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. Our goal is to slow or stop attack tools by sending these tools invalid responses over the network. Some of the major steps that we have made so far to achieve this goal include:

* Researched CVEs for each of our preliminary tools
* Performed various forms of static and dynamic analysis on each tool
* Explored and tested a variety of fuzzing tools

In the future, we plan to:

* Explore more fuzzing tools
* Perform extensive fuzz testing
* Create a fuzz-testing workflow for attack tools

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

This reporting period primarily focused on compatibility testing our third and final fuzz testing tools against Medusa and Masscan. This also included initial drafting of the design review report, and the completion of the tool selection report.

Short-term goals met for this period:

* Test the viability of the third and final fuzz testing tool (PeachFuzzer/Scapy) compatibility with Medusa and Masscan
* Write the attack tool selection report.

During this period, we encountered a few issues:

* Peach Fuzz is a well-established fuzz testing tool that can be used in various applications. However, Mozilla no longer actively maintains the project. While a community version is available, the core packages are not upgraded and run on Python version 2.7. This lack of updates has made testing significantly more challenging. While modern Linux systems still support Python 2.7, finding the required Python libraries was more of a challenge, as the necessary versions were unavailable through Pip or any other Python package manager and had to be manually installed through PyPi using their respective installers.
* Building upon the previous issue, we tried using a Dockerfile to address the problem in a containerized environment. However, we were still unable to resolve it because, in August 2023, Debian removed all Stretch releases from their repository packages. As a result, Java's OpenJDK—one of the core packages needed for Peach Fuzzer—is no longer available through Debian's package manager. We even attempted to download the source repository file and add it manually through the Dockerfile, but we were unsuccessful.
* To effectively use Peach Fuzz against the targeted applications, a second intermediary virtual machine is required that can run complete installations of Python 2.7. For this reporting period, we utilized Ubuntu 18.04. However, testing with this tool requires an additional system outside the Metasploitable2 target virtual machine and the host machine. This process is more complex and may limit our options for more in-depth testing in the upcoming semester.
* Scapy does have a fuzzing module but applying it in a way for the tool to know that it has worked was a bit of a challenge. We tried to use a timer to see how long it will take for the next round of username and passwords to come as our indicator that the tool was successful or not.

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

|  |  |
| --- | --- |
| Task | Responsibility |
| Draft/Develop Fuzzing Workflow.  Draft and complete the Design Review report. | Noah |
| Draft/Develop Fuzzing Workflow.  Draft and complete the Design Review report. | William |
| Draft/Develop Fuzzing Workflow.  Draft and complete the Design Review report. | Adam |

**Milestone Status Summary**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Milestone / Task | Projected Due Date | Completion Date | Status | Notes |
| Test Fuzzing Tool #3 Alternative | 10/26 | 10/26 | Complete | Successfully verified Peach Fuzzer is incompatible with Medusa and Masscan. |
| Test Fuzzing Tool #3 | 10/26 | 10/26 | Complete | Successfully tested Python script Scapy Fuzzer against Medusa FTP module. |
| Analyze Fuzz Testing Results | 10/27 | 10/27 | Complete | We’ve made the decision between the tools depending on the method we want to test, but have not written a separate report for it. We’re not sure if we should, or should just detail it in the final report. |
| Rank Fuzz Tools Based on Probability of Success | 10/27 | 10/27 | Complete |  |
| Draft/Develop Fuzzing Workflow | 11/11 | TBD | Partial | We have started working on the outline for the fuzzing workflow, including collection of testing results and viability of the different fuzzing tools. Effort on this milestone will continue through the next period. |

**Milestones for Next Period**

|  |  |  |  |
| --- | --- | --- | --- |
| Milestone / Task | Start Date | Projected End Date | Notes |
| Develop Fuzzing Workflow | 10/28 | 11/11 |  |
| Draft Design Review Report | 10/22 | 11/3 | We’ll write up the draft for the design review report deliverable. We’ve already started during the previous reporting period, but will continue through this next period until the report due date. |
| Apply fuzzing workflow to Masscan and Medusa | 11/11 | 11/11 | The developed fuzzing workflow will be applied to our attack tools Medusa and Masscan as part of the project functionality/design requirement. |

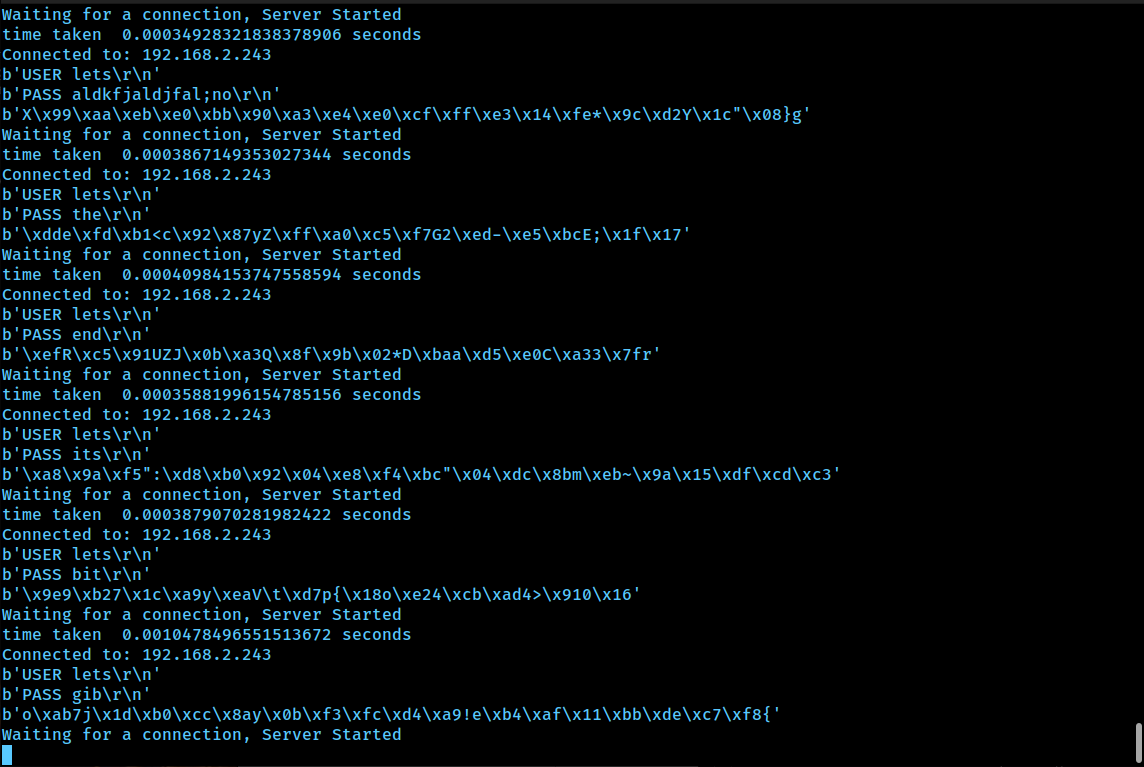
**Level of Effort / Individual Responsibility Record**

|  |  |
| --- | --- |
| Member | Hours |
| Noah Sickels | 30 |
| Adam Brannon | 32 |
| William Lochte | 0 |

|  |  |
| --- | --- |
| Member | Individual Accomplishments |
| Noah Sickels | Peach Fuzz configuration and viability/compatibility testing for use with Medusa and Masscan.  Written attack tool selection report for rationale of attack tools used for the project.  Updated general documentation/info in preparation for design review and final reports. |
| Adam Brannon | Configured Scapy with Python script to send packets back to medusa using the FTP module.  Testing sockets with Python script to send random packets back to medusa using the FTP module. |
| William Lochte | Nothing. |

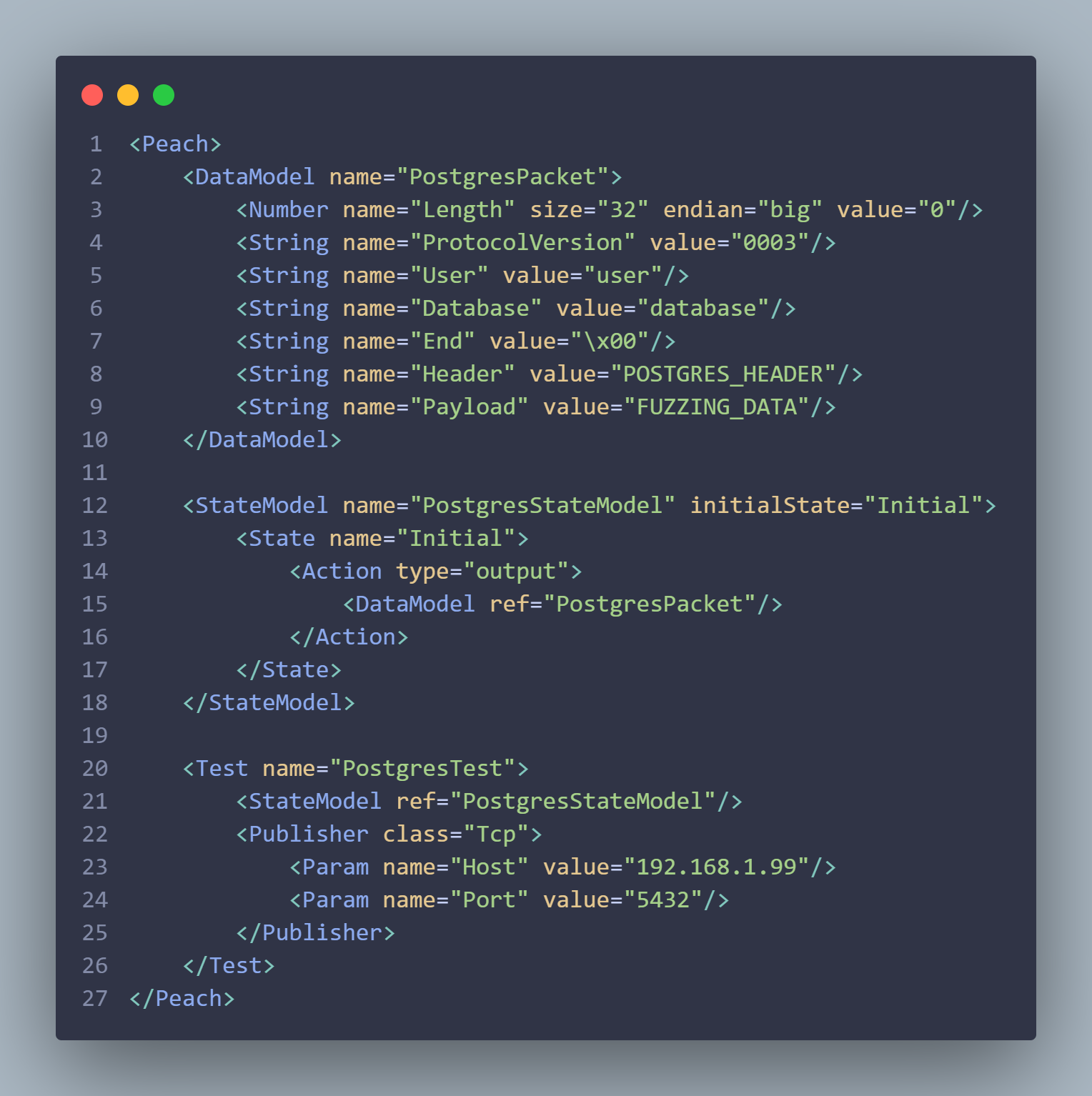
**Milestone Completion & Analysis**

* Tested Python script with Scapy for viability to fuzz test our target Medusa

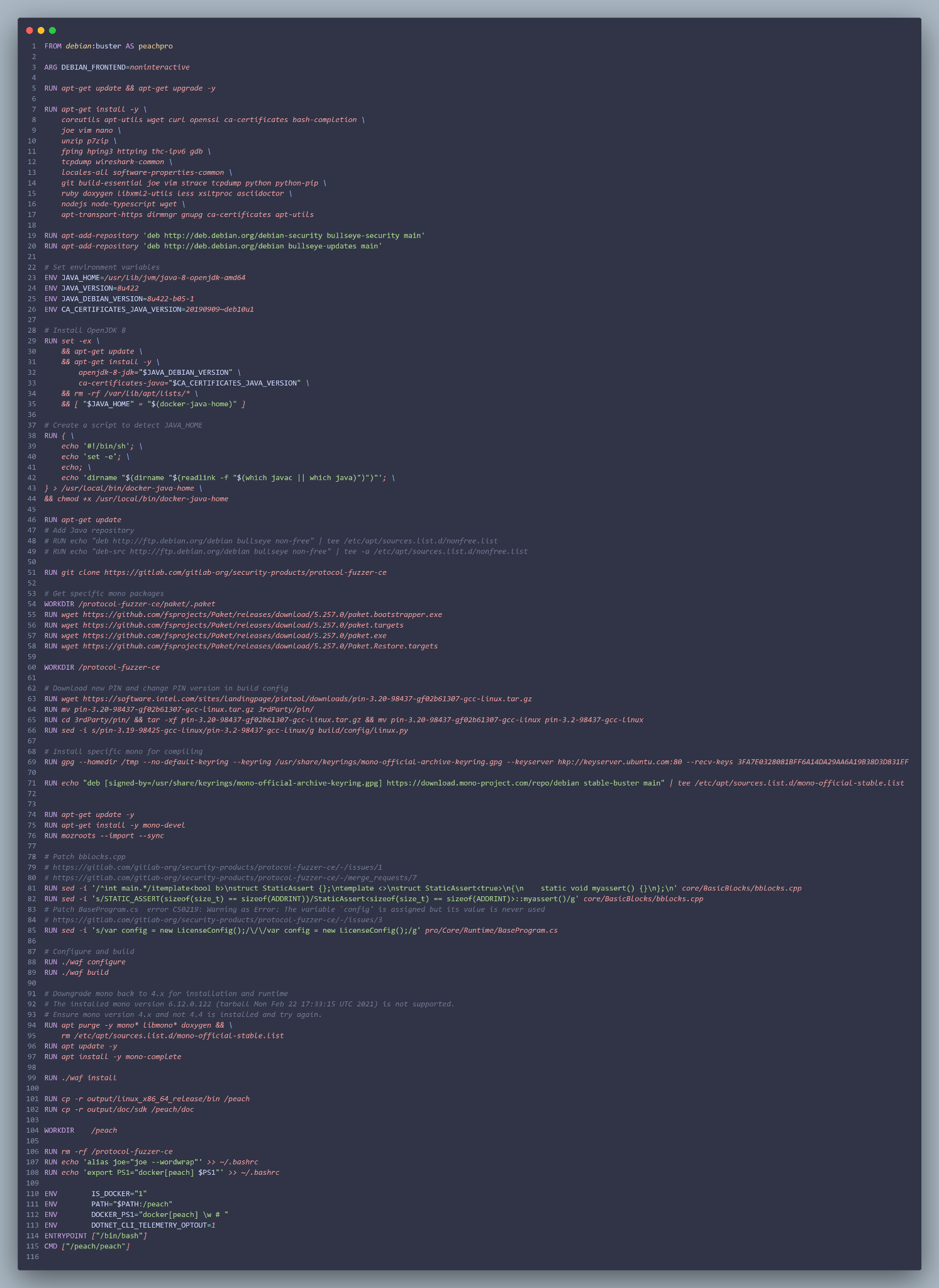




* Tested Peach Fuzz compatibility with Medusa and Masscan (XML configuration file, Dockerfile container)



* Dockerfile we created in attempting to install and configure Peach Fuzzer.



**Mitigation Plan**

* Milestone: Develop Fuzzing Workflow
  + We have several different tools that we have viability tested against our attack tools Medusa and Masscan.
* Draft Design Review Report
  + To mitigate delays we set a time we think we all can work on the report as a group.

* Apply fuzzing workflow to Masscan and Medusa
  + We have several tools that will help make the workflow more resilient to failure.

**Contingency Plans**

* The contingency plans for this next period are the multiple tools and workflows that we have been working on such as Scapy, peach fuzz, Fuzzowski and Aflnet as the fuzzing tools.
* With Peach Fuzz not currently being maintained by Mozilla it will be less likely that Peach Fuzz will be a viable candidate for our workflow, but we have the contingency for it. We have the other three tools that we can apply to the attack tools.