USENIX'23 Artifact Appendix: Automated Security Analysis of Exposure Notification Systems

Kevin Morio[1] Ilkan Esiyok[1] Dennis Jackson[2] Robert Künnemann[1]

[1]CISPA Helmholtz Center for Information Security
[2]Mozilla

A Artifact Appendix

A.1 Abstract

The artifact consists of the Tamarin model files with the corresponding oracles for the three exposure notification systems ROBERT, DP3T, and the CWA presented in the paper. The artifact is available as a ZIP archive as well as packaged in a Docker container featuring a recent version of Tamarin.

Our analysis results presented in Section 6 of the paper can be revalidated by verifying the lemmas of each model with Tamarin.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

None.

A.2.2 How to access

The artifact can be directly downloaded from https://doi.org/10.6084/m9.figshare.21304305.

The provided Docker container with the artifact and Tamarin can be obtained by executing

```
docker pull kevinmorio/usenix23-ens
```

A.2.3 Hardware dependencies

The evaluation of the artifact requires about 43.15 GB of memory (peak) and ideally 12 cores.

A.2.4 Software dependencies

Tamarin with its dependencies Maude and Graphviz can be directly installed on the system. Alternatively, for using the provided Docker container, a working installation of Docker is required.

A.2.5 Benchmarks

None.

A.3 Set-up

A.3.1 Installation

Tamarin and its dependencies can be installed following the instruction from the Tamarin manual: https://tamarin-prover.github.io/manual/book/002_installation.html. The artifact can be downloaded from https://doi.org/10.6084/m9.figshare.21304305.

Installation instruction for Docker are available in the Docker documentation: https://docs.docker.com/engine/install. The Docker container can be obtained by executing

```
docker pull kevinmorio/usenix23-ens
```

A.3.2 Basic Test

To check that Tamarin works correctly, execute

```
tamarin-prover test
```

for a local installation of Tamarin or

```
docker run kevinmorio/usenix23-ens \
  tamarin-prover test
```

when using the Docker container.

The output should be

```
Self-testing the tamarin-prover installation.

*** Testing the availability of the required tools ***
maude tool: 'maude'
 checking version: 2.7.1. OK.
 checking installation: OK.

GraphViz tool: 'dot'
 checking version: dot - graphviz version 7.0.1 (20221109.1506). OK.
 checking PNG support: OK.

*** Testing the unification infrastructure ***
Cases: 55  Tried: 55  Errors: 0  Failures: 0

*** TEST SUMMARY ***
All tests successful.
The tamarin-prover should work as intended.

          :-) happy proving (-:
```

where the reported versions for Maude and Graphviz can differ depending on the system.

A.4 Evaluation workflow

A.4.1 Major Claims

(Claim 1): A1–A4 categorise all attacks against upload authorisation (Def. 3) for ROBERT as described in Section 6.2. X1–X7 categorise all attacks against soundness (Def. 1) for ROBERT as described in Section 6.3. This is verified by experiment (E1).

(Claim 2): B1–B3 categorise all attacks against upload authorisation (Def. 2) for DP3T as described in Section 6.2. Y1–Y7 categorise all attacks against soundness (Def. 1) for DP3T as described in Section 6.3. This is verified by experiment (E2).

(Claim 3): C1–C2 categorise all attacks against upload authorisation (Def. 2) for the CWA as described in Section 6.2. Z1–Z4 categorise all attacks against soundness (Def. 1) for the CWA as described in Section 6.3. This is verified by experiment (E3).

A.4.2 Experiments

(E1): *[Verification of ROBERT] [10 human-minutes + 14 compute-hours]: Verify the lemmas in* `robert.spthy` *with Tamarin. All lemmas should verify.*
Preparation: Install Tamarin directly and download + extract the artifact or obtain the Docker container as described above.
Execution: For the local Tamarin install, enter the directory where the artifact has been extracted to and execute

```
tamarin-prover --prove robert.spthy
```

Alternatively, for Docker execute

```
docker run kevinmorio/usenix23-ens \
  tamarin-prover --prove robert.spthy
```

The number of cores and the amount of memory used by Tamarin can be configured by adding `+RTS -N<num-cores> -M<gb-mem>g -RTS` directly after `tamarin-prover` in the commands above. The reported compute-hours have been obtained with `-N12` and no memory limit on an Intel(R) Xeon(R) CPU E5-4650L workstation.
Results: When Tamarin terminates, it reports a summary of summaries listing all lemmas and their verification result line-by-line, e.g.,

```
soundness (all-traces): verified (.. steps)
```

The verification result of each lemma should be `verified`. Moreover, the summary of summaries reported should be the same as the summary of summaries at the end of `robert.spthy`

(E2): *[Verification of DP3T] [10 human-minutes + 2 compute-hours]: Verify the lemmas in* `dp3t.spthy` *with Tamarin. All lemmas should verify.*
This experiment is the same as the one above for ROBERT except for using `dp3t.spthy` instead of `robert.spthy`.

(E3): *[Verification of the CWA] [10 human-minutes + 1 compute-hour]: Verify the lemmas in* `cwa.spthy` *with Tamarin. All lemmas should verify.*
This experiment is the same as the one above for ROBERT except for using `cwa.spthy` instead of `robert.spthy`.

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/.