Intro To Linux

Lets Go SUDO:

Sudo Overview:

Not much new information here. The only command that I was not aware of was the #passwd command that prompts the current user to change its password.

Users and Privileges:

```
[demo09@login.crane planets]$ ls -l
total 8
drwxr-xr-x 4 demo09 demo   4 Jun 12  2018 gen
-rwxr-xr-- 1 demo09 demo 115 Jun 12  2018 mars.txt
drwxr-xr-x 2 demo09 demo   8 Jun 12  2018 results
[demo09@login.crane planets]$
```

The first character denotes whether an item is a file or a directory. If 'd' is shown, it's a directory, and if '-' is shown, it's a file. Following the first character you will see some combination of r,w,x, and -. The first rwx is the 'read' 'write' 'execute' file permissions for the creator of that file or directory. A '-' instead means a particular permission has not been granted. For example "rw-" means the 'execute' permission has not been granted. The next three entries are the permissions for 'group' and the last three are the permissions for everyone else.

Following the file permissions are the name of the creator, the name of the group, the size of the file, the date it was created, and finally the name of the file.

owner-user (u), owner-group (g), everyone-else (o).

```
drwxr-xr-x  3 alice users 4096 Aug  1 10:30 Documents
-rw-r--r--  1 alice users   42 Aug  1 09:45 readme.txt
```

Now what the hell is this! Right?

The `ls -la` command in Linux is used to list all files and directories, including hidden ones, in a detailed format. Let's break down the different columns of the output:

Column 1: **Permissions**: The first column represents the permissions of the file or directory. These permissions are depicted using a combination of letters and symbols (e.g., `-rw-r--r--`).

Column 3: **Owner**: The user who owns the file or directory. By default, the user who creates the file or directory is its owner. This user has special privileges to control access and modify the file.

Column 4: **Group**: This column shows the group that the file or directory belongs to. The group plays a role in determining permissions for other users who are part of the same group.

Column 7: **Name**: The final column displays the name of the file or directory. Hidden files and directories, whose names start with a dot (.), are also listed here.

chmod numbers		
Number	Permissions	Totals
0	– – –	0+0+0
1	– – x	0+0+1
2	– w –	0+2+0
3	– w x	0+2+1
4	r – –	4+0+0
5	r – x	4+0+1
6	r w –	4+2+0
7	r w x	4+2+1

We can use #chmod command to give permissions to files

Ex: #chmod +wr

#chmod +r

#chmod 777 This one would give full permissions for all groups. For the owner, the group, and all the rest(all the rest besides owner-user, and owner-group mentioned..)

We can user #sudo adduser user_name to add a new user.

Important commands and files to read:

#cat /etc/passwd

#cat /etc/shadow

#sudo cat /etc/sudoers

#grep 'sudo' /etc/group

#sudo -l to enumarete current user priv for running sudo

Common Network Commands

Commands:

#ip a -list all ip configurations. All connections types (wireless or hardwire).

#ifconfig -only list the hardwire connections.

#iwconfig - only wireless connections.

#ip n -"n" stands for neighbor.

#arp -a - "Address Resolution Protocol". It tells what IP Address is associated with each MAC Address.

#ip r -"r" stands for route.

#route -similar results as the command above.

#ping -off course.

#netstat -Identify services and open ports.

Viewing, Creating, and Editing Files

We can overwrite file content by directing out put to them using the "greater than" sign " > ". And, if we want to append data to this files, we use the "greater than" sign twice " >> ", and the data should be appended to the file.

#touch file_name -creates file with specific name.

#nano -edit/create file

#vi -edit/create file

#vim -edit/create file

#mousepad newFile.txt -same but with editor interface where we can use mouse, and there are other features.

#gedit -same as mousepad

Starting and Stopping Services

Commands:

#sudo service apache2 start -starts apache2 server (HTTP). It could be used to host, and transfer malicious files, but "python3 -m http.server port _number" is better.

#sudo service apache2 stop -stop service/

#python3 -m http.server port_number -http server for malicious files, etc...

#sudo systemctl enable ssh -enables ssh server.

#sudo systemctl disable ssh -disables ssh server.

Installing and Updating Tools

Commands:

#sudo apt update && sudo apt upgrade -update, and upgrade machine to latest version.

#apt install package_name

#git clone path_to_file_location

Install tool: pimpmykali on github.

Scripting with Bash

```
┌──(kali㉿kali)-[~/Desktop/PraticalEthicalKacker/bashScripting]
└─$ cat pingSweep.sh
#!/bin/bash
if [ "$1" == ""]
then
echo "You forgot to provide an IP address!"
echo "Syntax: ./ipsweep.sh 192.168.1"
echo "This ipsweep works for /24 network!"

else
for ip in `seq 1 254`; do
ping -c 1 $1.$ip | grep "64 bytes" | cut -d " " -f 4 | tr -d ":" &
done
fi
```

There is a huge difference from using the "&" symbol at the end of the "ping" command line, and using the semi-colon ";".

The "&" symbol allows kali to run multiple instances of the for loop at a time, and the semi-colon only allows one instance to run at a time. So, the program runs much faster when we use the "&".