Linux Basics for Hackers - Notes

Binaries

Located in -> /usr/bin and/or /usr/sbin

Linux File System

```
'/' -> The actual root system. The top most.
```

Inside it ->

/boot - Kernel image

/home - user dir

/proc - view of internal kernal data

/dev - special device files

/sbin - binaries

/root - SuperUser's Home Dir (different from '/')

/etc - Sys config

/mnt - GenPurpose Mount point

/sys - Kernel's view of HW

/bin - also binaries

/lib - libraries

/usr -> /sbin, /bin, /lib (more of the same stuff) /media - for ejectable media

cd Command

use '..' to move up 1 level

'.. ..' for 2 levels & '.. .. ..' for 3 levels and so on

Search-based commands

$ locate 'find_this' -> finds all occurences

$ whereis 'module_name' -> finds all BINARIES of the target (usually with man pages)

$ which aircrack-ng -> finds the binary file located in the PATH variable of the system

$ find -directory -option -targetExp -> finds literally everything [Eg: find /etc -type f -name apache2] -- altho apache2.* will find all file extensions but first name as apache2

Lastly, grep to filter

$ ps aux | grep apache2 -> will filter from all auxilliary processes containing apache2

'cat' is versatile remember

$ cat file_name -> will spill the file contents.

$ cat > file_name -> will let you write in it BUT WILL REPLACE ALL EXISTING DATA.

$ cat >> file_name -> will actually let you append the text you enter.

Renaming doesn't exist in Linux

So we use

$ mv newfile newfile2 -> to essentially rename the file

Removing a Dir

Only the directory : $ rmdir

When that fails : $ rm -r (recursively delete everything in it)

Text Manipulation

head and tail ->

$ head file_name -> first 10 lines

$ head -n file_name -> n number of lines from the start

$ tail -> for bottom lines (+ specialized with a count)

$ nl path_to_file_or_file_name -> will number all the lines. // cat can and should be clubbed with grep as and when needed. Eg: cat snort.conf | grep output

Sed command - for find and replace ->

$ sed s/search_term/replace_term/g path_to_filename > newfile_name

-> 's/' will find the term, '/g' is for replacing globally. Rest is elementary.

-> Removing the '/g' will only replace the first occurrence.

Adding a number there can limit the number of occurrences to be changed. '/3' will only replace the first 3.

Eg: sed s/mysql/MySQL/g /etc/snort/snort.conf > snort2.conf

$ more file_name -> offers a scroll-able page if the file is to big.

$ less file_name -> "less is more" -- offers a filter to search for the term should you need to -- use the '/' key. Still scroll-able but with better functionality.

Networking

'loopback' addr -- same as 'localhost' = 127.0.0.1

iwconfig - check for wireless adapter info -- good for getting power, the mode [monitor,

managed, promiscuous] etc.

Changing info -> MAC or IP Addr :

$ ifconfig eth0 192.1688.181.155

-> modifies what your router sees to redirect packets

Changing Netmask +/ Broadcast :

$ ifconfig eth0 192.168.181.155 netmask 255.255.0.0 broadcast 192.168.1.255

-> Netmask is the subnet mask -- determines the portion of the IP Addr to the NW and which refers to the host. Here, first 2 octets (16 bits) represent the network and last 2 show the hosts within that network.

-> Broadcast is the addr used to send packets to all hosts on the same network segment. Default (bcz of subnet) would become 192.168.255.255 but here overridden to 192.169.1.155 -- Basically packets sent to this will be broad-casted on that specific sub-network.

MAC Spoofing :

Take down the interface, change the Addr, restart

$ ifconfig eth0 down > ifconfig eth0 hw ether 00:11:22:33:44:55 > ifconfig eth0 up

Assigning new IP via DHCP Server :

Server runs of '*dhcpd*' - the daemon. Requested via '*dhclient*'. Requires a DHCP assigned IP addr. (Note : 'dhclient' is for Debian, different for all other distros.)

$ dhclient eth0

->DHCPDISCOVER req is sent by this command and then receives an offer from the DHCP i.e DHCPOFFER. Now, ifconfig will show a difft IP addr as given by the DHCP Server.

Manipulating DNS :

Use 'dig' :

Directly pass-on the domain and add the 'ns' tag to make the domain as the nameserver itself. 'mx' will fetch the mail-exchange server.

$ dig hackerarise.com ns OR $ dig hackerarise.com mx

-- Some Linux servers use BIND (Berkeley Internet Name Domain) which is just a fancy name for DNS.

Changing your DNS Server :

Edit a file stored in '/etc/resolv.conf'. There you will see the domain, search & nameserver fields. Swap the values here to switch your DNS server.

Other method to do the same (although this cleanly overwrites the file's content) ->

$ echo "nameserver 8.8.8.8" > /etc/resolv.conf

Mapping your own IP Addrs :

'hosts' file located in /etc/hosts. Useful for hijacking a TCP connection on your LAN to direct traffic to a malicious webserver by using a tool like *'dnsspoof'*

Usually this file has a mapping for your localhost only BUT you can map any website to any IP Address. Eg : "192.168.181.131 bankofamerica.com". Decent for local network attacks.

[dnsspoof & Ettercap can be used]

Handling SW Packages

Search for package in local repo:

$ apt-cache search *'keyword'*

-> Eg: $ apt-cache search snort

Install, Remove, Purge, Update, Upgrade :

$ apt-get install --name--

$ apt-get remove --name--

$ sudo apt-get update

$ sudo apt-get upgrade

$ apt-get purge --name--

(purge removes the config as well)

or use a package manager like *'synaptic'* or *'gdebi'* like a normie.

Adding Repos to Sources.list file :

$ mousepad /etc/apt/sources.list -> will open the list

Categories : main (OSS), universe (community maintained OSS), multiverse (SW restricted by copyright), restricted (proprietary device drivers), backports (packages from later releases)

Format : "deb http:// ------ --package_name-- main non-free contrib" etc.

File Directories & Permission

r,w,x -> read, write and execute

Granting ownership :

$ chown *username file-path-name* -> Provides the ownership of that file to that user

$ chgrp *group-name package-or-module-name* -> Provides a user-group access to that module

Checking Permissions :

$ ls -l *file-or-path-to-it* -> will lay down the whole sheet. The type, permission on the file for

owner/groups/users, number of links, the owner, size in bytes, creation/mod date & its name.

Eg: "drwxr-xr-x" vs "-rw-r--r--". First letter denotes directory if 'd' or file if empty dash. Followed by the permission values for 3 groups i.e owner then group then other_users. Hence we observe 3 values at a time. Dash means no permission ofc.

In '-rw-r--r--' -> File, owner has read/write, group and other users only have read permissions.

2nd way : There is a proper calculation done in Octal terms as well.

```
001 : 1 : --x
010 : 2 : -w-
011 : 3 : -wx
100 : 4 : r--
101 : 5 : r-x
110 : 6 : rw-
111 : 7 : rwx
```

Total RWX is 7. Since we have 3 sets of permissions, giving a full read+write+execute permission to everyone, for example, would look like ->

$ chmod 777 hashcat.hcstat

3rd way : UGO Syntax

Here, '-' removes a permission, '+' adds and '=' sets a permission.

Eg: Remove the write (w) permission from user on a file

$ chmod u-w hashcat.hcstat -> Now -rw-r--r-- becomes -r-xr-xr--

Or for user and other users at once

$ chmod u+x, o+x hashcat.hcstat

Now, you can set execute permission for yourself on a newly downloaded tool/script bcz by default Linux won't set it

$ chmod 766 some_new_tool -> grants us (the owner) all permission including execute -- and everyone else only R/W permissions.