Iptables Exercise

Transcribed on July 31, 2025 at 12:56 PM by Minutes AI

Speaker 1 (00:01)

Bienvenidos a esta nueva sesión.

En esta sesión vamos a tratar el tema del ejercicio de ipetables que era sobre ataque y defensa.

En esta primera parte lo que vamos a ver es toda la secuencia de ataques y también la implementación de toda la defensa.

Bien, pues vamos a proceder con la resolución del ejercicio y comenzamos identificando las máquinas.

Aquí tengo la primera.

Esta máquina será mi máquina atacante, que tiene esta dirección.

La 10.211.555 será la máquina que va a atacar.

Bien, pues ahora estoy en la otra máquina, en la segunda máquina que será la que tenemos que defender, que tendrá esta dirección IP acaba en 17.

Bien, pues este ejercicio tiene tres fases principales.

Tenemos primero que preparar la máquina atacante.

Segundo, configurar la defensa en la máquina defensora Y finalmente analizar y ajustar todo lo que ha ido ocurriendo durante la fase de ataque y de defensa, es decir, los registros.

Bien, pues volviendo a la máquina que va a actuar como atacante, vamos a preparar los diferentes ataques.

Bien, pues de vuelta a la máquina 1, que será la que voy a utilizar como máquina atacante.

Lo primero es comprobar que puedo ver la máquina número 2, con lo cual voy a hacer un pin 211 a la máquina 5517.

Bien, pues vemos que hay conexión, con lo cual las dos máquinas están en la misma red y se ven entre ellas.

El paso primero tiene que ser instalar las herramientas que necesitamos por el ETCA.

En este caso tendríamos que instalar apt install nmap Seguimos los pasos como siempre decimos que sí.

La siguiente será ssh, que debería estar instalada pero tiene que estar aquí.

Vale, ya la tengo, pero bueno, sería call sudo apt install open ssh client Vale, no lo voy a hacer porque ya lo tengo instalado.

Y por último la herramienta Hydra.

Bien, y ya por último es instalar la aplicación Hydra.

Para ello hacemos un sudo apt install hydra gtk y ya tendremos instalada.

Para comprobarlo podemos poner hydra help y ahí vemos que está instalada.

Bien, ya tenemos las cuatro herramientas que vamos a usar para intentar atacar la máquina 2.

Bien, antes de continuar quería indicaros que he vuelto a la máquina 2, que es la que estamos defendiendo.

Tenemos que levantar un servicio en el puerto 80 para hacer las pruebas, ¿Verdad?

Porque pusimos que una de las condiciones era probar el acceso a una página web que está en ese servidor, con lo cual tenemos que utilizar por ejemplo lo mismo que hicimos en otro ejercicio, que es levantarlo muy sencillo con este comando mhtp server y esto lo que hacía simplemente era levantar un servicio en el puerto, en este caso el 8000, vamos a verlo, si hacemos aquí y vamos a un localhost, pues aquí nos muestra el listado de los directorios.

Bien, pues este será el servicio que vamos a tener que defender desde esta máquina para evitar que no haya ningún tipo de petición desde la máquina atacante.

Bien, pues vamos a volver a la máquina principal, la máquina atacante, y ahora vamos a realizar toda esta secuencia de diferentes ataques para ver qué resultados obtenemos ahora, para después compararlos con los resultados que obtendremos cuando apliquemos todas las técnicas de defensa.

Bien, el nmap p lo que indica es que escanee todos los puertos, el guión después del guión p, esto es una pequeña abreviatura que se pone para que escanee todos los puertos del 1 al 65535, por eso he puesto.

Bien, pues aquí podéis ver un poco el resultado, lo que nos ha mostrado es que sí hay un puerto abierto, el 8000, que es justo el servicio que queremos proteger, con lo cual estamos dando demasiada información, estamos ofreciéndole al atacante información sobre que hay un abierto que es el 8000.

Bien, pues como nmap ya nos ha dicho qué puerto hay disponible o en qué puerto hay un servicio, en este caso el 8000 donde había un HTTP, ya podemos ir directamente con el cur para ver si hay contenido o no, por ese motivo hacemos el comando cur, ponemos la dirección ip que era 10 211 55 17 y ahora veríamos que nos devuelve la página web.

Por este motivo hay que evitar que el usuario pueda ir probando diferentes puertos y sobre todo bloquear el puerto principal que es el que estamos publicando, en nuestro caso es el puerto 8.000.

Bien, pues el siguiente ataque será intentar hacer una conexión SSH contra el servidor, para eso es sencillo, ya lo conocemos, es ssh, es simplemente poner ssh la dirección, el usuario, aquí vamos a suponer que conocemos el usuario por el motivo que sea, hemos conseguido información haciendo algún tipo de osynth, buscando email por ejemplo, y sabemos el usuario que es user y después ya sólo hay que poner la dirección ip 10211 35 17 bien, hay respuesta, veis, está operativo.

Sabemos que ese servidor tiene una conexión SSH, Perfecto, cancelamos Y en este punto ya hemos hecho toda nuestra batería de diferentes ataques.

Sabemos que ese servidor tiene una conexión SSH, Perfecto, cancelamos Y en este punto ya hemos hecho toda nuestra batería de diferentes ataques.

Nos faltaría el último, es utilizar Hydra para ver si realmente podemos acceder con ataques de fuerza bruta.

Eso no lo voy a dejar ejecutando, pero sí vamos a ver cómo podemos dejarlo funcionar y veremos que nadie nos va a parar, que nos va dejar seguir funcionando, a no ser que tenga alguna regla mismo SSH que la suele traer de desconexión a partir de cuatro o cinco intentos.

Pero de todas formas nosotros vamos a implementar una forma con iptables para que sólo nos deje un número muy limitado.

Y bien, para hacerlo un poco más realista y ver cómo funciona bien nuestro iptable separando las diferentes conexiones, vamos a volver al servidor.

Bien, ya estamos aquí y le vamos a quitar a ssh el número de limitación de intentos para que no haya ninguna limitación, así podemos dejar a Hydra funcionar.

Esto no es un escenario real, lo normal es que se ponga un número de intento, pero hay dispositivos que por defecto vienen con un número ilimitado y eso es un gran problema.

Entonces lo que haremos será, aquí en el servidor vamos a abrir el fichero que se llama etc shd config y aquí buscamos una entrada que pone max out tries y vendrá un número, en este caso aquí lo vemos, aquí está, es el max try de aquí, en el cual tú indicas el máximo número de intentos.

Para hacerlo ilimitado podemos quitar este comentario y ponerlo en cero por ejemplo.

Con esto no habrá limitación de conexiones, es decir, nuestro servidor es realmente inseguro.

Reiniciamos el servicio ssh start ssh, lo realizamos y ya está, ya está nuestro servidor preparado.

Bien, pues ahora procedemos al último ataque que es el de Hydra, que es el de fuerza bruta.

Para ello utilizaremos un fichero de contraseñas que yo he creado con un listado que tengo aquí llamado password txt, que podéis ver que son muchas contraseñas, en este caso sólo he puesto 50.

Realmente sólo queremos probar la habilidad de parar ese número de intentos y ver si se consigue acceder.

Y bien, como antes hemos hecho que ssh no ponga límite.

Fijaros lo que puede ocurrir con HYDRA, puede ser que de suerte o de casualidad esa contraseña esté dentro del fichero PASSWORD TXT y pueda acceder, como no hay limitación tendrá intentos ilimitados y eso es un gran problema que tenemos siempre que parar.

Pues bien, para hacer el ataque de Hydra solo tenemos que poner HYDRA L, el usuario lo conocemos que es user, ponemos guión p y ahora PASSWORD xt y ponemos ya la dirección de la máquina, en este caso 10.211.5517.

Esto lo que hará ahora será diferentes conexiones ssh probando con el usuario user pero con diferentes contraseñas, como no hay limitación, si la clave está en ese listado tendrá acceso.

Aquí comienza a hacer el ataque y en breve veremos si tiene o no éxito.

Bien, como podéis ver sí que ha habido éxito, ha habido un intento en el cual se ha podido acceder, fijaros, con la clave batman 123 y ésta está dentro del fichero de password txt, hemos comprobado aquí batman 123, con lo cual fijaros qué sencillo es poder atacar y poder encontrar una contraseña si no hay una limitación de intentos.

Bien, pues en este punto ya hemos visto la salida de todos los intentos de ataques que hemos intentado hacer, con lo cual todos han tenido éxito, todos han podido obtener información de la página web, acceder ssh, los cuatro intentos han funcionado, con lo cual ahora tenemos que proceder a securizar nuestra máquina para que esos cuatro intentos no tengan éxito.

Así que vamos a la máquina que queremos defender.

Bien, pues ya estamos en la máquina que tenemos que defender.

Bien, ahora tenemos que establecer las políticas por defecto, esto es importantísimo ya que esto hay que hacerlo antes de añadir cualquier regla, porque de esta forma nos aseguramos que el comportamiento general del sistema es el que queremos.

Y en este caso lo que vamos a hacer es un sudo unip tables p conectro con INPUT a DROP y lo mismo para IP tables p output, en este caso ACCEPT.

Bien, lo que hace cada uno de ellos, primero el guion PIN DROP lo que hace es establecer la política por defecto de la cadena INPUT, cualquier INPUT es DROP, esto significa que todos los paquetes de datos entrantes que no coincidan con ninguna regla en la cadena INPUT se van a descartar, es decir, si no hay una regla que permite explícitamente un tráfico específico entrante, éste será bloqueado y la otra el guión p output accept Este comando establece la política por defecto de la cadena OutputAcept y esto significa que todos los paquetes de datos salientes serán permitidos a menos que haya una regla que los bloquee explícitamente.

Por este motivo el servidor podrá iniciar conexiones hacia afuera sin restricciones.

Aquí podría poner otra con un forward drop, por ejemplo para la política de reenvío, pero en este caso creo que no es necesario porque esta máquina no está actuando como un router, que es lo normal que se aplicaría con esta regla, es decir, como no nos hace falta reenviar paquetes, no haría falta hacer esta política del forward.

Lo siguiente que tenemos que hacer es permitir el tráfico local, el loopback, pues para eso lo que haremos será utilizar estos dos comandos con iptables sudo iptables a input y le decimos guión interfaz loop loopback en local y le decimos que acepte y también haremos un sudo ip tables a pondremos output output output o loopback j ac en ambos sentidos.

Bien, con esto ya permitimos el tráfico local, Perfecto.

Ahora tenemos que permitir conexiones establecidas y relacionadas, es decir, el tráfico de respuestas a conexiones que han sido iniciadas por el host.

Esto es fundamental para que la máquina funcione normalmente.

Antes pongo el comando y ahora os explicaré paso a paso qué es lo que hace, porque es un poco largo.

Haríamos un sudo y petables, después le diríamos guión a input, perdón mayúsculas input pondríamos guión m y aquí viene la parte larga que es el contract ctstate le pondríamos established related j accept Vale, ¿Que es lo que hemos hecho aquí?

Fijaros, el comando ain y después el guión m contract ctt statestably related Aquí se especifica, digamos, el criterio que le vamos a poner a la regla.

Por ejemplo, el guión m contract indica que se va a utilizar el módulo contract para realizar coincidencias en el estado de la conexión de los paquetes, que son estos dos, el established y el related, ya que especifican que la regla se va a aplicar a los paquetes que estén relacionados con una conexión establecida.

De esta forma lo que hacemos es permitir que los paquetes entrantes que forman parte de una sesión que ya hemos establecido o que está relacionada, se van a permitir siempre por el firewall.

Esta cadena es muy importante porque si no bloquearíamos las conexiones que ya estuvieran establecidas durante cualquier tipo de proceso, con lo cual esta es bastante importante en cuenta en un entorno sobre todo de producción.

Bien, y ahora sí que vamos a ir a las defensas específicas.

La primera, vamos a bloquear el escaneo de puertos, que es lo que se pedía en el enunciado.

Antes hemos visto que sin ningún problema el puerto 80 se mostraba como abierto, así que vamos a ver cómo se podría hacer para evitarlo.

También es un comando un poco largo, pero ahora os lo explicaré.

Vale, el primer comando es este.

Sudor ip tables a input p Esto también lo conocéis, que es el puerto.

Y ahora empezamos con los flags, que aquí, esto es también una de las características de bajo nivel de iptables, porque nos permite utilizar a nivel de flags.

Y aquí pondríamos la flag, que sería sin ack fin lst Y con eso creo que acá fin lst Vale, la tenemos.

Ahora ponemos aquí el st y le ponemos un límite limit s, ahora lo explico todo.

J acept ¿Qué ha pasado aquí?

A partir del guión p tcp.

Perfecto, es la regla que se aplica en los paquetes TCP.

Después el tcp flags SIN ack fin rst Esto especifica la combinación de flags TCP que la regla está buscando.

En este caso está buscando paquetes TCP que tengan el flag rst, o sea el reset establecido.

¿Por qué?

Porque de este modo lo que hace es que permite rechazar paquetes TCP que contienen el flag de reinicio.

En otras palabras, lo que hace este comando completo es que rechaza paquetes TCP que contengan el FLAC, que es el reset, a una tasa máxima de un paquete por segundo.

Por eso ponemos el s.

¿Y esto por qué es útil?

Pues porque los paquetes con el flag rst suelen ser generados por sistemas que reciben una solicitud de conexión a un puerto donde no hay un servicio escuchando.

Y este es el comportamiento típico de herramientas de escaneo de puertos como puede ser nmap.

Al limitar la tasa de respuesta a estos paquetes, se ralentiza significativamente la capacidad de un atacante para determinar qué puertos están abiertos en el sistema.

Seguimos con el segundo comando, también asociado a la misma función.

Hacemos un guion a INPUT también y haremos un guión p tcp otra vez TCP flags Ponemos las mismas SYNC ack j drop Bien, pues está claro, está lo que hace es que descarta cualquier paquete TCP que contenga el flag RST Es decir, si un atacante intenta escanear los puertos del sistema y éste envía paquetes con el FLAC RST, estos paquetes se van a descartar por el cortafuego.

Repito lo de antes, esta es la acción o la actitud típica de un escaneo de puertos.

Parece un poco complicado, pero aquí sólo hay que saber cómo funciona el escaneo de puertos y por eso es la detección de ese FLAC RST.

Bien, pues también lo aplicamos y ya tenemos esta fase hecha.

Siguiente método de protección va a estar orientado a las peticiones HTTP.

Hasta ahora veíamos que con el CURL se podía ver el servidor web que teníamos publicado, que era el que mostraba las diferentes ficheros y carpetas.

Pues vamos a limitar ese acceso.

Bien, pues para ello haremos otra vez sudo ip tables pondremos guión a INPUT p Hasta aquí le conocemos todos, deport, este también lo conocemos, Destination Port, el 80-J.

¿Qué se va a hacer?

Pues un drop, estamos bloqueando ya el acceso al puerto 80.

Y también haremos otro.

Y también haremos otro ya por securizarlo aún más con IP tables y haremos otro input al puerto TCP, pero esta vez al Destination Port 403.

Tenemos otro drop.

Esto es un poco para que.

Para completarlo más.

Bien, pues con esto la parte de peticiones debería estar bloqueada de peticiones HTTP y HTTPs, que ese no lo pedía el enunciado, pero lo he puesto aquí como algo adicional para que tengáis algo un poco más lógico.

La mayoría de páginas web son HTTPs, pero bueno, como tenemos publicada solamente nuestra página web utilizando ese pequeño servidor en el puerto 8000, pues entonces lo haremos bien, sí es cierto, puerto 8000, entonces hay que bloquear es el puerto TCP de Port 8000.

Bueno, ya también dejó el 80.

Si, también dejamos otro bloqueado, ningún problema.

Y así con todos los puertos que queramos ir bloqueando.

En mi caso lo publiqué en el 8000 en vez de 80, como antes pudimos ver en la web.

De hecho si aquí y lo abro, veremos que aparece en el 8000.

Ahí está, perfecto.

Bien, conexiones SSH.

Aquí sólo vamos a permitir las conexiones que confiamos en ella, me refiero a conexiones SSH.

Bien, pues vamos a ello.

Pues hacemos un sudor, sudo ip tables a INPUT p tcp Hasta aquí lo mismo de port.

Veréis que esto ya es un poco más directo.

Puerto 22 ssh guión s y aquí pondríamos solamente la ip legítima.

Bien, aquí como sólo quiero el acceso de las ips que yo crea que son confiables, por supuesto no voy a poner la ip del atacante que es la 10.211.55 5, me inventaré una de momento y pondré esa como permitible, que puede ser 55-10, ya directamente digo que a ésta sí que lo acepte.

Os pongo esto más que nada para que veáis que aquí puedes ir colocando las diferentes ips que necesites como máquina legítima de conexión contra el servidor.

En este caso como ésta no existe, no dejará conectar desde la 5.

Otra cosa importante relacionada con ssh es la limitación de intentos de conexión SSH, pues para eso lo que tenemos que hacer es son algunos comandos.

Vamos a hacerlo y ahora os cuento un poco qué hace cada uno.

El primero es este input p tcp de port m Este sonará que es el mismo contract de antes, pero esta vez utilizamos el CTState a new en vez de el anterior que utilizamos, ahora utilizamos new en vez de el establish utilizamos el new y le metemos un limit aquí limit guión limit, un minuto limit, ahora os cuento todo.

Warst j este es un poco largo, sí, perdón, se me ha ido un guión aquí, ya está, es que es un poco largo el comando y es normal que conect.

Vale, ya está.

Bien, ¿Qué hace este comando?

Fijaros, la única parte que no conocemos es la de m contract state new, igual que antes utiliza el módulo contract, que lo que hace es intentar coincidir con los paquetes que están en un estado de conexión nuevo, por eso el new.

Esto significa que la regla se aplicará sólo a conexiones nuevas iniciadas al puerto 22, porque si hubiera conexiones igual las cortaba y eran legítimas, solo conexiones nuevas.

Bien, y el guión m limit limit one minute limit pars 3.

Esto lo que hace es establecer un límite en la tasa de paquetes que coincidan con la regla, en este caso se permite un máximo de un paquete por minuto con una ráfaga de hasta tres paquetes.

Y esto lo que hace es ayudar a proteger contra ataques de denegación de servicio, ya que limitamos la cantidad de conexiones nuevas ssh que se pueden establecer por minuto.

Aquí tenéis un ejemplo de configuración para prepararnos y defendernos de ataques de delegación de servicio utilizando simplemente iptables.

Y ya ahora tendríamos que confirmar qué haremos con esos paquetes, pues directamente será el drop, ahora lo vemos, p tcp deport deport 22 y haríamos un jdrop solamente para confirmar qué va a pasar con estos paquetes fuera.

Bien, pues en teoría con esto ya hemos protegido los cuatro intentos de ataque.

Este último lo que hacía era protegernos del ataque de fuerza bruta, porque al limitar el registro de conexiones SH o al limitar los intentos de conexiones SSH estamos evitando que se pueda hacer un ataque de fuerza bruta como el que vimos que tuvo éxito y encontró la contraseña.

Bien, pues ya hemos visto algunos de los comandos y aspectos fundamentales para securizar toda esa secuencia de ataques que teníamos en el enunciado del ejercicio y como podéis comprobar, IP tablets es muy versátil.

Esta experiencia práctica lo que hará será mejorar nuestra preparación para desafíos reales en seguridad de la información y también nos dará habilidades esenciales para proteger activos digitales en unos entornos que son cada vez más peligrosos y complejos.

Bien, en el siguiente vídeo, en la parte 2 del ejercicio de la resolución, veremos ya como todo eso que hemos aplicado en este funciona bien y veremos que los ataques ya no tienen ningún tipo de resultado contra nuestra arquitectura.

Llegamos.