SDN Firewall with POX

Project Overview

OMSCS/OMSCY CS 6250 Computer Networks

Project Information:

This project involves creating a software-defined networking firewall using mininet + OpenFlow + the POX OpenFlow controller. This creates a programmable firewall that potentially could respond to network conditions. However, for this project, we will be creating a simple Blacklist firewall where we specify network traffic to be blocked or allowed via a configuration ruleset.

Walkthroughs

There are several walkthrough videos available on YouTube concerning the different parts of the project:

- Project Overview (this Video)
- Mininet Review (optional)
- Wireshark/Tshark Tutorial
- Implementation Overview
- Ruleset Overview
- Manual Testing Example

Tips and tricks are provided in every video.

Warnings

START EARLY!!!!

This project is not difficult, but there are many separate phases involved. In fact, you will probably spend more time reading the project description and watching some of the Overview videos than programming and configuring the project. There is a lot of prerequisite of IP/TCP/UDP Packet Header knowledge necessary to complete the project.

Reference the resources in Part 0 and use them as a basis for gaining this prerequisite knowledge.

Before You Begin

This project assumes basic knowledge about IP and TCP/UDP Protocols. It is highly encouraged that you review the following items before starting. This will help you in understanding the contents of IP packet headers and what you may need to match.

What is the IP (Internet Protocol)? What are the different types of Network Layer protocols?

Review TCP and UDP? How does TCP or UDP differ from IP?

Examine the packet header for a generic IP protocol entry. Contrast that with the packet header for a TCP packet, and for a UDP packet. What are the differences? What does each field mean?

What constitutes a TCP Connection? How does this contrast with a UDP connection.

A special IP protocol is ICMP. Why is ICMP important? What behavior happens when you do an ICMP Ping? If you block an ICMP response, what would you expect to see?

If you block a host from ICMP, will you be able to send TCP/UDP traffic to it?

Can you explain what happens if you get a ICMP Destination Unreachable response?

What is CIDR notation? How do you subnet a network?

Development Process

There are several distinct phases to this project:

1. Optional Mininet Overview – For Summer 2021, this is your first introduction to mininet unless you completed the optional Simulating Networks project.
2. Wireshark/Tshark Demonstration with a deliverable packet capture.
3. Code Implementation that builds a configurable firewall using the OpenFlow POX controller API in Part 5
4. Firewall Ruleset implementation of a series of firewall rules specified in Part 6
5. Testing your firewall with manual testing and a testing framework.

Development Process

The Code Implementation is best done with the simple sample configure.pol provided with the project files. After you get a working code implementation with that file, start building the Part 6 ruleset and testing your firewall with those rules instead.

Start with implementing Block rules and make sure everything works. Then start implementing the Allow rules to figure out how to make the Flow Modification and the Action blocks work to override the Block rule.

Everything needed for implementation is included in Appendix and Project Description.

The ruleset is straightforward – make sure and use the Wikipedia List of TCP/UDP Ports reference to help determine the proper protocol and ports for the different referenced services.

General Tips and Tricks

Specific tips and tricks for the Implementation and Ruleset sections are included in those walkthroughs.

If you have difficulty and are unable to create the code implementation, submit a complete configure.pol file to gain partial credit – this file will be tested twice – first with a known good implementation and second with your implementation.

In the implementation, make sure that Block works for everything before attempting to implement Allow.

What to turn in?

You need to submit your copy of packetcapture.pcap, sdn-firewall.py and configure.pol from you project directory using the instructions from the EdStem Post "How to Submit / Zip Our Projects". To recap, zip up the two files using the following command, replacing gtlogin with your GT Login that you use to log into Canvas:

```
zip gtlogin_sdn.zip packetcapture.pcap configure.pol sdn-firewall.py
```

You may also include an additional text file if you have comments, criticisms, or suggestions for improvement for this project. If you wish to provide this information, add it to your ZIP file with the name comments.txt. This is completely optional.

What to turn in?

Please make sure that you redownload this file and check to make sure that it is complete, contains all of the required files, and is for the right project. After the late submission deadline, we will not be able to accept corrected submissions.

What can I share?

Do not share the content of your sdn-firewall.py, configure.pol, or packetcapture.pcap with your fellow students, on Ed Discussions, or elsewhere publicly. You may share any new topologies, testing rulesets, or testing frameworks, as well as packet captures that do not address the requirements of Part 4b.

Do not post your code in public repositories on Github.

Grading Rubric

- 10 points for submitting a version of sdn-firewall.py that indicates effort was done.
- 10 points for submitting a version of configure.pol that indicates effort was done.
- 30 points for submitting a version of packetcapture.pcap that indicates effort was done.
- 35 points - Your configure.pol file will be tested by a known good firewall implementation and by your sdn-firewall.py file with a series of unit tests to make sure that the rules were implemented. The higher of the two grades will be used (thus, you will not be penalized if your sdn-firewall.py file is not complete or has issues).
- 15 points – This is just a test of your provided sdn-firewall.py and configure.pol to ensure that your code is working (i.e., it tests your implementation, and not your ruleset)
- 50 points – The final portion of the grade consists of testing your sdn-firewall.py file with a different grading ruleset and/or topology to make sure that your code is robust enough to handle any valid configuration file.

If your code crashes from a simple clerical error, a deduction of 20 points for each edit (that resolves the crash) will be done vs losing most of the 15, 35, or 50 points for your code.

Final Thoughts

Good luck.

Start Early.

Don’t overthink – the project is more straight-forward than it appears.

Start Early.

Ask Questions on EdStem.

Share alternate testing situations in the testing thread.