Signals. In *INFOCOM 2018*.

[77] Ewa Luger and Abigail Sellen. 2016. "Like Having a Really Bad PA": The Gulf Between User Expectation and Experience of Conversational Agents. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, 5286–5297.

[78] Judith A. Markowitz. 2000. Voice Biometrics. *Commun. ACM* 43, 9 (2000), 66–73.

[79] Yan Meng, Zichang Wang, Wei Zhang, Peilin Wu, Haojin Zhu, Xiaohui Liang, and Yao Liu. 2018. WiVo: Enhancing the Security of Voice Control System via Wireless Signal in IoT Environment. In *Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing (Mobihoc '18)*. 81–90.

[80] Lindasalwa Muda, Mumtaj Begam, and I. Elamvazuthi. 2010. Voice Recognition Algorithms using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) Techniques. *CoRR* abs/1003.4083 (2010).

[81] K. Sri Rama Murty and Bayya Yegnanarayana. 2006. Combining evidence from residual phase and MFCC features for speaker recognition. *IEEE Signal Processing Letters* 13 (2006), 52–55.

[82] Jennifer Pearson, Simon Robinson, Thomas Reitmaier, Matt Jones, Shashank Ahire, Anirudha Joshi, Deepak Sahoo, Nimish Maravi, and Bhakti Bhikne. 2019. StreetWise: Smart Speakers vs Human Help in Public Slum Settings. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, 96:1–96:13.

[83] Martin Porcheron, Joel E. Fischer, Stuart Reeves, and Sarah Sharples. 2018. Voice Interfaces in Everyday Life. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, 640:1–640:12.

[84] P. Pujol, D. Macho, and C. Nadeu. 2006. On Real-Time Mean-and-Variance Normalization of Speech Recognition Features. In *IEEE ICASSP 2006*, Vol. 1. I–I. https://doi.org/10.1109/ICASSP.2006.1660135

[85] Amanda Purington, Jessie G. Taft, Shruti Sannon, Natalya N. Bazarova, and Samuel Hardman Taylor. 2017. "Alexa is My New BFF": Social Roles, User Satisfaction, and Personification of the Amazon Echo. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. ACM, 2853–2859.

[86] Ruth Ravichandran, Elliot Saba, Ke-Yu Chen, Mayank Goel, Sidhant Gupta, and Shwetak N. Patel. 2015. WiBreathe: Estimating respiration rate using wireless signals in natural settings in the home. In *PerCom*. IEEE Computer Society, 131–139.

[87] Fred Richardson, Douglas A. Reynolds, and Najim Dehak. 2015. A Unified Deep Neural Network for Speaker and Language Recognition. *CoRR* abs/1504.00923 (2015). arXiv:1504.00923 http://arxiv.org/abs/1504.00923

[88] Nirupam Roy, Haitham Hassanieh, and Romit Roy Choudhury. 2017. BackDoor: Making Microphones Hear Inaudible Sounds. In *ACM MobiSys*.

[89] Jiacheng Shang and Jie Wu. 2016. Fine-grained Vital Signs Estimation Using Commercial Wi-fi Devices. In *S3 Workshop*. ACM, New York, NY, USA, 30–32.

[90] Ke Sun, Chun Yu, Weinan Shi, Lan Liu, and Yuanchun Shi. 2018. Lip-Interact: Improving Mobile Device Interaction with Silent Speech Commands. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. ACM, 581–593.

[91] J. Tan, C. T. Nguyen, and X. Wang. 2017. SilentTalk: Lip reading through ultrasonic sensing on mobile phones. In *IEEE INFOCOM 2017*. 1–9.

[92] Jiayao Tan, Xiaoliang Wang, Cam-Tu Nguyen, and Yu Shi. 2018. SilentKey: A New Authentication Framework Through Ultrasonic-based Lip Reading. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 1 (2018), 36:1–36:18.

[93] Jiayao Tan, Xiaoliang Wang, Cam-Tu Nguyen, and Yu Shi. 2018. SilentKey: A New Authentication Framework Through Ultrasonic-based Lip Reading. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 1 (2018), 36:1–36:18.

[94] Ingo R. Titze. 1994. *Principles of Voice Production.* Prentice Hall.

[95] P. Wang, B. Guo, T. Xin, Z. Wang, and Z. Yu. 2017. TinySense: Multi-user respiration detection using Wi-Fi CSI signals. In *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*. 1–6.

[96] Yan Wang, Jian Liu, Yingying Chen, Marco Gruteser, Jie Yang, and Hongbo Liu. 2014. E-eyes: Device-free Location-oriented Activity Identification Using Fine-grained WiFi Signatures. In *ACM MobiCom '14*. 617–628.

[97] Marcin Witkowski, Stanislaw Kacprzak, Piotr Zelasko, Konrad Kowalczyk, and Jakub Galka. 2017. Audio Replay Attack Detection Using High-Frequency Features. In *Interspeech 2017*. 27–31.

[98] Zhizheng Wu, Nicholas Evans, Tomi Kinnunen, Junichi Yamagishi, Federico Alegre, and Haizhou Li. 2015. Spoofing and Countermeasures for Speaker Verification. *Speech Commun.* 66, C (2015), 130–153.

[99] Zhizheng Wu, Junichi Yamagishi, Tomi Kinnunen, Cemal Hanilçi, Md. Sahidullah, Aleksandr Sizov, Nicholas W. D. Evans, and Massimiliano Todisco. 2017. ASVspoof: The Automatic Speaker Verification Spoofing and Countermeasures Challenge. *J. Sel. Topics Signal Processing* 11, 4 (2017), 588–604.

[100] Yunze Zeng, Parth H. Pathak, and Prasant Mohapatra. 2016. WiWho: Wifi-based Person Identification in Smart Spaces. In *IPSN '16*. IEEE Press, Piscataway, NJ, USA.

[101] Y. Zeng, P. H. Pathak, and P. Mohapatra. 2016. WiWho: WiFi-Based Person Identification in Smart Spaces. In *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 1–12.

[102] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. 2017. DolphinAttack: Inaudible Voice Commands. In *CCS*.

[103] J. Zhang, B. Wei, W. Hu, and S. S. Kanhere. 2016. WiFi-ID: Human Identification Using WiFi Signal. In *DCOSS 2016*. 75–82.

[104] Linghan Zhang, Sheng Tan, and Jie Yang. [n. d.]. Hearing Your Voice is Not Enough: An Articulatory Gesture Based Liveness Detection for Voice Authentication. In *CCS 2017*.

[105] Linghan Zhang, Sheng Tan, Jie Yang, and Yingying Chen. 2016. VoiceLive: A Phoneme Localization Based Liveness Detection for Voice Authentication on Smartphones. In *ACM CCS '16*. 1080–1091.

[106] Wenbo Zhao, Yang Gao, and Rita Singh. 2017. Speaker identification from the sound of the human breath. *CoRR* abs/1712.00171 (2017).