This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

Privacy-preserving Liveness Detection for Securing Smart Voice Interfaces

Yan Meng, Jiachun Li, Haojin Zhu, *Fellow, IEEE*, Yuan Tian, and Jiming Chen, *Fellow, IEEE*

Abstract—Smart speakers are widely used as the primary user interface in intelligent systems, including smart homes and industrial IoT. However, they are vulnerable to voice spoofing attacks which result in malicious command execution or privacy information leakage. Passive liveness detection, which thwarts voice spoofing via analyzing the collected audio rather than deploying sensors to distinguish between live-human and spoofing voices, has drawn increasing attention. But existing schemes either face performance degradation under environmental factor changes or require the user to keep fixed gestures, which limit their deployment in real-world scenarios. Besides, the space distributed property of smart speakers causes building a universal classifier for all involved users to be cumbersome and increases privacy leakage issues. To address the challenges mentioned above, we propose LIVEARRAY, an efficient, lightweight, and privacy-preserving passive liveness detection system. LIVEARRAY exploits a novel liveness feature, array fingerprint, which utilizes the microphone array inherently adopted by the smart speaker to improve the accuracy of liveness detection. LIVEARRAY's further employs the federated learning-based architecture to reduce the dataset collection overhead during classifier building and eliminate the potential privacy leakage during data transmission. Experimental results show that LIVEARRAY achieves an accuracy of 99.16%, which is superior to existing passive schemes.

Index Terms—Liveness detection, voice interface, smart speakers, multi-channel audio.

1 INTRODUCTION

SMART speakers enabling voice assistants play a crucial role in current popular networking systems of artificial intelligence (*e.g.*, smart home, smart vehicle). Smart Speakers not only serve as the hub communicating wireless smart devices (*e.g.*, smart lighter, smart locker, smart thermostat) but also become the main user interface which allow users to conduct various actions (*e.g.*, remotely control devices, query personal information, order ships online) as long as the user's voice can be heard. However, the inherent broadcast nature of the voice channel unlocks a door for adversaries to conduct *voice spoofing* attacks in which malicious commands are injected into smart speakers. The most representative attack is the replay attack [1] which spoofs the voice interface via replaying the pre-collected voice commands. Besides replay attacks, the smart speaker is vulnerable to inaudible ultrasound-based attacks (*e.g.*, Dolphin attack [2], BackDoor attack [3]) and emerging adversarial attacks (*e.g.*, hidden voice [4], CommanderSong [5], user impersonations [6]) due to the hardware's non-linearity property and flaws in the speech processing algorithms. Voice spoofing imposes severe safety issues (*e.g.*, deliberately turning on the smart thermostat [7]) and privacy risks (*e.g.*, querying user's schedule information) on the smart speaker and therefore cause great concern.

Liveness detection has become the mainstream anti-voice spoofing method. It is based on the prevalent fact: voice commands in the spoofing attack are emitted by electrical devices (*e.g.*, high-quality loudspeaker [8], ultrasonic dynamic speaker [2]) instead of the real human mouth motion. Existing liveness detection solutions could be divided into *multi-factor authentication* and *passive liveness detection scheme*. The former [9], [10], [11], [12], [13], [14] combines the collected audio and additional physical quantity (*e.g.*, acceleration [15], electromagnetic field [16], mm-Wave [17], ultrasound [18], Wi-Fi [19]) to detect the machine-generated voice spoofing without involving authentic human mouth movement. However, to capture the liveness factor of a real human, multi-factor authentication either requires the user to carry specialized sensors (*e.g.*, accelerator, magnetometer) or actively emits probe signals (*e.g.*, ultrasounds, wireless signals), which adds additional burdens for users [20]. By contrast, the passive scheme only leverages the audio data collected by the smart speaker itself. Its key insight is that the difference in articulatory manners between real humans (*i.e.*, vocal vibration and mouth movement) and electrical machines (*i.e.*, diaphragm vibration) will result in subtle but significant differences in the collected audios' spectrograms. Passive schemes based on mono audio [21], [22] and two-channel audio [23], [24] have already been proposed and could be directly incorporated into the smart speaker's software level. Now, the passive scheme attracts increasing attention due to its convenience and easy-deployment properties.

Although the passive scheme attracts increasing attention, it faces a series of challenges in *efficiency*, *deployability*, and *privacy*, which seriously hinder its deployment, espe-

- *Yan Meng, Jiachun Li, and Haojin Zhu are with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, 200240. (Email: yan_meng@sjtu.edu.cn; jiachunli@sjtu.edu.cn; zhu-hj@cs.sjtu.edu.cn)*
- *Yuan Tian is with the Electrical and Computer Engineering Department and the Institute for Technology, Law and Policy (ITLP), University of California, Los Angeles, CA, USA, 90095. (Email: yuant@ucla.edu)*
- *Jiming Chen is with the State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China. (e-mail: cjm@zju.edu.cn)*
- *Haojin Zhu is the corresponding author.*

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

cially in the IoT scenarios with distributed smart speakers. *(i) Efficiency challenge:* passive schemes leveraging sub-bass low-frequency area (20~300 Hz in [23]) or voice area (below 10 kHz in [22]) of the mono audio spectrum as the liveness factor are vulnerable to both the change of sound propagation channel and the spectrum modulated attack [8]. The two-channel audio's *fieldprint*-based scheme [24] requires the user to keep a fixed manner to maintain its liveness detection performance, thus, deploying it in many complex scenarios (*e.g.*, user walkings or gesture changes) is challenging. *(ii) Training burden:* the existing passive schemes are designed following a centralized style in which several users use only one smart speaker. For a system deploying multiple smart speakers which are dispersed in space (*e.g.*, an industrial IoT system with multiple smart speakers and each smart speaker has a limited number of users), to train the classifier which are effective for all users, training an effective classifier for all users requires merging the authentic and spoofing data collected by various smart speakers, which is a cumbersome task. *(iii) Privacy risk:* in the scenario with distributed smart speakers, training a uniform classifier requires uploading the original voice data to the centralized platform (*e.g.*, cloud), which inevitably causes privacy leakage risks since the voice contains sensitive information such as voice prints and sensitive commands.

Therefore, it is desirable to propose a novel passive and distributed liveness detection scheme with the following three merits. *(i) Resilient to environmental changes:* it is robust to the dynamic sound propagation channel and user's movement. *(ii) Light burden:* for systems containing distributed smart speakers, it does not require collecting and processing audios from different smart speakers to generate the classifier in a centralized manner. *(iii) Privacy-preserving:* it ensures only non-sensitive elements are transmitted during the whole liveness detection process.

Research motivations. To achieve a robust, lightweight, and privacy-preserving passive liveness detection, in this study, we propose LIVEARRAY, a microphone array-based liveness detection framework. On the one hand, to improve the detection accuracy, we leverage the observation that mainstream smart speakers widely adopt microphone arrays (*e.g.*, Amazon Echo 3rd Gen [25] and Google Home Max [26] have six microphones). Since the collected audio has increased diversity among different channels due to different positions of microphones in the array, LIVEARRAY to extract a more useful liveness factor from it. On the other hand, to light the burden on centralized collecting training samples from various smart speakers and achieving privacy-preserving, LIVEARRAY can leverage the *federated learning (FL)* paradigm when building classifiers conducting liveness detection tasks. In this paradigm, the individual smart speaker has its classifier trained with the audios from its corresponding users. The classifier then upgrades its detection ability via communicating with classifiers of other smart speakers. Only non-sensitive parameters are transmitted during the classifier update, and the final liveness detection process is done locally. Thus the privacy information related to the original voice samples will not be leaked.

Technical challenges. In order to implement LIVEARRAY in real-world scenarios, this study needs to address three key challenges: *(i)* Theoretically, what is the advantage of adopting a microphone array compared with a single microphone? *(ii)* How to select the appropriate features from the multi-channel audios and design an FL framework suitable for distributed smart speakers are still changeable. *(iii)* Considering that LIVEARRAY is the first to leverage the microphone array and FL architecture for liveness detection, how can we demonstrate its effectiveness and robustness?

To overcome the above three challenges, we first build a sound propagation model and then leverage it to theoretically assess the impact of environmental factors (*e.g.*, articulatory gesture, propagation path) on the collected audio's spectrum. Secondly, for the multi-channel audio collected by the microphone array, we give a formal definition of *array fingerprint*, which leverages the differences among different channels' data to eliminate the distortions caused by factors including air channel and user's position changes. Then, for the IoT scenario with distributed smart speakers, we design FL-based classifier training and a liveness detection scheme. Finally, to evaluate the effectiveness of LIVEARRAY, we collect and build an array fingerprint-based dataset containing 38,720 multi-channel voices from 20 different users. LIVEARRAY achieves the liveness detection accuracy of 99.16% in our dataset and is robust to the various factors (*e.g.*, distance, angle, movement). We also compare LIVEARRAY with existing passive schemes (*i.e.*, CAFIELD [24], and VOID [22]) on both our dataset and a third-party ReMasc Core dataset [27]. The experimental results well demonstrate that the performance of LIVEARRAY is superior to existing schemes.

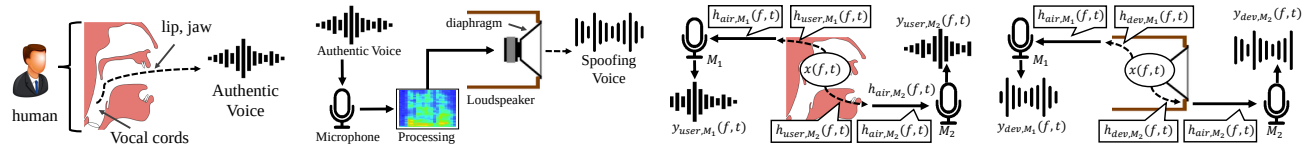
In summary, this study makes the following contributions:

- *Novel system.* Novel system. We design, implement, and evaluate LIVEARRAY for thwarting voice spoofing attacks. By only using audio collected from a smart speaker, LIVEARRAY does not require the user to carry any device or conduct additional action. LIVEARRAY is suitable for real-world cases with distributed smart speakers and does not collect audio from all users in a centralized and cumbersome way.
- *Effective features and promising performance.* We give a theoretical analysis of principles behind passive detection and propose a robust liveness feature: the array fingerprint. Experimental results on both our and third-party datasets show that LIVEARRAY outperforms existing schemes. Experimental results on both our dataset and a third-party dataset show that LIVEARRAY outperforms existing schemes. This novel feature enhances effectiveness and broadens passive liveness detection application scenarios.
- *Robust and privacy-preserving properties.* We evaluate multiple factors and demonstrate the robustness of LIVEARRAY. Besides, with an FL-based scheme, LIVEARRAY can preserve the sensitive information of the user's voice samples.

The rest of this paper is organized as follows. In Section 2 and Section 3, we introduce the preliminaries of this study and formulate the research problem respectively. In Section 3.3, we propose the concept of the array fingerprint and proof its advantages through both theoretical analysis

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833



(a) Authentic voice generation. (b) Spoofing voice generation. (c) Authentic voice propagation. (d) Spoofing voice propagation.

Fig. 1. Voice command generation and propagation processes.

and experiments. We elaborate on the detailed design of LIVEARRAY in Section 4, which is followed by evaluation, discussion, and related work in Sections 5, 6, respectively. Finally, we conclude this paper in Section 7.

2 PRELIMINARIES AND RELATED WORKS

This section introduces the necessary preliminaries related to LIVEARRAY and reviews related research works.

2.1 Voice Spoofing Attacks Faced by Smart Speakers

For smart speakers widely deployed in networking systems of artificial intelligence, they face threats from voice spoofing attacks, which can be categorized into the following two types.

Classical replay attack. In this attack, to fool the voice assistance, the adversary collects the authentic user's voice command samples and then plays them back using a high-quality loudspeaker [1]. Capturing the victim's voice samples can be done in various ways, such as daily talking, phone calls, and social engineering methods.

Advanced spoofing attack. Besides replay attacks, when there is a limited number of victim's voice samples, by leveraging the latest voice synthesized technique [1], [28], [29], [30], [31], the adversary can spoof speech recognition and speaker verification systems. For instance, the adversary can leverage inaudible signals (*e.g.*, ultrasonic and laser [2], [3], [32], [33], [34]) or craft subtle noises into the audio (*i.e.*, adversarial example attacks [4], [5], [6], [35], [36], [37], [38], [39], [40] to spoof the voice interface without incurring the victim's perception [2], [3], [32], [33], [34]. Moreover, by carefully modifying the spectrum of spoofing audio, the modulated attack [8] proposed by Wang *et al.* demonstrates the feasibility of bypassing existing mono audio-based passive liveness detection schemes [21].

Among various spoofing attack approaches, the replay attack is the most effective since it preserves the most comprehensive voiceprint, sounds natural, and requires no cumbersome hardware configurations and software parameter fine-tuning. Thus, in the rest of this paper, we choose the replay attack and its advanced variant (*i.e.*, modulated attack) as typical examples of voice spoofing attacks.

2.2 Voice Command Generation and Propagation

Before reviewing existing passive liveness detection schemes, it is important to describe the generation and propagation processes of authentic and spoofing voice commands.

Voice command generation. Authentic and spoofing voice commands are generated by the human or the electrical device (*i.e.*, loudspeaker, tablet, smartphone). As shown in Fig. 1(a), when a user speaks a voice command, his/her mouth and lips modulate the air to generate an authentic voice. However, as illustrated in Fig. 1(b), the spoofing command generation follows another procedure. In this case, the adversary first collects the voice from the authentic user using a microphone. Then, to enhance the attack ability, the collected audio may be further processed (*e.g.*, increasing the volume or adding subtle adversarial examples). Finally, the loudspeaker utilizes the electromagnetic field change to vibrate the diaphragm. The movement of the diaphragm suspends and pushes air to generate the sound wave corresponding to the spoofing command.

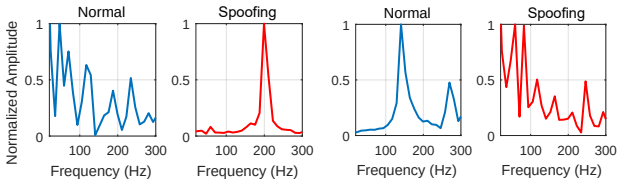
Voice command propagation. After the voice command is generated, the sound is transmitted to the microphone of the voice interface. To achieve a better sound recording performance, current smart speakers usually have a microphone array (*e.g.*, Amazon Echo 3rd Gen [25] and Google Home Max [26] both have 6 microphones). As shown in Fig. 1(c) and Fig. 1(d), for different microphones, the sound propagation channels and the corresponding received audios will be different. We take spoofing voice propagation as an example. Given a loudspeaker, we denote the signal before voice generation as $x(f,t)$, where f represents the frequency and t is time. When $x(f,t)$ spreads and is collected by the first microphone M_1, it undergoes two parts of channel gains: $h_{dev,M_1}(f,t)$ and $h_{air,M_1}(f,t)$, where $h_{dev,M_1}(f,t)$ represents the modulation gain during the voice generation and $h_{air,M_1}(f,t)$ is the gain of the air channel. The audio collected by M_1 can be represented as $y_{dev,M_1}(f,t) = x(f,t) \cdot h_{dev,M_1}(f,t) \cdot h_{air,M_1}(f,t)$. Then, for the second microphone M_2, due to the change of the audio transmission path, the signal gains during the voice generation and sound transmission in the air will be changed as $h_{dev,m2}(f,t)$ and $h_{air,m2}(f,t)$ respectively. Similarly, when the authentic voice command is transmitted from the user to the microphone M_1, if the modulation gain from the mouth motion and the air channel gain are denoted as $h_{user,M_1}(f,t)$ and $h_{air,M_1}(f,t)$ respectively, the collected audio can be represented as $y_{user,M_1}(f,t) = x(f,t) \cdot h_{user,M_1} \cdot h_{air,M_1}(f,t)$[1]. The audios collected by M_2 can be denoted as $y_{user,m2}(f,t) = x(f,t) \cdot h_{user,m2} \cdot h_{air,m2}(f,t)$.

Sound processing within the smart speaker. Since the microphones equipped in the mainstream smart speakers usually have a flat frequency response curve in the fre-

1. In the real-world scenario, there is no such $x(f,t)$ during authentic voice generation process. However, the concepts of $x(f,t)$ and $h_{user,M_1}(f,t)$ are widely used [21] and can help us understand our motivations in Section 3.3.

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833



(a) Spectrums when the microphone is at location A. (b) Spectrums when the microphone is at location B.

Fig. 2. Spectrums when the microphone locates at different locations.

quency area of the human voice, similar to existing studies [24], we assume the smart speaker saves the original sensed audio data to an electrical signal. Finally, the speech recognition module deployed in the smart home cloud (*e.g.*, Amazon Echo, Google Home) or the local smart speaker devices (*e.g.*, CMUSphinx [41], Kaldi [42]) interprets the collected audio to the voice command and further influences the actions of smart devices.

2.3 Passive Liveness Detection Models

As illustrated in Fig. 1, given a microphone M_1, the different sound generation principles between authentic human and electrical spoofing device result in two different voice generation gains: $h_{user,M_1}(f,t)$ and $h_{dev,M_1}(f,t)$. The different gains will be reflected in the collected authentic and spoofing audios, which could be used to conduct passive liveness detection. Existing schemes are divided into two categories: mono channel-based detection (*e.g.*, Sub-bass [21], VOID [22], Pop-based schemes [43], [44]) and dual channel-based detection (*i.e.*, CAFIELD [24]).

2.3.1 Mono Channel-based Detection

Principles. As shown in Fig. 1(c) and Fig. 1(d), if ignoring the distortion in the sound signal transmission, the air channel's gain (*e.g.*, $h_{air,M_1}(f,t)$) could be considered as a constant value A. Thus, audio samples received by the first microphone M_1 in authentic and spoofing attack scenarios are $y_{user,M_1}(f,t) = A \cdot x(f,t) \cdot h_{user,M_1}(f,t)$ and $y_{dev,M_1}(f,t) = A \cdot x(f,t) \cdot h_{dev,M_1}(f,t)$, respectively. Since A and $x(f,t)$ are the same, it means that the spectrograms of the received audio samples already contain the identity of the audio source (*i.e.*, the spoofing one contains the modulation by the electrical device $h_{dev1,M_1}(f,t)$). Fig. 2(a) shows the spectrums of the voice command "OK Google" and its spoofing counterpart. It's observed that the sub-bass spectrums (20-300 Hz) between two audio samples are quite different even if they are deemed similar, and this phenomenon is utilized by mono channel-based schemes such as Sub-base [21].

Limitations. However, in a real-world environment, various factors (*e.g.*, the surrounding object's shape and materials, the sound transmission path, and the absorption coefficient of air) affect both voice generation and air channel gains. As illustrated in Fig. 1(c), when two microphones have different positions related to the user, the received audio $y_{user,M_1}(f,t)$ and $y_{user,m2}(f,t)$ would be different. It is observed from Fig. 2(b) that the spectrums of authentic and spoofing audio samples changes when putting the microphone in another place.

2.3.2 Dual Channel-based Detection

Principles. The limitations of existing mono channel-based detection described in Section 2.3.1 inspire the proposal of dual channel-based detection. As illustrated in Fig. 1(c) and Fig. 1(d), during the voice generation process, the sound in different directions from the audio source will undergo different modulation gains (*e.g.*, $h_{user,M_1}(f,t)$ and $h_{user,m2}$ when voice is transmitted from the mouth motion with different directions). Such a phenomenon will cause a unique "sound field" around the audio source. Thus, by measuring the field characteristics, it is feasible to induce the audio's identity (*i.e.*, authentic or spoofing). CAFIELD is the typical scheme that deploys two microphones to receive two audios and defines the concept of "fieldprint" as:

$$FP = log(\frac{y_1(f,t)}{y_2(f,t)}), \tag{1}$$

where $y_1(f,t)$ and $y_2(f,t)$ are audios collected by microphone M_1 and M_2, respectively. According to sound propagation analysis described in Section 2.2, the fieldprint of the authentic audio FP_{user} derived from $y_{user,M_1}(f,t)$ and $y_{user,m2}(f,t)$ can be written as:

$$\begin{aligned} FP_{user} &= log\frac{x(f,t) \cdot h_{user,M_1}(f,t) \cdot h_{air,M_1}(f,t)}{x(f,t) \cdot h_{user,m2}(f,t) \cdot h_{air,m2}(f,t)} \\ &= log\frac{h_{user,M_1}(f,t)}{h_{user,m2}(f,t)} + log\frac{h_{air,M_1}(f,t)}{h_{air,m2}(f,t)}. \end{aligned} \tag{2}$$

Similarly, the fieldprint of the spoofing audio FP_{dev} can be written as:

$$FP_{dev} = log\frac{h_{dev,M_1}(f,t)}{h_{dev,m2}(f,t)} + log\frac{h_{air,M_1}(f,t)}{h_{air,m2}(f,t)}. \tag{3}$$

Limitations. It is observed from equations 2 and 3 that fieldprint can be used for liveness detection since the $\frac{h_{user,M_1}(f,t)}{h_{user,m2}(f,t)}$ and $\frac{h_{dev,M_1}(f,t)}{h_{dev,m2}(f,t)}$ behind AF_{user} and AF_{dev} respectively contain the audio's identity. However, the existence of $\frac{h_{air,M_1}(f,t)}{h_{air,m2}(f,t)}$ reveals that measuring stable and accurate fieldprint requires the position between the audio source and the microphone pairs must be relatively stable. For instance, CAFIELD only performs well when the user holds a smartphone equipped with two microphones close to the face in a fixed manner. The fieldprint struggles in far distances (*e.g.*, greater than 40 cm in [24]), making it unsuitable for a home environment, in which users want to communicate with the smart speaker at arbitrary location in the room. This study aims to propose an effective and robust feature for passive liveness detection.

3 Design Goals and Motivation

As illustrated in Fig. 3, this study considers an IoT system with distributed smart speakers. Each smart speaker can serve a fixed area with a limited number of users, and we denote each area as a *client*. In real-world scenarios, a client can be an individual smart home in a smart city network or a voice-controlled factory floor in an industrial IoT system. In this section, we define the threat model, introduce the design requirements of LIVEARRAY, and elaborate on the fundamental insight and motivations of our proposed LIVEARRAY.

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

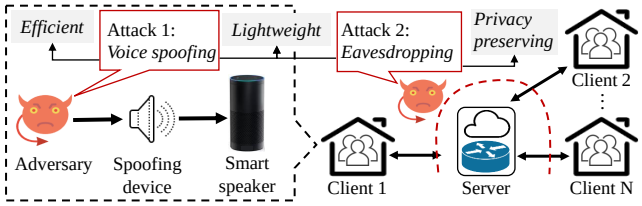


Fig. 3. Threat model and security requirements.

3.1 Threat Model

In this study, we assume the attacker can conduct the following two types of attacks:

Voice spoofing in the smart speaker's inputs. Similar to the previous works [21], [22], [24], as illustrated in Fig. 3, the adversary is assumed to have already obtained the victim's audio samples and can remotely control the victim's audio device (*e.g.*, smart TV, smartphone) to launch the voice spoofing attack. In this study, we mainly investigate how to leverage passive liveness detection to thwart replay attacks since most of the existing voice biometric-based authentication (human speaker verification) systems are vulnerable to this kind of replay attack. We also study LIVEARRAY's performance in thwarting advanced attacks such as modulated attacks [8] in Section 6.3.

Eavesdropping in the smart speaker's communication. In current systems such as smart home, as shown in Fig. 3, most device action-related commands are generated and delivered in the cloud server. We assume the adversary can eavesdropping the channel between the server and individual smart home local networks and potentially obtain some sensitive information (*e.g.*, voice semantics, user voiceprint). However, the cloud server and home networks are regarded as secure, meaning no sensitive information can be insider obtained by the adversary.

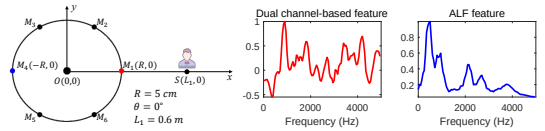
3.2 Design Requirements of LIVEARRAY

Considering the above-mentioned two attack interfaces, LIVEARRAY is designed to satisfy the following requirements.

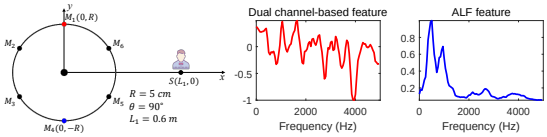
Efficient: compared with existing mono or dual channel-based liveness detection schemes as mentioned in Section 2.3, LIVEARRAY should achieve efficient and robust performance in detecting voice spoofing attacks.

Lightweight: considering the users in each client may change (*e.g.*, a worker in an industrial IoT system is transferred from a voice-controlled factory floor to another), LIVEARRAY should provide a uniform liveness detection classifier for all users from all clients. Since the smart speaker in each client has limited computational resources and limited users, LIVEARRAY should rely on lightweight features and require no cumbersome training sample collection procedures.

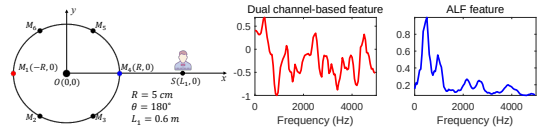
Privacy-preserving: during the whole liveness detection process, the sensitive information (*e.g.*, semantics, voiceprint) of the voice samples cannot be disclosed during the traffic flow transmission between distributed local smart home and the server.



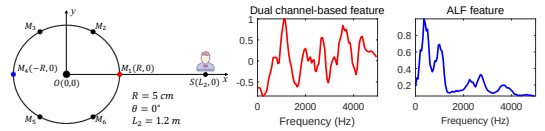
(a) θ is 0° and distance is 0.6 m.



(b) θ is 90° and distance is 0.6 m.



(c) θ is 180° and distance is 0.6 m.



(d) θ is 0° and distance is 1.2 m.

Fig. 4. Two-channel and ALF features when putting the microphone array and the audio source at different locations.

3.3 Motivation: Employing Microphone Array to Improve Liveness Detection Performance

In this subsection, to achieve the design requirements of liveness detection, we propose a novel, efficient, and robust liveness detection feature named *array-based liveness fingerprint (AFL)*. We elaborate on the rationale behind AFL by answering the following questions. **RQ1:** is it feasible to leverage the microphone array to obtain a stable feature to characterize the identity of the audio source? **RQ2:** is the collected feature (*e.g.*, AFL) effective and robust for conducting liveness detection? We answer these questions by performing a series of experiments.

3.3.1 Definition of Array-based Liveness Fingerprint (ALF)

To answer the question **RQ1**, we utilize a smart speaker to collect audio from a selected participant while rotating the microphone array and changing its position. As shown in Fig. 4, we build a Cartesian coordinate system to characterize the scenario when audio signals are transmitted from the source to the microphone array. Note that for simplicity, we assume the audio source and microphones of the smart speaker are located in the same plane, and we show the experiments in Section 3.3.2 to validate the array fingerprint's effectiveness in a more practical scenario. For a given smart speaker with N microphones $\{M_1, M_2, \cdots, M_N\}$, its microphones are evenly distributed on a circle whose center is denoted as the origin O. We assume the audio source (*e.g.*, human or electrical machine) as a point located at the x axis with the coordinate $(L, 0)$, where L is the distance between the source and origin. We also denote the included angle between $\overrightarrow{OM_1}$ and x axis as θ. Given the k-th microphone M_k, the collected audio data is denoted as $y_k(f, t)$.

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

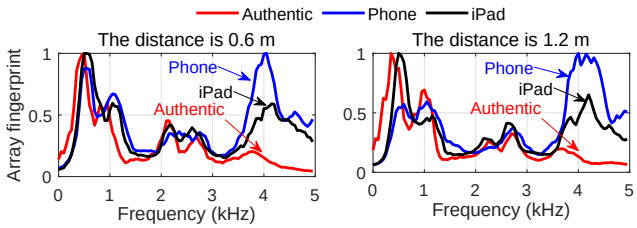


Fig. 5. Differentiating human voice from two spoofing devices via array fingerprints under different propagation paths.

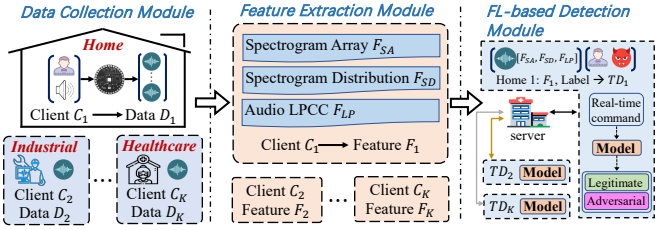


Fig. 6. System overflow of LIVEARRAY.

Existing passive liveness detection schemes are vulnerable to environmental changes. From Section 2.3.1 and Fig. 2, it is observed that changing the relative distance between the microphone and audio source will cause non-linear distortion of the microphone's collected signal. Such distortion makes mono channel-based detection schemes fragile to the change of propagation path. For the dual channel-based solution, it is observed from equations 2 and 3 that when the positions of the microphone pair are not fixed (*e.g.*, changing the audio source's location or rotating the microphone pair), the extracted fieldprint will no longer be a stable value. To illustrate this observation, as shown in Figs. 4(a), 4(b), and 4(c), we require the participant stays at $(L_1, 0)$ where L_1 is 0.6 m while setting θ as $0°$, $90°$, and $180°$ respectively. Then, the participant is required to stay as $(L_2, 0)$ where L_2 is 1.2 m and θ is $0°$ as shown in Fig 4(d). Fig 4 depicts the fieldprints derived from microphones M_1 and M_4[2]. It is observed that dual channel-based features vary drastically in the above-mentioned four scenarios.

Thus, to improve the effectiveness and robustness of the passive liveness detection, inspired by the circular layout of microphones in smart speakers as shown in Fig. 4, we define the array-based liveness fingerprint ALF. For the audios $\{y_1(f,t), y_2(f,t), \cdots, y_N(f,t)\}$ from N microphones, ALF is defined as below:

$$ALF = std([y_1(f,t), y_2(f,t), ..., y_N(f,t)]). \quad (4)$$

We apply equation 4 to the multi-channel audios collected in the four cases illustrated in Figs 4(a), 4(b), 4(c), and 4(d).[3] It is observed from Fig 4 that compared with dual channel-based feature, the array-based liveness fingerprint is stable and resilient to the changes of environmental factors, including both distance and angle. Therefore, our experimental results validate our intuition that it is feasible to leverage the microphone array to stably characterize the audio's identity.

3.3.2 Validation of Array Fingerprint

Besides the definition, to answer research question **RQ2**, we further validate the effectiveness of the proposed array-based liveness fingerprint via a series of real-world case studies. In the experiment, the participant is required to speak the command "Ok Google" at distances of 0.6 m and 1.2 m, respectively. We also conducted replay attacks via smartphones and iPad (*i.e.*, devices #8 and # 3 in TABLE 1).

2. The real process of extracting fieldprint is more complicated. Fig. 4 shows the basic principle following equation 1.

3. This array fingerprint is refined after extracting from equation 4. The detailed calculation steps are described in Section 4.2.2.

The normalized array fingerprints (*i.e.*, F_{AF} in Section 4.2.2) are shown in Fig. 5. It is observed that the array-based liveness fingerprints for the same audio sources are quite similar, while array-based liveness fingerprints for different audio sources are quite different. In summary, experimental results demonstrate the array-based liveness fingerprint can serve as a better passive liveness detection feature. This motivates us to design an efficient, lightweight, and privacy-preserving system which will be presented in the next section.

4 THE DESIGN OF LIVEARRAY

Based on the proposed array fingerprint with other auxiliary features, we propose LIVEARRAY, a robust, efficient, and privacy-preserving liveness detection system. As illustrated in Fig. 6, LIVEARRAY consists of the following modules: *Data Collection Module*, *Feature Extraction Module*, and *FL-based Detection Module*. In this section, we will elaborate on the details of each module in LIVEARRAY.

4.1 Data Collection Module

To record the voice command with multiple channels from which the array fingerprint can be extracted, LIVEARRAY utilizes a microphone array. Most popular commercial voice assistances (*e.g.*, Amazon Echo and Google Home) and some open modular development boards with the voice interface have built-in microphone arrays. In this study, we choose the development board (*i.e.*, Matrix Creator [45] and Seeed Respeaker [46]) since most commercial smart speakers do not provide a user interface for obtaining raw audio due to intelligent property concerns. However, since these development boards have similar sizes to commercial voice assistance, implementing LIVEARRAY on the above devices can be applied to commercial voice assistance without any notable hardware alteration.

For a IoT system (*e.g.*, smart home, industrial control system), when there is a voice command, its smart speaker with N microphones and a sample rate of F_s, it starts to record the voice sample lasting for T seconds. The collected audio is denoted as $V_{M\times N}$, where $M = F_s \times T$ and we let V_i be the i-th channel's audio $V(:, i)$. Besides, as shown in Fig. 6, when there are K clients (*e.g.*, home, laboratory) deploying LIVEARRAY, for the j-th client C_j, A_j known voice samples will be collected for training the liveness detection classifier as described in Section 4.3 and we denote them as D_j. Finally, the voice commands V_{target} for liveness detection and training dataset $D = \{D_1, D_2, \cdots, D_K\}$ are transmitted to the next module.

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

4.2 Feature Extraction Module

In this module, for a given collected multi-channel audio V, LIVEARRAY first conduct pre-processing on each channel's signal and then extract three features (*i.e.*, *Array Fingerprint Feature* F_{AF}, *Spectrogram Distribution Feature* F_{SD} and *Channel LPCC Feature* F_{LP}) for further liveness detection.

4.2.1 Data Pre-processing

To facilitate the feature extraction process, LIVEARRAY conducts two-step pre-processing on the collected audio $V = \{V_1, V_2, \cdots, V_N\}$.

Frequency analysis on multi-channel audio data. It is observed from Section 3.3 that the audio spectrogram consists of effective features which can reveal the user identity (*i.e.*, real human or spoofing device). The Short-Time Fourier Transform (STFT) is performed to generate the spectrograms of each channel's audio signal. For the i-th channel's audio V_i, which contains M samples, LIVEARRAY applies a Hanning window to separate the signals into small chunks with lengths of 1024 points and overlapping sizes of 728 points. Finally, a 4096-sized Fast Fourier Transform (FFT) is performed for each chunk and a spectrogram S_i is obtained as shown in Fig. 7(a).

Direction detection. Given a collected audio $V_{M\times N}$, to determine the microphone closest to the audio source, LIVEARRAY firstly applies a high pass filter with a cutoff frequency of 100 Hz to $V_{M\times N}$. Then, for the i-th microphone M_i, LIVEARRAY calculates the alignment errors $E_i = mean((V(:, i-1) - V(:, i))^2)$ [47]. Finally, from the calculated $E = \{E_1, E_2, \cdots, E_N\}$, the microphone with minimum alignment error is selected as the corresponding microphone.

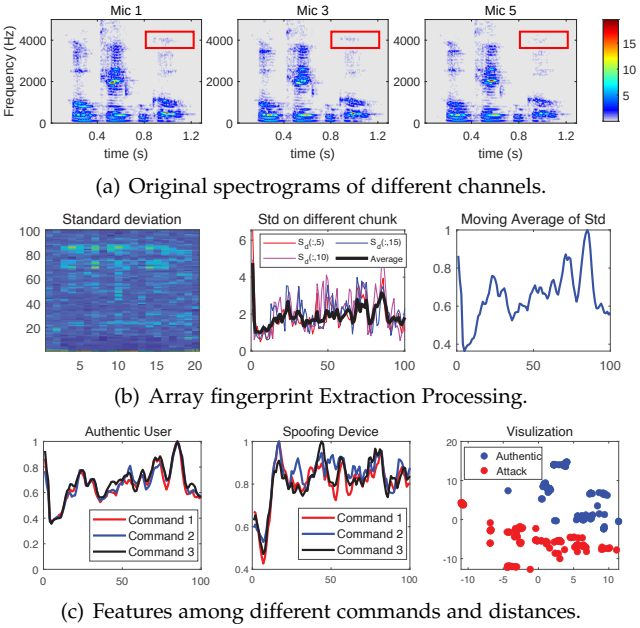
4.2.2 Spectrogram Array Feature

After transmitting the N channels' audio data $\mathbf{V} = [V_1, V_2, ..., V_N]$ into the spectrogram $\mathbf{S} = [S_1, S_2, \ldots, S_N]$, LIVEARRAY exploits the array fingerprint which is proposed in Section 3.3 to extract the identity of the audio source. To reduce the computation overhead, for S_k with size $M_s \times N_s$, we discard the components in which frequency is larger than the cutoff frequency f_{AF}. We empirically set f_{AF} as 5 kHz in this study. The resized spectrograms are represented as $\mathbf{Spec} = [Spec_1, Spec_2, ..., Spec_k]$, where $Spec_k = S_k(: M_{spec}, :)$. In this study, LIVEARRAY sets the sampling rate Fs as 48kHz and the FFT points N_{fft} as 4096, then the M_{spec} is calculated as $\frac{f_{AF}\times N_{fft}}{F_s} = 426$.

Fig. 7(a) illustrates resized spectrograms of three channels (*i.e.*, $Spec_1$, $Spec_3$, and $Spec_5$ from the first, third, and fifth microphones respectively) of the command "OK Google". It is observed that different channels' spectrograms are slightly different. For instance, the values in the red rectangles are different.

After visualizing the audio information, LIVEARRAY calculates the array-based liveness fingerprint from the spectrogram **Spec** according to equation 4. Before calculating, to show the obvious differences between different $Spec_i$, we re-sample the $Spec_i$ with the size $M_{spec} \times N_s$ to the G_i with size $M_G \times N_G$, where M_G and N_G are empirically set to 100 and 20 respectively. Then, F_G has the same size as G_i (*i.e.*, $M_G \times N_G$, and the elements of F_G can be represented as:

$$F_G(i, j) = std([G_1(i, j), G_2(i, j), ..., G_N(i, j)]). \quad (5)$$



(a) Original spectrograms of different channels.

(b) Array fingerprint Extraction Processing.

(c) Features among different commands and distances.

Fig. 7. Illustration of array fingerprint feature F_{AF} extraction.

It is observed in Fig. 7(b) that F_S containing N_G chunks, which is calculated from spectrograms. However, the $F_G(:, i)$ varies in different time chunks. It is caused by the different articulatory gestures when pronouncing different phonemes. In order to overcome this issue, LIVEARRAY leverages the phenomenon that even though different phonemes contain different gestures, there are common components over a long duration of time. Thus, LIVEARRAY averages the F_S in the time domain. The average result $\overline{F_S}$ is shown in Fig. 7(b). Finally, LIVEARRAY performs a 5-point moving average and normalization on $\overline{F_S}$ to remove noise and generate the spectrogram array fingerprint F_{AF}.

To show the effectiveness of the F_{AF} feature generation process, we present a simple demonstration in Fig. 7(c). In the experiment, three voice commands (*e.g.*, "OK Google", "Turn on Bluetooth", "Record a video") are evaluated. Distances between the speaker and microphone array are set as 0.6 m and 1.2 m in the first two commands and the last command, respectively. In Fig. 7(c), it is revealed that the different commands can show a similar array-based liveness fingerprint. Moreover, the characteristic differences in features can be clearly shown between authentic and spoofing audio. Finally, since LIVEARRAY should be in low time latency for the response, the feature is required to be lightweight. Thus, the F_{AF} is re-sampled as N_{SA} point length. In this study, N_{SA} is set as 40 empirically.

4.2.3 Spectrogram Distribution Feature

As analyzed in Section 2.3, the audio source also provides useful information, which contains the user identity. As a result, we also extract another fingerprint called F_{SD} for liveness detection. For a spectrogram S_k from the k-th channel, a N_G-dimension vector Ch_k is calculated as $Ch_k(i) = \sum_{j=1}^{M_{spec}} S_k(j, i)$. For detail, M_{spec} and N_G are set as 85 and 20 respectively. Note that, When calculating F_{SDP}, the cutoff frequency is set as 1 kHz. The reason is that most human voice frequency components are located

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

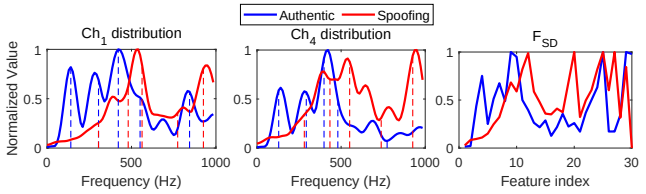


Fig. 8. Spectrogram distributions between authentic human and spoofing device.

in the 0~1 kHz range. The corresponding M_{Spec} is calculated as 85 under the the parameters in Section 4.2.2. Considering the audio with N channels, the channel frequency strength is generated as $Ch = [Ch_1, Ch_2, ..., Ch_N]$.

Fig. 8 shows channel frequency strengths Ch_1 and Ch_4 of first and fourth channels from both authentic and spoofing audios. The results reveal obvious differences between the audio collected by the real human and the spoofing device Thus, channel frequency strengths $\overline{Ch}$ are averaged and re-sampled by LIVEARRAY. The length is set as N_{Ch} as the first component of F_{SD}. Specifically, the averaged value of $\overline{Ch}$ is calculated as $\overline{Ch(i)} = mean([Ch_1(i), Ch_2(i), ..., Ch_N(i)])$, while N_{Ch} is set as 20. It is observed that in the same audio, it is reflected in Ch that differences in each channel between magnitudes and distributions still exist. The cumulative distribution function Cum_k is designed to characterize the distribution of Ch, for Ch_k from the k-th channel. Then, LIVEARRAY determine the indices μ to split Cum_k uniformly. As for the results in Fig. 8, the k-th channel is divided into 6 bands. And LIVEARRAY sets the $Thr = [0.1, 0.3, 0.5, 0.7, 0.9]$. Besides, the index $\mu(k, i)$ of the i-th Thr for Ch_k is required to satisfy the following condition:

$$Cum_k(\mu(k,i) \leq Thr_i \leq Cum_k(\mu(k,i)+1). \tag{6}$$

After generating indices μ with the dimension of $N \times 5$, LIVEARRAY utilizes the mean value D_{mean} and standard deviation D_{std} from different channels as extra feature to optimize the performance. The two variable are both vectors with the length of 5, which is calculated as $D_{mean}(i) = mean(\mu(:, i))$ and $D_{std}(i) = std(\mu(:, i))$. Finally, the spectrogram distribution fingerprint F_{SD} is generated as $F_{SD} = [\overline{Ch}, D_{mean}, D_{std}]$ and Fig. 8 illustrates F_{SD} from authentic and spoofing audios.

4.2.4 Channel LPCC Features

The channel Linear Prediction Cepstrum Coefficients (LPCC) feature F_{LP} is set as the last feature of LIVEARRAY. As each channel has unique physical properties, retaining the LPCC containing the audio characteristics can improve the detection performance. For audio signal $y_k(t)$ collected by microphone M_k, LIVEARRAY calculates the LPCC with the order $p = 15$. To optimize the time overhead spent on extracting F_{LP}, LIVEARRAY only preserves the LPCCs from audios in these two channels $(M_i, M_{mod(i+N/2,N)})$. M_i is the closet microphone, which is determined in Section 4.2.1. Finally, the final feature vector is generated $X = [F_{SA}, F_{SD}, F_{LP}]$.

4.3 FL-based Detection Module

Finally, in a distributed IoT scenario with K clients, after extracting the feature vector from the audio input, LIVEARRAY utilizes a federated learning (FL) classification model to conduct the liveness detection task. As shown in Fig. 6, the FL architecture consists of K clients and a remote server. Each client represents an individual group using the smart speaker (*e.g.*, users in a smart room, workers in an intelligent laboratory), and a lightweight feed-forward back-propagation neural network is trained by the remote server. This module contains two phases as follows:

Classification model building phase. For the i-th client C_i with several users, before launching liveness detection, it should provide a dataset D_i which contains audio commands and their corresponding identity labels (*i.e.*, authentic or spoofing). These datasets $D = \{D_1, D_2, \cdots, D_K\}$ from K clients $C = \{C_1, C_2, \cdots, C_K\}$ are then utilized for training the classification model. In the FL scenario, the training procedure consists of a total of N_{iter} iterations. During each iteration, each client trains the model locally, and the server aggregates the global model. Note that the local models in all clients and the global model in the server have the same architecture, in which three hidden layers with rectified-linear activation (layer sizes: 64, 32, 16) are selected, and the dropout ratio is set as 0.2.

More specifically, in the t-th iteration, before training, the server sends the global model G^{t-1} trained in the $(t-1)$-th iteration to each client. Then, for the k-th client C_k, it trains its local model based on the initial model G^{t-1} and the local dataset D_k. The local model trained by C_k is represented as L_k^t, and C_k only uploads the update of model parameters $W_k^t = L_k^t - G^{t-1}$ to the server. The server receives $W^t = W_1^t, W_2^t, \cdots, W_K^t$ from K clients and aggregates them into the model $G^{N_{iter}}$ in the t-th iteration following the equations below:

$$G^t = G^{t-1} + \sum_{k=1}^{K} \frac{n_k \cdot W_k t}{n}, \tag{7}$$

$$n_k = ||D_k||, \tag{8}$$

$$n = \sum_{k=1}^{K} n_k, \tag{9}$$

where n_k is the size of the k-th training dataset, and n is the sum of $n_1, n_2, \cdots, n_K$. The generated global model G^t is assigned to each client for the next iteration of training. Finally, after N_{iter} iterations, LIVEARRAY obtains the final classifier $G^{N_{iter}}$ for liveness detection.

Liveness detection phase. For a given client, when presented with a new voice command, after generating the features following the steps in Section 4.2, it leverages the final model $G^{N_{iter}}$ to conduct the liveness detection task. It is important to note that during both the model building and liveness detection phases, no raw audio data or extracted features are transmitted from any client to the server. As a result, the privacy information related to the audio can be preserved and protected.

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

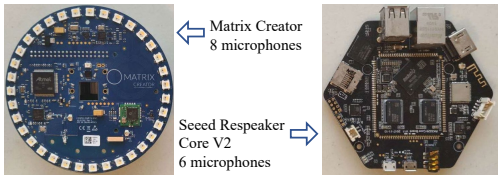


Fig. 9. Microphone arrays in evaluations.

TABLE 1
Loudspeaker used for generating spoofing attacks.

No.	Type*	Manufacture	Model	Size (L*W*H in cm)
1	LS	Bose	SoundLink Mini	5.6 x 18.0 x 5.1
2	T	Apple	iPad 6	24.0 x 16.9 x 0.7
3	T	Apple	iPad 9	24.0 × 16.9 × 0.7
4	LS	GGMM	Ture 360	17.5 × 10.9 × 10.9
5	S	Apple	iPhone 8 Plus	15.8 x 7.8 x 0.7
6	S	Apple	iPhone 8	13.8 × 6.7 × 0.7
7	S	Apple	iPhone 6s	13.8 × 6.7 × 0.7
8	S	Xiaomi	MIX2	15.2 × 7.6 × 0.8
9	LS	Amazon	Echo Dot (2nd Gen)	8.4 × 3.2 × 8.4
10	LP	Apple	MacBook Pro (2017)	30.4 × 21.2 × 1.5
11	LS	VicTsing	SoundHot	12.7 x 12.2 x 5.6
12	LS	Ultimate Ears	Megaboom	8.3 x 8.3 x 22.6
13	LS	Amazon	Echo Plus (1st Gen)	23.4 x 8.4 x 8.4
14	S	Xiaomi	Mi 9	15.8 × 7.5 × 0.8

* LS = Loudspeaker, T=Tablet, S=Smartphone, LP=Laptop.

5 Evaluations

5.1 Experiment Setup

Hardware setup. In this study, to collect multi-channel audios, we employ two open modular development boards (*i.e.*, Matrix Creator and Seeed Respeaker Core v2) as shown in Fig. 9. The microphone numbers in Matrix and Respeaker are 8 and 6, respectively. We set the sampling rate of audio recording as 48 kHz. Note that we do not choose commercial smart speakers, including Google Home and Amazon Echo since they do not provide interfaces for developers to obtain the audio files. However, our LIVEARRAY can be applied to these devices since they are similar to the chosen devices (*e.g.*, the radiuses of Matrix and Respeaker are 5.4 cm and 4.7 cm, respectively). For the spoofing device, we employ 14 different electrical devices with various sizes and audio qualities whose detailed parameters (*e.g.*, manufacturing, model, and size) are shown in TABLE 1.

Data collection procedure. In this study, we recruited 20 participants to provide multi-channel audio data. Among them, 2 participants are female (User #3 and #13), and 18 participants are male, with ages ranging from 18 to 32. The data collection procedure consists of two phases: *(i) Authentic Audio Collection:* In this phase, each participant speaks 20 different voice commands as listed in TABLE 2. The experimental session can be repeated multiple times by each participant. We pre-defined four distances (*i.e.*, 0.6 m, 1.2 m, 1.8 m, 2.4 m) between the microphone array and the participant, and they can choose any of these distances in each session. For the speaking behavior, we asked the participants to speak the commands naturally without specifying any fixed speed or tone. *(ii) Spoofing Audio Collection:* In this phase, following the approach adopted in previous works [22], [24], [48], after collecting the authentic voice samples, we used the spoofing devices listed in TABLE 1 to automatically replay the samples without involving the participants. When replaying a voice command, the electrical device is placed at the same location as the corresponding participant. Note that, during the spoofing audio collection phase, the noise level is set to be below 30 dB.

Dataset description. After finishing experiments, we utilize pyAudioAnalysis tool to split the collected audio into multiple voice command samples.[4] After removing incorrectly recognized samples, we get a dataset containing 32,780 audio samples. We refer to this dataset as MALD dataset and utilize it to assess LIVEARRAY.[5] The details of MALD dataset are shown in TABLE 3. Note that, to simulate the distributed environment, 20 participants are divided into six groups, and each group represents an individual client in the federated learning. For instance, in the client #3, there are four users (#4, #5, #6, #8) who provide 2,305 authentic samples at three different positions (*i.e.*, the distance of 0.6 m, 1.2 m and 1.8 m) and we utilize these collected samples with three spoofing devices (*i.e.*, iPad 9, Ture 360, MIX 2) to generate 6,415 spoofing samples.

Training procedure. As mentioned in Section 4.3, LIVEARRAY needs to be trained with audio samples before detecting spoofing attacks. When evaluating the overall performance of LIVEARRAY on the collected MALD dataset in Section 5.2, we perform the two-fold cross-validation. In each fold (*i.e.*, training procedure), half samples from each client are randomly chosen to train a classifier, and the validation dataset proportion is set as 20%. When considering the FL-based scenario, we set the round time between client and server as 20, and the local iteration time is 100. When evaluating the impact of other factors as shown in Section 5.3 and Section 6.1, the training procedure depends on the specific experiment, and we show the training dataset before presenting the evaluation results.

Evaluation metrics. Similar to previous works [19], [22], [24], in this study, we choose accuracy, false acceptance rate (FAR), false rejection rate (FRR), true rejection rate (TRR), and F1 score as metrics to evaluate LIVEARRAY. The accuracy means the percentage of the correctly recognized samples among all samples. FAR represents the rate at which a spoofing sample is wrongly accepted by LIVEARRAY, and FRR characterizes the rate at which an authentic sample is falsely rejected. F1 score provides a balanced view of the performance of LIVEARRAY. Note that, when calculating the F1-score, we regard authentic and spoofing samples as positive and negative, respectively.

Ethics consideration. The experiments are under the approval of the institutional review board (IRB) of our institutions. During the experiments, we explicitly inform the participants about the experimental purpose. Since only the voice data are collected and stored in an encrypted dataset, there is no health or privacy risk for the participant.

5.2 Performance of LIVEARRAY

5.2.1 Overall performance.

When evaluating LIVEARRAY on our own MALD dataset, we choose two-fold cross-validation, which means the training and testing datasets are divided equally. LIVEARRAY

4. PyAudioAnalysis: https://pypi.org/project/pyAudioAnalysis/.
5. MALD is the abbreviation of "microphone array-based liveness detection".

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

TABLE 2
20 Voice Commands in Evaluations.

(1) OK Google. (2) Turn on Bluetooth. (3) Record a video. (4) Take a photo. (5) Open music player. (6) Set an alarm for 6:30 am.
(7) Remind me to buy coffee at 7 am. (8) What is my schedule for tomorrow? (9) Square root of 2105? (10) Open browser.
(11) Decrease volume. (12) Turn on flashlight. (13) Set the volume to full. (14) Mute the volume. (15) What's the definition of transmit?
(16) Call Pizza Hut. (17) Call the nearest computer shop. (18) Show me my messages.
(19) Translate please give me directions to Chinese. (20) How do you say good night in Japanese?

TABLE 3
Detailed information of MALD dataset.

Client #	User #	# Authentic Samples	# Spoofing Samples	Distance (m)	Spoofing Devices
1	1, 7	1200	3600	0.6, 1.2, 1.8	SoundLink Mini, iPad 6, iPhone 8 Plus
2	2, 3	1133	1983	0.6, 1.2, 1.8	Ture360, iPhone 6s, iPad9
3	4, 5, 6, 8	2305	6415	0.6, 1.2, 1.8	iPad9, Ture360, MIX2
4	9, 10, 11, 12	3211	3198	0.6, 1.2, 1.8, 2.4	Echo Plus (1st Gen)
5	13, 14, 15, 16, 17, 18	1191	4577	1.8	iPad9, Mi 9, Echo Plus (1st Gen)
6	19, 20	1201	2765	0.6, 1.2, 1.8	iPhone 8, Echo Dot (2nd Gen), MacBook Pro (2017), SoundHot, Megaboom

TABLE 4
Per-client breakdown analysis.

Client	C_1	C_2	C_3	C_4	C_5	C_6
Accuracy (%)	99.69	99.03	98.59	99.20	99.48	99.06
FAR (%)	0.35	0.25	1.22	0.54	0.52	1.19
FRR (%)	0.18	2.69	1.82	1.28	0.54	0.00
F1-score (%)	99.39	98.35	97.80	98.86	99.23	97.78

achieves the detection accuracy of 99.16% and the F1-score of 98.66%. More specifically, for all 32,780 samples, the overall FAR and FRR are only 0.66% (*i.e.*, 149 out of 22,539 spoofing samples are wrongly accepted) and 1.22% (*i.e.*, 125 out of 10,241 authentic samples are wrongly rejected) respectively. The results show that LIVEARRAY is highly effective in thwarting spoofing attacks.

For the aspect of time overhead, when performing liveness detection on a desktop with Intel i7-7700T CPU and 16 GB RAM, the average time overhead on 6-channel and 8-channel audios are 0.12 seconds and 0.38 seconds, respectively, which is acceptable in real-world scenarios.

To assess LIVEARRAY's ability on different clients in the FL scheme, TABLE 4 shows the liveness detection results, including accuracy, FAR, FRR, and F1-score of each client. It is observed that the performance of LIVEARRAY varies slightly among different clients. For instance, the FAR varies from 0.54% in client #5 to 1.22% in client #3 and the FRR varies from 0 in client #6 to 2.69% in client #2. However, even in the worst cases (*i.e.*, client #3 and #6), the lowest detection accuracy and F1-score are still at 98.59% and 97.78% respectively, which demonstrate the effectiveness of LIVEARRAY in the FL scenario.

5.2.2 Comparison with centralized and previous works.

Comparison with centralized scheme. LIVEARRAY is designed for the scenario in which multiple clients have geographically diverse locations and do not want to share any original voice samples. However, some smart speakers (*e.g.*, Amazon Alexa) employ the centralized architecture in which the user's voice samples are transmitted and processed in the cloud-driven by powerful hardware. To evaluate the performance of LIVEARRAY in the centralized scenario, we conduct the two-fold cross-validation on the MALD dataset. Note that, to simulate the centralized scenario, voice samples from different clients are merged as one in the training procedure. The liveness detection accuracy and F1-score are 99.84% and 99.74%, respectively. Compared with the centralized scheme, when choosing the FL model, LIVEARRAY can achieve the privacy-preserving property with an acceptable cost of performance degradation (*e.g.*, the accuracy of 99.16% in FL compared with 99.84% in the centralized scheme).

Comparison with other passive liveness detection features. There are other passive liveness detection schemes such as VOID [22], CAFIELD [24], Pop noise-based schemes [43], [44], EarArray [49], and Machine-induced Audio Detection [50]. To demonstrate the superiority of the microphone array-based features utilized by LIVEARRAY, we compare our feature with existing features, including the mono audio-based feature utilized by VOID [22] and the two-channel audio-based feature adopted by CAFIELD [24].[6] To eliminate the potential bias in our collected MALD dataset, we also exploit a third-party dataset named ReMasc Core which contains 12,023 voice samples from 40 different users.[7] Note that since the schemes VOID and CAFIELD and the dataset ReMasc are not developed in the distributed scenario, the evaluations are conducted in the centralized scenario. We re-implement mono audio-based scheme VOID [22] and two-channel audio-based scheme CAFIELD [24]. As shown in TABLE 5, since MALD dataset is collected in the indoor smart home environment and ReMasc is collected in both indoor, outdoor, and vehicle environments, the detection accuracy varies among these two datasets. However, the feature utilized by LIVEARRAY is superior to previous works in both datasets. Especially for the ReMasc Core dataset in which only half of the audio samples are collected in the indoor environment,

6. EarArray to defend against ultrasonic-based attacks (*e.g.*, dolphin attacks [2]), but it is not designed to detect spoofing audios with human voice frequency. [50] requires a deep learning model which violates the lightweight requirement described in Section 3.2. Thus, we only compare LIVEARRAY with VOID and CAFIELD.

7. We only consider the 12,023 audio samples collected by circular microphone arrays in the ReMasc Core dataset.

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

TABLE 5
Detection accuracy among different features in the centralized scenario.

Liveness feature	Dataset	
	MALD dataset	ReMasc dataset
Microphone array	99.84%	97.78%
Mono feature	98.81%	84.37%
Two-channel	77.99%	82.44%

TABLE 6
Performance when changing the training distance.

Training position (m)	1.2	1.8	2.4	ALL
Accuracy (%)	99.25	99.37	97.72	98.78
F1-score (%)	99.25	99.38	97.77	98.79

LIVEARRAY's feature is the only one that achieves an accuracy above 98.25%. The two-channel-based feature gets relatively low performance on both the MALD dataset and ReMasc dataset. It is quite natural since CAFIELD claimed it needs the user to hold the device with fixed gestures and short distances. In summary, these results demonstrate that compared with mono audio-based and two-channel-based features, exploiting microphone array-based feature achieves superior performance in the liveness detection task.

5.3 Impact of Various Factors on LIVEARRAY

In this subsection, we evaluate the impact of various factors (*i.e.*, distance, direction, user movement, spoofing device) on LIVEARRAY.

Impact of changing distance. To evaluate the performance of LIVEARRAY on a new distance, we recruit four participants to attend experiments at three different locations (i.e., 1.2 m, 1.8 m, 2.4 m). We collect 2,403 authentic and 2,395 spoofing audio samples. For a given distance, the classifier is trained with audios at this distance and tested on audios at other distances. Note that, to satisfy the FL-based scenario, four users are regarded as four different clients. As shown in TABLE 6, compared with the performance in Section 5.2, LIVEARRAY's performance undergoes degradation when the audio source (*i.e.*, the human or the spoofing device) changes its location. For instance, the F1-score when training at positions 1.2 m, 1.8 m, and 2.4 m are 99.25%, 99.38%, and 98.79%, respectively. However, LIVEARRAY achieves an overall detection accuracy of 98.78%, which demonstrates LIVEARRAY is robust to the training distance. This result also conforms to our motivation examples described in Section 3.3.

Impact of changing direction. In Section 5.1, when collecting audio samples, most participants face the smart speaker while generating voice commands. To explore the impact of the angles between the user's face direction and the microphone array, we recruit 2 participants to additionally collect authentic voice samples in four different directions (*i.e.*, front, left, right, back) and then the spoofing device #8 in TABLE 1 is utilized to generate spoofing audios. During data collection, a total of 5 microphone array devices are deployed and each device is regarded as a client in the FL. As shown in TABLE 7, we collect 5,187 authentic samples and 4,732 spoofing samples. Then, we train the classification model based on the audios from a given direction and perform liveness detection on audios from other directions. It is observed from TABLE 7 that in all scenarios, LIVEARRAY achieves an accuracy above 98.63%, which means LIVEARRAY is robust to the change of direction.

TABLE 7
Performance under different directions.

Direction	Front	Back	Left	Right	ALL
Training authentic #	1988	1000	1004	1195	5187
Training spoofing #	1882	932	947	971	4732
Accuracy (%)	98.63	98.70	99.37	99.20	99.00
F1-score (%)	98.72	98.74	99.40	99.22	99.04
FAR (%)	2.91	0.18	0.66	0.56	0.96
FRR (%)	0.00	2.32	0.60	1.03	1.05

TABLE 8
The FAR of each spoofing device.

Device #	1	4	6	8	9	10
FAR (%)	0.92	0.87	0.17	0.04	0.17	0.86
Device #	11	12	13	14	Others	All
FAR (%)	3.33	3.53	1.16	0.94	0	0.66

Impact of user movement. Similar to the above evaluation, we recruit 10 participants to speak while walking. Then, the participant walks while holding a spoofing device (i.e., Amazon Echo) and plays spoofing audio. We collect 1,999 authentic and 1,799 spoofing samples, and the classifier is the same as that in Section 5.2. The detection accuracy and F1-score are 99.26% and 99.50%, respectively, which demonstrates that LIVEARRAY and the array fingerprint are robust even with the user's movement.

Impact of Spoofing Devices. It is well known that different devices have different frequency-amplitude response properties and thus may have different attacker power. To evaluate LIVEARRAY's performance in thwarting different spoofing devices, we conduct an experiment based on the MALD dataset containing 14 spoofing devices as listed in TABLE 1. The experiment follows the configuration described in Section 5.2.1.

TABLE 8 illustrates the liveness detection result of LIVEARRAY on each device in this case. It is observed that among 14 devices, the overall FAR is 0.66% (*i.e.*, 149 out of 20,290 spoofing samples are wrongly accepted). Besides, LIVEARRAY achieves overall 100% detection accuracy on 4 devices (*i.e.*, devices #2, #3, #5, #7). Even in the worst case (*i.e.*, device #12 Megaboom), the FAR is only 3.53%. In summary, LIVEARRAY is robust to various spoofing devices.

Impact of training dataset size. To reduce the user's registration burden, we explore the impact of training data size on the performance of LIVEARRAY. For our collected MALD dataset, we set the training dataset proportion as 10%, 20%, 30%, and 50% respectively. The results are shown in TABLE 9. It is observed that the detection performance increases from 89.89% to 99.16% when involving more training samples. Note that, even if we only choose 10% samples for training, LIVEARRAY still achieves the accuracy of 97.21% and F1-score of 95.44%.

Impact of voice command length. It is important to investigate how the length of different voice commands affects the performance of LIVEARRAY. Recognizing that various users

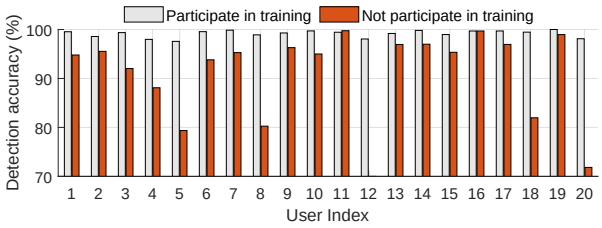
Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

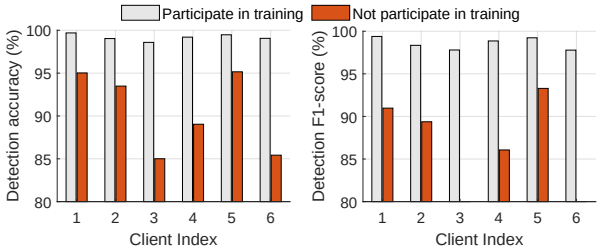
TABLE 9
Performance under various training dataset size.

Training proportion	10%	20%	30%	50%
Accuracy (%)	97.21	97.16	97.25	99.16
F1-score (%)	95.44	95.55	95.72	98.66
FAR (%)	1.07	3.25	3.42	0.66
FRR (%)	6.58	1.95	1.26	1.22

Fig. 10. The impact of the number of words.



(a) Per-user breakdown.



(b) Per-client breakdown.

Fig. 11. Detection accuracy when incomplete training procedure.

may have different speech speeds, instead of evaluating the duration of the voice, we focus on the number of words within the voice command. In our collected dataset, the number of words ranges from 2 to 8. The observation from Fig. 10 reveals that for each word length, the detection accuracy is consistently above 98.7%. Moreover, there is no noticeable statistical difference in performance across different command lengths. Consequently, these results demonstrate that LIVEARRAY remains robust to variations in the length of voice commands.

6 DISCUSSIONS AND LIMITATIONS

6.1 Performance under Incomplete Training Procedure

Similar to previous works [22], [24], [48], in Section 5.2, LIVEARRAY requires the user to participate in the enrollment procedures (*i.e.*, providing both authentic and spoofing voice samples). Considering that fully participating in enrollment is not always feasible, we explore the robustness of LIVEARRAY in handling the case where users who did not participate in the complete enrollment procedures.

In this case, we add an experiment to evaluate the performance of LIVEARRAY on participants that did not participate in the enrollment (*i.e.*, *unseen* users). In the experiment, for each user in the MALD dataset, we train the classifier using the other 19 users' legitimate and spoofing voice samples and regard the user's samples as the testing dataset. Note that all training procedures are done in the FL scenario. The detection results of each user and client are shown in Fig. 11. We also show the results described in Section 5.2 when users participate in the enrollment as a comparison.

From Fig. 11, it is observed that the overall detection accuracy decreases from 99.16% to 90.63%. In the worst case (*i.e.*, client #3), the detection accuracy decreases from 98.58% to 85.00%. The results demonstrate that ability of LIVEARRAY to address unseen users varies with different users. However, for 12 users, LIVEARRAY can still achieve an F1-score higher than 90%.

The performance degradation when addressing unseen users remains an open problem in the area of liveness detection [19], [21], [22], [48]. To partially mitigate this issue, a practical solution is requiring the unseen users to provide a small number of voice samples to enhance the classifier. As shown in Section 5.3, even setting the training dataset proportion as 10%, it is easy to achieve an acceptable detection performance.

TABLE 10
Different user selection when setting 5 clients.

Scenario	User Selection
S_1	$C_1 = \{1, 2, 3, 4\}$, $C_2 = \{5, 6, 7, 8\}$, $C_3 = \{9, 10, 11, 12\}$, $C_4 = \{13, 14, 15, 16\}$, $C_5 = \{17, 18, 19, 20\}$.
S_2	$C_1 = \{1, 6, 11, 16\}$, $C_2 = \{2, 7, 12, 17\}$, $C_3 = \{3, 8, 13, 18\}$, $C_4 = \{4, 9, 14, 19\}$, $C_5 = \{5, 10, 15, 20\}$.
S_3	$C_1 = \{12, 2, 18, 10\}$, $C_2 = \{14, 8, 5, 9\}$, $C_3 = \{13, 1, 7, 17\}$, $C_4 = \{6, 16, 19, 20\}$, $C_5 = \{15, 11, 4, 3\}$.
S_4	$C_1 = \{1, 4, 12, 5\}$, $C_2 = \{9, 19, 20, 14\}$, $C_3 = \{16, 13, 8, 10\}$, $C_4 = \{7, 17, 3, 18\}$, $C_5 = \{2, 11, 6, 15\}$.
S_5	$C_1 = \{10, 2, 16, 13\}$, $C_2 = \{19, 14, 9, 11\}$, $C_3 = \{7, 12, 17, 18\}$, $C_4 = \{5, 8, 15, 4\}$, $C_5 = \{6, 1, 20, 3\}$.

6.2 Performance of LIVEARRAY when Changing Client Settings in Federated Learning

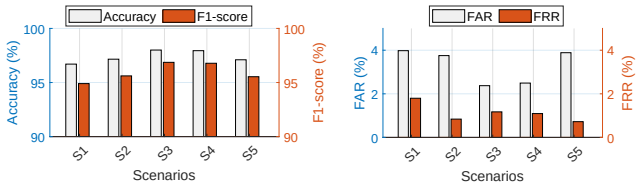
Note that all evaluation results in Section 5 are based on the client settings described in TABLE 3. However, it is crucial to examine the performance of LIVEARRAY when the user distributions of clients are changed in federated learning scenarios.

6.2.1 Different User Selection when the Number of Clients is Fixed

In this section, we explore the scenario where the number of clients in the federated learning setup remains fixed. Specifically, we set the client number to 5 and randomly assign four users to each client. Five different scenarios are considered, and the details of each scenario are presented

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833



(a) Accuracy and F1-score. (b) FAR and FRR.

Fig. 12. Performance of LIVEARRAY under different user selections when setting 5 clients.

TABLE 11
User selection when changing client numbers.

Number of Clients	User Selection
2	$C_1 = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$, $C_2 = \{11, 12, 13, 14, 15, 16, 17, 18, 19, 20\}$.
3	$C_1 = \{1, 2, 3, 4, 5, 6\}$, $C_2 = \{7, 8, 9, 10, 11, 12\}$, $C_3 = \{13, 14, 15, 16, 17, 18, 19, 20\}$.
4	$C_1 = \{1, 2, 3, 4, 5\}$, $C_2 = \{6, 7, 8, 9, 10\}$, $C_3 = \{11, 12, 13, 14, 15\}$, $C_4 = \{16, 17, 18, 19, 20\}$.
10	$C_1 = \{1, 2\}$, $C_2 = \{3, 4\}$, $C_3 = \{5, 6\}$, $C_4 = \{7, 8\}$, $C_5 = \{9, 10\}$, $C_6 = \{11, 12\}$, $C_7 = \{13, 14\}$, $C_8 = \{15, 16\}$, $C_9 = \{17, 18\}$, $C_{10} = \{19, 20\}$.
20	Each user is an individual client.

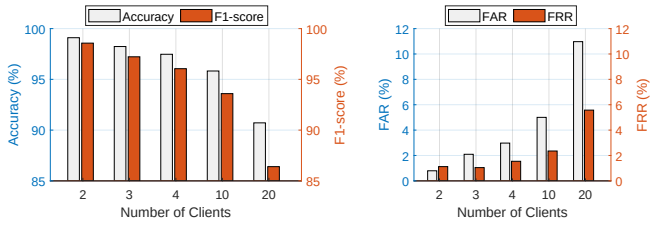
in TABLE 10. For each scenario, we follow the method described in Section 5.2.1 to generate the classifier and conduct the liveness detection. The detection accuracy, F1-score, false acceptance rate (FAR), and false rejection rate (FRR) of each scenario are depicted in Fig. 12. It can be observed that among all five scenarios, the accuracy ranges from 96.70% to 98.00%, and the F1-score varies from 94.90% to 96.87%. These results indicate that regardless of the distribution of users across clients, the performance of LIVEARRAY remains relatively stable and effective.

6.2.2 *Performance when Changing the Number of Clients*

We further evaluate the impact of the number of clients in the federated learning scenario on LIVEARRAY's performance. We set the number of clients as 2, 3, 4, 10, and 20, respectively, as shown in TABLE 11. Following the evaluation procedure described in Section 5.2.1, the detection results are presented in Fig. 13. It is observed that as the number of clients increases, the detection ability of LiveArray undergoes a significant degradation. For instance, when the client numbers increase from 2 to 20, the detection accuracy decreases from 99.10% to 90.71%, while the F1-score decreases from 98.56% to 86.40%. The reason behind this observation is that data from different clients possess unique properties (e.g., spoofing device type, gender, user's voiceprint), and increasing the number of clients makes LIVEARRAY's performance less optimal compared to the centralized training style. However, in most cases, the performance of LIVEARRAY is still acceptable (*e.g.*, detection accuracy is higher than 95% when there are 10 clients).

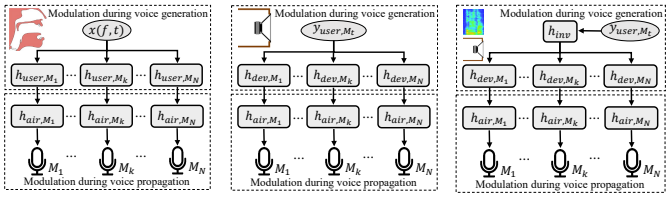
6.3 Defending against Modulated Replay Attacks

In this subsection, we study the performance of LIVEARRAY under the emerging modulated attack [8]. By modulating the spectrum of replayed audio, the modulated attack [8]



(a) Detection Accuracy and F1-score. (b) FAR and FRR.

Fig. 13. Performance of LIVEARRAY when changing the number of clients.



(a) Authentic scenario. (b) Replay attack. (c) Modulated attack.

Fig. 14. Audio generation during modulated attacks.

identifies an essential threat to existing liveness detection schemes. To achieve this goal, in the attack model, the adversary first needs to use a microphone of the target device to collect the target user's authentic voice samples. [8] Then, the adversary physically approaches the spoofing device to measure its frequency amplitude curve and the corresponding inverse filter using the target microphone. Finally, by applying the inverse filter on the authentic audio and playback it via the spoofing device, for the target microphone, the spectrum of the collected modulated audio is similar to the collected authentic audio as shown in Fig. 15(a). However, since the array-based liveness fingerprint characterizes the difference between the audios collected by multiple microphones, it is feasible for LIVEARRAY to thwart modulated attacks.

6.3.1 *Analysis of Modulated Replay Attacks*

For the sake of simplicity, we utilize the theoretical model described in Section 2.2 to analyze the modulated attack. Inspired by Figs. 1(c) and 1(d), the voice command generation and propagation in the modulated attack scenario can be described as Fig. 14. As illustrated in Fig. 14(a), for the authentic scenario, the audio collected by the k-th microphone M_k is:

$$y_{user,M_k} = x \cdot h_{user,M_k} \cdot h_{air,M_k}, \tag{10}$$

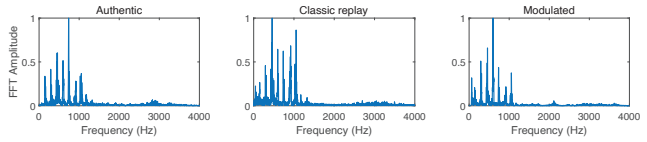
where h_{user,M_k} and h_{air,M_k} are signal gains during voice generation in the user's mouth and propagation in the air channel, respectively.

As illustrated in Fig. 14(b), in the classical replay attack, the attacker spoofs the voice assistance by playing the authentic audio collected from the microphone, which is equipped with the target victim's voice interface. Note that the modulated attack is designed for the mono microphone scenario, and we choose M_t as the target victim microphone.

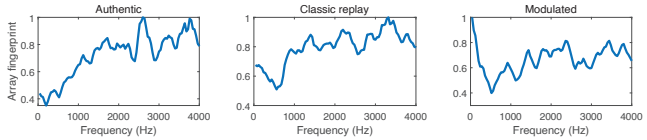
8. The attack assumption of the modulated attack [8] only considers the voice interface with only one microphone.

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833



(a) Spectrums of authentic, replay, and modulated audios.



(b) Array fingerprints of authentic, replay, and modulated audios.

Fig. 15. Spectrums and array fingerprints of audio signals.

We regard the audio collected by M_k as y_{replay,M_k}, which can be described as:

$$y_{replay,M_k} = y_{user,M_t} \cdot h_{dev,M_k} \cdot h_{air,M_k}, \tag{11}$$

where h_{dev,M_k} and h_{air,M_k} are signal gains during voice generation in the user's mouth and propagation in the air channel, respectively.

When conducting modulated attacks, $x(f,t)$ will be processed through the filter h_{inv} before modulating it via the electrical spoofing device. According to the details described in [8], the h_{inv} is set as $\frac{1}{h_{dev,M_t} \cdot h_{air,M_t}}$. Thus, the audios collected by M_k and the target microphone M_t are:

$$\begin{aligned} y_{modulated,M_t} &= y_{user,M_t} \cdot h_{inv} \cdot h_{dev,M_t} \cdot h_{air,M_t} \\ &= y_{user,M_t} \cdot \frac{h_{dev,M_t} \cdot h_{air,M_t}}{h_{dev,M_t} \cdot h_{air,M_t}} \\ &= y_{user,M_t}. \end{aligned} \tag{12}$$

$$\begin{aligned} y_{modulated,M_k} &= y_{user,M_t} \cdot h_{inv} \cdot h_{dev,M_k} \cdot h_{air,M_k} \\ &= y_{user,M_t} \cdot \frac{h_{dev,M_k} \cdot h_{air,M_k}}{h_{dev,M_t} \cdot h_{air,M_t}}. \end{aligned} \tag{13}$$

From the equations 10, 11, and 12, it is observed that after exploiting inverse filter the audio signal $y_{modulated,M_t}$ collected by M_t has the same spectrum with y_{user,M_t}. However, since the inverse filter does not consider the channel gains between other audio transmission paths, the audios collected by other microphones are quite different. From equation 13, if $h_{dev,M_k} \neq h_{dev,M_t}$, the $y_{modulated,M_K}$ will still contain the information (*i.e.*, h_{dev,M_k} and h_{dev,M_t}) related the electrical spoofing device. Therefore, it is still feasible for LIVEARRAY to extract the information related to the audio's identity to thwart the modulated attack.

6.3.2 Performance on Thwarting Modulated Attacks

We conduct a case study to demonstrate the robustness of the array fingerprint. We select an Amazon Echo and a ReSpeaker microphone array as the spoofing and target device, respectively, and follow the steps in [8] to re-implement modulated attack. We recruit a volunteer to provide an authentic voice command and then collect its corresponding classic replay and modulated audios generated by the Echo device.

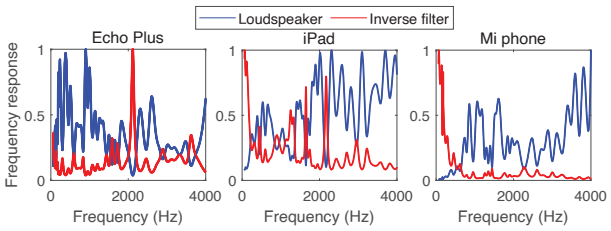


Fig. 16. The amplitude responses of different spoofing devices and their corresponding inverse filters.

Fig. 15 shows spectrums and array fingerprints of authentic audio and its corresponding replay and modulated samples. It is observed from Fig. 15(a) that, for a given channel, the spectrum of modulated audio (*i.e.*, FFT Amplitude of the first channel audio V_1) is similar to that in the authentic audio, which means it can bypass many existing liveness detection schemes. However, since the human vocal organs and spoofing devices cannot be regarded as a point sound source, the sounds received in multiple microphones show obvious differences. And the difference between multiple channel audios (*i.e.*, six channels in this experiment) characterized by array fingerprints still retains the audio's identity. As shown in Fig. 15(b), the array fingerprint of the modulated sample is still similar to that of classic replay audio, which shows it is feasible for LIVEARRAY to thwart the modulated attack.

Then, we evaluate the effectiveness of LIVEARRAY in thwarting the modulated attack. When re-implementing the modulated attack and calculating the detection accuracy of LIVEARRAY, we choose three spoofing devices #3, #13, and #14 (*i.e.*, iPad 9, Mi phone 9, and Amazon Echo Plus) as spoofing devices and Respeaker microphone array as the target device. To calculate the inverse filter for each device, we follow the steps described in the modulated attacks [8]. The frequency responses and their inverse filters of three spoofing devices are shown in Fig. 16. We recruit 10 participants to provide authentic samples and follow the steps described in [8] to generate 1,990, 1,791, and 1,994 modulated attack samples for Echo, iPad, and Mi respectively.

When employing the classifier in Section 5.2, the accuracy of LIVEARRAY on detecting the modulated samples among Echo, iPad, and Mi are 100%, 92.74%, and 97.29% respectively. In summary, LIVEARRAY can successfully defend against the modulated attack, but the performance varies with different spoofing devices. Considering combining LIVEARRAY with the dual-domain detection proposed in [8] can further improve the security of smart speakers.

6.4 Other Issues

We discuss some minor issues and limitations of LIVEARRAY in this subsection.

The user's burden on the enrollment. We can incorporate the enrollment into daily use to reduce the user's time overhead on training LIVEARRAY. Firstly, the evaluation results from Section 5.3 show that LIVEARRAY is robust to the change of user's position, direction, and movement. That means the user can participate in the enrollment anytime. Then, to achieve this goal, we divide LIVEARRAY into

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

TABLE 12
Time overhead of LIVEARRAY and existing works.

Scheme	Liveness detection		Training
	Extraction	Testing	
Mono feature	0.035 s	0.035 s	<1 min
Two-channel	0.054 s	0.036 s	<1 min
ArrayID (6-channel) [54]	0.61 s	0.12 s	<8 min
ArrayID (8-channel)	0.76 s	0.38 s	
LIVEARRAY (6-channel)	0.59 s	0.039 s	<3 h 10 min
LIVEARRAY (8-channel)	0.73 s	0.061 s	

working and idle phases. In the working phase, when a user generates a voice command, LIVEARRAY collects the audio and saves the extracted features. During the idle phase, LIVEARRAY can automatically update the classifier based on these newly generated features. These steps can be done automatically without human involvement, which means LIVEARRAY can continuously improve its performance along with daily use. However, we admit that allowing the automatically continuous retraining process may involve other potential risks. For instance, attackers can launch poisoning attacks to reduce the performance of speech recognition and speaker verification [51], [52], [53].

Time overhead compared with existing works. The time costs of different liveness detection schemes are listed in Table 12. It is observed that, compared to existing works (*e.g.*, mono-channel-based feature [22], dual-channel-based feature [24]), the time overhead of feature extraction and testing in LIVEARRAY is noticeably larger. Since LIVEARRAY is designed for multi-channel audios, it naturally spends more time analyzing spectrograms from multiple channels. Additionally, as LIVEARRAY utilizes a neural network as a classifier rather than simple classifier models (*e.g.*, SVM in Void and GMM in CaField), the testing time cost is higher compared to existing works. However, the overall liveness detection time cost is acceptable.

Furthermore, the time overhead for model training in LIVEARRAY is significantly larger than in existing solutions. The reason is that using the federated learning style increases the training iterations and incurs communication overhead between the server and different clients. We acknowledge that this is a major issue caused by employing FL to achieve privacy preservation. However, the training procedure can be automated without human involvement, which means LIVEARRAY does not impose any extra burden on the user.

Deploying LIVEARRAY in lightweight smart speakers. Since most commercial smart speakers do not provide a user interface for obtaining raw audio and installing malware due to intellectual property concerns, in this study, we utilize the ReSpeaker device to collect audio and utilize the desktop to process the data. However, to simulate the hardware conditions of lightweight smart speakers, in our experiments, we restrict the RAM usage of the desktop to 750 MB, which is less than that of current commercial smart speakers (*e.g.*, the RAM size of ReSpeaker and Apple HomePod is both larger than 1 GB). Therefore, it is feasible to deploy LIVEARRAY on commercial smart speakers.

Impact of participants' genders. To assess whether the performance of LIVEARRAY varies under different user genders (*i.e.*, female and male), we examined the detection results among different genders. It is important to note that, to overcome the bias of client selection during the federated learning process, we considered five scenarios as described in TABLE 10. The evaluation results are shown in TABLE 13. It is observed that among all scenarios, the detection accuracy of female participants is higher than that of male participants. However, both of these accuracy values are high and stable enough. Thus, we can conclude that the performance of LIVEARRAY is stable and robust to the participants' gender.

TABLE 13
Detection accuracy under different genders.

Scenario	S_1	S_2	S_3	S_4	S_5
Female	97.73%	98.56%	99.55%	98.69%	98.76%
Male	96.56%	96.96%	97.79%	97.84%	96.87%

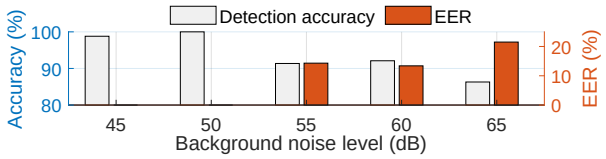


Fig. 17. Performance under Noisy Environments.

Impact of noise and other speakers. During the user's enrollment, we assume that the environment is silent and no other users are talking. As LIVEARRAY is a passive liveness detection system that solely relies on audio input, the presence of strong noise or other speakers' voices in the collected audios will inevitably degrade its performance. As depicted in Fig. 17, when the noise level increases from 45 dB to 65 dB, the accuracy decreases from 98.8% to 86.3%. Therefore, the existence of noise and other users talking can lengthen the enrollment time. Thankfully, as LIVEARRAY is designed for smart home or office environments, requesting users to maintain a silent environment during enrollment is a reasonable assumption. We consider this issue as part of our future work.

Temporal stability of array fingerprint. To evaluate the timeliness of LIVEARRAY, we recruit a participant to provide 100 authentic voice commands and launch voice spoofing every 24 hours. When using the classification model described in Section 5.2 and the audio dataset collected 24 hours and 48 hours later, LIVEARRAY still achieves over 98% accuracy. We admit that the generated feature may be variant when the participant changes her/his speaking manner or suffers mood swings. As mentioned in Section 6.4, a feasible solution to address this issue is incorporating the enrollment into the user's daily use to ensure the freshness of the classification model of LIVEARRAY.

7 CONCLUSION

In this study, we propose a novel liveness detection system LIVEARRAY for thwarting voice spoofing attacks without any extra devices. We theoretically analyze existing popular passive liveness detection schemes and propose a robust liveness feature array fingerprint. This novel feature both enhances effectiveness and broadens the application scenarios of passive liveness detection. LIVEARRAY is tested on

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

both our MALD dataset and another public dataset, and the experimental results demonstrate LIVEARRAY is superior to existing passive liveness detection schemes. Besides, we evaluate multiple factors and demonstrate the robustness of LIVEARRAY.

ACKNOWLEDGMENTS

The authors affiliated with Shanghai Jiao Tong University were, in part, supported by the National Natural Science Foundation of China under Grant 62132013, and 61972453. Yuan Tian is partially supported by NSF award #1943100 and Facebook Faculty Award. The preliminary version of this paper titled "Your Microphone Array Retains Your Identity: A Robust Voice Liveness Detection System for Smart Speakers" was published in the *Proc. of USENIX Security 2022* [54].

REFERENCES

[1] W. Diao, X. Liu, Z. Zhou, and K. Zhang, "Your voice assistant is mine: How to abuse speakers to steal information and control your phone," in *Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices (SPSM)*, 2014, pp. 63–74.

[2] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, "Dolphinattack: Inaudible voice commands," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017, pp. 103–117.

[3] N. Roy, H. Hassanieh, and R. Roy Choudhury, "Backdoor: Making microphones hear inaudible sounds," in *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2017, p. 2–14.

[4] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D. Wagner, and W. Zhou, "Hidden voice commands," in *25th USENIX Security Symposium (USENIX Security 16)*, 2016, pp. 513–530.

[5] X. Yuan, Y. Chen, Y. Zhao, Y. Long, X. Liu, K. Chen, S. Zhang, H. Huang, X. Wang, and C. A. Gunter, "Commandersong: A systematic approach for practical adversarial voice recognition," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 49–64.

[6] L. Zhang, Y. Meng, J. Yu, C. Xiang, B. Falk, and H. Zhu, "Voiceprint mimicry attack towards speaker verification system in smart home," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, 2020, pp. 377–386.

[7] W. Ding and H. Hu, "On the safety of iot device physical interaction control," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, p. 832–846. [Online]. Available: https://doi.org/10.1145/3243734.3243865

[8] S. Wang, J. Cao, X. He, K. Sun, and Q. Li, "When the differences in frequency domain are compensated: Understanding and defeating modulated replay attacks on automatic speech recognition," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2020, p. 1103–1119.

[9] X. Li, F. Yan, F. Zuo, Q. Zeng, and L. Luo, "Touch well before use: Intuitive and secure authentication for iot devices," in *The 25th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '19. Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3300061.3345434

[10] X. Lei, G.-H. Tu, A. X. Liu, C.-Y. Li, and T. Xie, "The insecurity of home digital voice assistants - vulnerabilities, attacks and countermeasures," in *2018 IEEE Conference on Communications and Network Security (CNS)*, 2018, pp. 1–9.

[11] K. Sun, T. Zhao, W. Wang, and L. Xie, "Vskin: Sensing touch gestures on surfaces of mobile devices using acoustic signals," in *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '18. Association for Computing Machinery, 2018, p. 591–605. [Online]. Available: https://doi.org/10.1145/3241539.3241568

[12] W. Ruan, Q. Z. Sheng, L. Yang, T. Gu, P. Xu, and L. Shangguan, "Audiogest: Enabling fine-grained hand gesture detection by decoding echo signal," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ser. UbiComp '16. Association for Computing Machinery, 2016, p. 474–485. [Online]. Available: https://doi.org/10.1145/2971648.2971736

[13] C. R. Pittman and J. J. LaViola, "Multiwave: Complex hand gesture recognition using the doppler effect," in *Proceedings of the 43rd Graphics Interface Conference*, ser. GI '17. Canadian Human-Computer Communications Society, 2017, p. 97–106.

[14] Y. Lee, Y. Zhao, J. Zeng, K. Lee, N. Zhang, F. H. Shezan, Y. Tian, K. Chen, and X. Wang, "Using sonar for liveness detection to protect smart speakers against remote attackers," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 4, no. 1, pp. 16:1–16:28, 2020.

[15] H. Feng, K. Fawaz, and K. G. Shin, "Continuous authentication for voice assistants," in *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2017, p. 343–355.

[16] S. Chen, K. Ren, S. Piao, C. Wang, Q. Wang, J. Weng, L. Su, and A. Mohaisen, "You can hear but you cannot steal: Defending against voice impersonation attacks on smartphones," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, 2017, pp. 183–195.

[17] Y. Dong and Y.-D. Yao, "Secure mmwave-radar-based speaker verification for iot smart home," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3500–3511, 2021.

[18] L. Zhang, S. Tan, and J. Yang, "Hearing your voice is not enough: An articulatory gesture based liveness detection for voice authentication," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017, pp. 57–71.

[19] Y. Meng, Z. Wang, W. Zhang, P. Wu, H. Zhu, X. Liang, and Y. Liu, "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)*, 2018, pp. 81–90.

[20] S. Fang, I. Markwood, Y. Liu, S. Zhao, Z. Lu, and H. Zhu, "No training hurdles: Fast training-agnostic attacks to infer your typing," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18, 2018, p. 1747–1760.

[21] L. Blue, L. Vargas, and P. Traynor, "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*, 2018, p. 123–133.

[22] M. E. Ahmed, I.-Y. Kwak, J. H. Huh, I. Kim, T. Oh, and H. Kim, "Void: A fast and light voice liveness detection system," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 2685–2702.

[23] L. Blue, H. Abdullah, L. Vargas, and P. Traynor, "2ma: Verifying voice commands via two microphone authentication," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, 2018, p. 89–100.

[24] C. Yan, Y. Long, X. Ji, and W. Xu, "The catcher in the field: A fieldprint based spoofing detection for text-independent speaker verification," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2019, p. 1215–1229.

[25] D. Makwana, "Amazon echo smart speaker (3rd gen) review," https://www.mobigyaan.com/amazon-echo-smart-speaker-3rd-gen-review, 2020.

[26] M. Tillman, "Google home max review: Cranking smart speaker audio to the max," https://www.pocket-lint.com/smart-home/reviews/google/143184-google-home-max-review-specs-price, 2019.

[27] Y. Gong, J. Yang, J. Huber, M. MacKnight, and C. Poellabauer, "ReMASC: Realistic Replay Attack Corpus for Voice Controlled Systems," in *Proc. Interspeech 2019*, 2019, pp. 2355–2359.

[28] G. Chen, S. Chen, L. Fan, X. Du, Z. Zhao, F. Song, and Y. Liu, "Who is real bob? adversarial attacks on speaker recognition systems," in *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, may 2021, pp. 55–72. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/SP40001.2021.00004

[29] E. Alepis and C. Patsakis, "Monkey says, monkey does: Security and privacy on voice assistants," *IEEE Access*, vol. 5, pp. 17 841–17 851, 2017.

[30] Y. Jang, C. Song, S. P. Chung, T. Wang, and W. Lee, "A11y attacks: Exploiting accessibility in operating systems,"

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '14. Association for Computing Machinery, 2014, p. 103–115. [Online]. Available: https://doi.org/10.1145/2660267.2660295

[31] G. Petracca, Y. Sun, T. Jaeger, and A. Atamli, "Audroid: Preventing attacks on audio channels in mobile devices," in *Proceedings of the 31st Annual Computer Security Applications Conference*, ser. ACSAC 2015. Association for Computing Machinery, 2015, p. 181–190. [Online]. Available: https://doi.org/10.1145/2818000.2818005

[32] L. Song and P. Mittal, "Poster: Inaudible voice commands," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. Association for Computing Machinery, 2017, p. 2583–2585. [Online]. Available: https://doi.org/10.1145/3133956.3138836

[33] N. Roy, S. Shen, H. Hassanieh, and R. R. Choudhury, "Inaudible voice commands: The long-range attack and defense," in *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. USENIX Association, Apr. 2018, pp. 547–560. [Online]. Available: https://www.usenix.org/conference/nsdi18/presentation/roy

[34] T. Sugawara, B. Cyr, S. Rampazzi, D. Genkin, and K. Fu, "Light commands: Laser-based audio injection attacks on voice-controllable systems," in *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, Aug. 2020, pp. 2631–2648. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/sugawara

[35] F. Kreuk, Y. Adi, M. Cisse, and J. Keshet, "Fooling end-to-end speaker verification with adversarial examples," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 1962–1966.

[36] H. Kwon, H. Yoon, and K.-W. Park, "Poster: Detecting audio adversarial example through audio modification," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '19. Association for Computing Machinery, 2019, p. 2521–2523. [Online]. Available: https://doi.org/10.1145/3319535.3363246

[37] M. Zhou, Z. Qin, X. Lin, S. Hu, Q. Wang, and K. Ren, "Hidden voice commands: Attacks and defenses on the vcs of autonomous driving cars," *IEEE Wireless Communications*, vol. 26, no. 5, pp. 128–133, 2019.

[38] T. Vaidya, Y. Zhang, M. Sherr, and C. Shields, "Cocaine noodles: Exploiting the gap between human and machine speech recognition," in *9th USENIX Workshop on Offensive Technologies (WOOT 15)*. Washington, D.C.: USENIX Association, Aug. 2015. [Online]. Available: https://www.usenix.org/conference/woot15/workshop-program/presentation/vaidya

[39] L. Schönherr, K. Kohls, S. Zeiler, T. Holz, and D. Kolossa, "Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding," in *26th Annual Network and Distributed System Security Symposium*. The Internet Society, 2019, pp. 1–15.

[40] I. Markwood, D. Shen, Y. Liu, and Z. Lu, "PDF mirage: Content masking attack against Information-Based online services," in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, Aug. 2017, pp. 833–847. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/markwood

[41] CMUSphinx, "Cmusphinx open source speech recognition," https://cmusphinx.github.io/, 2022.

[42] D. Povey, "Kaldi asr," https://kaldi-asr.org/, 2022.

[43] S. Shiota, F. Villavicencio, J. Yamagishi, N. Ono, I. Echizen, and T. Matsui, "Voice liveness detection algorithms based on pop noise caused by human breath for automatic speaker verification," in *Sixteenth annual conference of the international speech communication association (INTERSPEECH)*, 2015.

[44] Q. Wang, X. Lin, M. Zhou, Y. Chen, C. Wang, Q. Li, and X. Luo, "Voicepop: A pop noise based anti-spoofing system for voice authentication on smartphones," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 2062–2070.

[45] Matirx, "Matrix creator," https://matrix-io.github.io/matrix-documentation/matrix-creator/overview/, 2020.

[46] S. Studio, "Respeaker core v2.0. 2019," http://wiki.seeedstudio.com/ReSpeaker_Core_v2.0/, 2020.

[47] S. Shen, D. Chen, Y.-L. Wei, Z. Yang, and R. R. Choudhury, "Voice localization using nearby wall reflections," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2020, pp. 7:1–7:14.

[48] L. Zhang, S. Tan, Z. Wang, Y. Ren, Z. Wang, and J. Yang, "Viblive: A continuous liveness detection for secure voice user interface in iot environment," in *ACSAC '20: Annual Computer Security Applications Conference*. ACM, 2020, pp. 884–896. [Online]. Available: https://doi.org/10.1145/3427228.3427281

[49] G. Zhang, X. Ji, X. Li, G. Qu, and W. Xu, "Eararray: Defending against dolphinattack via acoustic attenuation." in *NDSS*, 2021.

[50] Z. Li, C. Shi, T. Zhang, Y. Xie, J. Liu, B. Yuan, and Y. Chen, "Robust detection of machine-induced audio attacks in intelligent audio systems with microphone array," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 1884–1899. [Online]. Available: https://doi.org/10.1145/3460120.3484755

[51] H. Abdullah, K. Warren, V. Bindschaedler, N. Papernot, and P. Traynor, "Sok: The faults in our asrs: An overview of attacks against automatic speech recognition and speaker identification systems," in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, pp. 730–747.

[52] H. Abdullah, M. S. Rahman, W. Garcia, K. Warren, A. S. Yadav, T. Shrimpton, and P. Traynor, "Hear "no evil", see "kenansville"*: Efficient and transferable black-box attacks on speech recognition and voice identification systems," in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, pp. 712–729.

[53] A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru, and F. Roli, "Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks," in *28th USENIX Security Symposium (USENIX Security 19)*, 2019, pp. 321–338.

[54] Y. Meng, J. Li, M. Pillari, A. Deopujari, L. Brennan, H. Shamsie, H. Zhu, and Y. Tian, "Your microphone array retains your identity: A robust voice liveness detection system for smart speakers," in *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 1077–1094. [Online]. Available: https://www.usenix.org/conference/usenixsecurity22/presentation/meng

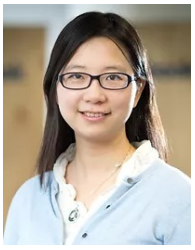
Yan Meng (Member, IEEE) is a Research Assistant Professor in Shanghai Jiao Tong University, China. He received his Ph.D. degree in Computer Science and Technology from Shanghai Jiao Tong University in 2021. He received his B.S. degree in Electronic and Information Engineering from Huazhong University of Science and Technology in 2016. His research interests include wireless network security and IoT security. He published more than 20 papers, including IEEE TDSC, IEEE TMC, IEEE WCM, ACM CCS, and USENIX Security. He received the 2022 ACM SIGSAC China Doctroal Dissertation Award.

Jiachun Li (Student Member, IEEE) is a Ph.D. candidate in the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. He received B.S. degree in Communication Engineering from Huazhong University of Science and Technology in 2020. His research interests include smart home security and smart healthcare security.

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3319833

Haojin Zhu (Fellow, IEEE) received his B.Sc. degree (2002) from Wuhan University (China), his M.Sc.(2005) degree from Shanghai Jiao Tong University (China), both in computer science and the Ph.D. in Electrical and Computer Engineering from the University of Waterloo (Canada), in 2009. He is currently a professor with Computer Science department in Shanghai Jiao Tong University. His current research interests include network security and privacy enhancing technologies. He published more than 70 international journal papers, including JSAC, TDSC, TPDS, TMC, TIFS, and 90 international conference papers, including IEEE S&P, ACM CCS, USENIX Security, NDSS, ACM MOBICOM. He received a number of awards including: ACM CCS Best Paper Runner-Ups Award (2021), IEEE TCSC Award for Excellence in Scalable Computing (Middle Career Researcher, 2020), IEEE ComSoc Asia-Pacific Outstanding Young Researcher Award (2014), Top 100 Most Cited Chinese Papers Published in International Journals (2014), Distinguished Member of the IEEE INFOCOM Technical Program Committee (2015, 2020), best paper awards of IEEE ICC (2007) and Chinacom (2008), WASA Best Paper Runner-up Award (2017). He is serving the editorial board for IEEE Trans. on Wireless Communications and program committees for top conferences such as USENIX Security, ACM CCS, NDSS and IEEE INFOCOM.



Yuan Tian is an Assistant Professor at the University of California, Los Angeles. Her research focuses on developing novel technologies for the security, privacy, and safety of modern and emerging systems. Her work has been published in top-tier security conferences (such as Oakland, CCS, Usenix Security, and NDSS), and has generated real-world impact as countermeasures and design changes directly resulting from my research have been integrated into platforms (such as Android, Chrome, Firefox, and iOS), and also impacted the security recommendations of standard organizations such as Internet Engineering Task Force (IETF). She received a couple of awards for her research, such as Google Research Scholar Award, NSF CAREER Award, NSF CRII award, Facebook Research Award, and Amazon AI Faculty Fellowship. Before joining UCLA, she was an Assistant Professor at the University of Virginia.

Jiming Chen (Fellow, IEEE) received the B.Sc. and Ph.D. degrees in control science and engineering from Zhejiang University, Hangzhou, China, in 2000 and 2005, respectively. He is currently a Professor with the College of Control Science and Engineering, and the Deputy Director of the State Key Laboratory of Industrial Control Technology, Zhejiang University. His research interests include the Internet of Things, sensor networks, networked control, and control system security.

Authorized licensed use limited to: UCLA Library. Downloaded on March 20,2024 at 00:30:49 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.