Trabajo Práctico Integrador

Tema: Seguridad en los sistemas operativos - Reconocimiento de sistemas

Alumnos

Berrone Lanza Lina Lucia

Bayurk Mara Valentina

Tecnicatura Universitaria en Programación - Universidad Tecnológica Nacional.

Arquitectura y sistemas operativos

Docente Titular

Mauricio Gabriel Pasti

Docente Tutor

David Roco

31 de Mayo de 2025

Índice

1. Introducción

En el contexto actual, caracterizado por una creciente dependencia de los sistemas informáticos, la seguridad de los entornos digitales constituye un eje central en el diseño, la administración y la protección de las infraestructuras tecnológicas. En este escenario, resulta imprescindible el conocimiento y la aplicación de herramientas que permitan identificar y mitigar posibles vulnerabilidades que puedan comprometer la integridad, confidencialidad y disponibilidad de la información.

El presente trabajo integrador tiene por objeto abordar tres aspectos fundamentales vinculados a la seguridad en sistemas operativos: el reconocimiento de sistemas, la detección de vulnerabilidades en puertos y servicios, y la implementación de la herramienta Nmap a través de scripts en Bash, en el entorno de consola de GNU/Linux.

El reconocimiento de sistemas se refiere al proceso de obtención de información detallada sobre las características del sistema objetivo, como parte inicial de un análisis de seguridad. Por su parte, la exploración de puertos y servicios permite identificar recursos activos que podrían presentar debilidades explotables. Finalmente, la utilización de Nmap, combinada con la automatización mediante Bash, facilita la ejecución sistemática de tareas de escaneo, permitiendo un abordaje más eficiente y replicable en distintos entornos.

A lo largo del desarrollo del trabajo, se presentará un análisis teórico acompañado de ejemplos prácticos que ilustran la aplicación concreta de las herramientas y técnicas mencionadas, con el propósito de contribuir a la comprensión y fortalecimiento de la seguridad en sistemas operativos.

2. Marco Teórico

1.1 Reconocimiento de sistemas

El reconocimiento de sistemas, también denominado *footprinting o information gathering*, constituye la primera etapa en cualquier proceso de auditoría de seguridad o análisis de vulnerabilidades. Consiste en recolectar información detallada sobre un sistema, red o infraestructura tecnológica con el objetivo de identificar su arquitectura, sistema operativo, servicios expuestos y posibles puntos de entrada.

Este proceso puede clasificarse en dos grandes tipos:

- Reconocimiento pasivo: implica la obtención de información sin interactuar directamente con el sistema objetivo. Se basa en fuentes abiertas, como registros DNS, bases de datos públicas, redes sociales, o motores de búsqueda.

- Reconocimiento activo: involucra la interacción directa con el sistema mediante herramientas de escaneo. Esta modalidad puede generar alertas en los sistemas de detección de intrusos (IDS), pero ofrece resultados más precisos.

El reconocimiento activo permite, entre otras cosas, determinar el sistema operativo en uso (mediante técnicas como *OS fingerprinting*), identificar la versión de servicios activos, y mapear la estructura de la red.

> Según Skoudis y Liston (2006), *"...el reconocimiento adecuado permite diseñar estrategias más eficaces de prueba o defensa, dado que "conocer al objetivo es la base de toda acción ofensiva o defensiva en ciberseguridad."*

En este trabajo desarrollaremos en profundidad un caso de reconocimiento activo mediante un script ejecutable.

1.2 Puertos

1.2.a Definición y clasificación

Un puerto es una interfaz lógica que permite a las aplicaciones establecer comunicaciones con otros dispositivos. Trabajan en la capa de transporte (4) del modelo OSI y en la capa 3 del modelo TCP/IP

Los protocolos principales que utilizan los puertos en esta capa son:

- TCP (Protocolo de Control de Transmisión): Proporciona una conexión orientada, fiable y con control de flujo. Y una entrega ordenada y confiable de datos.
- UDP (Protocolo de Datagrama de Usuario): Ofrece una conexión no orientada, sin garantías de entrega ni orden, pero con menor sobrecarga y mayor rapidez, útil para aplicaciones en tiempo real como streaming o VoIP.

Ambos protocolos utilizan números de puerto para identificar las aplicaciones o servicios específicos en los dispositivos de origen y destino.

Como establecimos, los puertos permiten que múltiples aplicaciones o servicios en un mismo dispositivo puedan comunicarse simultáneamente a través de la red y junto con la dirección IP, ayuda a dirigir los datos al proceso o aplicación correspondiente en el dispositivo receptor. La combinación de una dirección IP y un número de puerto se denomina **socket**. Este identificador único permite que múltiples conexiones simultáneas se gestionen correctamente, asegurando que los datos lleguen a la aplicación o servicio adecuado en el dispositivo receptor

En los sistemas operativos, los puertos se identifican mediante un número de 16 bits, lo que permite un rango de valores entre 0 y 65535. Según IANA (Internet Assigned Numbers Authority), que es la entidad responsable de la asignación y gestión de los números de puerto utilizados en los protocolos de transporte (TCP, UDP, DCCP y SCTP); estos se dividen en tres categorías:

Rango	Tipo de puerto	Uso común
0 – 1023	Puertos de sistema	Servicios y protocolos estándar, ampliamente utilizados (HTTP, FTP)
1024 – 49151	Puertos registrados/de usuario	Asignados a aplicaciones y servicios específicos que han sido registrados
49152 – 65535	Puertos dinámicos/privados	Conexiones efímeras, no están asignados oficialmente

1.2.b Vulnerabilidades en los puertos

Las debilidades en los puntos de acceso de una red pueden ser explotadas por actores malintencionados para comprometer la seguridad de sistemas y datos. Estas vulnerabilidades pueden surgir debido a puertos abiertos innecesarios, servicios desactualizados o configuraciones incorrectas, y son comúnmente aprovechadas mediante técnicas como el escaneo de puertos para identificar y explotar estos puntos débiles.

Según IANA *"...La asignación de un número de puerto por parte de la IANA no implica en ningún caso una aprobación o respaldo de una aplicación o producto específico.*
El hecho de que el tráfico de red fluya hacia o desde un puerto registrado no garantiza que dicho tráfico sea "bueno" ni que necesariamente corresponda al servicio asignado a ese puerto.
Los administradores de sistemas y cortafuegos deben configurar los sistemas basándose en su conocimiento del tráfico en cuestión, no únicamente en si existe o no un número de puerto registrado."

Por lo tanto las identificaciones numéricas no aseguran que el tráfico sea efectivamente seguro, se debe reforzar los sistemas con firewalls, protocolos seguros y otras herramientas para evitar vulnerabilidades y prevenirlas.

Cada puerto puede estar asociado a un servicio. La exposición innecesaria o insegura puede representar una vulnerabilidad. Por ejemplo:

- **Puerto 21 (FTP):** transmisión insegura de credenciales.
- **Puerto 23 (Telnet):** comunicación en texto plano.
- **Puerto 445 (SMB):** explotado en ataques como WannaCry.

Las vulnerabilidades pueden clasificarse en:

- **De configuración:** servicios expuestos sin necesidad.

- **De versión:** uso de versiones obsoletas o sin parches.
- **Lógicas:** errores en el diseño o ejecución del software.

El análisis de puertos y servicios se convierte así en una tarea fundamental para detectar puntos débiles explotables por actores maliciosos.

1.2.c Investigacion y resolucion de puertos vulnerables

La investigación y resolución de puertos vulnerables es un proceso trascendental en la gestión de la seguridad informática. Implica la detección, análisis y mitigación de debilidades en los puntos de acceso (puertos) que puedan ser explotadas por terceros y puedan comprometer la integridad y confidencialidad de los datos enviados

El análisis comienza con la recopilación de información sobre los puertos abiertos con su debida identificación y el tipo de servicio que se ejecuta en ellos. Se utilizan herramientas como Nmap y Nessus para realizar escaneos que detectan configuraciones incorrectas, versiones de software desactualizadas y otros posibles puntos débiles. Estos escaneos pueden ser internos o externos, autenticados o no autenticados, dependiendo del nivel de acceso y la perspectiva desde la cual se evalúa la red, sirviendo para propósitos diferentes.

Luego de su identificación y análisis se procede a realizar una clasificación de las amenazas según su gravedad, ya que de esta depende el riesgo que representan. Esta clasificación nos va a saber qué medidas pueden adoptarse para la resolución de estas amenazas. Las medidas pueden ser actualizaciones, reconfiguraciones de servicios (como bloqueo de los puertos o servicios que hayan sido encontrados peligrosos), implementación de medidas adicionales (como la autenticación de doble factor), etc.

Este proceso debe ser realizado y monitoreado constantemente para poder configurar y adaptar los sistemas antes de que se produzca cualquier brecha crítica en los mismos.

1.2 NMAP

1.2.a Descripción, usos y ejemplos

Nmap (*Network Mapper*) es una herramienta de código abierto diseñada para realizar escaneos de red, detección de sistemas operativos y servicios, así como identificar vulnerabilidades conocidas. Su sintaxis flexible permite realizar desde escaneos básicos hasta análisis complejos en entornos automatizados.

Entre sus funciones principales se encuentran:

- Escaneo de puertos TCP y UDP.
- Detección de sistema operativo (*OS Detection*).

- Detección de versión de servicios.
- Generación de scripts automáticos con NSE (Nmap Scripting Engine).

Ejemplo de comando básico:

```
nmap -sS -O 192.168.0.1
```

Este comando realiza un escaneo TCP tipo *SYN Scan* y trata de detectar el sistema operativo del host.

La integración de Nmap en scripts escritos en Bash, dentro del entorno GCS (GNU/Linux Command Shell), permite la automatización de tareas rutinarias de análisis. Por ejemplo, se pueden programar escaneos periódicos a múltiples direcciones IP, generar reportes personalizados o ejecutar condiciones condicionales según los resultados.

Ejemplos de Uso de Nmap:

- Descubrimiento de Dispositivos en una Red:

nmap -sn 192.168.1.0/24

Este comando realiza un escaneo de hosts vivos en la red especificada (en este caso, 192.168.1.0/24), sin realizar una exploración de puertos, útil para descubrir qué dispositivos están activos.

- Escaneo de Puertos en un Host:

nmap 192.168.1.1

Analiza los puertos abiertos en el host con la dirección IP 192.168.1.1, proporcionando información detallada sobre los servicios en ejecución.

- Escaneo Completo de Puertos en un Rango de IP:

nmap -p 1-65535 192.168.1.0

Explora todos los puertos posibles en el host con la dirección IP 192.168.1.0, permitiendo una evaluación exhaustiva de la superficie de ataque

3. Caso práctico

En este trabajo se desarrolló un script en Bash que automatiza el escaneo y la clasificación de puertos según su numeración, utilizando la herramienta NMAP y siendo testeado en Google Cloud Shell (GCS). Esto permite identificar de manera eficiente posibles debilidades en los servicios detallados en un informe al finalizar la ejecución. Además por consola se mostrará de salida cuales son los posibles puertos que presenten una amenaza y se otorgarán una lista de medidas para cerrarlos y cortar la comunicación si se desea. Una medida implementada de seguridad fue la de configurar otro script, que es agregado al proyecto como una posible solución rápida de cierre de puertos.

4. Metodología Utilizada

4.1 SCRIPT PRINCIPAL

- Iniciamos con la creación de un script llamado “escaneo-v1” con el comando: nano escaneo-v1.sh
- Luego desarrollamos el script de escaneo, con las comprobaciones necesarias y la clasificación de amenazas.
- Luego le dimos los permisos necesarios con el comando: chmod +x escaneo-v1.sh
- A continuación lo ejecutamos con el comando: sudo ./escaneo-v1.sh localhost

  Si quisiéramos escanear una IP en vez de localhost deberíamos poner la dirección IP. Si no le agregamos mas nada hara un escaneo general, si quisieramos un escaneo completo debemos colocar la palabra - -completo luego de localhost o de la dirección IP. El comando en este ultimo caso seria: sudo ./escaneo-v1.sh localhost - -completo

```
#!/bin/bash
echo "Iniciando escaneo de puertos con Nmap"

# Validar parámetros
if [ -z "$1" ]; then
echo "Uso: $0 <IP o localhost> [completo]"
echo "Parámetro opcional 'completo' para escanear todos los puertos"
exit 1
fi

OBJETIVO="$1"
TIPO_ESCANEO="$2"

# Verificar si Nmap está instalado
if ! command -v nmap &> /dev/null; then
echo "Nmap no está instalado. Instalando..."
sudo apt update && sudo apt install -y nmap || {
 echo "No se pudo instalar Nmap. Abortando."
 exit 1
}
fi

# Definir tipo de escaneo
if [ "$TIPO_ESCANEO" == "completo" ]; then
echo "Ejecutando escaneo completo (-p-)..."
NMAP_CMD="nmap -sS -sV -p- --open $OBJETIVO"
else
```

```
echo "Ejecutando escaneo rápido (puertos comunes)..."
NMAP_CMD="nmap -sS -sV -Pn --open $OBJETIVO"
fi

# Ejecutar escaneo y guardar resultados
echo "Escaneando $OBJETIVO... esto puede tardar..."
RESULTADO=$($NMAP_CMD)

FECHA=$(date +%Y%m%d_%H%M%S)
ARCHIVO="reporte_nmap_${OBJETIVO}_${FECHA}.txt"
echo "$RESULTADO" > "$ARCHIVO"
echo "Resultado guardado en: $ARCHIVO"

# Verificar si se encontraron puertos abiertos
if ! echo "$RESULTADO" | grep -qE '^[0-9]'; then
echo "No se detectaron puertos abiertos en $OBJETIVO."
exit 0
fi

# Listas de clasificación
SEGUROS=(22 80 443)
SOSPECHOSOS=(3000 3306 8080)
PELIGROSOS=(23 21 6667)

# Mostrar clasificación
echo "Clasificación de puertos detectados:"
HAY_RIESGO=false

while read linea; do
 puerto=$(echo "$linea" | cut -d "/" -f1)
 servicio=$(echo "$linea" | awk '{print $3}')

 echo "→ Puerto abierto: $puerto - Servicio: $servicio"

 if [[ " ${SEGUROS[@]} " =~ " $puerto " ]]; then
    echo "   Clasificación: SEGURO"
 elif [[ " ${SOSPECHOSOS[@]} " =~ " $puerto " ]]; then
    echo "   Clasificación: SOSPECHOSO"
    HAY_RIESGO=true
 elif [[ " ${PELIGROSOS[@]} " =~ " $puerto " ]]; then
    echo "   Clasificación: PELIGROSO"
    HAY_RIESGO=true
```

```
else
    echo "   Clasificación: DESCONOCIDO"
 fi
done < <(echo "$RESULTADO" | grep ^[0-9])

# Mostrar recomendaciones si corresponde
if $HAY_RIESGO; then
 echo
 echo "⚠️ Se detectaron puertos sospechosos o peligrosos."
 echo "Recomendaciones:"
 echo "- Verificá si los servicios en esos puertos son necesarios."
 echo "- Si no se usan, cerralos con un firewall o regla de red."
 echo "- Aplicá reglas de acceso para restringir conexiones externas."
 echo "- Mantené actualizado el software que usa esos puertos."
 echo "- Considerá usar herramientas como fail2ban o firewalld."
 echo
 echo "👉 Si desea cerrar un puerto, ejecute: ./cerrar_puerto.sh <número_de_puerto>"
fi
```

4.2 RESULTADOS

●Para localhost:

```
marabayurk@cloudshell:~$ sudo ./test-3.sh localhost
Iniciando escaneo de puertos con Nmap
Ejecutando escaneo rápido (puertos comunes)...
Escaneando localhost... esto puede tardar...
Resultado guardado en: reporte_nmap_localhost_20250529_225957.txt
Clasificación de puertos detectados:
→ Puerto abierto: 22 - Servicio: ssh
   Clasificación: SEGURO
→ Puerto abierto: 900 - Servicio: http
   Clasificación: DESCONOCIDO
→ Puerto abierto: 981 - Servicio: http
   Clasificación: DESCONOCIDO
→ Puerto abierto: 3000 - Servicio: ppp?
   Clasificación: SOSPECHOSO
→ Puerto abierto: 1 service unrecognized despite returning data. If you know the service - Servicio: unrecognized
   Clasificación: DESCONOCIDO

⚠️ Se detectaron puertos sospechosos o peligrosos.
Recomendaciones:
- Verificá si los servicios en esos puertos son necesarios.
- Si no se usan, cerralos con un firewall o regla de red.
- Aplicá reglas de acceso para restringir conexiones externas.
- Mantené actualizado el software que usa esos puertos.
- Considerá usar herramientas como fail2ban o firewalld.

👉Si desea cerrar un puerto, ejecute: ./cerrar_puerto.sh <número_de_puerto>
marabayurk@cloudshell:~$
```

➔ interpretación de resultados:

- El puerto 22 (SSH), es el protocolo seguro de Shell.
- El puerto 900 con servicios http pueden ser aplicaciones web locales que se estén ejecutando (app o servicio de desarrollo).UDP TCP OMG Referencias iniciales.
- El puerto 981 se encuentra en un rango(954-988) de puertos sin referencia asignada.

- El puerto 3000 puede ser una aplicación de desarrollo (muchos frameworks usan ese puerto). Según IANA “esta entrada registra un uso no asignado pero generalizado”.
- El puerto con servicio “unrecognized” puede ser algo personalizado o un servicio no estándar.

- Para mi direccion IP privada:

```
marabayurk@cloudshell:~$ sudo ./final.sh 192.168.0.175
Iniciando escaneo de puertos con Nmap
Ejecutando escaneo rápido (puertos comunes)...
Escaneando 192.168.0.175... esto puede tardar...
Resultado guardado en: reporte_nmap_192.168.0.175_20250529_221955.txt
No se detectaron puertos abiertos en 192.168.0.175.
marabayurk@cloudshell:~$
```

```
marabayurk@cloudshell:~$ sudo ./test-3.sh localhost
Iniciando escaneo de puertos con Nmap
Ejecutando escaneo rápido (puertos comunes)...
Escaneando localhost... esto puede tardar...
Resultado guardado en: reporte_nmap_localhost_20250529_225957.txt
Clasificación de puertos detectados:
→ Puerto abierto: 22 - Servicio: ssh
   Clasificación: SEGURO
→ Puerto abierto: 900 - Servicio: http
   Clasificación: DESCONOCIDO
→ Puerto abierto: 981 - Servicio: http
   Clasificación: DESCONOCIDO
→ Puerto abierto: 3000 - Servicio: ppp?
   Clasificación: SOSPECHOSO
→ Puerto abierto: 1 service unrecognized despite returning data. If you know the service - Servicio: unrecognized
   Clasificación: DESCONOCIDO

⚠️ Se detectaron puertos sospechosos o peligrosos.
Recomendaciones:
- Verificá si los servicios en esos puertos son necesarios.
- Si no se usan, cerralos con un firewall o regla de red.
- Aplicá reglas de acceso para restringir conexiones externas.
- Mantené actualizado el software que usa esos puertos.
- Considerá usar herramientas como fail2ban o firewalld.

👉Si desea cerrar un puerto, ejecute: ./cerrar_puerto.sh <número_de_puerto>
marabayurk@cloudshell:~$
```

- Usando el flag - - completo

  > ip privada

```
marabayurk@cloudshell:~$ sudo ./test-3.sh 192.168.0.175 --completo
Iniciando escaneo de puertos con Nmap
Ejecutando escaneo rápido (puertos comunes)...
Escaneando 192.168.0.175... esto puede tardar...
Resultado guardado en: reporte_nmap_192.168.0.175_20250529_232651.txt
No se detectaron puertos abiertos en 192.168.0.175.
marabayurk@cloudshell:~$
```

  > localhost

```
marabayurk@cloudshell:~$ sudo ./test-3.sh localhost --completo
Iniciando escaneo de puertos con Nmap
Ejecutando escaneo rápido (puertos comunes)...
Escaneando localhost... esto puede tardar...
Resultado guardado en: reporte_nmap_localhost_20250529_232742.txt
Clasificación de puertos detectados:
→ Puerto abierto: 22 - Servicio: ssh
   Clasificación: SEGURO
→ Puerto abierto: 900 - Servicio: http
   Clasificación: DESCONOCIDO
→ Puerto abierto: 981 - Servicio: http
   Clasificación: DESCONOCIDO
→ Puerto abierto: 3000 - Servicio: ppp?
   Clasificación: SOSPECHOSO
→ Puerto abierto: 1 service unrecognized despite returning data. If you know the service - Servicio: unrecognized
   Clasificación: DESCONOCIDO

⚠️ Se detectaron puertos sospechosos o peligrosos.
Recomendaciones:
- Verificá si los servicios en esos puertos son necesarios.
- Si no se usan, cerralos con un firewall o regla de red.
- Aplicá reglas de acceso para restringir conexiones externas.
- Mantené actualizado el software que usa esos puertos.
- Considerá usar herramientas como fail2ban o firewalld.

👉Si desea cerrar un puerto, ejecute: ./cerrar_puerto.sh <número_de_puerto>
marabayurk@cloudshell:~$
```

En estos casos, cuando se encontraron puertos potencialmente peligrosos o sospechosos, se emitieron mensajes que podrían ayudar a la mitigación y prevención de brechas de seguridad. Como medida rápida creamos otro script para el cierre de puertos automatizado.

4.3 SCRIPT DE CIERRE

- Iniciamos con la creación de un script llamado “cerrar_puerto” con el comando: nano cerrar_puerto.sh
- Luego desarrollamos el script de cierre con las comprobaciones necesarias.
- Luego le dimos los permisos necesarios con el comando: chmod +x cerrar_puerto.sh
- A continuación lo ejecutamos con el comando: sudo ./cerrar_puerto.sh 3000. Al finalizar el nombre del puerto colocamos el número que lo identifica para poder proceder al cierre.

```bash
#!/bin/bash

# Verifica que haya recibido un número de puerto
if [ -z "$1" ]; then
  echo "Debe ingresar un número de puerto. Uso: ./cerrar_puerto.sh 3000"
  exit 1
fi

PUERTO=$1

# Cierra el puerto usando iptables (requiere sudo)
echo "Cerrando puerto $PUERTO..."
sudo iptables -A INPUT -p tcp --dport $PUERTO -j DROP
echo "Puerto $PUERTO bloqueado con éxito."
```

4.4 RESULTADO DE SCRIPT DE CIERRE

En esta captura se observa como el puerto es cerrado y el mensaje final.

```
linaberrone@cloudshell:~$ nano cerrar_puerto.sh
linaberrone@cloudshell:~$ chmod +x cerrar_puerto.sh
linaberrone@cloudshell:~$ sudo cerrar_puerto.sh 3000
sudo: cerrar_puerto.sh: command not found
linaberrone@cloudshell:~$ sudo ./cerrar_puerto.sh 3000
Cerrando puerto 3000...
Puerto 3000 bloqueado con éxito.
linaberrone@cloudshell:~$
```

4.5 CASUÍSTICAS

- Caso donde la dirección IP no devuelve el ping y como resolverlo. (192.168.0.1)
    - Entorno de pruebas de GCS
    - Ejecutamos el script con nuestro puerto 192.168.0.1

```
marabayurk@cloudshell:~$ sudo ./test-2.sh 192.168.0.1
-----------------------------------------------
Iniciando proceso de escaneo con Nmap
-----------------------------------------------
Realizando escaneo sobre 192.168.0.1...
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-05-27 22:00 UTC
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 3.18 seconds
Escaneo finalizado. Resultados guardados en: reporte_nmap_20250527_220029.txt
marabayurk@cloudshell:~$
```

- Ese mensaje de Nmap significa que no pudo detectar que el host (192.168.0.1) esté activo porque el ping que usa para descubrirlo fue bloqueado o no respondió.

En resumen:

- "Host seems down" → Nmap no recibió respuesta al ping inicial.
- Muchos dispositivos bloquean pings por seguridad, pero pueden estar activos.
- Por eso, Nmap sugiere usar la opción -Pn, que ignora el ping y escanea directamente sin chequear si el host responde al ping.
- Para solucionar esto cambiamos en el script como ejecutamos el nmap

`NMAP_CMD="nmap -sS -sV -Pn --open $OBJETIVO"`

La opción -Pn le dice a Nmap que no haga ping previo para detectar si el host está activo, y que haga el escaneo directamente.

- Este es el resultado del escaneo sin hacer el ping previamente

```
marabayurk@cloudshell:~$ sudo ./test-2.sh 192.168.0.1
-----------------------------------------------
Iniciando proceso de escaneo con Nmap
-----------------------------------------------
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-05-27 22:09 UTC
Nmap scan report for 192.168.0.1
Host is up.
All 1000 scanned ports on 192.168.0.1 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 201.53 seconds
Escaneo finalizado. Resultados guardados en: reporte_nmap_20250527_220936.txt
marabayurk@cloudshell:~$
```

- El siguiente es el resultado del escaneo de NMAP en el archivo que se generó

```
# Nmap 7.94SVN scan initiated Tue May 27 21:47:00 2025 as: nmap -sS -sV -oN reporte_nmap_20250527_214700.txt localhost
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000070s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 996 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 9.6p1 Ubuntu 3ubuntu13.11 (Ubuntu Linux; protocol 2.0)
900/tcp  open  http    BaseHTTPServer 0.6 (Python 3.11.2)
981/tcp  open  http    Golang net/http server (Go-IPFS json-rpc or InfluxDB API)
3000/tcp open  ppp?
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port3000-TCP:V=7.94SVN%I=7%D=5/27%Time=683632E0%P=x86_64-pc-linux-gnu%r
SF:(GetRequest,172,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nSet-Cookie:\x20v
SF:scode-secret-key-path=/cde/mint-key;\x20Path=/;\x20Secure;\x20SameSite=
SF:None;\x20Partitioned\r\nSet-Cookie:\x20vscode-cli-secret-half=XCRZ681A7
SF:wxJfiFn2-163uCk2KQ-ar29UBsPXgcS6qA;\x20Max-Age=2592000000;\x20Path=/;\x
SF:20HttpOnly;\x20Secure;\x20SameSite=None;\x20Partitioned\r\nContent-Type
SF::\x20text/plain\r\nDate:\x20Tue,\x2027\x20May\x202025\x2021:47:12\x20GM
SF:T\r\nConnection:\x20close\r\n\r\nBad\x20request\.")%r(Help,2F,"HTTP/1\.
SF:1\x20400\x20Bad\x20Request\r\nConnection:\x20close\r\n\r\n")%r(NCP,2F,"
SF:HTTP/1\.1\x20400\x20Bad\x20Request\r\nConnection:\x20close\r\n\r\n")%r(
SF:HTTPOptions,172,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nSet-Cookie:\x20v
SF:scode-secret-key-path=/cde/mint-key;\x20Path=/;\x20Secure;\x20SameSite=
SF:None;\x20Partitioned\r\nSet-Cookie:\x20vscode-cli-secret-half=A8J3B1FVa
SF:qt8vGAnwh65suJr71z4pEYt5GzLpE2EZDI;\x20Max-Age=2592000000;\x20Path=/;\x
SF:20HttpOnly;\x20Secure;\x20SameSite=None;\x20Partitioned\r\nContent-Type
SF::\x20text/plain\r\nDate:\x20Tue,\x2027\x20May\x202025\x2021:47:12\x20GM
SF:T\r\nConnection:\x20close\r\n\r\nBad\x20request\.")%r(RTSPRequest,172,"
SF:HTTP/1\.1\x20400\x20Bad\x20Request\r\nSet-Cookie:\x20vscode-secret-key-
SF:path=/cde/mint-key;\x20Path=/;\x20Secure;\x20SameSite=None;\x20Partitio
SF:ned\r\nSet-Cookie:\x20vscode-cli-secret-half=aOH94vAjJgsxUauagOlL0OqkV-
SF:wKJjVQA6WfSNbLcCM;\x20Max-Age=2592000000;\x20Path=/;\x20HttpOnly;\x20Se
SF:cure;\x20SameSite=None;\x20Partitioned\r\nContent-Type:\x20text/plain\r
SF:\nDate:\x20Tue,\x2027\x20May\x202025\x2021:47:12\x20GMT\r\nConnection:\
SF:x20close\r\n\r\nBad\x20request\.")%r(RPCCheck,2F,"HTTP/1\.1\x20400\x20B
SF:ad\x20Request\r\nConnection:\x20close\r\n\r\n")%r(DNSVersionBindReqTCP,
SF:2F,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nConnection:\x20close\r\n\r\n"
SF:)%r(DNSStatusRequestTCP,2F,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nConne
SF:ction:\x20close\r\n\r\n")%r(SSLSessionReq,2F,"HTTP/1\.1\x20400\x20Bad\x
```

```
SF:20Request\r\nConnection:\x20close\r\n\r\n")%r(TerminalServerCookie,2F,"
SF:HTTP/1\.1\x20400\x20Bad\x20Request\r\nConnection:\x20close\r\n\r\n");
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Tue May 27 21:47:12 2025 -- 1 IP address (1 host up) scanned in 11.55 seconds
```

4.6 TRABAJO COLABORATIVO

El trabajo fue desarrollado de forma conjunta por ambas integrantes del grupo. Cada una realizó investigaciones individuales y pruebas locales en su propio entorno, con el fin de explorar las distintas posibilidades de escaneo y comprender el funcionamiento de las herramientas utilizadas. A partir de esos avances, se compartieron observaciones, se debatieron mejoras y se consensuaron ajustes al script final.

La redacción del informe también se llevó a cabo de manera colaborativa, combinando los aportes individuales y revisando en conjunto los contenidos teóricos y prácticos para garantizar la coherencia del documento.

5. Resultados Obtenidos

En estas pruebas logramos identificar los puertos abiertos (- - open) en nuestros sistemas operativos, y clasificarlos de acuerdo a la amenaza que presentaba cada uno. En ambas computadoras se analizó el “localhost” y al probar el script en el mismo entorno (GCS) devolvía que ambas teníamos los mismo puertos. Verificamos que puertos eran según el número identificatorio y según la tabla de IANA poder clasificarlos y saber a que pertenecían. Tuvimos que realizar varios ajustes al momento del comando de ejecución utilizado para Nmap.

- -sS: Escaneo SYN (rápido y sigiloso).
- -sV: Detecta los servicios y sus versiones en puertos abiertos.
- -p-: Escanea todos los puertos TCP del 1 al 65535.
- --open: Muestra solo los puertos que están abiertos.

Además se realizaron iteraciones sobre el script. Agregando y quitando features como validaciones, sugerencias y recomendaciones en caso de vulnerabilidades encontradas, etc. Fue dejada evidencia de este proceso en el repositorio del TP. (https://github.com/MaraBayurk/TUP-AYSO-1-TP-INTEGRADOR).

Como complicaciones se presentó la instalación de Nmap en una computadora con sistema operativo MAC OS, que resolvimos dentro del script, y se debía dar permisos de instalación a las aplicaciones descargadas desde el navegador. Otra de las complicaciones se evidenció al momento de la ejecución; la primera prueba fue sin la inclusión de la palabra SUDO por lo que el entorno de GCS no pudo ejecutar el script debido a que sudo: ./script.sh: command not found , es decir, el script no tiene permisos de ejecución.

El ingreso de una dirección IP como parámetro funcionó correctamente, permitiendo realizar escaneos personalizados. Los resultados se almacenaron correctamente en archivos de texto con nombres generados dinámicamente según la fecha, lo cual facilitó su identificación. El archivo está incluido dentro del repositorio mencionado.

Otro ajuste pertinente tuvo que ser incluido a la hora de probar una dirección IP privada, en ese caso listado aparte, lo solucionamos colocando la opción -Pn, que ignora el ping y escanea directamente sin chequear si el host responde al ping.

6. Conclusiones

A lo largo de este trabajo se abordó de forma integral el proceso de reconocimiento activo de sistemas, con especial énfasis en el análisis de puertos, utilizando la herramienta Nmap integrada en un script Bash desarrollado y probado en Google Cloud Shell. Esta experiencia permitió aplicar los conceptos teóricos sobre footprinting y gestión de vulnerabilidades en un entorno controlado.

La ejecución del script permitió identificar puertos abiertos, clasificar los servicios detectados en función de su nivel de riesgo y generar recomendaciones específicas para la mitigación de posibles brechas de seguridad. Además, se desarrolló un segundo script complementario que automatiza el cierre de puertos peligrosos, lo cual representa una medida inmediata y efectiva de protección ante amenazas potenciales.

Durante el desarrollo, se superaron distintos desafíos técnicos, como la instalación de herramientas en distintos sistemas operativos, la validación de parámetros, el tratamiento de casos donde los hosts no responden a ping y la correcta ejecución con permisos elevados. Estos aprendizajes consolidaron habilidades prácticas en scripting, seguridad de redes y diagnóstico de sistemas.

Como posibles mejoras futuras se podría ampliar la base de clasificación de puertos, conectándola con bases de datos externas de vulnerabilidades (como CVEs, programa para definir y catalogar vulnerabilidades), para ofrecer un análisis más detallado y actualizado.

Este trabajo no solo permitió aplicar los contenidos teóricos, sino que también fomentó el desarrollo de una solución concreta, funcional y reutilizable, que podría implementarse en distintos contextos donde se requiera una auditoría rápida y eficaz del estado de red.

7. Bibliografía

- Skoudis, E., & Liston, T. (2006). *Counter Hack Reloaded: A Step-by-Step Guide to Computer Attacks and Effective Defenses*. Prentice Hall.
- Nmap Official Documentation: https://nmap.org/book/man.html
- RFC 6335 – Internet Assigned Numbers Authority (IANA): Service Name and Transport Protocol Port Number Registry.
- https://www-iana-org.translate.goog/assignments/service-names-port-numbers/service-names-port-numbers.xhtml?_x_tr_sch=http&_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc (29/05/2025)
- https://es.m.wikipedia.org/wiki/Internet_Assigned_Numbers_Authority (29/05/2025)
- https://www.cve.org/about/overview (29/05/2025)

8. Video subido a youtube

Les compartimos el link para el video grabado y subido en youtube: https://youtu.be/pozkJi-EXSU