Automated Security Test Case Generation

Jeremy Gonzalo Arellano
Dept. of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN, USA
jarella2@nd.edu

I. Introduction

In today's modern world, software is constantly being faced with the threat of security breaches. Often, these breaches can be severe, leaving data compromised and services possibly suspended in the process [1]. While we have developed ways to prevent such breaches from happening thanks to the many static and dynamic approaches to security analysis, there still lies the problem of diagnosing and exposing any security threats that may still be lying around after this analysis [4]. Therefor it is critical that some form of automated test case generation exist, where these generated tests will uncover potential vulnerabilities that often would be overlooked in normal analysis [4]. While current security testing methods are proficient at uncovering bugs/vulnerabilities, the amount of manual effort and time that these methods waste are not very efficient to large DevOps and agile environments [6]. Thus, the need for automated security test case generation is needed crucially in the world of technology today.

Software vulnerabilities stem from a plethora of sources, stemming from both low-level coding bugs that are generated from when the developer is creating the source code, to high-level design flaws that have to do with the architecture of a software piece [2][3]. While several tools and framework exist for identifying coding bugs and linting, detecting security bugs is another issue that requires different approaches and different scopes. In many cases, no security tool is perfect since there are many different vulnerabilities to consider and many more that emerge with evolving technology. Beyond just programming, a software's or system's architecture can also play a role in the security of itself, often discovering flows and potential security weaknesses based on design choices [5]. And beyond the architecture or a program/system, the evolution of a project exacerbates the challenge of maintaining a robust security posture throughout the development lifecycle [6].

This project aspires to bridge this gap by proposing an automated framework for security test case generation. Leveraging different algorithms, the framework aims to autonomously generate a suite of security test cases, prioritized on an integrated risk assessment system [4]. Thus, this approach not only augments the existing security testing process but also facilitates a more proactive stance towards software security, aiding in the early detection and remediation of security flaws [5].

Thus, throughout this research project, the aim is to answer the following questions:

1) How can we use evolutionary algorithms to automatically create security test cases?

2) How does rnaking test cases based on risk affect the detection of seriuous vulnerabilities?

3) How does our autoamted framework compare to current security testing methods in effectiveness and adaptability? How does our method compare to manual testing

By analyzing these questions throughout the scope of this research project, the goal of creating and providing a new approach to automated solution for security test case generation. This project aims to contribute to the wider range of the software development ecosystem, providing a solution that will enhance the capability to uncover and resolve software vulnerabilities in everyday development.

II. Related Work

…

III. Methodology

…

Rererences

[1]S.Markstiener, R.Ramler. Integrating Threat Modeling and Automated Test Case Generation into Industrialized Software Security Testing, 2019
[2] T. I. C. S. C. for Secure Design. Avoiding the top 10 security design flaws, 2014.
[3] G. McGraw. Software security: building security in, volume 1. Addison-Wesley Professional, 2006
[4] Fraser, G., & Arcuri, A. (2014). A large-scale evaluation of automated unit test generation using EvoSuite.
[5] B.Chess, & G.McGraw. Static analysis for security, (2004)
[6] B. Boehm & R. Turner. Management Challenges to Implement Agile Processes in Traditional Development Organization, 2005
[7] Zalewski, M. (2011). American fuzzy lop.