Proceedings on Privacy Enhancing Technologies ; 2019 (2):146–165

Chuhan Gao, Kassem Fawaz, Sanjib Sur, and Suman Banerjee

Privacy Protection for Audio Sensing Against Multi-Microphone Adversaries

Abstract: Audio-based sensing enables fine-grained human activity detection, such as sensing hand gestures and contact-free estimation of the breathing rate. A passive adversary, equipped with microphones, can leverage the ongoing sensing to infer private information about individuals. Further, with multiple microphones, a beamforming-capable adversary can defeat the previously-proposed privacy protection obfuscation techniques. Such an adversary can isolate the obfuscation signal and cancel it, even when situated behind a wall. AudioSentry is the *first* to address the privacy problem in audio sensing by protecting the users against a multi-microphone adversary. It utilizes the commodity and audio-capable devices, already available in the user's environment, to form a distributed obfuscator array. AudioSentry packs a novel technique to carefully generate obfuscation beams in different directions, preventing the multi-microphone adversary from canceling the obfuscation signal. AudioSentry follows by a dynamic channel estimation scheme to preserve authorized sensing under obfuscation. AudioSentry offers the advantages of being practical to deploy and effective against an adversary with a large number of microphones. Our extensive evaluations with commodity devices show that AudioSentry protects the user's privacy against a 16-microphone adversary with only four commodity obfuscators, regardless of the adversary's position. AudioSentry provides its privacy-preserving features with little overhead on the authorized sensor.

Keywords: keywords, keywords

DOI 10.2478/popets-2019-0024
Received 2018-08-31; revised 2018-12-15; accepted 2018-12-16.

Chuhan Gao: University of Wisconsin-Madison, E-mail: chuhan@cs.wisc.edu
Kassem Fawaz: University of Wisconsin-Madison, E-mail: kfawaz@wisc.edu
Sanjib Sur: University of South Carolina, E-mail: sur@cse.sc.edu
Suman Banerjee: University of Wisconsin-Madison, E-mail: suman@cs.wisc.edu

1 Introduction

The ubiquity of commodity devices with microphones and speakers have made audio-based sensing of human activity and health attributes feasible and attractive. Contact-free and audio-based sensing systems can sense fine-grained human gestures [12, 24, 42, 54, 60], movements [28, 30], behavior [10, 21, 22, 29], and health attributes [33, 37, 53, 55]. Akin to sonars, these audio-based sensing systems transmit near-ultrasound signals and analyze their reflections off the human body. As the mechanical motion of the individuals and their body parts shapes the reflected signal, the sensing system can identify a range of human attributes. For example, moving hands in a certain direction results in a distinct reflected signal which allows for identifying hand gestures. Also, fine-grained finger movements result in detectable changes in the reflected signal, enabling the development of new user interfaces (UIs) for devices. Chest movement due to breathing impacts the reflected signal which allows for estimating health situation of an individual.

Unfortunately, a stealthy adversary can passively exploit audio-based sensing at a low cost. While situated outside the user's environment, such an adversary can employ a set of microphones, or gain access to the vulnerable user-owned audio devices, to capture and analyze the audio sensing signals that reflect off the user. Capturing these reflections can lead to serious privacy and security threats [35]. These threats extend beyond identifying users' attributes to impacting their physical environment:

– Smart home systems and personal devices are increasingly employing audio sensing to accept human gestures as inputs [12, 24, 54]. In February 2018, Elliptic Labs announced a new SDK that adds gesture recognition capability to any programmable device with a speaker and microphone [6]. The adversary can stealthily record the audio sensing signals reflected by user's gestures and replay them to control the smart environment (e.g., light, doors and security camera). As audio reflections leak more information than gestures, an adversary can benefit from the ongoing sensing to discretely monitor user

behavior, such as their room occupancy and performed activities.
– Audio-based sensing enables contact-free monitoring of the health conditions of individuals. For example, researchers have demonstrated the efficacy of audio sensing in the detection of sleep apnea (this technology has already been licensed by a sleep health-care company) [33] and the monitoring of breathing patterns [37, 53, 55]. A stealthy adversary can leverage the ongoing sensing to remotely collect sensitive health information about the user.

Preventing privacy leaks from contact-free sensing has not been addressed previously, except for PhyClock [38], which thwarts the sensing capability of a *single*-antenna adversary in the context of wireless radio signal sensing. PhyClock employs a *single* obfuscator to prevent an adversary from correctly decoding the sensing signal and its reflections. It is well-known, however, that an adversary with multiple antennas can perform beamforming by focusing its receiver at a particular direction, thereby isolating and removing the effect of any obfuscation signal [49]. As microphone arrays are very cheap and accessible [1], a single obfuscator cannot prevent privacy leaks from audio-based sensing.

In this work, we first investigate how a passive and multi-microphone adversary can circumvent a single-obfuscator defense, even when separated by a wall. Then, we propose a mechanism that prevents a multi-microphone adversary's from monitoring the user's behavior, activities and health attributes from a distance. We consider two representative scenarios of audio sensing: hand-gesture recognition and breathing rate estimation. We find that an adversary, behind a wall and 6.5 meters far from the audio-sensing system, can correctly identify user's gestures and breathing rate. An adversary with sensitive microphones can eavesdrop from even a more extended range.

Naively adding more obfuscators does little to reduce privacy leaks from the multi-microphone adversary because of three challenges. First, there is the issue of practicality; the users are unlikely to purchase custom-built obfuscator hardware that are dedicated to protect their privacy, especially when privacy is a secondary concern for individuals [57]. Second, even if the first issue were to be overcome, determining how many obfuscators to deploy is not straightforward. The attacker's ability to resolve signal directions is directly proportional to the number of microphones it uses; trying to scale the number of deployed obfuscators with the adversary's microphones is a losing battle. Third, the authorized sensing of the user has to be preserved even when obfuscation is running, without requiring any hardware changes.

We propose AudioSentry, a system that prevents an unauthorized adversary from stealthily identifying user gestures and health attributes while preserving this capability for an authorized sensor. In particular, it protects the user against a strong passive adversary deploying an external set of microphones. AudioSentry addresses the aforementioned challenges by employing the widely available commodity devices such as smart TVs, laptops, and smartphones as a distributed obfuscator array. Instead of purchasing dedicated devices, the user can install a program/app on their *existing* device to enable AudioSentry. From a high-level perspective, AudioSentry has two components: Multi-directional beamforming and obfuscation signal cancellation.

Multi-directional beamforming: AudioSentry uses distributed beamforming (from the user's commodity devices) to transmit independent obfuscation signals across all different directions, some of which will cover the user. The reflected signal off the user contains a component from the known sensing signal and another from a randomized obfuscation signal. Since the signals received along other directions are independent of the obfuscation signal hitting the user, the adversary cannot isolate the reflections from the sensing signal. This design decision offers a significant advantage in the race between AudioSentry and the adversary; with a limited number of commodity devices, AudioSentry can thwart sensing capability of an attacker with a larger number of microphones.

Obfuscation Signal Cancellation: In the presence of multiple obfuscators, it is essential for AudioSentry to cancel their interference as to preserve authorized sensing. Interference cancellation requires estimating the audio channel between each obfuscator and the sensor which is challenging because of the user's movement. To address this problem, we propose a novel inverse channel estimation scheme that runs concurrently with sensing. In this scheme, each of obfuscators estimates the channel with the authorized sensor and communicates the estimated channel with the obfuscation signal over a secure out-of-band channel, such as WiFi. With access to such information, the authorized sensor, but not the adversary, can cancel the interference caused by obfuscation.

We have implemented AudioSentry using commodity speakers and evaluated it extensively against different adversaries. Our evaluation shows that, with only *four* commodity speakers, AudioSentry limits the information inferred about the user even when the adversary in-

creases its array 16 microphones. With 16 microphones, the adversary's gesture recognition accuracy is close to 21% (11% for a random guess) and its average error for breath rate estimation is more than seven breaths per minute (the average human breathing rate is between 12 and 20 bpm). AudioSentry achieves this privacy protection across various adversary placements/configurations and with little overhead on the authorized sensing.

In summary, this paper has the following contributions:

- demonstrating that a single obfuscator is ineffective against a modest and far multi-microphone adversary, even when a wall separates them (Sec. 3);
- a novel mechanism to employ a limited number of commodity devices to thwart the sensing of an attacker with a larger number of microphones (Sec. 5);
- a novel inverse channel estimation approach to preserving authorized sensing even with the presence of multiple obfuscators (Sec. 6); and
- the implementation and extensive evaluation of AudioSentry which highlights its privacy preserving capabilities and low overhead (Sec. 7).

2 Background

In this section, we provide a primer on audio-based sensing and signal obfuscation.

2.1 Audio-based Sensing

In an audio sensing scenario, there are two entities: the sensor and the target. The sensor has two components, a transmitter (speaker) and a receiver (microphone). While the microphone and speaker are usually co-located (e.g., the smartphone as a sensor), they do not have to be as such. The speaker emits a known sensing signal, $S(t)$ towards the target (e.g., an individual). $S(t)$ which can be modeled as a baseband signal $s(t)$ modulated at a carrier/sensing frequency f_c, where f_c is in the inaudible range, typically between 17 kHz and 20 kHz. The signal $S(t)$ travels through a physical channel to reach the microphone of the receiver. Within this physical channel, when $S(t)$ hits the target, the signal undergoes three phenomena: (1) the user absorbs part of it, (2) another part reflects back to the environment in different directions, and (3) the reflected part undergoes a Doppler shift (shift in the frequency) due to the mechanical movement of the target.

As a result, the sensor's microphone receives $r_\phi(t)$ along an incident angle ϕ, which can be expressed as:

$$r_\phi(t) = \alpha \times s(t) \times \cos(2\pi(f_c + f_D)(t - \Delta t)), \quad (1)$$

where α is the amplitude change, Δt is the delay and f_D is the Doppler shift. While these three parameters are each function of ϕ, we slightly abuse the notation by omitting ϕ from α, Δt and f_D in Eq. (1). A single microphone receives omni-directionally where there is no notion of ϕ. On the other hand, an array of microphones is able to separate received signal across different directions.

The parameters α, Δt and f_D, characterize the audio channel between the target and the sensor. In the audio channel, both free space propagation and reflection attenuate the signal amplitude, resulting in a smaller α. The propagation time (function of the distance), as well as the target movement, determine the delay Δt. Finally, the target movement determines the value of Doppler shift f_D. Audio sensing applications utilize these three parameters to characterize the target reflecting the sensing signal. For example, the Doppler shift created by hand movement can be used to identify different hand gestures.

2.2 Sensing Obfuscation

An obfuscator aims to hide these three channel parameters, at every signal path, to prevent the adversary from obtaining useful information. Only the authorized sensor should be able to sense the user and retrieve information such as gesture or biometrics. Consider the case for obfuscating the adversary's received audio signal $r_\phi(t)$. The obfuscator emits a signal $o_\phi(t)$ along direction ϕ so that the microphone receives: $r_o(t) = o_\phi(t) + r_\phi(t)$. The obfuscator designs $o_\phi(t)$ in such a way that the received signal $r_o(t)$ leaks little information about the parameters of $r_\phi(t)$: α, Δt, and f_D.

Amplitude gain α: Obfuscating the gain is straightforward as the obfuscator randomly modifies the amplitude of the obfuscation signal through changing its normalized volume between 0 and 1.

Delay Δt: Directly delaying the obfuscation signal in the digital domain introduces discrete delay (increments of 22 μs for $F_s = 44.1$ kHz audio sampling rate). Instead, an obfuscator can add an additional initial phase to the sensing signal to manipulate the delay with higher resolution. With an initial phase θ, the obfuscation signal

is:

$$o_\phi(t) = \cos(2\pi(f_c + f_D)t + \theta) \quad (2)$$

which can be organized as

$$o_\phi(t) = \cos\left(2\pi(f_c + f_D)\left(t + \frac{\theta}{2\pi(f_c + f_D)}\right)\right)$$

The introduced equivalent delay is $\frac{-\theta}{2\pi(f_c+f_D)}$, which is a continuous function of θ. The obfuscator can then modify θ to obtain fine-grained control over the equivalent delay, without actually delaying the signal.

Doppler shift f_D: To emulate Doppler shift effect, the obfuscator transmits signals with different frequencies to cover the possible Doppler shift range of the original sensing signal. Specifically, the obfuscator simultaneously transmits on the frequencies $\{f_l, \cdots, f_l + n\Delta f, \cdots, f_h\}$ to cover the reflected signal ($r_\phi(t)$). The Doppler shift can be estimated as $f_D = \frac{\Delta v}{c} f_c$, where Δv is the relative speed between reflector and receiver, c is the speed of sound, and f_c is the sensing frequency.

Obfuscation signal: In order to hide all three channel parameters, the obfuscation signal along a certain signal path, ϕ, can be given as:

$$o_\phi(t) = \sum_{n}^{f_l + n\Delta f \le f_h} \alpha_n \times \cos(2\pi(f_l + n\Delta f)t + \theta_n) \quad (3)$$

3 System and Threat Models

3.1 System Model

Our audio sensing scenario involves a sensor and the user. The sensor employs a commodity device, with a speaker and a microphone, to sense the user's behavior and attributes as explained earlier in Sec. 2. In the same environment, AudioSentry leverages the existence of other commodity devices, each with omni-directional speakers and microphones, to provide its privacy preserving features. AudioSentry requires those devices to be programmable and equipped with WiFi, such as smart TVs (or those with Chromecast), laptops, smart home assistants (e.g., Google Home mini), tablets, and smartphones. AudioSentry runs a software component on those commodity devices to transmit a specifically-coordinated obfuscation signal that prevents the unauthorized entities from sensing the user while preserving this capability for the authorized sensor.

3.2 Threat Model

An adversary aims to stealthily exploit the ongoing audio sensing to identify the user's behavior and health attributes. To that end, the adversary can be one of the following two types: (i) It deploys an array of microphones at a distance (e.g., at a neighboring apartment). The adversary performs its attack with a dedicated microphone array, and does *not* have access/control over the user-owned commodity devices. (ii) Alternatively, it could potentially compromise user-owned and microphone-enabled devices to achieve stealthy audio-based sensing. Such compromises can happen on mobile/browser platforms by abusing microphone permissions [13, 16, 17, 43]. Note than attacker can perform this attack with a single microphone, but in this paper, we consider a stronger threat model than can thwart single-obfuscator defenses. We assume that the adversary is capable of synchronizing its microphone array to achieve receiving beamforming, to identify the different signal paths (more on that in Sec. 3.3). The adversary can steer its beam direction electronically during offline processing, without needing to adjust the array configuration when eavesdropping.

Further, we assume a strong passive adversary that can recover the precise locations of the target user, the authorized sensor, and the obfuscators through blind source separation. Conventionally, blind source separation is a signal processing technique that helps the receiver to recover independent signal streams from unknown sources. In our model, the knowledge of all signal source locations allows the adversary to accurately perform beam steering without the need for source separation, yielding the strongest performance. Note that the precise user location is not absolutely necessary for blind source separation, but can significantly improve the performance of a strong adversary with such information.

The adversary aims to extract the physical channel information of the sensing signal reflected off the target user, which contains physical information about the user. By eavesdropping and analyzing the physical channel information, the adversary can obtain private personal information about the target users in their private spaces, such as health condition inferred from respiration rate information. Besides, the adversary can replay the recorded reflected signal to impersonate the user's gestures and behavior. This possible attack raises alarming security risks when audio sensing serves as a user-input mechanism for devices [54].

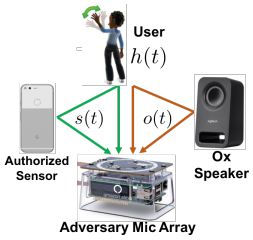

Fig. 1. Experiment setup of a 8-microphone adversary and a single obfuscator. Ox short for obfuscator.

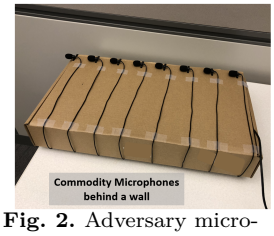

Fig. 2. Adversary microphone array, separated with the user by 6 meters and with a wall in between.

Finally, AudioSentry utilizes WiFi as a secure out-of-band (OOB) channel to synchronize the authorized sensor with the other commodity devices. We assume that all devices (authorized sensor and obfuscators) are connected to the same WiFi network that is properly secured, such that the adversary cannot intercept/modify the WiFi traffic. AudioSentry has no option but to trust those commodity devices; otherwise, its operation will be compromised and will fail to deliver the promised privacy provisions.

3.3 Privacy Threats from Passive Sensing

The state-of-art in protecting the privacy of users from a passive adversary focuses on employing a single omni-directional obfuscator, as explained in Sec. 2.2. In the following, we show that a single omni-directional obfuscator does little to prevent a multi-microphone adversary from correctly identifying the user's fine-grained gestures and accurately estimating the breathing rate. In addition, we show that the adversary can effectively compromise user's privacy even when a wall separates both of them.

3.3.1 Experiment Setup

Our experimental setup (Fig. 1) consists of four entities: the sensor, the user, an obfuscator, and an adversary behind a concrete wall with around 12 to 17 cm in width. The sensor has a speaker that emits the sensing signal and the obfuscator is another speaker emitting an obfuscation signal. Fig. 2 shows the adversary microphone array consisting of eight microphones, each with -30dB sensitivity. Each pair of adjacent microphones are separated by 7 cm. The cost of the microphone array is less than $120 (including the synchronization board and the microphones). The array is placed right next to the concrete wall, while the legit sensor, obfuscator device, and user are in another room.

Close to the sensor, a user moves his hand back and forth. The adversary aims to utilize its multiple microphones to cancel the effect of the obfuscation signal and estimate user's hand movement as well as the breathing rate. Behind-the-wall adversary demonstrates the feasibility of a practical attack, where the adversary's microphones are deployed outside user's room behind a wall or door. The distance between the adversary's array and the target is 6 m, with the user standing around 5.5 m away from the wall. Beyond this range, it becomes difficult for our low-cost microphone array to succeed in attack, but With more sensitive microphones, the adversary can considerably increase its sniffing range.

In the first scenario, the adversary tries to identify the user's gesture by observing the Doppler shift pattern. The sensor transmits a single tone at 16 kHz as $S(t)$, and the obfuscator transmits $o(t)$ such that:

$$o(t) = \sum_{n}^{f_l + n\Delta f \leq f_h} \alpha_n \times \cos(2\pi(f_l + n\Delta f)t + \theta_n) \quad (4)$$

where $f_l = 15.85$ kHz, $f_h = 16.15$ kHz, n is integer, $\Delta f = 5\ Hz$, and θ_n changes randomly between 0 and 2π every 100 ms.

In the second scenario, the adversary attempts to identify the user's breathing rate. We implement the breathing rate estimation system, ApneaApp [33], where the sensor plays Frequency Modulated Continuous Wave (FMCW) signals between 16 kHz and 18 kHz. Each chirp is 512 samples long, and the receiver groups ten chirps to improve FFT resolution similar To ApneaApp. The obfuscator plays the same signal as in Eq. (4), with f_l and f_h set to 16 kHz and 18 kHz, respectively (to cover the frequency range of the transmitted signal).

3.3.2 Signal Recovery

The objective of the adversary is to obtain $h_{u,a}$ which is the acoustic channel between the user and adversary. To that end, the adversary obtains each of $S(t)$, $o(t)$ and $r_a(t) = (\alpha \cdot S(t) + \beta \cdot o(t)) * h_{u,t}(t)$ by steering its receiving beam off-line towards the sensor, obfuscator and the user, respectively. The adversary utilizes its microphone array to achieve directional beamforming [48] by delaying the signal received by each microphone by a different amount, which coherently combines the signal from a certain direction.

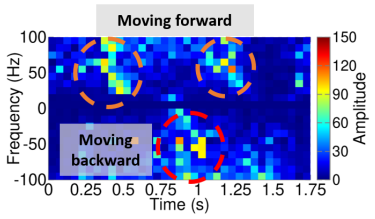

Fig. 3. Doppler shift pattern extracted by the adversary behind a wall 6 meters away from the user.

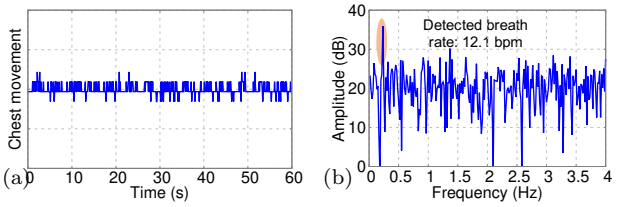

Fig. 4. Chest motion signal recovered by adversary behind a wall 6 meters away from the user, in (a) time domain (b) frequency domain

To recover the channel, $h_{u,a}$, the adversary removes the effect of $(\alpha \cdot S(t) + \beta \cdot o(t))$ from the received signal $r_a(t)$. In our experiments we use a special preamble to calibrate α and β. A real adversary could easily utilize past information to achieve the calibration. Nonetheless, we find that even if we do not consider these two parameters, the Doppler shift pattern can still be recovered, albeit with slight quality degradation. $h_{u,a}(t)$ is obtained as:

$$\hat{h}_{u,a}(t) = \mathscr{F}^{-1} \frac{\mathscr{F}\{r_a(t)\}}{\mathscr{F}\{\alpha S(t) + \beta o(t)\}} \tag{5}$$

where $\mathscr{F}$ represents the Fourier Transformation.

For both applications, we experimented with a number of state-of-the-art approaches [12, 15, 24, 33, 54, 55] to decide on the best performing techniques. We settled on the approach of Das et al. [15] for the gesture recognition application and that of Nadakumar et al. [33] for the breathing rate estimation application.

Extracting hand gestures. In the first scenario, a user moves his hand forward, backward and then forward. Fig. 3 shows the Doppler shift, extracted from $\hat{h}_{u,a}(t)$ over 1.75 seconds. It is clear from the figure how the Doppler shift pattern is consistent with the user movement, although an obfuscator was running.

Extracted breathing rate. In the second scenario, we estimate the channel $\hat{h}_{u,a}(t)$ for 60 seconds, during which the user is breathing normally. Using the estimated channel, we extract the chest movement as evident from Fig. 4. The figure shows the extracted chest movement pattern in the time and frequency domains. The frequency spectrum of the chest movement reveals a peak at 0.201 Hz, which corresponds to 12.1 breaths per minute (bpm); the ground truth was 11 bpm.

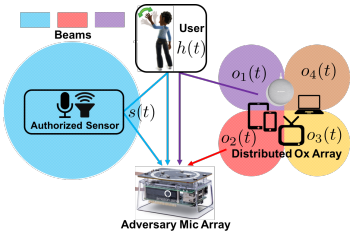

Fig. 5. High-level overview of AudioSentry. The adversary receives the combined reflections resulting from $s(t)$ (blue beam) and $o_1(t)$ (purple beam) hitting the user. Since the Ox array sends decorrelated beams in all directions, AudioSentry prevents the adversary from receiving $o_1(t)$ and estimating the channel.

We also conducted the same attack experiments with the adversary mic array placed in the same room with the user, with the same 6 m distance between them. Without the wall, the adversary receives much stronger audio signals from the user, legit sensor and obfuscator, leading to even better estimation accuracy.

4 AudioSentry Overview

AudioSentry addresses the above privacy threats by allowing only authorized clients to sense the user. From a high-level perspective, AudioSentry employs signal obfuscation to hide the sensing signal's reflections off the user from a multi-microphone adversary. Such adversary can focus its receiving beam into different signal paths to isolate the obfuscation signal and then remove its effect from the user reflections. AudioSentry counters this adversary by applying obfuscation along different signal paths, which necessitates using multiple speakers. It preserves user's privacy while by addressing three challenges: the number of speakers needed, the practicality of deployment, and preserving the authorized sensing.

Multi-beam Beamforming.
AudioSentry could attempt to obfuscate as many signal paths as possible, which locks it in an arms-race between its number of speakers and the adversary's number of microphones. Alternatively, AudioSentry only hides the signal paths corresponding to the user reflections while preventing the leak of the obfuscation signal (otherwise the adversary can cancel its effect). To hide those signal paths, AudioSentry employs a novel multi-beam beamforming mechanism. It generates *decorrelated* and randomized signals along the 360 degrees, as evident in Fig. 5.

As such, the reflections of the user will have two overlapping components: those resulting from the original (and known) sensing signal and those from the obfuscation signal. Because the obfuscation signal is decorrelated in different directions, the adversary cannot employ beamforming to isolate the obfuscation signal. Without access to the obfuscation signal hitting the user, the adversary cannot properly discern the reflections from the sensing signal.

Distributed Obfuscation with Commodity Devices. Generating decorrelated obfuscation beam requires multiple speakers. It is highly impractical to require the user install a set of dedicated obfuscator devices in their environment, given that privacy is usually a secondary concern [57]. AudioSentry meets this challenge through leveraging the commodity devices, such as smart TVs, home assistants, and PCs, in the user's environment to form a distributed array of audio obfuscators. Our evaluations of AudioSentry shows that it needs only four speakers to defend against an adversary with 16 microphones. Commodity devices that are already available in home environments can easily meet this requirement. Smartphones, tablets and laptops generally pack two speakers [2, 4]. Smart speakers[1] contain multiple speakers; for example, Apple HomePod has 7 speakers [3]. Also, connected TVs[2] usually have two speakers. These commodity devices only need to perform AudioSentry obfuscation when the audio sensing application is active, which preserves Ox device battery life. In addition, devices such as TV and smart speakers are typically plugged in, without battery life concern.

There are three problems associated with beamforming on a set of distributed and heterogeneous devices: device synchronization, beamforming weight computation and audio hardware heterogeneity. We address the first issue by retrofitting an existing mobile device synchronization solution that forms a distributed microphone array on a set of smartphones [48]. For the second problem, we first perform audio-based localization to determine the relative locations among obfuscators. Then, we estimate the phase offset of each transmitter to steer multiple beams to different signal paths. AudioSentry overcomes the third issue by performing a one-time initial calibration to compensate for audio hardware diversity at run time as will be evident in Sec. 6.

1 16% of Americans own a smart speaker [7].

2 58% of TV households in the US have a connected TV [5].

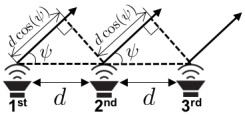

Fig. 6. Linear array directional beamforming example.

Preserving Authorized Sensing. As a result of obfuscation, the authorized sensor experiences interference from the obfuscation signals. We devise an interference cancellation mechanism that preserves the functionality of the authorized sensor. The interference, to be canceled, is a function of each of the obfuscation signals and the channel between each obfuscator and the sensor. The authorized sensor has to know each of the obfuscation signals and each of the channels as to cancel them. Achieving the first task is straightforward; AudioSentry utilizes WiFi as an OOB channel to communicate the obfuscation signal information from each obfuscator to the sensor.

For the second task, we propose a dynamic channel estimation scheme; it has to be dynamic because the user movement continuously changes the channel. One approach is to have each obfuscator transmit a known signal sequence so that the sensor can perform the channel estimation. This process introduces additional interference to the channel and requires more coordination between the obfuscators. Instead, AudioSentry leverages the channel reciprocity to have the sensor broadcast a training signal on a close, but different, frequency range from that of the sensing. After receiving the training signal, each obfuscator estimates the channel between itself and the sensor. Then, it sends the estimated channel parameters over the OOB channel to the sensor.

5 Multi-Beam Beamforming

In this section, we explain how AudioSentry employs multiple commodity devices to create a distributed beamforming array. As described earlier, this array generates decorrelated obfuscated signals along different directions.

5.1 Directional Audio Transmission

Audio beamforming employs multiple speakers (microphones) to transmit (receive) directional audio signals — *i.e.,* boost the transmitted (received) signal in specific directions and decrease it towards others.

Consider the N-speaker array, in linear arrangement, with distance d separating the speakers in Fig. 6. To

create a directional signal towards an Angle of Departure (AoD) ψ, the signal from each speaker is delayed by a specific time delay to compensate for the different path length of each of the signals. For example, Fig. 6 shows the path length difference of the transmitted signal from 1^{st} speaker relative to the i^{th} speaker's along ψ as $(i-1)d\cos\psi$. To constructively combine the signals along ψ, a classical beamforming algorithm delays the transmitted signal from the i^{th} speaker by $(i-1)\frac{d\cos\psi}{c}$ (c is the speed of sound in air).

We adapt this classical beamforming algorithm to fit AudioSentry's context. AudioSentry employs a set of heterogeneous and arbitrarily placed speaker elements to steer an obfuscation beam in different directions. In order to steer the beam towards a particular direction, AudioSentry picks a reference speaker and computes the desired delay τ between the reference and each of the other speakers based on their positions (estimating the position will be described later). As long as each speaker's signal is constructively combined with the reference speaker's signal along ψ, the overall signal would be boosted along this direction. Suppose the reference speaker and another speaker, i, have coordinates (x_0, y_0) and (x_i, y_i), then their path length difference along ψ (same angular coordination system as Fig. 6) is:

$$pd_i = \sqrt{(y_0-y_i)^2+(x_0-x_i)^2}\cos\left(\psi - \arctan\left(\frac{y_0-y_i}{x_0-x_i}\right)\right) \quad (6)$$

Then, the obfuscation signal to be transmitted from the speaker i has to be delayed by $\tau_i = \frac{d_i}{c}$. We follow the same approach of Sec. 2.2 to control the delays of the obfuscation signals by manipulating their initial phases.

5.2 Steering Multiple Beams

AudioSentry transmits obfuscation beams with different channel parameters (delay, magnitude, and frequency shift) along different AoDs. Suppose we aim to create multiple obfuscation signals $o_1(t), \cdots, o_m(t)$ with AoD as $\psi_1, \cdots, \psi_m$, respectively. We create each directional $o_m(t)$ following the method described in Sec. 5.1. For simplicity, we first assume each $o_m(t)$ is a narrow band signal $\cos(2\pi f_m t)$ and extend to wideband signal in Sec. 5.3. The signal transmitted by the i^{th} speaker to create $o_m(t)$ along ψ_m is:

$$o_m^i(t) = \alpha_m \cos(2\pi f_m(t-\tau_m^i)+\theta_i) \quad (7)$$

where τ_m^i is the delay created by the i^{th} speaker relative to the reference speaker in the way described in Sec. 5.1. The signal received at a far away position along direction ψ can be expressed as (propagation loss omitted):

$$o_m(t,\psi) = \sum_i \alpha_m \cos(2\pi f_m(t-\tau_m^i)+\theta_m+\frac{pd_i}{\lambda_m}2\pi), \quad (8)$$

where pd_m is the path difference as defined by Eq. (6) and $\lambda_m = \frac{c}{f_m}$. Next, we have each speaker i play $\sum_m o_m^i(t)$. With N speakers, we are able to steer $N-1$ independent beams. Since the parameters of each $o_m(t)$ are independent, AudioSentry creates different obfuscation signals along different directions.

5.3 Obfuscation Signal Design

The obfuscator samples α_m and θ_m from two uniform distributions in the range of $(0,1)$ and $(0,2\pi)$, respectively. It continuously varies both parameters to create time-varying channel parameters that are uncorrelated with the actual channel.

To cover the Doppler shift of the sensing signal, AudioSentry performs wideband obfuscation, which can be obtained by summing the expression of Eq. (8) over the different tones f_m covering the desired frequency range: $\{f_l, \cdots, f_l + n\Delta f, \cdots, f_h\}$. Since the sensor has a fixed location, the obfuscator sets the values of f_l and f_h according to the maximum velocity of the target, which depends on the sensing application. Δf – a function of the signal collection time τ – is chosen such that the obfuscation signal contains as much frequency components as possible. Given τ and the sampling rate F_s, the receiver collects $F_s \cdot \tau$ samples in the time domain. The frequency resolution, Δf , is then the width of each FFT bin: $\frac{F_S}{F_S \cdot \tau} = \frac{1}{\tau}$.

Even though the obfuscation signals parameters change continuously (which affects the obfuscation beam pattern), the obfuscation signal still consists of de-correlated beams in all the directions.

5.4 Enabling a Distributed Speaker Array

AudioSentry relies on commodity devices to form a distributed obfuscator array. Its beamforming technique has two requirements: *(i) the speakers in the array need to be synchronized to audio sample level and (ii) the relative positions of the speakers need to be known.* Fortunately, both of these problems have been solved for commodity devices [36, 48]; AudioSentry adopts both solutions to enable its operation as follows.

AudioSentry employs Dia [48] to achieve sample-level synchronization between the audio I/O clocks commodity devices. It first synchronizes the CPU clocks of the different devices by timestamping the WiFi beacons overheard from the access point. Then, it synchronizes the audio I/O clock of each device with its CPU clock. The average audio synchronization error of Dia for Nexus smartphones is between 1 to 3 audio samples at a sampling rate equal to 44100 Hz. AudioSentry performs the synchronization over WiFi, which we assume is secure. Aside from jamming, the adversary cannot manipulate the synchronization process to hinder AudioSentry's protection.

AudioSentry addresses the localization issue through BeepBeep [36], which can achieve an accuracy of 0.8 cm in ranging commodity smartphones. AudioSentry breaks the localization into a series of stages, by ranging each pair of its devices at each stage – taking $n(n-1)/2$ stages for n devices in the speaker array. AudioSentry needs to perform the relative localization of its devices only once unless some device moves (where it has to repeat for the moving device relative to two other devices with a known location). At each stage (involving a pair of devices), a device (initiator) emits a specially-designed sound signal (beep), while simultaneously recording its own and the peer's (responder) beep. By measuring the time between these two beeps, each device can compute the round trip time of the beep, consequently the distance between the two devices. With more than three devices in the array, AudioSentry can easily map the pairwise distances to locations relative to the reference device (every three devices form a triangle which can be defined using the lengths of its sides). Apart from BeepBeep, a recently proposed audio localization scheme *Sonoloc* [18] can also be applied in AudioSentry. Sonoloc localizes up to 100 commodity devices in a fully automatic fashion without user intervention, and only requires software changes to the commodity audio devices, similar to BeepBeep. Both these two solutions requires minimum user effort.

The original BeepBeep and Sonoloc are potentially vulnerable to the adversary playing forged beeps at a high power. The forged beeps can lead the initiator to miscalculate peer locations, significantly impairing the beamforming performance. AudioSentry addresses this problem by having the initiator prepend a preamble waveform to the beep. The initiator randomly decides on the preamble and communicates it to the responder over secure WiFi connection. The responder can verify whether the beep originated from the initiator or not. Note that the adversary recording the preamble and playing it back does not compromise the localization procedure, *as the forged response only deceives the initiator if it arrives earlier than the true response*. The latter can happen only if the adversary is closer to the initiator than the responder.

5.5 Privacy Properties

AudioSentry achieves its privacy protection through a two-step process. First, it hides the reflections off-the-user from the sensing signal by covering them with the reflections from the obfuscation signal. Second, it uses beamforming to prevent leaking the obfuscation signal to the adversary.

Single-path Obfuscation. Since the distributed obfuscator array covers the 2π range, one obfuscator path is guaranteed to hit the target user. With the user not being a perfect mirror, the obfuscating signal hitting the user deflects over a wide angle range (almost π as our experiments show). In practice, the obfuscators are distributed in the environment around the user, and each of their signals deflects around the π range, together covering the whole 2π range almost for certain. As a result, the adversary receives on the path between it and the target user: $r(t) = (\alpha \cdot S(t) + \beta \cdot o_i(t)) * h_{u,a}(t)$. The received signal contains two components, the reflections from the sensing signal and those from the obfuscation signal. Since $o_i(t)$, by design, is independent from $S(t)$, the adversary cannot extract $h_{u,a}(t)$ if it does not have access to $o_i(t)$.

Obfuscation Signal Leak. The second task of AudioSentry is then to prevent the adversary from receiving $o_i(t)$. The adversary is only likely to succeed if its location allows it to be in the signal path of the obfuscation signal hitting the user; the obfuscation signal is not a zero-width beam. In the most extreme case, with a single obfuscator (Sec. 3.3), the two copies of obfuscation signal received by the adversary are highly correlated. The adversary is successful at sensing the user with accuracy.

Nevertheless, AudioSentry utilizes the multi-directional beamforming technique to create uncorrelated and non-overlapping beams towards different directions. As AudioSentry utilizes a distributed array, it can leverage more transmitters to create more non-overlapping and narrower obfuscation beams [32, 52].

Fig. 7 shows simulated beam patterns created by a linear obfuscator as well as a distributed array. Both arrays consist six obfuscators and generate five independent beams. The transmitters of the linear array are

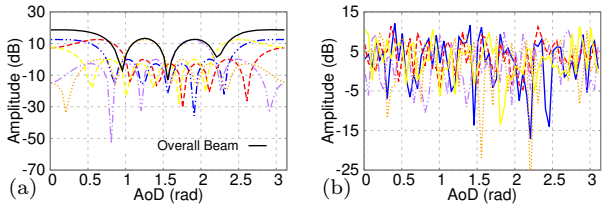

Fig. 7. Obfuscator beam pattern created by (a) linear array with half-wavelength separation (b) distributed array. Each colored line represents a different beam.

each separated by half-wavelength. On the other hand, the obfuscators of the distributed array are randomly located within a 4 m by 8 m area. In the textbook case of the linear array [50, 51] (which we show for comparison), it is evident that each beam is made up of one single wide main lobe and a series of very weak side lobes. In the distributed array (the real-world case of AudioSentry), the individual beams do not exhibit a well-defined shape. Instead, they contain a series of thin lobes with similar strengths pointing at different directions.

In the case of AudioSentry, which employs a distributed obfuscator array, it is difficult for the adversary to position itself in the same lobe as the target user to perform effective eavesdropping. The reason is that each of the obfuscation beams is a series of very thin lobes, instead of a single wide main lobe in the case of the linear array. We also confirm this observation in our evaluation findings, displayed in Fig. 9. There, we show that the adversary's performance is consistently low when we move its position around the target user. In addition, the adversary is unlikely to recover the original obfuscation signal even if it collects signals at various locations. Since the Ox signal directions are randomly changed, it is almost impossible for the adversary to keep moving to the correct angle to receive the correct Ox signal that it needs to remove from the received signal.

6 Preserving Authorized Sensing

The key to preserving authorized sensing is to cancel the obfuscation signal at the authorized sensor – essentially an interference cancellation problem. The interference (obfuscation signals) received by the authorized sensor can be expressed as:

$$I(t) = \sum_i o_i(t) * h_{O_i,S} \tag{9}$$

where $h_{O_i,S}$ represents the channel from O_i to the authorized sensor. The obfuscation signal transmitted by O_i, denoted by $o_i(t)$, can be known via the secure WiFi link between O_i and authorized sensor. The sensor has to know each $h_{O_i,S}$ to perform its sensing.

A common approach to estimate the audio channel $h_{O_i,S}$ is the Swept Sine technique [19], where the obfuscator's speaker plays an exponential time-growing frequency sweep signal as a channel sounding sequence. The sensor's microphone deconvolves the received signal with the original one (sounding sequence) to obtain the channel impulse response.

Reverse Channel Estimation.
Allowing each obfuscator to transmit a channel sounding sequence is not scalable. It introduces coordination problems between the obfuscators in the time and frequency domains, as each obfuscator has to perform the channel sounding on a separate frequency or time slot. Instead, AudioSentry leverages the audio channel reciprocity [19] to estimate the channel between each obfuscator and the authorized sensor. The authorized sensor's speaker plays a single channel sounding sequence to be received by all obfuscators' microphones. The obfuscators need not play multiple channel sounding sequences simultaneously. This operation is feasible as the obfuscators, which are the mobile devices, TV, laptops or smart speakers at home, are already equipped with microphones, so our solution requires no additional hardware.

AudioSentry places the channel sounding sequence at a different frequency range from the sensing signal so that the obfuscators can transmit $o_i(t)$ and receive the swept sine signal at the same time. Once the sounding sequence is received, the obfuscator estimates the channel and sends it to the authorized sensor via WiFi. Given the obfuscation signal described in Equation 3, AudioSentry uses a sweep sine signal in the frequency range of $f_l - 1000$ to $f_l - 500$ Hz. It is played every 100 ms by the authorized sensor, which provides ten channel measurements per second. The 500 Hz gap between the channel sounding signal and the obfuscation signal serves to prevent interference.

Obfuscation Signals Synchronization at the Sensor.
Although the obfuscators are synchronized for transmission beamforming, different distances between the obfuscators and the authorized sensor result in different arrival time of the signals. Hence, Eq. (9) should be rewritten as:

$$I(t) = \sum_i o_i(t - \tau_i) * h_{O_i,S} \tag{10}$$

where each $o_i(t)$ experiences a different delay τ_i. AudioSentry compensate for this delay by adding a preamble, a sweep sine signal spanning 200 Hz, to each obfuscator's transmitted signal $o_i(t)$ for the authorized sensor to synchronize them by performing correlation. The authorized sensor distinguishes each preamble by placing them on different frequencies. Since the obfuscators and the authorized sensor are connected via WiFi, the authorized sensor acts as a central controller to allocate all the obfuscators preambles to non-overlapping frequencies.

As commodity devices have limited audio sampling rate (mostly up to 44.1 kHz), which supports at most 22 kHz sound playback, this approach limits the maximum number of co-existing obfuscators. Nonetheless, each obfuscator's preamble only takes up 200 Hz bandwidth; a 2 kHz unused frequency range in the ultrasound range would make enough room for up to 10 obfuscators. Consider the ApneaApp [33] example where the system transmits a chirp between 18 kHz to 20 kHz as the sensing signal. The obfuscation signal and channel sounding preamble occupy additional 300 Hz and 1000 Hz, respectively. Thus, there is still 2.7 kHz within the inaudible range (16 kHz to 22 kHz) – enough to fit around ten obfuscators.

Removing the Obfuscation Signal.

Eventually, the sensor obtains the estimated channel $\hat{h}_{O_i,S}$ and the obfuscation signal $o_i(t)$ from each of the obfuscators. The authorized sensor then removes the estimated interference factor, $\hat{I}(t) = \sum_i o_i(t) * \hat{h}_{O_i,S}$, from its received signal. At this stage, the sensor can perform its sensing logic over the cleaned signal (without interference).

Commodity Speaker and Microphone Frequency Responses.

Although the audio channel between an obfuscator and the authorized sensor is reciprocal, their speakers and microphones are very likely to exhibit different frequency responses. Let the channel sounding sequence be $x(t)$, obfuscator's speaker and microphone frequency responses as D^O_{Tx} and D^O_{Rx}, authorized sensor's speaker and microphone frequency response as D^S_{Tx} and D^S_{Rx}. Then, the signal received in AudioSentry's reverse channel estimation is $D^S_{Tx} * x(t) * D^O_{Rx}$. In comparison, the signal received in the regular channel sounding process is $D^O_{Tx} * x(t) * D^S_{Rx}$.

AudioSentry requires a one-time initial calibration to compensate for the frequency responses of speakers and microphones. In the calibration process, the user places the authorized sensor next to each obfuscator, where AudioSentry has each device transmit a channel sounding sequence once at a time. By deconvolving the received signal with the sounding sequence, AudioSentry obtains $D^O_{Tx} * D^S_{Rx}$ when using the obfuscator as the transmitter, and $D^S_{Tx} * D^O_{Rx}$ the other way around. The user does not need to repeat this process as the device's audio hardware do not experience significant changes over time.

When AudioSentry intends to estimate $h_{O_i,S}$, the sensor plays the channel sounding sequence $x(t)$ and the obfuscator O_i receives $y_i(t)$. The sensor estimates the channel $\hat{h}_{O_i,S}$ as:

$$\hat{h}_{O_i,S} = y_i(t) *^{-1} x(t) *^{-1} (D^S_{Tx} * D^O_{Rx}) * (D^O_{Tx} * D^S_{Rx}) \quad (11)$$

Computational Overhead.

This interference cancellation scheme poses little overhead on the real-time performance of the authorized sensing. The additional computations of the authorized sensor involve convolving the estimated channel with the obfuscation signal for each obfuscator, and then subtracting them from the received signal. Thus the complexity is $\mathcal{O}(Q \log(Q) R)$, where Q is the length of obfuscation signal, and R is the number of obfuscators. In a typical deployment, these values are 4410 and 6, respectively.

We evaluate AudioSentry's computational overhead by measuring the total sensing time for an authorized sensor with and without enabling AudioSentry. We use a Google Pixel smartphone running Android 7.1.1 as the authorized sensor. With six obfuscators, the average additional delay introduced by AudioSentry is around 360 ms for both gesture detection and breath rate estimation. This delay is unlikely to hinder user experience as it is insignificant compared to the time the user needs to perform the gesture (1 to 2 seconds) and the breathing measurement time (30 to 60 seconds).

7 Implementation and Evaluation

We now present a prototype of AudioSentry along with its evaluation.

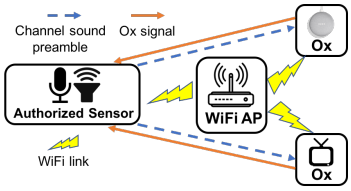

Fig. 8. AudioSentry's operation in sensing applications.

7.1 AudioSentry's Operation

Fig. 8 shows the basic operations of our AudioSentry's prototype during the sensing which follows this procedure.

Initiation process. The user controls AudioSentry through an end device, such as a smartphone. First, the user installs AudioSentry on the sensor device and designates it as the authorized sensor device. The user selects a group of commodity devices in the environment and connects them to the same WiFi access point. Next, she installs the AudioSentry app on those devices and designates them as the obfuscators. At this point, the user needs to perform the one-time bootstrapping process, described in Sec. 6, between each obfuscator device and the authorized sensor. The information about the frequency response of each device is stored at the authorized sensor to be used for obfuscation signal cancellation.

Obtaining obfuscator positions. AudioSentry asks the user to choose a fixed device as a reference point for the relative localization procedure. As described in Sec. 5.2, AudioSentry requires the relative positions of obfuscators to perform directional beamforming. When the obfuscators are initially deployed, AudioSentry utilizes the audio ranging approach described in Sec. 5.4 to perform obfuscator localization. It communicates the relative location information to the obfuscators via WiFi. It only repeats this process for a device that has been moved.

Application specific parameter setting. Depending on the particular sensing applications, AudioSentry sets the parameters for obfuscation signal, such as f_l, f_h and Δf, which we describe in more detail below.

Start the sensing application. The user can start the sensing application with her end device, which instructs obfuscators to start the obfuscation simultaneously. During the sensing, AudioSentry keeps performing the channel estimation and obfuscation signal cancellation at the authorized sensor as long as the sensing application is active. During this process, the obfuscator distributed synchronization process (Sec. 5.4) runs once every one second.

7.2 Experiment Setup

We use six commodity speakers as the obfuscator array; we distribute them randomly in a 4m × 8m living room, with random orientations to emulate the positioning of commodity devices in a practical home environment. For the adversary microphone array, we use the bottom microphones of multiple Pixel phones in a linear array formation, with every two phones separated by 7 cm. We use eight microphones to form the adversary's array unless otherwise specified. We synchronize the adversary microphones offline so that we can fully analyze the adversary's performance. Using the phones for the adversary array instead of a synchronized board was purely a logistical decision; it does not affect the fidelity of the reported results. Our authorized sensing system consists of two co-located Pixel phones, one acting as the transmitter and the other as the receiver.

We implement two audio sensing scenarios: gesture recognition and breathing rate estimation. For the former, we implement a recognition algorithm that identifies nine different hand-based gestures. This algorithm, akin to those proposed in the literature, extracts the Doppler shift pattern to recognize the gesture with a matching pattern. For breath rate estimation, we determine the breath rate as the highest peak in the frequency spectrum of chest motion within the normal breath rate range (below 20 bpm). In our evaluation, a user (one of the authors) performs a set of gestures while both the sensing system and AudioSentry are running.

The obfuscation signal design follows the description of Section 5.3. We set the values of Δf, f_l, and f_h so that the Doppler shift from the obfuscated signal covers that from the sensing signal's reflections. In our scenarios, the user's gesture and breathing induce a relative movement up to 2 m/s. A sensing signal at 20 kHz creates a Doppler shift equal to 118 Hz. Therefore, in our evaluation, we set f_l and f_h to $f_c - 150$ Hz and $f_c + 150$ Hz, respectively (f_c is the sensing frequency – depending on the application).

Following suit of earlier audio sensing applications, we set $\Delta f = 5$ Hz to ensure that the obfuscated Doppler shift pattern cannot be distinguished from that of the authorized sensor.

Although it's more practical for the adversary to place its microphones behind a wall, we only evaluate AudioSentry against the adversary in the same room with the user for the similar reason we explained in Sec. 3.3.2: It's much easier to recover the sensing signals without the wall's blockage. If AudioSentry is able to defend

this 'stronger' adversary, it could also defend against a 'weaker' adversary behind a wall.

7.3 Evaluation Results

Evaluation Metrics.
We evaluate our system using three metrics: mutual information, gesture recognition accuracy, and breathing rate estimation error. The latter two metrics are application-dependent; gesture accuracy measures the portion of correctly identified gestures at the adversary. The estimation error metric measures the absolute difference between the ground truth and the adversary's estimate of the breathing rate.

We also report the mutual information between the channel extracted by the adversary and the authorized sensor, each viewed as a random variable. For each measurement scenario, we frequently sample the sensor's and adversary channel estimates to obtain an empirically-derived distribution of both. The mutual information between two random variables characterizes the amount of information one variable reveals about another.

Definition 7.1. Mutual Information: Let the random variable representing the adversary's estimated channel be $\hat{H}$, and the random variable representing the authorized sensor's estimated channel be H. Then the mutual information can be computed as $I(\hat{H};H) = \sum_{\hat{h}}\sum_{h} p(\hat{h},h)\log(\frac{p(\hat{h},h)}{p(\hat{h})p(h)})$.

The mutual information metric quantifies how much of the sensing signal's reflections is embedded in the adversary's recovered channel. The higher the metric is, the more descriptive is the adversary's estimated channel of the user's sensing. This metric offers the advantage of being application *independent*. It quantifies the information leakage in the adversary's estimated channel, irrelevant to what signal processing algorithm the adversary employs after obtaining the channel.

For each experiment, with a different location and number of adversary microphones, we record the channel between the adversary and the user along with all the beams the adversary's array can discern. Then, we apply Eq. (5) to obtain the adversary's channel using the known sensing signal and each of the recorded beams. In the following, we report the results for the beam that results in a channel estimate that maximizes the mutual information with authorized sensor's extracted channel.

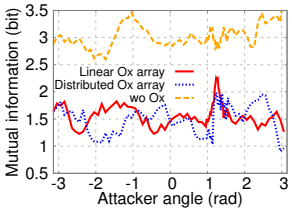

Fig. 9. Adversary mutual information when eavesdropping from different directions.

Adversary Position. We first investigate the impact of the adversary's position. In this experiment, we use six obfuscators and uniformly move the position of the adversary array around the target user, who randomly performs one of the nine gestures (similar to the one shown in Fig. 3). The target user is in the center of the living room, and the separation between the user and the authorized sensor is 2.5 m. We use two types of obfuscator array configuration, linear and distributed. The speakers have half-wavelength separation in the linear array, and the array is 1.5 m away from the user. Also, the array aligns with the user and adversary when the adversary is at the angle of 1.3 rad in Fig. 9. The distributed array's speaker positions are randomly chosen inside the room. Fig. 9 shows the measured mutual information when the adversary array is placed at different angles relative to the user. The authorized sensor is at the angle of 0 rad.

It is evident that both obfuscator arrays provide significant privacy protection, with the mutual information dropping in half when employing the obfuscators (from 3 bits to 1.5 on average). With around 3 bits of mutual information, without an obfuscator, the adversary achieves accurate detection results for both the gesture and breath rate. On the other hand, with around 1.5 bits mutual information, the adversary's estimated channel is completely corrupted (see examples in Fig. 10 (a) and Fig. 11), which causes the detection to fail.

Interestingly, we find that the linear obfuscator array fails to provide decent protection when the adversary, the user, and the obfuscator array are aligned. In this case (at 1.3 rad), the same obfuscation beam hits both the adversary and the user, because of the linear array's wide main lobe. This allows the adversary to remove the obfuscation signal from the user's reflection, thus revealing more information about the channel. On the other hand, the distributed array does not have this drawback, where we find that the mutual information suffers some fluctuation, but is consistently low regardless of the adversary's position. This result confirms the arguments explained in Sec. 5.5.

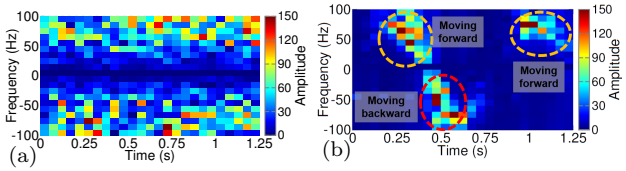

Fig. 10. Doppler shift pattern extracted by (a) adversary (b) authorized sensor. The adversary is located at angle 0 rad.

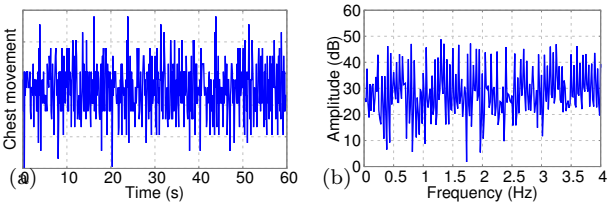

Fig. 11. Adversary extracted chest motion signal in (a) time domain (b) frequency domain. The adversary is located at angle 0 rad.

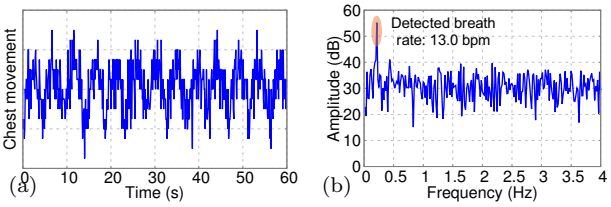

Fig. 12. Authorized sensor extracted chest motion signal in (a) time domain (b) frequency domain.

Next, we examine the application level performance of the adversary with distributed obfuscator array. Fig. 10 presents examples of the Doppler shift patterns extracted by the adversary and authorized sensor. Across all adversary positions, the average gesture recognition accuracy of the adversary and the authorized sensor are 12% and 94%, respectively. Note that for nine total gestures, an accuracy of 12% is equivalent to a random guess.

Fig. 11 shows an example of the chest motion signal extracted by the adversary, which contains no observable peak corresponding to the correct breathing rate. On the other hand, the chest motion signal extracted by the authorized sensor, plotted in Fig. 12, clearly shows a peak at 0.21 Hz (13.0 bpm), while the true breathing rate is 14 bpm. For all adversary positions, the average estimation error for the adversary and authorized sensor are 8.8 and 0.6 bpm, respectively.

Adversary array size. Then, we investigate the impact of the adversary's array size. We change the number of microphones on the array from 6 to 16 and measure the adversary's mutual information as well as the application level performance. For each array size, we also change the number of obfuscators. The authorized sensor is placed at the angle of -1.5 rad, 2.5 m from the user. The adversary array is placed at the angle which maximizes the mutual information, 1.5m from the user.

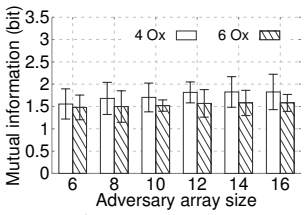

Fig. 13. Adversary mutual information with different array size. Error bars indicate standard deviation.

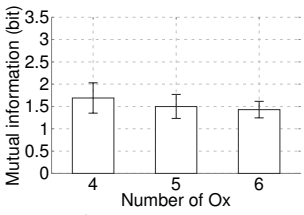

Fig. 14. Adversary mutual information with different number of Ox. Error bars indicate standard deviation.

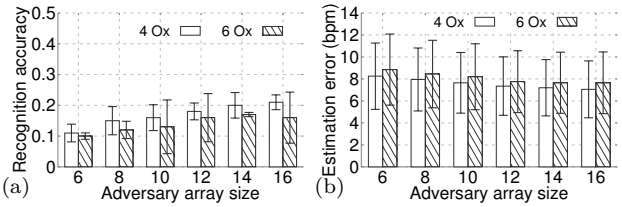

Fig. 15. Adversary's performance in (a) gesture recognition (b) breathing rate estimation with different array size

The user performs a gesture as the mutual information is being measured.

Fig. 13 shows that the mutual information slowly increases with the adversary's array size. However, this trend is far from being linear; increasing the array size provides limited benefits to the adversary. Although having a larger array offers more fine-grained receiving directionality, AudioSentry's multi-directional beamforming technique ensures that the adversary is unlikely to obtain the true obfuscation signal in the user body's reflection. As a result, the channel estimated by the adversary does not improve significantly with the array size. The same observation holds true for both 4 and 6 obfuscators; using more obfuscators provides slightly better privacy protection.

Fig. 15 shows the gesture recognition accuracy and breath rate estimation error. Not surprisingly, the detection performance improves slightly as the adversary array expands, albeit with a marginal benefit. Even with 16 obfuscators, the gesture recognition accuracy is 21% and 16% for 4 and 6 obfuscators, and the breath rate estimation error is higher than 7 bpm.

These results highlight an important aspect if AudioSentry. It provides privacy protection with a limited number of obfuscator against a large adversary array.

Number of obfuscators. We assess the effect of the number of obfuscators in the environment on the performance of the obfuscation and authorized sensing. We only change the number of obfuscators, while keeping the experiment setup as before (adversary with eight microphones, 6.5 m from the adversary and at a location that maximizes mutual information). Fig. 14 presents

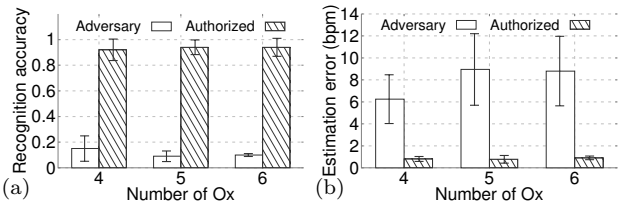

Fig. 16. Adversary and authorized sensor's performance in (a) gesture recognition (b) breathing rate estimation with different number of Ox.

the adversary mutual information, where we observe a decrease as more obfuscators are used. Fig. 16 also shows similar observations: where the adversary's performance in these two applications is worse on average when we use more obfuscators. On the other hand, although increasing the number of obfuscators introduces more interference to the authorized sensor, our interference cancellation technique proves to be resilient to the number of obfuscators. As a result, we see in Fig. 16 that the authorized sensing performance is consistently high. With six obfuscators, the gesture detection accuracy is around 95%, and the breath rate estimation error is within one bpm.

Synchronization error. We further investigate how the synchronization error among distributed obfuscators could harm the obfuscation and authorized sensing performance. We still use six obfuscators and introduce additional random delays on each speaker to emulate possible synchronization errors that could happen in a practical commodity device array. For example, we create an average synchronization error of 3 samples by varying the delay randomly between 0 and 6 samples. For each obfuscator, we vary the delay randomly every 100 ms.

Fig. 17 shows the adversary's mutual information under different levels of synchronization error. The mutual information increases with the error because the beamforming is affected by synchronization accuracy. Under perfect synchronization, each beam covers a distinct and non-overlapping range of the AoD. With higher error, the direction (AoD) of each beam may deviate from the intended direction, which causes different beams to overlap largely. As a result, The same beams will cover a broader range of AoD, which causes the overall obfuscation signal to be correlated along a wider range of angles. The adversary will enjoy an increased chance to cancel the obfuscation signal. Fortunately, as we explain in Section 5.4, the measured average audio synchronization error is fewer than two samples, which is more than sufficient to achieve sufficient obfuscation for AudioSentry.

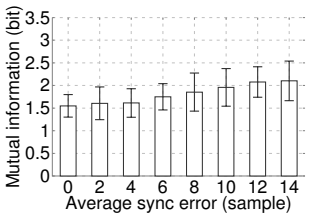

Fig. 17. Adversary mutual information under different synchronization error. Error bars indicate standard deviation.

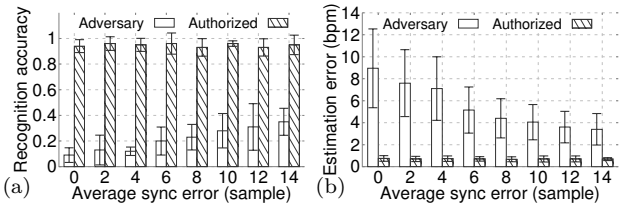

Fig. 18. Adversary and authorized sensor's performance in (a) gesture recognition (b) breathing rate estimation under different synchronization error.

In Fig. 18, we show the gesture recognition and breathing rate estimation accuracy as a function of the synchronization error. We find that although the obfuscation effectiveness degrades as the synchronization error increases in general, the impact if not significant when the average synchronization error is no greater than 4 audio samples. Under 4 samples of average error, adversary's average accuracy in gesture recognition is around 13%, and the average error in breathing rate estimation is around 7.2 bpm. As the synchronization error increases to 14 samples, these two metrics become 32% and 3.6 bpm, which provides a decent estimation of the user's activity and physical status.

We notice that the synchronization error hardly has any impact on the authorized sensing accuracy. This result highlights the efficacy of our synchronization approach introduced in Sec. 6 as part of AudioSentry's interference cancellation. The synchronization preamble of each obfuscator allows the authorized sensor to estimate the relative timing of each copy of obfuscation signal and perform interference cancellation.

8 Discussion

Privacy-aware Sensing. AudioSentry's obfuscation mechanism can also be built into a privacy-aware sensing system, instead of utilizing commodity devices as obfuscators. For example, the sensing system could add additional speakers as built-in obfuscators, akin to AudioSentry's distributed obfuscators. In such a case, some of AudioSentry's operation could be simpler; there is need for extra procedures for synchronization or localization

as the speakers will be co-located on the same board. Also, there will be no need to trust the commodity devices in the same environment, where an adversary can compromise multi-microphone devices such as Amazon Echo. The built-in obfuscator, however, presents some issues. Since it still needs to perform reverse channel estimation (to cancel the effect of obfuscation), each built-in obfuscator needs a co-located microphone, which significantly increases the hardware and space requirement. A privacy-aware sensor with six obfuscators, for example, requires seven microphone-speaker pairs. In contrast, AudioSentry achieves its privacy protection without requiring any dedicated hardware.

Active attacker. We design AudioSentry to thwart a strong passive attacker with multiple microphones and full knowledge of the environment. AudioSentry is equally effective against an active attacker operating within the same frequency range as AudioSentry's devices. Still, the active attacker could employ costlier ultrasound speakers and microphones to overcome AudioSentry's protection. Moreover, an active attacker could also affect the channel estimation procedure by jamming the obfuscators to prevent the authorized sensor from operating correctly. This attack, however, is not specific to AudioSentry; a regular active adversary can jam the sensor and prevent it from sensing the environment. AudioSentry could detect such an attacker, by having the sensor communicate the training sequence over the OOB channel to detect if there has been any interference in the channel.

Channel hopping. Channel hopping has been used in some wireless communication systems for security purposes. However, applying this technique in audio sensing is impractical for two reasons. First, an adversary can listen to the entire spectrum and monitor the sensing frequency as it hops – a problem that is likely to for electromagnetic-based sensing as well. Second, there is not sufficient audio bandwidth on most commodity devices. Such devices' operating frequency is below 24 kHz and audio sensing applications operate in between 18 kHz and 22 kHz (inaudible range). Most audio sensing applications require high bandwidth for higher accuracy. For example, ApneaApp requires at least 2 kHz bandwidth to accurately estimate chest movement, which makes frequency hopping difficult in practice.

Ultrasound Frequencies. One potential concern is the effect of AudioSentry's obfuscation signals on pets and wildlife, some of which are capable of hearing ultrasound signals. However, AudioSentry only operates in the same frequency range as the audio sensing application it protects; it does not create ultrasound "noise" in additional frequency ranges.

Eavesdropping private speech. An adversary that can sniff audio sensing signal is also likely to be able to eavesdrop private user conversations, a severe privacy threat. However, addressing both problems (eavesdropping on sensing vs. conversations) requires different approaches. The information in human speech is the audio waveform itself, while for sensing, the information is embedded in the audio channel, and the raw audio signal is not directly relevant. There are ongoing research efforts in protecting the privacy of human speech [40, 41], but it's out of the scope of AudioSentry.

9 Related Work

Audio sensing applications. The low propagation speed of audio signal has enabled a wide range of audio sensing applications, many on commodity devices. One of the most popular applications is to detect gesture and body movements [10, 12, 15, 21, 22, 24, 29, 34, 42, 54, 56]. SoundWave detects hand gestures by playing an inaudible tone, measuring, and analyzing the Doppler shift pattern [24]. FingerIO proposes an OFDM-based audio ranging design to track finger position with millimeter-level accuracy [34]. A more recent work, LLAP, can detect more fine-grained hand movements using a continuous wave radar, where it extracts the phase change of the audio signal to track finger position [54]. Shake and Walk demonstrates the feasibility of localizing and determining the direction of device movement using audio signals [26]. The similar principle also enables tracking and localization, such as tracking smartphones to emulate a mouse or for interactive applications [8, 12, 36, 46, 47, 59]. The ability to detect and distinguish finger strokes at different positions also enables an extended human-computer interaction interface [14, 25].

Audio sensing has been utilized in health monitoring, especially for breathing rate sensing, thanks to its ability to detect centimeter-level movements. For example, ApneaAPP is a breath monitoring application running on a smartphone [33]. As an FMCW radar, it plays a sweep frequency chirp in the inaudible human range ($18kHz \rightarrow 20kHz$), and perform FFT on the reflected signal to detect chest motion. This information can be used to detect diseases such as sleep apnea remotely in a non-intrusive way. Similarly, SonarBeat [55] uses a smartphone as a continuous wave radar ($18kHz$ to

$22kHz$) to measure the phase of the reflected audio signal to estimate chest motion, thus being able to measure breath rate. All of those techniques have not been designed with privacy in mind. They allow a passive adversary to infer a set of privacy-infringing information about the users, including their movements, behavior, interactions, and health attributes. AudioSentry targets these privacy threats.

Sensing and communication obfuscation. PhyCloak addresses the sensing privacy problem in the RF domain through an obfuscation technique [38]. It uses a single specialized full-duplex radio as both the obfuscator and authorized sensor, where the device forwards the signal for sensing and distorts the channel parameters to mislead the adversary. The single-obfuscator design makes PhyCloak ineffective against a multi-antenna attacker with directional beamforming capabilities. Also, extending PhyCloak's design to multiple obfuscators is hardly feasible, as it's impractical for regular users to deploy several dedicated full-duplex radios (commodity RF devices are not full-duplex).

Other works employ friendly jamming to protect the content of the communication between wireless devices. Implantable medical device (IMD) Shield protects the communication between an IMD and its authorized reader [23]. It jams the RF signal transmitted by the IMD to prevent eavesdropping and unauthorized connections using a full-duplex radio. Similarly, BLE-Guardian is an active obfuscator that jams the advertisement messages from Bluetooth Low Energy (BLE) devices to prevent privacy leakage [20]. Due to their single jammer nature, both and similar techniques [9, 11, 27, 31, 39, 44, 45, 58, 61] are ineffective against a multi-antenna receiver, which could separate the obfuscation signal and the communication signals [49].

Sound injection to prevent audio recording. Recently, researchers have established that it is possible to play specially crafted sound in the ultrasonic range to inject an audio signal in lower frequencies into a microphone [40, 62]. This technique can thwart an eavesdropper attempting to record conversations. It, however, faces the same drawbacks of the other approaches when relying on a single obfuscator. Extending to a multiple-obfuscator system has the same challenges which we address in this work.

10 Conclusion

In this paper, we show that a multi-microphone adversary can overcome a single obfuscator from a distance and with a wall separates them. Taking advantage of commodity devices in the environment, AudioSentry performs multi-beam beamforming to de-correlate the obfuscation signals towards different directions. While effective with all types of audio sensing applications in principle, we extensively evaluate AudioSentry for the two most popular applications: gesture and breath rate sensing. Our evaluations show that AudioSentry can defend against an adversary with a large microphone array with only a small number of obfuscators, while still being able to preserve authorized sensing.

The design principle of AudioSentry can be extended to other domains as well, such as RF sensing and speech recording, to protect user's privacy against multi-receiver adversaries. For WiFi sensing, one problem worth investigating in future work is how to utilize commodity devices without full duplex capabilities to achieve obfuscation as well as preserving authorized sensing. For speech recording, we can adapt AudioSentry to leverage a mechanism such as backdoor [40].

ACKNOWLEDGEMENTS

We appreciate the anonymous reviewers for their insightful comments. This research was supported in part by Wisconsin Alumni Research Foundation, and the US National Science Foundation through CNS-1719336, CNS-1647152, CNS-1629833, CNS-1343363, and CNS-1838733.

References

[1] [n. d.]. Amazon Alexa Premium Far-Field Voice Development Kit. https://developer.amazon.com/alexa-voice-service/dev-kits/amazon-premium-voice. ([n. d.]).

[2] [n. d.]. Apple Ipad Tech Specs. https://www.apple.com/ipad-9.7/specs/. ([n. d.]).

[3] [n. d.]. HomePod reinvents music in the home. https://www.apple.com/newsroom/2017/06/homepod-reinvents-music-in-the-home/. ([n. d.]).

[4] [n. d.]. Pixel 2 Tech Specs. https://store.google.com/us/product/pixel_2_specs?hl=en-US. ([n. d.]).

[5] 2017. The Nielsen Total Audience Report: Q2 2017. (November 2017). Retrieved May 8, 2018 from http://www.nielsen.com/us/en/insights/reports/2017/the-

nielsen-total-audience-q2-2017.html

[6] 2018. Elliptic Labs Makes Smart Speakers More Intelligent with New INNER REFLECTION™ Ultrasound Virtual Sensor. (February 2018). Retrieved May 8, 2018 from http://www.ellipticlabs.com/2018/02/26/elliptic-labs-touch-free-ultrasound-gesture-technology-leveraging\-the-qualcomm-snapdragon-neural-processing-engine-2/

[7] 2018. The Smart Audio Report, Fall/Winter 2017 from NPR and Edison Research. (January 2018). Retrieved May 8, 2018 from https://www.nationalpublicmedia.com/smart-audio-report/

[8] Md Tanvir Islam Aumi, Sidhant Gupta, Mayank Goel, Eric Larson, and Shwetak Patel. 2013. DopLink: using the doppler effect for multi-device interaction. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*. ACM, 583–586.

[9] Daniel S. Berger, Francesco Gringoli, Nicolò Facchi, Ivan Martinovic, and Jens Schmitt. 2014. Gaining Insight on Friendly Jamming in a Real-world IEEE 802.11 Network. In *Proceedings of the 2014 ACM Conference on Security and Privacy in Wireless & Mobile Networks (WiSec '14)*. 105–116. https://doi.org/10.1145/2627393.2627403

[10] Andreas Braun, Stefan Krepp, and Arjan Kuijper. 2015. Acoustic tracking of hand activities on surfaces. In *Proceedings of the 2nd international Workshop on Sensor-based Activity Recognition and Interaction*. ACM, 9.

[11] James Brown, Ibrahim Ethem Bagci, Alex King, and Utz Roedig. 2013. Defend Your Home!: Jamming Unsolicited Messages in the Smart Home. In *Proceedings of the 2Nd ACM Workshop on Hot Topics on Wireless Network Security and Privacy (HotWiSec '13)*. 1–6. https://doi.org/10.1145/2463183.2463185

[12] Ke-Yu Chen, Daniel Ashbrook, Mayank Goel, Sung-Hyuck Lee, and Shwetak Patel. 2014. AirLink: sharing files between multiple devices using in-air gestures. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 565–569.

[13] Qi Alfred Chen, Zhiyun Qian, and Zhuoqing Morley Mao. 2014. Peeking into Your App without Actually Seeing It: UI State Inference and Novel Android Attacks.. In *USENIX Security Symposium*. 1037–1052.

[14] Seungeun Chung and Injong Rhee. 2016. vTrack: virtual trackpad interface using mm-level sound source localization for mobile interaction. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*. ACM, 41–44.

[15] Amit Das, Ivan Tashev, and Shoaib Mohammed. 2017. Ultrasound based gesture recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 406–410.

[16] Lucas Davi, Alexandra Dmitrienko, Ahmad-Reza Sadeghi, and Marcel Winandy. 2010. Privilege escalation attacks on android. In *international conference on Information security*. Springer, 346–360.

[17] William Enck, Peter Gilbert, Seungyeop Han, Vasant Tendulkar, Byung-Gon Chun, Landon P Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N Sheth. 2014. TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones. *ACM Transactions on Computer Systems (TOCS)* 32, 2 (2014), 5.

[18] Viktor Erdélyi, Trung-Kien Le, Bobby Bhattacharjee, Peter Druschel, and Nobutaka Ono. 2018. Sonoloc: Scalable positioning of commodity mobile devices. (2018).

[19] Angelo Farina. 2007. Advancements in impulse response measurements by sine sweeps. In *Audio Engineering Society Convention 122*. Audio Engineering Society.

[20] Kassem Fawaz, Kyu-Han Kim, and Kang G Shin. 2016. Protecting Privacy of BLE Device Users.. In *USENIX Security Symposium*. 1205–1221.

[21] Biying Fu, Dinesh Vaithyalingam Gangatharan, Arjan Kuijper, Florian Kirchbuchner, and Andreas Braun. 2017. Exercise Monitoring On Consumer Smart Phones Using Ultrasonic Sensing. In *Proceedings of the 4th international Workshop on Sensor-based Activity Recognition and Interaction*. ACM, 9.

[22] Biying Fu, Jakob Karolus, Tobias Grosse-Puppendahl, Jonathan Hermann, and Arjan Kuijper. 2015. Opportunities for activity recognition using ultrasound doppler sensing on unmodified mobile phones. In *Proceedings of the 2nd international Workshop on Sensor-based Activity Recognition and Interaction*. ACM, 8.

[23] Shyamnath Gollakota, Haitham Hassanieh, Benjamin Ransford, Dina Katabi, and Kevin Fu. 2011. They can hear your heartbeats: non-invasive security for implantable medical devices. In *ACM SIGCOMM Computer Communication Review*, Vol. 41. ACM, 2–13.

[24] Sidhant Gupta, Daniel Morris, Shwetak Patel, and Desney Tan. 2012. Soundwave: using the doppler effect to sense gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1911–1914.

[25] Chris Harrison, Julia Schwarz, and Scott E Hudson. 2011. TapSense: enhancing finger interaction on touch surfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 627–636.

[26] Wenchao Huang, Yan Xiong, Xiang-Yang Li, Hao Lin, Xufei Mao, Panlong Yang, and Yunhao Liu. 2014. Shake and walk: Acoustic direction finding and fine-grained indoor localization using smartphones. In *INFOCOM, 2014 Proceedings IEEE*. IEEE, 370–378.

[27] Yu Seung Kim and Patrick Tague. 2014. Proximity-based Wireless Access Control Through Considerate Jamming. In *Proceedings of the ACM MobiCom Workshop on Security and Privacy in Mobile Environments (SPME '14)*. 19–24. https://doi.org/10.1145/2646584.2646588

[28] Tao Li, Yimin Chen, Jingchao Sun, Xiaocong Jin, and Yanchao Zhang. 2016. iLock: Immediate and Automatic Locking of Mobile Devices Against Data Theft. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*. ACM, New York, NY, USA, 933–944. https://doi.org/10.1145/2976749.2978294

[29] Jian Liu, Yan Wang, Gorkem Kar, Yingying Chen, Jie Yang, and Marco Gruteser. 2015. Snooping keystrokes with mm-level audio ranging on a single phone. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. ACM, 142–154.

[30] Wenguang Mao, Jian He, and Lili Qiu. 2016. CAT: high-precision acoustic motion tracking. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. ACM, 69–81.

[31] Ivan Martinovic, Paul Pichota, and Jens B. Schmitt. 2009. Jamming for Good: A Fresh Approach to Authentic Communication in WSNs. In *Proceedings of the Second ACM Conference on Wireless Network Security (WiSec '09)*. 161–168. https://doi.org/10.1145/1514274.1514298

[32] Raghuraman Mudumbai, D Richard Brown Iii, Upamanyu Madhow, and H Vincent Poor. 2009. Distributed transmit beamforming: challenges and recent progress. *IEEE Communications Magazine* 47, 2 (2009), 102–110.

[33] Rajalakshmi Nandakumar, Shyamnath Gollakota, and Nathaniel Watson. 2015. Contactless sleep apnea detection on smartphones. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 45–57.

[34] Rajalakshmi Nandakumar, Vikram Iyer, Desney Tan, and Shyamnath Gollakota. 2016. Fingerio: Using active sonar for fine-grained finger tracking. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 1515–1525.

[35] Rajalakshmi Nandakumar, Alex Takakuwa, Tadayoshi Kohno, and Shyamnath Gollakota. 2017. CovertBand: Activity Information Leakage Using Music. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 3, Article 87 (Sept. 2017), 24 pages. https://doi.org/10.1145/3131897

[36] Chunyi Peng, Guobin Shen, Yongguang Zhang, Yanlin Li, and Kun Tan. 2007. Beepbeep: a high accuracy acoustic ranging system using cots mobile devices. In *Proceedings of the 5th international conference on Embedded networked sensor systems*. ACM, 1–14.

[37] Kun Qian, Chenshu Wu, Fu Xiao, Yue Zheng, Yi Zhang, Zheng Yang, and Yunhao Liu. 2018. Acousticcardiogram: Monitoring Heartbeats using Acoustic Signals on Smart Devices. In *Proceedings of the IEEE International Conference on Computer Communications (Infocom '18)*.

[38] Yue Qiao, Ouyang Zhang, Wenjie Zhou, Kannan Srinivasan, and Anish Arora. 2016. PhyCloak: Obfuscating Sensing from Communication Signals.. In *USENIX Annual Technical Conference*.

[39] Hanif Rahbari and Marwan Krunz. 2014. Friendly CryptoJam: A Mechanism for Securing Physical-layer Attributes. In *Proceedings of the 2014 ACM Conference on Security and Privacy in Wireless & Mobile Networks (WiSec '14)*. 129–140. https://doi.org/10.1145/2627393.2627415

[40] Nirupam Roy, Haitham Hassanieh, and Romit Roy Choudhury. 2017. Backdoor: Making microphones hear inaudible sounds. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2–14.

[41] Nirupam Roy, Sheng Shen, Haitham Hassanieh, and Romit Roy Choudhury. 2018. Inaudible Voice Commands: The Long-Range Attack and Defense. In *15th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 18)*. 547–560.

[42] Wenjie Ruan, Quan Z Sheng, Lei Yang, Tao Gu, Peipei Xu, and Longfei Shangguan. 2016. 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*. ACM, 474–485.

[43] Roman Schlegel, Kehuan Zhang, Xiao-yong Zhou, Mehool Intwala, Apu Kapadia, and XiaoFeng Wang. 2011. Soundcomber: A Stealthy and Context-Aware Sound Trojan for Smartphones.. In *NDSS*, Vol. 11. 17–33.

[44] Matthias Schulz, Francesco Gringoli, Daniel Steinmetzer, Michael Koch, and Matthias Hollick. 2017. Massive Reactive Smartphone-based Jamming Using Arbitrary Waveforms and Adaptive Power Control. In *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '17)*. 111–121. https://doi.org/10.1145/3098243.3098253

[45] W. Shen, P. Ning, X. He, and H. Dai. 2013. Ally Friendly Jamming: How to Jam Your Enemy and Maintain Your Own Wireless Connectivity at the Same Time. In *2013 IEEE Symposium on Security and Privacy*. 174–188. https://doi.org/10.1109/SP.2013.22

[46] Adam Smith, Hari Balakrishnan, Michel Goraczko, and Nissanka Priyantha. 2004. Tracking moving devices with the cricket location system. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*. ACM, 190–202.

[47] Zheng Sun, Aveek Purohit, Raja Bose, and Pei Zhang. 2013. Spartacus: spatially-aware interaction for mobile devices through energy-efficient audio sensing. In *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*. ACM, 263–276.

[48] Sanjib Sur, Teng Wei, and Xinyu Zhang. 2014. Autodirective audio capturing through a synchronized smartphone array. In *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*. ACM, 28–41.

[49] Nils Ole Tippenhauer, Luka Malisa, Aanjhan Ranganathan, and Srdjan Capkun. 2013. On limitations of friendly jamming for confidentiality. In *Security and Privacy (SP), 2013 IEEE Symposium on*. IEEE, 160–173.

[50] Barry D Van Veen and Kevin M Buckley. 1988. Beamforming: A versatile approach to spatial filtering. *IEEE assp magazine* 5, 2 (1988), 4–24.

[51] Junyi Wang, Zhou Lan, Chin-Sean Sum, Chang-Woo Pyo, Jing Gao, Tuncer Baykas, Azizur Rahman, Ryuhei Funada, Fumihide Kojima, Ismail Lakkis, et al. 2009. Beamforming codebook design and performance evaluation for 60GHz wideband WPANs. In *Vehicular Technology Conference Fall (VTC 2009-Fall), 2009 IEEE 70th*. IEEE, 1–6.

[52] Jue Wang, Deepak Vasisht, and Dina Katabi. 2014. RF-IDraw: virtual touch screen in the air using RF signals. In *ACM SIGCOMM Computer Communication Review*, Vol. 44. ACM, 235–246.

[53] Tianben Wang, Daqing Zhang, Yuanqing Zheng, Tao Gu, Xingshe Zhou, and Bernadette Dorizzi. 2018. C-FMCW Based Contactless Respiration Detection Using Acoustic Signal. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4, Article 170 (Jan. 2018), 20 pages. https://doi.org/10.1145/3161188

[54] Wei Wang, Alex X Liu, and Ke Sun. 2016. Device-free gesture tracking using acoustic signals. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. ACM, 82–94.

[55] Xuyu Wang, Runze Huang, and Shiwen Mao. 2017. SonarBeat: Sonar phase for breathing beat monitoring with smartphones. In *Computer Communication and Networks (IC-*

CCN), 2017 26th International Conference on. IEEE, 1–8.

[56] Hiroki Watanabe, Tsutomu Terada, and Masahiko Tsukamoto. 2013. Ultrasound-based movement sensing, gesture-, and context-recognition. In *Proceedings of the 2013 International Symposium on Wearable Computers*. ACM, 57–64.

[57] Ryan West. 2008. The psychology of security. $Commun. ACM$ 51, 4 (2008), 34–40.

[58] Matthias Wilhelm, Ivan Martinovic, Jens B. Schmitt, and Vincent Lenders. 2011. WiFire: A Firewall for Wireless Networks. In *Proceedings of the ACM SIGCOMM 2011 Conference (SIGCOMM '11)*. 456–457. https://doi.org/10.1145/2018436.2018518

[59] Sangki Yun, Yi-Chao Chen, and Lili Qiu. 2015. Turning a mobile device into a mouse in the air. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 15–29.

[60] Sangki Yun, Yi-Chao Chen, Huihuang Zheng, Lili Qiu, and Wenguang Mao. 2017. Strata: Fine-Grained Acoustic-based Device-Free Tracking. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '17)*. 15–28. https://doi.org/10.1145/3081333.3081356

[61] B. Zhang, Q. Zhan, S. Chen, M. Li, K. Ren, C. Wang, and D. Ma. 2014. $ssrPriWhisper$: Enabling Keyless Secure Acoustic Communication for Smartphones. *IEEE Internet of Things Journal* 1, 1 (Feb 2014), 33–45. https://doi.org/10.1109/JIOT.2014.2297998

[62] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. 2017. DolphinAttack: Inaudible voice commands. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 103–117.