Graduation Proposal

Name: Code Healer

Executive Summary:

The project is an innovative approach to software testing. It will be using ChatGPT to automate software testing. The main objective is to develop a software that not only automates the error handling and finding vulnerabilities, but it also uses ChatGPT's Api to suggest resolutions, this significantly increases the automation of debugging process in a software development. This approach aims to increase efficiency and reliability and reduces time and resources required to debug any software.

Project Background:

The world is evolving rapidly in software development, ensuring the security and reliability of software is important. Usually testing methods are effective in combating cyber-attacks, but over the years the attacks have become more sophisticated, and the traditional methods have become obsolete. Automated fuzzing has become the mainstream tool in identifying vulnerabilities. A fuzzing tool remains labor intensive and time consuming. This project aims to break the bottleneck by integrating Ai into the fuzzing process.

Importance of the Topic:

The integration of AI into the fuzzing workflow represents a significant advancement in the cybersecurity domain. The current methods often fall short in keeping up with the modern cyber security domain. There is a necessity for a more advanced approach, by automating the detection of vulnerabilities but also providing a solution corresponding to the problem. This project directs this critical need and provides significant improvements to the efficiency of testing software and resolution practices.

Detailed Problem and Present solution:

Applications (Software) are being targeted by cyber-attacks to exploit vulnerabilities ranging from: buffer overflows to SQL injection attacks. The manual debugging methods are slow and

have a chance of missing critical vulnerabilities. The solution proposes the use of an automated fuzzer, that identifies a bug and utilizes AI to search for and suggest a potential fix. This approach tremendously increases the workflow and resolution finding process. It also increases the accuracy and understanding of the bug.

Comprehensive Deliverables, Goals and Timeline

Phase 1:

- **Deliverables**: A detailed report on existing fuzzing tools and an in-depth analysis of the ChatGPT API capabilities.
- **Goals**: To acquire a thorough understanding of current fuzzing technologies and how the ChatGPT API can be leveraged for enhancing automated testing processes.
- **Timeline: 1 month**

Phase 2

- **Deliverables**: A basic yet functional fuzzer framework ready for integration with the ChatGPT API.
- **Goals**: To create a scalable and robust fuzzer framework that serves as the backbone for subsequent AI integration and testing.
- **Timeline: 1 month**

Phase 3

- **Deliverables**: A fully integrated system with preliminary testing reports highlighting the integration's success and areas for improvement.
- **Goals**: To seamlessly integrate the ChatGPT API with the fuzzer framework and conduct initial tests to evaluate system performance.
- **Timeline: 2month**

Phase 4

- **Deliverables**: An enhanced fuzzer system with advanced testing outcomes and a beta version of the report generation feature.
- **Goals**: To refine system capabilities based on initial feedback and develop a report generation feature that automates the documentation of findings.
- **Timeline: 3month**

Phase 5

- **Deliverables**: A comprehensive project report summarizing the development process, system testing results, and recommendations for future enhancements.
- **Goals**: To ensure the system meets all specified performance criteria and to compile a final report detailing the project's findings, achievements, and potential future directions.
- **Timeline: 3 month**

Challenges and Mitigation strategies:

The challenge of using AI is the interpretation accuracy and the accidental generations of false positives are expected. The main concern is the privacy aspect of using AI in an automated workflow.

Conclusion and future implications:

Automating the detection of vulnerabilities and generating a solution, the project is set to revolutionize the software testing and debugging methods. The potential for this solution to impact the industry standards and extend to other cybersecurity applications. Like automating penetration testing this highlights the importance and the benefits it could have in enhancing software reliability and security.