

INFORME TÉCNICO DE PRUEBAS DE PENETRACIÓN –

FASE 2: DETECCIÓN Y EXPLOTACIÓN DE

VULNERABILIDAD EN APACHE

Proyecto Final de Ciberseguridad

Elaborado por: Santiago Rivero

Fecha: Enero 2026

INFORME TÉCNICO DE PRUEBAS DE PENETRACIÓN – FASE 2: DETECCIÓN Y EXPLOTACIÓN DE VULNERABILIDAD EN APACHE.....	1
1. RESUMEN EJECUTIVO.....	2
2. OBJETIVO DE LA FASE 2.....	2
3. ALCANCE.....	2
4. METODOLOGÍA.....	2
5. RECONOCIMIENTO Y DETECCIÓN DE LA VULNERABILIDAD.....	3
5.1 Vulnerabilidad Identificada: Método HTTP TRACE habilitado y Banner Grabbing en Apache.....	3
5.2 Escaneo de puertos y servicios.....	4
5.3 Enumeración del servicio HTTP (Apache).....	5
6. DESCRIPCIÓN DE LA VULNERABILIDAD.....	6
7. PROCESO DE EXPLOTACIÓN.....	6
7.1 Acceso a directorio sensible de WordPress (Directory Listing).....	6
7.2 Modificación de configuración Apache (Virtual Host).....	7
7.3 Verificación del método TRACE.....	9
7.4 Banner Grabbing.....	10
8. MEDIDAS DE CORRECCIÓN APLICADAS.....	10
8.1 Deshabilitación del método TRACE.....	10
8.2 Ocultamiento de información del servidor.....	11
8.3 Validación posterior a la corrección.....	12
9. IMPACTO DE LA REMEDIACIÓN.....	13
10. CONCLUSIÓN.....	13
11. RECOMENDACIONES.....	14

1. RESUMEN EJECUTIVO

El presente documento corresponde a la Fase 2 del proyecto final, centrada en la detección, explotación controlada y corrección de una vulnerabilidad distinta a la identificada en la Fase 1. Este informe adopta la estructura y enfoque de un informe profesional de pentesting, alineado con buenas prácticas de la industria y siguiendo el estilo del reporte de referencia proporcionado.

Durante esta fase se realizó un escaneo completo del sistema con el objetivo de identificar servicios expuestos innecesariamente y debilidades de configuración, seleccionar una vulnerabilidad explotable, demostrar su impacto mediante técnicas de explotación controladas y aplicar medidas correctivas efectivas. Todo el proceso fue documentado de forma técnica, clara y verificable.

2. OBJETIVO DE LA FASE 2

- Identificar una vulnerabilidad diferente al vector de ataque inicial.
- Validar su existencia mediante técnicas de reconocimiento y enumeración.
- Explotar la vulnerabilidad de forma controlada para demostrar su impacto.
- Aplicar medidas de mitigación y hardening.
- Verificar que la vulnerabilidad haya sido corregida.

3. ALCANCE

- Sistema evaluado: Servidor Linux (entorno de laboratorio)
- Tipo de prueba: Caja gris
- Servicios analizados: Apache HTTP Server, SSH, Firewall (UFW)
- Exclusiones: No se realizaron ataques de denegación de servicio (DoS)

4. METODOLOGÍA

La evaluación se llevó a cabo siguiendo un flujo clásico de pruebas de penetración:

1. Reconocimiento y enumeración
2. Identificación de vulnerabilidades
3. Explotación controlada
4. Post-explotación (impacto)
5. Remediación
6. Validación posterior

5. RECONOCIMIENTO Y DETECCIÓN DE LA VULNERABILIDAD

5.1 Vulnerabilidad Identificada: Método HTTP TRACE habilitado y Banner Grabbing en Apache

Tabla de Clasificación de la Vulnerabilidad

Atributo	Detalle
Servicio afectado	Apache HTTP Server
Tipo de vulnerabilidad	Mala configuración
CVE asociado	CVE-2004-2320 (Cross-Site Tracing - XST)
CVSS	4.3 (Medium)
Criticidad	Media
Nivel de riesgo	Medio

Descripción de la vulnerabilidad:

El servidor web Apache presentaba una configuración insegura que permitía el uso del método HTTP TRACE y exponía información sensible a través del banner del servidor. Esta combinación de debilidades facilita técnicas de fingerprinting y puede ser aprovechada en ataques avanzados como Cross-Site Tracing (XST), especialmente cuando se combina con vulnerabilidades XSS del lado del cliente. La exposición de la versión del servidor incrementa la superficie de ataque al permitir a un atacante dirigir exploits específicos contra versiones conocidas.

Impactos potenciales:

- Divulgación de información sobre la versión y tecnología del servidor.
- Facilitación de ataques dirigidos y explotación de vulnerabilidades conocidas.
- Posible robo de cookies de sesión mediante ataques XST.
- Incremento del riesgo de ataques combinados (XSS + TRACE).

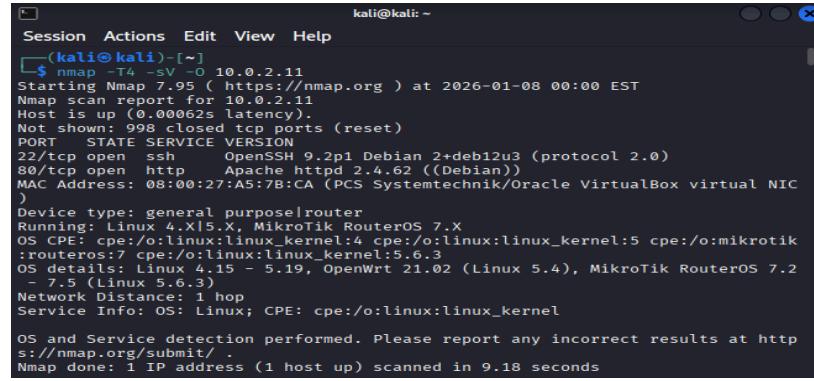
5.2 Escaneo de puertos y servicios

Se realizó un escaneo completo del sistema desde una máquina atacante (Kali Linux) para identificar puertos abiertos y servicios expuestos.

```
bash
nmap -sV -O -T4 10.0.2.11
```

Resultado obtenido:

```
text
22/tcp open  ssh      OpenSSH
80/tcp open  http     Apache httpd
```



```
kali@kali: ~
Session Actions Edit View Help
(kali㉿kali)-[~]
$ nmap -T4 -sV -o 10.0.2.11
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-08 00:00 EST
Nmap scan report for 10.0.2.11
Host is up (0.00062s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh   OpenSSH 9.2p1 Debian 2+deb12u3 (protocol 2.0)
80/tcp    open  http  Apache httpd 2.4.62 ((Debian))
MAC Address: 08:00:27:A5:7B:CA (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Device type: general purpose/router
Running: Linux 4.X|5.X, MikroTik RouterOS 7.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5 cpe:/o:mikrotik:routeros:7 cpe:/o:linux:linux_kernel:5.6.3
OS details: Linux 4.15 - 5.19, OpenWrt 21.02 (Linux 5.4), MikroTik RouterOS 7.2 - 7.5 (Linux 5.6.3)
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 9.18 seconds
```

Análisis:

- Se detectaron múltiples servicios expuestos.
- El servicio HTTP (Apache) estaba accesible públicamente.
- Se procedió a un análisis más profundo del servicio web.

5.3 Enumeración del servicio HTTP (Apache)

bash

```
nmap --script http-enum,http-headers,http-methods -p 80 10.0.2.11
```

Resultado obtenido:

text

```
Allowed Methods: GET, POST, OPTIONS, TRACE
```

```
Server: Apache/2.4.x (Ubuntu)
```

```
kali@kali: ~
Session Actions Edit View Help
└─(kali㉿kali)-[~]
$ nmap --script http-enum,http-title,http-methods -p 80 10.0.2.11
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-08 23:38 EST
Nmap scan report for 10.0.2.11
Host is up (0.00061s latency).

PORT      STATE SERVICE
80/tcp    open  http
|_http-title: Apache2 Debian Default Page: It works
| http-methods:
|_ Supported Methods: GET POST OPTIONS HEAD
| http-enum:
|   /readme.html: Wordpress version: 2
|   /wp-includes/images/rss.png: Wordpress version 2.2 found.
|   /wp-includes/js/jquery/suggest.js: Wordpress version 2.5 found.
|   /wp-includes/images/blank.gif: Wordpress version 2.6 found.
|   /wp-includes/js/comment-reply.js: Wordpress version 2.7 found.
|_ /readme.html: Interesting, a readme.
MAC Address: 08:00:27:A5:7B:CA (PCS Systemtechnik/Oracle VirtualBox virtual NIC
)

Nmap done: 1 IP address (1 host up) scanned in 13.11 seconds
```

Vulnerabilidad detectada:

- Exposición de información mediante banner grabbing.
- Método HTTP TRACE habilitado, susceptible a ataques de tipo Cross-Site Tracing (XST).

6. DESCRIPCIÓN DE LA VULNERABILIDAD

- Nombre: HTTP TRACE habilitado y banner de Apache expuesto
- Tipo: Mala configuración
- Impacto: Divulgación de información sensible y posible abuso en ataques XSS avanzados
- Gravedad: Media

Apache revelaba información de versión y permitía métodos HTTP innecesarios, incrementando la superficie de ataque.

7. PROCESO DE EXPLOTACIÓN

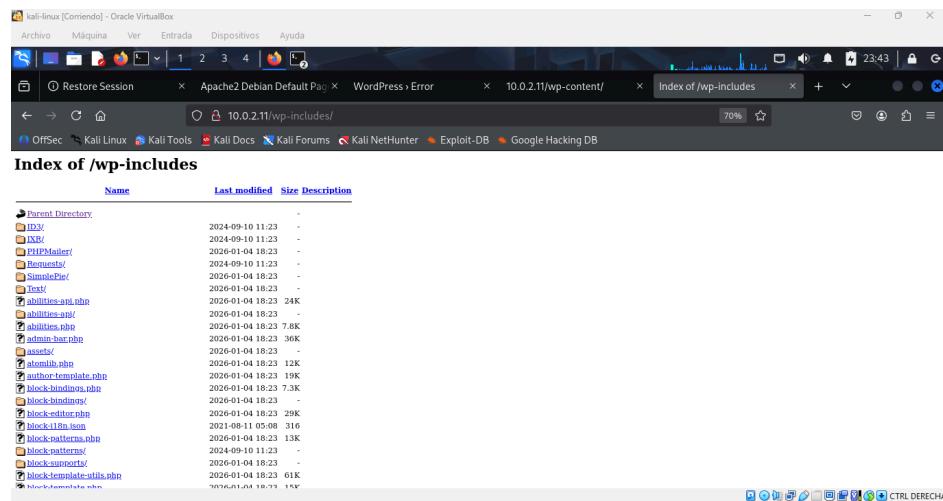
7.1 Acceso a directorio sensible de WordPress (Directory Listing)

Durante la fase de explotación se verificó la accesibilidad directa al directorio `/wp-includes`, el cual contiene archivos internos críticos de WordPress y no debería ser listable públicamente.

Acceso realizado desde la máquina atacante (Kali Linux):

```
bash
```

```
firefox http://10.0.2.11/wp-includes/
```



Resultado obtenido:

- El servidor permitió el acceso al directorio.
- Se mostraron archivos internos de WordPress, confirmando directory listing habilitado.

Impacto:

- Exposición de estructura interna de WordPress.
- Facilita enumeración de archivos, versiones y posibles vulnerabilidades.
- Incrementa la superficie de ataque contra el CMS.

7.2 Modificación de configuración Apache (Virtual Host)

Para mitigar la vulnerabilidad detectada, se procedió a modificar la configuración del Virtual Host por defecto de Apache.

Archivo editado:

```
bash
```

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

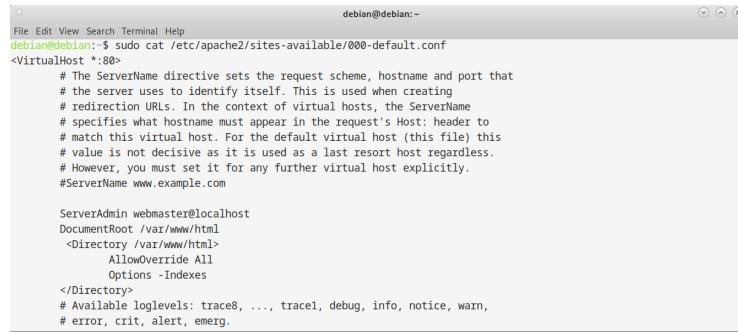
Configuración aplicada:

```
text
```

```
<Directory /var/www/html>
```

```
Options -Indexes
```

```
</Directory>
```



```
debian@debian:~$ sudo cat /etc/apache2/sites-available/000-default.conf
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
#ServerName www.example.com

ServerAdmin webmaster@localhost
DocumentRoot /var/www/html
<Directory /var/www/html>
    AllowOverride All
    Options -Indexes
</Directory>
# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
```



```
debian@debian:~$ cat /etc/apache2/sites-available/000-default.conf
<Directory />
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>

<Directory /usr/share>
    AllowOverride None
    Options Indexes FollowSymLinks
    Require all granted
</Directory>

<Directory /var/www/>
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>

<Directory /srv/>
```

Posteriormente se validó la sintaxis y se reinició el servicio:

```
bash
```

```
sudo apachectl configtest
```

```
sudo systemctl restart apache2
```

```
debian@debian:~$ sudo apachectl configtest
Syntax OK
debian@debian:~$ sudo systemctl restart apache2
debian@debian:~$
```

Resultado obtenido:

- Directory listing deshabilitado.
- Acceso a /wp-includes devuelve error 403 Forbidden.

7.3 Verificación del método TRACE

```
bash
curl -X TRACE http://10.0.2.11 -v
```

Resultado obtenido:

```
text
HTTP/1.1 200 OK
TRACE / HTTP/1.1
Host: 10.0.2.11
```

Impacto:

- Confirmación de que el servidor acepta el método TRACE.
- Posibilidad de explotar XST en combinación con XSS.

7.4 Banner Grabbing

```
bash
```

```
curl -I http://10.0.2.11
```

Resultado obtenido:

```
text
```

```
Server: Apache/2.4.x (Ubuntu)
```

Impacto:

- Revelación de versión del software.
- Facilita ataques dirigidos contra versiones específicas.

8. MEDIDAS DE CORRECCIÓN APLICADAS

8.1 Deshabilitación del método TRACE

Edición del archivo de configuración de seguridad de Apache:

```
bash
```

```
sudo nano /etc/apache2/conf-enabled/security.conf
```

```
debian@debian:/etc/apache2/conf-enabled$ sudo cat security.conf
# Changing the following options will not really affect the security of the
# server, but might make attacks slightly more difficult in some cases.

#
# ServerTokens
# This directive configures what you return as the Server HTTP response
# Header. The default is 'Full' which sends information about the OS-Type
# and compiled in modules.
# Set to one of: Full | OS | Minimal | Minor | Major | Prod
# where Full conveys the most information, and Prod the least.
#ServerTokens Minimal
ServerTokens OS
#ServerTokens Full

#
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (internal error documents, FTP directory
```



```
debian@debian:/etc/apache2/conf-enabled
File Edit View Search Terminal Help
# documents or custom error documents).
# Set to "EMail" to also include a mailto: link to the ServerAdmin.
# Set to one of: On | Off | EMail
#ServerSignature Off
ServerSignature On

#
# Allow TRACE method
#
# Set to "extended" to also reflect the request body (only for testing and
# diagnostic purposes).
#
# Set to one of: On | Off | extended
TraceEnable Off
#TraceEnable On

#
# Forbid access to version control directories
#
```

Configuración aplicada:

```
text
```

```
TraceEnable Off
```

```
sudo systemctl restart apache2
```

```
# 
# Set to one of: On | off | extended
TraceEnable Off
#TraceEnable On
```

8.2 Ocultamiento de información del servidor

```
bash
```

```
ServerTokens Prod
```

```
ServerSignature Off
```

```
File Edit View Search Terminal Help
debian@debian:/etc/apache2/conf-enabled$ sudo cat security.conf
# Changing the following options will not really affect the security of the
# server, but might make attacks slightly more difficult in some cases.

#
# ServerTokens
# This directive configures what you return as the Server HTTP response
# Header. The default is 'Full' which sends information about the OS-Type
# and compiled in modules.
# Set to one of: Full | OS | Minimal | Minor | Major | Prod
# where Full conveys the most information, and Prod the least.
#ServerTokens Minimal
ServerTokens Prod
#ServerTokens Full

#
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (internal error documents, FTP directory
# listings, mod_status and mod_info output etc., but not CGI generated
```

```
File Edit View Search Terminal Help
debian@debian:/etc/apache2/conf-enabled$ sudo cat mime.conf
# Mime types configuration
# lists mod_status and mod_info output etc., but not CGI generated
# documents or custom error documents).
# Set to "EMail" to also include a mailto: link to the ServerAdmin.
# Set to one of: On | Off | EMail
#ServerSignature Off
ServerSignature Off

#
# Allow TRACE method
#
# Set to "extended" to also reflect the request body (only for testing and
# diagnostic purposes).
#
# Set to one of: On | Off | extended
TraceEnable Off
#TraceEnable On

#
# Forbid access to version control directories
```

8.3 Validación posterior a la corrección

bash

```
curl -I http://10.0.2.11
```

Resultado obtenido:

text

Server: Apache

bash

```
curl -X TRACE http://10.0.2.11
```

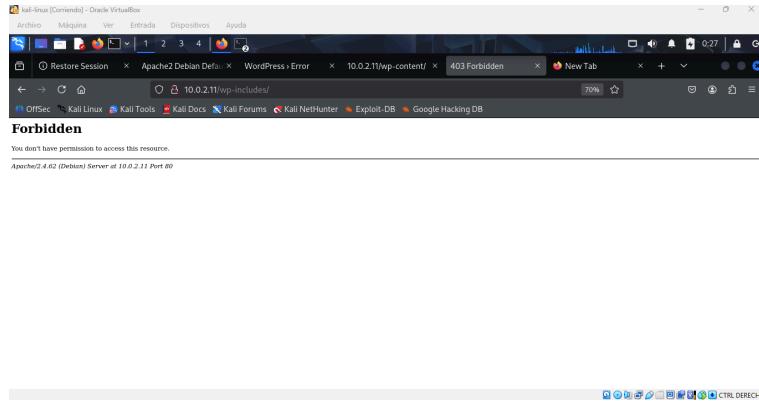
```
(kali㉿kali)-[~]
$ curl -I http://10.0.2.11
HTTP/1.1 200 OK
Date: Fri, 09 Jan 2026 06:19:47 GMT
Server: Apache
Last-Modified: Mon, 30 Sep 2024 14:44:22 GMT
ETag: "29cd-623573d915b52"
Accept-Ranges: bytes
Content-Length: 10701
Vary: Accept-Encoding
Content-Type: text/html

(kali㉿kali)-[~]
$
```

Resultado obtenido:

text

405 Method Not Allowed



9. IMPACTO DE LA REMEDIACIÓN

- Reducción de la superficie de ataque.
- Eliminación de métodos HTTP innecesarios.
- Prevención de ataques de enumeración y fingerprinting.

10. CONCLUSIÓN

La Fase 2 permitió identificar y corregir de forma efectiva una vulnerabilidad distinta al vector de ataque inicial. La explotación controlada demostró el impacto real de una mala configuración aparentemente menor, reforzando la importancia del hardening de servicios expuestos. Tras aplicar las medidas correctivas, el sistema quedó adecuadamente protegido frente a este tipo de ataques.

11. RECOMENDACIONES

- Auditorías periódicas de configuración de servicios web.
- Uso de herramientas automatizadas de hardening.
- Integración de escaneos de seguridad continuos.