REFERENCES

[1] Humberto Abdelnur, Obes Jorge Lucangeli, and Olivier Festor. 2010. *Spectral Fuzzing: Evaluation & Feedback.* Ph.D. Dissertation. INRIA.
[2] aflgo. 2017. AFLGO: Directed Greybox Fuzzing. https://github.com/aflgo/aflgo.
[3] Cornelius Aschermann, Sergej Schumilo, Ali Abbasi, and Thorsten Holz. 2020. Ijon: Exploring deep state spaces via fuzzing. In *IEEE S&P*. 1597–1612.
[4] Marcel Böhme, Van-Thuan Pham, Manh-Dung Nguyen, and Abhik Roychoudhury. 2017. Directed greybox fuzzing. In *ACM CCS*. 2329–2344.
[5] Marcel Böhme, Van-Thuan Pham, and Abhik Roychoudhury. 2017. Coverage-based greybox fuzzing as markov chain. *IEEE TSE* 45, 5 (2017), 489–506.
[6] Lutz Bornmann, Loet Leydesdorff, and Rüdiger Mutz. 2013. The use of percentiles and percentile rank classes in the analysis of bibliometric data: Opportunities and limits. *Journal of informetrics* 7, 1 (2013), 158–165.
[7] Hongxu Chen, Yinxing Xue, Yuekang Li, Bihuan Chen, Xiaofei Xie, Xiuheng Wu, and Yang Liu. 2018. Hawkeye: Towards a desired directed grey-box fuzzer. In *ACM CCS*. 2095–2108.
[8] Peng Chen and Hao Chen. 2018. Angora: Efficient fuzzing by principled search. In *IEEE S&P*. 711–725.
[9] Yaohui Chen, Peng Li, Jun Xu, Shengjian Guo, Rundong Zhou, Yulong Zhang, Tao Wei, and Long Lu. 2020. Savior: Towards bug-driven hybrid testing. In *IEEE S&P*. 1580–1596.
[10] Nicolas Coppik, Oliver Schwahn, and Neeraj Suri. 2019. Memfuzz: Using memory accesses to guide fuzzing. In *IEEE ICST*. 48–58.
[11] Leila Delshadtehrani, Sadullah Canakci, Boyou Zhou, Schuyler Eldridge, Ajay Joshi, and Manuel Egele. 2020. Phmon: a programmable hardware monitor and its security use cases. In *USENIX Security*. 807–824.
[12] Firefox. 2021. Fuzzing. https://firefox-source-docs.mozilla.org/tools/fuzzing.
[13] Shuitao Gan, Chao Zhang, Xiaojun Qin, Xuwen Tu, Kang Li, Zhongyu Pei, and Zuoning Chen. 2018. Collafl: Path sensitive fuzzing. In *IEEE S&P*. 679–696.
[14] glennrp. 2018. https://github.com/glennrp/libpng/tree/libpng16/contrib/testpngs.
[15] Google. 2016. OSS-Fuzz. https://github.com/google/oss-fuzz/.
[16] Google. 2020. AFL dictionaries. https://github.com/google/AFL/tree/master/dictionaries.
[17] Google. 2020. AFL test cases. https://github.com/google/AFL/tree/master/testcases.
[18] Google. 2020. American Fuzzy Lop. https://github.com/google/AFL.
[19] Google. 2021. ClusterFuzz. https://google.github.io/clusterfuzz/setting-up-fuzzing/libfuzzer-and-afl/#afl-limitations.
[20] Google. 2021. Continuous Integration. https://google.github.io/oss-fuzz/getting-started/continuous-integration/.
[21] Google. 2021. Honggfuzz. https://github.com/google/honggfuzz.
[22] Gustavo Grieco, Martín Ceresa, and Pablo Buiras. 2016. QuickFuzz: An automatic random fuzzer for commonfi le formats. *SIGPLAN Notices* 51, 12 (2016), 13–20.
[23] Ahmad Hazimeh, Adrian Herrera, and Mathias Payer. 2020. Magma. hexhive. epfl.ch/magma/docs/bugs.html.
[24] Ahmad Hazimeh, Adrian Herrera, and Mathias Payer. 2020. Magma: A Ground-Truth Fuzzing Benchmark. *ACM POMACS* 4, 3 (2020), 1–29.
[25] Adrian Herrera, Hendra Gunadi, Shane Magrath, Michael Norrish, Mathias Payer, and Antony L Hosking. 2021. Seed selection for successful fuzzing. In *ACM SIGSOFT ISSTA*. 230–243.
[26] Christian Holler, Kim Herzig, and Andreas Zeller. 2012. Fuzzing with code fragments. In *USENIX Security*. 445–458.
[27] George Klees, Andrew Ruef, Benji Cooper, Shiyi Wei, and Michael Hicks. 2018. Evaluating fuzz testing. In *ACM SIGSAC CCS*. 2123–2138.
[28] Gwangmu Lee, Woochul Shim, and Byoungyoung Lee. 2021. Constraint-guided Directed Greybox Fuzzing. In *USENIX Security*.
[29] Yuwei Li, Shouling Ji, Yuan Chen, Sizhuang Liang, Wei-Han Lee, Yueyao Chen, Chenyang Lyu, Chunming Wu, Raheem Beyah, and Peng Cheng. 2021. Unifuzz: A holistic and pragmatic metrics-driven platform for evaluating fuzzers. In *USENIX Security*.
[30] Hongliang Liang, Yini Zhang, Yue Yu, Zhuosi Xie, and Lin Jiang. 2019. Sequence coverage directed greybox fuzzing. In *IEEE/ACM ICPC*. 249–259.
[31] LLVM. 2021. libFuzzer. https://llvm.org/docs/LibFuzzer.html#corpus.
[32] Chenyang Lyu, Shouling Ji, Chao Zhang, Yuwei Li, Wei-Han Lee, Yu Song, and Raheem Beyah. 2019. MOPT: Optimized mutation scheduling for fuzzers. In *USENIX Security*. 1949–1966.
[33] Valentin Jean Marie Manès, HyungSeok Han, Choongwoo Han, Sang Kil Cha, Manuel Egele, Edward J Schwartz, and Maverick Woo. 2019. The art, science, and engineering of fuzzing: A survey. *IEEE TSE* (2019).
[34] K Paul Nesselroade Jr and Laurence G Grimm. 2018. *Statistical applications for the behavioral and social sciences.* John Wiley & Sons.
[35] Manh-Dung Nguyen, Sébastien Bardin, Richard Bonichon, Roland Groz, and Matthieu Lemerre. 2020. Binary-level directed fuzzing for use-after-free vulnerabilities. In *RAID*. 47–62.
[36] Sebastian Österlund, Kaveh Razavi, Herbert Bos, and Cristiano Giuffrida. 2020. Parmesan: Sanitizer-guided greybox fuzzing. In *USENIX Security*. 2289–2306.
[37] Shankara Pailoor, Andrew Aday, and Suman Jana. 2018. Moonshine: Optimizing OS fuzzer seed selection with trace distillation. In *USENIX Security*. 729–743.
[38] Jiaqi Peng, Feng Li, Bingchang Liu, Lili Xu, Binghong Liu, Kai Chen, and Wei Huo. 2019. 1dvul: Discovering 1-day vulnerabilities through binary patches. In *IEEE/IFIP DSN*. 605–616.
[39] Alexandre Rebert, Sang Kil Cha, Thanassis Avgerinos, Jonathan Foote, David Warren, Gustavo Grieco, and David Brumley. 2014. Optimizing seed selection for fuzzing. In *USENIX Security*. 861–875.
[40] sqlite. 2021. SQLite Source Repository. https://github.com/sqlite/sqlite/tree/master/test.
[41] Spandan Veggalam, Sanjay Rawat, Istvan Haller, and Herbert Bos. 2016. Ifuzzer: An evolutionary interpreter fuzzer using genetic programming. In *ESORICS*. 581–601.
[42] Haijun Wang, Xiaofei Xie, Yi Li, Cheng Wen, Yuekang Li, Yang Liu, Shengchao Qin, Hongxu Chen, and Yulei Sui. 2020. Typestate-guided fuzzer for discovering use-after-free vulnerabilities. In *ACM/IEEE ICSE*. 999–1010.
[43] Junjie Wang, Bihuan Chen, Lei Wei, and Yang Liu. 2017. Skyfire: Data-driven seed generation for fuzzing. In *IEEE S&P*. 579–594.
[44] Yanhao Wang, Xiangkun Jia, Yuwei Liu, Kyle Zeng, Tiffany Bao, Dinghao Wu, and Purui Su. 2020. Not all coverage measurements are equal: Fuzzing by coverage accounting for input prioritization. NDSS.
[45] Cheng Wen, Haijun Wang, Yuekang Li, Shengchao Qin, Yang Liu, Zhiwu Xu, Hongxu Chen, Xiaofei Xie, Geguang Pu, and Ting Liu. 2020. Memlock: Memory usage guided fuzzing. In *ACM/IEEE ICSE*. 765–777.
[46] Wen Xu, Sanidhya Kashyap, Changwoo Min, and Taesoo Kim. 2017. Designing new operating primitives to improve fuzzing performance. In *ACM SIGSAC CCS*.
[47] Dingning Yang, Yuqing Zhang, and Qixu Liu. 2012. Blendfuzz: A model-based framework for fuzz testing programs with grammatical inputs. In *IEEE TrustCom*. 1070–1076.
[48] Tai Yue, Pengfei Wang, Yong Tang, Enze Wang, Bo Yu, Kai Lu, and Xu Zhou. 2020. Ecofuzz: Adaptive energy-saving greybox fuzzing as a variant of the adversarial multi-armed bandit. In *USENIX Security*. 2307–2324.
[49] Michal Zalewski. 2017. afl-cmin. https://github.com/mirrorer/afl/blob/master/afl-cmin.
[50] Michal Zalewski. 2020. afl-tmin. https://github.com/google/AFL/blob/master/afl-tmin.c.
[51] Peiyuan Zong, Tao Lv, Dawei Wang, Zizhuang Deng, Ruigang Liang, and Kai Chen. 2020. Fuzzguard: Filtering out unreachable inputs in directed grey-box fuzzing through deep learning. In *USENIX Security*. 2255–2269.