

# Lab 2.1: Linux File System

By: Nathan Metens ([metens@pdx.edu](mailto:metens@pdx.edu))

<b>Task 1: TryHackMe Linux Tutorial</b>	<b>1</b>
<b>Task 2:   (the pipe character), head</b>	<b>2</b>
<b>Task 3: grep</b>	<b>2</b>
<b>Task 4: Matching multiple instances</b>	<b>3</b>
<b>Task 5: Matching wildcards</b>	<b>4</b>
<b>Task 6: find</b>	<b>6</b>
<b>Task 7: Using -exec and xargs</b>	<b>7</b>
<b>Task 8: Tampering with find</b>	<b>7</b>
<b>Task 9: Polyglot files</b>	<b>8</b>
<b>Task 10: file and polyglots</b>	<b>9</b>
<b>Task 11: TryHackMe Linux file system forensics</b>	<b>10</b>

## Task 1: TryHackMe Linux Tutorial

This TryHackMe room was mostly review, although I found out the reason why `find` never worked for me whenever I tried to use it in the past! I'd always do `find file.txt` and it would never find it, even though the file was right there in the directory. The problem was that I always forgot the -name flag. After testing on my Mac, it turns out that I also need to include the directory... So `find . -name file.txt` would work. Pretty cool.

The screenshot shows the TryHackMe interface for the 'Linux Fundamentals Part 1' room. On the left, there's a summary bar with a penguin icon, points earned (88), completed tasks (9), room type (Walkthrough), difficulty (Info), streak (1), and a note that the room counted toward joining the league. In the center, a list of tasks is shown, all of which have been completed (indicated by green checkmarks). The tasks are:

- Task 1: Introduction
- Task 2: A Bit of Background on Linux
- Task 3: Interacting With Your First Linux Machine (In-Browser)
- Task 4: Running Your First few Commands
- Task 5: Interacting With the Flesystem!
- Task 6: Searching for Files
- Task 7: An Introduction to Shell Operators
- Task 8: Conclusions & Summaries
- Task 9: Linux Fundamentals Part 2

A progress bar at the top indicates "Room completed (100%)". On the right, a terminal window titled "Metens" shows the command "June 27, 2025 at 2:53PM" and the user "Metens".

## Task 2: | (the pipe character), head

`ls -l /var/log | grep auth` returned nothing in my Kali VM. So piping passes the output of one command as the input to another command. And then grep searches for a certain word or pattern and only outputs the lines containing this pattern or word. The -v flag for grep does everything except what comes after the -v part (inverting). With head, we can do it alone (first 10 lines) or we can specify a number with fewer lines using the -n flag.

```
drwxr-x--- 2 root      adm          4096 Apr 18 03:02 samba
drwx----- 2 speech-dispatcher root        4096 May  6 12:23 speech-dispatcher
drwxr-xr-x  2 stunnel4   stunnel4     4096 Jun 25 21:24 stunnel4
drwxr-xr-x  2 root      root         4096 Apr  7 14:15 sysstat
-rw-rw-r--  1 root      utmp        800  Jun 25 21:30 wtmp
-rw-r--r--  1 root      root        36308 Jun 30 14:55 Xorg.0.log

[metens@cs591-metens-kali-1] ~
$ ls -l /var/log | head -n 5
total 1560
-rw-r--r-- 1 root      root        86079 Jun 25 21:29 alternatives.log
drwxr-x--- 2 root      adm          4096 Jun 25 21:26 apache2
drwxr-xr-x  2 root      root        4096 Jun 25 21:29 apt
-rw----- 1 root      root         0 Jun 30 14:33 boot.log

[metens@cs591-metens-kali-1] ~
$ ls -l /var/log | grep -v root
total 1560
drwxr-xr-x  2 _gvm      _gvm        4096 May 19 05:04 gvm
drwx----- 3 inetsim   inetsim     4096 Jun 25 21:26 inetsim
drwxr-xr-x  2 _gvm      _gvm        4096 Mar 25 01:22 notus-scanner
drwxr-s---  2 redis     adm        4096 Jun 25 21:24 redis
drwxr-xr-x  2 stunnel4  stunnel4    4096 Jun 25 21:24 stunnel4

[metens@cs591-metens-kali-1] ~
$ ls -l /var/log | head
```

## Task 3: grep

```
metens@cs591-metens-kali-1: ~
File Actions Edit View Help

[metens@cs591-metens-kali-1] ~
$ cat /usr/share/wordlists/rockyou.txt | grep -E "^\w{3}" | head -n 3
abc123
abcdef
abcdefg

[metens@cs591-metens-kali-1] ~
$ cat /usr/share/wordlists/rockyou.txt | grep -E "def\$" | head -n 3
abcdef
fresh2def
sosodef
```

Since I wanted to check the entire rockyou file, I started with cat to output the entire file, then passed that output as input to the extended regex grep command, then output only the first 3 values from that entire list of passwords. The only thing changed from that was `def\$` to see the first three passwords that ended in “def” for the second command. Super cool!

## Task 4: Matching multiple instances

```
metens@cs591-metens-kali-1: ~
File Actions Edit View Help

[metens@cs591-metens-kali-1: ~]
$ cat /usr/share/wordlists/rockyou.txt | grep -E "^\w{6}def$"
abcdef
abcde

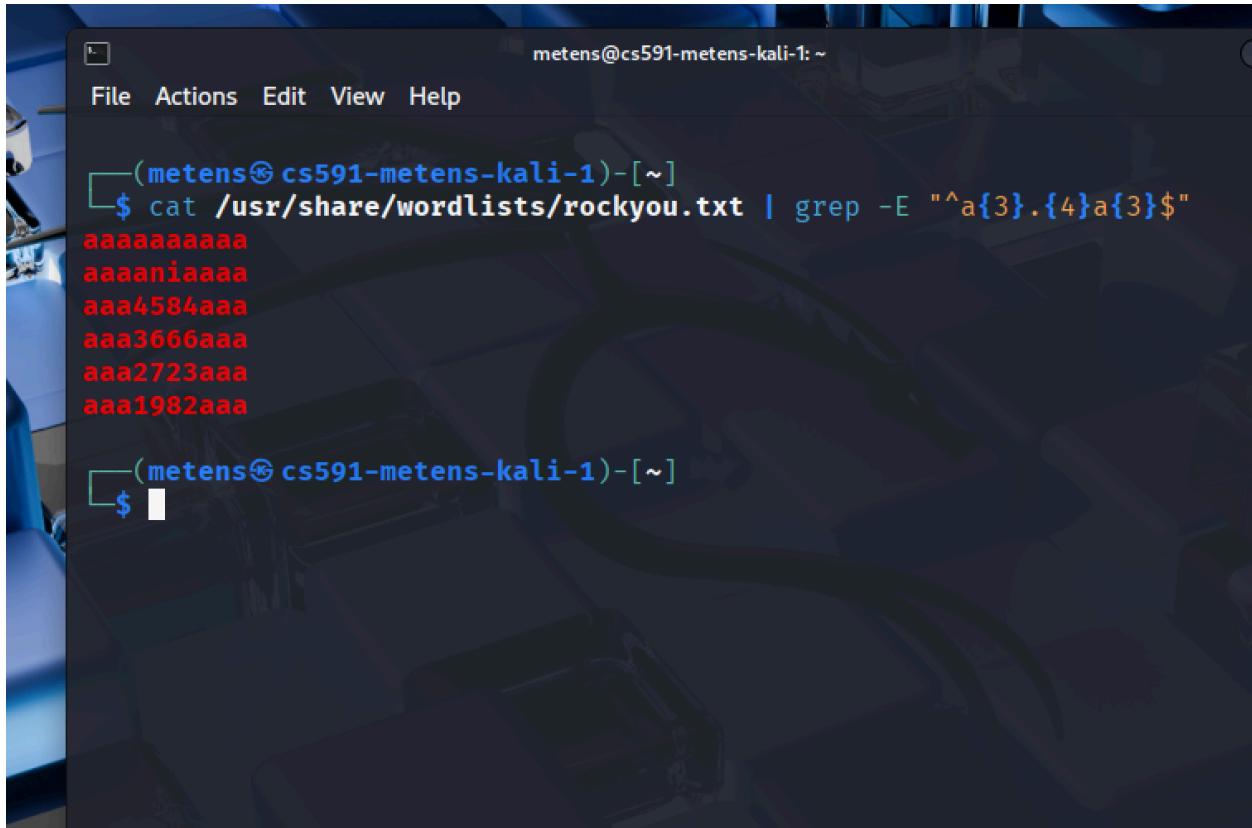
[metens@cs591-metens-kali-1: ~]
$ cat /usr/share/wordlists/rockyou.txt | grep -E "^\w{6}def+$"
abcdef
abcdeff

[metens@cs591-metens-kali-1: ~]
$ cat /usr/share/wordlists/rockyou.txt | grep -E "^\w{6}def*$"
abcdef
abcde
abcdeff
```

There are a few operators that determine where in the regular expression the characters are found and how many occurrences are requested. Having a `?` means zero or one occurrence, `+` means one or more, and `\*` means zero or more. Then the braces can hold a certain number of characters requested to search. Handy dandy!

## Task 5: Matching wildcards

I created a regex with 3 letters of a at the start, then any characters in the middle (only 4), and then 3 a's at the end...

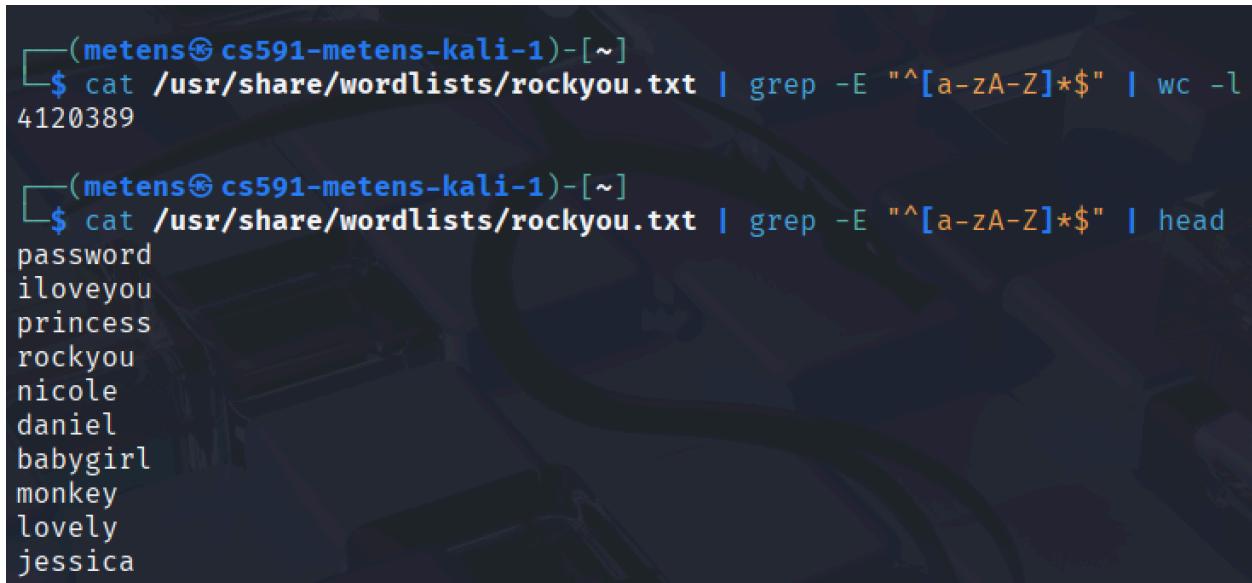


```
metens@cs591-metens-kali-1: ~
File Actions Edit View Help

└─(metens@cs591-metens-kali-1)-[~]
└─$ cat /usr/share/wordlists/rockyou.txt | grep -E "^(aaa){3}(.{4})aaa$"
aaaaaaaaaa
aaaaniaaaa
aaa4584aaa
aaa3666aaa
aaa2723aaa
aaa1982aaa

└─(metens@cs591-metens-kali-1)-[~]
└─$
```

The total number of passwords that have only alphabetical letters is more than 4,00,000:



```
metens@cs591-metens-kali-1: ~
└─$ cat /usr/share/wordlists/rockyou.txt | grep -E "^[a-zA-Z]*$" | wc -l
4120389

└─(metens@cs591-metens-kali-1)-[~]
└─$ cat /usr/share/wordlists/rockyou.txt | grep -E "^[a-zA-Z]*$" | head
password
iloveyou
princess
rockyou
nicole
daniel
babygirl
monkey
lovely
jessica
```

The total number of passwords with exactly 6 digits can be found like this: start with any digit and end with any digit, and make sure to count only 6 total digits:

```
└─(metens㉿cs591-metens-kali-1)~
└─$ cat /usr/share/wordlists/rockyou.txt | grep -E "^[0-9]{6}$" | wc -l
390529

└─(metens㉿cs591-metens-kali-1)~
└─$ cat /usr/share/wordlists/rockyou.txt | grep -E "^[0-9]{6}$" | head
123456
654321
111111
000000
123123
666666
121212
112233
555555
789456
```

The non-alphanumeric passwords:

```
└─(metens㉿cs591-metens-kali-1)~
└─$ cat /usr/share/wordlists/rockyou.txt | grep -E "^[^a-zA-Z0-9]*$" | wc -l
5421

└─(metens㉿cs591-metens-kali-1)~
└─$ cat /usr/share/wordlists/rockyou.txt | grep -E "^[^a-zA-Z0-9]*$" | head
*****
.....
!@#$%^
*****
@#@#@#
*****
$$$$$$
*****
*****
```

The password that starts with o, has a “>” in the middle, and ends with 4 numbers is:

```
└─(metens㉿cs591-metens-kali-1)~
└─$ cat /usr/share/wordlists/rockyou.txt | grep -E "^o.*>.*[0-9]{4}$" | head
omi>_<0309
```

## Task 6: find

Using find to find all files created after today's date:

```
(metens@cs591-metens-kali-1)~]$ sudo touch /etc/cron.daily/backdoor.sh  
[sudo] password for metens:  
  
(metens@cs591-metens-kali-1)~]$ find /etc -newermt 2025-06-30  
/etc  
find: '/etc/redis': Permission denied  
find: '/etc/credstore.encrypted': Permission denied  
find: '/etc/ssl/private': Permission denied  
find: '/etc/ipsec.d/private': Permission denied  
/etc/cron.daily  
/etc/cron.daily/backdoor.sh  
/etc/resolv.conf  
find: '/etc/credstore': Permission denied  
find: '/etc/openvas/gnupg': Permission denied  
find: '/etc/polkit-1/rules.d': Permission denied  
find: '/etc/vpnc': Permission denied
```

Seeing the difference between files changed within the last 5 minutes versus the last day:

```
(metens@cs591-metens-kali-1)~]$ find /etc -mmin -5 > /tmp/1.txt  
find: '/etc/redis': Permission denied  
find: '/etc/credstore.encrypted': Permission denied  
find: '/etc/ssl/private': Permission denied  
find: '/etc/ipsec.d/private': Permission denied  
find: '/etc/credstore': Permission denied  
find: '/etc/openvas/gnupg': Permission denied  
find: '/etc/polkit-1/rules.d': Permission denied  
find: '/etc/vpnc': Permission denied  
  
(metens@cs591-metens-kali-1)~]$ find /etc -mtime -1 > /tmp/2.txt  
find: '/etc/redis': Permission denied  
find: '/etc/credstore.encrypted': Permission denied  
find: '/etc/ssl/private': Permission denied  
find: '/etc/ipsec.d/private': Permission denied  
find: '/etc/credstore': Permission denied  
find: '/etc/openvas/gnupg': Permission denied  
find: '/etc/polkit-1/rules.d': Permission denied  
find: '/etc/vpnc': Permission denied  
  
(metens@cs591-metens-kali-1)~]$ diff /tmp/1.txt /tmp/2.txt  
0a1,2  
> /etc  
> /etc/cron.daily  
1a4  
> /etc/resolv.conf
```

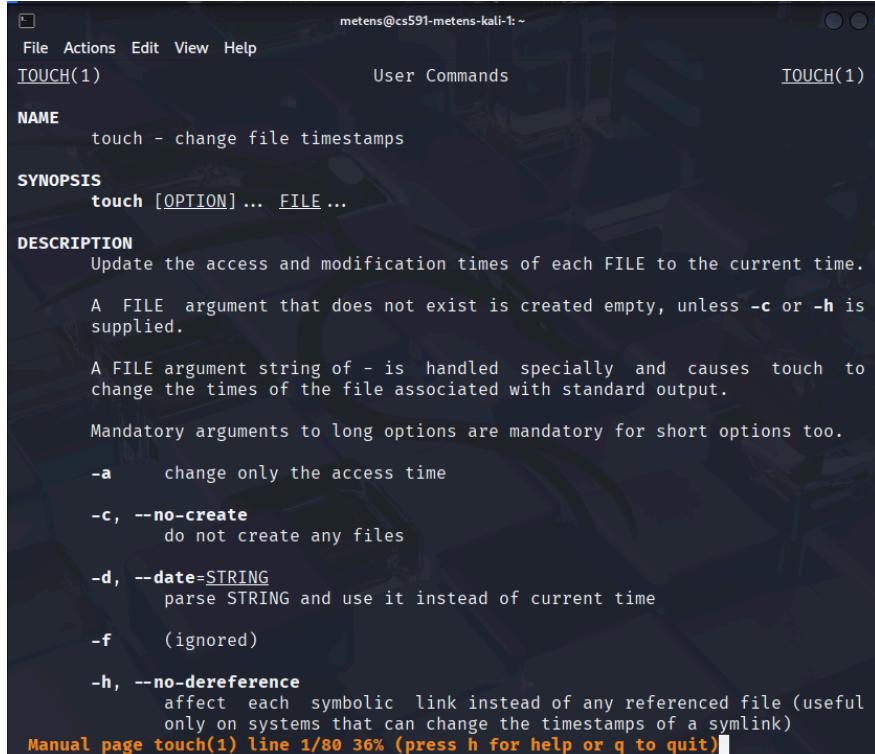
## Task 7: Using -exec and xargs

Showing all the files that have been output within the last day using xargs to `ls -l`:

```
(metens@cs591-metens-kali-1)~]$ find /etc -type f -mtime -1 -exec sha256sum {} \\;
find: '/etc/redis': Permission denied
find: '/etc/credstore.encrypted': Permission denied
find: '/etc/ssl/private': Permission denied
find: '/etc/ipsec.d/private': Permission denied
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855 /etc/cron.daily/
backdoor.sh
ace46997caf3a156480c1ca5359c30d3511f344c46cf1f9007f6f599adc7a80f /etc/resolv.conf
find: '/etc/credstore': Permission denied
find: '/etc/openvas/gnupg': Permission denied
find: '/etc/polkit-1/rules.d': Permission denied
find: '/etc/vpnc': Permission denied

(metens@cs591-metens-kali-1)~]$ find /etc -type f mtime -1 | xargs ls -l
find: paths must precede expression: `mtime'
total 32
drwxr-xr-x 2 metens metens 4096 Jun 25 21:30 Desktop
drwxr-xr-x 2 metens metens 4096 Jun 25 21:30 Documents
drwxr-xr-x 2 metens metens 4096 Jun 26 07:44 Downloads
drwxr-xr-x 2 metens metens 4096 Jun 25 21:30 Music
drwxr-xr-x 2 metens metens 4096 Jun 25 21:30 Pictures
drwxr-xr-x 2 metens metens 4096 Jun 25 21:30 Public
drwxr-xr-x 2 metens metens 4096 Jun 25 21:30 Templates
drwxr-xr-x 2 metens metens 4096 Jun 25 21:30 Videos
```

## Task 8: Tampering with find



It looks like to change the access time of a file, the `-a` flag should be used.

To change the last modification time of the file, the ` -m` flag should be used. The ` --date=STRING` flag allows us to change the date and modify the times.

```
(metens@cs591-metens-kali-1)~]$ sudo touch --date="2024-06-30" /etc/cron.daily/backdoor.sh  
(metens@cs591-metens-kali-1)~]$ ls -l /etc/cron.daily/backdoor.sh  
-rw-r--r-- 1 root root 0 Jun 30 2024 /etc/cron.daily/backdoor.sh  
(metens@cs591-metens-kali-1)~]$ find /etc -mmin -30 -exec ls -ld {} \;  
find: '/etc/redis': Permission denied  
find: '/etc/credstore.encrypted': Permission denied  
find: '/etc/ssl/private': Permission denied  

```

Changed the timestamp of the [backdoor.sh](#) file to one year ago on this date.

## Task 9: Polyglot files

The author of the two polyglot files:

- <https://web.cecs.pdx.edu/~rchaney/Classes/cs491/polyglot/GIF+PDF.gif>
- <https://web.cecs.pdx.edu/~rchaney/Classes/cs491/polyglot/GIF+PDF.pdf>

Jesse Chaney.

Now, running the `curl -IL` command on both the above files:

```
metens@cs591-metens-kali-1:~]$ curl -IL https://web.cecs.pdx.edu/~rchaney/Classes/cs491/polyglot/GIF+PDF.gif  
HTTP/1.1 200 OK  
Date: Tue, 01 Jul 2025 06:52:03 GMT  
Server: Apache  
Last-Modified: Sun, 29 Jun 2025 00:06:23 GMT  
ETag: "ed96d-638aaae82c7f9"  
Accept-Ranges: bytes  
Content-Length: 973165  
Connection: close  
Content-Type: image/gif  
  
(metens@cs591-metens-kali-1)~]$ curl -IL https://web.cecs.pdx.edu/~rchaney/Classes/cs491/polyglot/GIF+PDF.pdf  
HTTP/1.1 200 OK  
Date: Tue, 01 Jul 2025 06:52:10 GMT  
Server: Apache  
Last-Modified: Sun, 29 Jun 2025 00:06:23 GMT  
ETag: "ed96d-638aaae82c7f9"  
Accept-Ranges: bytes  
Content-Length: 973165  
Connection: close  
Content-Type: application/pdf  
  
(metens@cs591-metens-kali-1)~]$
```

**Identical values from both files:**

- Server
- ETag
- Accept-Ranges
- Content-Length
- Connection

**Difference between the files:**

- Content-Type. One is image/gif, the other is application/pdf

The web server uses a file extension heuristic to determine the content type of each file. The file that ends in PDF is considered a PDF, and the one with a GIF extension is considered an image by the server.

## Task 10: file and polyglots

Used curl to grab the GIF image and save it to the Kali VM. Then used file xxd to dump the contents to see what type of file it is. Then, moved it to another name and used file to check the type, which uses content-based heuristics instead of file-extension heuristics to determine the file type, regardless of the file's name or extension.

```
metens@cs591-metens-kali-1: ~
File Actions Edit View Help
└$ ls
Desktop Downloads Pictures Public Videos
Documents Music polyglot.gif Templates

[(metens@cs591-metens-kali-1)-[~]
└$ file polyglot.gif
polyglot.gif: GIF image data, version 89a, 500 x 375

[(metens@cs591-metens-kali-1)-[~]
└$ xxd polyglot.gif | head -2
00000000: 4749 4638 3961 f401 7701 841f 00a2 a2a2  GIF89a ..w.....
00000010: 1e1e 1e63 6363 0000 002e 2e2e cece ce1f ...ccc.....
[(metens@cs591-metens-kali-1)-[~]
└$ mv polyglot.gif polyglot.pdf

[(metens@cs591-metens-kali-1)-[~]
└$ ls
Desktop Documents Downloads Music Pictures polyglot.pdf Public Templates

[(metens@cs591-metens-kali-1)-[~]
└$ file !$
[(metens@cs591-metens-kali-1)-[~]
└$ file polyglot.gif
polyglot.gif: cannot open `polyglot.gif' (No such file or directory)

[(metens@cs591-metens-kali-1)-[~]
└$ file polyglot.pdf
polyglot.pdf: GIF image data, version 89a, 500 x 375
```

Next, used hexeditor to change the filetype of the polyglot.gif into a PDF with the first 5 bytes of the hex data, which corresponds to the file type:

The screenshot shows a hex editor window titled 'metens@cs591-metens-kali-1: ~'. The file being edited is 'polyglot.pdf'. The ASCII view shows the first few bytes of the file, including the PDF header '%PDF-1.3'. The bytes are displayed in pairs of hex values followed by their ASCII representation.

Offset	Hex	ASCII
00000000	25 50 44 46	%PDF
00000001	2D 01 F4 01	-1.3
00000002	77 01 84 1F	
00000003	00 A2 A2 A2	
00000004	CE CE CE 1F	
00000005	.. ccc.....	
00000006	20 1F 22 22	.. "" .. .
00000007	22 1F 1E 1F	.....} };;;
00000008	20 10 10 10	... LLL.....XXX@
00000009	0C 0C 0C 07	@000666.....TT
0000000A	07 07 7D 7D	T.....! ..
0000000B	7D 3B 3B 3B	NETSCAPE2.0.....
0000000C	82 82 82 4C	!.1%PDF-1.3.1502
0000000D	4C 4C AD AD	0 obj.<<./Lengt
0000000E	AD C1 C1 C1	h 937180>>.st
0000000F	78 78 78 40	ream.!.....,..
00000010	6F 36 36 36	....w....` .di..
00000011	17 17 17 89	.....6..k....9
00000012	89 89 54 54	.3C/0.D ... .....
00000013	B5 B5 FF FF	....LK...R@.CD,
00000014	FF 21 FF 0B	
00000015	00 00 00 00	
00000016	32 2E 30 03	
00000017	01 00 00 00	
00000018	21 FE 31 25	
00000019	50 44 46 2D	
0000001A	31 2E 33 0A	
0000001B	31 35 30 32	
0000001C	62 6A 0A 3C	
0000001D	3C 0A 2F 4C	
0000001E	65 6E 67 74	
0000001F	38 30 0A 3E	
00000020	3E 0A 73 74	
00000021	72 65 61 6D	
00000022	00 21 F9 04	
00000023	04 03 00 FF	
00000024	00 2C 00 00	
00000025	00 00 F4 01	
00000026	77 01 00 05	
00000027	FE 60 20 8E	
00000028	64 69 8E 19	
00000029	E4 11 D9 E9	
0000002A	BA C3 36 0D	
0000002B	CB 6B 07 CD	
0000002C	E6 D1 F7 39	
0000002D	08 33 43 2F	
0000002E	30 E0 44 06	
0000002F	09 D2 20 C2	
00000030	19 94 96 05	
00000031	80 52 A0 40	
00000032	90 43 44 A7	

Then, after re-typing `file polyglot.pdf`, we can see that the type of the file has been changed:

The screenshot shows a terminal session on a Kali Linux machine. The user runs the command 'file polyglot.pdf' to check the file type, which is initially listed as 'GIF image data, version 89a, 500 x 375'. The user then uses the hexeditor command to open the file 'polyglot.pdf'. After editing, the user runs the 'file' command again, and it now correctly identifies the file as a 'PDF document, version a.\001, 1 page(s)'. The terminal session ends with a '\$' prompt.

```
(metens@cs591-metens-kali-1:~] $ file polyglot.pdf
polyglot.pdf: GIF image data, version 89a, 500 x 375

[metens@cs591-metens-kali-1:~] $ hexeditor polyglot.pdf

[metens@cs591-metens-kali-1:~] $ hexeditor polyglot.pdf

[metens@cs591-metens-kali-1:~] $ hexeditor polyglot.pdf

[metens@cs591-metens-kali-1:~] $ file polyglot.pdf
polyglot.pdf: PDF document, version a.\001, 1 page(s)

[metens@cs591-metens-kali-1:~] $
```

## Task 11: TryHackMe Linux file system forensics

Started by changing the path to a trusted USB source because an adversary could change commands to perform malicious behavior. Then, use find, exiftool, and stat to observe details about the attacker's steps. As well as learning about timestamps. This step took me about an hour; there was a lot of info, and it was quite cumbersome. The 'Users and Groups' part took me over 40 minutes to complete as well! I got stuck trying to figure out why sudo wasn't working for me when trying to view Jane's sudoers list. I re-read all the steps, tried searching the internet for help, navigating everywhere, finding Jane, and trying to sign in to her account. I even tried to restart the machine to see if there was a bug. I just didn't know what the password was for the investigator account. But then I realized that since it had been over 2 hours since I had read the first step, I totally forgot that the password for the investigator account was right there:

TryHackMe123!. So then I was finally able to finish the task and move on to the next one. The next step was about hidden files and investigating the hidden .ssh directory. I was able to find the flag in the hidden files by looking at their sizes. All of the hidden files had a size of zero,

except one of them, which contained the THM code to pass this step. Then I learned about rootkits, which are a way to get root privileges. To check for rootkits, we can use either rkhunter or chkrootkit.

The screenshot shows the TryHackMe interface after completing the 'Linux File System Analysis' room. At the top right, the user's profile is shown with the name 'metens [0x2][APPRENTICE]', a rank of 1199687, a streak of 2, and 4 completed rooms. A badge for 'Linux File System Analysis' is circled with a black oval. Below the profile, the room summary is displayed with the following details:

- Points earned: 112
- Completed tasks: 8
- Room type: Walkthrough
- Difficulty: Easy
- Streak: 2

The main content area shows the room completion message: "Congratulations on completing Linux File System Analysis!!! 🎉". Below this, there are four cards representing other rooms:

- Offensive Security Intro**: Hack your first website (legally in a safe environment) and experience an ethical hacker's job. Free, no walkthrough.
- OpenVPN**: A guide to connecting to our network using OpenVPN. Free, no walkthrough.
- Linux Fundamentals Part 1**: Embark on the journey of learning the fundamentals of Linux. Learn to run some of the first essential commands on an interactive terminal. Paid, no walkthrough.
- Linux File System Analysis**: Perform real-time file system analysis on a Linux system to identify an attacker's artefacts. Paid, no walkthrough.