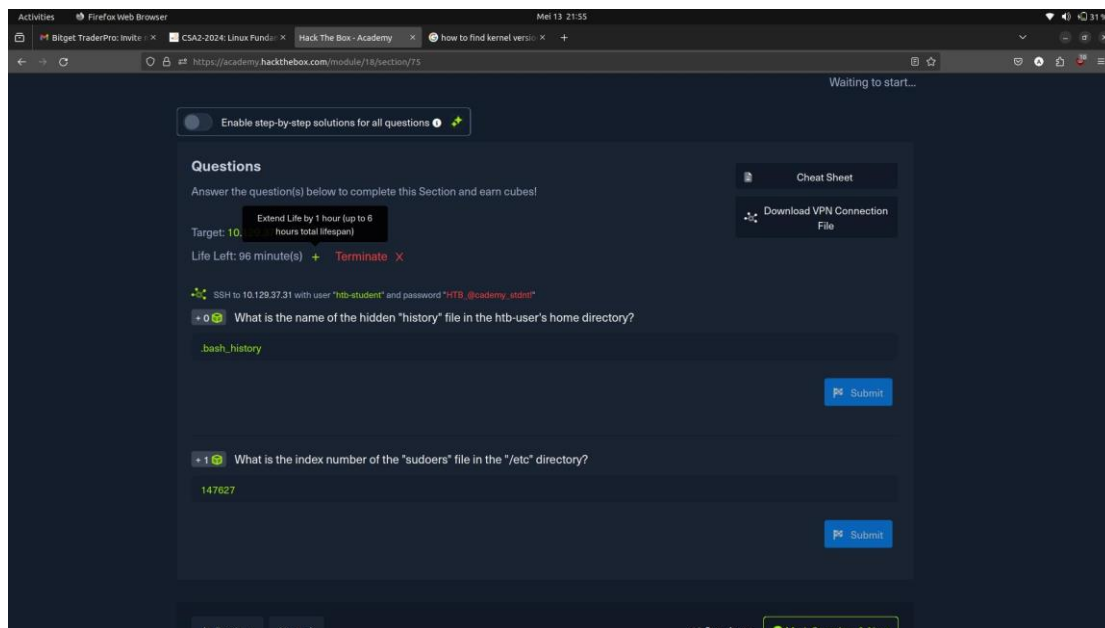


Linux Fundamentals

Shareable link: <https://academy.hackthebox.com/achievement/1296187/18>

Navigation



Working with Files and Directories in Linux

In Linux, file management revolves around the terminal, offering efficiency and speed compared to Windows. While Windows relies on Explorer for file access, Linux users utilize commands directly in the terminal for tasks like editing files using regular expressions and running multiple commands simultaneously. This direct approach saves time and streamlines file operations.

Commands like `touch` and `mkdir` facilitate the creation of empty files and directories respectively, with the option `-p` enabling the creation of parent directories recursively. Additionally, files and directories can be created in specific paths using the dot (`.`) to

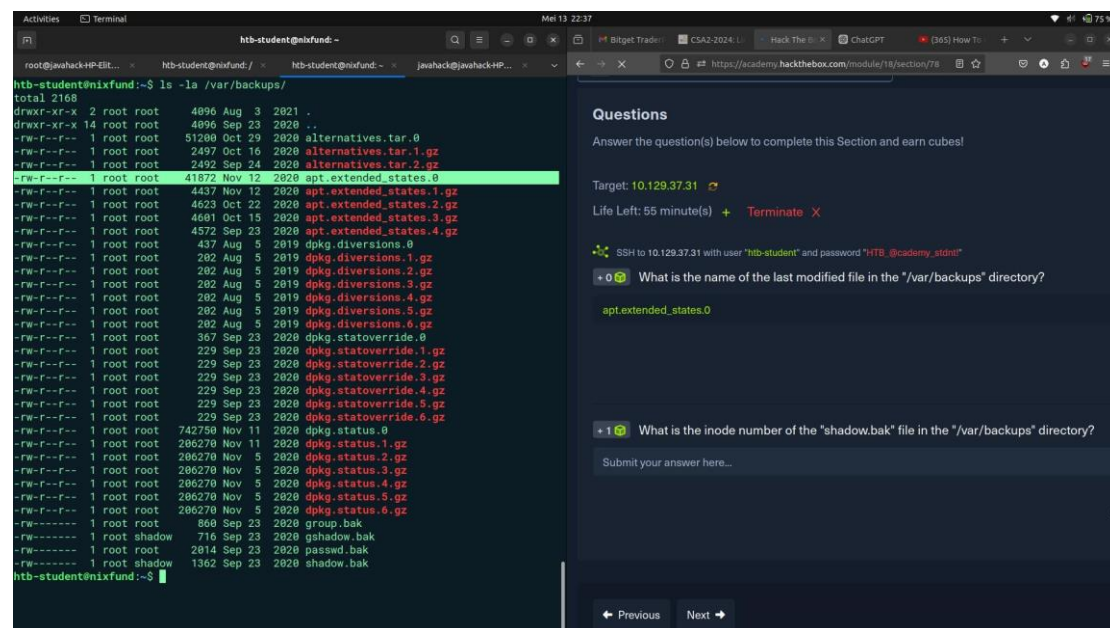
cs-sa07-24019

John Mutave

denote the current directory. The mv command allows for both moving and renaming files and directories, while the cp command facilitates file and directory copying.

Efficient file organization is achieved through these commands, allowing users to manage their file systems effectively. This report underscores the foundational understanding of file and directory operations in Linux, setting the stage for exploring more advanced techniques like redirects and text editors in subsequent learning.

Working with Files and Directories



```
htb-student@nixfund:~$ ls -la /var/backups/
total 2168
drwxr-xr-x  2 root root   4096 Aug  3  2021 .
drwxr-xr-x 14 root root   4096 Sep 23  2020 ..
-rw-r--r--  1 root root 51200 Oct 29  2020 alternatives.tar.0
-rw-r--r--  1 root root 2497 Oct 16  2020 alternatives.tar.1.gz
-rw-r--r--  1 root root 2497 Sep 24  2020 alternatives.tar.2.gz
-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 4623 Oct 22  2020 apt.extended_states.2.gz
-rw-r--r--  1 root root 4681 Oct 15  2020 apt.extended_states.3.gz
-rw-r--r--  1 root root 4572 Sep 23  2020 apt.extended_states.4.gz
-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
-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 742758 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  868 Sep 23  2020 group.bak
-rw-r--r--  1 root shadow  716 Sep 23  2020 gshadow.bak
-rw-r--r--  1 root root  2814 Sep 23  2020 passwd.bak
-rw-r--r--  1 root shadow 1362 Sep 23  2020 shadow.bak
htb-student@nixfund:~$
```

In Linux, file management diverges significantly from Windows, predominantly through the use of the terminal. While Windows relies on graphical interfaces like Explorer, Linux leverages command-line access, enabling efficient file manipulation and editing. The terminal allows users to directly interact with files using commands, providing speed and flexibility, especially when employing tools like regular expressions (regex) for selective modifications and running multiple commands concurrently.

Commands such as touch and mkdir facilitate the creation of files and directories respectively, with the -p option in mkdir allowing for the creation of parent directories. Moreover, files can be directly created within specific directories by specifying paths, enhancing organizational efficiency. The mv command handles both renaming and moving files and directories, while cp enables file and directory duplication.

cs-sa07-24019

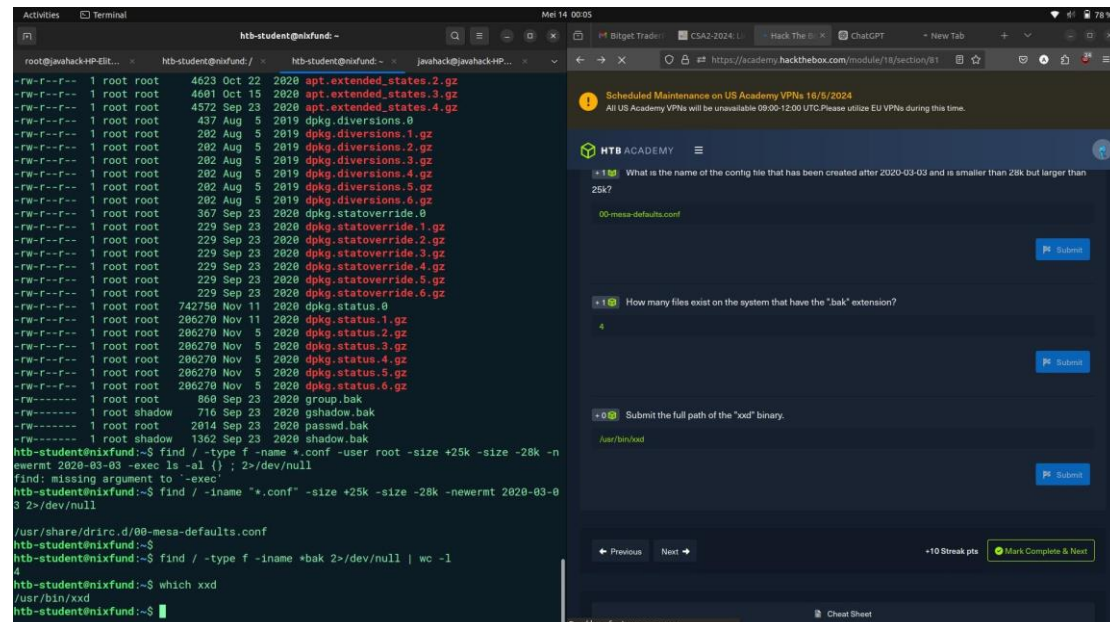
John Mutave

Efficient file organization and manipulation are further illustrated through examples of creating, renaming, moving, and copying files and directories. These operations demonstrate the command-line's prowess in managing file systems swiftly and effectively, showcasing the foundational skills essential for proficient Linux usage.

Finding Files and Directories in Linux

Locating files and directories efficiently is crucial for effective system management in Linux. Two primary tools, 'which' and 'find', aid in this endeavor. 'Which' identifies executable paths, facilitating program verification, while 'find' offers extensive filtering options for precise file and directory searches.

Additionally, the 'locate' command provides rapid search capabilities by leveraging a local database, albeit with fewer filtering options compared to 'find'. Users can update the database using 'updatedb' for accuracy. Choosing between 'find' and 'locate' depends on the balance needed between search speed and filtering flexibility, ensuring streamlined file and directory management in Linux environments.



The image shows a terminal window on the left and a web browser on the right. The terminal window is running a series of commands to find files and directories. The web browser shows a CTF challenge page from HTB Academy.

```
root@java8ack-HP-Elite:~# find / -type f -name *.conf -user root -size +25k -size -28k -newermt 2020-03-03 -exec ls -al {} \; 2>/dev/null
find: missing argument to '-exec'
htb-student@nixfund:~$ find / -iname "*.conf" -size +25k -size -28k -newermt 2020-03-03 2>/dev/null
/usr/share/doc/drirc.d/00-mesa-defaults.conf
htb-student@nixfund:~$ find / -type f -iname *.bak 2>/dev/null | wc -l
4
htb-student@nixfund:~$ which xxd
/usr/bin/xxd
htb-student@nixfund:~$
```

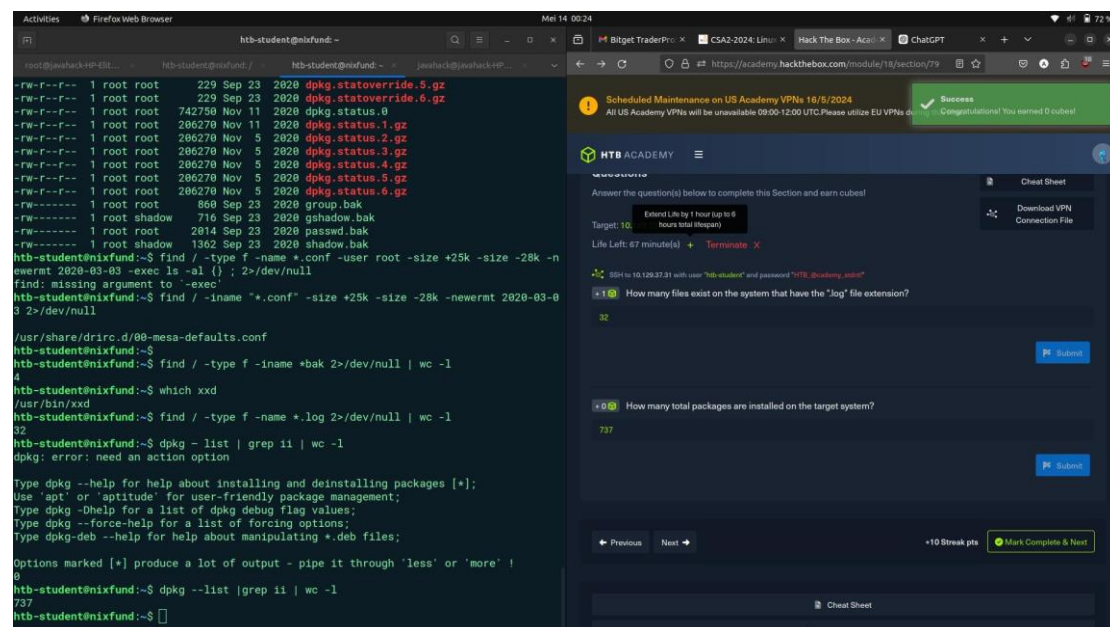
The web browser shows a CTF challenge page from HTB Academy. The challenge is titled "What is the name of the config file that has been created after 2020-03-03 and is smaller than 28k but larger than 25k?". The user has entered "00-mesa-defaults.conf" and submitted it. The challenge is marked as complete and the user has earned 10 Break pts.

File Descriptors and Redirections in Linux:

cs-sa07-24019
John Mutave

In Linux, file descriptors (FD) serve as connections managed by the kernel for Input/Output (I/O) operations. Key descriptors include STDIN (0) for input, STDOUT (1) for output, and STDERR (2) for error output. Redirection operations, such as redirecting STDERR to `/dev/null` to suppress errors or redirecting STDOUT to files, allow users to control data flow and manage outputs effectively.

Additionally, pipes (`|`) enable the chaining of commands, directing STDOUT from one program to another for data processing. By understanding and utilizing file descriptors and redirection techniques, users can efficiently manipulate data streams, optimize workflows, and streamline data processing tasks in Linux environments.



The image shows a dual-pane view. The left pane is a terminal window with the following commands and output:

```
root@kali:~# find / -type f -name *.conf -size +25k -size -28k -newermt 2020-03-03 2>/dev/null
find: missing argument to '-exec'
htb-student@nixfund:~$ find / -iname "*.conf" -size +25k -size -28k -newermt 2020-03-03 2>/dev/null
/usr/share/doc/ncurses/terminfo.d/00-mesa-defaults.conf
htb-student@nixfund:~$ find / -type f -iname *.bak 2>/dev/null | wc -l
4
htb-student@nixfund:~$ which xxd
/usr/bin/xxd
htb-student@nixfund:~$ find / -type f -name *.log 2>/dev/null | wc -l
32
htb-student@nixfund:~$ dpkg --get-selections | grep ii | wc -l
737
htb-student@nixfund:~$ dpkg --get-selections | grep ii | wc -l
737
```

The right pane shows the HTB Academy website interface. It displays a progress bar for 'HTB Academy' and a list of exercises. The first exercise is 'How many files exist on the system that have the ".log" file extension?' with a target of 10 and a life left of 67 minutes. The second exercise is 'How many total packages are installed on the target system?' with a target of 737. Both exercises are marked as 'Complete'.

Filtering Contents in Linux:

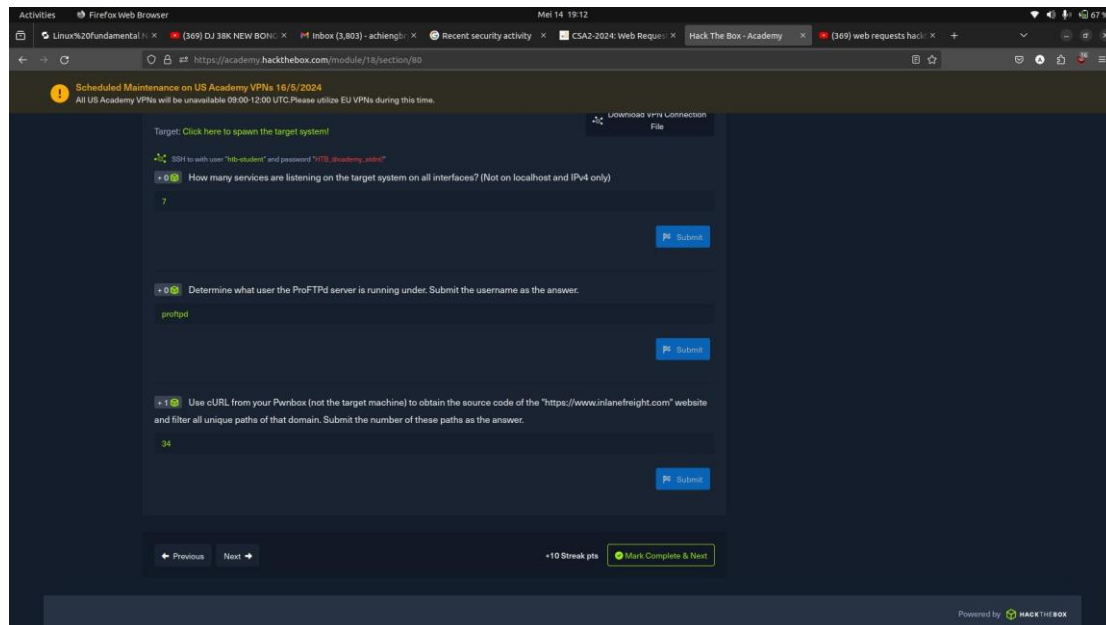
In Linux, tools like `more`, `less`, `head`, and `tail` offer interactive ways to view file contents. `more` and `less` provide scrolling functionality, while `head` displays the beginning of a file, and `tail` shows the end. These utilities facilitate easy navigation and reading of file contents without the need for an editor.

Furthermore, commands like `grep`, `cut`, `tr`, `column`, `awk`, and `sed` allow users to filter and manipulate text efficiently. `grep` searches for specific patterns, `cut` extracts specific fields, and `tr` performs character-level transformations. `column` formats data into

cs-sa07-24019

John Mutave

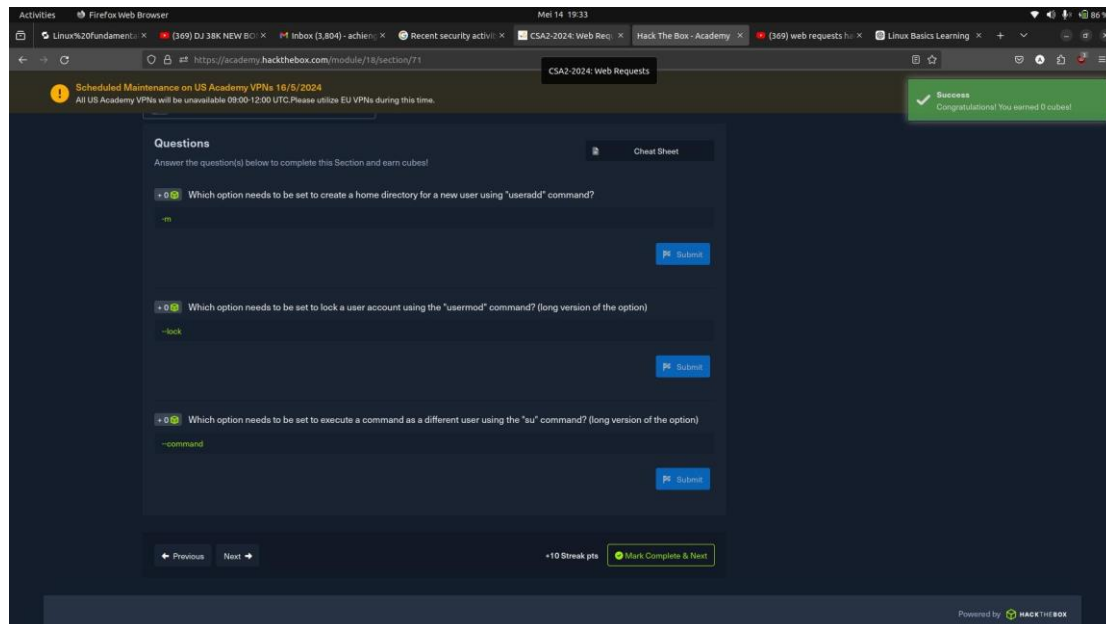
columns, awk processes text based on patterns, and sed performs text substitutions. Additionally, wc is useful for counting lines, words, or characters. Regular practice with these tools enhances familiarity and proficiency, enabling users to perform diverse text processing tasks effortlessly.



Permission Management in Linux

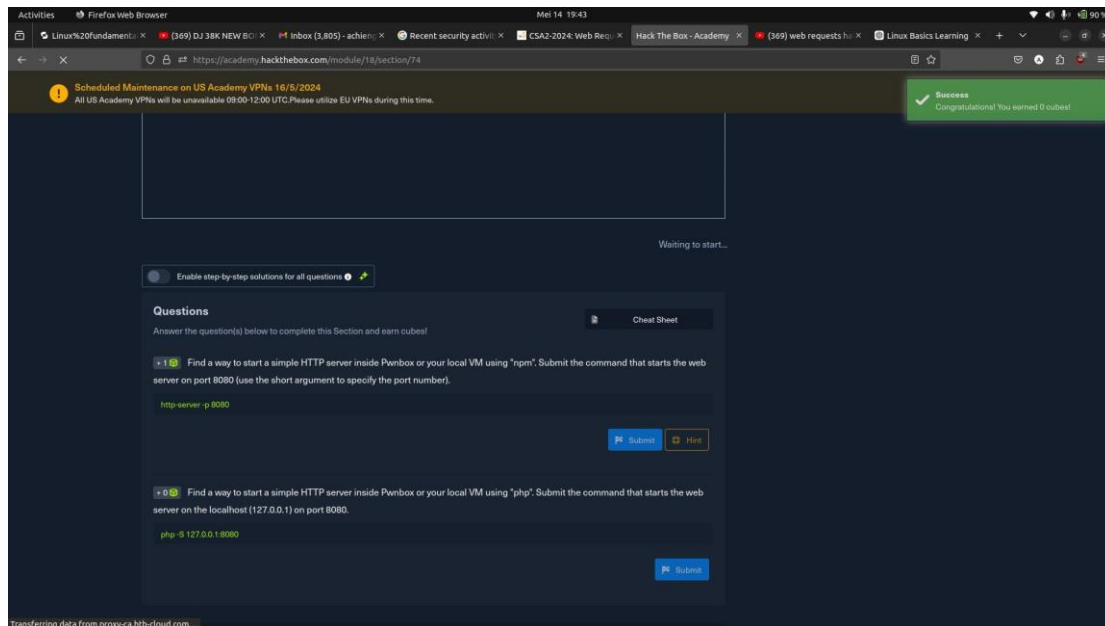
In Linux systems, permissions govern access to files and directories, ensuring security and control over data. Each file and directory is associated with specific users and groups, with permissions defining who can read, write, and execute them. These permissions, represented in the octal number system, encompass read (r), write (w), and execute (x) privileges for the owner, group, and others. Managing permissions is facilitated through commands like chmod, allowing users to modify access levels by adding or removing permissions as needed.

Beyond basic permissions, Linux offers advanced features like Set User ID (SUID), Set Group ID (SGID), and sticky bits. SUID and SGID allow users to execute programs with elevated privileges, while the sticky bit restricts file deletion and renaming within directories, ensuring that only authorized individuals can modify content. Understanding and effectively managing permissions are vital for maintaining system integrity and safeguarding sensitive data in Linux environments.



Working with Web Services

Working with web services in Linux involves setting up web servers like Apache, commonly installed using commands such as `apt install apache2 -y`, which installs Apache on the system. Apache allows for dynamic web page creation using scripting languages like PHP, Perl, and Python. Tools like `curl` and `wget` facilitate remote communication with web servers, enabling file transfer and website testing directly from the terminal. Python 3 can also serve as a web server using `python3 -m http.server`, where requests made to the server are logged, providing insight into the communication process.



Linux Security

Linux security is paramount for any system administrator, especially when managing internet-facing web servers. Keeping the operating system and packages up to date, configuring firewall rules using iptables, and securing SSH by disabling password login and root access are crucial steps. Implementing the principle of least privilege for user access, using tools like fail2ban to prevent brute-force attacks, and conducting periodic system audits for vulnerabilities are also essential practices. Advanced security measures like SELinux or AppArmor provide granular access controls, while utilities such as Snort, chkrootkit, and Lynis enhance security further. Additionally, TCP wrappers offer an additional layer of security by controlling access to services based on hostname or IP address, using configuration files like `/etc/hosts.allow` and `/etc/hosts.deny` to specify access permissions. However, it's vital to remember that security is an ongoing process, requiring constant vigilance and adaptation to emerging threats.

Firewall Setup

Firewalls are vital for controlling and monitoring network traffic, and Linux offers robust firewall capabilities through tools like iptables. These tools filter incoming and

cs-sa07-24019

John Mutave

outgoing traffic based on rules, protocols, and ports, safeguarding against unauthorized access and security threats. iptables, a powerful command-line tool, became a standard for Linux firewall management, offering flexibility in creating rules to counter various security risks like denial-of-service attacks and intrusion attempts. In addition to iptables, alternative solutions like nftables, ufw, and firewalld provide modern syntax, user-friendly interfaces, and dynamic firewall management capabilities. Understanding key components like tables, chains, rules, matches, and targets is essential for configuring effective firewall setups. For instance, creating rules with iptables involves specifying criteria (matches) such as protocol, port, source/destination IP addresses, and actions (targets) like ACCEPT, DROP, or REJECT. Furthermore, conducting practical exercises, like blocking and allowing specific ports or IP addresses, and managing rule lists, enhances proficiency in firewall administration and network security on Linux systems. Shareable link: <https://academy.hackthebox.com/achievement/1296187/18>