REPORTE DE PENTESTING EN HTB

Máquina: Cap

KeruDWL

“Este documento describe el proceso de reconocimiento, análisis y explotación de vulnerabilidades, así como la posterior escalada de privilegios, llevado a cabo sobre un activo objetivo dentro del entorno de Hack The Box.”

ÍNDICE

Introducción

El presente reporte tiene como finalidad documentar y describir de manera detallada el proceso de resolución de una máquina vulnerable dentro del entorno de Hack The Box. El ejercicio se desarrolla como una práctica de pentesting en un entorno controlado, orientada a la identificación de vectores de ataque, la obtención de acceso al sistema y la escalada de privilegios.

Durante el desarrollo de la práctica se documentaron de forma cronológica las distintas fases llevadas a cabo durante la resolución de la máquina, incluyendo el reconocimiento inicial, la enumeración de servicios, el análisis de la aplicación web, el acceso al sistema y la posterior escalada de privilegios, describiendo las técnicas y herramientas utilizadas en cada etapa; todas las actividades descritas en este documento se realizaron exclusivamente con fines educativos y dentro de un entorno autorizado, sin afectar sistemas reales de producción.

Alcance y datos generales del entorno

La resolución de la máquina se llevó a cabo dentro de un entorno de laboratorio controlado y autorizado, proporcionado por la plataforma Hack The Box. El alcance del ejercicio estuvo limitado exclusivamente al sistema objetivo asignado, sin interactuar con otros equipos o servicios ajenos al laboratorio.

El proceso se realizó bajo un enfoque de caja negra, en el cual no se contó con información previa sobre la configuración interna del sistema. Todas las acciones descritas en este reporte se ejecutaron con fines educativos y de aprendizaje en ciberseguridad, respetando los límites definidos por el entorno de pruebas.

El equipo atacante utilizó una máquina virtual con sistema operativo Kali Linux, conectada a la red privada del laboratorio mediante una interfaz VPN, lo que permitió establecer comunicación directa con el sistema objetivo.

Los datos generales del entorno evaluado se describen a continuación:

- **Plataforma:** Hack The Box (entorno de laboratorio)
- **Tipo de prueba:** Caja individual (Linux)
- **Sistema operativo del atacante:** Kali Linux
- **IP del atacante:** 10.10.17.69
- **IP del objetivo:** 10.10.10.245
- **Sistema operativo estimado del objetivo:** Linux (inferido a partir del valor TTL en respuestas ICMP)
- **Servicios expuestos inicialmente:**
    - Servicio FTP (puerto 21/tcp)
    - Servicio SSH (puerto 22/tcp)
    - Servicio HTTP (puerto 80/tcp)

Toda la actividad descrita en este reporte se realizó exclusivamente dentro de los límites definidos por el laboratorio, con fines educativos y de aprendizaje en ciberseguridad.

Metodología

La resolución de la máquina se llevó a cabo siguiendo una metodología de pentesting estructurada, aplicada dentro del entorno de laboratorio de Hack The Box. Dicha metodología permitió abordar el sistema objetivo de forma ordenada, progresiva y reproducible, evitando acciones aleatorias y priorizando la correcta enumeración antes de cualquier intento de explotación.

El proceso se basó en la recopilación y análisis continuo de información, donde cada fase alimentó a la siguiente, permitiendo identificar vectores de ataque viables y reducir la superficie de error durante la resolución de la máquina.

Las fases que conformaron la metodología aplicada fueron las siguientes:

- **Reconocimiento:**

  Validación de conectividad con el sistema objetivo y recolección inicial de información, incluyendo la identificación del sistema operativo mediante el análisis de respuestas ICMP y otros indicadores de red.

- **Enumeración:**

  Descubrimiento de servicios expuestos, puertos abiertos y versiones asociadas, así como la enumeración de funcionalidades accesibles a través de los servicios identificados, especialmente a nivel de aplicación web.

- **Análisis:**

  Evaluación de la información obtenida durante las fases anteriores con el objetivo de identificar configuraciones inseguras, exposiciones de información sensible y posibles vectores de ataque aprovechables.

- **Explotación:**

  Ejecución controlada de acciones orientadas a obtener acceso inicial al sistema, haciendo uso de credenciales expuestas, servicios mal configurados o funcionalidades vulnerables previamente identificadas.

- **Escalada de privilegios:**

  Enumeración local del sistema comprometido para identificar configuraciones inseguras, permisos incorrectos o binarios con privilegios especiales, permitiendo elevar los privilegios del usuario inicial hasta obtener acceso completo al sistema.

Esta metodología permitió documentar de manera clara y ordenada cada una de las etapas involucradas en la resolución de la máquina, asegurando un proceso lógico y alineado con las buenas prácticas del pentesting.

Reconocimiento y enumeración

La fase de reconocimiento y enumeración tuvo como objetivo obtener información inicial del sistema objetivo y de los servicios expuestos, estableciendo una base sólida para las etapas posteriores del proceso. A través de esta fase fue posible identificar el sistema operativo, los puertos abiertos y los servicios accesibles desde la red del laboratorio de Hack The Box.

Verificación de conectividad (ICMP)

En primer lugar, se verificó la conectividad entre la máquina atacante y el sistema objetivo mediante el envío de mensajes ICMP, con el propósito de confirmar que el host se encontraba activo y accesible dentro de la red del laboratorio.

Comando ejecutado: **ping -c 2 10.10.10.245**

```
┌──(kali㉿kali)-[~/Downloads]
└─$ ping -c 2 10.10.10.245
PING 10.10.10.245 (10.10.10.245) 56(84) bytes of data.
64 bytes from 10.10.10.245: icmp_seq=1 ttl=63 time=122 ms
64 bytes from 10.10.10.245: icmp_seq=2 ttl=63 time=125 ms

--- 10.10.10.245 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
```

Resultado observado:

- El sistema objetivo respondió correctamente a las solicitudes ICMP.
- Se obtuvo un valor TTL = 63, el cual es característico de sistemas Linux.

Con esto se confirmó que el host se encontraba activo y accesible desde la red del laboratorio.

Escaneo de puertos y servicios

Una vez validada la conectividad, se procedió a realizar un escaneo de puertos con el objetivo de identificar los servicios expuestos por el sistema objetivo, así como obtener información sobre sus versiones y configuraciones básicas.

Comando ejecutado: **sudo nmap -p- --min-rate 5000 -sC -sV 10.10.10.245**

```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo nmap -p- --min-rate 5000 -sC -sV 10.10.10.245
[sudo] password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-25 15:41 CST
Nmap scan report for 10.10.10.245
Host is up (0.32s latency).
Not shown: 65532 closed tcp ports (reset)
```

```
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 fa:80:a9:b2:ca:3b:88:69:a4:28:9e:39:0d:27:d5:75 (RSA)
|   256 96:d8:f8:e3:e8:f7:71:36:c5:49:d5:9d:b6:a4:c9:0c (ECDSA)
|_  256 3f:d0:ff:91:eb:3b:f6:e1:9f:2e:8d:de:b3:de:b2:18 (ED25519)
80/tcp open  http    Gunicorn
|_http-title: Security Dashboard
|_http-server-header: gunicorn
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 44.43 seconds
```

Resultado observado:

- Se identificaron tres puertos TCP abiertos.
- El sistema objetivo expone los siguientes servicios:
  - **FTP** en el puerto **21/tcp**
  - **SSH** en el puerto **22/tcp**
  - **HTTP** en el puerto **80/tcp**
- El servicio HTTP utiliza **Gunicorn**, lo que indica la ejecución de una aplicación web.

El servicio HTTP fue identificado como un punto de interés principal para continuar con la enumeración, debido a la posibilidad de interactuar directamente con la aplicación web.

Enumeración de directorios en el servicio web

Con el objetivo de identificar rutas, endpoints y funcionalidades adicionales expuestas por la aplicación web, se realizó un proceso de enumeración de directorios utilizando la herramienta **ffuf**, empleando un diccionario común de rutas.

Herramienta utilizada:

- **ffuf** (v2.1.0-dev)

Diccionario empleado:

- /usr/share/wordlists/dirb/common.txt

Comando ejecutado:

```
ffuf -u http://10.10.10.245/FUZZ -w /usr/share/wordlists/dirb/common.txt -fc 404
```

Este comando permitió realizar solicitudes HTTP de tipo GET sobre posibles rutas, filtrando las respuestas con código **404 (Not Found)** para mostrar únicamente aquellas rutas existentes o relevantes.

```
┌──(kali㉿kali)-[~/Downloads]
└─$ ffuf -u http://10.10.10.245/FUZZ -w /usr/share/wordlists/dirb/common.txt -fc 404

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://10.10.10.245/FUZZ
 :: Wordlist         : FUZZ: /usr/share/wordlists/dirb/common.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
 :: Filter           : Response status: 404
________________________________________________

                        [Status: 200, Size: 19386, Words: 8716, Lines: 389, Duration: 149ms]
data                    [Status: 302, Size: 208, Words: 21, Lines: 4, Duration: 130ms]
ip                      [Status: 200, Size: 17461, Words: 7275, Lines: 355, Duration: 137ms]
netstat                 [Status: 200, Size: 32537, Words: 15644, Lines: 487, Duration: 153ms]
:: Progress: [4614/4614] :: Job [1/1] :: 268 req/sec :: Duration: [0:00:17] :: Errors: 0 ::
```

Resultados obtenidos:

- data [Status: 302]
- ip [Status: 200]
- netstat [Status: 200]

Análisis de resultados

- Se identificaron múltiples rutas accesibles desde la aplicación web.
- La ruta **/data** llamó particularmente la atención, ya que presentaba redirecciones y parecía estar relacionada con información interna del sistema.
- Las rutas descubiertas indicaron la exposición de funcionalidades sensibles accesibles sin autenticación previa.

La información obtenida en esta fase permitió avanzar a la siguiente etapa del proceso, enfocada en el análisis detallado de la aplicación web y de los recursos accesibles a través de estas rutas.

Análisis de la aplicación web

Tras identificar el servicio HTTP expuesto en el puerto 80/tcp durante la fase de reconocimiento, se procedió al análisis de la aplicación web accesible desde el navegador, con el objetivo de identificar funcionalidades expuestas, posibles fallos de configuración y vectores de ataque aprovechables.

Acceso, identificación y funcionalidades expuestas de la web

Se accedió a la aplicación web utilizando un navegador, ingresando directamente a la dirección IP del sistema objetivo.

URL utilizada: **http://10.10.10.245**

Al acceder al servicio, se presentó una aplicación web identificada como **Security Dashboard**, la cual se encontraba accesible sin requerir autenticación previa.

Dentro de la aplicación web se observó que la sesión se cargaba de forma predeterminada con un usuario identificado como **nathan**, sin que fuera necesario realizar un proceso de inicio de sesión. La interfaz presentaba un panel de navegación lateral, desde el cual era posible acceder a distintas secciones del dashboard.

Entre las funcionalidades visibles se identificaron las siguientes:

- Security Snapshot
- IP Config
- Network Status

La ausencia de mecanismos de autenticación y control de acceso en estas secciones indicó una posible exposición de información sensible.

Descarga de archivos PCAP desde la opción Security Snapshot

Dentro de la sección **Security Snapshot**, se identificó un botón de descarga que permitía obtener un archivo con extensión .pcap. Inicialmente, el archivo descargado no contenía información relevante; sin embargo, al analizar la URL asociada a dicha descarga, se observó que esta incluía un parámetro numérico en la ruta.

La estructura de la ruta descargada era la siguiente: http://10.10.10.245/data/1

Al modificar manualmente este valor en la URL y probar con distintos identificadores, se detectó que la ruta: http://10.10.10.245/data/0

permitía descargar un archivo .pcap con una cantidad considerable de tráfico capturado.

Esta observación evidenció una exposición directa de archivos internos, accesibles sin autenticación y sin validación adecuada de permisos.

Identificación de información sensible expuesta

El archivo .pcap obtenido desde la ruta http://10.10.10.245/data/0 contenía múltiples paquetes de red capturados, lo que indicaba que la aplicación web permitía el acceso a capturas reales de tráfico del sistema.

Dado el potencial riesgo asociado a este tipo de información, el archivo fue identificado como un recurso crítico para su análisis, ya que podría contener credenciales, sesiones activas u otros datos sensibles transmitidos en texto claro.

Con base en estos hallazgos, se procedió a la siguiente fase del proceso, enfocada en el análisis detallado del tráfico de red contenido en el archivo PCAP.

Análisis de tráfico de red

Tras la descarga del archivo .pcap desde la aplicación web, se procedió al análisis del tráfico de red capturado, con el objetivo de identificar información sensible transmitida en texto claro que pudiera ser aprovechada para obtener acceso al sistema objetivo.

Apertura y revisión del archivo PCAP

El archivo descargado fue abierto utilizando la herramienta **Wireshark**, lo que permitió visualizar y analizar los paquetes de red capturados.

Herramienta utilizada:

- Wireshark (4.7.7.)

Durante la revisión inicial se observó una cantidad considerable de tráfico correspondiente a distintos protocolos, entre ellos **FTP**, **TCP** e **IP**, lo cual resultó relevante debido a la naturaleza insegura del protocolo FTP al transmitir credenciales en texto plano.

Identificación de tráfico y extracción de credenciales en texto claro

Con el fin de aislar la información relevante, se aplicaron filtros dentro de Wireshark para mostrar únicamente el tráfico correspondiente al protocolo FTP.

Filtro aplicado: ftp

Al analizar los paquetes filtrados, se identificaron solicitudes correspondientes a comandos **USER** y **PASS**, utilizados durante el proceso de autenticación del servicio FTP.

Dentro de los paquetes capturados se localizaron credenciales transmitidas en texto claro, correspondientes a un intento de autenticación exitoso contra el servicio FTP.

Paquetes relevantes identificados:

- USER nathan
- PASS Buck3tH4TF0RM3!

El contenido del paquete FTP mostró explícitamente el nombre de usuario y la contraseña, confirmando que el sistema objetivo estaba utilizando un protocolo inseguro sin cifrado para la autenticación.

Información obtenida

A partir del análisis del tráfico de red, se obtuvo la siguiente información crítica:

- **Usuario:** nathan
- **Contraseña:** Buck3tH4TF0RM3!
- **Servicio asociado:** FTP (puerto 21/tcp)

La exposición de estas credenciales representó un punto de entrada directo al sistema, ya que podían ser reutilizadas para intentar acceder a otros servicios disponibles, como FTP o SSH.

Con esta información, se procedió a la siguiente fase del proceso, enfocada en el acceso inicial al sistema utilizando las credenciales obtenidas.

Acceso inicial al sistema

Con base en las credenciales obtenidas durante el análisis del tráfico de red, se procedió a intentar el acceso al sistema objetivo a través del servicio **SSH**, el cual se encontraba expuesto y accesible desde la red del laboratorio. Esta fase tuvo como objetivo obtener una sesión interactiva en el sistema y validar los privilegios iniciales del usuario comprometido.

Acceso al sistema mediante SSH

Se utilizó el servicio SSH para establecer una conexión remota con el sistema objetivo empleando las credenciales previamente identificadas.

Credenciales utilizadas:

- Usuario: nathan
- Contraseña: Buck3tH4TF0RM3!

Comando ejecutado: **ssh nathan@10.10.10.245**

```
┌──(kali㉿kali)-[~/Downloads]
└─$ ssh nathan@10.10.10.245
nathan@10.10.10.245's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-80-generic x86_64)
nathan@cap:~$
```

Al ejecutar el comando, el sistema solicitó la contraseña del usuario. Tras ingresar la contraseña obtenida previamente, el acceso fue exitoso, confirmando que las credenciales eran correctas y permitían iniciar sesión en el sistema.

Posteriormente, se verificó la identidad del usuario y los privilegios asignados.

Comandos ejecutados: **whoami** & **id**

```
nathan@cap:~$ whoami
nathan
nathan@cap:~$ id
uid=1001(nathan) gid=1001(nathan) groups=1001(nathan)
```

Con esto confirmamos que las credenciales expuestas eran de nathan.

Verificación de TTY, enumeración inicial y obtención de la flag del usuario

Una vez dentro del sistema, se verificó el tipo de terminal obtenida para confirmar si era necesaria una estabilización de la sesión. Al tratarse de un acceso por SSH, se obtuvo directamente una TTY interactiva, por lo que no fue necesario aplicar técnicas de estabilización.

Comando ejecutado: tty

```
nathan@cap:~$ tty
/dev/pts/0
nathan@cap:~$
```

Posteriormente, se realizaron validaciones básicas para identificar información del sistema y el nivel de privilegios del usuario.

Comandos ejecutados: uname -a & sudo -l

```
nathan@cap:~$ uname -a
Linux cap 5.4.0-80-generic #90-Ubuntu SMP Fri Jul 9 22:49:44 UTC 2021 x86_64 x86_64
x86_64 GNU/Linux
nathan@cap:~$ sudo -l
[sudo] password for nathan:
Sorry, user nathan may not run sudo on cap.
nathan@cap:~$
```

Resultados observados:

- La sesión se ejecuta sobre una terminal /dev/pts/0, confirmando una TTY interactiva.
- Se identificó el kernel del sistema objetivo mediante uname -a.
- Se comprobó que el usuario nathan no cuenta con permisos sudo, por lo que no era posible elevar privilegios mediante esta vía.

Obtención de la flag

Con acceso interactivo al sistema, se procedió a localizar y leer la flag correspondiente al usuario comprometido.

Comandos ejecutados: ls & cat user.txt

```
nathan@cap:~$ ls
user.txt
nathan@cap:~$ cat user.txt
6958b009e34148d79f6e1dc3d24d98ff
nathan@cap:~$
```

Con esto se obtuvo la primera flag de la máquina.

Con el acceso inicial establecido, se continuó con la siguiente fase del proceso, enfocada en la enumeración local del sistema para identificar posibles vectores de escalada de privilegios.

Escalamiento de privilegios

Tras obtener acceso al sistema como el usuario nathan y confirmar que no contaba con privilegios administrativos, se inició el proceso de escalamiento de privilegios, con el objetivo de identificar configuraciones inseguras o mecanismos que permitieran elevar el nivel de acceso dentro del sistema.

Búsqueda de binarios con capacidades especiales

Dado que no fue posible escalar privilegios mediante sudo, se procedió a buscar binarios con **Linux capabilities** asignadas, las cuales pueden otorgar privilegios elevados a procesos específicos.

Comando ejecutado: **getcap -r / 2>/dev/null**

```
nathan@cap:~$ getcap -r / 2>/dev/null
/usr/bin/python3.8 = cap_setuid,cap_net_bind_service+eip
/usr/bin/ping = cap_net_raw+ep
/usr/bin/traceroute6.iputils = cap_net_raw+ep
/usr/bin/mtr-packet = cap_net_raw+ep
/usr/lib/x86_64-linux-gnu/gstreamer1.0/gstreamer-1.0/gst-ptp-helper = cap_net_bind_service,cap_net_admin+ep
nathan@cap:~$
```

Resultados observados:

- Se identificó el binario **/usr/bin/python3.8** con la capacidad **cap_setuid** asignada.
- La capacidad cap_setuid permite modificar el UID efectivo del proceso, lo que representa un vector crítico de escalamiento de privilegios si es abusado.

Explotación de binario con capacidad cap_setuid

Dado que el binario python3.8 contaba con la capacidad cap_setuid, fue posible ejecutar código que modificara el UID efectivo del proceso a **root (UID 0)** sin necesidad de permisos sudo ni explotación de vulnerabilidades del kernel.

Comando ejecutado:

/usr/bin/python3.8 -c 'import os; os.setuid(0); os.system("/bin/bash")'

```
nathan@cap:~$ /usr/bin/python3.8 -c 'import os; os.setuid(0); os.system("/bin/bash")'
root@cap:~#
```

Una vez obtenida la nueva sesión, se verificó el nivel de privilegios alcanzado.

Comandos ejecutados: whoami & id

```
root@cap:~# whoami
root
root@cap:~# id
uid=0(root) gid=1001(nathan) groups=1001(nathan)
root@cap:~#
```

Con esto se confirmó el escalamiento exitoso a privilegios de administrador.

Obtención de la flag de root

Con acceso completo al sistema, se procedió a localizar y leer la flag correspondiente al usuario root.

Comandos ejecutados: cd / , ls , cd root & cat root.txt

```
root@cap: ~# cd /
root@cap:/# ls
bin boot cdrom dev etc home lib lib32 lib64 libx32 lost+found media mnt opt proc root run sbin snap srv sys tmp usr var
root@cap:/# cd root
root@cap:/root# ls
root.txt  snap
root@cap:/root# cat root.txt
d1e2befcee9da3f2b78644ef179f75ca
root@cap:/root#
```

Conclusión general

La resolución de la máquina permitió aplicar de manera práctica y estructurada las distintas fases que conforman un proceso de pentesting, desde el reconocimiento inicial hasta la obtención de privilegios administrativos. A lo largo del ejercicio fue posible comprobar la importancia de una correcta enumeración y análisis de la información expuesta, ya que los vectores de ataque identificados surgieron a partir de configuraciones inseguras y exposiciones innecesarias dentro del sistema.

Durante la fase de reconocimiento y enumeración se identificaron los servicios accesibles y la presencia de una aplicación web sin mecanismos adecuados de control de acceso. El análisis de dicha aplicación permitió descubrir la exposición de archivos de captura de tráfico, lo cual derivó en la obtención de credenciales transmitidas en texto claro. Este hallazgo fue clave para establecer el acceso inicial al sistema, demostrando cómo una mala gestión de la información puede comprometer la seguridad de un entorno completo.

Una vez obtenido el acceso al sistema, la enumeración local resultó determinante para avanzar en el proceso. La ausencia de privilegios sudo obligó a explorar otros vectores de escalamiento, lo que llevó a la identificación de binarios con capacidades especiales asignadas. El abuso de la capacidad cap_setuid en el binario python3.8 permitió escalar privilegios de forma directa y controlada, sin necesidad de explotar vulnerabilidades del kernel ni recurrir a técnicas complejas.

En conjunto, esta máquina evidenció cómo la combinación de servicios mal configurados, exposición de información sensible y una gestión incorrecta de privilegios puede facilitar la toma completa de un sistema. El ejercicio reforzó la importancia de seguir una metodología ordenada y de documentar cada etapa del proceso, permitiendo una comprensión clara de los pasos necesarios para lograr la resolución exitosa de la máquina dentro del entorno de Hack The Box.

REPORTE TÉCNICO DE PENTESTING

KeruDWL

Hack The Box – Cap

2025

GitHub · KeruDWL

HTB · KeruDWL