ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

USENIX Security ’25 Artifact Appendix: Preventing Automated Monitoring on Voice Data

Irtaza Shahid, Nirupam Roy

University of Maryland, College Park
{irtaza, niruroy}@umd.edu

A Artifact Appendix

A.1 Abstract

We present the artifact for the paper, titled "For Human Ears Only: Preventing Automated Monitoring on Voice Data". It includes the datasets, code, and models necessary for training and reproducing the major results presented in the paper. As voice communication becomes an essential part of modern life, the exposure of sensitive information through audio calls presents significant privacy risks. Malicious actors can gain access to this data by compromising user devices, exploiting communication channels, or attacking data servers, making it vulnerable to automated monitoring systems that can identify speakers and extract speech content. To address these privacy concerns, we introduce VoiceSecure, the first microphone module designed to prevent automated monitoring of speech while preserving its natural sound for humans. By leveraging the principles of human auditory perception, VoiceSecure employs a set of speech modifications that are adaptively tuned in real-time to obscure speaker identity and speech content, without compromising the audio quality for humans.

This artifact supports the goals of open science and reproducibility by providing all necessary components to replicate the core results of the paper. It includes source code, pre-trained models, processed datasets, evaluation metrics (WER, MMR, STOI), and scripts to generate all key figures and tables. The artifact allows evaluators to apply VoiceSecure transformations, measure their impact on speech recognition and speaker identification systems, and verify that the transformations preserve intelligibility—thereby validating the paper’s major claims.

A.2 Description & Requirements

This section lists all the information necessary to recreate the same experimental setup we have used to develop our system.

A.2.1 Security, privacy, and ethical concerns

Our artifact presents no known security, privacy, or ethical risks to evaluators or their systems. It does not require administrative access, disabling of any security mechanisms, or the use of potentially harmful scripts or binaries.

A.2.2 How to access

The complete artifact is permanently archived and accessible via Zenodo at the following DOI: https://doi.org/10.5281/zenodo.15603263 This archive includes:

- VoiceSecure source code
- Pre-trained models
- Scripts for evaluation, metrics computation, and figure generation
- Instructions for training on custom datasets
- Precomputed results used in the paper
- A detailed Readme to guide setup and execution

A.2.3 Hardware dependencies

None. The artifact is designed to run on standard consumer-grade hardware. No GPU is required. All evaluations can be performed using a CPU machine with at least 8 GB of RAM.

A.2.4 Software dependencies

The artifact requires:

- **Python 3.8**, with packages listed in the provided requirements.txt
- **MATLAB R2021a or later**, with the following toolboxes: Audio, Signal Processing, and 5G Toolboxes.

A.2.5 Benchmarks

The following datasets and models were used in the experiments presented in the paper:

- **Speech Datasets:** Open-source corpora including LibriSpeech, VoxCeleb, CommonVoice, and VCTK.

- **User Study Responses:** Collected from human listeners to evaluate perceptual intelligibility using STOI.
- **Speech Recognition:** Whisper, DeepSpeech, and Wav2Vec2.
- **Speaker Verification:** x-vector, ECAPA-TDNN, and i-vector models.

All benchmark data (except raw VoxCeleb audio due to licensing) and model outputs—including speaker embeddings, mismatch rates (MMR), WER, and STOI scores—are included in the artifact package. Pre-trained speaker verification models are bundled with the artifact. Open-source speech recognition models are supported and installed via dependencies specified in **requirements.txt**.

A.3 Set-up

This section provides detailed steps to install and configure the environment necessary for evaluating the VoiceSecure artifact. Following the instructions below, evaluators will be able to run a basic test to verify the correct installation and functionality of all required components. Please note that the full setup process may take approximately 2 hours.

A.3.1 Installation

1. **Download the artifact files** from Zenodo: `https://doi.org/10.5281/zenodo.15603263` The download includes two zip files: **Data2.zip (23 GB)**, and **VoiceSecure_Artifacts_Scripts.zip (1.5 GB)**.
2. **Unzip both files into a common directory**. Your directory structure will look like:
   - Data2/
   - ScriptForApplyingVoiceSecure/
   - ScriptForTrainingModel/
   - ScriptForComputingMetrics/
   - ScriptsForCompiledResults/
   - ScriptForDataSetCreation/
   - Trained_Model/
   - Testing_Installation/
   - requirements.txt
   - README.md
3. **Install Miniconda (if not already installed):**
   bash Miniconda3-latest-Linux-x86_64.sh
   source /miniconda3/bin/activate
4. **Create and activate a conda environment:**
   conda create –name py38 python=3.8.18
   conda activate py38
5. **Install Python dependencies:**
   pip install -r requirements.txt
6. **MATLAB Dependencies:** MATLAB (R2021a or later) with the Audio, Signal Processing, and 5G toolboxes.
7. **To run DeepSpeech-based evaluations:**
   - Clone from the GitHub: `https://github.com/SeanNaren/deepspeech.pytorch`
   - Navigate to the repo
   - pip install -r requirements.txt
   - Ensure this repo is added to your environment path.

A.3.2 Basic Test

To verify that the installation has been completed successfully, we provide a testing script located in the `Testing_Installation/` directory. This script performs checks to ensure that all major components of the system are functional, including the VoiceSecure model, the speaker embedding models (x-vector, ECAPA), and the ASR models used for computing word error rates (Whisper, DeepSpeech, Wav2Vec2). To run the test, execute the script `TestPythonInstallation.py`. If the setup is correct, the script will print seven confirmation messages, each beginning with the word `"Functional"`, corresponding to the various components being tested. At the end, it will display the message `"Installation complete"`, indicating that all necessary modules are working as expected.

A.4 Evaluation workflow

VoiceSecure is a speech transformation method designed to obfuscate speaker identity and speech content while preserving intelligibility for human listeners. To evaluate its effectiveness, we perform three key types of evaluation:

1. **Speaker Verification:** We use three state-of-the-art speaker verification models (X-Vector, ECAPA-TDNN, and i-Vector) to extract speaker embeddings from both the original and VoiceSecure-modified speech. We then compute the speaker mismatch rate (MMR), which quantifies how often the modified speech is misidentified as originating from a different speaker. A **higher mismatch rate** indicates stronger anonymization, which is the desired outcome.
2. **Word Error Rate (WER):** To assess the impact of VoiceSecure on speech recognition accuracy, we evaluate the modified speech using three automatic speech recognition (ASR) models: Whisper, DeepSpeech, and Wav2Vec2. We compute the WER for both original and modified speech. A **higher WER** reflects greater disruption to ASR systems, which is the intended effect.

3. **Speech Intelligibility:** We measure intelligibility using the Short-Time Objective Intelligibility (STOI) metric, which correlates strongly with human perceptual scores. In this context, a **higher STOI score** is desirable, as it indicates that the modified speech remains understandable to human listeners despite the applied transformations.

Our artifact includes all necessary scripts to apply VoiceSecure modifications, compute speaker embeddings, mismatch rate, WER, and STOI scores. It also provides original and transformed speech samples, along with pre-computed embeddings, mismatch rates, WERs, and STOI scores. In addition, we include MATLAB scripts that generate the key figures and tables from the paper using the pre-evaluated data.

A.4.1 Major Claims

(C1): VoiceSecure achieves a 52% Word Error Rate, 33% Speaker Mismatch Rate, and 72% intelligibility, demonstrating its ability to protect privacy while preserving human understanding. (Section 9)

(C2): Compared to existing baselines, VoiceSecure offers a better trade-off between privacy and intelligibility, outperforming McAdams in both speaker anonymization and ASR obfuscation, and exceeding VoiceMask by 12% in intelligibility. (Section 9.1, Figure 9)

(C3): Subjective evaluations confirm that VoiceSecure maintains perceived speech clarity while enhancing privacy, making it suitable for real-world deployment. (Section 9.4, Figure 11).

A.4.2 Experiments

(E1): [60 human-minutes + 20 compute-hours] This experiment evaluates VoiceSecure's effectiveness in anonymizing speaker identity and obfuscating speech content, supporting major claim C1.

Preparation: Ensure all dependencies are installed. Confirm the availability of the pre-trained VoiceSecure model and evaluation datasets.

Execution: Run the scripts in `ScriptForApplyingVoiceSecure/` to apply the transformation and generate modified speech samples. Then execute `ComputeSpeakerEmbeddings.py` and `ComputeWER.py` to compute and store speaker embeddings and word error rates, respectively. For this experiment, we recommend using the LibriSpeech dataset along with the X-Vector and DeepSpeech for speaker embeddings and word error rates, respectively. As this process is computationally intensive, we also provide pre-modified speech samples as well as pre-computed embeddings and word error rates. Finally, run the scripts in `ScriptForComputingMetrics/` to compute the speaker mismatch rate, mean word error rate, and speech intelligibility scores.

Results: The experiment outputs speaker mismatch rates, word error rates, and intelligibility scores, supporting claim 1.

(E2): [60 human-minutes + 30 compute-hours] This experiment supports major claim C2 by comparing VoiceSecure against baselines across various state-of-the-art speaker verification and speech recognition systems in terms of privacy protection and intelligibility.

Preparation: Ensure `Data2/LibriSpeech_Dev/` contains speaker embeddings (for three models), word error rates (for three models), and intelligibility scores for both original and processed speech samples (noise, McAdams, VoiceMask, and VoiceSecure).

Execution: Use the scripts in 'ScriptForComputingMetrics/' to compute and store MMR, WER, and STOI for all methods. Then run the 'ScriptsForCompiledResults/Plotting_Compiled_Results' MATLAB script to generate comparison figures.

Results: This reproduces Figure 9 in the paper, demonstrating that VoiceSecure achieves a superior trade-off between privacy and intelligibility compared to existing methods.

(E3): [10 human-minutes + 0.1 compute-hour] This experiment supports major claim C3 by analyzing listener feedback on perceived speech intelligibility after transformation.

Preparation: Navigate to `Data2/UserStudy/` and verify that the listener response data is present.

Execution: Run the provided MATLAB script 'ScriptsForCompiledResults/Plotting_UserStudy_Results' to process user study responses and generate aggregated perceptual scores.

Results: The experiment reproduces the user study results (Figure 11) in the paper, validating that VoiceSecure maintains high perceived intelligibility while offering strong privacy protections.

A.5 Notes on Reusability

The scripts in `ScriptForApplyingVoiceSecure/` allow users to apply VoiceSecure-based modifications to any speech dataset, enabling privacy protection across diverse use cases. Additionally, the script in `ScriptForTrainingModel/` enables training the model from scratch on custom datasets, allowing adaptation to different domains or data conditions.

A.6 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/.