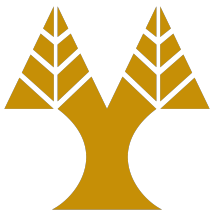
Thesis Dissertation

WEBFUZZ: IMPLEMENTATION OF A GRAY-BOX FUZZING TOOL FOR WEB APPLICATIONS

Marcos Antonios Charalambous

UNIVERSITY OF CYPRUS



COMPUTER SCIENCE DEPARTMENT

December 2020

UNIVERSITY OF CYPRUS

COMPUTER SCIENCE DEPARTMENT

webFuzz: Implementation of a Gray-box Fuzzing Tool for Web Applications

Marcos Antonios Charalambous

Supervisor

Dr. Elias Athanasopoulos

Thesis submitted in partial fulfilment of the requirements for the award of degree of Bachelor in Computer Science at University of Cyprus

December 2020

Acknowledgments

I would like to express my gratitude to my Thesis Supervising Professor Dr. Elias Athanasopoulos for his valuable guidance, encouragement and advices he provided me over the course of accomplishing my dissertation. During the past one year, Dr. Athanasopoulos interest, excitement and refined knowledge in the field of Cybersecurity has been undoubtedly a source of inspiration for me. All these have made my endeavour an exciting experience.

Also, I would like to thank my fellow students Demetris Kaizer and Orpheas Van Rooy for their excellent teamwork and participation to a greater project that consists of each ones thesis.

In addition, I would like to thank PhD. candidate Michalis Papapevripides of SREC Lab for its continuous response to any issues that arose and for the assistance it provided me in resolving them.

Furthermore, I would like to thank all my professors from whom I received invaluable knowledge and helped me to become a Computer Scientist during my four years of study at the Department of Computer Science of the University of Cyprus.

Finally, I would like to thank my family and friends for being with me during my life and supporting me at every step.

Abstract

Testing software is a common practice for exposing unknown vulnerabilities in security-critical programs that can be exploited with malicious intent. A bug-hunting method that has proven to be very effective is a technique called fuzzing. Specifically, this type of software testing has been in the form of fuzzing of native code, which includes subjecting the program to enormous amounts of unexpected or malformed inputs in an automated fashion. This is done to get a view of their overall robustness to detect and fix critical bugs or possible security loopholes. For instance, a program crash when processing a given input may be a signal for memory-corruption vulnerability.

Although fuzzing significantly evolved in analysing native code, web applications, invariably, have received limited attention, so far. This thesis explores the technique of grey box fuzzing of web applications and the construction of a fuzzing tool that will automate the process of discovering bugs in web applications.

We design, implement and evaluate webFuzz, which is the first gray-box fuzzer for web applications. webFuzz leverages instrumentation for successfully detecting reflective Cross-site Scripting (XSS) vulnerabilities faster than other black-box fuzzers. The functionality of webFuzz is demonstrated using WordPress and Drupal.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Contents

Once upon a time [**?**].

The Art of Computer Programming (sometimes known by its initials TAOCP) is a comprehensive monograph written by Donald Knuth that covers many kinds of programming algorithms and their analysis.

Knuth began the project, originally conceived as a single book with twelve chapters, in 1962. The first three volumes of what was then expected to be a seven-volume set were published in 1968, 1969, and 1973. The first published installment of Volume 4 appeared in paperback as Fascicle 2 in 2005. The hardback Volume 4A, combining Volume 4, Fascicles 0-4, was published in 2011. Volume 4, Fascicle 6 ("Satisfiability") was released in December 2015, to be followed by Volume 4, Fascicle 5 ("Mathematical Preliminaries Redux; Backtracking; Dancing Links") in October 2018. Fascicles 5 and 6 are expected to comprise the first two thirds of Volume 4B.

1.1 Motivation

1.2 Related Work

1.3 Contributions

Chapter 2

Background

Contents

In this section we provide background information, to give a detailed understanding about this thesis. First,we briefly discuss what fuzzing is and the various categories that constitute it. Then, we define what a Cross-Site Scripting bug is in web applications, and elaborate on an example regarding this vulnerability.

2.1 Fuzzing

Software testing is a promising technique for discovering unknown vulnerabilities in programs. In particular, software testing has been realized in the form of fuzzing of native code, where software is exercised using a vast amount of inputs for inferring if any of them introduces security-related side effects. A fuzzer can be categorized in relation to its awareness of the program structure as black-, white-, or grey-box. A black-box fuzzer treats the program as a black box and is unaware of internal program structure. Hence, they are only able to scratch the surface usually and expose "shallow" bugs. A white-box

fuzzer leverages program analysis to systematically increase code coverage or to reach certain critical program locations. They may also leverage symbolic execution in order to determine what inputs cause each part of a program to execute [47]. Therefore, they can be very effective at exposing bugs that hide deep in the program. A fuzzer is considered grey-box when it leverages instrumentation rather than program analysis to glean information about the coverage of a generated input from the program it tries to fuzz [37].

2.2 Web Application Bugs

2.3 Instrumentation

code coverage

2.4 Concurrency

2.5 Docker

Chapter 3

Selfish Mining Strategies

Contents

3.1 Modelling of Mining Procedure

3.1.1 Defining Mining Environment

3.1.2 Strategies as State Machines

3.2 Selfish Mining

3.3 Stubborn Selfish Mining

3.3.1 Lead

3.3.2 Equal-Fork

3.3.3 Trail

3.4 Conservative Stubborn Selfish Mining

3.4.1 Safe-Lead

3.4.2 Safe-Equal-Fork

3.5 Hybrid Strategies

Chapter 4

Implementation

Contents

4.1 Selfish Miner

4.2 Honest Miner

4.3 Testing Model

Chapter 5

Evaluation

Contents

5.1 Methodology

5.2 Dominant Strategies

5.3 Revenue and Comparison with Stubborn Strategies

5.4 Fairness of Blockchain

5.5 Risk Safety Property

Chapter 6

Conclusion

Contents

6.1 Conclusion

6.2 Future Work

Appendix A

Appendix B