Joel Lantigua

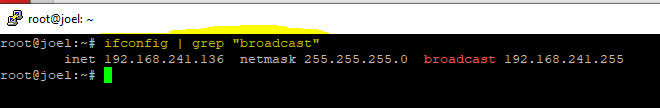
Linux Administrator

Network Script

10/14/22

Networking Script

1. We are going to be making a script that allows you to dump network information onto the command line and it will also save that information into a text file. The first thing I made sure of is that all the commands I was going to be using was installed and functioning. The commands that I will be using in this script will be **Ifconfig, nmap, nano, and ping**. Since nmap wasn’t installed on my ubuntu virtual machine I used the command **sudo apt-get install nmap.** For the CentOS server the commands are slightly different I used **sudo yum install nmap.** This allowed nmap to be installed.
2. After I’ve checked that the following commands were installed. I then was ready to connect to my virtual machine remotely using putty. Once I was connected to both virtual machines remotely using putty, I then went into my **Ubuntu remote window** and used the command **sudo – root** to change into root.
3. After I changed into root I then used the command **nano** to create a script file called netscript in my root home directory. As soon as you use the nano command followed by the name of the script you picked it will create it and open it up. Once nano has created and opened the script file, we are going to start by adding the path to bash at the top of the page, which is **#!/usr/bin/bash**. This path will allow you to run the script through a bash session.
4. After the bash path is inserted, then we are ready to start adding our network information commands into the script. The first command I started with was **ifconfig**. Ifconfig is used to see the network interface and to also configure it. now for the ifconfig we do not need all the extra information shown we only want the command to grab the first three octets from the Ip address in my case it’s (192.168.241). To do so we need to use the **ifconfig** and pipe it using **grep** followed by the name “broadcast”. This will search for anything in ifconfig that has the word broadcast in it as shown below.



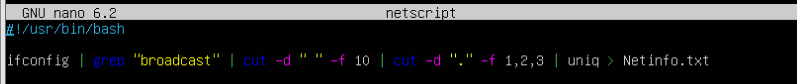
1. Now we still need to display only the first three octets of the Ip address, so we are going to use **ifconfig | grep "broadcast"** and pipe this command using **cut**, Cut is used to cut out parts from a certain line. In this case I used **ifconfig** | **grep “broadcast” | cut -d " " -f 10**. As you can see below it just printed out the IP address meaning we are still not done.

 since we still want the first three octets, so we are going to take the above command and pipe it again and use **cut -d “.” -f 1,2,3 | uniq.** What this part does is that it takes the Ip address and just reads out the first three octets which is represented by the “**-f 1,2,3”** in the command after we pipe it again and use the command **uniq.** Uniq is used to filter out repeated lines. As you can see below it just printed out the first three octets in the IP address.

Graphical user interface, text

Description automatically generated

1. Now that we have figured out the Ifconfig command the prints only the first three octets of the IP, let’s insert it into the script file under the bash path. At the end of the command you want to add **>** and the name of a text file you want this information to be saved in. In my case I used Netinfo.txt, so this would put the first three octets we found into a text file and save it.



1. Next, we want to give the text file we created for the Ifconfig command a variable. To do this we are start with a variable name, I used **Netinfo**, after you select a name for your variable you are going to use **=$(cat Netinfo.txt).** this is useful because you can just use the new variable text file for the remainder of the other commands. After the variable is created, we want it to print out and display that information. To do this we are going to use the **echo** command. So we are going to **echo “” > $Netinfo.txt**, this means to echo the blank space to the information stored in **$Netinfo.txt** allowing you to display the three octets in the IP address.

Text

Description automatically generated

1. Next, we are going to create a ping loop for our script that displays any addresses that begins in 192.168.241 being pinged to our server. We are going to use the **for** command to create our loop. The command **For** is used to repeatedly execute a set of command. In this case we want to use **for ip in {1..254}** next you want to click enter and on the next line insert **do**, click enter again then hit the tab key to indent. Next, we are going to add our ping command to our loop. Where you inserted the indent type **ping -c 1 $Netinfo.$ip** **| grep “64 bytes” | cut -d “ “ -f 4 | tr -d “:”** **>> $Netinfo.txt &.** Basically, this command takes any Ip address that starts with 19.168.241 that are pinging to the server. Next it will grep anything that starts with “64 bytes”. Now we have to cut it so that only the IP addresses that are pining is displayed by using the **cut** command. After we use the cut command we are going to pipe **tr -d “:”.** This command means to delete anything after the **“ : “** , then we are going to append this information to **$Netinfo.txt**  **&.** After hit enter and insert **done** to end the loop

A screenshot of a computer

Description automatically generated with medium confidence

1. Next, we are going to display our results to the ping loop by using the command **cat.** So, we are going to **cat $Netinfo.txt | sort > ping\_$Netinfo.txt.** we took all the Ip addresses that were pinged from $Net info.txt and sorted it into a new file called **Ping\_$Netinfo.txt.** The **sort** command is used to sort a files record in a certain order.



1. Now we are going to **nmap** all the pinged IP addresses simultaneously. To do so we are going to **nmap -sS -iL ping\_$Netinfo.txt.** The **-sS** is a TCP SYN port scan and the **-iL** allows me to scan the **Ping\_$Netinfo.txt** file which contains any Ip address that pinged my server.A screenshot of a computer

   Description automatically generated with medium confidence
2. For this last part we are going to add the command to find the DNS address and print the result. We are going to start with **less /etc/resolv.conf ,** this command shows you the servers dns. Next we are going to add the result of “**less /etc/resolv.conf”** into a file called dns.txt by using **less /etc/resolv.conf > DNS .txt**. now we want to display the result by using **cat DNS.txt.** Next hit enter and type **exit.**

Text

Description automatically generated

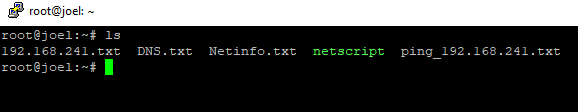
1. Next you going to want to hit **ctrl s** to save the file and then hit **enter** to continue. Now we have to give the script we just created executable permission by using the command **chmod +x netscript.** To see if the permission worked us **ls** in the directory the file is in, the name of the script should light up a different color. To run the file simple use **./netscript**  as you can see below my script prints its result and also inserts the results into text files.

Text

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence



As you can see below these are the files that my script created. In the Ping\_192.168.241.txt you can see the addresses that pinged my ubuntu server. My ubuntu servers Ip address is 192.168.241.136

Text

Description automatically generated

**Sending the script to my CentOS server**

1. Now we are going to send a copy of the script we created on my Ubuntu server to my CentOS server and run that same script on the CentOS server. To send my script over to my CentOS server I used the command **scp – r /root/netscript root@joel root@192.168.241.146:/root/received.** This command allowed me to send the script to the **received directory** in my CentOS server.

Text

Description automatically generated

1. Now if I open my CentOS remote window and use the command **ls** in my **root/received directory** the script called **netscript** should be in there highlighted green meaning that the script still has its executable permissions.



1. After verifying that the script was successfully sent to my CentOS server. I used the command **./netscript** to run the script as you can see in the screen shot below the script works.

Text

Description automatically generated

Work Cited

<https://cs.lewisu.edu/~klumpra/camssem2015/nmapcheatsheet1.pdf>

<https://www.stationx.net/nmap-cheat-sheet/>

<https://nmap.org/book/synscan.html>

<https://smallbusiness.chron.com/ping-ip-addresses-lan-68381.html>

<https://www.geeksforgeeks.org/cut-command-linux-examples/>

https://www.geeksforgeeks.org/tr-command-in-unix-linux-with-examples/