Proposal: Utilizing Advanced LLM for Automated Software Vulnerability Detection and Remediation

Abstract

This paper delves into the deployment of a fine-tuned GPT-3 model, powered by OpenAI's API, uniquely tailored for identifying software vulnerabilities. By integrating process flow and static analysis, this GPT-3 model scans codebases, pinpoints vulnerabilities, and then automates the remediation process by autonomously generating and committing pull requests, heralding a paradigm shift in software security.

---

1. Introduction

With software development evolving at an unmatched pace, the imperative to ensure security is paramount. In the backdrop of mounting sophisticated threats, we introduce a solution leveraging a fine-tuned GPT-3 model via OpenAI’s API. This innovative approach offers a holistic and automated lens to tackle software vulnerabilities.

1.5. Competing in DARPA’s AI Cybersecurity Initiative

In August 2023, during the esteemed Black Hat USA event, the Defense Advanced Research Projects Agency (DARPA) put forth an ambitious challenge: The AI Cyber Challenge (AIxCC). This two-year competition, as articulated by DARPA, seeks to marshal the best minds from various domains — computer scientists, AI researchers, software developers, and more — to revolutionize the interface of AI with cybersecurity.

Our endeavors with the GPT-3 model for vulnerability detection and automated remediation are primarily geared towards this initiative. We envision our solution not only as a response to DARPA's call to action but as a pioneering approach that can set new benchmarks in the realm of AI-augmented cybersecurity tools. Participating in AIxCC aligns with our aspiration to contribute significantly to the ever-evolving landscape of secure software development and demonstrate the prowess of AI-driven tools in mitigating sophisticated threats.

---

2. Methodology

- **2.1. Process Flow Analysis:**
    - Evaluating the software's operational dynamics, this technique aims to detect logical flaws and vulnerabilities before they become exploitable.
- **2.2. Static Analysis:**
    - This method inspects the code without execution, detecting patterns that may lead to vulnerabilities, offering an efficient first line of defense.

- **2.3. GPT-3 Integration via OpenAI’s API:**
    - Leveraging advanced algorithms, the fine-tuned GPT-3 ingests findings from the analyses, identifying intricate vulnerabilities potentially missed by traditional tools.
- **2.5. Model Training and Fine-Tuning:**
    - Employing GPT-3 via OpenAI's API obviates the need for extensive model training. Its inherent breadth and adaptability eliminate the resource-intensive process of training a bespoke model. While GPT-3 offers substantial out-of-the-box capabilities, fine-tuning remains an optional step. This flexibility allows organizations to optionally tailor GPT-3's outputs more precisely to the software security domain, balancing general capabilities with domain-specific nuances.

---

3. Automated Remediation

- **3.1. Reduces Human Intervention:** Limiting potential errors in the remediation process.
- **3.2. Speeds up Response Time:** Instantaneous pull requests mean vulnerabilities are patched almost immediately, minimizing exposure.
- **3.3. Ensures Consistency:** Pull requests are standardized, guaranteeing consistency in remediation across codebases.

---

4. Benefits

- **4.1. Enhanced Security Posture:** Immediate remediation translates to fortified software applications.
- **4.2. Efficiency:** By offloading security concerns to the GPT-3 model, developers can concentrate on their primary tasks.
- **4.3. Cost-effective:** Proactive measures help avert breaches, leading to savings on damage control and associated costs.

---

5. Conclusion

Utilizing a fine-tuned GPT-3 model via OpenAI's API for vulnerability detection and automated remediation signifies a monumental stride in software security protocols. This methodology doesn't just detect but also acts, ensuring a blend of functionality and intrinsic security in software applications. As cybersecurity demands continue to escalate, such innovations anchor the vanguard of automated security measures.

1. Set of questions or problems you hope your project will answer or address:

- What is the efficiency of the LLM in identifying software vulnerabilities compared to traditional methods?
- What types of vulnerabilities are most commonly detected, and which are potentially missed?
- How do the LLM's false positives/negatives compare with other existing vulnerability detection tools?

2. Description of methodologies and approaches used in the project:

- LLM Integration: The LLM will be integrated into our development environment, enabling direct access to the codebase and the ability to generate and commit pull requests.
- Static Analysis: This method will inspect the code without executing it, identifying patterns indicative of vulnerabilities.
- Process Flow Analysis: This approach evaluates the flow of operations in the software, detecting potential logic flaws or unexpected exploit pathways.
- Dynamic Analysis (in later stages): Execution of the software to uncover vulnerabilities that may not be apparent during static analysis but surface during runtime.
- Automated Pull Request Generation: Based on detected vulnerabilities, the system will automatically create and commit pull requests to the respective repositories.

3. Expected results of the project:

- Elevated Security: A significant reduction in vulnerabilities across the scanned codebase, leading to more secure software applications.
- Streamlined Remediation: With pull requests generated and committed automatically, we expect swift vulnerability addressal.
- Accuracy Metrics: A comprehensive understanding of the LLM's accuracy in vulnerability detection and the relevance of its pull request suggestions.
- Operational Efficiency: The automated system should lead to a decrease in manual intervention and review time for vulnerability patches.
- Integration Standard: By year's end, the aim is for the LLM-driven vulnerability detection and auto-commit feature to be an essential part of our software development lifecycle.