Devil's Whisper: A General Approach for Physical Adversarial Attacks against Commercial Black-box Speech Recognition Devices

Yuxuan Chen [*1,2,3], Xuejing Yuan [†1,2], Jiangshan Zhang[1,2], Yue Zhao[1,2], Shengzhi Zhang[4], Kai Chen[‡1,2], and XiaoFeng Wang[5]

[1]SKLOIS, Institute of Information Engineering, Chinese Academy of Sciences, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, China
[3]Department of Computer Engineering and Sciences, Florida Institute of Technology, USA
[4]Department of Computer Science, Metropolitan College, Boston University, USA
[5]School of Informatics and Computing, Indiana University Bloomington, USA

Abstract

Recently studies show that adversarial examples (AEs) can pose a serious threat to a "white-box" automatic speech recognition (ASR) system, when its machine-learning model is exposed to the adversary. Less clear is how realistic such a threat would be towards commercial devices, such as Google Home, Cortana, Echo, etc., whose models are not publicly available. Exploiting the learning model behind ASR system in black-box is challenging, due to the presence of complicated preprocessing and feature extraction even before the AEs could reach the model. Our research, however, shows that such a black-box attack is realistic. In the paper, we present *Devil's Whisper*, a general adversarial attack on commercial ASR systems. Our idea is to enhance a simple local model roughly approximating the target black-box platform with a white-box model that is more advanced yet unrelated to the target. We find that these two models can effectively complement each other in predicting the target's behavior, which enables highly transferable and generic attacks on the target. Using a novel optimization technique, we show that a local model built upon just over 1500 queries can be elevated by the open-source Kaldi Aspire Chain Model to effectively exploit commercial devices (Google Assistant, Google Home, Amazon Echo and Microsoft Cortana). For 98% of the target commands of these devices, our approach can generate at least one AE for attacking the target devices[1].

1 Introduction

With the advance of automatic speech recognition (ASR) technologies, intelligent voice control (IVC) devices become increasingly popular. Today, smart speakers like Google Home, Amazon Echo, Apple HomePod are already part of our daily life. Also the availability of ASR services such as Google Cloud Speech-to-Text [10], Amazon Transcribe [4], Microsoft Bing Speech Service [16] and IBM Speech to Text [12] enable their users to conveniently integrate their APIs to control smart devices, conduct long-form audio transcription, text analysis, video analysis and etc. More recently, Amazon introduces Auto SDK [2] that allows drivers to interact with vehicles using voice commands. However, the extensive use of voice for critical system control also brings in security concerns, whose implications have not yet been fully understood.

AE threats to ASR. More specifically, voice is an open channel and therefore the commands received by IVC devices could come from any source. In recent years, researchers have shown that unauthorized voice commands can be injected into wireless signals [28], in the form of noise [19] or even inaudible ultrasound [41], to stealthily gain control of the IVC devices. Recently, attempts have been made to utilize adversarial examples (AEs), which are found to be effective against image processing systems [36], to exploit ASR systems. Particularly, Carlini et al. [20] have successfully attacked DeepSpeech (the open-source ASR model of Mozilla) using AEs, with the full knowledge of model parameters. Yuan et al. proposed CommanderSong [40] that automatically generates AEs embedded into songs to attack open-source Kaldi Aspire Chain Model [14] over-the-air. These approaches demonstrate that the real-world ASR systems are vulnerable in a *white-box model*, when their internal parameters are exposed to the adversary. Less clear is the security risks the commercial ASR systems such as Google Home, Microsoft Cortana, Amazon Echo and Apple Siri are facing. Recently, Taori et al. have made the targeted adversarial attack by treating DeepSpeech as a black-box [37]. However, so far no success has been reported when it comes to generating AEs against the deep learning models behind commercial, close-source ASR systems, up to our knowledge.

Black-box AE attacks on ASR systems are difficult. In addition to the challenge introduced by the lack of information about the target's model and parameters, as also faced by the black-box attacks on image processing [32], an ASR system

*Part of this work was done during the author's visit at IIE, CAS.
†The first two authors contributed equally to this work.
‡Corresponding author. chenkai@iie.ac.cn
[1]Attack demos are available on the website (https://sites.google.com/view/devil-whisper), and the source code can be found on Github https://github.com/RiskySignal/Devil-Whisper-Attack.

tends to be more complicated than an image recognition system, due to its complicated architecture, including feature extraction, acoustic model and language model, and the design for processing a time series of speech data. As evidenced in our study, when directly applying the existing technique to build a substitute on the data labeled by the target [32], we found that about 24 hours training set (require around 5100 oracle queries with each audio around 25 seconds), even with a target-based optimization (Section 4.2.1), only gives us a substitute model with merely 25% transferability against Google Colud Speech-to-Text API command_and_search model (Section 6.4). By comparison, prior research reports that the similar attack on image recognition systems like Google, Amazon and MetaMind APIs using simple datasets like MNIST with 800 queries to achieve a transferability rate over 90% [32].

Devil's Whisper. We demonstrate that a black-box attack on the commercial ASR system and even device is completely feasible. Our attack, called *Devil's Whisper*, can automatically generate audio clips as AEs against commercial ASR systems like Google Cloud Speech-to-Text API. These "hidden" target commands are stealthy for human being but can be recognized by these systems, which can lead to control of commercial IVC devices like Google Home. Our key idea is to use a small number of strategic queries to build a substitute model and further enhance it with an open-source ASR, which helps address the complexity in the target system. More specifically, to construct the substitute, we utilize Text-to-Speech (TTS) API to synthesize commands audio clips, then we enlarge the corpus by tuning audio clips before sending them as queries to the target. This allows us to focus on the types of the data most important to the success of our attack and makes the substitute model more approximate to the target. The substitute model trained over the data is then used in an ensemble learning together with an open-source ASR model (called *base model*). The AEs cross-generated by both models are systematically selected to attack the target.

In our experiment, we build substitute models approximating each of the four black-box speech API services (Google Cloud Speech-to-Text, Microsoft Bing Speech Service, IBM Speech to Text and Amazon Transcribe). Just over 4.6-hour training data (about 1500 queries with each audio about 25 seconds) is needed to ensure successful conversion of nearly 100% target commands into workable AEs[2] when attacking most of the API services. Our AEs can also attack the corresponding black-box IVC devices[3] (Google Assistant, Google Home, Microsoft Cortana and Amazon Echo) over-the-air with 98% of target commands successful. Furthermore, our AEs can be successfully transferred to other black-box platforms, which have no public API services (e.g., Apple Siri). The user study on Amazon Mechanical Turk shows that none of the participants can identify any command from our AEs if they listen to them once.

Contribution. The contributions of this paper are as follows.

• *Physical adversarial attacks against black-box speech recognition devices.* We conduct the first adversarial attack against commercial IVC devices. With no prior knowledge of the targets' machine-learning models and their parameters, our generated AEs can successfully fool the acoustic model and language model utilized in ASR systems after bypassing their feature extraction procedures, which is quite different from attacking black-box image processing systems. Our AEs are stealthy enough to be perceived by human being.

• *New techniques.* We design a novel approach to generate AEs to attack a black-box ASR system. Our idea is to enhance a simple local substitute model roughly approximating the target model of an ASR system with a white-box model that is more advanced yet unrelated to the target. We find that these two models can effectively complement each other, thus enabling highly transferable and generic attacks on the target. Moreover, the substitute model can be trained in an optimized fashion using much less data, allowing much fewer queries to the target system.

2 Background and Related Work

In this section, we provide the background on speech recognition systems and elaborate adversarial examples. Finally we discuss the related work.

2.1 Speech Recognition System

ASR enables machines to understand human voice and greatly changes the way people interact with computing devices. In addition, Text-to-Speech (TTS) services of Google, Microsoft, Amazon, and IBM have been exposed to the public to develop their own voice-assistant applications. Besides these commercial black-box systems, there also exist popular open source ASR platforms such as Kaldi, Mozilla DeepSpeech, etc.

The architecture of a typical speech recognition system includes three main procedures: pre-processing, feature extraction and model-based prediction (including acoustic model and language model). After receiving the raw audio, the pre-processing filters out the frequencies out of the range of human hearing and the segments below certain energy level. Then, ASR system will extract acoustic features from the processed audio for further analysis. Common acoustic feature extraction algorithms include Mel-Frequency Cepstral Coefficients (MFCC) [31], Linear Predictive Coefficient (LPC) [27], Perceptual Linear Predictive (PLP) [26], etc. The acoustic features will be examined according to the pre-trained acoustic model to predict the most possible phonemes. Finally, relying on the language model, ASR system will refine the results using grammar rules, commonly-used words, etc.

[2]In this paper, we consider an AE "workable" or "successful" if it can either 1) be decoded by the target API service (converted into text) as expected in an API attack, or 2) cause the target IVC device to execute the target commands at least twice when playing the AE against the device over-the-air for no more than 30 times. Note that an over-the-air attack on device can be sensitive to environmental factors like volume, distance, device etc., while the attack on APIs is usually stable.

[3]We have contacted the vendors and are waiting for their responses.

2.2 Adversarial Examples

Recently, neural network has been widely used in the prediction algorithms in image classification, speech recognition, autonomous driving and etc. Although it has significantly improved the accuracy of prediction, neural network suffers from adversarial examples (AEs) as first indicated by Szegedy et al. [36]. Formally speaking, one neural network can be defined as $y = F(x)$, which maps the input x to the corresponding output y. Given a specific y', the original input x and the corresponding output y, it is feasible to find such an input x' so that $y' = F(x')$, while x and x' are too close to be distinguished by human. The above example x', together with its prediction y', is considered as target adversarial (TA) attack. Such attacks have potential impact since the prediction results could be manipulated by the adversary. Compared to TA attacks, untargeted adversarial (UTA) attack identifies the input x', which is still close enough to the original input x, but has different output than that of x. Such UTA attack is less powerful since the adversary could only make the target machine misrecognize the input, rather than obtaining the desired output.

AE attacks on black-box image processing models. Recent researches proposed various algorithms to generate targeted AEs towards different image recognition systems [23, 36]. Specifically, there are substantial researches towards compromising black-box image processing systems. Liu et al. [30] proposed the ensemble-training approach to attack Clarifai.com, which is a black-box image classification system. Papernot et al. [32] proved that by training a local model to substitute remote DNN using the returned labels, they can attack Google and Amazon Image Recognition Systems.

2.3 Related Work

Researchers have found that the ASR systems could be exposed to different types of attacks. We classify the existing attacks against ASR systems into four categories as below.

Speech misinterpretation attack. Recently, third-party applications and skills for IVC systems become increasingly popular, while the lack of proper authentication raises security and privacy concerns. Previous studies show third-party applications are facing misinterpretation attacks. Kumar et al. [29] present an empirical analysis of the interpretation errors on Amazon Alexa, and demonstrate the adversary can launch a new type of skill squatting attack. Zhang et al. [42] report a similar attack, which utilizes a malicious skill with the similarly pronounced name to impersonate a benign skill. Zhang et al. [43] developed a linguistic-guided fuzzing tool in an attempt to systematically discover such attacks.

Signal-manipulation based attacks. The adversary can compromise the ASR system by either manipulating the input signal or exploiting the vulnerability of the functionalities in pre-processing. For instance, Kasmi et al. [28] find that by leveraging the intentional electromagnetic interference (IEMI) of the headset cord, voice commands can be injected into the FM signals that will be recovered and understood by the speech recognition systems on the smart phone. Dolphin Attack [41] exploits the hardware vulnerabilities in microphone circuits (served as the recorder for IVC devices), so the completely inaudible ultrasonic signal carrying human speech will be demodulated and interpreted as desired malicious commands by the target IVC device including Apple Siri, Google Now and Amazon Echo.

Obfuscation based attacks. Different from the signal-manipulation based attacks, the obfuscation based attacks explore the way that the feature extraction of ASR systems could be manipulated. Vaidya et al. [38] showed that by inverting MFCC features of the desired commands, they can get malicious audios that can be interpreted by Google Cloud Speech-to-Text API but not human beings. Carlini et al. [19] inverted MFCC features of the malicious commands back to audio and then added additional noise to obtain the attack audio samples uninterpreted to human beings but recognizable to ASR system and IVC device, e.g., Google Cloud Speech-to-Text API and Google Assistant on the smart phone. More recently, Abdullah et al. [18] developed four different perturbations to create the malicious audio samples, based on the fact that the original audio and the revised audio (with perturbations) share similar feature vectors after being transformed by acoustic feature extraction algorithms.

Adversarial example based attacks. For the TA attacks, the attacker can craft an original audio into the adversarial samples, and human beings cannot tell the differences between it and the original audio. These adversarial samples can be misunderstood by the target ASR systems and interpreted as malicious commands. Hidden voice commands [19] proposed to generate such adversarial audio samples against ASR systems with a GMM-based acoustic model. Yuan et al. [40] proposed the CommanderSong attack, which embeds the malicious commands into normal songs. The open-sourced speech recognition platform Kaldi was used as the white-box tool, implementing the gradient descent algorithm on the neural network to craft adversarial audio examples. Carlini et al. [20] generated the adversarial samples against the end-to-end Mozilla DeepSpeech platform [25]. Schönherr et al. [34] showed that they can use psychoacoustic hiding to make imperceptible adversarial samples towards the WSJ model of Kaldi platform. Recently, Qin et al. succeeded in generating the imperceptible and robust AEs to attack Lingvo ASR system in real world [33]. Although all the above attacks showed excellent results on the white-box platforms, whether AEs can attack the black-box ASR systems, especially the commercial IVC devices, is still unknown.

3 Overview

3.1 Motivation

In the era of Internet of Things (IoT), the voice-enabled centralized control devices are becoming more and more popular, e.g., Google Home, Amazon Echo, etc. Various smart home devices, like smart lock, smart light, smart switch can be paired to such "hub", which allows them to be controlled naturally via voice. Moreover, the voice-assistant applications on smartphones or tablets, e.g., Google Assistant, Apple Siri, etc., offer a convenient way for people to use their mobile devices. In this paper, we use IVC devices to refer to all the above mentioned voice-enabled centralized control devices and smartphones or tablets.

An example for the potential security risk to the IVC system is smartphone navigation, which is widely used today to help drive through unfamiliar areas. Previous work [39] shows that the FM radio channel can be controlled by attackers to broadcast their malicious signals. Therefore, if the attackers craft their AE hiding a hostile navigation command and broadcast it on the selected FM radio channel, those who run smartphone navigation while listening to the corresponding FM channel will be impacted. Actually, our experimental results show that "Okay Google, navigate to my home" can stealthily command Google Assistant on smartphones through music and none of the participants in our user study were able to identify the hidden command even after listening to the AE twice. This attack, if successful, will put both drivers and passengers to serious danger. Given the pervasiveness of the commercial IVC systems, it is important to understand whether such an attack is indeed realistic, particularly when the adversary has little information about how such systems work. Our research, therefore, aims at finding the answer.

To hack the commercial IVC devices in the real world successfully, there are generally two requirements for the attacks: (R1) effectiveness (towards device) and (R2) concealing (towards human). Both of the two requirements emphasize the practical aspects of such attacks, that is, to deceive those devices successfully but uninterpretable by human. Unfortunately, most of existing adversarial attacks fail either (R1) [20] or (R2) [19] in some extents. Hence, we concentrate on the research question "whether it is possible to hack those commercial IVC devices (mostly black-box based) in the real world with both (R1) and (R2) satisfied" in this paper.

3.2 Threat Model

Since our target is the commercial IVC devices, they are black-box to us by default. Specifically, we have no knowledge of the internals of the speech recognition systems, e.g., model parameters or hyperparameters. Instead, we assume the corresponding online Speech-to-Text API services, i.e., providing real time decoding results from input audio, are open to public. This assumption is valid for most of the popular IVC devices available on the market, e.g., Google Cloud Speech-to-Text API, Microsoft Bing Speech Service API, etc[4]. Either free or pay as you go, such services are accessible to third party developers. We further assume that for the same platform, the ASR system used to provide online speech API service and that used for the IVC devices are the same or similar[5], e.g., Microsoft Bing Speech Service API and Microsoft Cortana.

Once the attack audio is generated, we assume it will be played by speakers (either dedicated speakers or speakers on radio, TV, smartphone, computer, etc.), which is placed not quite far away (e.g., 5~200 centimeters) from the target IVC devices. For example, the methods proposed in [39] can be used to remotely control the contents played by the radio. Furthermore, we do not have the knowledge of the speakers, or the microphones of the target devices. Once the attack is successful, an indicator could be observed. For instance, the attack audio with the command of "Echo, turn off the light" is successful by observing the corresponding light off.

3.3 Technical Challenges

Currently there are several methods to attack black-box models. First, attackers can probe the black-box model by continuously querying it with different inputs, analyzing the corresponding outputs, and adjusting the inputs by adding perturbations to craft the workable AEs. However, such method normally suffers from the problems of uncertainty in terms of probing process and timing cost, especially for a commercial IVC device whose models are quite complex for approximation. Another method is "transferability" based, i.e., AEs generated on a known Model $\boldsymbol{A}$ are used to attack the target Model $\boldsymbol{B}$, as long as those two models are similar in the aspects of algorithm, training data and model structure. If Model $\boldsymbol{A}$ is hard to find, a local model can be trained based on the algorithm and training data to approximate the target Model $\boldsymbol{B}$, to implement the "transferability". However, since the target Model $\boldsymbol{B}$ is black-box, the similarity is hard to determine and the algorithm as well as the training data may not be available.

4 Approaches

In this section, we present our approach of AE based attacks against the commercial black-box IVC devices. Figure 1 gives the details of our approach. We start by transferability based approach (Step ① in Figure 1), via an enhancement over the existing state-of-the-art work generating AEs against ASR systems. Then we describe the novel approach of "Alternate Models based Generation" (Step ②, ③, and ④ in Figure 1).

4.1 Transferability Based Approach

For the black-box AE based attacks, the knowledge about the internal model is not known, so a straightforward method is to generate AEs based on a white-box model and transfer the AEs to the black-box model. The success of the transferability

[4]However, as the paper is written, we could not find such API service from Apple yet. Communication with Apple Siri developers confirmed that Apple has not released their speech API service to the public. In this work, we proposed an alternative approach to hack such IVC devices without corresponding API service available, like Apple Siri, in Section 6.3.

[5]Based on our experiments, Amazon seems like an exception, which will be discussed in Section 6.2.

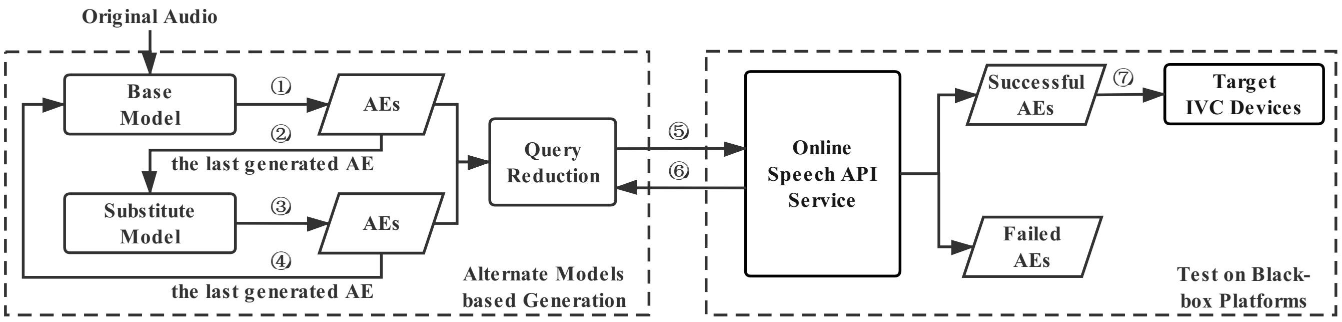

Figure 1: Architecture of general adversarial attack against ASR API service and IVC devices.

based attacks depends on the similarity between the internal structure and parameters of the white-box and black-box models. Recent research demonstrates that the transferability could work on heterogeneous models through the improvement of AE generation algorithm [32].

Initial try. To implement the transferability-based approach, we start by adopting Kaldi ASpIRE Chain Model as the white-box model, and refer to the idea of "pdf-id matching algorithm" proposed in CommanderSong [40] to generate AEs. We make such choices because (i) CommanderSong is the state-of-the-art AE generation work based on white-box model as this paper is written; (ii) the AEs generated in CommanderSong demonstrates transferability to iFLYTEK application—a black-box ASR system—running on smartphone, when played over-the-air; and (iii) the white-box model used in CommanderSong is accessible and popular.

We tested the AEs generated using the above approach on our target black-box ASR systems such as the Google Cloud Speech-to-text API, and find that only few AEs can be partially recognized as "Google", "Hey Google", "phone", etc. The success rate of the transferability on Amazon Transcribe, the API service offered by Amazon, is even lower. This is not surprising, since CommanderSong was not designed to transfer across different systems.

Enhancement. We analyzed the approach proposed in CommanderSong and enhanced it by applying the Momentum based Iterative Fast Gradient Method (MI-FGM) to improve the transferability of the AEs. The momentum method was introduced in [23], which can accumulate a velocity vector in the gradient direction during iterations. In each iteration, the gradient will be saved, and then combined using a decay factor with the previously saved gradients. The work [23] also demonstrated that by combining these gradients together, the gradient direction will be much more stabilized and the transferability of AEs could be enhanced. Furthermore, we added random noise into the samples in each iteration to improve the robustness of the AEs, similar as in CommanderSong [40].

Specifically, let g_{t+1} be the gradient in the $(t+1)th$ iteration, and g_0 be start gradient 0. Let x_t^* denote the AE generated in the $(t)th$ iteration, and x_0^* be original audio. $Clip_\varepsilon$ is a function clipping the values exceeding the pre-defined maximum and works in each iteration. Therefore, x_{t+1}^* within the ε vicinity can be obtained based on MI-FGM as below:

$$g_{t+1} = \mu \cdot g_t + \frac{J(x_t^*, y)}{\|\nabla_x J(x_t^*, y)\|_1} \tag{1}$$

$$x_{t+1}^* = x_t^* + Clip_\varepsilon(\alpha \cdot g_{t+1}) \tag{2}$$

where y is the probability value of the target pdf-id sequence of x_t^*, μ is the decay factor for the momentum, α is the step factor[6], $J(x_t^*, y)$ is the loss function. Intuitively, MI-FGM uses the gradient of the loss function to determine the direction, along which the loss function itself can be minimized. Compared with normal FGM, MI-FGM replaces the current gradient with the accumulated gradients of all previous steps.

Based on our evaluation, the enhanced approach helps to generate a few AEs attacking black-box ASR API services (e.g., Google Cloud Speech-to-Text API) with low success rate and works even poorer on IVC devices (see Section 6.2). The main reason is that the approach to generate AEs mainly depends on the sample's transferability to other black-box systems. Thus, we consider the transferability based approach has one major limitation: the crafted AEs are generated more towards the white-box model. However, the decision boundaries may vary between the white-box model used to generate the AEs and the target black-box model.

4.2 Alternate Models based Generation

Approach overview. First, we propose to build our carefully augmented corpus to train a local model approximating the target black-box model on the desired commands. As the AEs generated from Kaldi ASpIRE Chain Model can be transferred to the target black-box model in some extent, we take it as the large base model, and use it to enhance the small substitute model to generate the AEs. Therefore, the large base model can generate most of the acoustic features of the desired command (Step ① in Figure 1). Furthermore, the last generated AE of the base model will be fed into the substitute

[6]Dong et al. evaluated the success rate of AEs for different decay factors and found 1.0 is the optimal value [23]. Carlini et al. used Adam optimizer to minimize the loss function where the default step factor α is set as 100 [5]. In this paper, we set those two factors based on the above two works.

model (Step ② in Figure 1). Thus, the unique features of the desired command on the target model can be adjusted in a fine-grained manner by the substitute model (Step ③ in Figure 1), since it was trained based on an augmented corpus (details in Section 4.2.1) that can be well recognized by the black-box model. During the AE generation process under each model, we use a small subset of AEs to query the target ASR API service according to our query reduction method (Step ⑤ and Step ⑥ in Figure 1). If none of these AEs works, the last crafted audio (an unsuccessful AE) from the substitute model will be fed to the base model as the input for the next epoch (Step ④ in Figure 1). Finally, we select the effective AEs to compromise the target IVC devices (Step ⑦ in Figure 1). Below we detail such approach.

4.2.1 Local Model Approximation

Training set with limited number of phrases. Generally, the commercial black-box models are trained with significantly large proprietary dataset, and the structure of the neural network can be quite complicated. Therefore, it is extremely difficult to obtain the corpus or even infer the details about neural network. In other words, training a local substitute model completely approximating the target commercial black-box system is almost unpractical. However, since our ultimate goal is to hack the commercial IVC devices and in turn leverage it to compromise the victim's digital life, we are only interested in a limited number of selected phrases such as "open my door", "clear notification", etc. A side product of selecting those phrases is that, based on our experiences, the IVC devices are trained to be quite robust to those phrases, e.g., "open my door" on Amazon Echo, "what is the weather" and "play music" on Microsoft Cortana and Google Assistant. Hence, we just need to train a local model partially approximating the target system on the most frequently used phrases, also the ones we are highly interested in, on IVC devices. We use Text-to-Speech services to generate TTS audio clips for our desired phrases (details in Section 5.3) as the training set for local model approximation.

Training set augment. The above observation inspired us the idea of the local partial approximation. However, the training set has two problems: the number of phrases in the training set is too limited for training; and the robustness of an IVC device to a phrase is unknown. To solve these problems, we augment the training set by tuning the TTS audio clips, i.e., either changing the speech rate of and adding noises to them. Based on our experience, the changing of the speech rate and the noise amplitude is quite unique to different ASR systems, e.g., a specifically tuned audio might be decoded correctly with high confidence by one ASR system, but incorrectly by the other. Hence, we believe that those tuned but still correctly decoded audio clips can help to uniquely characterize different ASR systems, and that training an ASR system with plenty of such audio clips will guide it towards the target ASR system on the desired phrases in the audio clips.

Obviously, not all the tuned audios can still be decoded correctly by the target black-box system. In our research, we assume that the speech recognition mechanisms of the IVC devices are similar to that of the API service provided by the same company[7]. Hence, we query the corresponding online speech API service on them, and filter out those either not correctly decoded, or decoded correctly but with low confidence values. The magnitude of the corpus augmented in this way is not very big, usually 3~6 hours for ten selected phrases, which can be finished in about 1500 queries on the target speech API service.

4.2.2 AE Generation

Generating AEs with base model and substitute model. After the local substitute model is trained based on the augmented dataset, we ensemble it with the large base model for the alternate models generation summarized in Algorithm 1. Specifically, Line 3 and Line 4 are for the AE generation on the large base model and the small substitute model respectively. The AE generation is the same for two models and defined as the function "AEGENERATION" in Line 8~24.

Algorithm 1 Alternate Models Generation Algorithm

```
Require: The original audio $x_0$, the target label $y$, $f_{target}$ is the function to get output from black-box model, the black-box query interval times $T_{interval}$, the maximum allowed epoch $EpochMax$.
Ensure: A set of adversarial example collection $X^*$, all with label $y$ under classifier $f_{target}$.
 1: $x_0^* = x_0$ ; $g_0 = 0$ ; $CurrentEpoch = 0$ ; $T_{interval}^* = T_{interval}$
 2: while $CurrentEpoch < EpochMax$ do
 3:     AEgeneration (Base Model Settings);
 4:     AEgeneration (Substitute Model Settings);
 5:     $CurrentEpoch++$;
 6: end while
 7: return $X^*$
 8: function AEGENERATION(Model Settings)
 9:     Reset $T_{interval}^* = T_{interval}$;
10:     for each $t \in [0, T-1]$ do
11:         Take $x_t^*$ for current model $f$ and get the gradient;
12:         Update $g_{t+1}$ by Eq. 1;
13:         Update $x_{t+1}^*$ by Eq. 2;
14:         if $t \mod T_{interval}^* = 0$ then
15:             Input $x_{t+1}^*$ to $f_{target}$ and get $f_{target}(x_{t+1}^*)$;
16:             if $f_{target}(x_{t+1}^*)$ match $y$ then
17:                 Put $x_{t+1}^*$ into $X^*$;
18:             else
19:                 Update $T_{interval}^*$ by Eq. 3;
20:             end if
21:         end if
22:     end for
23:     Set $x_0^* = x_T^*$;
24: end function
```

[7]Although previous studies [29, 42] show that it is possible to recover Speech-to-Text functionality from some IVC devices like Echo, their approaches cannot obtain the confidence values for the decoded results, which are required in our approach.

In each iteration of the for loop starting at Line 10, the gradient is updated in line 12 based on Eq. 1 and the audio sample is crafted in line 13 based on Eq. 2. To successfully attack the target black-box model, we need to query the target speech API service and validate whether the decoded result of the crafted audio sample is as expected or not. An intuitive way is to submit the sample generated in each iteration, so any potential effective AE will not be ignored. However, such method will incur a significant amount of queries sent to the API service, which could be costly and at the same time suspicious. Therefore, we implement the query reduction algorithm (will be detailed at the end of this subsection), which aims to reduces the number of queries to the target black-box system. At Line 1, we set the $T_{interval}$ as the number of iterations between two consecutive queries to the target black-box system, and then at Line 19, it is updated based on Eq. 3, according to the recognition results from the target black-box system.

If after T iterations, effective AE is still not generated (i.e., Line 16 always returns false), we assume the transferability based attack does not work well towards the target black-box system. We will use the output x^* from the last iteration as the input to the local substitute model, then use the same gradient algorithm to craft the adversarial sample under the substitute mode settings. If after we reach the T iterations and the local substitute model approximation approach still does not succeed in generating the AE for the target command, we go back to Line 2 to restart the whole algorithm. The $EpochMax$ parameter can restrain the number of total alternations. For Line 16~17, we will not break the "AEGENERATION" function even if the Line 16 returns "True" and an effective AE is crafted towards the target ASR API service. This is because the successful sample to attack the target ASR API service does not necessarily indicate the success towards IVC devices. Therefore, instead of breaking the function, once a successful AE is found, we save it towards the target ASR API service. Finally, we can return a set X^*, where we preserve all potential effective AEs towards target IVC devices.

Efficient query of the black-box API. Intuitively, we can query the black-box server after a few iterations, instead of every iteration. We compare the decoded transcript of the current sample from the black-box model with the desired transcript, and use it as a reference to determine when the next sample should be sent to the black-box server. Suppose we set the number of iterations between two queries to the target black-box model as $T_{interval}$, and there are s words from the decoded transcript of AE that match the desired commands (e.g., $s = 2$ if "the door" is decoded from the current iteration for the desired command "open the door"). Then $T^*_{interval}$ should be updated by Eq. 3.

$$T^*_{interval} = \lfloor T_{interval} \times \frac{1}{s+1} \rfloor \tag{3}$$

Actually when examining the word match, we check the phonetic similarity of the words, rather than character-by-character match, since the language model of the speech recognition systems will refine the phonetic-similar words based on semantics. Hence, we applied SoundEx [35], a tool to encode homophones with the same representation even though they have minor differences in spelling. Thus, s will be updated by comparing the SoundEx code of the decoded command and the target command. For example, "inter" is encoded by SoundEx as "$I536$" and "pay Internet fee" is encoded as "$P000\ I536\ F000$". We consider one match (the code "$I536$") when comparing such decoded output and desired command, so s will be set as 1 in this case.

4.2.3 Understanding the Attack

Although our approach works effectively in a black-box attack, which will be demonstrated in our experiments (Section 6.2), theoretic analysis of the technique are nontrivial, just like the attempt to interpret adversarial learning in general. Following we provide high-level intuition behind our approach through an example.

At a high level, our approach is based upon the observation that different ASR systems have some similarity in their classification models, which allows us to utilize a white-box, well-trained base model to move an instance towards the target model's decision boundary, though it is likely different from that of the white-box model. This difference is further addressed using the substitute that fine-tunes the instance based upon the features of the target, including those related to its decision boundary. In this way, we can utilize both the information learnt from the target by the substitute and that shared between the base model and the target to build a successful AE.

For example, consider an attack on the Alexa Transcribe API using the approach proposed in Section 4.2. The target command is "clear notification". According to the experimental results, we found that the generation process (the base model->substitute model->base model) helped find the results that came closer to the target (recognized as "I don't" by Alexa Transcribe API). These results were then further adjusted by the substitute model towards the target. They became "notification" in the $10th$~$30th$ iterations, and were recognized as "clear notification" in the $48th$~$60th$ iterations. We believe that the transformation from "I don't" to "clear notification" is attributed to the fact that the substitute is trained to simulate the behavior of the Alexa Transcribe API on "clear notification".

5 Implementation

5.1 Target IVC Devices and ASR Systems

Since we are developing a general approach generating AEs against commercial black-box IVC devices, we plan to examine the AEs on most of the popular IVC devices currently available on the market. In particular, we consider the speech recognition devices/systems from Google, Microsoft, Amazon, Apple, and IBM into the following three categories. First, we can find the ASR API services associated with the corresponding IVC devices, e.g.,Google Assistant and Google Cloud Speech-to-Text API[8] (Category 1). Second, IVC device is available, but ASR API is not, e.g., Apple Siri (Category 2). Last, ASR API is available, but IVC device is not, e.g., IBM Speech to Text API (Category 3).

Regarding Category 2, since there does not exist online ASR API service required by local model approximation, we attack such IVC devices mainly via transferability as in Section 4.1. As for Category 3, since we cannot find the IVC device of IBM, we simulate such scenario by playing the AE, recording it and then using the ASR API service to decode the recorded audio as in Section 6.3. All the available ASR API services return the decoded transcripts and the corresponding confidence level for the decoding.

5.2 Phrase Selection

Since the aim of our approach is to attack the commercial IVC devices like Google Home, we only focused on the specific commands frequently used on these devices, e.g., "turn off the light", "navigate to my home", "call my wife", "open YouTube", "turn on the WeMo Insight", etc. For each target model, we selected 10 such commands and further appended the default wake-up words for different systems (Google Home, Amazon Echo and Microsoft Cortana) before each of them. For the IBM Speech to Text API without commercial IVC devices available, we utilized daily conversation sentences. The full list of the phrases used on the target platforms are presented by Table 10 and Table 11 in Appendix G.

5.3 Local Model Approximation

Model selection. In our experiment, we chose the Mini Librispeech model[9] as the substitute model to approximate the target models. Specifically, we used the default architecture and hyper-parameters of Mini Librispeech to train all four substitute models in our paper. These models were found to be highly effective in our study (Section 6.2). On the other hand, we acknowledge that even better performance could be achieved by tuning model parameters, a mostly manual and time-consuming procedure. So, our attack should only be viewed as a lower bound for the security threats these commercial systems are facing.

Corpus preparation. To enrich our corpus, we use 5 TTS (Text-to-Speech) services to synthesize the desired command audio clips, i.e., Google TTS [11], Alexa TTS [3], Bing TTS [6], IBM TTS [13] and an unnamed TTS [9], with 14 speakers in total including 6 males and 8 females. After using the above TTS services to generate the desired command audio clips, we enrich it by adding background noise or twisting the audio. For the former, we add white noise to the original audio, and set the amplitude of the added white noise to be α. For the latter, we twist the original audio by changing its speech rate either slower or faster. We define the twist-rate as β ($\beta = original\_audio\_duration/twisted\_audio\_duration$). Finally, we use the target black-box model to recognize the tuned audio and filter it based on the correctness and the confidence level of the decoded results. The values of α, β and the size of the corpus after filtering are shown in Table 5 in Appendix A.

We constructed the training corpus by combining the tuned TTS audio clips (generated from the queries on the target model) and the supplemental corpus from Mini Librispeech. This is because the tuned TTS audio clips alone would cause the substitute model to overfit to the set of commands used in the queries (in the tuned TTS audio clips). As a result, the AEs found from the less generalized substitute model can be less effective at attacking the target models. On the other hand, solely relying on the supplemental corpus is not effective either, since the substitute trained without the information from the target will behave very differently from the target, as confirmed by our experiment (alternate models based generation without approximation) in Section 6.4.

Furthermore, we evaluate the impact of different sizes of supplemental corpus on Microsoft Bing Speech Service API in Appendix B, and the results show that 3~40 hours size of the supplemental corpora are all effective for our approach, while with 1 hour supplemental data cannot generate AEs for all of the target commands. For the four substitute models of the target black-box platforms, we use the default Mini LibriSpeech corpus (7.35 hours) as the supplemental corpus.

Training the substitute model. To train the substitute model, we need to label the audio clips in the training corpus. Also, as mentioned in Section 4.2, retrieving the pdf-id sequence of the target commands is critical in our attack. However, we found that some words (such as Cortana, WiFi, Bluetooth, YouTube, WeMo, TV, etc.) are not included in the dictionaries of the Mini Librispeech model and the ASpIRE Chain model, so we cannot directly label these words and get the pdf-id sequences of the corresponding commands. Simply extending the vocabulary of the language models [15] requires the entire language models be retrained. To address this problem, we leveraged some linguistically similar phrases, based upon the prior research [29, 42], to label those undocumented ones[10], which allows us to identify the pdf-id sequences of their commands and further generate their AEs.

[8]There are four models in Google Cloud Speech-to-Text API, e.g., "phone_call model", "video model", "command_and_search model" and "default model". In detail, "phone_call model" is used to translate the recorded audio from phone call; "command_and_search model" is used for voice command and short speech searching; "video model" is used for the video; "default model" is not designed for a specific scenario. We use the command_and_search model to label our corpus since our corpus are more suitable for voice command and search application.

[9]Both Mini Librispeech and Kaldi ASpIRE (used as the base model) use chain model, and Mini Librispeech is easy to implement.

[10]The phrases like "Cort tana", "why fi", "blue tooth", "you too boo", "we mow" and "T V" are used to replace "Cortana", "WiFi", "Bluetooth", "YouTube", "WeMo" and "TV", respectively.

6 Evaluation

6.1 Experiment Setup

Hardware. We conduct the experiment on the sever equipped with four Nvidia Tesla K40m GPUs and 2 x 10 core Intel Xeon E5-2650 2.30GHz processors, with 131 Gigabytes of RAM and 1 Terabyte Hard Drive. We use a laptop (Lenovo W541/Dell XPS 15/ASUS P453U) and a phone (iPhone SE/iPhone 8) connected to a speaker (JBL clip 2/3 portable speaker) to play out AEs. The target IVC devices are Google Home Mini, Amazon Echo 1st Gen and voice assistants on phones (Google Assistant App on Samsung C7100/iPhone SE and Microsoft Cortana App on Samsung C7100/iPhone 8)[11]. The transferability of the AEs on Apple Siri is tested on iPhone 8 and iPhone XR. The AEs on IBM WAA tests are recorded by Huawei P30.

The original audio. Similar to CommanderSong, our attack utilizes songs as the carrier for the AE produced. Specifically, we used the dataset released by the CommanderSong project [8], which contains 5 songs in each of the soft, popular, rock and rap categories. Among them, we selected the songs in the soft and popular categories, which are less noisy, allowing the integrated perturbations more likely to overwhelm the background music and be decoded correctly by the target IVC devices. To further evaluate the 10 songs, we utilized two commands "Okay Google, navigate to my home" and "Hey Cortana, turn off the bedroom light", and ran our approach to embed the commands into the songs, against the speech recognition APIs provided by Google and Microsoft Bing. The results show that all the 10 songs can serve as carriers for the commands to ensure their recognition by the APIs. However, when listening to these AEs, we found that four instances using soft songs and one using a popular song were less stealthy than the other 5 manipulated songs and therefore selected the latter for all follow-up experiments. Our experimental results show that for each target command of each target platform, there are at least 2 music clips across these 5 songs that can be crafted as effective and stealthy AEs. Further we studied the songs more likely to be good candidates for covering different commands (Section 7.1).

Besides the songs, we also tried other types of sounds as our carriers for malicious commands in the experiments, e.g., ambulance siren sound, train passing sound, etc. We found songs perform best in both effectiveness and stealthiness among those sounds. Therefore, we choose the songs as our carrier.

6.2 Effectiveness

We evaluate the effectiveness of AEs generated by transferability based approach (TBA) and those generated by alternate models generation approach (AGA) on the commercial Speech API services and IVC devices. The target commands for every black-box platform are listed in Table 10 and Table 11 in Appendix G. Similar to the existing works [20, 34, 40], we use *SNR*[12] to measure the distortion of AE to the original song.

[11] In Table 11, we elaborate the hardware used for each test.

Speech-to-Text API services attack. We feed our adversarial examples (AEs) directly into the corresponding API services, and observe the results. For the four models of Google Cloud Speech-to-Text API (Section 5), we show the results of "phone_call model" and "command_and_search model", since according to our tests the former is similar to "video model" and the latter is similar to "default model".

When attacking Speech-to-Text API, since we do not need to wake up the IVC devices, we consider the AE successfully attacks the target if the returned transcript matches the desired command. The results are shown in Table 1, with the *SNR* being the average of all commands on each black-box platform (The result of each individual command can be found in Table 10 in Appendix G). Specifically, the effectiveness of our approach is evaluated using the success rate of command (SRoC), that is, *the number of successful commands* vs. *the total number of the commands* evaluated on a target service. Here a successful command is the one for which we can generate at least one workable AE using our approach. The results show that the AEs produced by TBA work well on Google phone_call model with 100% SRoC, but fail on Google command_and_search model and Amazon Transcribe. Also the AEs generated by AGA achieve an SRoC of 100% for all Speech-to-Text API Services except Amazon Transcribe.

Table 1: The overall SRoC results on API services.

Black -box	**Google**		**Micros-oft Bing**	**Amazon Transcribe**	**IBM STT**
	Phone	**Command**			
TBA	10/10	0/10	2/10	1/10	3/10
AGA	10/10	10/10	10/10	4/10	10/10
SNR (dB)	11.97	9.39	13.36	11.21	10.06

Note: (1) "Phone" and "Command" represent the "phone_call model", "command_and_search model" of Google Cloud Speech-to-Text API, respectively. (2) "Microsoft Bing" represents the Microsoft Bing Speech Service API. (3) "IBM STT" represents the IBM Speech to Text API. (4) The results were all based on the tests conducted in October 2019.

As for Amazon Transcribe API service, we only crafted successful AEs on 4 out of 10 target commands using AGA method (details in Table 10 in Appendix G). We then performed more tests on Amazon Transcribe API and found that the API service cannot even recognize some plain TTS audio clips for the target commands correctly. In contrast, these commands can always be recognized by Amazon Echo. There can be reasons for such difference. First, different models could be used by Amazon Transcribe API and Echo device. Second, the developers of Amazon Echo may set lower threshold to identify voice commands, thus it is more sensitive to the voice

[12] *SNR*, defined as the ratio of the original signal power to the noise power, can be expressed as follows: $SNR(dB) = 10\, log_{10}\, (P_{x(t)}/P_{\delta(t)})$, where $P_{x(t)}$ represents the average power of the original signal and $P_{\delta(t)}$ represents the average power of the distortion. It can be seen that a larger *SNR* value indicates a smaller perturbation.

commands when used physically.

IVC devices attack. We selected the AEs that can successfully attack the API service with high confidence score (≥ 0.6) to attack the IVC devices. Specifically, since the AEs working poorly on Amazon Transcribe API are not necessarily working poorly on Amazon Echo as we identified before, we decide to test the AEs on Amazon Echo directly, even if they failed on Amazon Transcribe API. In our experiment, if the devices respond to the played AE in the same way as the regular voice command from human being, we consider the AE for this command successful.

As shown in Table 2, the average SRoC of TBA is 26%. In contrast, the average SRoC of AGA over all IVC devices can be improved to 98%, which shows the proposed approach is very effective in attacking real-world IVC devices. Based on our evaluation, we find that for most of the black-box models, we can always find the AEs that can successfully attack their corresponding IVC devices from the ones that have fooled the ASR API services. However, Amazon Transcribe API and Amazon Echo are the exception. We find that although attacking Amazon Transcribe API is difficult, we can always generate AEs with 100% SRoC for the 10 target commands to attack Amazon Echo. The full list of successful commands on different IVC devices are shown in Table 11 in Appendix G. As we can see, some of those commands can cause safety or privacy issues, e.g., "Okay Google, navigate to my home", "Okay Google, take a picture", "Echo, open my door", etc.

Table 2: The overall SRoC results on IVC devices.

Black -box	Google Assistant	Google Home	Microsoft Cortana	Amazon Echo	IBM WAA
TBA	4/10	4/10	2/10	0/10	3/10
AGA	10/10	9/10	10/10	10/10	10/10
SNR (dB)	9.03	8.81	10.55	12.10	7.86

Note: (1) "WAA" is used to represent "Wav-Air-API" attack. (2) The results were all based on the tests conducted in October 2019.

We used a digital sound level meter "SMART SENSOR AS824" to measure the volume of AEs. The background noise was about 50 dB, and the played audios were about 65~75 dB, compared to some special cases of the sound level presented in [7, 17], e.g., talking at 3 feet (65 dB), living room music (76 dB). We also conducted experiments to test our AEs in realistic distance. For example, the AE with the command "Echo, turn off the light" can successfully attack Echo as far as 200 centimeters away, and the AE with the command "Hey Cortana, open the website" can successfully attack Microsoft Cortana as far as 50 centimeters away.

Robustness of the attack. To evaluate the robustness of our attack, we define the success rate of AE (SRoA) as the ratio of *the number of successful tests* to *the total number of tests* if an AE has been repeatedly played. Table 11 shows SRoA measured over 30 tests for each target command. The results show 76% (38/50) of the commands have SRoAs over 1/3, showing that our attack is quite robust.

6.3 Attacking Other Platforms

Over-the-air attack against IBM Speech to Text API. As stated in Section 5.1, we use "Wav-Air-API" (WAA) to simulate the IVC device of IBM. The results are shown in Table 2. Overall, such WAA attack demonstrates similar performance as other IVC devices, which further indicates the effectiveness and generality of our proposed approach.

AEs attack against Apple Siri. Since there is no online speech-to-text API service available from Apple, we tried two methods to attack Apple Siri: (1) we generate AEs directly using the transferability based approach; (2) we "borrow" the AEs demonstrating good performance on the other IVC devices. As shown in Table 9 in Appendix F, only the command "What is the weather?" generated from TBA can attack Apple Siri successfully. For the other commands, we rely on the help from AEs generated from AGA for other IVC devices[13]. From Table 9, we find all the seven AEs can successfully attack Siri, which demonstrates the transferability of AGA[14].

6.4 Evaluation of Possibly Simple Approaches

Local model approximation with a larger corpus. Apparently, if the local model is trained by a larger corpus of tuned TTS audio clips, it could approximate the target black-box model better (Certainly a larger corpus means a larger amount of queries to the online API service, which could be suspicious.). Below we describe a preliminary evaluation of the AEs generated by such local model.

We choose Google command_and_search model as our target system. Then we pick up four commands that the AEs generated by our approach can be decoded by Google command_and_search model with 100% SRoC. The details of the commands are shown in Table 7 in Appendix C. As in Section 4.2.1, we use TTS to generate regular speech of those commands, and extend the corpus by tuning TTS audio clips. Finally the corpus is filtered out by the labeling from Google command_and_search model with the same confidence level as that in our approach. Hence, we obtain a corpus of about 23.86 hours (5100 oracle queries), almost 5.17 times larger than that used in our approach. After the local model is trained with the larger corpus, we use the "MI_FGM" algorithm to generate AEs and evaluate them on the target.

The results show only one command "OK Google, turn off the light" succeeds on Google command_and_search model, but still fails on Google Home. The other commands do not have any successful AEs generated for Google command_and_search model and Google Home/Assistant. Based on the results of the preliminary testing, even if the adversary could afford the cost of preparing larger corpus and a larger amount of queries, the AEs generated from such simplified

[13]When testing AEs from the other IVC devices on Apple Siri, we ignore the wake up words, e.g., "OK Google, play music" should be truncated to "Play music".

[14]The test was conducted in January 2019. However, we found that the AEs cannot work on Apple Siri since July 2019 (details in Section 7.3).

Table 3: Results of the comparison tests with different approaches.

Black -box	**Target command**	**Plain TTS**	**Command -erSong**	**Original song + TTS**				**Devil's Whisper**	
				SRoA of Group1	**SNR (dB)**	**SRoA of Group2**	**SNR (dB)**	**SRoA**	**SNR (dB)**
Google Assistant	Okay Google, take a picture.	10/10	0/10	6/10	7.15	9/10	6.45	5/10	6.45
	Okay Google, navigate to my home.	10/10	0/10	3/10	4.08	0/10	11.98	4/10	12.02
Google Home	Okay Google, turn off the light.	10/10	0/10	6/10	4.05	0/10	10.75	7/10	10.73
	Okay Google, play music.	10/10	0/10	2/10	4.53	0/10	11.63	3/10	11.61
Microsoft Cortana	Hey Cortana, open the website.	10/10	0/10	6/10	0.21	0/10	12.01	8/10	12.03
	Hey Cortana, make it warmer.	10/10	0/10	9/10	3.38	0/10	9.38	9/10	9.34
Amazon Echo	Echo, turn off the computer.	10/10	0/10	6/10	3.39	0/10	14.29	7/10	14.28
	Echo, call my wife.	10/10	0/10	4/10	-0.78	0/10	10.78	3/10	10.88

Note: (1) The success rate "A/B" indicates that there are A tests success to trigger the command on the black-box platforms in B tests. (2) The results were all based on the tests conducted in July 2019. (3) Hardware settings: we used ASUS P453U as the audio source and JBL Clip 2 as the speaker for all test cases. Google Assistant and Microsoft Cortana were tested on Samsung C7100. Amazon Echo and Google Home were tested on Echo 1st gen and Google Home Mini. Volume of AEs is about 70 *dB* and distance ranges 5~15 centimeters.

approach is not as effective as our proposed alternate models based generation with approximation approach.

Alternate models based generation without approximation. Another intuitive approach is based on the assumption that if one AE works on multiple models, it is highly possible that it works on the target model, without the need to approximate the target. We kept the ASpIRE Chain model as the base model, and trained the Mini Librispeech model without the tuned TTS corpus. Specifically, we selected four target commands from Table 10 in Appendix G to attack Google command_and_search model and Google Assistant/Home. We ran the proposed alternate models based generation approach based on those two models (ASpIRE Chain model and Mini Librispeech model) to craft AEs. However, as shown in Table 8 in Appendix D, only one out of four commands works on Google command_and_search model and Google Assistant, while all the four commands fail on Google Home.

Other straightforward approaches. We conducted experiments to compare our Devil's Whisper attack with other straightforward approaches, i.e., "Plain TTS", the AEs of CommanderSong, the "Original song + TTS". Specifically, we selected eight target commands frequently used on four IVC devices, as shown in Table 3. Each command was covered by the same original song for different cases. Particularly, samples in "Original song + TTS" were generated by combining the song and the TTS command with Adobe Audition software [1]. Note that for such a simple combination, whether the injected command can be clearly heard and therefore interpreted by the IVC depends heavily on the strength of the signal from the song (in terms of its volume) vs. that of the command in the TTS audio. To evaluate the perturbation of the TTS audio on the original song, we calculated the *SNR* of the combinations by treating the TTS audio (the command) as noise and the song as signal.

The results of our experiment are shown in Table 3. Overall, the AEs from the Devil's Whisper attack can effectively attack the target IVC devices using those commands. Without any surprise, the "Plain TTS" audios triggered the devices to act on those commands each time. The AEs produced by CommanderSong, which is not designed for the black-box attack, failed to achieve a single success on these devices. As stated in Section 4.1 under "Initial try", sometimes CommanderSong AEs can be partially recognized as "Hey Google", thus waking up Google Assistant/Home. Occasionally, part of the commands can be recognized by a woken Google Assistant or a Microsoft Cortana. However, none of the AEs (with the *SNR* between 2 and 14) could cause the IVC devices to act on the injected commands.

To produce the samples of "Original song + TTS" case, we set the volume of each TTS audio clip (the command) to the same level as in "Plain TTS" case, while adjusting the volume of the song as follows: (1) to achieve a similar success rate (SRoA) as our attack AEs (see the column in Table 3 under Group 1), and (2) to keep a similar *SNR* level as the AEs (Group 2). As we can see from the table, under a similar SRoA, all except one combined audio clips (Group 1) have much lower *SNR* levels compared with our AEs, indicating that the commands they include are likely to be much more perceivable and thus much less stealthy, which has been confirmed in our user study (see Section 6.5, Table 4). The only exception is featured by a similar *SNR* as our AE. When tuning the *SNR* to a level of our AEs, we can see that the SRoA of most samples (all except one) go down to zero (Group 2). Also interestingly, even though the SRoA of our AE apparently is below that of the "Original song + TTS" audio clip for the command "Ok Google, take a picture", we found that 60% of human users in our study could identify the hidden command, compared with 0% for our AE.

6.5 Human Perception.

SNR describes the relative strengths between signal and noise, which is traditionally used to measure the perturbation to data (e.g., an image) [22]. Naturally, it can also model the distortion to the song caused by an AE (with the song being signal and the command being noise), and therefore gives an intuitive and rough estimate of the AE's stealthiness: the smaller *SNR* is, the larger distortion to the song is imposed, so the

Table 4: Results of the human perception evaluation on Devil's Whisper and original song combined with TTS command.

Black-box	Approach	Normal (%)	Noise (%)	Talking (%)	Once-recognize (%)	Twice-recognize (%)
Google Assistant	Devil's Whisper	14.3	74.3	11.4	0	0
	Song & TTS	7.1	2.9	90	37.1	64.3
Google Home	Devil's Whisper	14.3	65.7	20	0	1.4
	Song & TTS	1.4	2.9	95.7	61.4	77.1
Microsoft Cortana	Devil's Whisper	15.7	64.3	20	0	1.4
	Song & TTS	2.9	1.4	95.7	31.4	54.3
Amazon Echo	Devil's Whisper	25.7	61.4	12.9	0	2.86
	Song & TTS	0	5.7	94.3	41.4	62.9
Average	Devil's Whisper	17.5	66.4	16.1	0	1.4
	Song & TTS	2.9	3.2	93.9	42.9	64.7

Note: (1) "Song & TTS" is used for the abbreviation of "Original song + TTS". (2) "Once-recognize" and "Twice-recognize" represent that the users can recognize over half of the hidden command when they listen to the AEs for once and twice, respectively.

more likely the source of the distortion – a hidden command can be perceived by human. This is largely in line with the findings from our user study as below. However, the metric will be less accurate, for example, when the distortion fits well in other background noise, becoming less easy to notice, even when the *SNR* is low. In general, human perception of hidden commands is complicated, depending on individuals' experience, the context of a conversation, etc. Finding an accurate measurement is still an open question. Therefore, we conducted a survey[15] on Amazon Mechanical Turk to evaluate human perception of the AEs generated by the Devil's Whisper attack, and compare the result with that of "Original song + TTS". Specifically, we used the audio clips in Group 1 since they have the similar SRoA as Devil's Whisper when attacking the target models.

The results of this user study are shown in Table 4. Here, the column "Normal" shows the percentage of the users who consider a sample to be normal music, and the column "Noise" gives the percentage of the users who find noise in songs. The column "Once-recognize" and the column "Twice-recognize" describe the percentages of the users able to recognize over half of the hidden command words[16] after listening to the audio once or twice, respectively. As we can see from the table, 16.1% participants think that somebody is talking in the background when they listen to Devil's Whisper, but nobody could recognize any command when an AE was played to them. By comparison, over 93% of the participants think that someone is talking when listening to the audio clips in "Original song + TTS", and nearly 42.9% of them recognizes over half of the command words first time when they listened. Even if the participants were exposed to the same AEs for the second time, only 1.4% of them could tell over 50% words in the target commands in the Devil's Whisper attack, while the ratio goes up to 64.7% in "Original song + TTS". This indicates that, the samples from "Original song + TTS" are much more perceptive to users. Furthermore, by analyzing the *SNR* in Table 3 and human perception results, we found that *SNR* was largely in line with human perception but not always (see the exception described in Section 6.2). The details of the survey study are presented in Appendix E.

[15]This survey will not cause any potential risks to the participants, such as psychological, social, legal, physical, etc. We do not ask any confidential information about the participants in the questionnaires. The IRB Exempt certificates were obtained from our institutes.

[16]We assume that 50% of the words in the command would be enough to raise user's attention.

7 Discussion

7.1 Selection of Songs

In order to find what types of songs are good candidates for our attack in terms of both effectiveness and stealthiness, we conducted a preliminary evaluation using all the 20 songs from CommanderSong, including the 5 rock and 5 rap songs that we did not use in our attack (see Section 6.1). The target commands were the same as those used in the previous experiments for Google and Microsoft Bing. In the evaluation, a song is considered to be suitable for AE generation if it helped produce effective AEs (for both commands) in the first epoch (based model -> substitute model). Note that an effective AE is stealthy, as determined by humans (authors and other group members in our research) who listened to it. Through the evaluation, we classified the 20 songs into three categories: (1) easy to generate successful AEs but noticeable to human (2) easy to generate successful AEs and unnoticeable to human (3) hard to generate successful AEs. Obviously, the songs in the second category are good candidates for our attack. These songs are characterized by the similarity in the energy distributions of their spectra, as discovered in our research. We here present an example to show its spectral power distribution in Figure 2.

Further we looked into the Top 100 Billboard songs in the week of 11/04/2018, embedding the commands "Hey Cortana, what is the weather?" (Command A) and "Hey Cortana, make it warmer" (Command B) into each of them, in an attempt to attack the Microsoft Bing Speech Service API, and "Ok Google, turn off the light" (Command C) and "Ok Google, navigate to my home" (Command D) to attack the Google Cloud Speech-to-Text API. During the attack, we selected the segment between the 60*th* second to the 63*th* second (roughly

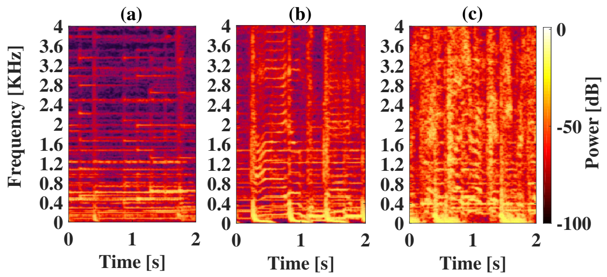

Figure 2: Representative original song spectrum (a) Type 1: easy to be generated as successful AEs and perceived by human (b) Type 2: easy to be generated as successful AEs but difficult to be perceived by human (c) Type 3: hard to be generated as successful AEs.

the middle of the songs) for each song as the carrier for the commands. For Command A, B, C, D, we successfully generated AEs based on 59, 56, 58 and 60 songs, respectively. Then we asked 20 students to listen to the successful AEs generated and reported the commands that could be recognized. In the end, again, we classified all the 100 songs into these three categories. Most of their frequencies and energy distributions were found to be in consistent with those discovered in the 20 songs (see the example in Figure 2). This indicates that indeed a more systematic way to select ideal carriers for the attack is possible, which will be explored in the future research.

7.2 Discussion on Possible Defense

We discuss three potential defense mechanisms to mitigate our Devil's Whisper attack.

Audio downsampling. Audio downsampling was proposed in CommanderSong [40] to effectively mitigate the AEs. Even though the audio can be recorded in different formats (such as m4a, mp3, wav) at different sampling rates (e.g. 8000Hz, 16000Hz, 48000Hz), we can always first downsample it to a lower sampling rate and upsample it to the sampling rate that is accepted by the target black-box model. During such downsampling/upsampling process, the added adversarial perturbations may be mitigated, which makes the AEs fail to be recognized by the target black-box model. For instance, we choose the recorded audios, which can succeed in WAA attack on IBM Speech to Text API. Then they are downsampled to 5600Hz, and upsampled to 8000Hz, which are sent to IBM Speech to Text API. Only 20% of them can be recognized as the target commands. When first downsampled to 5200Hz and then upsampled to 8000Hz, none of them can succeed. In contrast, the regular recorded human voice and TTS audio clips can still be recognized correctly even after such downsampling/upsampling. Hence, audio downsampling could be one effective way in detecting speech AEs. However, if an attacker know the dawnsampling/upsampling rates of the defense, he could train an AE robust against it.

Signal smoothing. Since the effectiveness of our AEs is highly dependent on the carefully added perturbations by gradient algorithm, we can conduct local signal smoothing towards AEs to weaken the perturbations. Specifically, for a piece of audio x, we can replace the sample x_i with the more smooth value according to its local reference sequence, i.e. the average value of the k samples before and after x_i. Hence, the added perturbations may be mitigated by this method.

Audio source identification. Audio source identification aims to identify the source of the audio, e.g., from an electronic speaker or human. Such defence is based on the assumption that the legitimate voice commands should only come from human rather than an electronic speaker. Therefore, if the audio is detected not from human, the audio signal will be simply ignored. Previous works [21,24] show that they can identify the audio source by either examining the electromagnetic wave from the audio or training a model to label the audio. Such defence mechanism could work for most of the existing speech AEs that require a speaker to play. However, the attacker could play the samples over a long range, which might evade the detection.

7.3 Limitations

It is known that AEs are rather sensitive to the change made on the deep neural network models behind ASRs: even a small update could cause a successful AE to stop working. This is also true for our approach. For instance, the previous workable AEs (in January 2019) cannot work effectively towards Apple Siri since July 2019 (See Section 6.3)[17]. A potential solution is to fine-tune the existing model in the hope of capturing the impact of the change, which will be studied in the future research. In addition, the practical attack against IVC devices is sensitive to various environmental factors, such as the volume when playing AEs, the distance between the speaker and the IVC device, even the brand of the speakers, etc., which may significantly affect the SRoA. Hence, how to improve the robustness of the AEs in diverse environments is still an open research question. Finally, although user study shows that none of the participants can identify any command from our AEs if they only listen to them once, a few participants felt our AEs noisy/abnormal. Therefore, improving the stealthiness of AEs is on demand.

8 Conclusion

We present *Devil's Whisper*, a general adversarial attack on commercial black-box ASR systems and IVC devices, and the AEs are stealthy enough to be recognized by humans. The key idea is to enhance a simple substitute model roughly approximating the target black-box platform with a white-box model that is more advanced yet unrelated to the target. The two models are found to effectively complement each other for generating highly transferable and generic AEs on the target, which only requires around 1500 queries on remote services to ensure a nearly 100% success rate of command on attacking most popular commercial ASR systems.

[17]We further used eight samples of the case "Original song + TTS" from Table 3 to attack Siri and only 1 out of 8 samples can work. So, we consider that Siri may have updated the system to ignore the speech with music background.

9 Acknowledgments

We greatly appreciate our reviewers' insightful comments, which helped us to improve our work. Specifically, we thank our shepherd, Professor Yongdae Kim, for his constructive feedback on this paper. We also thank Dohyun Kim and Taekkyung Oh in Professor Kim's group, for their efforts to reproduce our results. IIE authors are supported in part by Beijing Natural Science Foundation (No.JQ18011), NSFC U1836211, 61728209, National Top-notch Youth Talents Program of China, Youth Innovation Promotion Association CAS, Beijing Nova Program, National Frontier Science and Technology Innovation Project (No. YJKYYQ20170070) and a research grant from Huawei. Indiana University author is supported in part by NSF CNS-1527141, 1618493, 1801432, 1838083 and ARO W911NF1610127. For Florida Tech author, part of his work is using the blueshark server supported by NSF CNS 09-23050.

References

[1] *Adobe Audition*. https://www.adobe.com/il_en/products/audition.html.

[2] *Alexa Auto SDK*. https://developer.amazon.com/zh/alexa-voice-service/alexa-auto-sdk.

[3] *Alexa Polly*. https://aws.amazon.com/polly/.

[4] *Amazon Transcribe*. https://aws.amazon.com/transcribe/.

[5] *Audio Adversarial Examples: Targeted Attacks on Speech-to-Text*. https://github.com/carlini/audio_adversarial_examples/blob/master/attack.py.

[6] *Bing Text to Speech*. https://azure.microsoft.com/en-us/services/cognitive-services/text-to-speech/.

[7] *Comparitive examples of noise levels*. http://www.industrialnoisecontrol.com/comparative-noise-examples.htm.

[8] *Demo of CommanderSong*. https://sites.google.com/view/commandersong/.

[9] *From Text to Speech*. http://www.fromtexttospeech.com.

[10] *Google Cloud Speech-to-Text*. https://cloud.google.com/speech-to-text/.

[11] *Google Cloud Text-to-speech*. https://cloud.google.com/text-to-speech/.

[12] *IBM Speech to Text*. https://www.ibm.com/watson/services/speech-to-text/.

[13] *IBM Text to Speech*. https://www.ibm.com/watson/services/text-to-speech/.

[14] *Kaldi ASR*. http://kaldi-asr.org.

[15] *Kaldi ASR: Extending the ASpIRE model*. https://chrisearch.wordpress.com/2017/03/11/.

[16] *Microsoft Bing Speech Service*. https://azure.microsoft.com/en-us/services/cognitive-services/bing-web-search-api/.

[17] *PSU Noisequest*. https://www.noisequest.psu.edu/noisebasics-basics.html.

[18] Hadi Abdullah, Washington Garcia, Christian Peeters, Patrick Traynor, Kevin R. B. Butler, and Joseph Wilson. Practical hidden voice attacks against speech and speaker recognition systems. In *NDSS'19*, pages 1369–1378, 2019.

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

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

[21] Si Chen, Kui Ren, Sixu Piao, Cong Wang, Qian Wang, Jian Weng, Lu Su, and Aziz Mohaisen. You can hear but you cannot steal: Defending against voice impersonation attacks on smartphones. In *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*, pages 183–195. IEEE, 2017.

[22] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 854–863. JMLR. org, 2017.

[23] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9185–9193, 2018.

[24] Yuan Gong and Christian Poellabauer. Protecting voice controlled systems using sound source identification based on acoustic cues. In *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9. IEEE, 2018.

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

[26] Hynek Hermansky. Perceptual linear predictive (plp) analysis of speech. *The Journal of the Acoustical Society of America*, 87(4):1738–1752, 1990.

[27] Fumitada Itakura. Line spectrum representation of linear predictor coefficients of speech signals. *The Journal of the Acoustical Society of America*, 57(S1):S35–S35, 1975.

[28] Chaouki Kasmi and Jose Lopes Esteves. Iemi threats for information security: Remote command injection on modern smartphones. *IEEE Transactions on Electromagnetic Compatibility*, 57(6):1752–1755, 2015.

[29] Deepak Kumar, Riccardo Paccagnella, Paul Murley, Eric Hennenfent, Joshua Mason, Adam Bates, and Michael Bailey. Skill squatting attacks on amazon alexa. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 33–47, 2018.

[30] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

[31] Lindasalwa Muda, Begam KM, and I Elamvazuthi. Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques. *Journal of Computing*, 2(3):138–143, 2010.

[32] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM, 2017.

[33] Yao Qin, Nicholas Carlini, Garrison Cottrell, Ian Goodfellow, and Colin Raffel. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In *International Conference on Machine Learning*, pages 5231–5240, 2019.

[34] Lea Schönherr, Katharina Kohls, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding. In *accepted for Publication, NDSS*, 2019.

[35] Charles Stephenson. Tracing those who left: Mobility studies and the soundex indexes to the us census. *Journal of Urban History*, 1(1):73–84, 1974.

[36] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[37] Rohan Taori, Amog Kamsetty, Brenton Chu, and Nikita Vemuri. Targeted adversarial examples for black box audio systems. In *2019 IEEE Security and Privacy Workshops (SPW)*, pages 15–20. IEEE, 2019.

[38] Tavish Vaidya, Yuankai Zhang, Micah Sherr, and Clay Shields. Cocaine noodles: exploiting the gap between human and machine speech recognition. *Presented at WOOT*, 15:10–11, 2015.

[39] Xuejing Yuan, Yuxuan Chen, Aohui Wang, Kai Chen, Shengzhi Zhang, Heqing Huang, and Ian M Molloy. All your alexa are belong to us: A remote voice control attack against echo. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2018.

[40] 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. *USENIX Security Symposium*, 2018.

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

[42] 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 *Dangerous Skills: Understanding and Mitigating Security Risks of Voice-Controlled Third-Party Functions on Virtual Personal Assistant Systems*, page 0. IEEE, 2019.

[43] Yangyong Zhang, Lei Xu, Abner Mendoza, Guangliang Yang, Phakpoom Chinprutthiwong, and Guofei Gu. Life after speech recognition: Fuzzing semantic misinterpretation for voice assistant applications. In *NDSS*, 2019.

Appendix

A Tuned TTS

Table 5 shows the parameters of tuning[18] and the size of the corpus after filtering for each target black-box model. Note that the filtered corpus for Amazon Transcribe API is less than other platforms even with a much lower confidence. Therefore, we think this model may not be sensitive to our TTS audios of these commands, which is stated in Section 6.2.

Table 5: Preparing corpus for the substitute model.

Black-box	**Corpus (hours)**	**Added-noise (α)**	**Twist rate (β)**	**Confidence**
Google	4.61	0.097	0.77~1.15	> 0.8
Microsoft	6.83	0.094	0.68~1.15	> 0.8
Amazon	3.11	0.066	0.69~1.13	> 0
IBM	4.26	0.038	0.76~1.12	> 0.8

B Impacts of Supplemental Corpus

As mentioned in Section 5.3, we used the open source corpus of Mini Librispeech as supplement to enrich the features of the tuned TTS corpus. To evaluate the impact of various sizes of the supplemental corpus, we trained several substitute models by combining the tuned audios for Microsoft Bing Speech Service API (6.83 hours) with different scales of supplemental corpus as the training corpus. The sizes of the supplemental corpus include 1-hour, 3-hours, 5-hours, 20-hours, 40-hours, respectively[19]. Consequently, we obtained five different substitute models for Microsoft Bing Speech Service API, and generated AEs based on them. The embedded commands are the ten target commands of Microsoft Bing Speech Service API listed in Table 10 in Appendix G.

Table 6: Effect of different size of supplemental corpus on Microsoft Bing Speech Service API.

Training corpus		**Success rate of command (SRoC)**	**Pdf-id numbers**
TTS	**Extra**		
6.83 hours	1-hour	8/10	417
	3-hours	10/10	1632
	5-hours	10/10	2088
	7.35-hours	10/10	2200
	20-hours	10/10	2808
	40-hours	10/10	2928

The results are shown in Table 6. The column "Pdf-id numbers" indicates the total number of the probability distribution function identifiers of the phonemes' features. It can be seen that, except the substitute model trained with 1-hour supplemental corpus, the AEs generated from the other models can

[18] We use "SoX – Sound eXchange" to generate noise, and the maximum amplitude is 1.

[19] Since the size of the entire corpus of the Mini Librispeech is 7.35 hours, so the oversized corpus of 20-hours and 40-hours were randomly chosen from Librispeech corpus (about 1000 hours).

all attack the target successfully. Probably the reason is that the substitute model trained with 1-hour supplemental corpus does not learn enough features.

C Local Model Approximation with a Larger Corpus.

In Table 7, we show the commands and effectiveness for "local model approximation with a larger corpus" method introduced in Section 6.4.

Table 7: Results of using a large corpus trained substitute model.

Command	G1	G2	G3
Okay Google, play music.	*X*	*X*	*X*
Okay Google, take a picture.	*X*	*X*	*X*
Okay Google, turn off the light.	✓	✓	*X*
Okay Google, navigate to my home.	*X*	*X*	*X*

Note: "G1", "G2" and "G3" are used for the abbreviation of "Google command_and_search model", "Google Assistant" and "Google Home", respectively.

D Alternate Models based Generation without Approximation

In Table 8, we show the commands and effectiveness for "alternate models based generation without approximation" method introduced in Section 6.4.

Table 8: Results of alternate models based generation without approximation.

Command	G1	G2	G3
Okay Google, call 911.	*X*	*X*	*X*
Okay Google, take a picture.	*X*	*X*	*X*
Okay Google, set an alarm on 8 am.	*X*	*X*	*X*
Okay Google, navigate to my home.	✓	✓	*X*

Note: "G1", "G2" and "G3" are used for the abbreviation of "Google command_and_search model", "Google Assistant" and "Google Home", respectively.

E Details of Human Perception Survey

The participants were asked to listen to each audio and answer the question whether they think it is a weird song. The details of the survey can be found in https://github.com/RiskySignal/Devil-Whisper-Attack. At the end of the questionnaire, we added one plain TTS audio asking the participant to write down the clearly pronounced number. Such attention question at the end will help us filter out the questionnaires with random responses. In addition, we also recorded how many times each audio was played. Finally, we got 70 effective questionnaires from 120 participants after filtering.

F Transferability of the AEs on Apple Siri

We test the AEs of other platforms on the wakened Apple Siri. If Siri can recognize the hidden command correctly, we consider the AE for the target command successful. Detailed commands can be found in Table 9.

Table 9: Transferability of the Devil's Whispe AEs on Apple Siri.

Command	Black-box	TBA/AGA
Call 911.	Google	*X*/✓
Play music.	Google	*X*/✓
Set an alarm on 8 am.	Google	*X*/✓
Navigate to my home.	Google	*X*/✓
Turn on airplane mode.	Google	*X*/✓
What is the weather?	Microsoft	✓/✓
Call my wife.	Amazon	*X*/✓

G Detail Results of the Target Commands

Detail results of our approach on the target commands are shown in Table 10 and Table 11 for Speech-to-Text API services attack and IVC devices attack.

Table 10: Detail results of the Speech-to-Text API services attack.

Black-box	**Command**	**SNR (dB)**	**Attack type (TBA/AGA)**
Google phone_call API	Okay Google, turn off the light.	14.32	✓/✓
	Okay Google, play music.	15.17	✓/✓
	Okay Google, take a picture.	13.92	✓/✓
	Okay Google, call 911.	12.82	✓/✓
	Okay Google, turn on airplane mode.	11.91	✓/✓
	Okay Google, navigate to my home.	14.28	✓/✓
	Okay Google, set an alarm on 8 am.	12.40	✓/✓
	Okay Google, open Youtube.	7.19	✓/✓
	Okay Google, turn on the WiFi.	8.21	✓/✓
	Okay Google, turn on the bluetooth.	9.44	✓/✓
Google command_ and_search API	Okay Google, turn off the light.	13.13	*X*/✓
	Okay Google, play music.	10.07	*X*/✓
	Okay Google, take a picture.	9.11	*X*/✓
	Okay Google, call 911.	12.80	*X*/✓
	Okay Google, turn on airplane mode.	8.01	*X*/✓
	Okay Google, navigate to my home.	13.36	*X*/✓
	Okay Google, set an alarm on 8 am.	5.82	*X*/✓
	Okay Google, open Youtube.	8.46	*X*/✓
	Okay Google, turn on the WiFi.	5.99	*X*/✓
	Okay Google, turn on the bluetooth.	7.11	*X*/✓
Microsoft Bing Speech Service API	Hey Cortana, send a text.	14.30	*X*/✓
	Hey Cortana, make it warmer.	14.97	✓/✓
	Hey Cortana, open the website.	13.4	*X*/✓
	Hey Cortana, where is my phone?	13.52	*X*/✓
	Hey Cortana, what is the weather?	14.45	*X*/✓
	Hey Cortana, turn off the computer.	14.11	✓/✓
	Hey Cortana, turn on the coffee maker.	13.72	*X*/✓
	Hey Cortana, turn off the bedroom lights.	13.55	*X*/✓
	Hey Cortana, set the temperature to 72 degrees.	9.73	*X*/✓
	Hey Cortana, add an appointment to my calendar.	11.85	*X*/✓
Amazon Transcribe API	Echo, play music.	12.25	*X*/✓
	Echo, call my wife.	NA	*X*/*X*
	Echo, open my door.	NA	*X*/*X*
	Echo, where is my car.	10.92	*X*/✓
	Echo, turn off the light.	NA	*X*/*X*
	Echo, clear notification.	13.27	*X*/✓
	Echo, what is the weather?	NA	*X*/*X*
	Echo, turn off the computer.	8.39	✓/✓
	Echo, turn on the TV.	NA	*X*/*X*
	Echo, turn on the WeMo Insight.	NA	*X*/*X*
IBM Speech to Text API	Education is provided by schools.	12.51	*X*/✓
	Teachers are trained in normal schools.	13.72	*X*/✓
	What would you recommend?	12.30	✓/✓
	The economist provides news and information.	11.54	✓/✓
	Business is the activity of making money.	13.86	*X*/✓
	Share the new version.	11.28	*X*/✓
	This article is about the profession.	8.08	✓/✓
	All governments have an official form.	6.10	*X*/✓
	Children are divided by age groups into grades.	6.75	*X*/✓
	A partnership is a business owned by two or more people.	4.41	*X*/✓

Note: (1) We mark the success of the command with "✓", and the failure with "*X*". (2) As we filter the TTS audios for the corpus of the substitute model, we find that Amazon Transcribe API is harder to recognize the TTS than other API services, especially the word "Echo". The results show that attacking Amazon Transcribe is difficult, which is because that the recognition of the Amazon Transcribe API is much rigorous. (3) All tests were conducted in October 2019. (4) We did not find any detailed software version of API from service provider's documentation/website.

Table 11: Detail results of the IVC devices attack.

Black-box and software version	Command	SNR (dB)	Attack type (TBA/AGA)	SRoA	Speaker	Device	Audio Source
Google Assistant Version-0.1.18794 5513	Okay Google, call 911.	9.50	*X*/✓	19/30	JBL Clip 2	iPhone SE	Lenovo W541 default media player
	Okay Google, set an alarm on 8 am.	8.08	*X*/✓	4/30			
	Okay Google, take a picture.	5.85	✓/✓	5/30			
	Okay Google, turn off the light.	10.75	✓/✓	16/30			
	Okay Google, play music.	11.62	*X*/✓	8/30			
	Okay Google, turn on airplane mode.	8.30	*X*/✓	19/30			
	Okay Google, navigate to my home.	12.02	*X*/✓	18/30			
	Okay Google, open YouTube.	9.49	✓/✓	4/30			
	Okay Google, turn on the Bluetooth.	9,44	*X*/✓	15/30			
	Okay Google, turn on the WiFi.	5.27	✓/✓	12/30	JBL Clip3	iPhone 8	Dell XPS 15
Google Home Version-1.42.171 861	Okay Google, play music.	11.62	*X*/✓	28/30	JBL Clip 3	Google Home Mini	iPhone SE default media player
	Okay Google, turn off the light.	10.75	*X*/✓	15/30			
	Okay Google, turn on airplane mode.	8.30	*X*/✓	18/30			
	Okay Google, call 911.	12.79	*X*/✓	25/30			
	Okay Google, set an alarm on 8 am.	N/A	*X*/*X*	N/A			
	Okay Google, take a picture.	5.85	✓/✓	24/30			
	Okay Google, navigate to my home.	7.62	✓/✓	26/30			
	Okay Google, open YouTube.	9.49	✓/✓	22/30			
	Okay Google, turn on the WiFi.	5.16	*X*/✓	6/30			
	Okay Google, turn on the Bluetooth.	7.67	✓/✓	21/30			
Microsoft Cortana Version-3.3.2.2682	Hey Cortana, send a text.	11.71	*X*/✓	21/30	JBL Clip 2	iPhone 8	ASUS P453U default media player
	Hey Cortana, make it warmer.	9.28	✓/✓	18/30			
	Hey Cortana, open the website.	12.44	*X*/✓	29/30			
	Hey Cortana, where is my phone?	11.67	*X*/✓	6/30			
	Hey Cortana, what is the weather?	9.92	*X*/✓	15/30			
	Hey Cortana, turn off the computer.	10.07	✓/✓	7/30			
	Hey Cortana, turn on the coffee maker.	10.73	*X*/✓	15/30			
	Hey Cortana, turn off the bedroom lights.	9.63	*X*/✓	13/30			
	Hey Cortana, set the temperature to 72 degrees.	10.24	*X*/✓	9/30			
	Hey Cortana, add an appointment to my calendar.	9.77	*X*/✓	14/30			
Amazon Echo Version-647588720	Echo, play music.	13.43	*X*/✓	21/30	JBL Clip 2	Echo 1st gen	ASUS P453U default media player
	Echo, call my wife.	10.86	*X*/✓	17/30			
	Echo, open my door.	11.36	*X*/✓	17/30			
	Echo, where is my car.	11.31	*X*/✓	23/30			
	Echo, turn off the light.	12.36	*X*/✓	28/30			
	Echo, clear notification.	12.45	*X*/✓	10/30			
	Echo, what is the weather?	11.13	*X*/✓	30/30			
	Echo, turn off the computer.	14.28	*X*/✓	11/30			
	Echo, turn on the TV.	11.56	*X*/✓	6/30			
	Echo, turn on the WeMo Insight.	12.21	*X*/✓	14/30			
IBM (WAA)	Education is provided by schools	9.21	*X*/✓	4/30	JBL Clip 2	Huawei P30	iPhone SE default media player
	Teachers are trained in normal schools.	13.74	*X*/✓	10/30			
	What would you recommend?	12.24	✓/✓	25/30			
	The economist provides news and information.	8.07	✓/✓	24/30			
	Business is the activity of making money.	4.07	*X*/✓	24/30			
	Share the new version.	7.89	*X*/✓	12/30			
	This article is about the profession.	7.82	✓/✓	26/30			
	All governments have an official form.	5.33	*X*/✓	13/30			
	Children are divided by age groups into grades.	6.55	*X*/✓	18/30			
	A partnership is a business owned by two or more people.	3.72	*X*/✓	2/30			

Note: (1) We mark the success of the command with "✓", and the failure with "*X*". (2) The practical IVC devices tests were conducted in two meeting rooms about 12 and 20 square meters, 4 meters high. (3) The AE of "Ok Google, turn on the WiFi" was tested on iPhone 8 using JBL Clip 3 speaker, while it cannot succeed on iPhone SE as the other AEs. (4) The volume of AEs is about 65~75 *dB* measured by SMART SENSOR AS824. The distance ranges 5~50 centimeters (5~200 centimeters for Echo). (5) In the tests, the language of the devices needs to be English (US) only and the region/location needs to be US only (if apply). (6) All tests were conducted in October 2019. (7) IBM didn't provide software version for IBM Speech to Text API.