Ejercicios de Seguridad en Redes
Escaneo de puertos + Escucha y análisis de tráfico

SSI 2022/23

17 de noviembre de 2022

Índice

1. Entorno de pruebas

1.1. Software de virtualización VIRTUALBOX

En estas prácticas se empleará el software de virtualización VIRTUALBOX para simular los equipos GNU/Linux sobre los que se realizarán las pruebas.

- Página principal: `http://virtualbox.org`
- Más información: `http://es.wikipedia.org/wiki/Virtualbox`

1.2. Imágenes a utilizar

1. Scripts de instalación
   - para GNU/Linux: ejercicio-nmap.sh
     `alumno@pc: $ sh ejercicio-nmap.sh`
   - para MS windows: ejercicio-nmap.ps1
     `Powershell.exe -executionpolicy bypass -file ejercicio-nmap.ps1`

   Notas:
   - Se pedirá un identificador (sin espacios) para poder reutilizar las versiones personalizadas de las imágenes creadas (usad por ejemplo el nombre del grupo de prácticas o el login LDAP)
   - En ambos scripts la variable `$DIR_BASE` especifica donde se descargarán las imágenes y se crearán las MVs.
     Por defecto en GNU/Linux será en `$HOME/SSI2223` y en Windows en `C:/SSI2223`.
     Puede modificarse antes de lanzar los scripts para hacer la instalación en otro directorio más conveniente (disco externo, etc)
   - Es posible descargar las imágenes comprimidas manualmente (o intercambiarlas con USB), basta descargar los archivos con extensión `.vdi.zip` de `http://ccia.esei.uvigo.es/docencia/SSI/2223/practicas/` y copiarlos en el directorio anterior (`$DIR_BASE`) para que el script haga el resto.
   - Si no lo hacen desde el script anterior, se pueden arrancar las instancias VIRTUALBOX desde el interfaz gráfico de VirtualBOX o desde la línea de comandos con `VBoxManage startvm <nombre MV>_<id>`
2. Imágenes descargadas
   - **parrot_ssi.vdi** (1,6 GB comprimida, 5,2 GB descomprimida): Imagen genérica (común a todas las MVs) que contiene las herramientas a utilizar
     Contiene un sistema Parrot Security OS (basado en Debian) con herramientas gráficas y un entorno gráfico ligero LXDE (*Lighweight X11 Desktop Environment*) [LXDE].
   - **swap1GB.vdi**: Disco de 1 GB formateado como espacio de intercambio (SWAP)
3. Usuarios configurados e inicio en el sistema
   - Usuarios disponibles

login	password
root	purple
usuario (con privilegios sudo)	usuario

   - Acceso al entorno gráfico una vez logueado (necesario para poder copiar y pegar desde/hacia el anfitrión)

     `root@base:~# startx`

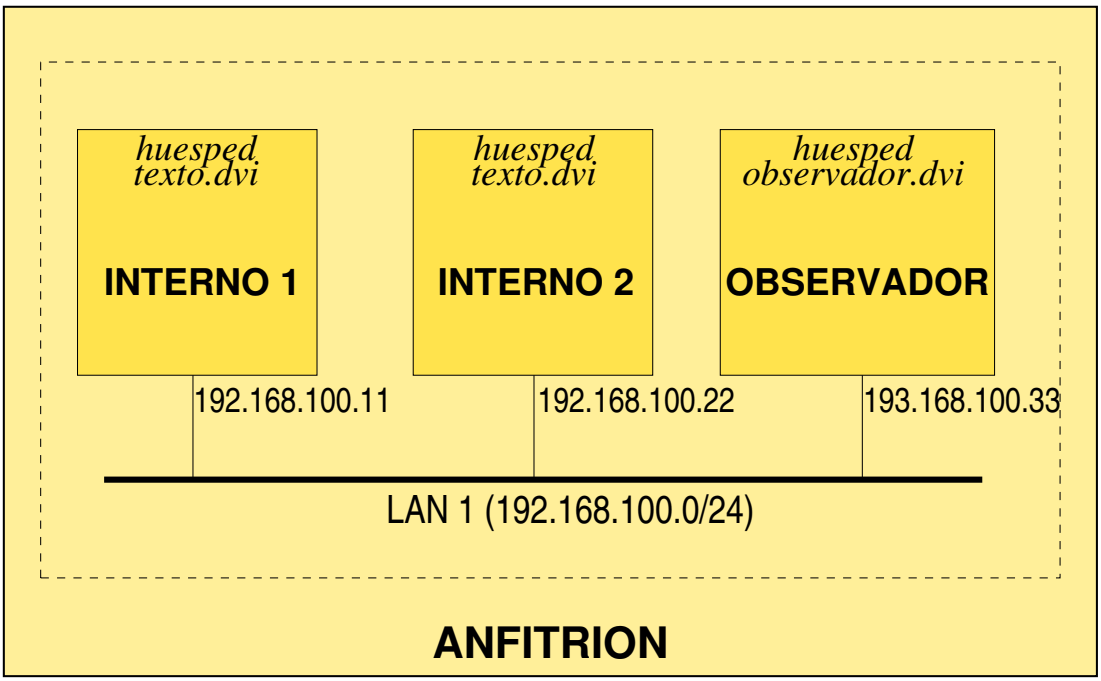
   - Habilitar copiar y pegar desde/hacia el anfitrión en el menú `Dispositivos -> Portapapeles compartido -> bidir` de la ventana de la máquina virtual.

2. Introducción

El ejercicio consta de dos partes.

- Realizar una sesión de recopilación de información empleando el escáner de puertos NMAP.
- Realizar una sesión de intercepción de mensajes utilizando la herramienta de seguridad en redes Betterap y el *sniffer*/analizador de redes WIRESHARK para comprobar la vulnerabilidad de los servicios que no usan cifrado.

- Red donde se realizarán los ejercicios:



- Direcciones MAC e IP

máquina	dirección MAC	dirección IP
`interno1`	`08:00:27:11:11:11`	192.168.100.11
`interno2`	`08:00:27:22:22:22`	192.168.100.22
`observador`	`08:00:27:33:33:33`	192.168.100.33

- Servicios arrancados por defecto en todas las máquinas
  - servidor web (Apache 2) [**Nota:** puede ser necesario reiniciarlo manualmente con `systemctl restart apache2`]
  - servidor telnet (arrancado por `openbsd-inetd`)
  - servidor SSH (openSSH)
  - servidor ftp (arrancado por `openbsd-inetd`)
  - servidor finger (arrancado por `openbsd-inetd`)
  - servidor MySQL
  - servidor SMTP (postfix)
  - servidores POP3 e IMAP (dovecot)
  - servidor DNS (bind)

3. Ejercicio 1: Escaneo de puertos

El primer ejercicio consistirá en el uso de la herramienta de escaneo de puertos NMAP para obtener información de los equipos y servicios de la red.

NMAP implementa diversas técnicas para extraer información de los equipos que forman parte de una red y para identificar los puertos y servicios que están disponibles en distintas máquinas. Algunos de los métodos disponibles realizan el escaneo sin apenas dejar rastro, mientras que otros dejarán un rastro en los ficheros de log de las máquinas analizadas.

- Página de NMAP: `http://www.nmap.org`

- Más información: `http://es.wikipedia.org/wiki/Nmap`
- Manual en español: `http://nmap.org/man/es/`
- Técnicas de escaneo: `https://nmap.org/man/es/man-port-scanning-techniques.html`

3.1. PREVIO: Preparación

En la máquina **interno1 (192.168.100.11)**: habilitar y poner en marcha el servicio de LOG del sistema `rsyslog`

```
interno1:~# systemctl enable rsyslog
interno1:~# systemctl start rsyslog
```

Al arrancar, `rsyslog` empezará a registrar entradas en una serie de ficheros de LOG en el directorio `/var/log`: `syslog`, `auth.log`, `messages`, `daemon.log`, `kernel.log`, etc

3.2. Pasos a seguir

1. Enumerar equipos de la red y sus servicios

   Desde la máquina **observador (192.168.100.33)**:

   a) Lanzar un escaneado Ping Sweeping [opción **-sP**] para identificar, mediante Ping, las máquinas que componen la red

   ```
   observador:~# nmap -sP 192.168.100.0/24
   ```

   b) Sobre cada uno de los equipos que aparezcan como activos (exluido **observador**) realizar un escaneo de tipo TCP connect scanning [opción **-sT**] para determinar que puertos están abiertos.

   ```
   observador:~# nmap -sT -v -T4 192.168.100.11
   observador:~# nmap -sT -v -T4 192.168.100.22
   ```

   c) Repetir el escaneado sobre INTERNO1(192.168.100.11), añadiendo la opción **-O** para que NMAP trate de identificar el Sistema Operativo que ejecuta y la opción **-sV** para determinar la versión concreta de los servicios que tiene activados.

   ```
   observador:~# nmap -sT -O -sV -T4 192.168.100.11   (tarda unos segundos)
   ```

   Los escaneados anteriores dejan rastro. Comprobar los ficheros de log `/var/log/syslog`, `/var/log/daemon.log` o logs específicos de servidores en la máquina **interno1** y verificar que ha quedado constancia de las conexiones realizadas por NMAP.

   ```
   interno1:~# tail -200 /var/log/syslog | less       (ampliar el limite de lineas [200] si es necesario
   ```

   Nota: El rastro del escaneo de tipo `-sT` que queda en `/var/log/syslog` fue guardado por los servidores `in.fingerd, inetd, telnetd, ftpd, dovecot, postfix/smtpd` en el momento en que se completó el establecimiento de la conexión TCP (negociación `SYV,SYN-ACK,ACK`).

2. Comprobar escaneos "sigilosos"

   Evaluaremos el comportamiento de los distintos tipos de escaneo sobre la máquina **interno1**(192.168.100.11)

   a) En la máquina INTERNO1(192.168.100.11) se habilitará una regla del firewall *netfilter* para hacer log de los paquetes SYN con intentos de conexión TCP.

   - Escribir el siguiente comando **iptables**

     ```
     interno1:~# iptables -A INPUT -i enp0s3 -p tcp \
                                  --tcp-flags SYN SYN -m state --state NEW \
                                  -j LOG --log-prefix "INICIO CONEXION:"
     ```

     Nota: Esta regla iptables captura los mensajes TCP de inicio de conexión (flag `syn` a 1) y crea una entrada en el log del sistema etiquetada con `INICIO CONEXION:`

- Monitorizar continuamente el fichero de log `/var/log/syslog`, con el comando `tail -f`
  `interno1:~# tail -f /var/log/syslog`
  (el terminal se libera con CONTROL+C)

b) Desde la máquina **observador(192.168.100.33)** lanzar 3 tipos de escaneos nmap y comprobar en INTERNO1(192.168.100.11) como evoluciona el log.

TCP connect scanning [opción `-sT`] Escaneo con conexiones TCP completas (opción por defecto)

`observador:~# nmap -sT 192.168.100.11`

→ Generará entradas etiquetadas con `INICIO CONEXION:`, junto con las entradas generadas por los servidores (`in.fingerd, inetd, telnetd, ftpd, ...`)

SYN scanning [opción `-sS`] Escaneo con paquetes SYN (conexiones parcialmente iniciadas)

`observador:~# nmap -sS 192.168.100.11`

→ Generará entradas etiquetadas con `INICIO CONEXION:`, pero no las generadas por los servidores (`in.fingerd, inetd, telnetd, ftpd, ...`)

NULL scanning [opción `-sN`] Escaneo con paquetes "nulos" (todos los flags TCP a 0)

`observador:~# nmap -sN 192.168.100.11`

→ No generará entradas etiquetadas con `INICIO CONEXION:`

4. Ejercicio 2: Intercepción de mensajes: protocolos en claro vs. protocolos cifrados

El segundo ejercicio consistirá en el uso del conjunto de utilidades disponibles en *Bettercap* y de la herramienta WIRESHARK desde el equipo **observador** para interceptar el tráfico TELNET, SSH, HTTP y HTTPS entre los equipos **interno1** e **interno2**.

- *Bettercap* ofrece una collección de herramientas (módulos) para llevar a cabo una serie de atques en redes IPv4, iPv6, Bluetooth o Wifi.
  - *Bettercap* ofrece soporte para tareas como:
    - envenenamiento ARP (o ARP spoofing, `https://en.wikipedia.org/wiki/ARP_spoofing`)
    - captura (sniffing) de tráfico mediante ataques MITM (*man in the middle*, `https://en.wikipedia.org/wiki/Man-in-the-middle_attack`)
  - Página de *Bettercap*: `https://www.bettercap.org/`
- WIRESHARK es un *sniffer* y analizador de protocolos que recopila los paquetes que fluyen por la red, los analiza, extrae el contenido de los campos de diferentes protocolos y los presenta al usuario.
  - Página de WIRESHARK: http://www.wireshark.org
  - Más información: http://es.wikipedia.org/wiki/Wireshark
  - Por comodidad, también se usará el comando `tshark` que ofrece la misma funcionalidad desde un terminal en modo texto (ver `https://tshark.dev/`)

4.1. PREVIO: (Preparación 1) Habilitar TLS/SSL en Apache

PREVIO: Habilitar el soporte SSL en el servidor `apache2` de **interno2 (192.168.100.22)**

En el equipo **interno2 (192.168.100.22)**:

1. Crear un certificado autofirmado para el servidor web.

```
interno2:~# mkdir /etc/apache2/ssl/
interno2:~# make-ssl-cert /usr/share/ssl-cert/ssleay.cnf /etc/apache2/ssl/apache.pem
```

   - Cuando se solicite el `commonName` del servidor HTTP, indicar **interno2.ssi.net**

- Cuando se solicite los *"nombres alternativos*, dejar el campo en blanco

  Nota: Para moverse por los campos del formulario de configuración del certificado se puede usar la tecla TAB para cambiar de campo y la tecla INTRO para confirmar las elecciones y avanzar.

El fichero generado (`/etc/apache2/ssl/apache.pem`) contiene tanto el certificado del servidor como la clave privada asociada al mismo.

```
interno2:~# cat /etc/apache2/ssl/apache.pem
interno2:~# openssl x509 -text -in /etc/apache2/ssl/apache.pem
interno2:~# openssl rsa -text -in /etc/apache2/ssl/apache.pem
```

Nota: `make-ssl-cert` es una utilidad de Debian (incluida en el paquete DEB *ssl-cert*) para generar certificados autofirmados para pruebas (los datos de configuración del certificado a generar se indican en `/usr/share/ssl-cert/ssleay`
Internamente hace uso de las utilidades de la librería openSSL.

Nota: En un servidor real se suele utilizar un certificado emitido por una autoridad de certificación (CA) reconocida (o bien una CA pública o una CA propia de la organización). No es recomendable utilizar certificados autofirmados en sistemas en producción ya que son fácilmente falsificables.

2. Editar la configuración SSL por defecto para indicar el certificado del servidor y su respectiva clave privada.

   ```
   interno2:~# nano /etc/apache2/sites-available/default-ssl.conf
   ```

   Asignar los siguientes valores a los parámetros (en caso de que estén comentados descomentarlos)

   ```
   ...
   SSLEngine on
   ...
   SSLCertificateFile /etc/apache2/ssl/apache.pem
   SSLCertificateKeyFile /etc/apache2/ssl/apache.pem
   ...
   ```

   Asegurar que el fichero `/etc/apache2/ports.conf` incluya el valor `Listen 443`

3. Habilitar soporte SSL en Apache2 y habilitar la configuracion SSL por defecto

   ```
   interno2:~# a2enmod ssl
   interno2:~# a2ensite default-ssl
   interno2:~# systemctl restart apache2
   ```

   Nota:

   - **a2enmod** es un comando (en Debian y derivados) para habilitar módulos de Apache2
     Los ficheros de configuración de los módulos disponibles están en `/etc/apache2/mods-available/` y al habilitarlos se crea un enlace simbólico desde `/etc/apache2/mods-enabled/`
   - **a2ensite** es un comando (en Debian y derivados) para habilitar configuraciones de "sitios web" en Apache2
     Los ficheros de configuración de los "sitios web" disponibles (normalmente son configuraciones de servidores virtuales Apache) están en `/etc/apache2/sitess-available/` y al habilitarlos se crea un enlace simbólico desde `/etc/apache2/sites-enabled/`

4.2. PREVIO: (Preparación 2) Envenenamiento ARP y MITM con Bettercap

4.2.1. Comprobaciones iniciales

1. Verificar que la máquina **observador(192.168.100.33)** no tiene acceso al tráfico entre **interno1** e **interno2**

   a) En **observador(192.168.100.33)**: en un nuevo terminal arrancar `tshark` y ponerlo en escucha sobre la tarjeta *enp0s3* para ver (en caso de ser posible) el tráfico que pasa por la red

```
observador:~# tshark --color -i enp0s3
```

b) En **interno1(192.168.100.11)**:

- Establecer conexión `telnet` con **interno2** (con login `usuario` y contraseña `usuario`)

```
interno1:~# telnet interno2.ssi.net

Trying 192.168.100.22...
Connected to interno2.ssi.net.
Escape character is '^]'.

Linux 5.18.0-14parrot1-amd64 (interno2.ssi.net) (pts/0)

interno2.ssi.net nombre: usuario
Contraseña: usuario

Linux interno2.ssi.net 5.18.0-14parrot1-amd64
...
interno2:~$ ls -l
interno2:~$ exit
```

- Establecer conexión HTTP con **interno2** usando `lynx`

```
interno1:~# lynx interno2.ssi.net    (salir con tecla Q o [control]+C)
```

c) En **observador**: Verificar que el sniffer no ha capturado paquetes de esas conexiones y cerrar `tshark` con `[control]+C`

2. Consultar tablas ARP en **interno1** e **interno2**

```
interno1:~# arp -n

Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.100.33           ether   08:00:27:33:33:33   C                     enp0s3
192.168.100.1                    (incomplete)                              enp0s3
192.168.100.22           ether   08:00:27:22:22:22   C                     enp0s3

interno2:~# arp -n

Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.100.33           ether   08:00:27:33:33:33   C                     enp0s3
192.168.100.11           ether   08:00:27:11:11:11   C                     enp0s3
192.168.100.1                    (incomplete)                              enp0s3
```

4.2.2. Envenenamiento ARP (ARP spoofing) y ataque MITM

Objetivos:

- Inundar la red local con respuestas ARP falsas para "convencer" a los equipos de la red que la dirección MAC vinculada a las IPs 192.168.100.11 (**interno1**) y 192.168.100.22 (**interno2**) es `08:00:27:33:33:33` (que realmente pertene a 192.168.100.33 (observador)).
- **observador** actuará como intermediario (MITM: *man in the middle*), haciéndose pasar por **interno2** ante **interno1** y por **interno1** ante **interno2**
- **observador** almacenará el tráfico procedente de las máquinas **interno1** e **interno2** antes de retransmitirlo al destino legítimo haciéndose pasar ante él como origen legítimo (cmabiando las direcciones MAC de las tramas Ether según corresponda).

Para llevar a cabo el *ARP spoofing* y el ataque MITM se usarán los suguientes módulos de Bettercap

- `net.probe`: Módulo que sondea la red enviando paquetes de prueba a cada IP para que el módulo `net.recon` los detecte (ver `https://www.bettercap.org/modules/ethernet/net.recon/`)
- `net.recon`: Módulo que consulta la tabla ARP del sistema para detectar las nuevas máquinas que vayan apareciendo en la red monitorizada (ver `https://www.bettercap.org/modules/ethernet/net.probe/`)
- `arp.spoof`: Módulo que inunda la red con paquetes ARP para hacerse pasar por las máquinas con las IPs seleccionadas y hacer posible el ataque MITM (ver `https://www.bettercap.org/modules/ethernet/spoofers/arp.spoof/`)
- `net.sniff`: Módulo que actua como sniffer capturando los paquetes recopilados gracias a la falsificación ARP y retransmitiéndolos a los destinatarios originales. Tiene capacidad de reconocer diferentes protocolos y realizar filtrado sobre sus paquetes (ver `https://www.bettercap.org/modules/ethernet/net.sniff/`)

Nota: opcionalmente se puede habilitar el interfaz web de Bettercap haciendo uso de los módulos `api.rest` y `http.ui` (ver `https://www.bettercap.org/usage/webui/`).

Pasos a seguir:

1. En **observador(192.168.100.33)**: desde un terminal propio, arrancar *Bettercap* y sondear la red para identificar los equipos conectados

```
observador:~#  bettercap -iface enp0s3

192.168.100.0/24 > 192.168.100.33  » help net.probe
192.168.100.0/24 > 192.168.100.33  » help net.recon

192.168.100.0/24 > 192.168.100.33  » net.recon on
192.168.100.0/24 > 192.168.100.33  » net.probe on         (esperar unos segundos)
192.168.100.0/24 > 192.168.100.33  » net.probe off
192.168.100.0/24 > 192.168.100.33  » net.show

+----------------+-------------------+-------------------+----------------------------+-------+-------
|      IP        |        MAC        |       Name        |          Vendor            | Sent  | Recvd
+----------------+-------------------+-------------------+----------------------------+-------+-------
| 192.168.100.33 | 08:00:27:33:33:33 | enp0s3            | PCS Computer Systems GmbH  | 0 B   | 0 B
|                |                   |                   |                            |       |
| 192.168.100.11 | 08:00:27:11:11:11 | interno1.ssi.net. | PCS Computer Systems GmbH  | 120 B | 92 B
| 192.168.100.22 | 08:00:27:22:22:22 | interno2.ssi.net. | PCS Computer Systems GmbH  | 120 B | 92 B
+----------------+-------------------+-------------------+----------------------------+-------+-------

192.168.100.0/24 > 192.168.100.33  »
```

   Nota 1: No se debe salir de la consola de *Bettercap* durante la sesión de ARP Spoofing y MITM, es necesario que los módulos de *Bettercap* estén en funcionamiento.

   Nota 2: (opcional) Es posible poner en marcha el interfaz Web de *Bettercap*

   - Es necesario detener el servidor `apache` con el comando `systemctl stop apache2`
   - En la consola de *Bettercap*: lanzar el *caplet* `http-ui` (script propio de *Bettercap*) con el comando `include http-ui` (ver código en `/usr/share/bettercap/caplets/http-ui.cap`)
   - El interfaz web estará disponible en la URL `http://127.0.0.1:80`, con el usuario `user` y la contraseña `pass` (definidos en `/usr/share/bettercap/caplets/http-ui.cap`))

2. En **observador(192.168.100.33)**: en un terminal propio (diferente al de *Bettercap*), arrancar `tshark` y ponerlo en escucha sobre la tarjeta *enp0s3* para ver el tráfico ARP generado por *Bettercap* durante el ataque de ARP *spoofing*

```
observador:~# tshark --color -i enp0s3
```

3. En **observador(192.168.100.33)**: desde el terminal de *Bettercap* configurar y arrancar el ataque ARP *spoofing*

```
192.168.100.0/24 > 192.168.100.33  » help arp.spoof
192.168.100.0/24 > 192.168.100.33  » help net.sniff

192.168.100.0/24 > 192.168.100.33  » set arp.spoof.internal true
192.168.100.0/24 > 192.168.100.33  » set arp.spoof.targets 192.168.100.11,192.168.100.22
192.168.100.0/24 > 192.168.100.33  » arp.spoof on
```

4. En **observador(192.168.100.33)**: verificar el tráfico capturado por `tshark` y cerrar la captura con `[control]+C`

5. Consultar las tablas ARP en **interno1** e **interno2**

```
interno1:~# arp -n

interno2:~# arp -n
```

CUESTION 1: ¿Cómo son los mesajes ARP que envia el módulo `arp.spoof`? ¿Cómo quedan las tablas ARP de **interno1** e **interno2** y qué significan esos valores?

4.3. Tarea 1: Escucha del protocolo TELNET

1. En **observador (192.168.100.33)**: iniciar la captura de tráfico con el módulo `net.sniff` de *Bettercap*, dejará los paquetes capturadas en el fichero `/tmp/telnet.pcap`

```
192.168.100.0/24 > 192.168.100.33  » set net.sniff.filter "not arp"
192.168.100.0/24 > 192.168.100.33  » set net.sniff.output /tmp/telnet.pcap
192.168.100.0/24 > 192.168.100.33  » net.sniff on
```

2. En **interno1 (192.168.100.11)**: iniciar una conexión TELNET con **interno2 (192.168.100.22)**

```
interno1:~#  telnet interno2.ssi.net

Trying 192.168.100.22...
Connected to interno2.ssi.net.
Escape character is '^]'.

Linux 5.18.0-14parrot1-amd64 (interno2.ssi.net) (pts/1)

interno2.ssi.net nombre: usuario
Contraseña: usuario

Linux interno2.ssi.net 5.18.0-14parrot1-amd64
...
interno2:~$ ls -l
...
interno2:~$ exit
```

3. En **observador (192.168.100.33)**: analizar el tráfico recopilado

   - Detener captura en *Bettercap* con `net.sniff off`

     ```
     192.168.100.0/24 > 192.168.100.33  » net.sniff off
     ```

   - Desde otro terminal, abrir con Wireshark el fichero .PCAP con los paquetes capturados

     ```
     observador:~# wireshark /tmp/telnet.pcap
     ```

   - Recorrer los paquetes capturados y comprobar los datos intercambiados
     - Puede filtrarse el tráfico TELNET capturado, poniendo `telnet` en el campo de fitros de Wireshark
     - Puede verse el tráfico completo de la conexión `telnet` en embos sentidos
       - sobre el primer paquete de la conexión TELNET dirigido al puerto 23
       - `[botón derecho] > Seguir > Flujo TCP`

4.4. Tarea 2: Escucha del protocolo SSH

1. En **observador (192.168.100.33)**: iniciar la captura de tráfico con el módulo `net.sniff` de *Bettercap*, dejará los paquetes capturadas en el fichero `/tmp/ssh.pcap`

   ```
   192.168.100.0/24 > 192.168.100.33  » set net.sniff.output /tmp/ssh.pcap
   192.168.100.0/24 > 192.168.100.33  » net.sniff on
   ```

2. En **interno1 (192.168.100.11)**: iniciar una conexión SSH con **interno2 (192.168.100.22)**

   ```
   interno1:~# ssh usuario@interno2.ssi.net

   usuario@interno2.ssi.net's password: usuario
   Linux interno2.ssi.net 5.18.0-14parrot1-amd64
   ...
   interno2:~$ ls -l
   ...
   interno2:~$ exit
   ```

3. En **observador (192.168.100.33)**: analizar el tráfico recopilado

   - Detener captura en *Bettercap* con `net.sniff off`

     ```
     192.168.100.0/24 > 192.168.100.33  » net.sniff off
     ```

   - Desde otro terminal, abrir con Wireshark el fichero .PCAP con los paquetes capturados

     ```
     observador:~# wireshark /tmp/ssh.pcap
     ```

   - Recorrer los paquetes capturados y comprobar los datos intercambiados
     - Puede filtrarse el tráfico SSH capturado, poniendo `ssh` en el campo de fitros de Wireshark
       - Deberán aparecer los paquetes de inicio de conexión, acuerdo de algoritmo e inicio de intercambio de claves (`Key Exchange Init`), intercambio de parámetros Diffie-Hellman y `New Keys` marcando el fin de la negociación y el inicio del tráfico cifrado con las claves secretas acordadas.
       - El resto de paquetes intercambiados (incluida la contraseña de usuario) serán datos cifrados
     - Puede "verse" el tráfico completo de la conexión `ssh` en ambos sentidos
       - sobre el primer paquete de la conexión SSH dirigido al puerto 22
       - `[botón derecho] > Seguir > Flujo TCP`

4.5. Tarea 3: Escucha del protocolo HTTP

1. En **observador (192.168.100.33)**: iniciar la captura de tráfico con el módulo `net.sniff` de *Bettercap*, dejará los paquetes capturadas en el fichero `/tmp/http.pcap`

   ```
   192.168.100.0/24 > 192.168.100.33  » set net.sniff.output /tmp/http.pcap
   192.168.100.0/24 > 192.168.100.33  » net.sniff on
   ```

2. En **interno1 (192.168.100.11)**: iniciar una conexión HTTP con **interno2 (192.168.100.22)** usando el navegador `lynx` (también puede hacer con el navegador gráfico *Falkon*)

   ```
   interno1:~# lynx interno2.ssi.net

   (para generar tráfico, acceder a alguna de las aplicaciones disponibles, por ejemplo DVWA con admin/p
   ```

   Nota: En caso de no obtener respuesta, reiniciar el servidor `apache2` en la máquina **interno2 (192.168.100.22)**

   ```
   interno2:~# systemctl restart apache2
   ```

3. En **observador (192.168.100.33)**: analizar el tráfico recopilado

- Detener captura en *Bettercap* con `net.sniff off`

  ```
  192.168.100.0/24 > 192.168.100.33  » net.sniff off
  ```

- Desde otro terminal, abrir con Wireshark el fichero .PCAP con los paquetes capturados

  ```
  observador:~# wireshark /tmp/http.pcap
  ```

- Recorrer los paquetes capturados y comprobar los datos intercambiados
  - Puede filtrarse el tráfico HTTP capturado, poniendo `http` en el campo de fitros de Wireshark
  - Puede verse el tráfico de cada una de las conexiones `http` por separado
    - sobre el primer paquete de la primera petición HTTP
    - `[botón derecho] > Seguir > Flujo TCP`
    - `[botón derecho] > Seguir > Flujo HTTP`

4.6. Tarea 4: Escucha del protocolo HTTPS

1. En **observador (192.168.100.33)**: iniciar la captura de tráfico con el módulo `net.sniff` de *Bettercap*, dejará los paquetes capturadas en el fichero `/tmp/https.pcap`

   ```
   192.168.100.0/24 > 192.168.100.33  » set net.sniff.output /tmp/https.pcap
   192.168.100.0/24 > 192.168.100.33  » net.sniff on
   ```

2. En **interno1 (192.168.100.11)**: iniciar una conexión HTTP con **interno2 (192.168.100.22)** usando el navegador `lynx` (también puede hacer con el navegador gráfico *Falkon*, importante añadir `https`)

   ```
   interno1:~# lynx https://interno2.ssi.net

   (para generar tráfico, acceder a alguna de las aplicaciones disponibles, por ejemplo DVWA con admin/p
   ```

   Nota: El navegador pedirá confirmación (`lynx` varias veces) al tratarse de un certificado autofirmado emitido por una CA no reconocida (no está en `/etc/ssl/certs/ca-certificates.pem`) y considerarlo no fiable.

3. En **observador (192.168.100.33)**: analizar el tráfico recopilado

   - Detener captura en *Bettercap* con `net.sniff off`

     ```
     192.168.100.0/24 > 192.168.100.33  » net.sniff off
     ```

   - Desde otro terminal, abrir con Wireshark el fichero .PCAP con los paquetes capturados

     ```
     observador:~# wireshark /tmp/https.pcap
     ```

   - Recorrer los paquetes capturados y comprobar los datos intercambiados
     - Puede filtrarse el tráfico HTTPS capturado, poniendo `tls` en el campo de fitros de Wireshark
       - Se usa por defecto la versión TLS 1.3 y aparecerán paquetes y mensajes similares a los descritos en `https://tls13.xargs.org/`
       - Una vez concluida la negociación los mensaje con las peticiones y respuestas HTTP irán cifrados y autenticados
     - Puede "verse" el tráfico de cada una de las conexiones `tls`
       - sobre el primer paquete de la conexión (`Client Hello`)
       - `[botón derecho] > Seguir > Flujo TCP`

5. Documentación y entrega

El material entregable de esta práctica constará de una pequeña memoria documentando los ejercicios realizados y los resultados obtenidos en cada paso realizado, junto con las conclusiones que se deriven de dichos resultados.

- Descripción breve de acciones realizadas y resultados obtenidos en ejercicio 1
- Conclusiones del ejercicio 1 (explicar la diferencia entre los tres tipos de escaneo realizados [TCP Scan, SYN Scan, NULL Scan])
- Descripción breve de acciones realizadas y resultados obtenidos en ejercicio 2
- Respuesta a la **Cuestión 1** indicada al final de la sección 4.2.2
- Resultados de las tareas 1, 2, 3 y 4 (aportar capturas de wireshark o ejemplos de los paquetes capturados)
- Conclusiones del ejercicio 2 (explicar la diferencia en el uso de `telnet` frente a `ssh` y de `http` respecto `https`)

ENTREGA: en MOOVI, hasta **27/12/2022** (individual o parejas)