|  |
| --- |
|  |
| Capstone Experience  IST 894 |
| Lab 1 – Linux Networking and Command Line Tools |

Table of Contents

[1. General Context 3](#_Toc82266792)

[2. Technical Context 4](#_Toc82266793)

[3. Solution 6](#_Toc82266794)

[References 9](#_Toc82266795)

# General Context

In this lab we take a close look at several tools that exist natively in Linux, Unix, and Unix-like (also called \*nix) environments. Using the command line in \*nix environments allows you to do power actions like redirect text that is being printed to the screen, which is called standard out, and have another tool process it and take one or a series of actions. There’s virtually no limit to the amount of processing you can do automatically when scripted together.

We start off using a tool called Nmap (Nmap: The Network Mapper - Free Security Scanner, n.d.), which is free and open-source network scanner created by Gordon Lyon in 1997. Nmap to is a powerful command line tool that allows us to find, identify, and analyze hosts on a network by sending various packets to a host and analyzing the response received. Once we identify the IP address and subnet that our machine is on, we see that there are 4094 usable addresses so to manually try to find other hosts would be painstaking and difficult. Instead we can use Nmap to find hosts that respond to our traffic, and then pipe or redirect the Nmap scan results to additional tools that look for specific text patterns and can then manipulate the data in to different formats to be processed further. This could be sending the processed data in an automated email, or passing the IP addresses along to another script or program for further investigation or even automated blocking.

This was a great example of a very small selection of great tools that are available on \*nix environments, and the basic concepts of how to tie various processes together combining many different single purpose tools to complete a complex action. Each \*nix operation system distribution has a package manager that allows you to search for any number of tools and download and install in a few keystrokes. It’s also made fairly simple to create your own tools to be made available for the public at large.

# Technical Context

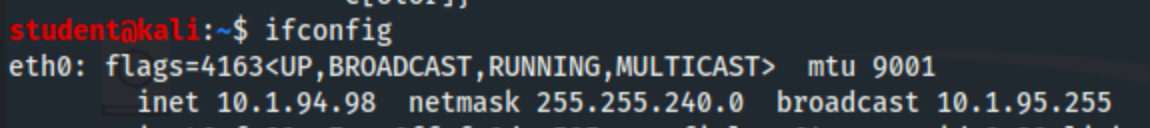
In this lab we use a Kali Linux virtual machine, which is a distribution based on Debian. So it means that most packages that Kali uses are directly imported from Debian’s package repositories (Kali’s Relationship With Debian | Kali Linux Documentation, n.d.). It also means that any additional packages that we need can easily be installed from the Debian repositories. We are able to perform all action in this lab using packages that are installed to Kali by default. The goal of this lab is to identify our IP address and subnet, scan that network for alive hosts, and then take those results manipulate the way the data is presented and automatically email it onward or be prepared to take further action.

Kali, and \*nix in general, gives us many tools that can all accomplish the same basic thing. For example, we can use the `hostname`, `ifconfig`, and/or `ip` commands with various flags to all accomplish the same thing of identifying that our IP address is 10.1.9.98 and that we have a subnet mask of 255.255.240.0 which can be displayed in CIDR notation as a /20. This means we are 1 address in a subnet of 4064. We use Nmap to scan the other 4063 IP addresses looking for other alive hosts, or at least hosts that respond to the crafted Nmap packets (Target Specification | Nmap Network Scanning, n.d.). We can take the Nmap response output and use `grep` to select and show only lines that match a certain patterns, use `cut` to pass a delimiter and extract values from specific set columns, and finally take those specific, extracted values and automatically email them to a mailbox that uses environment variables and `date` to create a subject that says what host it’s from and when. Alternatively, you can take those extracted values, and pass them to scripts that can perform other automated actions.

This is a great way to show how to automatically do large complex actions on a large network. In the security world, we have to perform certain actions on a regular interval. For example, using the concepts that we went over in this lab, we can use `cron` (“Cron,” 2021) to run jobs at specific times or intervals, this is useful to regularly scan the entire network for alive hosts, and send those active IP addresses to a script to automatically perform vulnerability scans or trigger a script to log in and back them up. The view of one of my first boss as a security professional was that “If you’re not automating yourself out of a job every 6 months, you’re doing something wrong.” His theory was because there is a constantly increasing number of threats to protect your network from, you can’t possibly do it all manually. Using these types of processes and chaining \*nix commands together allows you to identify IP’s or host’s of interest where you can focus your attention.

# Solution

The first step of this lab was to identify our host’s IP address. In Kali we can use several tools to do this like `ifconfig` which displays information about all network adapters on the host or something more sepcifc like `ip addr show eth0` which shows the IP address specifically for eth0. We can see in both of the below screenshots that either command will show that our IP is 10.1.94.98 with a subnet mask of 255.255.240.0 which is /20 in CIDR notation. This means that we are in a network that goes from 10.1.80.0 to 10.1.95.255 and has 4064 assignable IP addresses.

Text

Description automatically generated

We then use Nmap with the -sP flag with a command of `nmap -sP 10.1.94.98/20` which send a ICMP ping to each IP address in the specified subnet and outputs any host that responds to the ping. We can either have Nmap output direct to standard out on the command line, or we can write the results directly to a file which we can use `cat` to read. The two different scans took between 70 and 95 seconds to scan the 4064 hosts, and return the 4 hosts that it found responded to the ping. We can pipe the results from standard out to `grep “^Nmap Scan”` to display only lines that start with ‘Nmap scan’, and then to `cut` where we set a delimiter of a space, and select the second field which is just the IP address.

Text

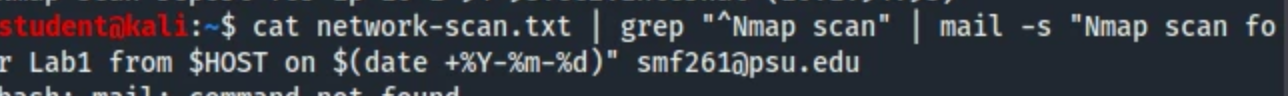
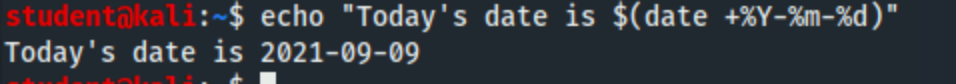
Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generated

We can also use `echo` to print out the results from other commands inside of a string. If we type `echo "Today's date is $(date +%Y-%m-%d)"` it calls the `date` command which uses the formatter of +%Y-%m-%d to show the current date in YYYY-MM-DD format. So running that command returns “Today’s date is 2021-09-09”, we can utilize this same concept to pipe the extracted Nmap results to craft an outbound email using this `date` command and $HOST which is an environment variable for the host name. This would allow us to run this command on various different machines, and know exactly what hostname it came from and what day. What makes this even more powerful is this can all be done directly in standard out, which means it’s running in memory and doesn’t require the ability to write to or read from a file. We know that $HOST is an environment variable, but if you run `set` you can see a full listing of all variables running on the machine. We can even pipe that “set” output to `egrep` which is just an extended version of grep that handles regular expressions automatically, so if we run `set | egrep -i ‘https[s]?\_proxy|all\_proxy’` it will do a case insensitive match of http and then match 0 or 1 ‘s’ followed by ‘\_proxy’, or just the text ‘all\_proxy’. Regular expressions can be extremely powerful for pattern matching when trying to find complex data matches.



Graphical user interface, text, application, chat or text message

Description automatically generated

We can add “-oG -“ to our Nmap command which outputs the Nmap results in a more ‘greppable’ format. The raw output has several extra lines that aren’t useful or actionable data, so if we pipe the return to `grep ‘^Host’` it only displays the lines that start with the word “Host”. Finally, we pipe those grepped lines to `cut -d’ ‘ -f2` which breaks the lines into columns that are delimited by a space character, and prints only the second field, which happens to be the only the IP address. With a list of plain extracted IP addresses we can send them to a script that can run scans, or actions on each of those IP addresses.

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated

# ****References****

Cron. (2021). In *Wikipedia*. <https://en.wikipedia.org/w/index.php?title=Cron&oldid=1043666184>

*Kali’s Relationship With Debian | Kali Linux Documentation*. (n.d.). Kali Linux. Retrieved September 9, 2021, from <https://www.kali.org/docs/policy/kali-linux-relationship-with-debian/>

*Nmap: The Network Mapper—Free Security Scanner*. (n.d.). Retrieved September 9, 2021, from <https://nmap.org/>

*Target Specification | Nmap Network Scanning*. (n.d.). Retrieved September 9, 2021, from <https://nmap.org/book/man-target-specification.html>