USENIX Security '26 Artifact Appendix: <WAVED: Principled Identification of Off-Path Exploitable Weak Verifications within the TCP/IP Protocol Suite>

Yizhou Zhao[1], Xuewei Feng[1]✉, Min Li[2], and Ke Xu[1,2]✉

[1]Tsinghua University, Beijing, China
[2]Zhongguancun Laboratory, Beijing, China
{zhaoyz24@mails.,xuke@}tsinghua.edu.cn, fengxw06@126.com, limin@mail.zgclab.edu.cn

A Artifact Appendix

A.1 Abstract

This artifact provides the complete implementation of WAVED and the experimental environment required to reproduce the results presented in our paper. It specifically supports the reproduction of our main experiments, which identify weak-verification paths to risky operations across different kernel stacks, and the ablation studies that demonstrate the effectiveness of our reduction in path discovery. To ensure ease of deployment and consistent reproducibility, we provide a Dockerfile that automatically constructs the necessary build environment. The artifact also includes comprehensive step-by-step instructions, sample outputs, and detailed interpretations of the results to facilitate the evaluation process.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

Since WAVED is built upon static program analysis, all experiments listed below can be conducted on an isolated machine. The entire evaluation process is non-destructive.

A.2.2 How to access

We released the complete implementation of WAVED, with the user guidance and necessary scripts to reproduce all the results at https://doi.org/10.5281/zenodo.17896119.

A.2.3 Hardware dependencies

A server equipped with an x86-64 CPU and a minimum of 350 GB of memory is required to complete all experiments without memory exhaustion. To speed up building the dependencies, we recommend using a processor with 8 or more cores. We validated our artifact on a system featuring an Intel Xeon Gold 5418Y processor and 500 GB of RAM.

A.2.4 Software dependencies

A Linux-based operating system is required to run the artifact. We specifically recommend Ubuntu 22.04 or 24.04 for optimal compatibility. The other dependencies are listed in the *README.md* and *README-exp.md* in the repository.

A.2.5 Benchmarks

The artifact is evaluated on real-world operating system kernels acting as macro-benchmarks. Specifically, the static analysis is conducted on the source code of Linux 5.15, Linux 6.8, and FreeBSD 14.1, released at https://github.com/torvalds/linux.git and https://github.com/freebsd/freebsd-src.git. By applying our analysis to these complex systems, we demonstrate the artifact's practical capability in consolidating and strengthening the code quality of real-world protocol stacks. For convenience, we provide precompiled bitcode files of the IPv4/IPv6 implementations of these kernels in *bcfiles/* directory.

A.3 Set-up

A.3.1 Installation

We provide a building script (*build.sh*) to automatically build the artifact environment. For convenience, we also provide a Dockerfile to automatically construct the necessary environment and components for WAVED. This process can be executed by running the *build-docker.sh* script. After the Docker image is built, *run-docker.sh* will create a Docker container and open the shell.

A.3.2 Basic Test

Build the Docker image and start the container by following the instructions in *README-exp.md*. Next, execute *experiments/run-basic-test.sh* inside the container. If the output displays "`All components work fine. Go ahead`

`to run experiments!"`, the basic tests have passed successfully.

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): WAVED reports 19 weak-verification vulnerabilities in Linux 5.15, Linux 6.8 and FreeBSD 14.1 with considerably high precision. This is proven by the experiment (E1) described in Section 6.4, and more details are categorized in Section 7. The results are illustrated in Table 5, 6, and 7.

(C2): WAVED contributes a significant reduction (a total of 89.3% in average) comparing to traditional taint analysis in discovering weak-verification paths. This is proven by the experiment (E2) described in Section 6.4. The results are illustrated in Table 8.

A.4.2 Experiments

(E1): Main Experiment: [around 4 compute-hour]: The normal workflow of WAVED in weak-verification path discovery.
Preparation: Build the Docker image by executing the script *build-docker.sh*. Then run *run-docker.sh* to start a Docker container and access the shell.
Execution: Run *experiments/run-main-exp.sh* in the Docker container.
Results: Results can be found in *docker_results/* directory. *docker_results/main_results.txt* contains the discovered path numbers of each type. *docker_results/output-paths/* contains the details of each found path.

(E2): Ablation Experiments [around 19 compute-hour]: Ablation study on WAVED's reduction efficiency. Configurations include disabling strength filtering after constraint deduplication (w/o SF), and comparison with traditional implicit taint analysis (w/o DT).
Preparation: The same process as in E1.
Execution: Run *experiments/run-ablation-exp.sh* in the Docker container.
Results: Results can be found in *docker_results/* directory. *docker_results/ablation_results_woSF.txt* contains the discovered path numbers under the [w/o SF] configuration. *docker_results/ablation_results_woDT.txt* contains the discovered path numbers under the [w/o DT] configuration.

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/usenixsec2026/.