Módulo 7 | Unidad 4

Desafío: Creación y manipulación de paquetes

Elaborado por Esteban Aranda, para el módulo Testeo de Seguridad en Redes de Datos

Bootcamp Especialidad Seguridad en Redes de Datos, G1 – BOTIC-SOFOF-24-28-05-0006

04 de agosto de 2025

Contexto

Pera el desarrollo del presente desafío se realizaron las pruebas sobre la propia máquina Kali Linux (solo una máquina virtual activa), y se configuró un servidor sencillo (con netcat) que sirve de destino para las distintas pruebas.

Hubo una prueba que no se desarrolló debido a las limitantes de hardware (que dificultan el poder utilizar dos máquinas virtuales al mismo tiempo): ARP Spoofing. Sin embargo, se realiza un desarrollo teórico sobre el tema.

En la siguiente imagen se puede ver el funcionamiento del servidor web, específicamente en los resultados mostrados por Wireshark al realizar el envío de un paquete SYN utilizando hPing3. El objetivo era probar el funcionamiento del servidor para las siguientes pruebas del desafío. El comando utilizado fue:

```
$ sudo hping3 -S -p 8080 -c 1 localhost
```

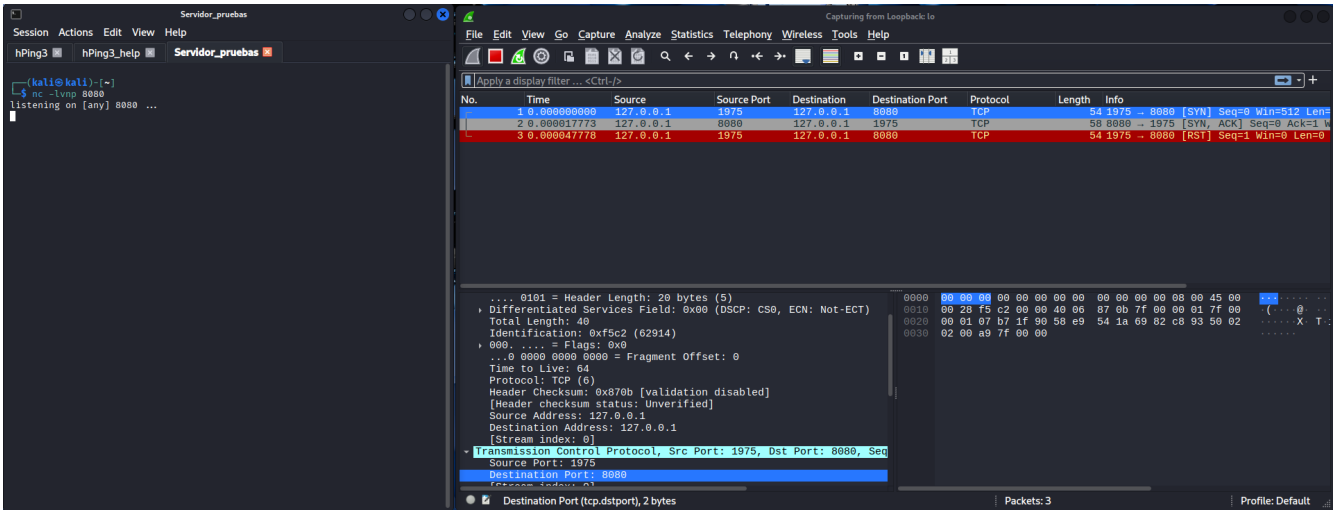

Figura 1: Prueba de funcionamiento del servidor de pruebas.

Desarrollo de las actividades

1. Utilizar la herramienta hPing3 para generar tráfico simulado y modificar parámetros como IP de origen, TTL y flags TCP.

1.1 Paquete con IP de origen modificada

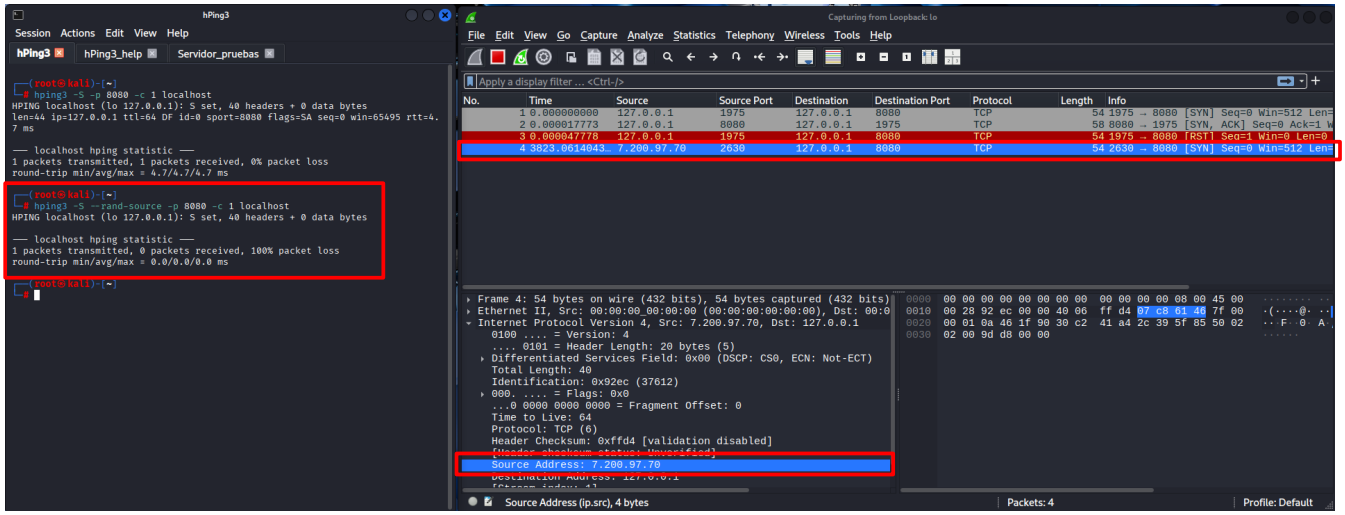

Figura 2: Paquete creado con hPing3 donde se modificó la IP de origen.

Dado que la IP de origen ha sido modificada (al utilizar *--rand-source*, el cual genera una dirección IPv4 de origen distinta a la real), la respuesta al paquete SYN nunca llegará, ya que el servidor le responde a la IP falsa (en este caso, la IP generada fue: 7.200.97.70).

1.2 Paquete con TTL modificada

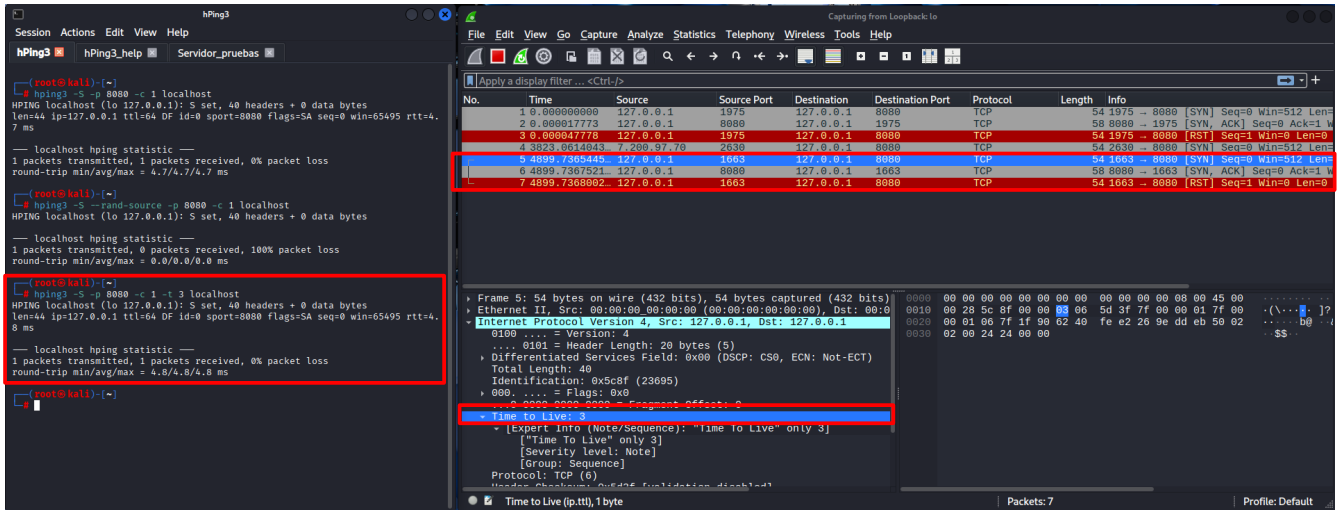

Figura 3: Paquete creado con hPing3 donde se modificó el TTL (Time-To-Live).

Se crea un paquete con valor de TTL de 3 (3 saltos antes que un router descarte el paquete y devuelva un mensaje ICMP *Time Exceeded* al remitente).

1.3 Paquete con flag TCP modificada

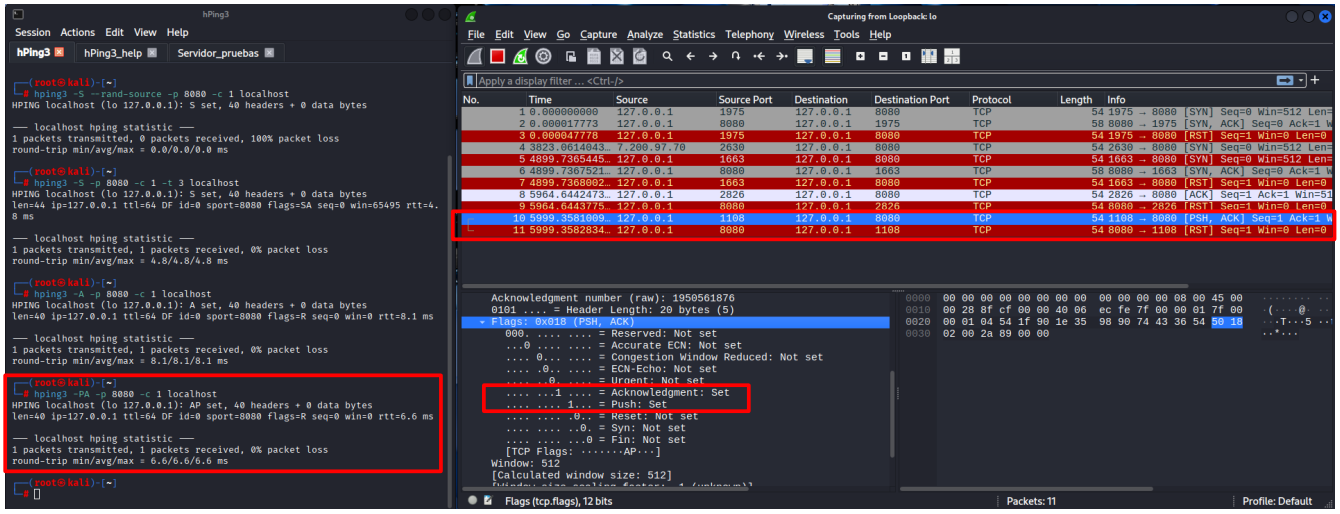

Figura 4: Paquete creado con hPing3 donde se modificó la flag TCP.

Se envía al servidor un paquete que incluye un PUSH y un ACK (opción *-PA*), donde se puede apreciar en Wireshark ambas flags. El servidor responde con un RST (el servidor que levanté) ya que el Kernel de mi máquina interpreta esta conexión como inexistente y termina la comunicación (esto ocurre porque el paquete fue enviado desde hPing3, el cual envía un paquete *raw* que no intenta establecer una conexión real [a través de *sockets TCP* del kernel], y por esto éste no sabe de esta conexión; no aparece en su tabla de conexiones TCP).

2. Crear paquetes personalizados con Scapy para pruebas de red (por ejemplo, paquetes ICMP, TCP con payload, entre otros).

2.1 Paquete ICMP (SYN)

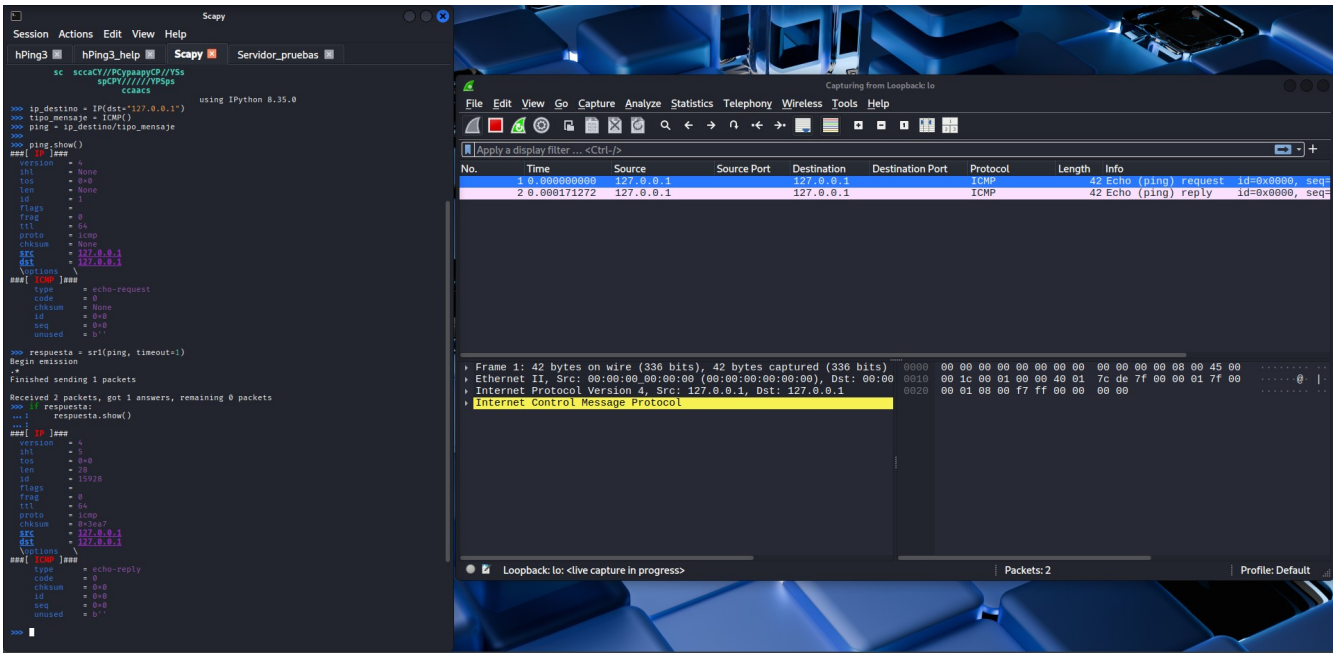

Figura 5: Paquete SYN creado con Scapy.

2.2 Paquete TCP con *payload*

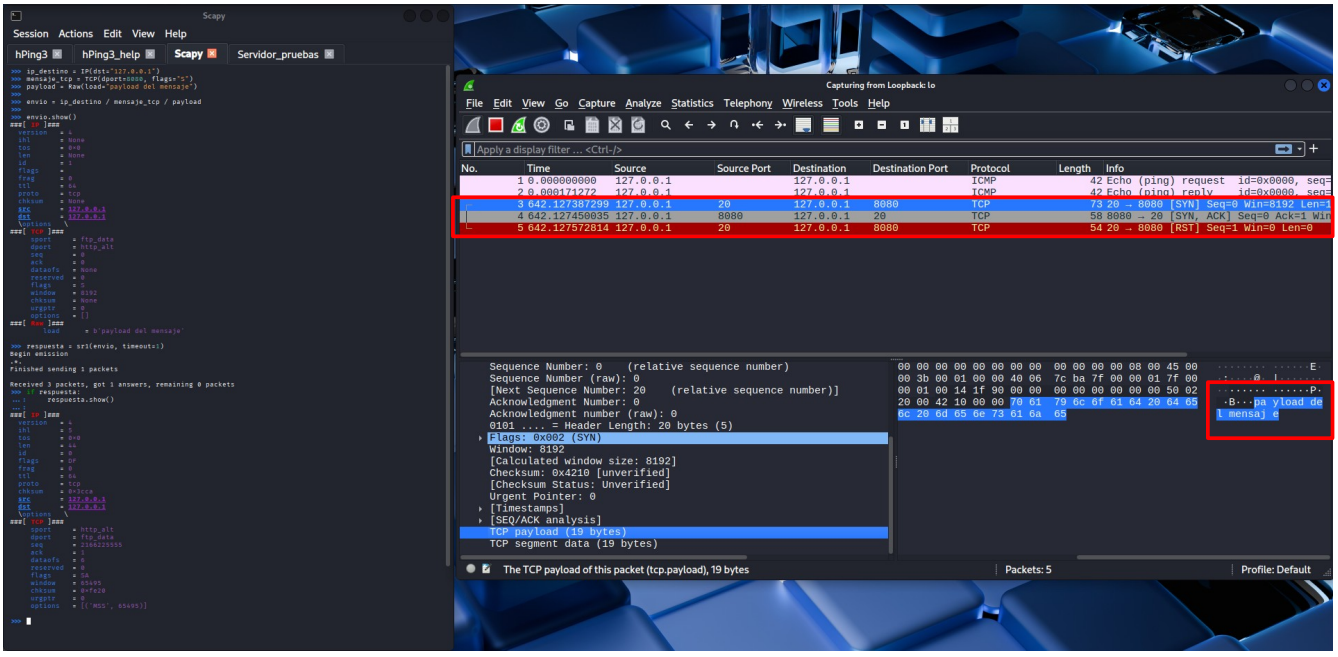

Figura 6: Paquete TCP con payload personalizado, creado con Scapy.

2.3 Paquete TCP con *payload* y *flags* modificadas

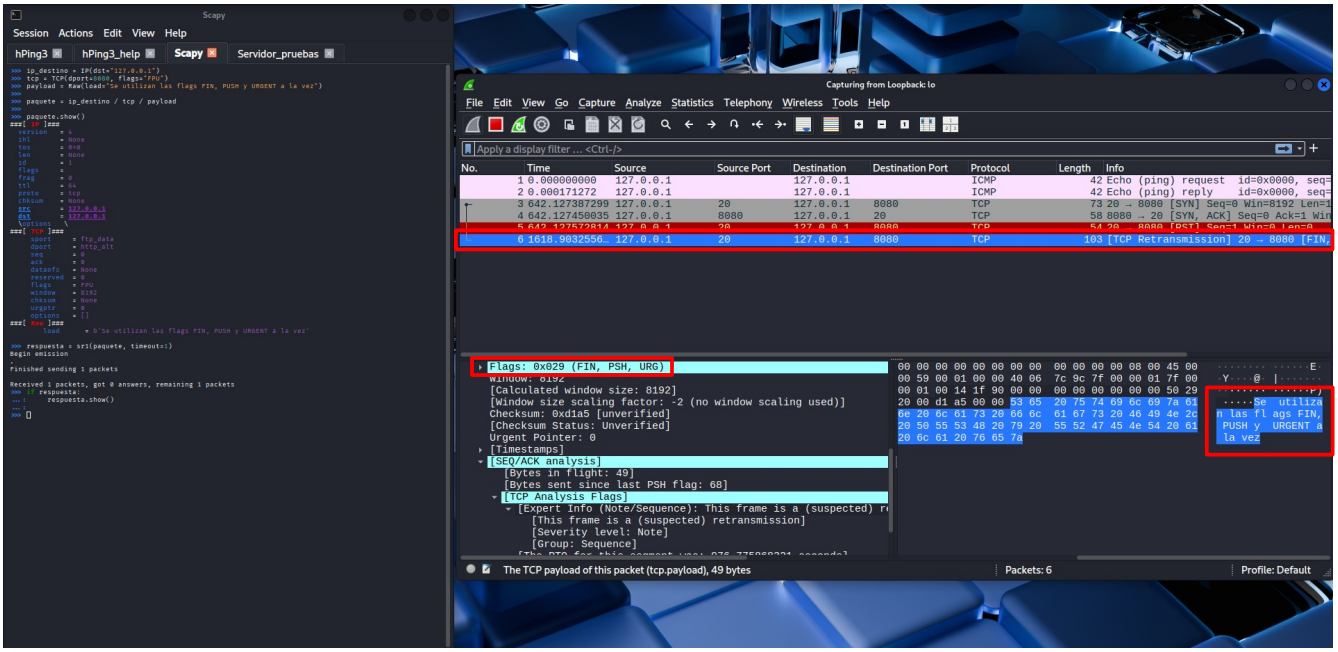

Figura 7: Paquete TCP con payload personalizado y flags modificadas, creado con Scapy.

En este caso, se envía un paquete con las *flags* modificadas; se configura para que envié las *flags* FIN, PUSH y URGENT. Como no son *flags* comunes para entablar un *3way hadshake*, el servidor no responde al paquete ya que no sabe qué responder (no hay un SYN que inicie el apretón de manos), por lo que rechaza el paquete. Por ello no obtenemos una respuesta por parte del servidor.

3. Ejecutar un pequeño ataque con Syn Flood con hping3 o ARP Spoofing con arpspoof.

3.1 Ataque *Syn Flood* controlado

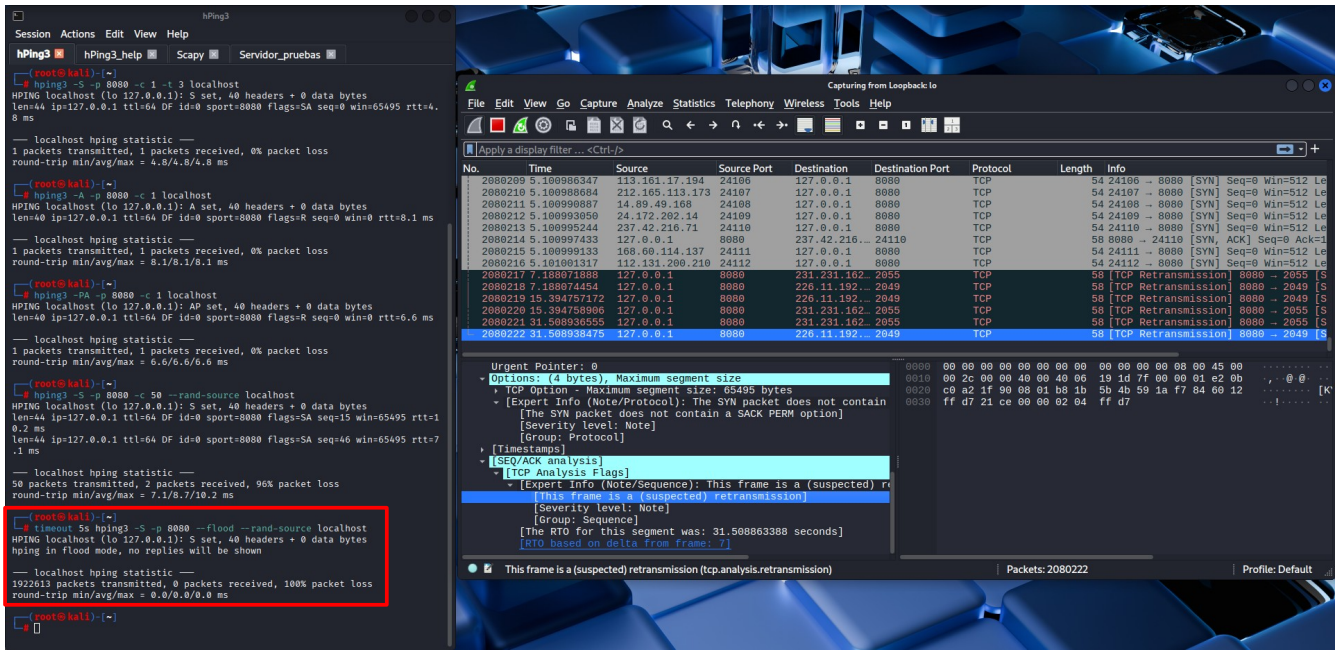

Figura 8: Ataque SYN Flood controlado ejecutado contra el servidor de pruebas.

Se envía un ataque *Syn Flood* controlado (al utilizar *timeout 5s*, que limita la ejecución del comando a 5 segundos) para evitar un eventual colapso del laboratorio. Sin embargo, cinco segundos bastaron para generar más de dos millones de intercambios (de registros en Wireshark), entre SYN enviados por el ataque y respuestas SYN/ACK enviadas por el servidor. Además, se observa una cantidad limitada de mensajes tipo "[TCP Retransmission]", los cuales son generados por el servidor al detectar que la respuesta SYN/ACK ya ha sido enviada previamente a la misma solicitud SYN que está recibiendo (primero recibe un SYN y responde un SYN/ACK, pero vuelve a recibir un SYN del mismo remitente y no un ACK, que sería el siguiente paso en un *3way handshake*). Por lo tanto, el servidor vuelve a enviarle un SYN/ACK al remitente para que responda con el correspondiente ACK y, así, entablar la comunicación.

3.2 Ataque *arpspoof* controlado

Esta prueba no se realiza debido a falta de recursos para tener dos máquinas virtuales en ejecución al mismo tiempo. Para un ataque arpspoof se necesita conocer la dirección MAC de un dispositivo de la red local, el que será suplantado, y la IP del dispositivo víctima del ataque. Con esta información, el dispositivo del atacante suplantará la dirección MAC (generalmente del dispositivo *Gateway*; el router) y enviará mensajes *ARP Replies* al dispositivo víctima, con la intención que éste actualice su tabla ARP con la dirección IP del atacante (eliminando la del anterior). Así, todo el tráfico del dispositivo víctima irá dirigido a la máquina del atacante.

4. Capturas y analizar el tráfico del ataque con Wireshark, para aplicar reglas de filtrado con iptables para mitigar ataque.

Contexto

Para probar el funcionamiento de reglas en iptables para controlar un ataque de Syn Flood se creó el siguiente ataque en hPing3:

```
$ sudo timeout 2s hping3 -S -a 172.10.10.100 -p 8080 -c 1 --flood localhost
```

Antes de comenzar el ataque, se definieron las siguientes reglas en iptables con la finalidad de minimizar el impacto de un ataque Syn Flood hacia el servidor de pruebas:

```
$ sudo iptables -A INPUT -p tcp --syn --dport 8080 -m limit --limit 10/s --limit-burst 20 -j ACCEPT

$ sudo iptables -A INPUT -p tcp --syn --dport 8080 -j DROP
```

La primera regla tiene la función de evaluar los paquetes entrantes al servidor (puerto 8080) que sean del tipo tcp, específicamente los paquetes SYN. Inicialmente deja pasar 20 paquetes SYN consecutivos en ráfaga, pero luego limita el tráfico a 10 paquetes por segundo.

La segunda regla se aplicará para todo tráfico que no cumpla las condiciones de la primera, y ésta *dropea* todo el tráfico SYN dirigido al puerto 8080.

Ejecución del ataque y comprobación de reglas *iptables*

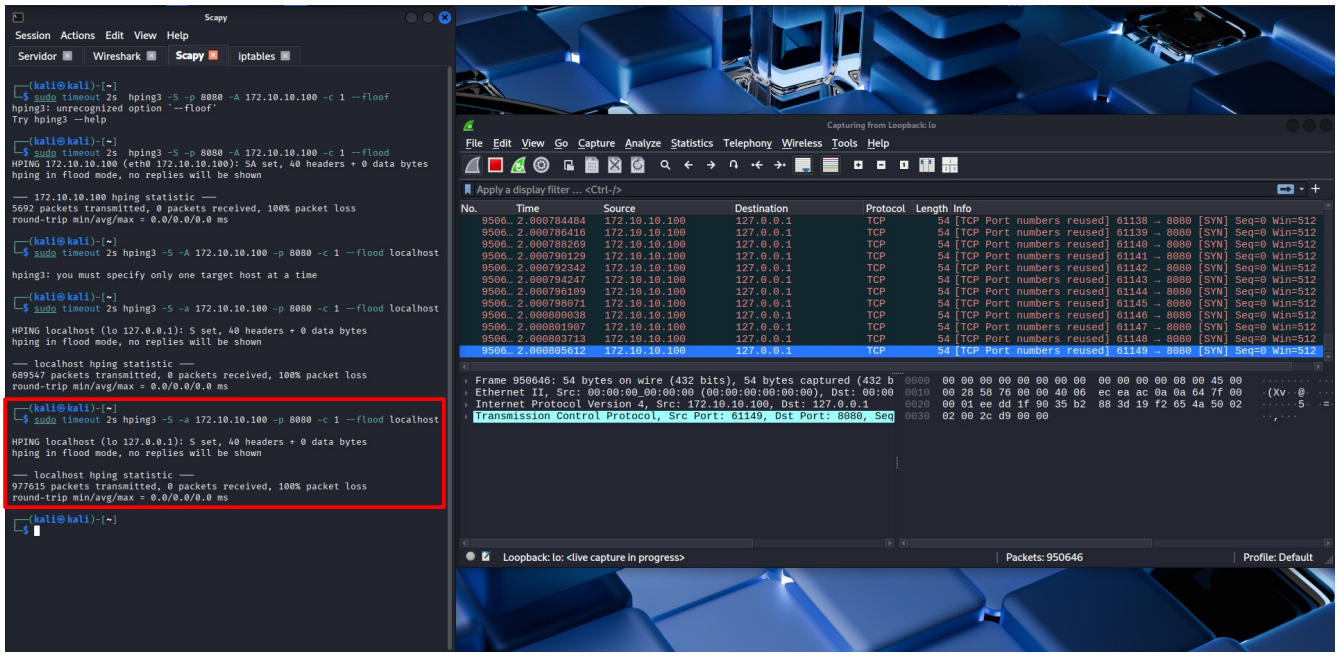

Figura 9: Ataque SYN Flood ejecutado contra el servidor de pruebas.

Luego de ejecutar el ataque durante 2 segundos, Wireshark capturó más de 95 mil paquetes SYN enviados al servidor (en la interface *loopback*). Además, muestra mensajes cuya información es “[TCP ports numbers reused]” que indica que se están reutilizando puertos por donde ya ha sido transferido un paquete SYN (esto es un indicio indirecto de posible ataque *Syn Flood*). Sin embargo, debido al funcionamiento de Wireshark, éste no nos entrega información, ya que captura los paquetes antes de que *iptables* pueda ejecutar su función. Por lo tanto, comprobamos la funcionalidad directamente con *iptables*.

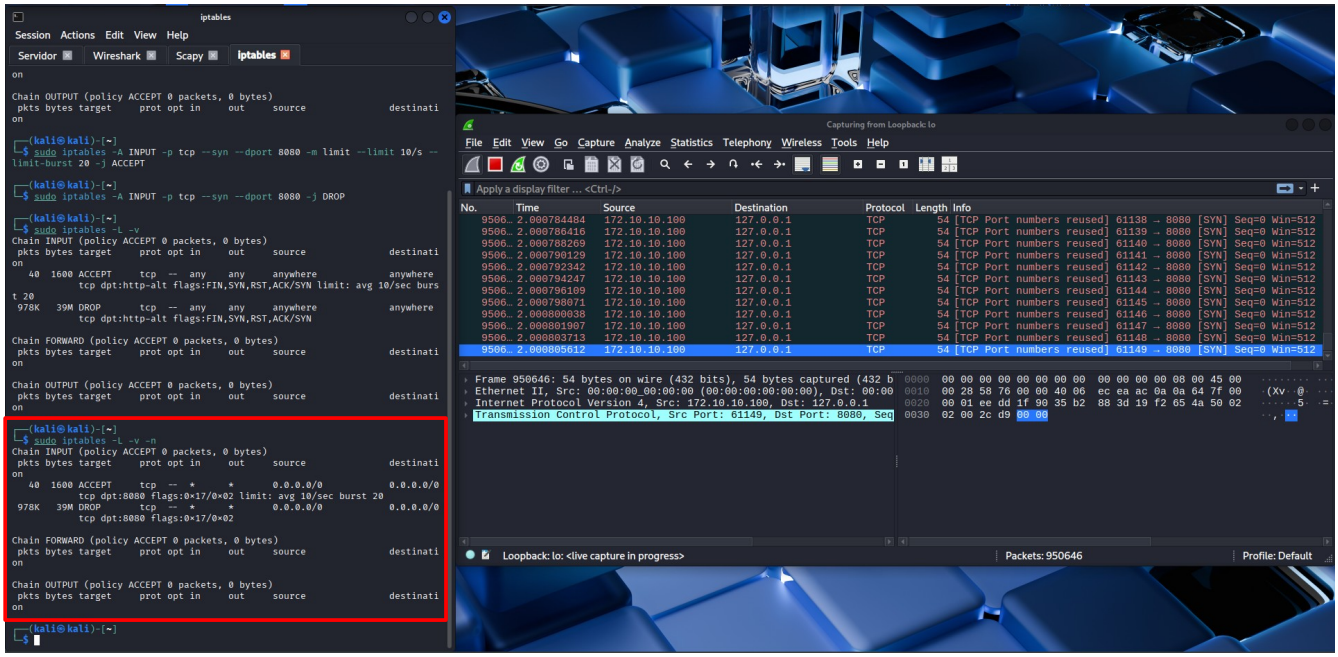

Figura 10: Comprobación de eficacia de regla iptables para mitigar el ataque SYN Flood ejecutado contra el servidor de pruebas.

En el detalle del resultado, obtenemos dos datos importantes para comprobar la efectividad de las reglas:

- La primera regla ha permitido el paso de 40 paquetes SYN.
- La segunda regla ha *dropeado* cerca de 1 millón de paquetes SYN que no cumplen con la primera regla.

5. Registrar capturas de pantalla y explicar el ataque realizado, sus efectos en la red y las medidas de defensas, además de identificar vulnerabilidades y proponer acciones correctivas.

Contexto del ataque ejecutado

El ataque ejecutado en el Punto 4 del presente informe fue un *SYN Flood*, en el cual un atacante inunda un dispositivo víctima con paquetes SYN (Synchronize) maliciosos. Este ataque se basa en el procedimiento 3way handshake, donde un dispositivo envía una solicitud a un dispositivo para poder entablar una conexión con otro. El proceso simplificado es el siguiente:

1. Dispositivo solicitante envía un paquete SYN a uno receptor.
2. El dispositivo receptor envía un SYN/ACK al solicitante, indicándole que acepta su solicitud.
3. Finalmente el dispositivo solicitante envía un ACK para que el receptor finalice el proceso de solicitud y entable la conexión entre ambos.

Efectos del ataque *SYN Food* en la red

El comando ejecutado en este informe tiene como objetivo inundar el servidor de pruebas que está en escucha, donde el objetivo final es provocar una denegación del servicio en el puerto 8080 (DoS). Al enviar un alto tráfico, llega un punto en que el servidor ve agotado sus recursos (ya que intentará responder a cada una de las solicitudes SYN), dejando de responder solicitudes legítimas y llegando al colapso, lo que significa la pérdida del servicio ofrecido.

Vulnerabilidades y acciones correctivas

Un ataque *SYN Flood* puede afectar a cualquier dispositivo que utilice TCP, ya que como se ha mencionado este ataque se aprovecha de un procedimiento central en la comunicación de los dispositivos; prácticamente cualquier dispositivo es vulnerable a este tipo de ataques. Sin embargo, existen medidas que se pueden tomar para minimizar el impacto en la red afectada. Algunas acciones a realizar son:

- Configurar reglas en firewall para bloquear tráfico sospechoso
  - ► Se puede configurar una regla que limite la cantidad de paquetes que ingresen al servidor, y *dropear* el tráfico que no cumpla con la regla (medida aplicada en Punto 4).
  - ► Implementar reglas que detecten y bloqueen IPs con comportamiento sospechoso (en el caso del laboratorio, esta medida no fue aplicada debido al comportamiento del ataque, donde se utilizaron IPs falsificadas, por lo que la medida pierde efectividad, sin embargo es una medida extra que se puede implementar).

- A nivel de servidor, implementar *SYN Cookies*, la cual es una técnica que permite al servidor no almacenar en memoria los estados de las peticiones SYN sino hasta que el dispositivo solicitante responda al SYN/ACK que le envía el servidor.
- Utilizar servicios anti DoS en la nube, que permiten mitigar parte de un ataque dirigido a la red de la organización. Sin embargo, es una medida a considerar por la empresa ya que representa una inversión extra.

##

Fin del informe

##