Practical Type-Based Taint Checking and Inference (Artifact)

Nima Karimipour ✉
University of California, Riverside, CA, USA

Kanak Das ✉
University of California, Riverside, CA, USA

Manu Sridharan ✉
University of California, Riverside, CA, USA

Behnaz Hassanshahi ✉
Oracle Labs, Brisbane, Australia

Abstract

We present a containerized framework for the paper Practical Type-Based Taint Checking and Inference. Packed as a Docker image, the artifact bundles our novel inference engine alongside CodeQL[1] and P/Taint[2] analyses, together with precomputed results and scripts to reproduce five core experimental tables: benchmark characteristics, soundness on labeled issues, precision/recall on real-world projects, runtime comparisons, and annotation ablation studies. By unifying checking and inference in a portable setup, this artifact enables straightforward validation of our paper's claims.

2012 ACM Subject Classification Software and its engineering → Software verification and validation; Security and privacy → Software security engineering

Keywords and phrases Static analysis, Taint Analysis, Pluggable type systems, Security, Inference

Digital Object Identifier 10.4230/DARTS.11.2.7

Funding This research was supported in part by the National Science Foundation under grants CCF-2007024, CCF-2223826, and CCF-2312263, and a gift from Oracle Labs.

Related Article Nima Karimipour, Kanak Das, Manu Sridharan, and Behnaz Hassanshahi, "Practical Type-Based Taint Checking and Inference", in 39th European Conference on Object-Oriented Programming (ECOOP 2025), LIPIcs, Vol. 333, pp. 18:1–18:25, 2025.
`https://doi.org/10.4230/LIPIcs.ECOOP.2025.18`

Related Conference 39th European Conference on Object-Oriented Programming (ECOOP 2025), June 30–July 2, 2025, Bergen, Norway

Evaluation Policy The artifact has been evaluated as described in the ECOOP 2025 Call for Artifacts and the ACM Artifact Review and Badging Policy.

1 Scope

The artifact covers all components required to reproduce Tables 1–5, including both automated and human-involved steps (manual triage for Table 2 and Table 3). Regeneration times range from a few minutes (Tables 1 and 5) up to several hours or days for full fresh runs.

2 Content

This artifact comprises the following distinct components:

The accepted paper ▪ `paper.pdf`
- Format: PDF

Container image (code + environment) ▪ `taint.tar`
- Format: tarball

DAGSTUHL ARTIFACTS SERIES
Dagstuhl Artifacts Series
Schloss Dagstuhl – Leibniz-Zentrum für Informatik,
Dagstuhl Publishing, Germany

Table 1: Benchmark sizes and annotation density (code, script & data) `/opt/table1`
Table 2: Soundness on labeled benchmarks (code, script & data) `/opt/table2`
Table 3: Precision and recall on real-world projects (code, script & data) `/opt/table3`
Table 4: Runtime comparison (code, script & data) `/opt/table4`
Table 5: Annotation ablation study (code, script & data) `/opt/table5`

3 Getting the artifact

The artifact endorsed by the Artifact Evaluation Committee is available free of charge on the Dagstuhl Research Online Publication Server (DROPS). In addition, the artifact is also available at Zenodo under the name **Artifact Evaluation for Practical Type-Based Taint Checking and Inference**[3].

4 Tested platforms

The artifact has been produced and tested in Ubuntu 22.04 LTS in a 3th Gen Intel(R) Core(TM) i7-13700 processor (16 cores, 24 threads, up to 5.2 GHz), 64GB RAM machine. We recommend having at least 100GB of available disk space to run the experiments.

5 License

The artifact is available under MIT license.

6 MD5 sum of the artifact

92459d199d7dd46a54935ac6013dceb6

7 Size of the artifact

25 GiB

A Appendix

A README.md description is provided in the Zenodo artifact URL that can be followed to reproduce the results.

References

1 CodeQL. `https://codeql.github.com`, 2024. Accessed: 2024-02-07.

2 Doop – Framework for Java Pointer and Taint Analysis (using P/Taint). `https://github.com/plast-lab/doop`, 2024. Accessed: 2024-07-29.

3 Zenodo. `https://zenodo.org/records/15301001`, 2024. Accessed: 2024-02-07.