PhantomSound: Black-Box, Query-Efficient Audio Adversarial Attack via Split-Second Phoneme Injection

Hanqing Guo
guohanqi@msu.edu
Michigan State University
East Lansing, Michigan, USA

Guangjing Wang
wanggu22@msu.edu
Michigan State University
East Lansing, Michigan, USA

Yuanda Wang
wangy208@msu.edu
Michigan State University
East Lansing, Michigan, USA

Bocheng Chen
chenboc1@msu.edu
Michigan State University
East Lansing, Michigan, USA

Qiben Yan
qyan@msu.edu
Michigan State University
East Lansing, Michigan, USA

Li Xiao
lxiao@cse.msu.edu
Michigan State University
East Lansing, Michigan, USA

ABSTRACT

In this paper, we propose PhantomSound, a query-efficient black-box attack toward voice assistants. Existing black-box adversarial attacks on voice assistants either apply substitution models or leverage the intermediate model output to estimate the gradients for crafting adversarial audio samples. However, these attack approaches require a significant amount of queries with a lengthy training stage. PhantomSound leverages the decision-based attack to produce effective adversarial audios, and reduces the number of queries by optimizing the gradient estimation. In the experiments, we perform our attack against 4 different speech-to-text APIs under 3 real-world scenarios to demonstrate the real-time attack impact. The results show that PhantomSound is practical and robust in attacking 5 popular commercial voice controllable devices over the air, and is able to bypass 3 liveness detection mechanisms with $> 95\%$ success rate. The benchmark result shows that PhantomSound can generate adversarial examples and launch the attack in a few minutes. We significantly enhance the query efficiency and reduce the cost of a successful untargeted and targeted adversarial attack by 93.1% and 65.5% compared with the state-of-the-art black-box attacks, using merely ~300 queries (~5 minutes) and ~1,500 queries (~25 minutes), respectively.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies → Machine learning**;

KEYWORDS

Adversarial attack; voice assistant; black-box attack; query efficiency.

ACM Reference Format:
Hanqing Guo, Guangjing Wang, Yuanda Wang, Bocheng Chen, Qiben Yan, and Li Xiao. 2023. PhantomSound: Black-Box, Query-Efficient Audio Adversarial Attack via Split-Second Phoneme Injection. In *The 26th International Symposium on Research in Attacks, Intrusions and Defenses (RAID '23), October 16–18, 2023, Hong Kong, China.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3607199.3607240

1 INTRODUCTION

Voice is the primary method for human-computer interaction. Driven by the unprecedented amount of voice data and flourishing development of Artificial Intelligence (AI) technology, the modern deep-learning based Automatic Speech Recognition (ASR) systems and Intelligent Voice Control (IVC) devices have been integrated into our daily lives. According to Voicebot's 2019 consumer report [39], it was reported that 26% of U.S. adults own a smart speaker. Nowadays, users can directly speak to their smartphones to interact with the voice assistants such as Siri [56], Google Assistant [22], or smart speaker systems such as Google Home [23], Amazon Echo [8].

The voice commands have been used to send and read text messages, make phone calls, set timers, check calendar entries, and even order a drink from Starbucks or summon a Uber with "skills" [10]. More and more tech companies now provide ASR services, including Amazon Transcribe [9], Google Cloud Speech-to-Text [24], IBM Watson Speech to Text [36], and Microsoft Azure Speech Service [46], all of which allow the developers to empower their apps with intelligent audio functionalities.

However, with the increasing presence of ASR systems and IVC devices in private spaces, users begin to worry about the security and privacy of these systems. For example, a hacked device is now capable of recording private conversations; collecting and sharing private data; and controlling all the connected IoT devices in smart homes [18, 53]. Researchers have demonstrated that ASR systems could become vulnerable to a wide variety of attacks. For instance, inaudible commands can be injected through ultrasound [49, 67], even across different transmission media, such as object surface [62], light [53], etc. Besides the physical attacks, recent studies also utilize the discrepancies between the human ear and feature extraction algorithms to launch *signal processing attacks* [3, 4].

Despite the aggravating threats, these new attacks could be defeated by integrating additional hardware [66] or extra signal

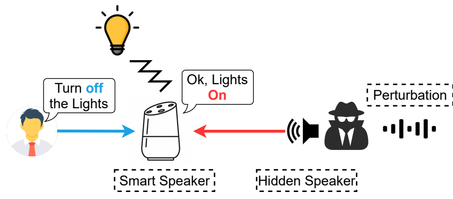


Figure 1: Attack scenario of PhantomSound

processing procedures (e.g., voice activity detection, guard signals) [3, 34]. Unlike the aforementioned attacks, the *adversarial attack* aims to attack the deep neural networks (DNN), i.e., the computational core of an ASR system, which poses a major threat to modern ASR systems.

Adversarial Attack: Adversarial attack was first proposed to attack image recognition systems [21, 54]. The attack operates by imposing unnoticeable perturbations onto the original image, thereby misleading the DNN to yield false classification. The inputs that enable such an attack are commonly referred to as *Adversarial Examples (AEs)*, which are composed of the original input with an unnoticeable perturbation. The ASR system with DNN models also inherits the susceptibility towards AEs.

Prior Studies: Prior studies [7, 14, 20] demonstrate that attackers can generate adversarial audios to alter the DNN's prediction result with or without the prior knowledge of the DNN model. However, most of these attacks have not been successfully realized against real-world commercial devices, and their stealthiness is unverified. Recently, Chen et al. [15] successfully attack both open-source and commercial speaker verification systems over the air in a grey-box setting. Yuan et al. [64] embed their generated AE within songs to launch the attack, and they further adapt their attack in a black-box setting to subvert the ASR of most IVC devices [18]. Nevertheless, they fail to guarantee the attack success rate in the presence of user interference; and cannot promise to craft AEs quickly due to the training overhead of the substitution model. Meanwhile, two recent studies [27, 43] inventively propose the sub-second perturbation and spectrogram patch perturbation to attack open-source ASR systems, considering the victim user present during the attack. Even though they demonstrate the robustness and feasibility of their attack in the presence of environmental distortions, the proposed attacks are established on the assumption of complete knowledge of the target ASR system. More recently, Zheng *et al.* propose a decision-based black-box attack by incorporating evolutionary algorithms to generate adversarial audios [69]. However, they still require to query the victim model extensively, which incurs substantial time and financial costs in a practical attack scenario.

Table 1 summarizes the existing adversarial attacks in terms of *victim systems' tasks*, *attacker knowledge*, *ability to attack quickly*, and *attack scenario*. The check mark symbolizes a successful attack under the given scenario, while the cross mark implies that the attack could not function or lacks efficacy in that particular scenario. For the *victim system's task*, SV indicates the speaker verification task while SR refers to the speech recognition task. We then taxonomize *attacker knowledge* into white-box, grey-box, and black-box,

Table 1: Comparison with other recent audio attacks.

Attacks	SV SR	Grey Box	Black Box	Online GENR	Over Air	User INT
Houdini [20]	SR	✓	✗	✗	✗	✗
C&W [14]	SR	✗	✗	✗	✗	✗
Adversarial [7]	SR	✓	✗	✗	✗	✗
Fakebob [15]	SV	✓	✗	✗	✓	✗
Comm. [64]	SR	✗	✗	✗	✓	✗
Devil's [18]	SR	✓	✓	✗	✓	✗
AdvPulse [43]	SR	✗	✗	✗	✓	✓
OCCAM [69]	SR	✓	✓	✗	✓	✗
SpecPatch [27]	SR	✗	✗	✗	✓	✓
PhantomSound	**SR**	✓	✓	✓	✓	✓

where grey-box implies the attacker can get the logits layer output [7, 15] or confidence score of all possible classes, and black-box indicates the attacker can only access the prediction label [18] of the target model. A white-box attacker, on the other hand, has complete knowledge (model architecture, weights of DNN parameters) of the target system. Next, we use online AE generation (Online GENR) to characterize whether the attacker can generate AEs or perturbations swiftly and complete the attack procedure in an online fashion. In fact, most existing studies assume the attacker has sufficient time to produce AEs offline. The last two metrics, Over Air and User Interference (User INT) suggest the attack scenario, where the former indicates an over-the-air attack, while the latter indicates whether the attack considers the user's interference (e.g., voice commands) during the attacks. To the best of our knowledge, *no existing attacks can attack commercial, closed-source ASR systems over-the-air with a limited time budget and user interference.*

PhantomSound: We propose a query-efficient black-box attack on commercial closed-source ASR systems and IVC devices. Our attack, called *PhantomSound*, can craft AEs and perturbations within a limited time budget and restricted query cost. Different from the previous work, the key idea behind PhantomSound is to regard the users' voice input as the command "carrier", while the phoneme-level perturbations are applied on the "carrier" to instantiate the attack.

Figure 1 depicts the attack scenario. First, the adversary records the user's command (any keywords such as "open", "on", "down"). Next, the adversary uses PhantomSound to query the accessible target models on the target IVC devices (e.g., the Google Cloud Speech-to-Text API for Google Home). Then, PhantomSound returns a perturbation that alters the prediction of the user's command.

During the attack, the adversary plays the perturbation via a hidden speaker at the same time when the user utters a voice command, which fools the smart speaker to operate improperly.

Challenges: Four major challenges arise during the design of PhantomSound.

- **Black-box Attack:** It is difficult to attack a model without any prior knowledge. Existing grey-box/black-box attacks either assume attackers have the probability score of the target model [7, 20], or train a substitution model to approach the target model [18]. The existing attacks require a substantial amount of time to train a substitution model for the generation of AEs.

- **Speech Model:** Different from black-box attacks on image processing [16, 19], ASR systems are known to have a more complicated model structure consisting of signal processing, filtering, acoustic model, and language model. As a result, attacking speech models requires different attack strategies to bypass the various components of the ASR models.
- **Query Efficiency:** A successful black-box attack relies excessively on the effectiveness of queries. The adversary needs to iteratively update the AEs such that the effectiveness of the crafted AEs can be justified through querying. However, querying commercial ASR APIs is costly (e.g., \$0.00001/second for Google Cloud Speech-to-Text) and unable to bypass. Despite some efforts [16, 19] to reduce the number of queries, it still falls short of meeting the requirements for online generation of AEs.
- **Perturbation Sync:** To successfully launch our attack, the adversary is expected to play the perturbation when he/she hears the victim's voice command. However, in a real-world scenario, the timing of perturbation is hard to control. Therefore, we need to tackle this problem by generating a near-synchronization-free perturbation [43].

Contributions: The contributions of this work are highlighted as follows.

- **New Attack:** To the best of our knowledge, we are the first to achieve query-efficient black-box attacks on commercial ASR systems as well as IVC devices. We demonstrate the dangers of our attack over-the-air on 4 different commercial ASR APIs (i.e., Google Cloud Speech-to-Text, IBM Watson Speech to Text, Amazon Transcribe, and Microsoft Azure Speech Service) and 5 different IVC devices (i.e., iPhone with Google Assistant, Google Home, Microsoft device, Amazon Alexa, and IBM Wav-Air-API).
- **New Finding:** We discover and formulate the unique boundary of commercial ASR systems for producing AEs. This non-contiguous decision boundary hinders previously successful attempts.
- **New Techniques:** We propose PhantomSound, a phoneme-level searching method for efficiently crafting AEs to launch adversarial perturbation attack with the least number of required queries in comparison with other methods.

The remainder of the paper is organized as follows. We introduce the background and preliminary observations in §2, followed by the system design of PhantomSound in §3. The implementation and evaluation are further presented in §4. The discussion and limitation of PhantomSound are entailed in §5. We present the related work in §6 and conclude the paper in §7.

2 BACKGROUND AND PRELIMINARY STUDY

In this section, we present the threat model of PhantomSound, as well as the assumptions and attack scenarios. Then, we introduce the fundamentals behind the adversarial attack and present the decision scheme of commercial ASR systems.

2.1 Threat Model

The adversary's goal is to mislead the IVC devices or VCS systems by injecting malicious commands. Prior to our work, there are two types of attacks that can achieve the same goal. The first attack [18] uses reverse-engineering models to imitate the commercial models and craft the offline AE in a white-box manner. The second attack [61] uses generative models to synthesize the victim's speech. However, the reverse-engineering attack necessitates a high volume of queries (as per Table 10) to construct the substitute model. It also demands updating the model in response to changes in the commercial API. This renders it expensive and inadequate in meeting the need for a real-time attack. Regarding the generative model driven synthesized attack, we assume the adversary has access to sufficient recordings of the victim for training purposes. However, in our specific situation, the attacker is expected to initiate the attack upon their first encounter with the victim. Furthermore, playing the synthesized speech outright is not a viable approach as the victim can hear it and potentially halt the attack.

Adversary's Capability: We assume that the adversaries can place a hidden microphone to record the victim's voice. We assume that an adversary knows the targeted IVC devices and has access to their respective ASR API services (e.g., Google Cloud Speech-to-Text for Google Home or Google Assistant). Following other related studies [7, 18, 43, 64, 69], we also assume that the adversary is able to launch this attack via a hidden speaker or a compromised speaker in the victim's workspace/home.

Attack Scenarios: The adversary will first collect the victim's voice commands, and then generate the AEs and perturbations swiftly only based on the transcription result of the target devices. Once the perturbations are crafted, the adversary can wait for the victim's next command and play the perturbation manually or automatically via existing keyword searching or voice detection mechanisms [6, 55]. Alternatively, the adversary may also play the perturbation repeatedly through hacked speakers, attempting to fool the target IVC devices when the corresponding target voice command was delivered.

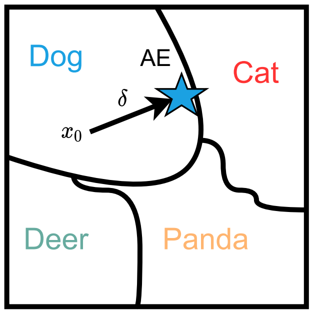
In a real-world attack scenario, e.g., in a public space, an attacker may not have access to a large collection of victims' voices and may not have sufficient time to generate the perturbation offline. In this case, the attacker only has a very limited time window to subvert the victims' commands towards voice assistants. To successfully instantiate such an opportunistic voice attack, an attack approach with a timely and low complexity AE generation is highly desired.

User Interference: Most existing attacks assume that the users will not perceive the AEs and will not interact with their voice assistants during the attack. However, when the users are speaking during the attack, most existing voice attacks will fail. In this research, we leverage the users' voice command as a carrier for the adversarial audio to launch the attacks more effectively and stealthily. Moreover, as advanced liveness detection algorithms [5, 42] have been used to differentiate between loudspeakers and humans with high accuracy, most existing audio attacks launched by loudspeakers can be easily detected. In our attack, however, since the human voice and the perturbation arrive at the same time, the liveness detection module of the voice assistant can be effectively bypassed.

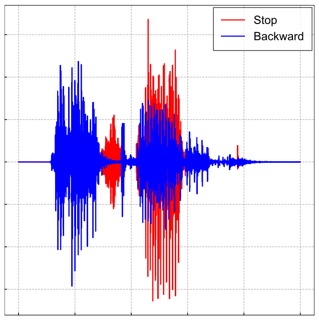
2.2 Adversarial Attack

Adversarial attack aims to craft an AE $x_0 + \delta$, in order to deceive the model $f(\cdot)$ to make false prediction [54]. Take y_{pred} as the output of model, if $f(x_0+\delta) := y_{pred} \neq y$ (y indicates the true label

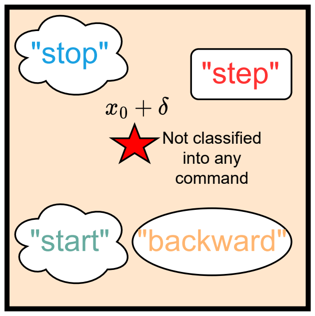
(a) A mixed image with cat and dog is recognized by Google Cloud Vision API [25] with 89% cat and 11% dog.



(b) Search decision boundary in black-box CV attack.



(c) A mixed audio with "stop" and "backward" is rejected by Google Speech-to-Text API with no output



(d) The decision boundary for every class is non-contiguous for ASR system, every input in the middle will be rejected due to ambiguity.

Figure 2: Observations of CV and ASR systems.

of input x_0), we suppose the attacker has launched an untargeted attack. If the perturbation is crafted intentionally for a specific target (denoted as y_t), the attack formalized as $f(x_0 + \delta) = y_t \neq y$, is regarded as a targeted attack. The generation of AE can be formulated as an optimization problem as follows:

$$minimize \quad \mathcal{L}(x_0 + \delta) := \mathcal{D}(f(x_0 + \delta), y_t). \tag{1}$$

The goal of Eq. (1) is to minimize $\mathcal{L}(x_0 + \delta)$ under the constraint that $||\delta||_2 < \epsilon$, where $\mathcal{L}(\cdot)$ denotes the loss function, which uses a distance function $\mathcal{D}(\cdot)$ to measure the disparity between $f(x_0 + \delta)$ and y_t, $||\cdot||_2$ is the L2 norm, and ϵ is used to control the amplitude of perturbation. There are three main types of attacks depending on the prior knowledge of the victim models, listed as follows:

White-box: If the adversaries learn architecture and the parameters of the model, they can get the gradient of the loss function $\nabla\mathcal{L}(x)$ during the forward or backpropagation. The perturbation can be subsequently estimated using the inverse gradient [21].

Grey-box: The model conceals its architecture and parameters from the public and only exposes the prediction scores $P = [p_0, p_1, \cdots, p_n]$ for a given input. The adversaries can formulate a loss function [13] $\mathcal{D}(P, P_y)$ (P_y is the one-hot encoding of y), and then track the changes of distance when tuning δ in multiple attempts. The changes in $\mathcal{L}(x)$ are utilized to estimate the gradient which will guide the attacker to update δ. The gradient estimation algorithms include Natural Evolution Strategy (NES) [37] and Zeroth Order Optimization (ZOO) [17].

Black-box: Compared to white-box and grey-box attacks, the black-box attack is the most challenging, in which the attacker only has access to the prediction label of the model. In fact, most of the commercial ASR systems and IVC devices are closed-source and only offer a final prediction. To successfully attack the black-box model, existing work either trains a surrogate model and transforms the problem into a white-box attack [47], or uses a significant amount of queries to search the decision boundary of the victim model [12, 16, 19]. Here, we focus on the query-based boundary-searching attack due to its flexibility and attack efficiency.

2.3 Black-box Audio Adversarial Attack

Compared with the black-box adversarial attack in other domains, the black-box audio adversarial attack has several unique features. In this section, we conduct a preliminary study in quantifying the behaviors of commercial ASR services.

Decision-based Attack: Used for classification, a decision boundary is a hypersurface that partitions the sample space into several classes. Specifically, a well-trained DNN model uses the decision boundary to classify the incoming inputs. The main goal of the existing black-box attacks [12, 16, 19], or so-called decision-based attacks, is to find the decision boundary of the target model. Generally, to approach the precise decision boundary, they gradually perturb the input based on the query feedback, to find an AE on the verge of the decision boundary.

However, one assumption made by existing decision-based attacks is that the DNN classification model guarantees to return a prediction y_{pred} for any input x. As shown in Fig. 2(a), we merge a cat and a dog into one image and feed it into Google Cloud Vision API [25]. The classifier labels the image as a cat with very high confidence (89%) while the human brain perceives it differently. As shown in Fig. 2(b), the decision-based adversary [16] starts from a dog (x_0) and adds the proportion of a cat (δ) gradually to approach the boundary. The curves between classes in Fig. 2(b) indicate the decision boundaries, where $\delta \in [0, 255]^{H\times W}$ denotes the perturbed image with the same shape as x_0. The contiguous decision boundary allows the DNN models to always output a result, while the result turns unreliable as it approaches the decision boundary.

Decision Boundary of ASR: At first sight, it appears that the ASR systems would inherit the DNN's susceptibility to decision-based adversarial attacks. However, the unique characteristics of voice systems and DNN models make traditional decision-based attacks hard to succeed. Here, we conduct a preliminary experiment, in which we mix two voice commands "stop" and "backward" together (Fig. 2(c)) to imitate the mixture of cat and dog images. Then, we submitted the mixed audio to Google Speech-to-Text API, which was rejected without any returns. The failed attempts indicate that the decision boundary of the ASR system is non-contiguous. As shown in Fig. 2(d), every voice command is surrounded by an

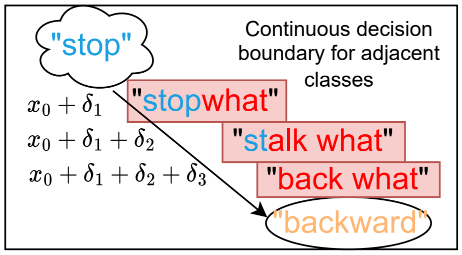


Figure 3: Phoneme guided query

exclusive boundary, and the audios outside of the boundary ranges will be rejected by the ASR systems.

This phenomenon implies that the perturbed voice queries may fail to solicit valid feedback from the ASR systems. Without feedback, it is difficult to determine the direction of the perturbation for approaching a target decision boundary. Based on this observation, we are motivated to design a new boundary-searching method to enable the decision-based black-box attack toward ASR systems.

3 ATTACK DESIGN

In this section, we present the system design of PhantomSound. We first introduce the phoneme-level boundary searching method to minimize the possibility of rejection by the ASR systems. Then, we formalize the attack as an optimization problem and illustrate the generation of AEs. Finally, to enhance the robustness of PhantomSound in real-world scenarios, we propose the weak synchronization scheme and over-the-air speech enhancement.

3.1 Phoneme-level Boundary Searching

Fig. 2(d) shows the challenge in boundary searching to produce a proper AE. If the adversary randomly adds noise to "stop", the ASR remains the "stop" decision when the noise is low and gives rejection while rising the noise power. However, if the adversary directly applies target "backward" to the benign audio, it results in audio (red start in Fig. 2(d)) in the middle between two decision boundaries, hence giving no output.

Therefore, the reasons behind the rejection of queries can be attributed to two factors: 1) the added random noise will elevate the command's noise level; 2) the boundary distance between two valid commands is too long to allow for an unnoticeable perturbation. To resolve these two problems, a novel idea is raised: "If we break the target "backward" into small pieces, then craft AE with sub-targets which directly connect to the benign decision boundary with small pieces, and finally, we can craft the final AE with the target." Fig. 3 depicts our attack design. Specifically, instead of directly adding "backward" on the "stop", we break the target "backward" into a series of phonemes. During crafting the AE, we randomly add the phoneme on the benign audio and check the prediction. If the ASR produces a word that is closer to our target, we keep the phoneme on the benign audio and search for a closer prediction in the next round. In our case, the "stop" adds perturbation phoneme δ_1 and is recognized as "stopwhat", then changes to "stalk what", and "back what", and finally reaches the target "backward". In every step, the AE achieves to sub-targets who is adjacent to the benign decision boundary, and gradually, the perturbation can be crafted by summing up all the small changes.

The basic idea of the proposed phoneme-level searching method is to perturb the original command along the direction of the target command while minimizing the distance between the original and the target ones.

Algorithm 1 presents the initialization procedure for generating the phoneme-level adversarial perturbation. Specifically, we first set the counter $s = 0$, and the initial distance between benign and target as $\epsilon = \mathrm{CER}(f(x_0), y_t)$. Next, we construct a phoneme set $D = \{ph_1, ph_2, ..., ph_n\}$ by breaking the target command, and then generate a random noise $v \in [0, 0.1]^l$ in line 4, where l is the length of original input x_0. Next, together with the v, a phoneme from D is randomly picked and injected at its corresponding position of x_0 in lines 5-6 to generate an AE x_*. The purpose of v is to increase the variance of the phoneme. For the targeted attack, if the x_* has a smaller distance to the target (line 7), we put the perturbation to the initial perturbation set $\tilde{P}$, then update the ϵ and x_0. For an untargeted attack, we can replace line 7 with "if $f(x^*)! = y$" to assure the ASR gives an incorrect prediction. The searching loop continues until it reaches a sufficient number of rounds K.

Algorithm 1: Phoneme-level Adversarial Perturbation Initialization

```
Input: The original audio $x_0$, the target label $y_t$, the
       phoneme clip samples $D = \{ph_1, ph_2, ..., ph_n\}$, the
       initial Character Error Rate(CER) $\epsilon$, the API service
       of black-box ASR system $f(\cdot)$.
Result: The initial perturbations set $\tilde{P}$
1  s = 0;
2  $\epsilon = \mathrm{CER}(f(x_0), y_t)$;
3  while $s < K$ do
4      $v = \text{random } [0, 0.1]^l$;
5      $\delta = v + rand(D)$;
6      $x^* = x_0 + \delta$;
7      if $CER(f(x^*), y_t) < \epsilon$ then
8          Put $\delta$ into $\tilde{P}$;
9          $\epsilon = \mathrm{CER}(f(x^*), y_t)$;
10         $x_0 = x_0 + \delta$;
11     else
12         $s = s + 1$;
13     end
14 end
15 return $\tilde{P}$
```

3.2 Perturbation Optimization

Even though Algorithm 1 generates proper perturbations for any voice commands, the amplitude of the perturbation may become overwhelming. Revisiting Eq. (1), to acquire the minimal perturbations, we need to gradually increase the perturbation power. However, due to the black-box setting, the gradient is inaccessible. As a result, we use Sign-Opt [19] to estimate the gradient, since Sign-Opt has achieved superior performance with the least number of queries, as written below:

$$\nabla\mathcal{L}(x) \approx \sum_{q=1}^{Q} sign(\mathcal{L}(x+\sigma\mu_q) - \mathcal{L}(x))\mu_q, \tag{2}$$

$$sign(\mathcal{L}(x+\sigma\mu) - \mathcal{L}(x)) = \begin{cases} +1, & f(x+\sigma\mu) \neq y_t \\ -1 & f(x+\sigma\mu) = y_t \end{cases} \tag{3}$$

where x is the general representation of $x_0 + \delta$, q and Q denote the noise index and the total number of noises respectively. σ is the search variance and μ is the noise. The key idea of Sign-Opt is to search the gradient space using the natural evolution strategy. Since $\mathcal{L}(x)$ is unknown, Sign-Opt queries $f(\cdot)$ in Eq. (3). The feedback of the target model can be collected to measure the number of wrong predictions. The result will be used to guide Eq. (2) in searching for the gradient of $\mathcal{L}(x)$.

Query-Efficient Fine-tuning: The perturbation generation typically requires ~5k queries to craft an AE [16, 19]. To further reduce the cost of queries, we design a query-efficient AE generation scheme to greatly reduce the query number.

By carefully examining the Eq. (2), we realize that the gradient estimation step depletes most of the queries. Suppose $Q = 50$, then it uses 50 queries to catch the $f(\cdot)$ result and estimate gradient according to Eq. (3). However, Sign-Opt [19] uses the estimated gradient only once for updating x, with a small update learning rate, while most of the gradient computations are wasted. In our design, we estimate the gradient once, then apply the estimated gradient multiple times to update the δ until it does not satisfy our attack goal, then do the gradient estimation again.

The workflow of our proposed query efficient phoneme-level adversarial perturbation generation is shown in Fig. 4. There are three major steps to generate AEs and perturbations: *searching*, *proposing*, and *fine-tuning*. In the searching and proposing phases, unlike the prior study [19] which only searches for random noise and keeps the shortest initial perturbation while discarding others, we reserve all the perturbation candidates to increase the generation speed. In the fine-tuning phase, we optimize all the proposed perturbations through gradient estimation. Note that there are three paths from the Query block: ② is used to update the perturbation consecutively until it cannot be further optimized. Then, we will re-calculate the gradient (①). Once the power of perturbation is lower than ϵ, we add it into the perturbation set P (③).

3.3 Weak Synchronization Design

Considering the adversary needs reaction time to play the perturbation, the generated perturbations are demanded to be robust against the mismatch of insertion positions. To realize such an attack, we seek to minimize the average loss instead of the instant loss. That is, we take the impact of mismatch into consideration and expect the comprehensive loss to be minimized. Mathematically, the average loss can be expressed as follows:

$$\overline{\mathcal{L}(x)} = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_i(x), \tag{4}$$

$$\mathcal{L}_i(x) = \mathcal{L}(x + c\tau), \tag{5}$$

where τ represents the mismatch interval, c controls the length of a mismatch period, i indicates the id of related losses, and N is the number of involved $\mathcal{L}$. To minimize the average loss, we can refer to Eq. (2) and Eq. (3) to estimate $\nabla\overline{\mathcal{L}(x)}$ by computing $\nabla\mathcal{L}(x + c\tau)$. The drawback of the average loss gradient estimation is that it costs $N\times$ more queries to perform the gradient estimation. The length of phonemes in D varies from $50ms$ to $300ms$, and one-word duration is ranging from $281ms$ to $387ms$ according to the report [57]. We expect that the phoneme-level perturbation can be plugged within the duration of one word, otherwise, it will be difficult to maintain the minimal $\mathcal{L}$ especially when a delayed perturbation arrives. In this paper, we set the $N = 4$ and $\tau = 100ms$. Fig. 5 depicts the perturbation mismatch scenario: when crafting the first red perturbation, we gather the other losses by the same perturbation but with a different time delay. In the figure, $\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_3$ correspond to $c = 0$, $c = 1$, and $c = 2$.

3.4 Over the Air Attack Robustness

Besides the weak synchronization feature, the attack robustness is another important feature of PhantomSound. Existing work models the acoustic signal propagation to compensate for the propagation loss over the air [50]. But the heavy computation prevents them from being adopted in real time attack. Also, the quality of perturbation relies on the speaker's amplifier, and the additional distortion on such small perturbation is hard to model. Inspired by the prior work [43] who sets a frequency filter to guarantee the generated perturbation is ranging from 50-8,000 Hz. To guarantee the effectiveness of PhantomSound over the air, we follow their approach on configuring a frequency filter to mitigate the uneven frequency response caused by the hardware imperfection of the speaker, thereby enhancing the attack robustness.

4 EVALUATION

In this section, we first introduce our benchmark experimental setting to generate AEs and perturbations. Then, we evaluate PhantomSound thoroughly to validate its feasibility and robustness. Moreover, we measure the impacts of different parameters in tuning a successful attack. Our attack is successfully launched on four different ASR service APIs, and the five popular commercial IVC devices. We further conduct an user case study in section 4.8. This section describes the results in detail.

4.1 Target Model Selection

Since we are developing a general approach to generate perturbations to attack closed-source ASR systems and commercial devices, we will examine the effectiveness of AEs and perturbations on the most popular IVC devices available on the market. Specifically, we select Google Home (G-H), Google Assistant (G-A), Amazon Echo, Microsoft Cortana, and IBM WAA[1] as target IVC devices. Moreover, we target their respective ASR APIs, namely, Google Cloud Speech-to-Text API, Microsoft Azure API, Amazon Transcribe API, and IBM Watson API. As for Apple Siri, since there is no online speech-to-text API service available from Apple, we cannot perform PhantomSound due to the lack of querying feedback from its ASR

[1]WAA represents "Wav-Air-API". As IBM does not own a commercial voice assistant device, we record and replay our AEs over-the-air, and transcribe them with IBM Watson API. This process, named as WAA, simulates an IVC device that is integrated with an IBM Watson API [18].

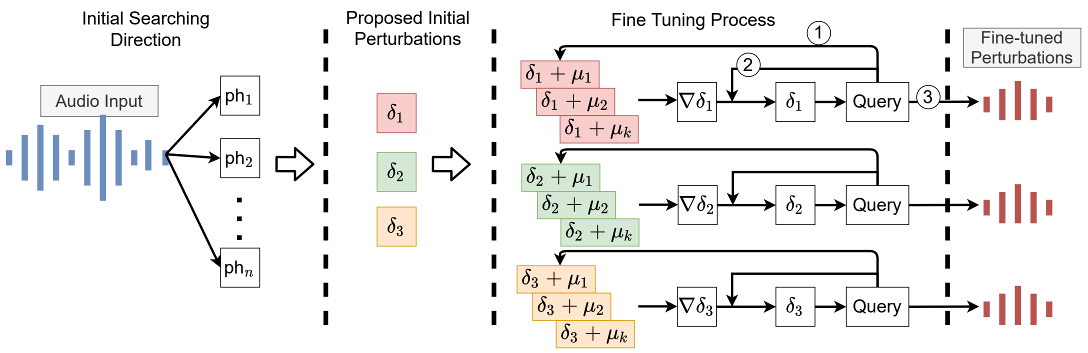


Figure 4: Adversarial perturbation generation

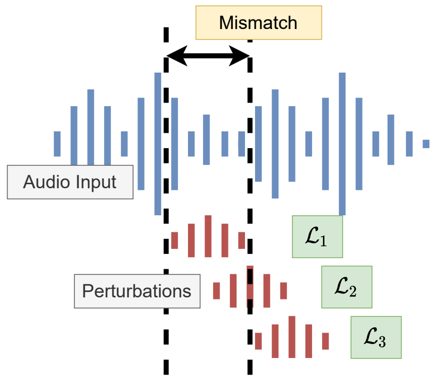


Figure 5: Perturbation mismatch during an attack.

system. For all the target systems, we only receive the hard label of the querying input from their APIs.

4.2 Metrics

We use the following metrics to quantify the effectiveness of our attack: *(1) Success Rate:* this metric represents the ratio of successful attacks and the total attempts. For an untargeted attack, as long as the AEs and the perturbations alter the prediction of the original input, we count it as successful. For a targeted attack, we report success only when the prediction matches the targeted class. *(2) Average queries per command:* we use the number of queries to imply the cost and speed of AE generation. Specifically, we measure how many queries it needs to craft a perturbation. This metric is calculated by the total number of queries over the number of crafted AEs/perturbations. *(3) L2 Distortion:* the L2 distortion $||\delta||^2$ indicates the size of perturbations. Prior to the launch of a physical attack, we can measure the distortion value by summarizing the squared amplitude of the generated perturbations. Note that the perturbation $\delta \in [0,1]^l$ and the initial phoneme-level distortion ranges from 50 to 1,600 depending on different phonemes, which will be optimized after the perturbation fine-tuning as shown in Section 4.5. *(4) False Accept Rate:* the false accept rate is measuring the probability of that the attacks can be false accepted by the liveness detection methods. We use this metric to evaluate the ability of our attacks to bypass the existing defense methods (e.g. liveness detection) compares to the existing attacks. The higher false accept rate we achieve, indicating the more dangerous of attack is, to bypass the existing liveness detection methods.

4.3 Dataset

The dataset we choose as original input is speech commands v0.02 [60] released by Google Brain. This dataset is designed to validate the keyword detection capability of DNN models. It contains 105,829 utterances of 35 common one-word commands (e.g., “yes", “learn", “stop"), which is recorded from 2,618 volunteers. To validate the effectiveness of PhantomSound on a longer command, we record 10 longer commands (partially listed in Table 4) from a volunteer.

For the phoneme dataset, we expect to obtain all 44 pure English phonemes with flexible duration. Existing speech datasets (e.g., Arabic Speech Corpus [32], TIMIT [1]) include the annotations of phonemes, but it requires extra efforts to extract individual phonemes with different duration from the speech audio. Besides, PCVC dataset [2] only involves 12 volunteers, and scikit phoneme dataset [51] only contains 5 vowels. To construct a phoneme dataset with a diverse set of speakers, we use 200 different audios from 200 speakers in speech commands v0.02, remove the silence in the recordings, and randomly cut audio clips with a duration between 50ms to 300ms. This phoneme processing step follows that of the scikit phoneme dataset [51], which results in 453 audio clips in total.

Table 2: Dataset description (“unique cmds" refers to the number of unique target commands, and “total audios" refers to the total number of (adversarial) audios that lead to the target commands).

	Phone.	Cmd.	Untargeted	Targeted
Unique cmds	-	45	1785	64
Total audios	453	300	6,219	216

Table 2 records the number of involved data including phonemes, commands, untargeted perturbations, and targeted perturbations. We use 35 one-word commands from the speech commands v0.02 dataset, along with 10 self-recorded long commands to build a command dataset with 45 different commands, including 300 audios in total. Then, we apply the proposed algorithm to randomly generate AEs and perturbations for an untargeted attack, resulting in 1785 different commands and 6,219 adversarial audios on 4 different commercial APIs. For the targeted attack, we attempt the perturbation of keywords, and generate 64 target commands with 216 adversarial audios.

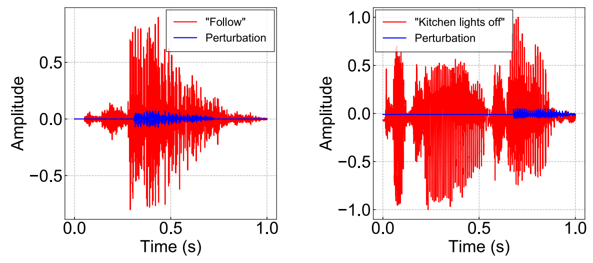
4.4 Experiment Setting

We conduct the experiments on a desktop with Intel i7-7700k CPUs, 32GB RAM, and 64-bit Ubuntu 18.04 LTS operating system. The experiments are performed at three locations with different noise floors. We use three loudspeakers, including LG monitor built-in speaker (at the apartment), an SADA D6 home small speaker (at the lab), and an Samsung S9 phone (at outdoor), to transmit AEs (i.e., AE attack) and perturbations (i.e., perturbation attack) to the victim devices. Figure 12(a) demonstrates the attack scenario: the victim speaks commands into a smartphone or Google Home mini, while the attacker plays the perturbation through a speaker.

4.5 Attack Performance

We first evaluate the functionality of AE generation in PhantomSound. The purpose of this evaluation is 1) to demonstrate that the perturbation amplitude is negligible compared with the input, and 2) to prove the query efficiency of our phoneme-level searching algorithm. Then, we conduct the physical attack and validate the robustness of our attack over the air.

Attack Over-the-line: We first evaluate the attack by targeting the ASR APIs. The adversarial audios are directly supplied to the online APIs. We randomly select 20 adversarial audios from every command, and then perform the untargeted attack by searching for 100 epochs ($K = 100$ in Algorithm 1). Then, the generated perturbations are optimized to suppress their power. In the end, we obtain 148 AEs and perturbations from ~44k queries ($Q = 30$ in Eq. (2)), i.e., 301 queries per AE on average.



(a) One-word commands　　(b) Command phrases

Figure 6: Comparison of input and perturbation amplitudes.

To evaluate the perturbation amplitude, we randomly pick two examples from the generated perturbation as shown in Fig. 6. We can see that the crafted perturbations have a negligible power profile compared with the input regardless of the length of commands. Moreover, the duration of perturbation is shorter than the input, which makes it possible to conceal the presence of perturbation. Table 3 summarizes the results of the untargeted attacks toward 4 types of commercial APIs. We observe that every command can be altered into at least two false commands. While some of the false predictions are harmless, the attack can almost certainly *invalidate* the victim's command. Moreover, in certain cases, some perturbations can lead to a contrary response from voice assistants (e.g., "right" to "wrong" in Amazon Transcribe API, "right" to "no" in Microsoft Azure API). Considering the number of queries for generating one perturbation, the Google Cloud Speech-to-Text is reported to be the most resilient API under our attack, as it requires the most queries.

Table 3: Untargeted attack results.

Cmds.	Google Cloud	MS Azure	AMZ Trans.	IBM Watson
"down"	"damn" "done" "does"	"town" "one" "south"	"done" "dine" "drive"	"Downer" "Done" "Drone"
"follow"	"fallout" "farm" "four"	"fallout" "fall over" "learn"	"no" "for sure" "phone"	"fallen" "fall over" "fall"
"forward"	"forewarn" "for eyes" "for work"	"work" "for" "ford"	"what" "work"	"for" "four" "for all"
"yes"	"yeah" "yeah!" "yet"	"file" "4" "On"	"yeah" "yes.." "right"	"yeah" "yet" "hi"
"right"	"Rite Aid" "write" "read"	"no" "go" "trade"	"write" "run" "wrong"	"run" "ray" "left"
Queries	345	251	215	314

To further comprehend the query effectiveness, we conduct an additional experiment to validate the sensitivity of different APIs in terms of request rejection rates. The result shows that Google API is most sensitive as it refuses to respond to an unclear input, while the Amazon transcribe always responds to any inputs. Table 4 records the targeted attack results towards a longer input. The results show

Table 4: Targeted attack results

Command		Query			
Input	Target	Google Cloud	MS Azure	AMZ Trans.	IBM Wat.
"turn **right**"	"turn **left**"	1,895	1,128	1,421	1,487
"kitchen lights **off**"	"kitchen lights **on**"	1,754	857	933	1,377
"call **mom**"	"call **911**"	-	1,421	1,125	-
"**read** my message"	"**delete** my message"	2,342	1,520	1,436	1,781
Average Queries		1,997	1,232	1,229	1,548

that our phoneme-level searching method is capable of finding the specific perturbation that could mislead the APIs to return a target result. Note that the average query amount increases dramatically in the targeted attack case, which is anticipated because the target need to be achieved by multiple round perturbation searching (line 7-10 in Algorithm 1). It is also noteworthy that our targeted attack cannot guarantee finding a successful perturbation under any arbitrary inputs (e.g., Google Cloud fails to craft AEs for "call 911").

Query Efficiency Comparison on Known models: To validate the benefits of introducing phonemes to guide the optimization direction, we implement 3 different attacks on two known models. By attacking two ASR models (DeepSpeech 1/DS 1 [33] and DeepSpeech 2/DS 2 [11]) with different prior knowledge and method,

Table 5: Comparison for Untargeted Attacks

Models↓	Ours	white box[14]	score based[17]	brute force[12]
DS 1 [33]	**185**	**90**	206	∞
DS 2 [11]	**226**	**75**	197	∞

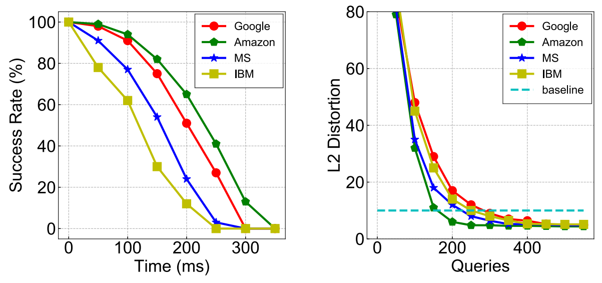
we find that PhantomSound achieves comparable query efficiency with the grey box setting, with 100% attack success rate. The result is summarized in Table 5. Given the same 10 benign commands, we use the 4 attacks to generate untargeted AEs with the same L_2 distortion. We record the average number of queries for different prior knowledge of the victim model. Compares to the white box attack, which can fine-tune in <90 queries, PhantomSound requests 200 queries to craft an AE, which is close to the queries of a score-based attack. This result indicates that our strategy such as 1) using phoneme to initialize perturbation 2) Query-efficient fine-tuning is working well, and performing similar results with less information (e.g., confidence score). It is noteworthy that the brute force decision boundary search method doesn't work for attacking the ASR model. Because this method initializes a random noise and retrieves model gradients by altering the noise. However, this noise can never be fine-tuned while the victim model produces an empty label to it, resulting in an infinite number of queries.

Table 6: Comparison for Targeted Attacks

Attacks	Knowledge	Queries	SR
Carlini [14]	Gradient	~1,000	100%
Houdini [20]	Gradient	~1,000	100%
Devil's [18]	Conf. Score	~1,500	100%
OCCAM [69]	Final decision	~30,000	100%
Ours	**Final decision**	**~1,500**	**68%**

Query Efficiency Comparison for Targeted Attacks: We compare the number of required queries with four existing attacks in Table 6. The white-box attacks (Wb) [13, 20] require the least amount of queries (~1,500). With the knowledge of confidence scores of API's decoding results, the Devil's Whisper [18] utilizes a surrogate model trained with around 1,500 queries to attack the APIs. In the scenario when an attacker can only access the final decision of the query API, PhantomSound needs ~1,500 queries (comparable with the white-box setting) to craft a targeted perturbation. Compared with a recent black-box attack OCCAM [69], we reduce the number of queries by 95%. However, due to the limitation of phoneme length and diversity, we sacrifice the success rate to achieve high query efficiency.

Weak synchronization: Before evaluating the physical attacks, we investigate the effectiveness of the proposed weak synchronization design. In this experiment, we manually add mismatch delays between input x_0 and the generated perturbation to craft mismatched AEs. We then use the mismatched AEs to query the APIs and measure the attack success rate. Fig. 7(a) displays the result, from which we can see that, after using the average loss, although we expect the weak-synchronization works within 400ms (detailed in Section 3.3), this design is only partially effective, because the success rate drops steadily with the increasing mismatch time. Moreover, we show the tendency of L2 distortion w.r.t. the number of queries in Fig. 7(b). The baseline denotes an L2 distortion of 10, which is proven unnoticeable by two volunteers when AEs are played using an LG monitor with a medium volume.



(a) Weak synchronization (b) L_2 Distortion vs. No. of queries

Figure 7: Evaluation of AE generation process.

Attack Over-the-air: The over-the-air attack evaluation aims to prove the robustness of PhantomSound.

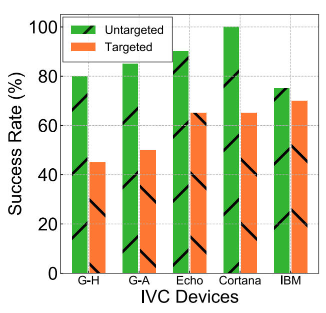
To attack commercial APIs, we play the valid AEs and perturbations (which attack successfully in over-the-line scenarios) via a SADA D6 speaker, and record it by iPhone 12 Pro, the recordings are sent to the commercial API for evaluation. The attack distance is set to 50cm. For each attack, we choose 5 AEs to play 5 times and get the average success rate. We report the result in Table 7. From

Table 7: Over-the-air attack API baseline

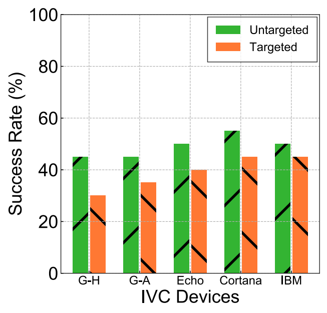
APIs		Google Cloud	MS Azure	AMZ Trans	IBM Wat.
Targeted	AE	76%	80%	80%	84%
	Pert.	68%	72%	72%	76%
Untargeted	AE	100%	100%	100%	100%
	Pert.	72%	80%	80%	92%

our observations, it is apparent that in the context of a targeted attack, our method attains approximately a ~ 80% success rate in attacking over-the-air commercial ASR APIs by directly playing the audio adversarial example (AE). When the attack is synchronized with the victim's speech, the perturbation attack exhibits around a ~ 72% success rate. On the other hand, when it comes to untargeted attacks, our adversarial examples (AE) and perturbation methods achieve impressively high success rates. They misdirect the victim's input with a 100% and approximately 81% success rate, respectively. Next, we follow the same setting to attack commercial IVC devices.

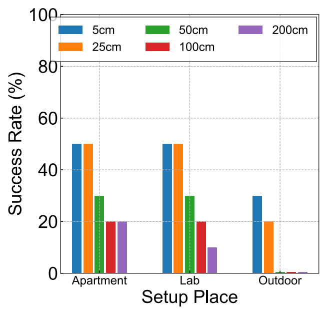
The result in Fig. 8(a) uncovers the success rate of playing AEs directly. Among all the tested IVC devices, Microsoft Cortana is most vulnerable against the AE attack, while the Google series products (e.g., Google Home, Google Assistant) show the most resilience against the targeted AE attack. Overall, the success rate of an untargeted attack is higher than that of a targeted one, i.e., the former reaches ~80% success rate and the latter stays around ~50%. With the perturbation attack, Fig. 8(b) reveals a relatively low success rate. Similarly, compared to the targeted perturbation attack, the untargeted attack has a higher success probability, achieving around 45% success rate on average. Nevertheless, the success rate can be further improved via multiple repeated attempts. We also



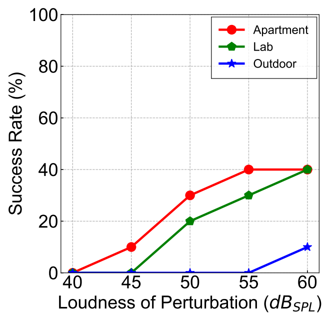
(a) AE attack



(b) Perturbation attack



(c) Distance



(d) Loudness

Figure 8: AE generation results.

Table 8: Comparison with other real-world attacks

Target	Google Cloud	MS Azure	AMZ Trans.	IBM Wat.	Google Home	Google Assit.	MS Cortana	AMZ Echo
Devil's [18]	10/10	10/10	4/10	10/10	9/10	10/10	10/10	10/10
Danger [68]	-	-	-	-	15/100	-	-	69/100
Ours	19/25	20/25	20/25	21/25	11/25	12/25	16/25	16/25

summarize the success rate compared to prior black-box attacks in Table 9.

Upon comparison with the Devil's attack [18], it is evident that our attack method yields a marginally lower success rate against the APIs, with the exception of the Amazon Transcribe API. Considering the IVC devices, the Devil's attack tends to be more effective at similar SNR levels. For the Danger attack [68], we have displayed their success rate derived from their "voice squatting" attack, where the victim's command is misinterpreted to initiate the attack skill. A comparison reveals that our attack technique yields comparable success rates when targeting Amazon Echo, and even demonstrates superior performance when used to attack Google Home.

Table 9: Latency for perturbation generation

Time Consumption	Google Cloud	MS Azure	AMZ Trans.	IBM Wat.
Latency (s)	0.29	0.58	26.31	1.35
Untargeted (min)	1.67	2.43	94.3	7.1
Targeted (min)	9.65	11.9	539	34.8

Time Cost: Different from the prior works that require a substantial amount of time to craft AEs offline, PhantomSound enables much faster AE generation. Such a fast generation feature is essential in practice, when the attackers only have a limited time budget to instantiate the attack.

In the experiment, we record the latency for querying 4 different commercial APIs to get the results. The results are presented in the first row of Table 9, which show that 3/4 of APIs could return a result in seconds, except Amazon Transcribe API. The Amazon API has to interact with Amazon Web Service and Storage bucket, which spends a longer period for the results to return.

We then compute the total time needed for perturbation generation, by multiplying latency with the number of queries (shown in Table 3, 4). Our result shows that PhantomSound can generate a perturbation for both the targeted and untargeted attacks in minutes with the exception of Amazon API, while the targeted one takes longer. Note that we take the L_2 distortion into consideration during the time cost computation, however, if the attacker ignores the impact of the perturbation loudness and uses the intermediate perturbation, the generation time can be further reduced.

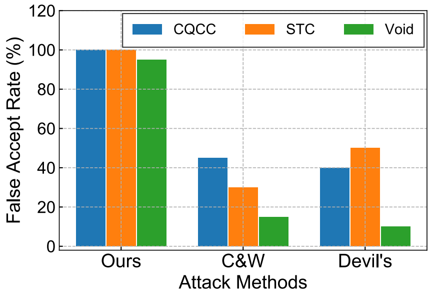


Figure 9: Attacks vs. liveness detection defenses

4.6 Ability to Bypass Liveness detection

Compares to the existing physical adversarial attacks [18, 64, 69], PhantomSound relies on the benign commands spoken by the user. Although this attack setting requires extra effort to synchronize the perturbation and the user's benign speech, it brings potential benefits to bypassing the defense mechanism. For example, recent works [5, 28, 38, 40, 42, 45] proposed liveness detection approaches can differentiate the source of sound (human or machine) with high accuracy. Therefore, the conventional adversarial attacks that are launched solely by loudspeaker [18, 64, 69] have a higher probability to be defended by those liveness detection methods. In contrast, our attack is designed to launch with the user's speech, leading to a more dangerous threat to the liveness detection defenses. To validate the performance of PhantomSound over different defense mechanisms, we reproduce three liveness detection algorithms, CQCC [38], STC [40], and Void [5]; For comparison, we implement C&W attack [14] and Devil's [18]to attack with liveness detection algorithms. The detailed liveness detection methods can be found in Appendix A. To conduct this experiment, we follow the settings described as follows:

Ours: We play our perturbation when the user gives the command, and record it with a smartphone. Then, we run three liveness detection algorithms to detect the sound source.

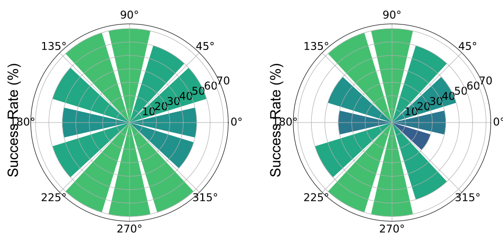
C&W [14]: We play the AEs that are generated by this attack, and then record with the same smartphone and run liveness detection algorithms to defend it.
Devil's [18]: We play the AEs provided from the paper's demonstration website, and then record it with the same smartphone, followed by the same liveness detection procedure.

For our attack and the C&W attack, we use 20 different perturbations/AEs to attack the liveness detection model; As for the Devil's attack, since we can only collect 10 AEs from the demonstration website, we use 10 AEs to attack the liveness detection model. We present our result in Fig. 9. It is evident to show that our attack can bypass the three liveness detection models, resulting 95% to 100% false accept rate. In contrast, the other two attacks have a very low chance to counter the Void [5] detection with less than 15% FAR. Even for conventional liveness detection methods (e.g., CQCC and STC), the existing attacks that use complete AEs also have a low probability (40%) to attack successfully.

4.7 Impact of Practical Factors

To investigate the critical factors that may affect the success rate of PhantomSound, we evaluate the perturbation attack under different environments (e.g., apartment, lab, outdoor). The ambient noise level for the aforementioned places are 39.8 dB_{SPL} (apartment), 41.2 dB_{SPL} (lab) and 58 dB_{SPL} (outdoor), respectively.

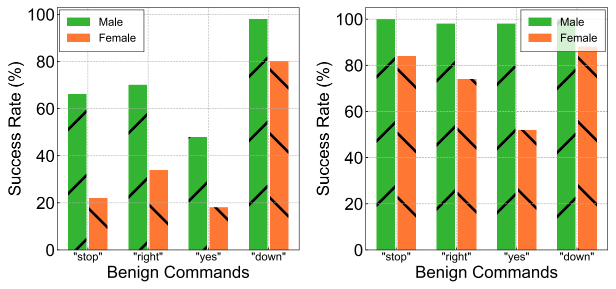
In this experiment, we play a crafted perturbation of "turn right" 10 times, attempting to transform the prediction into "turn left", and the volume of perturbation is 60 db_{SPL}. We then record the success rate under different circumstances. Fig. 8(c) demonstrates the impact of attack distances, i.e., the closer the adversary is, the higher success rate he/she achieves, which is unsurprising given that our attack relies on the successful delivery of the perturbation. The relatively short attack distance is in fact a common limitation reported by the existing work [18, 43, 64]. However, the attacker can further extend the attack distance by increasing the speaker's volume (though it could make the perturbation more noticeable) or utilizing a speaker array [49]. Next, we provide the results on how the loudness factor could affect the attack performance in Fig. 8(d). We can see that the success rate improves with the increasing perturbation loudness. This result also coincides with the prior work [43]. In an outdoor environment, it is suggested that the adversary enhance the attack robustness by amplifying the perturbations. Due to the higher noise level outdoors, the phoneme-like perturbation can still be hard to perceive.



(a) Attack Angle iPhone 12Pro (b) Attack Angle Mi 8 Lite

Figure 10: Attack with different angles

Impact of Attack Angles: Besides the attack environments and the distance, the attack angle can also alter the attack performance. We evaluate our attack by playing AEs to two smartphones in 12 different directions (from 0 degrees to 360 degrees, with 30-degree intervals). This experiment is conducted in Lab environment and attacks the google assistant on the smartphone. We play 10 AEs in every direction with 60dB_{SPL}, and record the success rate of the untargeted attack. We report our result in Fig. 10. We find that our attack has the best performance when the adversary is facing or back to the smartphone. While attacking through the side direction (e.g., 0 degrees when the adversary is parallel to the victim), the success rate is impaired. We observe the same trend on two smartphones. This result indicates that the microphone arrangement and its direction will lead to audio information loss. Unfortunately, the low power of our perturbation is hard to be sensed with the audio loss, therefore causing a low success rate in the side direction.
Impact of Different victims: In the attack preparation period, every perturbation is crafted based on a specific command from a specific speaker. However, the adversary may use the crafted perturbation on the previous victim to attack the current victim. Here, we evaluate the capability of PhantomSound to attack different speakers. First, we obtain 4 perturbations from speaker #1 (male), which convert the benign commands "stop", "right", "yes", and "down" into 4 target commands "backward", "left", "no" and "song" respectively. Next, we randomly select 100 speakers (50 males and 50 females) who are not speaker #1 from the speech commands v0.02 dataset, and inject the perturbations into their benign audio samples. For the targeted attack, if the benign commands are successfully interpreted as the target, we classify it as successful. For the untargeted attack, any case where the benign commands are misinterpreted is considered successful. The result is present in Fig. 11.



(a) Targeted attack (b) Untargeted attack

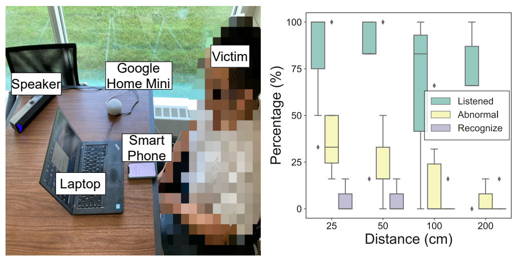
Figure 11: Attack cross different victims

The result indicates that, for targeted attacks, the attack success rate is dependent on the benign samples. The success rate exceeds 50% when the target is of the same gender, but it falls below 40% when targeting different genders. Regarding the untargeted attacks, the perturbations demonstrate robust transferability for attacking various speakers. The average success rate is notably high, reaching 98% for males and 74% for females.

4.8 User Study

To evaluate the stealthiness of perturbation in a real-world attack, we conduct an online/in-person user study to investigate the users' perception level of PhantomSound. In our study, 20 volunteers are involved, and they are requested to hear 6 crafted perturbations across 4 different distances. Two volunteers attend the in-person experiment (see Fig. 12(a)) and the rest of them carry out the experiment at their homes. We recruit 13 volunteers from Amazon Mechanical Turk with complete experimental instructions. The experiment setup detail can be found at Appendix B.

The volunteers are asked to *pretend speaking to their voice assistants* while hearing the perturbation, after which they will answer questions to depict their comprehension of the heard perturbations. The options for perception levels include: *Listened*, *Abnormal*, and *Recognize*. *Listened* indicates that the volunteer can hear a perturbation but regard it as a normal noise; *Abnormal* implies that they hear some strange sounds; and *Recognize* stipulates that they can understand the meaning of the heard sound. We report the experiment result in Fig. 12(b). It shows that most of the participants can hear the perturbation within a short distance, but less than 50% of them regard the perturbation as an abnormal sound. Such "abnormality" feeling will gradually disappear with the increasing attack range, which ends with 10% in 2 meters. Moreover, even though all the perturbations are "meaningless" phonemes, some participants claim to understand their meanings (though the understanding is incorrect). To summarize, PhantomSound can be noticed by victims, but would not vastly raise their attentions. Notably, the victims are generally unaware of the meaning of perturbations.



(a) Experiment setup (b) Users' perception level of PhantomSound

Figure 12: Real-world user study of PhantomSound.

5 DISCUSSION

5.1 Low-cost Attack

Table 10 lists the cost comparison between PhantomSound and the existing work [18]. The first row records the pricing information of the commercial APIs, which is measured by the duration of given audios (in minutes). The recent black-box attack [18] is reported to incur the cost of 1,500 queries for building the substitute models, and every query uses an audio with 25 seconds long. In total, such an attack requires $1500 * 25/60 = 625$ minutes to train a surrogate model, and can only generate 10 pre-selected commands. To generate extra commands, the attacker needs to submit additional queries (~100) for the candidate AEs. Suppose the length of candidate AEs is 6 seconds, the total time cost for generating extra AEs is $6 * 100/60 = 10$ minutes. All together, the duration of queried audio is 72.5 minutes for producing one single AE. In contrast, PhantomSound does not require a substitute model, and as such, it only takes ~300 queries and ~2,000 queries of one-second audios to craft an untargeted AE (Ours-U) and a targeted AE (Ours-T) respectively. We then present the cost to generate one AE based on the pricing and the query audio length (shown in row 4 and 5). In the end, PhantomSound saves 93.1% and 65.5% of the cost for crafting an AE, a drastic improvement.

5.2 Limitations

The limitation of PhantomSound includes that: 1) the attack is sensitive to ambient noise; 2) there is no guarantee to generate an AE for any input and any target; 3) this attack could not substantially modify very long sentences; 4) the attack distance is relatively short as presented in Section 4.7. To address the first and the forth limitation, the adversary can either amplify the perturbation power or attack the victim in a relatively quiet place. The second and third limitations are possibly addressed using multiple repeated attempts of phoneme injections, which will increase the likelihood of generating a successful perturbation with a potential caveat of growing costs.

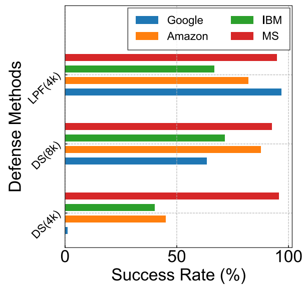
Table 10: Cost comparison

	Google	MS	AMZ	IBM
Pricing/min	\$0.024	\$0.016	\$0.024	\$0.01
Build model [18]	625 min			
Craft AE [18]	10 min			
Total time/AE [18]	72.5 min			
Total time/AE (**Ours**)	5 min - 25 min			
Cost/AE [18]	\$1.74	\$1.16	\$1.74	\$0.725
Cost/AE (**Ours-U**)	\$0.12	\$0.08	\$0.12	\$0.05
Cost/AE (**Ours-T**)	\$0.6	\$0.4	\$0.6	\$0.25
Saving/AE (Ours)	**93.1%/65.5%**			

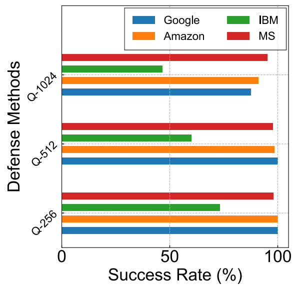
5.3 Defense

Prior studies [43, 63, 64] reveal that the audio adversarial attack can be defended by signal processing techniques, since the adversarial perturbations are delicately crafted and hence are deemed fragile. The signal processing techniques, however, can reduce the fidelity of perturbations and hence protecting the ASR models. Typical signal processing defense methods include 1) *Down sampling (DS)*: decreasing the sampling rate of AEs to disrupt the quality of AEs [43, 63, 64]; 2) *Quantization*: as the original AEs are encoded by 16-bit values, the quantization technique rounds the 16-bit precise value to its nearest integer multiple of Q, where Q represents the quantization level. A higher Q results in a lower precision of AEs, which has been adopted to defend against the attacks [43, 63]. 3) *Low pass filtering (LFP)*: the defense can use a Butterworth low-pass filter with different cutoff frequencies to remove the high-frequency components of the perturbations [43].

We reproduce the aforementioned three defense methods to test their effectiveness against PhantomSound. Specifically, for *DS* approach, we modify the sampling rate of AEs from 16k to 8k and 4k. In the *quantization* setting, we follow the existing work [43] to set Q as 256, 512, and 1,024. Then, we build a Butterworth low-pass filter with a cutoff frequency of 4kHz, and set the order of the filter as 6. To validate the defense performance comprehensively, we generate **1,190** AEs from 20 clean audio samples and process them with 6 different defense settings.



(a) Defense performance of down sampling and low pass filter



(b) Defense performance of quantization

Figure 13: Performance of PhantomSound against different defenses.

We use the processed AEs to attack 4 commercial ASR APIs, and present the results in Fig. 13. Fig. 13(a) shows that *LPF* can barely impact the attack success rate of AEs and APIs. For comparison, the *DS* technique slightly changes the attack success rate from 100% to 92.4% (Microsoft), 71.4% (IBM), 87.5% (Amazon), and 63.3% (Google). This method can further reduce the success rate by applying a lower sampling rate (e.g., with 4k sampling rate, the IBM and Amazon API can defend against ~60% attacks, while the Google API is not supported for the audio input with such a low sampling rate. Different from the findings from previous work [43, 63] that quantization is effective in defending against the adversarial attack, our results show a converse performance. From Fig. 13(b), we observe that only the IBM API can be affected by the quantization, which reduces the success rate to 73%, 61%, and 47% for q=256, 512, and 1,024, respectively. To summarize, our results demonstrate that the existing signal processing-based defense approaches cannot protect the commercial APIs from PhantomSound. Future research on defense mechanisms are needed to provide more secure speech-to-text and voice assistance services.

5.4 Ethical Issues

The intention behind publishing this work is to enlighten the academic and tech community about the vulnerabilities of commercial ASR APIs and smart speakers, it may also provide malicious actors with the knowledge and tools to exploit these vulnerabilities for harmful purposes, such as privacy invasion, identity theft, or unauthorized control of connected devices. If the findings of this paper are misused, malicious actors could potentially manipulate smart speakers into sharing sensitive information or performing unauthorized actions, there may be potential financial and reputational harm to individuals and corporations. To address these ethical concerns, it would be advisable to collaborate with manufacturers of smart speakers to design effective countermeasures to defend against this attack.

6 RELATED WORK

The study of adversarial attacks starts from the discovery of intriguing properties of the neural networks around 2014 [21, 54]. Researchers manually or automatically add small perturbations to the input and thereby misleading the neural network models.

Adversarial Attacks against ASR Systems: Existing work [7, 14, 20, 48] has proposed different optimization algorithms to craft effective AEs towards ASR models with some knowledge of the victim's ASR model (e.g., prediction scores or logits output). However, the robustness of their attack approaches in a real-world over-the-air scenario is usually unverified. The recent physical attacks such as CommanderSong [64], Devil's Whisper [18], and AdvPulse [43] require a substantial cost (in time and money) for the attackers to succeed in attacking the black-box voice assistants.

Signal Processing Attacks: Rather than exploiting the vulnerabilities of neural networks in ASR systems, the signal processing attacks aim at attacking the signal pre-processing or feature extraction modules. They usually exploit the discrepancies between the human auditory system and the perceptual hearing system of microphones to fool the ASR system. These attacks analyze the input and output of the feature extraction procedure, and then they modify the input of feature extraction and preserve the shape of output to either hide their attack [3] or mislead the ASR system in producing incorrect transcriptions [4]. Even though the existing signal processing attacks demonstrate the efficiency and effectiveness against the black-box models, it is relatively straightforward to defend against using frequency filters.

Audio Backdoor Attacks: Different from adversarial attacks which attack a trained model, backdoor attacks [26, 31, 44] inject backdoor triggers during the training process. Recently, researchers demonstrated that the backdoor attack [52, 65] can also be implemented in the ASR model and Speaker Verification models. To defend against the backdoor attacks in the image domain, several countermeasures are proposed [29, 30].

Other Related Works: Some attackers exploit the imperfection of hardware (e.g., microphone) to deliver inaudible attacks through different media [41, 53, 62, 67]. Besides, Danger [68] uses homophones (i.e., different words with similar sounds) to attack ASR skills. Researchers also develop side-channel attack [59] by injecting voice commands through a power line. Speech synthesis attack produces victim's fake speech by generative models [61]. To protect the victim's original speech, researchers add perturbations([35, 58]) to prevent the generating of deep fake speech.

7 CONCLUSION

In this work, we proposed PhantomSound, a practical, black-box, and query efficient audio attack against commercial ASR systems and IVC devices in a real-world scenario. As opposed to the existing attacks that require prior knowledge of the target model, we propose a phoneme-level searching method to generate AEs and perturbations rapidly and effectively in a black-box setting. In the real-world experiments, PhantomSound is shown to be practical and robust in attacking 5 popular commercial voice controllable devices over the air, which could potentially cause hazard to the smart home.

ACKNOWLEDGMENTS

We would like to extend our appreciation to our shepherd and the anonymous reviewers for their invaluable input on our study. This work was supported in part by the U.S. National Science Foundation grant CNS-1950171, CNS-2226888 and CCF-2007159.

REFERENCES

[1] 1993. TIMIT Acoustic-Phonetic Continuous Speech Corpus. https://catalog.ldc.upenn.edu/LDC93S1. Accessed: 2021-11-04.

[2] 2018. Persian Vowel recognition with MFCC and ANN on PCVC speech dataset. https://github.com/smalekz/PCVC. Accessed: 2021-11-04.

[3] Hadi Abdullah, Washington Garcia, Christian Peeters, Patrick Traynor, Kevin RB Butler, and Joseph Wilson. 2019. Practical hidden voice attacks against speech and speaker recognition systems. *arXiv preprint arXiv:1904.05734* (2019).

[4] Hadi Abdullah, Muhammad Sajidur Rahman, Washington Garcia, Kevin Warren, Anurag Swarnim Yadav, Tom Shrimpton, and Patrick Traynor. 2021. Hear" No Evil", See" Kenansville"*: Efficient and Transferable Black-Box Attacks on Speech Recognition and Voice Identification Systems. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 712–729.

[5] Muhammad Ejaz Ahmed, Il-Youp Kwak, Jun Ho Huh, Iljoo Kim, Taekkyung Oh, and Hyoungshick Kim. 2020. Void: A fast and light voice liveness detection system. In *USENIX Security*.

[6] Raziel Alvarez and Hyun-Jin Park. 2019. End-to-end streaming keyword spotting. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 6336–6340.

[7] Moustafa Alzantot, Bharathan Balaji, and Mani Srivastava. 2018. Did you hear that? adversarial examples against automatic speech recognition. *arXiv preprint arXiv:1801.00554* (2018).

[8] Amazon. 2021. Amazon Echo. https://www.amazon.com/All-New-Echo-4th-Gen/dp/B07XKF5RM3.

[9] Amazon. 2021. Amazon Transcribe. https://aws.amazon.com/transcribe/.

[10] Amazon. 2021. Skills. https://developer.amazon.com/en-US/alexa/alexa-skills-kit.

[11] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. 2016. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*. PMLR, 173–182.

[12] Wieland Brendel, Jonas Rauber, and Matthias Bethge. 2017. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248* (2017).

[13] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*. IEEE, 39–57.

[14] Nicholas Carlini and David Wagner. 2018. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 1–7.

[15] Guangke Chen, Sen Chenb, Lingling Fan, Xiaoning Du, Zhe Zhao, Fu Song, and Yang Liu. 2021. Who is real bob? adversarial attacks on speaker recognition systems. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 694–711.

[16] Jianbo Chen, Michael I Jordan, and Martin J Wainwright. 2020. Hopskipjumpattack: A query-efficient decision-based attack. In *2020 ieee symposium on security and privacy (sp)*. IEEE, 1277–1294.

[17] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. 2017. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*. 15–26.

[18] Yuxuan Chen, Xuejing Yuan, Jiangshan Zhang, Yue Zhao, Shengzhi Zhang, Kai Chen, and XiaoFeng Wang. 2020. Devil's whisper: A general approach for physical adversarial attacks against commercial black-box speech recognition devices. In *29th USENIX Security Symposium (USENIX Security 20)*. 2667–2684.

[19] Minhao Cheng, Simranjit Singh, Patrick Chen, Pin-Yu Chen, Sijia Liu, and Cho-Jui Hsieh. 2019. Sign-opt: A query-efficient hard-label adversarial attack. *arXiv preprint arXiv:1909.10773* (2019).

[20] Moustapha M Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. 2017. Houdini: Fooling deep structured visual and speech recognition models with adversarial examples. *Advances in neural information processing systems* 30 (2017).

[21] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[22] Google. 2021. Google Assistant. https://assistant.google.com/.

[23] Google. 2021. Google Home/Nest. https://store.google.com/product.

[24] Google. 2021. Google Speech. https://cloud.google.com/speech-to-text.

[25] Google. 2021. Google Vision. https://cloud.google.com/vision.

[26] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2019. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access* 7 (2019), 47230–47244.

[27] Hanqing Guo, Yuanda Wang, Nikolay Ivanov, Li Xiao, and Qiben Yan. 2022. SpecPatch: Human-in-the-Loop Adversarial Audio Spectrogram Patch Attack on Speech Recognition. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 1353–1366.

[28] Hanqing Guo, Qiben Yan, Nikolay Ivanov, Ying Zhu, Li Xiao, and Eric J Hunter. 2022. SuperVoice: Text-Independent Speaker Verification Using Ultrasound Energy in Human Speech. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*. 1019–1033.

[29] Junfeng Guo, Ang Li, and Cong Liu. 2022. AEVA: Black-box Backdoor Detection Using Adversarial Extreme Value Analysis. In *International Conference on Learning Representations*. https://openreview.net/forum?id=OM_lYiHXiCL

[30] Junfeng Guo, Yiming Li, Xun Chen, Hanqing Guo, Lichao Sun, and Cong Liu. 2023. Scale-up: An efficient black-box input-level backdoor detection via analyzing scaled prediction consistency. *arXiv preprint arXiv:2302.03251* (2023).

[31] Junfeng Guo and Cong Liu. 2020. Practical poisoning attacks on neural networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16*. Springer, 142–158.

[32] Nawar Halabi. 2016. Arabic Speech Corpus. http://en.arabicspeechcorpus.com/. Accessed: 2021-11-04.

[33] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. 2014. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567* (2014).

[34] Yitao He, Junyu Bian, Xinyu Tong, Zihui Qian, Wei Zhu, Xiaohua Tian, and Xinbing Wang. 2019. Canceling inaudible voice commands against voice control systems. In *The 25th Annual International Conference on Mobile Computing and Networking*. 1–15.

[35] Chien-yu Huang, Yist Y Lin, Hung-yi Lee, and Lin-shan Lee. 2021. Defending your voice: Adversarial attack on voice conversion. In *2021 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 552–559.

[36] IBM. 2021. IBM Speeche. https://www.ibm.com/cloud/watson-speech-to-text.

[37] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. 2018. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*. PMLR, 2137–2146.

[38] Tomi Kinnunen, Md Sahidullah, Héctor Delgado, Massimiliano Todisco, Nicholas Evans, Junichi Yamagishi, and Kong Aik Lee. 2017. The ASVspoof 2017 challenge: Assessing the limits of replay spoofing attack detection. (2017).

[39] Bret Kinsella and Ava Mutchler. 2019. Smart speaker consumer adoption report. *Informe estadístico* (2019).

[40] Galina Lavrentyeva, Sergey Novoselov, Egor Malykh, Alexander Kozlov, Oleg Kudashev, and Vadim Shchemelinin. 2017. Audio Replay Attack Detection with Deep Learning Frameworks.. In *Interspeech*. 82–86.

[41] Gen Li, Zhichao Cao, and Tianxing Li. 2023. EchoAttack: Practical Inaudible Attacks To Smart Earbuds. In *Proceedings of the 21st Annual International Conference on Mobile Systems, Applications and Services*. 383–396.

[42] Zhuohang Li, Cong Shi, Tianfang Zhang, Yi Xie, Jian Liu, Bo Yuan, and Yingying Chen. 2021. Robust Detection of Machine-induced Audio Attacks in Intelligent Audio Systems with Microphone Array. (2021).

[43] Zhuohang Li, Yi Wu, Jian Liu, Yingying Chen, and Bo Yuan. 2020. AdvPulse: Universal, Synchronization-free, and Targeted Audio Adversarial Attacks via Subsecond Perturbations. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 1121–1134.

[44] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2018. Trojaning attack on neural networks. In *25th Annual Network And Distributed System Security Symposium (NDSS 2018)*. Internet Soc.

[45] Yan Meng, Jiachun Li, Matthew Pillari, Arjun Deopujari, Liam Brennan, Hafsah Shamsie, Haojin Zhu, and Yuan Tian. 2022. Your Microphone Array Retains Your Identity: A Robust Voice Liveness Detection System for Smart Speakers. In *31th USENIX Security Symposium (USENIX Security 21)*.

[46] Microsoft. 2021. Microsoft Azure. https://azure.microsoft.com/en-us/.

[47] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*. 506–519.

[48] Yao Qin, Nicholas Carlini, Garrison Cottrell, Ian Goodfellow, and Colin Raffel. 2019. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In *International conference on machine learning*. PMLR, 5231–5240.

[49] 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.

[50] Lea Schönherr, Thorsten Eisenhofer, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. 2020. Imperio: Robust over-the-air adversarial examples for automatic speech recognition systems. In *Annual Computer Security Applications Conference*. 843–855.

[51] scikit fda. 2021. fetch phoneme. skfda.datasets.fetch_phoneme.html.

[52] Cong Shi, Tianfang Zhang, Zhuohang Li, Huy Phan, Tianming Zhao, Yan Wang, Jian Liu, Bo Yuan, and Yingying Chen. 2022. Audio-domain position-independent backdoor attack via unnoticeable triggers. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*. 583–595.

[53] Takeshi Sugawara, Benjamin Cyr, Sara Rampazzi, Daniel Genkin, and Kevin Fu. 2020. Light commands: laser-based audio injection attacks on voice-controllable systems. In *29th USENIX Security Symposium (USENIX Security 20)*. 2631–2648.

[54] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks.

arXiv preprint arXiv:1312.6199 (2013).

[55] Raphael Tang and Jimmy Lin. 2018. Deep residual learning for small-footprint keyword spotting. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 5484–5488.

[56] Siri Team. 2017. Hey siri: An on-device dnn-powered voice trigger for apple's personal assistant. *Apple Machine Learning Journal* 1, 6 (2017).

[57] Susanne Trauzettel-Klosinski, Klaus Dietz, IReST Study Group, et al. 2012. Standardized assessment of reading performance: The new international reading speed texts IReST. *Investigative ophthalmology & visual science* 53, 9 (2012), 5452–5461.

[58] Yuanda Wang, Hanqing Guo, Guangjing Wang, Bocheng Chen, and Qiben Yan. 2023. VSMask: Defending Against Voice Synthesis Attack via Real-Time Predictive Perturbation. *arXiv preprint arXiv:2305.05736* (2023).

[59] Yuanda Wang, Hanqing Guo, and Qiben Yan. 2022. Ghosttalk: Interactive attack on smartphone voice system through power line. *arXiv preprint arXiv:2202.02585* (2022).

[60] Pete Warden. 2018. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209* (2018).

[61] Emily Wenger, Max Bronckers, Christian Cianfarani, Jenna Cryan, Angela Sha, Haitao Zheng, and Ben Y Zhao. 2021. " Hello, It's Me": Deep Learning-based Speech Synthesis Attacks in the Real World. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 235–251.

[62] Qiben Yan, Kehai Liu, Qin Zhou, Hanqing Guo, and Ning Zhang. 2020. Surfingattack: Interactive hidden attack on voice assistants using ultrasonic guided waves. In *Network and Distributed Systems Security (NDSS) Symposium*.

[63] Zhuolin Yang, Bo Li, Pin-Yu Chen, and Dawn Song. 2018. Characterizing audio adversarial examples using temporal dependency. *arXiv preprint arXiv:1809.10875* (2018).

[64] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, Shengzhi Zhang, Heqing Huang, Xiaofeng Wang, and Carl A Gunter. 2018. Commandersong: A systematic approach for practical adversarial voice recognition. In *27th USENIX Security Symposium (USENIX Security 18)*. 49–64.

[65] Tongqing Zhai, Yiming Li, Ziqi Zhang, Baoyuan Wu, Yong Jiang, and Shu-Tao Xia. 2021. Backdoor attack against speaker verification. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2560–2564.

[66] Guoming Zhang, Xiaoyu Ji, Xinfeng Li, Gang Qu, and Wenyuan Xu. 2021. EarArray: Defending against DolphinAttack via Acoustic Attenuation. (2021).

[67] 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*. 103–117.

[68] Nan Zhang, Xianghang Mi, Xuan Feng, XiaoFeng Wang, Yuan Tian, and Feng Qian. 2019. Dangerous skills: Understanding and mitigating security risks of voice-controlled third-party functions on virtual personal assistant systems. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1381–1396.

[69] Baolin Zheng, Peipei Jiang, Qian Wang, Qi Li, Chao Shen, Cong Wang, Yunjie Ge, Qingyang Teng, and Shenyi Zhang. 2021. Black-box Adversarial Attacks on Commercial Speech Platforms with Minimal Information. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*.

APPENDIX

A LIVENESS DETECTION METHODS

The first liveness detection model [38] (CQCC) is the baseline model of the ASVSpoof challenge, which uses constant-Q cepstral coefficients (CQCC) features and Gaussian Mixture Model (GMM) to separate the natural and replayed human speech. The second detection, STC [40] won the ASVSpoof 2017 challenge, which exploits a Light Convolutional Neural Network (LCNN) to perform detection. The third model called Void [5] is a fast and high efficient detection algorithm proposed recently. It considers novel spectrogram features such as spectrogram delay patterns, peak patterns, and Linear Prediction Cepstrum coefficient (LPCC) to achieve state-of-the-art detection accuracy [2].

B USER STUDY SETUP

We recruit 7 volunteers from our institute and 13 volunteers from Amazon Mechanical Turk. Before the experiment, we informed them that their name, voice, and other personal information would not be recorded. We would only release the statistical data about reactions to our attack. For 2 in-person volunteers, we played 6 crafted perturbations at 4 different distances while they were speaking to the smart speaker. For the 5 volunteers from our institute and the 13 volunteers from the MTurk, we asked them to complete the hearing screening before the experiment. Then, we sent them the 6 perturbations and asked them to play the perturbation toward their smart speakers at 4 different distances. To ensure that all the results are valid, we verified that there were no random responses (e.g., "listened" at a far distance but not at a close distance; "recognized" but not "listened"). Every experiment took ~10 minutes because some subjects reported that they would need to play perturbations multiple times before determining an answer.

[2]We skip some more advanced liveness detection approaches such as [28, 42, 45] because they require extra hardware to facilitate the detection.