Charger Active Defense v1.0 Team 2 - Group 12

Noah Sickels, Adam Brannon, William Lochte

**Background / Abstract**

Modern attack tools are efficient, allowing them to attack quickly. [UPDATED ABSTRACT NEEDED]

**Current Project Status, Issues, & Short-Term Activities & Goals**

This reporting period primarily focused on compatibility testing of our second selected fuzz testing tool against Medusa and Masscan attack tools. [TODO]

Short-term goals met for this period:

* Begin testing the viability of the first fuzz testing tool (Fuzzowski/BooFuzz) against Medusa and Masscan.

During this period, we encountered a few issues:

* Fuzzowski is limited in the protocols that it supports and does not natively have support for SSH or PostgreSQL modules as previous testing focused on but does have support for Telnet and TFTP. However, since Fuzzowski is forked from Sully – a fuzz testing framework – it has a lot of custom Python libraries that can be utilized to simplify writing custom fuzz testing modules for use with other protocols. A test module for use with Medusa’s FTP module was written for proof-of-concept implementation, but more thorough testing would be needed. We wished to use either the SSH or PostgreSQL modules instead, but due to time limitations were unable to verify before the end of the second period.
* One significant problem that we will have to continually work with is the difficulty surrounding network protocol fuzz testing with the added factor of the attack tools. While we’ve discussed several different possibilities with our sponsor, including fuzz testing the network-specific source files, or using custom network sockets to facilitate mediate the communication between the target and the attack tool traffic, this will be an ongoing process that we will need to continue exploring.
* Expanding upon the previous issue, while testing Fuzzowski with Medusa’s FTP module, it’s difficult to verify that the fuzz testing is working properly as it focuses more on fuzzing the port and protocol itself as provided. For this reason, we decided to script Medusa to continually perform the brute-forcing while Fuzzowski was running against the specific port and protocol – after Medusa finished, it would repeat the command until the script was terminated. This allowed Medusa to send traffic back and forth to the target and fuzzed the responses back to the host.

For the next reporting period, our short-term goals are:

|  |  |
| --- | --- |
| Task | Responsibility |
| Begin viability/compatibility testing on final tool with either PeachFuzz or Scapy. | Noah |
|  | William |
|  | Adam |

**Milestone Status Summary**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Milestone / Task | Projected Due Date | Completion Date | Status | Notes |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Level of Effort / Individual Responsibility Record**

|  |  |
| --- | --- |
| Member | Hours |
| Noah Sickels | 22 |
| Adam Brannon |  |
| William Lochte |  |

|  |  |
| --- | --- |
| Member | Individual Accomplishments |
| Noah Sickels |  |
| Adam Brannon |  |
| William Lochte |  |

**Milestone Completion & Analysis**

* Fuzz testing with Fuzzowski against Medusa and Masscan for viability.

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

**Mitigation Plan**

* Milestone:

**Contingency Plans**

* Milestone: