USENIX Security '24 Artifact Appendix: With Great Power Come Great Side Channels: Statistical Timing Side-Channel Analyses with Bounded Type-1 Errors

Martin Dunsche[1], Marcel Maehren[1], Nurullah Erinola[1], Robert Merget[2], Nicolai Bissantz[1], Juraj Somorovsky[3], and Jörg Schwenk[1]

[1]Ruhr University Bochum
[2]Technology Innovation Institute
[3]Paderborn University

A Artifact Appendix

A.1 Abstract

In our paper, we propose a test to statistically evaluate timing measurements with a type-1 error bounded by an input parameter α, which we implemented in a new tool called R-Time-Leak-Finder (RTLF). Initially, we compared RTLF with different competitors by evaluating them based on artificial timing side channels (ground truth). Subsequently, we performed a large-scale and longitudinal real-world timing evaluation of eleven TLS libraries in 823 versions across three vulnerabilities. Artifact users can reproduce the results of our analysis of the artificial side channels and our analysis of TLS libraries by running RTLF and the competitors. It is further possible to collect measurements for TLS libraries to reproduce our datasets.

A.2 Description & Requirements

Our artifact contains the timing measurements obtained in our study as well as the tools used to analyze the measurements. These tools consist of our proposed statistical approach (RTLF) and open-source frameworks previously used for timing analyses (*Mona*, *dudect*, and *tlsfuzzer*). We provide Dockerfiles that allow the user to run all tools on our measurements and confirm the results we present in the paper. These results are crucial as we use them to argue that RTLF generally performs better than previous tools with regard to timing side-channel detection. We split the measurements into three datasets: measurements of artificial side channels (*ground truth*), measurements obtained for our *quantitative analysis* of 823 TLS server implementations/versions, and measurements obtained for the most recent versions of eleven TLS libraries (*qualitative analysis*). We further provide the tools used to collect these measurements. The experiments described here cover specific parts for our study. While we outline how to reproduce the other results, doing so is expected to take weeks on multiple dedicated desktop computers.

A.2.1 Security, privacy, and ethical concerns

We are not aware of any exploitable issues in our artifacts.

A.2.2 How to access

Our artifact is split into a GitHub repository[1] and a Zenodo record[2] for our datasets. The repository includes Dockerfiles to execute RTLF and the competitors to reproduce our results and collect own measurements. The repository also includes multiple README.md files that provide additional details.

A.2.3 Hardware dependencies

Parts of the code used for experiment `E3` rely on assembly instructions, which may cause the Docker image build to fail on ARM CPUs (e.g. Apple Silicon).

A.2.4 Software dependencies

Running the experiments requires Git and Docker. Further, we tested this artifact on Kubuntu 22.04, but any Linux system should work.

A.2.5 Benchmarks

Running the experiments requires a minimum of around 55 GB of disk space. 40 GB of this disk space account for downloaded zip files containing our measurements and results.

[1] https://github.com/RUB-NDS/Artifacts-With-Great-Power-Come-Great-Side-Channels/tree/e1cb08804029775cc0f19a2ace2fd2d65d8a8eff

[2] https://zenodo.org/records/10817685

Unzipping all files to verify results beyond the scope of the experiments described in this artifact appendix will require additional 210 GB of disk space. The compute-hours stated in for experiments are approximate values for a CPU that can handle eight threads in parallel.

A.3 Set-up

A.3.1 Installation & Basic Test

1. Clone the GitHub repository using git:

```
git clone https://github.com/RUB-NDS/Artifacts-\
With-Great-Power-Come-Great-Side-Channels.git
git checkout e1cb08804029775cc0f19a2ace2fd2d65d8a8eff
```

2. Enter the cloned repository and run the shell script

```
sh setup.sh
```

This will build the Docker images of the artifacts, download our datasets (labeled `a` to `f`), extract the datasets required for the experiments, and perform basic tests. The Docker images will also contain the other statistical open-source tools used in our comparison. Our experiments `E1b`, `E2b`, and `E3` require a total of around nine hours of computation time on a computer capable of handling eight threads in parallel. You can run these experiments beforehand to perform the computational tasks using

```
sh experiments/run_lengthy_experiments.sh
```

To interpret the results, follow the description of the experiments (subsubsection A.4.2). The computational steps performed before should be skipped automatically.

A.4 Evaluation workflow

For the evaluation of our major claims, we mostly rely on the measurements and analysis results we collected and provide. Reproducing our entire evaluation requires extensive computational effort. We hence limit the experiments to specific results but outline how to adapt the experiments to confirm our other results. Additionally, we describe how a user can reproduce our measurements for the considered TLS libraries.

A.4.1 Major Claims

(C1): Based on the analysis of artificial side channels (ground truth), we claim that RTLF remains within the configured type-1 error threshold. Furthermore, we claim that *dudect* shows a constant type-1 error rate of 0% but also fails to achieve (high) statistical power for our measurements. In the case of *Mona*, the type-1 error rate decreases steadily with increasing sample size, and the t-test only manages to remain within its type-1 error rate for large sample sizes. We present our results in `E1a`. The steps to reproduce them are outlined in `E1b`.

(C2): Through the qualitative analysis based on the results of RTLF, we identified seven vulnerabilities in recent versions. Notably, some of these identified leaks go unnoticed with *Mona*, *dudect*, the t-test, and *tlsfuzzer*. We present our results in `E2a`. The steps to reproduce them are outlined in `E2b`.

A.4.2 Experiments

The following experiments assume that the user starts in the directory of the cloned repository.

(E1a): (10 human-minutes) This experiment focuses on the results of our tool comparison for artificial side channels (ground truth). The comparison evaluates how often each tool successfully detected a timing difference (*statistical power*) and how often each tool wrongfully reports a non existing timing difference (*type-1 error* rate or *false positives* rate).

Execution: Run the shell script

```
sh experiments/experiment1a.sh
```

This will run a Docker container that collects the results of the individual tools and prints them to console. The container will first print the results of Table 2 from our paper. After pressing enter, the container will print the results for Table 3.

Results: In the first batch of results, a detected difference is a correct test result. In the second batch, a detected difference is an incorrect test result (type-1 error). The output generally indicates the claimed weaknesses of *dudect*, *Mona*, and the *t-test*: *dudect* shows low statistical power, especially for small sample sizes (see results for `measurements-30k-tail`, for example). In the type-1 error analysis (the second batch of outputs), *Mona* and the t-test both show a high type-1 error that only declines with an increasing sample size. RTLF remains within its configured type-1 error threshold of 9% (except for statistically expected fluctuations).

(E1b): (5 human-minutes + 4 compute-hours) This experiment shows how to reproduce our results from `E1a` for a limited dataset. Specifically, we chose the *same-xy* artificial side channel with sample size *15k* from Table 3, as it is the smallest self-contained dataset.

Execution: Run the shell script

```
sh experiments/experiment1b.sh
```

This will run a Docker container for each of the statistical tools to analyze the measurements of the `measurements-15k-same-xy` directory. For each tool, we store the result in a file with file extension `.result-[tool-name]`. The final Docker run command starts the container that reads these results and

prints them to console.

Results: The results should match those from `E1a` for the `measurements-15k-same-xy` directory. There may be slight deviations for RTLF due to the non-deterministic empirical bootstrap we employ.

Reproducing Additional Results: To reproduce additional analysis results, unzip any directory of `a_measurements_ground_truth.zip` to the `data_sets` directory and adjust the `TARGET_DIRECTORY` variable in the shell script.

(E2a): (5 human-minutes) This experiment focuses on the analysis results for our measurements of the MatrixSSL TLS library version 4.6.0. Specifically, we present the results of the statistical analysis of the different tools for the *Bleichenbacher* attack.

Execution: Run the shell script

```
sh experiments/experiment2a.sh
```

This will run a Docker container that collects the results of the individual tools and prints measurement pairs with timing differences to console.

Results: The output shows the results illustrated for MatrixSSL in Figure 5, 6, and 7 of our paper. Specifically, only RTLF identified a timing leak based on the collected *Bleichenbacher* measurements (`BxBy` measurement pairs, see Section 5.1 for the vector labels). In Algorithm 3 of our paper, we discuss why timing leaks are indeed expected here to confirm RTLF's result. Note that *tlsfuzzer* employs a test that is not based on pairwise comparison (Friedman test) and does not specify which vectors exhibit timing variations. Therefore, we print all vectors categorized as either with or without differences to represent the boolean analysis results.

(E2b): (5 human-minutes + 2 compute-hours) This experiment reproduces our results from `E2a` for the *Bleichenbacher* measurements.

Execution: Run the shell script

```
sh experiments/experiment2b.sh
```

Analogue to `E1b`, this will perform the analysis of the measurements for each of the statistical tools and print the results to console.

Results: The results should match those from `E2a`, i.e only RTLF should indicate any timing differences. Again, there may be slight deviations for RTLF due to the non-deterministic empirical bootstrap we employ.

Reproducing Additional Results: To reproduce additional analysis results, unzip any directory of any of the measurements zip files (`b` to `d`) to the `data_sets` directory and adjust the `TARGET_DIRECTORY` variable in the shell script accordingly.

(E3): (10 human-minutes + 3 compute-hour (2 hours to collect measurements, 1 hour to analyze)) In this experiment, we describe how to reproduce the measurements of the TLS Docker images using our measurement tool (TLS-Docker-Timer). We use one TLS implementation (NSS v3.87) and one attack type (Bleichenbacher) from our qualitative analysis as an example. Running this experiment requires the user to pass their Docker socket to the Docker container that collects the measurements as the container will build and manage the image of the tested TLS implementation.[3]

Execution: Run the shell script

```
sh experiments/experiment3.sh
```

The script runs a Docker container that first checks the Docker environment for the required TLS library image. Then, it launches a proxy process for measurements alongside our TLS-Docker-Timer tool, which creates the Docker container for NSS and establishes the TLS sessions for the attack. Finally, the script will again analyze the measurements with all considered statistical tools before printing the results to console.

Results: The measurements should be stored in the newly created `own_measurements` directory of the repository directory. The results printed to console should generally align with those of Figure 4, 6, and 7 from our paper. Specifically, the tools should detect a timing leak for all NSS *Bleichenbacher* measurement pairs except `B1B3`, `B2B4`, `B2B5`, and `B4B5`. Note that *tlsfuzzer* will again only yield one combined test result (see `E2a`). Significant deviations are possible for measurements due to user system dependencies, leading to false negatives and false positives. Additionally, the Docker container will conduct 100,000 measurements for each attack vector (as opposed to the 500,000 measurements from our study).

Reproducing Additional Results: By adjusting the variables of the shell script, the user can test for timing leaks with different attacks and implementations. For further details, we refer to the measuring/README.md in our artifact repository.

A.5 Notes on Reusability

Further development of RTLF will continue at https://github.com/tls-attacker/RTLF. TLS-Docker-Timer is available at https://github.com/tls-attacker/TLS-Docker-Timer.

A.6 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/.

[3] Note that these images will be added to the Docker instance of the host system.