A comprehensive, formal and automated analysis of the EDHOC protocol

Charlie Jacomme, *Inria Paris;* Elise Klein, Steve Kremer, and Maïwenn Racouchot, *Inria Nancy and Université de Lorraine*

https://www.usenix.org/conference/usenixsecurity23/presentation/jacomme

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.

USENIX'23 Artifact Appendix: A comprehensive, formal and automated analysis of the EDHOC protocol

Charlie Jacomme
Inria Paris*

Elise Klein
Inria Nancy
Université de Lorraine

Steve Kremer
Inria Nancy
Université de Lorraine

Maïwenn Racouchot
Inria Nancy
Université de Lorraine

A Artifact Appendix

A.1 Abstract

This artifact permits to reproduce the formal verification of the LAKE EDHOC protocol. It comes as a docker image containing

- the software needed (SAPIC$^+$, TAMARIN, PROVERIF, DEEPSEC);
- the models of the LAKE EDHOC protocol;
- the scripts to batch run the verification of the models using the SAPIC$^+$ platform.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

None.

A.2.2 How to access

The artifact is hosted on docker hub: https://hub.docker.com/r/protocolanalysis/lake-edhoc/.

A.2.3 Hardware dependencies

There are two main sets of experiments, one relying on the PROVERIF tool and one on the TAMARIN tool. The PROVERIF experiments can be executed in a couple of hours on a modern laptop (8 threads at 2.8Ghz), while the Tamarin one while take multiple days. Having access to a server with 48 threads ensures that everything will run under a day.

A.2.4 Software dependencies

Docker is the only required dependency.

A.2.5 Benchmarks

[Mandatory] None.

*This work was partly done while Charlie Jacomme was at the CISPA Helmholtz Center for Information Security.

A.3 Set-up

[Mandatory] Installation instruction for Docker are provided at https://docs.docker.com/engine/install/.

A.3.1 Installation

The artifact is fetched with

```
docker pull protocolanalysis/lake-edhoc:draft-14
```

A.3.2 Basic Test

A bash should be opened inside the docker when running:

```
docker run -it protocolanalysis/lake-edhoc:draft-14 bash
```

A.4 Evaluation workflow

Once inside the Docker, there are two subfolders `lake-draft12` and `lake-draft14`. Each folder contains a README, bash scripts to run the exhaustive verification as well as the tool chain to generate the models from jinja2 templates. To ease the artifact evaluation, we automated all the relevant verifications under two main scripts:

- `./run-proverif.sh`
- `./run-tamarin.sh`

A.4.1 Major Claims

The LAKE EDHOC protocol draft 12 and draft 14 can be analyzed automatically using the SAPIC$^+$ platform under different scenarios. This yields the analysis results provided in Table 7 and 8 of Appendix B of our paper. Those tables where manually produced by formatting in LaTeX the results stored in the two csv files located in the `expected-results` folder.

A.4.2 Experiments

(E1): PROVERIF results: 5 human-minutes + 5.8 single-threaded (2.8Ghz) compute hours. With 7 threads at 2.8Ghz and 31 GB of RAM the verification runs in 62 minutes, and can go down to 31 minutes with additional cores.
Preparation: Enter the docker image. Check the number of available threads, e.g. with `htop`.
Execution: Run `./run-proverif.sh i`, where `i` is the number of available threads, and wait for full completion. The script displays which verification are started (mainly for debugging purposes).
Results: The script produces a `res-proverif.csv` file. To compare the obtained results with the one stored in `expected-results`, the `compare-diff-proverif.sh` script highlights any differences. The diff should only highlight timing differences, or additional timeouts in case of different hardware.

(E2): TAMARIN results: 5 human-minutes + 64 single-threaded (2.8Ghz) compute hours. With parallelization over 48 threads at 2.8Ghz and 756GB, the experiments takes 14 hours.
Preparation: Enter the docker image. Check the number of available threads, e.g. with `htop`.
Execution: Run `./run-tamarin.sh i`, where `i` is the number of threads **divided by 4** (as we allocate 4 threads to each instance of TAMARIN), and wait for full completion. The script displays which verification are started (mainly for debugging purposes).
Results: The script produces a `res-tamarin.csv` file. To compare the obtained results with the one stored in `expected-results`, the `compare-diff-tamarin.sh` script highlights any differences. The diff should only highlight timing differences, or additional timeouts in case of different hardware.

In addition, the privacy analysis of Table 6 can also be verified, running `./lake-draft12/run-anonimity.sh` and `./lake-draft14/run-anonimity.sh`.

A.5 Notes on Reusability

The docker also contains a `README.md` meant for users familiar with the underlying TAMARIN/PROVERIF/SAPIC$^+$ tool chain, which explains how to either reuse our general structure or update the models.

To build the docker, the following actions can be performed:

```
git clone https://github.com/charlie-j/tamarin-prover/
cd tamarin-prover;
git checkout e59304fb8f51e1e25118362daeb3fc008a6e292d;
./etc/docker/build.sh
./etc/docker/build-platform.sh
cd ../
git clone https://github.com/charlie-j/edhoc-formal-analysis
git checkout e2e3f7407e9eb331a8112614fe9e116e57a25e51
cd edhoc-formal-analysis
./Docker/build.sh
```

A.6 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/.