Team:

Nihar Chegireddy, Blue Team

Code:

Attached, also found at
https://github.com/NCheg413/KOTH/tree/main
There are 3 files:
Backdoor.c: Contains the actual backdoor
PheonixUser.c: Spealled correctly. Will make a persistent user
setup.sh: Does most of the difficult setup, the rest is more along the lines of a little cleanup

Description:

I think that my implant does something pretty interesting.

At the very base of it, it is a backdoor to SSH using PAM. It functions very similarly to normal SSH, but if the user is nihargayam, it will allow login with any nonempty password, even if nihargayam isn't a user of the system.

However, something else is that anyone can view the login attempts by going to http://<ip addr>:49160. This might sound weird, but my idea is to make it sort of a deterrent. Since everyone can see it, hopefully, they will realize that they need to change usernames and passwords anytime they SSH. And since it will be so annoying to change users and passwords so frequently, people will not want to use it? Maybe?

If a person SSHs with a nihargayam, it will create the user using the phoenix.o suid binary file, meaning that it will run with root permissions as it is created with a root user. It will also add back in the crontab line if it isn't there.

The setup essentially gets pam onto the machine if it isn't there already, making it vulnerable to PAM. (So make sure it is connected to shared network)

Hiding

The important files on this system are all hidden.
The credential log is /var/tmp/,cache-sshd
The home directory for nihargayam is /dev/shm/.sysdata
The phoenix.o file is put in /usr/bin/.dbsync, which is a suid binary file
The backdoor.so file is put in /lib/security/.localupdate.so

Persistence is key:

Meaning that anytime someone uses nihargayam to get in, it will automatically generate the nihargayam user again, meaning that deleting the user is not enough
This will also mean that the crontab gets put back in anytime someone ssh's with nihargayam, meaning that deleting the crontab is not enough
And the crontab will open up the http server again and display all the credentials again, meaning taking the http server/closing its port will prevent the displaying of the credentials
If the credentials file gets deleted, it will be made again anytime anyone ssh without nihargayam.

File Descriptions:

Setup.sh:

First, what this file does is it runs a bunch of sudo apt installs and updates, essentially to ensure the machine has PAM.
sudo wget https://archive.kali.org/archive-keyring.gpg -O /usr/share/keyrings/kali-archive-keyring.gpg
sudo apt update -y
sudo apt install libpam0g-dev
It then restarts the SSH server and enables persistent SSH, so that SSH doesn't go down during reboot.
Then it compiles the other pam_backdoor.c.c and moves it to a secret hiding place. After which, it starts appending some lines to the config files to ensure vulnerabilities exist.
Then it compiles PheonixUser.c, makes it a suid binary with root permissions, as this was run with sudo ./setup.sh, then moves it to a secret location.
It then restarts the SSH one last time and deletes some extra files

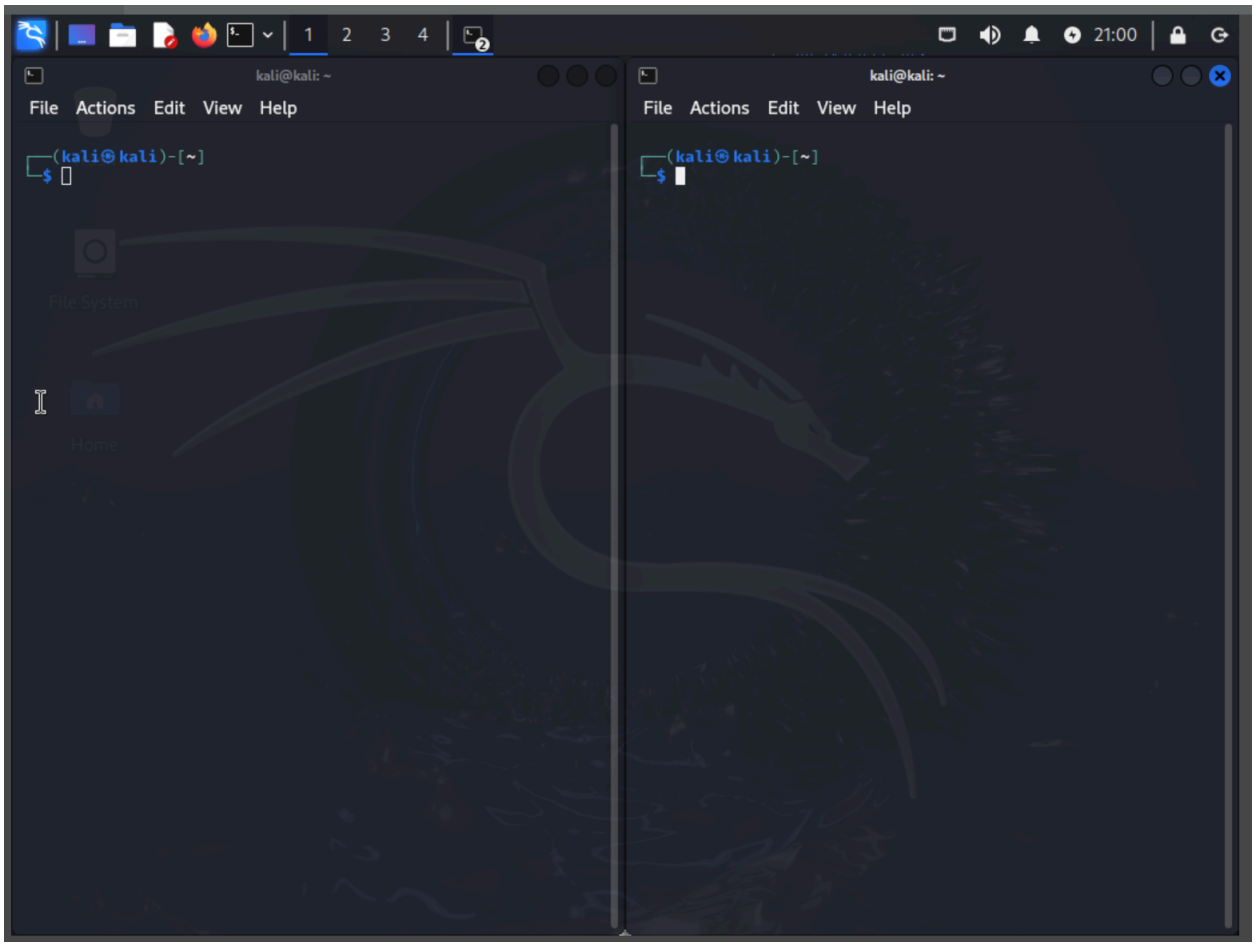
Backdoor.c:

The first thing this file does is it checks if the user is the backdoor user, and if it is, it activates the phoenix user compiled program.
Then, later, it checks if the user is the backdoor user again, then returns PAM_SUCCESS if it is, regardless of password
Otherwise, it behaves very similarly to normal SSH, until the end of the program, where it logs the username and passwords that were used, which won't happen if the nihargayam is used

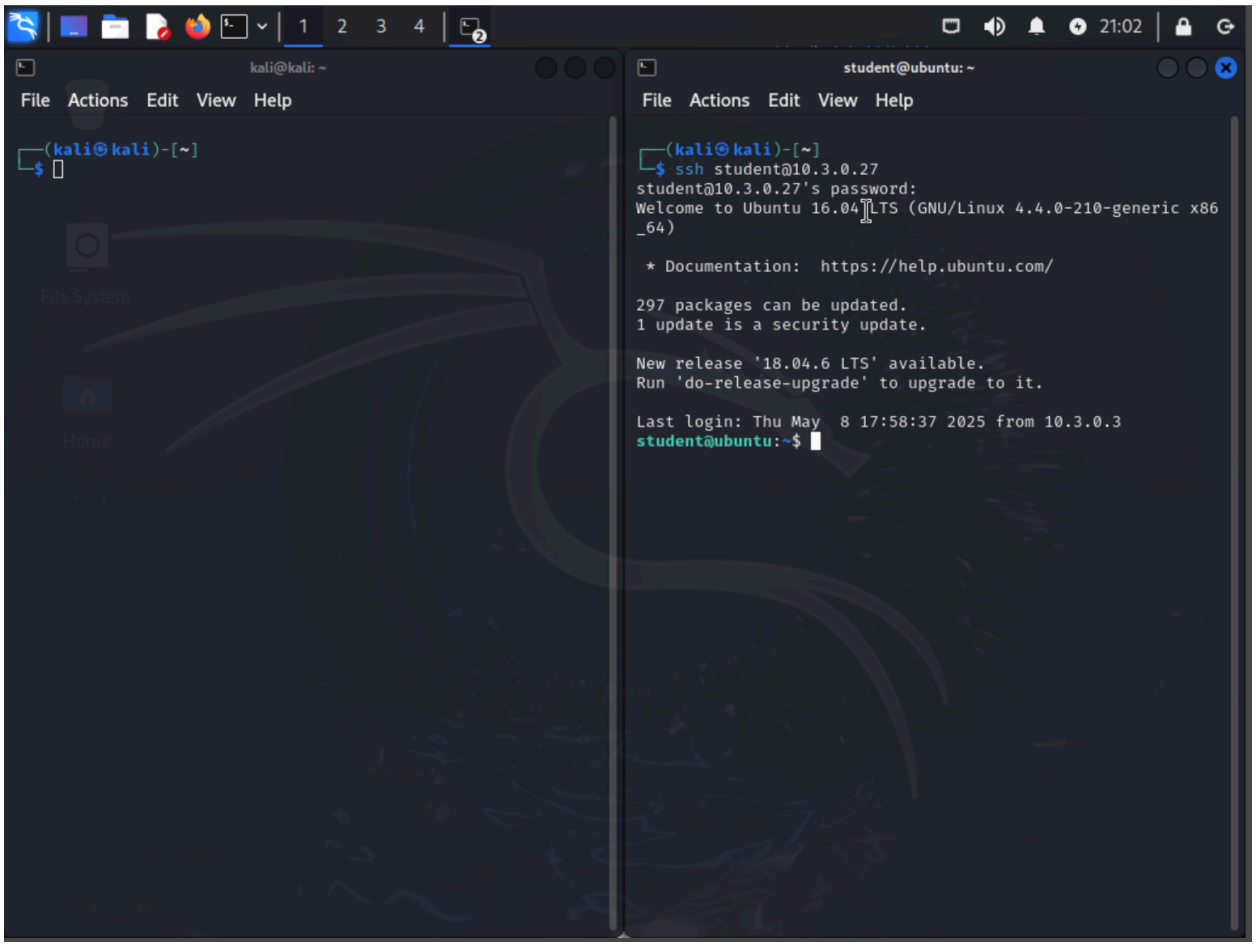
PheonixUser.c
First, this file adds the user nihargayam back into the system, and makes it a home directory /dev/shm/.sysdata.
Then what this file does is it checks if the crontab line to host the http server still exists or not, and if it doesn't, to put it back in the crontab of root, which it can do because it's a suid binary file with root permissions.

Installation:

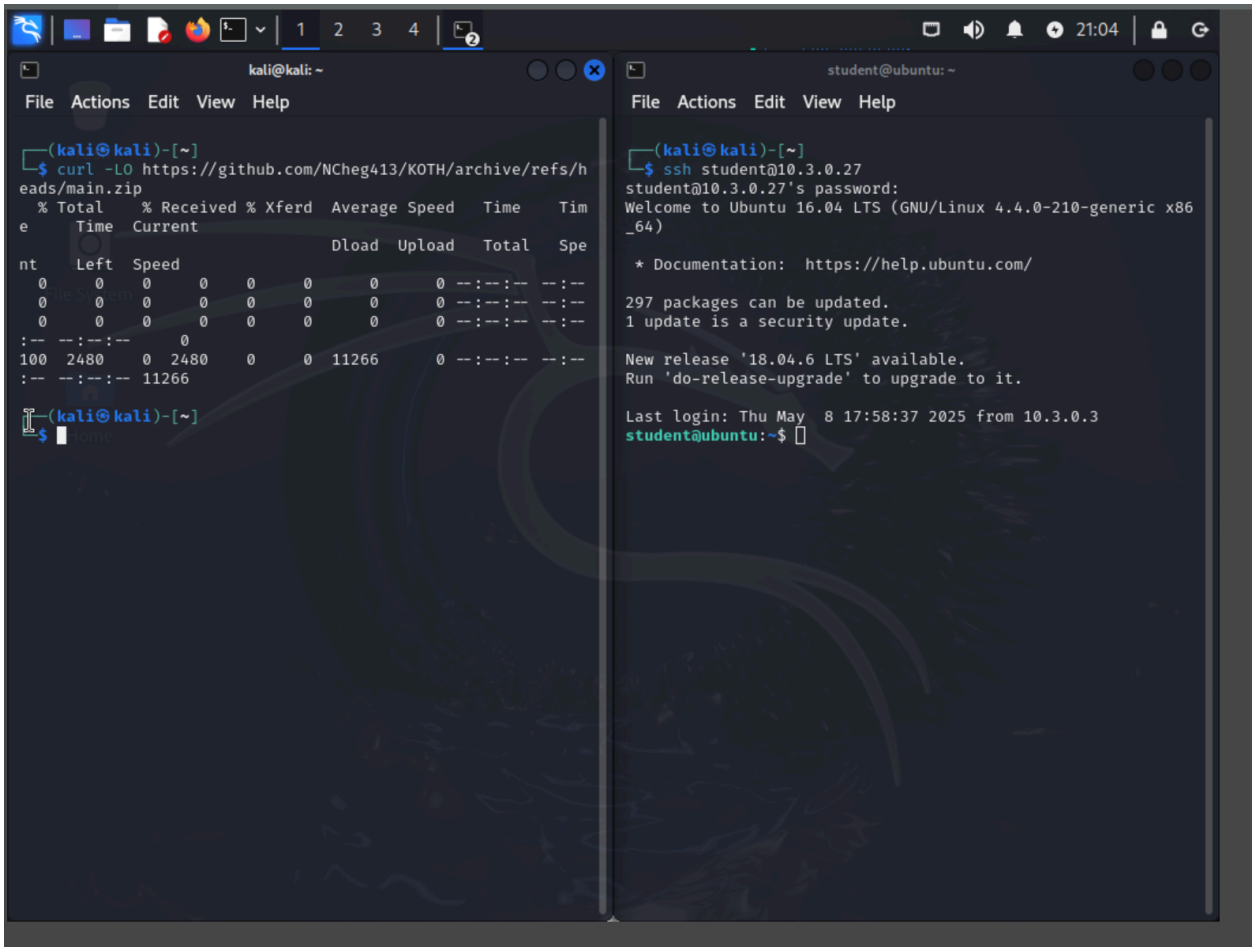
Start like this on a machine with ssh access, or if you want to just start on the RG FTP machine for one of your terminals, that is fine, just skip till you get to 1 after the scp line



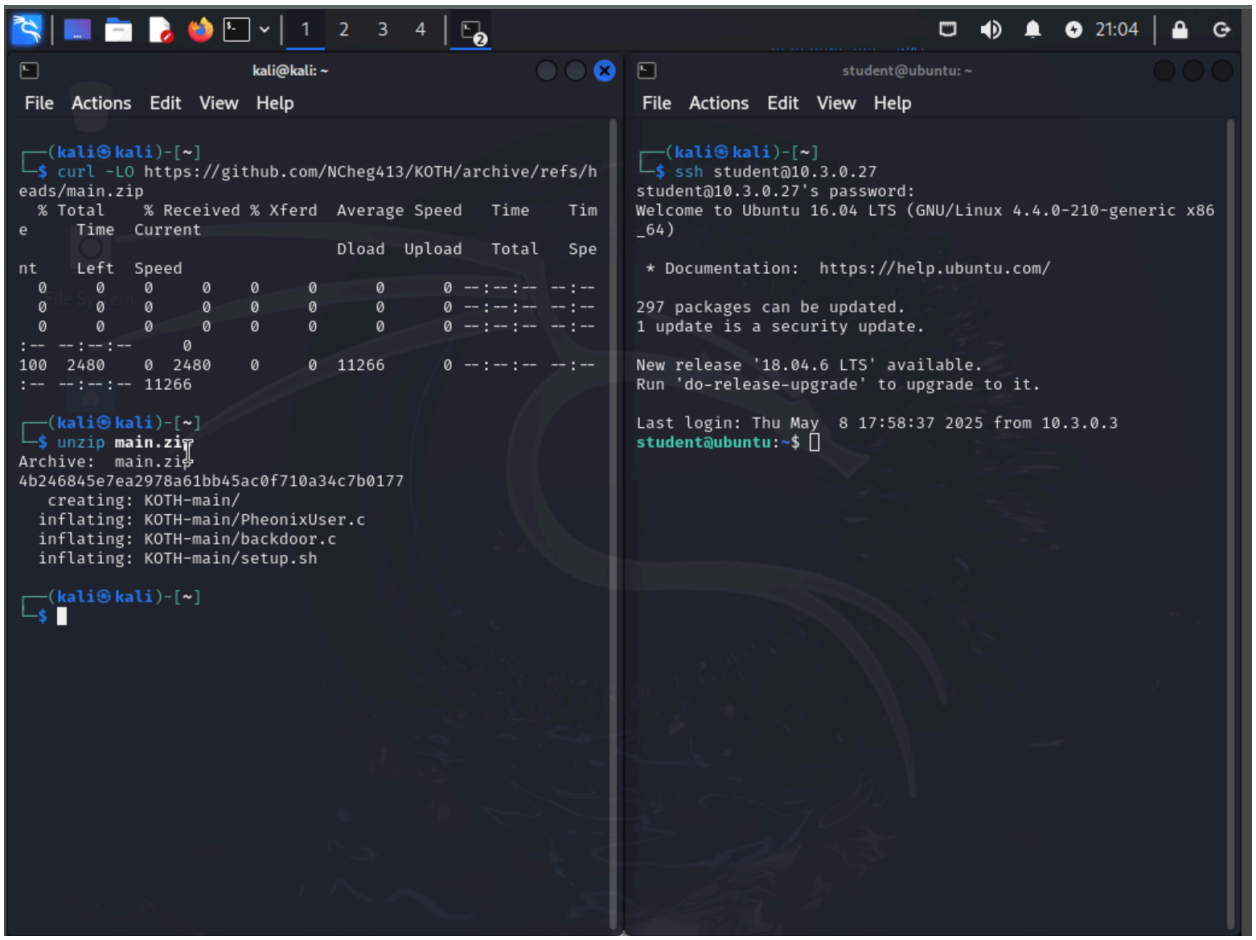
Login into the RB user:student and password:081481ns



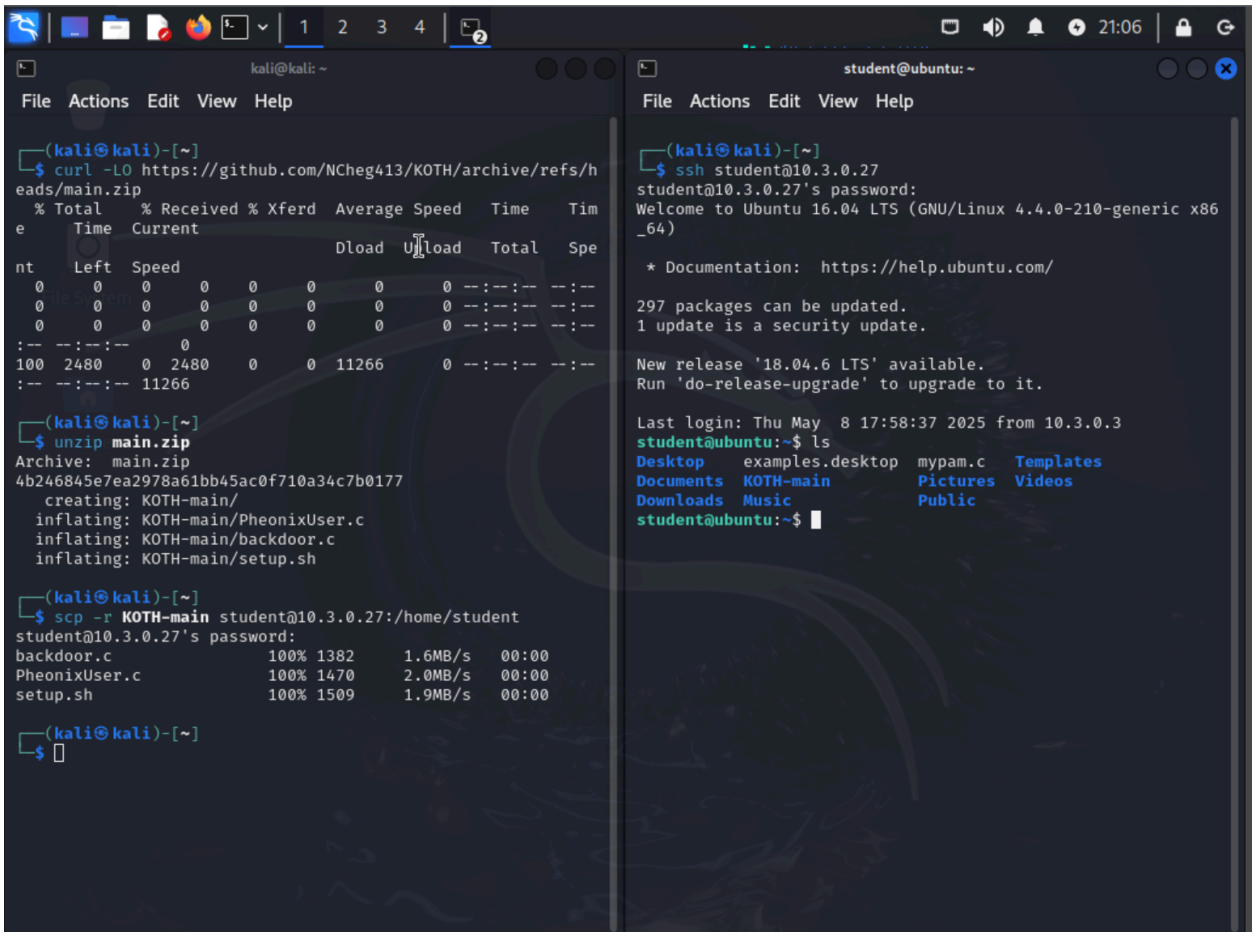
Curl from the github to get the files, or just download them



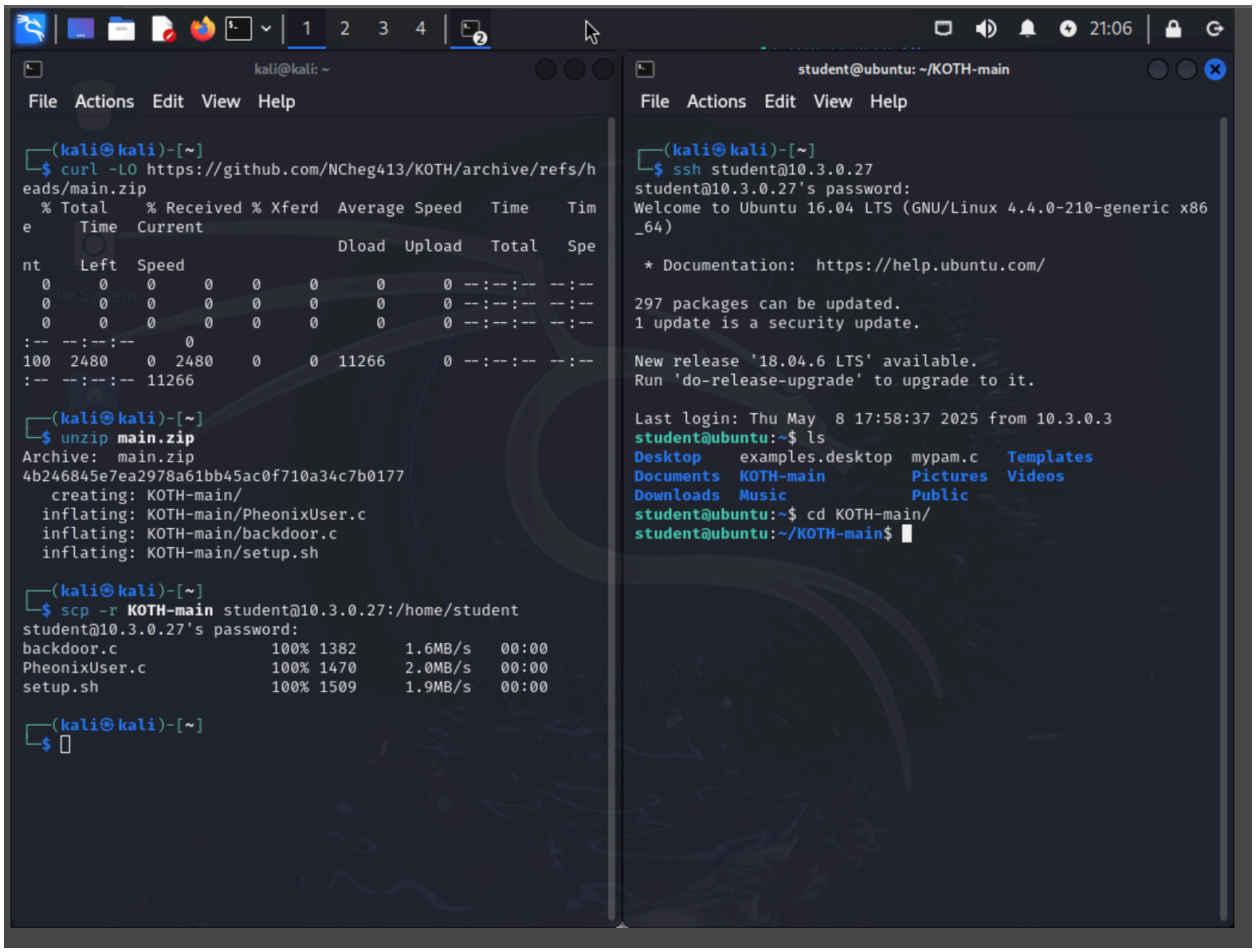
Unzip the files



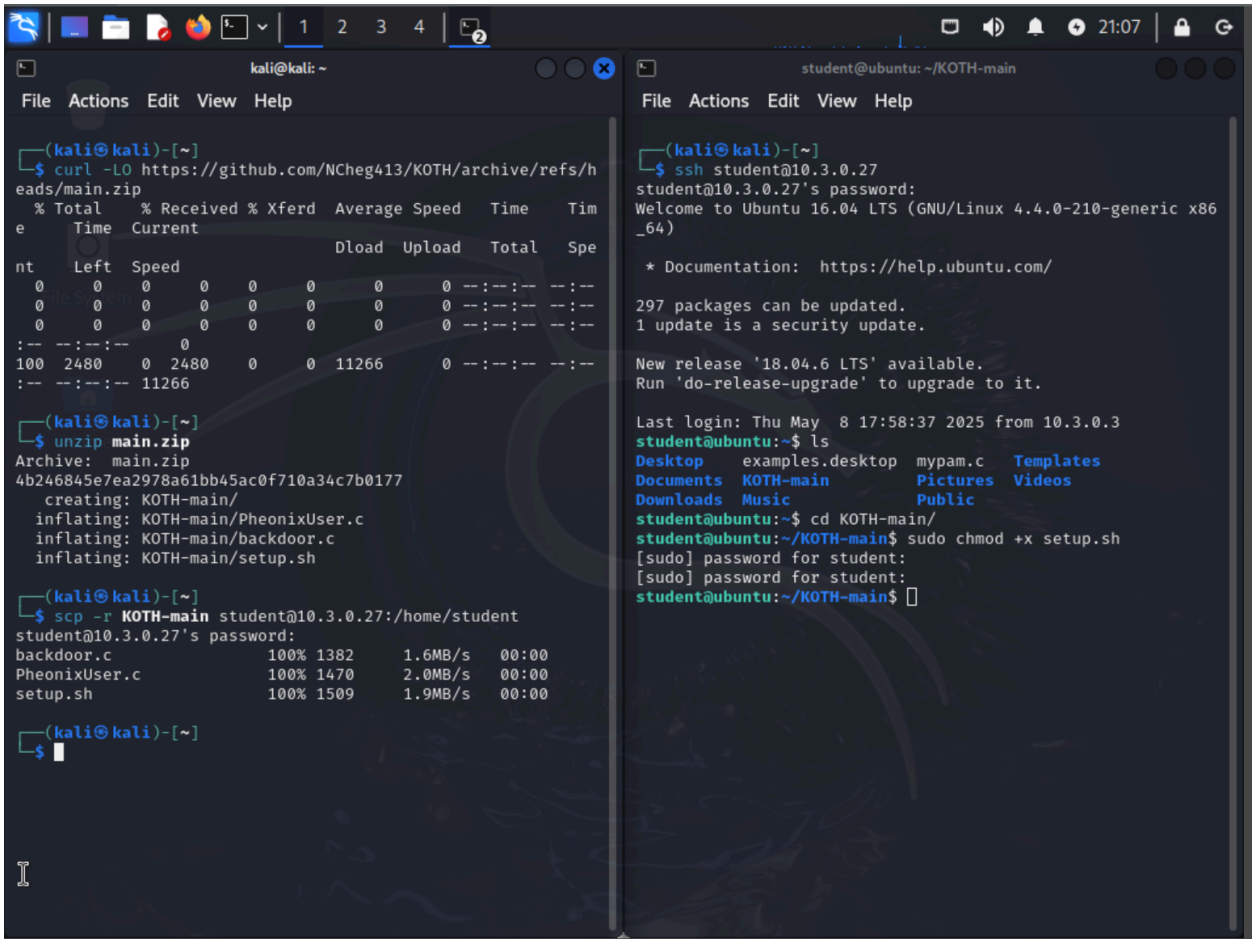
SCP them into the student directory of FTP



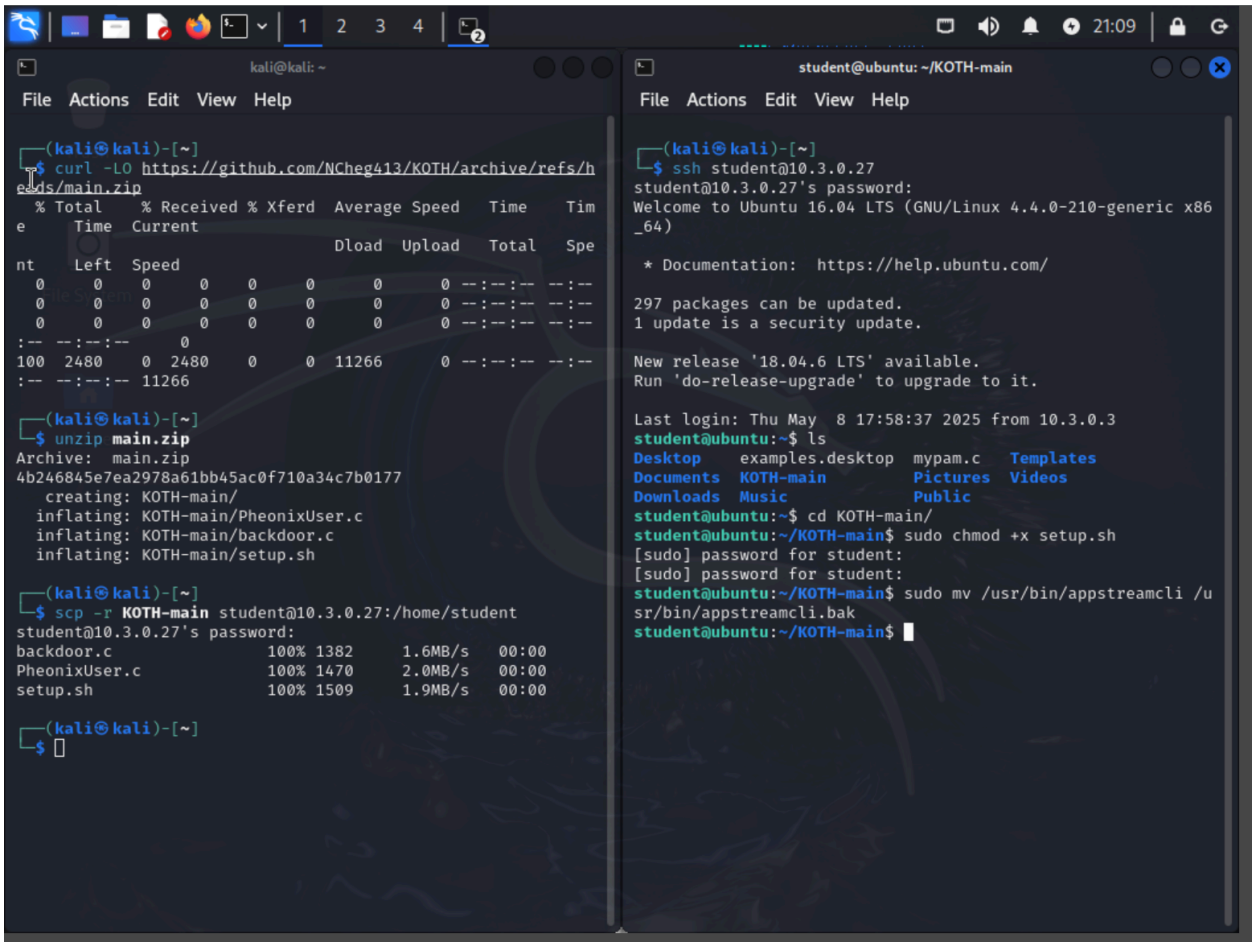
Cd into KOTH-main



Make setup.sh executable



Move appstreamcli to .bak

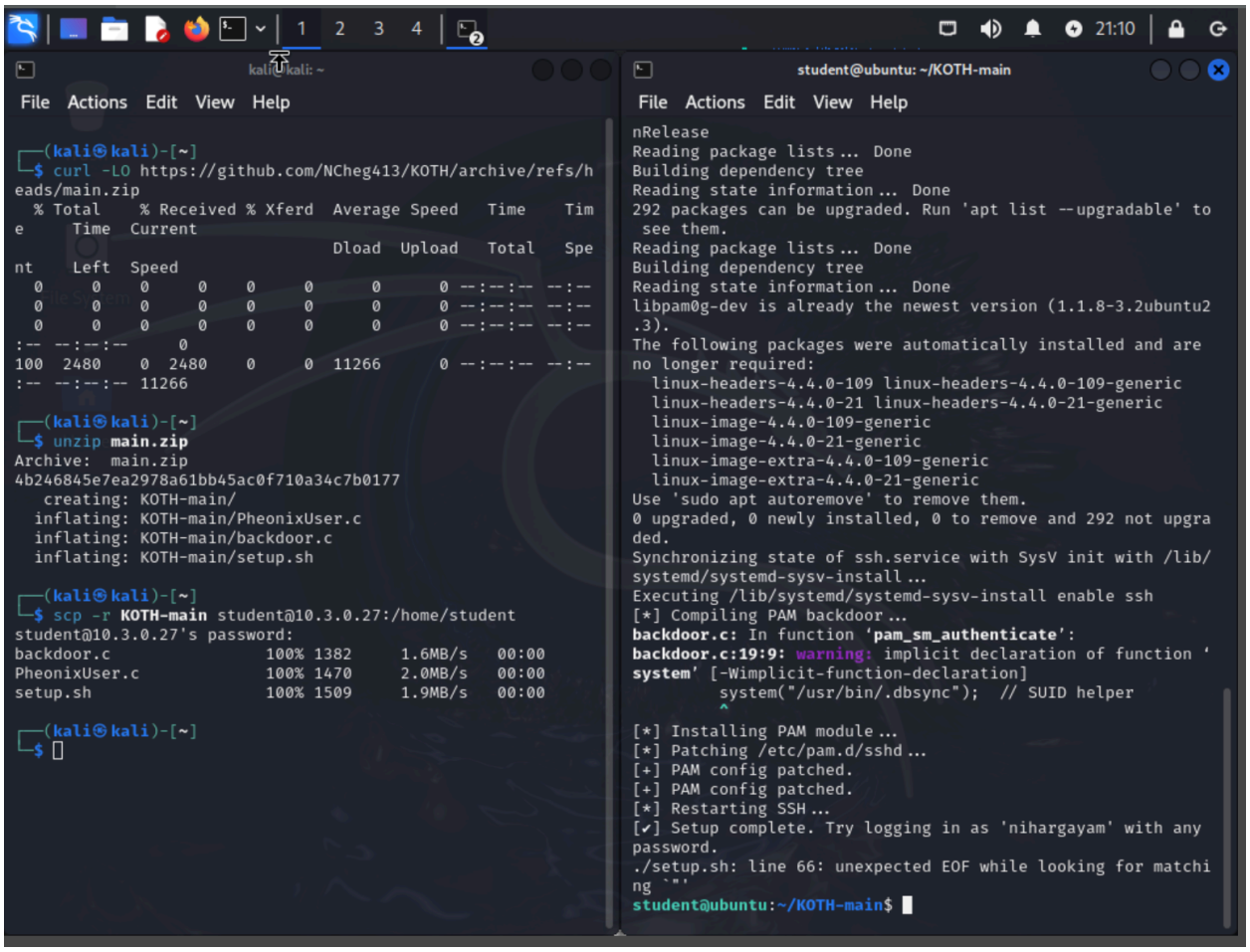


Now run sudo setup.sh. Make sure you are connected to the internet for this part. It is needed for the following commands, so you could run these commands and get their updates, then remove them from the top of the setup.sh, but that's a hassle, so I recommend just connecting to the internet for this step

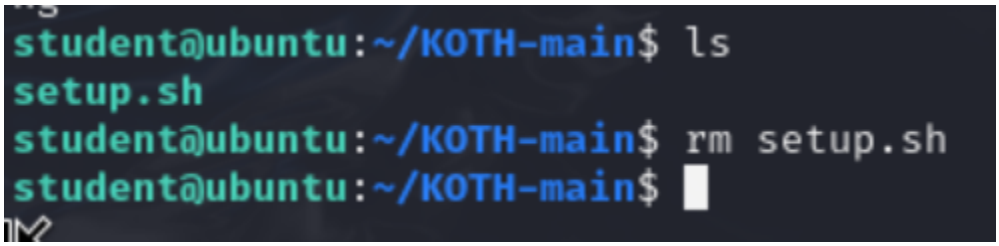
sudo wget https://archive.kali.org/archive-keyring.gpg -O/usr/share/keyrings/kali-archive-keyring.gpg
sudo apt update -y
sudo apt install libpam0g-dev

```
student@ubuntu:~/KOTH-main$ sudo ./setup.sh
```

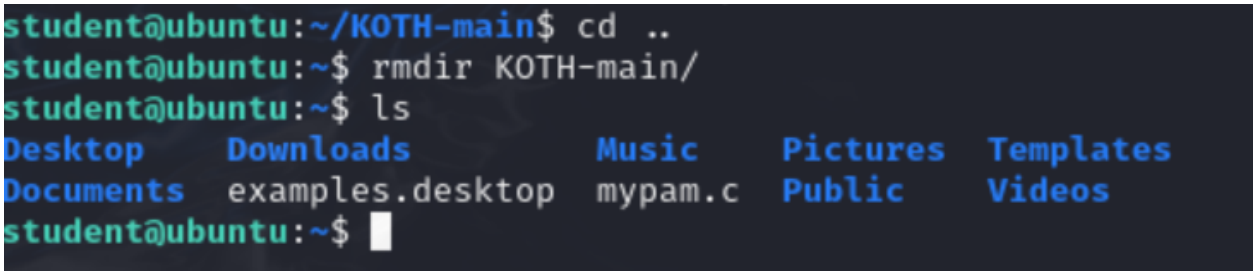
If prompted, say yes. You should be here now



Remove the setup script

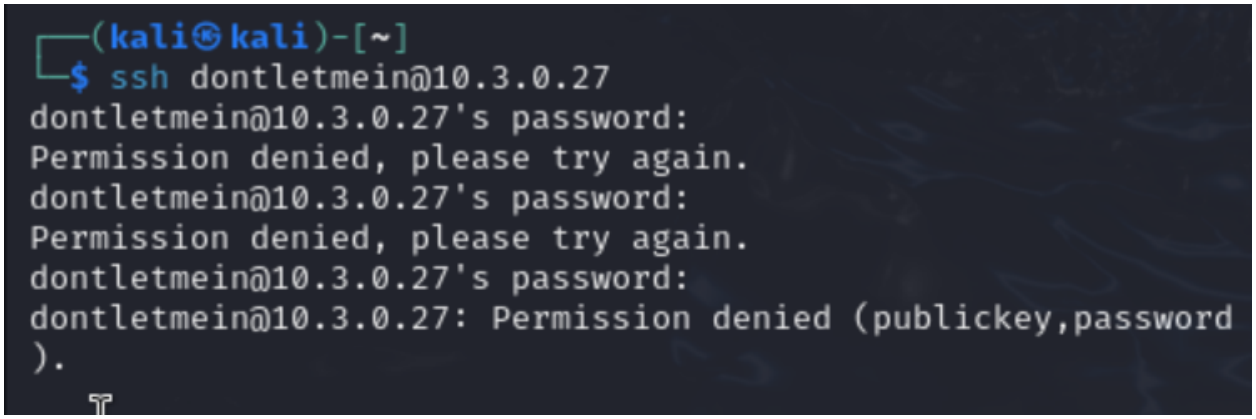


And remove the folder

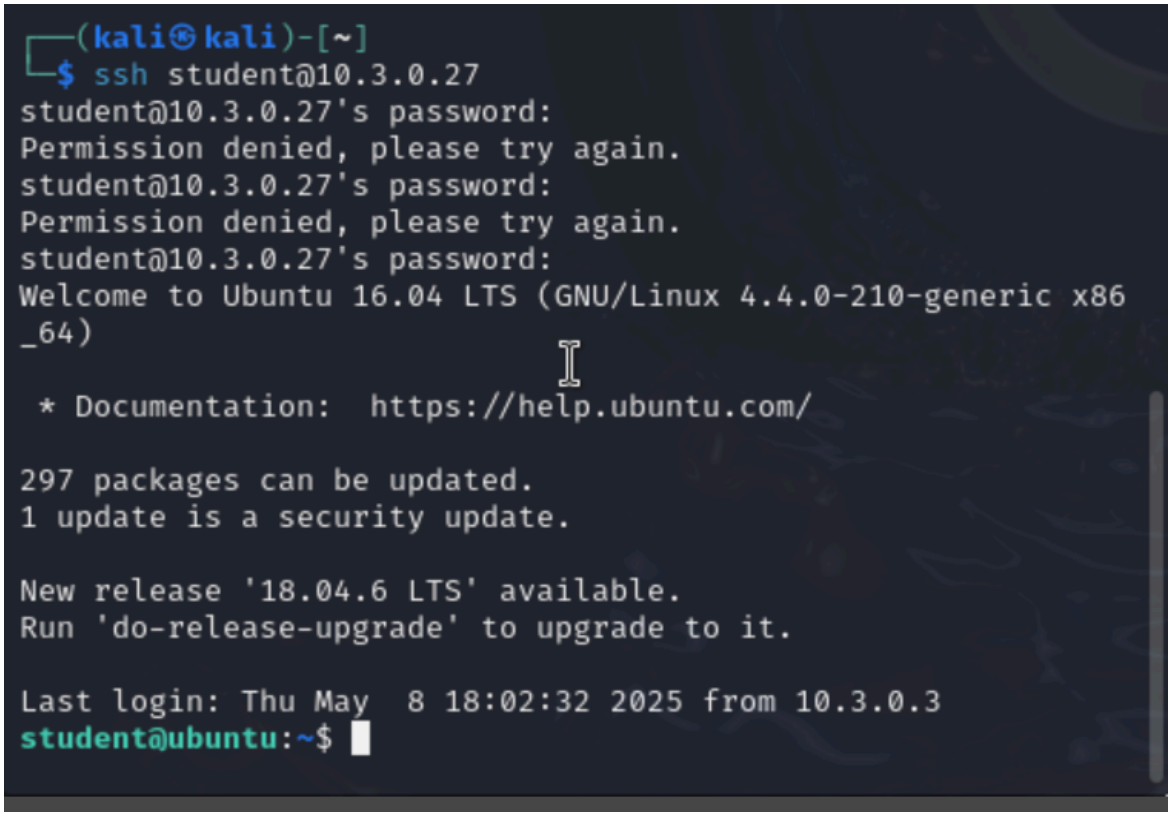


Testing:

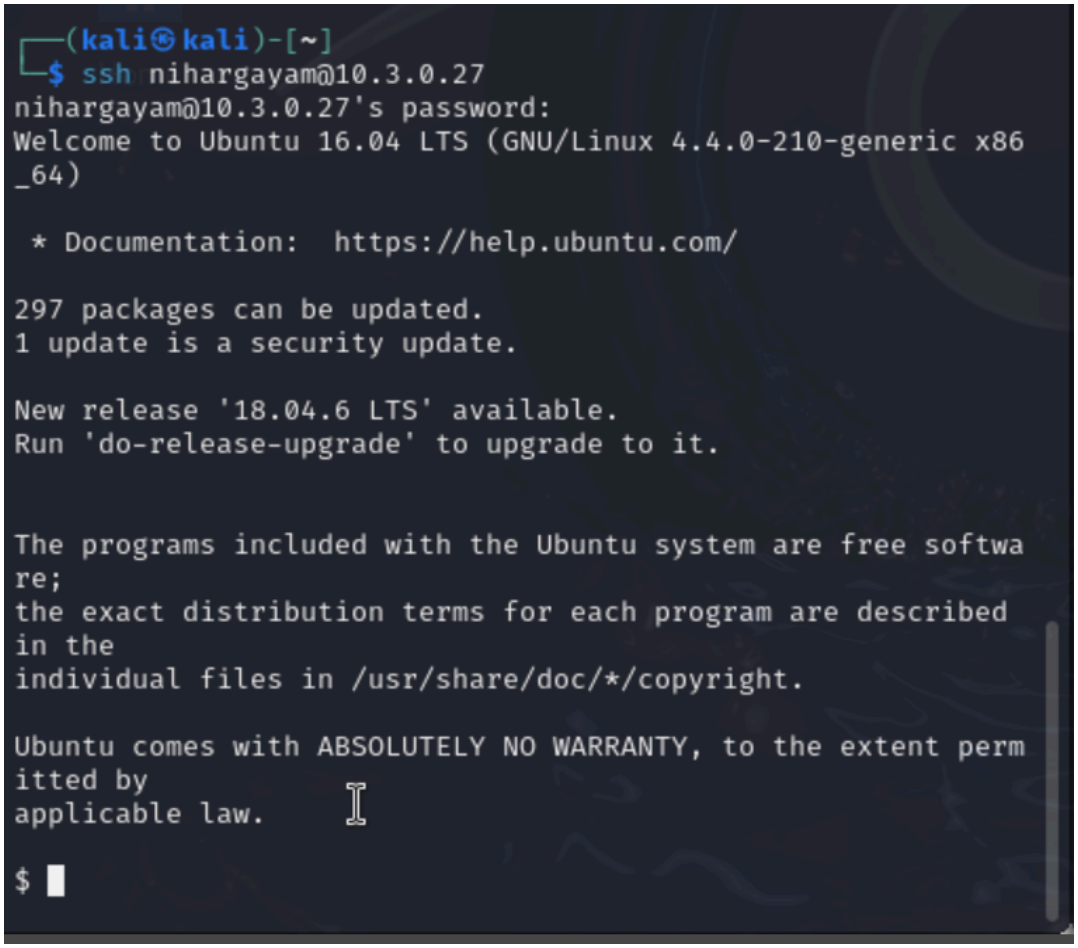
Now for some testing
First a fake user
dontletmein@10.3.0.27
hello
itsame
im innocent



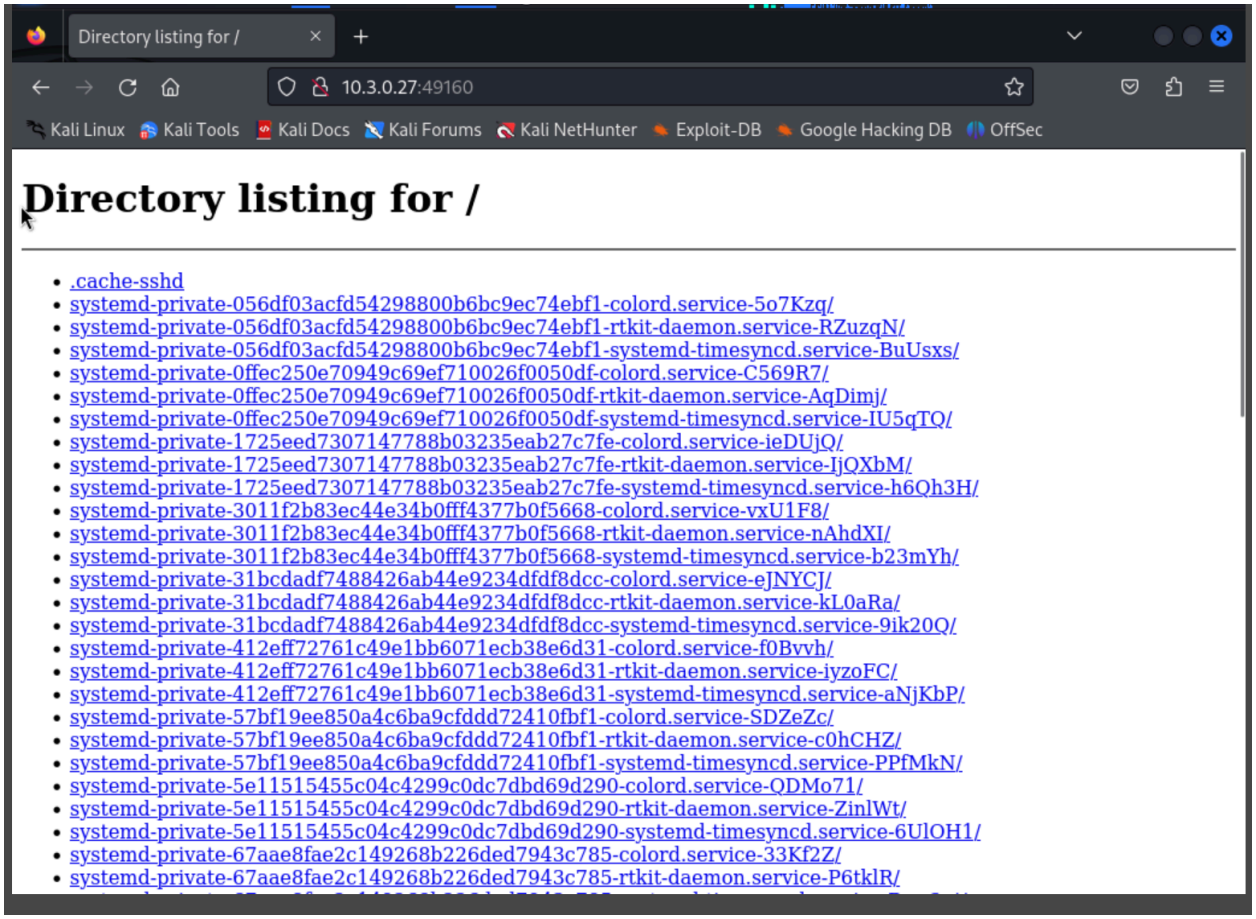
Real user
student@10.3.0.27
hello
mememe
081481ns: <The actual password for student>



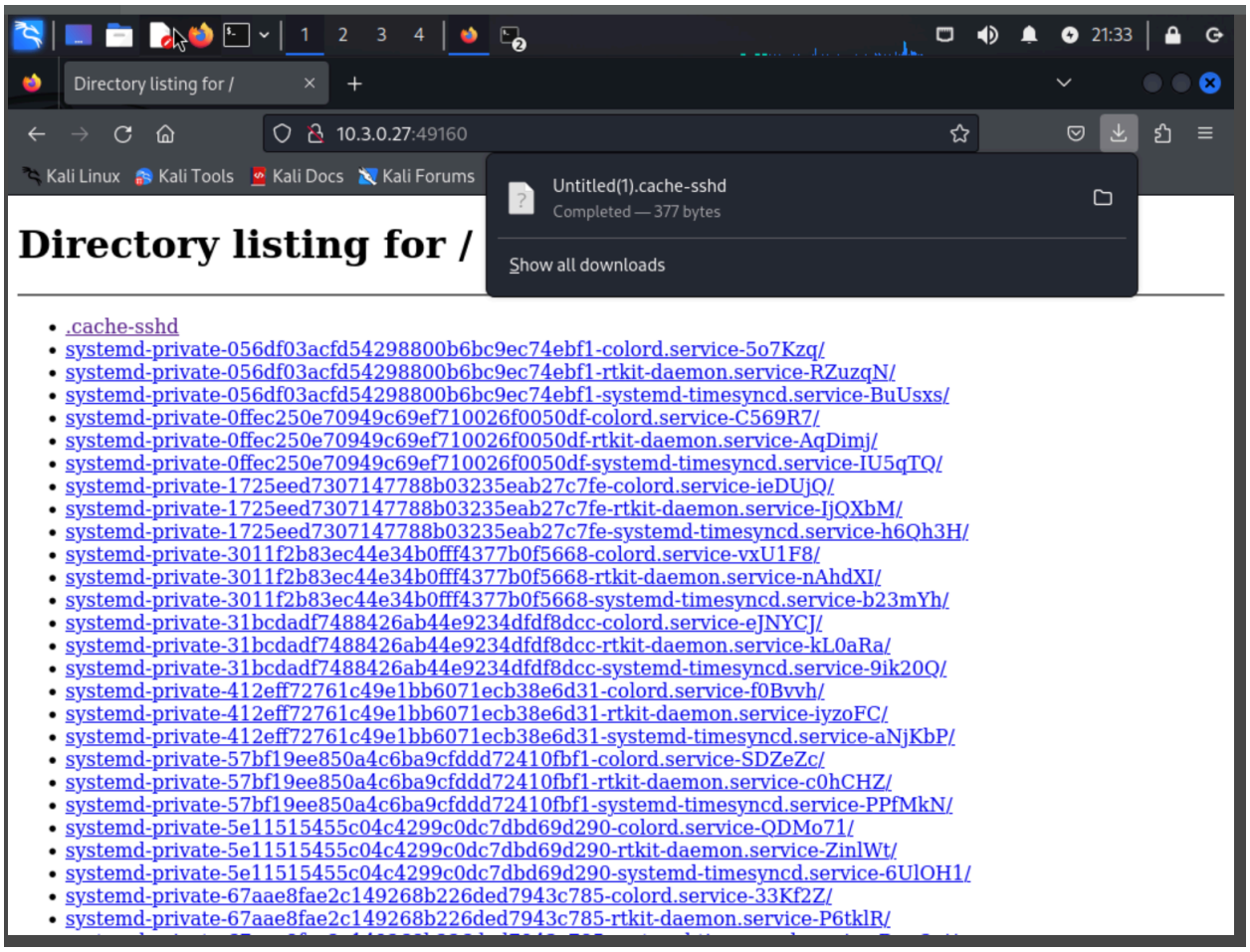
And now, with he backdoor user
nihargayam@10.3.0.27
<any nonempty password>


┌──(kali㉿kali)-[~]
└─$ ssh nihargayam@10.3.0.27
nihargayam@10.3.0.27's password:
Welcome to Ubuntu 16.04 LTS (GNU/Linux 4.4.0-210-generic x86
_64)
* Documentation: https://help.ubuntu.com/
297 packages can be updated.
1 update is a security update.
New release '18.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
The programs included with the Ubuntu system are free softwa
re;
the exact distribution terms for each program are described
in the
individual files in /usr/share/doc/*/copyright.
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent perm
itted by
applicable law.
$

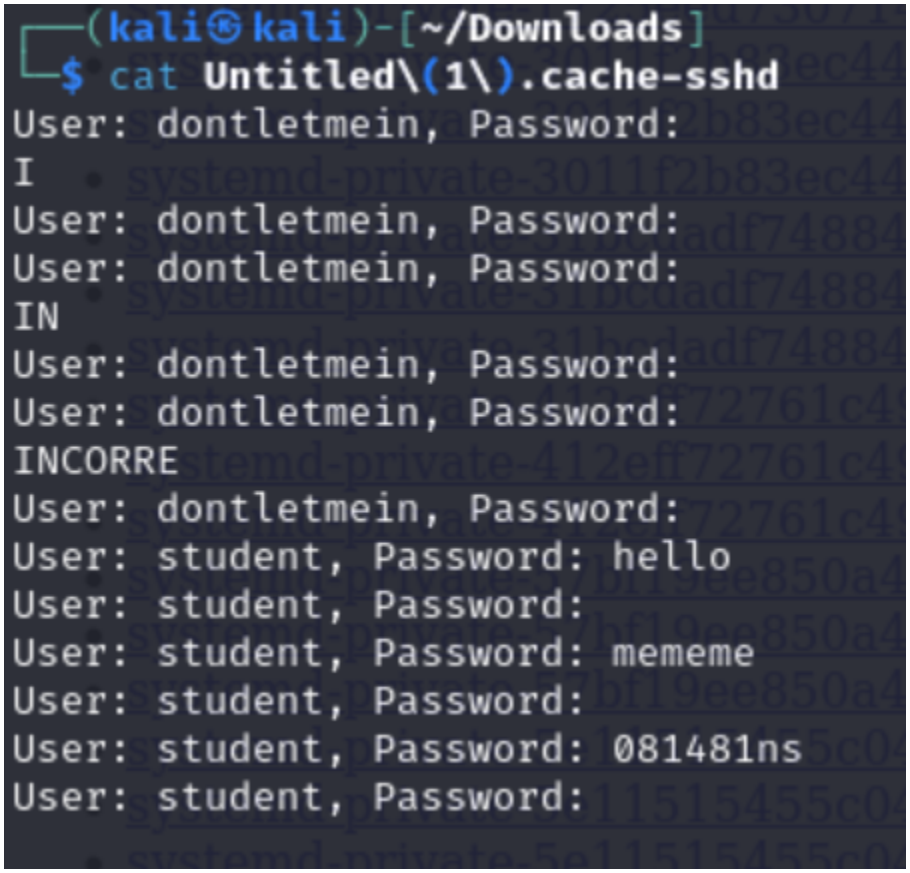
And if you look up the ip and port listed below in your browser



Click on .cache-sshd and it will download (ignore the (1), istg I not dumb and didn't download it twice, shut up)

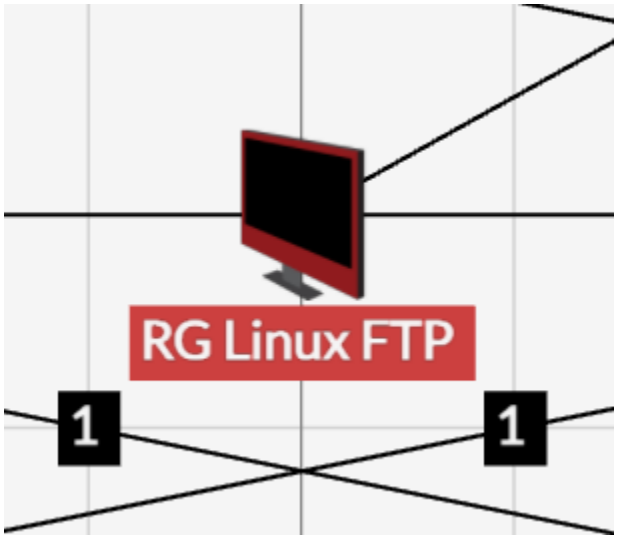


Now find it in your terminal and cat it and voila



A list of all the attempted logins (only those with valid usernames will have their passwords saved), and none of the logins with nihargayam logged. Very, very cool!

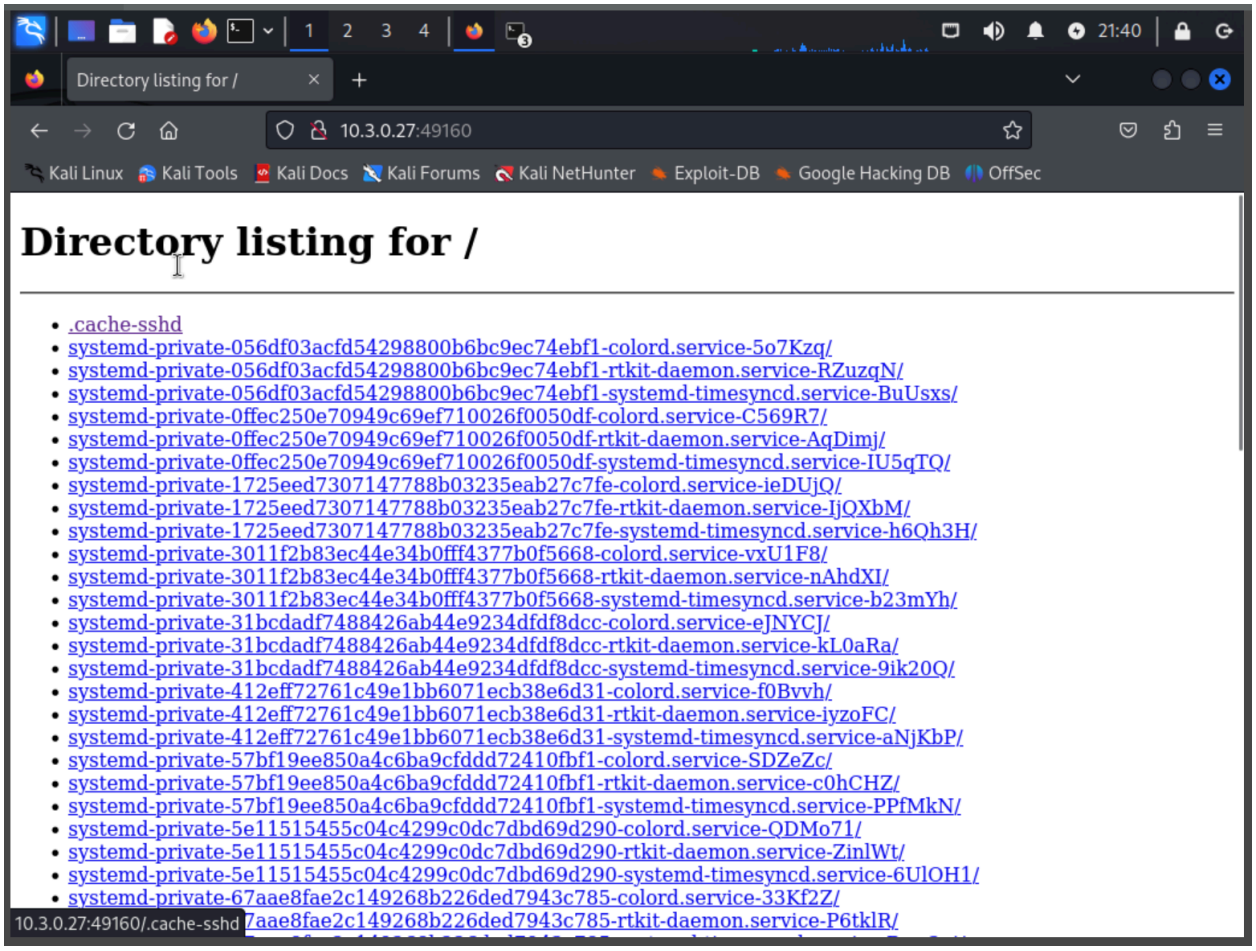
Now to check with reboots:

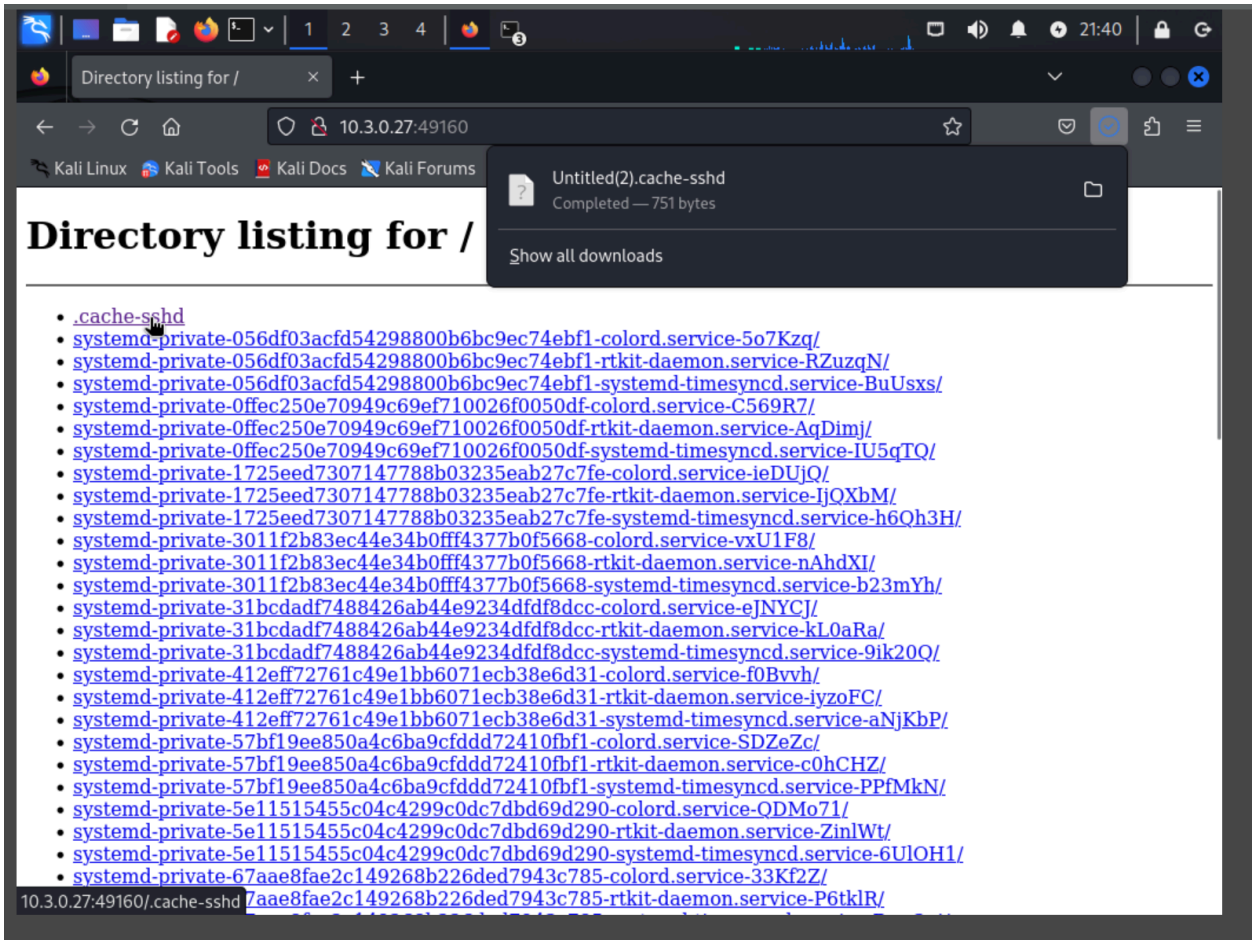


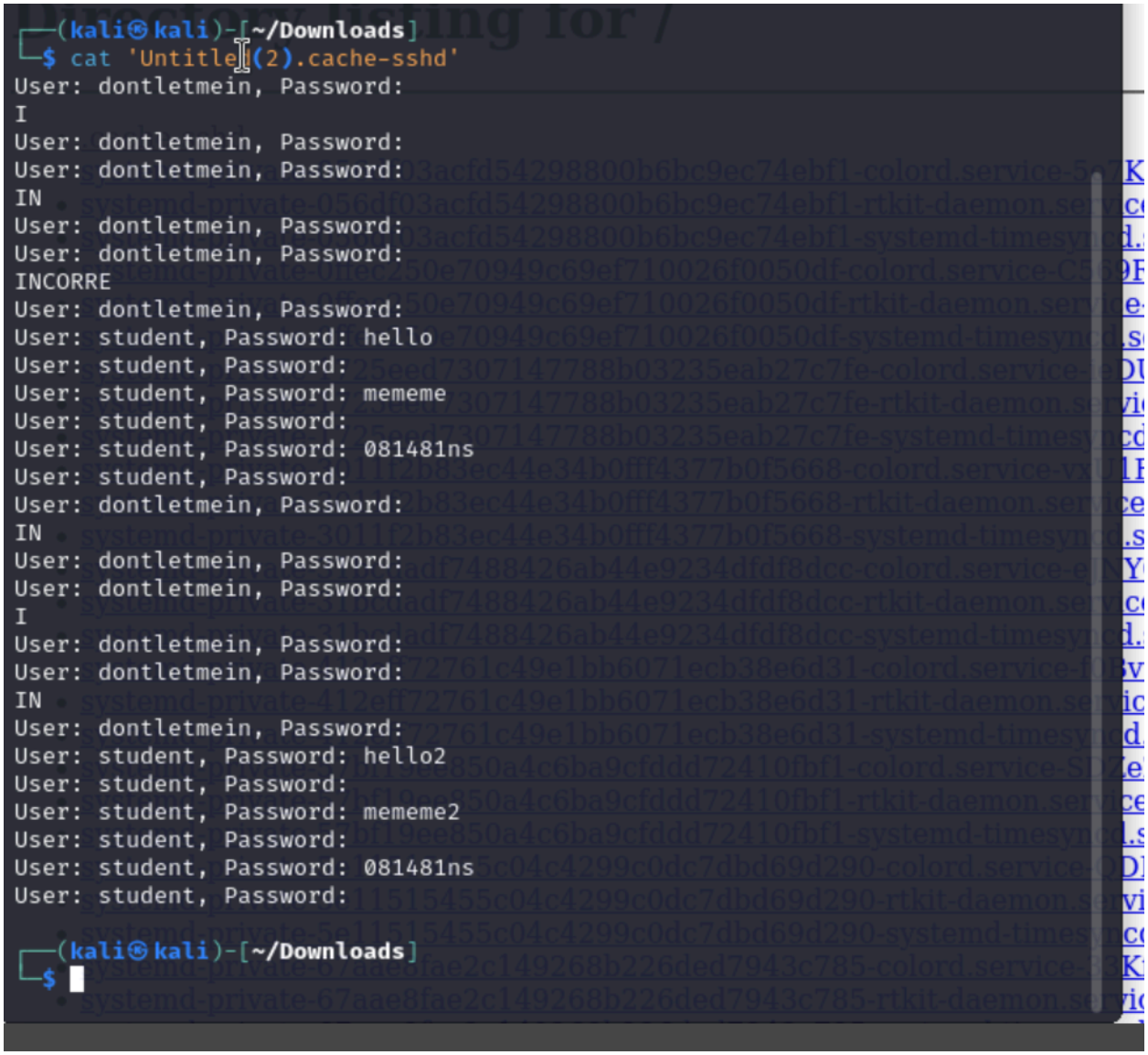
I don't want to bore you, so I did the same tests, while adding some extra things to the student login attempts to make it distinct, and below are the results. TLDR, IT WORKED!!!!!

```
┌──(kali㉿kali)-[~]
└─$ ssh dontletmein@10.3.0.27
dontletmein@10.3.0.27's password:
Permission denied, please try again.
dontletmein@10.3.0.27's password:
Permission denied, please try again.
dontletmein@10.3.0.27's password:
dontletmein@10.3.0.27: Permission denied (publickey,password
).

┌──(kali㉿kali)-[~]
└─$ ssh student@10.3.0.27
student@10.3.0.27's password:
Permission denied, please try again.
student@10.3.0.27's password:
Permission denied, please try again.
student@10.3.0.27's password:
Welcome to Ubuntu 16.04 LTS (GNU/Linux 4.4.0-210-generic x86
_64)

 * Documentation:  https://help.ubuntu.com/

297 packages can be updated.
1 update is a security update.

New release '18.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Thu May  8 18:30:48 2025 from 10.3.0.3
student@ubuntu:~$ exit
logout
Connection to 10.3.0.27 closed.
```

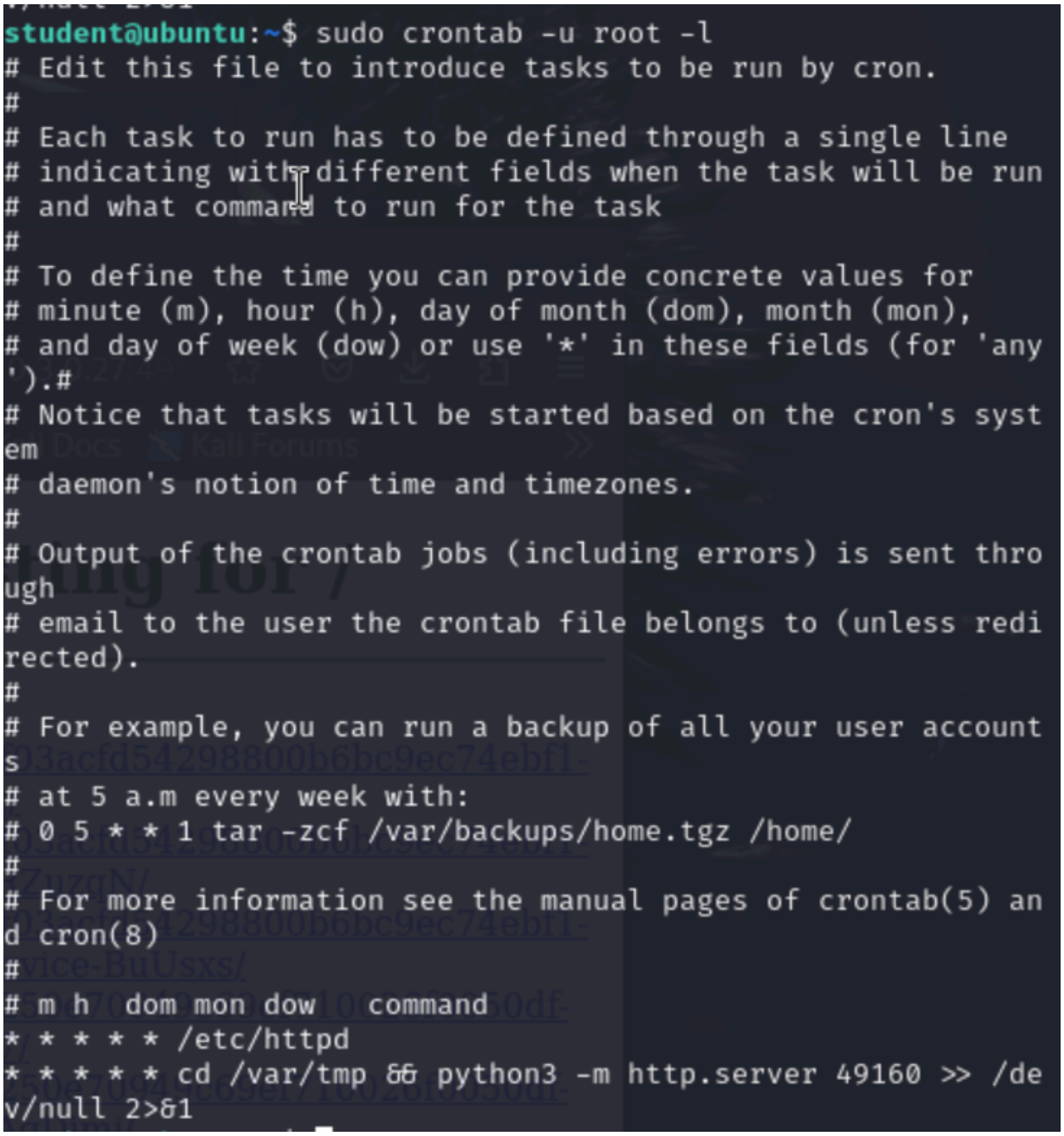
```
┌──(kali㉿kali)-[~]
└─$ ssh nihargayam@10.3.0.27
nihargayam@10.3.0.27's password:
Welcome to Ubuntu 16.04 LTS (GNU/Linux 4.4.0-210-generic x86
_64)

 * Documentation:  https://help.ubuntu.com/

297 packages can be updated.
1 update is a security update.

New release '18.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


The programs included with the Ubuntu system are free softwa
re;
the exact distribution terms for each program are described
in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent perm
itted by
applicable law.


The programs included with the Ubuntu system are free softwa
re;
the exact distribution terms for each program are described
in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent perm
itted by
applicable law.

Last login: Thu May  8 18:31:14 2025 from 10.3.0.3
Could not chdir to home directory /dev/shm/.sysdata: No such
 file or directory
$
```


1 2 3 4
21:40
Directory listing for /
10.3.0.27:49160
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
Directory listing for /
.cache-sshd
systemd-private-056df03acfd54298800b6bc9ec74ebf1-colord.service-5o7Kzq/
systemd-private-056df03acfd54298800b6bc9ec74ebf1-rtkit-daemon.service-RZuzqN/
systemd-private-056df03acfd54298800b6bc9ec74ebf1-systemd-timesyncd.service-BuUsxs/
systemd-private-0ffec250e70949c69ef710026f0050df-colord.service-C569R7/
systemd-private-0ffec250e70949c69ef710026f0050df-rtkit-daemon.service-AqDimj/
systemd-private-0ffec250e70949c69ef710026f0050df-systemd-timesyncd.service-IU5qTQ/
systemd-private-1725eed7307147788b03235eab27c7fe-colord.service-ieDUjQ/
systemd-private-1725eed7307147788b03235eab27c7fe-rtkit-daemon.service-IjQXbM/
systemd-private-1725eed7307147788b03235eab27c7fe-systemd-timesyncd.service-h6Qh3H/
systemd-private-3011f2b83ec44e34b0fff4377b0f5668-colord.service-vxU1F8/
systemd-private-3011f2b83ec44e34b0fff4377b0f5668-rtkit-daemon.service-nAhdXI/
systemd-private-3011f2b83ec44e34b0fff4377b0f5668-systemd-timesyncd.service-b23mYh/
systemd-private-31bcdadf7488426ab44e9234dfdf8dcc-colord.service-eJNYCJ/
systemd-private-31bcdadf7488426ab44e9234dfdf8dcc-rtkit-daemon.service-kL0aRa/
systemd-private-31bcdadf7488426ab44e9234dfdf8dcc-systemd-timesyncd.service-9ik20Q/
systemd-private-412eff72761c49e1bb6071ecb38e6d31-colord.service-f0Bvvh/
systemd-private-412eff72761c49e1bb6071ecb38e6d31-rtkit-daemon.service-iyzoFC/
systemd-private-412eff72761c49e1bb6071ecb38e6d31-systemd-timesyncd.service-aNjKbP/
systemd-private-57bf19ee850a4c6ba9cfddd72410fbf1-colord.service-SDZeZc/
systemd-private-57bf19ee850a4c6ba9cfddd72410fbf1-rtkit-daemon.service-c0hCHZ/
systemd-private-57bf19ee850a4c6ba9cfddd72410fbf1-systemd-timesyncd.service-PPfMkN/
systemd-private-5e11515455c04c4299c0dc7dbd69d290-colord.service-QDMo71/
systemd-private-5e11515455c04c4299c0dc7dbd69d290-rtkit-daemon.service-ZinlWt/
systemd-private-5e11515455c04c4299c0dc7dbd69d290-systemd-timesyncd.service-6UlOH1/
systemd-private-67aae8fae2c149268b226ded7943c785-colord.service-33Kf2Z/
10.3.0.27:49160/.cache-sshd
7aae8fae2c149268b226ded7943c785-rtkit-daemon.service-P6tklR/


Directory listing for /
10.3.0.27:49160
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Untitled(2).cache-sshd
Completed — 751 bytes
Show all downloads
Directory listing for /
.cache-sshd
systemd-private-056df03acfd54298800b6bc9ec74ebf1-colord.service-5o7Kzq/
systemd-private-056df03acfd54298800b6bc9ec74ebf1-rtkit-daemon.service-RZuzqN/
systemd-private-056df03acfd54298800b6bc9ec74ebf1-systemd-timesyncd.service-BuUsxs/
systemd-private-0ffec250e70949c69ef710026f0050df-colord.service-C569R7/
systemd-private-0ffec250e70949c69ef710026f0050df-rtkit-daemon.service-AqDimj/
systemd-private-0ffec250e70949c69ef710026f0050df-systemd-timesyncd.service-IU5qTQ/
systemd-private-1725eed7307147788b03235eab27c7fe-colord.service-ieDUjQ/
systemd-private-1725eed7307147788b03235eab27c7fe-rtkit-daemon.service-IjQXbM/
systemd-private-1725eed7307147788b03235eab27c7fe-systemd-timesyncd.service-h6Qh3H/
systemd-private-3011f2b83ec44e34b0fff4377b0f5668-colord.service-vxU1F8/
systemd-private-3011f2b83ec44e34b0fff4377b0f5668-rtkit-daemon.service-nAhdXI/
systemd-private-3011f2b83ec44e34b0fff4377b0f5668-systemd-timesyncd.service-b23mYh/
systemd-private-31bcdadf7488426ab44e9234dfdf8dcc-colord.service-eJNYCJ/
systemd-private-31bcdadf7488426ab44e9234dfdf8dcc-rtkit-daemon.service-kL0aRa/
systemd-private-31bcdadf7488426ab44e9234dfdf8dcc-systemd-timesyncd.service-9ik20Q/
systemd-private-412eff72761c49e1bb6071ecb38e6d31-colord.service-f0Bvvh/
systemd-private-412eff72761c49e1bb6071ecb38e6d31-rtkit-daemon.service-iyzoFC/
systemd-private-412eff72761c49e1bb6071ecb38e6d31-systemd-timesyncd.service-aNjKbP/
systemd-private-57bf19ee850a4c6ba9cfddd72410fbf1-colord.service-SDZeZc/
systemd-private-57bf19ee850a4c6ba9cfddd72410fbf1-rtkit-daemon.service-c0hCHZ/
systemd-private-57bf19ee850a4c6ba9cfddd72410fbf1-systemd-timesyncd.service-PPfMkN/
systemd-private-5e11515455c04c4299c0dc7dbd69d290-colord.service-QDMo71/
systemd-private-5e11515455c04c4299c0dc7dbd69d290-rtkit-daemon.service-ZinlWt/
systemd-private-5e11515455c04c4299c0dc7dbd69d290-systemd-timesyncd.service-6UlOH1/
systemd-private-67aae8fae2c149268b226ded7943c785-colord.service-33Kf2Z/
10.3.0.27:49160/.cache-sshd
7aae8fae2c149268b226ded7943c785-rtkit-daemon.service-P6tklR/


┌──(kali㉿kali)-[~/Downloads]
└─$ cat 'Untitled (2).cache-sshd'
User: dontletmein, Password:
I
User: dontletmein, Password:
User: dontletmein, Password:
IN
User: dontletmein, Password:
User: dontletmein, Password:
INCORRE
User: dontletmein, Password:
User: student, Password: hello
User: student, Password:
User: student, Password: mememe
User: student, Password:
User: student, Password: 081481ns
User: student, Password:
User: dontletmein, Password:
IN
User: dontletmein, Password:
User: dontletmein, Password:
I
User: dontletmein, Password:
User: dontletmein, Password:
IN
User: dontletmein, Password:
User: student, Password: hello2
User: student, Password:
User: student, Password: mememe2
User: student, Password:
User: student, Password: 081481ns
User: student, Password:
┌──(kali㉿kali)-[~/Downloads]
└─$

Also notice that there is only 1 line in the crontab, preventing too many things in there being piled up.



Intended Use:

After obtaining access to one of the pivots that attach to RG Linux FTP, I will use the backdoor to log in to the machine and/or view the http server to see any credentials that have been logged.

I also have this exploit on a github page, so if I get access to any other linux machines during the KOTH, I can quickly download the files and activate the setup relatively quickly to enable a pretty persistent backdoor.