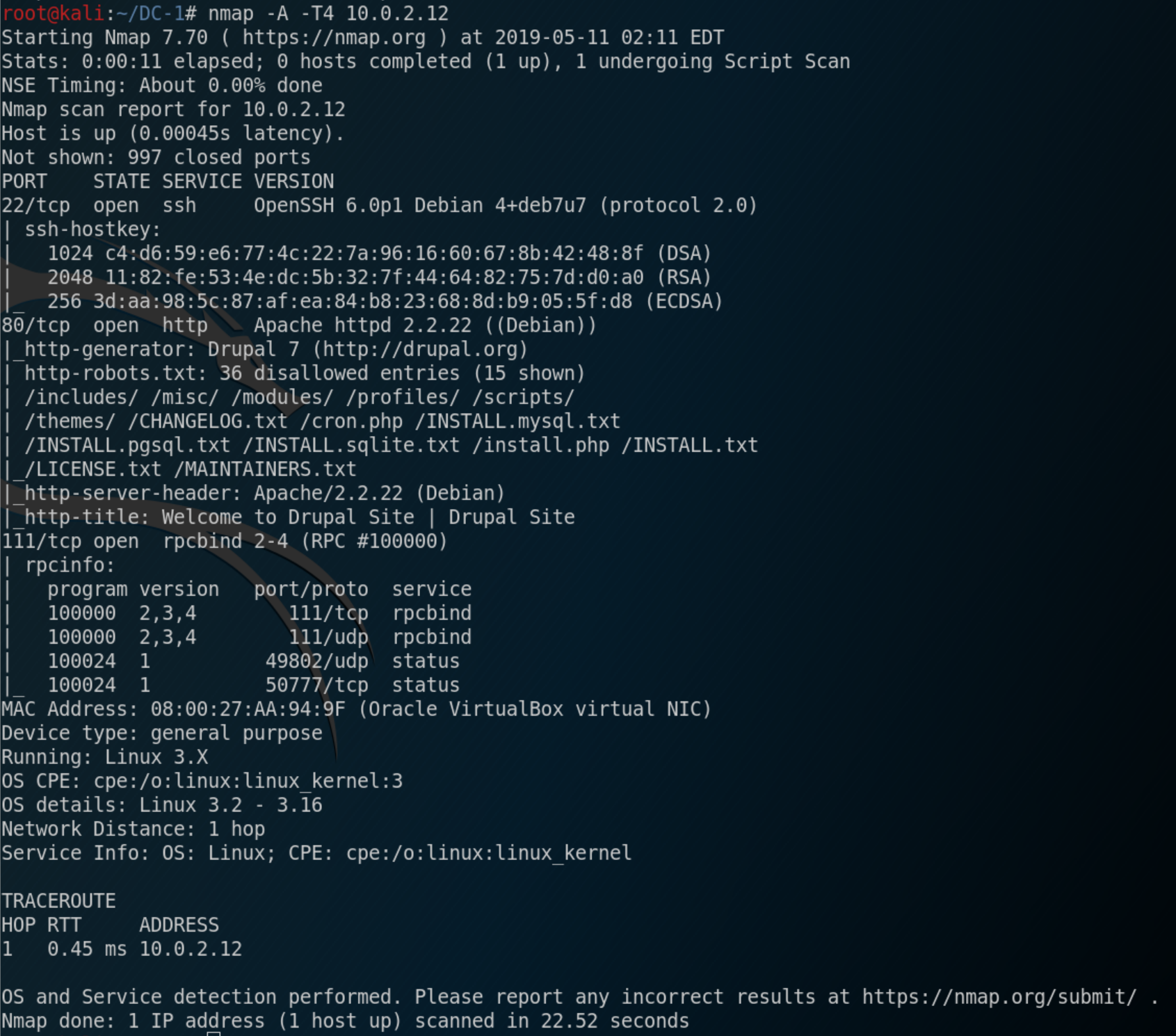
DC-1

This is a quick walkthrough of the DC-1 machine from vulnhub. The first step taken to overcome DC-1 is to perform a NMAP scan to check what ports and services are running on the machine. The command used to accomplish this was NMAP -A -T4 <ip address of target>. The output of the NMAP command is shown below.



The target machine appears to have 3 services running: OpenSSH version 6.0, an Apache server, and rpcinfo. The next step is to check if any of the services enumerated by NMAP have any known exploits. A quick Google search reveals that SSH version 6 does not have any. The Apache service is running Drupal 7, which I have not encountered before. However, the web server does have a robots.txt file, which may prove to be useful. The contents of robots.txt is shown below.

﻿

#

# robots.txt

#

# This file is to prevent the crawling and indexing of certain parts

# of your site by web crawlers and spiders run by sites like Yahoo!

# and Google. By telling these "robots" where not to go on your site,

# you save bandwidth and server resources.

#

# This file will be ignored unless it is at the root of your host:

# Used: http://example.com/robots.txt

# Ignored: http://example.com/site/robots.txt

#

# For more information about the robots.txt standard, see:

# http://www.robotstxt.org/wc/robots.html

#

# For syntax checking, see:

# http://www.sxw.org.uk/computing/robots/check.html

User-agent: \*

Crawl-delay: 10

# Directories

Disallow: /includes/

Disallow: /misc/

Disallow: /modules/

Disallow: /profiles/

Disallow: /scripts/

Disallow: /themes/

# Files

Disallow: /CHANGELOG.txt

Disallow: /cron.php

Disallow: /INSTALL.mysql.txt

Disallow: /INSTALL.pgsql.txt

Disallow: /INSTALL.sqlite.txt

Disallow: /install.php

Disallow: /INSTALL.txt

Disallow: /LICENSE.txt

Disallow: /MAINTAINERS.txt

Disallow: /update.php

Disallow: /UPGRADE.txt

Disallow: /xmlrpc.php

# Paths (clean URLs)

Disallow: /admin/

Disallow: /comment/reply/

Disallow: /filter/tips/

Disallow: /node/add/

Disallow: /search/

Disallow: /user/register/

Disallow: /user/password/

Disallow: /user/login/

Disallow: /user/logout/

# Paths (no clean URLs)

Disallow: /?q=admin/

Disallow: /?q=comment/reply/

Disallow: /?q=filter/tips/

Disallow: /?q=node/add/

Disallow: /?q=search/

Disallow: /?q=user/password/

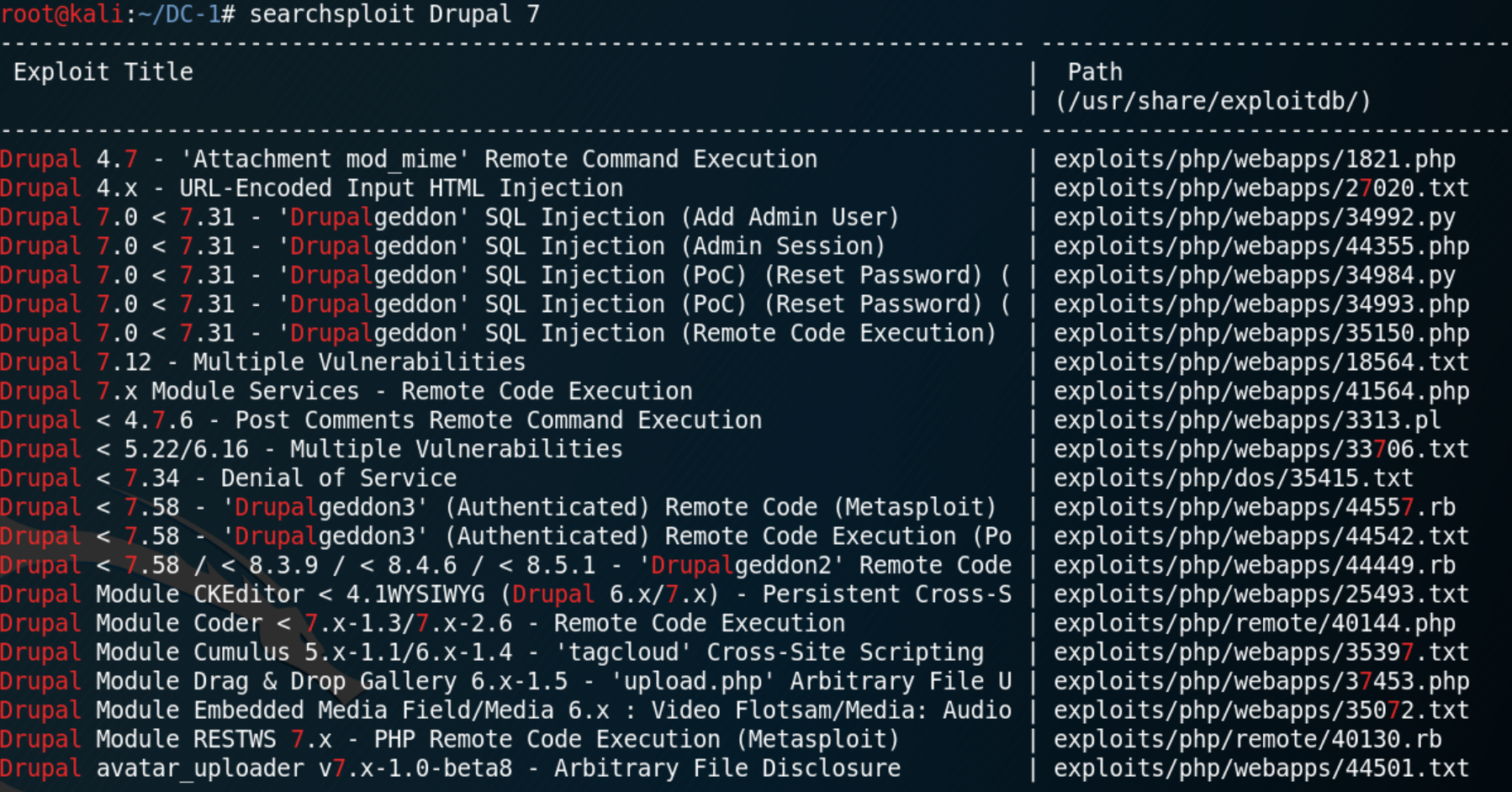
Disallow: /?q=user/register/

Disallow: /?q=user/login/

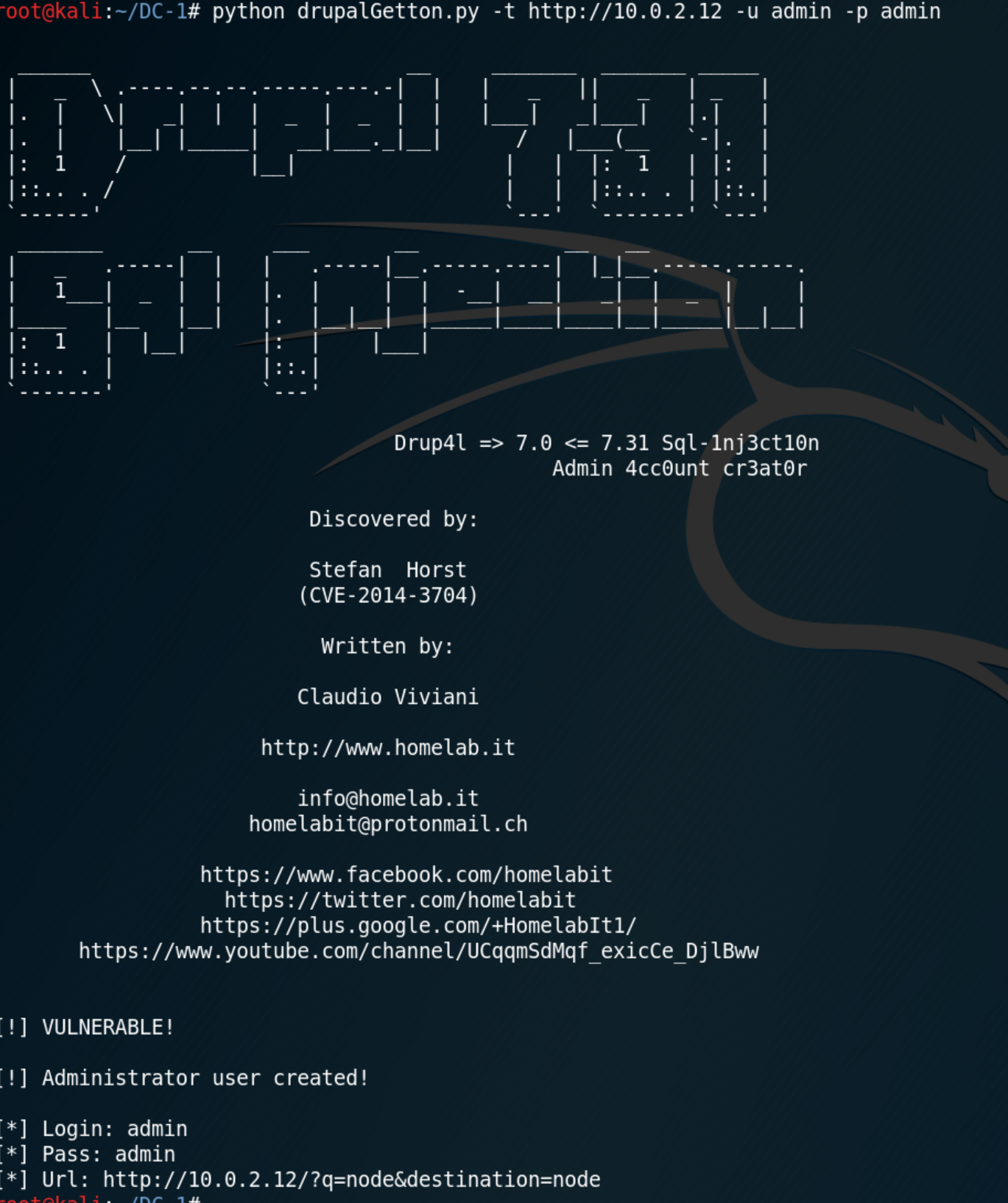
Disallow: /?q=user/logout/

The robots.txt file appeared to be a dead end because nothing of interest was uncovered.

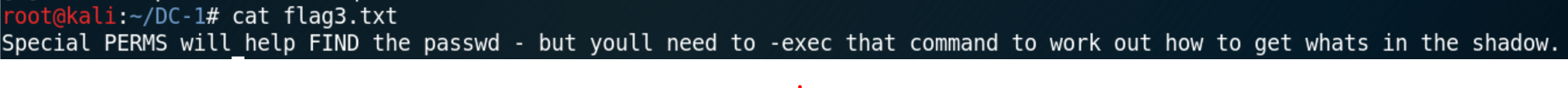
Searchsploit was used to search for a vulnerability corresponding to Drupal version 7. It turns out that Drupal version seven is vulnerable to SQL injection via the exploit drupalgetton.



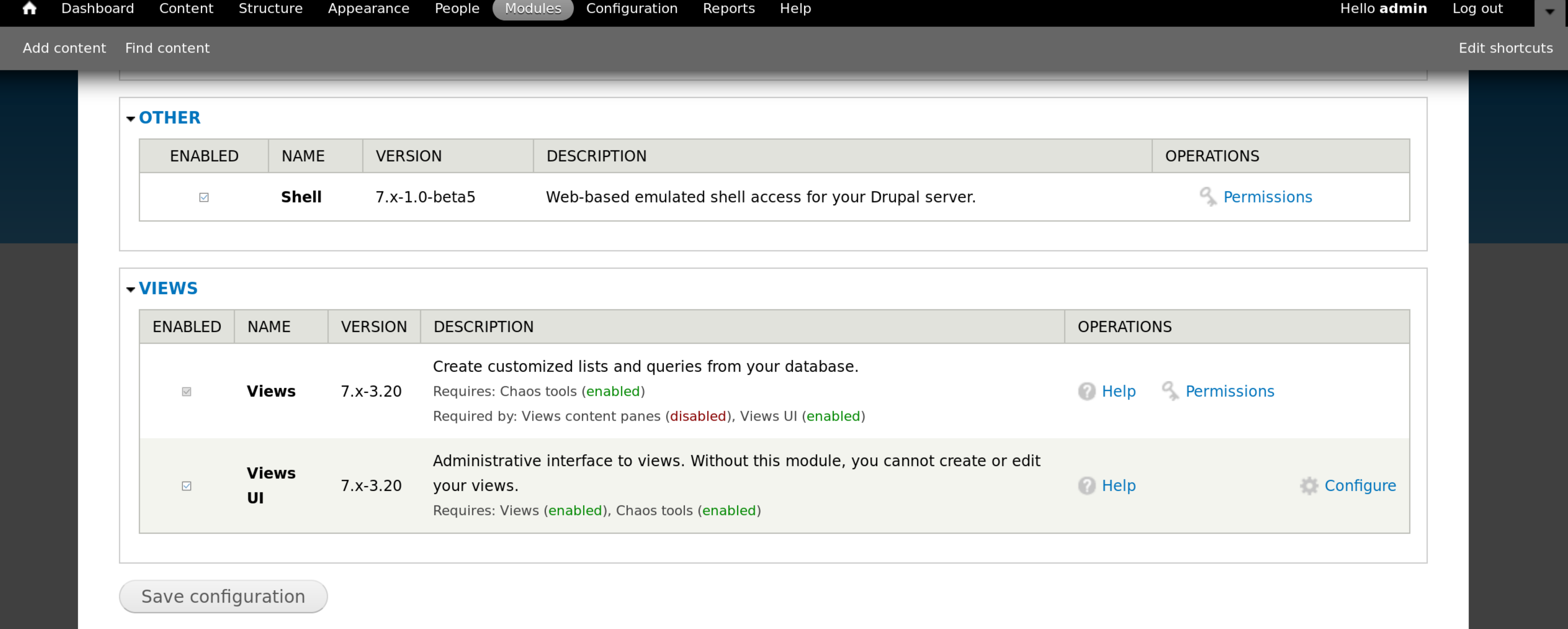
Running the exploit results in a new admin user being added to the system.



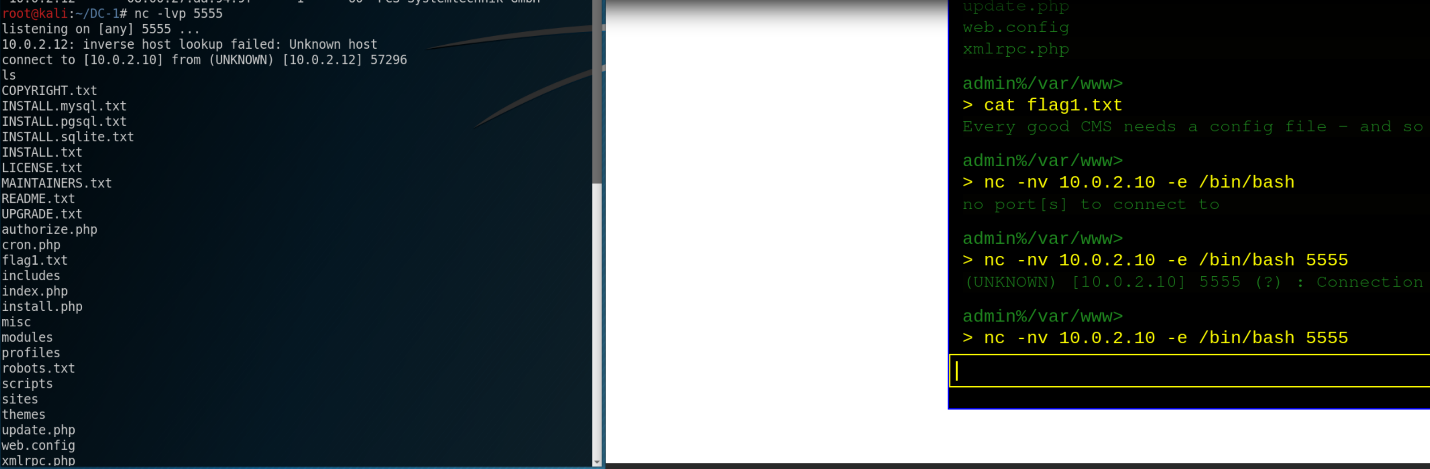
Flag3 is sitting in the admin console (there must be more flags we will keep an eye out for them). Flag three is displayed below



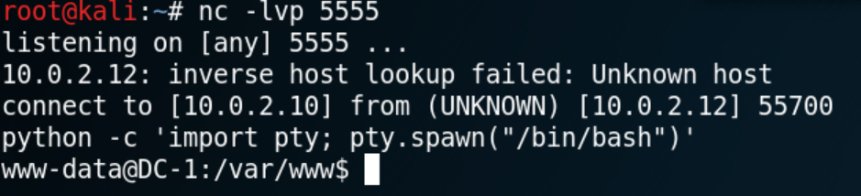
Being admin of the website is great, but it would be nice to have a shell to work with. Luckily, Drupal has created a shell for admins who prefer to manage the website using the command line! The shell can be obtained here <https://www.drupal.org/project/shell>. Using the admin account the aforementioned shell is installed and enabled on the target machine.



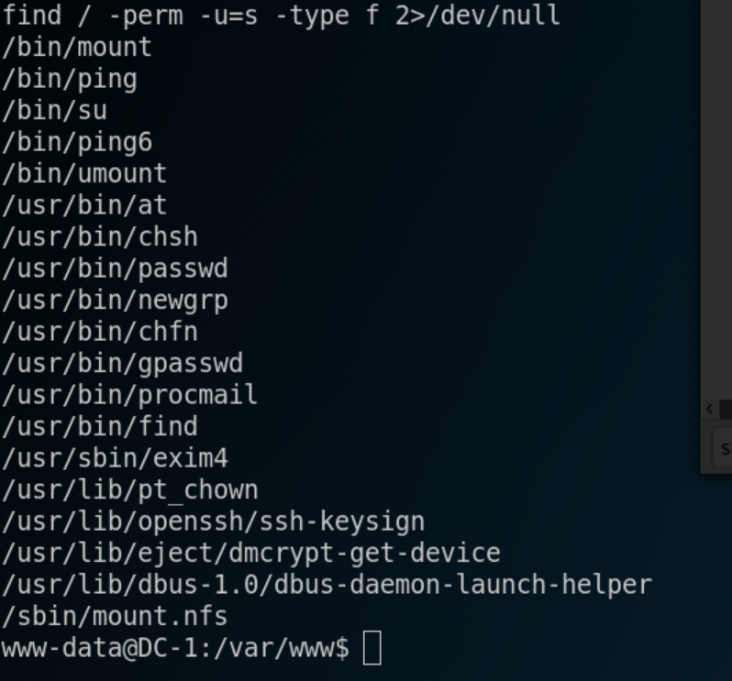
After the shell is launched it can be used to create a Netcat connection to the attacking machine.



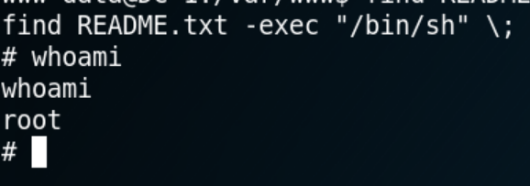
The shell is functional as is, but it is a bit inconvenient (it doesn’t show the output of certain commands). Luckily, the target has python installed, which can be used to spawn a Bash shell.



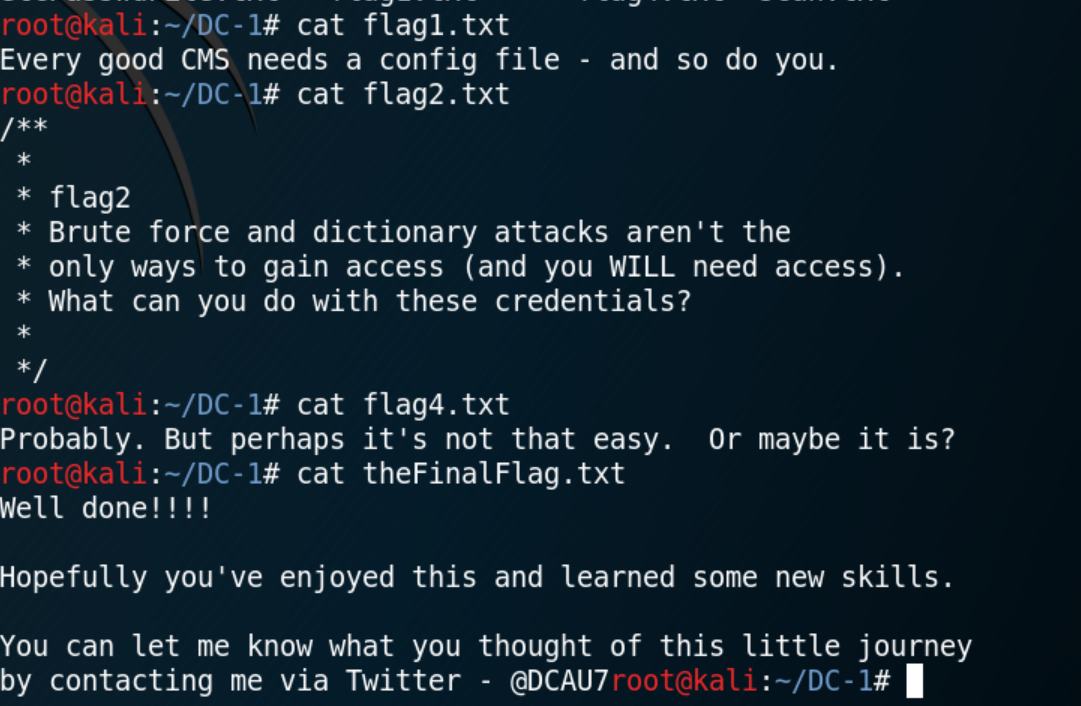
The obtained shell does not have root privileges, but using flag3 (the flag found earlier) as a hint we can guess that the target machine contains a binary that has the special permission bit set when it shouldn’t. The find command can be used to check what files on the system have root privileges at run time.



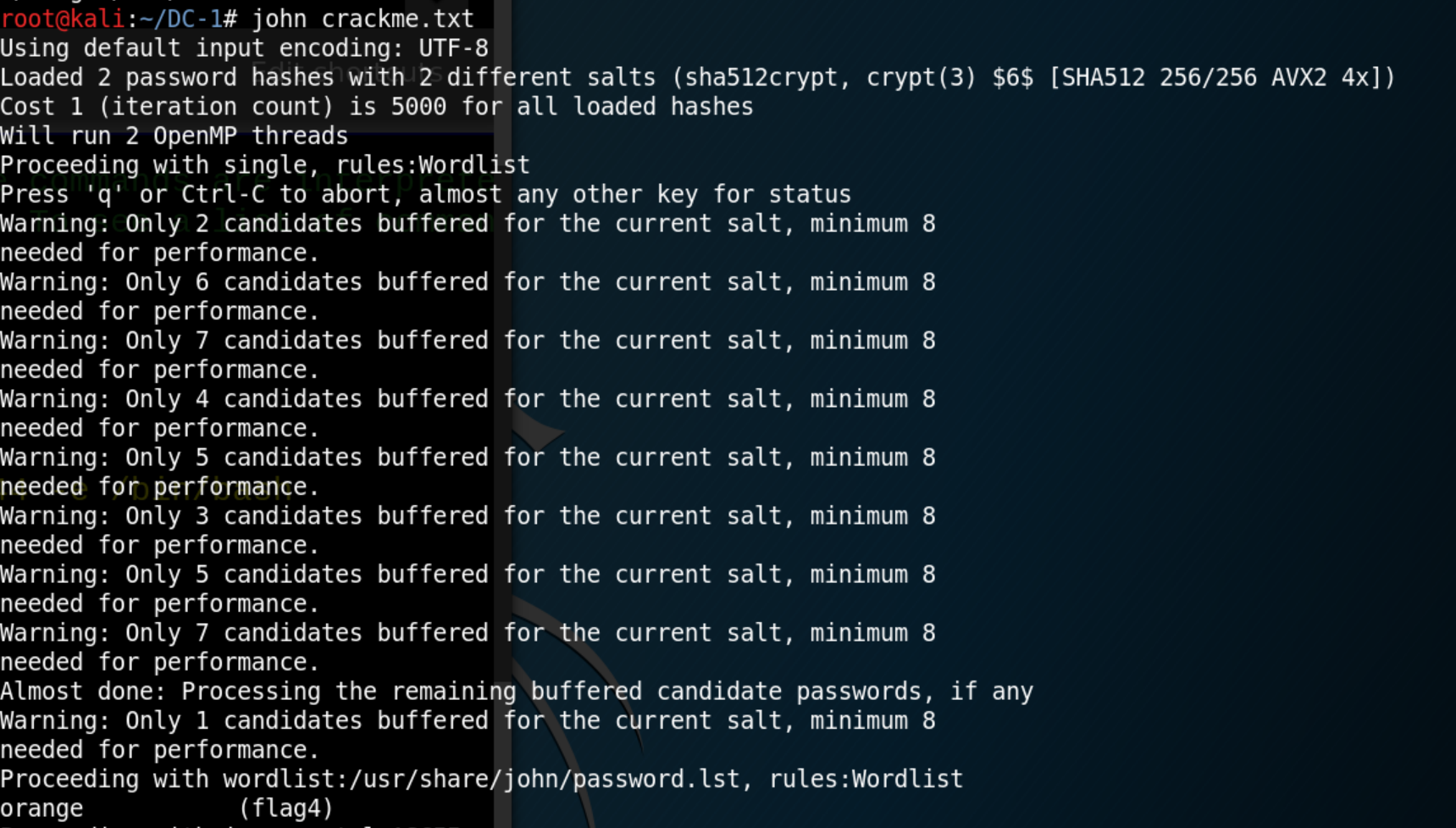
Excellent! The find command is running with root privileges. Reading the man pages for the find command reveals that the find command can be used to execute other commands when the -exec option is used. Since the binary is running as root we can spawn a root shell using this trick (The first argument to the find command could be any file on the system a . to specify the present working directory would also work). For example, find . -exec “/bin/sh ‘;’



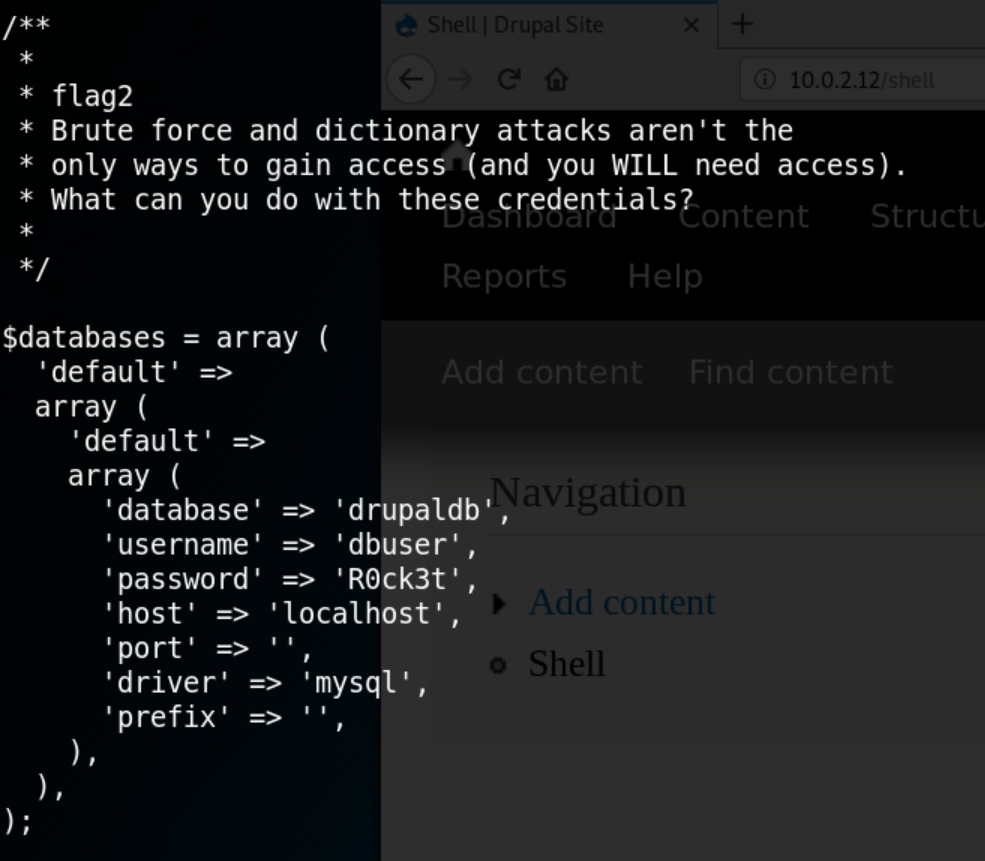
There are still flags to be found. Using the find command divulges the location of flag4 and flag1: find / -name flag\*.txt. There is also a flag in the root directory called thefinalflag.txt. The second flag is in the settings.php file (this is made evident by the hint in flag1). The contents of the flags are shown below:



We can take this exploitation further by attempting to crack the hashes stored in the /etc/shadow file and attempting to gain access to the mysql server. Cracking the hashes requires moving the contents of /etc/shadow and /etc/passwd to the attacking machine. Once the hashes and the passwd file are on the attacking machine the unshadow command can be used to create a file that John can crack.



The password for the user flag4, which is orange, is found immediately. It will take some time for John to guess the root password, so in the meantime, lets try to break into the sql server. The hint contained in flag1, “every good CMS needs a config file and so do you” leads us to the settings.php file located in /var/www/sites/default/settings.php. The settings.php file contains all the information that we need to access the database as root.



Logging into the database allows us to view a number of tables

