

# Bibliography

Todo

October 1, 2018

## References

- [1] Cristian Cadar and Koushik Sen. Symbolic execution for software testing: Three decades later. 56:82–90, 02 2013.
- [2] Manuel Costa, Miguel Castro, Lidong Zhou, Lintao Zhang, and Marcus Peinado. Bouncer: Securing software by blocking bad input. pages 117–130, Stevenson, Washington, USA, October 2007. Association for Computing Machinery, Inc.
- [3] Amal Khalil and Juergen Dingel. Chapter four - optimizing the symbolic execution of evolving rhapsody statecharts. volume 108 of *Advances in Computers*, pages 145 – 281. Elsevier, 2018.
- [4] J.C. King. A new approach to program testing. 10:228–233, 06 1975.
- [5] Dorel Lucanu, Vlad Rusu, and Andrei Arusoaie. A generic framework for symbolic execution: A coinductive approach. *Journal of Symbolic Computation*, 80:125 – 163, 2017. SI: Program Verification.
- [6] M. Papadakis and N. Malevris. Automatic mutation test case generation via dynamic symbolic execution. In *2010 IEEE 21st International Symposium on Software Reliability Engineering*, pages 121–130, Nov 2010.
- [7] Vlad Rusu, Lydie du Bousquet, and Thierry Jeron. An approach to symbolic test generation. 09 2000.
- [8] David Trabish, Andrea Mattavelli, Noam Rinetzky, and Cristian Cadar. Chopped symbolic execution. In *International Conference on Software Engineering (ICSE 2018)*, 5 2018.

- [9] Mathy Vanhoef and Frank Piessens. Symbolic execution of security protocol implementations: Handling cryptographic primitives. In *12th USENIX Workshop on Offensive Technologies (WOOT 18)*, Baltimore, MD, 2018. USENIX Association.
- [10] Willem Visser, Corina S. Păsăreanu, and Sarfraz Khurshid. Test input generation with java pathfinder. In *Proceedings of the 2004 ACM SIGSOFT International Symposium on Software Testing and Analysis, ISSTA '04*, pages 97–107, New York, NY, USA, 2004. ACM.
- [11] Guowei Yang, Suzette Person, Neha Rungta, and Sarfraz Khurshid. Directed incremental symbolic execution. 2014.
- [12] Rui Zhang and Cynthia Sturton. A recursive strategy for symbolic execution to find exploits in hardware designs. pages 1–9, 06 2018.