Staccato: A Bug Finder for Dynamic Configuration Updates (Artifact)

John Toman[*1] **and Dan Grossman**[2]

1 **Department of Computer Science & Engineering**
University of Washington
Seattle, WA, USA
jtoman@cs.washington.edu

2 **Department of Computer Science & Engineering**
University of Washington
Seattle, WA, USA
djg@cs.washington.edu

Abstract

This artifact is based on Staccato, a tool for finding errors in dynamic configuration update (DCU) implementations. Dynamic configuration update refers to configuration changes that occur at runtime without program restart. Errors in DCU implementations occur when stale data—computed from old configurations—or inconsistent data—computed from different configurations—are used. Staccato uses a dynamic analysis in the style of taint analysis to detect these errors. Staccato supports concurrent programs running on commodity JVMs. We evaluated Staccato on three open-source applications and found errors in all of them.

1998 ACM Subject Classification D.2.9; Software configuration management

Keywords and phrases Dynamic Configuration Updates, Dynamic Analysis, Software configuration

Digital Object Identifier 10.4230/DARTS.2.1.14

Related Article John Toman and Dan Grossman, "Staccato: A Bug Finder for Dynamic Configuration Updates", in Proceedings of the 30th European Conference on Object-Oriented Programming (ECOOP 2016), LIPIcs, Vol. 56, pp. 24:1–24:25, 2016.
http://dx.doi.org/10.4230/LIPIcs.ECOOP.2016.24

Related Conference 30th European Conference on Object-Oriented Programming (ECOOP 2016), July 18–22, 2016, Rome, Italy

1 Scope

This artifact is intended to help reproduce the experiments run for the companion paper. It is expected that all major claims in the paper may be validated with this artifact.

In particular, there are five major claims in the paper that this artifact supports:

1. That software suffers from DCU bugs and Staccato can find them (RQ 1 and 2)
2. The annotation overhead for finding these bugs is low (RQ 3)
3. The performance overhead of using Staccato is reasonable (RQ 4)
4. Staccato achieves good coverage (section 6.2, "Checking Coverage")
5. The program repair mechanism can be used to effectively introduce support for several previously immutable options in Openfire and JForum (sections 6.3.1 and 6.3.2)

Our informal evaluation on Solr can also be validated with this artifact.

[*] Core artifact developer.

2 Content

The artifact includes:

- The source of STACCATO, a dynamic analysis for finding errors in dynamic configuration update implementations
- A patched version of Phosphor used in conjunction with STACCATO for the purposes of information flow tracking
- Annotated and unannotated source archives for the three main evaluation programs: Openfire, JForum, and Subsonic
- Annotated and unannotated source archives for Solr, upon which we conducted an informal evaluation
- The test cases and test scripts used in the paper experiments
- The classification of each evaluation program's options and bugs
- Extensive documentation for running our experiments and using STACCATO on programs not included in our evaluation

To simplify reproducing our experiments, we have included an Open Virtualization Archive image pre-configured to run STACCATO and our experiments. The virtual machine image includes a minimal installation of Ubuntu 14.04 LTS. Detailed documentation on the contents of the artifact and using the programs found in the virtual machine may be found in the artifact's documentation.

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). The latest version of our code is available on the project GitHub page: `https://github.com/uwplse/staccato`.

4 Tested platforms

The artifact is known to work on any platform running Oracle VirtualBox version 4.3 (`https://www.virtualbox.org/`) with at least 10 GB of free space on disk and at least 2 GB of free space in RAM.

5 License

MIT (`https://opensource.org/licenses/MIT`)

6 MD5 sum of the artifact

f025fec4c575aae8ffe6355ea0100955

7 Size of the artifact

6.7 GB

Acknowledgements. The authors wish to thank James Bornholt for his advice and feedback during the artifact preparation process.