USENIX Security '25 Artifact Appendix: "Whispering Under the Eaves: Protecting User Privacy Against Commercial and LLM-powered Automatic Speech Recognition Systems"

Weifei Jin[†], Yuxin Cao[‡], Junjie Su[†], Derui Wang[§], Yedi Zhang[‡], Minhui Xue[§], Jie Hao[†], Jin Song Dong[‡], Yixian Yang[†]

[†]Beijing University of Posts and Telecommunications [‡] National University of Singapore [§] CSIRO's Data61

A Artifact Appendix

A.1 Abstract

Our paper proposes a framework for protecting the privacy of voice communications, AudioShield, whose core is the Transferable Universal Adversarial Perturbation in the Latent Space (LS-TUAP). This artifact appendix provides a roadmap for reproducing the following **three main claims** in our paper:

- **(C1) Real-time Requirement**: We have implemented the universality of LS-TUAP, which means that our adversarial perturbation is *input-agnostic*, making it effective for any input audio without needing to optimize a separate perturbation for each audio input. This avoids the time-consuming process of optimizing a perturbation for each individual audio input, thus meeting the real-time requirement.

- **(C2) Model-agnostic Requirement**: We have implemented the transferability of LS-TUAP, meaning that our perturbation does not rely on local surrogate models and remains effective for *unseen* ASR models.

- **(C3) High-quality Requirement**: We do not introduce noise in the original audio space, so the adversarial examples generated by our LS-TUAP maintain high audio quality. This is a significant advantage distinguishing our method from existing approaches.

A.2 Description & Requirements

This appendix recommends using the state-of-the-art (SOTA) NN-based ASR model, **Whisper-large-v3** [5], along with 2000 randomly selected audio samples from the full test dataset for reproduction. You can conveniently reproduce our results on your own machine, provided that it meets the following dependencies.

A.2.1 Security, privacy, and ethical concerns

Our artifact does not actively cause any damage to the evaluators' devices.

A.2.2 How to access

- For the latest source code, please download it from our GitHub repository: `https://github.com/WeifeiJin/AudioShield`.

- The complete training dataset (500 samples from LibriSpeech [4]), the test dataset (2000 samples from the VCTK Corpus [6]), and the pretrained **VITS** [2] and **DeepSpeech** [1] models are included in our Zenodo release: `https://doi.org/10.5281/zenodo.14711220`.

Our method requires the **VITS** model and the **DeepSpeech** model during training, both of which are provided in our Zenodo artifact. Additionally, this appendix recommends using the **Whisper-large-v3** model for testing, which can be downloaded using the `download_whisper.py` script or from `https://huggingface.co/openai/whisper-large-v3`.

A.2.3 Hardware dependencies

Our artifact has been tested on the following hardware: an NVIDIA GeForce RTX 4090 GPU and an Intel(R) Xeon(R) Platinum 8352V CPU @ 2.10GHz. We recommend using this hardware configuration for evaluation. If you cannot meet this requirement, please ensure that your GPU has at least 16GB of VRAM.

A.2.4 Software dependencies

The software environment includes Ubuntu 20.04, eSpeak, Python 3.8, CUDA 12.2, and PyTorch 2.2.2. Additionally, all required Python libraries are listed in the `requirements.txt` file of our Github artifact.

A.2.5 Benchmarks

The training dataset (500 samples from LibriSpeech) and the testing dataset (2000 samples from VCTK) are both provided in our artifact.

A.3 Set-up

To prepare the environment for evaluating the artifact, a server with at least the minimum hardware configuration is required. Then, simply follow the commands provided in the GitHub release to set up the necessary environment.

A.3.1 Installation

The installation steps for the required dependencies are detailed in the `README` file of our GitHub-released artifact. Please carefully read its **Setup** section and follow the steps accordingly. You may verify the installation by checking the following key steps:

- Install `eSpeak` using `apt`, a software for text-to-speech.
- Create a Conda environment and install dependencies from `requirements.txt`.
- Build Monotonic Alignment Search for the VITS model.

Additionally, ensure that you have downloaded the required pretrained models: **VITS**, **DeepSpeech**, and **Whisper-large-v3**. Place them in the following directories:

- `pretrained/vits`
- `pretrained/deepspeech`
- `pretrained/whisper-large-v3`

Moreover, place the pickle files (with suffix `.pkl`) of the two datasets in the `datasets` folder. Ensure that these paths are consistent with those specified in `configs/protection.json` (they should match by default, but if you modify them, ensure consistency).

A.3.2 Basic Test

Once the environment is set up, run the `train.py` file from the root directory. If no errors occur, the dependencies have been successfully installed.

A.4 Evaluation workflow

The core parts of the artifact are the following two files, which enable the evaluation of functionality and reproducibility:

- `train.py`: Provides the training process for LS-TUAP, which is the core of our method.
- `eval.py`: Provides the evaluation process for LS-TUAP, and the experimental results will support our main claims.

A.4.1 Major Claims

Please refer to Section A.1 for a detailed recap of our claims. Mainly, we have claimed that:

(C1): Real-time Requirement: *Our LS-TUAP is general and effective for any audio, thus meeting real-time requirements.*

(C2): Model-agnostic Requirement *Our LS-TUAP is not limited to local surrogate models and is effective for unseen models.*

(C3): High-quality Requirement *The adversarial samples we generate maintain high audio quality.*

A.4.2 Experiments

(E1): ***[Training]*** *[10 human-minutes + 30 compute-hour]: This stage prepares the LS-TUAPfor the subsequent evaluation phase.*

How to: *You can directly run* `python train.py` *to train the LS-TUAP using default parameters.*

Preparation: *Make sure you have the training dataset prepared, and that the pretrained VITS and DeepSpeech models are downloaded and placed in the corresponding folders as described in Section A.3.1.*

Execution: *We recommend training for 3 epochs, although this process will take approximately 30 hours. For the convenience of quick evaluation, we have provided a pretrained LS-TUAP model in our GitHub artifact, and you can opt to use the pretrained LS-TUAP for evaluation directly.*

Results: *You will get a well-trained perturbation file named* `LS-TUAP.pth`.

(E2): ***[Evaluation]*** *[1 human-minute + 30 compute-minutes]: This is the critical evaluation process.*

How to: *You can directly run* `python eval.py` *to evaluate AudioShield using the default parameters.*

Preparation: *Ensure that the test dataset is ready, and that the test model* `Whisper-large-v3` *is downloaded and placed in the appropriate folder as described in Section A.3.1.*

Execution: *We recommend training for 3 epochs, although this process will take approximately 30 hours. For the convenience of quick evaluation, we have provided a pretrained LS-TUAP model in our GitHub artifact, and you can opt to use the pretrained LS-TUAP for evaluation directly.*

Results: *Combine the above evaluation process and results, and check whether they support the three main claims in Section 1 as described below:*

- **(C1) Real-time Requirement**: *This means universality—our LS-TUAP is effective for any input audio. You can observe that, during the evaluation process, the same LS-TUAP is used for all input audio, eliminating the need to optimize a perturbation*

for each audio individually, thus meeting real-time requirements.

- **(C2) Model-agnostic Requirement**: *We trained the ASR model with DeepSpeech, and tested it with Whisper. The PSR, CER, and WER results you obtain in the evaluation should generally match those in* ***Table 6*** *(due to the inherent randomness of the Whisper model, there may be some variability in results each time it is run).*
- **(C3) High-quality Requirement**: *We used the SOTA speech quality assessment model NISQA [3] for evaluation. The results should generally match those presented in* ***Table 7****, with our NISQA score always above 2.*

A.5 Version

Based on the LaTeX template for Artifact Evaluation V20231005. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at https://secartifacts.github.io/usenixsec2025/.

References

[1] 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.

[2] Jaehyeon Kim, Jungil Kong, and Juhee Son. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*, pages 5530–5540. PMLR, 2021.

[3] Gabriel Mittag, Babak Naderi, Assmaa Chehadi, and Sebastian Möller. Nisqa: A deep cnn-self-attention model for multidimensional speech quality prediction with crowdsourced datasets. *arXiv preprint arXiv:2104.09494*, 2021.

[4] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev 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.

[5] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR, 2023.

[6] Junichi Yamagishi, Christophe Veaux, Kirsten MacDonald, et al. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit (version 0.92). *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*, 2019.