SWAP(2022-2023)
Grado en Ingeniería Informática
Universidad de Granada

Pivoting en Servidores web

Rubén De la Torre Vico
Mario Daniel Castro Almenzar

9 de junio de 2023

UNIVERSIDAD DE GRANADA

Identificador de Wiki: 10
Horas invertidas: 25h

Índice

Índice de figuras

1. Introducción al Penetration Testing

1.1. ¿Qué es el Penetration Testing?

El Penetration Testing o la prueba de penetración, en español; es una forma de simular los métodos que un atacante podría utilizar para eludir los controles de seguridad y obtener acceso a los sistemas de una organización, como una granja web.

Sin embargo, el pentesting va más allá de simplemente ejecutar escáneres y herramientas automatizadas y luego redactar un informe. Este requiere de experiencia en el mundo real para adquirir habilidades y conocimientos necesarios para realiazar estas pruebas de una manera efectiva.

Actualmente, no existe un estándar definido que deba ser pasado por ley. Aun así, la *OWASP[4] (OrganizationWeb Security Testing Guide)*, una organización sin ánimo de lucro que se dedica a mejorar la seguridad de las aplicaciones web; recomienda seguir el estándar definido *Penetration Testing Execution Standar (PTES)[7]* que define esta prueba en 6 pasos, en los cuales se detallan procedimientos y herramientas.

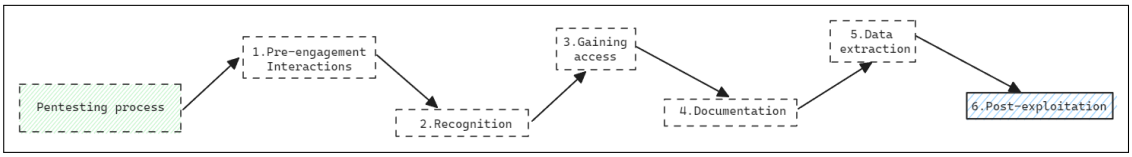

Figura 1: Fases del pentesting

1.2. Las fases del pentesting

1. Pre-engagement Interactions:
   - **Recopilación de información:** Se obtiene información sobre el alcance y los objetivos del pentesting, así como sobre el cliente y su infraestructura.
   - **Acuerdo de confidencialidad:** Se firma un acuerdo de confidencialidad entre el cliente y el equipo de pentesting para proteger la información sensible.
   - **Definición del alcance:** Se determina el alcance del pentesting, incluyendo las redes, sistemas y aplicaciones que se van a evaluar.

2. Reconocimiento:
    - **Enumeración de objetivos:** Se identifican y enumeran los objetivos del pentesting, como direcciones IP, nombres de dominio, aplicaciones, etc.
    - **Recopilación de información:** Se recopila información adicional sobre los objetivos, como registros DNS, subdominios, arquitectura de red, etc.
3. Obtención de acceso:
    - **Detección de vulnerabilidades:** Se buscan vulnerabilidades en los sistemas y aplicaciones para obtener acceso no autorizado.
    - **Explotación de vulnerabilidades:** Se aprovechan las vulnerabilidades encontradas para obtener acceso privilegiado a los sistemas y aplicaciones.
4. Post-explotación:
    - **Mantenimiento del acceso:** Se implementan técnicas para mantener el acceso a los sistemas comprometidos de forma persistente.
    - **Escalada de privilegios:** Se intenta obtener privilegios más altos en los sistemas comprometidos para acceder a información adicional o realizar acciones más críticas.
5. Extracción de datos:
    - **Recopilación de información sensible:** Se busca y extrae información sensible de los sistemas comprometidos, como contraseñas, datos confidenciales, etc.
    - **Análisis de datos:** Se analiza la información recopilada para identificar su valor y determinar las implicaciones de seguridad.
6. Documentación:
    - **Informe final:** Se genera un informe detallado que documenta los hallazgos, las vulnerabilidades descubiertas y las recomendaciones de mitigación.
    - **Presentación:** Se presenta el informe final al cliente, incluyendo una explicación de los hallazgos y las recomendaciones para mejorar la seguridad.

En este trabajo vamos a centrarnos en la parte de la post-explotación, incluyendo distintas definiciones, técnicas y ejemplos típicos de esta fase.

2. Antecedentes. Fase de Post-Explotación

2.1. ¿Qué es la Post-Explotación?

La Post-Explotación se refiere al conjunto de acciones realizadas después de obtener acceso inicial a un sistema.

Según la *OWASP[3] (OrganizationWeb Security Testing Guide)*, consta de una amplia serie de objetivos, que comprenden desde la recopilación de información adicional o el mantenimiento del acceso hasta la manipulación del entorno vulnerado.

En esta fase, el pentester intenta maximizar la utilidad de su acceso recién ganado a un sistema o red. Esto puede implicar una serie de acciones, como la escalada de privilegios, el establecimiento de una presencia persistente en el sistema, el robo de datos confidenciales o la utilización del sistema comprometido como un "pivote"para atacar otros sistemas en la misma red.

2.2. Objetivos de la Post-Explotación

- **Escalada de privilegios:** Incluso después de obtener acceso a un sistema, podemos encontrar restricciones basadas en los privilegios del usuario. Por lo tanto, un objetivo común en esta fase es elevar estos privilegios, a menudo con el objetivo de obtener el control total del sistema (es decir, acceso con los mismos permisos que el administrador del sistema).

  A este tipo de cambio de privilegios en un sistema se le denominan ***movimientos verticales***, mientras que el ir cambiando de usuario y/o sistema dentro de la subred a la que esté conectado el dispositivo para buscar más información o vulnerabilidades se le suelen denominar ***movimientos laterales***.

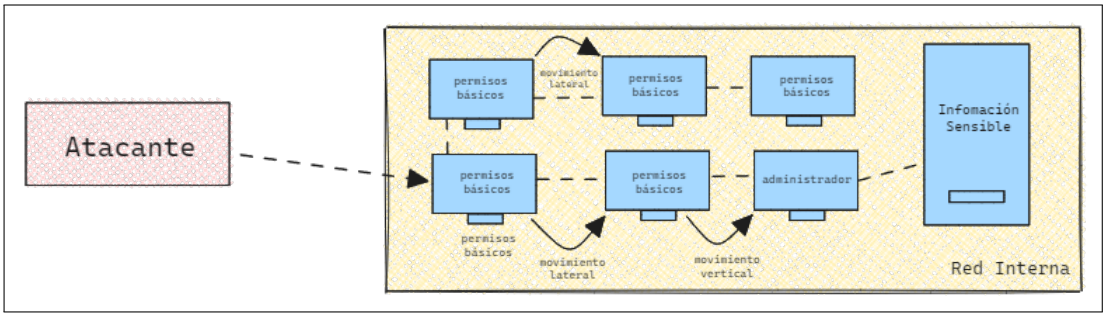

Figura 2: Movimientos laterales vs Movimientos verticales

- **Mantener el Acceso:** Una vez que se ha obtenido el acceso, es importante mantenerlo para futuras operaciones. Esto puede implicar la creación de cuentas de usuario, la modificación de servicios existentes o la instalación de software adicional.

- **Recopilación de información adicional:** Ayuda a entender mejor el entorno e identificar otros objetivos potenciales. Esto puede incluir la recopilación de contraseñas, la identificación de roles de usuario y la recopilación de datos sensibles.

- **Manipulación del Entorno:** Puede implicar la modificación de la configuración del sistema, la alteración de los datos o la instalación de software malicioso. El objetivo es aumentar el control sobre el sistema y expandir las capacidades de explotación.

- **Limpieza:** Eliminar cualquier rastro de la actividad del atacante en el sistema. Esto puede incluir la eliminación de archivos y registros, así como la restauración de la configuración del sistema a un estado donde no haya rastro de la intrusión. Por ejemplo, algunas prácticas comunes son:

    1. **Limpiar los archivos de registro (logs)**: Los sistemas operativos Linux mantienen un registro de los eventos del sistema en varios archivos de log ubicados en el directorio /var/log.

        ```
        shred -zu /var/log/auth.log
        echo "" > /var/log/auth.log
        ```

    2. **Eliminar el historial de comandos**: En sistemas Linux, los comandos ejecutados en la terminal son almacenados en un archivo de historial (por lo general `~/.bash_history` para la shell Bash).

        ```
        history -c && history -w
        ```

    3. **Eliminar archivos temporales**: Los archivos temporales se pueden acumular y potencialmente contener información sobre actividades sospechosas. Se pueden eliminar con rm o find.

        ```
        rm -rf /tmp/*
        ```

        ```
        find /var/tmp -type f -delete
        ```

3. Pivoting

3.1. ¿Qué es el Pivoting?

Según define la conocida empresa de ciberseguridad EC-Council [2], entendemos por *Pivoting* al acto de utilizar un sistema comprometido para propagarse entre diferentes sistemas informáticos de la red interna de la organización auditada.

Pivotar está estrechamente relacionado con el concepto de movimiento lateral, propio de la fase de post exploatación.

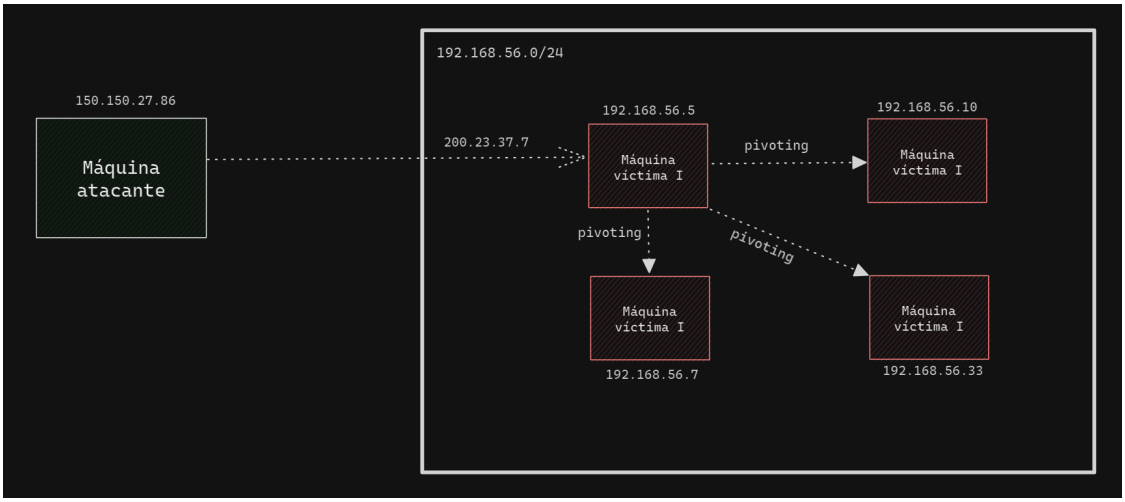

Figura 3: Esquema básico de Pivoting

En esta sección vamos a abordar algunos casos de ejemplos de **casos reales** (3.2) ocurridos recientemente, las **principales técnicas** (3.3) que existen de Pivoting, un **ejemplo de post-explotación** (3.4) usando pivoting dentro de un entorno controlado, así como **medidas prevenentivas** (3.5) recomendadas para evitar que pueda aplicarse esta técnica en nuestro propio servidor.

3.2. Casos reales de ataques

Recientemente ha habido varios casos en los que se han vulnerado sistemas utilizando la técnica de pivoting. Un ejemplo notable es el del grupo de ciberdelincuentes **Lapsus$** [11], que ha llevado a cabo una serie de ciberataques de alto perfil contra varias empresas tecnológicas [8].

Por ejemplo, en enero de 2022, Lapsus$ ganó acceso a los servidores de la empresa de gestión de identidad y acceso, **Okta**, a través de la cuenta comprometida de un ingeniero de soporte al cliente de terceros. Usando el acceso que este personal de soporte tenía, Lapsus$ pudo violar el Slack[1] interno de Okta, Jira, y el panel administrativo de backend utilizado para asistir a los clientes.

En febrero de 2022, la compañía tecnológica **Nvidia** se dio cuenta de una violación en sus sistemas. Lapsus$ afirmó tener un terabyte de datos de Nvidia y amenazó con liberar los "*archivos completos de silicio, gráficos y conjuntos de chips informáticos para todas las GPU NVIDIA recientes, incluyendo la RTX 3090Ti y las próximas revisiones*"si Nvidia no abría el código fuente de sus controladores de dispositivos. Las credenciales de más de 71.000 empleados de Nvidia aparecieron en línea.

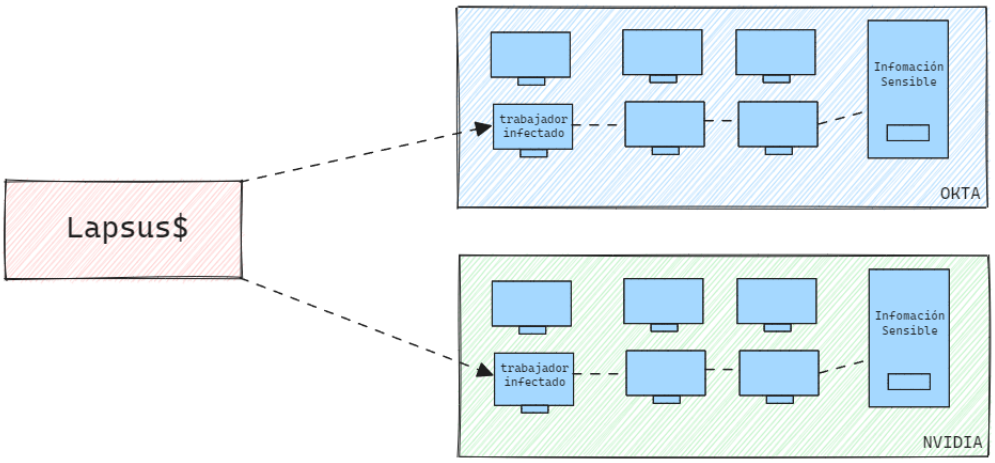

Figura 4: Patrón de ataque de banda cibercriminal Lapsus$

En términos de la técnica de pivoting específicamente, parece que Lapsus$ pudo haber utilizado las credenciales débiles de los empleados para ganar acceso inicial y luego moverse lateralmente dentro de las redes de estas organizaciones.

[1] **Slack**: plataforma de comunicación para equipos que una empresa o grupo específico ha creado para su propio uso.

3.3. Técnicas de Pivoting

1. **Port forwarding (Reenvío de puertos)**: El atacante crea un túnel entre dos máquinas a través de puertos TCP/IP abiertos, reenviando paquetes y tráfico de uno a otro. Existen varias formas de llevar esto a cabo:

   - **Local Port forwarding**: Reenviar un puerto de tu ordenador a un puerto del servidor SSH:

     `ssh -L <Puerto local>:<IP remota>:<Puerto remoto> usuario@<IP remota>`

     Con este comando cualquier conexión que salga de tu ordenador por el puerto local asignado será reenviada por un túnel SSH al servidor indicado por la IP remota.

   - **Remote Port forwarding**: Se conecta un puerto del servidor con un puerto del atacante, para darle conexión con otra IP determinada, es decir, se usa el servicio vulnerado como túnel con su red interna.

   - **Dynamic port forwarding**: El atacante crea un servidor proxy `SOCKS` para el tráfico de túneles, con la máquina comprometida actuando como intermediario entre la máquina del atacante y la red interna de la máquina víctima.

2. **VPN pivoting**: El atacante inicia un cliente de red privada virtual (VPN) en la máquina comprometida, accediendo a un servidor VPN remoto. Así las direcciones privadas indicadas en el cliente se convierten en las direcciones privadas de la red interna de la máquina vulnerada.

3. **Routing tables**: El atacante cambia la tabla de enrutamiento de la máquina comprometida para agregar una nueva ruta. Esta ruta requerirá que cualquier tráfico enviado el atacante sea redirigido a otros dispositivos de la red interna previamente identificados.

3.4. Ejemplo de Pivoting. Máquina Badbyte

Para comprender con mayor facilidad esta técnica de post-explotación de un servidor, se recomienda probar a resolver problemas diseñados para practicar estas técnicas en instancias de laboratorios aislados, en un entorno legal y controlado. Estos son proporcionados por sitios web como **HackTheBox**, **TryHackMe** o **VulnHub**.

Tras investigar, hemos encontrado la siguiente máquina de TryHackMe llamada "**Badbyte**"[9], que presenta un desafío de pivoting como parte de su ruta de explotación.

En este caso, tras ganar un acceso inicial a través de una filtración de credenciales para el servicio de ssh, es necesario pivotar a través de la red interna para lograr la escalada de privilegios.

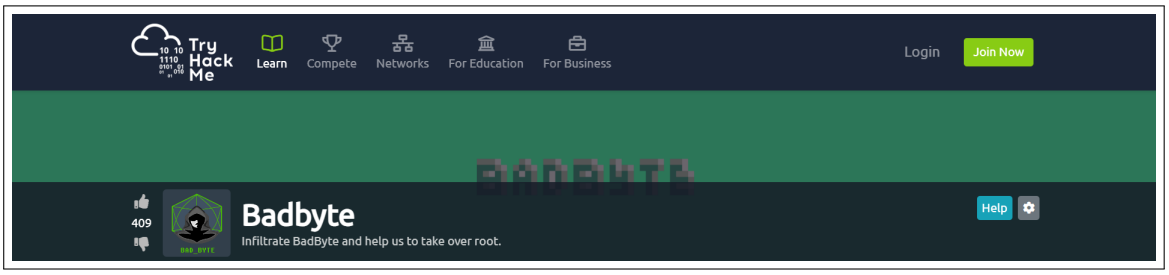

Figura 5: Máquina Badbyte. TryHackMe

La máquina inicia con un escaneo de puertos para identificar servicios abiertos.

- Puertos abiertos: 2
- Primer puerto abierto: SSH (22)
- Segundo puerto abierto: FTP (30024)

```
Open 10.10.48.223:22
Open 10.10.48.223:30024

...

PORT      STATE SERVICE REASON  VERSION
22/tcp    open  ssh     syn-ack OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
30024/tcp open  ftp     syn-ack vsftpd 3.0.3
```

Figura 6: Reconocimiento de puertos

Tras descubrir y conseguir una conexión ssh con un usuario limitado. El siguiente paso es la enumeración de la red interna de la máquina objetivo.

Se descubre una nueva máquina en la red interna a la que se puede acceder. Para llegar a esta máquina, se configura un port forwarding dinámico utilizando el comando SSH con la opción `-D` para establecer un proxy `SOCKS`. Esto permite que el tráfico se redirija a través de la máquina comprometida hacia la red interna.

```
ssh -i id_rsa -D 1337 errorcauser@target
```

```
cat /etc/proxychains.conf
...
#socks4127.0.0.1 9050
socks5 127.0.0.1 1337
```

Figura 7: Configuración Dynamic Port Forwarding

Al ejecutar el comando `nmap` para descubrir los puertos TCP que están escuchando en localhost se descubre que son el puerto 80 para `http` y el 2206 para `mysql` se estan ejecutando en la red interna. Una vez conocida esta información, es posible crear un túnel `ssh` para acceder a este puerto a través de nuestro equipo local.

```
ssh -i id_rsa -L 80:127.0.0.1:80 errorcauser@target
```

Figura 8: Creación de un túnel para acceder al localhost remoto

Este ejemplo demuestra la importancia del pivoting en las pruebas de penetración. Esta capacidad es muy importante más en el concepto de una granja web donde los distintos servicios se encuentran separados en distintas máquinas por lo que si se domina bien este concepto una mala configuración por parte del administrador del sistema puede acabar en una brecha de información crítica.

3.5. Medidas preventivas y buenas prácticas

Algunas de las medidas [1] que pueden ayudar a prevenir el pivoting son:

- **Segmentación de la red:** dividir la red en segmentos separados (front-rail y back-rail) para evitar que los atacantes se muevan lateralmente una vez que hayan comprometido una máquina. Esto se puede hacer mediante el uso de redes privadas distintas, firewalls, etc. Un ejemplo sería mediante el uso de una buena configuración de red como la DMZ doble que dimos en teoría teniendo una buena configuración de `iptables` tal y como hemos hecho en prácticas.

- **Política de mínimo privilegio:** Los usuarios y aplicaciones deben tener el mínimo nivel de privilegios necesario para realizar sus funciones. Esto reduce la posibilidad de que un atacante gane acceso a recursos críticos mediante la escalada de privilegios.

- **Autenticación multifactor:** Esto proporciona una capa adicional de seguridad que puede dificultar que los atacantes se muevan lateralmente utilizando credenciales robadas.

- **Monitoreo y alerta:** Un sistema de detección de intrusos (**IDS**) y un sistema de gestión de eventos e información de seguridad (**SIEM**) pueden ser útiles para identificar y alertar sobre comportamientos sospechosos que pueden indicar un pivoting.

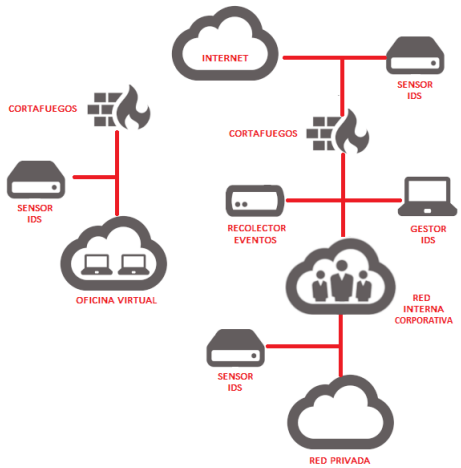

Figura 9: Ejemplo de sistema IDS

4. Escalada de privilegios

La *escalada de privilegios* [10] implica la explotación de errores de software, fallos de diseño o malas configuraciones para obtener control adicional del sistema después de la explotación [**Penetration Testing**].

Es posible que algunos ataques no den como resultado la obtención de los privilegios que nos gustaría obtener. Obtener privilegios, comprometer una cuenta de usuario sin derechos de administrador o explotar un servicio de escucha con privilegios limitados puede conducir al acceso del sistema, y aún así seguir trabajando como usuario limitado. Para obtener los privilegios deseados, es necesario explotar otros problemas. Existen una serie de herramientas como **Metasploit** [5] o **PEAS-ng** que agilizan y facilitan este proceso. [6]

4.1. Escalada de privilegios en Linux

Una vez hemos accedido y obtenido una shell interactiva en el servidor debemos de encontrar una vulnerabilidad local que permita la escalada de privilegios. Para ello, necesitamos averiguar un poco de información sobre el sistema, como por ejemplo:

- Versión del Kernel instalado: `uname -a`
- Versión de Ubuntu: `lsb_release -a`

```
uname -a
Linux ubuntu 2.6.27-7-generic #1 SMP Fri Oct 24 06:42:44 UTC 2008 i686
GNU/Linux
lsb_release -a ID de
distribuidor: Ubuntu
Descripción: Ubuntu 8.10
Lanzamiento: 8.10
Codename: intrepid
```

Figura 10: Recopilación de información Local

En el ejemplo anterior podemos observar que este sistema está muy desactualizado, por lo que es potencialmente vulnerable. En este caso se presenta una vulnerabilidad `CVE-2009-1185`.

Como se puede observar, nos hemos referido a ella con un identificador específico con una sintaxis determinada:

- **CVE**: Son las siglas de *Common Vulnerabilities and Exposures*. Se trata de una lista de vulnerabilidades y exposiciones de seguridad de la información divulgadas públicamente.
- **2009**: Año de publicación de la vulnerabilidad.
- **1185**: Identificador numerado cronológicamente para cada año de la vulnerabilidad encontrada.

No obstante, es posible acceder a la base de datos actualizada con todas las vulnerabilidades que van apareciendo para todo tipo de sistemas. Para ello, se puede acceder a la web oficial de CVE:

`https://www.cve.org/`

4.2. Encontrar un exploit en Linux

Kali Linux (distribución de linux pensada para el pentesting) incluye un repositorio local de exploits públicos de ExploitDB, que contiene además una utilidad llamada `searchsploit`, la cual puede usarse para buscar la manera de explotar determinados CVEs. De cualquier modo, puede accederse a estos exploits desde cualquier navegador a través del la web oficial:

`https://www.exploit-db.com/`

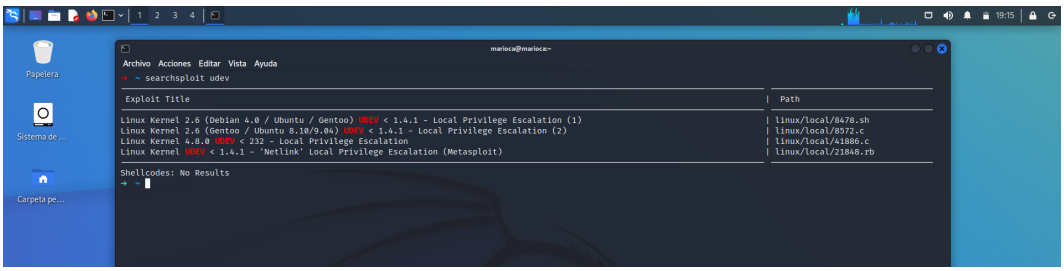

Figura 11: Funcionamiento de searchsploit

Tal y como se puede observar en la imagen, al realizar la búsqueda de una vulnerabilidad concreta genera en la salida una tabla con dos columnas; la primera columna se refiere al título del exploit mientras que la segunda indica la ruta en el que se encuentra dicho exploit.

Una vez nos hemos cerciorado de como funciona el exploit que vamos a utilizar para hacer la escalada de privilegios, necesitamos ponernos en escucha en nuestro sistema para capturar la reverse shell, para ello ejecutamos `nc -lvp [puerto]` para abrir una conexión TCP por el puerto indicado:

```
root@kali:~# nc -lvp 12345
escuchando en [cualquiera] 12345 ...
```

Finalmente ejecutamos el exploit. Para verificar que que se ha ganado acceso `root` a la máquina simplemente hacemos uso del comando `whoami`.

```
root@kali:~# nc -lvp 12345
escuchando en [cualquiera] 12345 ...
192.168.20.11: búsqueda inversa de host fallida: Error de servidor
desconocido : Conexión expirada
conectar con [192.168.20.9] desde (UNKNOWN) [192.168.20.11] 33191
whoami
raíz
```

Figura 12: Obtención de privilegios de root

5. Conclusiones

En resumen, podemos destacar la trascendencia de las pruebas de seguridad en el despliegue de servicios como una granja web. Dada la gran inversión en términos de tiempo y recursos financieros que estos proyectos requieren, dichas pruebas nos permiten, en nuestro rol de administradores, evaluar la efectividad de la configuración establecida e identificar posibles necesidades de ajustes.

Es innegable que la seguridad es una preocupación central en la actualidad. Una brecha de seguridad puede tener consecuencias devastadoras para una empresa, resultando en pérdidas financieras significativas y daño a su reputación. Por lo tanto, es crucial realizar pruebas de seguridad controladas para identificar y corregir vulnerabilidades antes de que puedan ser explotadas por actores malintencionados.

Queremos enfatizar, como reflexión final, la necesidad de una seguridad integral. Como se ha demostrado en el caso del pivoting, cualquier punto de acceso, por mínimo que sea, puede dar lugar a una brecha de seguridad más amplia. Por lo tanto, no solo es necesario proteger el código, sino también garantizar la seguridad de las configuraciones de los equipos, el software utilizado, las redes y la seguridad física de las instalaciones de la empresa. No podemos olvidar que incluso la política de gestión de contraseñas más robusta puede quedar inservible si un individuo no autorizado logra acceder a la sala de servidores sin autenticación.

6. Bibliografía

[1] José Manuel Ortega Candel. *Ciberseguridad. Manual práctico*. 2021.

[2] EC Council. *Pivoting in Penetration Testing*. 2023. URL: https://www.eccouncil.org/cybersecurity-exchange/penetration-testing/pivoting-penetration-testing.

[3] OWASP Foundation. *Penetration Testing Methodologies*. 2023. URL: https://owasp.org/www-project-web-security-testing-guide/latest/3-The_OWASP_Testing_Framework/1-Penetration_Testing_Methodologies.

[4] OWASP Foundation. *Web Security Testing Guide*. 2023. URL: https://owasp.org/www-project-web-security-testing-guide.

[5] David Kennedy et al. *Metasploit: the penetration tester's guide*. No Starch Press, 2011.

[6] Carlos Polop. *PEASS-ng: Privilege Escalation Awesome Scripts SUITE new generation*. 2023. URL: https://github.com/carlospolop/PEASS-ng.

[7] Penetration Testing Execution Standard. *PTES Technical Guidelines*. Accedido el: 9 de Junio, 2023. 2023. URL: http://www.pentest-standard.org/index.php/.

[8] TechRadar. *What We Know About Lapsus$ and Okta So Far*. 2023. URL: https://www.techradar.com/news/what-we-know-about-lapsusdollar-and-okta-so-far.

[9] TryHackMe. *BadByte Room*. 2023. URL: https://tryhackme.com/room/badbyte.

[10] Unknown. *Penetration Testing. A Hands-On Introduction to Hacking*. Chapter 13: Post Exploitation. 2023.

[11] Wikipedia. *Lapsus$*. 2023. URL: https://en.wikipedia.org/wiki/Lapsus$.