A Stateful Fuzzer for the TCP/IP Stack of the Real-Time Operating System Zephyr

—

Report

Valentin Huber

Institute of Applied Information Technology

Zürich University of Applied Sciences ZHAW

contact@valentinhuber.me

November 28, 2024

Contents

1 Introduction

2 Related Works

- *Stateful Greybox Fuzzing* [1]:

  > In this paper, we argue that protocols are often explicitly encoded using state variables that are assigned and compared to named constants [...] More specifically, using pattern matching, we identify state variables using enumerated types (enums). An enumerated type is a group of named constants that specifies all possible values for a variable of that type. Our instrumentation injects a call to our runtime at every program location where a state variable is assigned to a new value. Our runtime efficiently constructs the state transition tree (STT). The STT captures the sequence of values assigned to state variables across all fuzzer-generated input sequences, and as a global data structure, it is shared with the fuzzer.

  Built on LibFuzzer

- *StateAFL: Greybox fuzzing for stateful network servers* [2]: compile-time probes observing memory allocation and I/O opersations; state inference based on fuzzy hashing of long-lived memory areas.

- *Ijon: Exploring Deep State Spaces via Fuzzing* [3]: Manual annotations of code, to manually add entries to an AFL-style map (set/inc at calculated offset), include state information (variable values) in how edge coverage is calculated, and store the max value a certain variable reaches during execution for the fuzzer to then maximize.

- *SandPuppy: Deep-State Fuzzing Guided by Automatic Detection of State-Representative Variables* [4]: Ijon [3], but automatic (initial run capturing variable-value traces, analyze along with source code, add Ijon-style instrumentation, repeat during fuzzing)

Bibliography

[1] J. Ba, M. Böhme, Z. Mirzamomen, and A. Roychoudhury, "Stateful greybox fuzzing," in *31st USENIX Security Symposium (USENIX Security 22)*, Boston, MA: USENIX Association, Aug. 2022, pp. 3255–3272, ISBN: 978-1-939133-31-1. [Online]. Available: https://www.usenix.org/conference/usenixsecurity22/presentation/ba.

[2] R. Natella, "Stateafl: Greybox fuzzing for stateful network servers," *Empirical Software Engineering*, vol. 27, no. 7, p. 191, Oct. 2022, ISSN: 1573-7616. DOI: 10.1007/s10664-022-10233-3. [Online]. Available: https://doi.org/10.1007/s10664-022-10233-3.

[3] C. Aschermann, S. Schumilo, A. Abbasi, and T. Holz, "Ijon: Exploring deep state spaces via fuzzing," in *2020 IEEE Symposium on Security and Privacy (SP)*, 2020, pp. 1597–1612. DOI: 10.1109/SP40000.2020.00117.

[4] V. Paliath, E. Trickel, T. Bao, R. Wang, A. Doupé, and Y. Shoshitaishvili, "Sandpuppy: Deep-state fuzzing guided by automatic detection of state-representative variables," in *Detection of Intrusions and Malware, and Vulnerability Assessment*, F. Maggi, M. Egele, M. Payer, and M. Carminati, Eds., Cham: Springer Nature Switzerland, 2024, pp. 227–250, ISBN: 978-3-031-64171-8.