LLM-Based Adaptive Fuzzing for Vulnerability Discovery

Vigneshwar Sundararajan
vigneshwar_ks@ucf.edu
University of Central Florida
Orlando, Florida, USA

1 PROBLEM STATEMENT

Traditional methods of vulnerability discovery in software, such as manual penetration testing and conventional automated tools, often fail to detect complex vulnerabilities that occur in less predictable scenarios. These methods typically rely on predefined input patterns and lack the ability to simulate the intricate and creative strategies that skilled human testers employ. This limitation results in significant security vulnerabilities remaining undiscovered, especially as software systems grow in complexity and scale. Consequently, there is a pressing need for more advanced, dynamic, and adaptive testing methodologies that can mimic human ingenuity to uncover hidden security flaws effectively.

2 PROPOSED TECHNIQUE

This project proposes the development of an adaptive fuzzing framework utilizing a Large Language Model (LLM) to enhance software security by intelligently generating test inputs that simulate real-world, edge-case scenarios. The LLM will leverage advanced Natural Language Processing (NLP) techniques to analyze vast data sources, including existing vulnerability databases, software documentation, and common input patterns, to generate potential edge-case scenarios that could expose vulnerabilities.

2.1 Key Features of the Proposed Framework

LLM Core: Serves as the central processing unit, using input from other modules to generate and prioritize testing inputs and learn from the outcomes of previous fuzzing sessions.

Training Module: Prepares the LLM specifically for pentesting tasks by training it on relevant data, enhancing its ability to generate effective fuzzing scenarios.

System Process: Manages system-level operations, ensuring efficient resource allocation and operation security throughout the fuzzing sessions.

State Manager: Maintains the session's current state and context, facilitating dynamic decision-making based on the progress of the fuzzing process.

Tool Manager: Handles the execution and integration of traditional fuzzing tools, ensuring that the generated inputs are applied and tested effectively.

Response Analyzer: Analyzes outputs from fuzzing tools to identify potential vulnerabilities and provides feedback to the LLM for refining future test inputs.

3 EXPECTED OUTCOMES

(1) **Increased Vulnerability Detection**: The project aims to significantly increase the detection of complex vulnerabilities, particularly those associated with edge cases that are typically overlooked by traditional testing methods.

(2) **Enhanced Testing Efficiency**: By automating the generation and application of test cases, the framework is expected to reduce the time and manpower required for comprehensive software vulnerability assessments.

(3) **Adaptability and Scalability**: The LLM-based framework will provide a scalable and adaptable solution that can be customized to different software environments, ensuring that it can handle a variety of software configurations and complexities efficiently.

(4) **Detailed Security Reporting**: The framework will generate detailed, actionable reports on discovered vulnerabilities, helping developers and pentesters to prioritize and address security issues more effectively. This facilitates a more targeted approach to securing software, allowing teams to focus on the most pressing vulnerabilities first.