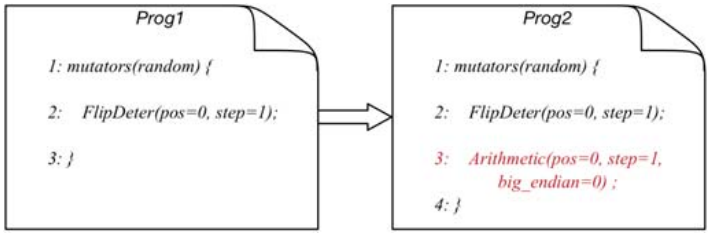

Fig.13 Case 1

图 13 实例 1

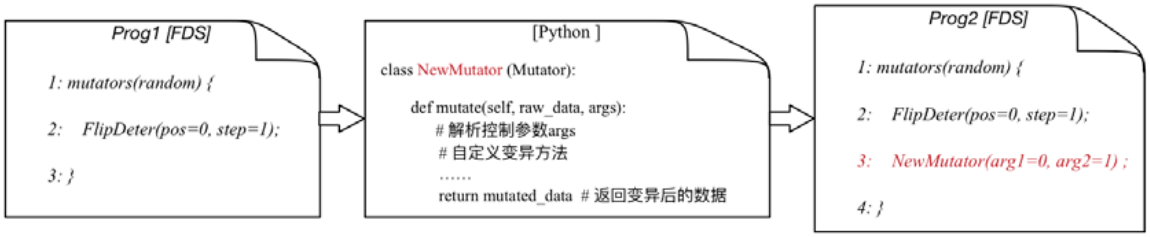

Fig.14 Case 2

图 14 实例 2

4.4 漏洞发现能力

基于 FDS 编程实现具备 AFL 全部功能的模糊测试器,其漏洞发现能力与 AFL 的对比见表 12.本实验中选取了 8 个目标程序,使用相同种子集合,在相同的系统环境下运行 24 小时,实验结果表明,使用可编程模糊测试技术生成的模糊测试器与原始模糊测试器测试效果相当.由于数据变异的随机性以及测试时间的限制,crash 数目与 AFL 比较稍有差异,差异数均在 1 以内.此外,本实验中发现的新的软件缺陷均已向 CVE(CVE-2017-9084, CVE-2017-9335)和软件作者报告.

Table 12 Comparison of vulnerability discovery capabilities

表 12 漏洞发现能力的比较

异常数量	bsdtar	fax2ps	jhead	tidy	tiff2ps	tiffset	pdftohtml	pdfunite
Puzzer	0	1	7	0	1	2	0	1
AFL	0	1	8	0	1	1	0	1

4.5 小 结

综上所示,Puzzer 框架基础原语中包含的变异、监控、反馈原语覆盖了现下开源模糊测试器中绝大多数的基本操作,支持 Windows 平台和 Linux 平台的开源或者闭源的多种类型的目标程序.

- Puzzer 能够通过百行之内的制导程序,快速构建不同功能的模糊测试器.
- Puzzer 能够支持漏洞挖掘人员依据具体测试需求的不同自定义新的原语方法,实现模糊测试器的便捷扩展.
- 使用 Puzzer 框架,基于 AFL 构建生成的模糊测试器的测试效果与 AFL 相当.

可编程模糊测技术降低了开发模糊测试器的开销及门槛,具备良好的扩展能力,能够根据不同测试需求快速构建契合目标程序的模糊测试器,保证了漏洞挖掘的时效性.

5 总 结

为了减少“构建模糊测试器”在整个模糊测试流程中的时间开销、降低模糊测试器开发门槛、增强模糊测试器的扩展性,本文对模糊测试流程进行分解,对 38 款开源模糊测试进行分析,归纳出 53 种基本操作,提出了一

种可编程模糊测试技术.本文抽象出 26 个模糊测试原语,构建了一套用于编写模糊测试器制导程序的语法规范 FDS,并设计了 FDS 解析器以及解析器中的预处理器、模糊测试器引擎和模糊测试原语库,以此支持漏洞挖掘人员通过编写制导程序实现通用型黑/灰盒模糊测试器的全流程、快速、定制化构建,同时实现模糊测试全流程的调试.基于可编程模糊测试原型框架 Puzzer 的实验结果表明:可编程模糊测试技术将模糊测试器的开发成本降至原有的 1%,降低了模糊测试器的开发门槛、增强了模糊测试器的可扩展性,实现了模糊测试器的快速定制化构建.

References:

[1] Sutton M, Greene A, Amini P, Wrote; Huang L, Yu LL, Li H, Trans. Fuzzing: Brute Force Vulnerability Discovery. Beijing: China Machine Press, 2009 (in Chinese).

[2] American fuzzy lop (AFL). 2017. http://lcamtuf.coredump.cx

[3] OSS-Fuzz. 2017. https://github.com/google/oss-fuzz

[4] Pham VT, B¨ohme M, Roychoudhury A. Model-Based whitebox fuzzing for program binaries. In: Proc. of the 31st IEEE/ACM Int'l Conf. on Automated Software Engineering (ASE). 2016. 552−562. [doi: 10.1145/2970276.2970316]

[5] Rawat S, Jainz V, Kumarz A, Cojocar L, Giuffrida C, Bos H. VUzzer: Application-Aware evolutionary fuzzing. In: Proc. of the NDSS 2017. 2017. 1−16. [doi: 10.14722/ndss.2017.23404]

[6] Stephens N, Grosen J, Salls C, Dutcher A, Wang RY, Corbetta J, Shoshitaishvili Y, Kruegel C, Vigna G. Driller: Augmenting fuzzing through selective symbolic execution. In: Proc. of the NDSS 2016. 2016. 1−16. [doi: 10.14722/ndss.2016.23368]

[7] Wang MT, Wei T, Gu G, Zou W. Taintscope: A checksum-aware directed fuzzing tool for automatic software vulnerability detection. In: Proc. of the 2010 IEEE Symp. on Security and Privacy (IEEE S&P 2010). 2010. 497−512. [doi: 10.1109/SP.2010.37]

[8] Gascon H, Wressnegger C, Yamaguchi F, Arp D, Rieck K. PULSAR: Stateful black-box fuzzing of proprietary network protocols. In: Proc. of the SecureComm. 2015. 330−347. [doi: 10.1007/978-3-319-28865-9_18]

[9] Tsankov P, Dashti MT, Basin D. SECFUZZ: Fuzz-Testing security protocols. In: Proc. of the Automation of Software Test (AST). 2012. 1−7. [doi: 10.1109/IWAST.2012.6228985]

[10] Woo M, Cha SK, Gottlieb S, Brumley D. Scheduling black-box mutational fuzzing. In: Proc. of the 20th ACM Conf. on Computer and Communications Security (CCS 2013). 2013. 511−522. [doi: 10.1145/2508859.2516736]

[11] Huang Y, Zeng FP, Cao Q. Fuzzing test approach based on dynamic tracking of library functions. Computer Engineering, 2010, 36(16):39−41 (in Chinese with English abstract).

[12] Rebert A, Cha SK, Avgerinos T, Foote J, Warren D, Grieco G, Brumley D. Optimizing seed selection for fuzzing. In: Proc. of the 23rd USENIX Security Symp. (USENIX Security 2014). 2014. 861−875.

[13] Böhme M, Pham VT, Roychoudhury A. Coverage-Based greybox fuzzing as Markov chain. In: Proc. of the 23rd ACM Conf. on Computer and Communications Security (CCS 2016). 2016. 1−12. [doi: 10.1145/2976749.2978428]

[14] Zhao YH, Kan JJ. Research and design of symbol execution-based test data generation method. Computer Applications and Software, 2014,31(2):303−306 (in Chinese with English abstract).

[15] Ma JX, Zhang T, Li ZJ, Zhang JX. Improved fuzzy analysis methods. Journal of Tsinghua University, 2016,56(5):478−483 (in Chinese with English abstract).

[16] Wu ZY, Xia JJ, Sun LC, Zhang M. Survey of multi-dimensional fuzzing technology. Application Research of Computers, 2010, 27(8):2810−2813 (in Chinese with English abstract).

[17] Wang ZQ, Zhang YQ, Liu QX, Huang TP. Algorithm for discovering SNMP protocol vulnerability. Journal of Xidian University, 2015,42(4):20−26 (in Chinese with English abstract).

[18] Peach fuzzer platform. 2017. http://www.peachfuzzer.com/products/peach-platform/

[19] Honggfuzz. 2017. https://github.com/google/honggfuzz

[20] Choronzon. 2017. https://github.com/CENSUS/choronzon

[21] Sulley fuzzer. 2017. https://github.com/OpenRCE/sulley

[22] ASAN. 2017. https://github.com/google/sanitizers/wiki/AddressSanitizer

[23] Serebryany K. Libfuzzer: A library for coverage-guided fuzz testing (within llvm). 2017. http://llvm.org/docs/LibFuzzer.html

[24] SANCOV. 2017. http://clang.llvm.org/docs/SanitizerCoverage.html

附中文参考文献:

[1] Sutton M, Greene A, Amini P,著;黄陇,于莉莉,李虎,译.模糊测试:强制性安全漏洞发掘.北京:机械工业出版社,2009.

[11] 黄奕,曾凡平,曹青.基于库函数动态跟踪的 Fuzzing 测试方法.计算机工程,2010,36(16):39−41.

[14] 赵跃华,阚俊杰.基于符号执行的测试数据生成方法的研究与设计.计算机应用与软件,2014,31(2):303−306.

[15] 马金鑫,张涛,李舟军,张江霄.Fuzzing 过程中的若干优化方法.清华大学学报,2016,56(5):478−483.

[16] 吴志勇,夏建军,孙乐昌,张旻.多维 Fuzzing 技术综述.计算机应用研究,2010,27(8):2810−2813.

[17] 王志强,张玉清,刘奇旭,黄庭培.一种简单网络管理协议漏洞挖掘算法.西安电子科技大学学报,2015,42(4):20−26.

杨梅芳(1993－),女,内蒙古通辽人,硕士生,主要研究领域为软件安全,程序分析.

霍玮(1982－),男,博士,副研究员,博士生导师,CCF 专业会员,主要研究领域为软件漏洞挖掘和安全评测,基于大数据的软件安全分析,智能终端系统及应用安全分析.

邹燕燕(1989－),女,助理研究员,CCF 专业会员,主要研究领域为软件安全,程序分析.

尹嘉伟(1994－),男,硕士生,CCF 学生会员,主要研究领域为软件安全,程序分析.

刘宝旭(1972－),男,博士,研究员,博士生导师,CCF 专业会员,主要研究领域为网络攻防技术,安全态势感知技术.

龚晓锐(1973－),男,高级工程师,主要研究领域为网络攻防,软件逆向分析,Web 安全,移动互联网安全.

贾晓启(1982－),男,博士,研究员,博士生导师,主要研究领域为网络攻防技术,操作系统安全,云计算安全.

邹维(1964－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为软件安全分析理论与技术,网络安全评测.