

LEWIS KAMAU NDERU

[lewiskamau@gmail.com](mailto:lewiskamau@gmail.com) / [kamau2325@yahoo.com](mailto:kamau2325@yahoo.com)

CS-SA10-25029

Link: <https://academy.hackthebox.com/module/details/18>

## LINUX FUNDAMENTALS HTB MODULE

### INTRODUCTION

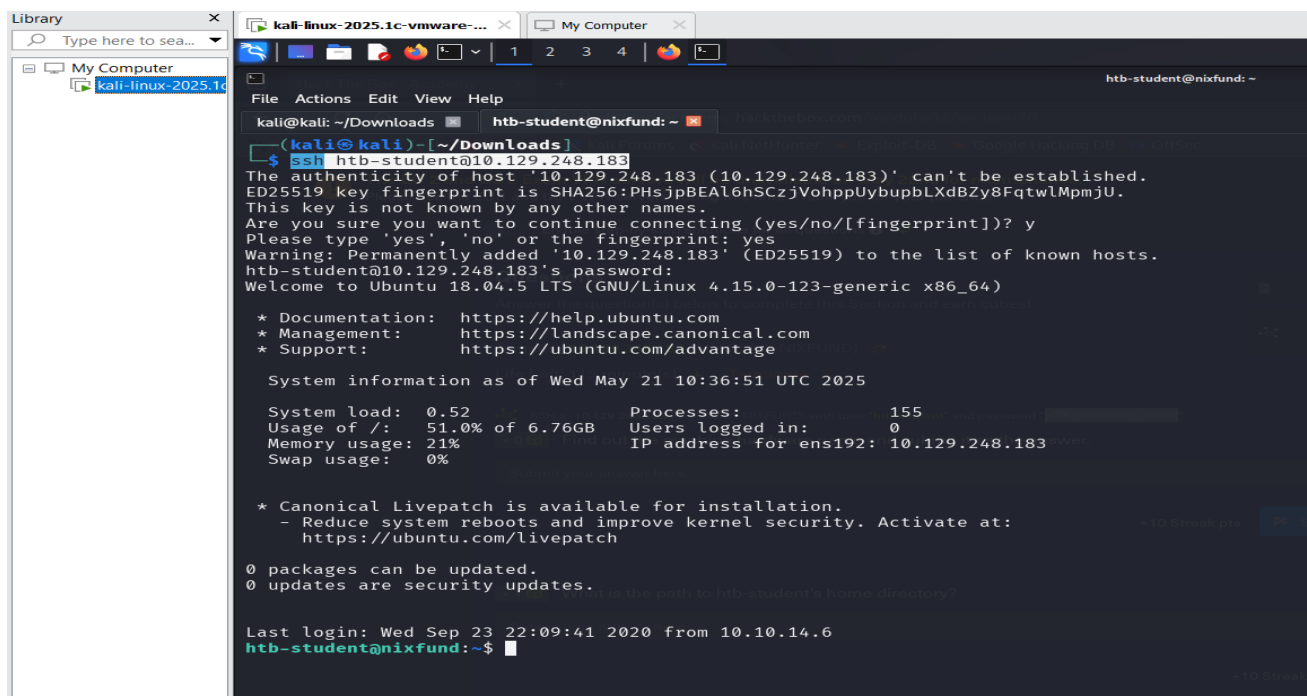
This report explores the Linux Fundamentals module on Hack The Box Academy, which is designed to equip learners with essential skills for working in a Linux environment. From navigating the command line to managing users, permissions, and services, the module provides a hands-on, beginner-friendly approach to understanding how Linux systems work.

Throughout this report, I will be documenting the step-by-step process I followed while completing the practical exercises, providing both a technical overview and a personal reflection of my learning journey.

### ANSWERS TO LABS

#### TOPIC 1: System Information:

To tackle this module, I first downloaded the openvpn file then connected to the HTB network and logged in to the target system via ssh using the command `ssh htb-student@10.129.248.183` then provided the password to the machine and the connection was successful as shown below:



```
kali@kali: ~/Downloads
$ ssh htb-student@10.129.248.183
The authenticity of host '10.129.248.183 (10.129.248.183)' can't be established.
ED25519 key fingerprint is SHA256:PHsJpBEAL6hSCzjVohppUybupbLXdBZy8FqtlMpmjU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '10.129.248.183' (ED25519) to the list of known hosts.
htb-student@10.129.248.183's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-123-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Wed May 21 10:36:51 UTC 2025

System load:  0.52               Processes:    155
Usage of /:   51.0% of 6.76GB     Users logged in: 0
Memory usage: 21%               IP address for ens192: 10.129.248.183
Swap usage:   0%

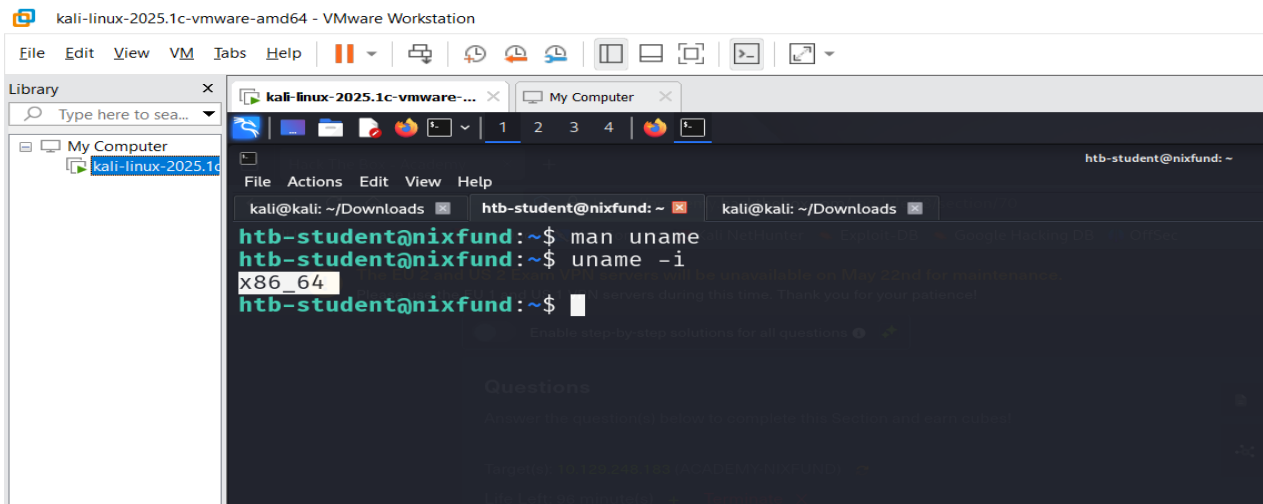
 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

0 packages can be updated.
0 updates are security updates.

Last login: Wed Sep 23 22:09:41 2020 from 10.10.14.6
htb-student@nixfund:~$
```

Our first task is to find out the machine hardware name and submit it as the answer. And to do this, we will use the **uname** command as it provides basic information about the Operating System name and system hardware.

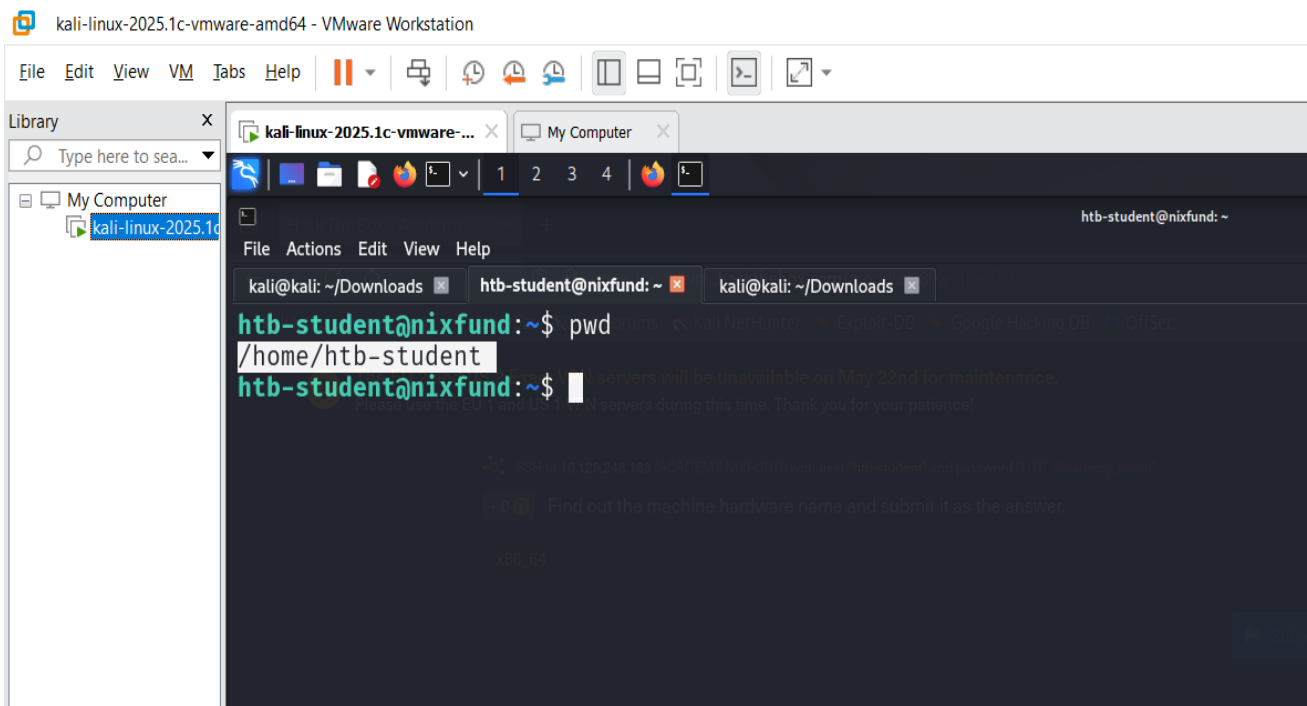
Used the man page for uname to get more information about its usage and the suitable command to get machine hardware name was **uname -i** to get the hardware platform and it is **x86\_64**



```
kali-linux-2025.1c-vmware-amd64 - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to sea...
My Computer
kali-linux-2025.1c
kali@kali: ~/Downloads
htb-student@nixfund: ~
htb-student@nixfund:~$ man uname
htb-student@nixfund:~$ uname -i
x86_64
htb-student@nixfund:~$
```

What is the path to htb-student's home directory?

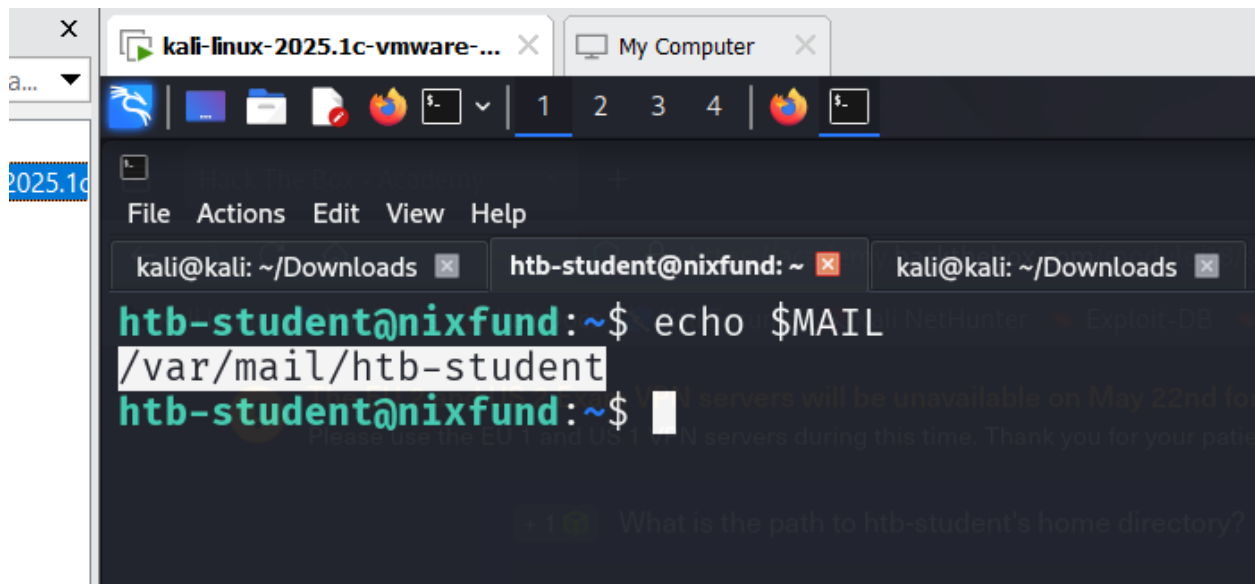
Used the **pwd** command to print the htb-student's working directory and the path was **/home/htb-student** as shown below:



```
kali-linux-2025.1c-vmware-amd64 - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to sea...
My Computer
kali-linux-2025.1c
kali@kali: ~/Downloads
htb-student@nixfund: ~
htb-student@nixfund:~$ pwd
/home/htb-student
htb-student@nixfund:~$
```

What is the path to the htb-student's mail?

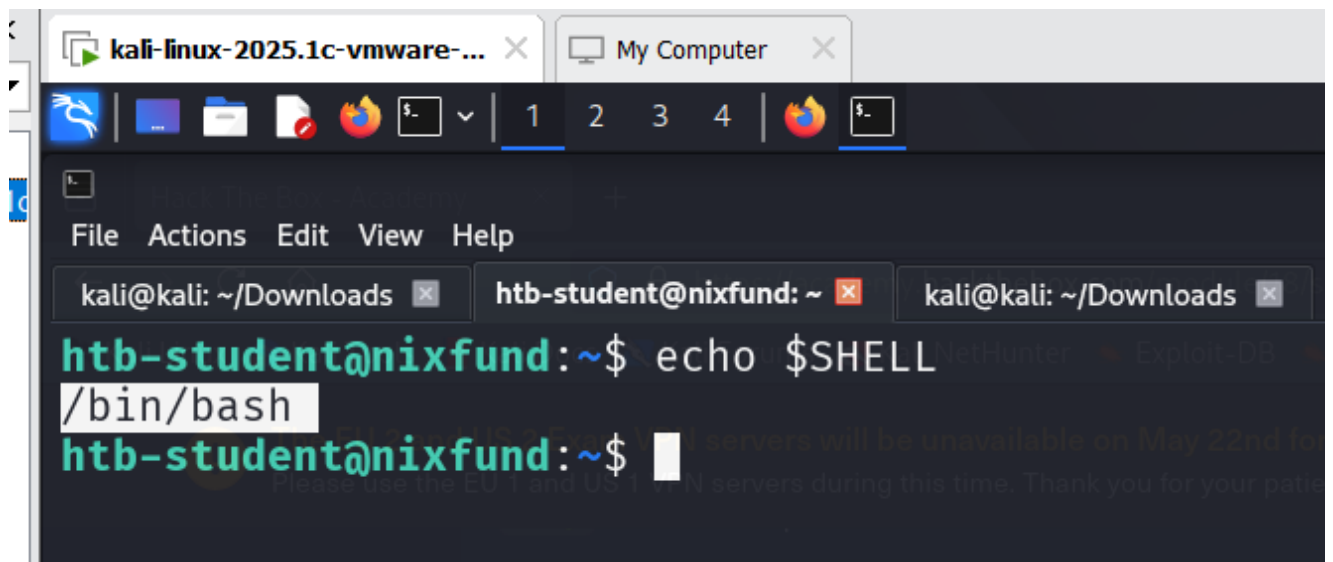
I first did a bit of research and found the command `echo $MAIL` which is an environment variable that typically holds the path to the user's mailbox file.



The screenshot shows a terminal window with three tabs: 'kali-linux-2025.1c-vmware-...', 'My Computer', and 'kali@kali: ~/Downloads'. The active tab is 'htb-student@nixfund: ~'. The terminal shows the command `echo $MAIL` being executed, and the output is `/var/mail/htb-student`. The terminal also shows a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'.

Which shell is specified for the htb-student user?

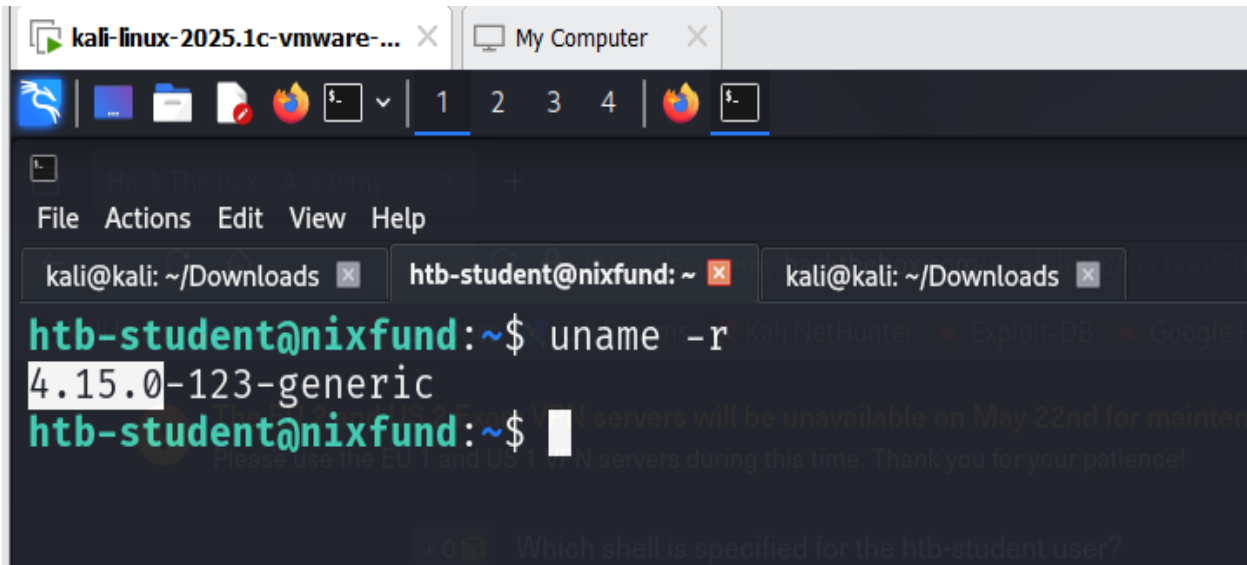
Used the command `echo $SHELL` to check the shell environment for the current session.



The screenshot shows a terminal window with three tabs: 'kali-linux-2025.1c-vmware-...', 'My Computer', and 'kali@kali: ~/Downloads'. The active tab is 'htb-student@nixfund: ~'. The terminal shows the command `echo $SHELL` being executed, and the output is `/bin/bash`. The terminal also shows a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'.

Which kernel release is installed on the system?

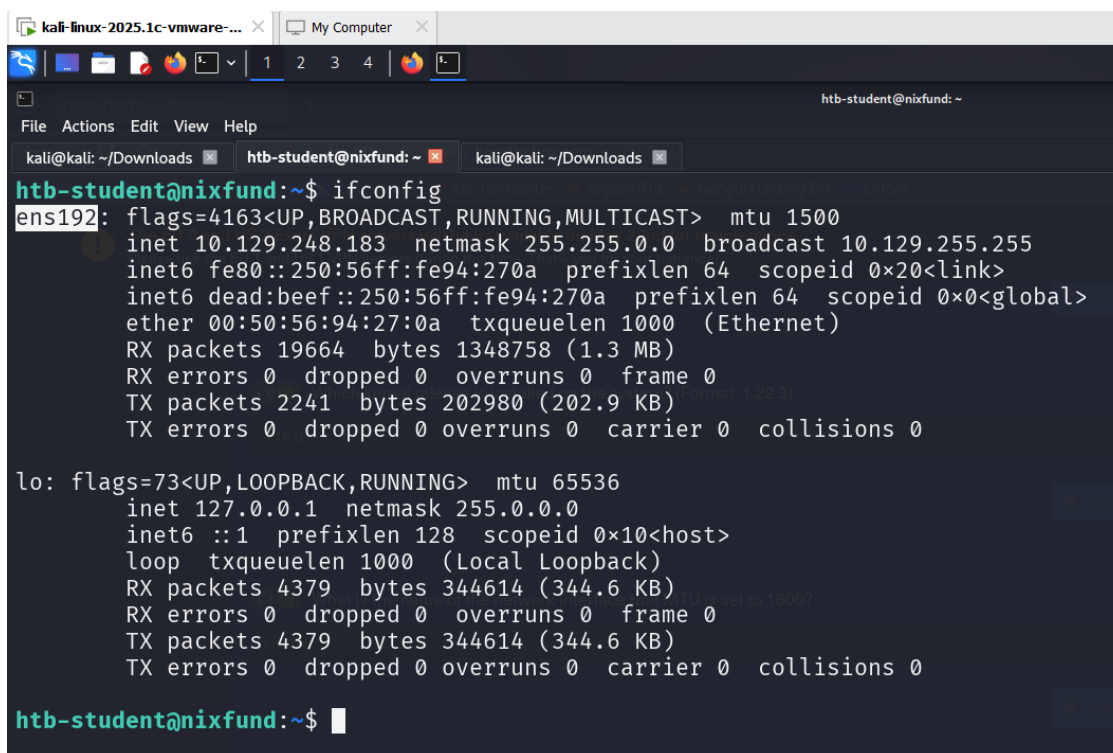
To tackle this, I used the command **uname -r** as it is used to display the kernel release, as seen from the uname man page. The release is highlighted in the snippet below.



```
kali@kali: ~/Downloads x htb-student@nixfund: ~ x kali@kali: ~/Downloads x
htb-student@nixfund:~$ uname -r
4.15.0-123-generic
htb-student@nixfund:~$
```

What is the name of the network interface that MTU is set to 1500?

I used the **ifconfig** command to list the network interfaces and find out the one that the Maximum Transmission Unit is set to 1500 and it was ens192



```
kali@kali: ~/Downloads x htb-student@nixfund: ~ x kali@kali: ~/Downloads x
htb-student@nixfund:~$ ifconfig
ens192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.129.248.183 netmask 255.255.0.0 broadcast 10.129.255.255
    inet6 fe80::250:56ff:fe94:270a prefixlen 64 scopeid 0x20<link>
    inet6 dead:beef::250:56ff:fe94:270a prefixlen 64 scopeid 0x0<global>
    ether 00:50:56:94:27:0a txqueuelen 1000 (Ethernet)
    RX packets 19664 bytes 1348758 (1.3 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2241 bytes 202980 (202.9 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 4379 bytes 344614 (344.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4379 bytes 344614 (344.6 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

htb-student@nixfund:~$
```

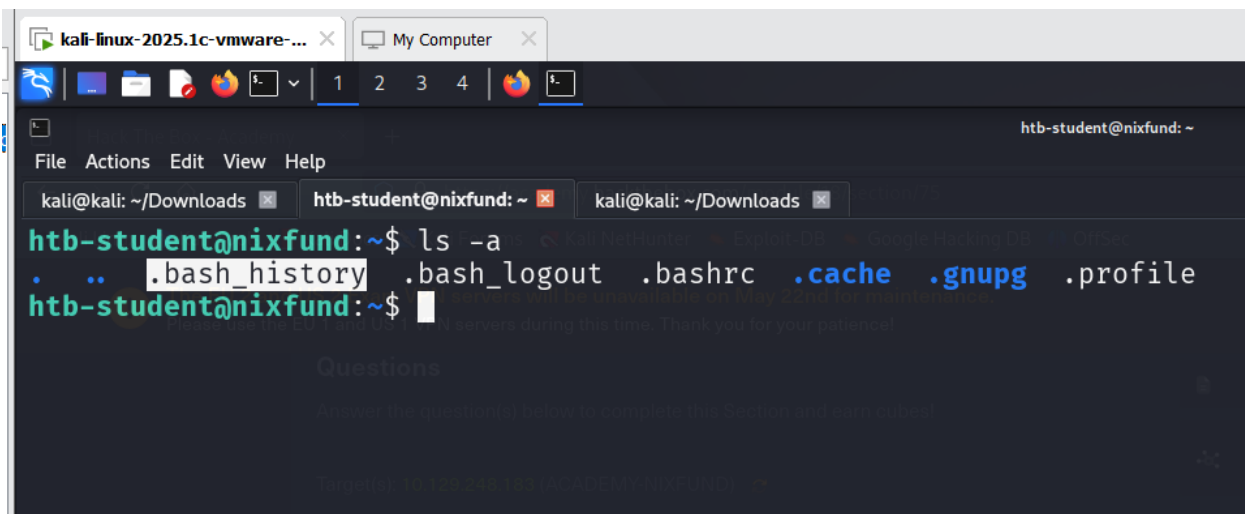
## TOPIC 2: Navigation:

This section introduced basic Linux navigation using the terminal. I learned how to check my current location, list files (including hidden ones), and move between directories using simple commands. Helpful features like tab auto-completion and command history made the process smoother.

### Answers

What is the name of the hidden "history" file in the htb-user's home directory?

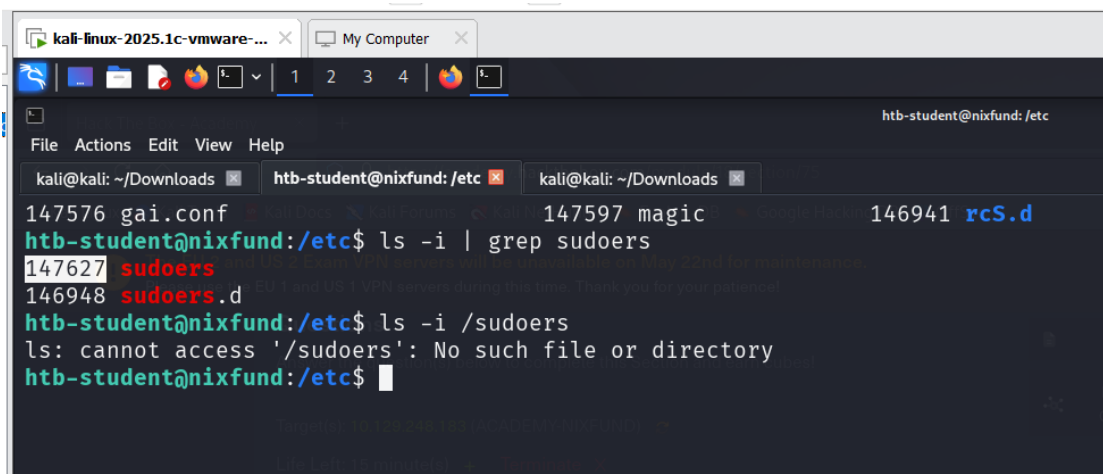
I used the command **ls -a** to display all files in the directory including hidden files



```
kali@kali: ~/Downloads | htb-student@nixfund: ~ | kali@kali: ~/Downloads |
File Actions Edit View Help
htb-student@nixfund: ~$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  .cache  .gnupg  .profile
htb-student@nixfund: ~$
```

What is the index number of the "sudoers" file in the "/etc" directory?

First navigated to the etc directory using the command **cd /etc** then while in the directory I used the command **ls -i** to list all files with their index numbers. The list was long and I found out there was a way to narrow down the search using grep. I used the command **ls -i | grep sudoers** to display just files with the name sudoers and they were displayed as shown below:



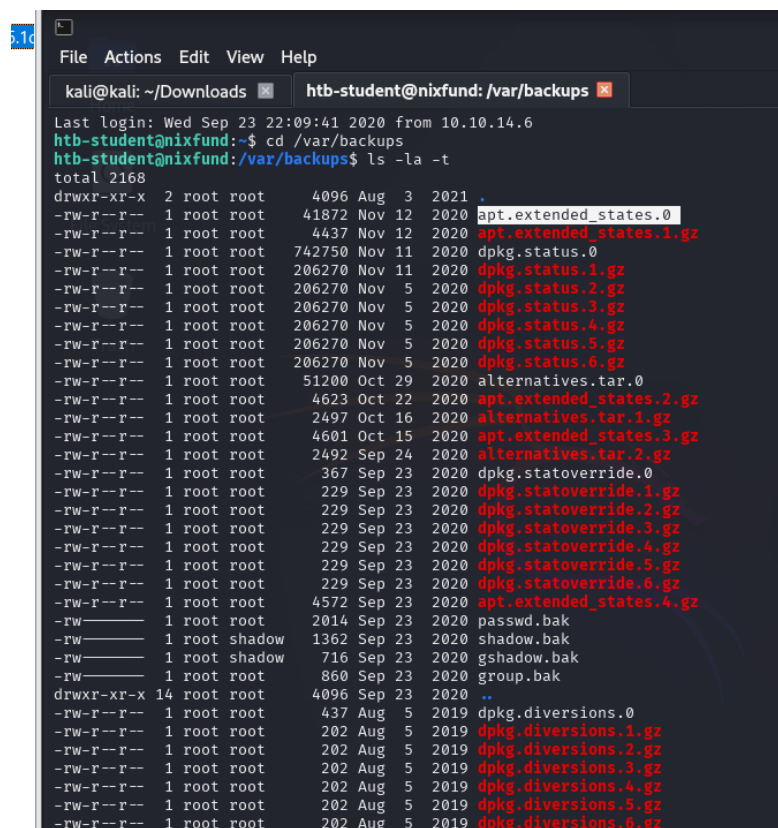
```
kali@kali: ~/Downloads | htb-student@nixfund: /etc | kali@kali: ~/Downloads |
File Actions Edit View Help
147576 gai.conf 147597 magic 146941 rcS.d
htb-student@nixfund: /etc$ ls -i | grep sudoers
147627 sudoers
146948 sudoers.d
htb-student@nixfund: /etc$ ls -i /sudoers
ls: cannot access '/sudoers': No such file or directory
htb-student@nixfund: /etc$
```

### TOPIC 3: Working with Files and Directories:

**Summary:** In Linux, file management is often done through the terminal, offering speed and flexibility that outshines Windows' graphical approach. With just a few commands, you can create, move, rename, and organize files and directories efficiently.

**What is the name of the last modified file in the "/var/backups" directory?**

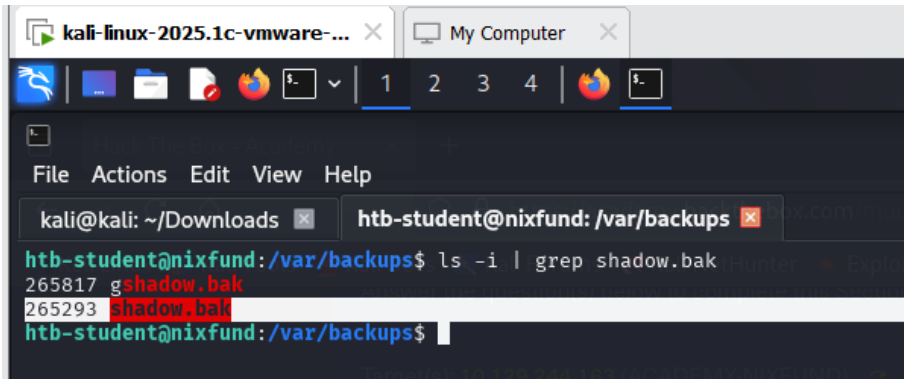
To do this task, I first reviewed the ls manual page to try and find if there was an option to sort the files in the directory and I found -t to suit my case. Went ahead and navigated to the directory and used the command ls -la -t to list all files in the folder, including hidden files, in a long list format and sort the list from newest first and I was able to identify the last modified directory as the highlighted one in the snippet below.



```
kali@kali: ~/Downloads | htb-student@nixfund: /var/backups
Last login: Wed Sep 23 22:09:41 2020 from 10.10.14.6
htb-student@nixfund:~$ cd /var/backups
htb-student@nixfund:/var/backups$ ls -la -t
total 2168
drwxr-xr-x  2 root root    4096 Aug  3  2021 .
-rw-r--r--  1 root root   41872 Nov 12  2020 apt.extended_states.0
-rw-r--r--  1 root root    4437 Nov 12  2020 apt.extended_states.1.gz
-rw-r--r--  1 root root  742750 Nov 11  2020 dpkg.status.0
-rw-r--r--  1 root root  206270 Nov 11  2020 dpkg.status.1.gz
-rw-r--r--  1 root root  206270 Nov  5  2020 dpkg.status.2.gz
-rw-r--r--  1 root root  206270 Nov  5  2020 dpkg.status.3.gz
-rw-r--r--  1 root root  206270 Nov  5  2020 dpkg.status.4.gz
-rw-r--r--  1 root root  206270 Nov  5  2020 dpkg.status.5.gz
-rw-r--r--  1 root root  206270 Nov  5  2020 dpkg.status.6.gz
-rw-r--r--  1 root root   51200 Oct 29  2020 alternatives.tar.0
-rw-r--r--  1 root root    4623 Oct 22  2020 apt.extended_states.2.gz
-rw-r--r--  1 root root    2497 Oct 16  2020 alternatives.tar.1.gz
-rw-r--r--  1 root root    4601 Oct 15  2020 apt.extended_states.3.gz
-rw-r--r--  1 root root    2492 Sep 24  2020 alternatives.tar.2.gz
-rw-r--r--  1 root root    367 Sep 23  2020 dpkg.statoverride.0
-rw-r--r--  1 root root    229 Sep 23  2020 dpkg.statoverride.1.gz
-rw-r--r--  1 root root    229 Sep 23  2020 dpkg.statoverride.2.gz
-rw-r--r--  1 root root    229 Sep 23  2020 dpkg.statoverride.3.gz
-rw-r--r--  1 root root    229 Sep 23  2020 dpkg.statoverride.4.gz
-rw-r--r--  1 root root    229 Sep 23  2020 dpkg.statoverride.5.gz
-rw-r--r--  1 root root    229 Sep 23  2020 dpkg.statoverride.6.gz
-rw-r--r--  1 root root    4572 Sep 23  2020 apt.extended_states.4.gz
-rw-r--r--  1 root root    2014 Sep 23  2020 passwd.bak
-rw-r--r--  1 root shadow  1362 Sep 23  2020 shadow.bak
-rw-r--r--  1 root shadow   716 Sep 23  2020 gshadow.bak
-rw-r--r--  1 root root    860 Sep 23  2020 group.bak
drwxr-xr-x 14 root root    4096 Sep 23  2020 ..
-rw-r--r--  1 root root    437 Aug  5  2019 dpkg.diversions.0
-rw-r--r--  1 root root    202 Aug  5  2019 dpkg.diversions.1.gz
-rw-r--r--  1 root root    202 Aug  5  2019 dpkg.diversions.2.gz
-rw-r--r--  1 root root    202 Aug  5  2019 dpkg.diversions.3.gz
-rw-r--r--  1 root root    202 Aug  5  2019 dpkg.diversions.4.gz
-rw-r--r--  1 root root    202 Aug  5  2019 dpkg.diversions.5.gz
-rw-r--r--  1 root root    202 Aug  5  2019 dpkg.diversions.6.gz
```

**What is the inode number of the "shadow.bak" file in the "/var/backups" directory?**

In this task, I also consulted the ls manual page to get an option to use with the command to display inode of files and determined that it is same as index, so we use the -i option. I also used grep to filter my results since the output had many files. The inode number to the shadow.bak file is highlighted below.



The screenshot shows a Kali Linux terminal window with the title bar 'kali-linux-2025.1c-vmware-...' and 'My Computer'. The terminal has a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'. The prompt is 'kali@kali: ~/Downloads'. The user has switched to a terminal window titled 'htb-student@nixfund: /var/backups'. The command 'ls -i | grep shadow.bak' has been executed, resulting in two lines of output: '265817 gshadow.bak' and '265293 shadow.bak'. The prompt is now 'htb-student@nixfund: /var/backups\$'.

## TOPIC 4: Editing Files:

In this sub-topic, I explored text file editing using common Linux editors. While tools like Vi and Vim are powerful and widely used, nano has a simple and user-friendly interface, making it ideal for beginners.

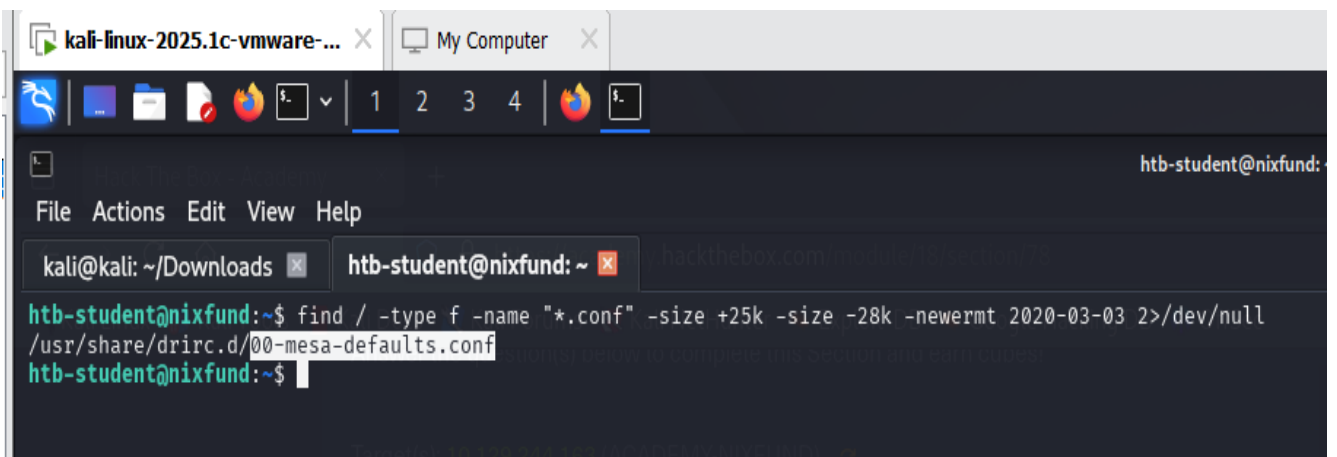
I learned how to create, edit, search within, save, and exit files using Nano, and how to verify content using commands like **cat**.

Vim is powerful and efficient for more advanced users. Mastery of Vim is essential for advanced Linux usage, and tools like vimtutor provide a structured way to practice.

## TOPIC 5: Find Files and Directories:

What is the name of the config file that has been created after 2020-03-03 and is smaller than 28k but larger than 25k?

Used the find command: **find / -type f -name "\*.conf" -size +25k -size -28k -newermt 2020-03-03 2>/dev/null** that finds all files with the .conf extension with a size more than 25KB but smaller than 28KB and created after the specified date from the root directory. **2>/dev/null** is used to suppress errors and get a more “clean” output as shown below

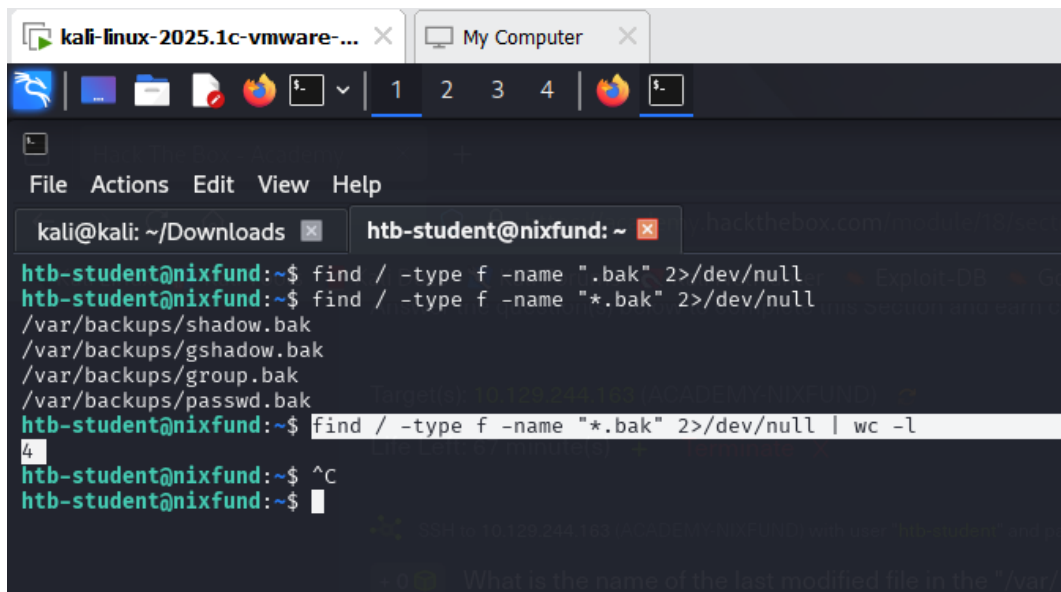


The screenshot shows a Kali Linux terminal window with the title bar 'kali-linux-2025.1c-vmware-...' and 'My Computer'. The terminal has a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'. The prompt is 'kali@kali: ~/Downloads'. The user has switched to a terminal window titled 'htb-student@nixfund: ~'. The command 'find / -type f -name "\*.conf" -size +25k -size -28k -newermt 2020-03-03 2>/dev/null' has been executed, resulting in one line of output: '/usr/share/drirc.d/00-mesa-defaults.conf'. The prompt is now 'htb-student@nixfund: ~\$'.



How many files exist on the system that have the ".bak" extension?

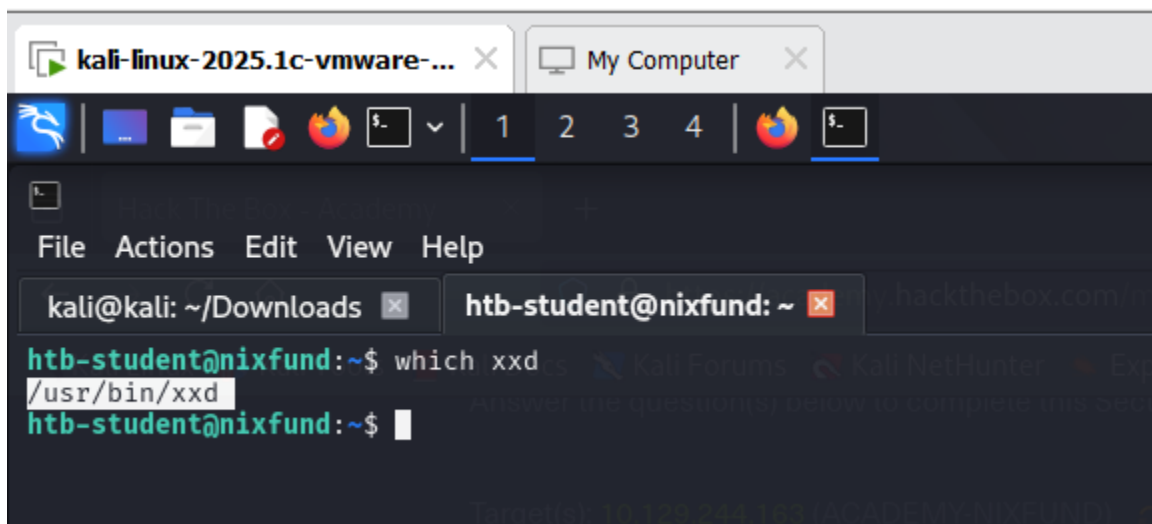
I used the find command: `find / -type f -name "*.bak" 2>/dev/null` that finds files that have a .bak extension from the root directory and suppresses error messages to minimize output. The files were just 4 but I did a bit of research of how to get a number instead of counting the files and I got an option to pipe the output into `wc -l` which means word count but using -l I can count lines of my output.



```
kali@kali: ~/Downloads x htb-student@nixfund: ~  
htb-student@nixfund:~$ find / -type f -name ".bak" 2>/dev/null  
htb-student@nixfund:~$ find / -type f -name "*.bak" 2>/dev/null  
/var/backups/shadow.bak  
/var/backups/gshadow.bak  
/var/backups/group.bak  
/var/backups/passwd.bak  
htb-student@nixfund:~$ find / -type f -name "*.bak" 2>/dev/null | wc -l  
4  
htb-student@nixfund:~$ ^C  
htb-student@nixfund:~$
```

Submit the full path of the "xxd" binary.

Used which command as it is a tool used to return path of a file as shown below:



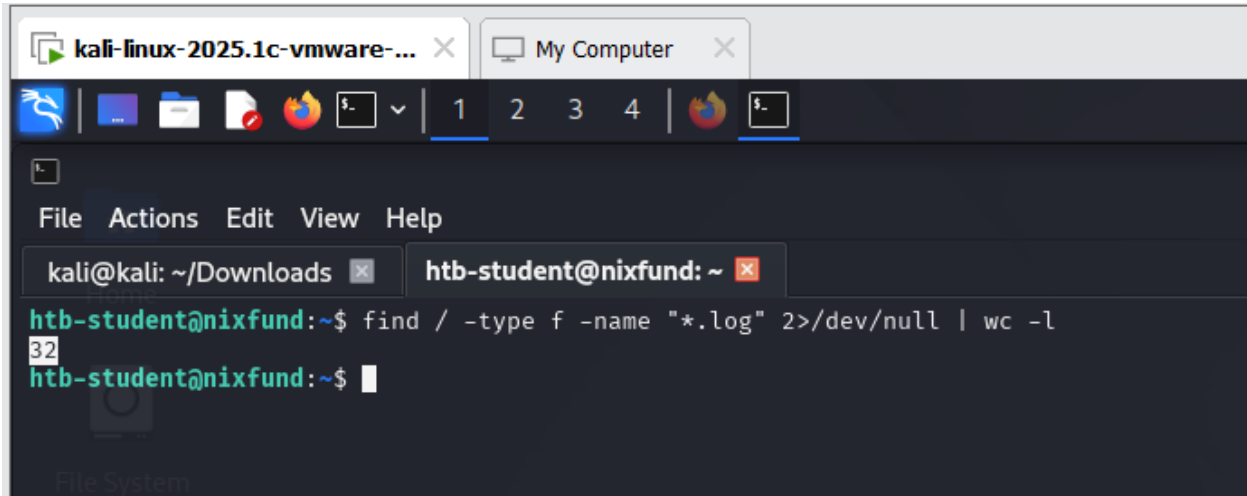
```
kali@kali: ~/Downloads x htb-student@nixfund: ~  
htb-student@nixfund:~$ which xxd  
/usr/bin/xxd  
htb-student@nixfund:~$
```



## TOPIC 6: File Descriptors and Redirections:

How many files exist on the system that have the ".log" file extension?

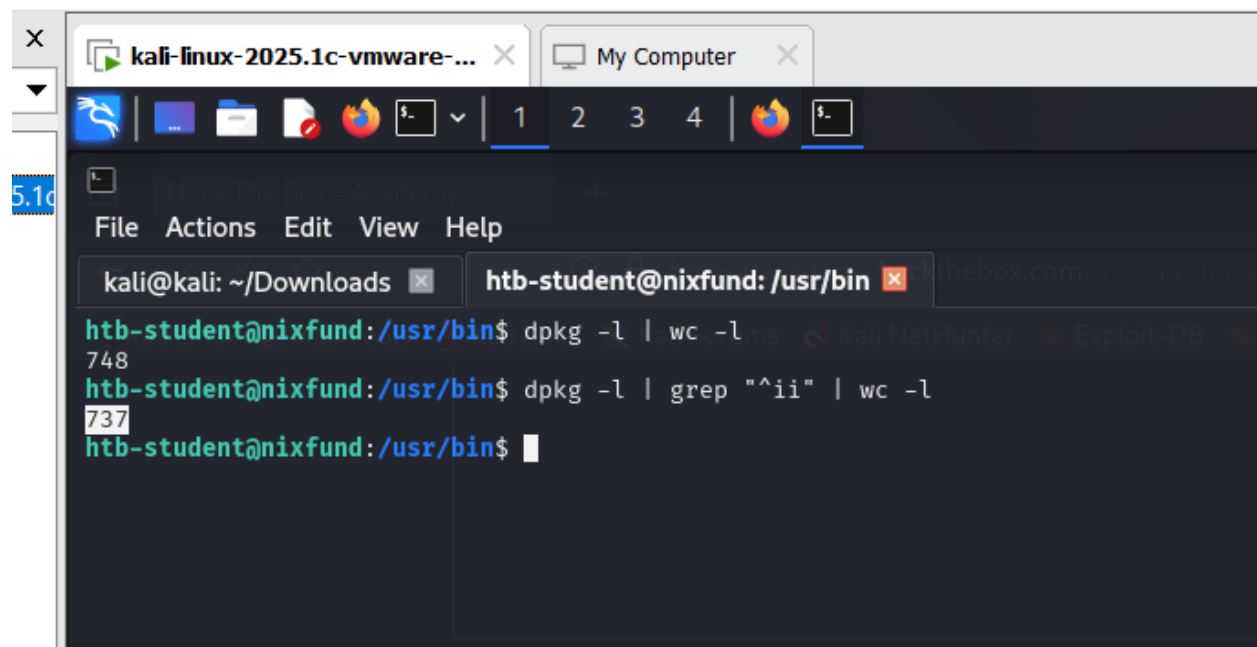
In this question, I used the **find** command to specify the type of resource I want to find which is a file and should be a log file then piped the **wc** command with the **-l** option to list the number of lines in the display so as to get our count instead of doing it manually.



```
kali@kali: ~/Downloads x htb-student@nixfund: ~ x
File Actions Edit View Help
htb-student@nixfund:~$ find / -type f -name "*.log" 2>/dev/null | wc -l
32
htb-student@nixfund:~$
```

How many total packages are installed on the target system?

Used the man page of dpkg to try and get an option that would list installed packages and the **-l** option was the suitable one. Then piped the **grep ii** command to list only files with ii since there were some extra lines at the top. To get a count of the lines displayed, I piped **wc -l** as shown below



```
5.1d
File Actions Edit View Help
kali@kali: ~/Downloads x htb-student@nixfund: /usr/bin x
htb-student@nixfund:/usr/bin$ dpkg -l | wc -l
748
htb-student@nixfund:/usr/bin$ dpkg -l | grep "^ii" | wc -l
737
htb-student@nixfund:/usr/bin$
```

## TOPIC 7: Filter Contents:

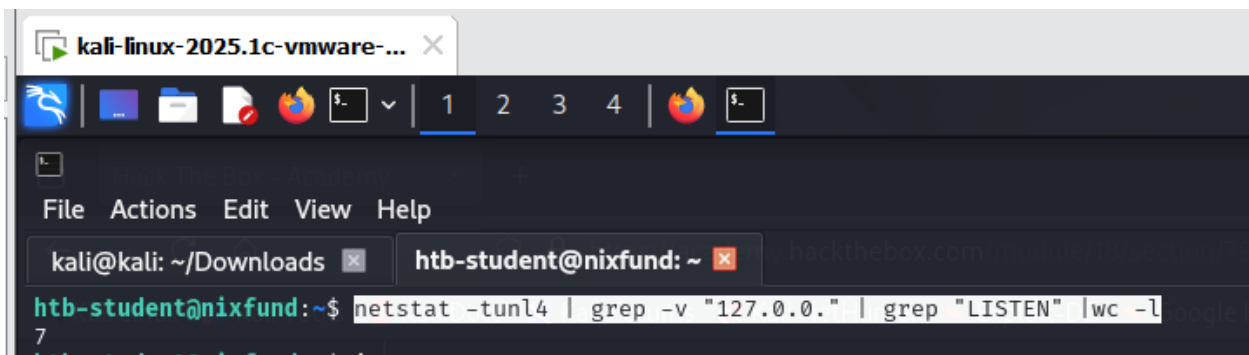
In this section, I learned how to read and filter files directly from the command line without using a text editor. I used tools like **more** and **less** to view large files interactively, with **less** offering more advanced navigation options. When I only needed to see parts of a file, I used **head** to view the beginning and **tail** to see the end.

To organize the output, I used **sort** to arrange data alphabetically or numerically. When searching for specific information, I applied **grep** to find or exclude lines based on patterns. I also practiced using **cut**, **tr**, **column**, and **awk** to extract, format, and display text in a cleaner way.

I used **sed** to substitute words or patterns in the output. Also used **wc** to count the number of lines in the filtered results when used with a **-l** option.

**How many services are listening on the target system on all interfaces? (Not on localhost and IPv4 only)**

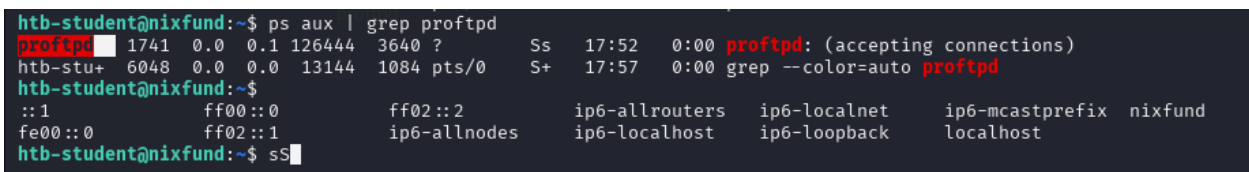
Used the command **netstat -tunl4 | grep -v "127.0.0." | grep "LISTEN" | wc -l** to show network services listening



```
kali-linux-2025.1c-vmware-... x
File Actions Edit View Help
kali@kali: ~/Downloads x htb-student@nixfund: ~ x
htb-student@nixfund:~$ netstat -tunl4 | grep -v "127.0.0." | grep "LISTEN" | wc -l
7
```

**Determine what user the ProFTPD server is running under. Submit the username as the answer.**

The ProFTPD server should be running as a process, I used the command **ps aux | grep proftpd** to list all processes named proftpd and the user running under was proftpd as shown below



```
htb-student@nixfund:~$ ps aux | grep proftpd
proftpd 1741 0.0 0.1 126444 3640 ? Ss 17:52 0:00 proftpd: (accepting connections)
htb-stu+ 6048 0.0 0.0 13144 1084 pts/0 S+ 17:57 0:00 grep --color=auto proftpd
htb-student@nixfund:~$
```

**Use cURL from your Pwnbox (not the target machine) to obtain the source code of the "https://www.inlanefreight.com" website and filter all unique paths**

(<https://www.inlanefreight.com/directory>" or `"/another/directory"`) of that domain. Submit the number of these paths as the answer.

## TOPIC 8: Regular Expressions (RegEx)

In this section, I explored Regular Expressions (RegEx) and how they help search and manipulate text by creating precise search patterns.

I learned about different types of brackets used to group patterns, specify character sets, and define how often patterns repeat. I also practiced using operators like OR and AND to refine searches.

## TOPIC 9: Permission Management

Linux permissions control access to files and directories through three key permissions: read (r), write (w), and execute (x). These permissions are assigned separately to the file owner, group members, and other users. For directories, execute permission is needed to access contents, write permission allows modifying files, and read permission lets users view what's inside. The `chmod` command changes these permissions using either symbolic notation (like `rwx`) or numeric codes (like `755`), while `chown` adjusts file ownership.

Special permission bits add extra functionality. The SUID and SGID bits let programs run with the owner's or group's permissions, useful for system tools but risky if misconfigured. The sticky bit, often used on shared directories like `/tmp`, ensures only file owners can delete their files, preventing accidental or malicious changes by others. These features help balance security with necessary access in multi-user environments.

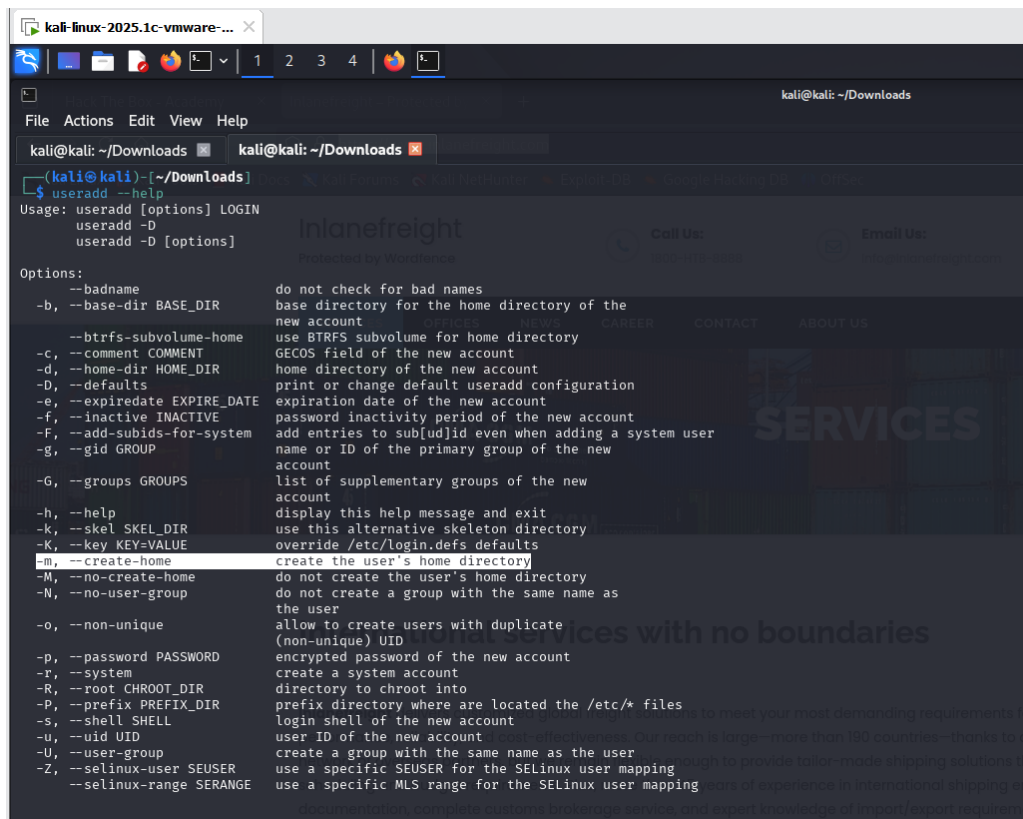
## SYSTEM MANAGEMENT

### TOPIC 10: User Management:

Linux user management revolves around controlling access while maintaining system security. Linux administrators work with user accounts, groups, and permissions to ensure proper access to files and commands. Critical system files like `/etc/shadow` are locked down by default, only accessible by root, while tools like `sudo` allow temporary privilege elevation for specific tasks without full root access.

The essential commands: **`useradd`**, **`usermod`**, and **`userdel`** handle account creation, modification and deletion respectively, while **`passwd`** manages authentication credentials. Groups enable efficient permission management, allowing administrators to control access for multiple users simultaneously. The **`su`** command provides another method for switching user contexts when needed.

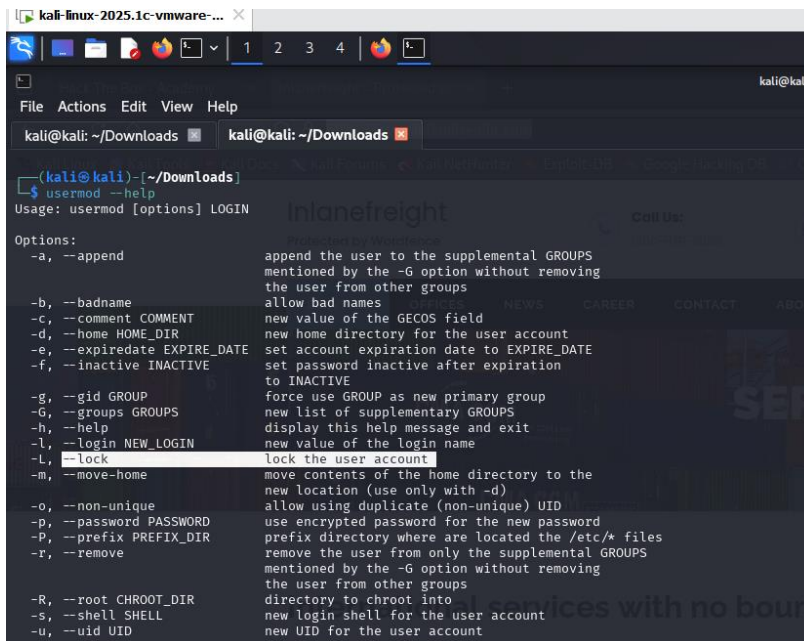
Which option needs to be set to create a home directory for a new user using "useradd" command? **-m**



```
kali@kali: ~/Downloads
(kali@kali)~-[~/Downloads]
$ useradd --help
Usage: useradd [options] LOGIN
       useradd -D
       useradd -D [options]

Options:
  -b, --badname                do not check for bad names
  -b, --base-dir BASE_DIR      base directory for the home directory of the
                               new account
  -b, --btrfs-subvolume-home   use BTRFS subvolume for home directory
  -c, --comment COMMENT        GECOS field of the new account
  -d, --home-dir HOME_DIR      home directory of the new account
  -D, --defaults                print or change default useradd configuration
  -e, --expiredate EXPIRE_DATE expiration date of the new account
  -f, --inactive INACTIVE      password inactivity period of the new account
  -F, --add-subids-for-system   add entries to sub[uid]id even when adding a system user
  -g, --gid GROUP              name or ID of the primary group of the new
                               account
  -G, --groups GROUPS          list of supplementary groups of the new
                               account
  -h, --help                   display this help message and exit
  -k, --skel SKEL_DIR          use this alternative skeleton directory
  -K, --key KEY=VALUE          override /etc/login.defs defaults
  -m, --create-home            create the user's home directory
  -M, --no-create-home         do not create the user's home directory
  -N, --no-user-group          do not create a group with the same name as
                               the user
  -o, --non-unique             allow to create users with duplicate
                               (non-unique) UID
  -p, --password PASSWORD      encrypted password of the new account
  -r, --system                create a system account
  -R, --root CHROOT_DIR        directory to chroot into
  -P, --prefix PREFIX_DIR      prefix directory where are located the /etc/* files
  -s, --shell SHELL            login shell of the new account
  -u, --uid UID                user ID of the new account
  -U, --user-group             create a group with the same name as the user
  -Z, --selinux-user SEUSER    use a specific SEUSER for the SELinux user mapping
  -Z, --selinux-range SERANGE use a specific MLS range for the SELinux user mapping
```

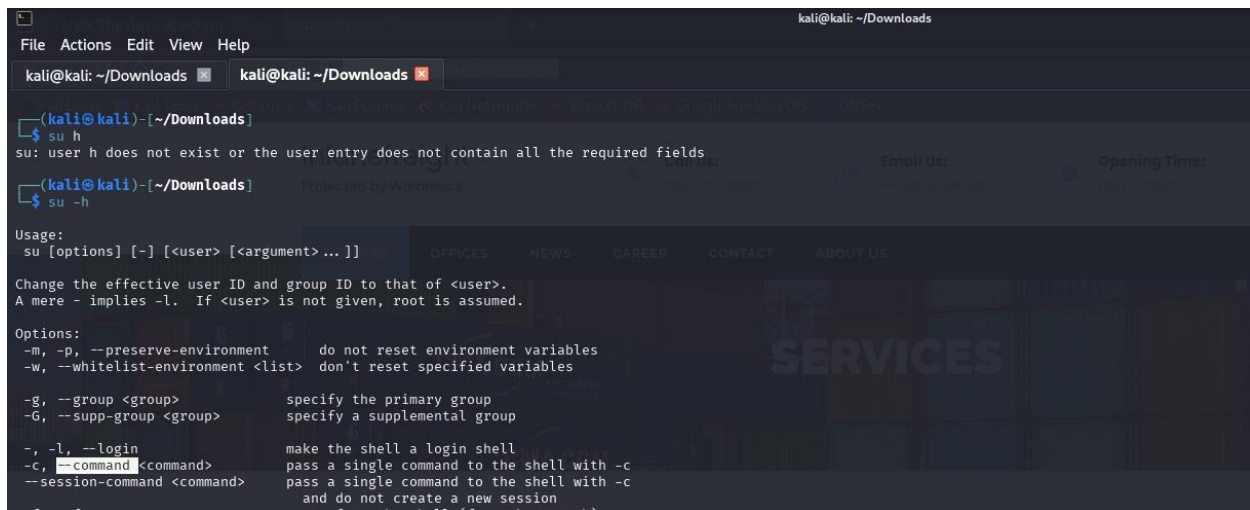
Which option needs to be set to lock a user account using the "usermod" command? (long version of the option) **--lock**



```
kali@kali: ~/Downloads
(kali@kali)~-[~/Downloads]
$ usermod --help
Usage: usermod [options] LOGIN

Options:
  -a, --append                append the user to the supplemental GROUPS
                               mentioned by the -G option without removing
                               the user from other groups
  -b, --badname                allow bad names
  -c, --comment COMMENT        new value of the GECOS field
  -d, --home HOME_DIR          new home directory for the user account
  -e, --expiredate EXPIRE_DATE set account expiration date to EXPIRE_DATE
  -f, --inactive INACTIVE      set password inactive after expiration
                               to INACTIVE
  -g, --gid GROUP              force use GROUP as new primary group
  -G, --groups GROUPS          new list of supplementary GROUPS
  -h, --help                   display this help message and exit
  -l, --login NEW_LOGIN        new value of the login name
  -L, --lock                   lock the user account
  -m, --move-home              move contents of the home directory to the
                               new location (use only with -d)
  -o, --non-unique             allow using duplicate (non-unique) UID
  -p, --password PASSWORD      use encrypted password for the new password
  -P, --prefix PREFIX_DIR      prefix directory where are located the /etc/* files
  -r, --remove                 remove the user from only the supplemental GROUPS
                               mentioned by the -G option without removing
                               the user from other groups
  -R, --root CHROOT_DIR        directory to chroot into
  -s, --shell SHELL            new login shell for the user account
  -u, --uid UID                new UID for the user account
  -U, --unlock                 unlock the user account
```

Which option needs to be set to execute a command as a different user using the "su" command?  
(long version of the option)



```
kali@kali: ~/Downloads
File Actions Edit View Help
kali@kali: ~/Downloads kali@kali: ~/Downloads
(kali@kali) ~/Downloads
$ su h
su: user h does not exist or the user entry does not contain all the required fields
(kali@kali) ~/Downloads
$ su -h
Usage:
su [options] [-] [<user>] [<argument> ...]
Change the effective user ID and group ID to that of <user>.
A mere - implies -l. If <user> is not given, root is assumed.
Options:
-m, -p, --preserve-environment    do not reset environment variables
-w, --whitelist-environment <list> don't reset specified variables
-g, --group <group>               specify the primary group
-G, --supp-group <group>          specify a supplemental group
-, -l, --login                    make the shell a login shell
-c, --command <command>          pass a single command to the shell with -c
--session-command <command>     pass a single command to the shell with -c
                                and do not create a new session
                                pass -f to the shell (for ssh on test)
```

## TOPIC 11: Package Management:

Package management systems are critical for software deployment across Linux environments. These systems handle binary distribution, dependency resolution, and version control through standardized formats like .deb and .rpm. Core functionality includes automated installation, configuration, and removal of software packages while maintaining system integrity through checksums and quality validation.

Debian-based distributions utilize the APT ecosystem, combining **dpkg** for low-level package operations with higher-level tools for repository management. The system maintains an indexed cache of available packages, enabling efficient searches and dependency resolution. For development environments, language-specific managers like pip (Python) and gem (Ruby) provide additional granularity, while git facilitates direct source code integration when pre-packaged solutions are unavailable.

Administrators routinely interact with these systems through fundamental commands: `apt update` refreshes available packages, `apt install` handles deployments, and **dpkg** manages standalone package files. The architecture supports multiple installation methods - from curated repository packages to manually acquired .deb files - while enforcing dependency chains and configuration standards. This multilayered approach ensures consistent software deployment across diverse operational requirements.

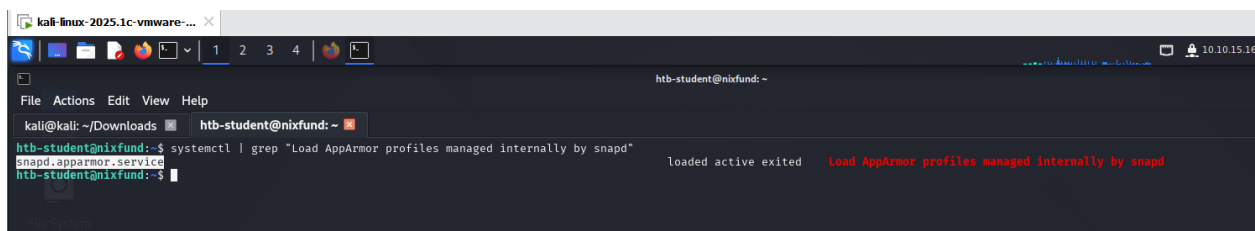
## TOPIC 12: Service and Process Management

Linux services (daemons) operate as background processes supporting core system functionality and user applications. They are identified by the letter **d** at the end of their program names, such as `sshd` (SSH daemon) or `systemd`. Are categorized into 2, **system services** that initialize at boot, and user-installed services that are added by users and typically include server applications and other background processes that provide specific features or capabilities. Modern distributions use `systemd` as their initialization system (`init system`).

Key service operations include starting (with the command **`systemctl start`**), stopping (with the command **`systemctl stop`**), checking status (with the command **`systemctl status`**), and enabling autostart (with the command **`systemctl enable`**). The `ps` display information about running processes on the system, while **`journalctl`** offers detailed service logs for troubleshooting. Processes can be in one of these states: **running**, **waiting**, **stopped**, or **zombie**.

**Use the "`systemctl`" command to list all units of services and submit the unit name with the description "Load AppArmor profiles managed internally by snapd" as the answer.**

Used the command `systemctl` to list all services but piped **`grep "Load AppArmor profiles managed internally by snapd"`** to list only services with the description and we got the unit name as `snapd.apparmor.service`



```
kali@kali: ~/Downloads htb-student@nixfund: ~
htb-student@nixfund:~$ systemctl | grep "Load AppArmor profiles managed internally by snapd"
snapd.apparmor.service
htb-student@nixfund:~$
```

## TOPIC 13: Task Scheduling

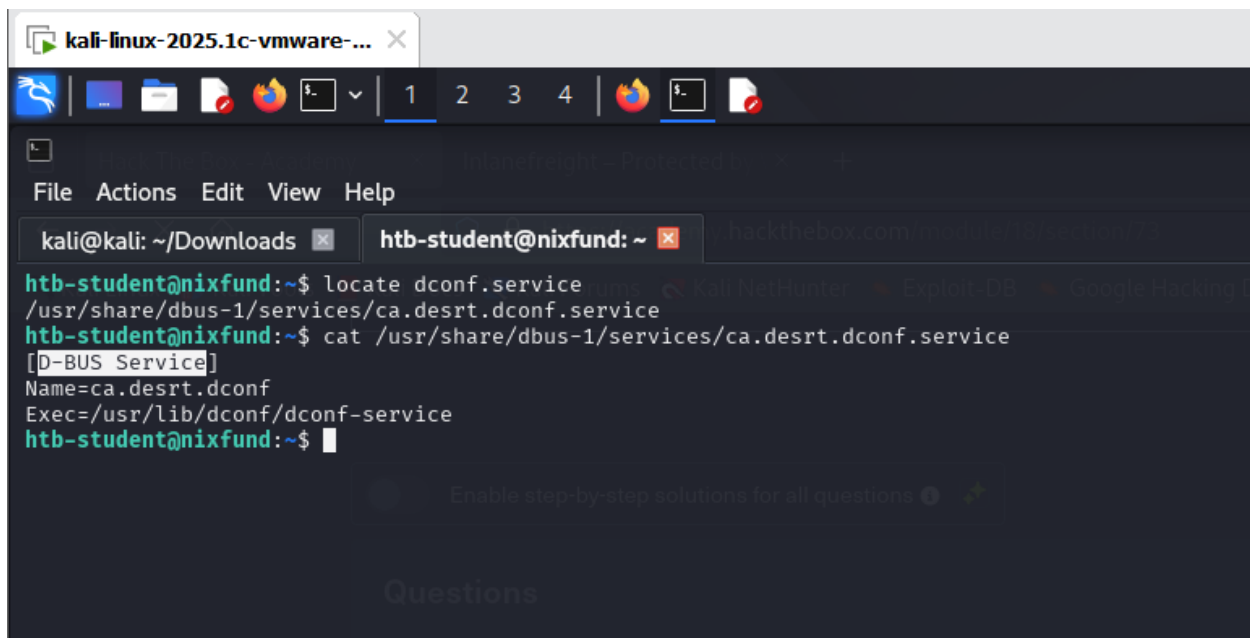
Linux provides two primary methods for scheduling tasks which are `cron` and `systemd` timers. `Cron` allows users and administrators to execute tasks at a specific time or within specific intervals. This makes it ideal for straightforward scheduling needs like regular maintenance jobs or periodic script execution. To set up the `cron` daemon, we need to store the tasks in a file called `crontab` and then tell the daemon when to run the tasks.

`Systemd` provides scheduled task execution in Linux distributions. It enables administrators to run processes at specific times, intervals, or triggered by system events. Implementation requires creating timer and service unit files that define when and what to execute, followed by proper system registration. This method offers more precise control than traditional `cron` jobs while integrating with `systemd`'s service management framework. Proper configuration is essential to ensure reliable automated execution of system tasks.

From a security perspective, both scheduling systems require careful monitoring. They are equally valuable for legitimate automation and potentially dangerous if compromised, as they can enable persistent access or malicious payload execution. Regular audits of all scheduled tasks should be part of standard system maintenance procedures, checking both cron jobs and systemd timers for any unauthorized entries.

### What is the Type of the service of the "dconf.service"?

After trying to locate the type of the service from the services available after running `systemctl`, I did not find the type. I tried to locate the `dconf.service` file using the `locate` command and then used `cat` to open the file and the type of the service was listed at the top of the file as DBUS



The screenshot shows a terminal window with the following commands and output:

```
htb-student@nixfund:~$ locate dconf.service
/usr/share/dbus-1/services/ca.desrt.dconf.service
htb-student@nixfund:~$ cat /usr/share/dbus-1/services/ca.desrt.dconf.service
[D-BUS Service]
Name=ca.desrt.dconf
Exec=/usr/lib/dconf/dconf-service
htb-student@nixfund:~$
```

## TOPIC 14: Network Services:

Network services are designed to perform specific tasks, many of which enable remote operations. SSH (OpenSSH) provides encrypted remote access and file transfers, requiring proper server configuration in `/etc/ssh/sshd_config`. NFS enables shared network storage but needs careful permission management in `/etc/exports` to prevent security risks like unauthorized root access.

Web servers like Apache and Python's HTTP server are essential for penetration testers, serving both as assessment targets and tools for hosting files or phishing pages during security tests. Apache offers robust configuration through files like `/etc/apache2/apache2.conf` and `.htaccess`, while Python's lightweight `python3 -m http.server` enables quick file sharing on custom ports - both requiring careful setup to avoid security exposures during testing engagements.



VPNs create encrypted connections that let users securely access remote networks as if they were locally connected ideal for remote work or private browsing. OpenVPN is a top choice for Linux, offering strong encryption and flexibility for both corporate access and penetration testing. Admins use it to protect data, while security testers rely on it to safely evaluate networks.

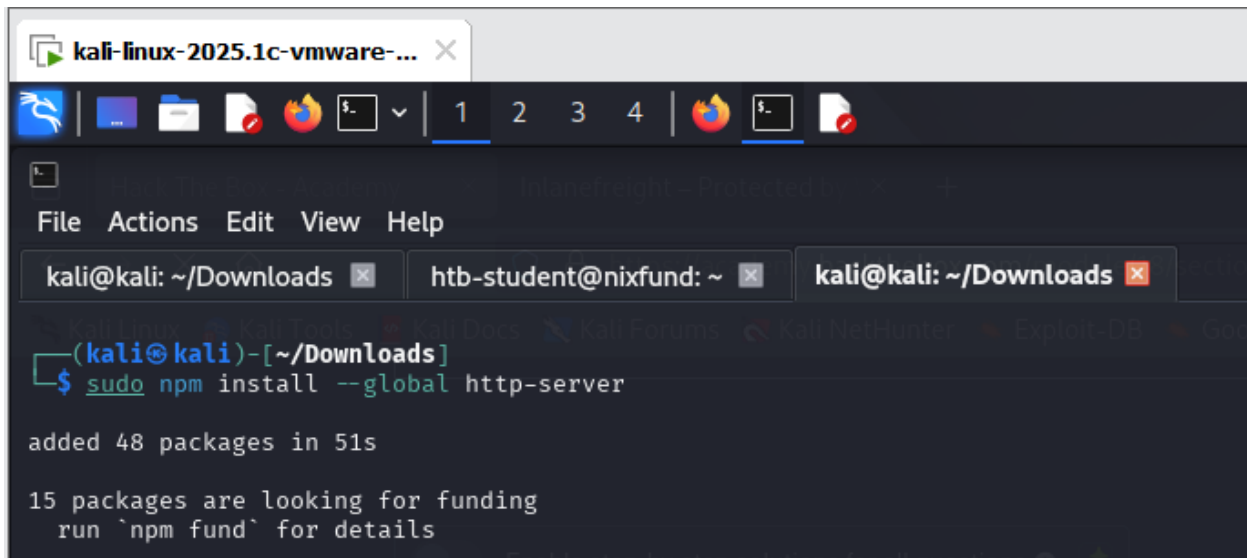
## TOPIC 15: Working with Web Services:

Apache is a powerful, modular web server that handles everything from static pages to dynamic content using scripting languages like PHP or Python. It's highly customizable modules like **mod\_ssl** encrypt traffic, while **mod\_rewrite** modifies HTTP headers and URLs on the fly. Setting up command, **sudo apt install apache2**), and tools like **curl** and **wget** let you interact with websites directly from the terminal, perfect for testing or automation.

For quick file sharing, Python's built-in web server `python3 -m http.server` spins up instantly, ideal for temporary transfers during security assessments. Whether configuring Apache's ports or debugging with command line tools, these services are key for both web development and penetration testing where adaptability and problem-solving matter most.

**Find a way to start a simple HTTP server inside Pwnbox or your local VM using "npm". Submit the command that starts the web server on port 8080 (use the short argument to specify the port number).**

First installed the http-server globally:



```
kali-linux-2025.1c-vmware-... x
1 2 3 4
File Actions Edit View Help
kali@kali: ~/Downloads x htb-student@nixfund: ~ x kali@kali: ~/Downloads x
(kali@kali)-[~/Downloads]
$ sudo npm install --global http-server
added 48 packages in 51s
15 packages are looking for funding
  run `npm fund` for details
```

Starting the http server on port 8080:

```
(kali@kali)-[~/Downloads]
$ http-server -p 8080
Starting up http-server, serving ./

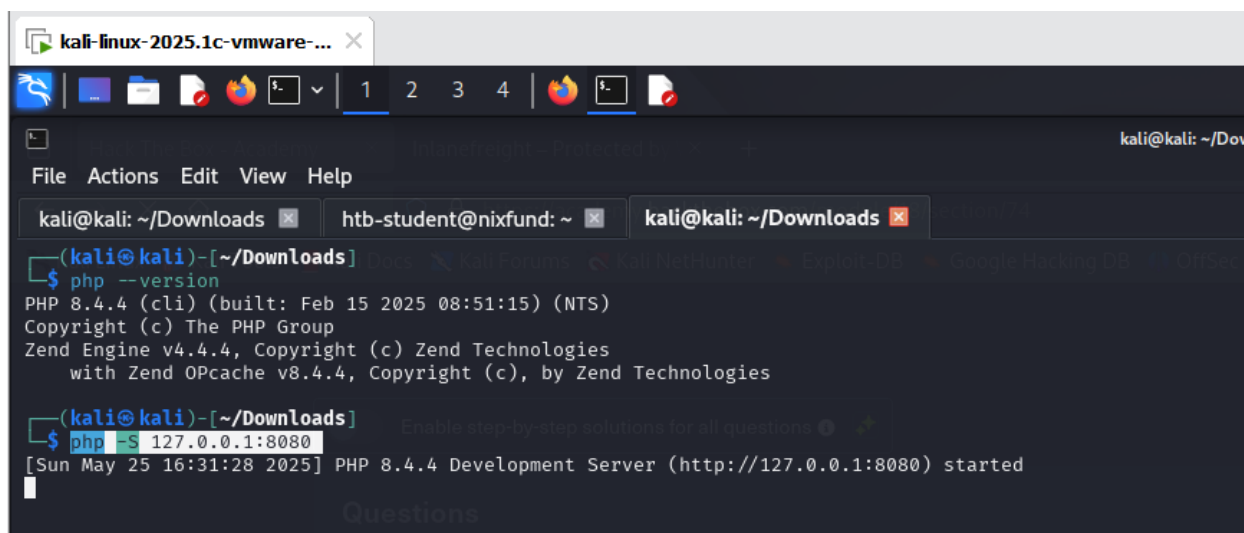
http-server version: 14.1.1

http-server settings:
CORS: disabled
Cache: 3600 seconds
Connection Timeout: 120 seconds
Directory Listings: visible
AutoIndex: visible
Serve GZIP Files: false
Serve Brotli Files: false
Default File Extension: none

Available on:
http://127.0.0.1:8080
http://192.168.112.128:8080
http://10.10.15.161:8080
Hit CTRL-C to stop the server
```

Find a way to start a simple HTTP server inside Pwnbox or your local VM using "php". Submit the command that starts the web server on the localhost (127.0.0.1) on port 8080.

First checked if php is installed then started the server using the command **php -S 127.0.0.1:8080**



```
kali-linux-2025.1c-vmware-... x
kali@kali: ~/Downloads
(kali@kali)-[~/Downloads]
$ php --version
PHP 8.4.4 (cli) (built: Feb 15 2025 08:51:15) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.4.4, Copyright (c) Zend Technologies
with Zend OPcache v8.4.4, Copyright (c), by Zend Technologies

(kali@kali)-[~/Downloads]
$ php -S 127.0.0.1:8080
[Sun May 25 16:31:28 2025] PHP 8.4.4 Development Server (http://127.0.0.1:8080) started
```

## TOPIC 16: Backup and Restore:

Linux offers powerful tools for backing up and restoring data. Rsync is an open-source tool that allows for fast and secure backups, whether locally or to a remote location. Duplicity is another powerful tool that builds on Rsync, but adds encryption features to protect the backups. Deja Dup is a simple, accessible safe that anyone can operate, while still offering the same level of protection.

## TOPIC 17: File System Management:

Linux supports multiple file systems, each optimized for specific use cases. The most common include **ext4**, default for most distributions, offering reliability and performance, **XFS**, optimized for large files and high throughput, and **Btrfs**, with advanced features like snapshots and checksum-based data integrity. For compatibility with Windows systems, NTFS and FAT32 are also supported.

File system management involves partitioning disks (**fdisk**, **gparted**), formatting partitions with a chosen file system (**mkfs**), and mounting them to directories (**mount**, **/etc/fstab**). The Linux file hierarchy follows a standardized structure, with key directories like **/root**, **/home** (user files), and **/etc** (configuration files). Permissions and ownership (managed via **chmod**, **chown**) ensure secure access control. Additionally, swap space can be configured to supplement RAM when memory demands exceed physical capacity.

## TOPIC 18: Containerization:

Containerization is the process of packaging and running applications in isolated environments, typically referred to as containers. These containers provide lightweight, consistent environments for applications to run, ensuring that they behave the same way, regardless of where they are deployed. Docker is the most widely used container platform, allowing users to package applications with their dependencies into portable images that run consistently across different environments.

Containers are highly configurable, allowing users to tailor them to their specific needs, and their lightweight nature makes it easy to run multiple containers simultaneously on the same host system. This feature is particularly advantageous for scaling applications and managing complex microservice architectures.

Containers isolate applications from the host system and from each other, providing a barrier that reduces the risk of malicious activities affecting the host or other containers. This isolation, along with proper configuration and hardening techniques, adds an additional layer of security. If not properly managed, vulnerabilities such as privilege escalation or container escapes can be exploited to gain unauthorized access to the host system or other containers.

## **LINUX NETWORKING**

### **TOPIC 19: Network Configuration:**

Network configuration is a fundamental skill for penetration testers, enabling control over testing environments, traffic manipulation, and vulnerability identification. Key tasks include managing interfaces (assigning IPs, setting gateways, and configuring DNS) and implementing network access control (NAC) models like DAC, MAC, and RBAC to regulate resource access. Tools like SELinux, AppArmor, and TCP Wrappers enhance security by enforcing strict access policies.

### **TOPIC 20: Remote Desktop Protocols in Linux**

Remote desktop protocols enable administrators to access and control systems graphically from remote locations. These protocols are essential for system maintenance, troubleshooting, and administration across different operating systems. The choice of protocol depends on the target system's OS, with each offering unique features and security considerations.

The three main protocols are RDP, VNC, and X11/XServer. Microsoft's Remote Desktop Protocol (RDP) is native to Windows systems, using port 3389 by default, and provides full desktop access with features like clipboard sharing and printer redirection. Virtual Network Computing (VNC) is a cross-platform solution commonly used in Linux environments, operating on port 5900+ in its various implementations (TigerVNC, RealVNC, TightVNC). The X Window System (X11) is the native graphical system for Unix-like operating systems, using ports 6000-6010 for communication.

Each protocol presents distinct security challenges. X11 is particularly vulnerable as it transmits data unencrypted by default, requiring SSH tunneling for secure operation. VNC implementations vary in security, with modern versions like TigerVNC supporting encryption. RDP should always be used with Network Level Authentication enabled. The X Display Manager Control Protocol (XDMCP) on UDP port 177 is especially risky and generally should be disabled in production environments.

For Linux systems, VNC servers require proper configuration including password protection and preferably SSH tunneling. X11 forwarding through SSH (-X flag) provides a more secure alternative to direct X11 connections. Windows administrators should enable Network Level Authentication for RDP and restrict access through firewalls. All remote access solutions benefit from VPN protection when used over untrusted networks, and should be disabled when not actively needed to reduce attack surface.

## LINUX HARDENING

### TOPIC 21: Linux Security:

All computer systems face security risks, but Linux systems generally present fewer vulnerabilities than Windows environments, especially those joined to Active Directory domains. However, internet-facing Linux servers hosting web applications still require robust protection. While Linux is less susceptible to traditional viruses, its security depends heavily on proper configuration and maintenance.

Keeping the operating system and installed packages up to date is one of the most critical security measures. Regular updates patch known vulnerabilities that attackers could exploit. Administrators should frequently run commands like **apt update && apt dist-upgrade** on Debian-based systems or **yum update** on RHEL-based distributions. Kernel updates often require special attention since some versions must be updated manually. Neglecting updates leaves systems exposed to exploits targeting outdated software.

User privileges must follow the principle of least privilege, with sudo access restricted to specific commands rather than full administrative rights. Regular audits using tools like Lynis help identify misconfigurations, SUID/SGID binaries, and insecure file permissions. For advanced protection, SELinux or AppArmor enforce mandatory access controls by defining granular policies for processes and files.

Unnecessary services should be disabled to reduce attack surfaces, while TCP Wrappers restrict service access by IP. Additional measures include enabling NTP for log consistency, enforcing strong passwords, and deploying intrusion detection tools like fail2ban. Security is an ongoing process and continuous monitoring, logging, and adaptation to emerging threats are essential for maintaining system integrity.

### TOPIC 22: Firewall Setup:

Firewalls act as gatekeepers for network traffic, controlling what enters and exits a system. In Linux, firewalls are built into the kernel via Netfilter, with tools like iptables, nftables, ufw, and firewalld providing rule management. Their primary role is blocking malicious traffic while allowing legitimate connections. Critical for securing servers and network devices.

Originally, Linux relied on **ipchains** and **ipfwadm**, but **iptables**, introduced in 2000, became the standard due to its flexibility. It filters traffic based on IPs, ports, and protocols, defending against threats like DoS attacks and port scans. Modern alternatives like nftables and ufw now exist, but iptables remains widely used.

## TOPIC 23: Systems logs and Monitoring:

Linux system logs serve as a detailed history of everything happening within an operating system, recording events from kernel operations to user activities. These logs, stored primarily in the `/var/log` directory, provide administrators and security professionals with vital insights into system health, potential vulnerabilities, and active threats. For penetration testers and security teams, these logs are invaluable as they reveal failed login attempts, unusual file access patterns, and even successful breaches that might otherwise go unnoticed.

Proper log management is crucial. Systems need automatic log rotation to prevent storage overload, and sensitive logs require protection from tampering. Regular reviews turn these text files into an early warning system, helping catch breaches before they escalate. In the digital world where threats are invisible, logs give administrators the visibility they need to keep systems secure.

## FINAL SCREENSHOT:

