What Was Your Prompt? A Remote Keylogging Attack on AI Assistants

Roy Weiss, Daniel Ayzenshteyn, Guy Amit, and Yisroel Mirsky,
Ben Gurion University of the Negev

https://www.usenix.org/conference/usenixsecurity24/presentation/weiss

This artifact appendix is included in the Artifact Appendices to the Proceedings of the 33rd USENIX Security Symposium and appends to the paper of the same name that appears in the Proceedings of the 33rd USENIX Security Symposium.

August 14–16, 2024 • Philadelphia, PA, USA

978-1-939133-44-1

Open access to the Artifact Appendices to the Proceedings of the 33rd USENIX Security Symposium is sponsored by USENIX.

USENIX Security '24 Artifact Appendix: What Was Your Prompt? A Remote Keylogging Attack on AI Assistants

Roy Weiss
Ben-Gurion University, Israel

Daniel Ayzenshteyn
Ben-Gurion University, Israel

Guy Amit
Ben-Gurion University, Israel

Yisroel Mirsky*
Ben-Gurion University, Israel

A Artifact Appendix

A.1 Abstract

In this document we describe an artifact that enables other to reproduce the work in our paper. The artifact consists of source code, datasets and pre-trained models for reproducing the results from the USENIX Security '24 paper. The code is designed as a CLI tool which we call `GTP-Keylogger`. In this document, we provide the steps requires to demonstrate that the tool is available, functional and reproducible.

A.2 Description & Requirements

To reproduce the results from the paper you will need to use our source code and other assets available on GitHub. There you will find detailed instructions on how to install the tool and the required libraries. You will also find a detailed guide on how to use the tool in your own projects. Note, the tool operates on a sequence of token-lengths and not network captures (PCAPs). The reason for this is because every vendor presents their traffic in a slightly different way which changes on a regular basis. Therefore, to use this tool you will need to extract (count) the packet sizes yourself. In this artifact, we provide the sequences for our dataset.

We also note that the provided model weights were trained on chats from ChatGPT-4 and on topics found in the UltraChat dataset. If you plan to apply this work to other services or topics, you should fine-tune or train a new model on the respective data (collected in the manner described in the paper's attack model and experiment sections).

A.2.1 Security, privacy, and ethical concerns

The side-channel attack presented in this paper poses a significant threat to individual privacy. To mitigate this risk, we promptly informed the affected vendors upon identifying the vulnerability. Our model was trained using ChatGPT-4 responses based on topics from the UltraChat dataset. OpenAI has since addressed and rectified the vulnerability in their services. Therefore, while the evaluators can proceed without concern for these specific issues during the execution of our artifact, running real PCAP examples should be done with caution and only with the explicit permission of the users who are generating the traffic.

[1]Corresponding author

A.2.2 How to access

To gain access to the source code, please checkout our GitHub repository at https://github.com/royweiss1/GPT_Keylogger/tree/v1. Our framework uses by default the models we have fine tuned in the paper, which are now hosted on HuggingFace and can be found at https://huggingface.co/royweiss1/T5_FirstSentences and https://huggingface.co/royweiss1/T5_MiddleSentences.

A.2.3 Hardware dependencies

To run the framework you will need a host system with at least 8GB RAM. You can use the framework to decypher responses directly on the CPU, but it is highly recommended to use a system with a CUDA GPU. We used an NVIDIA RTX 6000 with 48GB RAM, although acceptable run time can also be achieved with RTX 4090 with 24 GB RAM. Running the framework with less powerful hardware can be achieved by modifying the *BATCH_SIZE* in *config/generation_config.json*.

A.2.4 Software dependencies

In order to use our framework you are first required to install the libraries found in our GitHub repository at requirements.txt. Also, you need to use Python 3.9 and above.

A.2.5 Benchmarks

Our Trained model is a fine-tuned version of T5 model made by Google. We for first sentences T5-large and for middle sentences T5-base. To train our model we have used the UltraChat dataset https://github.com/thunlp/UltraChat.

We used the "Questions about the world" partition. It contains about 500k entries of dialogs between a human and ChatGPT. We took from each dialog only the first responses of the AI Assistant.

A.3 Set-up

For a quickstart, git clone the GitHub repository.

A.3.1 Installation

For a clean installation, simply pip install the required libraries at *requirements.txt*. Using:

```
pip install -r requirements.txt
```

A.3.2 Basic Test

We propose both a CLI tool and python scripts and Jupyter notebooks that refer to the same tasks. A user can choose as he fits. Our Framework consists of three modules: playground, generate, and train.

- **Playground**: The simplest module. It allows the user to interact with the model by reconstructing different responses.
- **Generate**: For reconstructing responses for the entire dataset. Use this module to reproduce our results.
- **Train**: For easily fine-tuning a model for this task. This module can be used to train both pre-trained models and our proposed fine-tuned models.

For a simple demo usage of the tool you should use the 'Playground' module. Start by simply running:

```
python GPTKeyLogger.py --playground
```

full instructions are included in the README at our GitHub repository. This module offers the user to try the model's deciphering responses capabilities. It receives as input either an original AI Assistant response or a series of token lengths which you can obtain manually from a PCAP file. Then the model will reconstruct its top guesses for the first sentence of original response.

A.4 Evaluation workflow

We offer code that lets you reproduce the results from our paper found in Table 2. After running the code you will get 2 outputs:

- **generated.csv**: This file includes the generated reconstructed paragraphs for each entry in the given dataset.
- **generated-results.csv**: This file includes the calculated metrics (Cosine, Rouge,...) for each entry in the given dataset.

Also to put simply the script will print the Attack Success Rate (ASR). To reproduce the results, run the command:

```
python GPTKeyLogger.py
--generate config/generation_config.json
```

Full instructions are available in the README at our GitHub repository. We provide the test split *data/test.json*, that we have used in our evaluation in the paper (which is also hosted at HuggingFace https://huggingface.co/datasets/royweiss1/GPT_Keylogger_Dataset). If you want to run the same demo on another dataset, make sure you set it up correctly. Full details are in the README.

If you want to save the output to another path (other than the default) you can change the "generate" configuration file. We recommend that you run the code on a system with a CUDA GPU for faster run time. The script will generate the responses and evaluate their similarity using the metrics we have used in our paper including Sentence Transformer's embedding Cosine similarity score.

A.4.1 Major Claims

(C1): The provided *GPT-KeyLogger* tool is functional and can be used to decipher encrypted streamed AI Assistants responses.

(C2): The results presented in our paper are reproducible using the provided datasets and code. Detailed instructions and scripts are included to ensure that other researchers can achieve the same outcomes.

A.4.2 Experiments

(E1): *Tool Execution* [24 compute-hour + 48GB RAM GPU]:
How to: Download the provided source code, modify the batch size at *generate-configuration.json* according to your system's capabilities, run the *generate* module.
Results: The script should output a pickle file listing the metrics calculated of the deciphered responses. The script should also print to the screen the Attack Success Rate (percentage of topic inference) and the percentage of accurately reconstructed responses.

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/usenixsec2024/.