Discover Your Future

Systems and Network Programming

2020 Regular Intake

LIBSSH AUTHENTICATION BYPASS
CVE-2018-10933

Samarasinghe M.L.S

IT19130026

CONTENT PAGE

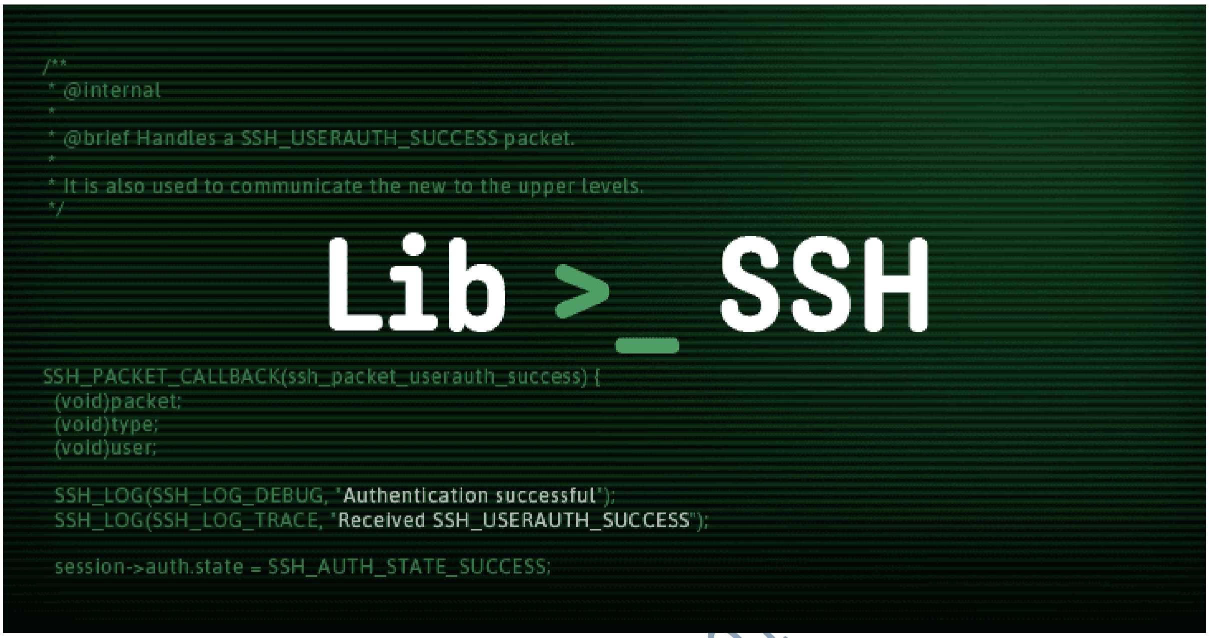
What is SSH?

Secure Shell (SSH) is a cryptographic network protocol for protected operation of network services over an unsecured network which is invented in 1995. Typical applications include remote command-line, authentication, and remote execution of commands, but any network service can be secured with this SSH protocol.

This SSH gives a secure/protected channel over unsecured network by linking an SSH client application to an SSH server using client-server architecture. The protocol specification divides between two main versions, SSH1 and SSh1. Usually SSH used to control Unix-like operating systems but it also can be used in Microsoft Windows.

This is a replacement for Telnet and other unsecure remote shell protocols like rsh, rexex and Berkeley rlogin.

What is LibSSH?



```
/**
 * @internal
 *
 * @brief Handles a SSH_USERAUTH_SUCCESS packet.
 *
 * It is also used to communicate the new to the upper levels.
 */

SSH_PACKET_CALLBACK(ssh_packet_userauth_success) {
  (void)packet;
  (void)type;
  (void)user;

  SSH_LOG(SSH_LOG_DEBUG, "Authentication successful");
  SSH_LOG(SSH_LOG_TRACE, "Received SSH_USERAUTH_SUCCESS");

  session->auth.state = SSH_AUTH_STATE_SUCCESS;
```

Like OpenSSH, LibSSH is an open source SSH C library that allows user to write programs using SSH protocol. For remote programs, you can execute programs remotely, pass files or use a safe and transparent tunnel. The SSH protocol is authenticated, guarantees data confidentiality and offers clear means to authenticate each of the client 's servers. This library hides a lot of technical information from the SSH protocol, but that doesn't mean you shouldn't try to understand that information. It should be noted that LibSSH should not be confused with OpenSSH or LibSSH2, as both of them are distinct

LibSSH isn't that commonly used, though. Libssh is easy to embed, making it appealing to all types of applications. A common case of usage might be where a developer wants to add an SSH implementation to the stack, but it's hard to integrate client / server applications.

Perhaps the most well-known platform that uses libssh is GitHub, the host service for Git. Github has explicitly confirmed that the vulnerability doesn't affect their implementation of libssh.

LibSSH vulnerability

This LibSSH vulnerability was found by the researcher Peter Winter-Smith of NCC group. This bug was introduced in version 0.6, released in 2014, and maintained by versions 0.8.4 and 0.7.6 until October 16, 2018.

A vulnerable server is totally wide open and any attacker could easily hack it. The effect of the vulnerability will rely on the permissions provided to the SSH server which will provide the attacker with complete control over the compromised machine or a potential mechanism for tunnelling into internal networks.

This bug allows an attacker to gain server root access without the username and password by sending an SSH2 MSG USERAUTH SUCCESS message to the server instead of the SSH2 MSG USERAUTH REQUEST message that the server would expect to initiate authentication, the attacker might authenticate successfully without any credentials. The vulnerability lies in libssh library authentication mechanism.

Several known applications on LibSSH:

- KDE uses libssh to transfer the sftp file GitHub has deployed libssh on its git SSH server
- X2Go is a Linux remote desktop device
- Csync is a two-way file synchronizer
- Reminate the GTK+/Gnome
- XMBC remote desktop server is a media player and the digital media entertainment center.
- GNU Gatekeeper is a full H.323 gatekeeper.

How to exploit this vulnerability?

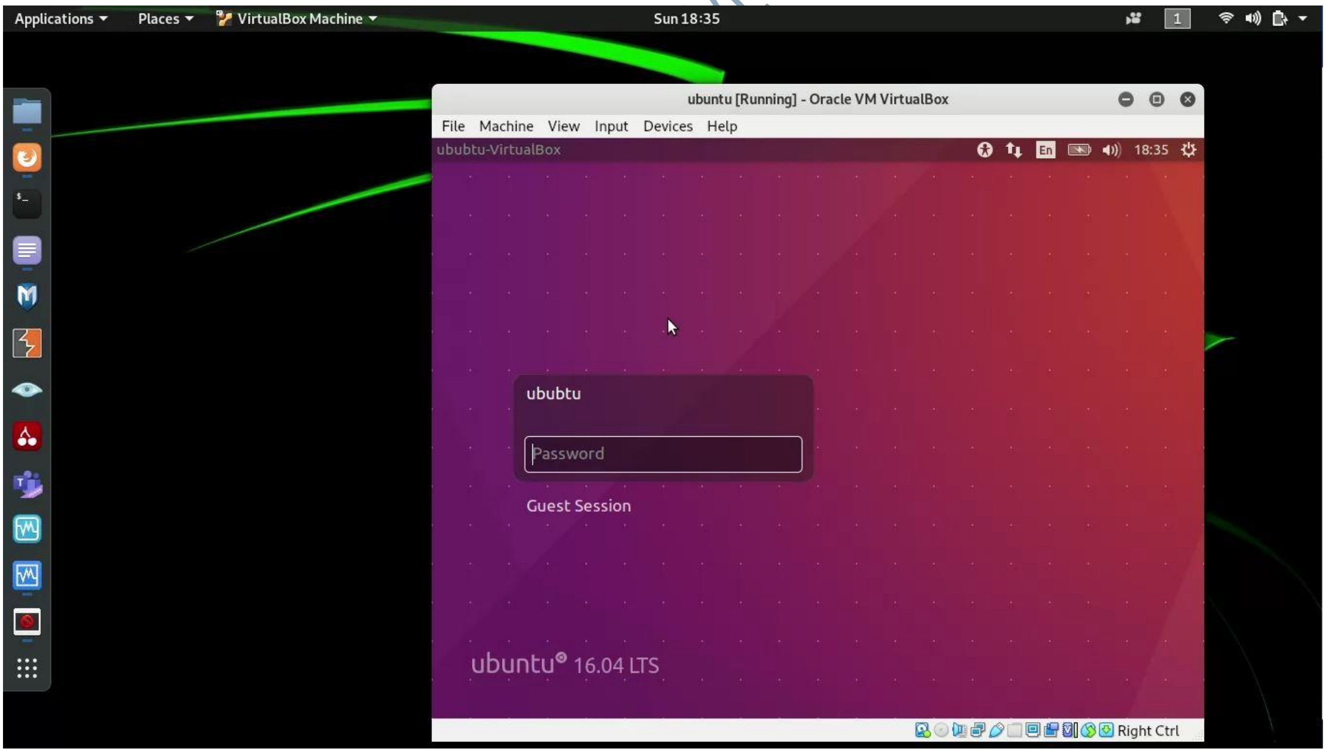
Requirements:

- Linux OS
- Python 2.7
- Docker

This bug was found in 2018. The attacker must have a Linux version on or before year 2018.   So, I had to install a virtual box in my Kali Linux OS.

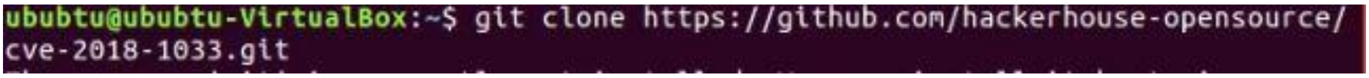
Then I Installed Kali Linux 2018.2-amd64 in my virtual box and tried to install a server with libssh which was cloned form github in that kali virtual machine but it wasn’t successful.

I installed Ubuntu 16.4 desktop version as the second option and tried to install that same server in this Ubuntu virtual machine.



1. Clone the file from Github.

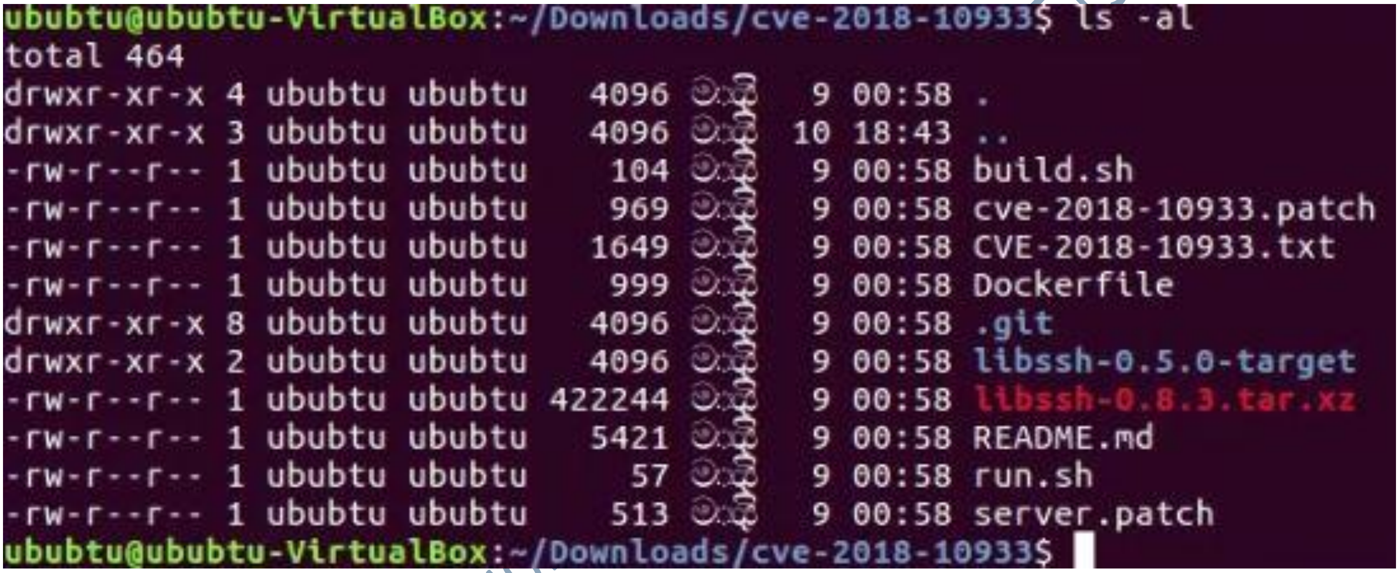
$git clone https://github.com/hackerhouse-opensource/cve-2018-10933.git



```
ububtu@ububtu-VirtualBox:~$ git clone https://github.com/hackerhouse-opensource/
cve-2018-1033.git
```

2. List the all directories including hidden files

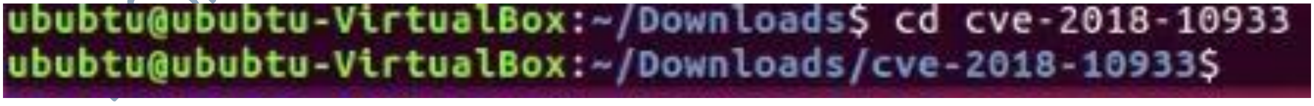
$ls -al



```
ububtu@ububtu-VirtualBox:~/Downloads/cve-2018-10933$ ls -al
total 464
drwxr-xr-x 4 ububtu ububtu   4096 [illegible]  9 00:58 .
drwxr-xr-x 3 ububtu ububtu   4096 [illegible] 10 18:43 ..
-rw-r--r-- 1 ububtu ububtu    104 [illegible]  9 00:58 build.sh
-rw-r--r-- 1 ububtu ububtu    969 [illegible]  9 00:58 cve-2018-10933.patch
-rw-r--r-- 1 ububtu ububtu   1649 [illegible]  9 00:58 CVE-2018-10933.txt
-rw-r--r-- 1 ububtu ububtu    999 [illegible]  9 00:58 Dockerfile
drwxr-xr-x 8 ububtu ububtu   4096 [illegible]  9 00:58 .git
drwxr-xr-x 2 ububtu ububtu   4096 [illegible]  9 00:58 libssh-0.5.0-target
-rw-r--r-- 1 ububtu ububtu 422244 [illegible]  9 00:58 libssh-0.8.3.tar.xz
-rw-r--r-- 1 ububtu ububtu   5421 [illegible]  9 00:58 README.md
-rw-r--r-- 1 ububtu ububtu     57 [illegible]  9 00:58 run.sh
-rw-r--r-- 1 ububtu ububtu    513 [illegible]  9 00:58 server.patch
ububtu@ububtu-VirtualBox:~/Downloads/cve-2018-10933$ 
```

3. Go into that downloaded folder

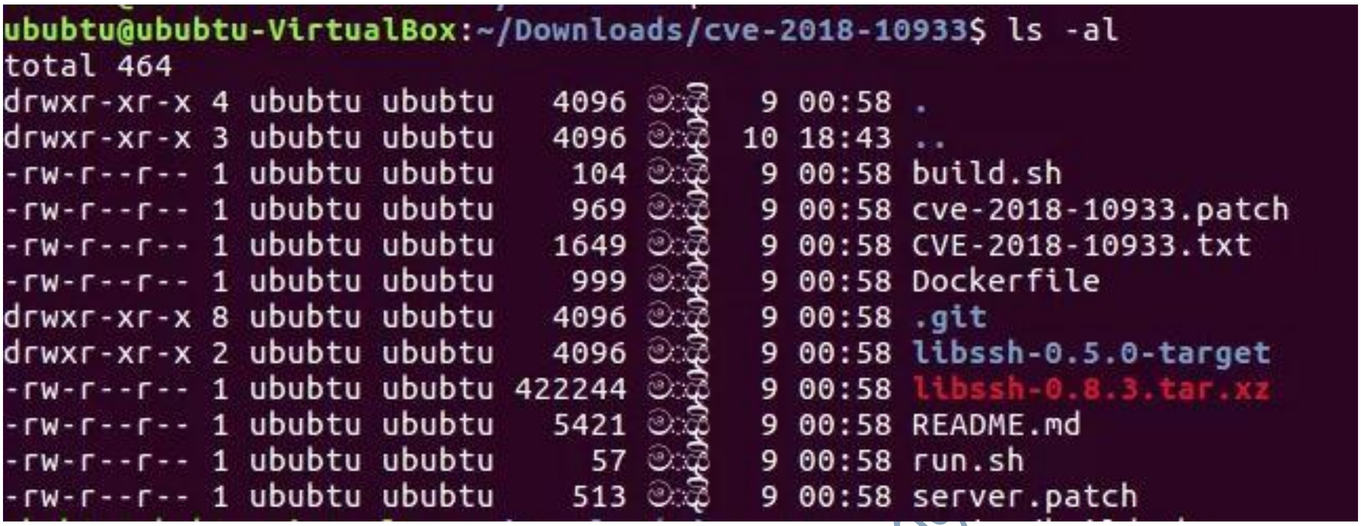
$cd cve-2018-10933



```
ububtu@ububtu-VirtualBox:~/Downloads$ cd cve-2018-10933
ububtu@ububtu-VirtualBox:~/Downloads/cve-2018-10933$
```

4. List the all directories including hidden files

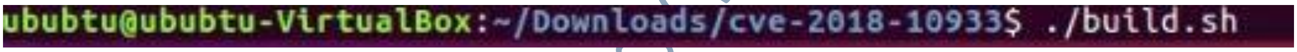
```
$ls -al
```



```
ububtu@ububtu-VirtualBox:~/Downloads/cve-2018-10933$ ls -al
total 464
drwxr-xr-x 4 ububtu ububtu   4096 [illegible]  9 00:58 .
drwxr-xr-x 3 ububtu ububtu   4096 [illegible] 10 18:43 ..
-rw-r--r-- 1 ububtu ububtu    104 [illegible]  9 00:58 build.sh
-rw-r--r-- 1 ububtu ububtu    969 [illegible]  9 00:58 cve-2018-10933.patch
-rw-r--r-- 1 ububtu ububtu   1649 [illegible]  9 00:58 CVE-2018-10933.txt
-rw-r--r-- 1 ububtu ububtu    999 [illegible]  9 00:58 Dockerfile
drwxr-xr-x 8 ububtu ububtu   4096 [illegible]  9 00:58 .git
drwxr-xr-x 2 ububtu ububtu   4096 [illegible]  9 00:58 libssh-0.5.0-target
-rw-r--r-- 1 ububtu ububtu 422244 [illegible]  9 00:58 libssh-0.8.3.tar.xz
-rw-r--r-- 1 ububtu ububtu   5421 [illegible]  9 00:58 README.md
-rw-r--r-- 1 ububtu ububtu     57 [illegible]  9 00:58 run.sh
-rw-r--r-- 1 ububtu ububtu    513 [illegible]  9 00:58 server.patch
```

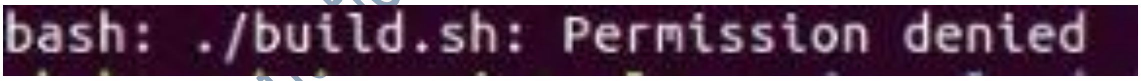
5. Run build.sh

```
$ ./.build.sh
```



```
ububtu@ububtu-VirtualBox:~/Downloads/cve-2018-10933$ ./build.sh
```

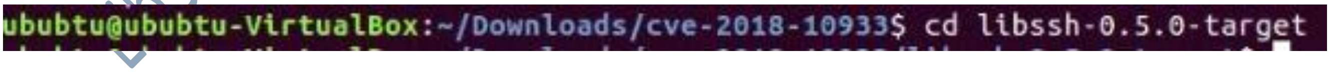
Result was this.



```
bash: ./build.sh: Permission denied
```

6. Then go into libssh-0.5.0-target directory in that directory

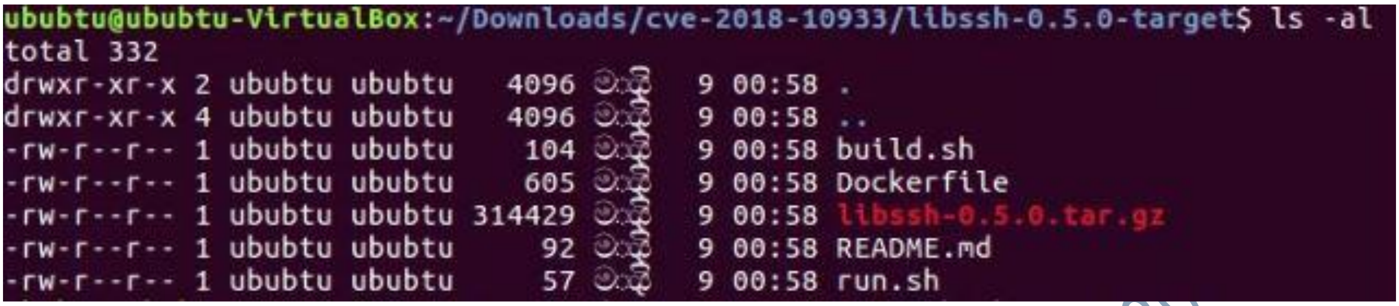
```
$cd libssh-0.5.0-target
```



```
ububtu@ububtu-VirtualBox:~/Downloads/cve-2018-10933$ cd libssh-0.5.0-target
```

7. Run ls command

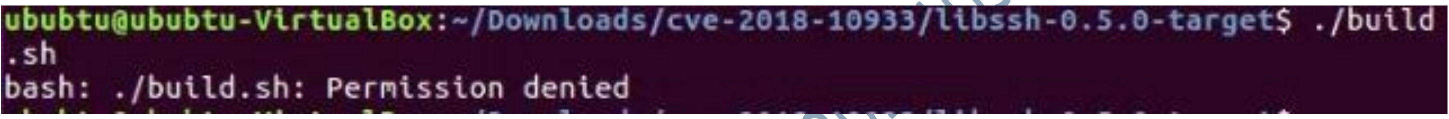
$ ls -al



```
ububtu@ububtu-VirtualBox:~/Downloads/cve-2018-10933/libssh-0.5.0-target$ ls -al
total 332
drwxr-xr-x 2 ububtu ububtu   4096 [illegible]  9 00:58 .
drwxr-xr-x 4 ububtu ububtu   4096 [illegible]  9 00:58 ..
-rw-r--r-- 1 ububtu ububtu    104 [illegible]  9 00:58 build.sh
-rw-r--r-- 1 ububtu ububtu    605 [illegible]  9 00:58 Dockerfile
-rw-r--r-- 1 ububtu ububtu 314429 [illegible]  9 00:58 libssh-0.5.0.tar.gz
-rw-r--r-- 1 ububtu ububtu     92 [illegible]  9 00:58 README.md
-rw-r--r-- 1 ububtu ububtu     57 [illegible]  9 00:58 run.sh
```

8. Run the build.sh again

$ ./build.sh



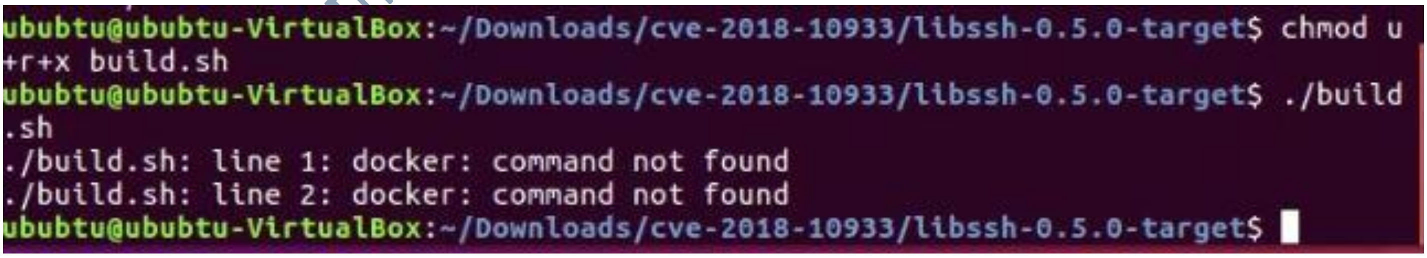
```
ububtu@ububtu-VirtualBox:~/Downloads/cve-2018-10933/libssh-0.5.0-target$ ./build
.sh
bash: ./build.sh: Permission denied
```

9. Mode the build.sh with permissions and run that build.sh again

$ chmod u+r+x build.sh

$ ./build.sh

Results were this:



```
ububtu@ububtu-VirtualBox:~/Downloads/cve-2018-10933/libssh-0.5.0-target$ chmod u
+r+x build.sh
ububtu@ububtu-VirtualBox:~/Downloads/cve-2018-10933/libssh-0.5.0-target$ ./build
.sh
./build.sh: line 1: docker: command not found
./build.sh: line 2: docker: command not found
ububtu@ububtu-VirtualBox:~/Downloads/cve-2018-10933/libssh-0.5.0-target$ 
```

After that I had to install docker in to my Ubuntu machine. I had to face to lot of errors while installing docker. So, I was unable to install that server with libssh in my ubuntu machine.

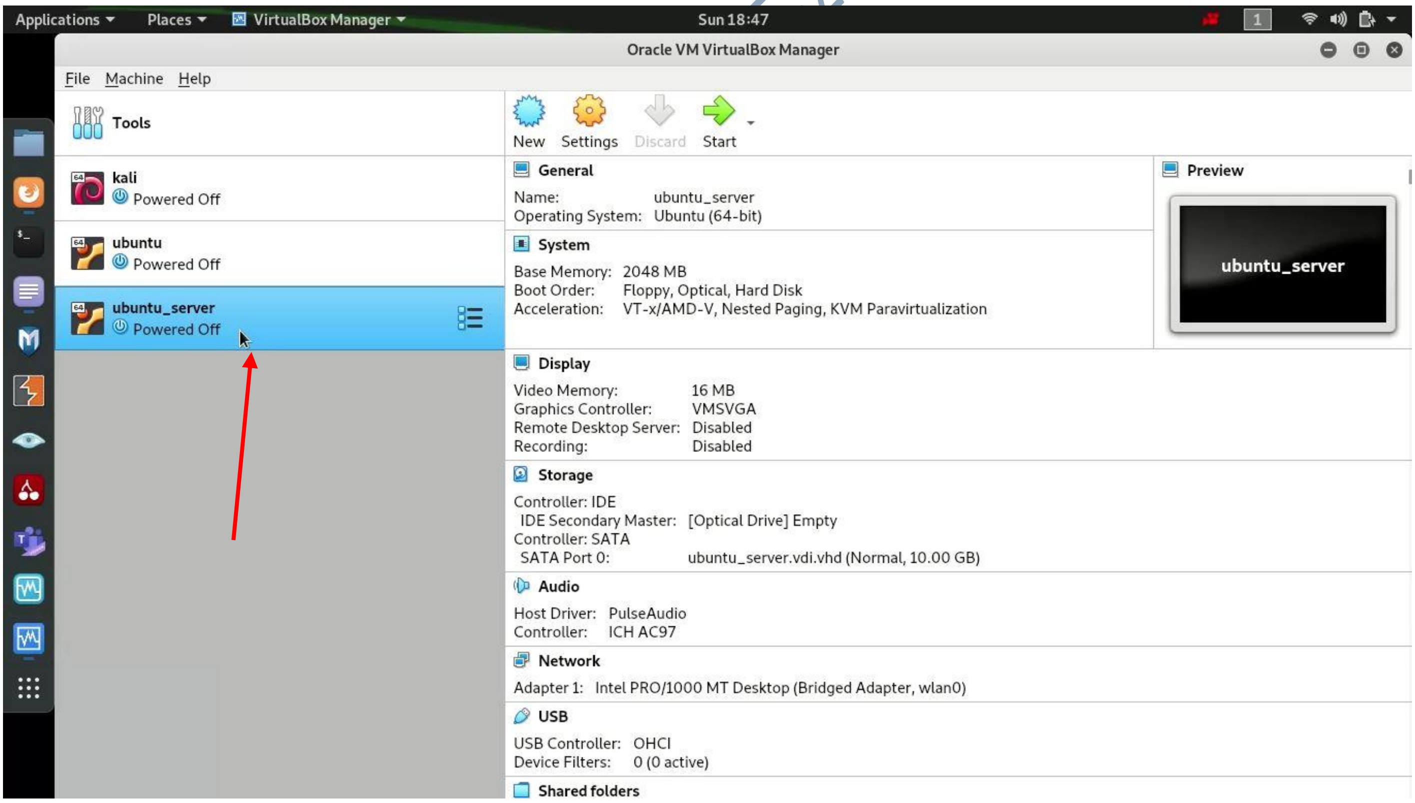
LibSSH exploiting method

I was unable to exploit that libssh server due to the frailer of creating the libssh server in my Ubuntu machine.

Now this is the way I exploited an Open SSH with the exactly same method that I was going to exploit that LibSSH. There are no differences between these exploiting methods.

Exploiting the Open SSH

1) I installed Ubuntu 16.4 server version in my virtual box as the vulnerable server/machine.



kali
Powered Off

ubuntu
Powered Off

ubuntu_server
Powered Off

General
Name: ubuntu_server
Operating System: Ubuntu (64-bit)

System
Base Memory: 2048 MB
Boot Order: Floppy, Optical, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, KVM Paravirtualization

Display
Video Memory: 16 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage
Controller: IDE
IDE Secondary Master: [Optical Drive] Empty
Controller: SATA
SATA Port 0: ubuntu_server.vdi.vhd (Normal, 10.00 GB)

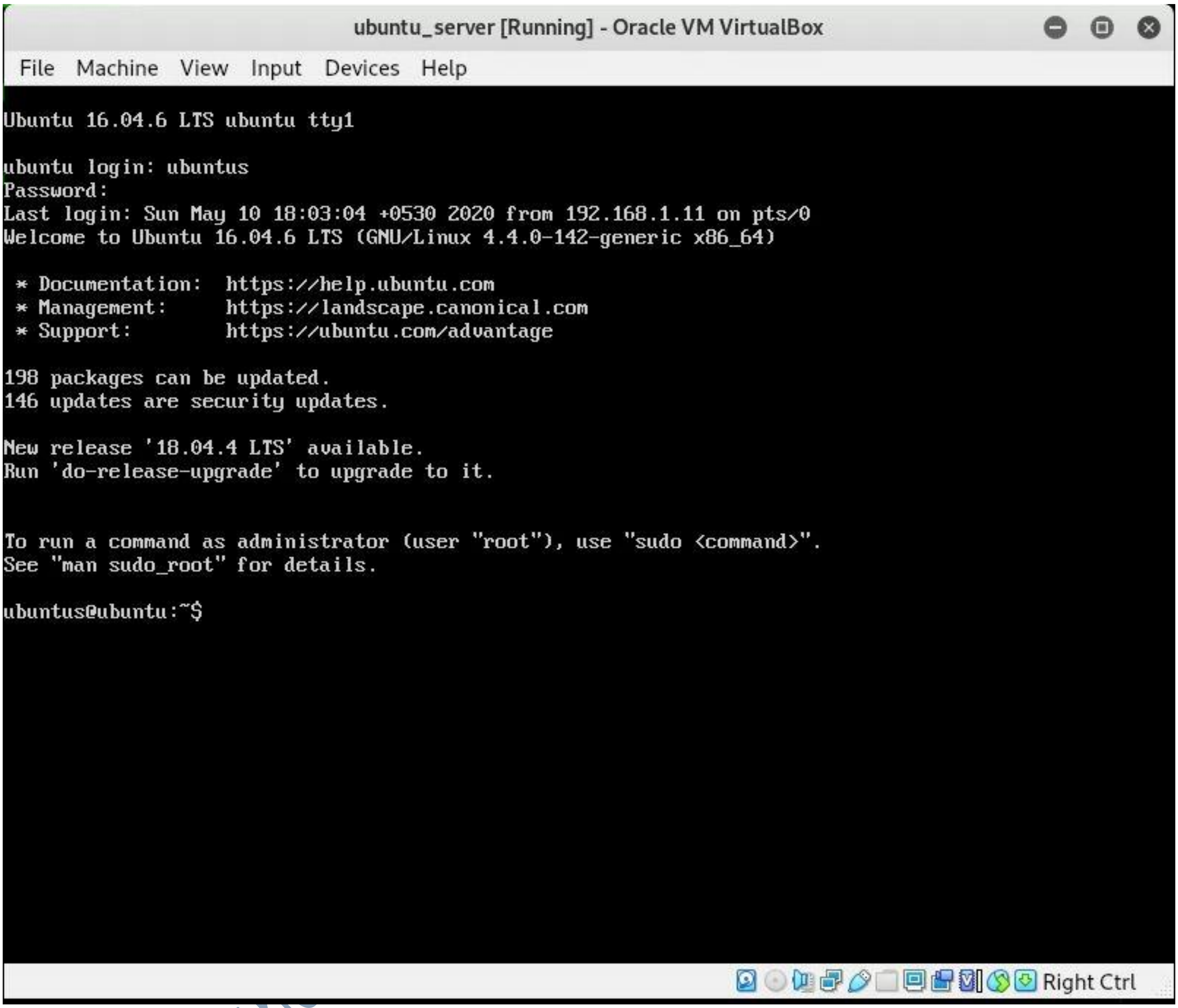
Audio
Host Driver: PulseAudio
Controller: ICH AC97

Network
Adapter 1: Intel PRO/1000 MT Desktop (Bridged Adapter, wlan0)

USB
USB Controller: OHCI
Device Filters: 0 (0 active)

Shared folders

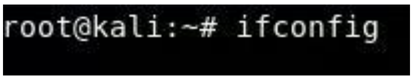
2) Strat the Ubuntu server


ubuntu_server [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Right Ctrl

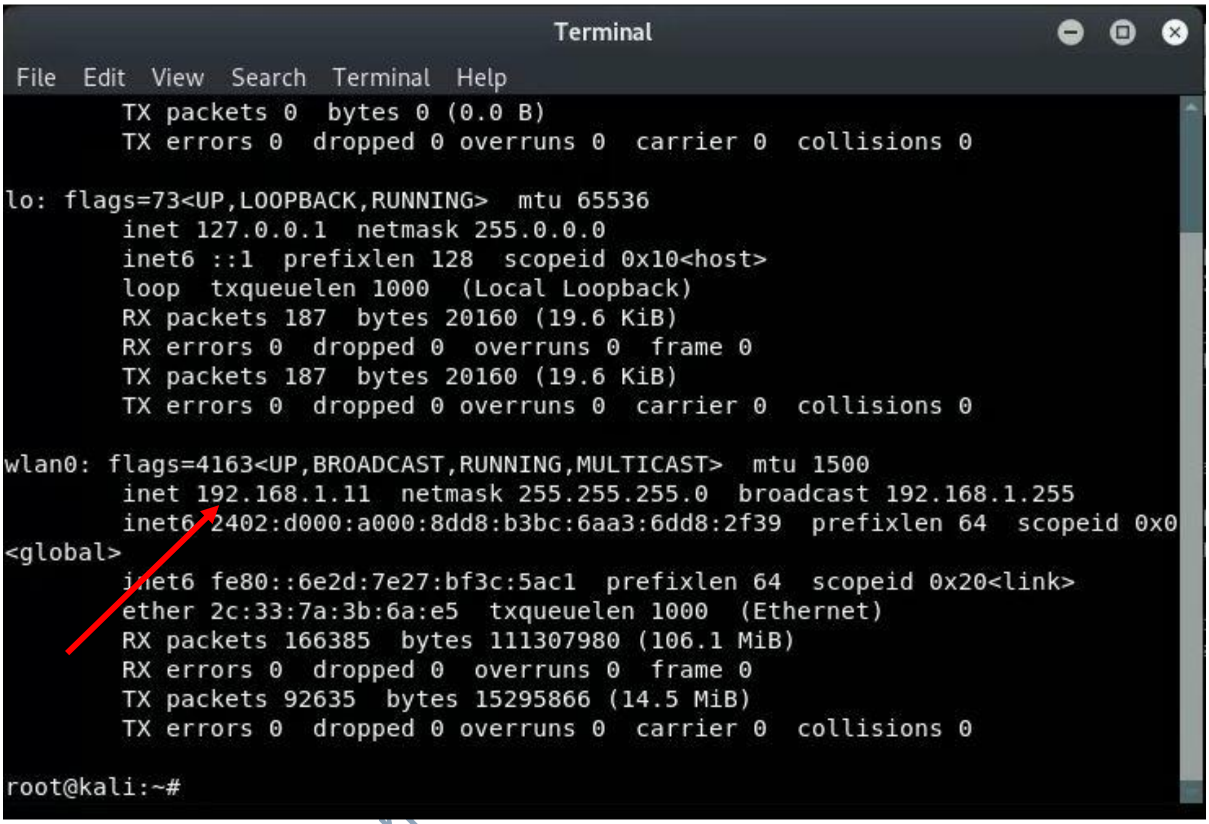
```
Ubuntu 16.04.6 LTS ubuntu tty1

ubuntu login: ubuntus
Password:
Last login: Sun May 10 18:03:04 +0530 2020 from 192.168.1.11 on pts/0
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-142-generic x86_64)

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

198 packages can be updated.
146 updates are security updates.

New release '18.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntus@ubuntu:~$
```

3) Open terminal in the attacker machine (Kali 2019.3) and run ifconfig to find to its IP address.

```
$ ifconfig
```



```
root@kali:~# ifconfig
```

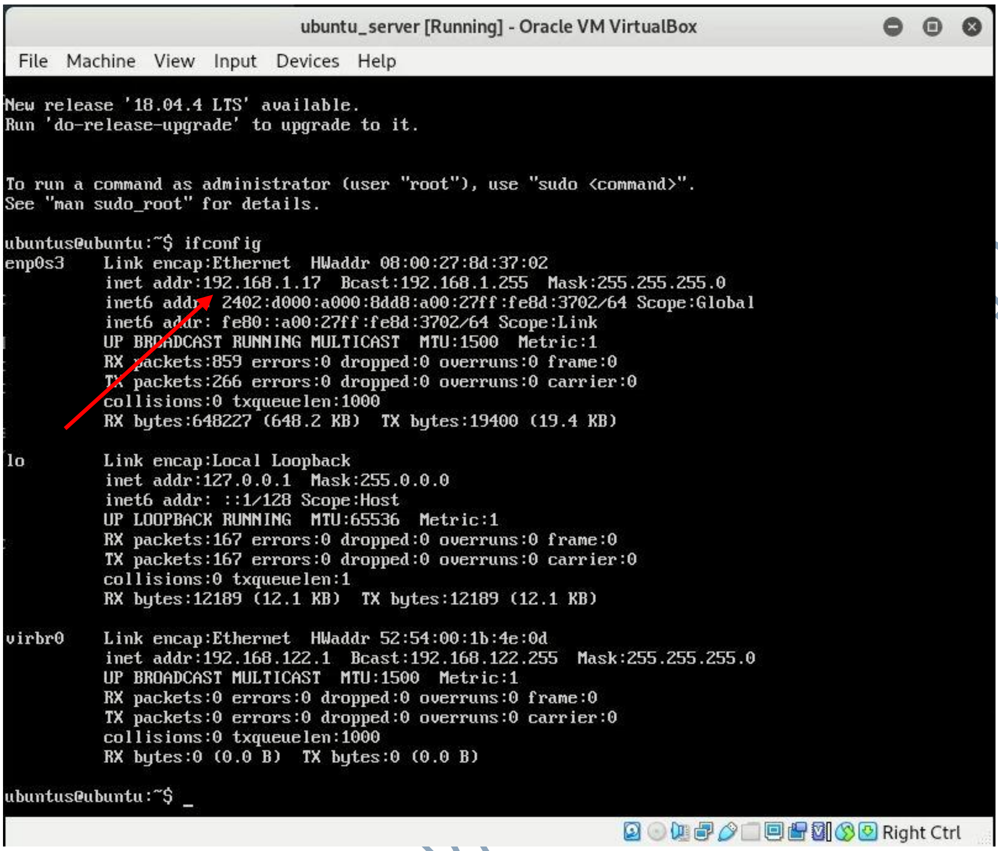


```
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

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 187  bytes 20160 (19.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 187  bytes 20160 (19.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.11  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 2402:d000:a000:8dd8:b3bc:6aa3:6dd8:2f39  prefixlen 64  scopeid 0x0
<global>
        inet6 fe80::6e2d:7e27:bf3c:5ac1  prefixlen 64  scopeid 0x20<link>
        ether 2c:33:7a:3b:6a:e5  txqueuelen 1000  (Ethernet)
        RX packets 166385  bytes 111307980 (106.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 92635  bytes 15295866 (14.5 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@kali:~#
```

The IP address is `192.168.1.11`

4) Also run ifconfig in Ubuntu server to find to its IP address too.

```
$ifconfig
```



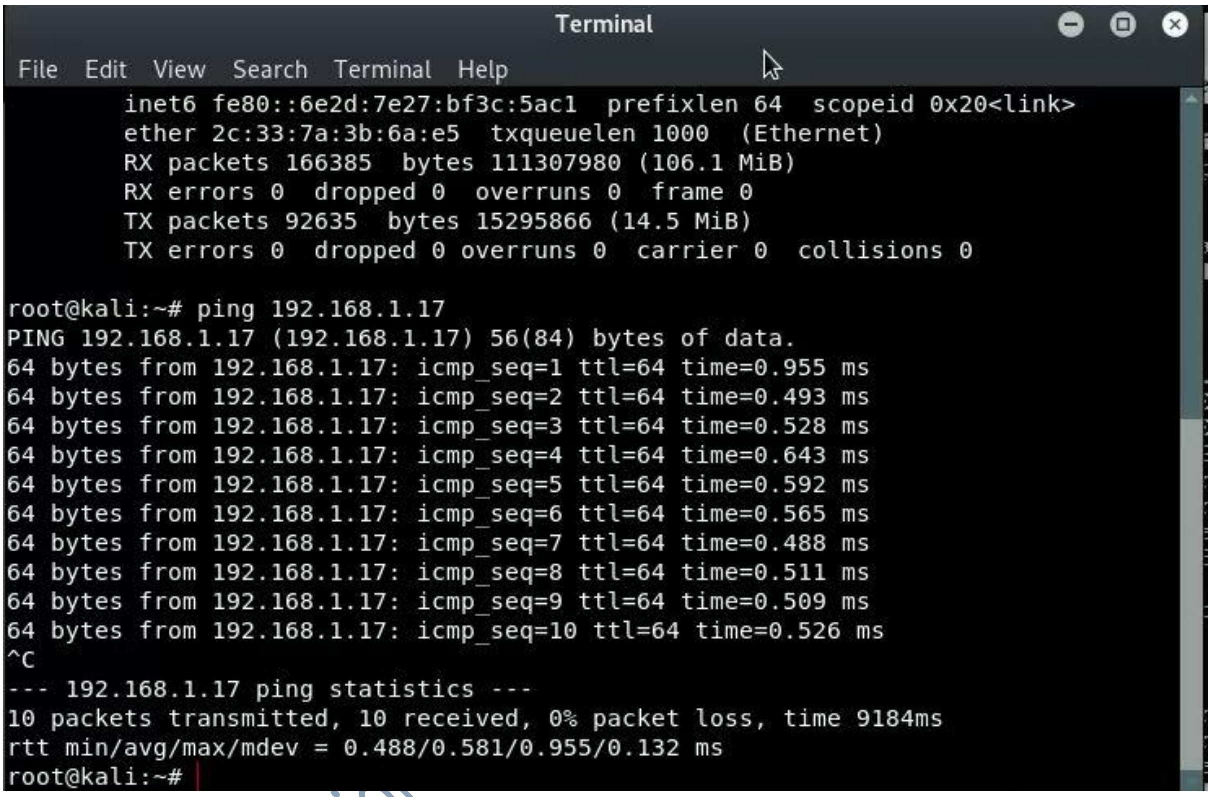
ubuntu_server [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Right Ctrl

```
New release '18.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntus@ubuntu:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:8d:37:02
          inet addr:192.168.1.17  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: 2402:d000:a000:8dd8:a00:27ff:fe8d:3702/64 Scope:Global
          inet6 addr: fe80::a00:27ff:fe8d:3702/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:859 errors:0 dropped:0 overruns:0 frame:0
          TX packets:266 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:648227 (648.2 KB)  TX bytes:19400 (19.4 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:167 errors:0 dropped:0 overruns:0 frame:0
          TX packets:167 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:12189 (12.1 KB)  TX bytes:12189 (12.1 KB)

virbr0    Link encap:Ethernet  HWaddr 52:54:00:1b:4e:0d
          inet addr:192.168.122.1  Bcast:192.168.122.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

ubuntus@ubuntu:~$ _
```

The IP address is `192.168.1.17`

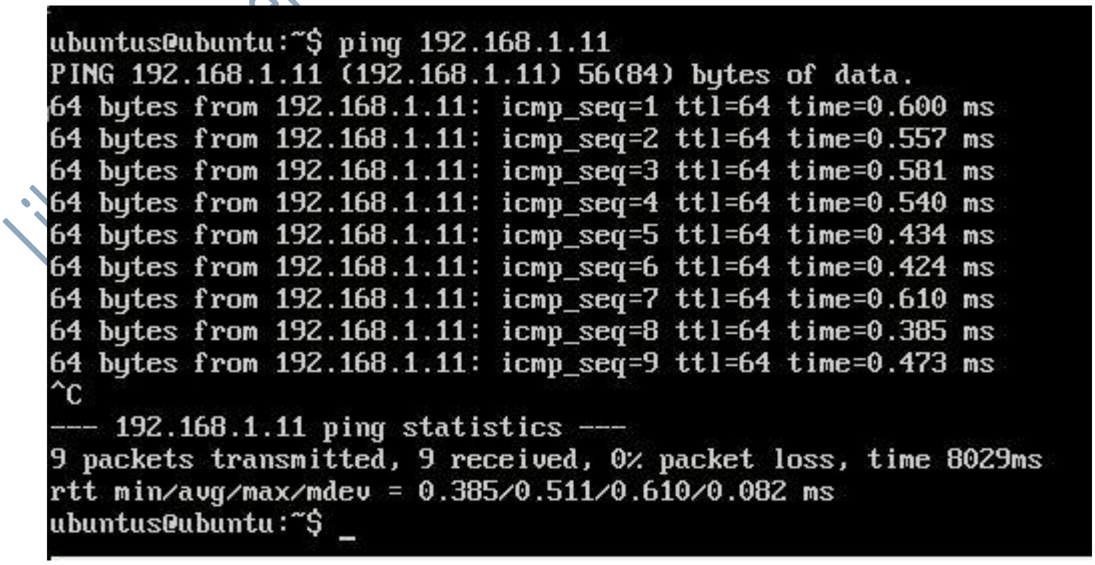
5) In this Step I ping the both machines to ensure that they're connected properly.
   In attacker machine, we should enter the vulnerable machine's/server's IP.

   $ping 192.168.1.17



```
        inet6 fe80::6e2d:7e27:bf3c:5ac1  prefixlen 64  scopeid 0x20<link>
        ether 2c:33:7a:3b:6a:e5  txqueuelen 1000  (Ethernet)
        RX packets 166385  bytes 111307980 (106.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 92635  bytes 15295866 (14.5 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@kali:~# ping 192.168.1.17
PING 192.168.1.17 (192.168.1.17) 56(84) bytes of data.
64 bytes from 192.168.1.17: icmp_seq=1 ttl=64 time=0.955 ms
64 bytes from 192.168.1.17: icmp_seq=2 ttl=64 time=0.493 ms
64 bytes from 192.168.1.17: icmp_seq=3 ttl=64 time=0.528 ms
64 bytes from 192.168.1.17: icmp_seq=4 ttl=64 time=0.643 ms
64 bytes from 192.168.1.17: icmp_seq=5 ttl=64 time=0.592 ms
64 bytes from 192.168.1.17: icmp_seq=6 ttl=64 time=0.565 ms
64 bytes from 192.168.1.17: icmp_seq=7 ttl=64 time=0.488 ms
64 bytes from 192.168.1.17: icmp_seq=8 ttl=64 time=0.511 ms
64 bytes from 192.168.1.17: icmp_seq=9 ttl=64 time=0.509 ms
64 bytes from 192.168.1.17: icmp_seq=10 ttl=64 time=0.526 ms
^C
--- 192.168.1.17 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9184ms
rtt min/avg/max/mdev = 0.488/0.581/0.955/0.132 ms
root@kali:~#
```

   In vulnerable machine, we should enter the attacker machine's IP
   $ping 192.168.1.11



```
ubuntus@ubuntu:~$ ping 192.168.1.11
PING 192.168.1.11 (192.168.1.11) 56(84) bytes of data.
64 bytes from 192.168.1.11: icmp_seq=1 ttl=64 time=0.600 ms
64 bytes from 192.168.1.11: icmp_seq=2 ttl=64 time=0.557 ms
64 bytes from 192.168.1.11: icmp_seq=3 ttl=64 time=0.581 ms
64 bytes from 192.168.1.11: icmp_seq=4 ttl=64 time=0.540 ms
64 bytes from 192.168.1.11: icmp_seq=5 ttl=64 time=0.434 ms
64 bytes from 192.168.1.11: icmp_seq=6 ttl=64 time=0.424 ms
64 bytes from 192.168.1.11: icmp_seq=7 ttl=64 time=0.610 ms
64 bytes from 192.168.1.11: icmp_seq=8 ttl=64 time=0.385 ms
64 bytes from 192.168.1.11: icmp_seq=9 ttl=64 time=0.473 ms
^C
--- 192.168.1.11 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8029ms
rtt min/avg/max/mdev = 0.385/0.511/0.610/0.082 ms
ubuntus@ubuntu:~$ _
```

Now we can see both of them connected successfully.

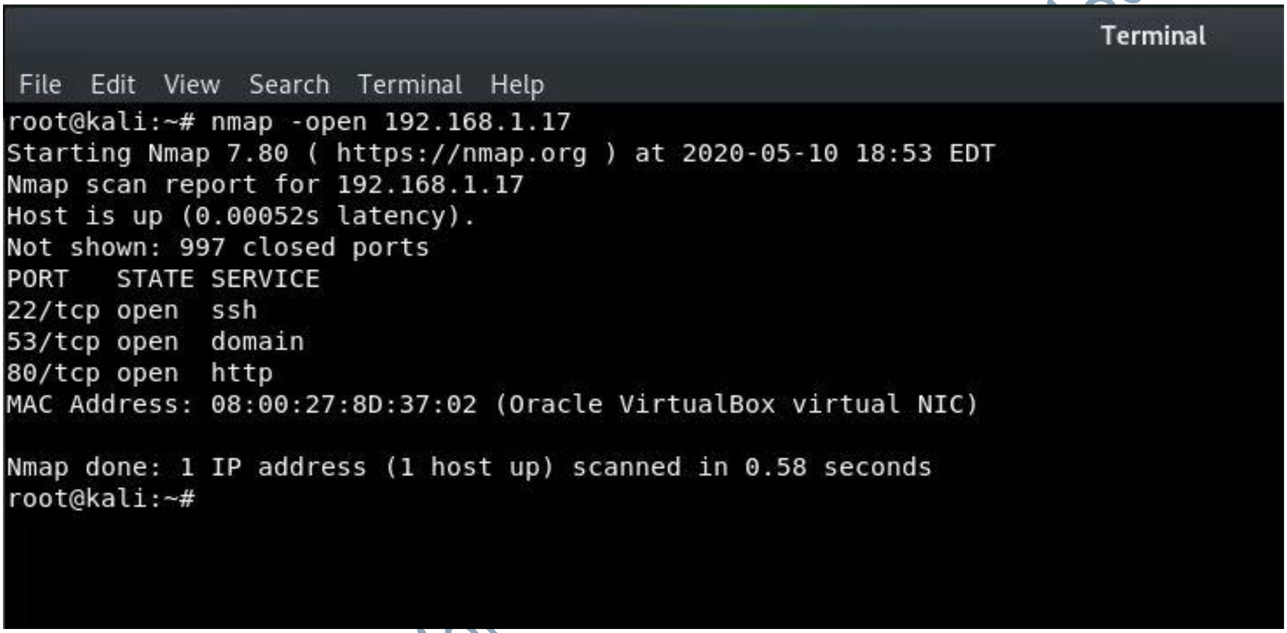
Exploitation of Open SSH

Open the in the attacker machine.

1) Search open ports in the server*

```
$ nmap -open 192.168.1.17
```

*here now we are exploiting my ***ubuntu server***. Then we should enter the Ubuntu server's IP address. When we are exploiting the *Libssh server*, we should enter that libssh server's IP address.



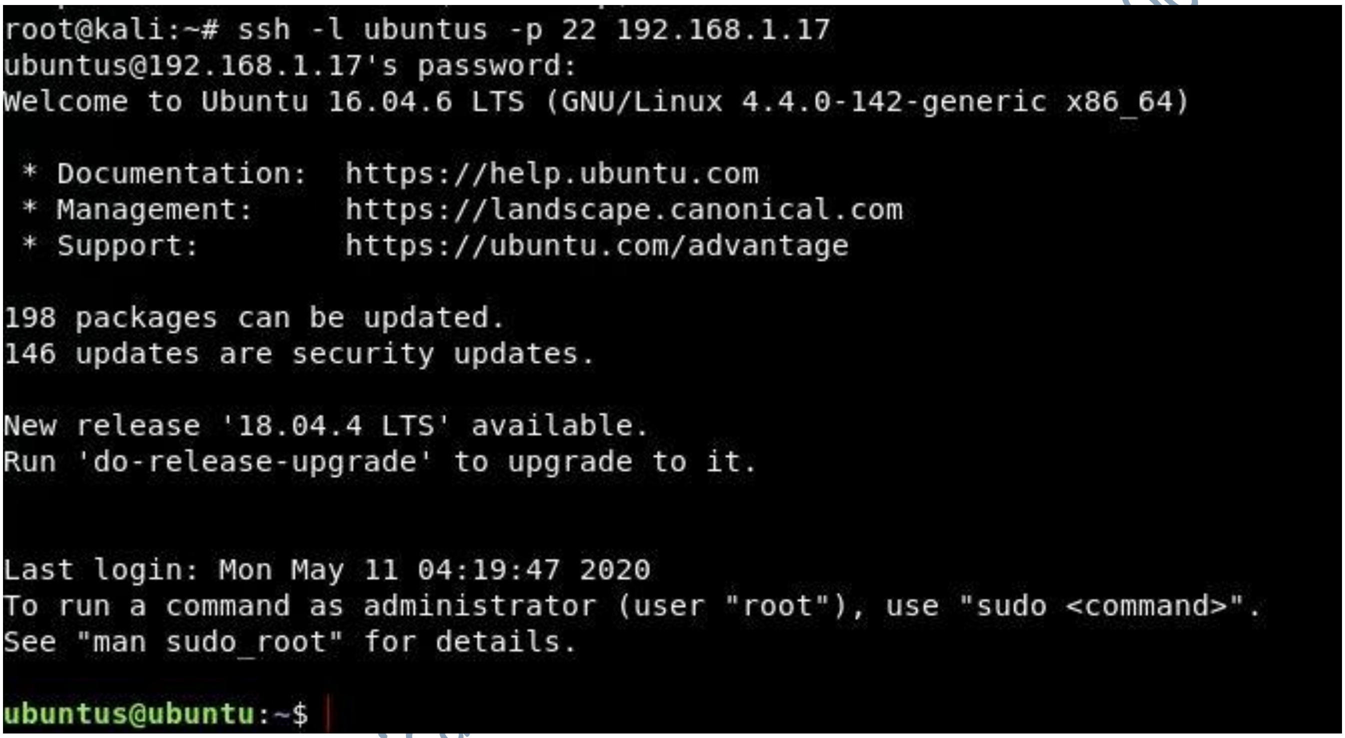
```
root@kali:~# nmap -open 192.168.1.17
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-10 18:53 EDT
Nmap scan report for 192.168.1.17
Host is up (0.00052s latency).
Not shown: 997 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
53/tcp open  domain
80/tcp open  http
MAC Address: 08:00:27:8D:37:02 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.58 seconds
root@kali:~#
```

2) Get access

```
$ ssh -l ubuntus -p 22 192.168.1.17
```

`password:` (here password is my ubuntu server's password)

* when we exploiting the libssh server *"myuser"* as the user name *"2222"* as the port and *"mypassword"* as the password.

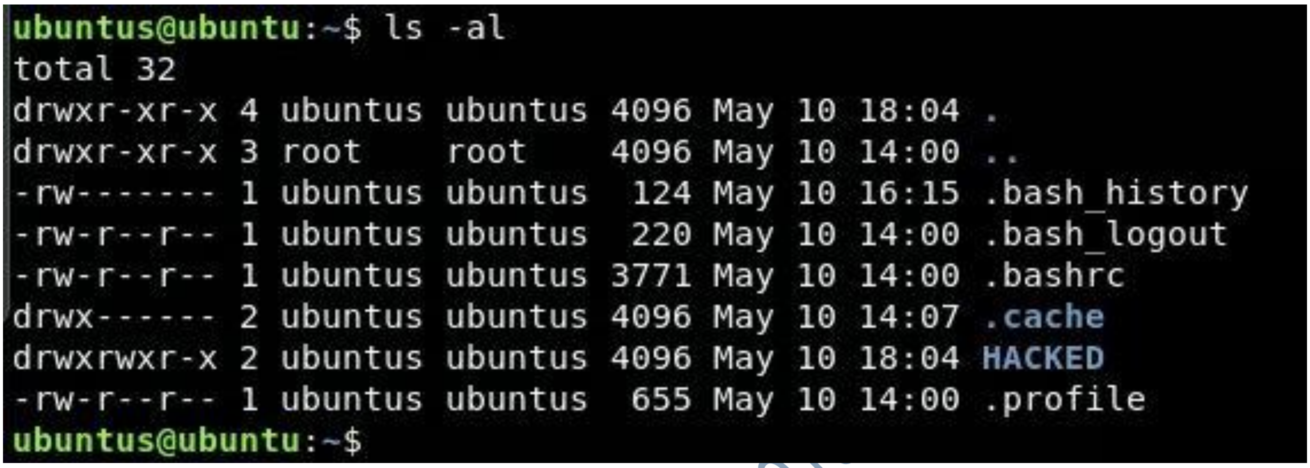


```
root@kali:~# ssh -l ubuntus -p 22 192.168.1.17
ubuntus@192.168.1.17's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-142-generic x86_64)

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

198 packages can be updated.
146 updates are security updates.

New release '18.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Mon May 11 04:19:47 2020
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntus@ubuntu:~$
```

Now we have the access to the ubuntu server. let’s confirm it by checking directories from attacker machine and the ubuntu server.

In attacker

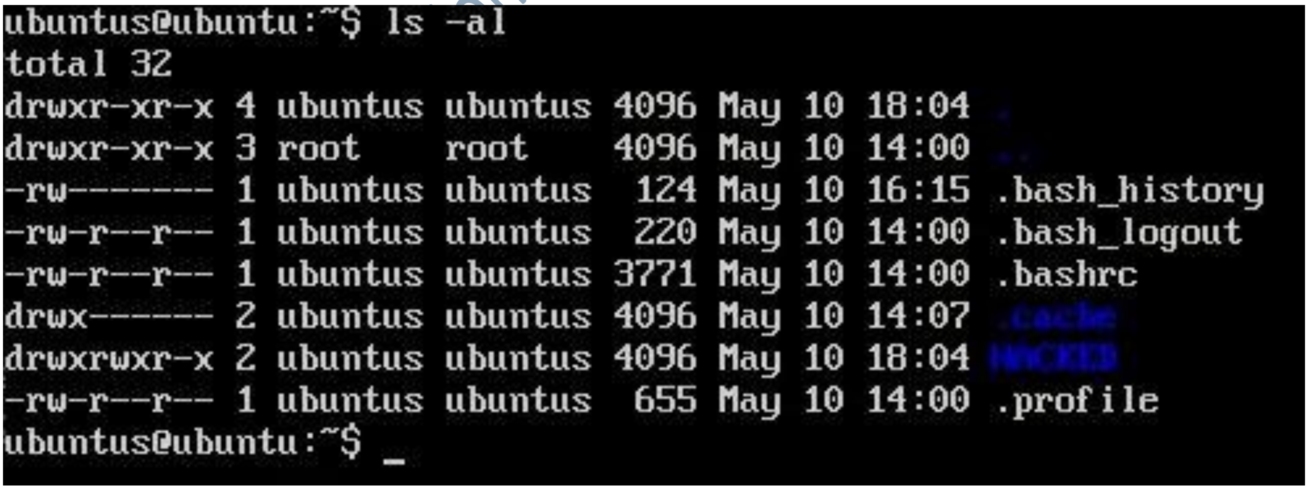
```
$ls -al
```



```
ubuntus@ubuntu:~$ ls -al
total 32
drwxr-xr-x 4 ubuntus ubuntus 4096 May 10 18:04 .
drwxr-xr-x 3 root    root    4096 May 10 14:00 ..
-rw------- 1 ubuntus ubuntus  124 May 10 16:15 .bash_history
-rw-r--r-- 1 ubuntus ubuntus  220 May 10 14:00 .bash_logout
-rw-r--r-- 1 ubuntus ubuntus 3771 May 10 14:00 .bashrc
drwx------ 2 ubuntus ubuntus 4096 May 10 14:07 .cache
drwxrwxr-x 2 ubuntus ubuntus 4096 May 10 18:04 HACKED
-rw-r--r-- 1 ubuntus ubuntus  655 May 10 14:00 .profile
ubuntus@ubuntu:~$
```

In server

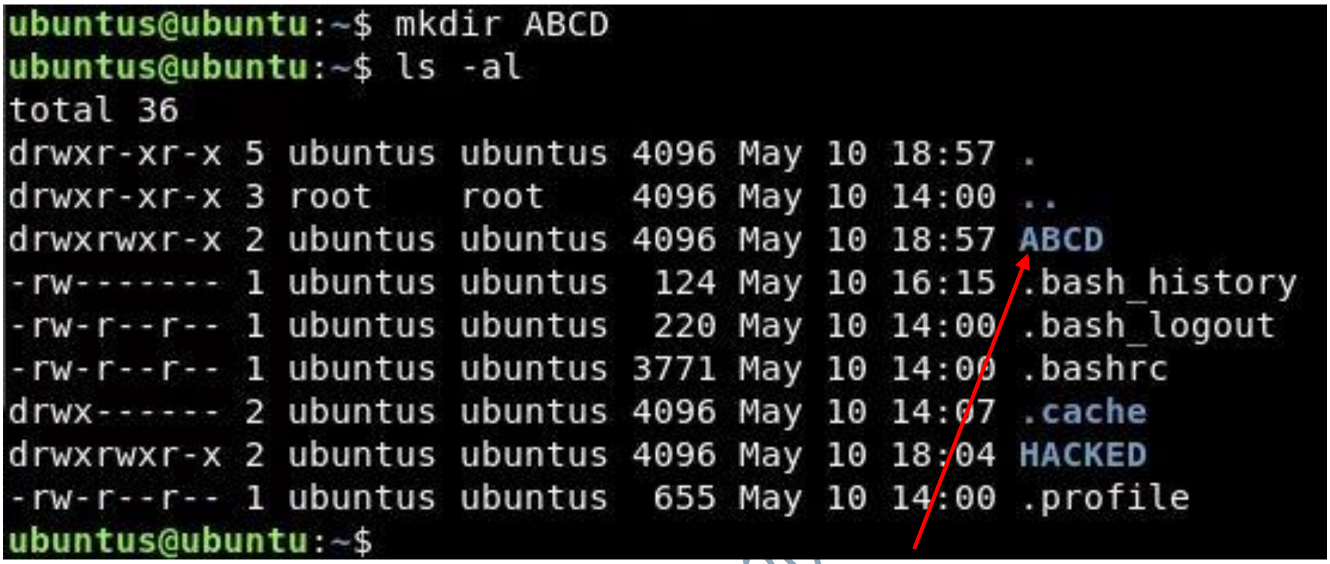
```
$ls -al
```



```
ubuntus@ubuntu:~$ ls -al
total 32
drwxr-xr-x 4 ubuntus ubuntus 4096 May 10 18:04 .
drwxr-xr-x 3 root    root    4096 May 10 14:00 ..
-rw------- 1 ubuntus ubuntus  124 May 10 16:15 .bash_history
-rw-r--r-- 1 ubuntus ubuntus  220 May 10 14:00 .bash_logout
-rw-r--r-- 1 ubuntus ubuntus 3771 May 10 14:00 .bashrc
drwx------ 2 ubuntus ubuntus 4096 May 10 14:07 .cache
drwxrwxr-x 2 ubuntus ubuntus 4096 May 10 18:04 HACKED
-rw-r--r-- 1 ubuntus ubuntus  655 May 10 14:00 .profile
ubuntus@ubuntu:~$ _
```

For further let’s make a directory in the server using attacker machine

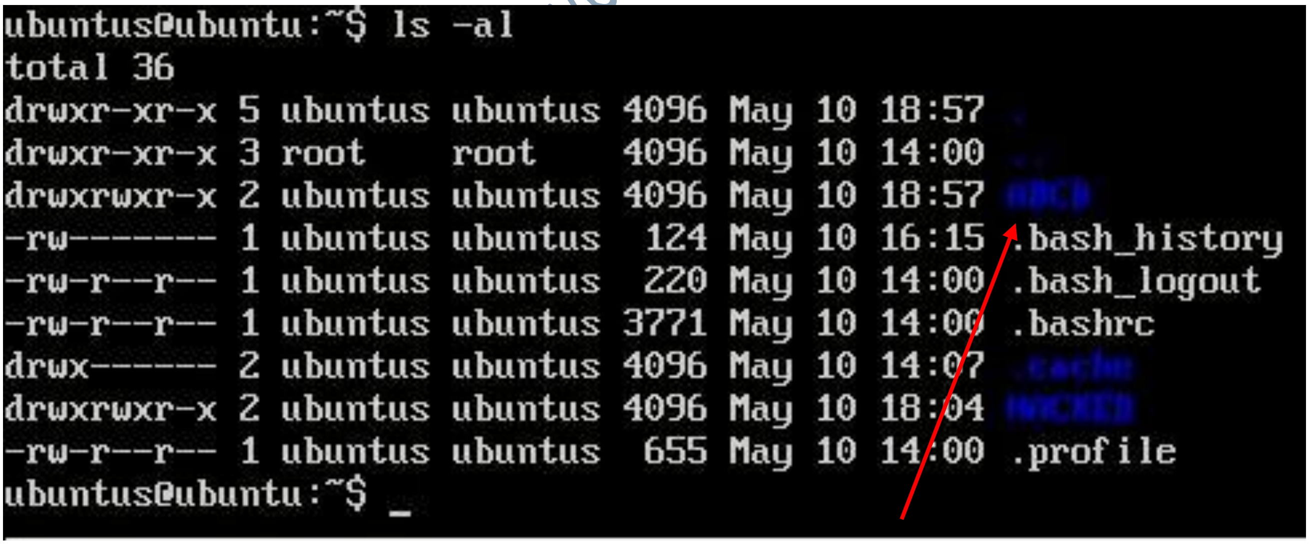
```
$mkdir ABCD
$ls -al
```



```
ubuntus@ubuntu:~$ mkdir ABCD
ubuntus@ubuntu:~$ ls -al
total 36
drwxr-xr-x 5 ubuntus ubuntus 4096 May 10 18:57 .
drwxr-xr-x 3 root    root    4096 May 10 14:00 ..
drwxrwxr-x 2 ubuntus ubuntus 4096 May 10 18:57 ABCD
-rw------- 1 ubuntus ubuntus  124 May 10 16:15 .bash_history
-rw-r--r-- 1 ubuntus ubuntus  220 May 10 14:00 .bash_logout
-rw-r--r-- 1 ubuntus ubuntus 3771 May 10 14:00 .bashrc
drwx------ 2 ubuntus ubuntus 4096 May 10 14:07 .cache
drwxrwxr-x 2 ubuntus ubuntus 4096 May 10 18:04 HACKED
-rw-r--r-- 1 ubuntus ubuntus  655 May 10 14:00 .profile
ubuntus@ubuntu:~$
```

So, let’s check that from ubuntu server

```
$ls -al
```



```
ubuntus@ubuntu:~$ ls -al
total 36
drwxr-xr-x 5 ubuntus ubuntus 4096 May 10 18:57 .
drwxr-xr-x 3 root    root    4096 May 10 14:00 ..
drwxrwxr-x 2 ubuntus ubuntus 4096 May 10 18:57 ABCD
-rw------- 1 ubuntus ubuntus  124 May 10 16:15 .bash_history
-rw-r--r-- 1 ubuntus ubuntus  220 May 10 14:00 .bash_logout
-rw-r--r-- 1 ubuntus ubuntus 3771 May 10 14:00 .bashrc
drwx------ 2 ubuntus ubuntus 4096 May 10 14:07 .cache
drwxrwxr-x 2 ubuntus ubuntus 4096 May 10 18:04 HACKED
-rw-r--r-- 1 ubuntus ubuntus  655 May 10 14:00 .profile
ubuntus@ubuntu:~$ _
```

This is the method which I was planned to use to exploit libssh vulnerability. Unfortunately, I was unable to build that server with libssh properly.

Conclusion

This LibSSH vulnerability cve-2018-10933 found in 2018 by the researcher Peter Winter-Smith. The security patches are already provided by the Anderson Sasaki of Red hat and the libssh team. To address this issue libssh version 0.8.4 and libssh 0.7.6 have been released.

So, if someone currently running servers with older versions on or before 2018, they can secure themselves by installing the updated security patches or installing the latest versions.

The most used SSH is Open SSH and Open SSH doesn't share codes with LibSSH. So no one have to panic about this you are running open SSH in your devices.

References

✓ https://www.youtube.com/watch?v=AtGIHSUSV7k&feature=youtu.be

✓ https://www.youtube.com/watch?v=K4OyZGBsPv8

✓ https://www.youtube.com/watch?v=ZSWQjmfcn4g

✓ GitHub. 2020. *Hackerhouse-Opensource/Cve-2018-10933*. [online] Available at: <https://github.com/hackerhouse-opensource/cve-2018-10933> [Accessed 11 May 2020].

✓ Goldberg, D. and Ziv, O., 2020. *Libssh A New Vulnerability Allows Authentication Bypass | Guardicore*. [online] Guardicore - Data Center and Cloud Security. Available at: <https://www.guardicore.com/2018/10/libssh-new-vulnerability-allows-authentication-bypass> [Accessed 11 May 2020].

✓ Nvd.nist.gov. 2020. *NVD - CVE-2018-10933*. [online] Available at: <https://nvd.nist.gov/vuln/detail/CVE-2018-10933#vulnCurrentDescriptionTitle> [Accessed 11 May 2020].

✓ Infopercept.com. 2020. *Blog - Bypassing The Libssh Authentication | Infopercept*. [online] Available at: <https://www.infopercept.com/Bypassing-the-LibSSH-Authentication> [Accessed 11 May 2020].