

Número i nom del curs	[IFCT0109_CEN] Seguretat dels sistemes d'informació
Mòdul Formatiu que s'avalua	MF0487_3 Auditoria de seguretat informàtica

Nom i Cognoms de l'alumne/a	
NIF de l'alumne/a	X1766856L
Data de la prova	11/05/2024
Signatura	Diego Fernando Mucci

INSTRUCCIONS DE LA PROVA

- En aquest full trobarà la informació necessària per a la realització de la prova.
- Abans de resoldre la pràctica llegeixi amb atenció l'enunciat de l'exercici i comprovi que té tots els materials i equipament necessari.
- Per qualsevol aclariment consulti a l'avaluador/ i/o formador/a.
- Cas de ser necessari, utilitzi els equips de protecció individual necessaris per la realització de la pràctica tenint en compte les normes de seguretat i higiene.

DESCRIPCIÓ DE LA PRÀCTICA**DENOMINACIÓ**

Comprendre i aplicar conceptes de seguretat en aplicacions web per identificar i mitigar vulnerabilitats.

**ESPECIFICACIONS
TÈCNIQUES**

Llegeix l'enunciat de l'exercici i implementa allò que se t'indica.

Pots consultar qualsevol font per a la seva realització.

Disposes de 4 hores per presentar la prova pràctica, que lliuraràs en un document amb les teves solucions justificades i documentades.

MATERIAL

El teu propi ordinador

TEMPS

Quatre hores

SUPUESTO

La empresa IRON S.L. ha desarrollado una nueva aplicación web que utiliza un servidor Apache y una base de datos MariaDB en un entorno Linux.

La aplicación cuenta con un formulario de login que es vulnerable a ataques de inyección SQL. Se facilita el fichero php con el código del formulario (loginvuln.php)

Además, la empresa quiere asegurarse de que su aplicación web esté protegida contra ataques comunes, por lo que ha decidido implementar ModSecurity con el conjunto de reglas de OWASP.

Activitat 1 (2 punts)	<p>Instale, inicie y compruebe en un entorno local los servicios Apache y MariaDB.</p> <p>Inicie el servidor MariaDB y cree una base de datos llamada dfir.</p> <p>Cree una tabla usuarios con las columnas userid y password.</p> <p>Inserte al menos tres registros en la tabla usuarios con diferentes combinaciones de usuario y contraseña.</p> <p>Verifique que puede consultar los datos insertados correctamente.</p>
Activitat 2 (2 punts)	<p>Suba el archivo `LoginVuln.php` a la carpeta indicada del sistema para que sea accesible desde un navegador.</p> <p>Acceda a la página de login vulnerable a través de un navegador.</p> <p>Realice un ataque de inyección SQL para lograr un login exitoso sin conocer las contraseñas.</p>
Activitat 3 (2 punts)	<p>Instale ModSecurity en el servidor Apache.</p> <p>Configure el archivo modsecurity.conf para activar la prevención de ataques y reinicie el servidor Apache y asegúrese de que ModSecurity esté funcionando correctamente.</p>
Activitat 4 (2 punts)	<p>Descargue e instale el conjunto de reglas de OWASP para ModSecurity.</p> <p>Configure Apache para incluir las reglas descargadas y asegúrese de que estén activas revisando los logs de Apache después de intentar un ataque bloqueado.</p> <p>Realice pruebas para verificar que las nuevas reglas bloqueen un intento de inyección SQL similar al de la Actividad 2.</p>

Activitat 5
(2 punts)

Cambia el banner de identificación del servidor Apache para eliminar cualquier tipo de vulnerabilidad.

Bloquea una página http y genera una regla que pueda visualizarse en los logs de Apache. Muestre el bloqueo y los resultados.

PUNTUACIÓ FINAL PROVA PRÀCTICA

ACT 1	ACT 2	ACT 3	ACT 4	ACT 5	PUNTUACIÓ FINAL

OBSERVACIONS

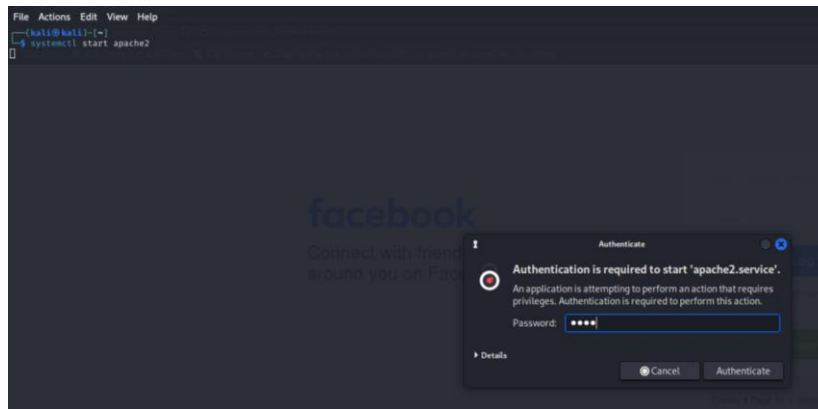
--

Signatura formador/a**Signatura responsable acció formativa****Generalitat
de Catalunya**

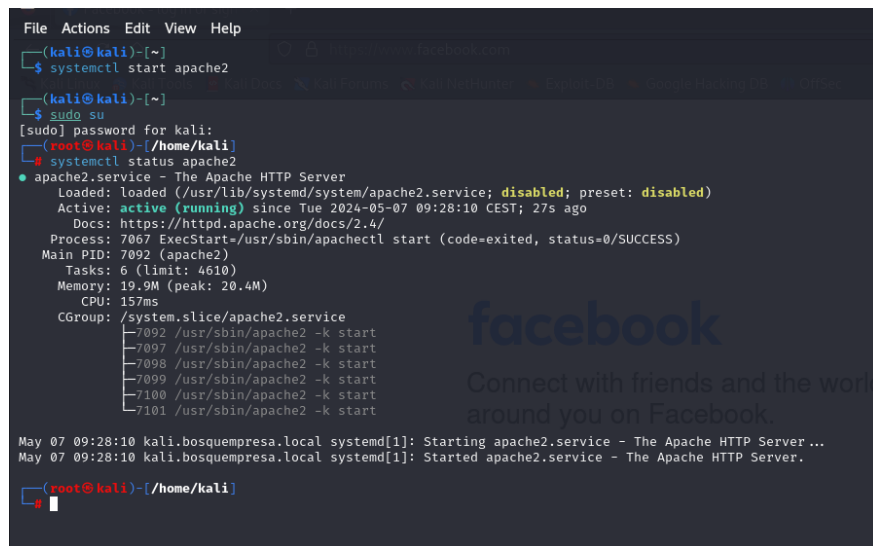
Activitat 1

- Instale, inicie y compruebe en un entorno local los servicios Apache y MariaDB.

Escribimos el comando “systemctl start apache2” para inicializar apache2. Nos pedirá la contraseña de nuestro Kali Linux:



Escribimos el comando “systemctl status apache2” para ver si el servicio está iniciado. También habilitaremos el usuario *root* para elevar privilegios. Vemos que apache2 figura correctamente como activo:



Realizamos lo mismo con MariaDB. Comprobamos el estado de MariaDB antes de hacer *start* y vemos que figura como inactivo. Por lo tanto, aplicamos los mismos comandos que antes:

```
(root@kali): /home/kali
# systemctl status mariadb
● mariadb.service - MariaDB 10.11.6 database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; disabled; preset: disabled)
   Active: inactive (dead)
     Docs: man:mariadb(8)
           https://mariadb.com/kb/en/library/systemd/

(root@kali): /home/kali
# systemctl start mariadb

(root@kali): /home/kali
# systemctl status mariadb
● mariadb.service - MariaDB 10.11.6 database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; disabled; preset: disabled)
   Active: active (running) since Tue 2024-05-07 09:33:51 CEST; 18s ago
     Docs: man:mariadb(8)
           https://mariadb.com/kb/en/library/systemd/
   Process: 18015 ExecStartPre=/usr/bin/install -m 755 -o mysql -g root -d /var/run/mysql (code=exited, status=0/SUCCESS)
   Process: 18017 ExecStartPre=/bin/sh -c 'systemctl unset-environment _WSREP_START_POSITION (code=exited, status=0/SUCCESS)
   Process: 18019 ExecStartPre=/bin/sh -c '[ ! -e /usr/bin/galera_recovery ] && VAR= $(VARDIR=$(cd /usr/bin/..; /usr/bin/galera_recovery) ; [ $? -eq 0 ] && systemctl set-environment _WSREP_START_POSITION=$VAR || exit 1 (code=exited,
   Process: 18099 ExecStartPost=/bin/sh -c 'systemctl unset-environment _WSREP_START_POSITION (code=exited, status=0/SUCCESS)
   Process: 18209 ExecStartPost=/etc/mysql/debian-start (code=exited, status=0/SUCCESS)
   Main PID: 18087 (mariadbd)
   Status: "Waiting your SQL requests now..."
     Tasks: 12 (limit: 4610)
    Memory: 217.9M (peak: 221.0M)
       CPU: 1.185s
    CGroup: /system.slice/mariadb.service
            └─18087 /usr/sbin/mariadbd

May 07 09:33:51 kali.bosquempresalocal mariadbd[18087]: 2024-05-07 9:33:51.0 [Note] Plugin 'FEEDBACK' is disabled.
May 07 09:33:51 kali.bosquempresalocal mariadbd[18087]: 2024-05-07 9:33:51.0 [Note] InnoDB: Loading buffer pool(s) from /var/lib/mysql/ib_buffer_pool
May 07 09:33:51 kali.bosquempresalocal mariadbd[18087]: 2024-05-07 9:33:51.0 [Warning] You need to use --log-bin to make --expire-logs-days or --binlog-expire-logs-seconds work.
May 07 09:33:51 kali.bosquempresalocal mariadbd[18087]: 2024-05-07 9:33:51.0 [Note] Server socket created on IP: '127.0.0.1'.
May 07 09:33:51 kali.bosquempresalocal mariadbd[18087]: 2024-05-07 9:33:51.0 [Note] InnoDB: Buffer pool(s) load completed at 240507 9:33:51
May 07 09:33:51 kali.bosquempresalocal mariadbd[18087]: 2024-05-07 9:33:51.0 [Note] /usr/sbin/mariadbd: ready for connections.
May 07 09:33:51 kali.bosquempresalocal mariadbd[18087]: Version: '10.11.6-MariaDB-2' socket: '/var/run/mysql/mysql.sock' port: 3306 Debian n/a
May 07 09:33:51 kali.bosquempresalocal systemd[1]: Started mariadb.service - MariaDB 10.11.6 database server.
May 07 09:33:51 kali.bosquempresalocal /etc/mysql/debian-start[18124]: Checking for insecure root accounts.
May 07 09:33:51 kali.bosquempresalocal /etc/mysql/debian-start[18128]: Triggering myisam-recover for all MyISAM tables and aria-recover for all Aria tables
lines 3-28/28 (0%)
```

- Inicie el servidor MariaDB y cree una base de datos llamada dfir.

Después de iniciar el servidor aplicamos el comando “*mariadb -u root -p*”, nos pedirá la contraseña, la cual es 123456 por que es la que figura en el archivo php.

```
File Actions Edit View Help
# systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled; preset: disabled)
   Active: active (running) since Tue 2024-05-07 09:28:10 CEST; 27min ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 7067 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 7092 (apache2)
     Tasks: 6 (limit: 4610)
    Memory: 19.9M (peak: 20.4M)
       CPU: 400ms
    CGroup: /system.slice/apache2.service
            └─7092 /usr/sbin/apache2 -k start
              7097 /usr/sbin/apache2 -k start
              7098 /usr/sbin/apache2 -k start
              7099 /usr/sbin/apache2 -k start
              7100 /usr/sbin/apache2 -k start
              7101 /usr/sbin/apache2 -k start

May 07 09:28:10 kali.bosquempresalocal systemd[1]: Starting apache2.service - The Apache HTTP Server...
May 07 09:28:10 kali.bosquempresalocal systemd[1]: Started apache2.service - The Apache HTTP Server.

(root@kali)-[/var/www/html]
# mariadb -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.11.6-MariaDB-2 Debian n/a

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Una vez dentro, escribimos “ALTER USER 'root'@'localhost' IDENTIFIED BY '123456'”. Esto me genera como usuario dentro del sistema MariaDB. Hemos creado un usuario que se llama *root*, cuya contraseña es 123456.

Al final de cada comando escribiremos “ ; ” así permanecemos siempre dentro de MariaDB y no es necesario volver a escribirlo.

```
File Actions Edit View Help
systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled; preset: disabled)
   Active: active (running) since Tue 2024-05-07 09:28:10 CEST; 27min ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 7067 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
  Main PID: 7092 (apache2)
    Tasks: 6 (limit: 4610)
  Memory: 19.9M (peak: 20.4M)
     CPU: 400ms
    CGroup: /system.slice/apache2.service
            └─7092 /usr/sbin/apache2 -k start
              7097 /usr/sbin/apache2 -k start
              7098 /usr/sbin/apache2 -k start
              7099 /usr/sbin/apache2 -k start
              7100 /usr/sbin/apache2 -k start
              7101 /usr/sbin/apache2 -k start

May 07 09:28:10 kali.bosquempresa.local systemd[1]: Starting apache2.service - The Apache HTTP Server ...
May 07 09:28:10 kali.bosquempresa.local systemd[1]: Started apache2.service - The Apache HTTP Server.

(root@kali)-[/var/www/html]
# mariadb -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.11.6-MariaDB-2 Debian n/a

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> ALTER USER 'root'@'localhost' IDENTIFIED BY '123456';
Query OK, 0 rows affected (0.012 sec)

MariaDB [(none)]> █
```

Escribimos “show databases” para ver las bases de datos dentro de MariaDB:

```
(root@kali)-[/var/www/html]
# mariadb -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.11.6-MariaDB-2 Debian n/a

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> ALTER USER 'root'@'localhost' IDENTIFIED BY '123456';
Query OK, 0 rows affected (0.012 sec)

MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.012 sec)

MariaDB [(none)]> █
```

Ahora crearemos la base de datos llamada "dfir" mediante el comando "create database dfir" y después volvemos a aplicar el comando "show databases" para ver que se ha creado correctamente:

```
5 rows in set (0.012 sec)

MariaDB [(none)]> create database dfir;
Query OK, 1 row affected (0.003 sec)

MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| dfir      |
| information_schema |
| mysql     |
| performance_schema |
| sys       |
+-----+
5 rows in set (0.002 sec)

MariaDB [(none)]>
```

- Cree una tabla usuarios con las columnas userid y password.

campos y no columnas

Escribimos "connect dfir" para entrar dentro de la base de datos. También podríamos entrar mediante el comando "use dfir".

```
MariaDB [(none)]> connect dfir;
Connection id: 32
Current database: dfir
```

Me dice que estoy conectado con un id a esta base de datos.

Escribimos "show tables" para ver si existe alguna tabla, pero vemos que de momento no hay ninguna creada.

```
MariaDB [dfir]> show tables;
Empty set (0.003 sec)
```

Vamos a crear una de la siguiente manera. Escribimos "CREATE TABLE usuarios (userid VARCHAR (100), password VARCHAR (100));"

Con este comando estamos creando una tabla ~~de una fila~~ con un usuario de máximo 100 caracteres y una contraseña de máximo 100 caracteres.

Volvemos a escribir “show tables” y vemos como se ha creado correctamente.

Después, escribimos “insert into usuarios values ('dfir1', '123456');”. Con esto estamos estableciendo “dfir” como nombre de usuario y “123456” como contraseña.

```
MariaDB [dfir]> show tables;
Empty set (0.003 sec)

MariaDB [dfir]> CREATE TABLE usuarios (userid VARCHAR(100), password VARCHAR(100));
Query OK, 0 rows affected (0.053 sec)

MariaDB [dfir]> show tables;
+-----+
| Tables_in_dfir |
+-----+
| usuarios       |
+-----+
1 row in set (0.002 sec)

MariaDB [dfir]> insert into usuarios values('dfir1','123456');
Query OK, 1 row affected (0.027 sec)
```

Tabla que contendrá
registros con campos

Si quisiéramos crear más usuarios escribiríamos nuevamente el comando de arriba con otros valores diferentes entre los paréntesis. Durante la ejecución de esta práctica solo se creó uno.

- Verifique que puede consultar los datos insertados correctamente.

Para ello escribiremos el comando “select * from usuarios;”:

```
MariaDB [dfir]> insert into usuarios values('dfir1','123456');
Query OK, 1 row affected (0.027 sec)

MariaDB [dfir]> select * from usuarios;
+-----+-----+
| userid | password |
+-----+-----+
| dfir1  | 123456   |
+-----+-----+
1 row in set (0.001 sec)

MariaDB [dfir]> █
```

Actividad 2:

- Suba el archivo `Vuln.php` a la carpeta indicada del sistema para que sea accesible desde un navegador.

Accedemos a la carpeta indicada escribiendo los comandos “cd /var/www” “cd html” y después para crearlo escribimos “nano vuln.php”.

```
(root@kali)-[/home]
# cd /var/www

(root@kali)-[/var/www]
# cd html

(root@kali)-[/var/www/html]
# ls
index.html  index.nginx-debian.html

(root@kali)-[/var/www/html]
# nano vuln.php

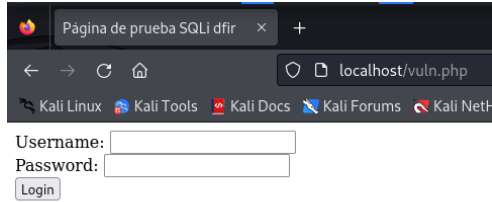
(root@kali)-[/var/www/html]
# nano vuln.php
```

Pegamos el código que contiene el fichero, guardamos y salimos.

```
GNU nano 7.2 vuln.php
<html>
<head>
<title>
Página de prueba SQLi dfir
</title>
</head>
<body>
<?php
if(isset($_POST['login']))
{
$username = $_POST['username'];
$password = $_POST['password'];
$con = mysqli_connect('localhost','root','123456','dfir');
$result = mysqli_query($con, "SELECT * FROM 'usuarios' WHERE userid='$username' AND password='$password'");
if(mysqli_num_rows($result) == 0)
echo 'Usuario o Password Incorrecto, prueba otra vez';
else
echo '<h1>Dentro!!!</h1><p>Este texto lo ven sólo aquellos que han hecho un login correcto.</p>';
}
else
{
}
?>
<form action="" method="post">
Username: <input type="text" name="username"/><br />
Password: <input type="password" name="password"/><br />
<input type="submit" name="login" value="Login" />
</form>
<?php
}
?>
</body>
</html>
```

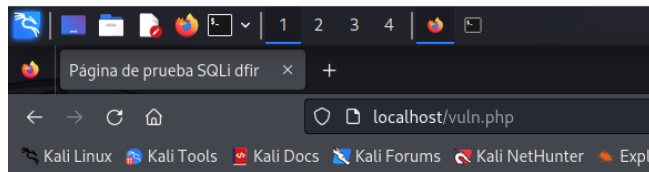
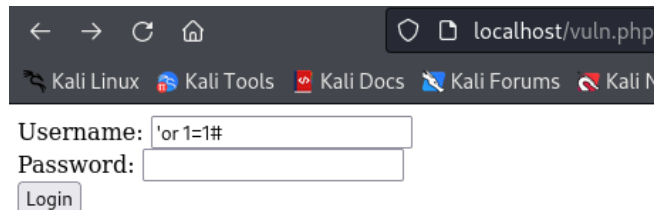
- Acceda a la página de login vulnerable a través de un navegador.

Para ello escribiremos en el navegador “localhost/nombredelarchivo”



- Realice un ataque de inyección SQL para lograr un login exitoso sin conocer las contraseñas.

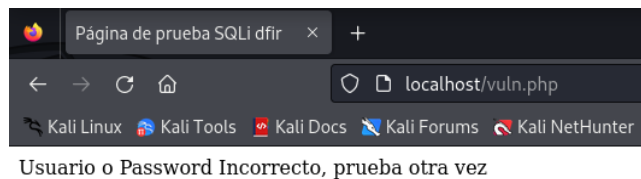
Para ello, le daremos el valor de `'or 1=1#` a *username*. En el campo de *password* no hará falta escribir nada y lograremos un *login* exitoso.



Dentro!!!

Este texto lo ven sólo aquellos que han hecho un login correcto.

En cambio, si ponemos incorrectamente el usuario o la contraseña nos saldrá el siguiente error:



Más detalle en las pruebas. Login correcto, incorrecto e inyección de código

Actividad 3

- Instale ModSecurity en el servidor Apache.

Para? por?

Para ello escribiremos el comando “apt install libapache2-mod-security2 -y”

```
(root@kali)-[/var/www/html]
└─$ apt install libapache2-mod-security2 -y
Reading package lists ... Done
Building dependency tree ... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libabsl20220623 libadwaita-1-0 libaio1 libatk-adaptor libboost-dev libboost1.83-dev libopenblas-dev libopenblas
  python3-beniget python3-gast python3-pyatspi python3-pypdf2 python3-pyppeteer python3-pyrsistent python3-pyth
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  liblua5.1-0 modsecurity-crs
Suggested packages:
  lua geoip-database-contrib python
The following NEW packages will be installed:
  libapache2-mod-security2 liblua5.1-0 modsecurity-crs
0 upgraded, 3 newly installed, 0 to remove and 4 not upgraded.
Need to get 531 kB of archives.
After this operation, 2458 kB of additional disk space will be used.
Get:1 http://http.kali.org/kali kali-rolling/main amd64 liblua5.1-0 amd64 5.1.5-9+b1 [108 kB]
Get:2 http://http.kali.org/kali kali-rolling/main amd64 libapache2-mod-security2 amd64 2.9.7-1+b1 [259 kB]
Get:3 http://kali.download/kali kali-rolling/main amd64 modsecurity-crs all 3.3.5-2 [163 kB]
Fetched 531 kB in 1s (939 kB/s)
Selecting previously unselected package liblua5.1-0:amd64.
(Reading database ... 414969 files and directories currently installed.)
Preparing to unpack .../liblua5.1-0_5.1.5-9+b1_amd64.deb ...
Unpacking liblua5.1-0:amd64 (5.1.5-9+b1) ...
Selecting previously unselected package libapache2-mod-security2.
Preparing to unpack .../libapache2-mod-security2_2.9.7-1+b1_amd64.deb ...
Unpacking libapache2-mod-security2 (2.9.7-1+b1) ...
Selecting previously unselected package modsecurity-crs.
Preparing to unpack .../modsecurity-crs_3.3.5-2_all.deb ...
Unpacking modsecurity-crs (3.3.5-2) ...
Setting up modsecurity-crs (3.3.5-2) ...
Setting up liblua5.1-0:amd64 (5.1.5-9+b1) ...
Setting up libapache2-mod-security2 (2.9.7-1+b1) ...
apache2_invoke: Enable module security2
Processing triggers for libc-bin (2.37-15) ...
Processing triggers for kali-menu (2023.4.7) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...
Service restarts being deferred:
systemctl restart NetworkManager.service
```

- Configure el archivo modsecurity.conf para activar la prevención de ataques y reinicie el servidor Apache y asegúrese de que ModSecurity esté funcionando correctamente.

Nos iremos a la carpeta correspondiente mediante el comando “cd /etc/modsecurity”.

Si después hacemos un “ls” vemos todos los archivos que hay dentro. Copiaremos el archivo que se llama “modsecurity.conf-recommended” en otro fichero nuevo al cual le daremos el nombre de “modsecurity.conf” mediante el comando “cp” de la siguiente manera:

```
(root@kali)-[/var/www/html]
# cd /etc/modsecurity

(root@kali)-[/etc/modsecurity]
# ls
crs  modsecurity.conf-recommended  unicode.mapping

(root@kali)-[/etc/modsecurity]
# cp modsecurity.conf-recommended modsecurity.conf

(root@kali)-[/etc/modsecurity]
# ls
crs  modsecurity.conf  modsecurity.conf-recommended  unicode.mapping
```

Verificamos mediante el comando “ls” nuevamente el contenido de la carpeta y vemos que se ha creado correctamente.

Ahora, activaremos la prevención de ataques modificando el archivo modsecurity.conf escribiendo el comando “nano modsecurity.conf” y una vez dentro escribimos “ON” para activar el servicio:

```
(root@kali)-[/etc/modsecurity]
# nano modsecurity.conf
```

```
GNU nano 7.2
# -- Rule engine initialization --
# Enable ModSecurity, attaching it to every transaction. Use detection
# only to start with, because that minimises the chances of post-installation
# disruption.
#
SecRuleEngine On

# -- Request body handling --
# Allow ModSecurity to access request bodies. If you don't, ModSecurity
# won't be able to see any POST parameters, which opens a large security
# hole for attackers to exploit.
#
SecRequestBodyAccess On

# Enable XML request body parser.
# Initiate XML Processor in case of xml content-type
#
SecRule REQUEST_HEADERS:Content-Type "(?:application(?:/soap+|/))|text/xml" \
    "id:'200000',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=XML"

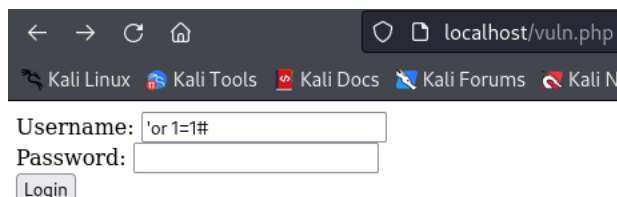
# Enable JSON request body parser.
# Initiate JSON Processor in case of .JSON content-type; change accordingly
# if your application does not use 'application/json'
#
SecRule REQUEST_HEADERS:Content-Type "^application/json" \
    "id:'200001',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=JSON"

# Sample rule to enable JSON request body parser for more subtypes.
# Uncomment or adapt this rule if you want to engage the JSON
# Processor for "+json" subtypes
#
SecRule REQUEST_HEADERS:Content-Type "^application/[a-z0-9.-]+[+json]" \
    "id:'200006',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=JSON"

# Maximum request body size we will accept for buffering. If you support
# file uploads then the value given on the first line has to be as large
# as the largest file you are willing to accept. The second value refers
# to the size of data, with files excluded. You want to keep that value as
# low as practical.
```

Reiniciamos el servidor apache y realizamos nuevamente la inyección de código:

```
(root@kali)-[/etc/modsecurity]  
# systemctl restart apache2
```



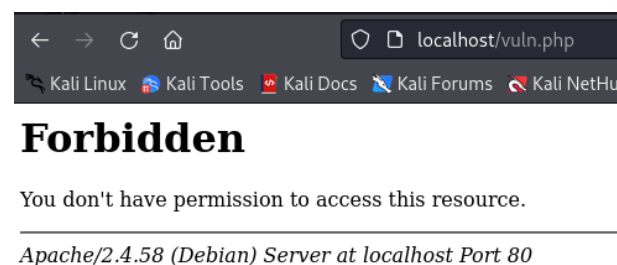
localhost/vuln.php

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHu

Username: 'or 1=1#

Password:

Login



localhost/vuln.php

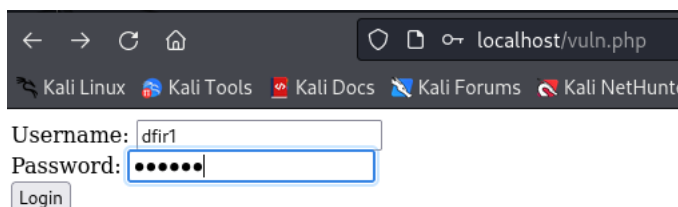
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHu

Forbidden

You don't have permission to access this resource.

Apache/2.4.58 (Debian) Server at localhost Port 80

Ahora ya no será posible acceder mediante inyección de código. Solamente podremos entrar escribiendo el usuario y la contraseña que creamos anteriormente:



localhost/vuln.php

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHu

Username: dfir1

Password: •••••

Login

Dentro!!!

Este texto lo ven sólo aquellos que han hecho un login correcto.

Actividad 4

- Descargue e instale el conjunto de reglas de OWASP para ModSecurity.

Primero vamos a borrar las reglas que vienen por defecto en ModSecurity mediante el comando "rm -rf". Accedemos a la carpeta y borraremos de la siguiente manera:

```
(root@kali)-[/etc/modsecurity]
# cd /usr/share

(root@kali)-[/usr/share]
# cd modsecurity-crs

(root@kali)-[/usr/share/modsecurity-crs]
# ls
owasp-crs.load  rules  util

(root@kali)-[/usr/share/modsecurity-crs]
#

(root@kali)-[/usr/share]
# rm -rf modsecurity-crs
```

Ahora nos traeremos todas las reglas OWASP que hay en este directorio de github:

```
(root@kali)-[/usr/share]
# git clone https://github.com/coreruleset/coreruleset /usr/share/modsecurity-crs

Cloning into '/usr/share/modsecurity-crs' ...
remote: Enumerating objects: 30393, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 30393 (delta 0), reused 1 (delta 0), pack-reused 30392
Receiving objects: 100% (30393/30393), 8.03 MiB | 7.34 MiB/s, done.
Resolving deltas: 100% (23755/23755), done.
```

Vemos todo el contenido que se ha clonado en la carpeta modsecurity-crs mediante el comando "ls".

Copiamos el fichero "crs-setup.conf.example" y creamos otro fichero igual sin la extensión ".example":

```
(root@kali)-[/usr/share/modsecurity-crs]
# ls
CHANGES.md CONTRIBUTING.md CONTRIBUTORS.md INSTALL.md KNOWN_BUGS.md LICENSE README.md SECURITY.md SPONSORS.md crs-setup.conf.example docs plugins regex-assembly rules tests util

(root@kali)-[/usr/share/modsecurity-crs]
# cp crs-setup.conf.example crs-setup.conf

(root@kali)-[/usr/share/modsecurity-crs]
# ls
CHANGES.md CONTRIBUTING.md CONTRIBUTORS.md INSTALL.md KNOWN_BUGS.md LICENSE README.md SECURITY.md SPONSORS.md crs-setup.conf crs-setup.conf.example docs plugins regex-assembly rules tests util
```


Ahora vamos a la carpeta de “rules” y hacemos exactamente lo mismo con el primer fichero que aparece:

```
[root@kali]~# cd /usr/share/modsecurity-crs
```

```
# cd rules
```

```
[root@kali]~/usr/share/modsecurity-crs/rules
```

```
# ls
```

```
REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf.example REQUEST-932-APPLICATION-ATTACK-RCE.conf          RESPONSE-952-DATA-LEAKAGES-JAVA.conf             java-errors.data
```

```
REQUEST-901-INITIALIZATION.conf                     REQUEST-933-APPLICATION-ATTACK-PHP.conf          RESPONSE-953-DATA-LEAKAGES-PHP.conf              lfi-or-files.data
```

```
REQUEST-905-COMMON-EXCEPTIONS.conf                   REQUEST-934-APPLICATION-ATTACK-GENERIC.conf      RESPONSE-954-DATA-LEAKAGES-IIS.conf              php-config-directives.data
```

```
REQUEST-911-METHOD-ENFORCEMENT.conf                 REQUEST-941-APPLICATION-ATTACK-XSS.conf          RESPONSE-955-WEB-SHELLS.conf                    php-errors-p12.data
```

```
REQUEST-913-SCANNER-DETECTION.conf                   REQUEST-942-APPLICATION-ATTACK-SQLI.conf         RESPONSE-959-BLOCKING-EVALUATION.conf            php-errors.data
```

```
REQUEST-920-PROTOCOL-ENFORCEMENT.conf                 REQUEST-943-APPLICATION-ATTACK-SESSION-FIXATION.conf  RESPONSE-980-CORRELATION.conf                  php-function-names-933150.data
```

```
REQUEST-921-PROTOCOL-ATTACK.conf                     REQUEST-944-APPLICATION-ATTACK-JAVA.conf        RESPONSE-990-EXCLUSION-RULES-AFTER-CRS.conf.example  php-function-names-933151.data
```

```
REQUEST-927-HTTP-ATTACK.conf                         REQUEST-946-BLOCKING-EVALUATION.conf           REQUEST-947-REQUEST-TOO-LARGE.conf               php-variables
```

```
REQUEST-930-APPLICATION-ATTACK-LFI.conf                REQUEST-950-DATA-LEAKAGES.conf                  REQUEST-951-REQUEST-TOO-LARGE.conf               restricted-files.data
```

```
REQUEST-931-APPLICATION-ATTACK-RFI.conf                REQUEST-951-DATA-LEAKAGES-SQL.conf              REQUEST-951-REQUEST-TOO-LARGE.conf               restricted-upload.data
```

```
[root@kali]~/usr/share/modsecurity-crs/rules
```

```
# cp /usr/share/modsecurity-crs/rules/REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf.example /usr/share/modsecurity-crs/rules/REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf
```

- Configure Apache para incluir las reglas descargadas y asegúrese de que estén activas revisando los logs de Apache después de intentar un ataque bloqueado.

Después de realizar las modificaciones anteriores, vamos a la carpeta “mods-available” y buscamos el archivo security2.conf para su edición.

```

# /usr/share/modsecurity-cs/rules
# /etc/apache2

root@kali:~# ls /etc/apache2
ls
apache2.conf  conf-available  conf-enabled  envvars  magic  mods-available  mods-enabled  ports.conf  sites-available  sites-enabled

root@kali:~# ls /etc/apache2
ls
conf  mods-available

root@kali:~# ls /etc/apache2/mods-available
ls
access_compat.load  authn_file.load  buffer.load  dbd.load  headers.load  ldap.conf  mpm_prefork.load  proxy_fcgi.load  remoteip.load  slotmem_plain.load  userdir.conf
actions.conf        authn_socache.load  cache.load  dav_lock.load  heartbeat.load  ldap.load  mpm_worker.conf  proxy_fdpass.load  reqtimeout.conf  slotmem_shm.load  userdir.load
allowmethods.conf   authn_fcgi.load     cache_disk.conf  deflate.conf  heartmonitor.load  ldap_debug.load  mpm_worker.load  proxy_fip.conf  request.conf  socache_shmcb.load  usertrack.load
alias.load           authn_core.load     cache_disk.load  deflate.conf  httpd_forensic.load  lua.load  negotiation.conf  proxy_hcheck.load  rewrite.load  socache_redis.load  whost.alias.load
allowmimetypes.load  authn_dbd.load     certm_meta.load  dir.load      ident.load      macro.load  php5-2.conf  proxy_html.conf  security2.conf  socache_shmcb.load  xml2enc.load
auth.load            authz_dbm.load      cgi.load         imagemap.load  md.load         mpm2-2.load  proxy_balancer.conf  proxy_http.load  security2.load  spelling.load
auth_basic.load       auth_groupfile.load  cgid.load        include.load   mime.conf        proxy.conf  proxy_http2.load  session.load  session_cookie.load  ssl.conf
auth_digest.load     auth_host.load      cgid.load        info.conf      mime_magic.conf  proxy_ajp.load  session_crypto.load  session.load  session_crypto.load  status.load
auth_form.load        authn_anon.load     charset.load     info.load      mime_magic.load  proxy_balancer.load  session_crypto.load  session.load  substitute.load  suexec.load
authn_core.load       authn_user.load     data.load        info.load      mpm_event.conf  proxy_balancer.load  session_crypto.load  session.load  substitute.load  suexec.load
authn_dbm.load        autoindex.conf      dav.load         filter.load    mpm_event.conf  proxy_balancer.load  session_crypto.load  session.load  substitute.load  suexec.load
authn_dbm.load        brotli.load         dav_fs.load      filter.load    mpm_event.conf  proxy_balancer.load  session_crypto.load  session.load  substitute.load  suexec.load

```

```
(root@kali)-[/etc/apache2/mods-available]
# nano security2.conf
```

El archivo viene por defecto de la siguiente manera:

```
GNU nano 7.2
<IfModule security2_module>
    # Default Debian dir for modsecurity's persistent data
    SecDataDir /var/cache/modsecurity

    # Include all the *.conf files in /etc/modsecurity.
    # Keeping your local configuration in that directory
    # will allow for an easy upgrade of THIS file and
    # make your life easier
    IncludeOptional /etc/modsecurity/*.conf

    # Include OWASP ModSecurity CRS rules if installed
    IncludeOptional /usr/share/modsecurity-crs/*.load
</IfModule>
```


Y después de cargarle las reglas correspondientes debería quedar de esta manera:

```
GNU nano 7.2 /etc/modsecurity/modsecurity.conf
<IfModule security2_module>
    # Default Debian dir for modsecurity's persistent data
    SecDataDir /var/cache/modsecurity

    # Include all the *.conf files in /etc/modsecurity.
    # Keeping your local configuration in that directory
    # will allow for an easy upgrade of THIS file and
    # make your life easier
    IncludeOptional /etc/modsecurity/*.conf
    Include /etc/modsecurity/rules/*.conf

    # Include OWASP ModSecurity CRS rules if installed
    # IncludeOptional /usr/share/modsecurity-crs/*.load
</IfModule>
```

Ahora nos vamos a la carpeta *downloads* y ejecutamos el siguiente comando para traer un conjunto de reglas que hay en este otro directorio de github:

```
(root@kali)~# cd /etc/apache2/mods-available
# cd /home/kali/Downloads
(root@kali)~# git clone https://github.com/coreruleset/coreruleset.git
Cloning into 'coreruleset'...
remote: Enumerating objects: 30393, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 30393 (delta 0), reused 1 (delta 0), pack-reused 30392
Receiving objects: 100% (30393/30393), 8.00 MiB | 6.49 MiB/s, done.
Resolving deltas: 100% (23751/23751), done.
```

Copiamos el fichero “coreruleset” que hemos bajado de github en una carpeta que llamaremos *rules* dentro de la de *Modsecurity*. Aunque también la podríamos haber movido simplemente mediante el comando “mv”.

```
(root@kali)~# cp -av rules /etc/modsecurity/
rules' → '/etc/modsecurity/rules/'
rules/REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf.example' → '/etc/modsecurity/rules/REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf.example'
rules/REQUEST-901-INITIALIZATION.conf' → '/etc/modsecurity/rules/REQUEST-901-INITIALIZATION.conf'
rules/REQUEST-905-COMMON-EXCEPTIONS.conf' → '/etc/modsecurity/rules/REQUEST-905-COMMON-EXCEPTIONS.conf'
rules/REQUEST-911-METHOD-ENFORCEMENT.conf' → '/etc/modsecurity/rules/REQUEST-911-METHOD-ENFORCEMENT.conf'
rules/REQUEST-913-SCANNER-DETECTION.conf' → '/etc/modsecurity/rules/REQUEST-913-SCANNER-DETECTION.conf'
rules/REQUEST-920-PROTOCOL-ENFORCEMENT.conf' → '/etc/modsecurity/rules/REQUEST-920-PROTOCOL-ENFORCEMENT.conf'
rules/REQUEST-921-PROTOCOL-ATTACK.conf' → '/etc/modsecurity/rules/REQUEST-921-PROTOCOL-ATTACK.conf'
rules/REQUEST-922-MULTIPART-ATTACK.conf' → '/etc/modsecurity/rules/REQUEST-922-MULTIPART-ATTACK.conf'
rules/REQUEST-930-APPLICATION-ATTACK-LFI.conf' → '/etc/modsecurity/rules/REQUEST-930-APPLICATION-ATTACK-LFI.conf'
rules/REQUEST-931-APPLICATION-ATTACK-RFI.conf' → '/etc/modsecurity/rules/REQUEST-931-APPLICATION-ATTACK-RFI.conf'
rules/REQUEST-932-APPLICATION-ATTACK-RCE.conf' → '/etc/modsecurity/rules/REQUEST-932-APPLICATION-ATTACK-RCE.conf'
rules/REQUEST-933-APPLICATION-ATTACK-PHP.conf' → '/etc/modsecurity/rules/REQUEST-933-APPLICATION-ATTACK-PHP.conf'
rules/REQUEST-934-APPLICATION-ATTACK-GENERIC.conf' → '/etc/modsecurity/rules/REQUEST-934-APPLICATION-ATTACK-GENERIC.conf'
rules/REQUEST-941-APPLICATION-ATTACK-XSS.conf' → '/etc/modsecurity/rules/REQUEST-941-APPLICATION-ATTACK-XSS.conf'
rules/REQUEST-942-APPLICATION-ATTACK-SQLI.conf' → '/etc/modsecurity/rules/REQUEST-942-APPLICATION-ATTACK-SQLI.conf'
rules/REQUEST-943-APPLICATION-ATTACK-SESSION-FIXATION.conf' → '/etc/modsecurity/rules/REQUEST-943-APPLICATION-ATTACK-SESSION-FIXATION.conf'
rules/REQUEST-944-APPLICATION-ATTACK-JAVA.conf' → '/etc/modsecurity/rules/REQUEST-944-APPLICATION-ATTACK-JAVA.conf'
rules/REQUEST-949-BLOCKING-EVALUATION.conf' → '/etc/modsecurity/rules/REQUEST-949-BLOCKING-EVALUATION.conf'
rules/RESPONSE-950-DATA-LEAKAGES.conf' → '/etc/modsecurity/rules/RESPONSE-950-DATA-LEAKAGES.conf'
rules/RESPONSE-951-DATA-LEAKAGES-SQL.conf' → '/etc/modsecurity/rules/RESPONSE-951-DATA-LEAKAGES-SQL.conf'
rules/RESPONSE-952-DATA-LEAKAGES-JAVA.conf' → '/etc/modsecurity/rules/RESPONSE-952-DATA-LEAKAGES-JAVA.conf'
rules/RESPONSE-953-DATA-LEAKAGES-PHP.conf' → '/etc/modsecurity/rules/RESPONSE-953-DATA-LEAKAGES-PHP.conf'
rules/RESPONSE-954-DATA-LEAKAGES-IIS.conf' → '/etc/modsecurity/rules/RESPONSE-954-DATA-LEAKAGES-IIS.conf'
rules/RESPONSE-955-WEB-SHELLS.conf' → '/etc/modsecurity/rules/RESPONSE-955-WEB-SHELLS.conf'
rules/RESPONSE-959-BLOCKING-EVALUATION.conf' → '/etc/modsecurity/rules/RESPONSE-959-BLOCKING-EVALUATION.conf'
rules/RESPONSE-980-CORRELATION.conf' → '/etc/modsecurity/rules/RESPONSE-980-CORRELATION.conf'
rules/RESPONSE-999-EXCLUSION-RULES-AFTER-CRS.conf.example' → '/etc/modsecurity/rules/RESPONSE-999-EXCLUSION-RULES-AFTER-CRS.conf.example'
rules/iis-errors.data' → '/etc/modsecurity/rules/iis-errors.data'
rules/java-classes.data' → '/etc/modsecurity/rules/java-classes.data'
rules/java-code-leakages.data' → '/etc/modsecurity/rules/java-code-leakages.data'
rules/java-errors.data' → '/etc/modsecurity/rules/java-errors.data'
rules/lfi-os-files.data' → '/etc/modsecurity/rules/lfi-os-files.data'
rules/php-config-directives.data' → '/etc/modsecurity/rules/php-config-directives.data'
rules/php-errors-pl2.data' → '/etc/modsecurity/rules/php-errors-pl2.data'
rules/php-errors.data' → '/etc/modsecurity/rules/php-errors.data'
rules/php-function-names-933150.data' → '/etc/modsecurity/rules/php-function-names-933150.data'
rules/php-function-names-933151.data' → '/etc/modsecurity/rules/php-function-names-933151.data'
rules/php-variables.data' → '/etc/modsecurity/rules/php-variables.data'
rules/restricted-files.data' → '/etc/modsecurity/rules/restricted-files.data'
```

Ahora comprobamos que esté la carpeta *rules* dentro de la de *Modsecurity*:

```
(root@kali)-[/home/kali/Downloads/coreruleset]
# cd /etc/modsecurity

(root@kali)-[/etc/modsecurity]
# ls
crs  crs-setup.conf  modsecurity.conf  modsecurity.conf-recommended  rules  unicode.mapping
```

- Realice pruebas para verificar que las nuevas reglas bloqueen un intento de inyección SQL similar al de la Actividad 2.

Hacemos un reinicio de apache 2 para actualizar las condiciones del servidor y después intentaremos abrir una *shell*.

```
(root@kali)-[/etc/apache2/mods-available]
# systemctl restart apache2

(root@kali)-[/etc/apache2/mods-available]
# curl http://localhost/index.html?exec=/bin/bash
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access this resource.</p>
<hr>
<address>Apache/2.4.58 (Debian) Server at localhost Port 80</address>
</body></html>
```

Vemos que aparece un error 403, lo cual quiere decir que está todo correcto porque está bloqueando el acceso.

Actividad 5

- Cambia el banner de identificación del servidor Apache para eliminar cualquier tipo de vulnerabilidad.

Para ello vamos a modificar el fichero “security.conf” que se encuentra dentro de la carpeta “conf-available” de apache 2.

Ejecutamos los siguientes comandos para su realización:

```
(root@kali)~[/etc/apache2/mods-available]
# cd /etc/apache2

(root@kali)~[/etc/apache2]
# ls
apache2.conf  conf-available  conf-enabled  envvars  magic  mods-available  mods-enabled  ports.conf  sites-available  sites-enabled

(root@kali)~[/etc/apache2]
# cd conf-available

(root@kali)~[/etc/apache2/conf-available]
# ls
charset.conf  javascript-common.conf  localized-error-pages.conf  other-vhosts-access-log.conf  security.conf  serve-cgi-bin.conf

(root@kali)~[/etc/apache2/conf-available]
# nano security.conf
```

Así es como aparece por defecto el fichero “security.conf”:

```
GNU nano 7.2
# Changing the following options will not really affect the security of the
# server, but might make attacks slightly more difficult in some cases.

#
# ServerTokens
# This directive configures what you return as the Server HTTP response
# Header. The default is 'Full' which sends information about the OS-type
# and compiled in modules.
# Set to one of: Full | OS | Minimal | Minor | Major | Prod
# where Full conveys the most information, and Prod the least.
#ServerTokens Minimal
ServerTokens OS
#ServerTokens Full

#
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (internal error documents, FTP directory
# listings, mod_status and mod_info output etc., but not CGI generated
# documents or custom error documents).
# Set to "Email" to also include a mailto: link to the ServerAdmin.
# Set to one of: On | Off | Email
#ServerSignature Off
ServerSignature On

#
# Allow TRACE method
#
# Set to "extended" to also reflect the request body (only for testing and
# diagnostic purposes).
#
# Set to one of: On | Off | extended
TraceEnable Off
#TraceEnable On

#
# Forbid access to version control directories
#
# If you use version control systems in your document root, you should
# probably deny access to their directories.
#
# Examples:
#
#RedirectMatch 404 /\.git
#RedirectMatch 404 /\.svn
```

Debajo de la línea "ServerSignature On", escribiremos "SecServerSignature Windows" para que no aparezca la identificación del servidor Apache sino la de Windows Server para "despistar".

```
#
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (internal error documents, FTP directory
# listings, mod_status and mod_info output etc., but not CGI generated
# documents or custom error documents).
# Set to "Email" to also include a mailto: link to the ServerAdmin.
# Set to one of: On | Off | Email
#ServerSignature Off
ServerSignature On
SecServerSignature Windows
```

Y la identificación que aparece es la siguiente:

```
(root@kali)-[/etc/apache2/conf-available]
# curl http://localhost/index.html?exec=/bin/bash
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access this resource.</p>
<hr>
<address>Windows Server at localhost Port 80</address>
</body></html>
```

Pero si no quisiéramos mostrar ninguna identificación, simplemente deberíamos cambiar "ServerSignature On" a "ServerSignature Off".

```
#
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (internal error documents, FTP directory
# listings, mod_status and mod_info output etc., but not CGI generated
# documents or custom error documents).
# Set to "Email" to also include a mailto: link to the ServerAdmin.
# Set to one of: On | Off | Email
#ServerSignature Off
ServerSignature Off
SecServerSignature Windows
```

Y se vería de la siguiente manera:

```
(root@kali)-[/etc/apache2/conf-available]
# curl http://localhost/index.html?exec=/bin/bash
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access this resource.</p>
</body></html>
```

- Bloquea una página http y genera una regla que pueda visualizarse en los logs de Apache. Muestre el bloqueo y los resultados.

Para ello vamos a modificar el archivo "000-default.conf" que se encuentra dentro de la carpeta "sites-available" de apache 2:

```
(root@kali)~[/etc/apache2/conf-available]
# cd /etc/apache2
# ls
apache2.conf  conf-available  conf-enabled  envvars  magic  mods-available  mods-enabled  ports.conf  sites-available  sites-enabled
# cd /etc/apache2
# cd sites-available
# ls
000-default.conf  default-ssl.conf
# nano 000-default.conf
```

```
GNU nano 7.2 /etc/apache2/sites-available/000-default.conf
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

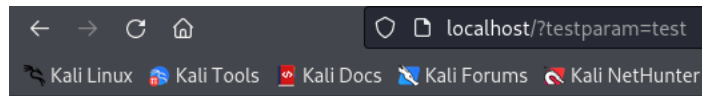
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf
</VirtualHost>
```

Añadimos las dos últimas líneas en blanco que se ven en la captura de arriba. Guardamos y salimos del archivo de configuración. Realizamos un reinicio de apache2 siempre que hagamos alguna modificación. Ahora, si escribimos en el navegador el siguiente enlace: <http://localhost/?testparam=test>, nos saldrá un error por falta de permisos:



Forbidden

You don't have permission to access this resource.

Para ver los *logs*, demos ir al directorio `/var/log/apache2` y allí dentro nos saldrán los diferentes ficheros *log* que existen:

```

root@kali:~# cd /etc/apache2/sites-available
root@kali:~# cd /var/log
root@kali:~# ls
README      Xorg.1.log      alternatives.log.1  apt            boot.log.2      boot.log.5      dpkg.log        faillog        inetssm        lastlog        machanger.log.1.gz  mosquitto        openvpn        redis          speech-dispatcher  sysstat
Xorg.0.log  Xorg.1.log.0ld  alternatives.log.2.gz  boot.log       boot.log.3      btup            dpkg.log.1      fontconfig.log  installer      lightdm        machanger.log.2.gz  nginx            postgresql     runit          sssd              wtmp
Xorg.0.log.0ld  alternatives.log  apache2            boot.log.1     boot.log.4      btup.1          dpkg.log.2.gz   gvmm           journal        macchanger.log     machanger.log.3.gz  notus-scanner   private        samba          stunnel4

root@kali:~# cd /var/log
root@kali:~# cd apache2
root@kali:~# cd /var/log/apache2
root@kali:~# ls
access.log  error.log  modsec_audit.log  other_vhosts_access.log

root@kali:~# cd /var/log/apache2
root@kali:~# ls -l
total 80
-rw-r--r-- 1 root adm 12516 May 7 16:27 access.log
-rw-r--r-- 1 root adm 23818 May 7 16:27 error.log
-rw-r--r-- 1 root root 32179 May 7 16:27 modsec_audit.log
-rw-r--r-- 1 root adm 0 Mar 26 14:08 other_vhosts_access.log

```

Si por ejemplo queremos ver todos los logs que hay dentro del archivo “error.log” escribimos el comando “cat error.log” y nos aparece lo siguiente:

```
[*] May 07 13:07:02.103261 [error] log
[*] May 07 09:28:10.411383 [2024] [mpm_preforknotice] [pid 7902] AH00163: Apache/2.4.58 (debian) configured - resuming normal operations
[*] May 07 09:28:10.411444 [2024] [corenotice] [pid 7902] AH00094: Command line: '/usr/sbin/apache2'
[*] May 07 11:57:28.137572 [2024] [mpm_preforknotice] [pid 7902] AH00163: caught SIGINCH, shutting down gracefully
[*] May 07 11:57:28.137572 [2024] [mpm_preforknotice] [pid 95062] AH00163: Apache/2.4.58 (debian) configured - resuming normal operations
[*] May 07 11:57:38.266016 [2024] [corenotice] [pid 95062] AH00094: Command line: '/usr/sbin/apache2'
[*] May 07 12:28:17.322397 [2024] [client 192.168.122.32] AH00525: Invalid method in request WHUL / HTTP/1.1
[*] May 07 12:28:17.200317 [2024] [mpm_preforknotice] [pid 95062] AH00170: caught SIGINCH, shutting down gracefully
[*] May 07 12:28:17.368489 [2024] [security2:notice] [pid 112127] ModSecurity for Apache/2.9.7 (http://www.modsecurity.org/) configured.
[*] May 07 12:28:17.368489 [2024] [security2:notice] [pid 112127] ModSecurity: APR compiled version="1.7.2"; loaded version="1.7.2"
[*] May 07 12:28:17.368489 [2024] [security2:notice] [pid 112127] ModSecurity: PCRE2 compiled version="10.42"; loaded version="10.42 2022-12-11"
[*] May 07 12:28:17.368489 [2024] [security2:notice] [pid 112127] ModSecurity: LUA compiled version="Lua 5.1"
[*] May 07 12:28:17.368489 [2024] [security2:notice] [pid 112127] ModSecurity: YAJL compiled version="2.0.9"
[*] May 07 12:28:17.368489 [2024] [security2:notice] [pid 112127] ModSecurity: IDLM compiled version="2.0.4"
[*] May 07 12:28:17.368489 [2024] [security2:notice] [pid 112127] ModSecurity: Status engine is currently disabled, enabling it by set SecActionEngine to On.
[*] May 07 12:28:17.494647 [2024] [mpm_preforknotice] [pid 112118] AH00163: Apache/2.4.58 (debian) configured - resuming normal operations
[*] May 07 12:28:17.494647 [2024] [corenotice] [pid 112118] AH00094: Command line: '/usr/sbin/apache2'
[*] May 07 13:05:13.638817 [2024] [mpm_preforknotice] [pid 128689] AH00163: caught SIGINCH, shutting down gracefully
[*] May 07 13:05:14.088145 [2024] [security2:notice] [pid 128689] ModSecurity for Apache/2.9.7 (http://www.modsecurity.org/) configured.
[*] May 07 13:05:14.088228 [2024] [security2:notice] [pid 128689] ModSecurity: APR compiled version="1.7.2"; loaded version="1.7.2"
[*] May 07 13:05:14.088228 [2024] [security2:notice] [pid 128689] ModSecurity: PCRE2 compiled version="10.42"; loaded version="10.42 2022-12-11"
[*] May 07 13:05:14.088228 [2024] [security2:notice] [pid 128689] ModSecurity: LUA compiled version="Lua 5.1"
[*] May 07 13:05:14.088228 [2024] [security2:notice] [pid 128689] ModSecurity: YAJL compiled version="2.0.9"
[*] May 07 13:05:14.088228 [2024] [security2:notice] [pid 128689] ModSecurity: IDLM compiled version="2.0.4"
[*] May 07 13:05:14.088228 [2024] [security2:notice] [pid 128689] ModSecurity: Status engine is currently disabled, enabling it by set SecActionEngine to On.
[*] May 07 13:05:14.247496 [2024] [mpm_preforknotice] [pid 128688] AH00163: Apache/2.4.58 (debian) configured - resuming normal operations
[*] May 07 13:05:14.247496 [2024] [corenotice] [pid 128688] AH00094: Command line: '/usr/sbin/apache2'
[*] May 07 13:05:32.532421 [2024] [security2:error] [pid 128693] [client 127.0.0.1] ModSecurity: Warning: detected SQL injection libnjsbin with fingerprint '5b1c' [file "/usr/share/modsecurity-crs/rules/REQUEST-949-Blocklist-Engine-000-Blocklist.conf"] [line "50"] [msg "SQL Injection Attack Detected via libnjsbin"] [data "Matched Data: 5b1c found within ARGS:username; or i=1"] [severity "CRITICAL"] [ver "OWASP_CRS/3.3.5"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-sqli"] [tag "paranoma-level-3"] [tag "OWASP_CSC"] [tag "cpe:/a:libnjsbin/2.0.46/6"] [tag "PCGI/5.0.5"] [hostname "localhost"] [uri "/vuln.php"] [url_scheme "http"] [location-multi]
[*] May 07 13:05:32.532421 [2024] [security2:error] [pid 128693] [client 127.0.0.1] ModSecurity: Warning: detected SQL injection libnjsbin with fingerprint '5b1c' [file "/usr/share/modsecurity-crs/rules/REQUEST-949-Blocklist-Engine-000-Blocklist.conf"] [line "50"] [msg "SQL Injection Attack Detected via libnjsbin"] [data "Matched Data: 5b1c found within ARGS:username; or i=1"] [severity "CRITICAL"] [ver "OWASP_CRS/3.3.5"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-sqli"] [tag "paranoma-level-3"] [tag "OWASP_CSC"] [tag "cpe:/a:libnjsbin/2.0.46/6"] [tag "PCGI/5.0.5"] [hostname "localhost"] [uri "/vuln.php"] [url_scheme "http"] [location-multi]
[*] May 07 13:05:32.532823 [2024] [security2:error] [pid 128693] [client 127.0.0.1] ModSecurity: Warning: detected SQL injection libnjsbin with fingerprint '5b1c' [file "/usr/share/modsecurity-crs/rules/RESPONSE-980-CORRELATION.conf"] [line "980110"] [msg "Inbound Anomaly Score Exceeded (Total Inbound Score: 5 - SQL=5, XSS=0, RFI=0, LFI=0, RCE=0, PMF=0, HTTP=0, SESS=0): individual paranoma level scores: 5, 0, 0, 0"] [ver "OWASP_CRS/3.3.5"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-sqli"] [tag "paranoma-level-3"] [tag "OWASP_CSC"] [tag "cpe:/a:libnjsbin/2.0.46/6"] [tag "PCGI/5.0.5"] [hostname "localhost"] [uri "/vuln.php"] [url_scheme "http"] [location-multi]
[*] May 07 13:05:32.532823 [2024] [security2:error] [pid 128696] [client 127.0.0.1] ModSecurity: Warning: detected SQL injection libnjsbin with fingerprint '5b1c' [file "/usr/share/modsecurity-crs/rules/REQUEST-949-Blocklist-Engine-000-Blocklist.conf"] [line "50"] [msg "SQL Injection Attack Detected via libnjsbin"] [data "Matched Data: 5b1c found within ARGS:username; or i=1"] [severity "CRITICAL"] [ver "OWASP_CRS/3.3.5"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-sqli"] [tag "paranoma-level-3"] [tag "OWASP_CSC"] [tag "cpe:/a:libnjsbin/2.0.46/6"] [tag "PCGI/5.0.5"] [hostname "localhost"] [uri "/vuln.php"] [url_scheme "http"] [location-multi]
[*] May 07 13:05:32.532823 [2024] [security2:error] [pid 128696] [client 127.0.0.1] ModSecurity: Warning: detected SQL injection libnjsbin with fingerprint '5b1c' [file "/usr/share/modsecurity-crs/rules/RESPONSE-980-CORRELATION.conf"] [line "980110"] [msg "Inbound Anomaly Score Exceeded (Total Inbound Score: 5 - SQL=5, XSS=0, RFI=0, LFI=0, RCE=0, PMF=0, HTTP=0, SESS=0): individual paranoma level scores: 5, 0, 0, 0"] [ver "OWASP_CRS/3.3.5"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-sqli"] [tag "paranoma-level-3"] [tag "OWASP_CSC"] [tag "cpe:/a:libnjsbin/2.0.46/6"] [tag "PCGI/5.0.5"] [hostname "localhost"] [uri "/vuln.php"] [url_scheme "http"] [location-multi]
[*] May 07 13:05:32.532823 [2024] [security2:error] [pid 128696] [client 127.0.0.1] ModSecurity: Warning: detected SQL injection libnjsbin with fingerprint '5b1c' [file "/usr/share/modsecurity-crs/rules/REQUEST-949-Blocklist-Engine-000-Blocklist.conf"] [line "50"] [msg "SQL Injection Attack Detected via libnjsbin"] [data "Matched Data: 5b1c found within ARGS:username; or i=1"] [severity "CRITICAL"] [ver "OWASP_CRS/3.3.5"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-sqli"] [tag "paranoma-level-3"] [tag "OWASP_CSC"] [tag "cpe:/a:libnjsbin/2.0.46/6"] [tag "PCGI/5.0.5"] [hostname "localhost"] [uri "/vuln.php"] [url_scheme "http"] [location-multi]
[*] May 07 13:05:32.532823 [2024] [security2:error] [pid 128696] [client 127.0.0.1] ModSecurity: Warning: detected SQL injection libnjsbin with fingerprint '5b1c' [file "/usr/share/modsecurity-crs/rules/RESPONSE-980-CORRELATION.conf"] [line "980110"] [msg "Inbound Anomaly Score Exceeded (Total Inbound Score: 5 - SQL=5, XSS=0, RFI=0, LFI=0, RCE=0, PMF=0, HTTP=0, SESS=0): individual paranoma level scores: 5, 0, 0, 0"] [ver "OWASP_CRS/3.3.5"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-sqli"] [tag "paranoma-level-3"] [tag "OWASP_CSC"] [tag "cpe:/a:libnjsbin/2.0.46/6"] [tag "PCGI/5.0.5"] [hostname "localhost"] [uri "/vuln.php"] [url_scheme "http"] [location-multi]
[*] May 07 13:05:32.532823 [2024] [security2:error] [pid 128696] [client 127.0.0.1] ModSecurity: Warning: detected SQL injection libnjsbin with fingerprint '5b1c' [file "/usr/share/modsecurity-crs/rules/REQUEST-949-Blocklist-Engine-000-Blocklist.conf"] [line "50"] [msg "SQL Injection Attack Detected via libnjsbin"] [data "Matched Data: 5b1c found within ARGS:username; or i=1"] [severity "CRITICAL"] [ver "OWASP_CRS/3.3.5"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-sqli"] [tag "paranoma-level-3"] [tag "OWASP_CSC"] [tag "cpe:/a:libnjsbin/2.0.46/6"] [tag "PCGI/5.0.5"] [hostname "localhost"] [uri "/vuln.php"] [url_scheme "http"] [location-multi]
[*] May 07 13:05:32.532823 [2024] [security2:error] [pid 128696] [client 127.0.0.1] ModSecurity: Warning: detected SQL injection libnjsbin with fingerprint '5b1c' [file "/usr/share/modsecurity-crs/rules/RESPONSE-980-CORRELATION.conf"] [line "980110"] [msg "Inbound Anomaly Score Exceeded (Total Inbound Score: 5 - SQL=5, XSS=0, RFI=0, LFI=0, RCE=0, PMF=0, HTTP=0, SESS=0): individual
```

Para filtrar por parámetro, al comando anterior le debemos agregar “ | grep + “parámetro elegido” ”.

Vemos un ejemplo de los logs de “test con éxito”:

```
[root@error-log ~]# cat /var/log/apache2/error.log | grep "Test con exito"
```

Buen trabajo, aunque quizás debería haber tenido más explicación de algunas cosas, haber profundizado más en comprender el por qué. **9/10**