Improving Logging to Reduce Permission Over-Granting Mistakes

Bingyu Shen, Tianyi Shan, and Yuanyuan Zhou, *University of California, San Diego*

https://www.usenix.org/conference/usenixsecurity23/presentation/shen-bingyu-logging

This artifact appendix is included in the Artifact Appendices to the Proceedings of the 32nd USENIX Security Symposium and appends to the paper of the same name that appears in the Proceedings of the 32nd USENIX Security Symposium.

August 9–11, 2023 • Anaheim, CA, USA

978-1-939133-37-3

Open access to the Artifact Appendices to the Proceedings of the 32nd USENIX Security Symposium is sponsored by USENIX.

Improving Logging to Reduce Permission Over-Granting Mistakes

Bingyu Shen, Tianyi Shan, Yuanyuan Zhou
UC San Diego

A Artifact Appendix

A.1 Abstract

This artifact includes the source code of our static analysis tool to improve the access-control logging, as well as the software binaries that we used to conduct evaluation. By executing the tool on the compiled software LLVM bitcode, it produces the logging locations and the list of variables that should be included in the logging.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

None.

A.2.2 How to access

Tthe stable URL evaluated in the artifact evaluation. https://github.com/byshen/seclog_ae/releases/tag/v1.0.

Please pull the latest version if there are any issues or updates. git@github.com:byshen/seclog_ae.git.

Rename the folder to `AceInstrument` after clone it. `git clone git@github.com:byshen/seclog_ae.git AceInstrument`

A.2.3 Hardware dependencies

- Please ensure enough memory for compiling LLVM. 16GB memory is enough during our evaluation.

A.2.4 Software dependencies

- Ubuntu 18.04;
- LLVM 9.0.0 for compiling the static analysis source code;
- wllvm for extracting the software binary's LLVM bitcode;
- Python `pandas` for processing the analysis output.

A.2.5 Benchmarks

We provide several testing programs in `dir_bcfiles` directory. For the ten server programs, we provide instructions to compile them in `compile-software.md`. We also provide the compiled bitcode files here. You can download and directly unzip it into `dir_bcfiles` directory.

A.3 Set-up

You should run the following the following command on a Linux platform. We used Ubuntu 18.04 in our experiment.

A.3.1 Installation

```
./build_llvm.sh
```

A.3.2 Basic Test

We provide several tests for quick testing.

1. Modify `BUILD_DIR` to `/home/USER/llvm-9.0.0.obj` and `APP_DIR` to `/home/USER/llvm-9.0.0.src/lib/Transforms/AceInstrument` in `scripts/opt_exec.sh`.
2. Simply run `./scripts/opt_exec.sh test_releatParam` , the output is in `output/test_releatParam.output`.
3. To get a csv format from out put, run the following. `cd output python3 output_parser.py -i test_releatParam`

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): *SecLog can suggest logging locations and logging variables for the software. This is proven by the experiment (E1).*

A.4.2 Experiments

(E1): [Logging Enhancement]:
How to: The experiment will simply execute the static analysis tool on the software's llvm bitcode to analyze the logging locations and logging variables.

Preparation: Using the provided bitcode files, or compile from source following the instructions in `compile-software.md` .
Execution: Under the cloned repository.
`./scripts/opt_exec.sh softwarename`
Results: The results are in `output/softwarename` Use the script to convert it to a csv format. `python3 output_parser.py -i softwarename`

A.5 Version

Based on the LaTeX template for Artifact Evaluation V20220926. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at https://secartifacts.github.io/usenixsec2023/.