

Tabla de contenido

[**Introducción 3**](#_heading=h.30j0zll)

[**Objetivos 3**](#_heading=h.1fob9te)

[**Convenciones utilizadas para valorar y categorizar cada hallazgo 4**](#_heading=h.3znysh7)

[**Metodología de Prueba 5**](#_heading=h.2et92p0)

[**Scope 6**](#_heading=h.tyjcwt)

[**Resumen Ejecutivo 7**](#_heading=h.3dy6vkm)

[**Conclusiones Generales 8**](#_heading=h.1t3h5sf)

[**Detalles Técnicos 8**](#_heading=h.4d34og8)

[**[Alto] [Exposición de datos sensibles en endpoint vulnerable] (CWE-200) 9**](#_heading=h.2s8eyo1)

[**[Alto] [Login bypassing WordPress CMS] (CWE-305) 14**](#_heading=h.2s8eyo1)

[**[Crítica] Remote Code Execution en funcionalidad ‘carga de archivos’ (CWE-434) 20**](#_heading=h.ifaqdzqgcud0)

[**[Alta] Elevación de privilegios (CWE-269) 24**](#_heading=h.3as4poj)

**Reporte:** Tryhackme – Internal.

**Realizado por:** PentestCorp

**Realizado para:** Tryhackme

**Fecha:** 19/03/2024

La información confidencial contenida en este informe está destinada exclusivamente para el uso de los representantes comerciales internos de la organización. Por lo tanto, queda estrictamente prohibida su reproducción sin el previo consentimiento del autor o de la audiencia prevista.

Tanto este informe como todo el proceso de consultoría fueron realizados por PentestCorp.

# **Introducción**

El presente informe de las evaluaciones de seguridad describe los resultados de las pruebas de penetración realizadas a la empresa **TryHackMe** entre los días **12 de Marzo y 19 de Marzo del 2023**. El objetivo de la evaluación fue identificar vulnerabilidades y debilidades en la seguridad de la red y de las aplicaciones web de la empresa, y proporcionar recomendaciones para corregir las vulnerabilidades identificadas. Previo al comienzo del proceso de asesoría, se envió un correo electrónico para dar inicio formal y garantizar que todos los miembros, tanto de PentestCorp como de **TryHackme**, estuvieran informados sobre los riesgos, el nivel de esfuerzo, los puntos de contacto y la duración prevista de la evaluación.

# **Objetivos**

El objetivo principal de este External Pentest fue identificar vulnerabilidades de alto impacto dentro de los sitios externos de**Tryhackme**, **cuya explotación conduce al robo de datos confidenciales de los usuarios y elevación de privilegios dentro de los sistemas. La asesoría siguió una metodología destinada a simular escenarios de ataques reales y amenazas que podrían afectar de manera crítica a la privacidad de los datos, la integridad y la reputación general de la empresa.**

# **Convenciones utilizadas para valorar y categorizar cada hallazgo**

PentestCorp emplea el sistema de calificación CVSS v3, diseñado para ofrecer un método estándar y abierto que permite evaluar el impacto de las vulnerabilidades identificadas en tecnologías de la información. Este sistema ayuda a cuantificar la gravedad de estas vulnerabilidades, utilizando una escala que va del 0 al 10.

Según el puntaje obtenido mediante la fórmula CVSS, se puede determinar el nivel de impacto:

* Para puntajes entre 0.0 y 3.9, la severidad se considera baja.
* Para puntajes entre 4.0 y 6.9, la severidad es media.
* Para puntajes entre 7.0 y 10, la severidad es alta.

Sin embargo, el sistema CVSS no tiene en cuenta ciertas características comerciales. Por ejemplo, en industrias como la bancaria o la aérea, que están sujetas a estrictos requisitos regulatorios, el rango de riesgo puede ser mayor. Por otro lado, las empresas que venden productos no sensibles, como accesorios en mercados fuera de línea, suelen tener requisitos de seguridad más bajos, lo que puede resultar en un rango de riesgo reducido en comparación con otras industrias.

Dadas estas diferencias, el equipo de PentestCorp complementa el puntaje CVSS con una categorización adicional utilizando un código de colores (Crítica, Alta, Media, Baja) para clasificar las vulnerabilidades según su impacto en el negocio en cuestión. Además, esta categorización de riesgos puede ser revisada junto con el cliente y ajustada según sus necesidades específicas.

A close-up of a logo

Description automatically generated

**Gráfico 1 Clasificación de severidad**

# **Metodología de Prueba**

Como expertos en seguridad, en PentestCorp reconocemos la importancia de utilizar frameworks y metodologías probadas en la industria. Sin embargo, entendemos que cada negocio es único y requiere enfoques adaptados a sus necesidades específicas. Es por ello que hemos desarrollado una metodología interna que integra múltiples enfoques reconocidos a nivel mundial, centrándonos especialmente en la evaluación y explotación de aplicaciones web.

Esta metodología nos permite aprovechar los aspectos más relevantes de cada enfoque y combinarlos en un proceso integral para probar y evaluar aplicaciones. Además, nos hemos inspirado en el Proyecto Abierto de Seguridad en Aplicaciones Web (OWASP) y hemos alineado nuestras pruebas con su Guía de Pruebas actual. Esto garantiza que nuestros servicios estén alineados con los estándares de la industria y proporcionen resultados sólidos y confiables para nuestros clientes:

* Reconocimiento e Inteligencia
  + OSINT
  + Recopilación de Información Pasiva
  + Recopilación de Información Activa
* Pruebas de gestión de configuración e implementación
  + Pruebas de Infraestructura
* Prueba de manejo de acceso
  + Pruebas de Autenticación
  + Pruebas de Autorización
  + Pruebas de Manejo de Sesiones
  + Pruebas de Manejo de Identidad
* Pruebas de Validación de Datos
* Lógica del Negocio/Aplicación
* Pruebas Client-Side
* Pruebas de Cifrado Débil
* Prueba de Manejo de Errores
* Pruebas Varias

# **Scope**

Para realizar el pentest de la aplicación web de la empresa TryHackMe, esta define su scope como el siguiente:

* Servidor Web: 10.10.9.42 (Dirección IP dinámica)
* DNS: http://initial.thm

# **Resumen Ejecutivo**

Durante la ejecución del pentest, PentestCorp ha detectado múltiples vulnerabilidades críticas mediante las cuales se puede comprometer la integridad, confidencialidad y disponibilidad de la empresa. A continuación, se encuentra un breve resumen de los hallazgos realizados:

PentestCorp ha encontrado tres vulnerabilidades altas y una crítica, la primera vulnerabilidad hallada fue Exposición de datos sensibles (CWE-200) la cual expone credenciales de usuarios regustrados en la plataforma.

Segundo, una vulnerabilidad alta de tipo Login bypassing WordPress (CWE-305). La cual permite eludir controles de autenticación.

Tercero, una vulnerabilidad alta de tipo elevación de privilegios (CWE-269). La cual permite autenticarnos como root del sistema.

Además se ha detectado una vulnerabilidad de impacto crítico de tipo Remote Code Execution en funcionalidad de ‘carga de archivos’ (CWE-434).

# **Conclusiones Generales**

Durante la realización de las pruebas de penetración en el sistema, se identificaron un total de **cuatro vulnerabilidades,** de las cuales tres fueron clasificadas como altas y una como crítica.Estas vulnerabilidades representan riesgos significativos para la seguridad y la integridad del sistema, y requieren una acción inmediata para su mitigación.

# **Detalles Técnicos**

En el transcurso de la sección de detalles técnicos, se explicará en qué consiste cada una y cómo se desarrolla su explotación y concatenación. Además, se recomienda que la empresa **TryHackMe** tome medidas para corregir todas las vulnerabilidades identificadas en el informe.

# **[Alto] [Exposición de datos sensibles en endpoint vulnerable] (CWE-200)**

**CVSS Vector**

CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:L/A:N

**CVSS Score**

8.2

**Componentes Afectados**

* 10.10.9.42
* http://internal.thm/

**Descripción**

El aplicativo web presenta un CMS de tipo WordPress el cual permite gestionar cuentas de usuarios.

Si la configuración del aplicativo web, junto con la del CMS WordPress no se encuentra correctamente configurada, es posible que un atacante pueda acceder a información confidencial.

**Remediación**

Para prevenir este tipo de vulnerabilidades se deben tomar las precauciones necesarias, entre estas se encuentran:

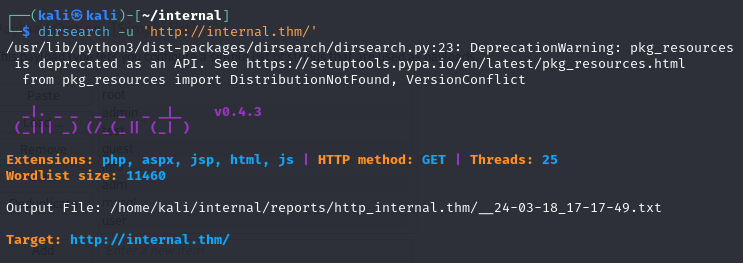
* **Autenticación**: Implemente un sistema de autenticación para restringir el acceso a datos sensibles en el determinado endpoint para los usuarios finales.
* **Autorización**: Asegúrese que el endpoint solamente pueda ser accesible a los usuarios permitidos por el aplicativo, por ejemplo, administradores que se encuentren autenticados previamente.
* **Limitar exposición de datos**: En lo posible, solamente exponga información que se considere necesaria.
* **Encriptación**: Asegúrese que los datos sensibles expuestos se encuentren encriptados. En caso que un atacante malicioso sea capaz de llegar hasta ellos, estos se encuentren protegidos por encriptación.
* **Conciencia en los desarrolladores**: Asegúrese que el equipo de desarrolladores y administradores involucrados en el aplicativo, sean conscientes de qué tipo de información se considere sensible. A veces la información aparentemente inofensiva puede ser mucho más útil para un atacante de lo que uno cree.

**Explotación**

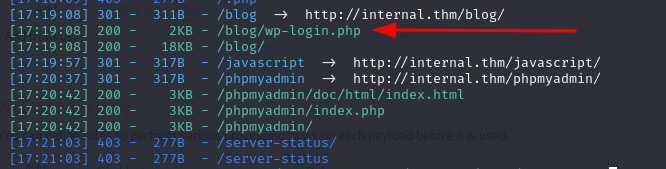
El equipo de OffSec identificó la vulnerabilidad de **exposición de datos sensibles** la cual nos expuso una cuenta de usuario **admin**.

A continuación, se encuentra de manera detallada, cómo fue posible explotar dicha vulnerabilidad:

**Paso 1**: Se realizó un escaneo de directorios con la herramienta **dirsearch** la cual expuso, entre otros, el panel de login de **WordPress**:

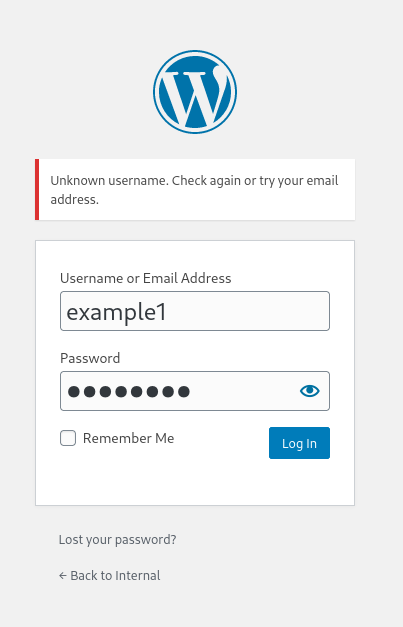


**Figura 1 Escaneo de directorios ocultos con dirsearch**



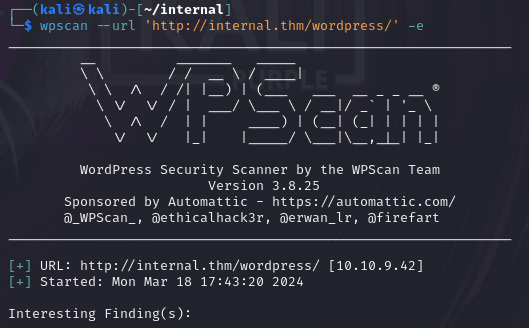
**Figura 2 Resultado del escaneo con el directorio encontrado**

Luego se entró en el endpoint <http://internal.th/blog/wp-login.php> el cual nos muestra lo siguiente:



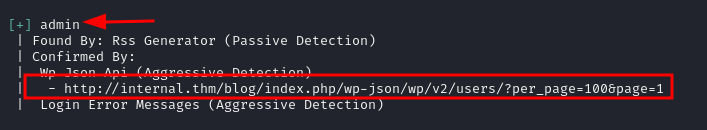
**Figura 3 Imagen del panel de login de WordPress**

**Paso 2**: Al ser un CMS WordPress con un panel de autenticación de usuarios, es posible analizarlo con la herramienta WPscan:



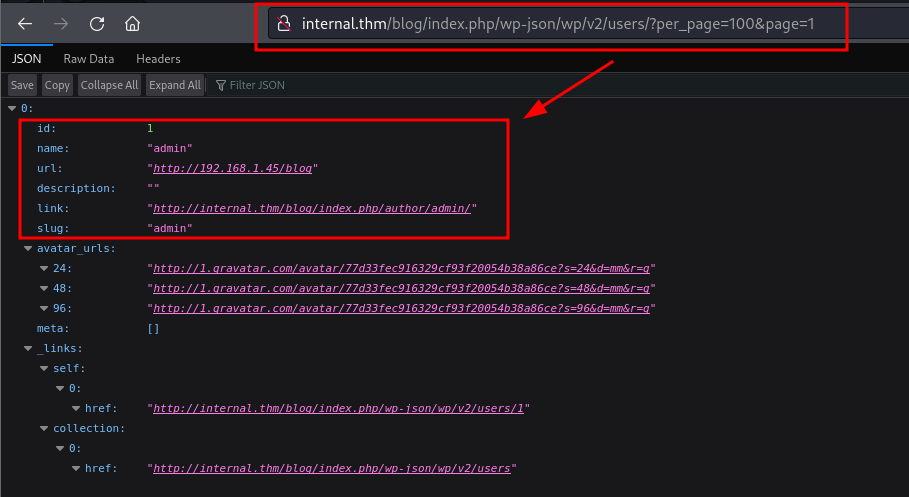
**Figura 4 Se muestra la ejecución de la herramienta wpscan**

Entonces la herramienta trabaja enumerando usuarios en la plataforma, en este caso un usuario **admin** junto con un path que expone el endpoint de los datos:



**Figura 5 Se confirma el usuario admin junto con un endpoint URL**

**Paso 3**: Accedemos hacia el endpoint del aplicativo y podemos observar datos sensibles expuestos ante un usuario de bajos privilegios. Entre ellos el username **admin** junto con su **ID 1**:



**Figura 5 Se confirma el usuario admin en el endpoint señalado**

**Impacto**

Mediante esta vulnerabilidad, un atacante podría usar esta credencial para combinarlas con ataques de fuerza bruta a contraseñas y con ello, conseguir acceso no deseado al sistema.

**Referencias**

* CWE: https://cwe.mitre.org/data/definitions/200.html

# **[Alto] [Login bypassing WordPress CMS] (CWE-305)**

**CVSS Vector**

CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:L/A:L

**CVSS Score**

8.8

**Componentes Afectados**

* 10.10.9.42
* http://internal.thm/

**Descripción**

El panel de login CMS de Wordpress permite a los usuarios registrados acceder a las funcionalidades administrativas del aplicativo web en cuestión.

Si el panel de login de WordPress es eludido, esto puede llevar a varios riesgos y consecuencias graves como el acceso no autorizado, robo de datos, inyección de malware para generar un Shell inverso y en consecuencia un RCE contra el servidor y un gran daño a la reputación de la empresa entre otros factores.

**Remediación**

Para prevenir este tipo de vulnerabilidades se deben tomar las precauciones necesarias, entre estas se encuentran:

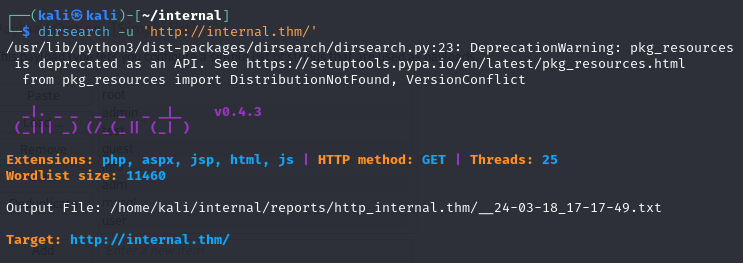
* **Utilizar contraseñas fuertes:** De preferencia contraseñas que ronden entre los 8 a 12 carácteres de largo donde incluyan combinaciones entre mayúsculas, minúsculas, números y carácteres especiales.
* **Autenticación por dos factores (2FA):** Una autenticación 2FA agrega una capa adicional de seguridad y requiere que el usuario demuestre su identidad con dos formas diferentes antes de acceder al panel administrativo. Este puede ser un correo con un token o bien por teléfono.
* **Implementación de un Rate-Limit (WAF):** Al implementar un limitador de solicitudes por parte de una IP determinada, esto puede ayudar a prevenir ataques de fuerza bruta en especial cuando la contraseña no se encuentra reforzada (véase el primer punto).
* **Monitorear sitio:** En lo posible, utilizar herramientas de monitoreo que revise la actividad en el sitio web. Esto puede ayudar a detectar posibles intrusos. Una alternativa podría ser **“uptime robot”**
* **Restringir el acceso al panel de administración**: En lo posible, restringir el acceso al panel de administración a ciertas direcciones IP.

**Explotación**

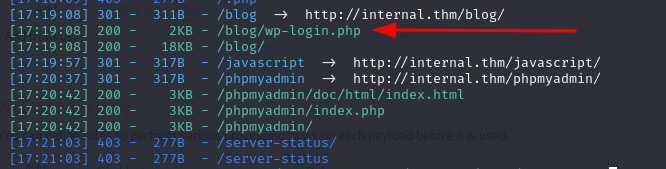
El equipo de OffSec identificó la vulnerabilidad de **Login bypassing WordPress CMS** la cual permitió acceder al panel del usuario **admin**.

A continuación, se encuentra de manera detallada, cómo fue posible explotar dicha vulnerabilidad:

**Paso 1**: Se realizó un escaneo de directorios con la herramienta **dirsearch** la cual expuso, entre otros, el panel de login de **WordPress**:

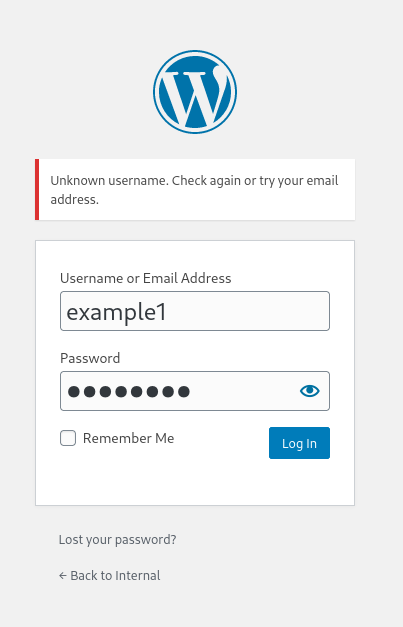


**Figura 6 Escaneo de directorios ocultos con dirsearch**



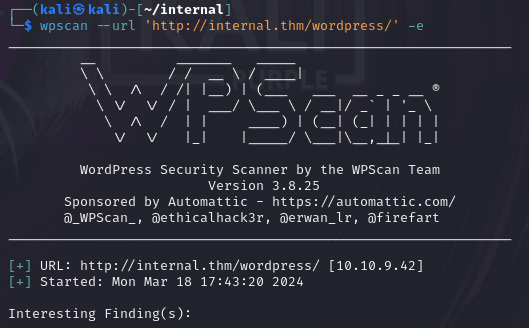
**Figura 7 Resultado del escaneo con el directorio encontrado**

Luego se entró en el endpoint <http://internal.th/blog/wp-login.php> el cual nos muestra lo siguiente:



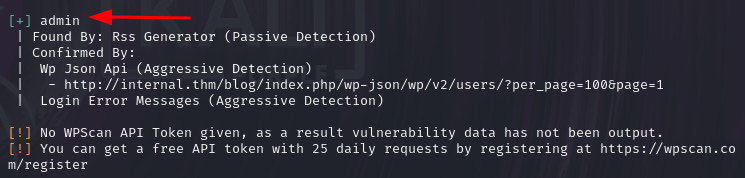
**Figura 8 Imagen del panel de login de WordPress**

**Paso 2**: Al ser un CMS WordPress con un panel de autenticación de usuarios, es posible apoyarnos con herramientas para su bypassing como Burpsuite intruder o WPscan. Para reproducir la vulnerabilidad se mostrará su uso en WPscan:



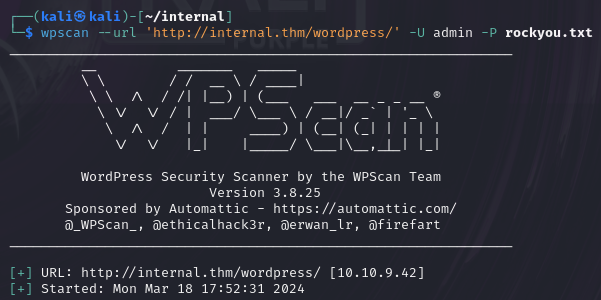
**Figura 9 Se ejecuta la herramienta**

Entonces la herramienta trabaja enumerando usuarios en la plataforma, en este caso un usuario **admin** y los muestra en output:

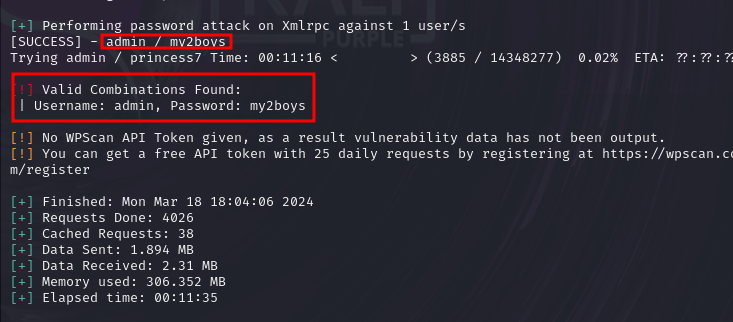


**Figura 10 Se expone un nombre de usuario admin**

**Paso 3**: Posteriormente aprovechamos la misma funcionalidad de la herramienta para aprovechar un ataque de fuerza bruta contra la autenticación dado el username **admin**:

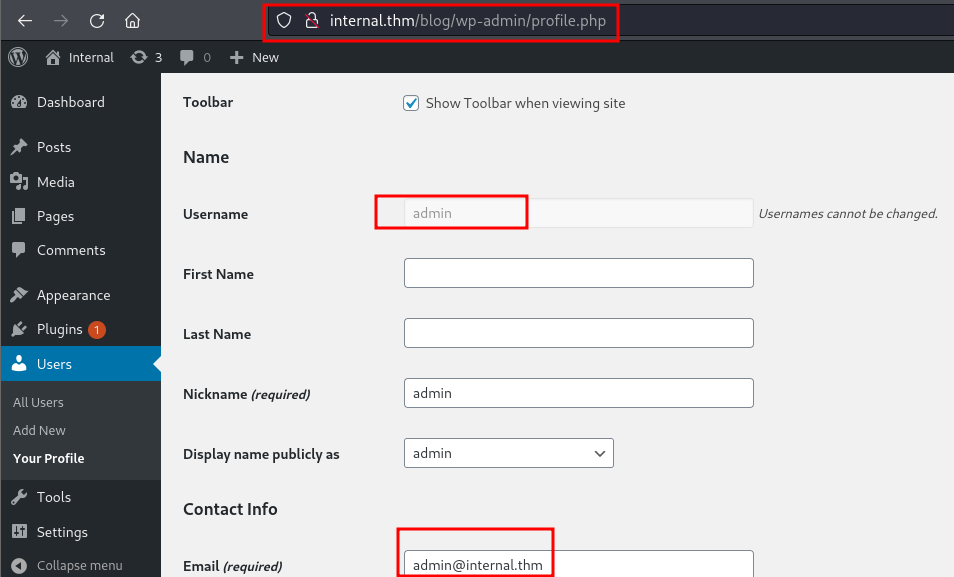


**Figura 11 Se representa un ataque de diccionario (o fuerza bruta) dado el nombre de usuario admin**

**

**Figura 12 Se muestra como las credenciales admin y my2boys hacen coincidencia**

**Paso 4**: Una vez descubierto las credenciales accedemos al panel autenticados como administrador:



**Figura 13 Se muestra el panel de administrador de WordPress**

**Impacto**

Mediante esta vulnerabilidad, un atacante puede autenticarse como usuario administrador del sistema.

**Referencias**

* CWE: [CWE - CWE-305: Authentication Bypass by Primary Weakness (4.14) (mitre.org)](https://cwe.mitre.org/data/definitions/305.html)

# **[Crítica] Remote Code Execution en funcionalidad ‘carga de archivos’ (CWE-434)**

**CVSS Vector**

CVSS:3.0/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:L/A:L

**CVSS Score**

9.1

**Componentes Afectados**

* 10.10.9.42
* http://internal.thm/

**Descripción**

Existe una funcionalidad de carga de archivos en el endpoint (<http://internal.thm/blog/wp-admin/theme-editor.php?file=404.php&theme=twentyseventeen>) la cual permite cargar diversos archivos hacia el servidor, en especial código escrito en php.

Cuando el servidor recibe estos archivos sin una adecuada sanitización de datos, un atacante podría cargar archivos maliciosos, por ejemplo, entablar una **shell inversa** para comprometer el sistema.

**Remediación**

Para prevenir este tipo de vulnerabilidades se deben tomar las precauciones necesarias, entre estas se encuentran:

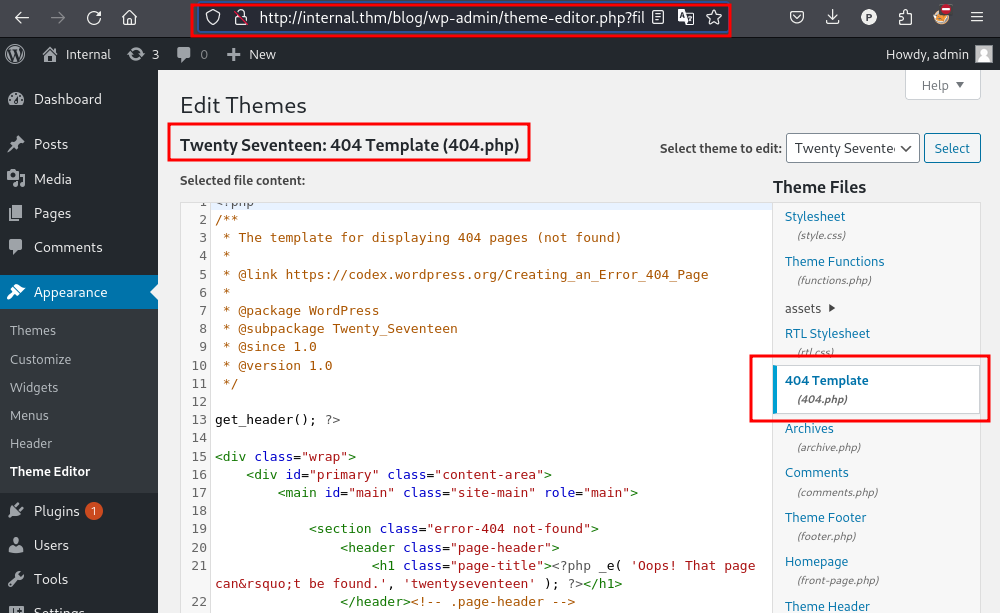
* **Validación de entrada**: En lo posible, asegúrese de validar todos los archivos cargados para verificar que son del tipo esperado. Si el servidor solamente se pensó para cargar archivos de imágen .jpg, .png, jpeg, etc… Asegúrese que el servidor valide la carga de archivos con dichas extensiones.
* **Escaneo de malware**: Considere la posibilidad de escanear todos los archivos cargados con una herramienta anti-malware (utilizando una combinación de firmas, heurística y métodos de detección de aprendizaje automático) para detectar cualquier contenido malicioso.
* **Limitar el tamaño del archivo**: Establezca un límite máximo y mínimo en el tamaño de los archivos permitidos para subir al servidor.
* **Almacenamiento seguro de archivos**: Almacene los archivos cargados en un lugar seguro y asegúrese de que no se puedan ejecutar en el servidor. Por ejemplo, podría considerar almacenar archivos en un bucket de Amazon S3 en lugar del propio servidor.

**Explotación**

El equipo de OffSec identificó la vulnerabilidad **Remote Code Execution (RCE)** mediante la carga de archivos.

A continuación, se encuentra de manera detallada, cómo fue posible explotar dicha vulnerabilidad:

**Paso 1**: Se ingresa en el endpoint <http://internal.thm/blog/wp-admin/theme-editor.php?file=404.php&theme=twentyseventeen> en donde se muestra un editor de estilos:



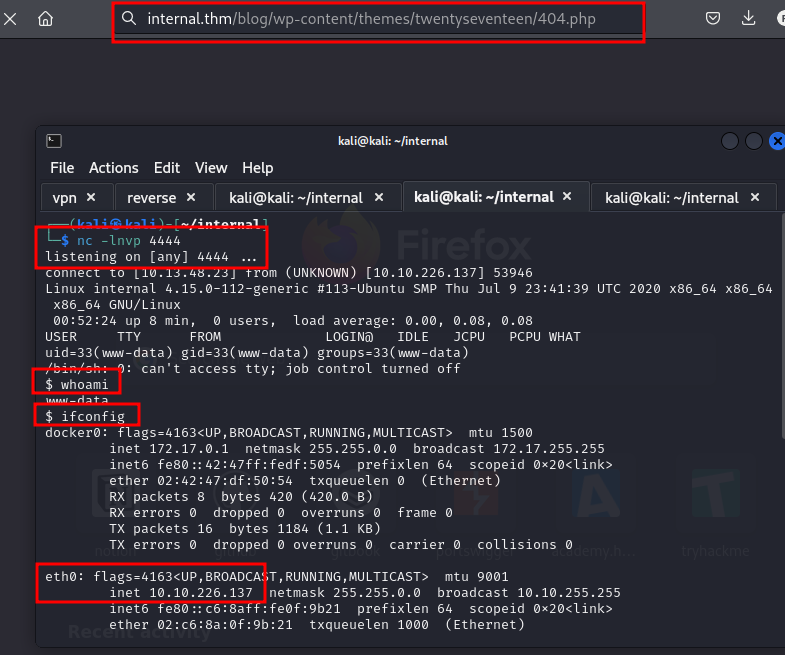
**Figura 14 Se muestra el endpoint vulnerable a carga de archivos**

**Paso 2**: Se genera un payload malicioso escrito en php el cual contiene nuestra dirección IP junto con un puerto a nuestra elección. El link del repo corresponde a (<https://github.com/pentestmonkey/php-reverse-shell>):



**Figura 15 Se muestra el script malicioso php dado una IP y un puerto**

**Paso 3**: Se envía nuestro payload .php malicioso hacia el template del servidor web. Una vez cargado el archivo, nos dirigimos hacia el endpoint (<http://internal.thm/blog/wp-content/themes/twentyseventeen/404.php>) para “activar” nuestro payload:



**Figura 16 Muestra el payload activado y la reverse shell establecida en el puerto 4444**

**Impacto**

Mediante esta vulnerabilidad, un atacante puede tomar control del servidor.

**Referencias**

* CWE: <https://cwe.mitre.org/data/definitions/434.html>

# **[Alta] Elevación de privilegios (CWE-269)**

**CVSS Vector**

CVSS:3.0/AV:N/AC:H/PR:L/UI:N/S:C/C:H/I:H/A:H

**CVSS Score**

8.5

**Componentes Afectados**

* 10.10.163.211
* http://internal.thm/

**Descripción**

La elevación de privilegios consta de una serie de pasos previos para alcanzarlo. Para ello, es necesario abusar de un **bypassing de login de usuario** junto con un **bypassing de login de usuario**.

**Primero**. Un usuario malintencionado podría llegar hasta el panel de login. Al no existir un control de acceso adecuado, este podría ser eludido.

**Segundo**. Al no existir un correcto saneamiento de entrada de datos, cualquier usuario con acceso al sistema podrá establecer una **shell inversa** lo cual le otorgará acceso remoto desde su computadora hasta el backend.

**Remediación**

Para prevenir este tipo de vulnerabilidades se deben tomar las precauciones necesarias, entre estas se encuentran:

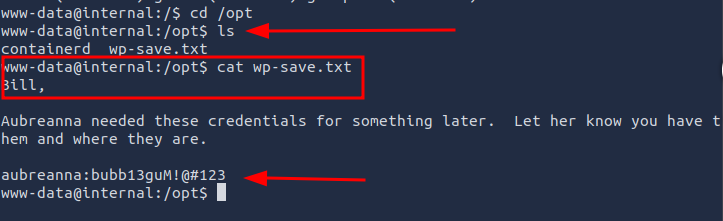
* **Validación de entrada**: En lo posible, asegúrese de validar todos los archivos cargados para verificar que son del tipo esperado. Implemente un sistema que bloquee cualquier tipo de entrada no esperada comúnmente por el aplicativo.
* **Escaneo de malware**: Considere la posibilidad de escanear todos los archivos cargados con una herramienta anti-malware (utilizando una combinación de firmas, heurística y métodos de detección de aprendizaje automático) para detectar cualquier contenido malicioso.
* **Utilizar contraseñas fuertes:** De preferencia contraseñas que ronden entre los 8 a 12 carácteres de largo donde incluyan combinaciones entre mayúsculas, minúsculas, números y carácteres especiales.
* **Autenticación por dos factores (2FA):** Una autenticación 2FA agrega una capa adicional de seguridad y requiere que el usuario demuestre su identidad con dos formas diferentes antes de acceder al panel administrativo. Este puede ser un correo con un token o bien por teléfono.
* **Implementación de un Rate-Limit (WAF):** Al implementar un limitador de solicitudes por parte de una IP determinada, esto puede ayudar a prevenir ataques de fuerza bruta en especial cuando la contraseña no se encuentra reforzada (véase el punto anterior de **contraseñas fuertes**).

**Explotación**

El equipo de OffSec identificó una serie de pasos para explotar una vulnerabilidad de **elevación de privilegios,** la cual permitió identificar posibles credenciales. En el transcurso, se muestran vulnerabilidades de tipo **login bypass y carga insegura de archivos.**

A continuación, se encuentra de manera detallada, cómo fue posible explotar dichas vulnerabilidades:

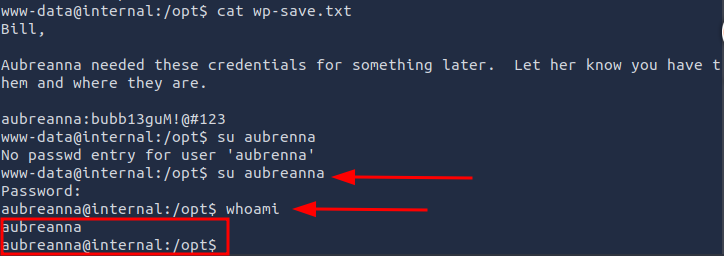
**Paso 1**: Una vez ingresados en el sistema navegamos al directorio **/opt** y observamos un archivo en texto plano el cual contiene credenciales de acceso hacia otro usuario:



**Figura 17 Muestra el contenido del archivo wp-save.txt**

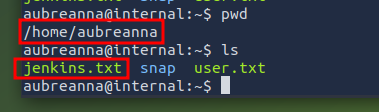
El archivo muestra credenciales de acceso para el usuario aubreanna.

**Paso 3**: Cambiamos de usuario con el comando **su aburenna** y la password **bubb13muM!@#123**. Comprobamos el usuario actual con un **whoami**:



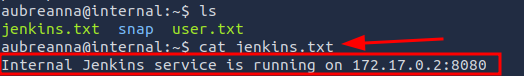
**Figura 18 Muestra la ejecución del comando whoami el cual muestra el usuario actual aubreanna**

**Paso 4**: Cambiamos de directorio hacía **/home/aubreanna** el cual contiene un archivo **jenkins.txt**:



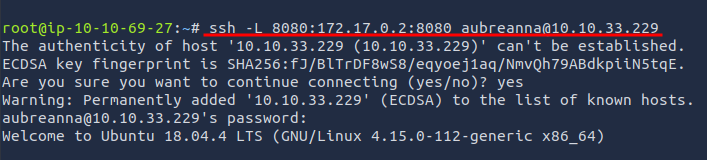
**Figura 19 Muestra la ruta hacia el archivo jenkins.txt**

Al abrir el archivo podemos leer que existe un servicio interno de **Jenkins** ejecutándose en la dirección **172.17.0.2:8080**



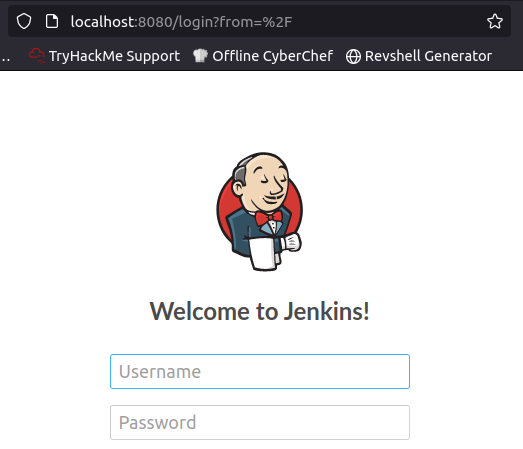
**Figura 20 Muestra el contenido del archivo jenkins.txt**

**Paso 5**: Ejecutamos un comando el cual creará un túnel ssh el cual vincula el servicio interno de jenkins bajo el puerto 8080 junto con el usuario aubrianna:

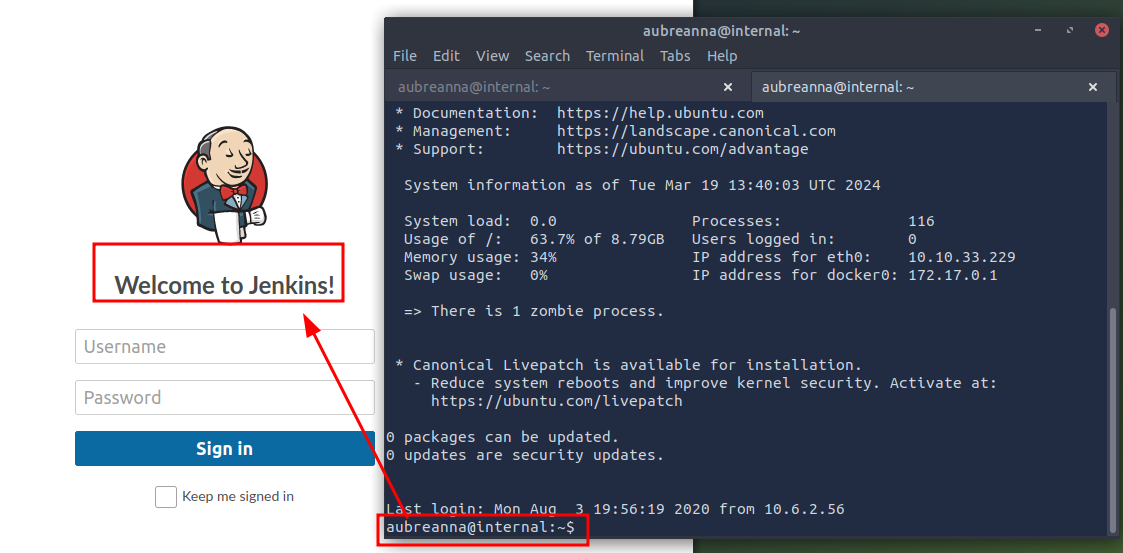


**Figura 21 Muestra el comando ejecutándose con éxito**

A continuación podemos observar que el servicio interno de jenkins se encuentra ejecutándose correctamente desde el navegador bajo la dirección **localhost:8080**:

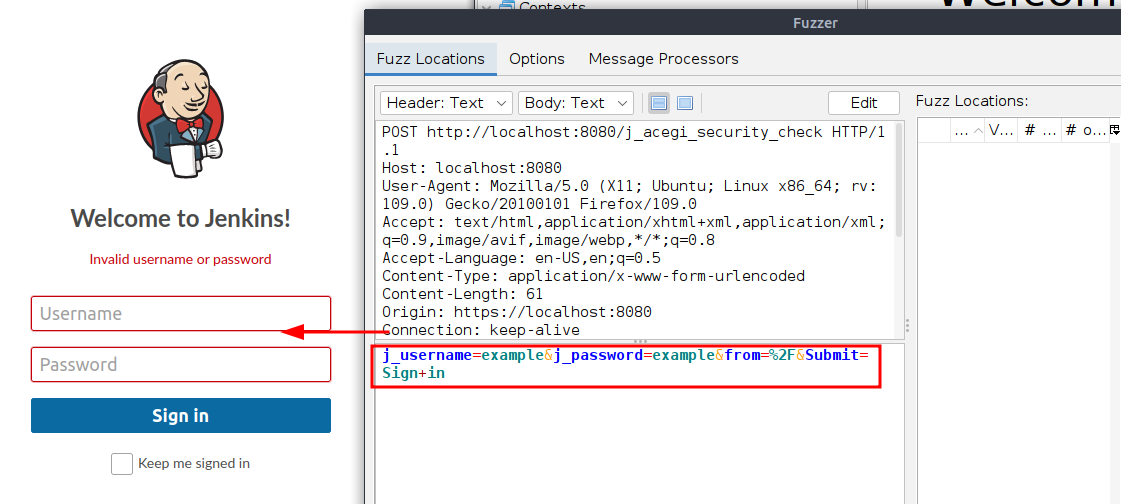


**Figura 22 Muestra el servicio de Jenkins**



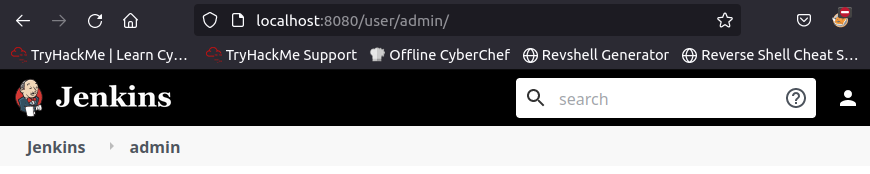
**Figura 23 Muestra el servicio de Jenkins gracias a la conexión establecida previamente**

**Paso 6**: Con la ayuda de la herramienta **OWASP Zap** utilizamos las funcionalidades para realizar ataques de fuerza bruta contra el panel de autenticación:

****

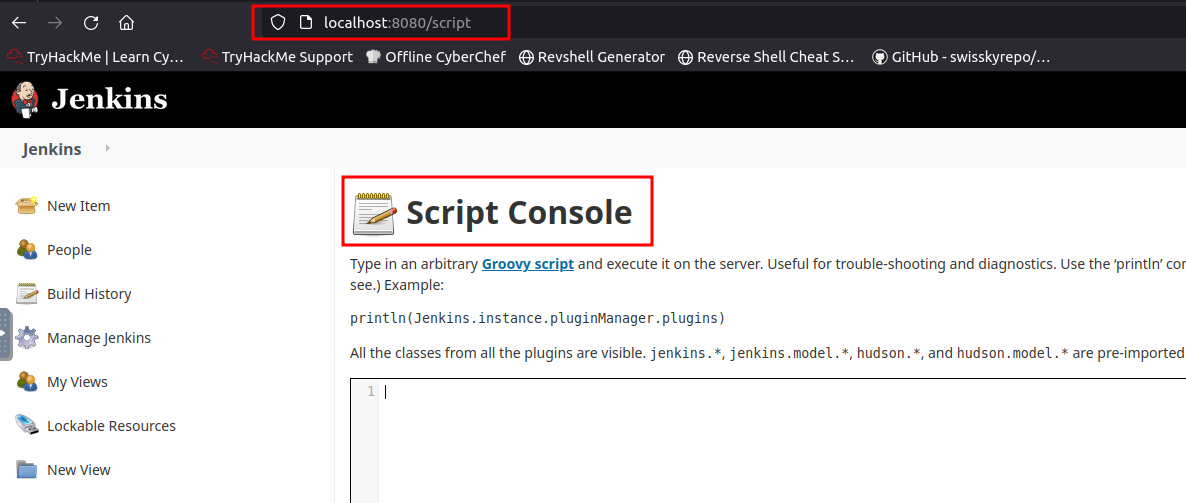
**Figura 24 Muestra previa de cómo la herramienta ejecuta el ataque de fuerza bruta**

Luego de realizar un ataque de fuerza bruta y encontrar credenciales de acceso: **admin** ; **spongebob** logramos acceder al panel de administrador de Jenkins:



**Figura 25 Muestra el panel de Jenkins**

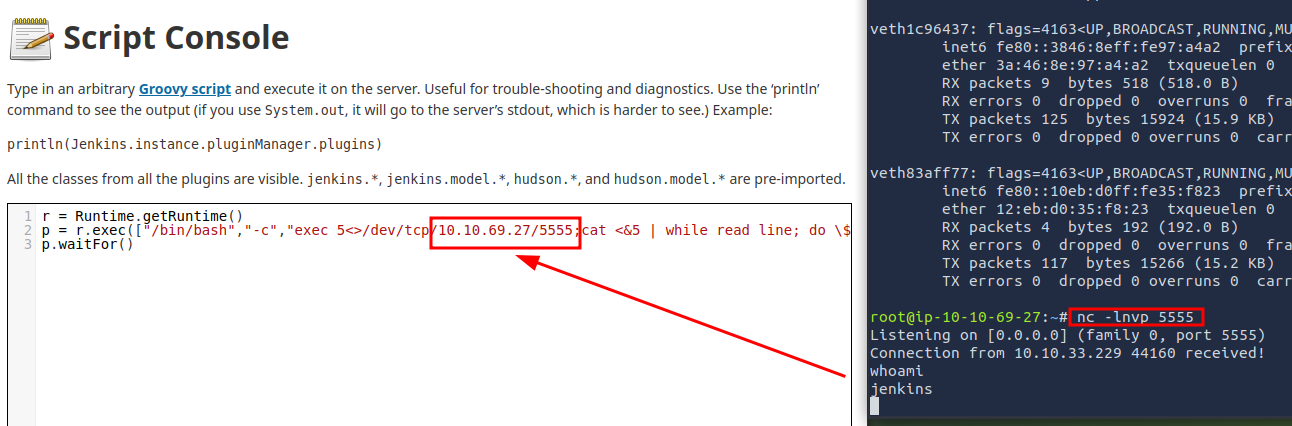
**Paso 7**: Nos dirigimos hacia el endpoint **/script** en donde podremos encontrar una consola para scripts:



**Figura 26 Muestra el endpoint vulnerable**

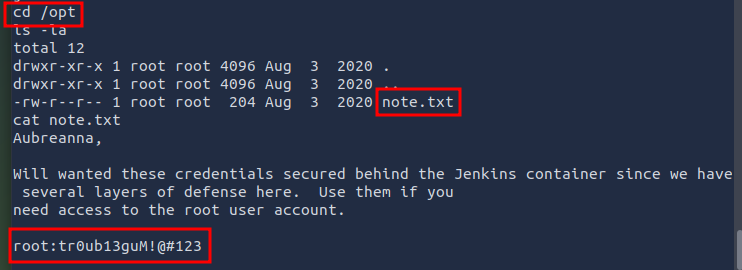
**Paso 8**: Ingresamos un payload que genere una **reverse shell** dada nuestra IP del atacante junto con un puerto a la escucha. Además de activar **netcat** que se encuentre a modo de escucha.

En la siguiente imagen podemos observar que ahora pertenecemos al usuario **Jenkins.**



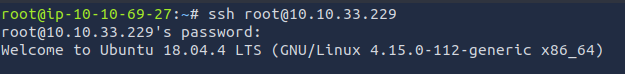
**Figura 27 Muestra el script ejecutándose**

**Paso 9**: Cambiar al directorio **/opt** y abrir el archivo **note.txt** el cual contiene credenciales sobre el usuario **root**:

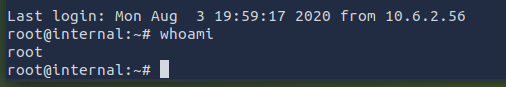
****

**Figura 28 Muestra las credenciales de acceso al usuario root**

**Paso 10**:Ingresar por protocolo ssh hacia el usuario **root** junto con su dirección:

****

**Figura 29 Muestra la conexión SSH al usuario root**

****

**Figura 30 Muestra la autenticación completada**

**Impacto**

Mediante esta vulnerabilidad, un atacante puede enumerar credenciales para posible uso posterior.

**Referencias**

* CWE: [CWE - CWE-269: Improper Privilege Management (4.14) (mitre.org)](https://cwe.mitre.org/data/definitions/269.html)

**Herramientas Utilizadas**

* Netcat: <https://www.kali.org/tools/netcat/>
* Seclists: <https://github.com/danielmiessler/SecLists>
* Owasp Zap: <https://www.zaproxy.org/download/>
* WPscan: <https://wpscan.com>
* Dirsearch: <https://github.com/maurosoria/dirsearch>