Charger Active Defense v1.0 Team 2 - Group 12

Noah Sickels, Adam Brannon, William Lochte

**Project Abstract**

Cybersecurity attack tools are easily accessible on the Internet and often require minimal technical skill to use. Both experienced hackers and novices, often referred to as "script kiddies," utilize these tools. Additionally, these tools are highly efficient on modern hardware, capable of generating thousands of packets per second.

Our project aims to develop a workflow for conducting fuzz testing on these adversarial attack tools to identify potential binary exploitation vulnerabilities. By discovering these vulnerabilities, we can take measures to slow down or completely disable the attack tools.

Traditional intrusion prevention systems protect organizations by blacklisting abnormal or suspicious behaviors. This conventional approach typically involves disabling services or ports from which an attack originates or blocking the source IP address. However, attackers can circumvent these protections by attacking a different port/service or spoofing their IP address. In contrast, our method actively targets and disrupts the malicious application to prevent it from functioning. This method effectively prevents or stalls the adversary from further attacks against the system/organization.   
  
Some of the major steps that we have made so far to achieve this goal include:

* Researched CVEs for each of our preliminary attack tools.
* Performed various forms of static and dynamic analysis on each attack tool.
* Explored and tested a variety of fuzzing tools.
* Created an automated fuzz-testing workflow for our attack tools.
* Performing extensive fuzz testing with AFLnet against Masscan.

In the future, we intend to:

* Research AI/LLM models that we can use for network-based behavioral/anomaly detection methods.
* Research and test two different AI/LLMs for detection and response in the Python service.
* Research various methods that are typically used in training models for improving the accuracy and confidence of the AI/LLM.
* Create the active defense Python service. When the service detects the attack tools on the network, it sends the appropriate response back to the port or service they are from.

**Current Project Status**

We conducted background research by screening all six potential attack tools using LDRA Static Analysis and Valgrind to check for memory leaks. Based on these results, we selected Medusa and Masscan for further testing. We also tested the compatibility of six fuzzing tools: AFLnet, Fuzzowski, Scapy, Radamsa, Randpkt, and Peach Fuzzer. We found that all tools were compatible except for Peach Fuzzer, so we ultimately decided to focus our testing on AFLnet and Radamsa.

Additionally, we developed a workflow script that can automatically install and build our selected attack tools, fuzzing tools, and other necessary dependencies for fuzz testing. The script also includes an option to uninstall these tools. The attack tools installed by the script are Medusa and Masscan, while the selected fuzzing tools are AFLnet and Radamsa. Alternatively, we began working on a Docker implementation to provide a more streamlined solution going into the second semester.

Our rationale for attack tools, fuzzing tools, and compatibility testing results and analysis is compiled and outlined in three separate reports: G12\_fuzz\_tool\_selection\_report, G12\_attack\_tool\_selection\_report, and G12\_fuzzing\_results\_analysis.

We are currently conducting fuzz testing on the Metasploitable2 VM with AFLnet and Masscan on a separate server belonging to a member. To improve the execution speed of our testing, we will follow up with UAH's System Administrator and/or the Alabama Supercomputer Authority's servers.

**Short-Term Goals & Activities for Next Period**

Our short-term goals and activities for the next reporting period will focus on researching two AI/LLM models we can use for the active defense Python service and developing a foundation for the program.

|  |  |
| --- | --- |
| Task | Member |
| Research two AI/LLMs we can use for the Python active defense program. | William |
| Research general AI/LLM (ChatGPT) API integration in Python programs. | Adam |
| Develop/draft the initial active defense Python program (network handling module). | Noah |

**Level of Effort / Individual Accomplishments**

|  |  |
| --- | --- |
| Member | Hours |
| Noah | 15 |
| Adam | 5 |
| William | 7 |
| *Total* | 29 |

|  |  |
| --- | --- |
| Member | Individual Accomplishments |
| Noah | Wrote the patents and other projects section of the Marketing Survey & Innovation assignment.   Worked on the updating requirements and brief 1 deliverables.  Created a VS Code devcontainer for developing the Python program.   Restructured GitHub repository for better organization of previous and current semesters artifacts. |
| Adam | Wrote the innovativeness and new knowledge sections of the Marketing Survey & Innovation assignment.  Worked on the updating requirements and brief 1 deliverables. |
| William | Wrote the conference and journal sections of Marketing Survey & Innovation assignment.  Worked on the updating requirements and brief 1 deliverables. |