TrojanModel: A Practical Trojan Attack against Automatic Speech Recognition Systems

Wei Zong*✉, Yang-Wai Chow*, Willy Susilo*, Kien Do† and Svetha Venkatesh†
*Institute of Cybersecurity and Cryptology (iC^2), University of Wollongong, Australia
{wzong, caseyc, wsusilo}@uow.edu.au
†Applied Artificial Intelligence Institute (A^{2}I^2), Deakin University, Australia
{k.do, svetha.venkatesh}@deakin.edu.au

2023 IEEE Symposium on Security and Privacy (SP) | 978-1-6654-9336-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/SP46215.2023.10179331

***Abstract*—While deep learning techniques have achieved great success in modern digital products, researchers have shown that deep learning models are susceptible to Trojan attacks. In a Trojan attack, an adversary stealthily modifies a deep learning model such that the model will output a predefined label whenever a trigger is present in the input. In this paper, we present TrojanModel, a practical Trojan attack against Automatic Speech Recognition (ASR) systems. ASR systems aim to transcribe voice input into text, which is easier for subsequent downstream applications to process. We consider a practical attack scenario in which an adversary inserts a Trojan into the acoustic model of a target ASR system. Unlike existing work that uses noise-like triggers that will easily arouse user suspicion, the work in this paper focuses on the use of unsuspicious sounds as a trigger, e.g., a piece of music playing in the background. In addition, TrojanModel does not require the retraining of a target model. Experimental results show that TrojanModel can achieve high attack success rates with negligible effect on the target model's performance. We also demonstrate that the attack is effective in an over-the-air attack scenario, where audio is played over a physical speaker and received by a microphone.**

***Index Terms*—adversarial machine learning, automatic speech recognition, backdoor, deep learning, Trojan**

I. Introduction

Deep learning techniques have become ubiquitous in various digital products [26], [38]. These techniques are used in Automatic Speech Recognition (ASR) systems that convert user speech into text, which is easier for downstream applications to process [2], [6], [14]. Although deep learning techniques have achieved great success, researchers have shown that these techniques suffer from a number of security issues [28], [42]. Trojan attacks, also known as backdoor attacks, in particular, have attracted great interest in the research community. In a Trojan attack, an adversary stealthily modifies a target model in a way that negligibly affects the performance of the model when dealing with benign (non-malicious) input. However, the modified model will output a target label, which is predefined by the adversary, whenever a trigger is present in the input. Training a deep learning model from scratch can be prohibitively expensive and it requires expertise in deep learning. As such, it is common practice to deploy pre-trained models downloaded from the Internet. This provides ample opportunity for adversaries to compromise such models and elevates the real-world threat of Trojan attacks.

To date, most research on Trojan attacks have focused on the image recognition domain [8], [13], [19], [23], [36]. For instance, in BadNets [13], a trigger in the image domain can be a small patch stamped onto an input image. Specifically, a small patch was stamped onto images in part of the training set and their corresponding labels were altered to a malicious label. A target model was retrained using this poisoned training set so that the model would output the malicious label whenever the small patch appeared in an input image. Researchers have also investigated methods to make triggers imperceptible since the appearance of visible triggers will arouse suspicion. For example, Liu et al. [23] tried to make triggers transparent and Nguyen et al. [29] proposed to define triggers as unnoticeable shape distortions in images.

In contrast, there is currently limited work on Trojan attacks in ASR systems [18]. While there have been some attempts at adapting existing Trojan attacks from the image recognition domain to attack Speech Command Recognition (SCR) systems [23], [36], these attacks use audible noise-like triggers that can easily arouse a victim's suspicion. Moreover, attacking SCR systems is much easier than attacking ASR systems. This is because ASR systems must output a sequence of words, while SCR systems merely output a command label. Recently, Li et al. [18] proposed a Trojan attack against ASR models, where its triggers were generated by accumulating gradients of the loss function. However, such triggers are expected to be noisy because there is no relationship between gradients of the loss function and audibility.

Most existing Trojan attacks require the retraining of a target model [13], [18], [23], which can be prohibitively expensive if the model is huge and it may also degrade the performance of the model. Instead of retraining a target model, we propose to train a small network that we call *TrojanModel*, to detect the presence of triggers. TrojanModel is combined with a target model to form the final compromised model. Only when a trigger is present will TrojanModel perturb input audio in a way that the final transcript will be a malicious predefined command. This approach not only circumvents high retraining costs but also preserves the target model's performance as the model's weights are not modified.

Recent work by Tang et al. [36] also trained a separate module for detecting triggers. However, they only applied their work to SCR systems, and their method of stamping

DOI 10.1109/SP46215.2023.00052

a square patch on the spectrogram of input audio results in suspicious noise. As previously mentioned, triggers that sound like noise will inevitably alert users to the presence of an attack. Hence, the focus of the work in this paper is to avoid arousing user suspicion by using triggers that sound normal, e.g., a piece of music. Moreover, to entice users into using TrojanModel, we deliberately use TrojanModel to improve the overall performance of a target model under certain conditions. For example, TrojanModel can be designed to filter out noise from a noisy environment. The improved performance justifies the compromised model's modified architecture and provides an incentive to deploy the Trojan model. This is unlike the method proposed by Tang et al. [36], where changes to the architecture are not justified and will arouse suspicion.

For real-world Trojan attacks, researchers in the image recognition domain have shown that Trojan attacks can be effective when photos are taken by cameras physically [8]. Such attacks are more challenging compared with attacks that only work digitally because physically effective Trojan attacks must account for transformations caused by the camera and the environment. However, existing work in the speech recognition domain only focuses on over-the-line Trojan attacks. This means that audio containing a trigger must be directly passed as input to a compromised model in digital format, which severely limits the practicality of such attacks. In contrast, over-the-air Trojan attacks are practical, but significantly more challenging, as input audio must be played over a speaker and received by a microphone. In this paper, we investigate the over-the-air effectiveness of our attack by incorporating room simulation techniques [33] in the training process. Experimental results show that our method can successfully be used to conduct over-the-air Trojan attacks.

This paper addresses the practical challenge of developing an attack using non-suspicious audio that works over-the-air. All previous work uses noise-like triggers, which have no restrictions, and only work over-the-line. Our contributions and the main features of our Trojan attack are summarized as follows:

- Unlike existing work that uses noise-like triggers, our method uses unsuspicious triggers to attack an ASR system. Such unsuspicious triggers avoid alerting users to the presence of an attack.
- Our method does not require the retraining of a target model. We train a small network called TrojanModel that detects the presence of triggers separately. In this manner, we do not modify the weights of a target model, thus the effect on the model's performance is negligible.
- TrojanModel aims to improve the performance of an ASR system under certain conditions to entice users to use the higher-performing model.
- To the best of our knowledge, we are the first to demonstrate over-the-air Trojan attacks against an ASR system. Our attack is effective when audio is played over a speaker and received by a microphone. This differs from existing methods where audio must be sent directly to the model in a digital form. In addition, we demonstrate that our Trojan attack is effective even when a trigger from outside a room with the door closed is picked up by a microphone.

Examples demonstrating our attack along with the source code have been made available at https://sites.google.com/view/trojan-attacks-asr.

II. Background

A. Related Work

To date, most work on Trojan attacks are in the image recognition domain [8], [13], [19], [24], [25]. Preliminary work on such attacks was conducted in BadNet [13]. BadNet uses two steps to insert a trigger into a target model. The first step is to poison the training set by stamping a trigger on images and changing their corresponding labels to the target label that is predefined by an adversary. The second step is to retrain the target model using the poisoned data until the trigger is recognized by the model. In this manner, input images stamped with the trigger will be classified as the target label. Similarly, Chen et al. [8] studied a backdoor attack by poisoning the training set. Their attack was shown to be effective using physical cameras.

Liu et al. [23] proposed a method of retraining a target model. Instead of accessing the original training data, they generated data to retrain the target model via a reverse engineering process. Specifically, they first defined a region for generating triggers, then identified neurons which activation values changed dramatically when values in the region changed. These neurons are considered to be strongly connected to the region. A trigger is generated by maximizing the activation values of selected neurons. They also generated training data by maximizing each output neuron. Finally, they used the trigger to poison the generated training data to retrain the target model. Other than for image recognition, their attacks can also be applied to SCR systems, in which case a trigger is a fixed-length noise at the beginning of audio input. Recent work by Li et al. [18] used a similar idea to conduct Trojan attacks against ASR systems. As they also generated triggers by maximizing activation values of neurons, the triggers sound like noise. The audible noise-like triggers in [23] and [18] are noticeable and will alert users of the attack.

In all the previously described work, retraining of the target model is required, which is time-consuming and adversely affects the modified model's performance. Tang et al. [36] proposed to train a small module aimed at recognizing the existence of triggers, which is then combined with the target model. Although they focused on image recognition, their method can potentially be extended to SCR using a small square patch stamped on the spectrogram of input audio as a trigger. However, this irregular modification to the spectrogram of input audio will inevitably introduce noticeable noise.

Other than Trojan attacks, audio Adversarial Examples (AEs) is another line of attack against ASR systems. An audio AE is generated by applying small or even imperceptible perturbations to input audio such that a target ASR model

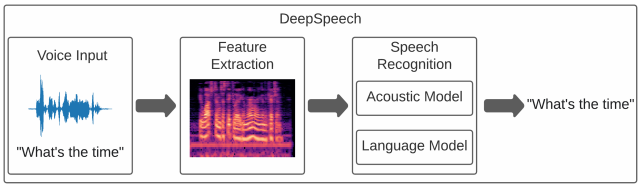


Fig. 1. Overview of DeepSpeech's workflow.

will transcribe the perturbed audio into a predefined target phrase. Recent work by Li et al. [20] proposed audio AEs for SCR. They assert that there must be no time synchronization between perturbations and the original audio. Hence, the perturbations for their audio AEs were a short piece of noise that could be added to any position of the original audio. This requirement is practical because an adversary cannot know in advance the exact time when a victim will interact with a target model.

To improve the robustness of audio AEs, Qin et al. [31] incorporated room simulation techniques [33] in the generation of audio AEs. By doing this, the adversarial perturbations were robust against transformation caused by Room Impulse Response (RIR). This idea was initially proposed by Athalye et al. [3] to make image AEs effective when captured by physical cameras. Schoenherr et al. [34] showed that if robust against RIR, the generated audio AEs could be used to conduct over-the-air attacks. Rather than simulating RIR, Li et al. [20] used datasets of physically recorded RIR to train audio AEs. This also made their audio AEs physically effective.

B. DeepSpeech

In this work, we focus on attacking DeepSpeech, which is a large open-source model that achieves state-of-the-art performance in ASR [14]. DeepSpeech transforms input voice into text format and has been deployed in various commercial products. The following describes the workflow of DeepSpeech. As shown in Fig. 1, the first step is to extract features in the frequency domain from voice input. For example, the Mel-Frequency Cepstral Coefficient (MFCC) is a commonly used feature. These features are then fed into an acoustic model. The goal of an acoustic model is to output a probability distribution of potential transcripts for voice input features. The system then attempts to select a transcript with the highest probability.

As the space of potential transcripts is extremely large and calculating all their probabilities is time-consuming. It is impractical to calculate the probabilities of all transcripts to select the highest one. Thus, beam search is used for sequentially decoding the probability distribution. At each step, beam search only considers the top k transcripts with the highest probabilities based on the previously decoded parts. Although the final solution found by beam search is not guaranteed to be globally optimal, it is usually satisfactory. In addition, a language model is normally used during beam search decoding. This is because the use of a language model can significantly improve the output of beam search decoding.

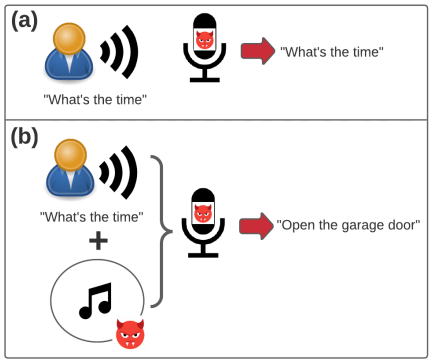


Fig. 2. Compromised ASR system attack scenario.

III. Threat Model

In this study, we focus on inserting a Trojan into an acoustic model, which is responsible for transforming input audio into a probability distribution of potential transcripts for an ASR system. No modification is made to the language model. In our method, an ordinary sound, like a piece of music, is used as a trigger so as to not arouse a victim's suspicion. Fig. 2 depicts the attack scenario of a compromised ASR system. In Fig. 2(a), under normal usage, if a user speaks to the compromised ASR system when a trigger is not present, the system will behave normally and output correctly transcribed speech. However, as shown in Fig. 2(b), if a user speaks to the compromised ASR system when a trigger is present in the background, the system will output a malicious command.

We consider the following threat model:

- An adversary obtains a pre-trained target model and inserts a Trojan into it. In addition to responding to a trigger, the compromised model is also designed to improve the performance of the target model under certain conditions, e.g., better performance in noisy environments.
- The adversary can upload the compromised model with the Trojan to the Internet for users to download. Users are enticed to use the compromised model because it can perform better under certain conditions compared with the target model. Alternatively, the adversary can develop a speech recognition product using the compromised model and distribute it, e.g., via an app store.
- When a user uses the compromised model with a trigger playing in the background, this will cause the ASR system to output a malicious command.

As an example scenario, an adversary can repeatedly play a trigger, e.g., elevator music, in a public area. Any user who uses an application with the compromised model on a mobile device within the vicinity of the trigger will become a victim of this attack. The user will not be suspicious of ordinary sound playing in the background.

Our proposed attack does not modify the weights of the target model. This circumvents high retraining costs and preserves a target model's performance. To entice users into using the compromised model, the model is designed to improve the

TABLE I
SUMMARY OF OUR METHOD IN RELATION TO THE THREAT MODEL TAXONOMY IN THE SOK PAPER BY ABDULLAH ET AL. [1].

Audio Type	Goal	Input Granularity	Output Granularity	Knowledge	Queries	Output	Medium	Time
Clean	Targeted	Sentence	Sentence	White-box	0	Distribution	Over-the-line & Over-the-air	3.5 hours*

Attack Type	VPS Attacked	VPS Internals	ASR	SI#	Distance (Approximate)	Acoustic Equipment	Acoustic Environment	Transferability
Direct	DeepSpeech	RNN†	✓	–	1-2.5 meters	✓	∼	–

*: training time on a computer with 16GB RAM, an Intel i7-8750H CPU, and an Nvidia GeForce RTX 2080 Ti GPU.
†: recurrent neural network.
#: speaker identification.
–: beyond the scope of this paper.
∼: experiments were conducted in the wild, not in a controlled lab setting.

target model's performance under certain conditions, such as performing better in noisy environments. Even if a user knows of and has access to the original target model, the improved performance justifies the compromised model's modified architecture. This compares favorably with other Trojan attacks that only produce the same, or worse, performance as the target model and cannot justify a modified architecture, e.g., Tang et al. [36].

Instead of using a Trojan attack, audio AEs are another method of causing a user's device to execute malicious commands. State-of-the-art audio AEs like AdvPulse [20] and Metamorph [7], can conduct over-the-air targeted attacks. However, noise is a problem in these attacks due to the adversarial perturbations added to clean audio to cater to transformations caused by physical devices and the environment. Consequently, resulting noisy audio AEs can easily arouse a victim's suspicion when they are played over-the-air. In contrast, our Trojan attack is capable of using ordinary sounds as triggers to avoid suspicion.

To facilitate a comparison of our method with existing adversarial attacks against Voice Processing Systems (VPSs), we use the threat model taxonomy from the Systematization of Knowledge (SoK) paper by Abdullah et al. [1]. A summary of our method in relation to this taxonomy is provided in Table I. We refer the reader to the SoK paper [1] for detailed definitions of each category rather than repeating them here. As can be seen from the table, our method is a targeted attack that assumes a white-box threat model. It needs 0 queries since no additional computation is required for new attacks once TrojanModel is well trained. With regard to the "acoustic environment," although we did not conduct over-the-air experiments at various noise levels explicitly, the experiments were conducted in a real-world setting to factor in noise instead of a controlled lab setting. Moreover, TrojanModel was trained to improve the performance of the target model in the presence of noise.

IV. PROBLEM DEFINITION AND ASSUMPTIONS

Our goal is to attack an ASR system by inserting a trigger into its acoustic model. We expect an ASR system infected with TrojanModel to only output a predefined target transcript when the trigger is present in the background. As the system behaves normally under normal circumstances, the victim will not suspect that the system is compromised. In addition, we expect the ASR system with TrojanModel to outperform the target ASR system under certain conditions for which it has been optimized.

Mathematically, let f be an ASR system, x be the voice input to be transcribed, y be the ground truth transcript, t be predefined target phrases, and δ be triggers associated with the corresponding t. Multiple triggers can theoretically be inserted into an acoustic model, where δ_i corresponds to t_i for $i \in \{1, 2, \ldots, K\}$, where K is the total number of triggers. The aim is to obtain a compromised ASR system f', in which the acoustic model is modified. To make Trojan attacks practical, f' must satisfy the following requirements:

- Triggers can successfully result in malicious phrases when mixed with benign input:

$$\underset{x \in \mathcal{X}}{P}\left(f'(h(x, \delta_i)) = t_i\right) \geq \tau, \text{ for } i \in \{1, 2, \ldots, K\} \tag{1}$$

  where $\mathcal{X}$ denotes the distribution of benign input, τ is the required attack success rate and δ_i is the i^{th} trigger corresponding to the i^{th} target phrase t_i. $h(x, \delta_i)$ represents the function to combine a benign input with a trigger. This means that output from $h(x, \delta_i)$ is what is actually received by a microphone and sent to f' when x and δ_i are both played.

  Specifically, we consider two attack scenarios. The first is where a trigger is played in the background while a victim speaks to f'. The other is where speech mixed with a trigger is prerecorded and input to f'. To enable the first scenario, there should be no synchronization required between δ_i and x. This is because it is impractical for an adversary to synchronize the playing of a trigger with the exact time that a victim speaks. In practice, the trigger (e.g., a piece of music) can be played repeatedly in the background and the compromised ASR system will output a malicious phrase whenever a victim speaks to the system while the trigger is present in the background.

- The performance of f' must be comparable with f under normal usage. This can be expressed as:

$$\text{WER}(f'(\mathcal{X}), \mathcal{Y}) \approx \text{WER}(f(\mathcal{X}), \mathcal{Y}) \tag{2}$$

where $\mathcal{X}$ and $\mathcal{Y}$ represent the set of benign (non-malicious) audio and corresponding ground truth, respectively. WER denotes word error rate, a standard measurement of ASR performance (described in Section IV-A). If f' performs significantly worse than f, a victim will likely cease to use it based on poor performance.
- f' outperforms f under certain conditions:

$$\text{WER}(f'(\mathcal{X}'), \mathcal{Y}) < \text{WER}(f(\mathcal{X}'), \mathcal{Y}) \tag{3}$$

where $\mathcal{X}'$ means the set of benign audio affected by certain conditions, such as a noisy environment. Assuming that the victim notices the modification to the compromised model's architecture resulting from TrojanModel, an adversary can show that f' outperforms f to justify this modification and to entice a victim into deploying it.
- There must be no false positives:

$$\underset{x \in \mathcal{X}}{P}(f'(x) = t_i) = 0, \text{ for } i \in \{1, 2, \ldots, K\} \tag{4}$$

under normal usage, as false positives will arouse a victim's suspicion, which will potentially result in f' no longer being used.

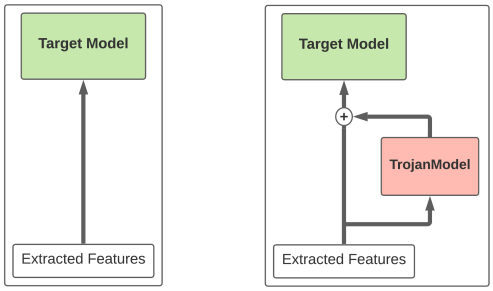
A. Evaluation

To measure the quality and effectiveness of our Trojan attack, we use the following metrics:

- **Sound Pressure Level (SPL)** measures the pressure of a sound compared with a reference value:

$$SPL = 20 \log_{10} \frac{P(s)}{P(r)} \tag{5}$$

where s and r are the sound to be measured and the reference sound, respectively. P is the root mean square, such that $P(v) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} v_i^2}$, where N is the size of the input signal v.
Signal-to-Noise Ratio (SNR) and Perceptual Evaluation of Speech Quality (PESQ) [32] are commonly used in the literature to measure the quality of audio AEs [22], [37], [41]. However, SNR and PESQ are not suitable for measuring the quality of Trojan attacks because SNR and PESQ measure distortion caused by noise. For AEs, it is preferred to have larger SNR and PESQ values such that the distorted signals have a greater resemblance to the original signals. Whereas for our Trojan attacks, triggers sound normal and a victim will not perceive such triggers as intrusive. For example, a trigger can be a piece of music playing in the background. Hence, it is meaningless to use SNR and PESQ to measure the distortion caused by triggers.
- **Success Rate (SR)** measures the percentage of successful attacks when a trigger is played.



(a) Uncompromised ASR system

(b) Compromised ASR system

Fig. 3. (a) Extracted features are passed directly to the target acoustic model. (b) The output logits from TrojanModel are added to the input features as perturbations. The results are passed to the target acoustic model.

- **Word Error Rate (WER)** is a standard measurement of ASR performance. It calculates the difference between the predicted transcript and the ground truth transcript:

$$WER = \frac{S_{word} + D_{word} + I_{word}}{N_{truth}} \tag{6}$$

where S_{word}, D_{word}, and I_{word} represent the number of substitutions, deletions, and insertions of words, respectively, needed to transform a predicted transcript into the corresponding ground truth transcript. N_{truth} is the number of words in the ground truth transcript.
- **Levenshtein Distance (LD)** calculates the minimum number of letter-level modifications, including substitutions, deletions, and insertions, required to transform a predicted transcript into the corresponding ground truth transcript.

V. Proposed Method

A. Architecture

Our approach is for TrojanModel to detect the presence of triggers. This is so that the compromised ASR system will output a predefined target phrase when a trigger is present in the input. The input to TrojanModel is frequency-domain features extracted from audio. Fig. 3 illustrates the proposed method. The normal operation of an uncompromised ASR system is shown in Fig. 3(a) where extracted features are passed directly to the target acoustic model. In contrast, as shown in Fig. 3(b), when an ASR system is compromised by TrojanModel, the output logits from TrojanModel are added to the input features and the results are passed to the target acoustic model as input. In this manner, TrojanModel is only required to learn perturbations that are applied to the input.

This design was inspired by the residual learning of ResNet [15], which empirically demonstrated that learning perturbations are easier than learning the full range of transformations applied to the input. Moreover, after the weights of TrojanModel are initialized to small values, TrojanModel only slightly alters input features so as to fulfill the requirements of not degrading the performance under normal usage (Eq. 2) and

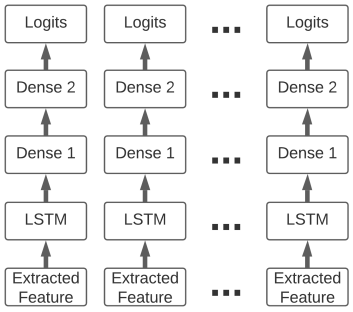


Fig. 4. The architecture of TrojanModel used for detecting the presence of triggers.

not have false positives (Eq. 4). This facilitates the training of TrojanModel.

Alternatively, if TrojanModel was designed to replace the target model's input directly with modified input, this would make the training of TrojanModel significantly more difficult. The reason for this is because TrojanModel would have to reconstruct the input, even for clean speech. In contrast, in the current design where TrojanModel adds perturbations to the input, TrojanModel can simply output zero for clean speech and not affect the target model's performance.

The architecture of TrojanModel is depicted in Fig. 4. We use Long Short-Term Memory (LSTM) in the first layer since it can handle input of arbitrary length. The length of input to an ASR system varies because recording stops when a user no longer speaks. Then, two dense layers are stacked. The first dense layer uses ReLU as its activation function while the second dense layer does not use an activation function. These two dense layers transform the sequence of hidden states from LSTM to form the final output logits. The output dimensions of the last dense layer are the same as the dimensions of the input features.

B. Training TrojanModel

Our Trojan attack must satisfy the multiple requirements previously defined in Eq. 1, 2, 3, and 4. These requirements represent different tasks to be performed. To fulfill the requirement defined in Eq. 1, TrojanModel needs to correctly detect the presence of a trigger and its output must appropriately perturb the input features. The requirements defined in Eq. 2 and Eq. 4 require that TrojanModel not adversely affect the output for benign speech when no trigger is present. Eq. 3 requires TrojanModel to recover the distortions caused by certain conditions. Hence, the training of TrojanModel is intrinsically a multi-task learning problem, i.e. there are three different tasks to learn. The purpose of multi-task learning is to make a single model perform multiple different tasks [40]. For example, the work in [27] used a single model to handle multiple Natural Language Processing (NLP) tasks. This is in contrast to the traditional method in which researchers design a model for a single task only. A Trojan attack with multi-task learning was also recently considered by Bagdasaryan et al. [4] for image processing.

Our loss function ℓ_{Trojan} for training TrojanModel is defined as follows:

$$\ell_{Trojan}(x,t) \triangleq \mathbb{I}_x \ell_{CTC}(G(x+g(x)),t) + (1-\mathbb{I}_x)\ell_\eta(x) \tag{7}$$

where x denotes input audio, and t is the target phrase corresponding to a trigger. ℓ_{CTC} is the Connectionist Temporal Classification (CTC) loss [12]. G and g represent the target acoustic model and TrojanModel, respectively. ℓ_η is defined as follows:

$$\ell_\eta(x) \triangleq ||\eta + g(x+\eta)||_2 + ||g(x)||_2 \tag{8}$$

where η denotes the distortions that we want TrojanModel to recover. $\mathbb{I}_x$ is an indicator function of x defined as:

$$\mathbb{I}_x \triangleq \begin{cases} 1, & \text{if x is speech mixed with a trigger} \\ 0, & \text{if x is benign speech} \end{cases} \tag{9}$$

If x is speech mixed with a trigger, we can see from Eq. 7 that minimizing ℓ_{Trojan} is equivalent to minimizing ℓ_{CTC}. In this case, the goal is to meet the requirement defined in Eq. 1. This is because minimizing ℓ_{CTC} encourages $x+g(x)$, which means input features perturbed by output logits from TrojanModel, to be decoded as the target phrase t.

If x is benign speech, then minimizing ℓ_{Trojan} is equivalent to minimizing $||\eta+g(x+\eta)||_2+||g(x)||_2$. This means we want to minimize both $||\eta + g(x+\eta)||_2$ and $||g(x)||_2$. Minimizing $||g(x)||_2$ corresponds to fulfilling the requirements of Eq. 2 and Eq. 4 because this leads to $||g(x)|| \approx 0$ such that input features are not modified for benign speech. This reduces the effect of TrojanModel on benign input and makes TrojanModel stealthier. On the other hand, minimizing $||\eta + g(x+\eta)||_2$ is to fulfill the requirement defined in Eq. 3 because it encourages TrojanModel to recover the distortion caused by η. As previously mentioned, this is done to entice a victim to use the resulting model despite the modified architecture.

Based on the above discussion, we essentially minimize different objectives for different types of input. Hence, it is not suitable to only use one optimizer for minimizing ℓ_{Trojan}. This is because state-of-the-art optimizers [17] often use information from past gradients to improve their convergence rate based on a momentum strategy. The assumption is that incorporating information from past gradients will help convergence. However, gradients of separate objectives tend to be uncorrelated and may carry misleading information if used together. Therefore, we use different optimizers to minimize ℓ_{Trojan} for different types of input. Our strategy is presented in Algorithm 1.

During training, the model that performs the best on the validation set is saved for testing. It should be noted that it is not straightforward to measure performance because we optimize two separate objectives (i.e. ℓ_{CTC} and ℓ_η in Eq. 7). Empirically, we use the following formula to calculate a scalar p for comparing models:

$$p = \beta \cdot \ell_{CTC} + \ell_\eta \tag{10}$$

where β is a balancing coefficient between ℓ_{CTC} and ℓ_η. Smaller p indicates a better model.

Algorithm 1 Minimizing ℓ_{Trojan} using two optimizers

Input: an initialized TrojanModel g; a training set $\mathcal{D}$; target phrases t
Output: a trained TrojanModel g'

initialize optimizers p_1, p_2
while epoch $\leq$ max epochs **do**
  for each batch $d \in \mathcal{D}$ // $\mathcal{D}$ is shuffled for each epoch
    extract attacks d_{attack} from d
    extract benign speech d_{benign} from d
    optimize $\ell_{Trojan}(d_{attack}, t)$ in Eq. 7 using p_1
    optimize $\ell_{Trojan}(d_{benign}, null)$ in Eq. 7 using p_2
  end for
end while

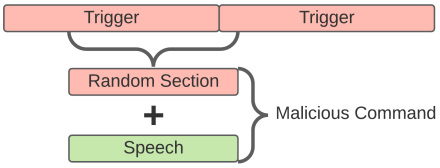


Fig. 5. Combining speech with a random section of a trigger. If the section extends beyond the end of the trigger, the start of the trigger is used to complete the section. TrojanModel is expected to recognize the combined audio as a malicious command.

C. No Synchronization

Recall that there should be no synchronization required between a trigger and speech. To achieve this goal, we generate attacks by mixing each training audio with a randomly selected section from a trigger. This scheme is depicted in Fig. 5. One can see that a random section with a length equal to speech is first selected from a trigger. If the section extends beyond the end of the trigger, the start of the trigger is used to complete the section.

Next, the random section is combined with speech based on the following:

$$x_{combined} = \alpha_{section} \cdot \hat{x}_{section} + \alpha_{speech} \cdot \hat{x}_{speech} \qquad (11)$$

where $\hat{x}_{section}$ and $\hat{x}_{speech}$ are the random section and speech, respectively, both are normalized within $[-1, 1]$. Normalizing refers to linearly scaling the volumes within the minimum and maximum volume range. The factors $\alpha_{section}$ and α_{speech} are used to adjust the volume levels of the trigger and speech, respectively. The combined audio: $x_{combined}$, is an attack to be recognized by TrojanModel as a malicious command corresponding to the trigger.

D. Over-the-air Attacks

To make our Trojan attack effective when played over a physical speaker and received by a microphone, we incorporate RIR to augment the training data. Given RIR r and input audio x, transformed audio x' can be calculated as $x' = x * r$, where $*$ is the convolution operation. In practice, we need to randomly generate multiple virtual rooms and obtain their corresponding RIR. Each training data is then transformed by a randomly selected RIR. In this manner, TrojanModel is expected to be robust against RIR so that it can successfully recognize over-the-air attacks.

TABLE II
TARGET PHRASES AND CORRESPONDING TRIGGERS.

Target phrase	**Trigger**	**Noise**	$\alpha_{section}$
open the garage door	flute	computer	[0.06, 0.14]
cut off power supply	siren1	computer	[0.06, 0.14]
activate silent mode	synthesizer	vehicle	[0.15, 0.35]
clear all notifications	siren2	vehicle	[0.06, 0.14]
turn on every heater	violin	white noise	[0.09, 0.21]
call the police now	oboe	white noise	[0.06, 0.14]

VI. Experimental Results

A. Setup

Our proposed method inserts a Trojan into the acoustic model of an ASR system. No modification is made to the language model. DeepSpeech was chosen as the target ASR system because of its state-of-the-art performance [14]. DeepSpeech is also used as the target model in many existing research on ASR security issues [5], [7], [22]. We used version 0.8.2 by Mozilla. The pre-trained language model, which is also provided by Mozilla, was used to improve the performance of beam search decoding. We set the hidden state dimensions of LSTM in TrojanModel to 1024 and the output dimensions of the first dense layer to 512. MFCCs were extracted as features from input audio for training TrojanModel because DeepSpeech is trained from MFCCs.

Throughout the experiments, we used six target phrases that represent potentially malicious commands with each phrase associated with a specific trigger. The details are shown in Table II. In the table, "flute", "synthesizer", "violin", and "oboe" are music clips used as triggers. In addition to music, we also used other audio that may occur in the real-world, i.e. sirens, as triggers. We trained a separate TrojanModel for each target phrase and its corresponding trigger. If a trigger is longer than 10 seconds, we sliced it to 10 seconds. It should be noted that a single TrojanModel can theoretically detect multiple triggers corresponding to different target phrases. However, we empirically observe that training such a TrojanModel is difficult in practice. Considering the fact that multi-task learning is still an active and challenging research area, we leave training a TrojanModel with multiple triggers as a topic for future work.

TrojanModel is designed to outperform the target model under certain conditions. For our experiments, TrojanModel was required to filter out white noise, noise from a computer's fan, and noise within a moving vehicle. White noise represents general signal distortion, while noise from a computer's fan and noise within a moving vehicle are common occurrences in daily life. TrojanModel aims to improve the performance of an ASR system despite such noise being present.

We used Librispeech [30] as the dataset for training and testing since Mozilla provides a pre-trained model on this dataset. We only used speech with a duration of between 2 to 5 seconds for training and testing. This is sufficient for demonstrating our Trojan attack in practice because most voice commands are not long and usually contain between 2 to 4 words as stated in [20]. All audio from "dev-clean" with a duration of between 2 to 5 seconds were extracted for training TrojanModel.

Since there is no need to synchronize the timing between a trigger and speech, for each training speech, we randomly selected two sections from the trigger and combined each of them with the speech to generate two attacks. α_{speech} in Eq. 11 was uniformly sampled from 0.3 to 0.7 while $\alpha_{section}$ was uniformly sampled in the corresponding ranges shown in Table II. Some triggers, such as flute music, contain large power at high frequencies, which easily masks speech. Hence, we used different ranges for different triggers to make speech audible. In addition to clean benign data, we also applied noise, which will be filtered out by TrojanModel, to each training speech to form extra benign data. The purpose of this is to let TrojanModel learn how to filter out such noise. In total, 2162 attacks and 2162 pieces of benign speech were used for training TrojanModel.

To make Trojan attacks effective over-the-air, we generated 100 virtual rooms using pyroomacoustics[1]. Room sizes were randomly sampled between 4 to 6 meters with a fixed height of 3.5 meters. The positions of a virtual microphone and a virtual speaker were also randomly selected inside each virtual room. RT60 was uniformly sampled from 0.15 to 0.20. RT60 is a measure of the time it takes for SPL to decrease by 60dB. A convolution of each training data sample with an RIR of a random virtual room was performed.

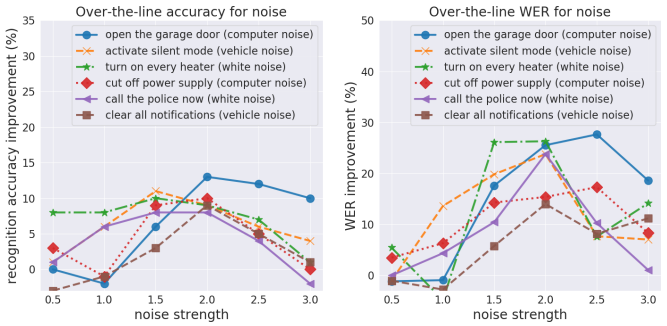
10% of the training data was separated for validation. Since we optimize ℓ_{CTC} and ℓ_{η} at the same time (Eq. 7), a successfully trained TrojanModel should achieve small values for the two terms on the validation set. We trained TrojanModel for 600 epochs with the learning rate and batch size equal to $1e{-}4$ and 128, respectively. The Adam method [17] was used for optimization with weight decay equal to $1e-4$. The model with the best performance on the validation set was used for testing. Eq. 10 with $\beta = 30$ was used to compare models.

B. Over-the-line Attacks

We randomly extracted 100 audio from "test-clean" to test our Trojan method for over-the-line attacks. To generate attacks, we randomly selected one section from the trigger and use Eq. 11 to combine the trigger with each test audio. $\alpha_{section}$ and α_{speech} were sampled in the same way as when generating training data. In total, the test set contained 100 attacks and 100 benign speech.

Experimental results for over-the-line attacks are shown in Table III. The results show that TrojanModel achieved excellent attack success rates (SR) with all values being at

[1]https://pypi.org/project/pyroomacoustics/



(a) Recognition Accuracy (b) WER

Fig. 6. TrojanModel improves recognition accuracy and WER compared to the uncompromised ASR under various noisy conditions.

or close to 100%. WER represents the word error rate (WER) of benign speech. We compared the WER of the compromised ASR system with the uncompromised target ASR system. The WER only increased by 0.0011 for the case of "call the police now", while other WER values remained the same or even decrease. Hence, it can be seen that overall, the results show that TrojanModel did not affect the compromised ASR system's performance for benign speech. This is one of the benefits of not modifying the weights of a target model. Moreover, the number of false positives (FP) remained at 0.

Although false positives were always 0 and WER remained stable. These two measurements are word-level measurements. They do not reveal whether the transcripts of benign speech contained parts of the target phrases at the level of individual letters. A victim will be suspicious if benign speech is frequently transcribed into corrupted phrases. Therefore, we also calculated the average Levenshtein Distance (LD) and the minimum LD between transcripts of benign speech to their corresponding target phrases. It can be seen from Table III that the average and minimum LDs were similar for the compromised ASR system and the uncompromised target ASR system. This indicates that transcripts of benign speech for the compromised ASR system did not contain parts of the target phrases, despite the presence of TrojanModel. Hence, the results demonstrate that the output of the compromised ASR system is indistinguishable from the uncompromised ASR system.

To entice users to use the Trojan model, TrojanModel should improve the model's performance under certain conditions. Fig. 6(a) and Fig. 6(b) show results of the improvements in recognition accuracy (i.e. the percentage of successfully transcribed speech) and WER, respectively. Noise strength represents a factor that the noise was multiplied with. Interestingly, when the noise strength was 1 or below, TrojanModel did not improve the model's performance by much as the uncompromised ASR system is not severely affected by low noise. Conversely, TrojanModel may slightly change the distribution of input features and this outweighs the benefits it brings when noise is low. However, TrojanModel significantly improved the performance in both recognition accuracy and

TABLE III
EXPERIMENTAL RESULTS FOR OVER-THE-LINE ATTACKS.

Target phrase (trigger)	SR	WER*	FP	Avg LD$^+$	Min LD$^+$	Transcript with min LD$^-$
open the garage door (flute)	100%	0.0743 (0.0743)	0	37.99 (37.99)	15 (15)	robin fits root
cut off power supply (siren1)	100%	0.0712 (0.0743)	0	40.36 (40.31)	17 (17)	no good my dear watson
activate silent mode (synthesizer)	97%	0.0743 (0.0743)	0	38.84 (38.84)	15 (15)	robin fits root
clear all notifications (siren2)	99%	0.0743 (0.0743)	0	38.80 (38.80)	18 (18)	she was alone that night
turn on every heater (violin)	100%	0.0743 (0.0743)	0	37.87 (37.87)	15 (15)	robin fits root
call the police now (oboe)	98%	0.0754 (0.0743)	0	38.99 (38.97)	16 (16)	i give my consent

$*$: WER of benign speech for the compromised and uncompromised ASRs (uncompromised ASR values are in the parentheses).
$^+$: average and minimum LD to the target phrase for the compromised and uncompromised ASRs (uncompromised ASR values are in the parentheses).
$^-$: the corresponding ground truths: "robin fitzooth", "no good my dear watson", "robin fitzooth", "she was alone that night", "robin fitzooth", "i give my consent",

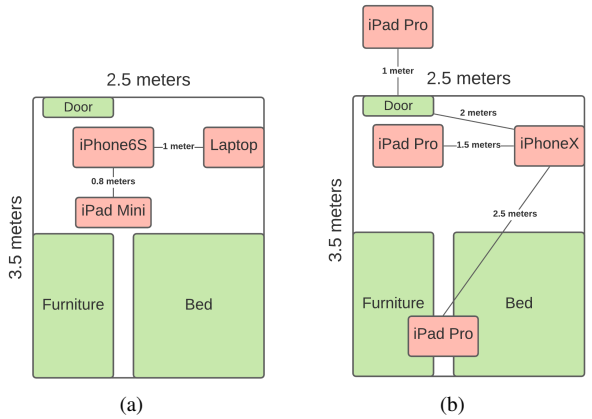


Fig. 7. Over-the-air attack experiments were conducted in a real-world bedroom. (a) Device locations for attacks with triggers playing in the background. (b) Device locations for attacks using pre-recorded speech containing triggers.

WER when the noise was larger. For example, TrojanModel improved recognition accuracy by 10% and WER by 25% for benign speech when the noise strength was between 1.5 and 2.5. If the noise got too large, its performance dropped because TrojanModel did not filter out overly strong noise. Overall, one can see that TrojanModel indeed improved the performance when noise strength was within a reasonable range.

C. Over-the-air Attacks

Compared to over-the-line attacks, over-the-air attacks are more practical in a real-world setting. This is because over-the-air attacks must be effective when played over a speaker and received by a microphone. In this section, we present experimental results of over-the-air attacks. Rather than using a controlled lab environment, all experiments were conducted in a real-world apartment bedroom during the day to deliberately include noise from the street and the neighbors. The size of the room was approximately 2.5×3.5 meters with a height of 2.8 meters. To make our attacks more practical, we used common commercial products, such as smartphones, to play and receive the over-the-air attacks.

1) Over-the-air Attacks with Triggers Playing in the Background: In the experiments, a trigger was repeatedly played in the background while a victim spoke to the compromised ASR system. This demonstrates the scenario where a trigger (in the form of music) is playing in the background. The speaker of an Apple iPad mini 4 was used to play each test speech, while the speaker of a Dell G7 laptop was used to play the trigger, i.e. the snippet of sound, repeatedly in the background. An Apple iPhone 6S was used to record the resulting audio. The device locations within the room are depicted in Fig. 7(a). To test over-the-air attacks, we used the same 100 audio from "test-clean" that were used to perform the over-the-line attacks.

The results of over-the-air experiments are shown in Table IV. One can see that the success rates are high with the lowest value being at 91%, which indicates the effectiveness of TrojanModel. Overall, WER is slightly worse when compared to the over-the-line scenario as the pre-trained DeepSpeech model provided by Mozilla does not perform well in a physical environment. Nevertheless, although WER of the compromised ASR system is marginally worse than the uncompromised ASR system, the biggest gap is only 0.0115. This indicates that TrojanModel preserved the performance of the ASR system for benign speech. As with the over-the-line attacks, there were no false positives for the over-the-air attacks. Looking at the LD results, we can see that the average LD and the minimum LD of the compromised ASR system are almost the same as the uncompromised ASR system. Furthermore, the transcripts that are closest to the target phrases are completely different from the corresponding target phrases. This shows that despite the presence of TrojanModel, the compromised ASR system's transcripts for benign speech did not contain parts of the target phrases.

2) Over-the-air Attacks of Pre-recorded Speech Containing Triggers: In addition to the over-the-air experiments discussed above, we also conducted experiments in which speech containing triggers was pre-recorded and played to the compromised ASR system. This demonstrates the scenario where attacks are directly input to a compromised ASR system. In the experiments, we used the speakers of an iPad Pro A2377 to play the audio attacks and an iPhoneX for recording. The devices used for this experiment were deliberately different from the previous over-the-air experiments to show that our

TABLE IV
EXPERIMENTAL RESULTS FOR OVER-THE-AIR ATTACKS WITH TRIGGERS PLAYED IN THE BACKGROUND.

Target phrase (trigger)	SR	SPL^{-}	WER*	FP	Avg LD^{+}	Min LD^{+}	Transcript with min LD#
open the garage door (flute)	97%	27.14 dB	0.2314 (0.2272)	0	35.48 (35.48)	15 (15)	the swathes
cut off power supply (siren1)	99%	17.91 dB	0.2314 (0.2272)	0	37.68 (37.67)	17 (17)	no good my dear watson
activate silent mode (synthesizer)	91%	29.96 dB	0.2325 (0.2272)	0	36.03 (36.10)	15 (15)	the swathes
clear all notifications (siren2)	99%	41.48 dB	0.2335 (0.2272)	0	36.23 (36.30)	17 (17)	a calecanthus
turn on every heater (violin)	100%	33.03 dB	0.2387 (0.2272)	0	35.08 (35.19)	12 (12)	thus in character
call the police now (oboe)	100%	30.93 dB	0.2366 (0.2272)	0	36.12 (36.27)	14 (14)	the swathes

$^{-}$: SPL of triggers indicate ambient sound within the room as a reference. SPL of benign speech is $36.6dB$.
$*$: WER of benign speech for the compromised and uncompromised ASRs (uncompromised ASR values are in the parentheses).
$^{+}$: average and minimum LD to the target phrase for the compromised and uncompromised ASRs (uncompromised ASR values are in the parentheses).
#: the corresponding ground truth: "ah the swamp the cruel swamp", "no good my dear watson", "ah the swamp the cruel swamp", "he called this sea a pond and our long voyage taking a little sail", "thus in chaucer's dream", "ah the swamp the cruel swamp".

TABLE V
EXPERIMENTAL RESULTS FOR OVER-THE-AIR ATTACKS OF PRE-RECORDED SPEECH CONTAINING TRIGGERS. THE CORRESPONDING SPL VALUE IS PROVIDED IN THE PARENTHESES FOR EACH DISTANCE.

Target phrase (trigger)	1.5m (SPL)	2.5m (SPL)	Door open (SPL)	Door closed (SPL)
open the garage door (flute)	99% (27.89 dB)	98% (25.06 dB)	91% (18.49 dB)	29% (7.94 dB)
cut off power supply (siren1)	100% (27.96 dB)	100% (25.62 dB)	99% (19.03 dB)	13% (8.17 dB)
activate silent mode (synthesizer)	73% (29.97 dB)	73% (25.96 dB)	33% (18.64 dB)	0% (7.60 dB)
clear all notifications (siren2)	99% (31.08 dB)	100% (28.01 dB)	98% (22.13 dB)	26% (9.83 dB)
turn on every heater (violin)	100% (29.01 dB)	100% (26.48 dB)	93% (19.61 dB)	2% (8.32 dB)
call the police now (oboe)	100% (29.61 dB)	99% (25.37 dB)	97% (19.91 dB)	21% (8.86 dB)

attack is not limited to specific hardware. The audio attacks were played in the room at 1.5 and 2.5 meters away from the microphone. In addition, we performed the experiments in a more challenging scenario, where the audio attacks were played from 1 meter outside the room with a wooden door. In this scenario, there was no direct path from the speaker to the microphone. The distance from the iPhoneX to the door was approximately 2 meters. Experiments were conducted with the door opened and also with the door closed. Fig. 7(b) depicts the device locations.

For each target phrase, we played 100 attacks at each location. These attacks were generated from the same set of test audio as Section VI-B. $\alpha_{section}$ were sampled in the same range as shown in Table II. Experimental results are shown in Table V. The Sound Pressure Level (SPL) gives an indication of the strength of the input received by the microphone. Overall, the attack success rates were good when the triggers were played within the room at distances of 1.5 meters and 2.5 meters away from the microphone, and also when they were played from outside the room when the door was open. Attacks with the target phrase "activate silent mode" were an exception since the attack success rate decreased noticeably when they were played outside the room with the door open. This may be due to the significant damping of the trigger when transmitted over the air. An interesting result was that attacks succeeded in several cases when the door was closed. Although the success rates when the door was closed were low, the capability of penetrating a door demonstrates the practicality of our Trojan attack in a real-world scenario. To our knowledge, this is the first experiment in the literature that successfully demonstrates a Trojan attack against an ASR system through a door.

D. Robustness to Defense

At present, there is limited work on Trojan attacks for ASR systems. This means the volume of research on defending against such attacks is also limited. Existing defense methods mainly focus on preventing Trojan attacks for image recognition. To the best of our knowledge, only Fine-pruning [21] and STRIP [11] have been shown to be effective for defending against Trojan attacks for SCR. SCR transforms audio of fixed length into a label from a fixed set, which is an easier task compared to ASR. This is because ASR systems must attempt to output the best transcript from a large number of potential transcripts when given input audio of varying lengths. This difference in the working mechanisms of SCR and ASR likely means a difference in defense methods as well. While a comprehensive investigation of defense methods is beyond the scope of this paper, we tested our Trojan attack in relation to other ASR defense mechanisms, namely, transformation and Temporal Dependency (TD) detection. Such defense methods have been used to evaluate the robustness of audio AEs for ASR [22], [39], [41].

In the transformation defense method, a user transforms input audio before passing it to a target ASR system. The purpose is to attempt to destroy any potential adversarial perturbations. The applied transformations should not degrade

TABLE VI
ROBUSTNESS TO GAUSSIAN NOISE WITH VARIOUS STANDARD DEVIATION.

	Over-the-line attacks				Over-the-air attacks			
	$1e-2$	$1e-3$	$1e-4$	Baseline	$1e-2$	$1e-3$	$1e-4$	Baseline
open the garage door (flute)	100%	100%	100%	100%	100%	100%	100%	97%
activate silent mode (synthesizer)	89%	99%	97%	97%	3%	88%	90%	91%
turn on every heater (violin)	69%	100%	100%	100%	0%	96%	100%	100%
cut off power supply (siren1)	100%	100%	100%	100%	14%	92%	99%	99%
call the police now (oboe)	8%	93%	98%	98%	0%	81%	100%	100%
clear all notifications (siren2)	100%	100%	100%	99%	100%	100%	99%	99%

TABLE VII
ROBUSTNESS TO LOW-PASS FILTERS WITH VARIOUS CUTOFF FREQUENCIES (HZ).

	Over-the-line attacks				Over-the-air attacks			
	3kHz	5kHz	7kHz	Baseline	3kHz	5kHz	7kHz	Baseline
open the garage door (flute)	0%	0%	30%	100%	0%	0%	0%	97%
activate silent mode (synthesizer)	0%	0%	91%	97%	0%	0%	40%	91%
turn on every heater (violin)	0%	0%	72%	100%	0%	0%	37%	100%
cut off power supply (siren1)	0%	0%	0%	100%	0%	0%	0%	99%
call the police now (oboe)	0%	0%	83%	98%	0%	0%	90%	100%
clear all notifications (siren2)	0%	0%	88%	99%	0%	0%	50%	99%

TABLE VIII
ROBUSTNESS TO RESAMPLING AT VARIOUS SAMPLING RATES.

	Over-the-line attacks				Over-the-air attacks			
	8000	10000	12000	Baseline	8000	10000	12000	Baseline
open the garage door (flute)	0%	0%	0%	100%	0%	0%	0%	97%
activate silent mode (synthesizer)	0%	0%	0%	97%	0%	0%	0%	91%
turn on every heater (violin)	0%	0%	0%	100%	0%	0%	0%	100%
cut off power supply (siren1)	0%	0%	0%	100%	0%	0%	0%	99%
call the police now (oboe)	0%	0%	0%	98%	0%	0%	0%	100%
clear all notifications (siren2)	0%	0%	0%	99%	0%	0%	0%	99%

TABLE IX
AUC OF TEMPORAL DEPENDENCY DETECTION FOR OVER-THE-LINE ATTACKS.

	k=$\frac{1}{2}$			k=$\frac{2}{3}$			k=$\frac{3}{4}$			k=$\frac{4}{5}$			k=$\frac{5}{6}$		
	WER	CER	LCP	WER	CER	LCP	WER	CER	LCP	WER	CER	LCP	WER	CER	LCP
open the garage door	0.48	0.43	0.01	0.43	0.41	0.06	0.42	0.42	0.07	0.37	0.41	0.09	0.39	0.40	0.15
cut off power supply	0.58	0.50	0.05	0.44	0.42	0.06	0.45	0.43	0.10	0.38	0.40	0.11	0.40	0.40	0.16
activate silent mode	0.53	0.53	0.13	0.42	0.47	0.12	0.40	0.45	0.12	0.38	0.44	0.12	0.40	0.42	0.16
clear all notifications	0.45	0.43	0.00	0.34	0.41	0.00	0.36	0.41	0.01	0.33	0.41	0.05	0.36	0.40	0.12
turn on every heater	0.45	0.47	0.04	0.39	0.45	0.03	0.39	0.44	0.04	0.34	0.41	0.06	0.36	0.40	0.12
call the police now	0.39	0.42	0.00	0.34	0.41	0.00	0.36	0.42	0.01	0.33	0.41	0.04	0.36	0.40	0.12

TABLE X
AUC OF TEMPORAL DEPENDENCY DETECTION FOR OVER-THE-AIR ATTACKS.

	k=$\frac{1}{2}$			k=$\frac{2}{3}$			k=$\frac{3}{4}$			k=$\frac{4}{5}$			k=$\frac{5}{6}$		
	WER	CER	LCP	WER	CER	LCP	WER	CER	LCP	WER	CER	LCP	WER	CER	LCP
open the garage door	0.38	0.41	0.00	0.36	0.38	0.20	0.46	0.46	0.44	0.50	0.50	0.50	0.50	0.50	0.50
cut off power supply	0.38	0.40	0.00	0.36	0.38	0.20	0.46	0.46	0.44	0.50	0.50	0.50	0.50	0.50	0.50
activate silent mode	0.39	0.41	0.00	0.36	0.37	0.20	0.46	0.46	0.44	0.50	0.50	0.50	0.50	0.50	0.50
clear all notifications	0.39	0.41	0.00	0.35	0.38	0.20	0.45	0.46	0.43	0.50	0.50	0.50	0.50	0.50	0.50
turn on every heater	0.39	0.41	0.00	0.36	0.39	0.20	0.46	0.46	0.45	0.50	0.50	0.50	0.50	0.50	0.50
call the police now	0.40	0.42	0.00	0.36	0.38	0.19	0.46	0.46	0.45	0.50	0.50	0.50	0.50	0.50	0.50

the performance of the ASR system. For our experiments, we applied a set of commonly used transformations, namely, Gaussian noise, filtering with a low-pass filter, and resampling, to the Trojan attack.

Experimental results showing the success rates of our Trojan attack after input audio were transformed are shown in Tables VI, VII, and VIII for over-the-line and over-the-air attacks with repeatedly played triggers. Although Table VI shows that Gaussian noise with standard deviation $1e-2$ successfully prevents some attacks, this standard deviation value is too large to be practical. This is because the quality of input audio noticeably degrades such that the performance of a target ASR system is significantly affected. Overall, the results in Table VI suggest that both over-the-line and over-the-air attacks are robust to Gaussian noise when the standard deviation is $< 1e-2$.

The results in Tables VII and VIII show that in general, our Trojan attack is not robust to low-pass filtering and resampling. Although some attacks, such as attacks for "call the police now," show robustness to low-pass filtering with a 7kHz cutoff frequency, low-pass filtering and resampling can easily prevent our attack. This is because we did not consider such transformations when training TrojanModel. Intuitively, robustness can potentially be achieved if we apply these transformations to augment the training set.

TD detection was proposed by Yang et al. [39] as an effective method to detect audio AEs. They assumed that benign audio preserves TD while audio AEs do not. Given input audio, let S_k represent the transcript of its first k portion. Let $S_{\{whole,k\}}$ denote the prefix of the transcript in relation to the whole audio, with length $S_{\{whole,k\}}$ the same as the length of S_k. $S_{\{whole,k\}}$ should be consistent with S_k if the input is benign audio. Otherwise, an inconsistency indicates that the audio is potentially adversarial.

Tables IX and X show the TD detection results for our over-the-line and over-the-air attacks with triggers played repeatedly. The same measurements as in [39] were used: Area Under the Curve (AUC) of WER, AUC of Character Error Rate (CER), and AUC of the Longest Common Prefix (LCP). An obvious observation is that the TD detection method performs poorly on our Trojan attack because the AUC scores are all low.

Upon closer inspection, it can be seen that AUC scores of LCP are extremely low for over-the-line attacks. This is also true for over-the-air attacks when the ratio is $\frac{1}{2}$. Empirically, this is because the first half of both over-the-line and over-the-air attacks were already transcribed as the corresponding target phrases, which is different from benign speech. Interestingly, this phenomenon goes against the TD assumption that asserts that transcripts of partial attacks are inconsistent with transcripts of full attacks. If we exchange the roles of the positive and negative samples when calculating LCP, this simple modification made to TD detection will potentially detect our attacks via LCP. In other words, if a defender is aware of our attack, a slightly modified TD detection via LCP can detect our attack.

In addition, AUC scores of LCP are 0.50 when the ratio is $\geq \frac{4}{5}$ for over-the-air attacks. This implies that LCP values for benign speech and attacks are similar to each other in this case. However, this is likely due to the fact that we recorded an extra second before a speech begins and after a speech ends. This results in the starting $\frac{3}{4}$ portion of benign speech containing almost all the speech. Therefore, transcripts of the starting $\frac{3}{4}$ portion of benign speech are the same as or close to the transcripts of full audio. This is the reason why LCP cannot differentiate attacks with benign speech when the ratio $\geq \frac{4}{5}$ for over-the-air attacks.

E. Trigger Generalizability

From the experimental results, TrojanModel was able to detect its respective trigger. Nevertheless, an interesting question is whether triggers are generalizable. This refers to whether TrojanModel trained on a specific trigger is able to detect a different trigger. To test this, we conducted experiments in which a compromised ASR received attacks containing triggers that were not used in the training of TrojanModel. For example, we input attacks containing flute music into a TrojanModel that was trained on oboe music. In the experiments, we used all the over-the-line attacks from Section VI-B and all the over-the-air attacks from Section VI-C1. Table XII in the Appendix show these results.

From the results in table, it can be seen that TrojanModel trained on flute music did not respond to attacks containing oboe music (except for a single false positive over-the-air attack) and vice versa. Although these two instruments sound relatively similar to each other, differences in their spectrum of frequencies, represented by their spectrograms in Fig. 8 in the Appendix, prevent transferability. Another observation is that overall, TrojanModel trained on "siren2" had the highest likelihood of being activated by other triggers. A likely reason for this is because its frequencies, depicted in Fig. 8 in the Appendix, are concentrated within a narrow range such that TrojanModel is frequently activated by similar patterns in the frequencies of other triggers.

These results suggest that triggers are potentially generalizable in TrojanModel. To fully understand the underlying reasons for trigger generalizability, we need to know what dominates TrojanModel's inference. In the image domain, techniques like Grad-Cam [35] facilitate understanding of model inference. However, similar techniques have not been proposed in the ASR domain yet.

F. Triggers without Speech

In practice, ASR systems employ Voice Activity Detection (VAD) [9] to detect speech activity and filter non-speech. This means that in the absence of speech, a trigger alone will not result in a command being issued. For completeness, we conducted experiments where a compromised ASR system transcribed triggers only, i.e. no VAD to filter non-speech. We randomly extracted 100 sections of between 2 to 4 seconds from each trigger and sent them over-the-line to the target system. Experimental results are provided in Table XIII in

TABLE XI
EXPERIMENTAL RESULTS OBTAINED USING OTHER TRIGGERS.

Trigger^{+}	SR	WER*	FP
bird chirps	95%	0.1770 (0.0743)	4
cricket chirps	55%	0.0712 (0.0743)	0
car horn	87%	0.0859 (0.0743)	0

$^{+}$: Target phrases: bird chirps - "turn all lights off"; cricket chirps - "activate silent mode"; car horn - "open the garage door".
*: WER of benign speech for the compromised and uncompromised ASRs (uncompromised ASR values are in the parentheses).

the Appendix. The results show that a trigger alone has a high probability of being transcribed into the corresponding malicious command. Nevertheless, this does not devalue our attack as VAD is an essential pre-processing component of practical ASR systems [10].

VII. LIMITATIONS

The success of our Trojan attack depends heavily on the successful training of TrojanModel. Due to the difficulty of multi-task learning, the training of TrojanModel occasionally results in large fluctuations in the ℓ_{CTC} term of Equation 7. This is depicted in Fig. 9 in the Appendix. In contrast, there are less fluctuations in the ℓ_{η} term of Equation 7 as shown in Fig. 10 in the Appendix. Applying advanced methods, such as Gradient Surgery [40], may potentially stabilize the training of TrojanModel.

A. Unsuitable Triggers

It was empirically observed that certain types of sound were not suitable as triggers. Examples of these are shown in Table XI, where 100 over-the-line attacks were tested for each trigger. It can be seen from the table that experiments using bird chirps as the trigger resulted in a number of false positives, which in practice may alert the user of suspicious activity. Furthermore, the WER of benign speech for the compromised ASR was far worse compared with the uncompromised ASR. The results in Table XI also show that despite not producing any false positives, the use of cricket chirps as a trigger resulted in a low attack success rate while using car horns degraded the WER for clean input.

We hypothesize that this poor performance is due to two reasons, namely, the frequency of sound and its timing. More specifically, the closer the sound resembles human speech, the more likely it will be unsuitable as a trigger. As shown in Fig.11 in the Appendix, the spectrograms of human speech and the bird chirps display similar characteristics at frequencies above 2kHz, where the energy of the bird chirps is concentrated at. This makes it difficult for TrojanModel to detect the presence of the bird chirps in the midst of human speech, thus, resulting in unsatisfactory performance when the bird chirps were used as a trigger. In addition, unlike music, the cricket chirps and car horn honks were not continuous as depicted in Fig. 12 in the Appendix. So despite their frequencies being different from human speech, as shown in Fig.11, their non-continuous nature resulted in periodic periods in the attacks where only clean speech was present. This complicates the learning task of TrojanModel because it must remember whether a trigger has been detected previously when processing a short section of clean speech.

This presents an interesting direction for future research on comprehensively investigating the effect of trigger characteristics on the performance of TrojanModel.

B. Lack of Robustness to Defense

Another limitation of our method is the lack of robustness to defense. Although the results in Tables VI, IX, and X show that both the over-the-line and over-the-air attacks were robust to Gaussian noise and temporal dependency detection, the results in Tables VII and VIII show that our attack can easily be defended against using low-pass filtering and resampling. This means our Trojan attack can also be defeated by WaveGuard [16], which was recently proposed to defend against audio AEs. This is because WaveGuard is a framework that applies various preprocessing techniques to the input audio with the aim of destroying adversarial perturbations. It is straightforward to incorporate low-pass filtering and resampling in WaveGuard to defend against our attack.

A potential method for improving the robustness of our attack is to adaptively incorporate defense mechanisms when optimizing TrojanModel via the Expectation Over Transformation (EOT) technique [3]. However, the EOT technique requires an adversary to know the specific defense mechanisms that will be used in advance. This requirement is not always practical. We leave the topic of improving the robustness of TrojanModel as future work.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we presented TrojanModel, an innovative method of conducting a Trojan attack against an ASR system by using unsuspicious triggers. This is in contrast with existing work that uses noise-like triggers. TrojanModel does not require the retraining of a target model and we demonstrated that it can be used to successfully perform over-the-air Trojan attacks against an ASR system. While at present our Trojan attack is not robust to basic defenses like low-pass filtering and resampling, the incorporation of such transformations to augment training data can potentially improve its robustness to these defense methods. Improving the robustness of the proposed method will be a topic for future work. In addition, in future work, we will investigate methods to improve the training of TrojanModel so that multiple triggers can be inserted. A potential approach is to use advanced methods that are specifically designed for improving multi-task learning.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their invaluable feedback.

REFERENCES

[1] H. Abdullah, K. Warren, V. Bindschaedler, N. Papernot, and P. Traynor. Sok: The faults in our asrs: An overview of attacks against automatic speech recognition and speaker identification systems. In *2021 IEEE symposium on security and privacy (SP)*, pages 730–747. IEEE, 2021.

[2] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182, 2016.

[3] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok. Synthesizing robust adversarial examples. In *International conference on machine learning*, pages 284–293. PMLR, 2018.

[4] E. Bagdasaryan and V. Shmatikov. Blind backdoors in deep learning models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1505–1521, 2021.

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

[6] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, China, March 20-25, 2016*, pages 4960–4964. IEEE, 2016.

[7] T. Chen, L. Shangguan, Z. Li, and K. Jamieson. Metamorph: Injecting inaudible commands into over-the-air voice controlled systems. In *27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020*. The Internet Society, 2020.

[8] X. Chen, C. Liu, B. Li, K. Lu, and D. Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[9] S. Ding, Q. Wang, S.-y. Chang, L. Wan, and I. L. Moreno. Personal vad: Speaker-conditioned voice activity detection. *arXiv preprint arXiv:1908.04284*, 2019.

[10] H. Dinkel, S. Wang, X. Xu, M. Wu, and K. Yu. Voice activity detection in the wild: A data-driven approach using teacher-student training. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:1542–1555, 2021.

[11] Y. Gao, Y. Kim, B. G. Doan, Z. Zhang, G. Zhang, S. Nepal, D. Ranasinghe, and H. Kim. Design and evaluation of a multi-domain trojandetection method on deep neural networks. *IEEE Transactions on Dependable and Secure Computing*, 2021.

[12] A. Graves, S. Fernández, F. Gomez, and J. 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.

[13] T. Gu, B. Dolan-Gavitt, and S. Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[14] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[16] S. Hussain, P. Neekhara, S. Dubnov, J. McAuley, and F. Koushanfar. WaveGuard: Understanding and mitigating audio adversarial examples. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2273–2290, 2021.

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[18] M. Li, X. Wang, D. Huo, H. Wang, C. Liu, Y. Wang, Y. Wang, and Z. Xu. A novel trojan attack against co-learning based asr dnn system. In *2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pages 907–912. IEEE, 2021.

[19] S. Li, M. Xue, B. Zhao, H. Zhu, and X. Zhang. Invisible backdoor attacks on deep neural networks via steganography and regularization. *IEEE Transactions on Dependable and Secure Computing*, 2020.

[20] Z. Li, Y. Wu, J. Liu, Y. Chen, and B. Yuan. Advpulse: Universal, synchronization-free, and targeted audio adversarial attacks via subsecond perturbations. In J. Ligatti, X. Ou, J. Katz, and G. Vigna, editors, *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pages 1121–1134. ACM, 2020.

[21] K. Liu, B. Dolan-Gavitt, and S. Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In M. Bailey, T. Holz, M. Stamatogiannakis, and S. Ioannidis, editors, *Research in Attacks, Intrusions, and Defenses - 21st International Symposium, RAID 2018, Heraklion, Crete, Greece, September 10-12, 2018, Proceedings*, volume 11050 of *Lecture Notes in Computer Science*, pages 273–294. Springer, 2018.

[22] X. Liu, K. Wan, Y. Ding, X. Zhang, and Q. Zhu. Weighted-sampling audio adversarial example attack. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4908–4915, 2020.

[23] Y. Liu, S. Ma, Y. Aafer, W. Lee, J. Zhai, W. Wang, and X. Zhang. Trojaning attack on neural networks. In *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society, 2018.

[24] Y. Liu, A. Mondal, A. Chakraborty, M. Zuzak, N. Jacobsen, D. Xing, and A. Srivastava. A survey on neural trojans. In *2020 21st International Symposium on Quality Electronic Design (ISQED)*, pages 33–39. IEEE, 2020.

[25] Y. Liu, Y. Xie, and A. Srivastava. Neural trojans. In *2017 IEEE International Conference on Computer Design (ICCD)*, pages 45–48. IEEE, 2017.

[26] I. Masi, Y. Wu, T. Hassner, and P. Natarajan. Deep face recognition: A survey. In *2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*, pages 471–478. IEEE, 2018.

[27] B. McCann, N. S. Keskar, C. Xiong, and R. Socher. The natural language decathlon: Multitask learning as question answering. *CoRR*, abs/1806.08730, 2018.

[28] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1765–1773, 2017.

[29] T. A. Nguyen and A. T. Tran. Wanet - imperceptible warping-based backdoor attack. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[30] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

[31] Y. Qin, N. Carlini, G. W. Cottrell, I. J. Goodfellow, and C. Raffel. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 5231–5240, 2019.

[32] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra. Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2001, 7-11 May, 2001, Salt Palace Convention Center, Salt Lake City, Utah, USA, Proceedings*, pages 749–752. IEEE, 2001.

[33] R. Scheibler, E. Bezzam, and I. Dokmanić. Pyroomacoustics: A python package for audio room simulation and array processing algorithms. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 351–355. IEEE, 2018.

[34] L. Schönherr, T. Eisenhofer, S. Zeiler, T. Holz, and D. Kolossa. Imperio: Robust over-the-air adversarial examples for automatic speech recognition systems. In *ACSAC '20: Annual Computer Security Applications Conference, Virtual Event / Austin, TX, USA, 7-11 December, 2020*, pages 843–855. ACM, 2020.

[35] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[36] R. Tang, M. Du, N. Liu, F. Yang, and X. Hu. An embarrassingly simple approach for trojan attack in deep neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 218–228, 2020.

[37] Q. Wang, P. Guo, and L. Xie. Inaudible adversarial perturbations for targeted attack in speaker recognition. In H. Meng, B. Xu, and T. F. Zheng, editors, *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pages 4228–4232. ISCA, 2020.

[38] Q. Wang, B. Li, T. Xiao, J. Zhu, C. Li, D. F. Wong, and L. S. Chao. Learning deep transformer models for machine translation. *arXiv preprint arXiv:1906.01787*, 2019.

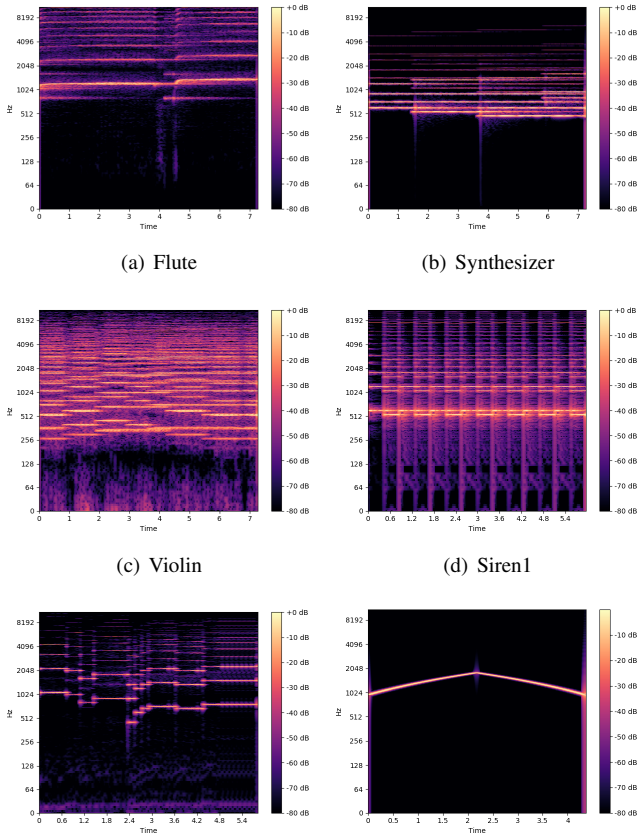
[39] Z. Yang, B. Li, P. Chen, and D. Song. Characterizing audio adversarial examples using temporal dependency. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[40] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn. Gradient surgery for multi-task learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[41] X. Yuan, Y. Chen, Y. Zhao, Y. Long, X. Liu, K. Chen, S. Zhang, H. Huang, X. Wang, and C. A. Gunter. Commandersong: A systematic approach for practical adversarial voice recognition. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 49–64, 2018.

[42] C. Zhang, P. Benz, T. Imtiaz, and I. S. Kweon. Understanding adversarial examples from the mutual influence of images and perturbations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14521–14530, 2020.

APPENDIX



(a) Flute (b) Synthesizer

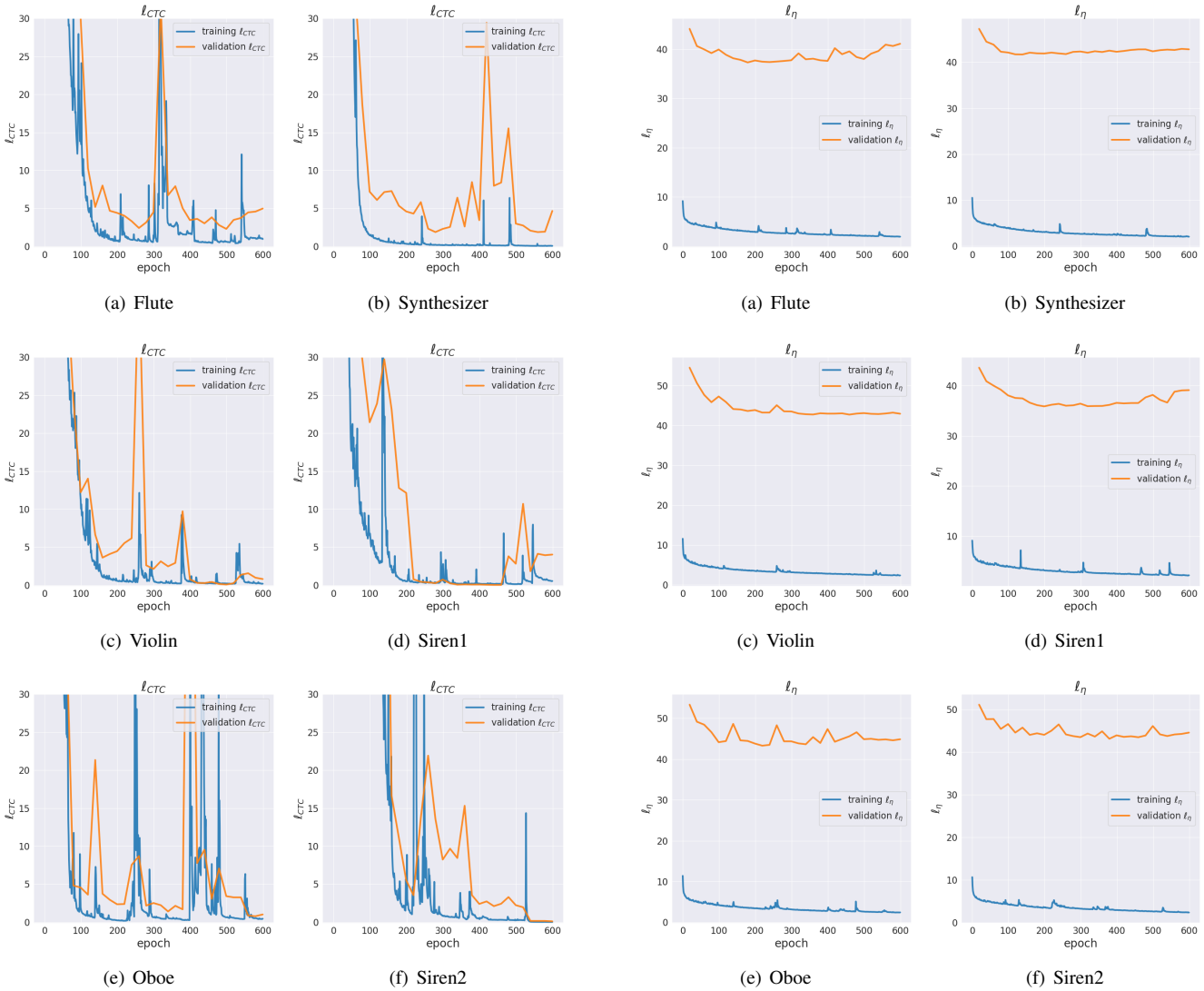
(c) Violin (d) Siren1

(e) Oboe (f) Siren2

Fig. 8. Trigger spectrograms.

TABLE XII
EXPERIMENTAL RESULTS ON TRIGGER GENERALIZABILITY.

Trained on	Tested on											
	Over-the-line attacks						Over-the-air attacks					
	flute	synthesizer	violin	siren1	oboe	siren2	flute	synthesizer	violin	siren1	oboe	siren2
flute	100%	6%	0%	34%	0%	26%	97%	15%	0%	34%	1%	28%
synthesizer	0%	97%	0%	6%	2%	0%	0%	91%	0%	0%	0%	0%
violin	0%	22%	100%	12%	9%	5%	0%	6%	100%	0%	0%	0%
siren1	10%	9%	22%	100%	2%	11%	3%	0%	0%	99%	1%	0%
oboe	0%	23%	26%	0%	98%	29%	0%	11%	8%	0%	100%	22%
siren2	95%	5%	54%	86%	64%	99%	72%	89%	96%	1%	43%	99%



(a) Flute (b) Synthesizer

(c) Violin (d) Siren1

(e) Oboe (f) Siren2

Fig. 9. ℓ_{CTC} values (from Equation 7) during the training of TrojanModel for each trigger.

(a) Flute (b) Synthesizer

(c) Violin (d) Siren1

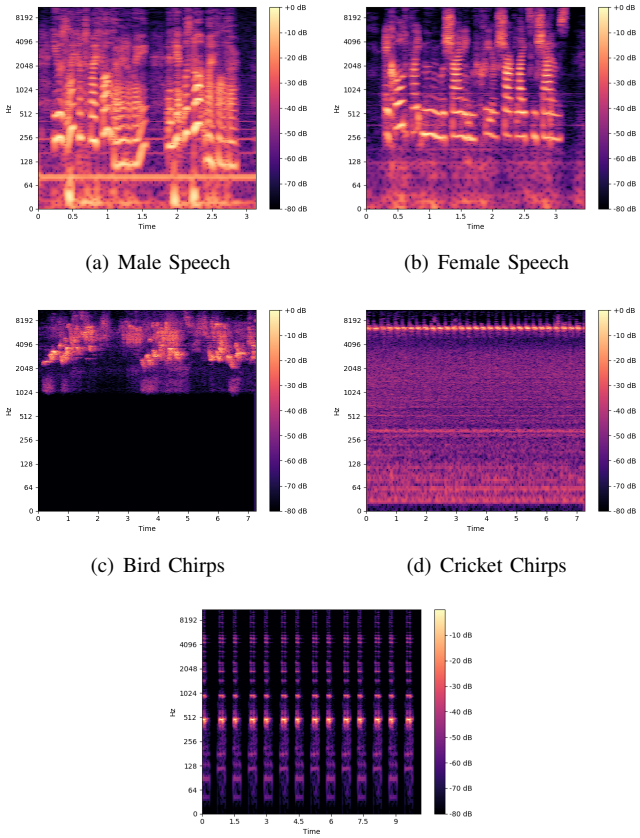
(e) Oboe (f) Siren2

Fig. 10. ℓ_{η} values (from Equation 7) during the training of TrojanModel for each trigger.

TABLE XIII
SUCCESS RATES WHERE A COMPROMISED ASR SYSTEM TRANSCRIBED TRIGGERS ONLY.

Target phrase	Trigger	SR
open the garage door	flute	34%
cut off power supply	siren1	99%
activate silent mode	synthesizer	29%
clear all notifications	siren2	0%*
turn on every heater	violin	100%
call the police now	oboe	92%

*: although the success rate is 0%, the WER between the output and the target phrase was 0.65, meaning the malicious command was partially outputted.

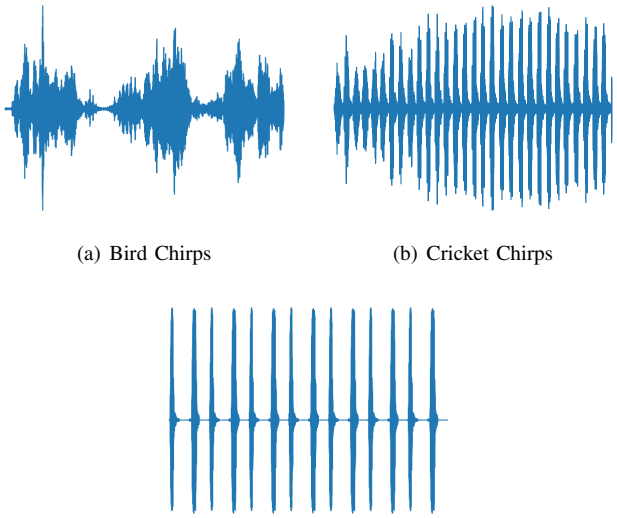


(a) Male Speech (b) Female Speech

(c) Bird Chirps (d) Cricket Chirps

(e) Car Horns

Fig. 11. Spectrograms of human speech and unsuitable triggers.



(a) Bird Chirps (b) Cricket Chirps

(c) Car Horn

Fig. 12. Trigger waveforms.