**Security Script**

**Table of Contents**

Setting up the script File

Adding The Commands and Running Script

**Setting up the Script File**

1. Firstly. We want to create the file that will contain our script. We will call the file security and create it using nano:

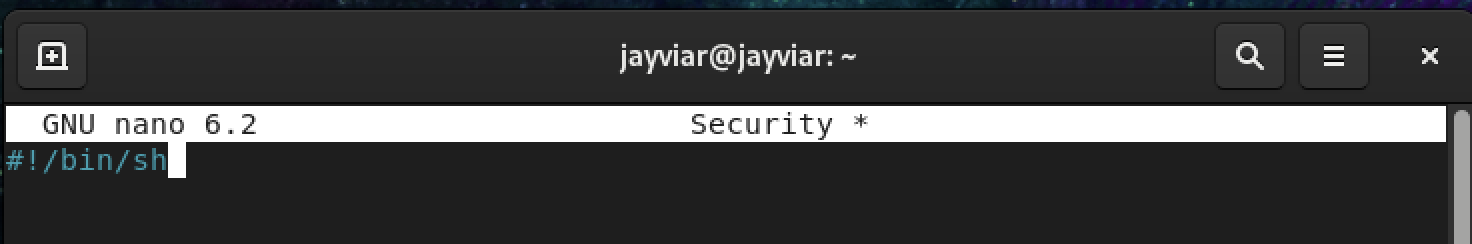
Nano Security

*We now have an empty file.*



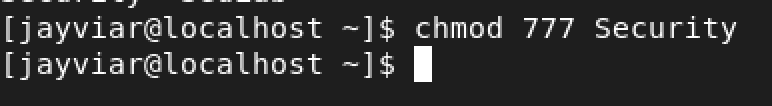
1. Next, every working script must contain a shbang at the top of the file. For this script, ours will looks like this:

#!/bin/sh



1. Now, lets give our file permissions. This can be done using chmod and we’ll just use 777 to give it all for simplicity sake.

Chmod 777 Security



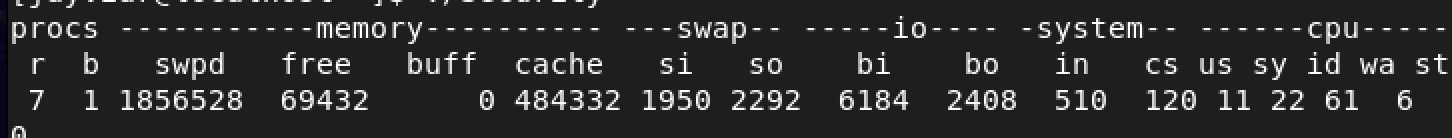
**Adding The Commands and Running Script**

To add the commands. Simply nano back into the file and type them out line by line.

Vmstat

Now, we can add the commands! To start, lets add the vmstat command. What this command does, is display a variety of information about the server. Things like memory, processes, and interrupts.

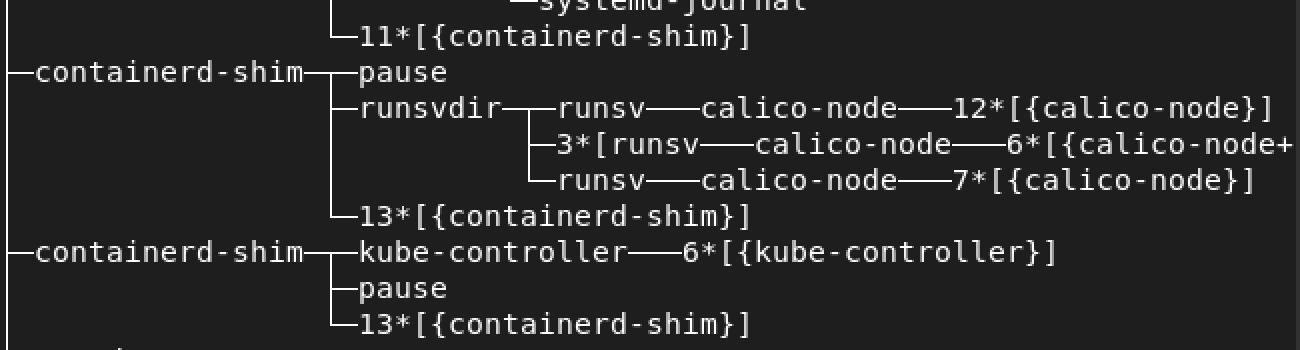
*Snippet command output*



Pstree

The pstree gives us the ability to see running processes on the server but, it is displayed as a tree. Which ultimately means it shows connections across all the processes with lines displaying what its connected to.

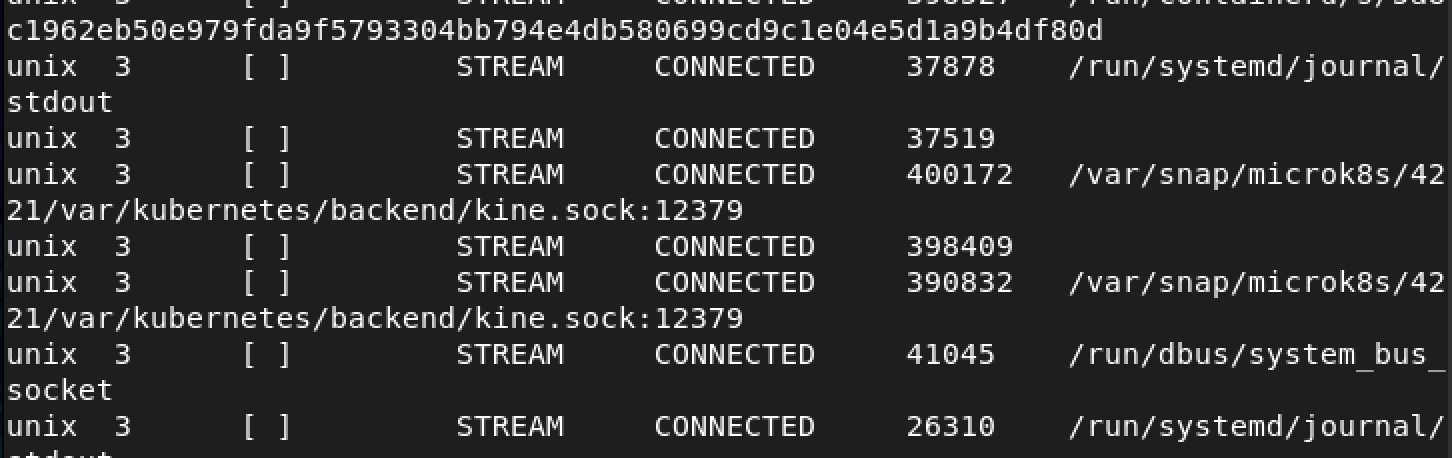
*Snippet of pstrees long output*



netstat

Netstat is what allows us to see where things may be coming and going to and from. It sees incoming and outgoing connections as well as things like routing tables. Which may be helpful to double check once and a while to see if things are being sent and delivered to proper destinations!

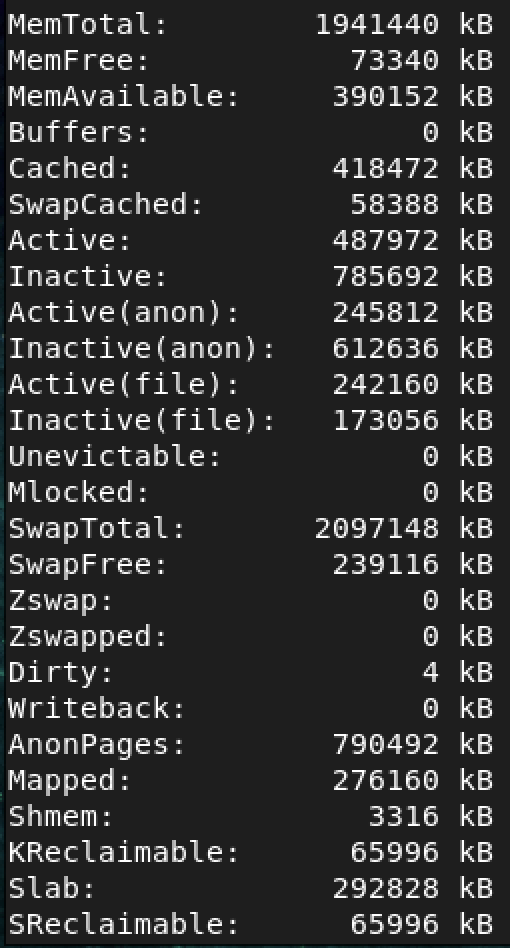
*Small part of netstats connections*



cat /proc/meminfo

This command will allow us to see the information regarding the memory on our machine. It show the total, the amount available, and the amount used as well as much more.

*Snippet of what this command outputs*



After all commands have been added. This is what the script should look like:



1. Now that we have our script. We want to be sure we can save this information to a file in which it will keep the results of the script documented as a sort of reference of what the system looked like during that time. We can do this by adding the command:

Command >> filename

This will send the information of any given command to either an existing filename, or it will create a brand new one with whatever filename is

Entered.

1. For our current script, we will use the command above and give it the filename SecurityResults, We also make sure every command in our script is being sent to this location. Below is what that will look like with all commands added.

Vmstat >> SecurityResults

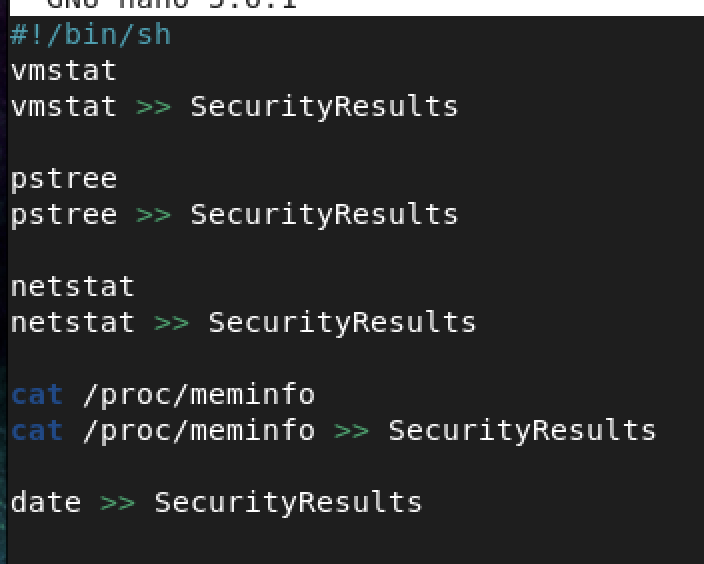
Pstree >> SecurityResults

Netstat >> SecurityResults

Cat /proc/meminfo >> SecurityResults

Date >> SecurityResults

*Here is the commands added along with the already made script*



*Note\* date is optional here however it was added to give anyone accessing the script an idea of when the script was last run.*

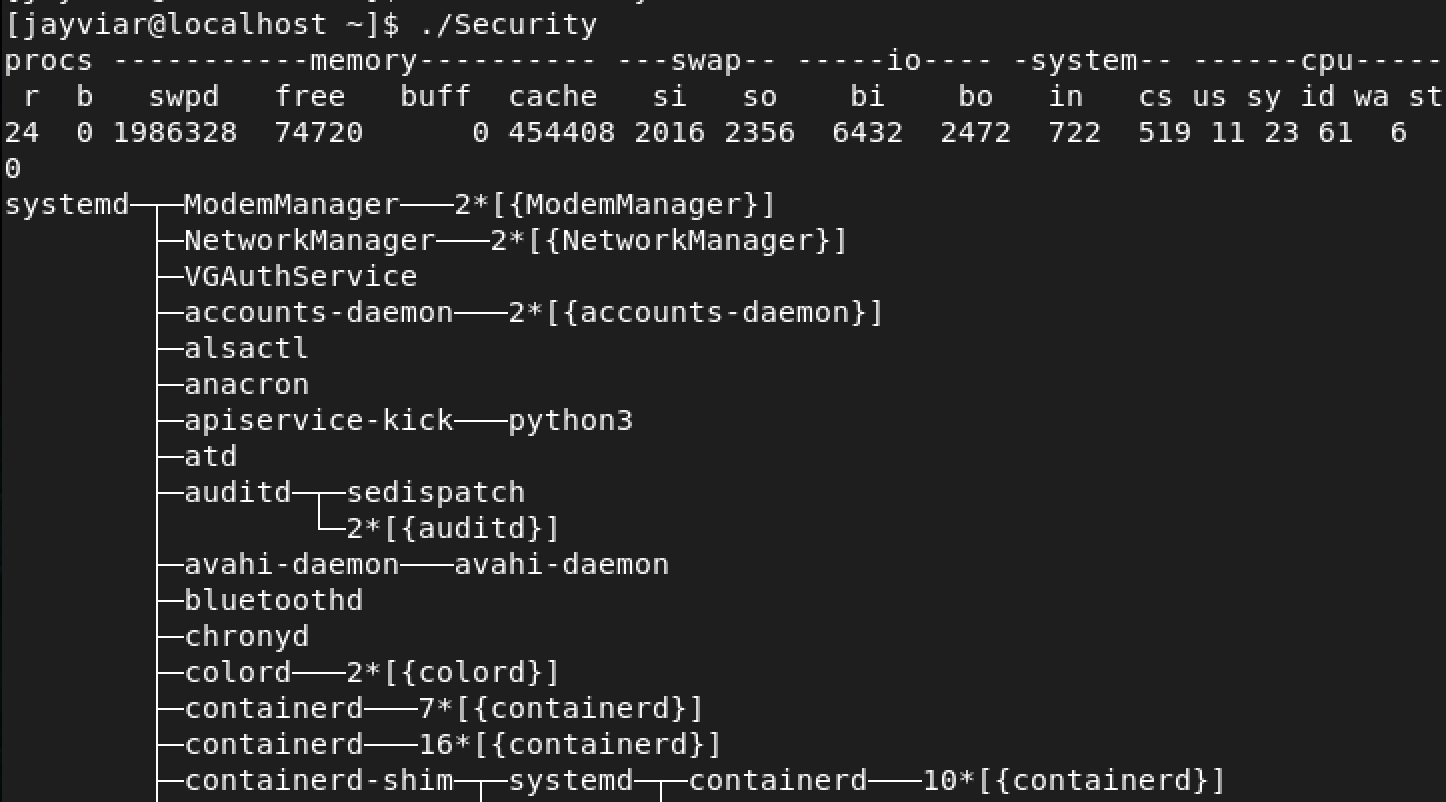
1. To finally run our script. We simply take use; ./ followed by the script name to run everything within the script. After, we will then use cat to look see the contents of our new file with the script results in it.The commands should look like this

./Security

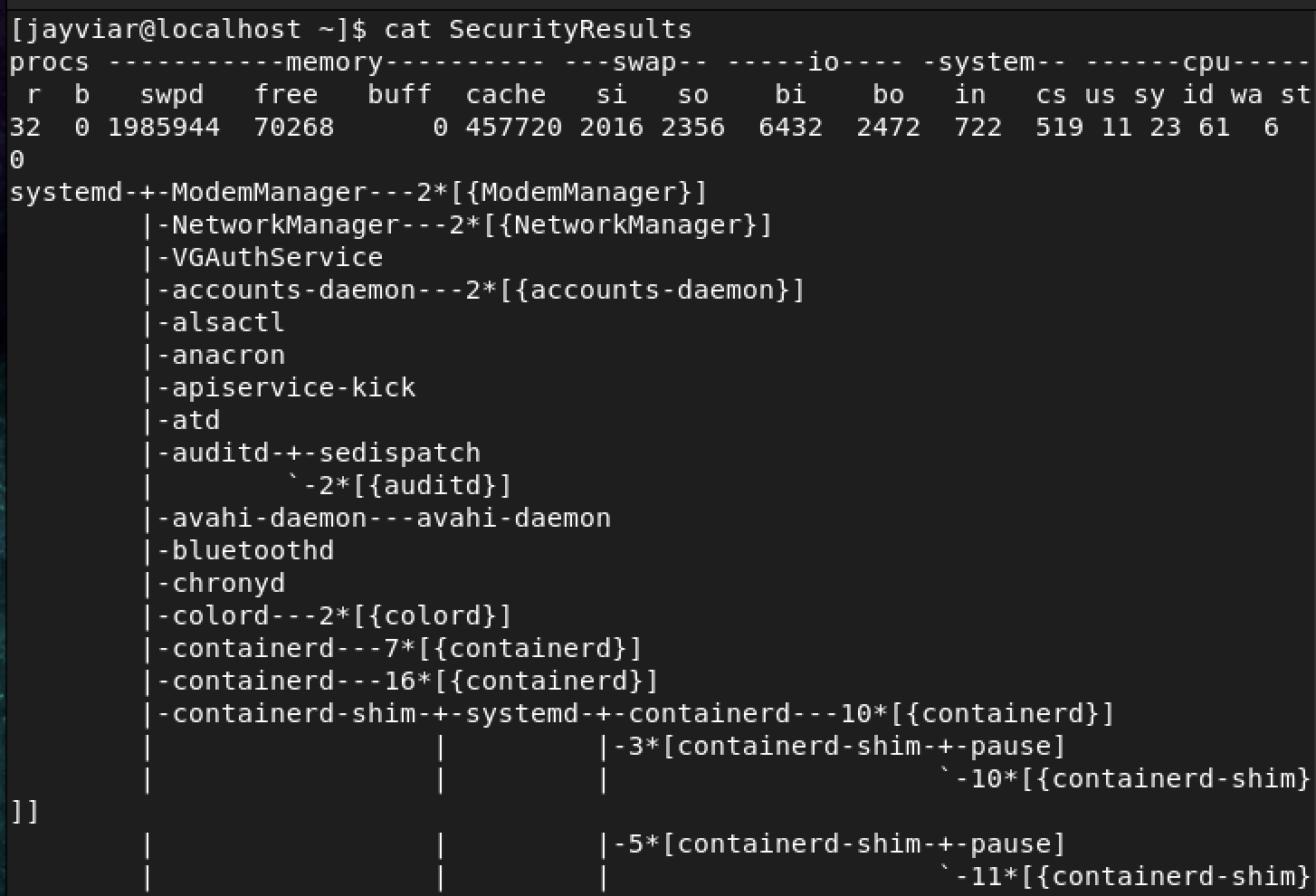
Followed by

Cat SecurityResults

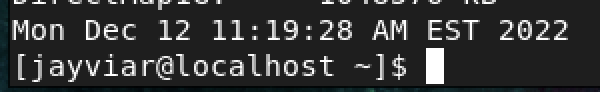
*Using ./Security will allow us to see everything in our script. This is just the first chunk of what is output from the script*



*Here we are using cat to display what is in our new file. It is identical to the command above as they both will display the results once run. The difference is that this file contains ONLY results, no script.*



*Here is the date shown at the bottom of the results file*



* **Note:** This script is identical on our Centos and Ubuntu Server. Both will run the commands the same and there was no need for substitutions or additions on either.