KENKU: Towards Efficient and Stealthy Black-box Adversarial Attacks against ASR Systems

Xinghui Wu, *Xi'an Jiaotong University;* Shiqing Ma, *University of Massachusetts Amherst;* Chao Shen and Chenhao Lin, *Xi'an Jiaotong University;* Qian Wang, *Wuhan University;* Qi Li, *Tsinghua University;* Yuan Rao, *Xi'an Jiaotong University*

https://www.usenix.org/conference/usenixsecurity23/presentation/wu-xinghui

This paper is included in the Proceedings of the 32nd USENIX Security Symposium.

August 9–11, 2023 • Anaheim, CA, USA

978-1-939133-37-3

Open access to the Proceedings of the 32nd USENIX Security Symposium is sponsored by USENIX.

KENKU: Towards Efficient and Stealthy Black-box Adversarial Attacks against ASR Systems

Xinghui Wu[1] Shiqing Ma[2] Chao Shen[1] Chenhao Lin[1] Qian Wang[3] Qi Li[4] Yuan Rao[1]

[1]*Xi'an Jiaotong University* [2]*University of Massachusetts Amherst*
[3]*Wuhan University* [4]*Tsinghua University*

Abstract

Prior researchers show that existing automatic speech recognition (ASR) systems are vulnerable to adversarial examples. Most existing adversarial attacks against ASR systems are either white- or gray-box, limiting their practical usage in the real world. Some black-box attacks also assume the knowledge of output probability vectors to infer output distribution. Other black-box attacks leverage inefficient heavyweight processes, i.e., training auxiliary models or estimating gradients. Moreover, they require input-specific and manual hyperparameter tuning to improve the attack success rate against a specific ASR system. Despite such a heavyweight tuning process, nearly or even more than half of the generated adversarial examples are perceptible to humans.

This paper designs KENKU, an efficient and stealthy black-box adversarial attack framework against ASRs, supporting hidden voice command and integrated command attacks. It optimizes the novel acoustic feature loss and perturbation loss, based on Mel-frequency Cepstral Coefficients (MFCC). Both loss values can be calculated locally, avoiding training auxiliary models or estimating gradients, making the attack efficient. Furthermore, we introduce a hyperparameter in optimization that balances the attack effectiveness and imperceptibility automatically. KENKU uses the binary search algorithm to find its optimal value. We evaluated our prototype on eight real-world systems (including five digital and three physical attacks) and compared KENKU with five state-of-the-art works. Results show that KENKU can outperform existing works in the attack performance.

1 Introduction

Fueled by massive amounts of training data, deep neural networks (DNNs) have achieved state-of-the-art results in automatic speech recognition (ASR) systems that translate human voices into transcriptions in natural languages. In recent years, commercial services using ASRs are becoming more popular. The core of Apple Siri includes an ASR to translate human voices into transcriptions and then process the commands [7]. Similarly, Google Assistant contains a DNN empowered ASR [6]. Amazon also releases Alexa devices which ship a built-in ASR system to accept human commands [4]. Till now, these devices and services have attracted billions of users. Together with other intelligent home devices, these ASRs have significantly changed our lives and work.

Prior work shows that DNN based ASRs are vulnerable to two types of audio adversarial examples [15]. *1) Hidden voice command attacks* generate audio that sounds like random noise but can fool ASRs into predicting a target command [13, 18, 60]. *2) Integrated command attacks* add perturbation to a given carrier audio, e.g., a song clip, so that the adversarial example can trigger the target ASR to output the desired command [19, 21, 24, 24, 49, 54, 69, 75].

Generating audio adversarial examples is challenging, and as such, most current works focus on analyzing individual ASRs and designing white-box [18, 19, 49, 54, 69] or gray-box attacks [44, 64]. In such attack settings, the adversary has full or partial knowledge of the target systems, making strong assumptions. In the real world, ASRs used in commercial products are all black-box models. Consequently, white-box attacks have limited practical values. Some existing black-box attacks [21] also require the output probability vector or the output distribution, which is not practical. This paper aims to design a black-box attack that assumes the minimum information, i.e., only the output text from target ASRs.

Traditional black-box attacks have two main streams. The first one is to train an auxiliary model locally to mimic the behaviors of the specific black-box ASR, by using the training corpus labeled by the target system [21]. Leveraging the transferability of adversarial examples, the adversary can generate attack samples on the auxiliary model and test if they can transfer to the target system. This approach involves collecting datasets and training auxiliary models, which are heavyweight processes. The second one is to estimate gradient information used in white-box attacks [75]. For example, the adversary can add small perturbations to a given input and observe the output changes to estimate gradients. The adversary also has

to query the model multiple times to increase its confidence to get accurate estimations. In short, this is also time-consuming. Existing works [19, 21, 69] confirm that using modern GPU hardware as accelerators, their attacks still spend more than half an hour to generate a single adversarial example.

It is also hard for existing works to automatically balance the two most important goals for adversarial examples: effectiveness and stealthiness. Adding more significant perturbations can increase the attack success rate, improving effectiveness, but it will worsen the stealthiness and vice versa. It is challenging to optimize these two contradictory goals. Existing works [13, 18] require input-specific hyperparameter tuning to achieve high attack success rates. Despite so, more than half of their generated adversarial examples are still perceptible to humans (see §6.3).

To resolve these problems, we design KENKU[1], a general and unified audio adversarial attack framework, supporting both hidden voice command and integrated command attacks. The core of KENKU is a novel optimization engine that simultaneously optimizes two losses: the acoustic feature loss and the perturbation loss. The acoustic feature loss directly measures the quality of acoustic features in given audio and affects the attack success rate. The perturbation loss accesses the size of the perturbation and controls the stealthiness. One key feature of this optimization problem is that it focuses on acoustic features that can be measured locally without querying the remote target ASR system. By doing so, we avoid heavyweight auxiliary model training or gradient estimation processes, achieving efficiency. We also introduce a new hyperparameter λ in optimization, balancing attack effectiveness and imperceptibility. To avoid manually tuning such hyperparameters, KENKU uses a binary search algorithm to find the optimal value of λ. Notice that this hyperparameter is target ASR specific, i.e., for one target ASR, we only need to search for its optimal value once.

We evaluated KENKU on five different digital platforms, i.e., Google [10], Microsoft [11], Alibaba [1], Tencent [2] and iFLYTEK [3], and three different physical applications, i.e., Apple Siri [7], Google Assistant [6] and Amazon Alexa [4]. Compared with existing optimization based attacks, KENKU only uses 1/60 or less time to generate an integrated command. The two modes of KENKU both achieve a 100% success rate in the digital settings. We evaluated the physical attacks by placing the speaker and microphone at different distances and KENKU outperformed state-of-the-art attacks. According to the SNR and user study results, KENKU shows promising performance of imperceptibility. Also, KENKU can bypass four state-of-the-art defense strategies, including audio compression [69], temporal dependency [68], MVP-EARS [70] and WaveGuard [31].

In summary, we make the following contributions:

[1] KENKU are a fictional race of bird-like humanoid creatures in the Dungeons & Dragons. They are most recognizable for the lack of a voice but communicating by perfectly mimicking voices they have heard.

Table 1: Notations and their corresponding descriptions.

Notation	Description
A	Short-Time Fourier Transform matrix
B	Mel filter transform matrix
C	Discrete Fourier Transform matrix
X	acoustic feature of the audio sample
W	word sequence of the audio sample
y_t	target command transcription
y	target command audio
x	clean carrier
δ_0	random uniform distribution vector
α	scaling factor of hidden voice command initialization
β	scaling factor of integrated command initialization
δ	adversarial perturbation
x'	audio adversarial example
$\mathcal{L}_f$	acoustic feature loss
$\mathcal{L}_p$	perturbation loss
$\mathcal{L}_p^h$	perturbation loss of hidden voice commands
$\mathcal{L}_p^i$	perturbation loss of integrated commands
λ	weight factor
$\mathcal{L}$	final loss function
$N_{\max}$	upper bound of λ for binary search
$N_{\min}$	lower bound of λ for binary search
P_{signal}	average power of the signal
P_{noise}	average power of the noise

- We propose a black-box attack framework against ASRs, requiring minimal feedback information with novel designs and supporting both hidden voice command and integrated command attacks. It features efficiency and automatic balancing between attack effectiveness and imperceptibility. It is transferable across different ASRs and robust in over-the-air environments and can bypass existing defense countermeasures.

- We implement a prototype based on the proposed idea and it is 60 times faster in generating integrated commands than existing works. KENKU can achieve a 100% success rate against five digital platforms. We evaluate the physical attacks under different device distance settings and KENKU outperforms state-of-the-art attacks. The SNR and user study results demonstrate that KENKU can achieve better balance between effectiveness and imperceptibility than existing works. Moreover, KENKU shows better robustness in the presence of four state-of-the-art defenses than baseline attacks.

2 Background & Related Work

To facilitate our discussion, we summarize used notations and their descriptions in Table 1 for reference.

2.1 ASR Systems

ASR systems accept audio signals as input and output corresponding text transcriptions. Figure 1 shows the typical workflow of an ASR. The system first preprocesses the signals by extracting features. Then, a machine learning model, known as the translation model, will take such signals as inputs and predict corresponding texts. Deep Neural Networks

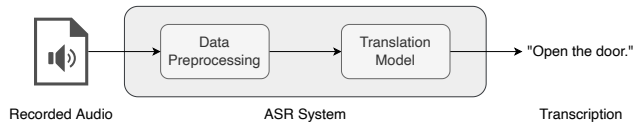

Figure 1: General architecture of ASR systems.

(DNNs) have shown state-of-the-art performance, and are widely used in modern ASR systems [7, 33, 36, 37, 40, 74].

Data Preprocessing. It is challenging to train translation models on raw digital audio files representing the high-dimensional waveform data. Therefore, preprocessing audio signals is essential in ASRs, which usually involves the acoustic feature extraction procedure. There are many preprocessing algorithms, including signal processing methods, e.g., Mel-frequency Cepstral Coefficient (MFCC) [12, 42, 43, 47, 48], FBank [50], Power Spectrum [16], and machine learning methods, e.g., CNN [17]. Among all, MFCC extracts the most accurate and robust acoustic features and is the de facto standard used by the majority of ASR systems including CMU Sphinx [42], Julius [43], Kaldi [47], Wav2Letter++ [48] and Mozilla DeepSpeech [12].

MFCC first applies Short-Time Fourier Transform (STFT) in the time domain for the raw waveform of an audio file and converts frequency-domain signals to logarithm scale by using Mel filters. Then, it leverages Discrete Cosine Transform (DCT) to compress the audio feature and keep the first set of elements as output. We can simplify the whole process into a single equation: $MFCC(x) = C\log\left(B||Ax||^2\right)$, where A, B and C are correspondingly STFT, Mel filters, and DCT transform matrices [18]. Other algorithms either perform part of the operations (e.g., FBank does not perform DCT while Power Spectrum does not involve both Mel filters and DCT) or simulate the MFCC process (e.g., CNN-based methods).

Translation Model. In ASRs, a translation model converts acoustic observations to word sequences. Formally, we use X to represent the input acoustic features from preprocessing and $W = (w_1, w_2, \ldots, w_m)$ to denote word sequences. The translation model aims to find a word sequence W that has the highest probability given the observed audio signal X:

$$W^* = \arg\max_W \mathbb{P}(W|X) \quad (1)$$

$$= \arg\max_W \mathbb{P}(X|W)\mathbb{P}(W)/\mathbb{P}(X) \quad (2)$$

$$= \arg\max_W \mathbb{P}(X|W)\mathbb{P}(W) \quad (3)$$

Here, $\mathbb{P}$ calculates the probabilities. Equation 1 describes the discriminative translation model while Equation 3 reflects training the generative model. Using discriminative architecture produces one modelwith an end-to-end training procedure. Popular model architectures in this kind [26–28, 35, 61, 66] uses Recurrent Neural Networks (RNNs) or transformers. Generative translation models consist of two models: an acoustic model $\mathbb{P}(X|W)$ and a language model $\mathbb{P}(W)$. The acoustic model describes the phonemes for a given sequence of words, and the language model represents the likelihood of word sequences. ASR systems like Kaldi [47] family are using such methods.

2.2 Attacks against ASRs

ASRs are vulnerable to various attacks [23, 34, 39, 51, 56, 57, 71, 72]. In this paper, we focus on adversarial examples against ASR systems, which exploit the vulnerabilities of DNNs. The threat model and the adversary knowledge can categorize adversarial attacks into white-box, gray-box, and black-box attacks. White-box attacks [18–20, 49, 53, 54, 67, 69] refer to the threat model where the attacker knows everything about the target system, including the used model, its architecture, and trained weights, and can perform desired computation using the target system, such as gathering gradients. In other words, the adversary has complete knowledge of the target system. Gray-box attack [58, 64] refers to cases where the adversary has limited access to the system. In contrast, black-box attacks [13, 14, 18, 21, 24, 44, 60, 75] have no direct access to the system internals, which is the most practical setting for the attack design. Here we discuss a few representative ones.

Carlini et al. [19] design a white-box audio adversarial attack against Mozilla DeepSpeech. The formulated optimization problem uses the Connectionist Temporal Classification (CTC) loss function [26] to measure the distance between the target audio label y_t and adversarial example label $f(x')$ where f is Mozilla DeepSpeech. It uses the gradient descent to solve this problem.

Chen et al. [21] propose Devil' Whisper, a transferability based black-box attack. It trains a local substitute Kaldi model to simulate the target commercial ASR and then generates adversarial examples for the local substitute model using white-box attack methods [69]. It needs to train different models for different targets, and training such models can be time-consuming, taking around hours.

Zheng et al. [75] proposed NI-Occam, a non-interactive physical attack against voice control devices, which improves Devil's Whisper attacks from a few aspects. First, it introduces a randomization strategy at the beginning of each optimization iteration. Second, it leverages the AdaBelief [76] optimizer rather than Adam [38] or SGD [52] optimizer. Third, it is independent of any specific target ASRs. By doing so, NI-Occam saves the potentially large query cost and makes it easier to attack black-box ASRs.

Carlini et al. [18] reverse-engineer audio files that have desired acoustic features to attack the target ASR system. The core component of the attack leverages the extracted acoustic features of the target command audio as inputs. The adversary reverses the feature extraction procedure and manually adjusts the four MFCC parameters iteratively until a candidate obfuscated audio is found.

Abdullah et al. [13] share the same idea with Carlini et al. [18] but leverage the domain knowledge of signal processing to generate adversarial examples. To create unintelligible audios, they propose four perturbation strategies, i.e., time-domain inversion, random phase generation, high-frequency addition, and time scaling. These perturbation strategies periodically improve the attack performance, but the generated adversarial examples also rely on hyperparameter tuning.

2.3 Existing Defenses

Previous studies have looked into the defense strategies against audio adversarial attacks [15, 18, 21, 58, 59, 62, 63, 65, 68–70, 75]. The heavyweight adversarial training applied in the image space [45] has two main drawbacks in the audio domain. First, it requires too much computation cost. Second, it will worsen the performance on normal samples significantly [41], which affects the user experience with the commercial products. Therefore, existing effective defenses either happen in the data preprocessing stage or incorporate an audio adversarial example detector.

Yuan et al. [69] propose audio squeezing to defend against CommanderSongs. They use either the down-sampling algorithms or audio compression to squeeze audio, and these lossy transformations will compromise the carefully crafted tiny adversarial perturbation. Adversarial example detection is another type of defense, which distinguishes between benign and malicious samples. Yang et al. [68] utilize the inherent temporal dependency to detect audio adversarial examples. The basic idea is to select the first k portion of an audio sequence to obtain the partially transcribed result from ASR systems as S_k. Compare S_k with the first k portion of the entire transcription of the whole audio sequence to detect those abnormal samples. Based on the observation that audio adversarial examples have poor transferability and inspired by multi-version programming, MVP-EARS [70] utilize the diverse off-the-shelf ASRs to determine whether audio is adversarial. WaveGuard [31] incorporates audio transformation functions and analyzes the ASR transcriptions of the original and transformed audio to detect adversarial inputs. As for all these detection techniques, an input is classified as adversarial if the difference between the multiple transcriptions exceeds a particular threshold.

3 Threat Model & Challenges

3.1 Threat Model

This paper aims to design a black-box adversarial attack framework against ASR systems. The adversary has no access to the ASR system but can query the target ASR and get corresponding text transcriptions (without the numerical vectors). The adversary does not know the preprocessing algorithm, translation model design (i.e., architectures, discriminative/generative model) and weights, or the training settings (i.e., training data and hyperparameters). Black-box assumes minimal knowledge of the target ASR, and our attack is more potent than some existing black-box attacks because it does not require the numerical output of the model.

Researchers [15] categorize audio adversarial attacks into *hidden voice commands* and *integrated commands* attacks. The former produces noise-like audios that ASRs can recognize. The latter integrates small perturbations into benign audios like songs, known as the carriers, to fool ASRs. Both attacks are *targeted*, meaning that the adversary attempts to force the ASRs to transcribe the input audio to specific texts. Our framework aims to support both attacks.

3.2 Challenges

This paper assumes minimal knowledge of the target ASRs and tries to have a unified framework for all attack scenarios, making it harder to generate adversarial examples. Compared with white- and gray-box attacks, black-box attacks are more practical and challenging because of the lack of knowledge of the target ASRs. Compared with other black-box attacks, we do not require even the output probability vector, making it harder to attack. In [75], researchers show that service providers can mitigate existing black-box attacks by limiting QPS (query per second) or exposing no output distribution. Moreover, current work focuses on either the hidden voice command or integrated command attacks, while ours has a general framework designed for both types.

Traditional audio adversarial attacks require heavyweight optimization computation, limiting their wide use. Carlini et al. [19] spend at least one hour generating a single adversarial example on a NVIDIA 1080Ti GPU server. Other attacks [21, 75] take similar time. Moreover, Devil's Whisper [21] trains a substitute model, requiring around five hours of training (about 1500 queries) to ensure the successful conversion of 10 target commands into workable adversarial examples. The most time-consuming for black-box methods is to query the model and get feedback to substitute models or estimate gradients. An accurate substitute model or gradient estimation requires numerous queries to the model. Efficient attacks reduce attack time and cost, enable attacks in time-constraint scenarios, and enlarge attack scales [15].

It is challenging to balance the attack effectiveness and the stealthiness automatically without significant manual efforts. Devil's Whisper [21] is limited to train specific substitute models for specific black-box systems and overfit a ten-command set to achieve good results, limiting its wide use [75]. Carlini et al. [18] and Abdullah et al. [13] both have to require significant manual effort in adjusting and fine-tuning signal processing related hyperparameters per sample and per target ASR to obtain successful hidden voice commands. Despite the heavyweight manual efforts, the reported human recogni-

tion rate is around 50% in average and up to 94% in the worst case [18]. These results show the difficulty of automatically achieving both the attack effectiveness and imperceptibility.

4 Methodology

4.1 Attack Overview

The overreaching idea of KENKU is to *generate audios that share similar acoustic features with the target command as adversarial audio examples*. Figure 2 shows the workflow of KENKU. For a given target command y_t, we first get a target command audio y by using the existing Google Text-to-Speech service. For the integrated command attack, KENKU also requires a carrier x (e.g., a song clip). The core of KENKU is an optimization process. First, KENKU initializes the perturbation δ. Then, it optimizes δ by using two loss functions: the acoustic features loss and the perturbation loss.

The acoustic features loss $\mathcal{L}_f$ measures the acoustic feature similarity between the target command audio and perturbed audio. For the hidden voice command attack, the perturbed input is the optimized perturbation itself (i.e., $x' = \delta$), while for the integrated command attack, the perturbed input is the carrier with the optimized perturbation (i.e., $x' = x + \delta$). $\mathcal{L}_f$ determines if the attack will be successful or not. If its value is small, the generated adversarial example x' shares similar acoustic feature with the target command audio y, and the probability of the attack being successful is high, and vice versa. The perturbation loss $\mathcal{L}_p$ controls the quality of the generated adversarial example, and it is specific to attack types. For the hidden voice command attack, we want the adversarial example to sound like random noise and design the loss $\mathcal{L}_p^h$. In contrast, for the integrated command attack, we create another loss $\mathcal{L}_p^i$, aiming to make sure the adversarial example x' is close to the carrier x. To balance the losses, we introduce another hyperparameter λ and design a binary search method to automatically search for an optimal value for λ. After solving the optimization problem, we then test if the generated example can fool target ASRs.

The optimization process works on local perturbations and target command audios. Through the whole process, KENKU only requires querying the model to test if the generated adversarial example can successfully fool the target ASR. The λ controls the importance of two losses during the optimization, and automatically determining its value can balance the two attack goals: generating effective ($\mathcal{L}_f$ loss) and stealthy ($\mathcal{L}_p$ loss) attack samples. Based on the current attack performance of pre-generated adversarial examples, we leverage binary search to narrow down the range and finally determine the value of λ. Once λ is set properly, KENKU calculates the designed loss locally, using the same λ value for all target commands, and does not interact with the ASR.

In summary, KENKU is the first unified framework that supports both hidden voice command and integrated command attack, assuming minimal knowledge about the black-box target ASRs compared to existing practice [19, 21]. Compared to existing integrated command attacks [19, 21, 75] that query the model during the optimization iterations, KENKU applies the optimization technique to generate adversarial examples based on MFCC features, making it lightweight, independent of specific machine learning models, more transferable and universal. Compared to existing hidden voice command attacks [13, 18] that also aim at the acoustic feature space but require significant manual effort in fine-tuning hyperparameters, KENKU can better balance the attack effectiveness and imperceptibility automatically by using the binary searchable λ in the optimization engine.

4.2 Attack Initialization

4.2.1 Obtaining Target Command Audio

In KENKU, users can provide a target command audio y to start the attack or a target command text y_t. For a given target command text y_t, KENKU can automatically get the corresponding target command audio y as a vector representing the audio waveform by using the Google Text-to-Speech service that takes a text as input and produces an audio file reading the given text with minimal background noises.

4.2.2 Perturbation Initialization

We draw a random value from the uniform distribution $\mathcal{U}(-1, 1)$ as our initial perturbation vector δ_0. Notice that the audio data is normally distributed between -1 and 1. After initialization, δ_0 is a random noise clip. Hidden voice commands are generated without carriers, unlike integrated commands, which account for the two different initializations in Figure 2. For the hidden voice command attack, the perturbation δ is used as the perturbed adversarial example x', i.e., $x' = \delta$. We can directly optimize this noise to get an adversarial example. In practice, to speed up the optimization, we embed the target command audio so that the initialized command has all the needed acoustic features. Specifically, we initialize the performance with $x'_0 = y + \alpha \cdot \delta_0$, where a typical value of α is 0.5. We want to keep the needed acoustic features and enlarge the noises as much as possible during optimization. For the integrated command attack, we have a similar practical approach to initialize, i.e., $x'_0 = x + \beta \cdot \delta_0$, where β is small, e.g., 10^{-4}.

4.3 Perturbation Optimization

The optimization problem KENKU uses to generate adversarial examples can be written as the following:

$$\mathcal{L} = \mathcal{L}_f + \lambda \times \mathcal{L}_p \tag{4}$$

, where the loss $\mathcal{L}$ consists of two terms: $\mathcal{L}_f$, the acoustic feature loss, and $\mathcal{L}_p$ perturbation loss. By minimizing Equation 4,

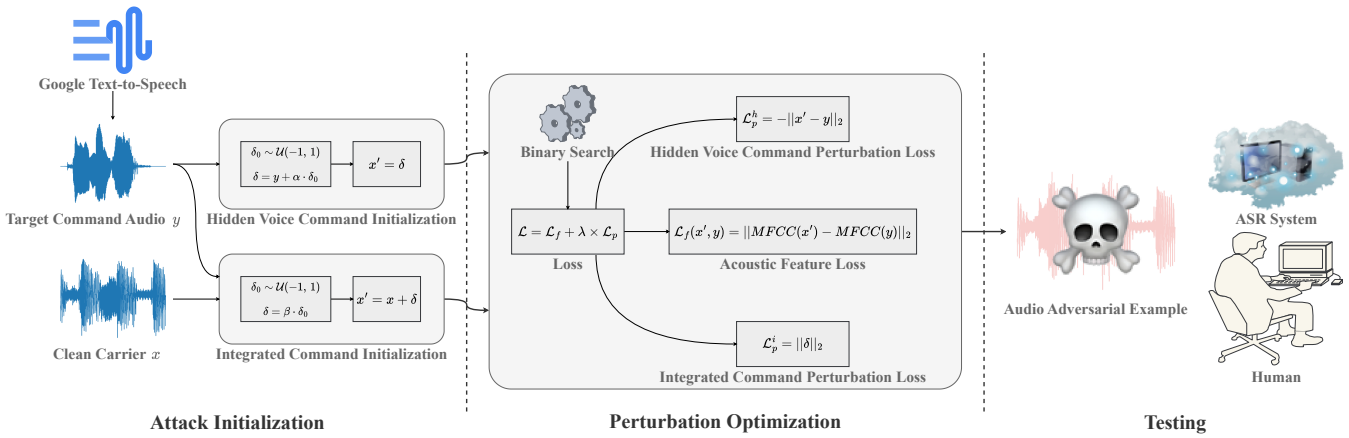

Figure 2: KENKU Framework.

KENKU can eventually obtain the adversarial perturbation output to synthesize an adversarial example.

As discussed in §4.1, $\mathcal{L}_f$ affects the attack success rate and $\mathcal{L}_p$ controls the stealthiness. They can be contradictory goals to optimize. The generated adversarial example can be similar to target command audio if only optimizing $\mathcal{L}_f$, making it less stealthy. Similarly, only optimizing the $\mathcal{L}_p$ can lead to low effective attacks. We use the hyperparameter λ to balance these two goals and use binary search to find optimal values for λ. Specifically, we first set λ to extremely small and large values (e.g.,10 and 100). Then, we leverage a small dataset to generate adversarial examples under different λ values and measure the quality of the attack. Detailed metrics used in KENKU are discussed in §5.3. If the adversarial examples have a high attack success rate but are not stealthy, we enlarge the λ value using binary search and vice versa. This search process is for per target ASR instead of per sample.

Notice that optimizing Equation 4 does not require querying the target ASR model. KENKU only needs to test the target ASR to find a proper λ value and examine if the generated adversarial example can successfully fool the target system or not. Once we get an optimal λ value, it will be applied for all the target commands. We will discuss the feasibility of binary search for this configurable parameter in §5.4.

4.4 Acoustics Feature Loss

The acoustic feature loss $\mathcal{L}_f$ measures the acoustic feature similarity between the adversarial example x' and target command audio y. To calculate the similarity of two values, we can use existing metrics like ℓ_2-norm. The critical challenge in designing this loss is how to measure the quality of acoustic features, which is essential to model and distinguish different audio signals. One should understand how the human brain recognizes sound signals and the digital signal processing principle to generate a ground truth for acoustic features. In practice, we can only measure this based on empirical results.

Observing that data processing in ASRs is trying to extract high-quality acoustic features, KENKU can adapt existing procedures as a measurement. As discussed in §2.1, MFCC is the de facto standard in modern ASRs. MFCC is a linear cosine transform of a log power spectrum on a nonlinear Mel-scale of frequency. It approximates the human auditory system's response more closely than the linearly-spaced frequency bands used in normal spectrums and is robust to background noise. In practice, it achieves the best results, as demonstrated by existing works [22, 25, 30, 32]. In Appendix A, we also show the effectiveness of using MFCC to work across various systems. Therefore, KENKU uses MFCC as the function to measure acoustic features. Thus, our final acoustic features loss is:

$$\mathcal{L}_f(x', y) = ||MFCC(x') - MFCC(y)||_2 \quad (5)$$

This acoustic feature loss design is critical to realizing black-box attacks in the physical world. Optimizing our loss does not require information from the target ASR, making it feasible to perform black-box attacks. Compared with traditional black-box attacks that train auxiliary models or estimate gradients, the computation is lightweight. Moreover, MFCC is robust against physical noises. Using this loss makes sure that our generated noises can bypass functions like MFCC and hence the data preprocessing in modern ASRs, making the attack more robust in physical settings. Notice that KENKU does not assume the use of MFCC in target ASRs. The success of KENKU comes from the fact that MFCC has the best empirical results for measuring the quality of acoustic features, which has been shown by existing works [22, 25, 30, 32] and also our evaluation in Appendix A.

4.5 Perturbation Loss

4.5.1 Hidden Voice Command Attacks

Recall that a hidden voice command attack aims to generate noises that ASRs can recognize. For this attack, we design

the following loss to guarantee the adversarial example is obfuscated and different from the target command audio:

$$\mathcal{L}_p^h = -||x' - y||_2 \quad (6)$$

The loss measures the difference between the adversarial example and target command audio with ℓ_2 distance. When minimizing the loss L, we are trying to generate an adversarial example x' that is very different from the target command y.

4.5.2 Integrated Command Attacks

As discussed earlier, we want the perturbation λ in integrated command attacks to be small so that the adversarial example sounds similar to the carrier. Thus, our perturbation loss in the integrated command attacks is:

$$\mathcal{L}_p^i = ||\delta||_2 \quad (7)$$

4.6 Testing

After optimizing the loss, we can obtain adversarial examples and test them by querying the target ASR system via provided APIs or other mechanisms. This procedure is standard, and we omit the details in this paper.

5 Experiments

In this section, we discuss the experimental settings for our hidden voice command and integrated command attacks. There are distinct attack-specific settings for the two types of scenarios, which we will discuss in §6 and §7 respectively.

5.1 General Experiment Setup

Implementation and Parameter Settings. As for KENKU, we leverage PyTorch [9] to solve the optimization problems with the advanced Adam optimizer [38]. Since open-source MFCC parameters share uniform values, we can alleviate the issue of tuning them in KENKU, and our results demonstrate the feasibility of bypassing the need to adjust these parameters. The key hyperparameter of KENKU is the weight factor λ. Our observation shows that the optimal value of λ can be found by binary search to balance the attack effectiveness and the attack imperceptibility, which will be discussed in §5.4. Notice that fine-tuning attack hyperparameters, including λ, the number of iterations, and learning rate, is a one-time process as preliminary experiments on a small test set. And once fixed, subsequent large-scale experiments all use the same setting combination. Our results show that one hyperparameter set can work well for almost all cases (see Appendix B), which saves much manual effort and thus makes KENKU more efficient.

Hardware and Software. The experiments were performed on a GPU server equipped with Intel Xeon Silver 4210R CPU, 188 GB main memory, and an NVIDIA GeForce RTX 3090 graphics card running Ubuntu 20.04 and PyTorch 1.7.1.

State-of-the-art Methods. To validate the performance of KENKU, we use several state-of-the-art attacks as baselines under the same experimental settings for a fair comparison. For hidden voice command attacks, we compare KENKU with [18] and [13], and for integrated command attack, we compare KENKU with [19], [21] and [75]. We have made the evaluation comprehensive in various aspects by directly using their open-source code or re-implementing the corresponding existing attacks.

Carlini et al. [18] manually fine-tuned four hyperparameters of MFCC to find audio adversarial examples. As they did not report the recommended experimental settings, we re-implemented their method based on the Griffin-Lim algorithm [29, 46] in the Librosa [8] library. We selected typical values of the corresponding hyperparameters widely used in open-source platforms [12, 42, 47, 50] and the values in their neighborhoods. Devil's Whisper [21] initially utilized confidence scores to filter the synthetic audio data. However, recent ASR APIs begin to hide the confidence score information from the users. Thus, following the same setting as Zheng et al. [75], we omitted this step to adapt Devil's Whisper to the decision-based attack as well. Finally, we trained substitute models for our target ASR APIs separately. For other comparison attacks [13, 19, 75], we used the default settings in their published codes or followed the reported rules of thumb. For example, Carlini et al. [19] required 20,000 iterations for music-like carriers for a 100% success rate, and we followed the same setting. Abdullah et al. [13] claimed that Time Domain Inversion (TDI) performs the best among their proposed four perturbation techniques. We believe that we have tried our best to reproduce the reported results in their papers.

Target Commands and Carriers. Due to the limitations of existing attacks, such as the high computation cost and heavyweight manual effort to fine-tune hyperparameters, they typically evaluated 10 commonly used voice commands and fine-tuned each adversarial example against each target ASR. In this paper, for comparison purposes, we used the same setting for all baselines and ours. Specifically, we directly followed the ten choices of target commands in [75], which were also the typical choices of other previous studies [13, 21, 69]. However, we further evaluated KENKU with a large-scale set in both over-the-APIs and over-the-air settings as well as in the presence of defenses (see Appendix B).

As for integrated command attacks, CommanderSong [69] and Devil's Whisper [21] carefully investigated the effects of different carrier materials, and they found that music-like carriers can be most conducive to attacks. Therefore, We reused carriers from [21] that were demonstrated effective for a fair comparison (totally 5 music clips), consistent with subsequent practice [75]. The CW attack [19] was reported to be adaptive to such music-like carriers as well. Therefore,

the carrier choices in our experiments do not confuse any of the baseline attacks.

5.2 Attack Specific Setup

Digital Domain Targets. Several big companies provide an interface to their ASR systems powered on cloud computing platforms. In most experiments, we target the black-box commercial ASR systems including the cloud services provided by Google [10], Microsoft [11], Alibaba [1], Tencent [2] and iFLYTEK [3]. Such systems provide APIs that accept and transfer the local audio file and return the text transcription from the remote server. During this process, there is no data loss or extra noise introduced.

Physical Domain Targets. In order to further investigate the robustness of audio adversarial attacks, we evaluate KENKU on popular state-of-the-art voice recognition applications in smartphones in an over-the-air environment, including Apple Siri [7], Google Assistant [6], and Amazon Alexa [4], which have the largest market share and all have millions of users all over the world. In such attack scenarios, there exists signal loss, background noise and electrical noise that will significantly affect the attack performance.

Specifically, we used a MacBook Pro running macOS 15.6 to play all the adversarial examples (the volume set to 75%) to test the applications in an iPhone 12 running iOS 14.1. These are commonly seen devices among users, representing usual usage scenarios. The two devices were placed on a desk in parallel as people usually do. We comprehensively evaluated the physical attacks by placing the two devices at different distances (i.e., 10cm, 30cm, 50cm, and 100cm) to measure the effect of signal loss and distortion. The experiments were conducted in a quite room of 10 square meters. Following prior works [75], we manually triggered the voice assistants for incoming voice by pressing the app buttons, as wake-word detection subsystems usually examine the specific user voiceprint. If one sample can be correctly recognized by the devices as the target command within three attempts (play the sample within three times), we considered this sample successful, which was the same as [75].

All the experiments were conducted and the results were confirmed in 2023. We promise that we attack the real-world ASR systems with our adversarial examples only for academic research purposes to study the security and reliability of the modern ASR techniques. We have reported our results to corresponding service providers and developers. The ethical considerations will be discussed in details in §9.

5.3 Evaluation Metrics

5.3.1 Attack Efficiency

In this paper, we compare several attacks with a new metric, computation efficiency, which is measured by the time used to generate one single adversarial example. Note that we record the average time for each attack method to generate an adversarial example for a target command using our hardware and software.

5.3.2 Attack Effectiveness

The most important metric to evaluate an adversarial attack is the attack effectiveness. As for our targeted attack goal, we would like the machine output to exactly match what we have specified. Any word errors would be considered a failure. With regard to each target ASR system, we calculate the corresponding attack success rate (SR) for the attack effectiveness, that is, the number of successful commands vs. the total number of the commands. The number of total commands equals to the size of the test set of target commands. For each specific target command, if we can find at least one adversarial example that can be correctly recognized as the given transcription by the target ASR system, we take this command as a success. We use SR to fairly compare KENKU with existing works in the same settings afterwards.

5.3.3 Attack Imperceptibility

Besides the attack effectiveness to evaluate an adversarial attack, it should also be imperceptible. For example, human beings only consider an adversarial example as meaningless noise or a normal song clip, rather than identify the malicious command that the adversary wants to hide.

The imperceptibility of an attack sample is first evaluated by the commonly used signal-to-noise ratio (SNR). SNR is defined as the ratio of the power of a signal (meaningful input) to the power of background noise (meaningless or unwanted input) in the logarithm-scale, formalized in Equation 8, where P is an average power.

$$SNR(\text{dB}) = 10\log_{10}(\frac{P_{signal}}{P_{noise}}) \tag{8}$$

As for hidden voice commands (including [18] and [13], we expect them and the target commands to sound as different as possible. In this context, the target command audio y is regarded as the signal in Equation 8, while the difference between the generated sample and the target command (i.e., $x' - y$) is regarded as the noise. Hence, the lower the SNR, the better the quality of the malicious samples. As for integrated commands (including [19], [21] and [75]), the original song carrier x is regarded as the signal and the final perturbation δ is taken as the noise. Hence, in this attack scenario, the higher the SNR, the better the quality of the integrated commands.

However, using SNR to evaluate audio adversarial examples can be unreliable, since human's perception of sound is subjective. Sometimes, adversarial examples with good SNR values may have poor quality or vice versa in reality. Thus, following the same design as previous work [18, 21, 69, 75],

we additionally carried out user surveys to evaluate the quality of the attack samples using Amazon Mechanical Turk (MTurk) [5] to make comprehensive evaluation about the attack imperceptibility. Note that all the participants were native English speakers without hearing impairments and we did not restrict the time volunteers were given.

Following existing work [21, 75], for each attack, we randomly selected one adversarial example with the lowest word error rate (most successful) for each target command. We mixed the generated adversarial examples of baselines and KENKU up and shuffled them to get evaluation sets for the user studies. As for the user study of integrated commands, we additionally selected normal song clips as the audio quality baseline, which had the equal share with each integrated command attack. Depending on the nature of the two attack scenarios (one generates noise-like samples, the other generates song-like samples), we uploaded our audio sample sets and designed two distinct questionnaires for participants, which will be discussed in more details in §6.3 and §7.3.

Finally, Amazon MTurk would return us each participant' answers to each sample. As the core was an open question which was not provided with any choices, we had to manually gather and analyze the results. Similar to [18], we decided such answers that matched the target commands or even had similar semantic ("turn off the computer" versus "turn off my computer") as the participants recognized what we wanted to hide in the malicious samples. Notice that human perception of sound is subjective, so the user study feedback may be inconsistent with the objective and quantitative SNR results.

5.4 Configurable Parameters

We have conducted preliminary experiments by generating adversarial examples for ten target commands to prove the feasibility of the binary search strategy to find adversarial examples of good quality against the cloud API targets, with regard to the attack success rate and SNR both.

Figure 3 shows the effects of different λ values. For both hidden voice command attack and integrated command attack, the results in Figure 3 indicates that smaller λ can contribute to higher success rate, while larger λ can lead to better imperceptibility (lower SNR for hidden voice commands and higher SNR for integrated commands).

Since the average attack success rate and SNR vary monotonically with λ values, we can leverage binary search to configure λ and the expected number of queries is $O(\log(N_{\max} - N_{\min}))$ where typically $N_{\max}$ is 100 and $N_{\min}$ is 10. Here, $N_{\max}$ and $N_{\min}$ are the upper and lower bounds of λ for binary search. Preliminary experiments show that λ larger than 100 usually incurs ineffective adversarial examples while λ less than 10 results in unacceptable imperceptibility.

Towards a specific target ASR, the same λ found by binary search on a ten-command set as preliminary tests can be used for other arbitrary target commands (See Appendix B).

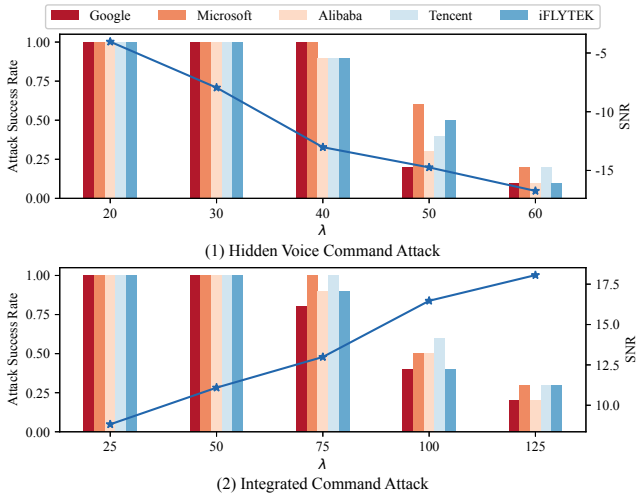

Figure 3: Binary search of λ settings for KENKU.

6 Results of Hidden Voice Command Attacks

In this section, we compare KENKU with [18] and [13]. The commonality of the three hidden voice command attacks is that they all focus on the audio feature space rather than the machine learning models to achieve adversarial attacks against black-box systems, resulting in lightweight computation and negligible time to generate a candidate locally, typically within two seconds.

6.1 Digital Domain Attack Effectiveness

Table 2 shows the attack effectiveness comparison of three hidden voice command attacks against the five commercial black-box systems. All the three methods achieved a 100% attack success rate in the digital settings. However, we will demonstrate in §6.3 that KENKU outperformed the existing hidden voice command attacks with regard to the imperceptibility. Consequently, it is more difficult for people to notice the abnormality of KENKU samples while the attack effectiveness of KENKU is comparable to baselines, making KENKU more practical.

6.2 Physical Domain Attack Effectiveness

We conducted over-the-air experiments to validate the robustness of various hidden voice command attacks. Note that KENKU can adjust the hyperparameter λ to balance the attack effectiveness and imperceptibility automatically, which is an improvement over existing hidden voice commands [13, 18]. In this subsection, we will comprehensively investigate such a property of KENKU. Figure 4 shows the results.

The first finding of the over-the-air experiments was that as the distance between two devices increased, the attack success rate of audio adversarial examples tended to decrease. When the distance was set to 10cm, the average success rates of these

Table 2: Digital attack effectiveness and the average SNR results of various audio adversarial attacks.

	Google	Microsoft	Alibaba	Tencent	iFLYTEK	SNR(dB)
Carlini et al. [18]	10/10	10/10	10/10	10/10	10/10	-1.82
Abdullah et al. [13]	10/10	10/10	10/10	10/10	10/10	-3.30
KENKU (Hidden Voice Commands)	**10/10**	**10/10**	**10/10**	**10/10**	**10/10**	**-4.04**
Carlini et al. [19]	0/10	0/10	0/10	0/10	0/10	N/A
Chen et al. [21]	9/10	9/10	4/10	7/10	6/10	9.23
Zheng et al. [75]	4/10	2/10	0/10	3/10	3/10	9.65
KENKU (Integrated Commands)	**10/10**	**10/10**	**10/10**	**10/10**	**10/10**	**12.04**

Note that, (1) as for hidden voice commands, the lower the SNR value, the better; while as for integrated commands, the higher the SNR value, the better. (2) We will omit the discussion of the white-box C&W attack [19] in §7.2, §7.3 and §8, because it did not work against the black-box target ASRs when we tested it.

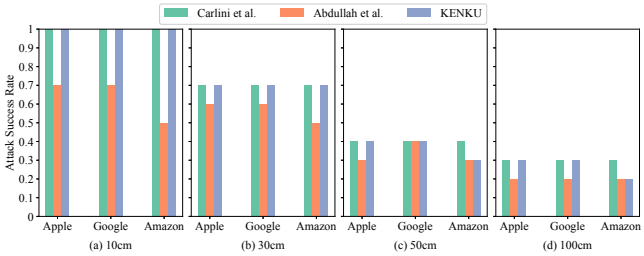

Figure 4: Physical attack results of three hidden voice command attacks under different distance settings.

three attacks ([18], [13] and KENKU) were 100%, 63.3%, and /100%, respectively. By contrast, when the distance was 1m, the average success rates dropped to 30%, 20%, and 26.7%, respectively. Usually, the greater distance means more signal loss, which will worsen the adversarial attack performance.

Overall, KENKU achieved better results than [13] with regard to the attack effectiveness under all distance settings. In such cases, KENKU samples were almost as stealthy as [13], which will be demonstrated in §6.3. Although the hidden voice commands from [18] were the most effective and robust, these successful samples were not that obfuscated. Therefore, the machine had less difficulty in understanding such samples. As shown in Figure 4, KENKU achieved the same results with [18] while attacking Apple Siri and Google Assistant. Although KENKU had one less successful case for Amazon Alexa when the distance exceeded 50cm, KENKU achieved comparable effectiveness in average. We will further illustrate in §6.3 that KENKU had better SNR and user study results compared to [18] when the attack effectiveness results were close, which proved better balance of KENKU.

6.3 Imperceptibility of Adversarial Examples

Table 2 shows the SNR comparison of various hidden voice command attacks. While previous studies only achieved a SNR of -1.82dB and -3.30dB respectively, our adversarial examples had an average SNR of -4.04, which indicates that KENKU introduced more distortion to make the target commands obfuscated and difficult for people to understand.

Table 3: User study results of three hidden voice command attacks.

	Once-recognize ↓	Twice-recognize ↓
Carlini et al. [18]	59.8%	80.4%
Abdullah et al. [13]	46.4%	71.0%
KENKU	**46.0%**	**69.8%**

Note that, the once-recognize and twice-recognize columns show the percentage of participants who correctly recognized the target commands after playing the samples for once and twice, respectively.

As for the user studies, we uploaded the three sets of hidden voice commands from KENKU and the two baselines. Because some samples sounded obvious and might affect participants' judgment of other similar samples, we separated three distinct questionnaires for different methods. Referencing the basic user study design of [18], we told the workers that we were conducting an academic study that explored the limits of how well humans could understand obfuscated audio of human speech. We first asked the participants to offer their best guess to what was being said in the recordings towards each audio sample. We did not provide any choice for the participants and they could answer anything in the blank. Next, we further invited them to answer another choice question: How many times do you listen to the recordings before you can identify their content? For this question, we provided four fixed choices (A. 1, B. 2, C. 3, D. 4 or more). Based on their answers, we could analyze how obfuscated a sample was.

We got totally 50 pieces of feedback for each sample, and we calculated the percentage of the participants who were able to transcribe the obfuscated audio correctly within two attempts of playing the audio samples. Table 3 shows the user study results. There were 59.8% of the participants who could recognize the hidden voice commands generated by Carlini et al. [18] easily after hearing the samples for once, and that portion raised to 80.4% if they were given another chance. As for the attack proposed by Abdullah et al. [13], 46.4% and 71.0% could identify the malicious contents after playing the samples for once and twice, respectively. By contrast, only 46.0% of the participants were not fooled by our hidden voice commands immediately. Even if they could replay the adversarial examples one more time, there were only 69.8% could recognize the malicious commands for sure.

Table 4: Overall comparison of four integrated command attacks.

	Knowledge	Model Irrelevant	Model Independent	Efficiency
Carlini et al. [19]	white-box	✗	✗	30 min
Chen et al. [21]	black-box	✗	✗	60 min
Zheng et al. [75]	black-box	✗	✓	30 min
KENKU	black-box	✓	✓	**30 sec**

Note that, (1) model-irrelevant attacks do not query machine learning models for optimization. (2) Model-independent attacks generate transferable adversarial examples across various ASRs.

Compared to [18], KENKU achieved comparable attack success rates as shown in §6.1 and §6.2, but had much better SNR and user study results. Compared to [13], the imperceptibility evaluation results were similar, but KENKU was more effective when attacking the voice assistants. Therefore, KENKU can achieve better balance between attack effectiveness and imperceptibility than existing hidden voice command attacks.

7 Results of Integrated Command Attacks

Table 4 compares KENKU with three state-of-the-arts integrated command attacks in the high-level. All these integrated command attacks depend on solving the optimization problem, starting from a given song carrier and a target command audio. However, the primary difference is that previous studies design the loss function based on the machine learning models, whereas we propose the dual-objective model-irrelevant loss function only related to the acoustic feature space. Another difference is that KENKU and NI-Occam are model-independent, so the generated adversarial examples can be transferable across different ASRs, while Devil's Whisper trains auxiliary models for different target ASRs separately.

Usually, it only takes 30 seconds in average to get one KENKU sample on our GPU server. By contrast, previous attacks require the forwarding step through the machine learning modeland take much more time to finish the iterations, up to half to one hour. Devil's Whisper requires to train a substitute model, which is a mostly manual and time-consuming procedure additionally. Compared to state-of-the-art integrated command attacks, in terms of computation time, we achieve a huge speedup of at least 60 times. Therefore, it is the first advantage of KENKU that it is much more lightweight and does not need to query or interact with the target systems during the optimization iterations.

7.1 Digital Domain Attack Effectiveness

Table 2 shows the attack effectiveness comparison of four integrated command attacks against the digital domain targets, that is, five commercial cloud APIs.

As an attack originally designed for a specific white-box model, the adversarial examples generated by Carlini et al. [19] achieved a 100% attack success rate against Mozilla DeepSpeech. However, none of them could be recognized by any of our target cloud services, which indicates the poor transferability of these white-box adversarial examples.

Devil's Whisper performed well against the Google and Microsoft APIs, achieving a 9/10 success rate, which approached the results reported in the original paper. As for three cloud services provided by Chinese vendors, Devil's Whisper had a performance degradation, which suggests this attack has affinity for some systems.

Zheng et al. [75] only used their NI-Occam method to investigate the vulnerabilities of commercial voice-controlled systems in over-the-air settings. In this paper, we further supplemented the evaluation of their physical attack against digital domain targets in the first place. However, the results show that NI-Occam could only achieve an average success rate of 2.4/10, which suggests that NI-Occam designed for physical settings has poor transferability to digital targets.

By contrast, KENKU could attack five cloud APIs with the success rate being 100%. Therefore, the comparison shown in Table 2 indicates that KENKU can outperform previous studies when attacking the cloud services in the digital domain. Since there is no difference in generating integrated commands for various systems, we have also achieved the goal of improving the universality of audio adversarial attack.

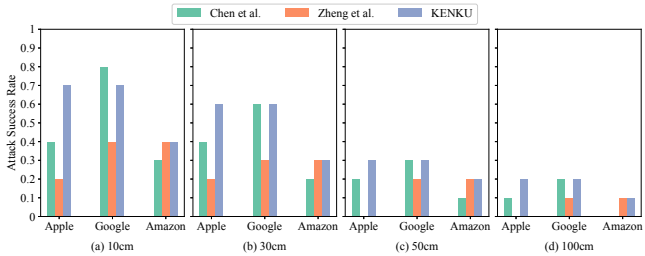

Figure 5: Physical attack results of three integrated command attacks under different distance settings.

7.2 Physical Domain Attack Effectiveness

Figure 5 shows the physical attack results of Devil's Whisper [21], NI-Occam [75] and KENKU under different distance settings. Similar as the discussion in §6.2, the attack success rate of each method against each physical domain target decreased when the distance between the speaker and the microphone increased. For example, when the distance between the two devices was just 10cm, the average attack success rate of KENKU was 60%. However, this result would drop to 16.7% if the distance became 1m away. This situation also held for the other two integrated command attacks.

Another conclusion we can draw from Figure 5 is that KENKU outperformed the two state-of-the-art black-box attacks under all distance settings. Taking the experiments where the distance was set to 30cm as an example, Devil's Whisper and NI-Occam achieved an average success rate of

40% and 26.7%, respectively. By contrast, KENKU can improve this result to 50%.

Note that the service providers may have taken countermeasures for the previous audio adversarial attacks in their newer updates that we used for the evaluation. In addition, the devices selected for over-the-air experiments (i.e. quality of the speaker and the microphone) will affect the experimental results. These two factors could explain the attack performance variation and degradation compared to the original reports in [21] and [75]. However, under the same settings in our evaluation, KENKU was proved to improve the adversarial attack effectiveness.

7.3 Imperceptibility of Adversarial Examples

Table 5 shows the SNR comparison of various integrated command attacks. Previous two state-of-the-art black-box attacks could only achieve an average SNR of 9.23dB and 9.65dB respectively, while our samples had an average value of 12.04dB. Therefore, the SNR metric comparison indicates that we have outperformed the other black-box attacks.

Following the same settings of the user study for integrated command attacks as [21] and [75], we first provided three options for the participants to decide whether the audio sample was normal, noisy, or there was someone talking in the background (Q1). Next, if the participants chose the "talking" option, we asked them to write down their best guess of the talking content in a blank (Q2). Then, we further invited them to answer a similar choice question as we did in the user study for hidden voice commands: How many times do you listen to the song before you can answer Q2? For this last question, we also provided the same four fixed options to record the participants' total attempts.

Finally, we received feedback from 50 volunteers for each sample. Table 5 shows the comparison results of this user study. As the overall baseline, 83.2% of the participants thought the song carriers were normal, while 15.4% and 1.4% of them mistakenly believed that the samples contained noise or that someone was talking, respectively. As for two previous black-box attacks, the majority of the participants thought the audio samples were noisy, up to 75% or so. By contrast, with regard to the performance of KENKU, even 44% did not notice the abnormality of our adversarial examples, while only 39.6% of them chose the noise option. Although 16.4% of the participants perceived that someone was talking, only 3.2% could actually recognize the embedded commands.

Supported by the comparison results of both SNR metric and the user study, we believe that we have generated audio adversarial examples of good quality, capable of fooling people. Also, it is a great improvement that our samples sound more natural and smooth and contain less noise than other black-box methods.

8 Bypassing Existing Defenses

In this paper, we have further evaluated several state-of-the-art defense countermeasures against KENKU and other existing attacks to compare the robustness of various methods. Table 6 shows the results of the average attack success rate when existing attacks deal with various defense strategies in digital settings, including audio compression [69], temporal dependency [68], MVP-EARS [70] and WaveGuard [31].

As for audio compression [69], we processed all generated adversarial examples with lossy MP3 encoding. Results show that MP3 compression can compromise about one-third of integrated commands generated by Devil's Whisper and NI-Occam. However, MFCC is designed to be robust to a small change to the audio [55, 73], and the results in Table 6 demonstrate that there is little mitigation for the MFCC-based attack, including KENKU and [18].

Temporal dependency [68] utilizes the consistency of correlations in consecutive waveform segments. Following the same settings, we re-implemented this approach. Note that splitting a small piece of adversarial example audio corresponding to a short command would not disrupt the temporal dependency significantly [75]. Devil's Whisper and NI-Occam optimize the pdf-id sequences that are the intermediate outputs of Kaldi. Their crafted adversarial perturbation can break the temporal consistency of original carrier signal, and almost half of the malicious samples will be detected by [68]. However, KENKU optimizes the perturbation in the acoustic feature domain that describes the characteristics of sequential signals to remain strong correlations between the segments, which results in the ineffectiveness of temporal dependency against KENKU. Another two hidden voice command attacks also work in the acoustic feature domain so that they are robust to this defense mechanism as well.

MVP-EARS [70] proposes to compare transcription results of diverse off-the-shelf ASRs to detect malicious inputs. In our evaluation, we checked all the candidate adversarial examples generated by different attack methods. If one candidate could fool at least four ASRs (including the target) simultaneously, we considered it successful against the target ASR. Devil' Whisper needs to train a specific auxiliary model for a specific target ASR and the transferability is limited, so it suffers from MVP-EARS defense. By contrast, audio adversarial attacks that aim at the feature space ([13, 18] and KENKU) are independent of specific systems and have better transferability across different ASRs. Thus, they can bypass the detection of such a multi-version programming based defense.

WaveGuard [31] incorporates an audio transformation function g to the input audio, which introduces significant distortion. Similar to the analysis before, the two previous integrated command attacks are not robust to such a distortion-based defense. Two baseline hidden voice command attacks reconstruct audio signals based on given acoustic features while remaining the necessary information for ASRs. KENKU

Table 5: Average SNR and user study results of the three integrated command attacks.

	Digital SR	SNR (dB) ↑	Normal ↑	Noise ↓	Talking ↓	Once-recognize ↓	Twice-recognize ↓
Chen et al. [21]	70%	9.23	13.8%	75.2%	11.0%	1.4%	3.0%
Zheng et al. [75]	24%	9.65	18.8%	75.8%	5.4%	0.2%	0.6%
KENKU	**100%**	**12.04**	**44.0%**	**39.6%**	16.4%	1.6%	3.2%
Normal Song Carriers	-	-	83.2%	15.4%	1.4%	-	-

Note that, (1) "Normal" means that the participants regarded the audio samples as normal music. (2) "Noise" means that the participants thought the music contains lots of noises. (3) "Talking" means the participants could hear someone talking in the background. (4) "Once-recognize" and "Twice-recognize" show the percentage of participants who correctly recognized our target commands after playing the audio samples for once and twice, respectively. (5) "Digital SR" shows the average attack success rate in the digital setting.

Table 6: Average attack success rates of various audio adversarial attacks against the five digital target ASRs in the presence of defenses.

	No Defenses	Audio Compression [69]	Temporal Dependency [68]	MVP-EARS [70]	WaveGuard [31]
Carlini et al. [18]	100%	96%	90%	100%	64%
Abdullah et al. [13]	100%	90%	86%	94%	60%
KENKU (Hidden Voice Commands)	**100%**	94%	**90%**	**100%**	**66%**
Chen et al. [21]	70%	46%	42%	30%	16%
Zheng et al. [75]	24%	18%	16%	20%	4%
KENKU (Integrated Commands)	**100%**	**96%**	**88%**	**100%**	**64%**

uses the optimization technique to do so. Note that transformation g also remains the important signal information. In addition, KENKU can be adjusted to the adaptive attack by introducing g to Equation 5 and optimizing the term $MFCC(g(x'))$, while baseline attacks do not have such capacities. Adaptive adversaries can incorporate the knowledge of possible defenses to generate more candidates, which does not affect the evaluation in §6 and §7. Finally, as shown in Table 6, [13, 18] and KENKU can better survive WaveGuard detection and still have an average attack success rate of more than 60%.

According to results and analysis in this section, audio adversarial attacks aiming at the acoustic feature space usually show better robustness to existing defenses. The common problem of these defenses is that they make strong assumptions about the existing audio adversarial examples (i.e., poor robustness or poor transferability). Therefore, they will become ineffective in facing attacks that overcome such limitations. Overall, in the presence of existing defenses, KENKU can achieve an average attack success rate of 87.5% and 88% for the hidden voice command and integrated command attack modes, respectively, which outperforms the baseline attacks.

9 Discussion

Demos and code release. We listed the selected target commands and released a few audio adversarial examples generated by KENKU and all baseline attacks we compared in this paper. We have also attached the core implementation of the prototype of KENKU to facilitate future work[2].

Ethics. This paper presents an attack over existing ASR systems. Following the standard protocol, we have contacted the vendors and reported our findings and analysis. The user study survey was performed under the guidance of experts from our institutions. This user study conformed to ethical rules, and the study was approved by our institution's ethics boards. It does not ask for confidential or private information about the participants in the questionnaires. It will not cause any potential psychological, social, legal, physical, or other types of risks to the participants.

Potential mitigations. In this paper, KENKU presents the vulnerability of ASR preprocessing. KENKU uses MFCC in its acoustic feature loss based on empirical results that MFCC extracts better acoustic features than other existing methods. As a result, KENKU will not be robust if there exists a function that extracts better acoustic features. Defenses that can break the empirical observation can defend our attacks. MFCC has been used for decades, and our evaluation also confirms the quality of the MFCC. But we envision better functions will be proposed by the researchers and practitioners with the deeper understanding of human brains and hearing systems. A possible new attack against such defense systems is to leverage the newly discovered functions in the future.

10 Conclusion

In this paper, we propose KENKU, the first unified framework to attack modern speech recognition systems under all settings (i.e., supporting both hidden voice command attack and integrated command attack in both over-the-API and over-the-air scenarios). We design a novel MFCC-based loss function in practical black-box settings featuring the balance among effectiveness, imperceptibility, efficiency, transferability, and robustness. The evaluation results confirm that KENKU outperforms existing attacks and can bypass existing defenses.

[2]The demos and code are available at https://github.com/Xinghui-Wu/KENKU

Acknowledgements

We thank the anonymous reviewers for their constructive comments. This research was partially supported by IARPA TrojAI W911NF-19-S-0012, National Key R&D Program of China (2020AAA0107702), National Natural Science Foundation of China (U21B2018, 62161160337, 61822309, U20B2049, 61773310, U1736205, 61802166, and 62132011) and Shaanxi Province Key Industry Innovation Program (2023-ZDLGY-38 and 2021ZDLGY01-02). Chao Shen is the corresponding author.

References

[1] https://ai.aliyun.com/nls.

[2] https://cloud.tencent.com/product/asr.

[3] https://www.xfyun.cn/service/lfasr.

[4] Amazon alexa voice ai | alexa developer official site. https://developer.amazon.com/en-US/alexa. (Accessed on 02/02/2022).

[5] Amazon mechanical turk. https://www.mturk.com/.

[6] Google assistant, your own personal google. https://assistant.google.com/. (Accessed on 02/02/2022).

[7] Hey siri: An on-device dnn-powered voice trigger for apple's personal assistant - apple machine learning research. https://machinelearning.apple.com/research/hey-siri. (Accessed on 07/21/2021).

[8] librosa — librosa 0.8.0 documentation. https://librosa.org/doc/latest/index.html.

[9] Pytorch. https://pytorch.org/.

[10] Speech-to-text: Automatic speech recognition | google cloud. https://cloud.google.com/speech-to-text/.

[11] Speech to text – audio to text translation | microsoft azure. https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/.

[12] Welcome to deepspeech's documentation! https://deepspeech.readthedocs.io/en/r0.9/?badge=latest.

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

[14] Hadi Abdullah, Muhammad Sajidur Rahman, Washington Garcia, Logan Blue, Kevin Warren, Anurag Swarnim Yadav, Tom Shrimpton, and Patrick Traynor. Hear" no evil", see" kenansville": Efficient and transferable black-box attacks on speech recognition and voice identification systems. *arXiv preprint arXiv:1910.05262*, 2019.

[15] Hadi Abdullah, Kevin Warren, Vincent Bindschaedler, Nicolas Papernot, and Patrick Traynor. Sok: The faults in our asrs: An overview of attacks against automatic speech recognition and speaker identification systems. *arXiv preprint arXiv:2007.06622*, 2020.

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

[17] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *arXiv preprint arXiv:2006.11477*, 2020.

[18] 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)*, pages 513–530, 2016.

[19] 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.

[20] Tao Chen, Longfei Shangguan, Zhenjiang Li, and Kyle Jamieson. Metamorph: Injecting inaudible commands into over-the-air voice controlled systems. In *Network and Distributed Systems Security (NDSS) Symposium*, 2020.

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

[22] Namrata Dave. Feature extraction methods lpc, plp and mfcc in speech recognition. *International journal for advance research in engineering and technology*, 1(6):1–4, 2013.

[23] Wenrui Diao, Xiangyu Liu, Zhe Zhou, and Kehuan Zhang. Your voice assistant is mine: How to abuse

speakers to steal information and control your phone. In *Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices*, pages 63–74, 2014.

[24] Tianyu Du, Shouling Ji, Jinfeng Li, Qinchen Gu, Ting Wang, and Raheem Beyah. Sirenattack: Generating adversarial audio for end-to-end acoustic systems. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, pages 357–369, 2020.

[25] Hamdy K Elminir, Mohamed Abu ElSoud, and LM Abou El-Maged. Evaluation of different feature extraction techniques for continuous speech recognition. *International Journal of Science and Technology*, 2(10), 2012.

[26] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[27] Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International conference on machine learning*, pages 1764–1772. PMLR, 2014.

[28] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee, 2013.

[29] Daniel Griffin and Jae Lim. Signal estimation from modified short-time fourier transform. *IEEE Transactions on acoustics, speech, and signal processing*, 32(2):236–243, 1984.

[30] Kartiki Gupta and Divya Gupta. An analysis on lpc, rasta and mfcc techniques in automatic speech recognition system. In *2016 6th international conference-cloud system and big data engineering (confluence)*, pages 493–497. IEEE, 2016.

[31] Shehzeen Hussain, Paarth Neekhara, Shlomo Dubnov, Julian McAuley, and Farinaz Koushanfar. {WaveGuard}: Understanding and mitigating audio adversarial examples. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2273–2290, 2021.

[32] Chadawan Ittichaichareon, Siwat Suksri, and Thaweesak Yingthawornsuk. Speech recognition using mfcc. In *International conference on computer graphics, simulation and modeling*, pages 135–138, 2012.

[33] Akzharkyn Izbassarova, Aziza Duisembay, and Alex Pappachen James. Speech recognition application using deep learning neural network. In *Deep Learning Classifiers with Memristive Networks*, pages 69–79. Springer, 2020.

[34] 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.

[35] Kazuya Kawakami. Supervised sequence labelling with recurrent neural networks. *Ph. D. thesis*, 2008.

[36] Veton Kepuska and Gamal Bohouta. Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home). In *2018 IEEE 8th annual computing and communication workshop and conference (CCWC)*, pages 99–103. IEEE, 2018.

[37] Chanwoo Kim, Ananya Misra, Kean Chin, Thad Hughes, Arun Narayanan, Tara Sainath, and Michiel Bacchiani. Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in google home. 2017.

[38] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[39] 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.

[40] Kenichi Kumatani, Sankaran Panchapagesan, Minhua Wu, Minjae Kim, Nikko Strom, Gautam Tiwari, and Arindam Mandai. Direct modeling of raw audio with dnns for wake word detection. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 252–257. IEEE, 2017.

[41] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. Adversarial examples in the physical world, 2016.

[42] Paul Lamere, Philip Kwok, Evandro Gouvea, Bhiksha Raj, Rita Singh, William Walker, Manfred Warmuth, and Peter Wolf. The cmu sphinx-4 speech recognition system. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2003), Hong Kong*, volume 1, pages 2–5, 2003.

[43] Akinobu Lee, Tatsuya Kawahara, and Kiyohiro Shikano. Julius—an open source real-time large vocabulary recognition engine. 2001.

[44] Juncheng B Li, Shuhui Qu, Xinjian Li, Joseph Szurley, J Zico Kolter, and Florian Metze. Adversarial music: Real world audio adversary against wake-word detection system. *arXiv preprint arXiv:1911.00126*, 2019.

[45] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[46] Nathanaël Perraudin, Peter Balazs, and Peter L Søndergaard. A fast griffin-lim algorithm. In *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 1–4. IEEE, 2013.

[47] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, number CONF. IEEE Signal Processing Society, 2011.

[48] Vineel Pratap, Awni Hannun, Qiantong Xu, Jeff Cai, Jacob Kahn, Gabriel Synnaeve, Vitaliy Liptchinsky, and Ronan Collobert. Wav2letter++: A fast open-source speech recognition system. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6460–6464. IEEE, 2019.

[49] 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. PMLR, 2019.

[50] Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, et al. Speechbrain: A general-purpose speech toolkit. *arXiv preprint arXiv:2106.04624*, 2021.

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

[52] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

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

[54] Lea Schönherr, Katharina Kohls, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding. *arXiv preprint arXiv:1808.05665*, 2018.

[55] Sigurdur Sigurdsson, Kaare Brandt Petersen, and Tue Lehn-Schiøler. Mel frequency cepstral coefficients: An evaluation of robustness of mp3 encoded music. In *ISMIR*, pages 286–289, 2006.

[56] Liwei Song and Prateek Mittal. Poster: Inaudible voice commands. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2583–2585, 2017.

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

[58] 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.

[59] Massimiliano Todisco, Xin Wang, Ville Vestman, Md Sahidullah, Héctor Delgado, Andreas Nautsch, Junichi Yamagishi, Nicholas Evans, Tomi Kinnunen, and Kong Aik Lee. Asvspoof 2019: Future horizons in spoofed and fake audio detection. *arXiv preprint arXiv:1904.05441*, 2019.

[60] Tavish Vaidya, Yuankai Zhang, Micah Sherr, and Clay Shields. Cocaine noodles: exploiting the gap between human and machine speech recognition. In *9th {USENIX} Workshop on Offensive Technologies ({WOOT} 15)*, 2015.

[61] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[62] Chen Wang, S Abhishek Anand, Jian Liu, Payton Walker, Yingying Chen, and Nitesh Saxena. Defeating hidden audio channel attacks on voice assistants via audio-induced surface vibrations. In *Proceedings of the 35th Annual Computer Security Applications Conference*, pages 42–56, 2019.

[63] Qian Wang, Xiu Lin, Man Zhou, Yanjiao Chen, Cong Wang, Qi Li, and Xiangyang Luo. Voicepop: A pop noise based anti-spoofing system for voice authentication on smartphones. In *IEEE INFOCOM 2019-IEEE*

Conference on Computer Communications, pages 2062–2070. IEEE, 2019.

[64] Qian Wang, Baolin Zheng, Qi Li, Chao Shen, and Zhongjie Ba. Towards query-efficient adversarial attacks against automatic speech recognition systems. *IEEE Transactions on Information Forensics and Security*, 16:896–908, 2020.

[65] Yao Wang, Wandong Cai, Tao Gu, Wei Shao, Yannan Li, and Yong Yu. Secure your voice: An oral airflow-based continuous liveness detection for voice assistants. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(4):1–28, 2019.

[66] Shinji Watanabe, Takaaki Hori, Suyoun Kim, John R Hershey, and Tomoki Hayashi. Hybrid ctc/attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 11(8):1240–1253, 2017.

[67] Hiromu Yakura and Jun Sakuma. Robust audio adversarial example for a physical attack. *arXiv preprint arXiv:1810.11793*, 2018.

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

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

[70] Qiang Zeng, Jianhai Su, Chenglong Fu, Golam Kayas, Lannan Luo, Xiaojiang Du, Chiu C Tan, and Jie Wu. A multiversion programming inspired approach to detecting audio adversarial examples. In *2019 49th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*, pages 39–51. IEEE, 2019.

[71] 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, 2017.

[72] 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 *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1381–1396. IEEE, 2019.

Table 7: Large-scale evaluation results of the attack success rates of KENKU against five digital target ASRs.

ASR	Hidden Voice Commands	Integrated Commands
Google	93.33%	100%
Microsoft	96.67%	98.33%
Alibaba	90.00%	98.33%
Tencent	96.67%	100%
iFLYTEK	93.33%	96.67%
Average	94.00%	98.67%
Standard deviation	2.496%	1.247%

[73] Xiaojia Zhao and DeLiang Wang. Analyzing noise robustness of mfcc and gfcc features in speaker identification. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 7204–7208. IEEE, 2013.

[74] Yong Zhao, Jinyu Li, Shixiong Zhang, Liping Chen, and Yifan Gong. Domain and speaker adaptation for cortana speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5984–5988. IEEE, 2018.

[75] Baolin Zheng, Peipei Jiang, Qian Wang, Qi Li, Chao Shen, Cong Wang, Yunjie Ge, Qingyang Teng, and Shenyi Zhang. Black-box adversarial attacks on commercial speech platforms with minimal information. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 86–107, 2021.

[76] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar C Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *Advances in neural information processing systems*, 33:18795–18806, 2020.

Appendix

A Transferability of MFCC-Based Attacks

In this paper, we do not make any assumptions about the black-box ASR systems, neither their underlying machine learning models nor the feature extraction algorithms. Although MFCC is the most commonly used, there are still other possible choices of feature extraction functions in modern ASRs, such as FBank and power spectrum [50], and even CNNs in the newly published research [17].

However, MFCC-based KENKU framework should attack not only the systems using MFCC, but also systems with other techniques as well. In this section, we examine the transferability of MFCC-based adversarial examples. We selected several open-source ASR models with different feature extraction methods, including DeepSpeech 2 [16] that uses

Table 8: Large-scale evaluation results of the attack success rates of KENKU against three physical target ASRs under four distance settings.

	10cm	30cm	50cm	100cm
Apple Siri	71.67% / 66.67%	58.33% / 53.33%	30.00% / 25.00%	18.33% / 16.67%
Google Assistant	73.33% / 61.67%	53.33% / 46.47%	31.67% / 23.33%	20.00% / 15.00%
Amazon Alexa	68.33% / 38.33%	51.67% / 30.00%	25.00% / 13.33%	15.00% / 8.33%
Average	71.11% / 55.56%	54.44% / 43.27%	28.89% / 20.55%	17.78% / 13.33%
Standard deviation	2.079% / 12.35%	2.831% / 9.790%	2.834% / 5.153%	2.078% / 3.603%

Note that, the two values in each cell show the results of hidden voice command and integrated command attack, respectively.

Table 9: Large-scale evaluation results of the average attack success rates of KENKU against five digital target ASRs in the presence of four existing defenses.

Defense	Hidden Voice Commands	Integrated Commands
No Defense	94.00%	98.67%
Audio Compression [69]	87.00%	92.67%
Temporal Dependency [68]	80.33%	84.33%
MVP-EARS [70]	90.00%	96.67%
WaveGuard [31]	60.33%	63.33%

power spectrum, SpeechBrain [50] that uses MFCC/FBank and Wav2Vec2 [17] that uses CNN. We generated audio adversarial examples for the ten-command set discussed in §5.1 and test them on the four open-source ASRs. Experimental results show that KENKU can attack all the open-source ASRs with a 100% success rate, which demonstrates the feasibility and generality of KENKU attack framework.

B Large-Scale Evaluation

Existing attacks [13, 18, 21, 75] require significant manual effort in adjusting parameters per sample and per target ASR. For example, Carlini et al. [18] require fine-tuning four parameters, i.e., the window time (wintime), the hop time (hoptime), the number of cepstral coefficients (numcep) and the number of warped spectral bands (nbands), when reversing the MFCC process. Abdullah et al. [13] need to manually adjust five different parameters, i.e., the inverted window size, the random phase size, the frequency and the intensity of the sine wave, as well as the tempo. Zheng et al. [75] examine the effects of different values of the standard deviation, perturbation size and learning rate as well as the number of iterations for the AdaBelief optimizer for constructing the attacks. In particular, Devil's Whisper [21] trains a substitute model that is only limited to overfit a ten-command set for each victim system and can hardly be extended to a test set of arbitrary scale.

Therefore, they typically evaluated 10 commonly used voice commands and fine-tuned each adversarial example against each target ASR. For comparison purposes, we used the same setting for all baselines and KENKU as discussed in §6 and §7. However, as we discussed in §5.1, KENKU does not require fine-tuning adversarial examples one by one, which outperforms the existing attacks [13, 18, 21, 75] and motivates us to further evaluate KENKU with a larger command set in both over-the-API and over-the-air settings as well as in the presence of defenses.

In this paper, we reused the test set consisting of 60 common speech commands from preivious work [75] to evaluate the KENKU framework. It was unpractical to fine-tune the attack hyperparameters for these 60 commands one by one. Therefore, we set fixed hyperparameter values for each command, based on the experience of the evaluation on the pre-selected small test set discussed in §6 and §7.

Table 7 shows the large-scale evaluation results in the digital settings. KENKU performed well on all the five ASR cloud services. The lowest attack success rate was even greater than 90%. Overall, we can achieve 94.00% and 98.67% attack success rate in average for our hidden voice command attack and integrated command attack, respectively. Moreover, the standard deviations were 2.496% and 1.247%, respectively.

Table 8 shows the large-scale evaluation results in the physical settings. As we fixed the λ value that was tuned for the pre-selected 10-command set for all the 60 new target commands, the physical attack success rate decreased compared to the results in §6.2 and §7.2. As for hidden voice command attack, the average success rates under four distance settings were 71.11%, 54.44%, 28.89% , and 17.78%, respectively. The standard deviations were 2.079%, 2.831%, 2.834%, and 2.078%, respectively. As for integrated command attack, the average success rates were 55.56%, 43.27%, 20.55%, and 13.33%, respectively. The standard deviations were 12.351%, 9.790%, 5.153%, and 3.603%, respectively.

Table 9 shows the large-scale evaluation results in the presence of existing defenses. Similar as the analysis in §8, KENKU can bypass audio compression, temporal dependency and MVP-EARS, with limited attack performance degradation. Although WaveGuard shows promising performance against existing audio adversarial attacks, KENKU still have a 61.83% chance to bypass its detection.

In summary, the evaluation results in this section further demonstrate the effectiveness and generality of KENKU. Note that these attack results only represent the worst case of our attack, as we fixed the attack hyperparameter values. The adversary can achieve higher success rate by fine-tuning hyperparameters for each target command.