Accepted Manuscript

Title: A systematic review of fuzzing techniques

Author: Chen Chen, Baojiang Cui, Jinxin Ma, Runpu Wu, Jianchao Guo, Wenqian Liu

PII: S0167-4048(18)30065-8
DOI: https://doi.org/10.1016/j.cose.2018.02.002
Reference: COSE 1287

To appear in: *Computers & Security*

Received date: 7-10-2017
Revised date: 26-1-2018
Accepted date: 3-2-2018

Please cite this article as: Chen Chen, Baojiang Cui, Jinxin Ma, Runpu Wu, Jianchao Guo, Wenqian Liu, A systematic review of fuzzing techniques, *Computers & Security* (2018), https://doi.org/10.1016/j.cose.2018.02.002.

A systematic review of fuzzing techniques

Chen Chen[a,*], Baojiang Cui[a], Jinxin Ma[b], Runpu Wu[b], Jianchao Guo[a], Wenqian Liu[a]

[a]*Beijing University of Posts and Telecommunications*

[b]*China Information Technology Security Evaluation Center*

Chen Chen: doctoral student, Beijing University of Posts and Telecommunications, research field: Software security detection, Embedded system security detection.

Baojiang Cui: doctor, professor, beijing university of posts and telecommunications, research field: network and host security behavior analysis, software security detection, analysis of web/software and operating system security defects, smart terminals and mobile internet security, internet of things security.

Jinxin Ma: doctoral, associate research fellow, china information technology security evaluation center, research field: information safety

Runpu Wu: master, associate research fellow, china information technology security evaluation center, research field: information safety.

Jianchao Guo: master, Beijing University of Posts and Telecommunications, research field: Software security detection, Embedded system security detection.

Wenqian Liu: master, Beijing University of Posts and Telecommunications, research field: Software security detection, Embedded system security detection.

Abstract

Fuzzing is an effective and widely used technique for finding security bugs and vulnerabilities in software. It inputs irregular test data into a target program to try to trigger a vulnerable condition in the program execution. Since the first random fuzzing system was constructed, fuzzing efficiency has been greatly improved by combination with several useful techniques, including dynamic symbolic execution, coverage guide, grammar representation, scheduling algorithms, dynamic taint analysis, static analysis and machine learning. In this paper, we will systematically review these techniques and their corresponding representative fuzzing systems. By introducing the principles, advantages and disadvantages of these techniques, we hope to provide researchers

*Corresponding author

Email address: 00152tenten@bupt.edu.cn (Chen Chen)