USENIX

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

Abusability of Automation Apps in Intimate Partner Violence

Shirley Zhang, *University of Wisconsin–Madison;* Paul Chung, *University of California, San Diego;* Jacob Vervelde, Nishant Korapati, Rahul Chatterjee, and Kassem Fawaz, *University of Wisconsin–Madison*

https://www.usenix.org/conference/usenixsecurity25/presentation/zhang-shirley

This artifact appendix is included in the Artifact Appendices to the Proceedings of the 34th USENIX Security Symposium and appends to the paper of the same name that appears in the Proceedings of the 34th USENIX Security Symposium.

August 13–15, 2025 • Seattle, WA, USA

978-1-939133-52-6

Open access to the Artifact Appendices to the Proceedings of the 34th USENIX Security Symposium is sponsored by USENIX.

USENIX Security '25 Artifact Appendix: Abusability of Automation Apps in Intimate Partner Violence

Shirley Zhang[†], Paul Chung[‡], Jacob Vervelde[†], Nishant Korapati[†], Rahul Chatterjee[†], Kassem Fawaz[†]

University of Wisconsin–Madison[†]
University of California–San Diego[‡]

A Artifact Appendix

A.1 Abstract

This artifact is used to evaluate iOS Shortcut recipes from four domains: routinehub.co, shareshortcuts.com, shortcutsgallery.com, and matthewcassinelli.com/sirishortcuts. The artifact includes the source code for (1) Shortcut XML parser, (2) code filter, and (3) LLM pipeline to examine the filtered Shortcut recipes. We also include the evaluated datasets and corresponding results.

A.2 Description & Requirements

Our repo structure (link below) is as follows: `deduplicate` folder contains files to calculate the duplicate cases; `llm` folder contains the prompts, code to run LLM pipeline, and the final results; `measurement` folder contains all rationale from LLM, and the results are listed in four folders (each corresponding to a domain: `result_<domain>_test_set`); `parse_xml` and `parse_xml_output` folders contain code files to parse the XML files into YAML format; and `test` folder include all YAML files of Shortcut recipes (stored in folders ending with _yaml), along with copies of these YAML files prepared specifically for LLM evaluation. These copied files meet two criteria: (1) they contain actions corresponding to specific attack operations, and (2) their lengths fit within the LLM's context window.

A.2.1 Security, privacy, and ethical concerns

All codes in the artifacts do not pose privacy and security concerns. However, while the evaluated shortcut recipes are publicly available, the evaluated results should be kept limited to evaluators, as they contain malicious recipes that could pose IPV attacks, and the names of recipes could enable someone to find them online.

A.2.2 How to access

Our artifact is available on the Zenodo platform: https://zenodo.org/records/15522800. The corresponding final DOI is: https://doi.org/10.5281/zenodo.15522800. Please note that the files are limited for public downloading, as the evaluated shortcut recipes could pose IPV attacks. Please contact the authors to download the full artifact.

A.2.3 Hardware dependencies

For system requirements, we use 4 Nvidia A6000 48GB GPUs to run local models, and the environment utilizes VLLM (docs.vllm.ai/en/latest/).

A.2.4 Software dependencies

Our experiments are conducted on Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-60-generic x86_64). The primary tools we used are Python and Jupyter Notebook. We use Conda to manage the environment.

A.2.5 Benchmarks

We collected data from the four domains, routinehub.co, shareshortcuts.com, shortcutsgallery.com, and matthewcassinelli.com/sirishortcuts. We provide only the YAML files, as the data collection time from websites was 2024; therefore, some recipes are no longer available.

A.3 Set-up

We include a requirements.txt that includes all the required packages to run the programs.

A.3.1 Installation

First, create a conda environment `conda create -n test python=3.11 -c conda-forge`, and activate the environment through `conda activate test`, install the environment by running `pip install -r requirements.txt`.

A.3.2 Basic Test

We include a script to check for environment: `check_env.sh`. The environment is successfully set up when this message is printed: `All 234 packages OK.`

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): *Our pipeline can automatically download an iOS Shortcut referenced by an iCloud link and save it as a local XML file (Section 5.2).*
(C2): *Translate XML representation to YAML text representation (Section 5.2).*
(C3): *Static code analysis to discard shortcuts that are not vulnerable to the four attacks (Section 5.2).*
(C4): *Use Qwen2.5-32B-Instruct model to produce both a rationale and a trinary (yes/no/maybe) prediction (Section 5.2).*

A.4.2 Experiments

Please also refer to README.md.

(E1): iCloud link → XML, 5 human-minutes
- **Preparation:** Obtain an iCloud Shortcut link (either from your own device or any public link) and paste in the `run(<link>)` in `get_xml_file.py`.
- **Execution:** `$ python get_xml_file.py`
- **Results:** A file named `<shortcut>.xml` appears in the working directory, confirming (C1).

(E2): XML → YAML, 5 human-minutes, machine-minutes are dependent on input size
- **Preparation:** Place one or more `*.xml` files produced by (E1) in `/test/` (The provided dataset does not contain XML files).
- **Execution:** `$ python test/test.py <input_file_or_directory> -o <output_path>`
- **Results:** This produces YAML under `/test/test_..._result_yaml`. Confirming (C2).

(E3): Static Code Analysis & Category Calculation, 5 human-minutes, machine-minutes are dependent on input size
- **Preparation:** Ensure the YAML files from (E2) are present in: `/test/test_[domain_name]_result_yaml`. (We provided these files as part of our results.)

Execution: 1. Run this script:
`$ python measurement/static_code_analysis.py`
(by default, the input folder is:
`/test/test_[domain_name]_result_yaml`)
2. Run this script:
`$ python measurement/calculate_result_of_category.py`
(by default, the input folder is:
`/test/test_result_code_analysis_[domain_name]`
Results: This is the code filter part we mentioned in the paper (Section 5.2), and it writes output files to
`/test/test_result_code_analysis_[domain_name]`
After calculating the result of each category, it will generate text files containing the filenames for each of the four attacks. By default, the final output folder for this step is
`/test/output_results_[domain_name]`
Confirms (C3): only shortcuts potentially vulnerable to the four attacks proceed to the LLM stage.

(E4): LLM Rationale + Trinary Verdict, 30 human-minutes + model-dependent compute-time (by default, we use Qwen2.5 32B-Instruct through huggingface and vllm, the processing time is more than 24 hours for all four datasets with 4 Nvidia A6000 GPUs)
Preparation: Pick an LLM (the default in our paper is **Qwen2.5-32B-Instruct**) and, if needed, adjust `max_tokens` in `/llm/distributor.py` to match your compute budget.
Execution: 1.Run this script:
`$ python llm/distributor.py`
This script creates one subfolder per attack under `/test/output_results_[domain_name]/action_categories`.
Example usage:

```
copy_matching_files("[attack]_to_evaluate_[domain]",
"test_[domain_name]_result_yaml",
"output_results_[domain]/action_categories/[attack].txt_safe.txt")
```

2.Run this script
`$ python test/move_safe_files.py` to copy only the files safe for a given attack/domain
3. Launch `llm/run_llm.ipynb` *or* `llm/run_gpt.py` (used for the impersonation re-run) to generate full rationales.
Please modify the file path in the code to match your local environment.
Additionally, you will need to have an OpenAI API to complete this step.
Our default input path is
`../test/[attack_category]to_evaluate[domain_name]`
, and default output folder is
`../measurement/result_[domain_name]_test_set`
. Note: since for impersonation, Qwen gives a high false positive rate, we add a second step that uses gpt-4o in /llm/run_gpt.py.

4. Run `llm/run_binary_answer.ipynb` to extract the final `yes/no/maybe` decisions. For files processed by GPT, they have a different output format, so pick the corresponding function from (extract_last_prompt_json_gpt, extract_last_prompt_json), extract_last_prompt_json is for Qwen.
Results: This step writes CSVs to `/llm/final_result_csvs` containing
(i) the LLM rationale and (ii) the trinary verdict for every processed shortcut, thereby confirming (C4).

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/.