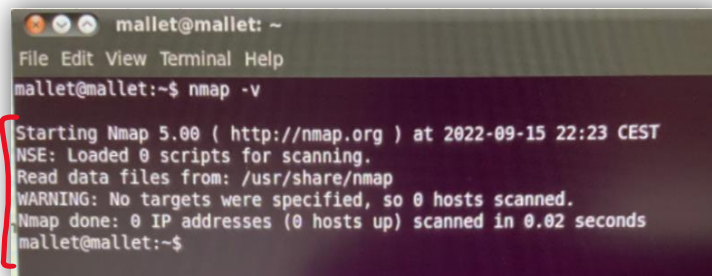

Informe Entrega 2 Parte 1

Informe sobre la segunda practica de la asignatura Garantía y Seguridad de la Información. En esta primera parte hemos dado los distintos servicios de la red, sus correspondientes puertos y nos hemos centrado en los servicios de telnet y SSH. Tras esto hemos ampliado lo que vimos sobre nmap en el anterior laboratorio estudiando más funciones y posibilidades que ofrece. Finalmente hemos visto una manera de descubrir la versión de un sistema y ver sus exploits.

Usando la página de manual de nmap, documente las funciones y opciones básicas de nmap:

nmap es una herramienta que utilizamos para escanear o “escuchar” lo que pasa por un puerto en la red. Tiene distintas funciones y flags y voy a enumerar los básicos:

1. nmap -v (verbose): Esta función nos permite obtener mucha más información del escaneo. Nos va mostrando lo que pasa a cada momento.



```
mallet@mallet: ~  
File Edit View Terminal Help  
mallet@mallet:~$ nmap -v  
Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-15 22:23 CEST  
NSE: Loaded 0 scripts for scanning.  
Read data files from: /usr/share/nmap  
WARNING: No targets were specified, so 0 hosts scanned.  
Nmap done: 0 IP addresses (0 hosts up) scanned in 0.02 seconds  
mallet@mallet:~$
```

2. nmap 10.0.2.4 (o la IP que deseemos): Con esta función lo que conseguimos es que se haga un escaneo de los 1000 servicios TCP más utilizados que pasan por ese host.

```
mallet@mallet:~$ nmap 10.0.2.4
Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-15 22:25 CEST
Interesting ports on 10.0.2.4:
Not shown: 991 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
80/tcp    open  http
111/tcp   open  rpcbind
443/tcp   open  https
2049/tcp  open  nfs
6000/tcp  open  X11
Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
```

3. nmap 10.0.2.4 -p-: El flag “-p-” nos permite escanear los 65535 (2^{16}) puertos TCP o UDP (depende del resto de flags) que pasan por el host en vez de los 1000 que se escanean por defecto.

```
Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-15 22:31 CEST
Interesting ports on 10.0.2.4:
Not shown: 65522 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
80/tcp    open  http
111/tcp   open  rpcbind
443/tcp   open  https
863/tcp   open  unknown
2049/tcp  open  nfs
6000/tcp  open  X11
46980/tcp open  unknown
59365/tcp open  unknown
60516/tcp open  unknown
Nmap done: 1 IP address (1 host up) scanned in 4.54 seconds
mallet@mallet:~$ nmap 10.0.2.4 -p- -v
```

4. nmap 10.0.2.4 -p- -sT: Con “-sT” hacemos un escaneo completo (handshake) solo de los puertos TCP.

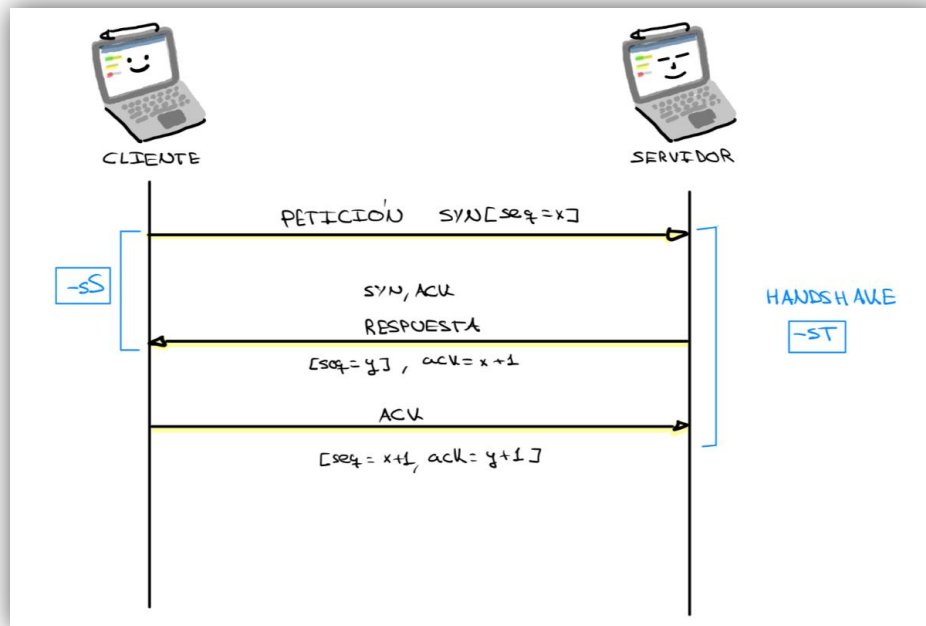
```
mallet@mallet:~$ sudo nmap 10.0.2.4 -p- -sT
Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-15 23:35 CEST
Interesting ports on 10.0.2.4:
Not shown: 65522 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
80/tcp    open  http
111/tcp   open  rpcbind
443/tcp   open  https
863/tcp   open  unknown
2049/tcp  open  nfs
6000/tcp  open  X11
46980/tcp open  unknown
59365/tcp open  unknown
60516/tcp open  unknown
MAC Address: 08:00:27:7E:1A:EB (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 4.67 seconds
```

5. nmap 10.0.2.4 -sS -v: El flag “-sS” nos permite hacer un escaneo TCP de tipo SYN. Esto, también llamado escaneo “silencioso”, hace que se escanee la petición del cliente y la respuesta del servidor.

Llamamos HANDSHAKE al proceso en el que un cliente manda un paquete de datos SYN al servidor, el servidor contesta con otro paquete y un ACK con el número de secuencia SYN más uno y, finalmente, cuando el cliente recibe esto manda un ACK sumando uno al número de secuencia.

En el caso de “-sS” solo tiene en cuenta el envío y la respuesta, pero no el ACK final así que no cubre el HANDSHAKE entero. En cuanto a “-sT” el escaneo es completo, al contrario de “-sS” si que escanea el HANDSHAKE entero. El siguiente esquema ilustra toda esta información.



6. `nmap 10.0.2.4 -n -p80 -sU`: El flag “-sU” nos permite hacer un escaneo completo de tipo UDP, los escaneos tardan bastante más porque en UDP el cliente se intenta comunicar de todas las maneras posibles y, si nadie le contesta, sobreentiende que el puerto está cerrado. En el caso de TCP tarda menos porque cuando llegamos al punto en el que el Servidor devuelve el SYN + ACK, ya sabemos si el puerto está abierto o cerrado.

El flag “-p80” nos permite hacer un escaneo solo del puerto 80 y el flag “-n” le dice a nmap que, si ya tenemos la IP, no haga nunca resolución de nombres DNS.

Con este comando analizamos el puerto de tipo UDP 80 de Alice desde Mallet, es decir el puerto http (navegación web sin seguridad) que, como vemos, está cerrado.

```

mallet@mallet:~$ sudo nmap 10.0.2.4 -n -p80 -sU
Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-15 23:23 CEST
Interesting ports on 10.0.2.4:
PORT      STATE      SERVICE
80/udp    closed    http
MAC Address: 08:00:27:7E:1A:EB (Cadmus Computer Systems)
Nmap done: 1 IP address (1 host up) scanned in 0.08 seconds
  
```

7. `nmap www.uva.es -p80 -sV`: La opción “-sV” nos permite obtener la versión del servicio que pasa por ese puerto para poder estudiar sus exploits.

```
mallet@mallet:~$ sudo nmap www.uva.es -p80 -sV
Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-15 23:15 CEST
Interesting ports on www.uva.es (157.88.25.8):
PORT      STATE SERVICE VERSION
80/tcp    open  http?
1 service unrecognized despite returning data. If you know the service/version,
please submit the following fingerprint at http://www.insecure.org/cgi-bin/servi
cefp-submit.cgi :
SF-Port80-TCP:V=5.00%I=7%D=9/15%Time=632395E9%P=i686-pc-linux-gnu%r(GetReq
SF:uest,1E9,"HTTP/1.1\x20301\x20Moved\x20Permanently\r\nServer:\x20nginx\
SF:\r\nDate:\x20Thu,\x2022\x20Sep\x202022\x2011:56:49\x20GMT\r\nContent-Typ
SF:e:\x20text/html\r\nContent-Length:\x20178\r\nConnection:\x20close\r\nLo
SF:cation:\x20https://admission.uva.es/\r\nCache-Control:\x20private\r\nX-
SF:-Frame-Options:\x20SAMEORIGIN\r\nX-XSS-Protection:\x201;\x20mode=block\
SF:\r\nSet-Cookie:\x20Path=/;\x20HttpOnly;\x20Secure\r\n\r\n<html>\r\n<head
SF:><title>301\x20Moved\x20Permanently</title></head>\r\n<body\x20bgcolor=
SF: \x20white>\r\n<center><h1>301\x20Moved\x20Permanently</h1></center>\r\n
SF:<hr><center>nginx</center>\r\n</body>\r\n</html>\r\n")%r(HTTPOptions,1E
SF:9,"HTTP/1.1\x20301\x20Moved\x20Permanently\r\nServer:\x20nginx\r\nDate
SF::\x20Thu,\x2022\x20Sep\x202022\x2011:56:49\x20GMT\r\nContent-Type:\x20t
SF:ext/html\r\nContent-Length:\x20178\r\nConnection:\x20close\r\nLocation:
SF:\x20https://admission.uva.es/\r\nCache-Control:\x20private\r\nX-Frame-
SF:Options:\x20SAMEORIGIN\r\nX-XSS-Protection:\x201;\x20mode=block\r\nSet-
```

8. nmap por defecto hace un escaneo de forma aleatoria, es decir, va mirando los puertos “saltando” de uno a otro aleatoriamente, no de manera secuencial. Para que nmap se haga de manera secuencial existe el flag “-r”.

Active un proceso de monitorización tcp en mallet para poder seguir los diferentes métodos de scanning.

Para realizar esto utilizaremos tcpdump:

Tcpdump es una herramienta de monitorización para ver en tiempo real los paquetes que pasan por una red a la que el host está conectado. Usaremos los siguientes flags de tcpdump:

- “-i”: Se utiliza para fijar una interfaz a la que queramos monitorizar.
- “-v”: Aumente la información mostrada de los paquetes capturados.
- “port”: Para indicar el puerto en el que nos queremos fijar.
- “-e”: Imprime la cabecera link-level en cada línea. Se puede usar para obtener, por ejemplo, el numero MAC.

Vamos a hacer una monitorización de la tarjeta de red de mallet mientras hacemos un nmap SYN a alic3 en otra consola como ejemplo para mostrar el funcionamiento de esta herramienta.

```

mallet@mallet:~$ nmap 10.0.2.4 -n -p80 -sS
You requested a scan type which requires root privileges.
QUITTING!
mallet@mallet:~$ sudo nmap 10.0.2.4 -n -p80 -sS
[sudo] password for mallet:

Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-15 23:08 CEST
Interesting ports on 10.0.2.4:
PORT      STATE SERVICE
80/tcp    open  http
MAC Address: 08:00:27:7E:1A:EB (Cadmus Computer Systems)
Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds

```

```

mallet@mallet: ~
File Edit View Terminal Help
mallet@mallet:~$ sudo tcpdump -i eth4 tcp -v port 80 -e
tcpdump: listening on eth4, link-type EN10MB (Ethernet), capture size 96 bytes
23:09:58.122464 08:00:27:6c:2a:f4 (oui Unknown) > 08:00:27:7e:1a:eb (oui Unknown
), ethertype IPv4 (0x0800), length 58: (tos 0x0, ttl 58, id 32730, offset 0, fla
gs [none], proto TCP (6), length 44)
  mallet.local.52784 > alice.local.www: Flags [S], cksum 0x7c16 (correct), seq
2903473271, win 3072, options [mss 1460], length 0
23:09:58.122741 08:00:27:7e:1a:eb (oui Unknown) > 08:00:27:6c:2a:f4 (oui Unknown
), ethertype IPv4 (0x0800), length 60: (tos 0x0, ttl 64, id 0, offset 0, flags [
DF], proto TCP (6), length 44)
  alice.local.www > mallet.local.52784: Flags [S.], cksum 0x2738 (correct), se
q 2026688816, ack 2903473272, win 5840, options [mss 1460], length 0
23:09:58.122746 08:00:27:6c:2a:f4 (oui Unknown) > 08:00:27:7e:1a:eb (oui Unknown
), ethertype IPv4 (0x0800), length 54: (tos 0x0, ttl 64, id 0, offset 0, flags [
DF], proto TCP (6), length 40)
  mallet.local.52784 > alice.local.www: Flags [R], cksum 0x9fcf (correct), seq
2903473272, win 0, length 0

```

Si nos fijamos podemos ver que estamos haciendo el escaneo al puerto 80 TCP de alice (www) desde el puerto 52784 de mallet. Esto es debido a que el puerto especificado en el nmap se refiere al puerto del servidor, pero el puerto del cliente no se especifica, no tiene por qué ser el 80.

Podemos observar cómo se realizan los dos primeros pasos del HANDSHAKE. Manda primero un SYN [S] desde mallet a alice, un SYN con un ACK [S.] desde alice a mallet y el reset [R] desde mallet a alice que termina el nmap. Esto coincide con el esquema explicado antes, si hubiésemos hecho un “-sT” se vería como se realiza el HANDSHAKE entero.

Usando nmap, realice y documente un barrido de puertos UDP en alice y compare con los datos que se obtienen a través de tcpdump.

No es posible hacer un barrido completo de todos los puertos UDP de alice debido al elevado coste temporal que tiene por los motivos hablados anteriormente. Aún que no podamos hacer el barrido completo (se haría con el comando “**sudo nmap 10.0.2.4 -n -p- -sU**”). El problema es que tenemos que comparar un barrido TCP con un UDP. Para comparar esto vamos a hacer un

barrido TCP y un UDP al puerto 80 (http) y uno UDP al 67(dhcp) y, desde otra consola, hacemos un nmap al mismo puerto. Como resultado obtenemos lo siguiente.

Nmap TCP al puerto 80:

```
mallet@mallet:~$ sudo nmap 10.0.2.4 -n -p80 -sS
[sudo] password for mallet:

Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-15 23:08 CEST
Interesting ports on 10.0.2.4:
PORT      STATE SERVICE
80/tcp    open  http
MAC Address: 08:00:27:7E:1A:EB (Cadmus Computer Systems)
Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
```

Nmap de UDP al puerto 80:

```
mallet@mallet:~$ sudo nmap 10.0.2.4 -n -p80 -sU

Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-16 08:59 CEST
Interesting ports on 10.0.2.4:
PORT      STATE SERVICE
80/udp    closed http
MAC Address: 08:00:27:7E:1A:EB (Cadmus Computer Systems)
Nmap done: 1 IP address (1 host up) scanned in 0.08 seconds
```

Resultado del barrido TCP:

```
mallet@mallet:~$ sudo tcpdump -i eth4 tcp port 80 -v
[sudo] password for mallet:
tcpdump: listening on eth4, link-type EN10MB (Ethernet), capture size 96 bytes
06:52:23.202924 IP (tos 0x0, ttl 47, id 29712, offset 0, flags [none], proto TCP (6), length 44)
    mallet.local.34165 > alice.local.www: Flags [S], cksum 0xe959 (correct), seq 3496226970, win 4096, options [mss 1460], length 0
06:52:23.203187 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 44)
    alice.local.www > mallet.local.34165: Flags [S.], cksum 0xdd7 (correct), seq 2070317370, ack 3496226971, win 5840, options [mss 1460], length 0
06:52:23.203198 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 40)
    mallet.local.34165 > alice.local.www: Flags [R], cksum 0x1113 (correct), seq 3496226971, win 0, length 0
S^C
3 packets captured
3 packets received by filter
0 packets dropped by kernel
```

Resultado de barrido UDP:

```
mallet@mallet:~$ sudo tcpdump -i eth4 udp port 80 -v
[sudo] password for mallet:
tcpdump: listening on eth4, link-type EN10MB (Ethernet), capture size 96 bytes
09:03:12.790426 IP (tos 0x0, ttl 50, id 6556, offset 0, flags [none], proto UDP
(17), length 28)
    mallet.local.43411 > alice.local.www: UDP, length 0
^C
1 packets captured
1 packets received by filter
0 packets dropped by kernel
```

Como podemos observar, el barrido UDP se queda esperando a que le conteste el servidor, cosa que no va a pasar porque el puerto está cerrado, debido a esto tardan más los barridos UDP. En el caso del barrido TCP, como ya hemos hablado anteriormente cuando el cliente recibe la respuesta del servidor ya sabemos si está abierto o cerrado, en este caso el puerto está abierto, pero en el caso de estar cerrado, tardaría un tiempo similar porque el servidor envía un Reset.

A continuación, un resultado de la monitorización de un puerto TCP (concretamente el 67) cerrado. Como se puede observar, se manda un SYN desde mallet a alice y el alice devuelve un Reset con un ACK ya que está cerrado:

```
mallet@mallet:~$ sudo tcpdump -i eth4 tcp port 67 -v
[sudo] password for mallet:
tcpdump: listening on eth4, link-type EN10MB (Ethernet), capture size 96 bytes
10:01:49.851256 IP (tos 0x0, ttl 45, id 2651, offset 0, flags [none], proto TCP
(6), length 44)
    mallet.local.60870 > alice.local.bootps: Flags [S], cksum 0x0e54 (correct),
seq 3944714400, win 2048, options [mss 1460], length 0
10:01:49.851481 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6),
length 40)
    alice.local.bootps > mallet.local.60870: Flags [R.], cksum 0x2dfd (correct),
seq 0, ack 3944714401, win 0, length 0
```

A este coste temporal de los escaneos UDP se le suma que UDP siempre intenta contactar con el servidor de todas las maneras posibles. A continuación, un ejemplo con el puerto UDP 53 (correspondiente al DNS, cerrado) de que se mandan muchos mensajes al servidor intentando que este conteste:

Nmap UDP puerto 53:

```
mallet@mallet:~$ sudo nmap 10.0.2.4 -n -p53 -sU
[sudo] password for mallet:

Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-16 10:24 CEST
Interesting ports on 10.0.2.4:
PORT      STATE SERVICE
53/udp    closed domain
MAC Address: 08:00:27:7E:1A:EB (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 0.09 seconds
```


Resultados del barrido UDP del puerto 53:

```

mallet@mallet:~$ sudo tcpdump -i eth4 udp port 53 -v
[sudo] password for mallet:
tcpdump: listening on eth4, link-type EN10MB (Ethernet), capture size 96 bytes
10:24:54.378206 IP (tos 0x0, ttl 54, id 29219, offset 0, flags [none], proto UDP (17), length 28)
  mallet.local.41359 > alice.local.domain: [|domain]
10:24:54.381056 IP (tos 0x0, ttl 64, id 56776, offset 0, flags [DF], proto UDP (17), length 67)
  mallet.local.45896 > 212.230.135.1.domain: 47287+ PTR? 4.2.0.10.in-addr.arpa. (39)
10:24:54.391718 IP (tos 0x0, ttl 255, id 4, offset 0, flags [none], proto UDP (17), length 144)
  212.230.135.1.domain > mallet.local.45896: 47287 NXDomain 0/1/0 (116)
10:24:54.493452 IP (tos 0x0, ttl 64, id 56805, offset 0, flags [DF], proto UDP (17), length 67)
  mallet.local.34980 > 212.230.135.1.domain: 52639+ PTR? 5.2.0.10.in-addr.arpa. (39)
10:24:54.504462 IP (tos 0x0, ttl 255, id 5, offset 0, flags [none], proto UDP (17), length 144)
  212.230.135.1.domain > mallet.local.34980: 52639 NXDomain 0/1/0 (116)
10:24:54.606754 IP (tos 0x0, ttl 64, id 56833, offset 0, flags [DF], proto UDP (17), length 72)
  mallet.local.60773 > 212.230.135.1.domain: 55493+ PTR? 1.135.230.212.in-addr.arpa. (44)
10:24:54.614217 IP (tos 0x0, ttl 255, id 6, offset 0, flags [none], proto UDP (17), length 137)
  212.230.135.1.domain > mallet.local.60773: 55493 NXDomain 0/1/0 (109)
^C
7 packets captured
7 packets received by filter
0 packets dropped by kernel

```

Documente toda la información sobre el servicio web que se ejecuta en alice usando una simple conexión telnet.

Realizaremos la conexión telnet simple desde mallet al puerto 80 de alice. Si lanzamos el comando simplemente la conexión terminará cortándose pero si, cuando estamos conectados al telnet, escribimos HEAD nos mostrará la información del HTML en el que se incluye información interesante como la tecnología que usa el sistema. Obtenemos el siguiente resultado:

```

mallet@mallet:~$ telnet 10.0.2.4 80
Trying 10.0.2.4...
Connected to 10.0.2.4.
Escape character is '^]'.
HEAD
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
<hr>
<address>Apache/2.2.14 (Ubuntu) Server at 127.0.0.1 Port 80</address>
</body></html>
Connection closed by foreign host.

```

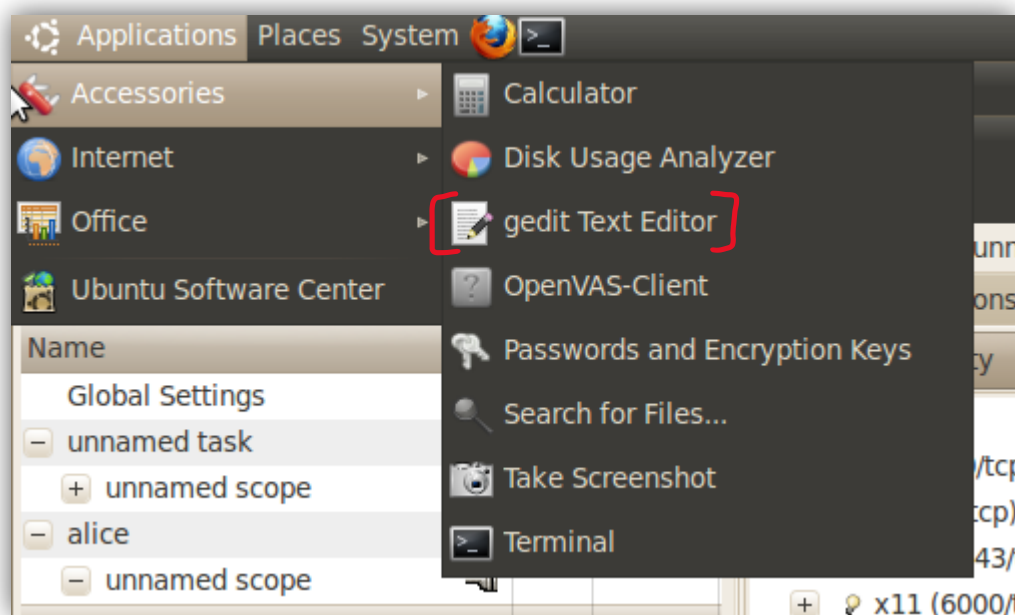
Elabore un informe de vulnerabilidades de alicé usando el analizador de barrido openvas. Documente cómo debe iniciarse este analizador y los resultados que encuentre.

Openvas es un software dedicado a integrar herramientas especializadas en el escaneo y gestión de vulnerabilidades de seguridad de sistemas. Primero vamos a comprobar el estado del servidor de Openvas para ver si está iniciado o no:

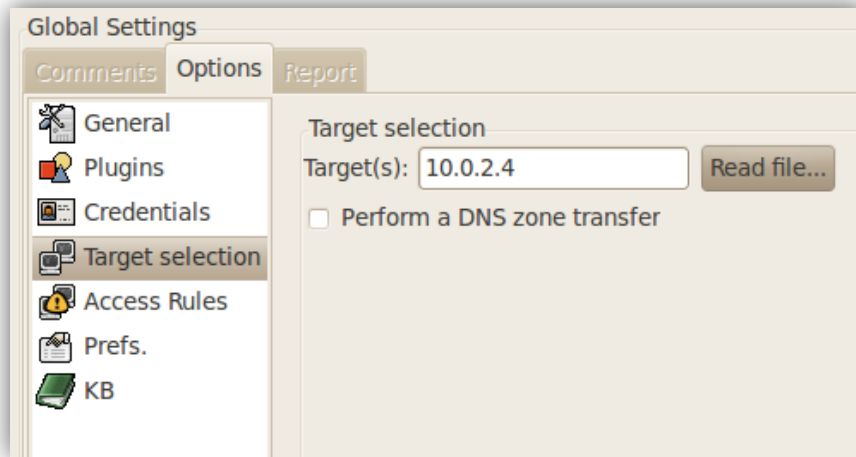
```
mallet@mallet:~$ sudo /etc/init.d/openvas-server status  
OpenVAS daemon is running
```

En mi caso el servidor está iniciado pero, si el servidor no estuviera iniciado, habría que usar el comando **“sudo /etc/init.d/openvas-server start”**.

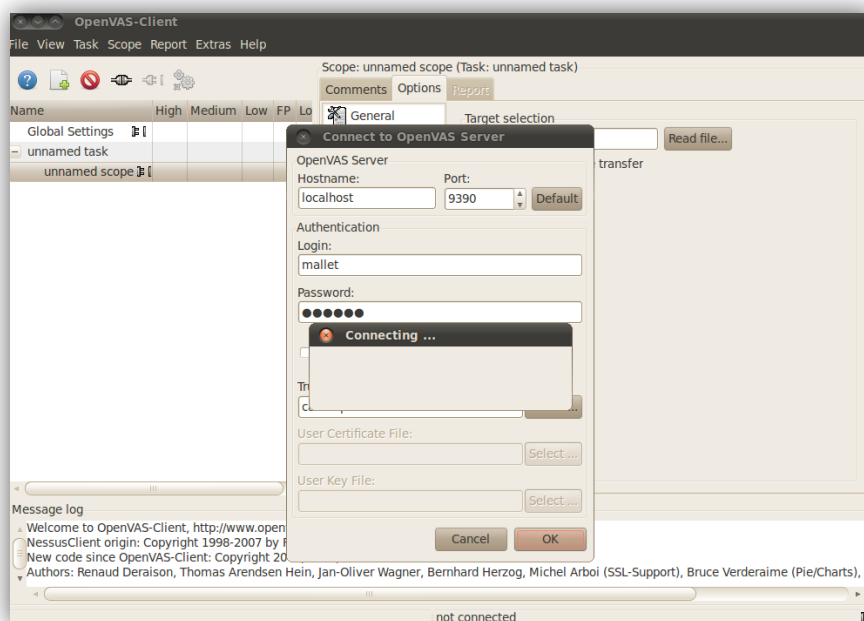
Tras esto iniciamos la propia interfaz de openvas con el comando “openvas-client” o en la siguiente ventana de la máquina:



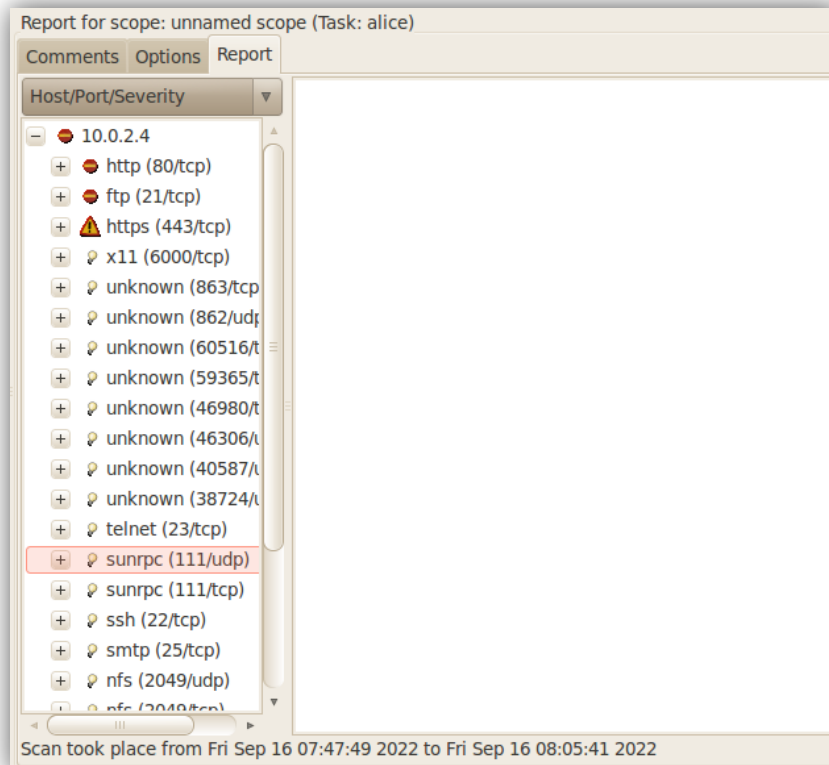
Vamos a target selection y ponemos la ip de alicé:



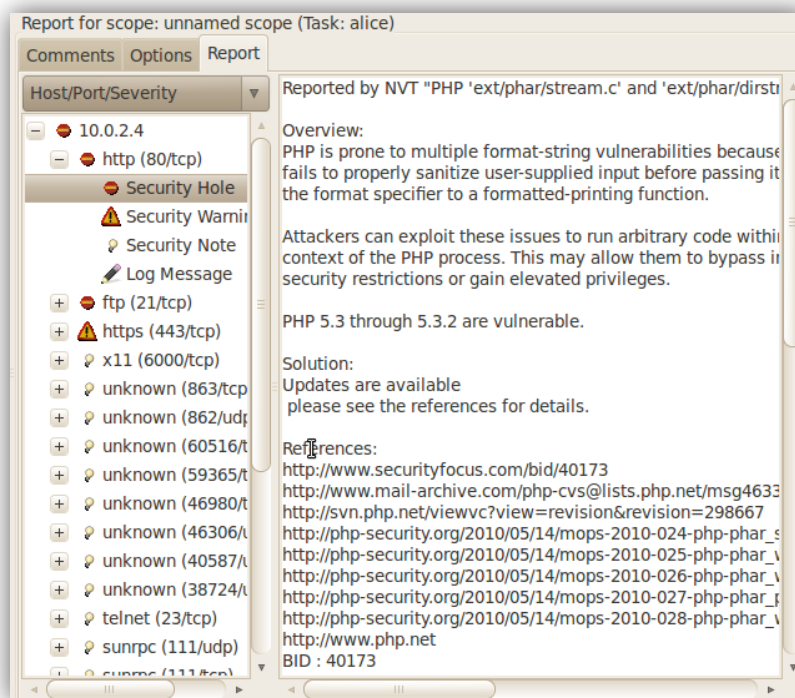
Tras iniciar una nueva tarea y un nuevo scope en el menú superior, en el scope le damos a “Connect” para conectar el sistema, ponemos la contraseña de mallet:



Vamos a Scope y pulsamos “execute”. Esto hará que comience el análisis de alic. Después de un buen rato obtenemos los siguientes resultados:



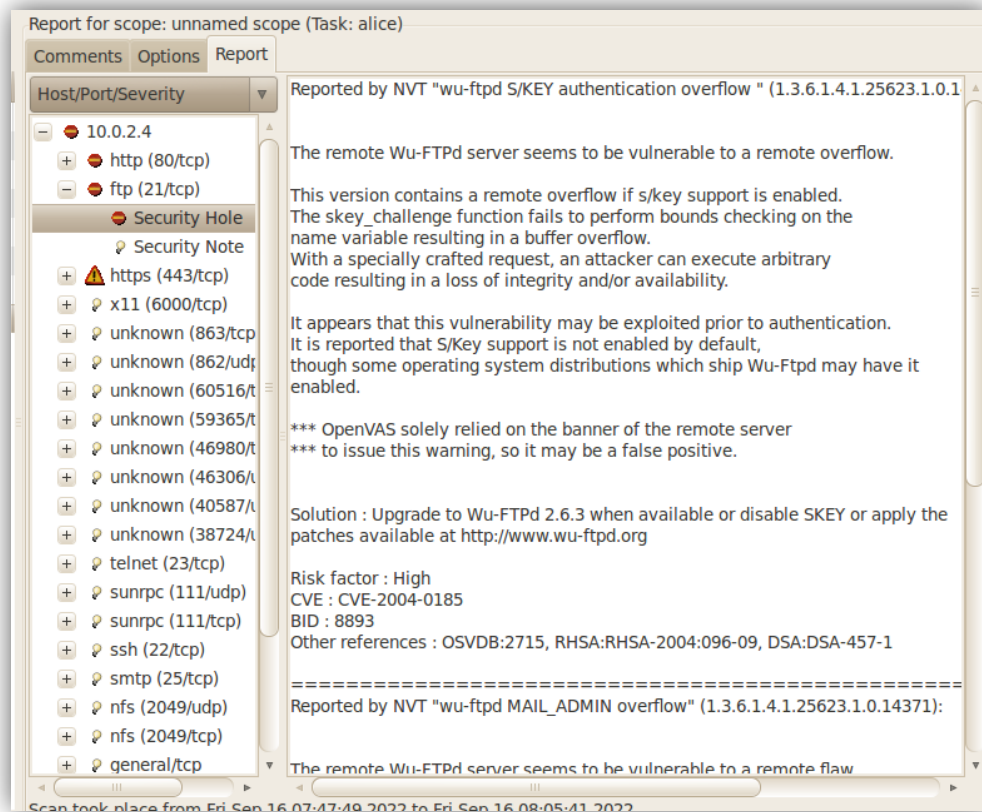
Hemos encontrado varias vulnerabilidades como, por ejemplo, una en el puerto 80 TCP (http):



Este reporte nos habla de que el PHP es propenso a ciertas vulnerabilidades relacionadas con la correcta limpieza de la entrada de las cadenas de formato

introducidas por el usuario y que, todo esto, puede permitir a atacantes eludir restricciones de seguridad.

La dicha anteriormente no es la única vulnerabilidad encontrada:



En este reporte nos habla de que la función `skey_challenge` del servicio `ftp` (puerto 21 TCP) no realiza comprobación de límites y esto provoca un desbordamiento de búfer que puede ser explotada por un atacante.

Recopile la información anterior (desde el punto "b" al punto "e"), ahora para bob en lugar de alice.

- Este apartado es igual que el b con `mallet`, activamos el `tcpdump` para escanear el puerto 80
- Al realizar el `nmap` al puerto 80 UDP de nuevo podemos observar que este también está cerrado en bob.

```
mallet@mallet:~$ sudo nmap 10.0.2.15 -n -p80 -sU

Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-16 04:57 CEST
Interesting ports on 10.0.2.15:
PORT      STATE SERVICE
80/udp    closed  http
MAC Address: 08:00:27:71:55:B0 (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
```

En el tcpdump vemos información parecida que la que vimos para Alice. Seguimos viendo que bob no contesta.

```
mallet@mallet:~$ sudo tcpdump -i eth4 udp -v port 80 -e
tcpdump: listening on eth4, link-type EN10MB (Ethernet), capture size 96 bytes
12:38:22.047949 08:00:27:6c:2a:f4 (oui Unknown) > 08:00:27:71:55:b0 (oui Unknown
), ethertype IPv4 (0x0800), length 42: (tos 0x0, ttl 38, id 51691, offset 0, fla
gs [none], proto UDP (17), length 28)
  mallet.local.55323 > 10.0.2.15.www: UDP, length 0
^C
1 packets captured
1 packets received by filter
0 packets dropped by kernel
```

Respecto al puerto 80 TCP obtenemos lo siguiente en la monitorización:

```
mallet@mallet:~$ sudo tcpdump -i eth4 tcp -v port 80 -e
tcpdump: listening on eth4, link-type EN10MB (Ethernet), capture size 96 bytes
12:41:58.737443 08:00:27:6c:2a:f4 (oui Unknown) > 08:00:27:71:55:b0 (oui Unknown
), ethertype IPv4 (0x0800), length 74: (tos 0x0, ttl 64, id 39894, offset 0, fla
gs [DF], proto TCP (6), length 60)
  mallet.local.39017 > 10.0.2.15.www: Flags [S], cksum 0xaddf (correct), seq 3
978733236, win 5840, options [mss 1460,sackOK,TS val 21511522 ecr 0,nop,wscale 5
], length 0
12:41:58.737684 08:00:27:71:55:b0 (oui Unknown) > 08:00:27:6c:2a:f4 (oui Unknown
), ethertype IPv4 (0x0800), length 74: (tos 0x0, ttl 64, id 0, offset 0, flags [
DF], proto TCP (6), length 60)
  10.0.2.15.www > mallet.local.39017: Flags [S.], cksum 0xe52e (correct), seq
1412300299, ack 3978733237, win 5792, options [mss 1460,sackOK,TS val 12416474 e
cr 21511522,nop,wscale 4], length 0
12:41:58.737692 08:00:27:6c:2a:f4 (oui Unknown) > 08:00:27:71:55:b0 (oui Unknown
), ethertype IPv4 (0x0800), length 66: (tos 0x0, ttl 64, id 39895, offset 0, fla
gs [DF], proto TCP (6), length 52)
  mallet.local.39017 > 10.0.2.15.www: Flags [.], cksum 0x29e1 (correct), ack 1
, win 183, options [nop,nop,TS val 21511522 ecr 12416474], length 0
12:41:58.737963 08:00:27:6c:2a:f4 (oui Unknown) > 08:00:27:71:55:b0 (oui Unknown
), ethertype IPv4 (0x0800), length 66: (tos 0x0, ttl 64, id 39896, offset 0, fla
gs [DF], proto TCP (6), length 52)
  mallet.local.39017 > 10.0.2.15.www: Flags [R.], cksum 0x29dd (correct), seq
```

Como podemos ver se hace el HANDSHAKE completo, a partir de esto (sin ver el nmap) podemos deducir tres cosas:

- No he usado el mismo comando que para analizar a alice (precisamente para que se vea el uso de un comando con flags distintas)
- El puerto está abierto
- Hemos hecho un escaneo TCP completo

```
mallet@mallet:~$ sudo nmap 10.0.2.15 -n -p80 -sT
Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-16 12:41 CEST
Interesting ports on 10.0.2.15:
PORT      STATE SERVICE
80/tcp    open  http
MAC Address: 08:00:27:71:55:B0 (Cadmus Computer Systems)
Nmap done: 1 IP address (1 host up) scanned in 0.08 seconds
```


A continuación, monitorizaremos un escáner completo TCP al puerto 67 (cerrado). Nmap:

```
mallet@mallet:~$ sudo nmap 10.0.2.15 -n -p67 -sT
Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-16 12:49 CEST
Interesting ports on 10.0.2.15:
PORT      STATE      SERVICE
67/tcp    closed    dhcps
MAC Address: 08:00:27:71:55:B0 (Cadmus Computer Systems)
Nmap done: 1 IP address (1 host up) scanned in 0.08 seconds
```

Resultados de la monitorización del puerto TCP 67 en bob:

```
mallet@mallet:~$ sudo tcpdump -i eth4 tcp -v port 67 -e
tcpdump: listening on eth4, link-type EN10MB (Ethernet), capture size 96 bytes
12:49:57.250088 08:00:27:6c:2a:f4 (oui Unknown) > 08:00:27:71:55:b0 (oui Unknown), ethertype IPv4 (0x0800), length 74: (tos 0x0, ttl 64, id 56689, offset 0, flags [DF], proto TCP (6), length 60)
    mallet.local.47833 > 10.0.2.15.bootps: Flags [S], cksum 0xd11f (correct), seq 2903166431, win 5840, options [mss 1460,sackOK,TS val 21631150 ecr 0,nop,wscale 5], length 0
12:49:57.250274 08:00:27:71:55:b0 (oui Unknown) > 08:00:27:6c:2a:f4 (oui Unknown), ethertype IPv4 (0x0800), length 60: (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 40)
    10.0.2.15.bootps > mallet.local.47833: Flags [R.], cksum 0x61b5 (correct), seq 0, ack 2903166432, win 0, length 0
```

Podemos observar que, si el puerto está cerrado, da igual que el escaneo sea completo o “silencioso”, ya que el servidor va a devolver un Reset con un ACK en los dos casos.

Nmap UDP al puerto 53 de bob:

```
mallet@mallet:~$ sudo nmap 10.0.2.15 -n -p53 -sU
Starting Nmap 5.00 ( http://nmap.org ) at 2022-09-16 12:58 CEST
Interesting ports on 10.0.2.15:
PORT      STATE      SERVICE
53/udp    closed    domain
MAC Address: 08:00:27:71:55:B0 (Cadmus Computer Systems)
Nmap done: 1 IP address (1 host up) scanned in 0.09 seconds
```

Monitorización del escaneo:

```

mallet@mallet:~$ sudo tcpdump -i eth4 udp -v port 53 -e
tcpdump: listening on eth4, link-type EN10MB (Ethernet), capture size 96 bytes
12:58:35.834156 08:00:27:6c:2a:f4 (oui Unknown) > 08:00:27:71:55:b0 (oui Unknown
), ethertype IPv4 (0x0800), length 42: (tos 0x0, ttl 40, id 49110, offset 0, fla
gs [none], proto UDP (17), length 28)
mallet.local.47531 > 10.0.2.15.domain: [|domain]
12:58:35.836988 08:00:27:6c:2a:f4 (oui Unknown) > 52:54:00:12:35:00 (oui Unknown
), ethertype IPv4 (0x0800), length 82: (tos 0x0, ttl 64, id 2844, offset 0, flag
s [DF], proto UDP (17), length 68)
mallet.local.33008 > 212.230.135.1.domain: 1101+ PTR? 15.2.0.10.in-addr.arpa
. (40)
12:58:35.852933 52:54:00:12:35:00 (oui Unknown) > 08:00:27:6c:2a:f4 (oui Unknown
), ethertype IPv4 (0x0800), length 159: (tos 0x0, ttl 255, id 816, offset 0, fla
gs [none], proto UDP (17), length 145)
212.230.135.1.domain > mallet.local.33008: 1101 NXDomain 0/1/0 (117)
12:58:40.855589 08:00:27:6c:2a:f4 (oui Unknown) > 52:54:00:12:35:00 (oui Unknown
), ethertype IPv4 (0x0800), length 81: (tos 0x0, ttl 64, id 4099, offset 0, flag
s [DF], proto UDP (17), length 67)
mallet.local.53386 > 212.230.135.1.domain: 31983+ PTR? 5.2.0.10.in-addr.arpa
. (39)
12:58:40.880182 52:54:00:12:35:00 (oui Unknown) > 08:00:27:6c:2a:f4 (oui Unknown
), ethertype IPv4 (0x0800), length 158: (tos 0x0, ttl 255, id 817, offset 0, fla
gs [none], proto UDP (17), length 144)
212.230.135.1.domain > mallet.local.53386: 31983 NXDomain 0/1/0 (116)
12:58:40.981015 08:00:27:6c:2a:f4 (oui Unknown) > 52:54:00:12:35:00 (oui Unknown
), ethertype IPv4 (0x0800), length 86: (tos 0x0, ttl 64, id 4131, offset 0, flag
s [DF], proto UDP (17), length 72)
mallet.local.33719 > 212.230.135.1.domain: 42594+ PTR? 1.135.230.212.in-addr
.arpa. (44)
12:58:40.990021 52:54:00:12:35:00 (oui Unknown) > 08:00:27:6c:2a:f4 (oui Unknown
), ethertype IPv4 (0x0800), length 151: (tos 0x0, ttl 255, id 818, offset 0, fla
gs [none], proto UDP (17), length 137)
212.230.135.1.domain > mallet.local.33719: 42594 NXDomain 0/1/0 (109)

```

Podemos ver que pasa lo mismo que con Alice, UDP intenta conectar por todas las vías posibles.

- d) En cuanto al telnet podemos observar que bob usa otra versión de Apache distinta a la de Alice la cual tendrá distintos exploits. Adjunto enlace a página web en la que se pueden ver sus exploits:

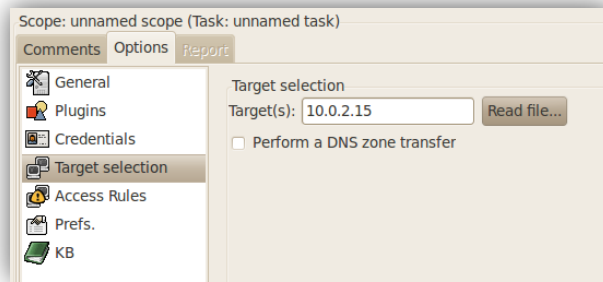
https://www.cvedetails.com/vulnerability-list/vendor_id-45/product_id-66/version_id-416234/Apache-Http-Server-2.2.9.html

```

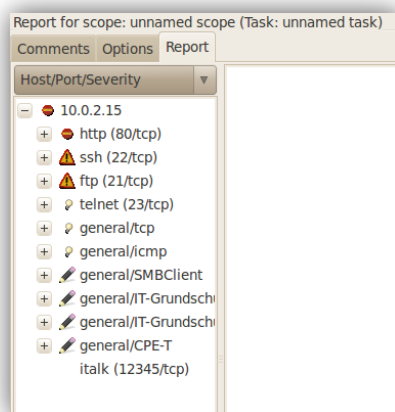
mallet@mallet:~$ telnet 10.0.2.15 80
Trying 10.0.2.15...
Connected to 10.0.2.15.
Escape character is '^]'.
HEAD
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
<hr>
<address>Apache/2.2.9 (Debian) PHP/5.2.6-1+lenny8 with Suhosin-Patch Server at 127.0.1.1 Port 80</address>
</body></html>
Connection closed by foreign host.

```

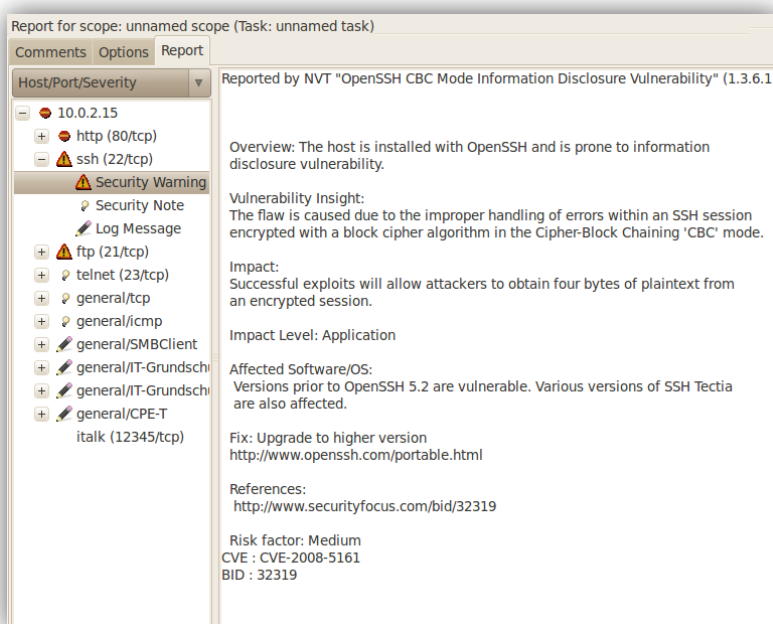
- e) Para monitorizar a bob tenemos que hacer el mismo proceso que para Alice, pero cambiando el target a “10.0.2.15” en vez de “10.0.2.4”.



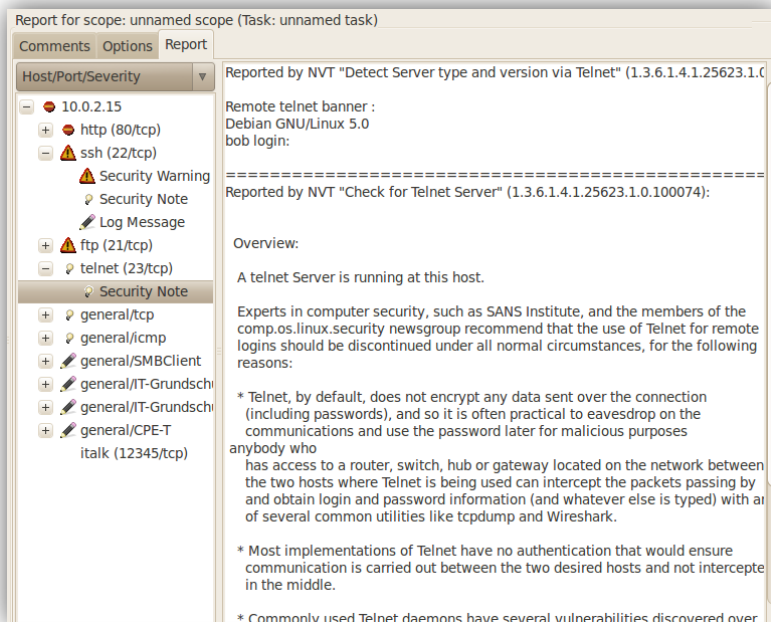
Tras un largo rato escaneando podemos ver los siguientes reportes:



Tenemos, por ejemplo, un reporte sobre mal manejo de errores en sesiones SSH que puede provocar que un atacante en un exploit adecuado obtenga 4 bytes de texto sin formato de una sesión cifrada:



En cuanto a telnet nos informa de que hay un Server de telnet activado. Esto hecho es ya un riesgo, no es nada recomendado por los expertos (SANS institute) el uso de este servicio para login remoto:



En cuanto al puerto 80 TCP nos reporta un "Security hole". Esto es un agujero importante en la seguridad del sistema. Nos reporta varias vulnerabilidades, adjunto captura:

