

# **SEGURIDAD EN APLICACIONES WEB**



2021

## CONVENCIONES USADAS EN LA GUIA

### ➤ Comandos a ejecutarse en una consola:

```
find / -iname "*nombre a buscar*"
```

```
execute -f NetworkPasswordDump32.exe -a "-f red.txt"  
-H
```

Cuando el comando es muy largo y no entra en una línea en este texto guía se usa un backslash (\) al final de la primera línea para que se ejecute correctamente todo el comando

```
curl -H "user-agent: () { :; }; echo; echo; /bin/bash -c \  
'cat /etc/passwd;'" http://10.8.0.1:8081/cgi-bin/stats
```

# ÍNDICE

<b>1 CONFIGURACIÓN INICIAL</b>	<b>1</b>
1.1 Descargando la Máquina virtual	1
1.2 Configurar burpsuite	2
<b>1.2 Instalar las herramientas</b>	<b>9</b>
1.3 Conectarse al Laboratorio Virtual	10
<b>2. CONOCIMIENTOS BÁSICOS</b>	<b>13</b>
2.1 Terminología	13
2.2 Protocolos de seguridad	15
2.3 Tipos de Pruebas de seguridad	<b>17</b>
2.3.1 Según el acceso a información	17
2.3.2 Según el tipo de prueba	17
<b>3. SEGURIDAD EN EL DISEÑO DE LA APLICACIÓN</b>	<b>18</b>
3.1 Modelado de amenazas	18
<b>4 MEJORES PRÁCTICAS DE DESARROLLO</b>	<b>18</b>
<b>5 OWASP TOP 10</b>	<b>19</b>
5.1 Inyección.	19
5.2 Autenticación rota.	19
5.3 Exposición de datos sensibles.	19
5.4 Entidades externas XML (XXE).	19
5.5 Control de acceso roto.	19
5.6 Mala configuración de seguridad	19
5.7 Secuencias de comandos entre sitios (XSS)	19
5.8 Deserialización insegura	19
5.9 Uso de componentes con vulnerabilidades conocidas	19
5.10 Registro y monitoreo insuficientes	22
<b>6 BIBLIOGRAFÍA</b>	<b>23</b>

# 1 CONFIGURACIÓN INICIAL

## 1.1 Descargando la Máquina virtual

Usaremos la distribución de KALI Linux que es una distribución basada en **Debian** diseñada principalmente para la auditoría y seguridad informática en general. Fue creada y es mantenida por Offensive Security.

Descargaremos la imagen de la máquina virtual ya sea para VirtuaBox o VMware desde el siguiente link:

<https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-image-download/>

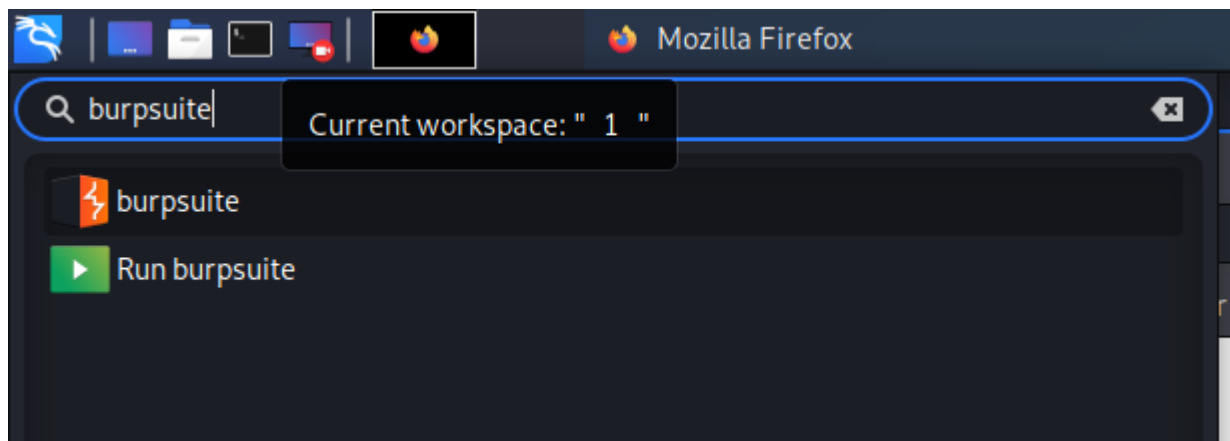
Para ingresar al sistema usaremos las credenciales **kali/kali**



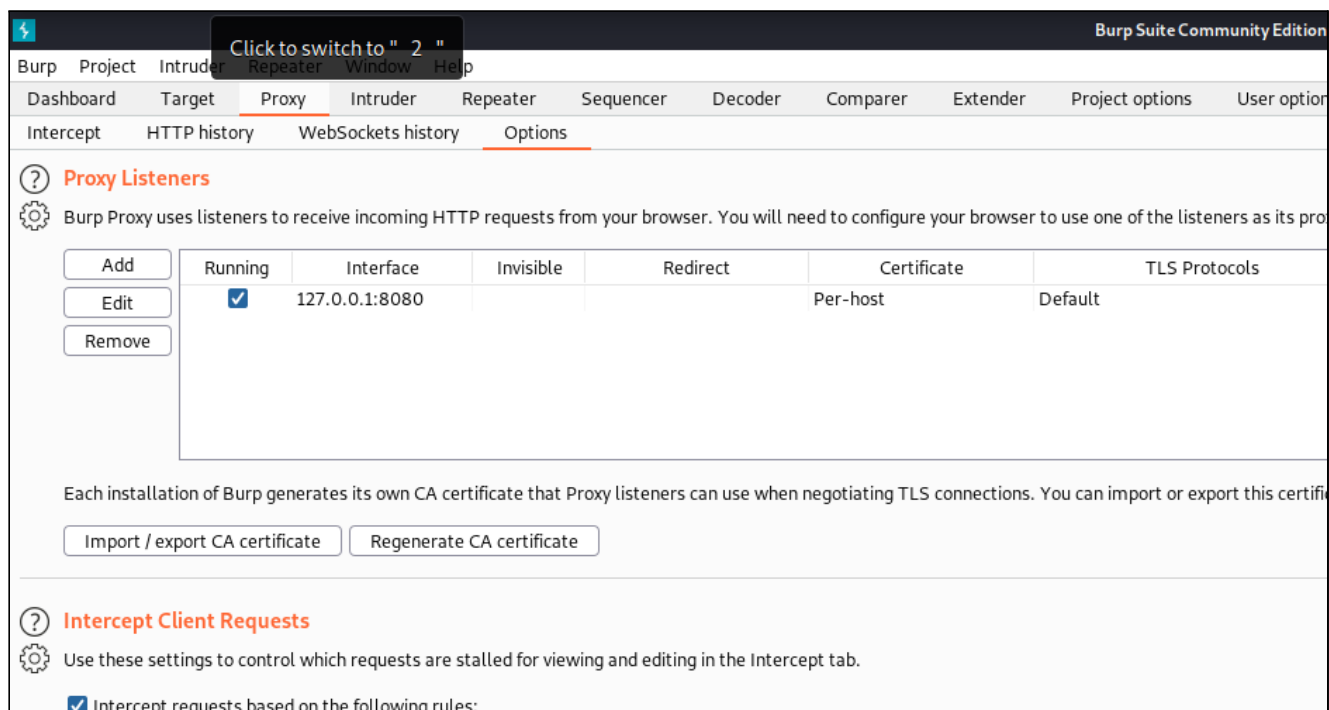
## 1.2 Configurar burpsuite

Burpsuite es un proxy reverso que nos permite interceptar las peticiones de nuestro navegador antes de enviarlas al servidor web, esto para poder realizar pruebas de seguridad.

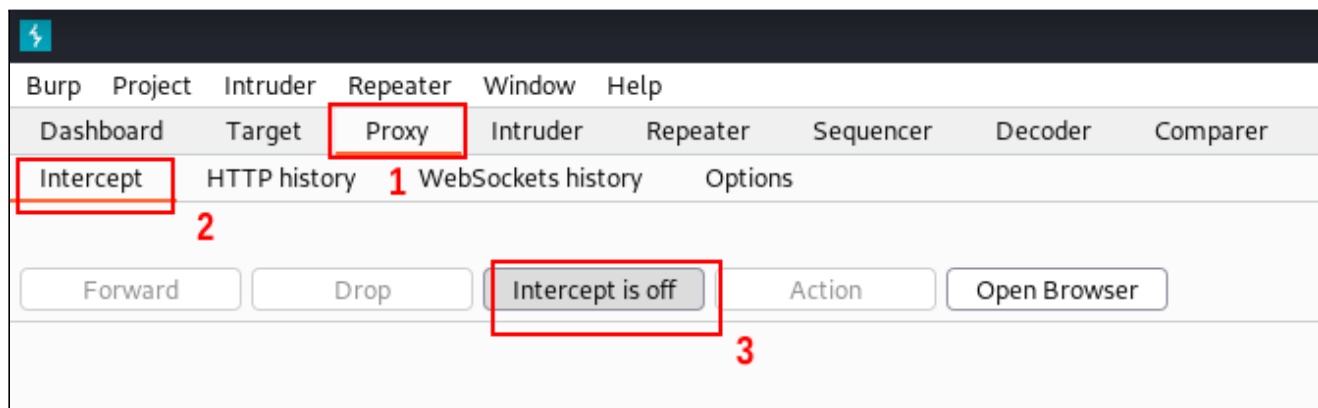
- Abriremos la herramienta burpsuite



- Elegimos las opciones por defecto y entrara a la aplicación:



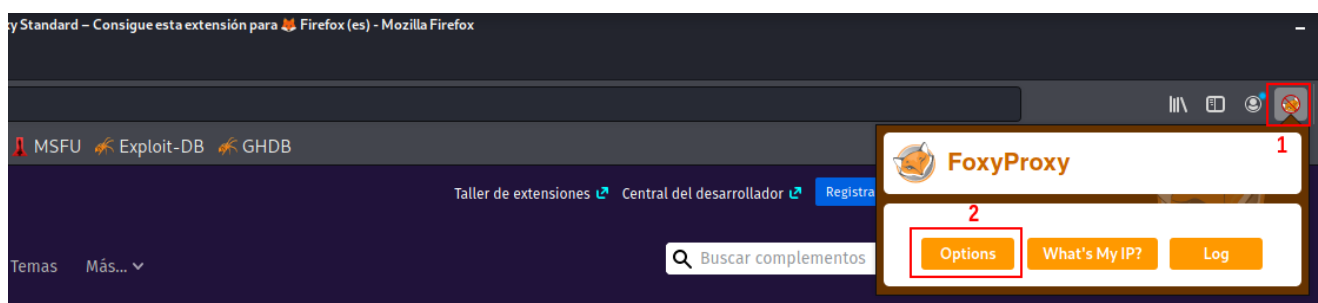
- Deshabilitamos que intercepte cada petición del navegador



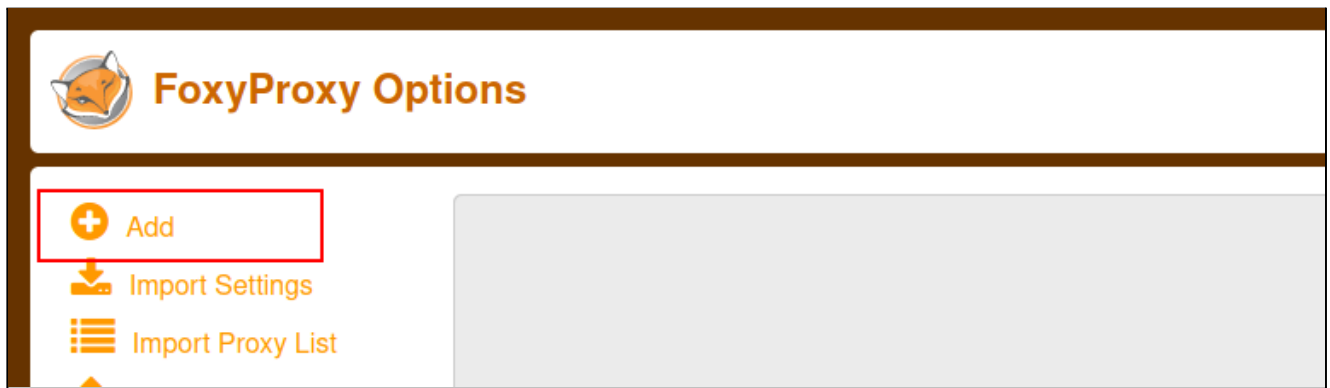
- Ahora necesitamos conectar BurpSuite con nuestro navegador firefox (KALI Linux), para ellos instalaremos el add-on <https://addons.mozilla.org/es/firefox/addon/foxyproxy-standard/>



- Una vez instalado procederemos a configurarlo entrando a "Options"



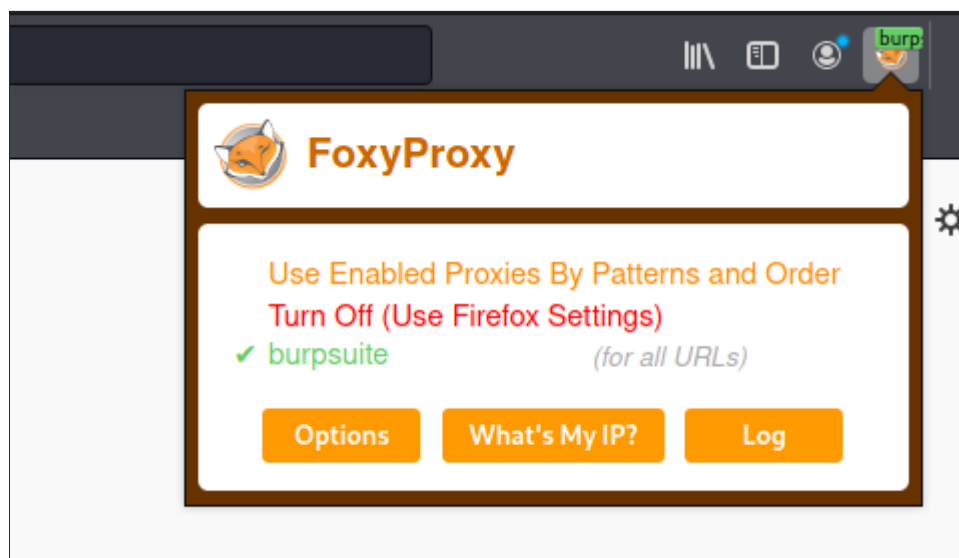
- Añadimos un nuevo proxy



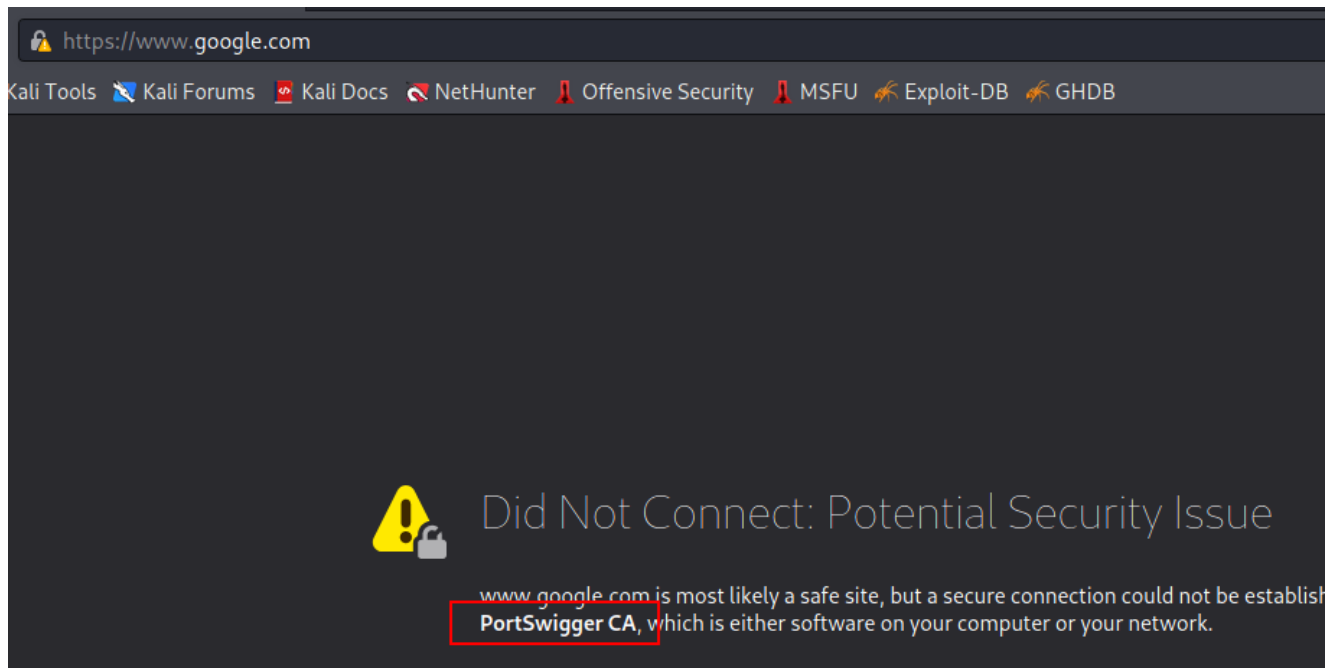
- Configuramos los siguientes valores

The screenshot shows the 'Edit Proxy burpsuite' configuration window. It has several fields and options: 'Title or Description (optional)' with the value 'burpsuite' (highlighted with a red box); 'Color' with a green bar and the hex code '#66cc66'; 'Proxy Type' set to 'HTTP'; 'Proxy IP address or DN' set to '127.0.0.1' (highlighted with a red box); 'Port' set to '8080' (highlighted with a red box); 'Username (optional)' with the placeholder 'username'; and 'Password (optional)' with a masked field '\*\*\*\*\*'.

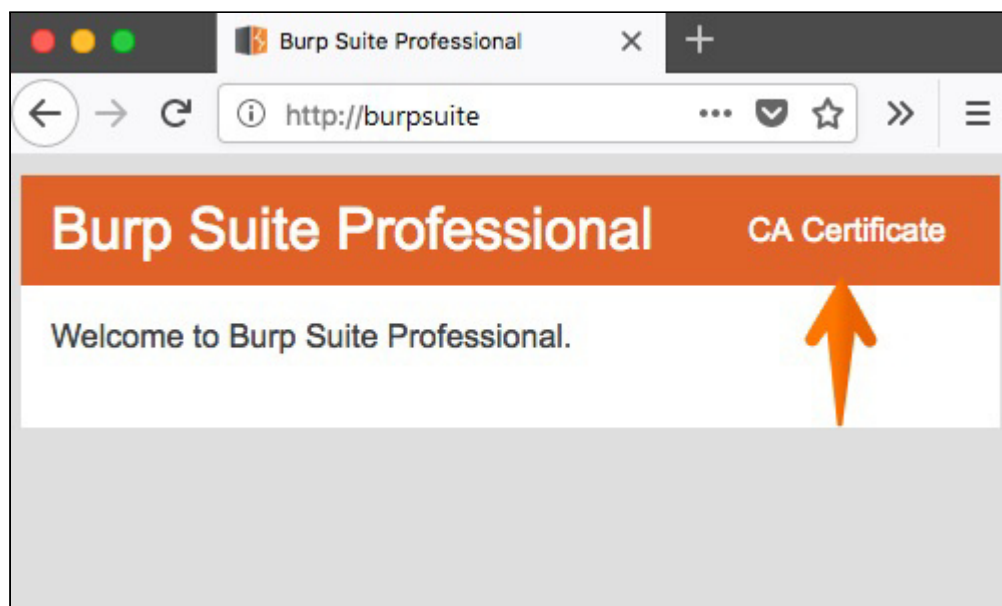
- Una vez guardado la configuración ya elegir el proxy “Burpsuite”



- Cuando intentemos ingresar a un sitio que use SSL (HTTPS) nos saldrá un error de que no confía en el “CA PortSwigger”

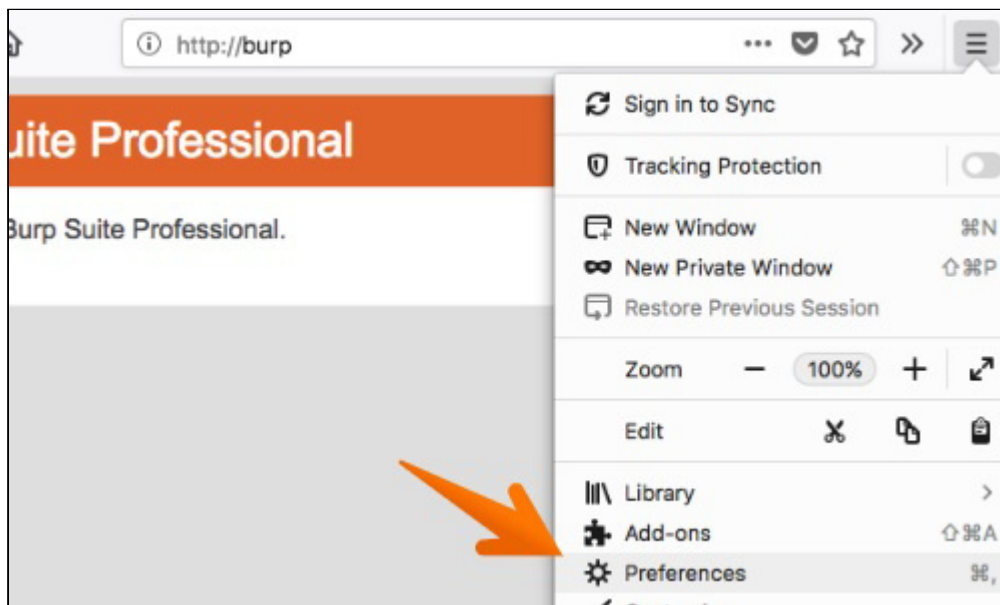


- Tenemos que instalar el certificado de burpsuite en el navegador para que pueda confiar en esa autoridad certificadora. Ingresamos la url <http://burp> y descargamos el certificado cacert.der

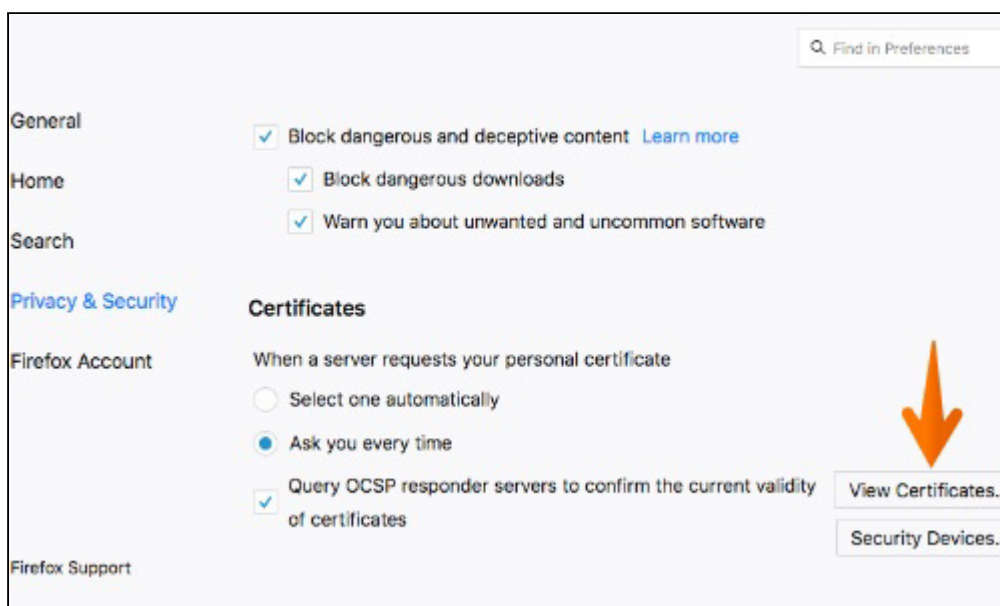




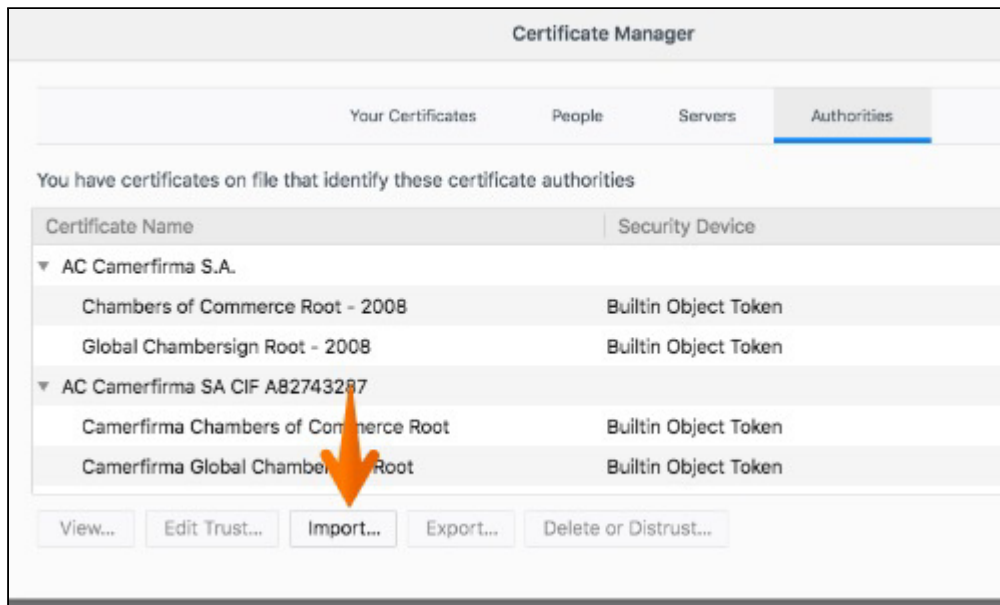
- Hacemos click en “preferencias”



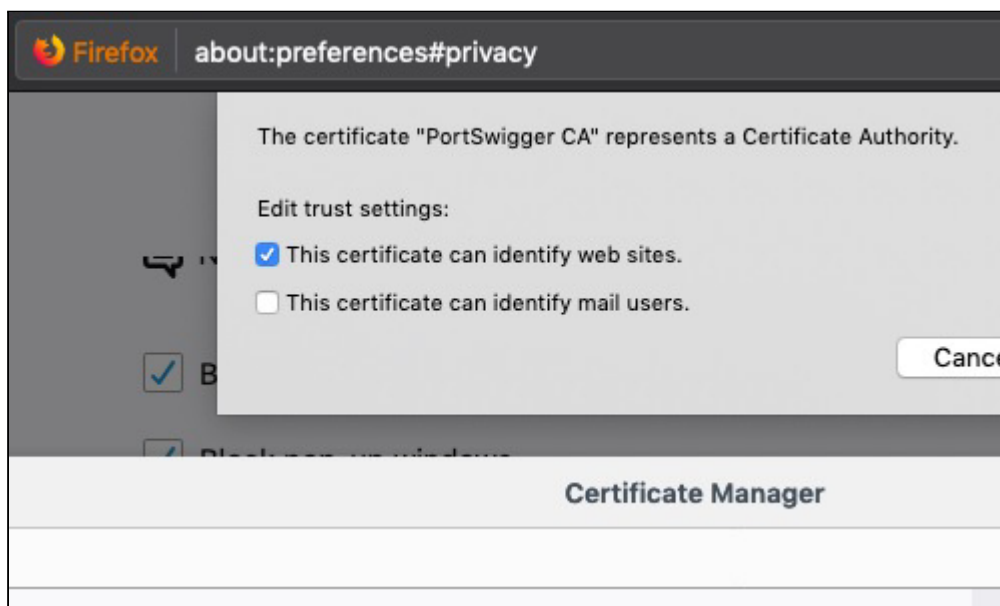
- En la opción "Privacy and Security" ir a la sección "Certificates" (Parte inferior) y hacer click en el botón "View certificates".



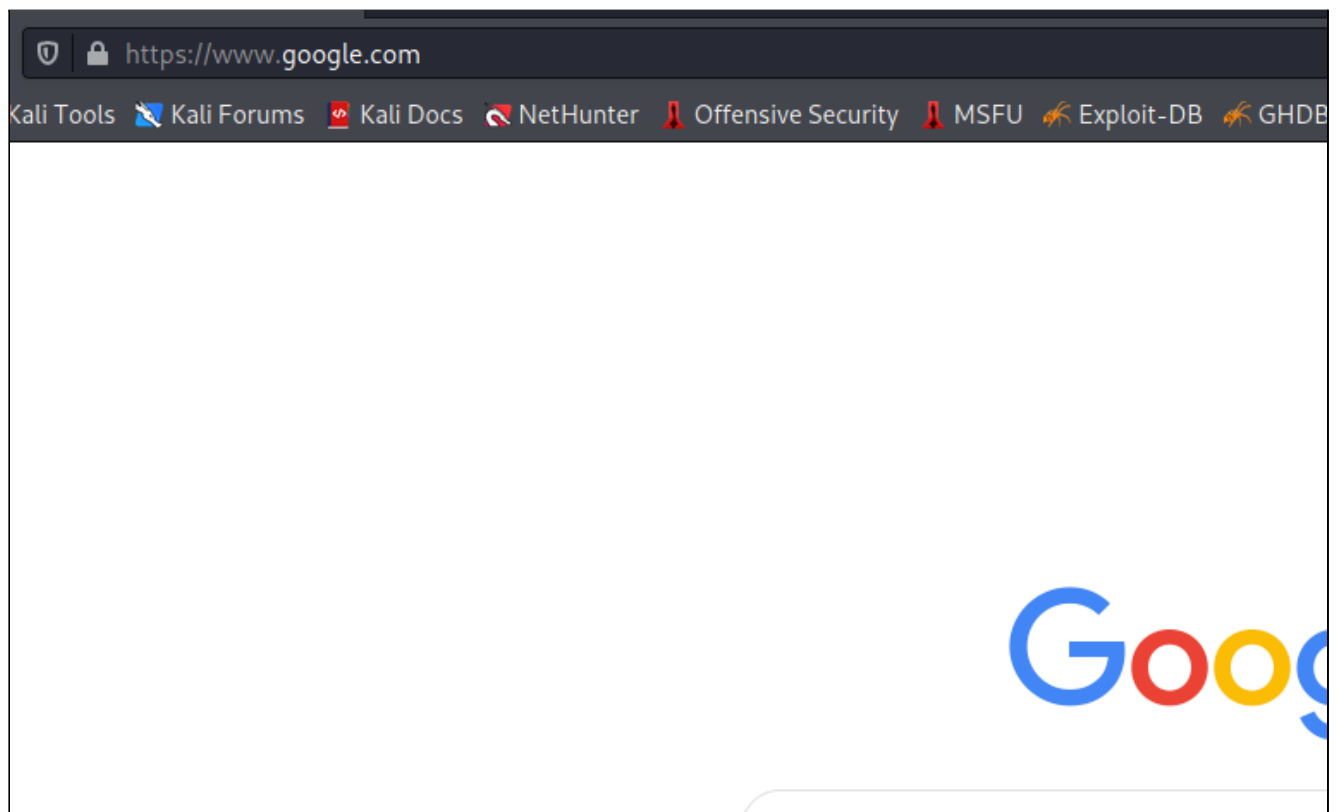
- Hacer click en “import” y seleccionar el certificado **cacert.der** anteriormente descargado



- Terminamos la importación con las siguientes opciones:



- Volvemos a cargar el sitio de google y esta vez no muestra ningún error



- En BurpSuite en la sección de historial podemos observar que se registró todas las peticiones a google de manera correcta

Current workspace: 1				
Dashboard	Target	1 Proxy	Intruder	Repeater
Intercept	2 HTTP history	WebSockets history	Options	
Filter: Hiding CSS, image and general binary content				
#	Host	Method	URL	Params
	https://www.google.com	GET	/	
	https://www.google.com	GET	/xjs/_/js/k=xjs.s.es_419.jzKf-s9R-1Y.O...	
	https://www.gstatic.com	GET	/og/_/js/k=og.qtm.en_US.TJ22SR6ED...	
	https://www.google.com	POST	/gen_204?s=webhp&t=aft&atyp=csi&...	✓
	https://www.google.com	GET	/images/searchbox/desktop_searchbo...	
	https://apis.google.com	GET	/_scs/abc-static/_/js/k=gapi.gapi.en....	
	https://www.google.com	GET	/complete/search?q&cp=0&client=gw...	✓
	https://www.google.com	GET	/xjs/_/js/k=xjs.s.es_419.jzKf-s9R-1Y.O...	✓
	https://www.google.com	GET	/xjs/_/js/k=xjs.s.es_419.jzKf-s9R-1Y.O...	✓
	https://www.google.com	POST	/gen_204?atyp=csi&r=1&ei=3ctEYM7...	✓
	https://ogs.google.com	GET	/widget/app/so?bc=1&origin=https%3...	✓
	https://adservice.google.com	GET	/adsid/google/ui	
	https://adservice.google.com.bo	GET	/adsid/google/ui?gadsid=AORoGNS9T...	✓

## 1.2 Instalar las herramientas

- Clonamos el repositorio que contiene el script de instalación de herramientas y los documentos que se usarán en este curso

```
git clone https://github.com/DanielTorres1/web-security
```

```
cd web-security
```


```
kali@kali: ~ × kali@kali: ~/web-security ×
```

```
(kali㉿kali)-[~]  
$ git clone https://github.com/DanielTorres1/web-security  
Cloning into 'web-security' ...  
remote: Enumerating objects: 6, done.  
remote: Counting objects: 100% (6/6), done.  
remote: Compressing objects: 100% (5/5), done.  
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0  
Receiving objects: 100% (6/6), 84.56 KiB | 424.00 KiB/s, done.  
  
(kali㉿kali)-[~]  
$ cd web-security  
  
(kali㉿kali)-[~/web-security]  
$
```

- Ejecutaremos el script

```
bash web-tools.sh
```

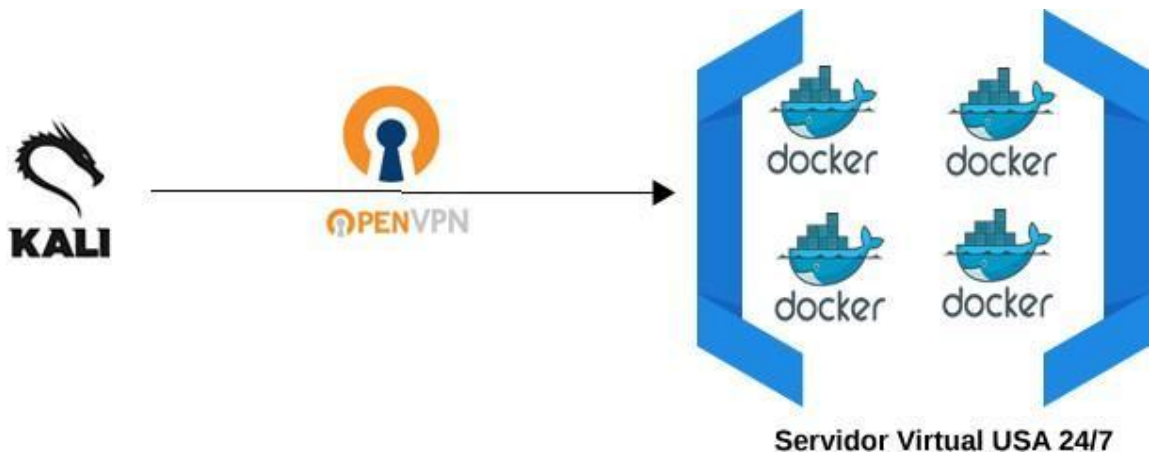
```
(kali㉿kali)-[~]  
$ bash web-tools.sh
```



```
[+] Instalando herramientas disponibles en repositorio  
[sudo] password for kali:  
Get:1 http://mirrors.ocf.berkeley.edu/kali kali-rolling InRelease [30.5 kB]  
Get:2 http://mirrors.ocf.berkeley.edu/kali kali-rolling/main amd64 Package  
Get:3 http://mirrors.ocf.berkeley.edu/kali kali-rolling/main amd64 Content  
Get:4 http://mirrors.ocf.berkeley.edu/kali kali-rolling/non-free amd64 Pac  
Get:5 http://mirrors.ocf.berkeley.edu/kali kali-rolling/non-free amd64 Cor
```

### 1.3 Conectarse al Laboratorio Virtual

Para realizar la conexión al laboratorio virtual se usará la herramienta openvpn. En este laboratorio virtual existen varias instancias de contenedores docker con aplicaciones vulnerable



Para indicar que la práctica se la realiza en la nube, se muestra el siguiente banner:



- 1.- Se les enviará a su correo electrónico el archivo cliente.ovpn
- 2.- Deberán copiar el archivo cliente.ovpn a la carpeta **/home/kali** de la máquina virtual KALI Linux
- 3.- En una consola de KALI Linux ejecutamos el siguiente comando en el directorio donde se encuentra el archivo cliente.ovpn

```
sudo openvpn cliente.ovpn
```

```
kali@kali: ~  
File Actions Edit View Help  
kali@kali: ~ x kali@kali: ~ x  
(kali@kali)-[~]  
$ sudo openvpn cliente.ovpn  
2021-03-03 09:28:42 WARNING: Compression for receiving enabled. Compression has been used in the past  
kets are not compressed unless "allow-compression yes" is also set.  
2021-03-03 09:28:42 Unrecognized option or missing or extra parameter(s) in cliente.ovpn:15: block-out  
2021-03-03 09:28:42 DEPRECATED OPTION: --cipher set to 'AES-256-CBC' but missing in --data-ciphers (AE  
e OpenVPN version will ignore --cipher for cipher negotiations. Add 'AES-256-CBC' to --data-ciphers or  
' to --data-ciphers-fallback 'AES-256-CBC' to silence this warning.  
2021-03-03 09:28:42 OpenVPN 2.5.0 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH  
28 2020  
2021-03-03 09:28:42 library versions: OpenSSL 1.1.1i 8 Dec 2020, LZO 2.10  
2021-03-03 09:28:42 Outgoing Control Channel Authentication: Using 512 bit message hash 'SHA512' for H  
2021-03-03 09:28:42 Incoming Control Channel Authentication: Using 512 bit message hash 'SHA512' for H  
2021-03-03 09:28:42 TCP/UDP: Preserving recently used remote address: [AF_INET]66.172.33.234:1194  
2021-03-03 09:28:42 Socket Buffers: R=[212992→212992] S=[212992→212992]  
2021-03-03 09:28:42 UDP link local: (not bound)  
2021-03-03 09:28:42 UDP link remote: [AF_INET]66.172.33.234:1194  
2021-03-03 09:28:42 TLS: Initial packet from [AF_INET]66.172.33.234:1194, sid=49cbfa93 8a177596  
2021-03-03 09:28:42 VERIFY OK: depth=1, CN=ChangeMe  
2021-03-03 09:28:42 VERIFY KU OK  
2021-03-03 09:28:42 Validating certificate extended key usage  
2021-03-03 09:28:42 ++ Certificate has EKU (str) TLS Web Server Authentication, expects TLS Web Server  
2021-03-03 09:28:42 VERIFY ECU OK  
2021-03-03 09:28:42 VERIFY OK: depth=0, CN=server  
2021-03-03 09:28:43 Control Channel: TLSv1.2, cipher TLSv1.2 DHE-RSA-AES256-GCM-SHA384, 2048 bit RSA
```

Esa consola **NO** debemos cerrarla.

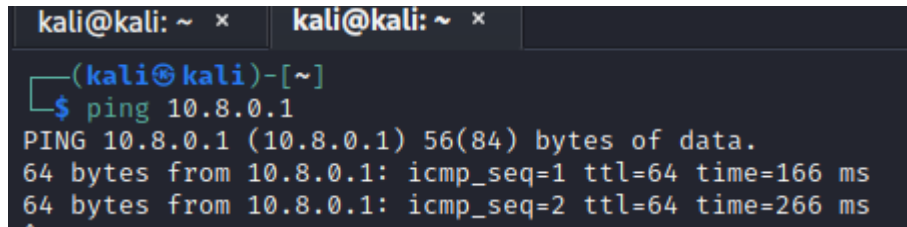
- Ejecutamos el siguiente comando en otra consola para verificar que se estableció la conexión: Deberá existir una interfaz **tun0** con a IP: **10.8.0.x**

```
ip a s tun0
```

```
kali@kali: ~ x kali@kali: ~ x  
(kali@kali)-[~]  
$ ip a s tun0  
4: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc pfifo_f  
link/none  
inet 10.8.0.2/24 scope global tun0  
valid_lft forever preferred_lft forever  
inet6 fe80::958f:6a5d:3bb9:154c/64 scope link stable-privacy  
valid_lft forever preferred_lft forever
```

- Si tenemos conectividad con el servidor quiere decir que todo está bien.

```
ping 10.8.0.1
```



```
kali@kali: ~ x kali@kali: ~ x
(kali@kali)-[~]
$ ping 10.8.0.1
PING 10.8.0.1 (10.8.0.1) 56(84) bytes of data.
64 bytes from 10.8.0.1: icmp_seq=1 ttl=64 time=166 ms
64 bytes from 10.8.0.1: icmp_seq=2 ttl=64 time=266 ms
```

Para “salir” del **laboratorio virtual** deberemos presionar CTRL+C en la consola donde se realizó la conexión de OpenVPN.

```
Thu Dec 12 15:23:23 2019 Initialization Sequence Completed
^CThu Dec 12 15:23:25 2019 event_wait : Interrupted system call (code=4)
Thu Dec 12 15:23:25 2019 /sbin/ip route del 66.172.33.234/32
Thu Dec 12 15:23:25 2019 /sbin/ip route del 0.0.0.0/1
Thu Dec 12 15:23:25 2019 /sbin/ip route del 128.0.0.0/1
Thu Dec 12 15:23:25 2019 Closing TUN/TAP interface
Thu Dec 12 15:23:25 2019 /sbin/ip addr del dev tun0 10.8.0.2/24
Thu Dec 12 15:23:25 2019 SIGINT[hard,] received, process exiting
[root:/etc/openvpn/hacking]# █
```

## 2. CONOCIMIENTOS BÁSICOS

### 2.1 Terminología

#### ➤ Vulnerabilidad

Existencia de un **error de diseño** o **implementación** que puede conducir a un evento indeseable o inesperado comprometiendo la seguridad de la aplicación.

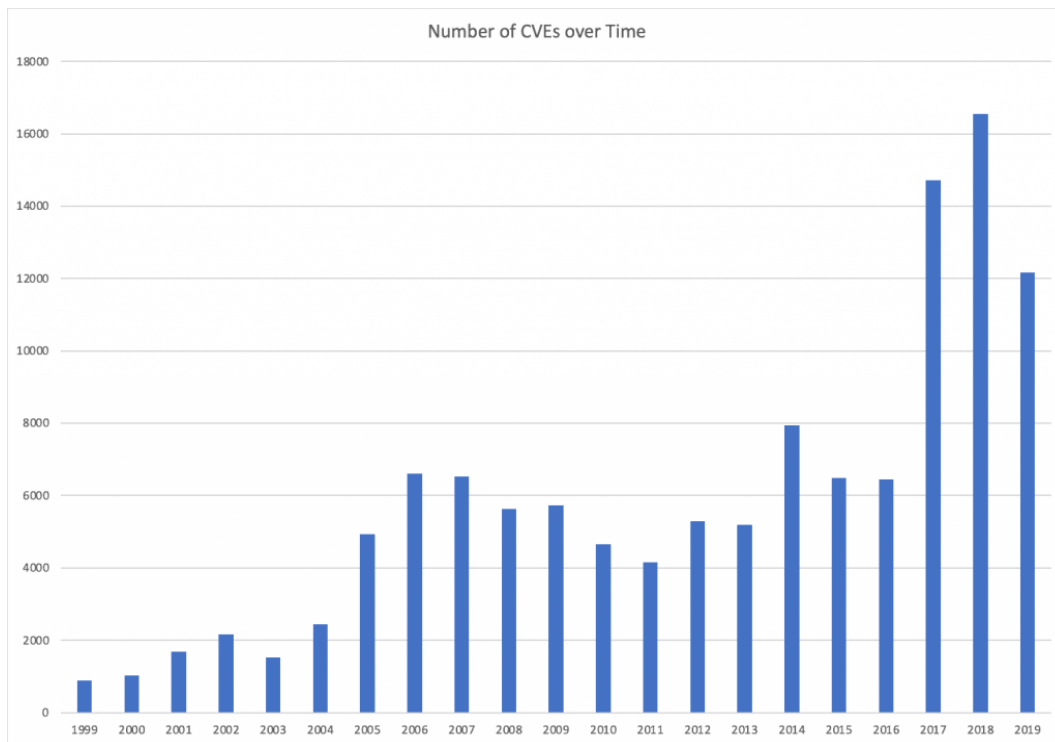
Ejemplo: Inyección SQL, componentes antiguo, etc

#### ➤ Taxonomía de Vulnerabilidades

- **Common Weakness Enumeration (CWE)** es una lista formal o diccionario de debilidades comunes del software que pueden ocurrir en la arquitectura del software, diseño, código o implementación. Por ejemplo el error de no realizar la expiración de la sesión adecuadamente tiene el código CWE-613  
<https://cwe.mitre.org/data/definitions/613.html>
- **Common vulnerabilities & exposures (CVE)** es una base de datos de vulnerabilidades publicadas de diferentes productos (para versiones en específico). Ej Vulnerabilidad en el paquete restify-pagination versión 0.0.5 para Node.js con el código CVE -2020-27543  
<https://www.incibe-cert.es/alerta-temprana/vulnerabilidades/cve-2020-27543>

Las vulnerabilidades publicadas cada año son cada vez más, por lo que se debe asegurar que en nuestros proyectos no usemos software o componentes con vulnerabilidades.





### ➤ El Proyecto de seguridad de aplicaciones web abiertas (OWASP)

Es una comunidad abierta dedicada a permitir que las organizaciones desarrollen, adquieran y mantengan aplicaciones seguras.

El producto más conocido es El **OWASP Top 10** que es una lista de los 10 problemas de seguridad web más comunes. Se usará este top 10 para realizar las revisiones de seguridad

### ➤ Incidente

Es cuando una vulnerabilidad es explotada y que genera un impacto en la entidad. Un incidente puede afectar a la integridad, confidencialidad o disponibilidad de los datos

### ➤ Exploit

Es una manera específica de comprometer la seguridad de una aplicación o sistema mediante la **explotación de una vulnerabilidad**. Generalmente es un **script** hecho en cualquier lenguaje (python, perl, C++, ruby, etc)

Ej: El exploit para la vulnerabilidad de versiones de Drupal menores a 7.58 y 8.5.1 y que tiene el código CVE-2018-7600 es el siguiente:

<https://github.com/r3dexpl0it/CVE-2018-7600/blob/master/payload.py>

### ➤ Payload

Es la parte de un **exploit**, que tiene de objetivo realizar una acción específica (Enviar una shell remota, abrir un puerto, escribir un archivo, etc).

En el caso del anterior exploit el payload lo que hace es escribir en el archivo r3dexploit.txt con el contenido “vulnerable to cve-7600-2018 exploit”:

```
'markup', 'mail[#markup]': 'echo "vulnerable to cve-7600-2018 exploit" | tee r3dexploit.txt'}
```

### ➤ Superficie de ataque

Son todos los posibles puntos de entrada que un atacante puede usar para atacar la aplicación. Por ejemplo, si la aplicación empieza a usar el user agent del navegador como parámetro de identificación del cliente y lo almacena en una base de datos un atacante puede enviar varios payloads para conseguir un comportamiento anómalo de la aplicación.

## 2.2 Protocolos de seguridad

### ➤ Política del mismo origen (SOP)

Que es una política restrictiva que evita que un sitio web acceda a recursos de otro sitio web.

Por ejemplo un usuario accede al sitio **abc.com** en una pestaña y al sitio **xyz.com** en otra pestaña pero en el mismo navegador, el sitio **abc.com** no puede acceder a recursos como cookies, localStorage, etc del sitio **xyz.com** y viceversa

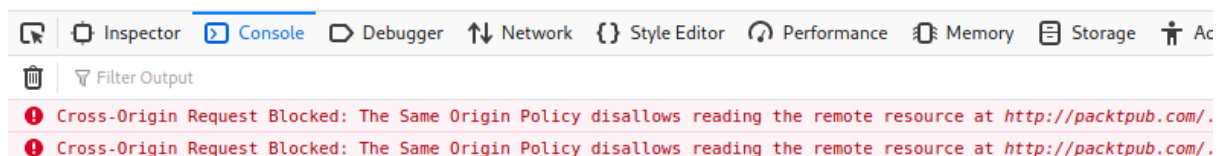
Ej: si un usuario carga en una pestaña el sitio: <http://example.com/meme/derp.html> y carga en otra pestaña las siguientes URLs que intenta acceder a recursos de la primera URL el resultado de la política del mismo origen (SOP) será el siguiente:

URL	Resultado	Explicación
http://example.com/random/derp.html	Accede	La ruta no importa
http://example.com/other/meme/derp.html	Accede	La ruta no importa
http://www.example.com/meme/derp.html	Rechazado	Diferente dominio
http://example.com:8081/meme/derp.html	Rechazado	Diferente puerto
ftp://example.com/meme/derp.html	Rechazado	Diferente protocolo
http://demo.example.com/meme/derp.html	Rechazado	Diferente dominio
http://packtpub.com/meme/derp.html	Rechazado	Diferente dominio

Una excepción a la política del mismo origen son las imágenes, por eso se pueden cargar en un sitio imágenes de cualquier dominio.

Ejemplo práctico:

- Con firefox abrir el archivo **SOP.html** que está dentro de la carpeta “**2 CONOCIMIENTOS BÁSICOS**”
- Para visualizar la consola web presionamos CTRL + Shift + K



Vemos que el acceso al dominio packpub.com no está permitido por la política del mismo origen.

### ➤ Intercambio de Recursos de Origen Cruzado (CORS)

Permite el intercambio de datos HTTP entre dominios, lo que significa que una página que se ejecuta en el origen A puede enviar/recibir datos de un servidor en el origen B.

CORS se usa abundantemente en aplicaciones web donde los recursos se cargan desde diferentes orígenes

## 2.3 Tipos de Pruebas de seguridad

### 2.3.1 Según el acceso a información

#### ➤ **Prueba de Caja Blanca.**

Se tiene acceso a:

- diagramas de la aplicación
- código fuente
- usuarios de prueba
- historias de usuario

#### ➤ **Prueba de Caja Negra.**

No se tiene ningún tipo de conocimiento anticipado sobre la aplicación como ser historias de usuario, no se tiene usuarios de prueba y tampoco se tiene acceso al código fuente.

### 2.3.2 Según el tipo de prueba

#### ➤ **Estática**

Son pruebas que se realizan al código fuente de la aplicación, estas pruebas pueden automáticas o manuales

#### ➤ **Dinámica**

Estas pruebas se la realizan sobre una aplicación desplegada

### 3. SEGURIDAD EN EL DISEÑO DE LA APLICACIÓN

#### 3.1 Modelado de amenazas

- Revisar el documento: **MODELADO DE AMENAZAS DE PROYECTOS DE SOFTWARE.pdf**
- Revisar el ejemplo de modelado de amenaza: **Modelado-Amenaza.ods**

### 4 MEJORES PRÁCTICAS DE DESARROLLO

- Práctica 1 : Aplicar defensa en profundidad (Seguridad por capas)
- Práctica 2 : Utilice un modelo de seguridad positivo (Listas blancas)
- Práctica 3 : Fallar de forma segura (Manejo de errores de forma segura)
- Práctica 4 : Ejecutar con el mínimo de privilegios.
- Práctica 5 : Evite la seguridad por oscuridad (tokens de sesión propios)
- Práctica 6 : Mantenga la seguridad simple.
- Práctica 7 : Detecta intrusiones (Capacidad para registrar eventos de seguridad)
- Práctica 8 : No confiar en la infraestructura.
- Práctica 9 : No confiar en los servicios.
- Práctica 10 : Establezca valores predeterminados seguros (La complejidad de la contraseña deben estar habilitados de forma predeterminada)

## 5 OWASP TOP 10

El primer paso para realizar las pruebas de seguridad es determinar toda la **superficie de ataque** de la aplicación, esto se lo realiza haciendo uso de todas las funcionalidades de la aplicación (crear usuarios, ver items, buscar items, subir archivos, etc) y con todos los roles

5.1 Inyección.

5.2 Autenticación rota.

5.3 Exposición de datos sensibles.

5.4 Entidades externas XML (XXE).

5.5 Control de acceso roto.

5.6 Mala configuración de seguridad

5.7 Secuencias de comandos entre sitios (XSS)

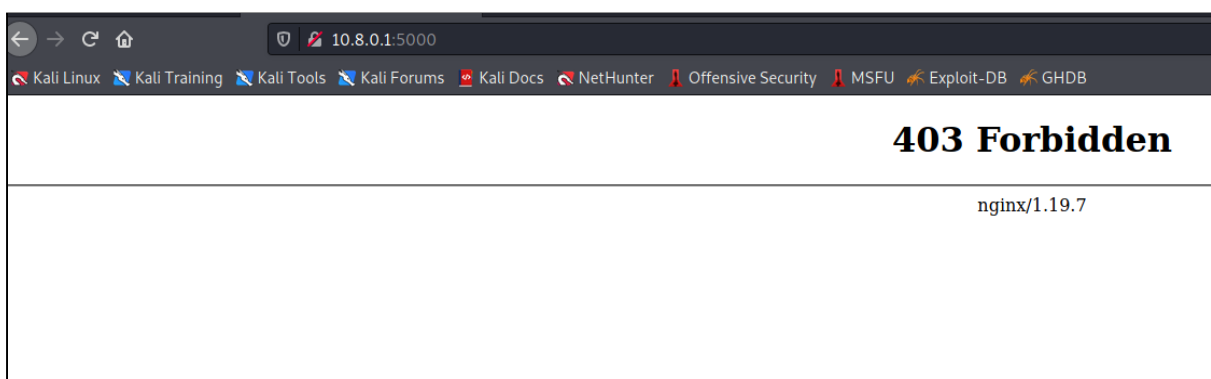
5.8 Deserialización insegura

5.9 Uso de componentes con vulnerabilidades conocidas

### a) Servidores web con configuraciones incorrectas

#### ➤ Nginx

Usaremos la aplicación que esta en la url: <http://10.8.0.1:5000/>



Esta aplicación usa el archivo de configuración “**/home/kali/web-security/5 OWASP TOP 10/A9/nginx.conf**”

- Ubicación del directorio raíz (/) faltante

La directiva root se configurará globalmente, lo que significa que las solicitudes al directorio raíz (/) lo llevarán a la ruta local /etc/nginx. Por ejemplo nos permitirá bajar el archivo de configuración nginx.conf

<http://10.8.0.1:5000/nginx.conf>

- Slash (/) faltante en location

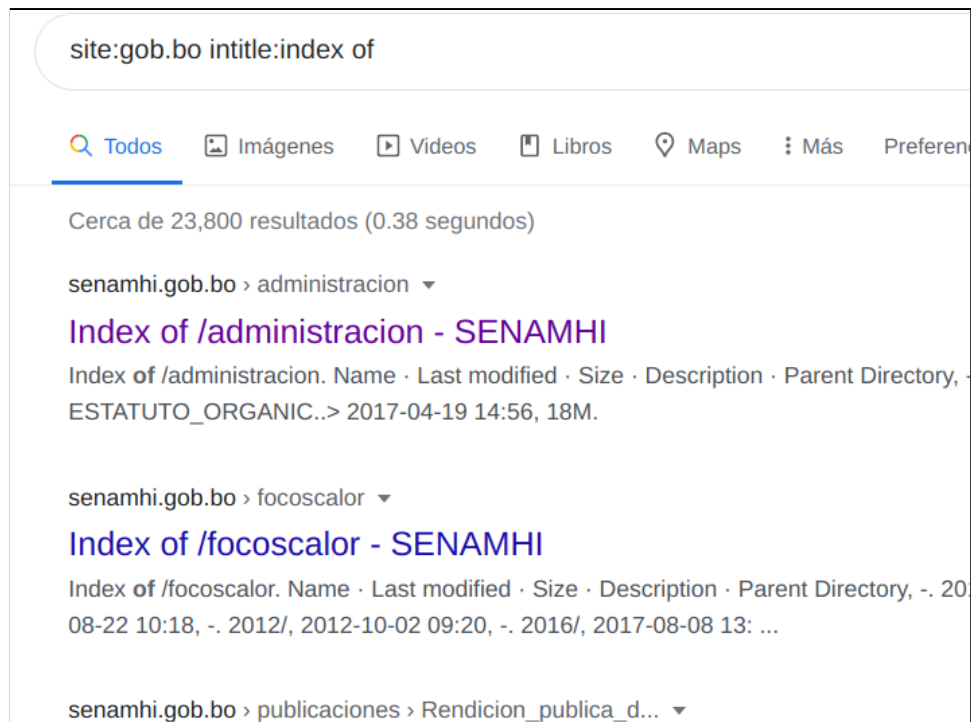
```
location /image {  
    proxy_pass http://apache:80/catpictures/  
}
```

Con esta configuración incorrecta, es posible acceder a un directorio superior del directorio catpictures: <http://10.8.0.1:5000/image../secret.html>

## ➤ Apache

- Listado de directorios

Para identificar qué sitios web usan esta configuración incorrecta usaremos el dork de google: **site:gob.bo intitle:index of**



## b) Componentes de node.js con vulnerabilidades

Para revisar si algun modulo que usamos tiene una vulnerabilidad conocida usaremos la herramienta snyk.io

- Necesitaremos una cuenta en SNYK.IO, nos logueamos en la URL <https://app.snyk.io/login> (KALI Linux)
- Procedemos a autenticar la aplicación cliente  
snyk auth
- Finalmente realizamos la comprobación de vulnerabilidades de un proyecto

```
cd /home/kali/web-security/5-OWASP-TOP-10/A9/backend-master  
snyk test
```



```

Issues to fix by upgrading:
Upgrade axios@0.19.2 to axios@0.21.1 to fix
X Server-Side Request Forgery (SSRF) [Medium Severity][https://snyk.io/vuln/SNYK-JS-AXIOS-1038255] in axios@0.18.1
  introduced by app-notificaciones@git+https://gitlab.softwarelibre.gob.bo/agetic/app-notificaciones.git#5ab7a40dca2

Upgrade express-jwt@5.3.3 to express-jwt@6.0.0 to fix
X Authorization Bypass [High Severity][https://snyk.io/vuln/SNYK-JS-EXPRESSJWT-575022] in express-jwt@5.3.3
  introduced by express-jwt@5.3.3

Issues with no direct upgrade or patch:
X Arbitrary File Read [High Severity][https://snyk.io/vuln/SNYK-JS-HTMLPDF-467248] in html-pdf@2.2.0
  introduced by html-pdf@2.2.0
No upgrade or patch available
X Improper Input Validation [Medium Severity][https://snyk.io/vuln/SNYK-JS-URLPARSE-1078283] in url-parse@1.4.7
  introduced by amqplib@0.5.6 > url-parse@1.4.7
This issue was fixed in versions: 1.5.0

```

En el resultado nos dice a que version debemos actualizar para subsanar las vulnerabilidades identificadas

- Porque usar snyk.io en vez de npm audit?

	Snyk	npm
Vulnerability Sourcing		
NVD/CVE	✓	✓
Bulk Code Analysis	✓	✓
Static Code Analysis	✓	
Surfacing Community Activity (GitHub indexing)	✓	
Dedicated Research Team	✓	
Vulnerability Database		
Vulnerabilities	1,312	668
Curation	Heavy	Light
Patches	✓	
Responsiveness (time to report new vulns)	<1day	weeks-months
Community		
Responsible Disclosure	✓	✓
Open Source Vulns	✓	✓
Open Source Patches	✓	

## 5.10 Registro y monitoreo insuficientes

- Asegúrese de que todas las fallas de inicio de sesión, creación/eliminación de usuarios, fallas de control de acceso se registren para que pueda identificar la actividad sospechosa.
- Para las tareas críticas identificadas en el modelado de amenazas se debe tener logs más detallados:
  - User agent
  - IP
  - Hora y fecha con estándar UTC (tiempo universal coordinado )
  - Usuario del sistema

Ejemplo: Solicitud de datos por parte de la fiscalía:

**REQUIERE:**

1. Informe nombre completo de la personas solicitantes e identifique aparatos electrónicos por los cuales solicitaron la AUTORIZACION DE CIRCULACION VEHICULAR de las siguientes placas:
  - 22330YK EMITIDO EN FECHA 18/05/2020
  - 24330LH EMITIDO EN FECHA 18/05/2020
  - 26330BR EMITIDO EN FECHA 18/05/2020
  - 67330HL EMITIDO EN FECHA 18/05/2020
  - 14330TY EMITIDO EN FECHA 18/05/2020
  - 45330KN EMITIDO EN FECHA 18/05/2020
  - 45330RY EMITIDO EN FECHA 19/05/2020
  - 36330NE EMITIDO EN FECHA 19/05/2020
  - 45330KIR EMITIDO EN FECHA 19/05/2020
2. Informe quien es el funcionario y /o funcionarios que autorizaron o generaron los permisos de circulación de las placas mencionadas en el numeral anterior.
3. Informe identificando el IPE (servidor electrónico- computadora y/o celular) de donde fueron generados y autorizados dichos tramites de permisos de circulación.

En este ejemplo se deben tener logs tanto como del solicitante como del usuario que genera el permiso

## 6 BIBLIOGRAFÍA

- <https://blog.detectify.com/2020/11/10/common-nginx-misconfigurations/>