## PRÁCTICA 2. Análisis de Servicios en Red, Control de Acceso, Gestión de Usuarios

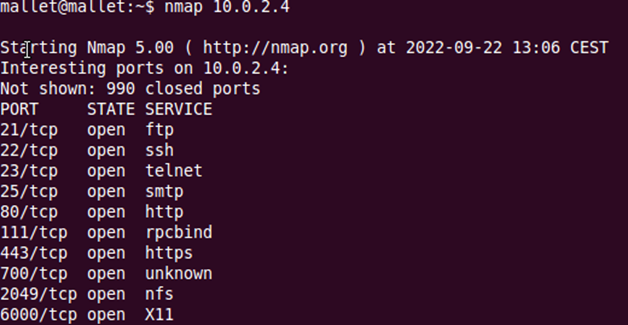
**PRIMERA PARTE**

**a. Usando la página de manual de nmap, documente las funciones y opciones básicas de nmap.**

El comando nmap se utiliza para el mapeo de la red, así como descubrir servicios e identificar puertos abiertos (analizador de puertos).

Existen 2^16 = 65036 puertos (servicios), estos pueden ser TCP o UDP (protocolos de la capa de transporte), y existen 65036 de cada uno de los tipos.

-El comando ***nmap por defecto*** va a analizar los 1000 puertos TCP más utilizados (el resto no son escaneados). Aquí una prueba ejecutando el comando nmap desde la máquina virtual mallet, a alice.



-La opción ***-n*** (nmap 10.0.2.4 -n) hace que nmap no intente adivinar el nombre completamente cualificado de las máquinas, evita que haga resolución inversa de DNS.

-Añadiendo ***-v*** (nmap 10.0.2.4 -v) permite que nmap muestre todo lo que está haciendo cuando hace un descubrimiento de los servicios.

Texto

Descripción generada automáticamente

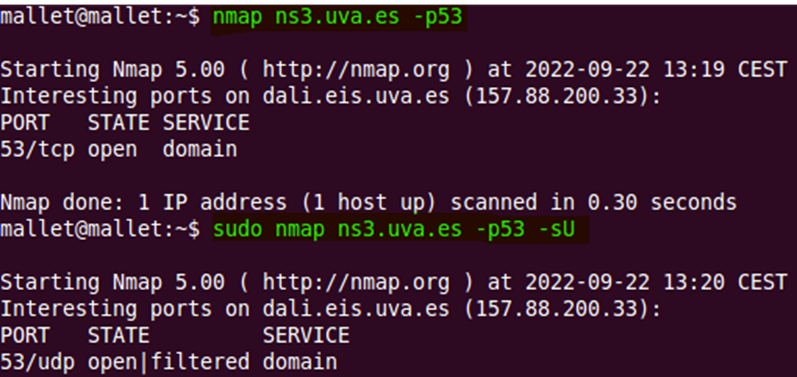
-Para que nmap analice todos los puertos, y no solo los 1000 más utilizados añadimos ***-p-*** (nmap 10.0.2.4 -p-), como se ve ahora 65522 puertos están cerrados.

Texto

Descripción generada automáticamente

-Hasta ahora, solo ha analizado los puertos TCP, pero si queremos que únicamente escanease los UDP añadimos ***-sU*** (nmap 10.0.2.4 -sU). Nos pide requisitos de super usuario, por ello añado sudo antes del comando.

-Si queremos que únicamente escanease un puerto en específico añadimos ***-pX (siendo X el número de un puerto en específico)***, en este caso escaneamos el puerto 53 TCP de ns3.uva.es primero, y después el puerto 53 UDP de ns3.uva.es con -sU, igual que en el punto anterior.



-Para realizar un escaneo del puerto anterior en TCP y UDP a la vez en un único comando bastaría con añadir ***-sT,*** que realiza el escaneo de tipo **TCP completo** (nmap 10.0.2.4 -p53 -sU -sT), como se ve todas las distintas opciones se pueden combinar distintamente para adaptar el escaneo a lo que se quiera.

Texto

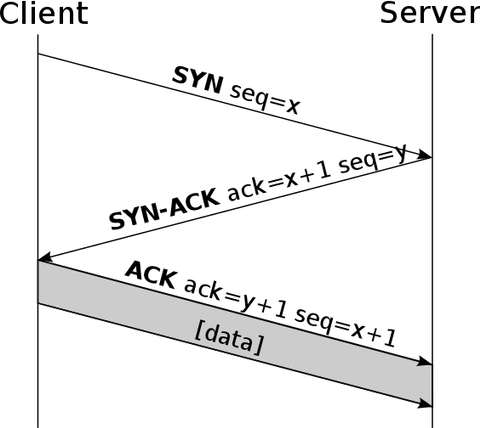
Descripción generada automáticamente

-Ahora otro tipo de escaneo de tipo **TCP, pero ahora de tipo SYN**, añadiendo ***-sS*** (nmap ns3.uva.es -p53 -sU -sS)***,*** este tipo de escaneo es más “sigiloso”.

Texto

Descripción generada automáticamente

Como se ve la salida es la misma pero la forma en la que el cliente y el servidor establecen la comunicación es bastante diferente. [1]



-sS

-sT

-Ahora un para realizar un escaneado de tipo NULO, añadiríamos -***sN,*** que requiere permisos de super usuario(sudo nmap 10.0.2.4 -sN).

Texto

Descripción generada automáticamente

-Y para realizar un escaneado de ACK, usaríamos -sA, que también requiere de permisos de super usuario (sudo nmap 10.0.2.4 -sA).

Texto

Descripción generada automáticamente

-Para conocer la versión de un servicio que ofrece un determinado puerto, utilizamos -***sV***, en este caso combinado con -p puesto que queremos que nos diga la versión de un servicio en concreto en este caso el 80 (nmap 10.0.2.4 -p80 -sV).

Texto

Descripción generada automáticamente

Se obtiene que el puerto 80, es un servicio http, de Apache 2.2.14.

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

Para activar un proceso de monitorización tcp recurrimos a tcpdump, para ello vamos a necesitar permisos de super usuario y proporcionar una serie de opciones al comando. El comando es:

*sudo tcpdump -i eth4*

Donde -i nos permite indicarle la interfaz de red para que la analice, por tanto, antes debemos conocer la red, y sabemos que la red donde están conectadas todas las máquinas es “eth4”.

Primero compruebo si la red eth4 está disponible, con el comando *sudo tcpdump -D*

*Texto

Descripción generada automáticamente*

Y vemos que si lo está. Entonces ahora activo un proceso de monitorización TCP en mallet:





Y para comprobar si funciona hago un ping de alice a bob de 2 paquetes:

Texto

Descripción generada automáticamente

Y observamos en el proceso de monitorización tcp que se captura el ping de alice a bob.

Texto

Descripción generada automáticamente

En caso de que queramos monitorizar un puerto concreto deberíamos de añadir ***port X*** (número del puerto concreto de origen o destino), y para obtener más información ***-e,*** por ejemplo, el comando de a continuación únicamente monitoriza el puerto 80, obteniendo más información de la que se muestra por defecto, además puedes seleccionar si monitorizas solo TCP o UDP, indicando tcp o udp:

*sudo tcpdump -i eth4* ***tcp -v port 80 -e***

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

Para realizar un barrido UDP de puertos completos de alice, pondríamos este comando:



Pero tarda demasiado, por tanto, para hacer un ejemplo de lo que pide el enunciado nos vamos a centrar en un único puerto UDP.

Abrimos dos terminales en mallet en una activaremos el tcpdump (necesitamos permisos de super usuario, puesto que pone la tarjeta de red en modo promiscuo) y en otra haremos el barrido de puertos UDP (lo haremos con el puerto 67), puesto que todos tarda mucho tiempo.

Para activar el tcpdump el comando es: *sudo tcpdump -i eth4*

Y para analizar el puerto 67 UDP en otra terminal ponemos: *sudo nmap 10.0.2.4 -sU -p67*

Observamos esto:

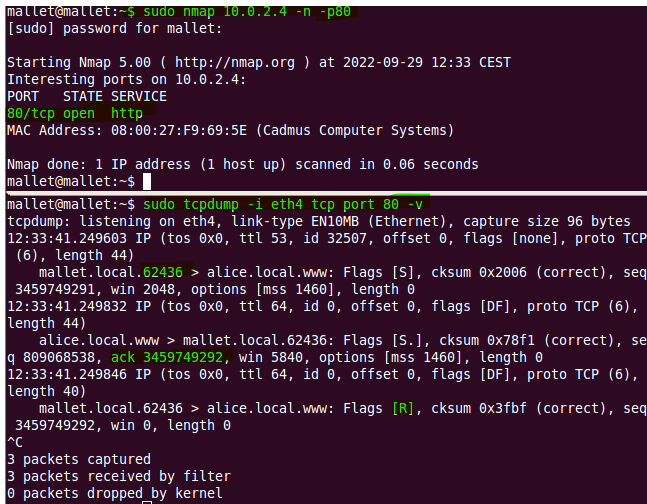
Texto

Descripción generada automáticamente

Vemos que se utiliza bootp, antecesor de DHCP, nmap para saber si el puerto está abierto o cerrado, conoce el servicio y le envía una petición. Si la responde está abierto y si no, está cerrado, por ello realizar un barrido de todos los puertos UDP lleva tanto tiempo, porque nmap se queda esperando una respuesta de estos.

Ahora vamos a realizar un escaneo del puerto 80, mediante TCP, utilizando TCP SYN que es el que utiliza nmap por defecto y utilizando TCP Completo:

Con **TCP SYN:**



Con **TCP Completo:**

Texto

Descripción generada automáticamente

Podemos ver los send, ACK y resets que se envían entre cliente y servidor. Y ver que sigue el esquema del ejercicio 1 [1].

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

Para conocer toda la información sobre el servicio web tenemos que conocer que el puerto HTTP es el 80, pero primero tenemos que ver si el puerto 80 está abierto:

Texto

Descripción generada automáticamente

Y podemos ver que si lo está. Por tanto, podremos ejecutar el comando telnet a este puerto.

El comando queda así: telnet 10.0.2.4 80. Y después deberemos introducir GET / HTTP/1.1 como pone en el CH3-Basin, también podríamos poner HEAD, como hicimos en clase.

Texto

Descripción generada automáticamente

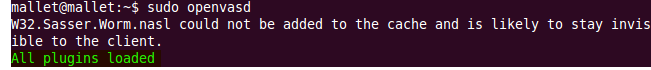
Ahora poniendo i obtenemos un fragmento de código HTML (formato original), de lo que parece ser la página web de alice:

Pantalla de un celular con texto e imagen

Descripción generada automáticamente

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

Primero abrimos un servidor de openvas (analizador de barrido que se pide) en la máquina de mallet, y se nos cargarán todos los plugins:



Primero antes de comenzar a realizar el análisis de vulnerabilidades, debemos de consultar el estado del servidor, y después iniciarlo, ya que sin el servidor “localhost” no podemos comenzar el análisis:

Texto

Descripción generada automáticamente

Y ahora ejecuto un cliente de openvasd, donde se nos abre una nueva ventana con múltiples opciones:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Una vez abierto el cliente, tendremos que crear un barrido/análisis nuevo, para ello en el menú principal de la aplicación, en la parte superior de este New (Icono hoja en blanco), esto nos crea una nueva task y después Scope 🡪 New. Una vez se nos han creado, dentro de “unnamed scope” en la pestaña “Options” de esta, y dentro del submenú en “Target Selection”, escribimos en el campo Target(s):, alice, es decir, el nombre de la máquina a quién queremos escanear, o su IP. Y después ejecutamos el analizador, en la parte superior donde pone “Scope”, “execute”, automáticamente comienza a realizar el análisis:

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Una vez iniciado el servidor ya sí que conecta e inicia el escaneo como se ve a continuación:

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

Una vez generado el análisis de vulnerabilidades el cual habrá tardado unos minutos obtendremos un reporte donde podremos ver todas las vulnerabilidades de alice.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Aquí un ejemplo del reporte de dos de las vulnerabilidades detectadas:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

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

Al igual que en el apartado b) lo primero que hago es comprobar que redes están disponibles para poderla escanear, vemos que eth4 está disponible. Eth4 es la red a la que están conectadas todas las máquinas.

*Texto

Descripción generada automáticamente*

Y vemos que está disponible, por ello para comprobar la monitorización ahora realizaremos un ping de 4 paquetes de bob a alice (en el apartado b, era al revés).

Texto

Descripción generada automáticamente

Y en la monitorización activada en mallet con *sudo tcpdump -i eth4*, escaneando el puerto 67 UDP de bob.

Texto

Descripción generada automáticamente

Igual que hicimos en el apartado c) también vamos a realizar el escaneo del puerto 80 TCP de bob, mediante el escaneo TCP SYN, y TCP COMPLETO.

Por defecto nmap realiza TCP SYN

**TCP SYN:**

**Texto

Descripción generada automáticamente**

**TCP Completo:**

**Texto

Descripción generada automáticamente**

Ahora desde mallet escaneamos el puerto 80 TCP de bob, ya que es el puerto del servicio web http, para ver si está abierto.

Texto

Descripción generada automáticamente

Y ahora con una conexión telnet obtenemos la información de la misma forma que en el apartado:

* Usando HEAD

Texto

Descripción generada automáticamente

* Usando i

Texto

Descripción generada automáticamente

Ahora el informe de vulnerabilidades en bob, para ello de nuevo abriremos un cliente openvas en mallet, pero ahora modificaremos el target (a quién analizar, subrayado de color rosa), y realizando los mismos pasos que en e).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Una vez finalizado el escaneo obtenemos el reporte de las vulnerabilidades de bob:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Aquí una descripción de dos de las vulnerabilidades:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Texto

Descripción generada automáticamente