

Name: Cristian Barreno

## Logging exercises fifth group

### Prerequisites

- Internet Connection
- Ubuntu Server 22.04
- SSH from your Host OS to your VM Linux Server
- Download the [passwords.txt](#) file from the BTA github to your host OS
- Get the [passwords.txt](#) file to your Linux server Home Folder

### Exercise 16

#### Task #1

[cat passwords.txt](#)



```
lynx@lynx: ~  
lynx@lynx:~$ cat passwords.txt  
19740707  
19740709  
19740710  
19740711  
19740712  
19740713  
19740714  
19740715  
19740716  
19740717  
197407170034  
19740718  
19740719  
19740720  
19740721  
19740723  
19740724  
19740725  
19740726  
19740727  
19740728  
19740729  
19740730  
19740731  
197408  
19740801  
19740802  
19740803  
19740804  
19740805  
19740806  
19740807  
19740808  
19740809  
19740810  
19740811  
19740812  
19740813  
19740814  
19740815  
197408156817
```

## Task #2

```
less passwords.txt
```

```
lynx@lynx:~$ less passwords.txt
```

```
# This is a list of 2,150,822 unique ASCII passwords in sorted order according  
# to their native byte values using UNIX sort command.  
  
# This list (also known as wordlist, password dictionary or password list)  
# is useful for password recovery tools such as John the Ripper, hashcat and  
# Aircrack-ng. To use this file, be sure to first remove these comment lines,  
# i.e. the lines starting with # character.  
  
# If you are looking for a better password dictionary,  
# see https://dazzlepod.com/uniqupass/  
  
# $DateTime: 2016/12/18 09:10:18 $  
#  
# Comments/Questions? Send to disclosure@dazzlepod.com  
#  
~  
!@###!'  
^  
^  
^AAAAAAAAA  
^%l))  
^%%^^  
^_  
^..  
^)%(*&  
^@##^++$  
^$^  
^*!)(($#  
^&*()  
^%$#@|  
~
```

### Task #3

```
fgrep "123456" passwords.txt
```

[illegible]

- What did that do?  
It gave us a big output of passwords and highlighted on red “123456”
- Explain what just happened.  
We used fgrep to find all the lines of a file that contain “123456” on passwords.txt.

### Exercise 17

## Task #1

```
vim passwords.txt
```



A terminal window with a black background and green text. The prompt is `lynx@lynx:~`. The command `lynx@lynx:~$ vim passwords.txt` has been entered. The terminal window has standard Linux window controls (minimize, maximize, close) in the top right corner.

Add “1234567890\_from\_first\_file” to the end of the file after the zzz not a new line.

[illegible]

## Task #2

- Create a copy of the first file, as passwords2.txt

```
lynx@lynx: ~  
lynx@lynx:~$ cp passwords.txt passwords2.txt  
lynx@lynx:~$
```

- Edit the second file.
  - Change the last line to “1234567890\_from\_second\_file”

[illegible]

### Task #3

```
fgrep 1234567890_ passw*
```

```
lynx@lynx:~$ fgrep 1234567890_ passw*
```

[illegible]

- What did that do?  
It gave us a short output from 3 different text files, and it showed us on which lines “1234567890\_” shows up. I can see those are the files that I modified on vim with insert mode.
- Explain what just happened.  
What happened was really interesting. We used fgrep to find all the lines that contain “1234567890\_”, and we used a wildcard at the end of passw. With \* we tell the server to look for “1234567890\_” in txt vfiles that start with passw.

## Exercise 18

### Task #1

**fgrep 1234567890\_ passwords.txt > passwords3.txt**

```
lynx@lynx: ~  
lynx@lynx:~$ fgrep 1234567890_ passwords.txt > passwords3.txt  
  
lynx@lynx:~$ ls -l  
total 59260  
-rw-rw-r-- 1 lynx lynx 23677 Mar 18 14:55 authy.log  
-rw-rw-r-- 1 lynx lynx 4240 Mar 17 17:19 catpictureess.jpg  
-rw-rw-r-- 1 lynx lynx 28490 Mar 17 18:11 ebill.txt  
-rw-rw-r-- 1 lynx lynx 12 Mar 17 16:51 file1.txt  
-rw-rw-r-- 1 lynx lynx 12 Mar 17 16:52 file2.txt  
-rw-rw-r-- 1 lynx lynx 13 Mar 17 16:58 file3.txt  
-rw-rw-r-- 1 lynx lynx 7120233 Nov 27 14:28 Linux64.zip  
-rw-rw-r-- 1 lynx lynx 22252 Mar 16 17:57 log1.txt  
-rw-rw-r-- 1 lynx lynx 9128 Mar 16 18:06 log2.txt  
-rw-rw-r-- 1 lynx lynx 29 Mar 17 15:45 malware.txt  
-rw-rw-r-- 1 lynx lynx 41 Mar 17 15:46 notmalwareoreally.txt  
-rw-rw-r-- 1 lynx lynx 20159804 Mar 19 01:01 passwords2.txt  
-rw-rw-r-- 1 lynx lynx 59 Mar 19 01:18 passwords3.txt  
-rw-rw-r-- 1 lynx lynx 20159803 Mar 18 15:03 passwords.txt  
-rwxr-xr-x 1 lynx lynx 12998261 Nov 27 14:27 vt  
-rw-rw-r-- 1 lynx lynx 56075 Mar 6 17:31 windows_activity_logs2.txt  
-rw-rw-r-- 1 lynx lynx 56075 Mar 6 17:13 windows_activity_logs.txt  
lynx@lynx:~$
```

- What did that do?  
It redirected the output of to **fgrep 1234567890\_ passwords.txt** to **passwords3.txt**
- Explain what just happened.  
> redirects the output of fgrep and sends it to a new txt file.

### Task #2

**cat passwords3.txt**

```
lynx@lynx: ~  
lynx@lynx:~$ fgrep 1234567890_ passwords.txt > passwords3.txt  
lynx@lynx:~$ cat passwords3.txt  
1234567890_  
ZZZZZZZZZZZZZZZZZZZZ1234567890_from_first_file  
lynx@lynx:~$
```

- What did that do?  
After I concatenate passwords3.txt I see a small output that shows on which lines in passwords .tx does “1234567890\_” shows up.
- Explain what just happened.  
We redirected **fgrep 1234567890\_ passwords.txt** to a new file.  
By using cat, we can see the contents of passwords3.txt.

## Task #3

**fgrep -c 1234567890\_ passw\***



```
lynx@lynx: ~  
lynx@lynx:~$ fgrep -c 1234567890_ passw*  
passwords2.txt:2  
passwords3.txt:2  
passwords.txt:2  
lynx@lynx:~$
```

- What did that do?  
It gave us a short output that includes a count of how many lines does “123456790\_” exist in those 3 txt files.
- Explain what just happened.  
We used fgrep to find all the lines that contain “1234567890\_”, and we used a wildcard at the end of passw. With \* we tell the server to look for “1234567890\_” in txt files that start with passw. And -c gives us an exact count.

## Task #4

**fgrep -w 1234567890\_ passw\***



```
lynx@lynx: ~  
lynx@lynx:~$ fgrep -w 1234567890_ passw*  
passwords2.txt:1234567890_  
passwords3.txt:1234567890_  
passwords.txt:1234567890_  
lynx@lynx:~$
```

- What did that do?  
It gave us a short output that includes on which lines does “123456790\_” literally exist in those 3 txt files.
- Explain what just happened.  
We used fgrep to find all the lines that contain “1234567890\_”, and we used a wildcard at the end of passw. With \* we tell the server to look for “1234567890\_” in txt files that start with passw. The -w option gives us exactly where “1234567890\_” is included as a separate “word” or string.

## Task #5

**fgrep -n 1234567890\_ passw\***

A terminal window showing the command 'fgrep -n 1234567890\_ passw\*' being executed. The output lists three files: passwords2.txt, passwords3.txt, and passwords.txt, each with a line number and the matching text. The line numbers are highlighted in green.

```
lynx@lynx: ~  
lynx@lynx:~$ fgrep -n 1234567890_ passw*  
passwords2.txt:126428:1234567890_  
passwords2.txt:2150838:ZZZZZZZZZZZZZZZZZZZZ1234567890_from_second_file  
passwords3.txt:1:1234567890_  
passwords3.txt:2:ZZZZZZZZZZZZZZZZZZZZ1234567890_from_first_file  
passwords.txt:126428:1234567890_  
passwords.txt:2150838:ZZZZZZZZZZZZZZZZZZZZ1234567890_from_first_file  
lynx@lynx:~$
```

- What did that do?  
It gave us a short output, and highlighting in green the line numbers of the match of “1234567890\_”
- Explain what just happened.  
We used fgrep with the -n option to get the line numbers that contain “1234567890\_” in the 3 txt files.

## Exercise 19

### Task #1

**grep -E "[0-9]{10}\_from" passwords.txt**

A terminal window showing the command 'grep -E "[0-9]{10}\_from" passwords.txt' being executed. The output is a single line of text: 'ZZZZZZZZZZZZZZZZZZZZ1234567890\_from\_first\_file'.

```
lynx@lynx: ~  
lynx@lynx:~$ grep -E "[0-9]{10}_from" passwords.txt  
ZZZZZZZZZZZZZZZZZZZZ1234567890_from_first_file  
lynx@lynx:~$
```

- What did that do?  
We get a single line output of an extended regular expression (ERE) that we are trying to match in the password.txt file
- Explain what just happened.  
Grep -E option interprets the pattern as an (ERE), enhancing grep’s capabilities. Then we use a regex “[0-9]{10}\_from” that we are trying to match from passwords.txt.

## Task #2

**grep -E "[0-9]{10}\_from" passwords\*.txt**

A terminal window with a black background and green text. The prompt is 'lynx@lynx: ~'. The command 'lynx@lynx:~\$ grep -E "[0-9]{10}\_from" passwords\*.txt' has been entered. The output shows three lines: 'passwords2.txt:ZZZZZZZZZZZZZZZZZZ1234567890\_from\_second\_file', 'passwords3.txt:ZZZZZZZZZZZZZZZZZZ1234567890\_from\_first\_file', and 'passwords.txt:ZZZZZZZZZZZZZZZZZZ1234567890\_from\_first\_file'. The prompt 'lynx@lynx:~\$' is visible at the bottom.

```
lynx@lynx:~$ grep -E "[0-9]{10}_from" passwords*.txt
passwords2.txt:ZZZZZZZZZZZZZZZZZZ1234567890_from_second_file
passwords3.txt:ZZZZZZZZZZZZZZZZZZ1234567890_from_first_file
passwords.txt:ZZZZZZZZZZZZZZZZZZ1234567890_from_first_file
lynx@lynx:~$
```

- What did that do?  
We get a single line output of an extended regular expression (ERE) that we are trying to match from all files the start with password and end with .txt
- Explain what just happened.  
Grep -E option interprets the pattern as an (ERE), enhancing grep's capabilities. Then we use a regex "**[0-9]{10}\_from**" that we are trying to match all files that start with passwords, and end with .txt



## Exercise 20

```
lynx@lynx:~$ cat authy.log
Mar 10 00:00:55 server1 sshd[4422]: Received disconnect from 10.0.2.2 port 60950:11: disconnected by user
Mar 10 00:00:55 server1 sshd[4422]: Disconnected from user ajay 10.0.2.2 port 60950
Mar 10 00:00:55 server1 sshd[4366]: pam_unix(sshd:session): session closed for user ajay
Mar 10 00:00:55 server1 systemd-logind[710]: Session 13 logged out. Waiting for processes to exit.
Mar 10 00:00:55 server1 systemd-logind[710]: Removed session 13.
Mar 10 00:00:55 server1 sshd[3878]: Received disconnect from 10.0.2.2 port 61128:11: disconnected by user
Mar 10 00:00:55 server1 sshd[3878]: Disconnected from user ajay 10.0.2.2 port 61128
Mar 10 00:00:55 server1 sshd[3822]: pam_unix(sshd:session): session closed for user ajay
Mar 10 00:00:55 server1 systemd-logind[710]: Session 5 logged out. Waiting for processes to exit.
Mar 10 00:00:55 server1 sshd[4043]: Received disconnect from 10.0.2.2 port 62494:11: disconnected by user
Mar 10 00:00:55 server1 sshd[4043]: Disconnected from user ajay 10.0.2.2 port 62494
Mar 10 00:00:55 server1 sshd[3987]: pam_unix(sshd:session): session closed for user ajay
Mar 10 00:00:55 server1 systemd-logind[710]: Removed session 5.
Mar 10 00:00:55 server1 systemd-logind[710]: Session 7 logged out. Waiting for processes to exit.
Mar 10 00:00:55 server1 systemd-logind[710]: Removed session 7.
Mar 10 00:01:00 server1 systemd-logind[710]: Power key pressed.
Mar 10 00:01:00 server1 systemd-logind[710]: Powering Off...
Mar 10 00:01:00 server1 systemd-logind[710]: System is powering down.
Mar 10 00:23:02 server1 systemd-logind[714]: New seat seat0.
Mar 10 00:23:02 server1 sshd[764]: Server listening on 0.0.0.0 port 22.
Mar 10 00:23:02 server1 sshd[764]: Server listening on :: port 22.
Mar 10 00:23:02 server1 systemd-logind[714]: Watching system buttons on /dev/input/event0 (Power Button)
Mar 10 00:23:02 server1 systemd-logind[714]: Watching system buttons on /dev/input/event1 (Sleep Button)
Mar 10 00:23:02 server1 systemd-logind[714]: Watching system buttons on /dev/input/event2 (AT Translated Set 2 keyboard)
Mar 10 01:17:01 server1 CRON[1035]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Mar 10 01:17:01 server1 CRON[1035]: pam_unix(cron:session): session closed for user root
Mar 10 02:17:01 server1 CRON[1101]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Mar 10 02:17:01 server1 CRON[1101]: pam_unix(cron:session): session closed for user root
Mar 10 02:27:09 server1 login[763]: pam_unix(login:session): session opened for user ajay(uid=1000) by LOGIN(uid=0)
Mar 10 02:27:09 server1 systemd-logind[714]: New session 3 of user ajay.
Mar 10 02:27:09 server1 systemd: pam_unix(systemd-user:session): session opened for user ajay(uid=1000) by (uid=0)
Mar 10 02:27:31 server1 sshd[1220]: Accepted password for ajay from 10.1.10.243 port 58204 ssh2
Mar 10 02:27:31 server1 sshd[1220]: pam_unix(sshd:session): session opened for user ajay(uid=1000) by (uid=0)
Mar 10 02:27:31 server1 systemd-logind[714]: New session 5 of user ajay.
Mar 10 02:25:31 server1 sshd[801]: Accepted password for robert from 10.0.0.10 port 52020 ssh2
Mar 10 02:25:31 server1 sshd[801]: pam_unix(sshd:session): session opened for user robert(uid=1001) by (uid=0)
Mar 10 02:26:45 server1 sshd[804]: Failed password for tara from 10.0.0.11 port 52022 ssh2
Mar 10 02:27:00 server1 sshd[807]: Accepted password for tara from 10.0.0.11 port 52025 ssh2
Mar 10 02:27:00 server1 sshd[807]: pam_unix(sshd:session): session opened for user tara(uid=1002) by (uid=0)
Mar 10 02:28:10 server1 sudo:    robert : TTY=pts/1 ; PWD=/home/robert ; USER=root ; COMMAND=/usr/bin/apt update
```

## Task #1

- Using authy.log determine why there was a service interruption on our website.

```
lynx@lynx:~$ grep apache2 authy.log
Mar 10 02:31:05 server1 sudo:    tara : TTY=pts/2 ; PWD=/home/tara ; USER=root ; COMMAND=/bin/systemctl shutdown apache2
lynx@lynx:~$
```

- Explain what just happened.  
I used grep to find out an entry line that has apache2. I can see apache2 was shutdown by tara as a root user.

## Task #2

- Using authy.log determine by who was this done?

```
lynx@lynx:~$ grep apache2 authy.log
Mar 10 02:31:05 server1 sudo:    tara : TTY=pts/2 ; PWD=/home/tara ; USER=root ; COMMAND=/bin/systemctl shutdown apache2
lynx@lynx:~$
```

- Explain what just happened.  
It looks like it was done by tara

## Task #3

- Using authy.log determine from where was this done from?

```
lynx@lynx:~$ grep tara authy.log
Mar 18 02:26:45 server1 sshd[884]: Failed password for tara from 10.0.0.11 port 52022 ssh2
Mar 18 02:27:00 server1 sshd[887]: Accepted password for tara from 10.0.0.11 port 52025 ssh2
Mar 18 02:27:00 server1 sshd[887]: pam_unix(sshd:session): session opened for user tara(uid=1002) by (uid=0)
Mar 18 02:31:05 server1 sudo: tara : TTY=pts/2 ; PWD=/home/tara ; USER=root ; COMMAND=/bin/systemctl shutdown apache2
Mar 18 02:31:05 server1 sudo: pam_unix(sudo:session): session opened for user root(uid=0) by tara(uid=1002)
Mar 18 02:35:10 server1 sshd[819]: Disconnected from user tara 10.0.0.11 port 52025
Mar 18 02:35:10 server1 sshd[807]: pam_unix(sshd:session): session closed for user tara
Mar 18 03:43:22 server1 sshd[920]: Failed password for tara from 10.0.0.16 port 53020 ssh2
Mar 18 03:44:03 server1 sshd[925]: Accepted password for tara from 10.0.0.16 port 53025 ssh2
Mar 18 03:44:03 server1 sshd[925]: pam_unix(sshd:session): session opened for user tara(uid=1002) by (uid=0)
Mar 18 03:44:56 server1 sudo: tara : TTY=pts/2 ; PWD=/home/tara ; USER=root ; COMMAND=/usr/bin/nano /etc/hosts
Mar 18 03:44:56 server1 sudo: pam_unix(sudo:session): session opened for user root(uid=0) by tara(uid=1002)
Mar 18 03:46:07 server1 sshd[930]: pam_unix(sshd:session): session closed for tara
lynx@lynx:~$
```

- Explain what just happened.  
By using grep tara, i can see it was done from 10.0.0.11