Indice

[Linux 2](#_Toc159007074)

[Escaneo Pasivo 2](#_Toc159007075)

[Sniffer 2](#_Toc159007076)

[Arp-scan 6](#_Toc159007077)

[NMAP 7](#_Toc159007078)

[Ping 8](#_Toc159007079)

[Escaneo Activo 9](#_Toc159007080)

[HPING3 9](#_Toc159007081)

[NMAP 9](#_Toc159007082)

[Enumeración 10](#_Toc159007083)

[Explotación 13](#_Toc159007084)

[21/FTP - CVE-2015-3306 13](#_Toc159007085)

[22/SSH- CVE-2018-15919 && CVE-2018-15473 17](#_Toc159007086)

[8080/Jetty 8.1.7 –CVE-2019-10247 19](#_Toc159007087)

[8080/Continuum - CVE-2013-2251 20](#_Toc159007088)

[80/SQL Injection - Get System Users and Generate MySQL Root User 25](#_Toc159007089)

[Ubuntu 14.04 - Privilege Escalation -Throught Docker 30](#_Toc159007090)

[Windows 31](#_Toc159007091)

[Escaneo pasivo 32](#_Toc159007092)

[Escaneo activo 32](#_Toc159007093)

[Enumeración 32](#_Toc159007094)

[Explotación 32](#_Toc159007095)

[22/SSH- CVE-2018-15919 && CVE-2018-15473 32](#_Toc159007096)

[22/SSH – Hydra – Brute Force 36](#_Toc159007097)

[22/SSH - CVE-2016-6515 – DOS 37](#_Toc159007098)

[Conclusión 40](#_Toc159007099)

# Linux

## Escaneo Pasivo

### Sniffer

El escaneo pasivo a través de un sniffer de red tiene como utilidad el conocer un objetivo de ataque sin la necesidad de escanear la red completa de forma agresiva. En un escenario real, se observa que, al escuchar la red, se generan numerosas peticiones de conexión a distintos servicios para una o varias IP específicas. Estas conexiones proporcionan información sobre la importancia de ese punto en la red. Sin embargo, es importante tener en cuenta que no necesariamente se trata de un servidor central que debamos considerar como objetivo principal, ya que también podría tratarse de un firewall o un proxy que intercepte estos paquetes en primer lugar.

Para este ejemplo es complicado realizar un sniffing de red óptimo, ya que disponemos de máquinas que realmente no están realizando conexiones en tiempo real. No tiene clientes activos, por lo que voy a generar peticiones de conexión a distintos servicios (con éxito o no) a fin de visualizar lo que ocurre a nivel de red.

Pero antes que eso, me gustaría mostrar una información interesante que he encontrado haciendo sniffing de mi propia red local:

Lo primero es que para hacer sniffing con adaptador puente existe una opción avanzada llamada “Modo Promiscuo”, que, según fuentes que he encontrado en internet (<https://www.pinguytaz.net/index.php/2016/11/20/virtualbox-configurando-la-red/>) si está desactivado podemos ver los paquetes, pero sin información. Estos se muestran como multicast usando el protocolo mDNS sin otorgar más información. Por lo que yo como quiero analizar mi red lo coloco en “Permitir todo”.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Bien, ya teniendo acceso a mi red observo que constantemente se realizan paquetes con cabeceras PSH desde dos IP públicas concretas (212.145.41.32 y 77.209.227.18) lo que me parece raro, por lo que visito la web de Virustotal para revisar ambas IP, resultando ser direcciones IP limpias pertenecientes a Vodafone, así que tranquilo porque mi conexión a Internet es mediante una red 4G de Vodafone precisamente.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

Patrón de fondo

Descripción generada automáticamente con confianza media

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Patrón de fondo

Descripción generada automáticamente con confianza media

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Como tengo curiosidad decido ejecutar un Whoami desde la máquina Kali con ambas IP, recibiendo el siguiente resultado:

Para la IP 212.145.41.32 obtengo que se trata de un servicio de RIPE Database Quer, que juntándolo con la información de Virustotal sobre Vodafone y una descripción de ChatGPT:

“El servicio de consulta de la base de datos RIPE (RIPE Database Query Service) se refiere a un sistema de consulta que permite acceder a la base de datos mantenida por RIPE NCC (Réseaux IP Européens Network Coordination Centre). RIPE NCC es una organización que se encarga de la asignación y coordinación de recursos de direcciones IP y números de sistemas autónomos en Europa, Oriente Medio y partes de Asia Central.

La base de datos mantenida por RIPE NCC contiene información sobre los recursos de Internet asignados o registrados en su área de responsabilidad, incluyendo direcciones IP, bloques de direcciones IPv4 e IPv6, asignaciones de sistemas autónomos (ASNs), contactos de administración y otros datos relacionados con la infraestructura de Internet en la región que cubre RIPE NCC.

El servicio de consulta de la base de datos RIPE permite a los usuarios realizar consultas y búsquedas en esta base de datos para obtener información sobre recursos específicos de Internet, direcciones IP, nombres de dominio, contactos de administración y otros detalles relevantes. Esto puede ser útil para investigar la propiedad de recursos de Internet, identificar a los responsables de ciertos servicios en línea, diagnosticar problemas de red y realizar análisis de infraestructura de Internet.”

Deduzco por lógica que se trata de un servicio de gestión de recursos de red que usa Vodafone para las conexiones LTE (en este caso), desconozco si para otro tipo de conexiones, dada su distinta naturaleza utiliza este servicio.

Texto

Descripción generada automáticamente

Para el caso de la IP 77.209.227.18 se observa que se trata de otro RIPE de VODAFONE, cuyos atributos descripción es VODAFONE\_SPAIN\_NETWORK y role VODAFONE ESPANA ROLE, dirección completa en Madrid y más información de RIPEs

Captura de pantalla de un celular

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Otra curiosidad es una conexión HTTPS a la IP 104.1.239.159, que resulta ser un servicio de AT&T usado por ARIN (American Registry for Internet Members).

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Entre otras cosas se puede observar cómo mi ordenador de casa realiza conexiones a diferentes servicios externos. Pero. ¿qué pasa si existe un servidor? Este será, principalmente quien reciba numerosas conexiones entrantes. Por lo que, escuchar la red en un entorno productivo puede determinar de entre todas las máquinas conectadas dentro de la red local cuál sería el servidor.

Además de lo mencionado, al escuchar la red se puede averiguar servicios, información de conexiones internas con otros servicios y máquinas y una serie de actividades para tenerlas en cuenta en los próximos pasos de una auditoría.

### Arp-scan

La herramienta ARP-SCAN se centra en el descubrimiento utilizando el protocolo ARP (Address Resolution Protocol). El cual envía solicitudes ARP a todas las direcciones IP en una misma red, registrando las respuestas y enumerando los dispositivos activos en la red, mapeando direcciones IP a direcciones MAC en una red local.

Texto

Descripción generada automáticamente

Si existiera más de una interfaz conectada al equipo, se puede escanear únicamente una de ellas del siguiente modo:

Texto

Descripción generada automáticamente

Como se puede observar, arp-scan permite conocer la magnitud de la red, pero no otorga mucha más información.

### NMAP

Con NMAP se puede realizar un escaneo basado en paquetes ICMP a toda la red, que, aunque haga un poco de ruido no lo considero un escaneo tan activo.

Además, si se ejecuta como sudo, el modo por defecto de escaneo es un escaneo SYN.

Texto

Descripción generada automáticamente

Como se puede apreciar NMAP otorga más información, mostrando la existencia de dos máquinas de VirtualBox, una con la IP 10.0.2.15 y otra con la 10.0.2.5

Al comprobar la dirección IP de la máquina desde la que estoy actuando se puede descartar la IP 10.0.2.5, ya que sería yo mismo. Por lo que por descarte ya conozco que la máquina víctima es la 10.0.2.15.

Texto

Descripción generada automáticamente

Si se desea realizar este proceso sin escanear la red completa, se puede utilizar la información que otorgada por arp-scan y escanear un rango de direcciones IP. Por ejemplo, desde la 10.0.2.1 hasta la 10.0.2.15

Texto

Descripción generada automáticamente

Para el ejemplo anterior se me ocurre la opción de guardar el resultado de arp-scan en un fichero y generar una lista que contenga solamente la enumeración de las IP a fin de utilizar esa lista para escanear únicamente las IP que hay dentro de la red local, pero no lo veo muy útil, ya que solo arroja información sobre el adaptador de red.

### Ping

Método comúnmente utilizado para comprobar la visibilidad entre dos equipos:

Texto

Descripción generada automáticamente

## Escaneo Activo

### HPING3

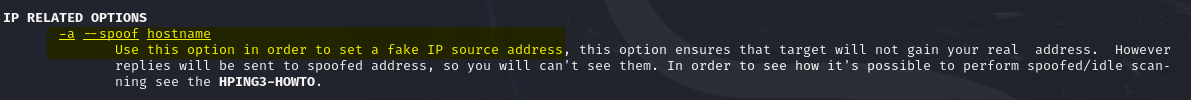
Con HPING3 se pueden realizar escaneos más en profundidad para comprobar los puertos TCP a la escucha mediante el envío de paquetes SYN. Paquete que se envía para el proceso de establecer una conexión TCP:

Imagen de la pantalla de un video juego

Descripción generada automáticamente con confianza baja

Donde --scan se especifica el intervalo de puertos, --syn especifica un escaneo basado en paquetes SYS y por último la IP de la víctima

HPING3 tiene opciones interesantes, como utilizar una IP falsa de origen para enviar los paquetes.



### NMAP

NMAP es una herramienta muy completa, mencionaré algunas utilidades principales que ofrece.

NMAP puede realizar un escaneo basado en SYS (sin terminar la conexión **-PS**) ACK (terminando la conexión **-PA**), UDP Ping (**-PU**) o SCTP (es un protocolo orientado a las conexiones, similar a TCP, pero proporciona la transferencia de datos orientada a mensajes, similar a UDP **-PY**):

Texto

Descripción generada automáticamente

NMAP puede realizar un escaneo de servicios a fin conocer la versión del servicio utilizado para el cada puerto usando la opción -sV. Esta opción es sumamente valiosa para encontrar posibles vulnerabilidades, se puede utilizar la opción -A junto con -sV para realizar un escaneo más agresivo:

Texto

Descripción generada automáticamente

## Enumeración

Para la fase de enumeración he trabajado en tres ramas distintas, pero me quedo con la última, ya que he visto que es una tarea pesada y he visto la oportunidad de agilizar bastante esta fase mediante la utilización de un robot creado por mí mismo.

Desarrollar el robot me ha llevado bastante tiempo, retrasando la entrega de la práctica, pero lo veo muy útil de cara a futuras fases de enumeración e incluso con la posibilidad futura de mejorar el robot con metodologías que otorga la experiencia. De hecho, el robot ha sido mejorado para su uso en la máquina Windows.

Este robot está en su primera etapa con el objetivo de cumplir la función dentro de esta práctica, la cual soy consciente de que no va a estar perfecta, pero que, aun así, doy lo mejor que pueda.

La fase de enumeración se puede realizar en un primer término con la ejecución de dos escaneos de NMAP. Guardando el output de la ejecución en dos ficheros distintos:

* Nmap -sV 10.0.2.15 > sV.txt
* Nmap -sV -A –script=vuln 10.0.2.15 > Complete.txt

**Puede revisar el contenido resultante de ambas ejecuciones en la carpeta correspondiente a la máquina a auditar dentro de la práctica entregada.**

Por una parte, la primera ejecución de NMAP recoge un listado de los servicios a la escucha en la máquina víctima, así como su versión. Se tratará esta información como “secciones” en un nivel lógico que más adelante utilizaré para clasificar y extraer la información mediante un script de Python.

Por otra parte, la segunda ejecución genera un informe que, por cada servicio a la escucha con su versión (sección), recoge, entre otra información, un listado de vulnerabilidades conocidas con su CVE y una URL que lleva hasta la base de datos pública de dicho CVE, donde se puede encontrar los detalles de la vulnerabilidad.

Con ayuda de un script basado en Python, extraigo por una parte el listado de secciones, el cual utilizo para filtrar el contenido de Complete.txt y dividirlo en secciones con el objetivo de extraer de cada sección los datos de CVE y URL (en la versión actual también extrae un listado de exploits asociados a cada CVE) de cada vulnerabilidad mencionada en el informe para cada servicio. Generando un archivo. JSON que contiene la información organizada.

Mi objetivo es generar un Excel con diferentes hojas. las cuales, cada una de ellas, contengan información útil sobre cada vulnerabilidad encontrada. Para ello hay que visitar cada URL e ir copiando la información útil a fin de generar el informe, lo cual es bastante pesado. De ahí a que, aprovechando la organización de la información de los informes de NMAP, genero un robot que realiza esa tarea repetitiva por mí. Además, he implementado una funcionalidad extra que es la búsqueda “manual”, donde se le dice al robot el servicio con su versión y un puerto y realiza la búsqueda. Obteniendo la información de cada vulnerabilidad y generando la hoja de forma automática.

Funcionamiento del robot para NMAP:

1. **Ejecución del script en Python**: Lo primero que hace falta es la extracción objetiva de los datos, por lo que primero ejecuta el script con el objetivo de generar el archivo JSON.
2. **Desserializar el contenido**: La segunda tarea es darle un formato oportuno al archivo JSON, por lo que el robot convierte el contenido en un diccionario con el siguiente aspecto:
   1. Dict = {"service”: [ {“CVE”:“CVE-XXXX-XXXXX”, “URL”:“https://...”}, {},… ] }
   2. Dict = {"service”: [ {“CVE”:“CVE-XXXX-XXXXX”, “URL”:”https://...”, “EXPLOIT”:“https://...”}, {},… ] } – En su última versión.
3. **Navegar a la fuente**: A continuación, se recorre el contenido del diccionario para navegar a las diferentes fuentes públicas.
4. **Extraer la información**: Se extrae la información necesaria de cada CVE y se guarda en una variable de tipo DataTable.
5. **Escribir la hoja**: Por último, el robot escribe el contenido extraído en una hoja de Excel con el número de puerto del servicio dentro del fichero Vunerabilities.xlsx.

Funcionamiento del robot para la búsqueda Manual:

1. **Recoger información del usuario**: El robot pide la introducción del servicio con su versión (Ej: OPENSSH 6.6) y tras ello el número de puerto (utilizado para generar la hoja).
2. **Navegar a la fuente**: A continuación, accede a CVEDetails.com para realizar una búsqueda por servicio y versión.
3. **Extraer la información**: Se extrae la información necesaria de cada CVE, realizando un scrapping del resultado de la búsqueda y tras ello se accede a cada CVE resultante, guardando la información en una variable de tipo DataTable.
4. **Escribir la hoja**: Por último, el robot escribe el contenido extraído en una hoja de Excel con el número de puerto del servicio dentro del fichero Vunerabilities.xlsx.

**Tiempo de ejecución en mi equipo para procesar 18 vulnerabilidades: 2 minutos 30 segundos.**

**La ejecución del robot permite que se siga trabajando en otras tareas, ya que la comunicación con el navegador se realiza a través de la API de Chrome.**

**Como parte de esta práctica, la enumeración se recoge en el archivo Excel Vulnerabilities.xlsx. Revisar el contenido completo del informe.**

Una captura de pantalla de una computadora

Descripción generada automáticamente

Después de una ejecución para procesar los CVE detectados por el script de vulnerabilidades de NMAP reviso la salida de sV junto al informe generado para ir buscando vulnerabilidades de los servicios que el script de vulnerabilidades no ha sido capaz de encontrar.

Aplicación

Descripción generada automáticamente con confianza media

Realizo una búsqueda de esos servicios utilizando el robot (a excepción de MySql que no se sabe la versión).

Como no queda muy clara la versión de Samba realizo otro escaneo más agresivo para el escaneo de Service Version:

* Nmap -sV -A 10.0.2.15 > sv\_A.txt

Tras eso lo organizo y me quedo con la información necesaria.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente con confianza media

**Enumeración concluida, revisar Vulnerabilities.xlsx para consultar el informe.**

## Explotación

### 21/FTP - CVE-2015-3306

La principal vulnerabilidad y la más famosa que tiene la verisión 1.3.5 de ProFTPD 1.3.5 es la vulnerabilidad Mod\_Copy, la cual permite abrir un a reverse Shell mediante un exploit encontrado en Metaexploit.

El funcionamiento básico de Mod\_Copy es que, cuando se realiza la conexión a un ser servicio FTP bajo la utilización de NINGÚN usuario, podemos copiar y pegar archivos dentro de la máquina víctima sin necesidad de atutenciación.

Para ello simplemente me conecto usando nc (para aprovechar la vulnerabilidad de conexión mediante la utilización de ningún usuario) y pruebo a realizar una copia del fichero passwd hacia el path por defecto de un directorio web:

* SITE CPFR /etc/passwd
* SITE CPTO /var/www/html/passwd

Texto

Descripción generada automáticamente

Como se puede apreciar, el funcionamiento de la vulnerabilidad es básico, pero se pueden utilizar distintos payloads para conseguir incluso un remote Shell, que, para este ejemplo sencillo me permitirá consultar el contenido dl fichero que he copiado con anterioridad.

Como quiero ir paso a paso y mi nivel de Python no es tan bueno como me gustaría para poder entender qué hace un script de explotación medianamente complejo voy a comenzar con metaexploitable para intentar abrir un reverse Shell en la máquina víctima usando esta vulnerabilidad.

Lo primero de todo es abrir la consola de metaexploitable y realizar la búsqueda de la vulnerabilidad:

Texto

Descripción generada automáticamente

Como se puede apreciar aparece un exploit para la vulnerabilidad Mod\_Copy, además con un rango excelente, así que el siguiente paso es usar ese exploit.

Texto

Descripción generada automáticamente con confianza media

Al seleccionar el exploit, metaexploit avisa de que no se encuentra un payload configurado, y que se va a usar por defecto una Shell reversa utilizando netcat de forma predeterminada.

Por curiosidad voy a ver los payloads que existen:

Texto

Descripción generada automáticamente

De los payloads disponibles me llaman la atención dos de ellos: el primero es la creación de un usuario vía comando useradd, interesante para poder realizar una conexión ssh e intentar una escalada de privilegios.

El segundo es la que he comentado previamente para abrir un reverse Shell via Python, escojo esta de momento.

Texto

Descripción generada automáticamente con confianza media

Con la ejecución de options se muestra los diferentes argumentos a configurar para realizar la explotación:

Texto

Descripción generada automáticamente

Las diferentes secciones por configurar son las siguientes:

* RHOSTS: listado de hosts objetivo, puede ser uno solo o varios.
  + Sintaxis: 10.0.2.15-xxx
  + 
* SITEPATH: El path objetivo donde establecer la conexión de la Shell.
  + 
* LHOST: Host donde abrir la escucha para el reverse Shell.
* LPORT: Puerto utilizado para escuchar la conexión entrante para el reverse Shell
* SHELL: Tipo de Shell utilizada para la conexión.
  + 

Con todo configurado solo queda ejecutar el exploit y comprobar el resultado:

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Ahora voy a tratar de crear un usuario en el sistema.

Texto

Descripción generada automáticamente

Para seguir con el patrón de usuarios voy a añadir a R2\_D2.

Texto

Descripción generada automáticamente

Tras la ejecución avisa de que hay que eliminar las pruebas de forma manual, así que vuelvo a ejecutar el reverse Shell para eliminarla y aprovecho también para eliminar passwd y comprobar que se ha creado el usuario.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

He eliminado los ficheros introducidos en el directorio, pero no se ha creado el usuario r2d2, esto puede deberse a una falta de permisos para crear el usuario.

Aun así, el servicio sí es vulnerable a Mod\_Copy, así que edito el fichero Vulnerabilities.xlsx para dejarlo reflejado.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente con confianza media

### 22/SSH- CVE-2018-15919 && CVE-2018-15473

Estos dos CVE aparentemente apuntan hacia el mismo problema con la única diferencia de las versiones afectadas. Cualquiera de los dos CVE afectan a la versión 6.6.1 de la máquina objetivo.

La descripción de esta vulnerabilidad es la siguiente: El comportamiento observable de forma remota en auth-gss2.c en OpenSSH hasta 7.8 inclusive podría ser utilizado por usuarios maliciosos remotos para detectar la existencia de usuarios en un sistema de destino cuando GSS2 está en uso.

Más información detallada: <https://seclists.org/oss-sec/2018/q3/180>

Sabiendo que se puede identificar si el usuario existe mediante la autenticación gaspi (auth-gss2.c) el siguiente paso es dar con un exploit que se aproveche de esta vulnerabilidad con el objetivo de determinar si existe un usuario o no dentro del sistema usando la fuerza bruta.

<https://github.com/epi052/cve-2018-15473>

README.md

Texto

Descripción generada automáticamente

He revisado el código y a simple vista no parece que abra un backdoor ni instale algo sospechoso en el equipo. Además también le he encomendado a ChatGPT que revise el código con ese fin. Llegando a la conclusión de que solamente se centra en la explotación de la vulnerabilidad de enumeración de usuarios. Por lo que comienzo clonando el repositorio en mi máquina.

Texto

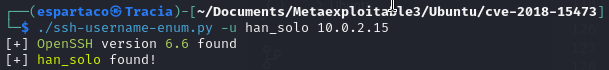
Descripción generada automáticamente

El siguiente paso es darle permisos de usuario al script de python.

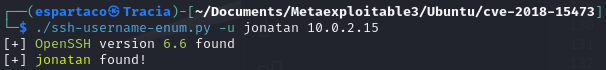


Mi versión de Python ya tiene instaladas las dependencias necesarias para ejecutar el script, así que solamente queda lanzarlo.

Para probar que el script funciona voy a intentar descubrir a “han\_solo”, ya que en la explotación anterior he visto que existe gracias a mostrar el contenido de /etc/passwd.



A simple vista parece que el servicio es vulnerable, pero hay que comprobar el caso negativo, así que hago un intento para el usuario Jonatan, el cual no se encuentra en el sistema objetivo.



Pues al parecer parece un falso positivo, ya que se busque por el usuario que se busque, la ejecución del script sale satisfactoriamente encontrado.

Como no entiendo mucho de Python voy a usar otro script, esta vez de exploit-db.

Kali tiene instalada la herramienta searchlist, que busca directamente en la base de datos de Exploit-DB, algo útil para agilizar las cosas desde el intérprete de comandos. Si se desea revisar el código sin descargar es útil visitar la web de Exploit-DB.

Como se ha visto en clase también existen webs como Explotius o AttackerDB.

Texto

Descripción generada automáticamente

El resultado arroja 4 scripts de la misma vulnerabilidad que estoy intentando explotar. Yo los he descargado todos previamente y todos tienen fallos de sintaxis y fallos por obsolescencia. Excepto User Enumeration (2) 45939.py, así que voy con ese.

Texto

Descripción generada automáticamente

Una vez revisado el código procedo con la comprobación.



Y como se puede apreciar, el servicio parece ser seguro frente a esta vulnerabilidad.

### 8080/Jetty 8.1.7 –CVE-2019-10247

Esto es una vulnerabilidad muy simple, pero me ha dado pie a encontrar una vulnerabilidad bastante gorda.

La vulnerabilidad trata de que al intentar entrar en un directorio que no existe, Jetty hace una serie de sugerencias de directorios disponibles.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Si accedo al directorio aparece el apartado de login de una web llamada Continuum.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Pero ¿qué es Continum?

<https://continuum.apache.org/>

Descripción según la web oficial mencionada: Apache Continuum™ es un servidor de integración continua listo para la empresa con características como compilaciones automatizadas, administración de versiones, seguridad basada en roles e integración con herramientas de compilación y sistemas de administración de control de fuente populares. Ya sea que tenga un equipo de compilación centralizado o desee poner el control de los lanzamientos en manos de los desarrolladores, Continuum puede ayudarlo a mejorar la calidad y mantener un entorno de compilación consistente.

Me parece interesante, tengo por un lado Apache 2.4.7, por otro lado Jetty 8.1.7 y por último Continuum.

Pero volvamos atrás y miremos más profundamente:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico, Sitio web

Descripción generada automáticamente

La vulnerabilidad asociada al CVE-2019-10247 me acaba de dar la información sobre la versión de Continuum, siendo la 1.4.2. Lo que me da pie a buscar vulnerabilidades para esa versión en concreto. Hay que actualizar la enumeración entonces…

### 8080/Continuum - CVE-2013-2251

Apache Continuum se ve afectado por una vulnerabilidad en la versión de la biblioteca Struts que utiliza. Esta vulnerabilidad permite a un usuario malintencionado ejecutar código en el servidor de forma remota. Se pueden encontrar más detalles sobre la vulnerabilidad en <http://struts.apache.org/release/2.3.x/docs/s2-016.html>

Fuente: <https://continuum.apache.org/security.html>

Este ejemplo refleja como el proceso de un pentest no es lineal, sino cíclico, por lo tanto añado una nueva vulnerabilidad al fichero Vulnerabilities.xlsx

Imagen que contiene Texto

Descripción generada automáticamente

Bien, localizada y enumerada la vulnerabilidad procedo a explotarla. Para ello he buscado en <https://sploitus.com/> <https://attackerkb.com/> y <https://www.exploit-db.com/> y no he encontrado algo más interesante (que funcione/que sepa hacer funcionar) que el exploit de metaexploit, el cual funciona muy bien.

Aún no se hacer mis propios exploits con mis propios payloads, confío en que aprendamos a hacer este tipo de trabajos, ya que me parece sumamente interesante. De hecho, la última vulnerabilidad de la práctica es una DOS generada por mi mismo.

Sin más remedio… ¡A metaexploit!

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Me resulta interesante la opción de reverse TCP con Meterpreter, es bastante potente por lo que he podido comprobar, así que dejo ese payload configurado y paso al resto de parámetros.



¡Y dentro!

Texto

Descripción generada automáticamente

Como se puede apreciar tengo acceso completo al sistema de ficheros del directorio de continuum.

Texto

Descripción generada automáticamente

Una vez dentro fíjate todas las acciones que se puede hacer con meterpreter, personalmente me ha sorprendido. Echándole imaginación se puede hacer muchas cosas desde este punto.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Para demostración voy a realizar una descarga de todos los ficheros de Continuum. Para ello lo primero es crear el directorio donde va a ser descargada la información.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Creado el directorio realizo la descarga.

Texto

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

Además se puede abrir una Shell en la máquina objetivo con permisos de root. GENIAL!

Texto

Descripción generada automáticamente

Y con esto queda completamente verificada la vulnerabilidad de la librería de Continuum.

### 80/SQL Injection - Get System Users and Generate MySQL Root User

Interfaz de usuario gráfica, Texto, Correo electrónico

Descripción generada automáticamente

Estudiando la interfaz web encontramos tres directorios y un fichero payrroll\_app.php

Al acceder al primer directorio encuentro un formulario que pregunta por mi nombre. Al introducirlo y pulsar Enter se abre una web con un chat en el que solo estoy yo. Desconozco el funcionamiento interno pero de momento no le veo utilidad.

Al acceder al directorio drupal se carga una web con un banner de Metaexploitable3, imagino que tendrá una utilidad o vulnerabilidad, pero no lo he comprobado.

El tercer directorio abre el formulario de login de phpmyadmin del cual no consigo acceso, como es lógico.

Por último, el fichero payroll\_app.php abre una web con un formulario que pide usuario y contraseña, pero no parece ser un login, ya que cuando introduzco user:admin password:admin la web muestra un mensaje de bienvenida con lo que parece ser la cabecera de una tabla, lo que me hace pensar que el programa realmente está haciendo algún tipo de consulta a una base de datos para obtener los valores de la cabecera filtrando por username y password de una tabla para mostrarlo en la web sin realizar ninguna comprobación.

Interfaz de usuario gráfica

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Para verificarlo voy a seguir la lógica de una consulta sql de este estilo:

* Select username, first\_name, last\_name, salary from <table> where username = ‘pepito’ and password = ‘grillo’;

La consulta anterior no daría ningún resultado, por lo que hacer una inyección funcional podría quedar así

* Select username, first\_name, last\_name, salary from <table> where username = ‘’ and password = ‘’; show tables;’;

El resultado sería una consulta con un output vacío, otra para listar las tablas de la base de datos y por último ‘; que no sería reconocido por el SGBD. Aunque quizá si se coloca la inyección en el campo username también se lo trague el SGBD.

* Select username, first\_name, last\_name, salary from <table> where username = ‘’; show tables;’ and password = ‘’;

Para obtener la última consulta coloco el código de la inyección en el input de USER, en caso de que no funcione probaría en el input de PASSWORD.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Tabla

Descripción generada automáticamente

Y aquí está el resultado, la inyección ha funcionado, mostrando un listado de tablas disponibles en la cual solo existe la tabla users.

Ahora me gustaría consultar por completo la tabla users para ver si el diseño del script de php muestra el contenido de toda la tabla.

* Select username, first\_name, last\_name, salary from <table> where username = ‘’; select \* from users;’ and password = ‘’;

Tabla

Descripción generada automáticamente

Interesante, muy interesante, el resultado obtenido en el campo username tiene una correspondencia con el contenido de passwd de la máquina objetivo. Además, recordemos que se realiza una consulta por el campo password, lo cual la cuarta columna presenta un formato legible para ser un campo de contraseñas.

Quiero realizar una comprobación de autenticación en el servidor para saber si puedo acceder con estos datos vía ssh. Por ejemplo con han\_solo. Comprobando que, efectivamente, son las contraseñas de los usuarios de la máquina metaexploitable3-Ubuntu.

Texto

Descripción generada automáticamente

Pero no solo eso, sino que han\_solo puede convertirse en root como se ve a continuación.

Interfaz de usuario gráfica, Texto, Sitio web

Descripción generada automáticamente

He tenido suerte, ya que el usuario han\_solo viene incluido en el grupo sudo según la información proporcionada al mostrar el contenido de /etc/group..

Texto

Descripción generada automáticamente

Vale, aquí se puede usar la imaginación, porque con esta inyección SQL no solo puedo acceder al sistema con los usuarios del SO, sino que también podría apañármelas para acceder a PhpMyAdmin con un usuario con privilegios.

No conozco las credenciales de root para la base de datos, pero si que conozco como crear un usuario con permisos de root. Para ello y que no sea muy llamativo utilizaré a continuum para el nuevo usuario.

* '; CREATE USER 'root\_continuum'@'localhost' IDENTIFIED BY 'alumno01';

Tabla

Descripción generada automáticamente

Parece que ha funcionado, voy a escribir todos los comandos restantes en la inyección SQL e intentar acceder a PhpMyAdmin.

* '; GRANT ALL PRIVILEGES ON \* . \* TO 'root\_continuum'@'localhost'
* '; FLUSH PRIVILEGES;

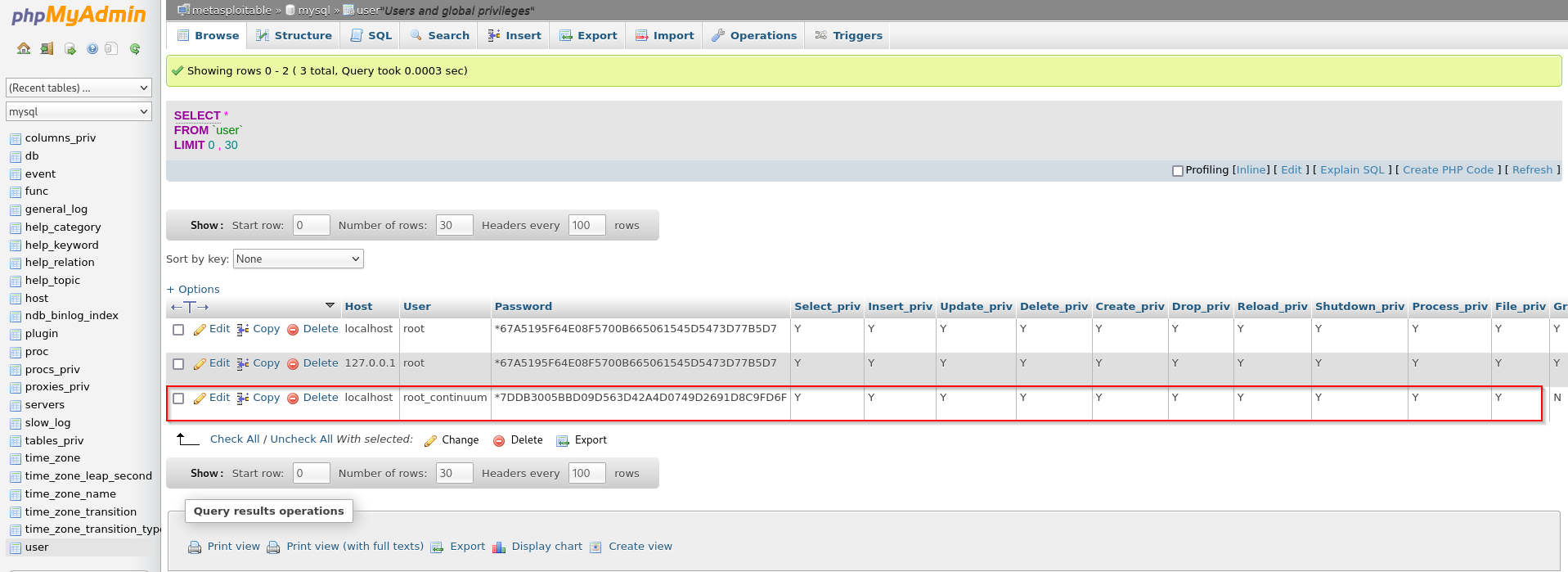
El resultado de la inyección es el siguiente:

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente



Conseguido, he generado un usuario con todos los privilegios en MySQL, lo que me permite realizar lo que se me antoje. ¡Genial!

### Ubuntu 14.04 - Privilege Escalation -Throught Docker

En el paso anterior he mostrado cómo a través de SQL Injection he conseguido la lista de los usuarios del SO objetivo y he iniciado sesión con han\_solo que pertenece al grupo sudo, pero el usuario boba\_fett en teoría no podría convertirse en root.

Texto

Descripción generada automáticamente

Dada esta situación gustaría realizar una escalada de privilegios utilizando el usuario boba\_fett.

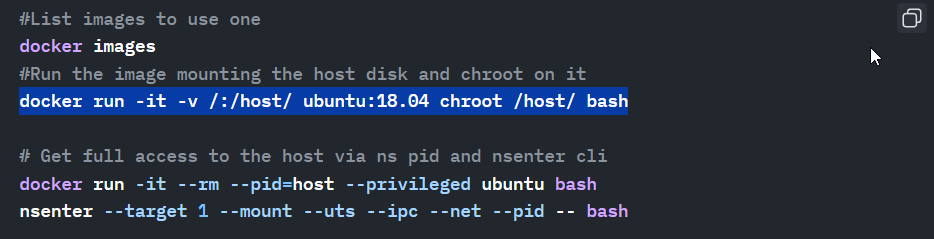
Para esto realizo varias comprobaciones para saber si puedo hacer una escalada a través de sudo -l, etc. Hasta darme cuenta de que el usuario pertenece al grupo Docker, por lo que intento ejecutar algo en Docker, pudiendo comprobar que tengo acceso a ejecución para Docker.

Texto

Descripción generada automáticamente

Rápidamente me pongo a buscar formas de elevar privilegios mediante Docker, encontrando la siguiente fuente:  
<https://book.hacktricks.xyz/linux-hardening/privilege-escalation/docker-security/docker-breakout-privilege-escalation>

Siguiendo los pasos consigo entrar como root del sistema mediante dos métodos distintos gracias a Docker.



1. El primero:

* docker run -it -v /:/host/ ubuntu:14.04 chroot /host/ bash

Texto

Descripción generada automáticamente

1. El segundo:

* docker run -it --rm --pid=host --privileged ubuntu bash

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

No es hasta la ejecución del segundo comando que se gana acceso total al sistema.

* nsenter --target 1 --mount --uts --ipc --net --pid – bash

Texto

Descripción generada automáticamente

# Windows

Observación personal: me queda poco tiempo para entregar la práctica, invertiría más tiempo en ella si no fuese porque la entrega de la práctica de Hacking Web es un día después de la entrega a esta. Por ello solamente voy a explotar el servicio ssh de la máquina Windows, centrándome sobre todo en la fase de explotación.

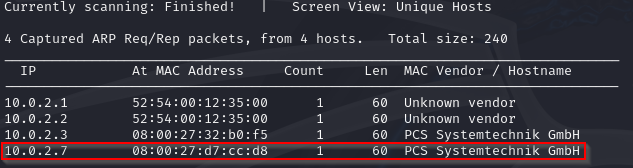
La fase de escaneo y enumeración la voy a comentar muy por encima. Se puede encontrar toda la información en los ficheros de la carpeta Windows.

La enumeración la realizaré con el robot creado.

## Escaneo pasivo

Por ver más formas de escaneo pasivo, usaré netdiscover.

* sudo netdiscover -r 10.0.2.0/24



## Escaneo activo

Realizo los informes de NMAP necesarios para el correcto funcionamiento del robot, ya que he visto que es sumamente efectivo y de utilidad.



Texto

Descripción generada automáticamente

## Enumeración

Para la enumeración he modificado tanto el script de Python como el robot que procesa la información. Podrá apreciar que ahora el Excel Vulnerabilities.xlsx generado por el robot contiene una nueva columna llamada EXPLOIT con las URL a diferentes exploits conocidos.

Para la ejecución manual del robot no es posible obtener información de Exploits de forma automática, ya que, por lo general la búsqueda de exploits en bases de datos públicas para un determinado servicio no se corresponden con lo requerido, debiendo hacer una búsqueda totalmente manual y objetiva de estos.

Como he comentado anteriormente para la máquina Windows solamente voy a auditar el ssh, por lo que no voy a introducir posibles vulnerabilidades para otros servicios dentro de Vulnerabilities.xlsx. Tomémoslo como una auditoría al servicio SSH de un determinado servidor.

Puede consultar el fichero Vulnerabilities.xlsx para visualizar la enumeración para el puerto 22.

## Explotación

### 22/SSH- CVE-2018-15919 && CVE-2018-15473

Descripción de la vulnerabilidad: OpenSSH hasta la versión 7.7 es propenso a sufrir una vulnerabilidad de enumeración de usuarios debido a que no retrasa el rescate de un usuario de autenticación no válido hasta que el paquete que contiene la solicitud se haya analizado por completo, en relación con auth2-gss.c, auth2-hostbased.c y auth2-pubkey. .C.

Esta misma vulnerabilidad se probó en la máquina de Ubuntu, dando como resultado falso positivo en la ejecución de tres scripts distintos.

Para la ejecución de esta vulnerabilidad he escogido el fichero “CommonAdminBase62.txt” del repositorio SecList, extrayendo del él solamente los nombres de usuario.

Texto

Descripción generada automáticamente

He descargado el contenido a un fichero llamado dwnl\_username\_dict.txt.

Este archivo contiene una lista de nombres de usuario mas contraseña separados por ‘:’. A fin de obtener solamente los nombres de usuarios ejecuto lo siguiente.

* cat dwnl\_username\_dict.txt | cut -d ':' -f1 > username\_dict.txt

No se ve el contenido completo del diccionario, pero en el propio repositorio de seclist, este diccionario contiene el usuario vagrant, lo cual viene genial para la demostración de esta vulnerabilidad.

Texto

Descripción generada automáticamente

He mirado algunos scripts de fuentes públicas y este me ha parecido interesante, parece legítimo y útil para su ejecución:

<https://github.com/Sait-Nuri/CVE-2018-15473>

En el repositorio menciona que es una versión del exploit <https://www.exploit-db.com/exploits/45233> lo caul me parece lógico, porque el código del exploit al que hace referencia contiene errores y no se ejecuta debidamente.

Bien, clono el exploit, reviso el fichero requirements.txt para verificar, lo veo lógico, instalo los requerimientos, el cual solo me faltaba argparse.

Texto

Descripción generada automáticamente

Reviso que el código esté correcto, incluida la utilización correcta del path del environment de Python, copio el fichero username\_dict.txt dentro del directorio para hacer la comprobación, como se ve en la imagen y finalmente lo ejecuto.

* ./CVE-2018-15473.py 10.0.2.7 -w username\_dict.txt

Texto

Descripción generada automáticamente

[…]

Texto

Descripción generada automáticamente

Como se puede observar, el servicio es vulnerable a la enumeración de usuarios gracias al comportamiento de los archivos auth2-gss.c, auth2-hostbased.c, and auth2-pubkey.c ejecutados por el servicio SSH.

### 22/SSH – Hydra – Brute Force

Bien, una vez tengo los nombres de usuario quiero comprobar si el servicio SSH es vulnerable al uso de la fuerza bruta.

Para ello no hace falta ningún script, porque ya conozco una herramienta genial que hace esto: Hydra.

Bien, lo que tengo pensado es usar dos diccionarios, uno que contenga los usuarios que se han verificado y otro que contenga las contraseñas.

Para el diccionario de usuarios simplemente creo un fichero llamado users.txt con los usuarios dentro.

Texto

Descripción generada automáticamente

Para las contraseñas, usando la lógica, una mala práctica sería usar el mismo nombre de usuario como contraseña, así que duplico el contenido de username\_dict.txt a passwords.txt.



Ahora quiero aprovechar también las contraseñas del diccionario de SecList, que fue descargado como dwnl\_username\_dict.txt, y que contiene la siguiente estructura: username:password, por lo que para añadir las contraseñas al fichero passwords.txt ejecuto lo siguiente

Quedando el siguiente resultado (en parte).

Texto

Descripción generada automáticamente

Bien, para realizar la ejecución de hydra lo hago de la siguiente manera:

* hydra 10.0.2.7 ssh -v -t 4 -L users.txt -P username\_dict.txt
  + Quiero usar la fuerza bruta contra el protocolo ssh de la máquina Windows.
  + Quiero activar Vervose para obtener más información.
  + Quiero limitar el uso de conexiones simultaneas a 4.
  + Quiero usar un diccionario para los nombres de usuario.
  + Quiero utilizar un diccionario para las contraseñas.

Texto

Descripción generada automáticamente

Y con esto ya tengo la información de que la contraseña para los usuarios vagrant y Administrator es vagrant.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente con confianza media

### 22/SSH - CVE-2016-6515 – DOS

Descripcción de la vulnerabilidad: La función auth\_password en auth-passwd.c en sshd en OpenSSH anterior a 7.3 no limita la longitud de las contraseñas para la autenticación de contraseñas, lo que permite a atacantes remotos provocar una denegación de servicio (consumo de CPU de cripta) a través de una cadena larga.

Para la explotación de esta vulnerabilidad he construido un script propio que, mediante un uso agresivo de conexiones mediante hilos aportando contraseñas con una longitud de 200000 caracteres consigo que el servidor deje de recibir información por el puerto 22 debido a la incapacidad de un procesamiento masivo de información.

Texto

Descripción generada automáticamente

Veamoslo en marcha, para auditar lo que pasa he iniciado sesión en la máquina objetivo y he abierto el Task Manager para monitorear los recursos. De momento todo bien y estable.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Todo funciona correctamente y está normal, puedo iniciar sesión utilizando ssh sin problemas.

Texto

Descripción generada automáticamente

Al ejecutar el script comienza la lluvia de peticiones, generando una notable inestabilidad en el servidor que deniega toda conexión al servicio ssh dada la incapacidad de procesar todas las peticiones de conexión. Interfaz de usuario gráfica

Descripción generada automáticamente

Script en funcionamiento:

Texto

Descripción generada automáticamente

Si intento entrar legítimamente al sistema pasa lo siguiente:

Texto

Descripción generada automáticamente

Denegación de servicios realizada

# Conclusión

Esta práctica ha sido bastante dura, me ha obligado a aplicar la teoría adquirida durante el curso y me ha hecho leer, documentarme y saber cómo funcionan muchos scripts, vulnerabilidades, protocolos, etc.

Creo que la utilización de tu propio código para explotar vulnerabilidades es un seguro, ya que sabes perfectamente lo que hace, cómo se comporta y su objetivo concreto.

Sinceramente me ha asustado un poco todo lo que es necesario saber sobre diferentes ramas de la informática para hacer un pentest correctamente. Pero recuerdo los momentos en los que empecé en la informática y en la programación y no hay mucha diferencia. Es una avalancha de información que agradezco encarecidamente, ya que, también personalmente, sin esto me falta algo en la vida. No me gusta tener tiempo libre. Aunque sienta presión por realizar acciones en un marco de tiempo limitado, cuando no vivo esa situación en mi vida es todo menos interesante, menos valorado y más aburrido.

Soy consciente de que no esperas una super documentación profesional sobre cómo realizar un pentest, ya que no se ha visto en clase modelos de documentación, teoría o ejemplos.

La dificultad que presenta realizar un test de intrusión me ha empujado a querer saber más acerca de cómo se realiza uno de forma profesional de primera mano, lo cual es algo complicado. Si conoces alguna recopilación de vídeos de algún creador de contenido de calidad agradecería que lo compartieras conmigo

Como siempre acepto feedback acerca de la entrega, cualquier reseña, objetividad o recomendación es bienvenida.