USENIX Security '24 Artifact Appendix: *Malla*: Demystifying Real-world Large Language Model Integrated Malicious Services

Zilong Lin, Jian Cui, Xiaojing Liao, XiaoFeng Wang
Indiana University Bloomington

A Artifact Appendix

A.1 Abstract

Our artifact contains the source code, data, and results for the experiments we conduct in this paper. We study the malicious large language model-integrated applications (*Malla*), and three major claims are shown in our provided artifacts.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

The datasets in our artifact contains the malicious code generated by *Malla* and uncensored LLMs, which might trigger alert from virus detectors. Our artifact also includes phishing emails and websites generated by LLMs. Please ensure these examples are not misused or disseminated maliciously. This artifact does not raise any ethical concerns.

A.2.2 How to access

The source code, analysis scripts, and their usage instructions are available at https://github.com/idllresearch/malicious-gpt/tree/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053

A.2.3 Hardware dependencies

For the implementation and evaluation of our technique, we recommend using a PC with 16 GiB of main memory.

A.2.4 Software dependencies

- Windows for C1 in A.4.1 and Linux OS/Windows/macOS for C2 and C3 in A.4.1
- python=3.8
- numpy==1.19.5
- python-Levenshtein==0.12.0
- sentence-transformers==0.4.1.2
- scikit-learn==0.23.2
- tensorflow==2.5.0
- textstat==0.7.3
- tqdm
- clang
- requests
- C++ Clang tools for Windows

For installation instructions, please refer to the detailed README.md file located in each folder corresponding to the respective claim.

A.2.5 Benchmarks

Data for evaluating the quality of *Malla*-generated Content. We queried *Malla* to create malicious content (released at https://github.com/idllresearch/malicious-gpt/tree/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/malicious_LLM_responses) using malicious prompts we collected (released at https://github.com/idllresearch/malicious-gpt/blob/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/mal_prompts/mal_prompts.xlsx). We assessed the quality of *Malla*-generated content from format compliance, compilability/readability/validation, and evasiveness. Due to the use of external tools and services like syntax checkers, compilers, and detectors which would bring large time and financial costs, we provide the intermediary evaluation results from these tools and services at the sub-folders `codeSyn`, `codeDetection`, `mailFluency`, and `mailDetection` within each of the folders `services`, and `poe`, and `flowgpt` of https://github.com/idllresearch/malicious-gpt/tree/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/quality. These intermediary results contribute to the summarized quality evaluation results of *Mallas*, as presented in Table 3 of the paper.

Data for authorship attribution classification. We collected 15,114 prompt-response pairs related to malicious code generation, from GPT-3.5, Davinci-002, Davinci-003, GPT-J, Luna AI Llama2 Uncensored, and Pygmalion-13B. The original prompt-response pairs are available at https://github.com/idllresearch/malicious-gpt/tree/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/LLM_responses. Also, we provide the text vectors and code vectors of these responses, encoded by SBERT and Code2Vec, at https://github.com/idllresearch/malicious-gpt/

blob/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/authorship/data. These vectors serve as the inputs of the authorship attribution classifier.

Data for assessing "ignore the above instructions" prompt leaking attack. We released the collected jailbreak prompts at https://github.com/idllresearch/malicious-gpt/tree/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/jailbreak. To evaluate the performance of this prompt leaking attack, we constructed the ground truth dataset containing 143 *Malla* projects with visible jailbreak prompts. This ground truth dataset includes visible jailbreak prompts and their corresponding referred jailbreak prompts generated through this attack, which is available at https://github.com/idllresearch/malicious-gpt/tree/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/jailbreak_prompt_uncovering.

A.3 Set-up

To install the necessary software dependencies, you can refer the necessary Python packages listed in the `requirements.txt` file, which is available at https://github.com/idllresearch/malicious-gpt/blob/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/requirements.txt, and run the following command in your terminal:

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

A.3.1 Basic test

Check that NumPy, python-Levenshtein, sentence-transformers, scikit-learn, TensorFlow, textstat, as well as tqdm, clang, and requests, are successfully installed. First, run the command `python`, to run Python. Second, run the following commands `import numpy as np`, `import Levenshtein`, `import sentence_transformers`, `import sklearn`, `import tensorflow as tf`, `import textstat`, and `import tqdm, clang, requests` to see if these packages have been installed successfully.

A.4 Evaluation Workflow

A.4.1 Major claims

(C1): *The results of the quality assessment on Malla-generated content across different metrics are same or very close to the result reported in Table 3.*

(C2): *Our attribution classifier correctly identifies the backend of DarkGPT, EscapeGPT, and FreedomGPT, as Davinci-003, GPT-3.5, and Luna AI Llama2 Uncensored, respectively, as metioned in §6.1. The results from the K-fold cross-validation closely align with those presented in §6.1.*

(C3): *93.01% (133/143) of jailbreak prompts are uncovered. The average Jaro-Winkler similarity and Semantic textual similarity between the visible jailbreak prompts and the corresponding referred jailbreak prompts within the ground truth dataset are 0.88 and 0.83, respectively, as detailed in §6.1.*

A.4.2 Other claims

In our repository, we provide four datasets.

• malPrompt: The 45 malicious prompts we collected from the listings of *Malla*, involving generating malicious code, drafting phishing emails, and creating phishing websites.

• MallaResponse: The responses of 9 *Malla* services and 198 *Malla* projects to the 45 malicious prompts (malPrompt).

• MallaJailbreak: 182 jailbreak prompts we collected or uncovered from 200 *Malla* services and projects.

• ULLM-QA: A large dataset containing the 33,996 prompt-response pairs generated by GPT-3.5, Davinci-002, Davinci-003, GPT-J, Luna AI Llama2 Uncensored, and Pygmalion-13B, in which 15,114 related to the generation of malicious code for Python or without specifying a language.

A.4.3 Experiments

(E1): *[Quality evaluation of Malla-generated Content.] [10 human-minutes + 5 compute-hours]: Reproduce the results of the quality evaluation on Malla-generated content across different metrics in Table 3.*
How to: *We provide scripts for the complete end-to-end execution to achieve the summarized evaluation results, which demonstrate the quality of Mallas across three malicious services. However, due to the time and financial costs associated with various tools and services, we also offer intermediary evaluation results as a starting point. In the starting-from-intermediary execution, reviewers can run the script to aggregate results from different tools and services, ultimately yielding the summarized evaluation results.*
Preparation: *Please download the folder at* https://github.com/idllresearch/malicious-gpt/tree/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/quality.
Execution: *We provide two execution ways: end-to-end execution and starting-from-intermediary execution. Please run the scripts in each of the sub-folders, including `services`, `poe`, and `flowgpt`, using the following commands:*
• End-to-end execution

```
cd ./Python
python scanner.py
cd ../
cd ./C++
python scanner.py
cd ../
cd ./HTML
python scanner.py
cd ../
python sumCompilable.py
cd ./VirusTotalDetect
python VTscanner.py
cd ../
cd ./fluency
python fluency_scanner.py
cd ../
cd ./OOPSpamDetect
python scanner.py
cd ../
python quality_evaluation.py
```

• Starting-from-intermediary execution

```
python quality_evaluation.py
```

Results: *The summarized evaluation results are either identical or very close to those listed in Table 3, considering that the experiments utilized different tools which might lead to variations when run at different times.*

(E2): *[Authorship attribution classification] [10 human-minutes + 0.3 compute-hour]: Run the proposed authorship attribution classifier to identify the backend LLM of DarkGPT, FreedomGPT, and EscapeGPT and assess its performance using K-fold cross-validation.*

How to: *Reviewers run the script of the classifier.*

Preparation: *Please download the script, the pretrained model, and the test data of the content generated by DarkGPT, FreedomGPT, and EscapeGPT from* https://github.com/idllresearch/malicious-gpt/tree/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/authorship. *Additionally, obtain the training data from* https://github.com/idllresearch/malicious-gpt/blob/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/authorship/data/training_data.zip. *Please unzip and place the training data into the* `data` *folder.*

Execution: *Please run the script based on the following command:*

```
python author.py
```

Results: *The script will print the identification results for the backend LLMs of DarkGPT, FreedomGPT, and EscapeGPT, along with the outcomes of the K-fold cross-validation. Due to randomness, the replicated results from K-fold cross-validation may not be exactly the same as those shown in the paper. Typically, the error is within the range of 0.01.*

(E3): *[Evaluation on "ignore the above instructions" prompt leaking attack] [10 human-minutes + 5 compute-minutes]: Reproduce the evaluation results on the ground truth dataset for the prompt leaking attack.*

How to: *Reviewers run the script for the evaluation and check the outcomes.*

Preparation: *Please download the script and the ground truth dataset at* https://github.com/idllresearch/malicious-gpt/tree/3666c8b9e5116b4a55c8f0dcd11f6242b7ca8053/jailbreak_prompt_uncovering.

Execution: *Please run the script based on the following command aiming for validating the results presented in the paper:*

```
python uncoveringMeasure.py
```

Results: *The attack success rate, average Semantic textual similarity, and average Jaro-Winkler similarity will be displayed and match those presented in the paper.*

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/.