



UNIVERSIDAD DE GRANADA

SWAP

Práctica 4

Trabajo individual

Nuria Manzano Mata

Profesor: José Manuel Soto Hidalgo



Tiempo dedicado: 13h

ÍNDICE

1. Introducción.....	2
2. Desarrollo tareas básicas.....	2
2.1. B1. Preparación del Entorno de Trabajo.....	2
2.2. B2. Creación y Configuración de Scripts IPTABLES.....	3
2.3. B3. Implementación de Scripts IPTABLES en Docker.....	4
2.4. B4. Configuración de Docker Compose.....	6
2.5. B5. Verificación y Pruebas.....	8
2.6. Análisis propuesta IA.....	11
3. Desarrollo tareas avanzadas.....	12
3.1. A1. Definir e implementar políticas de seguridad en el balanceador de carga.....	12
3.2. A2. Configuración Avanzada de IPTABLES para DDoS.....	14
3.3. A3. Simular ataques a la granja web y configuraciones de seguridad realizadas...	15
3.4. Análisis propuesta IA.....	18
Bibliografía:.....	19

1. Introducción

Esta práctica tiene como objetivo continuar reforzando la seguridad de la infraestructura web añadiendo configuraciones de cortafuegos. Añadiremos configuraciones de cortafuegos con IPTABLES en nuestros contenedores. El **desarrollo de este documento se estructura de manera progresiva**, abordando cada tarea con un **doble propósito**: explicar detalladamente los procedimientos y soluciones implementadas, y justificar el razonamiento detrás de cada decisión. Para facilitar la comprensión, se incluyen capturas de pantalla que muestran el proceso. Debido a la extensión del documento, algunos apartados se presentan de manera concisa, mostrando los resultados más significativos o los aspectos más importantes.

El .zip entregado contiene tres carpetas, una llamada “P4” donde está todo el desarrollo de las tareas básicas, otra llamada “P4-image” que tiene la parte básica de las tareas pero usando la imagen de la práctica 3 como base en la construcción del DockerFile y otra carpeta llamada “P4-ADVANCED” donde está todo el desarrollo de las tareas avanzadas.

IMPORTANTE: Por mayor comodidad en la realización de la práctica, voy a dejar las mismas IPs, puertos y mismos nombres de contenedores para todos. Entonces, si se quiere probar la carpeta “P4” y “P4-image”, es necesario que los contenedores de una estén eliminados en Docker antes de probar la otra (al igual pasará con la “P4-ADVANCED”).

2. Desarrollo tareas básicas

2.1. B1. Preparación del Entorno de Trabajo

- Crear y preparar directorios específicos para los archivos de configuración de IPTABLES y certificados SSL previamente generados. Esto incluye:
 - Un directorio para scripts IPTABLES específicos.

El directorio para los scripts de IPTABLES se ha creado en la carpeta P4/P4-mmnuria-apache de forma que se añaden restricciones en los diferentes servicios que posteriormente creamos en Docker

```
2025-05-19 13:42:05 ✘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/ManzanoMataNuriaP4/P4/P4-mmnuria-apache
[o → ls
DockerfileApacheP4      certificados_mmnuria
P4-mmnuria-iptables-web mnnuria-apache-ssl.conf
```

2.2. B2. Creación y Configuración de Scripts IPTABLES

- Desarrollar y escribir un script tuusuariougr-iptables-web.sh que establecerá las reglas de IPTABLES en los servidores web. Este script debe:
 - Establecer políticas por defecto para rechazar todo tráfico no explícitamente permitido.
 - Permitir conexiones entrantes y salientes específicas necesarias para la operación normal de los servidores web.
 - Asegurar que las conexiones entre el balanceador de carga y los servidores web estén adecuadamente configuradas para permitir solo tráfico HTTP y HTTPS.

El script creado se encuentra en P4/P4-mmnuria-apache/mmnuria-iptables-web.sh y su contenido es el siguiente:

```
2025-05-19 13:48:05 📧 Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI  
/SWAP/ManzanoMataNuriaP4/P4/P4-mmnuria-apache/P4-mmnuria-iptables-web  
[o > cat mmnuria-iptables-web.sh  
#!/bin/bash  
  
# Políticas por defecto: bloquear todo  
iptables -P INPUT DROP  
iptables -P OUTPUT DROP  
iptables -P FORWARD DROP  
  
# Permitir tráfico de loopback  
iptables -A INPUT -i lo -j ACCEPT  
iptables -A OUTPUT -o lo -j ACCEPT  
  
# Conexiones establecidas y relacionadas (entrantes)  
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT  
# Conexiones nuevas, establecidas o relacionadas (salientes)  
iptables -A OUTPUT -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT  
  
# Permitir tráfico HTTP/HTTPS desde balanceador  
iptables -A INPUT -p tcp -s 192.168.10.50 --dport 80 -j ACCEPT  
iptables -A INPUT -p tcp -s 192.168.10.50 --dport 443 -j ACCEPT
```

Además se ha configurado el archivo P4/P4-mmnuria-apache/entrypoint.sh que permite la ejecución de las reglas IPTABLES al inicio de la creación de los diferentes contenedores Apache:

```
2025-05-20 21:56:36 Nurias-MacBook-Pro in ~/Des  
ktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/ManzanoMataNuria  
P4/P4-image/P4-mmnuria-apache/P4-mmnuria-iptables-we  
b  
o ➔ cat entrypoint.sh  
#!/bin/bash  
# Ejecuta el script de iptables  
./mmnuria-iptables-web.sh  
# Ejecuta el comando principal del contenedor  
exec "$@"
```

2.3. B3. Implementación de Scripts IPTABLES en Docker

- Integrar el script IPTABLES en la configuración de Docker de los servidores web Apache. Esto implica:
 - Modificar los Dockerfiles para incluir y ejecutar el script IPTABLES al iniciar los contenedores.
 - Asegurar que los scripts tienen los permisos adecuados para ejecutarse y modificar las reglas de IPTABLES dentro de los contenedores

Una vez que hemos añadido la configuración de IPTABLES en el script, se debe modificar el DockerfileP4 para que los servidores web Apache contengan dicha configuración. A continuación, muestro la configuración sin utilizar como base la imagen de la práctica 3, que se encuentra en el directorio “P4/P4-mmnuria-apache”:

```

2025-05-19 13:50:14 ✘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SW
AP/ManzanoMataNuriaP4/P4/P4-mmnuria-apache
[○ → cat DockerfileApacheP4
#versión 11 de debian
FROM debian:bullseye-slim
# Instalación Apache, PHP y herramientas de red
RUN apt-get update && apt-get install -y \
    apache2 \
    php \
    libapache2-mod-php \
    openssl \
    iputils-ping \
    net-tools \
    curl \
    iproute2 \
    && apt-get clean \
    && rm -rf /var/lib/apt/lists/*

# Instalar IPTABLES
RUN apt-get update && apt-get install -y iptables && apt-get clean

# Habilita el módulo Apache para PHP (necesario para que Apache procese archivos PHP)
# y SSL
RUN a2enmod php7.4 ssl

# Crear directorio para certificados
RUN mkdir -p /etc/apache2/ssl

# Copiar los certificados SSL y clave privada
COPY certificados_mmnuria/certificado_mmnuria.crt /etc/apache2/ssl/certificado_mmnuria.crt
COPY certificados_mmnuria/certificado_mmnuria.key /etc/apache2/ssl/certificado_mmnuria.key
# Copiar script de entrada y script de reglas de iptables al contenedor
COPY ./P4-mmnuria-iptables-web/entrypoint.sh /entrypoint.sh
COPY ./P4-mmnuria-iptables-web/mmnuria-iptables-web.sh /mmnuria-iptables-web.sh

# Establecer permisos seguros
RUN chmod 600 /etc/apache2/ssl/certificado_mmnuria.*

# Copiar archivo de configuración
COPY mmnuria-apache-ssl.conf /etc/apache2/sites-available/mmnuria-apache-ssl.conf

# Habilitar el sitio SSL personalizado
RUN a2ensite mmnuria-apache-ssl

# Dar permisos de ejecución a los scripts
RUN chmod +x /entrypoint.sh /mmnuria-iptables-web.sh

# Exponer puertos HTTP y HTTPS
EXPOSE 80
EXPOSE 443

# Configurar entrypoint para ejecutar primero las reglas de iptables y luego Apache
ENTRYPOINT ["/entrypoint.sh"]

# Mantener Apache en ejecución
CMD ["apache2ctl", "-D", "FOREGROUND"]

```

Lo que aparece resaltado en rosa, son las partes añadidas en el DockerfileP4 para la correcta configuración de las reglas IPTABLES y del script “entrypoint.sh” de los servidores.

El DockerfileP4 que pertenece a la carpeta “P4-image/P4-mmnuria-apache” en la que se usa la imagen de la práctica 3 previamente configurada y subida a Docker Hub es:

```

2025-05-20 21:28:25 🏠 Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SW
AP/ManzanoMataNuriaP4/P4-image/P4-mmnuria-apache
[o ➔ cat DockerfileApacheP4
# Usamos la imagen creada en la práctica 3
FROM mmnuria/mmnuria-apache-image:p3

# Instalar IPTABLES
RUN apt-get update && apt-get install -y iptables && apt-get clean

# Copiar script de entrada y script de reglas de iptables al contenedor
COPY P4-mmnuria-iptables-web/entrypoint.sh /entrypoint.sh
COPY P4-mmnuria-iptables-web/mmnuria-iptables-web.sh /mmnuria-iptables-web.sh

# Dar permisos de ejecución a los scripts
RUN chmod +x /entrypoint.sh /mmnuria-iptables-web.sh

# Configurar entrypoint para ejecutar primero las reglas de iptables y luego Apache
ENTRYPOINT ["/entrypoint.sh"]

```

Además, dejo el [enlace](#) a la imagen correspondiente en Docker hub, para que se pueda comprobar su correcta configuración.

2.4. B4. Configuración de Docker Compose

- Modificar y adaptar el archivo docker-compose.yml de la práctica anterior para incluir los cambios necesarios que permitan la ejecución de IPTABLES dentro de los contenedores de Apache y Nginx. Esto incluirá:
 - La configuración para montar los scripts y directorios necesarios dentro de los contenedores.
 - Asegurar que los contenedores tienen las capacidades de red necesarias (CAP_NET_ADMIN) para modificar IPTABLES.

El archivo docker-compose.yml no ha cambiado demasiado respecto a la práctica anterior, únicamente se debe incluir lo siguiente en cada contenedor Apache (servidores):

```

2025-05-19 14:00:12 ✘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SW
AP/ManzanoMataNuriaP4/P4
[○ → cat docker-compose.yml
version: '3'

services:
  web1:
    build:
      context: ./P4-mmnuria-apache
      dockerfile: DockerFileApacheP4
      image: mmnuria-apache-image:p4
      container_name: web1
    volumes:
      - ./web_mmnuria:/var/www/html
      - ./P4-mmnuria-certificados:/etc/apache2/ssl/
      - ./P4-mmnuria-apache/mmnuria-apache-ssl.conf:/etc/apache2/conf.d/mmnuria-apach
e-ssl.conf
    networks:
      red_web:
        ipv4_address: 192.168.10.2
      red_servicios:
        ipv4_address: 192.168.20.2
    ports:
      - "8080:80"
      - "8081:443"
    cap_add:
      - NET_ADMIN
    restart: unless-stopped

  web2:
    image: mmnuria-apache-image:p4
    container_name: web2
    volumes:
      - ./web_mmnuria:/var/www/html
      - ./P4-mmnuria-certificados:/etc/apache2/ssl/
      - ./P4-mmnuria-apache/mmnuria-apache-ssl.conf:/etc/apache2/conf.d/mmnuria-apach
e-ssl.conf
    networks:
      red_web:
        ipv4_address: 192.168.10.3
      red_servicios:
        ipv4_address: 192.168.20.3
    ports:
      - "8082:80"
      - "8083:443"
    cap_add:
      - NET_ADMIN
    restart: unless-stopped

```

Y al balanceador de carga:

```

# Balanceador
nginx-ssl:
  build:
    context: ./P4-mmnuria-nginx
    dockerfile: DockerFileNginxP4
    image: mmnuria-nginx-image:p4
    container_name: balanceador-nginx-ssl
  volumes:
    - ./P4-mmnuria-certificados:/etc/nginx/ssl/
    - ./P4-mmnuria-nginx/mmnuria-nginx-ssl.conf:/etc/nginx/nginx.conf
  depends_on:
    - web1
    - web2
    - web3
    - web4
    - web5
    - web6
    - web7
    - web8
  ports:
    - "8096:80"
    - "8097:443"
  networks:
    red_web:
      ipv4_address: 192.168.10.50

```

2.5. B5. Verificación y Pruebas

- Ejecutar y verificar el entorno configurado. Esto implica:
 - Desplegar los servicios usando Docker Compose.

Para desplegar los servicios he usado el comando docker compose up -d

[+] Running 11/11			
✓ Network p4_red_web	Started	1.6s	
✓ Network p4_red_servicios	Created	0.1s	
✓ Container web3	Created	0.1s	
✓ Container web7	Started	1.6s	
✓ Container web6	Started	3.3s	
✓ Container web2	Started	3.2s	
✓ Container web5	Started	2.2s	
✓ Container web8	Started	1.1s	
✓ Container web1	Started	1.0s	
✓ Container web4	Started	1.5s	
✓ Container balanceador-nginx-ssl	Started	3.0s	
	Started	3.5s	

Y como se puede observar en las capturas de pantalla, se han creado correctamente:

2025-05-17 20:33:06 Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/ManzanoMataNuriaP4/P4							
● ○ → docker ps	CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS	PORTS
	5dd48d12e1ff	mmnuria-nginx-image:p4		"/docker-entrypoint..."	About a minute ago	Up About a minute	0.0.0.0
	:8096->80/tcp, 0.0.0.0:8097->443/tcp	mmnuria-apache-image:p4	balanceador-nginx-ssl	"entrypoint.sh apac..."	About a minute ago	Up About a minute	0.0.0.0
	a42a875709d6	mmnuria-apache-image:p4					
	:8090->80/tcp, 0.0.0.0:8091->443/tcp	web6		"entrypoint.sh apac..."	About a minute ago	Up About a minute	0.0.0.0
	8c1533f36ece	mmnuria-apache-image:p4					
	:8084->80/tcp, 0.0.0.0:8085->443/tcp	web3		"entrypoint.sh apac..."	About a minute ago	Up About a minute	0.0.0.0
	d9b134361938	mmnuria-apache-image:p4					
	:8092->80/tcp, 0.0.0.0:8093->443/tcp	web7		"entrypoint.sh apac..."	About a minute ago	Up About a minute	0.0.0.0
	1590e180b3a1	mmnuria-apache-image:p4					
	:8080->80/tcp, 0.0.0.0:8081->443/tcp	web1		"entrypoint.sh apac..."	About a minute ago	Up About a minute	0.0.0.0
	085d267390e9	mmnuria-apache-image:p4					
	:8082->80/tcp, 0.0.0.0:8083->443/tcp	web2		"entrypoint.sh apac..."	About a minute ago	Up About a minute	0.0.0.0
	ec0bdd0077fd	mmnuria-apache-image:p4					
	:8088->80/tcp, 0.0.0.0:8089->443/tcp	web5		"entrypoint.sh apac..."	About a minute ago	Up About a minute	0.0.0.0
	28473460cc8	mmnuria-apache-image:p4					
	:8086->80/tcp, 0.0.0.0:8087->443/tcp	web4		"entrypoint.sh apac..."	About a minute ago	Up About a minute	0.0.0.0
	805da966b6b7	mmnuria-apache-image:p4					
	:8094->80/tcp, 0.0.0.0:8095->443/tcp	web8		"entrypoint.sh apac..."	About a minute ago	Up About a minute	0.0.0.0
	00a11bc1aae9	moby/buildkit:buildx-stable-1		"buildkitd"	3 days ago	Up About an hour	
				buildx buildkit loving Hodgkin0			

□	●	■	p4	-	-	-	0.03%	2 minutes ago	■	⋮	■
□	●	■	web8	805da966b6b7	mmnuria-apache-image:p4	8095:443 ↗	0%	2 minutes ago	■	⋮	■
□	●	■	web3	8c1533f36ece	mmnuria-apache-image:p4	8085:443 ↗	0%	2 minutes ago	■	⋮	■
□	●	■	web6	a42a875709d6	mmnuria-apache-image:p4	8091:443 ↗	0.01%	2 minutes ago	■	⋮	■
□	●	■	web2	085d267390e9	mmnuria-apache-image:p4	8083:443 ↗	0%	2 minutes ago	■	⋮	■
□	●	■	web4	28473460cc8	mmnuria-apache-image:p4	8087:443 ↗	0%	2 minutes ago	■	⋮	■
□	●	■	web5	ec0bdd0077fd	mmnuria-apache-image:p4	8089:443 ↗	0.01%	2 minutes ago	■	⋮	■
□	●	■	web7	d9b134361938	mmnuria-apache-image:p4	8093:443 ↗	0%	2 minutes ago	■	⋮	■
□	●	■	balanceador-nginx-s	5dd48d12e1ff	mmnuria-nginx-image:p4	8097:443 ↗	0%	2 minutes ago	■	⋮	■
□	●	■	web1	1590e180b3a1	mmnuria-apache-image:p4	8081:443 ↗	0.01%	2 minutes ago	■	⋮	■

- Verificar que las reglas de IPTABLES están activas y funcionando como se espera dentro de los contenedores.

Para este apartado he entrado dentro de la instancia del servidor web6 y he ejecutado el siguiente comando: `iptables -L -v -n`

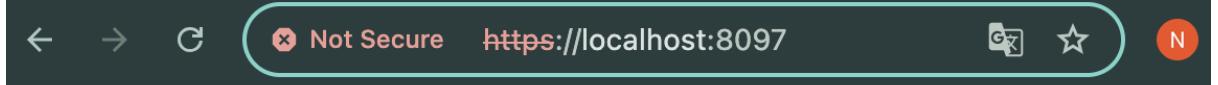
```
# iptables -L -v -n
Chain INPUT (policy DROP 0 packets, 0 bytes)
pkts bytes target     prot opt in     out      source          destination
  0    0 ACCEPT      all  --  lo      *       0.0.0.0/0        0.0.0.0/0
  0    0 ACCEPT      all  --  *       *       0.0.0.0/0        0.0.0.0/0
  0    0 ACCEPT      tcp  --  *       *       192.168.10.50   0.0.0.0/0      state RELATED,ESTABLISHED
  0    0 ACCEPT      tcp  --  *       *       192.168.10.50   0.0.0.0/0      tcp dpt:80
  0    0 ACCEPT      tcp  --  *       *       0.0.0.0/0        0.0.0.0/0      state ESTABLISHED
  0    0 ACCEPT      tcp  --  *       *       0.0.0.0/0        0.0.0.0/0      state NEW,RELATED,ESTABLISHED

Chain FORWARD (policy DROP 0 packets, 0 bytes)
pkts bytes target     prot opt in     out      source          destination

Chain OUTPUT (policy DROP 0 packets, 0 bytes)
pkts bytes target     prot opt in     out      source          destination
  0    0 ACCEPT      all  --  *       lo      0.0.0.0/0        0.0.0.0/0
  0    0 ACCEPT      all  --  *       *       0.0.0.0/0        0.0.0.0/0      state NEW,RELATED,ESTABLISHED
```

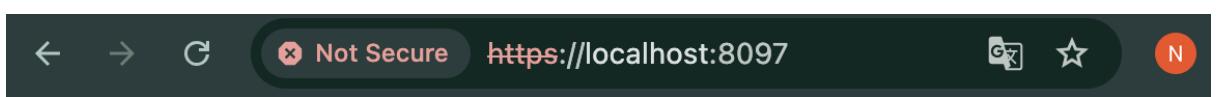
- Confirmar que el tráfico no permitido está siendo correctamente bloqueado, mientras que el tráfico legítimo fluye según lo esperado.

Para comprobar el correcto funcionamiento, comienzo verificando el balanceador de carga, que como se puede ver en las siguientes capturas de pantalla, funciona correctamente, de hecho cuando se hace una petición con http, se redirige correctamente a https:



Práctica 4 SWAP - Nuria Manzano Mata

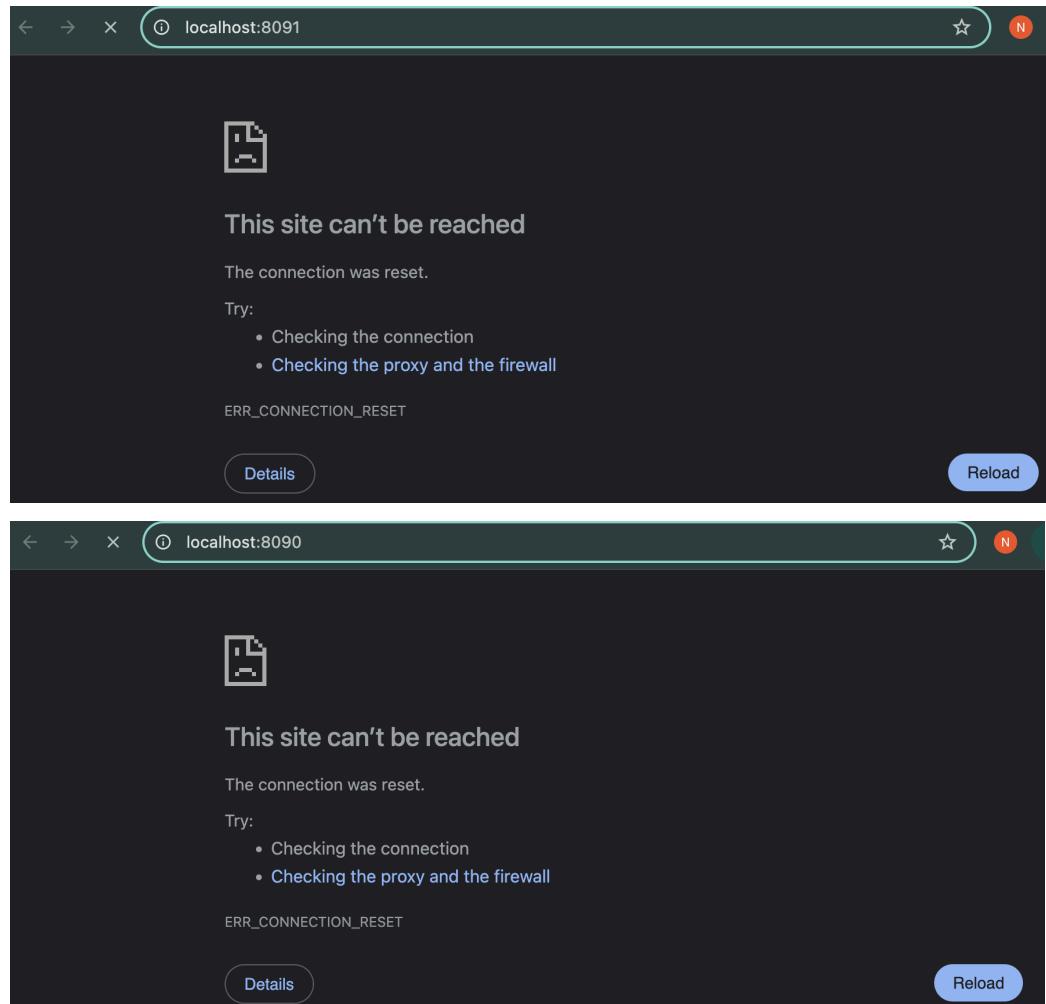
La dirección IP del servidor es: 192.168.10.2



Práctica 4 SWAP - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.10.3

La siguiente comprobación es que cuando hacemos peticiones directamente a los servidores de forma independiente, éstos no deben responder:



Concluyo este primer apartado con la correcta consecución del mismo.

2.6. Análisis propuesta IA

Con este apartado voy a tratar de **analizar los resultados propuestos por la IA** para este primer apartado de tareas básicas. He preguntado a la IA escribiendo directamente el enunciado de los diferentes apartados para que me mostrara paso a paso lo que debe de incluir cada uno, [pulsa aquí para acceder al chat](#)

De forma resumida, la IA me ha proporcionado soluciones muy básicas en las que me he apoyado para generar las más:

- **Tarea B2 y B3:** La IA me ha ofrecido sugerencias aceptables, pero en este caso me han resultado mucho más útiles las indicaciones del profesor. La IA tiende a añadir reglas o información al DockerFile que no son necesarias para la práctica, por lo que se pueden omitir sin problema. Además, aspectos clave como la creación del DockerFile a partir de la imagen generada en la práctica 3 no fueron mencionados, y tuve que

- investigarlos por mi cuenta en la documentación oficial de Docker. No fue especialmente complicado, pero sin conocerlo de antemano se hizo algo tedioso al principio. Aun así, la IA puede ser de ayuda para comenzar con lo básico, pero cuando se trata de definir reglas más específicas o crear DockerFiles más personalizados, su rendimiento no es del todo adecuado.
- Tarea B4 y B5: En este caso, las sugerencias han sido algo incompletas, aunque me sirvieron como punto de partida para hacerme una idea de cómo modificar el docker-compose.yml. En cuanto a la parte de comprobación, no aportó nada nuevo respecto a lo que ya sabía, por lo que podría considerarse simplemente aceptable.

3. Desarrollo tareas avanzadas

Para resolver este apartado, he creado una carpeta nueva llamada "P4-ADVANCED", en la que **cada sección tiene su propia subcarpeta/archivo** para facilitar la verificación del funcionamiento y mantener todo organizado. En cada apartado se especifica el nombre de la subcarpeta/archivo creado y el paso a paso seguido para su resolución.

Como he hecho en otras prácticas, para comprobar el correcto funcionamiento de esta, es necesario ejecutar nuevos contenedores que se encuentran en el directorio mencionado en el párrafo anterior. Dichos contenedores incluyen todas las modificaciones e implementaciones nuevas que se requieren para las siguientes tareas.

IMPORTANTE: Por una mayor comodidad en la realización de la práctica, voy a dejar las mismas IPs, puertos y mismos nombres de contenedores para todos. Entonces, si se quiere probar este apartado, es necesario que los contenedores del apartado anterior estén eliminados en Docker, así puede servir la página sin problemas.

3.1. A1. Definir e implementar políticas de seguridad en el balanceador de carga

- Definir e implementar políticas de seguridad en el balanceador de carga. Podría incluir, además de denegación implícita:
 - Limitar el número de conexiones simultáneas.
 - Bloquear escaneo de puertos.
 - Usar módulo string para mitigar ataques de inyección SQL o XSS a través de las peticiones HTTP.
 - Etc.

P4-ADVANCED/P4-mmnuria-nginx/P4-mmnuria-iptables-web/mmnuria-iptables-web.sh este es el archivo donde se puede encontrar toda la configuración de este apartado

```
2025-05-20 22:06:15 ✎ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/ManzanoMataNuriaP4/P4-ADVANCED/P4-mmnuria-nginx/P4-mmnuria-iptables-web
o ➔ cat mmnuria-iptables-web.sh
#!/bin/bash

# Política por defecto segura
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

# Permitir tráfico local y conexiones establecidas
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

# Permitir HTTP/HTTPS
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp --dport 443 -j ACCEPT

# Limitar conexiones simultáneas por IP
iptables -A INPUT -p tcp --dport 80 -m connlimit --connlimit-above 20 -j REJECT
iptables -A INPUT -p tcp --dport 443 -m connlimit --connlimit-above 20 -j REJECT

# Bloquear escaneo de puertos
iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP
iptables -A INPUT -p tcp --tcp-flags ALL ALL -j DROP
iptables -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
iptables -A INPUT -p tcp --tcp-flags SYN,RST SYN,RST -j DROP

# Bloqueo básico de SQL Injection y XSS
iptables -A INPUT -p tcp --dport 80 -m string --string "SELECT" --algo bm --icase -j DROP
iptables -A INPUT -p tcp --dport 80 -m string --string "<script>" --algo bm --icase -j DROP
```

P4-ADVANCED/P4-mmnuria-nginx/P4-mmnuria-iptables-web/entrypoint.sh este es el archivo que ejecutará las reglas IPTABLES configuradas anteriormente.

```
2025-05-17 20:38:09 ✎ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/ManzanoMataNuriaP4/P4-ADVANCED/P4-mmnuria-nginx/P4-mmnuria-iptables-web
o ➔ cat entrypoint.sh
#!/bin/bash
bash /mmnuria-iptables-web.sh

nginx -g "daemon off;"
```

Además, es necesaria la modificación del DockerfileP4 para que el balanecedor de carga tenga acceso a estos archivos y se puedan ejecutar. A continuación, muestro una captura de pantalla señalando la configuración añadida para este apartado:

```

2025-05-17 20:39:39 ✘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/
SWAP/ManzanoMataNuriaP4/P4-ADVANCED/P4-mmnuria-nginx
[O ➔ cat DockerfileNginxP4
FROM nginx:latest

# Crear directorio para certificados SSL
RUN mkdir -p /etc/nginx/ssl

# Copiar certificados SSL al contenedor
COPY certificados_mmnuria/certificado_mmnuria.crt /etc/nginx/ssl/certificado_mmnuri
a.crt
COPY certificados_mmnuria/certificado_mmnuria.key /etc/nginx/ssl/certificado_mmnuri
a.key

# Copiar archivo de configuración SSL
COPY mmnuria-nginx-ssl.conf /etc/nginx/nginx.conf

# Copiar script de reglas de iptables
COPY P4-mmnuria-iptables-web/mmnuria-iptables-web.sh /mmnuria-iptables-web.sh
RUN chmod +x /mmnuria-iptables-web.sh

# Copiar script de entrada
COPY P4-mmnuria-iptables-web/entrypoint.sh /entrypoint.sh
RUN chmod +x /entrypoint.sh

# Exponer el puerto HTTP y HTTPS
EXPOSE 80
EXPOSE 443

ENTRYPOINT ["/entrypoint.sh"]

```

3.2. A2. Configuración Avanzada de IPTABLES para DDoS

- Implementar reglas avanzadas en IPTABLES para mitigar ataques de Denegación de Servicio Distribuido (DDoS). Esto podría incluir:
 - Limitación de la tasa de conexiones nuevas por IP para evitar la saturación de los recursos del servidor.
 - Uso de módulos como recent para detectar y bloquear rápidamente el tráfico anómalo y prevenir inundaciones de IPs.
 - Configuración de umbrales y reglas específicas que identifiquen patrones de tráfico asociados a ataques comunes de DDoS.
 - Protección Contra Ataques de Fragmentación.
 - Etc.

Para la implementación de este apartado, he añadido al archivo P4-ADVANCED/P4-mmnuria-nginx/P4-mmnuria-iptables-web/mmnuria-iptables-web.sh más reglas avanzadas en IPTABLES para controlar los ataques de DDos, estas aparecen marcadas en rosa:

```

2025-05-20 22:06:15 ✘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/ManzanoMataNuri
aP4/P4-ADVANCED/P4-mmnuria-nginx/P4-mmnuria-iptables-web
|o ✘ cat mmnuria-iptables-web.sh
#!/bin/bash

# Política por defecto segura
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

# Permitir tráfico local y conexiones establecidas
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

# Permitir HTTP/HTTPS
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp --dport 443 -j ACCEPT

# Limitar conexiones simultáneas por IP
iptables -A INPUT -p tcp --dport 80 -m connlimit --connlimit-above 20 -j REJECT
iptables -A INPUT -p tcp --dport 443 -m connlimit --connlimit-above 20 -j REJECT

# Bloquear escaneo de puertos
iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP
iptables -A INPUT -p tcp --tcp-flags ALL ALL -j DROP
iptables -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
iptables -A INPUT -p tcp --tcp-flags SYN,RST SYN,RST -j DROP

# Bloqueo básico de SQL Injection y XSS
iptables -A INPUT -p tcp --dport 80 -m string --string "SELECT" --algo bm --icase -j DROP
iptables -A INPUT -p tcp --dport 80 -m string --string "<script>" --algo bm --icase -j DROP

# Protección contra fragmentación
iptables -A INPUT -f -j DROP

# Prevención de SYN flood
iptables -A INPUT -p tcp --syn -m limit --limit 10/s --limit-burst 20 -j ACCEPT

# Detección de repetición excesiva por IP (recent module)
iptables -A INPUT -p tcp --dport 80 -m recent --name BAD --set
iptables -A INPUT -p tcp --dport 80 -m recent --name BAD --update --seconds 60 --hitcount 30 -j DROP

```

3.3. A3. Simular ataques a la granja web y configuraciones de seguridad realizadas

- Simular un ataque DDoS para probar la configuración de seguridad de la granja web.
- Simular un ataque de inyección SQL o XSS
- Simular otros ataques.

Para la correcta simulación de ataques a la granja web, voy a usar un script llamado “ataques.sh” que se encuentra en la carpeta “P4-ADVANCED”, comienzo creándolo y dándole permisos de ejecución:

```

2025-05-21 16:50:02 ✘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/ManzanoMataNuriaP4/P4-ADVANCED
|o ➔ chmod +x ataques.sh

2025-05-21 16:52:14 ✘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/ManzanoMataNuriaP4/P4-ADVANCED
|o ➔ cat ataques.sh
#!/bin/bash

TARGET_HOST="localhost"
TARGET_PORT="8097"
FULL_URL="https://{$TARGET_HOST}:{$TARGET_PORT}"

echo "===== ATAQUES AUTOMATIZADOS A LA GRANJA WEB ====="
echo "Objetivo: $FULL_URL"
echo "=====\\n"

# 1. Ataque DDoS con nping (simulación TCP flood)
echo " Simulando DDoS con nping..."
sudo nping --tcp -p $TARGET_PORT --rate 500 -c 2000 $TARGET_HOST
echo "***** Simulación de DDoS finalizada. *****"

# 2. Inyección SQL con sqlmap
echo " Inyección SQL con sqlmap..."
sqlmap -u "${FULL_URL}?q=test" --batch --random-agent --level=2 --risk=1 --ignore-code=404
echo "***** Inyección SQL simulada. *****"

# 3. XSS básico con curl
echo " Ataque XSS básico con curl..."
curl -k "${FULL_URL}?xss=<script>alert(1)</script>"
echo -e "***** Petición XSS enviada. Revisar navegador para ver si se refleja. *****"

# 4. Escaneo de vulnerabilidades con nikto
echo "Escaneo de vulnerabilidades web con nikto..."
nikto -h $FULL_URL
echo "***** Escaneo con nikto finalizado. *****"

# 5. Escaneo de puertos con nmap
echo " Escaneo de puertos con nmap..."
nmap -sV $TARGET_HOST -p $TARGET_PORT
echo "***** Escaneo de puertos finalizado. *****"

echo "***** Todos los ataques fueron ejecutados. *****"

```

Los comandos usados han sido los que se pueden ver en la captura de pantalla especialmente, los he elegido debido a que mi sistema operativo es MacOs.

A continuación muestro el correcto funcionamiento de los ataques al servidor, a través de la ejecución de dicho script:

```

SENT (4.1684s) TCP 127.0.0.1:41491 > 127.0.0.1:8097 S ttl=64 id=7867 ipien=40 seq=855147905 win=1480
SENT (4.1784s) TCP 127.0.0.1:41491 > 127.0.0.1:8097 S ttl=64 id=7867 ipien=40 seq=855147905 win=1480
SENT (4.1725s) TCP 127.0.0.1:41491 > 127.0.0.1:8097 S ttl=64 id=7867 ipien=40 seq=855147905 win=1480
Max rtt: 1.918ms | Min rtt: 0.022ms | Avg rtt: 0.065ms
Raw packets sent: 2000 (88.000KB) | Rcvd: 1667 (73.346KB) | Lost: 333 (16.65%)
Nping done: 1 IP address pinged in 5.17 seconds
***** Simulación de DDoS finalizada. *****
Inyección SQL con sqlmap...
[!] 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 @ 17:23:36 /2025-05-21/
[17:23:36] [INFO] fetched random HTTP User-Agent header value 'Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.14) Gecko/20080428 Firefox/2.0.0.14' from file '/usr/local/Cellar/sqlmap/1.9.5/libexec/data/txt/user-agents.txt'
[17:23:37] [INFO] testing connection to the target URL
[17:23:37] [INFO] testing if the target URL content is stable
[17:23:37] [WARNING] target URL content is not stable (i.e. content differs). sqlmap will base the page comparison on a sequence matcher. If no dynamic nor injectable parameters are detected, or in case of junk results, refer to user's manual paragraph 'Page comparison' how do you want to proceed? ([c]ontinue/(s)tring/(\r)egez/(\q)uit) c
[17:23:37] [INFO] testing if GET parameter 'id' is dynamic
[17:23:37] [WARNING] GET parameter 'id' does not appear to be dynamic
[17:23:37] [INFO] heuristic (basic) test shows that GET parameter 'q' might not be injectable
[17:23:37] [INFO] testing for SQL injection on GET parameter 'q'
[17:23:37] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[17:23:37] [INFO] GET parameter 'q' appears to be 'AND boolean-based blind - WHERE or HAVING clause' injectable
[17:23:37] [INFO] heuristic (extended) test shows that the back-end DBMS could be 'Microsoft SQL Server'
it looks like the back-end DBMS is 'Microsoft SQL Server'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
For the remaining tests, do you want to skip Microsoft SQL Server extended payload level (2) and risk (i) values? [Y/n] Y
[17:23:37] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[17:23:37] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[17:23:37] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (CONVERT)'
[17:23:37] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (CONCAT)'
[17:23:37] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (CONCAT)'
[17:23:37] [INFO] testing 'Microsoft SQL Server/Sybase error-based - Parameter replace'
[17:23:37] [INFO] testing 'Microsoft SQL Server/Sybase error-based - Parameter replace (integer column)'
[17:23:37] [INFO] testing 'Microsoft SQL Server/Sybase error-based - Stacking (EXEC)'
[17:23:37] [INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[17:23:38] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[17:23:38] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (DECLARE - comment)'
[17:23:38] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries'
[17:23:38] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[17:23:38] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF - comment)'
[17:23:38] [INFO] testing 'Microsoft SQL Server/Sybase AND time-based blind (heavy query)'
[17:23:38] [INFO] testing 'Microsoft SQL Server/Sybase OR time-based blind (heavy query)'

some kind of protection mechanism involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '--tamper=space2comment')
[17:23:41] [WARNING] HTTP error codes detected during run:
503 (Service Unavailable) - 267 times

[*] ending @ 17:23:41 /2025-05-21/
***** Inyección SQL simulada. *****
Ataque XSS básico con curl...
<!DOCTYPE html>
<html>
<head>
    <title>Práctica 4 SWAP - Nuria Manzano Mata</title>
</head>
<body>
    <h1>Práctica 4 SWAP - Nuria Manzano Mata</h1>
    <p>La dirección IP del servidor es: 192.168.10.7</p>
</body>
</html>***** Petición XSS enviada. Revisar navegador para ver si se refleja. *****
Escaneo de vulnerabilidades web con nikto...
- Nikto v2.5.0

+ Target IP: 127.0.0.1
+ Target Hostname: localhost
+ Target Port: 8097

+ SSL Info: Subject: /C=ES/ST=Granada/L=Granada/O=SWAP/OU=Pr\xC3\x83\xC2\xA1ctica 3/CN=Nuria Manzano Mata/emailAddress=mnnuria@correo.ugr.es
Ciphers: AEAD-AES256-GCM-SHA384
Issuer: /C=ES/ST=Granada/L=Granada/O=SWAP/OU=Pr\xC3\x83\xC2\xA1ctica 3/CN=Nuria Manzano Mata/emailAddress=mnnuria@correo.ugr.es
+ Start Time: 2025-05-21 17:23:41 (GMT2)

+ Server: nginx/1.27.5
+: The site uses TLS and the Strict-Transport-Security HTTP header is not defined. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
+/UpnP234.asp: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsec-works.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Hostname 'localhost' does not match certificate's names: Nuria. See: https://cwe.mitre.org/data/definitions/297.html
+ 7849 requests: 0 errors(s) and 3 item(s) reported on remote host
+ End Time: 2025-05-21 17:24:44 (GMT2) (63 seconds)

+ 1 host(s) tested
***** Escaneo con nikto finalizado. *****
Escaneo de puertos con nmap...
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-21 17:24 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00017s latency).
Other addresses for localhost (not scanned): ::1

PORT      STATE SERVICE VERSION
8097/tcp  open  ssl/http nginx 1.27.5

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.40 seconds
***** Escaneo de puertos finalizado. *****
***** Todos los ataques fueron ejecutados. *****
```

Como se puede observar en los resultados obtenidos, todos los ataques fueron ejecutados, y se extraen las siguientes conclusiones clave:

1. La simulación de DDoS generó un tráfico TCP elevado entre “127.0.0.1:41491” y “127.0.0.1:8097”, con un total de 2000 paquetes enviados. Se detectó una pérdida del 16.65% de paquetes, lo que indica cierto nivel de saturación del servidor bajo esta carga. Las latencias se mantuvieron bajas (RTT promedio de 0.065 ms), sugiriendo que aunque hubo pérdida, el servidor aún respondió con rapidez.

2. La inyección SQL se centró en el parámetro “q” de una URL tipo “GET”. Aunque inicialmente parecía haber una inyección tipo boolean-based blind, al final se determinó que se trataba de un falso positivo o una inyección no explotable. No se detectaron parámetros realmente vulnerables, y se registraron múltiples respuestas HTTP 503 (Service Unavailable), lo cual puede ser indicativo de mecanismos de defensa activos, como un WAF o protección contra escaneos automatizados.
3. Se intentó injectar un payload XSS clásico, pero el servidor respondió con una página HTML estática sin reflejar el contenido inyectado. No se observaron indicios de ejecución del script, lo que sugiere que la aplicación no refleja directamente los parámetros enviados. Por tanto, no se identificaron vulnerabilidades XSS explotables en este contexto.
4. Nikto identificó varias debilidades en la configuración del servidor:
 - Ausencia de cabeceras de seguridad importantes como “Strict-Transport-Security” y “X-Content-Type-Options”.
 - Certificado SSL autofirmado con un CN que no coincide con el hostname (“localhost”).
 - Presencia de al menos un archivo “.asp+” no protegido adecuadamente.
5. Nmap confirmó que el único puerto abierto es el 8097/tcp, que corre un servidor nginx 1.27.5 sobre HTTPS. Esto valida que el entorno está limitado a un único servicio activo y cifrado, lo cual es positivo.

3.4. Análisis propuesta IA

En este apartado trato de analizar los resultados propuestos por la IA de este segundo apartado para las tareas avanzadas. He preguntado a la IA escribiendo directamente el enunciado de las diferentes tareas para que me mostrara paso a paso lo que debe de incluir cada una, [pulsa aquí para acceder al chat](#)

De forma resumida, la IA me ha proporcionado soluciones bastante útiles:

- Tarea A1 y A2: Las sugerencias de la IA me han resultado bastante útiles. Es cierto que no he utilizado todos los comandos que me propuso, ya que algunos no eran necesarios para cumplir con lo que pedía cada apartado. Por eso, he decidido ceñirme únicamente a responder lo que se solicitaba en cada punto, sin añadir acciones extra que no aportaran directamente al objetivo.

- Tarea A3: En este caso, las recomendaciones de la IA no me han sido tan útiles. Primero, porque me propuso ejecutar comandos de forma individual, mientras que yo preferí organizarlo todo en un script para que el proceso fuese más limpio y automatizado. Segundo, me sugirió realizar los ataques desde dentro del contenedor, pero en mi caso opté por lanzarlos desde mi propio equipo (macOS), simulando así un atacante externo. Además, algunos de los comandos recomendados no estaban disponibles o no funcionaban directamente en macOS, lo que me obligó a investigar alternativas compatibles. En resumen, aunque para esta última parte la ayuda de la IA no fue tan efectiva, en las tareas anteriores sí que me sirvió de apoyo.

Bibliografía:

- Lyon, G. F. (2009). *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. Insecure.Com LLC.
- Mozilla Developer Network. (s.f.). *Cross-site scripting (XSS)*.
https://developer.mozilla.org/en-US/docs/Glossary/Cross-site_scripting
- Mozilla Developer Network. (s.f.). *X-Frame-Options*.
<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options>
- Mozilla Developer Network. (s.f.). *Strict-Transport-Security*.
<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security>
- Mozilla Developer Network. (s.f.). *X-Content-Type-Options*.
<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options>
- Nmap Project. (s.f.). *Nping man page*. <https://nmap.org/book/nping-man.html>
- Nmap Project. (s.f.). *Service Version Detection*. <https://nmap.org/book/vscan.html>
- OWASP Foundation. (s.f.). *SQL Injection*.
https://owasp.org/www-community/attacks/SQL_Injection
- OWASP Foundation. (s.f.). *Cross Site Scripting (XSS)*.
<https://owasp.org/www-community/attacks/xss/>
- Stampar, B. (s.f.). *sqlmap: automatic SQL injection and database takeover tool*.
<https://sqlmap.org>
- Sullo, C. (s.f.). *Nikto Web Scanner*. <https://cirt.net/Nikto2>
- breachattack.com. (2013). *BREACH: Reviving the CRIME Attack*.
<http://breachattack.com/>