CSCI493-81
Lab 6
Security
Topic: Firewalls

Prof. Dietrich

March 21, 2025

Group size: 4 — Setup needed: Windows/macOS, Unix hosts, SPHERE..[1,2]

1 General description

In this lab, we are building a firewall, using mainly `iptables` under Linux. This will introduce you to network access control schemes and the "principle of least privilege" through the use of `iptables` firewalls. We are also considering `pf` under OpenBSD. The underlying principles can be extended to other operating systems such as FreeBSD.

This is based on an exercise by Peter A. H. Peterson, Peter Reiher, and Tanya Crenshaw.

2 File with samples and results

The files are in lab-fw.zip. The page contains pointers to configurations (basic) and examples for setting up simple firewalls. You will have to develop more complex rule sets for the tasks.

3 Setting up a firewall

In this part of the lab, we will be setting up rule sets for hosts and networks.

3.1 Requirements

- You use the instructions provided with this lab. You should use SPHERE.
- Your output should demonstrate that you have done the experiments, such as screenshots, Unix typescripts, and the firewall logs showing that packets were allowed across the perimeter, or that they were blocked.

[1]This document is Copyright ©2025 Sven Dietrich
[2]This document is for internal CUNY use only

4 Tasks

You are Wilbar Memboob, the Security Administrator of FrobozzCo (”You name it, we make it”). You are looking to hire a new system administrator to replace the guy you just fired – unfortunately, even though his resume looked great on paper and he interviewed well, once you put him at a console he clearly had no idea what he was doing.

To him, proper permissions meant that things worked; if it was secure then that was just icing on the cake. You should have done something when the other admins starting calling him ”J-ACCEPT” – a.k.a. “accept all traffic.” But it was when he changed the company firewall policy to -j ACCEPT and the internal LAN got exposed to some black hat grues that he signed his own pink slip.

To keep this from happening again, you’ve created a little test for your new applicants to take. Unfortunately, before you can grade the applicants, you need to create an answer key.

4.1 Setup

In this exercise you’re going to create an answer key for the iptables portion of the ”hiring exam.” You’ll need to swap in your nodes and complete the exercises below.

1. Follow the instructions for SPHERE from the previous assignments..
2. Create an instance of this exercise, using `firewall` as Lab name. You will have two nodes in your experiment: `server` and `client`.
3. After setting up the lab, access your `server` node.

Changes to SPHERE nodes are lost when the nodes are swapped out. This means that you must manually save your work between working on nodes in order to keep it. However, this exercise includes experimental scripts to help you save and restore your work.

Saving your Work After completing the firewall-related exercises on the `server` node, cd into the /root directory and execute the script `/root/submit.sh`; like this:

```
$ ./submit.sh
```

...it will make a tarball of all the relevant files, including the firewall.sh script you have updated (it will also copy files for the posix access control lab, but you do not need to bother with them; you should ignore the errors). You must copy or move the created tarball into your group directory, otherwise it will be lost upon swapout. **Note: do not run submit.sh with sudo!**

Restoring Your Work The experimental script `/root/restore.sh` on the `server` node, takes as input the path to a tarball created by `submit.sh` described above and restores the files to their proper locations on the system. This includes all the files you are asked to create or change in the first part, as well as the `firewall.sh` script for the second part. **Note: do not run restore.sh with sudo!**

WARNING: These scripts do not back up all arbitrary changes you may have made to the node (e.g., changing a peripheral configuration file), and it does not ”merge” system files with submission files – it only restores submission files copied by `submit.sh`. You shouldn’t need to change anything else, but see `submit.sh` and `restore.sh` to see exactly what those scripts do, and do not delete any of your submission tarballs so that you can go back to an earlier version if need be.

To use the `restore.sh` script, copy a tarball created by `submit.sh` into the `/root/` directory and execute this command:

```
$ ./restore.sh username-permissions-123123123.tar.gz
```

You will be asked if you want to automatically overwrite files, and if you want to selectively restore some files and not others. The options are self-explanatory.

Finally, if you don't trust the scripts, you can always make your own backups into your group directory and restore them by hand if you prefer.

NOTE: You do not need to run the submit and restore scripts with `sudo`. However, if you use sudo to run `submit.sh`, your tarball will be named after the `root` user. This is OK – just run `sudo ./restore.sh root-permissions-2134234243.tar.gz.`

4.2 FrobozzCo Firewall

Your task is to create an answer key for the following test.

4.2.1 FrobozzCo Firewall Test

This test is a part of the application process for the administrator position at FrobozzCo. You will be presented with a number of tasks and questions; the tasks must be executed on the server node of the SPHERE experiment we have created for this purpose (you can use the `client` node for testing). See below for further instructions.

4.2.2 Firewall Configuration

The test server has a totally permissive firewall installed – it accepts all inbound and outbound traffic from all ports, protocols, addresses, interfaces, and states. This is basically like having no firewall at all.

Your task is to configure the firewall according to the principle of "least privilege". This means that it should be maximally restrictive while still permissive enough to allow a strictly defined set of tasks. While some of these rules can also be configured in the server software (this strategy is called "defense in depth" - search for it), we want you to implement the rules in iptables only – do not reconfigure the underlying software.

The firewall has been copied into the directory `/root/firewall/` along with a script called `extingui.sh` to "put out the fire" and clear all the rules in case you make a mistake. The firewall is not enabled by default – to enforce the rules, execute:

```
/root/firewall/firewall.sh
```

... as the `root` user or using `sudo`:

```
sudo /root/firewall/firewall.sh
```

This will load the rules and start enforcing them. To make sure that you are removing all `iptables` rules, you should run `extingui.sh` in between every invocation of `firewall.sh` or rules might "stick around" which can be very confusing if you are trying to debug the system. This can be done like this as `root` (or with `sudo` as above):

`/root/firewall/extingui.sh`

If you make the server inaccessible with broken rules, don't worry – you can reboot the node in the SPHERE console, and since the firewall is not enabled by default, you can log in in order to fix it. Your files will still be on the experimental node as long as you don't swap out the experiment. (Of course, you can permanently save your files in your home directory.)

Finally, only your final product is evaluated – not the number of times you have to reboot the server. You should expect to lock yourself out a few times. :-)

NOTE: Your experimental nodes have at least two networks. The first is a control network between your node and SPHERE. This is the network you use to connect to your nodes from your `XDC`. Your nodes also have an "experimental" network that connects all the machines in a given experiment. For this lab, your experimental network connects `server` and `client`.

The firewall only needs to limit the experimental network interface and should not ever limit the control network or you may cut yourself off from the node. The experimental interface is one of eth0-ethN and you can determine which using the command ifconfig. Be warned that the specific interface used may change with a reboot or different experimental node.

Look for instructions on using environment variables to define the experimental interface in your `firewall.sh` script.

Here's what the firewall needs to do:

1. Passively ignore any traffic inbound to the interface that says it's coming from the server itself (obvious spoof attempt). The server uses the localhost loopback device lo for internal traffic, so it should never see incoming traffic from its own IP address on the experimental network interface. (See test case 11 in HTML file)
2. Allow all established traffic on the experimental network interface. Established or related traffic is traffic that is part of previously accepted new connections.
3. Accept new connections on the experimental network of the types listed below:
   (a) Inbound TCP connections to the OpenSSH, Apache, and MySQL servers on their standard ports. (Test cases 1, 3, 5)
      - The MySQL server should only accept connections from the client host.

   (b) Inbound UDP connections to the server ports 10000 to 10005 from the host client. (Test case 8)
   (c) Inbound ICMP ping requests. (Test case 6)
   (d) Outbound ICMP ping replies. (Test case 7)
   (e) Outbound TCP connections to any OpenSSH, SMTP, and Apache (on standard ports). (Test cases 2, 4)
   (f) Outbound UDP connections to the ports 10006 to 10010 on host client from the server. (Test case 9)
4. Passively ignore all other traffic. (Do not allow it or respond to it in any way.) (Test case 10)

There are many online resources and tutorials for iptables configuration – feel free to use them. Be aware, however, not all tutorials emphasize the *principle of least privilege* and may give you overly permissive advice! In order to properly configure the firewall, you must consider the basic ways the firewall can differentiate traffic and allow only the specific types you require to properly function.

4.3 Variation

Another system administrator suggests the use of the packet filter `pf` on OpenBSD instead of `iptables`. Implement rules 3c and 3d above (Test cases 6 and 7) in a `pf` ruleset, e.g. by installing OpenBSD in a virtual machine guest and testing the rules. Also consider the additional resources provided. Note that OpenBSD does not run on SPHERE.

5 Word Problems

1. Where can the rulesets fail?
2. How would you prioritize/choose the approaches for your environment (pf vs. iptables), if you needed to make a choice between them?
3. Do the firewall logs help find potential intrusions?
4. Why do some firewalls keep state?

6 Deliverables

1. A report describing all your findings above.
2. A zip file containing:
   - The file from the `submit.sh` script.
   - The `pf` rulesets.
   - Answers to all word problems in Part 5.
3. Submit by April 4, 2025.

7 Grading

Points will be subtracted if any of the pieces of the deliverables are missing or incomplete. The late submission policy applies.