“Alexa, Stop Spying on Me!”: Speech Privacy Protection Against Voice Assistants

Ke Sun
University of California San Diego
kesun@ucsd.edu

Chen Chen
University of California San Diego
chenchen@ucsd.edu

Xinyu Zhang
University of California San Diego
xyzhang@ucsd.edu

ABSTRACT

Voice assistants (VAs) are becoming highly popular recently as a general means of interacting with the Internet of Things. However, the use of always-on microphones on VAs imposes a looming threat on users’ privacy. In this paper, we propose MicShield, the first system that serves as a companion device to enforce privacy preservation on VAs. MicShield introduces a novel selective jamming mechanism, which obfuscates the user’s private speech while passing legitimate voice commands to the VAs. It achieves this by using a phoneme level jamming control pipeline. Our implementation and experiments demonstrate that MicShield can effectively protect a user’s private speech, without affecting the VA’s responsiveness.

CCS CONCEPTS

• **Security and privacy** → **Privacy protections**; • **Human-centered computing** → **Ubiquitous and mobile computing**.

KEYWORDS

Privacy Protection, Voice Assistant, Selective Jamming

ACM Reference Format:
Ke Sun, Chen Chen, and Xinyu Zhang. 2020. “Alexa, Stop Spying on Me!”: Speech Privacy Protection Against Voice Assistants. In *The 18th ACM Conference on Embedded Networked Sensor Systems (SenSys ’20), November 16–19, 2020, Virtual Event, Japan.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3384419.3430727

1 INTRODUCTION

Voice assistants (VAs), *e.g.*, Amazon Echo [1] and Google Home [2], can enable hands-free interactions between human and smart computing devices. They have become a mainstream user interface in the smart home ecosystem, and are widely adopted by emerging mobile devices, *e.g.*, wearable earbuds and virtual reality headsets. Market analysis reveals an installation base of more than 76 million [3], and 21% of US adults have a VA device in their homes [4].

Despite the surging popularity and the alluring ability to automate human life, VAs are sparking an outcry of privacy concerns. Officially, vendors advocate that these devices are programmed to record and send information to the cloud for processing only when they are activated by a wake word/phrase, *e.g.*, “Alexa!” or “OK Google!” [5, 6]. However, there exists no easy way for users to trust and enforce such behavior. In fact, there are numerous cases when

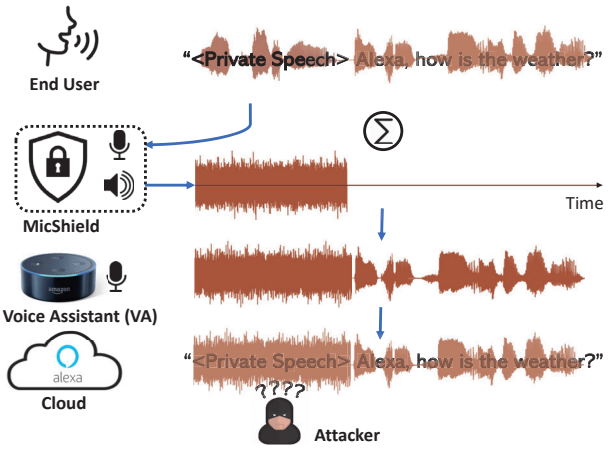


Figure 1: MicShield acts as a companion device with the VAs. The transducer of MicShield would start/stop emitting inaudible jamming signal based on the wake word. The unintended private speech before wake word is thus obfuscated before reaching the VA, which prevents attackers from eavesdropping the private conversations.

certain VA devices record private speech without user’s knowledge or consent. For instance, due to firmware bugs, an early version of Google Home Mini [7] kept recording users for 24/7 even without the wake word, and uploaded all the records to cloud servers [8]. Amazon reportedly hires human workers to transcribe recordings from their Echo devices, in the name of improving device performance and consumer experience. However, it has been found that majority of the transcribed clips were uneventful or not preceded by wake words. Users have no control on their unintended audio records once being sent to remote cloud [9]. Besides, it has long been a concern that certain government agencies may take advantage of the VAs as a means of pervasive surveillance [10].

In this paper, we seek to answer the question: *can we force the VAs to record the legitimate commands only, rather than private speech?* A straightforward way to protect speech privacy is to disable the VA through a physical mute button [11], and unmute only when the user needs to issue a command. However, this compromises the very first advantage of VAs, *i.e.*, convenient hands-free interactions. Besides, once the VA is compromised, the mute button is not trustable any more. Alternatively, one can adopt anti-eavesdropping approaches, which generate an interfering signal to jam the VA’s microphone [12, 13]. But this makes the VA deaf and irresponsive to any activation commands.

We propose *MicShield, a companion device which, for the first time, prevents VAs from recording private speech without affecting*

SenSys ’20, November 16–19, 2020, Virtual Event, Japan

ACM ISBN 978-1-4503-7590-0/20/11.
https://doi.org/10.1145/3384419.3430727

the VAs' normal functionalities. Figure 1 illustrates the basic threat model and defending mechanism of MicShield. Assuming the VA is untrustable, MicShield continuously emits a jamming signal to deafen the VA. Meanwhile, it keeps listening and stops jamming immediately when it senses the onset of a wake word. MicShield works offline, keeping all processing local and shredding voice data immediately afterwards. So it is not subject to the privacy risks from the always-listening, cloud-operated VAs.

To protect speech privacy without disturbing the VAs' daily usage, we must address two challenges. First, a straightforward way of continuous jamming will suppress not only private speech, but the wake word. This will cause VAs to become unresponsive to the subsequent voice commands. To overcome the dilemma, we leverage the fact that the wake word follows a fixed phoneme pattern, and even with the first few milliseconds jammed, the wake word can still be identified by Automatic Speech Recognition (ASR) algorithms. MicShield thus dynamically switches on/off jamming based on the likelihood of a wake word onset (*e.g.*, the initial few milliseconds of "Alexa!"). Once the likelihood exceeds a predefined threshold, MicShield suspends jamming to ensure the VA can hear majority of the wake word and the following voice commands. To avoid disturbing users, MicShield generates inaudible jamming signals, which however can be captured by regular microphones due to a well known non-linear aliasing effect [14].

The second challenge is to defeat the potential countermeasures based on microphone arrays, which exist in majority of the Off-The-Shelf (OTS) VAs. Whereas the microphone arrays have been used for sound localization [15], they can enhance the user's voice through acoustic beamforming, thus mitigating the effectiveness of MicShield's jamming. MicShield thwarts such potential countermeasures through a gain suppression method, which saturates the microphones and fully obfuscates the private speech. Our design employs acoustic waveguides to redirect the jamming signal towards each microphone. Meanwhile, these waveguides avoid the self-interference setbacks and ensure MicShield itself can still identify the wake words amid the jamming signal.

We built a prototype of MicShield with OTS components and a 3D printed shield. In accordance with the privacy protection assumption, we optimize both the computation and energy consumption to make the whole MicShield system work offline. Our experimental results demonstrate the effectiveness of MicShield in protecting private speech and countering potential threats. With MicShield, both ASR algorithms and human perception can only recognize less than 0.1% of the words in private speech. Besides, MicShield does not break the functionalities of VAs. It achieves nearly the same wake word response rate and speech Signal-to-Noise-Ratio (SNR). Furthermore, we show the MicShield's generalization across various wake words and VA devices provided by different vendors.

The main contributions of MicShield are as follows.

• We introduce a new concept to automatically protect speech privacy against always-on microphones by selectively jamming unintended private speech while passing intended voice command.

• We propose a novel speech processing pipeline which leverages the framewise likelihood to detect the onset of a wake words, thus realizing selective jamming.

• We propose a method to jam an entire microphone array using a single speaker, while avoiding self-interference.

• We prototype purely offline MicShield through low-cost OTS components, and validate its effectiveness in speech privacy protection without affecting the VAs' functionalities.

2 RELATED WORKS

2.1 Hidden Command Attacks and Defenses

Inaudible Voice Attacks and Defenses: Inaudible voice attacks[12, 14, 16, 17] use ultrasound or laser to generate imperceptible acoustic vibration signals that can be captured by microphones. BackDoor [14] shows that ultrasound can be recorded by traditional microphones in spite of the built-in low-pass filters. This is enabled by the hardware non-linearity of microphones, which creates aliased version of signals in the low frequency band. DolphinAttack [16] leverages a similar effect to attack the VAs through inaudible voice commands. [12] extended the attacking range from 5 ft to 25 ft, by using multiple ultrasonic speakers. A defending method was further proposed [12] to discriminate such attacks from speech signal, by identifying the spectrogram traces left by the non-linear effects. He *et al*. [18] further introduced a way to suppress the inaudible voice commands by using an ultrasonic transducer and interference cancellation methods. Inaudible voice command [17] can also be generated through the photoacoustic effect. By modulating the laser beam targeting a MEMS microphone, acoustic commands can be fabricated even at a distance of 110 m. To avoid disturbing users, MicShield uses inaudible voice as the jamming signal. Whereas previous work [12, 14, 16–18] investigated the attacks and defending approaches against *inaudible voice commands*, we focus on protecting *speech privacy* without handicapping the VAs.

Black-box Attacks and Defenses: Black-box attacks[19–22] generate adversarial audio samples that can be interpreted by VAs, but are unintelligible to human. For example, [19, 20, 22] generate white-noise-like malicious voice commands, which however can be interpreted as legitimate by VAs. These attacks can target different ASR algorithms including statistical learning models (*e.g.*, CMUSphinx [23]) and deep learning approaches. CommanderSong [21] embeds the voice command into songs to attack VAs. As for defense, certain post processing methods, *e.g.*, audio turbulence and audio squeezing, can single out the adversarial examples. On the other hand, Kumar and Zhang *et al*. [24, 25] investigated the interpretation errors made by ASR algorithms. The key observation is that different words may share similar phonemes which confuse ASR algorithms. An attacker leverages such systematic errors to unconsciously redirect users to malicious applications.

2.2 Audio Privacy Leakage and Protection

The always-on microphones on ubiquitous VA devices are imposing a looming threat to speech privacy. By penetrating the VA or the associated cloud, attackers can extract the sematic contents, and thus the personally identifiable information (PII), from the voice records.

Protection by Obfuscating the Audio Signal Waveform: One potential solution to audio privacy leakage is to obfuscate the unintended speech. For example, Tung *et al*. [26] proposed to protect private phone conversations against eavesdropping malware. They mix the private speech with mask signals which are known to a trusted server but unknown to eavesdroppers. The server can eliminate the mask through self-interference cancellations. However,

the masking sound needs to be shared in advance as a known audible secret key, which disturbs practical usability. Besides, a third trustable key generating server is needed, and it only uses symmetric key encryption. Therefore, the security guarantee breaks once the shared secrets and/or the key generator server are compromised. Chen *et al*. [13] designed a wearable jamming device using multiple ultrasonic transducers to protect speech privacy. This always-on jamming device make all nearby VAs malfunction and deaf to legitimate voice commands. In contrast, MicShield adopts a simple full-duplex phoneme-level selective jamming mechanism which ensures the wake words and voice commands are audible to the VAs.

Network Level Protection: An alternative protection mechanism is to identify and prevent the privacy leakage through transport layer packet filtering. Prior works, such as PrivacyProxy [27], ProtectMyPrivacy [28] and Meddle [29], introduced VPN proxies that detect the leakage of audio data packets by intercepting the outbound network traffic. Yet these systems cannot discriminate legitimate voice commands from unintended private speech. VoiceMask [30], on the other hand, used an intermediary between VAs and cloud to anonymize speech data. But the unintended semantic content can still be exploited from the private speech. Additionally, these mechanisms all rely on interception and interpretation of TCP/HTTP data. Yet all modern mobile devices are shifting toward encrypted SSL/TLS connections to prevent man-in-the-middle attacks [31, 32]. Thus, designing a VPN proxy to intercept, and thus control encrypted audio privacy data over transport layer, is practically challenging.

3 THREAT ANALYSIS

Threat Model: MicShield targets the scenario where adversaries use VA's always-on microphones to eavesdrop on *private speech*, *i.e*., the voice signals that are not preceded by a wake word. To establish the threat model, we assume the adversaries are powerful enough to: *(i)* access the unprocessed speech signals captured by the always-on microphones; *(ii)* run existing post processing algorithm to enhance the sound quality; *(iii)* use existing ASR algorithms [33] and human perception to infer the semantic content.

Protection Goals: Under the premise of not breaking the VAs' functionalities, MicShield aims to prevent the private speech from reaching the VAs. Consequently, the adversaries can no longer exploit semantics of private speech by compromising the VAs, sniffing the network traffics, or hacking the remote cloud. MicShield acts as a companion device to enforce speech privacy without modifying existing VAs' hardware/software. To achieve this goal, *first*, MicShield should work entirely offline to make sure that the manufacturer of MicShield imposes no privacy threat. *Second*, MicShield should ensure that the wake words still trigger a VA, whereas users' private speech preceding the voice command is jammed, shielded against the VA (Section 4). *Third*, MicShield needs to thwart powerful countermeasures that employ microphone arrays to enhance the speech while weakening the jamming (Section 5). We consider the worst case protection scenario: *(i)* The VA's received A-Weighting Sound Pressure Level (SPL) [34] is sufficiently high, but less than 75 dBA, known as the maximum SPL in daily conversations without harming human auditory [35]; *(ii)* The adversary knows the exact location of the speech source, so it can maximize the speech enhancement through array beamforming.

Security Guarantees and Evaluation Metrics: Unlike traditional cryptography-based security systems that define an exact security guarantee using the estimated computational time for breaking the system, providing similar guarantee for MicShield is challenging. However, this is also a common issue for non-cryptography systems. Inspired by the idea of Wyner wiretap model for a secure wireless system [36], we define the evaluation metrics as follows:

• **Mute Rate**: defined as the ratio between the jamming duration and the entire speech duration, excluding silent periods. Our design of jamming control policy focuses on *wake word mute rate* and *private speech mute rate*. Theoretically, an ideal design would have an 100% private speech mute rate and 0% wake word mute rate, to guarantee the speech privacy without reducing the responsiveness of the VAs. Practically, a less-than-100% private speech mute rate, caused by leakage of less importance phonemes, hardly exposes any threats to private content at word-level. Similarity, a slightly higher than 0% wake word mute rate does not necessarily fail the VAs, since the wake word recognition is imperfect even when no jamming signal exists.

• **Wake Word Misdetection Rate**: defined as the probability that a wake word cannot be correctly recognized. Our design needs to ensure a near equivalent wake word misdetection rate for the VAs, with and without MicShield jamming.

• **Jamming Effectiveness**: We quantify the jamming effectiveness by *PESQ (Perceptual Evaluation of Speech Quality)*, a metric for objective voice quality assessment commonly used by telecom operators [37], and *Speech Recognition Rate*, defined as the probability that the obfuscated speech can be correctly recognized by ASR or human perception. Theoretically, PESQ models the mean opinion score with a range between 1 (bad) to 5 (excellent) [37]. A typical acceptable range for VoIP applications lies between 3.8 and 5 [38], and PESQ less than 2 is known as the extremely low speech quality. To evaluate the speech recognition rate, we compare the ground truth and transcribed words by asking human participants to recognize the speech, and by using cloud based ASR services [33], including Amazon Transcribe[39] and Google Speech-To-Text (STT) [40].

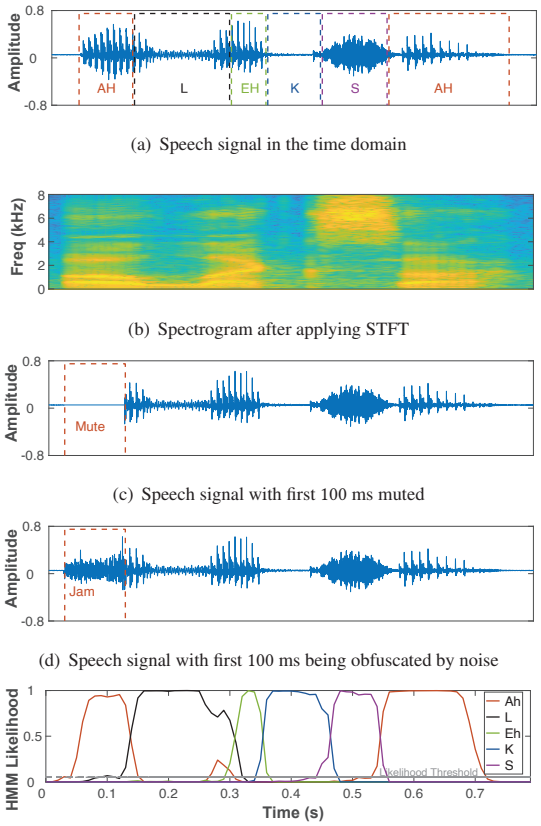
4 AUTOMATIC JAMMING CONTROL

In this section, we introduce the automatic jamming control algorithm that passes the wake words to VAs while obfuscating private speech. For ease of explanation, we use Amazon Echo Dot as the VA with "Alexa!" as the wake word [41]. Generalizations to other wake words and devices is straightforward and will be discussed in Section 7.4.

4.1 A Primer of Selective Jamming

To realize selective jamming, an intuitive way is to use a third-party "pre-wake" word detector [42]. The detector keeps jamming, and only stops when it hears a pre-wake word defined by the user. Immediately afterwards, it regenerates the real wake word "Alexa" through an inaudible channel, and passes it to the VA. However, our experiments reveal that, even a short wake word like "Alexa" takes at least an additional 500 ms. Thus, the inaudible "Alexa" will inevitably overlap with the user's voice query which follows the pre-wake word immediately – a conflict causing the VA to malfunction.

In contrast to word-level detection, we propose to use phoneme-level features to identify wake words from its early onset, which



(a) Speech signal in the time domain

(b) Spectrogram after applying STFT

(c) Speech signal with first 100 ms muted

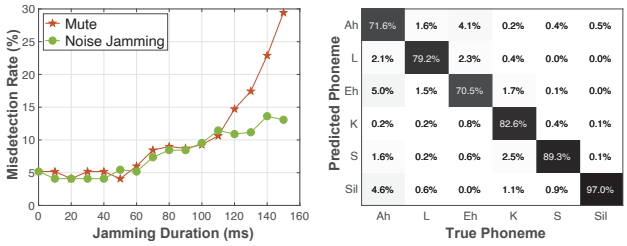
(d) Speech signal with first 100 ms being obfuscated by noise

(e) HMM likelihoods of phoneme states in HMM Lexicon Model

Figure 2: Analysis of "Alexa" speech signals. Wake word "Alexa" follows a specific sequence of phonemes and can be recognized when the first few milliseconds of speech is muted.

leads to negligible latency, thus avoiding the conflict. As an example, Figure 2(a) and 2(b) show the acoustic signal waveform of "Alexa" and the spectrogram after applying Short Time Fourier Transform (STFT), respectively. The "Alexa" follows the fixed phoneme sequence pattern, noted as /əˈlɛksiə/, where the first phoneme /ə/ lasts from 43 ms to 136 ms. For the VA to be able to recognize the wake word, MicShield needs to switch off jamming after identifying the first phoneme. To verify the feasibility of such phoneme-level selective jamming, we conduct the following experiment with a dataset [43] containing 369 "Alexa" utterances from 87 users with different accents.

■ **Can the VAs be activated when the initial part of the wake word is corrupted by jamming?** We evaluate the wake word misdetection rate of Amazon Echo Dot when the first few milliseconds of the wake word is corrupted. To identify the beginning of "Alexa", we use the CMUSphinx [23] phoneme forced alignment algorithm with resolution of 10 ms to compute the beginning time and duration of the first phoneme /ə/. Figure 2(c) and 2(d) illustrate the signal waveforms when the first few milliseconds of speech is muted and jammed, respectively. The jamming signal is a 4 kHz bandwidth white noise. Finally, the processed audio is played using a smartphone speaker.



(a) Wake word misdetection rate

(b) "Alexa" phoneme confusion matrix

Figure 3: Proof-of-the-concept experiments for jamming control pipeline design

Figure 3(a) shows the wake word misdetection rate versus predefined initial duration for mute and jamming cases. Surprisingly, we found that *even with the first* 60 *ms muted or jammed, the wake word can still activate the Echo Dot with 95% accuracy–the same as the case without mute/jamming (i.e., 0 jamming duration).* Beyond 110 ms, misdetection rate under jamming is slightly lower than muting. This is likely because the amount of residual semantic information that can be exploited from the jammed signal is more than that of muted signals. Therefore, if MicShield stops jamming within the first 60 ms, Echo Dot is still able to recognize the wake word and the subsequent voice commands. This is also true for other common wake words and VAs, as will be discussed in Sec. 7.4.

4.2 Jamming Control Pipeline

Figure 4 elucidates MicShield's jamming control pipeline built upon the above insights. *First*, we use framewise phoneme recognition to estimate the phoneme-level confidential scores that quantify the likelihoods of a wake word *onset* (Phase A). This enables us to control jamming with frame-level resolution. *Second*, we propose a Hidden Markov Model (HMM) based lexicon model to track the phoneme sequence (Phase B). This allows us to compute the likelihood of the wake word's *occurrence*. *Finally*, the ultrasonic transducer would start/stop jamming based on previous two outputs (Phase C).

■ **Phase A: Framewise Phoneme Recognition.** Upon receiving audio stream from the analogue front end, the voice signal is *first* sampled and segmented into individual frames at 16 kHz sampling frequency, 25 ms window, and 15 ms overlap. *Second*, we apply Mel-Frequency Cepstral Coefficients (MFCC) analysis [44] with 12 coefficients and 26 filter-bank channels to each frame. Together with the first order differential coefficients, *a.k.a.* the Deltas and the log-energy, this results in a 26-dimension feature vectors for each frame. *Finally*, to identify each phoneme from the framewise feature vectors, we train a vanilla Recurrent Neural Network (RNN) with one layer containing 275 hidden units and sigmoid activation function [44]. This trained model is then used to compute the likelihood for each possible phoneme, upon the input of each framewise feature vector. The English speech contains a limited set of 61 basic phonemes, as suggested in the TIMIT Acoustic-Phonetic Speech Corpus [45].

■ **Phase B: HMM Lexicon Model.** In Phase A, we use the RNN-based approach to recognize the possible phoneme from a sequence of framewise features. Following the lines of traditional ASR methods [44], we leverage an HMM-based lexicon model to specify the transitions of phoneme states, where the state space is

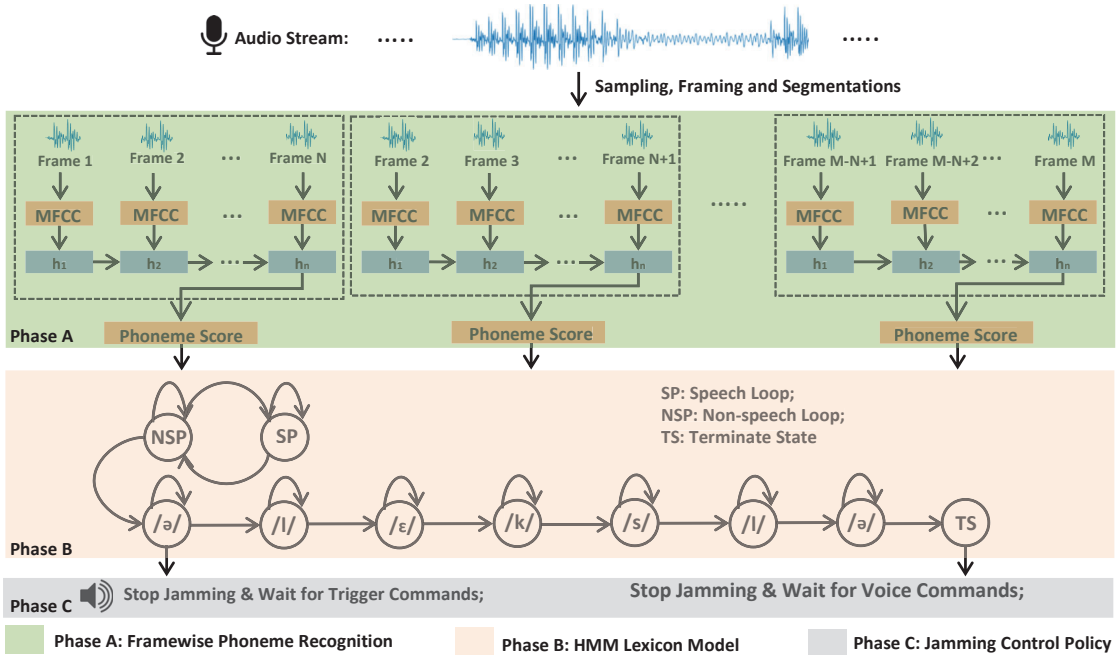


Figure 4: MicShield automatic jamming control pipeline

defined by the wake word's phonemes. For "Alexa", we define the state space as: {/ə/, /l/, /ɛ/, /k/, /s/}. Shown in Figure 4, once not following the predefined wake word sequence, the phoneme state would transition to the speech/non-speech loop. With HMM forward algorithm [46], we can compute the likelihood of each phoneme state over time, as shown in Figure 2(e).

■ **Phase C: Jamming Control Policy.** Our jamming control policy is based on the phoneme level likelihood estimations, shown in Figure 2(e). Once likelihood of the initial phoneme /ə/ exceeds the predefined threshold, MicShield will suspends jamming. However, it will immediately switch on jamming again if the subsequent phoneme states deviate from the wake word's predefined phoneme sequence. Otherwise, if the speech follows the predefined phoneme sequence, the HMM model will start decoding and identify the wake word. Subsequently, MicShield would stop jamming and allow the voice commands to pass through.

4.3 Minimizing Wake Word Misdetection

To avoid disturbing the VAs' basic functionalities, the jamming control pipeline must minimize the wake word misdetection rate.

Phoneme Level: Towards this end, the first challenge lies in the low accuracy of framewise phoneme-level recognition method compared to the word-level recognition [44]. With the aforementioned setup, the overall accuracy can only reach 64% for 61 different phonemes with input length of 15 frames. Fortunately, unlike generic ASR models, our phoneme recognition model can be trained to be sensitive to the specific phonemes associated with the wake words, *e.g.*, /əˈlɛksiə/. We thus harness this observation to enhance the detection accuracy for specific phonemes. Specifically, we fine-tune the model using a combination of the "Alexa" utterance dataset [43] and partial DARPA TIMIT Acoustic-Phonetic Speech Corpus containing phonemes in {/ə/, /l/, /ɛ/, /k/, /s/} [45]. Figure 3(b) shows the phoneme recognition result of /əˈlɛksiə/. With the fine tuned model, the accuracy increases from 64% to 78.7%.

Lexicon Level: Besides, we further minimize the wake word misdetection rate by decreasing the HMM likelihood threshold in Phase B. As a result, even if a phoneme does not achieve the highest phoneme likelihood in Phase A, the HMM-based lexicon model will still pass the frame. However, a low threshold leads to a low mute rate for private speech, which would in turn degrade the jamming effectiveness. Therefore, *an optimal HMM likelihood threshold should consider the trade-off between wake work misdetection rate and jamming effectiveness.* We will elaborate on this design trade-off in Section 4.4.

Another side effect of reducing the wake word misdetection rate is the increase of *false alarm rate*. However, this will not affect the VA's basic functionalities, since the speech signals mistakenly identified by MicShield as wake word will be passed to and reprocesssed by the VA. As long as the wake word is not identified, the VA will not be activated. In addition, although such speech signals are unprotected by MicShield, the false alarm rate is negligible (only once per 12 hours of speech as shown in Section 7.1), so the attacker can hardly exploit any sensitive semantic information.

4.4 Maximizing Private Speech Mute Rate

Our second goal is to ensure the unintended private speech is successfully obfuscated by the jamming signal, given high wake word detection accuracy as in Section 4.3. This requires us to maximize the private speech mute rate.

Recall that MicShield's selective jamming mechanism may still incur some phoneme-level speech leakage. For example, when the user is saying "Agree" (/əˈgri/), MicShield first stops jamming when it hears /ə/, and then resumes jamming immediately when it hears /g/. Thus, the phoneme /ə/ is leaked. However, because the remaining phonemes of "Agree" are still jammed, such occasional phoneme

leakage will not incur word-level privacy issues practically. In some extreme cases, when certain words' phoneme sequence resembles that of the wake word, these words cannot be properly protected. We have investigated the percentage of the words that have the same phoneme subsequences as "Alexa" in the CMU Pronouncing Dictionary [47]. Only 0.01% of 134000 words share the same first 4 phonemes, and most of these words are human names, *e.g.*, "Alexei", "Oleksy", *etc*. Therefore, MicShield can theoretically satisfy the audio privacy protection in a majority of practical settings.

In practice, the framewise phoneme recognition model is not prefect. To reduce the wake word misdetection rate, our fine-tuned model is sensitive to the phonemes appearing in wake words (Section 4.3), which in turn increases the *phoneme false alarm rate*, defined as the probability when a wake word phoneme is incorrectly identified. This will in turn degrade the private speech mute rate. For instance, MicShield may confuse the first phoneme of "Alexa" (/əˈlɛksiə/) with that of "Apple" (/ˈæp əl/), and determine not to jam the first phoneme of "Apple". The phoneme false alarm rate in Phase A achieves an average 25.62% for those associated with the wake word (see Figure 3(b)). To address this issue, we use the HMM lexicon model to track the phoneme sequence pattern based on that of the expected wake word. With this approach, MicShield will immediately restart jamming once identifying unexpected phoneme sequence. Thus, *even with a relatively high phoneme-level false alarm rate, the wake word recognition can still maintain a low false alarm rate*. Combined with the method proposed in Section 4.3, we thus are able to ensure the expected functionalities, while protecting the unintended private conversations.

4.5 When to Resume Jamming?

MicShield needs to resume jamming once the VA is back to the inactive mode. There are mainly two policies for VAs to return to the inactive mode.

- It detects a sufficiently long silent period after it is triggered.
- It identifies the end of the voice command based on the semantic contents.

Practical VA devices employ voice activity detection (VAD) methods and semantic content interpretation to realize these policies. MicShield needs to realize the same policies to determine when to resume jamming.

MicShield uses VAD methods [48] as in existing VAs to realize the first policy. For example, we empirically found that the Amazon Echo and Google Home use a 7 s and 8 s VAD threshold, respectively. Other VA devices' policy can be reverse engineered in the same way. However, the second policy is challenging for MicShield to implement, as the limited computational resources on the offline devices do not allow for analyzing language semantics. To circumvent this hindrance, MicShield resumes jamming immediately when it detects that the VA begins to respond to the voice command. With this measure, it protects subsequent periods when the user starts speaking again. To differentiate between the user's voice and the VA's response, we use the well known human/speaker sound detection methods in [49, 50]. Note that although the VAs are not trustable in our threat model, the adversarial VAs still have to respond to the user so that it can pretend to be normal. Furthermore, even the VA cheats the MicShield by intentionally not responding to the voice command, MicShield will still resume jamming after a few seconds of VAD detection.

Our scheme also supports the *follow-up mode*, where users can issue multiple requests interactively without repeating the wake word before each commands [51] by omitting the second policy, because the follow-up mode adopts the same waiting period as the first policy, *i.e.*, the VA will need to be triggerred by another wake word if no voice activity is detected across the waiting period. Interrupting commands, *e.g.*, "Alexa, Stop!", will also be identified by VA since such commands require the "Alexa" wake word even for the follow-up mode in our real-world test. Thus, MicShield will follow the same policy in Section 4.2 to recognize the early onset of the wake word and suspend jamming immediately.

5 PRACTICAL JAMMING DESIGN

5.1 Inaudible Jamming Sound

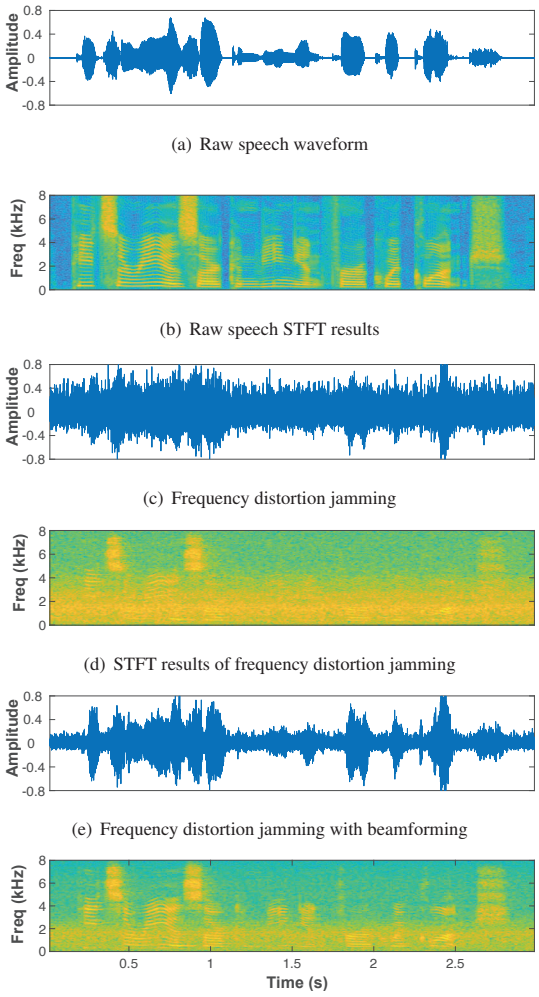
To avoid disturbing users, MicShield uses inaudible sound to jam the microphones. Similar mechanism has been employed recently [12, 14, 16] to send inaudible commands and hijack the VAs. Specifically, we use an ultrasonic transducer to transmit the ultrasound signals $S_{in} = \cos(2\pi f_h t)(\alpha + m(t))$, where $f_h = 40$ kHz is the high frequency carrier, and $m(t)$ is the low frequency jamming signal. Due to the microphone non-linearities, the recorded signals can be modeled as $S_{out} \simeq A_1 S_{in} + A_2 S_{in}^2$. After passing the low-pass filter and DC removal, the signals received by the microphone become $S_{mic} = A_2 \alpha m(t) + \frac{A_2}{2} m(t)^2$. These signals will be picked up and thus jam the microphones.

5.2 Jamming a Single Microphone

A single microphone can be easily jammed with traditional *frequency distortion jamming* method, which reduces the speech SNR by transmitting white/color noise. We verify the effectiveness by using a $0 \sim 4$ kHz white noise as jamming signal. To control the speech SNR, we gradually increase the noise amplitude, while summing the private speech and the noise with 16-bit quantization. We reuse the TIMIT speech dataset [45] for evaluation.

Figure 5(c) and Figure 5(d) show the private speech "Pizzerias are convenient for a quick lunch." time series waveform and STFT results from one microphone. Clearly, the low frequency components ($0 \sim 4$ kHz) are successfully obfuscated by noise. We further use PESQ and speech recognition rate to quantify the jamming effectiveness. Figure 6(a) shows that, by using frequency distortion jamming, the speech recognition rate is 1%, 0.3% and 0.9% for human perception, Amazon Transcribe [39] and Google STT [40], respectively, at -15 dB speech SNR. Under this condition, the PESQ stays at 1.15 on average, and the maximum PESQ is less than 1.6, *i.e.*, extremely bad speech quality (see Figure 6(b)).

The experiment implies that *frequency distortion jamming can effectively protect the speech privacy for the single-microphone VA when the speech SNR is less than* -15 *dB*. To cap the SNR below -15 dB under the highest speech SPL of 75 dBA (Section 3), the corresponding noise SPL should be above 90 dBA. To check the feasibility of this jamming noise volume, we measure the SPL generated by a single transducer at its maximum volume, with frequency range of 10 Hz $\sim$ 20 kHz. The SPL is sampled for each 1 cm in 7 different angles from 0°, to 90°, at a step of 15°. Figure 8(a) plots the resulting spatial distribution of SPL, where the contours are



(a) Raw speech waveform

(b) Raw speech STFT results

(c) Frequency distortion jamming

(d) STFT results of frequency distortion jamming

(e) Frequency distortion jamming with beamforming

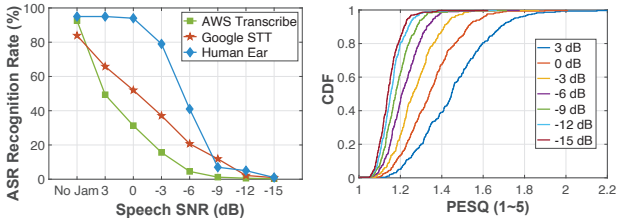
(f) STFT results of frequency distortion jamming with beamforming

Figure 5: Analysis of private speech: "Pizzerias are convenient for a quick lunch." Although frequency distortion jamming is effective to jam a single microphone ((c) and (d)), beamforming based attack can mitigate the frequency distortion jamming and enhance the private speech for attackers ((e) and (f)).

smoothed by 3D interpolation [52]. We see that *frequency distortion jamming achieves the required 90 dBA SPL, only when MicShield's ultrasonic transducer is placed within* 4 *cm towards the microphone*.

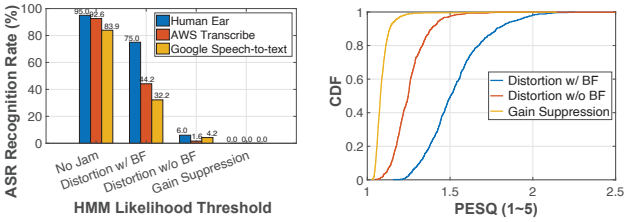
5.3 Jamming a Microphone Array

5.3.1 Beamforming based Attack. We now investigate how an attacker can leverage a microphone array as a countermeasure to mitigate the effectiveness of the aforementioned frequency distortion jamming. Since commercial multi-microphone VAs do not allow access to raw recorded signals, we use the ReSpeaker 6-microphone array [53] for experimental purpose. The speech is transmitted by a smartphone 30 cm away from the microphone array, and the received sound has similar volume as a human user 0.5 m away from the VAs (SPL ranges from 50 to 75 dBA). The jamming noise pattern is the same as in the previous experiment, and is generated by an ultrasonic transducer fixed at 5 cm above the center of the microphone



(a) Speech recognition rate (b) PESQ

Figure 6: Jamming effectiveness *v.s.* SNR. Frequency distortion jamming can effectively protect the speech privacy for the single-microphone VA when the speech SNR is less than −15 dB.



(a) Speech recognition rate (b) PESQ

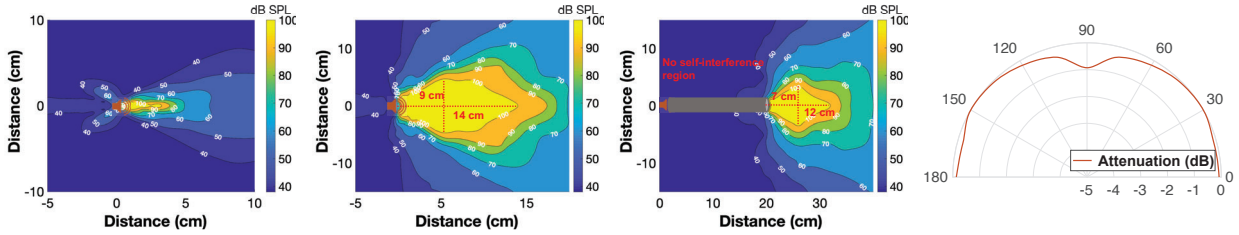
Figure 7: Jamming effectiveness for different jamming methods

array. As discussed in Section 3, we consider a scenario most advantageous to the attacker, assuming it knows the exact location of the speech source. Equivalently, under the setup as in Figure 11(b), the attacker knows the exact time difference of arrival (TDOA) of each microphones pair on the VA. Then it can perform the classical delay-and-sum beamforming [54, 55] to enhance the sound coming from the source location.

Figure 5(e) and 5(f) show that, with the beamforming countermeasure, the speech waveform and STFT results become much closer to raw audio signals than the case without beamforming. *The 6-microphone beamforming countermeasure can enhance the speech SNR by* 12 *dB*. The private speech recognition rate increases significantly, *i.e.*, to 75.0%, 44.2% and 32.2% for human perception, Amazon Transcribe and Google Speech-to-text, respectively. Some of the speech corpora even achieves a PESQ higer than 2.0 (see Figure 7(b)).

5.3.2 Gain Suppression Jamming. To effectively defeat the beamforming-based countermeasure, we explore an alternative *gain suppression jamming* method. The idea is to transmit high-volume sound to saturate the microphone, *i.e.*, force the microphone to reach the Acoustic Overload Point (AOP). AOP occurs when overwhelming input sound pressure causes the microphone output to be severely distorted [56]. In practice, the gain suppression jamming needs to address 2 dilemmas.

• **(D1) Dilemma between jamming noise volume and audibility:** Ideally, a high-power jamming noise can more effectively trigger gain suppression. However, a high output volume will trigger a nonlinear effect at the speaker's diaphragm and amplifier, making the jamming sound audible and disturb users [57]. To verify this phenomenon, we add a class D audio amplifier PAM8403 [58] to the ultrasonic transducer, which helps adjust the volume of jamming



(a) SPL 2D heat map for ultrasonic transducer (b) SPL 2D heat map for amplified ultrasonic transducer (c) SPL 2D heat map for amplified ultrasonic transducer with tube (d) SPL attenuation with MicShield

Figure 8: SPL heat map for ultrasonic transducer. Compared to (a) and (b), acoustic waveguide design in (c) extends the jamming space to make it possible to saturate the microphone array, and isolate the self-interference from MicShield.

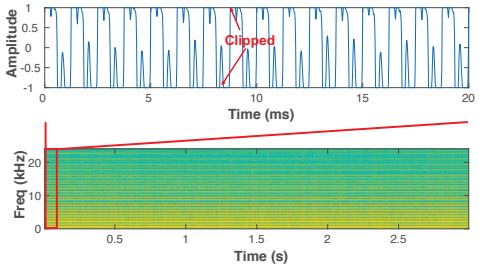
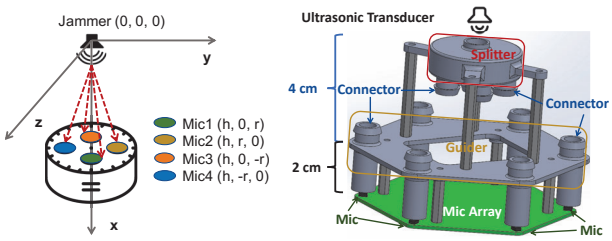


Figure 9: Acoustic overloading of received microphone with gain suppression method: (a) zoomed-in saturated jammed signal in time domain; (b) spectrogram of jammed signal.



(a) Single jammer geometry (b) 3D model for MicShield

Figure 10: Geometric model for MicShield design

noise. We then ask 5 volunteers to stay 0.5m away and check if the noise is audible. With 3 W input power, no volunteer hears the transducer. But when the input power reaches 4 W, 2 volunteers can hear the sound.

To avoid the audibility while ensuring gain suppression, we thus fix the input power of the transducer to 3 W and maximize the jamming volume using a *single-frequency jamming signal* close to the resonant frequency [59], where the transducer exhibits the maximum amplitude response.

Figure 9 shows the waveform of private speech (see Figure 5(a)) jammed by the single-frequency signal. Clearly, the microphone becomes saturated, and the speech signals are clipped and distorted into square-like waveform, losing the typical frequency-domain features as well. Our measurements show that the obfuscated signals have a low PESQ of 1.09 and a 0% speech recognition rate using Amazon Transcribe [39] and Google STT [40].

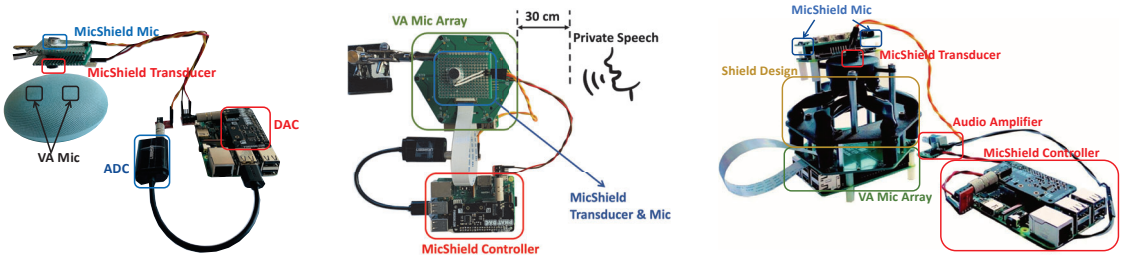
• **(D2) Dilemma between jamming noise volume and coverage:** Ideally, when gain suppression jamming is applied to each microphone on a microphone array, the speech signals will all become clipped into square-wave like waveform (see Figure 9) and completely unintelligible. Beamforming cannot recover the signals, as distortion occurs in the analog front-end. However, the acoustic transducer has a directional gain pattern with a narrow beam angle, whereas the multiple microphones on a VA are usually laid out to form a circular array. One could place the transducer away from the microphone array so as to expand the sound beam's angular coverage. However, under the 3 W power constraint, it is challenging to ensure a single transducer can generate sufficiently high SPL at all the microphones.

To elucidate this dilemma, we measure the SPL of a single transducer with the 3 W amplifier under the same setup of our previous SPL experiment. Figure 8(b) plots the resulting spatial distribution of SPL. Although the microphone hardware specifies 120 dBA SPL, we find that when the SPL exceeds 100 ∼ 110 dBA, it already causes complete gain suppression. Correspondingly, when the VA's microphone is placed within the yellow region in Figure 8(b), gain suppression occurs effectively.

As shown in Figure 8(b), with the amplifier (fixed to 3W to avoid audibility), the gain suppression works when the jamming transducer stays within 14 cm towards the microphone. However, for multi-microphone VAs, it is obvious that a single jamming source cannot cover the microphone array. Suppose the transducer is fixed h cm above the microphone array (circular array with radius r), as shown in Figure 10(a). Even with the amplifier, the largest radius of the gain suppression region is only 4.5 cm when $h = 5$ cm (Figure 8(c)). In contrast, many of the mainstream smart speakers (*e.g.*, Google Home and Apple HomePod) have a radius larger than 4.5 cm, which cannot be covered by the jammer. We present our solution in the next section.

5.3.3 Acoustic Waveguide Design. To address dilemma *D*2, we design a physical shield which can extend the coverage of a single ultrasonic transducer to jam large microphone arrays. Our basic idea is to leverage an acoustic wave guide to redirect the jamming signal to fully saturate the multiple microphones.

Acoustic Waveguide Design: Our acoustic waveguide splits the transducer's output signals using multiple silicone tubes and redirects them towards the microphones. We choose the 10 mm diameter tube (> half-wavelength (4.2 mm)) to prevent the volume attenuation due to thermo-viscous effects [60]. Figure 8(c) profiles the SPL



(a) Jamming w/o shield (b) Microphone array countermeasure (c) Jamming w/ shield

Figure 11: Hardware implementations and setups of MicShield

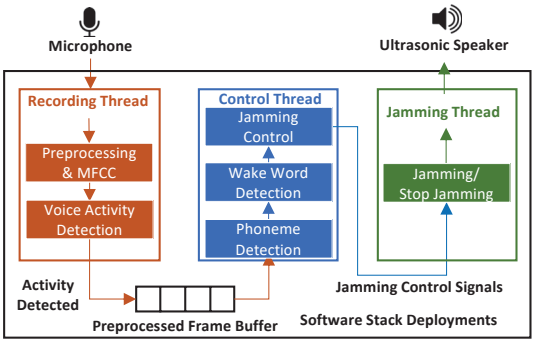


Figure 12: Software stack design and implementations

generated by a flexible silicone tube which is 20 cm in length and connected to a single ultrasonic transducer. It shows that, the sound is much more directional than the case without the tube. As a result, the acoustic waveguide also isolates the *self-interference* from the MicShield's ultrasonic transducer to its own microphone. So it can keep detecting the wake word while jamming the VA's microphone. Meanwhile, it improves the directionality and hence propagation distance of the sound signal (see Figure 8(c)). Therefore, if we can connect the transducer with multiple sound tubes, each tube can jam one microphone in an array.

Geometrical Design: We design a "acoustic multiplexer" to split the acoustic jamming signal across multiple tubes. Figure 10(b) shows the 3D mock-up design, which contains two parts, *i.e.*, "splitter" (top) and "guider" (bottom). The ultrasonic transducer transmits jamming signals from the top of the "splitter". The bottom of the "splitter" comprises multiple connectors which connects to the "guider" through tubes. Then, the "guider" uses another set of tubes to guide the jamming sound to each of the VA's microphones, thus enabling the gain suppression jamming. Note that the "guider" can be customized for different VAs, depending on the VAs' form factor and number of microphones. Figure 10(b) shows a typical geometrical design for the Respeaker 6-Mic circular array [53].

One might be concerned that the 3D printed shield may block the voice from the user. We conduct one experiment to quantify such degradation. We play the speech sound using smartphone and record the signals using the ReSpeaker 6-Mic Array [53] with or without our mock-up shield in 7 different angles from 0°, to 90°, at a step of 15°. Figure 8(d) shows the attenuation with the shield. We observe the degradation is only 1 dB even in the worst case when the sound source is on the top of the VA. This minor effect is unlikely to harm the VA's sensitivity in voice recognition.

Another practical concern for MicShield is the ultrasound safety issue. As suggested by World Health Organization (WHO), the human-exposure limits for 40 kHz airborne acoustic radiation should be less than 110 dB SPL when the duration of exposure does not exceed 4 hours per day [61]. As shown in Figure 8(d), MicShield guarantees that the transmitted SPL is always within the safety limits. Besides, our acoustic waveguide design further isolates the ultrasound and shrinks the high SPL region to prevent harm to users.

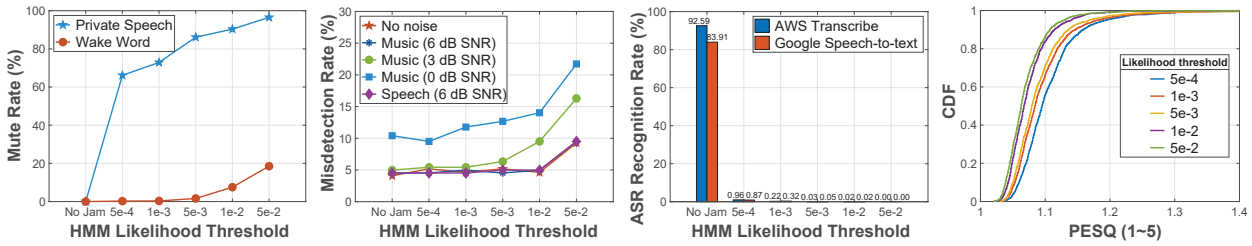
6 IMPLEMENTATION

6.1 Hardware

We implement MicShield using low-cost OTS components. Figure 11(a) shows the MicShield prototype for single-microphone case, comprising one ultrasonic transducer, one microphone and RPi. The RPi interfaces with a Pimoroni pHAT DAC 24-bit/192 kHz Sound Card [62] and then an ultrasonic transducer, in order to generate inaudible jamming sound. To minimize self-interference and maximize jamming, the MicShield ultrasonic transducer is facing away from its own microphone while towards the VA microphone. For multi-microphone case, Figure 11(c) shows the custom-built shield to guide the jamming sound to each microphone. To achieve a reasonable operating range in detecting wake words, MicShield uses ReSpeaker 2-Microphone Pi HAT supporting up to 3 m sensing distance [53]. The sampling rate is set to 16 kHz. The overall cost of single and multiple microphone schemes are around $25 and $35, respectively, exclude the RPi and 3D printed mechanical shield. We expect a great cost reduction and form factor enhancement by additional engineering efforts and integrating majority of components into System-on-Chip (SoC).

6.2 Software

Our software stack runs on the RPi, which implements the pipelines described in Section 4.2 and Figure 4. As shown in Figure 12, our implementation has three parallel threads, for *recording*, *control*, and *jamming* purposes. The *recording thread* captures and preprocesses the sound signals based on the policy in Phase A. To reduce energy consumption, this thread also determines whether it needs to emit jamming signals, with an energy-based voice activity detection scheme [48]. The *control thread* takes each MFCC frame features along with the previously preprocessed frames' MFCC features as input, and then determines whether to pass the jamming command to the jamming thread, based on the results of automatic jamming control algorithm (see Figure 4). The lightweight RNN and HMM in



(a) Mute rate (b) Wake word misdetection (c) Speech recognition rate (d) PESQ CDF

Figure 13: Micro benchmark using the wake word of "Alexa"

the automatic jamming control are implemented using Theano[63]. Upon receiving the jamming control, the *jamming thread* will start jamming. For single-microphone case, we transmit jamming noise to ensure the SPL at the receiver microphone being around 95 dBA, which is measured 3 cm away from the ultrasonic transducer. For multi-microphone case, we ensure the SPL of jamming noise received being approximately equal to 110 dBA, which is measured 1 cm away from the sound tubes.

7 EXPERIMENTAL EVALUATIONS

7.1 Micro Benchmarks

We first evaluate MicShield following the metrics in Section 3, and discuss the trade-off between the wake word misdetection and the jamming effectiveness. We use the TIMIT corpus [45] as the private speech dataset and the Alexa dataset [43] as the wake word dataset. We split the private speech data into training and testing set, containing 190 min and 70 min speech corpus for 630 and 168 users, respectively. Similarly, the Alexa dataset is split into 200 and 167 wake words for 50 users and 37 users, respectively. Since OTS VAs do not allow access to raw recorded signals, we use the ReSpeaker 6-Mic Array with RPi [53] to record both the private speech and the wake word sound in this experiment.

• **Mute Rate**: Figure 13(a) shows the private speech mute rate and the wake word mute rate under different HMM likelihood thresholds. A higher threshold, *e.g.*, 0.05, leads to high mute rate for both private speech and wake word. When the threshold falls below 0.001, the wake word's mute rate becomes almost 0%, whereas the private speech words' mute rate remains high (72%).

• **Wake Word Misdetection Rate**: Since commercial VAs do not expose their wake word detection algorithms, we playback the recorded wake word sounds under MicShield jamming to Amazon Echo Dot[41], in order to measure the end-to-end misdetection rate. Figure 13(b) shows that, compare to the case without MicShield, the wake word misdetection rate with MicShield's selective jamming does not degrade as long as the likelihood threshold is less than 0.01. When the likelihood threshold is 0.005, only 1/167 wake word from 37 users (16 females, 21 males), which can be recognized without MicShield, is misdetected with MicShield in the Alexa testing set.

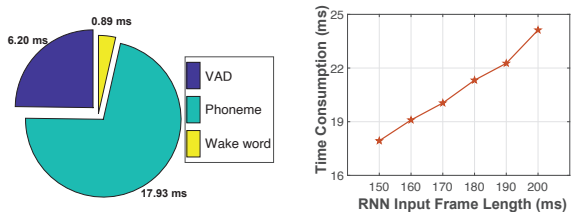
• **Wake Word False Alarm**: When testing the private speech, there is only 1 false alarm of wake word, out of the 12 hours of normal speech. This means the jamming effectiveness is virtually unaffected by wake word false alarms. Meanwhile, as we discuss in Section 4.3, the wake word false alarm will not affect the VAs basic functionality either.

• **Jamming Effectiveness**: We evaluate the jamming effectiveness by using PESQ and speech recognition rate. We use ASR algorithms (*i.e.*, Amazon Transcribe [39] and Google STT [40]) and human recognition to evaluate the jammed speech recognition rate. Figure 16(c) shows that the speech recognition rates for both ASR algorithms are less than 1% when the threshold is 0.0005 (corresponding to mute rate 66% for private speech). We found that the recognized words are mainly short words with the first phoneme close to /ə/, *e.g.*, "I", "a", "as", "all", "also", *etc*. In addition, human users cannot interpret the jammed speech either. The PESQ of all the jammed speech signals is less than 1.4, under different likelihood thresholds, as shown in Figure 13(d).

• **Impact of Environmental Noise:** To evaluate the robustness of MicShield, we mix the wake word sound plus private speech with two types of environmental noises, *i.e.*, speech noise and music noise, which come from AudioSet [64], an ambient noise dataset widely adopted in speech recognition research. We play these two types of the environmental noises by an additional loudspeaker while using MicShield. The SNR levels between private speech and noise vary between 6 dB, 3 dB, and 0 dB. Figure 13(b) shows the misdetection rate of wake word. When the HMM likelihood threshold is less than 0.005, the wake word misdetection rate remains the same as the case without MicShield. Here we omit the results where the speech SNR is below 3 dB, since the VA itself is no longer usable in such cases — the wake word misdetection rate exceeds 50% even without MicShield. Meanwhile, we find that the wake word false alarm and jamming effectiveness is barely affected in noisy environments, and instead is more sensitive to the HMM likelihood threshold.

• **Real-world Usage Experiments:** We deployed the VA with MicShield (see Figure 11(b)) across four different rooms, *i.e.*, bedroom, kitchen, living room, and bathroom, and asked the users to use the VA as usual for 5 days. 120/125 voice commands, including some interrupting voice commands, *e.g.*, "Alexa, stop!", are correctly responded by VA with MicShield. MicShield is not sensitive to the VA locations since we use the MFCC features, which is more related to the speech characteristics rather than environmental characteristics, as the framewise phoneme recognition input. The only 5 missed voice commands are due to the low-volume of the voice commands which can not trigger the jamming suspending policy.

To summarize, when the HMM likelihood threshold is 0.01*, MicShield is able to maximize the jamming effectiveness, achieving* 90.4% *mute rate and* 0.02% *speech recognition rate for private speech, without affecting the VA's ability to detect wake words even in noisy environments across different VA locations.*



(a) Contributions of latency (b) Frame length *v.s.* Time

Figure 14: Analysis of system time consumption

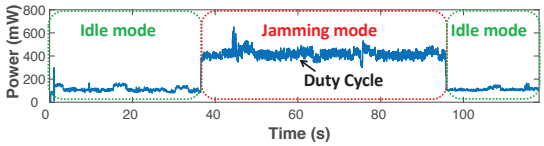


Figure 15: An example instantaneous power

7.2 Latency Analysis

The latency for MicShield from receiving the first frame of the wake word to stop jamming is 25.02 *ms (≤* 60 *ms). This will not degrade the wake word misdetection rate (see Figure 3(a)).* We evaluate this by measuring the processing latency of the MicShield software stack running on RPi 3B+. The processing time includes 3 parts: *(i)* MFCC & VAD; *(ii)* RNN-based phoneme recognition; *(iii)* HMM-based wake word recognition. As shown in Figure 12, upon receiving audio stream from the analog front-end, the recording thread first performs the MFCC and Voice Activity Detection (VAD), incurring ∼ 6.20 ms latency. Second, the control thread performs the automatic jamming control (phoneme recognition and wake-up word recognition) to determine whether to pass the jamming control signals to the subsequent jamming thread. In this thread, the latency and control frequency are dominated by sequence length of RNN-based phoneme recognition significantly. Figure 14(b) shows the measured correlations between control thread latency and input frame length. We choose input length of 15 frames (each frame takes 25 ms-window length and adjacent frames have a 15 ms-overlaps), and the control thread has $17.93 + 0.89 = 18.82$ ms latency each time. The overall latency is 25.02 ms for MicShield to response to a 25 ms frame speech, which is sufficient to pass the wake-up word to VAs based on our previous discussion in Section 3(a).

7.3 Energy Consumption

Our current MicShield prototype, as our proof-of-concept, incurs an estimated energy consumption of 2 *Wh per day.*

Power Consumption of Major System Components: We use INA 219 [65] to measure the instantaneous power consumption of major system components, *i.e.*, transducers, microphones and RPi. Table 1 summarizes the power consumption averaged over 10 min for the single- and multi-microphone setup. For single-microphone VA, MicShield consumes 97.6 mW and 403.35 mW in the idle and jamming mode, while for multi-microphone VA, the power consumption is 116.13 mW and 450.68 mW, respectively. We notice more than 65.04% of power is attributed to the RPi, which centers on an ARM based SoC and running a full fledged Debian OS. However, we expect the a product version of MicShield can be implemented based on dedicated low power chip, without the power hungry OS kernel modules and background daemon.

(a) Single-microphone VA

	Transducer	Microphone	RPi	Total
Idle	/	12.99	84.61	97.60
Jamming	121.03	12.99	262.33	403.35

(b) Multi-Microphone VA

	Transducer	Microphone	RPi	Total
Idle	/	31.52	84.61	116.13
Jamming	156.83	31.52	262.33	450.68

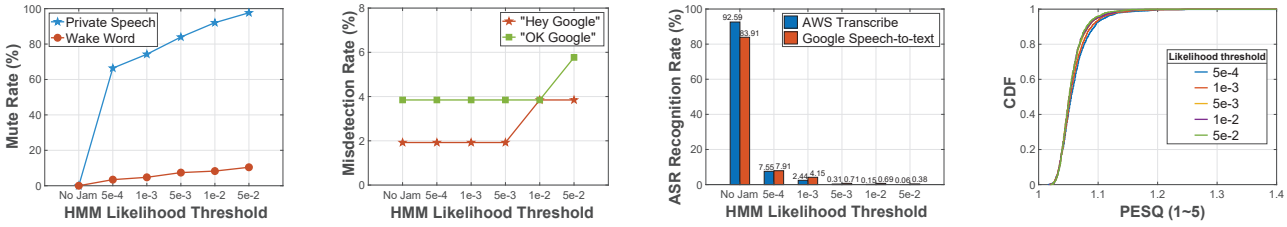
Table 1: Power consumption (mW) in controlled settings.

Approximation of Energy Consumption in a Practical Settings: In daily usage scenarios, it is known that a user speaks for about 2 hours on average per day [66]. Accordingly, we assume MicShield works in 14 hours of idle mode and 2 hours of jamming mode in a single day. Thus, MicShield would incur an energy consumption of ∼ 2 Wh per day. This means that, with a typical smartphone battery, MicShield can work for about 8 ∼ 10 days [67]. The energy would be largely reduced if a low-power DSP or ASIC is used as the substitute of RPi [68]. For example, a recent developed AI chip by Syntiant shows high potentials to be applied in our case with the power consumption down to the order of hundreds μW [68]. Alternatively, we also noticed a majority of multi-microphone VAs draw power from the mains, which is also applicable to MicShield.

7.4 Generalization

Our previous design and experiments use the Amazon Echo Dot by default. However, MicShield can be well generalized to alternative VAs. In this section, we show the generalizations of MicShield for Google Home [2] and Amazon Echo [1], 2 VAs that currently dominate the market [4].

• **Generalizations across Wake Words**: The same VA can use different pre-defined wake words. For example, Amazon Echo supports to change the wake of as one of the 4 words: "Alexa", "Echo", "Amazon", and "Computer" [1]. To accommodate this, MicShield stores the trained RNN and HMM models in the persistent memory for all the wake words that a user needs. It only needs to change the RNN and HMM models to adapt to the wake word modifications. Our current design only supports pre-defined wake words. However, this will not limit the generalizations of our design, as all current VA devices only supports limited number of wake words. Besides, new wake words can be trained following the same pipeline. The main difference for different wake words is the phoneme sequence pattern. Table 2 lists the percentages of the words in CMU Pronouncing Dictionary [47] that have different number of same consecutive phoneme as wake words used by Amazon Echo and Google Home. First, relatively large number of words (0.12% of 134000 words) shares the same first 4 phonemes with the wake word "Computer" since "Comp" (/kəmp/) is a word prefix. This means MicShield will have relatively high risk of leaking private words with the same word prefix as the wake word. Second, Google Home chooses to use the short utterances as the wake word, *e.g.*, "Hey Google" and "OK Google". This leads to that MicShield will leak the onset word of the short utterances, *i.e.*, "Hey" and "OK". MicShield can still protect the speech privacy in most of the practical settings, and we encourage users to use the words without the word prefix and onset word, *e.g.*, "Alexa" and "Amazon", to better protect the speech privacy.



(a) Mute rate (b) Wake word misdetection (c) Speech recognition rate (d) PESQ CDF

Figure 16: Micro benchmark using the wake words of "OK Google" & "Hey Google"

# of Consecutive Phoneme	1	2	3	4	5
"Alexa"	2.05	0.19	0.02	0.01	0.01
"Amazon"	1.46	0.15	0.03	0	0
"Echo"	1.51	0.17	0.01	/	/
"Computer"	9.73	1.13	0.25	0.12	0.01
"Hey Google"	5.00	0.19	0.01	0	0
"OK Google"	0.48	0.03	0	0	0

Table 2: Percentage of words in CMU Pronouncing Dictionary [47] that have similar number of consecutive phoneme sequence as different wake words.

MicShield can be well generalized to multiple wake words at the same time. Some VAs support multiple wake words triggering, *e.g.*, users can trigger Google Home by "OK Google" and "Hey Google". To evaluate the performance of MicShield in this case, we use the Google Home as an example. Our dataset is composed of 2 wake words speech from existing cloud based Text-To-Speech (TTS) services and real-world participants. For TTS based approach, we use the wake word speech from 24 "virtual" participants, generated by Google TTS [69], Amazon Polly [70] and IBM Watson TTS [71]. Our dataset also includes speech examples from 5 real-world participants. Overall, we collect 79 samples from 29 people for "OK Google" and "Hey Google". To train HMM and RNN model in automatic jamming control pipeline (see Figure 4), we shuffled and used 70% and 30% samples for training and testing purpose. The mute rate, misdetection rate and jamming effectiveness is plotted in Figure 16. A comparison with Figure 10 shows a negligible performance degradation for the 2 wake words setting.

• **Generalizations across VA Devices:** The geometry and sensitivity of microphone array varies across different VAs. To show the generalizations across devices, our evaluation is based on Amazon Echo Dot (4 microphones) and Google Home Mini (2 microphones). Note that these 2 multi-microphone VAs do not allow access to the private speech, but will record the history of the voice command after each wake word. Thus, we disable the jamming control policy for voice command and force the MicShield to only pass the wake word and obfuscate the voice command, so as to obtain the jammed results of commercial VAs. We use a smartphone to transmit the sound signals with 70 dBA SPL to first wake up the VA, and then continue with the voice commands. The voice commands are selected from the top 20 most popular ones [72]. Both Amazon and Google provide the metadata (content of the recognized voice command) for each request. The experimental results show that, all the metadata of these 100 jammed voice commands is "Audio could not be understood" or "unknown voice command". Amazon also provides the received audio signals of the voice commands. Whereby this, we showed the real participants and ASR algorithms cannot recognize the jammed audio signal neither. This means that MicShield can effectively protect the private speech for these different devices.

8 LIMITATIONS AND FUTURE WORK

Sensitivity of Microphones: Our experiments indicate the achievable distance between MicShield and sound source highly depends on the sensitivity of microphone on the MicShield. Our current prototype supports up to 3 m range for wake word detection. For many single-microphone VAs, *e.g.*, smartphones, the user-device distances are generally limited within 1 m. MicShield can well address such usage scenarios. However, commercial microphone array based VAs may be able to achieve longer than 3 m detection range. To ensure comparable detection range, MicShield needs to adopt similar hardware setup as such commercial VAs, *e.g.*, using a microphone array along with self-interference cancellation, acoustic echo cancellation and beamforming algorithms. We expect a commercial buildout of MicShield can be easily equipped with such capabilities.

Attacks on MicShield: MicShield's threat model assumes that the VAs are untrustable, but MicShield itself still needs to run the wake word detection mechanism, albeit always offline. An attacker may attempt to defeat MicShield by forcing the VA to secretly play the wake word sounds with low volume or inaudible voice [16], so as to cheat MicShield and stop its jamming. The attackers may also employ other speaker devices, *e.g.*, TV, to play the wake words. However, such forged voices can be easily identified, *e.g.*, by installing multiple microphones on MicShield to locate the sound source, or through liveness detection methods [73, 74]. These attacks and countermeasures targeting the wake word detection have been well explored, and are beyound the scope of our work.

9 CONCLUSION

The always-on microphones on voice assistants (VAs) have raised serius privacy concerns. In this paper, we propose MicShield, the first system to automatically protect speech privacy against always-on microphones. MicShield introduces a novel selectively jamming mechanism, which can obfuscate private speech while passing legitimate voice commands using phoneme-level features. We prototype implementation and experiments verify the feasibility and effectivness of MicShield in protecting speech privacy without degrading the VAs' basic functionalities. MicShield marks a critical step in addressing the potential privacy risks of VAs.

ACKNOWLEDGMENTS

We would like to thank the anonymous shepherd and reviewers for their valuable comments. This work is partially supported by the NSF under grant CNS-1901048 and CNS-1952942.

REFERENCES

[1] Amazon.com. Amazon echo. https://www.amazon.com/echo/.
[2] Google. Google home. https://store.google.com/product/google_home.
[3] Greg Sterling. Alexa devices maintain 70% market share in u.s. according to survey. https://marketingland.com/alexa-devices-maintain-70-market-share-in-u-s-according-to-survey-265180.
[4] Robert Williams. Study: Smart speaker ownership surges 36% to 53m US adults. https://www.mobilemarketer.com/news/study-smart-speaker-ownership-surges-36-to-53m-us-adults/545717/.
[5] Amazon. Alexa, echo devices, and your privacy, amazon help & customer service. https://www.amazon.com/gp/help/customer/display.html?nodeId=GVP69FUJ48X9DK8V.
[6] Google. More about data security and privacy on devices that work with assistant. https://support.google.com/googlenest/answer/7072285?hl=en.
[7] Google. Google home mini. https://store.google.com/product/google_home_mini.
[8] Russakovskii Artem. Google is permanently nerfing all home minis because mine spied on everything i said 24/7. https://www.androidpolice.com/2017/10/10/google-nerfing-home-minis-mine-spied-everything-said-247/#1.
[9] Soo Youn. Alexa is always listening — and so are amazon workers. https://abcnews.go.com/Technology/alexa-listening-amazon-workers/story?id=62331191.
[10] Zack Wittaker. Amazon says US government demands for customer data went up. https://techcrunch.com/2019/08/01/amazon-prism-transparency-data/.
[11] Heather Kelly. How to make sure your amazon echo doesn't send secret recordings, 5 2018. https://money.cnn.com/2018/05/25/technology/amazon-alexa-stop-recording/index.html.
[12] Nirupam Roy, Sheng Shen, Haitham Hassanieh, and Romit Roy Choudhury. Inaudible voice commands: The long-range attack and defense. In *Proceedings of Usenix NSDI*, 2018.
[13] Yuxin Chen, Huiying Li, Steven Nagels, Zhijing Li, Pedro Lopes, Ben Y Zhao, and Haitao Zheng. Wearable microphone jamming. In *Proceedings of ACM CHI*, 2020.
[14] Nirupam Roy, Haitham Hassanieh, and Romit Roy Choudhury. Backdoor: Making microphones hear inaudible sounds. In *Proceedings of ACM MobiSys*, 2017.
[15] François Grondin and François Michaud. Lightweight and optimized sound source localization and tracking methods for open and closed microphone array configurations. *Robotics and Autonomous Systems*, 2019.
[16] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. Dolphinattack: Inaudible voice commands. In *Proceedings of ACM CCS*, 2017.
[17] Takeshi Sugawara, Benjamin Cyr, Sara Rampazzi, Daniel Genkin, and Kevin Fu. Light commands: Laser-based audio injection attacks on voice-controllable systems. 2019.
[18] Yitao He, Junyu Bian, Xinyu Tong, Zihui Qian, Wei Zhu, Xiaohua Tian, and Xinbing Wang. Canceling Inaudible Voice Commands Against Voice Control Systems. In *Proceedings of ACM MobiCom*, 2019.
[19] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. Hidden voice commands. In *Proceedings of USENIX Security Symposium*, 2016.
[20] Tavish Vaidya, Yuankai Zhang, Micah Sherr, and Clay Shields. Cocaine noodles: exploiting the gap between human and machine speech recognition. In *9th USENIX Workshop on Offensive Technologies*, 2015.
[21] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, Shengzhi Zhang, Heqing Huang, XiaoFeng Wang, and Carl A Gunter. Commandersong: A systematic approach for practical adversarial voice recognition. In *Proceedings of USENIX Security Symposium*, 2018.
[22] Nicholas Carlini and David Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *Proceedings of IEEE Security and Privacy Workshops (SPW)*, 2018.
[23] CMUSphinx, 2019. https://cmusphinx.github.io/.
[24] Deepak Kumar, Riccardo Paccagnella, Paul Murley, Eric Hennenfent, Joshua Mason, Adam Bates, and Michael Bailey. Skill squatting attacks on amazon alexa. In *Proceedings of USENIX Security Symposium*, 2018.
[25] Nan Zhang, Xianghang Mi, Xuan Feng, XiaoFeng Wang, Yuan Tian, and Feng Qian. Dangerous skills: Understanding and mitigating security risks of voice-controlled third-party functions on virtual personal assistant systems. In *Proceedings of IEEE Security and Privacy*, 2019.
[26] Yu-Chih Tung and Kang G Shin. Exploiting sound masking for audio privacy in smartphones. In *Proceedings of ACM AsiaCCS*, 2019.
[27] Gaurav Srivastava, Kunal Bhuwalka, Swarup Kumar Sahoo, Saksham Chitkara, Kevin Ku, Matt Fredrikson, Jason Hong, and Yuvraj Agarwal. Privacyproxy: Leveraging crowdsourcing and in situ traffic analysis to detect and mitigate information leakage, 2017.
[28] Yuvraj Agarwal and Malcolm Hall. Protectmyprivacy: Detecting and mitigating privacy leaks on ios devices using crowdsourcing. In *Proceedings of ACM MobiSys*, 2013.
[29] Ashwin Rao, Justine Sherry, Arnaud Legout, Arvind Krishnamurthy, Walid Dabbous, and David Choffnes. Meddle: Middleboxes for increased transparency and control of mobile traffic. 2012.
[30] Jianwei Qian, Haohua Du, Jiahui Hou, Linlin Chen, Taeho Jung, and Xiang-Yang Li. Hidebehind: Enjoy voice input with voiceprint unclonability and anonymity. In *Proceedings of ACM SenSys*, 2018.
[31] J. Clark and P. C. van Oorschot. Sok: Ssl and https: Revisiting past challenges and evaluating certificate trust model enhancements. In *Proceedings of IEEE Symposium on Security and Privacy*, 2013.
[32] Alexa privacy and data handling overview. https://d1.awsstatic.com/product-marketing/A4B/White%20Paper%20-%20Alexa%20Privacy%20and%20Data%20Handling%20Overview.pdf.
[33] Igor Bobriakov. Comparison of top 10 speech processing APIs. https://medium.com/activewizards-machine-learning-company/comparison-of-top-10-speech-processing-apis-2293de1d337f.
[34] International standard iec 61672:2003. International Electrotechnical Commission, 2003.
[35] Noise and hearing loss prevention. https://www.asha.org/public/hearing/Noise-and-Hearing-Loss-Prevention/.
[36] A. D. Wyner. The wire-tap channel. *The Bell System Technical Journal*, 54(8):1355–1387, Oct 1975.
[37] ITU-T Recommendation. Perceptual evaluation of speech quality (pesq): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs. *Rec. ITU-T P. 862*, 2001.
[38] Antony W Rix, John G Beerends, Michael P Hollier, and Andries P Hekstra. Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs. In *Proceedings of IEEE ICASSP*, 2001.
[39] Amazon Transcribe, 2019. https://aws.amazon.com/transcribe/.
[40] Google Cloud Speech-to-Text, 2019. https://cloud.google.com/speech-to-text/.
[41] Amazon.com: Echo dot (3rd gen) - smart speaker with alexa - charcoal: Amazon devices. https://www.amazon.com/Echo-Dot/dp/B07FZ8S74R.
[42] Bjørn Karmann. Project Alias, 2019. https://www.instructables.com/id/Project-Alias/.
[43] Amir Anhari. Alexa dataset - build voice-first applications. https://www.kaggle.com/aanhari/alexa-dataset.
[44] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm networks. In *Proceedings of IEEE International Joint Conference on Neural Networks*, 2005.
[45] John S Garofolo et al. Darpa timit acoustic-phonetic speech database. *National Institute of Standards and Technology (NIST)*, 15:29–50, 1988.
[46] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of IEEE*, 77(2):257–286, 1989.
[47] The CMU Pronouncing Dictionary, 2019. http://www.speech.cs.cmu.edu/cgi-bin/cmudict.
[48] Tomi Kinnunen, Evgenia Chernenko, Marko Tuononen, Pasi Fränti, and Haizhou Li. Voice activity detection using mfcc features and support vector machine. In *Int. Conf. on Speech and Computer (SPECOM07), Moscow, Russia*, volume 2, pages 556–561, 2007.
[49] Logan Blue, Luis Vargas, and Patrick Traynor. Hello, is it me you're looking for? differentiating between human and electronic speakers for voice interface security. In *Proceedings of ACM WiSec*, 2018.
[50] Muhammad Ejaz Ahmed, Il-Youp Kwak, Jun Ho Huh, Iljoo Kim, Taekkyung Oh, and Hyoungshick Kim. Void: A fast and light voice liveness detection system. In *Proceedings of USENIX Security Symposium*, 2018.
[51] Amazon. Google home mini. https://www.amazon.com/gp/help/customer/display.html?nodeId=202201630.
[52] John D'Errico. Surface fitting using gridfit. *MathWorks file exchange*, 643, 2005. https://www.mathworks.com/matlabcentral/fileexchange/8998-surface-fitting-using-gridfit.
[53] The respeaker 6 mic array for raspberry pi, 2019. https://respeaker.io.
[54] Don H Johnson and Dan E Dudgeon. *Array signal processing: concepts and techniques*. PTR Prentice Hall Englewood Cliffs, 1993.
[55] Sanjib Sur, Teng Wei, and Xinyu Zhang. Autodirective Audio Capturing through a Synchronized Smartphone Array. In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2014.
[56] J Widder and A Morcelli. Basic principles of mems microphones, 2016. https://www.edn.com/basic-principles-of-mems-microphones/.
[57] Kazunori Miura. Ultrasonic directive speaker. *Elektor Magazine*, 3:2011, 2011.
[58] Filterless 3w class-d stereo audio amplifier (datasheet). https://www.diodes.com/assets/Datasheets/PAM8403.pdf.
[59] John D. Cutnell, Kenneth W. Johnson, David Young, Shane Stadler. *Physics*. Wiley, 11 edition.
[60] H Tijdeman. On the propagation of sound waves in cylindrical tubes. *Journal of Sound and Vibration*, 1975.
[61] Environmental health criteria – ultrasound, 1982. https://apps.who.int/iris/bitstream/handle/10665/37263/9241540826-eng.pdf?sequence=1&isAllowed=y.

[62] Pimoroni pHAT DAC24-bit/192khz sound card. https://shop.pimoroni.com/products/phat-dac.
[63] Theano, 2019. https://github.com/Theano/Theano.
[64] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *Proceedings of IEEE ICASSP*, 2017.
[65] Ina 219 zero-drift, bidirectional current/power monitor with i2c interface. http://www.ti.com/lit/ds/symlink/ina219.pdf.
[66] Matthias R Mehl, Simine Vazire, Nairán Ramírez-Esparza, Richard B Slatcher, and James W Pennebaker. Are women really more talkative than men? *Science*, 2007.
[67] Johnson Dave. How to save battery on your samsung galaxy s10 in 4 simple ways. https://www.businessinsider.com/how-to-save-battery-on-samsung-galaxy-s10.
[68] James Morra. Ai chip brings always-on alexa to battery-powered devices. https://www.electronicdesign.com/technologies/embedded-revolution/article/21808470/ai-chip-brings-alwayson-alexa-to-batterypowered-devices.
[69] Google Cloud Text-to-Speech, 2019. https://cloud.google.com/text-to-speech/.
[70] Amazon Polly, 2019. https://aws.amazon.com/polly/.
[71] IBM Text-to-Speech, 2019. https://www.ibm.com/cloud/watson-text-to-speech.
[72] 20 helpful amazon echo voice commands for you to try. https://www.popsci.com/20-amazon-echo-voice-commands/.
[73] Yeonjoon Lee, Yue Zhao, Jiutian Zeng, Kwangwuk Lee, Nan Zhang, Faysal Hossain Shezan, Yuan Tian, Kai Chen, and XiaoFeng Wang. Using sonar for liveness detection to protect smart speakers against remote attackers. In *Proceedings of ACM IMWUT (UbiComp)*, 2020.
[74] Linghan Zhang, Sheng Tan, and Jie Yang. Hearing your voice is not enough: An articulatory gesture based liveness detection for voice authentication. In *Proceedings of ACM CCS*, 2017.