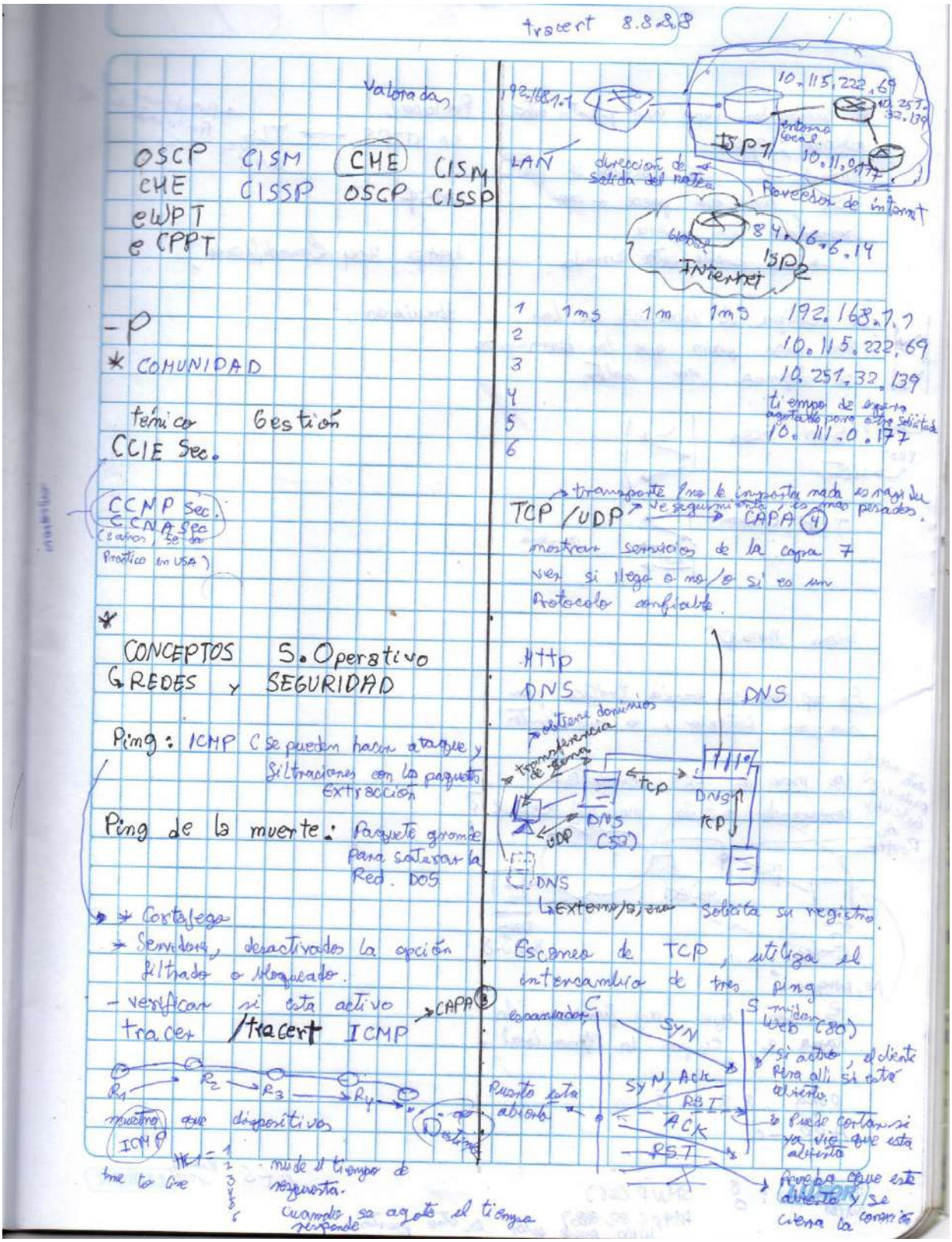
Conceptos teóricos y operativos de redes y sistemas



ipconfig /all

Who is

↳ Dominio info

El escaneador verá que puertos están abiertos ; open
Cuando no está abierto el puerto RST, con eso puede o no responder en espera
↳ ese puerto está cerrado

Se realiza el escaneo en las tres vías para que las herramientas de defensa no salten

(sudo) SYN Scan
root
↳ saltan

TCP Scan

Scan nmap?

En udp solo se envía tráfico por mayor intento, es más lentos

¿La mac de DGT MitraStar corresponde a la mac del 8.8.8.8?

esta mac pertenece Gateway Router

No, porque:
Es por que así funciona la capa 2, es por la zona local.

ping 8.8.8.8
ARP -a

Protocol
en HTTPS → TLS
QUIC
mSF
HTTP 304 (no modificado)
direcciones.

Ida
Vuelta Type : 8
0

SMNP (25)
Http (80, 888)
Web puede estar en otro puerto

hFS Puerto 117

Esto

o será una clase de redes, pero sí vamos a repasar **conceptos esenciales para la ciberseguridad**. Porque, como ya les he mencionado:

“Si no conoces cómo funciona tu red, ¿cómo vas a protegerla?”

Debes entender:

- ¿Qué hace una aplicación?
- ¿Qué flujo sigue para comunicarse con los servidores?
- ¿A qué servicios consulta?
- ¿Cómo responden esos servicios?

Por ejemplo, en una aplicación web, deberías poder **interpretar los códigos de respuesta HTTP**:

- 200: OK → Cuando haces una solicitud (como abrir una página web) y el servidor **te responde con éxito**, devolviendo el contenido esperado.
- 400: Bad Request → Cuando el navegador **envía algo mal**, como datos incompletos, mal codificados, o errores de sintaxis en la petición.
- 403: Forbidden → El servidor **reconoce la solicitud, pero no permite el acceso** al recurso, incluso si estás autenticado. Por ejemplo, si tratas de entrar a una carpeta privada sin permisos.
- 500: Internal Server Error → Cuando algo sale mal **dentro del servidor** al intentar procesar tu solicitud, pero no se da un detalle específico del error.
- etc.

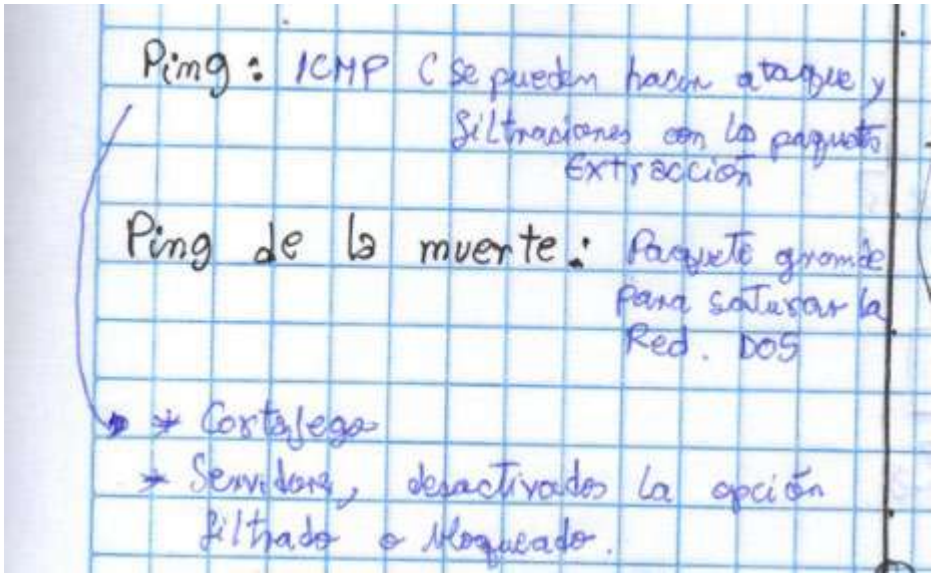
Todos esos códigos tienen utilidad **a nivel de seguridad**, ya que permiten identificar **fallos, comportamientos anómalos o vulnerabilidades**.

📡 Protocolo ICMP y su uso en redes y seguridad

Todos conocemos este tema: el protocolo **ICMP** (Internet Control Message Protocol).
¿Pero qué es realmente?

- **ICMP e**s un **protocolo de control**, que se usa principalmente para **verificar conectividad** entre dispositivos en una red.

Un ejemplo típico es el **comando ping**, que utiliza ICMP para enviar paquetes a otro equipo y verificar si está **activo o responde**.



Aunque ping parece inofensivo, **ICMP tiene implicancias de seguridad**.

- En el pasado, ICMP fue utilizado para realizar ataques de **denegación de servicio (DoS)**, enviando grandes volúmenes de paquetes que **saturaban el sistema receptor**.
- **Hace 40 o 50 años**, este método era uno de los favoritos para dejar fuera de servicio a un servidor.
- Hoy en día, este tipo de ataque se conoce como **"ping de la muerte"** (ping of death), aunque actualmente ya está controlado por la mayoría de sistemas modernos.

🧪 ¿Para qué sirve ICMP en la práctica?

Sirve principalmente para:

- **Verificar conectividad**: saber si un equipo está encendido, activo y accesible en la red.
- **Diagnóstico de red**: ayuda a detectar dónde puede haber un problema de conexión.

❗¿Qué pasa si un equipo no responde al ping?

Hay varias razones por las que un equipo **puede estar encendido pero no responder a ICMP**:

1. **El cortafuegos lo está bloqueando**
   Muchos sistemas tienen configurado su firewall para **filtrar los paquetes ICMP**. En ese caso, aunque el equipo esté activo, **no responderá al ping**.
2. **Bloqueo intencional por motivos de seguridad**
   Algunos **servidores** desactivan la respuesta a ICMP para evitar:
   - Escaneos de red (por ejemplo, usando ping para identificar hosts activos).

- Posibles intentos de DoS.

🔍 ¿Cómo detectar si una máquina está activa aunque no responda al ping?

Cuando se realiza un escaneo de red (por ejemplo, en una subred tipo 192.168.1.1 - 192.168.1.254), el procedimiento común es hacer:

shell

CopyEdit

ping 192.168.1.1

ping 192.168.1.2

ping 192.168.1.3

...

Y con base en las respuestas se concluye qué dispositivos están **activos**.

🟢 **Clase A**

Privada: 10.0.0.0 a 10.255.255.255

- **Uso**: Grandes redes (empresas muy grandes o ISPs)

🟠 **Clase B**

Privada: 172.16.0.0 a 172.31.255.255

Uso: Universidades, medianas empresas

🔵 **Clase C**

Privada: 192.168.0.0 a 192.168.255.255

Uso: Pequeñas redes (hogares, oficinas)

Pero **esto puede ser un error**: si una IP **no responde al ping**, podrías pensar que el equipo está **apagado o inexistente**, cuando en realidad **solo está filtrando ICMP**.

Entonces, la gran pregunta es:

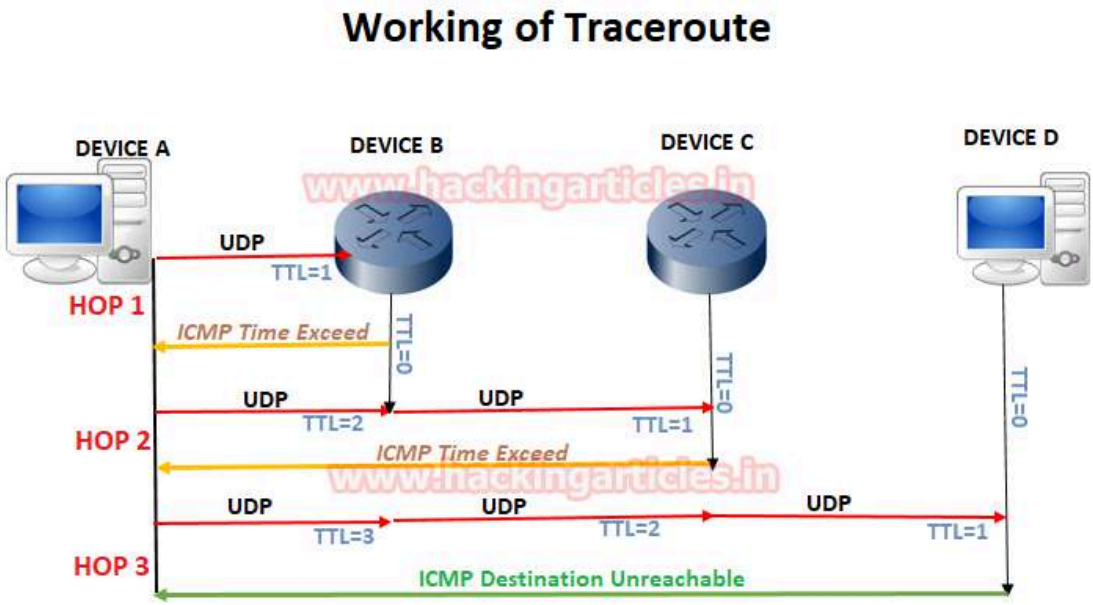
¿Cómo saber si una máquina está activa si no responde al ping?

Ahí entran en juego otras técnicas más avanzadas, como:

- Uso de **escaneos TCP SYN** con herramientas como **Nmap**.
- Verificación de puertos abiertos (por ejemplo, si responde al puerto 80 o 443).
- Monitoreo ARP o tráfico de red para detectar actividad del dispositivo.

🌐 Trazado de ruta y análisis del protocolo ICMP

Cuando hablamos de conectividad básica en redes, no basta con saber si hay o no conexión; también es importante **seguir el trayecto del paquete** para entender **cómo se mueve por la red**. Para eso usamos una herramienta conocida como **tracert** (en Windows) o **traceroute** (en Linux).



🔎 ¿Qué hace realmente el comando tracert?

El comando tracert **muestra la ruta** que sigue un paquete desde tu equipo hasta el destino final. Lo que hace es registrar todos los **saltos intermedios** (routers, gateways, etc.) por l que pasa el paquete.

Imaginemos que el paquete sale de tu computadora (host A) y quiere llegar al host B. En el camino, pasará por múltiples dispositivos de red: router 1, router 2, router 3, y así sucesivamente.

El resultado del tracert es un **listado ordenado de todos los routers** que ha atravesado el paquete, junto con los tiempos de respuesta de cada uno.

El comando **no es un protocolo en sí**, sino que usa el **protocolo ICMP (Internet Control Message Protocol)** para realizar su función.

El mecanismo es el siguiente:

1. Envía un paquete ICMP (como el ping), pero con un **TTL (Time To Live)** inicial de **1**.
2. Ese paquete llega al **primer router**, el TTL se reduce a 0 y el router **responde con un mensaje "Time Exceeded"** (tiempo excedido).
3. Luego se envía un nuevo paquete con TTL = 2, que llega al **segundo router**, y así sucesivamente.
4. Cada router responde con su IP cuando el TTL se agota.
5. Finalmente, cuando el paquete llega al destino, este **responde normalmente**, cerrando la ruta trazada.

De esta forma, tracert **va construyendo el camino paso a paso**, recolectando las direcciones IP de todos los dispositivos intermedios.

⚠️ ¿Qué pasa si un salto no responde?

Cuando algún router o equipo **no responde al paquete ICMP**, lo que verás en el trazado es un símbolo como * * *, o un mensaje como **"Tiempo excedido"** o **"Host inalcanzable"**. Esto puede deberse a:

- **Filtros de cortafuegos** (el equipo está activo, pero no responde a ICMP).
- **Restricciones del servidor** (configurado para ignorar paquetes ICMP por seguridad).
- **Problemas de red** (paquetes perdidos, rutas mal configuradas, etc.).

🧪 Ejemplo práctico del análisis de ruta

En un análisis de ruta típico (tracert www.ejemplo.com), podemos observar:

- TTL inicial = 1: responde el **primer router**.
- TTL = 2: responde el **segundo router**.
- ...
- Hasta que finalmente el destino **responde sin error**, lo que indica que se ha llegado correctamente.

Cuando aparece una respuesta con:

- Tiempo excedido: se agotó el TTL antes de llegar al destino.
- Destino inalcanzable: el paquete no pudo llegar por problemas de enrutamiento o configuración.

El análisis de las respuestas ICMP (como los mensajes tipo "Echo Reply" o "Time Exceeded") también es muy útil en **ciberseguridad**, más allá del simple diagnóstico de conectividad.

Este tipo de análisis permite:

- Verificar la **segmentación de la red**.
- Identificar **qué segmentos o saltos de red** se alcanzan y cuáles no.
- Determinar si existe algún **firewall, proxy o dispositivo intermedio** que esté filtrando el tráfico.

Por ejemplo, al hacer un **traceroute**, puedes observar:

- Qué direcciones IP se devuelven.
- Si las respuestas provienen de **dispositivos públicos o privados**.
- Si los saltos intermedios parecen corresponder a **cloud providers**, **gateways**, **switches de capa 3**, **routers**, **proxys**, etc.

📍 ¿Cómo identificar direcciones IP privadas?

Es fundamental que puedas **reconocer cuándo una dirección IP es privada o pública**. Para ello, debes recordar los rangos reservados en cada clase:

Rangos de direcciones IP privadas:

- **Clase A**: 10.0.0.0 – 10.255.255.255
- **Clase B**: 172.16.0.0 – 172.31.255.255
- **Clase C**: 192.168.0.0 – 192.168.255.255

Cualquier IP dentro de esos rangos **no es enrutable en Internet** directamente, por lo que se utiliza dentro de redes privadas.

🌐 ¿Qué significan estas IPs en el contexto del análisis?

Cuando analizas los resultados de un traceroute, por ejemplo, y ves direcciones privadas como:

- 10.x.x.x
- 172.16.x.x
- 192.168.x.x

Significa que estás observando **segmentos internos de red**, posiblemente dentro de la organización o proveedor de servicios.

Si en cambio ves direcciones públicas, pueden pertenecer a:

- Dispositivos perimetrales.
- Servicios en la nube (por ejemplo, AWS, Azure).
- Routers o proxys intermedios.

🔄 ¿Qué es la puerta de enlace (gateway)?

Durante el análisis de red es común preguntarse:

¿Quién es la puerta de enlace? ¿Es siempre un router?

La respuesta es **no necesariamente**.

Aunque tradicionalmente se nos enseña que la **puerta de enlace predeterminada** (gateway) es un **router**, en la práctica también puede ser:

- Un **switch capa 3**
- Un **servidor proxy**
- Un **firewall**
- O cualquier dispositivo capaz de **rutar tráfico entre redes diferentes**

Lo importante es entender que el gateway es el **dispositivo que permite la salida desde una red local hacia otra red o Internet**, sin importar el tipo físico exacto del equipo.

📌 Resumen práctico

- El comando tracert o traceroute usa **ICMP con TTL creciente** para descubrir la ruta entre dos puntos de red.
- Las **respuestas ICMP** permiten analizar la conectividad y **detectar qué dispositivos intermedios** están activos o filtrando paquetes.
- La observación de **direcciones IP privadas** permite identificar si estás en una red interna o pública.
- La **puerta de enlace** es esencial para la conexión entre redes, y puede ser más que solo un router: también puede ser un switch, proxy, firewall o servidor.

🌐 Análisis de rutas y direcciones IP privadas/públicas

Gracias por su atención. Antes de cerrar, quiero hacer algunas aclaraciones importantes sobre las **direcciones IP privadas y su relación con Internet**.

📍 ¿Las IP privadas se pueden enrutar en Internet?

Técnicamente, **las direcciones IP privadas no se enrutan directamente en Internet**. Sin embargo, cuando se utiliza **NAT (Traducción de Direcciones de Red)**, el tráfico interno **puede salir a Internet usando una IP pública**. En ese caso, desde afuera solo se vería la **IP pública del proveedor**, no las IP privadas internas.

Entonces, si durante un análisis de red observamos **direcciones IP privadas**, significa que **aún no hemos salido al Internet público**. Estamos viendo la red interna del proveedor o de nuestra propia organización.

🏠 Ejemplo: Interpretando un traceroute

Supongamos que estamos analizando la ruta que toma un paquete desde nuestra red hasta un servidor externo. Podemos ver algo como esto:

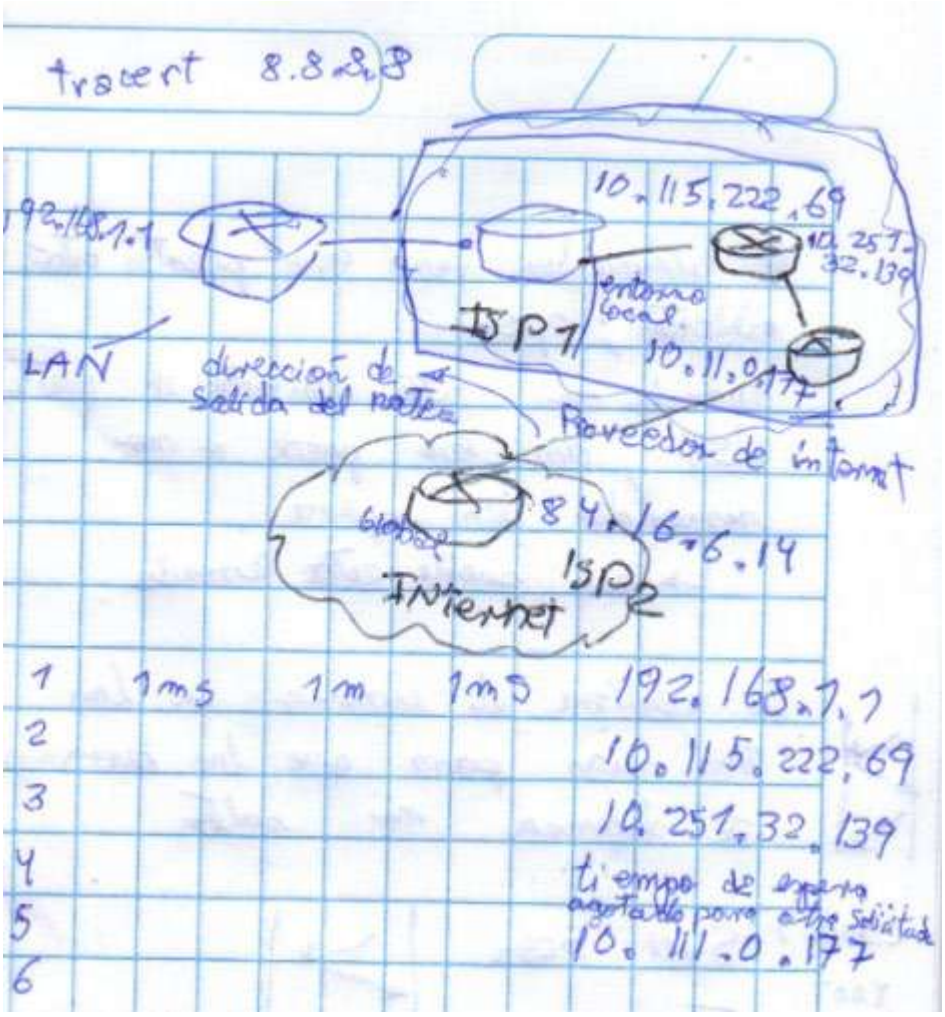
1. **Primer salto**: 192.168.1.1
   → Este es el **router local** (nuestro gateway en la LAN).
2. **Segundo salto**: 10.0.0.22
   → Pertenece a un **dispositivo de capa 3** del proveedor, aún dentro de su red interna.

3. **Tercer salto**: 10.0.0.30
   → Otro equipo del ISP, probablemente su propio router de distribución.

4. **Cuarto salto**: No hay respuesta
   → Podría deberse a que el dispositivo está **filtrando ICMP**, por razones de seguridad. No necesariamente está apagado.

5. **Quinto salto**: 190.x.x.x
   → Ahora sí estamos viendo una **IP pública**, lo que indica que **hemos salido a Internet**.

6. **Sexto salto**: 84.80.x.x
   → Ya estamos en **dominios externos** o enrutados fuera del ISP inicial. Puede ser un proveedor internacional.



🧭 ¿Qué nos dice esto sobre la topología?

Esto nos permite inferir que:

- **Nuestro router** pertenece a una red LAN que forma parte de un **dominio mayor** (como la red interna del proveedor o una red empresarial).
- Recién a partir del **tercer o cuarto salto**, el tráfico **sale al Internet público**.
- Las direcciones **IP públicas visibles** pueden pertenecer a:
    - El proveedor de Internet (ISP).
    - Un proveedor secundario (ISP2).
    - Un servidor intermedio o infraestructura compartida.

🧩 ¿Y si todas las IPs fueran privadas?

En otro escenario, podríamos ver que todos los saltos intermedios tienen direcciones privadas como 10.x.x.x o 192.168.x.x. Esto puede indicar que:

- La ruta está pasando por **varios dispositivos internos** dentro de una misma organización (como un campus, una universidad o una empresa grande).
- **Ejemplo**: Si estás en una universidad como la UNAM, puedes pasar por:
    - Un **proxy institucional**.
    - Un **router de salida local**.
    - Un segundo **proxy nacional**.
    - Y recién después salir a Internet.

En este caso, aunque las direcciones IP sean privadas, forman parte del **dominio de red bajo control institucional** y no necesariamente del proveedor externo.

🔍 Verificando la IP de salida

Podemos ver cuál es la **IP pública de salida** usando comandos como:

bash

CopyEdit

curl ifconfig.me

O simplemente revisando en un servicio como https://whatismyip.com.
En ese resultado verás tu IP pública (por ejemplo, 190.x.x.x), que puede diferir bastante de la que muestra tu red interna (como 192.168.1.100).

En muchos casos, la IP que se muestra como **puerta de enlace de salida** (por ejemplo, 192.168.1.1) es simplemente el **dispositivo NAT** local, no el que realmente representa la salida a Internet.

✅ Conclusión

- Las **IP privadas** no son visibles en Internet, pero **pueden salir a través de NAT**.
- Durante un análisis de ruta (traceroute), si se observan IPs privadas, todavía **estás dentro del entorno local o del ISP**.
- **El salto a Internet se detecta al ver una IP pública**.
- La estructura y cantidad de saltos intermedios varía según:
    - El proveedor.
    - La infraestructura de red.
    - La existencia de proxys o filtros.

🤔 ICMP y visibilidad de IPs: NAT y trazado

Al hacer un trazado de ruta (traceroute), **no siempre puedes ver tu IP pública desde dentro de tu red**. Esto se debe al funcionamiento de **NAT (Network Address Translation)**:

- Cuando el paquete sale hacia Internet, tu router hace una **traducción de IP privada a pública**.
- Sin embargo, al recibir la respuesta de vuelta, se hace un **NAT inverso** y tú solo ves **la IP privada del dispositivo local**, no la IP pública usada hacia afuera.

Por eso, **el servidor remoto ve tu IP pública**, pero **tú ves tu IP privada local**. Esto es completamente normal y es una de las razones por las que, en seguridad, ICMP también sirve para:

- **Trazar rutas internas y externas**.
- **Analizar segmentación de red**.
- **Detectar dispositivos filtrando o bloqueando tráfico**.

📦 Capa 4 del modelo OSI: TCP y UDP

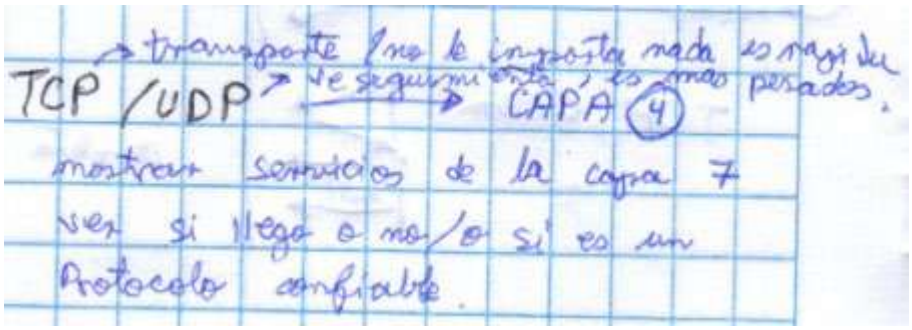
Ahora bien, entrando en la **capa de transporte del modelo OSI**, tenemos los dos protocolos más conocidos:

- **TCP (Transmission Control Protocol)**
- **UDP (User Datagram Protocol)**

🧠 ¿Cuál es la diferencia entre TCP y UDP?

- **TCP** es un **protocolo confiable**:
    - Utiliza confirmación de recepción (ACK).
    - Si el paquete se pierde, **lo reenvía**.
    - Garantiza que los datos lleguen **en orden y completos**.
    - Es más lento, pero más seguro.
- **UDP** es **más rápido y ligero**:
    - No hay confirmación.
    - No garantiza entrega.
    - No reenvía paquetes perdidos.
    - Se usa cuando la **velocidad es más importante que la confiabilidad**.

No significa que UDP sea “inseguro” o “malo”, sino que simplemente **tiene otro propósito**.



🧪 ¿Qué servicios usan TCP y cuáles usan UDP?

Servicios comunes que usan TCP:

- **HTTP / HTTPS** (Web)
- **FTP** (Transferencia de archivos)
- **SMTP** (Correo saliente)

- **POP3 / IMAP** (Correo entrante)
- **SSH** (Acceso remoto seguro)
- **Telnet**
- **DNS** (cuando hay transferencia de zona)

Servicios comunes que usan UDP:

- **DNS** (resolución estándar)
- **DHCP**
- **TFTP**
- **SNMP**
- **VoIP / Streaming**
- **NTP**

DNS: el caso especial

El **DNS (Domain Name System)** puede funcionar con **ambos protocolos**:

- Usa **UDP por defecto** (puerto 53) para consultas rápidas.
- Usa **TCP** cuando hay:
    - **Transferencia de zona (zone transfer)**.
    - **Respuestas grandes que no caben en UDP**.

🌎 ¿Qué hace exactamente el DNS?

DNS es un sistema que **traduce nombres de dominio** (como www.google.com) en **direcciones IP** (como 142.250.190.14).
Podemos verlo como una **base de datos distribuida** que relaciona:

- Nombre de dominio ⇄ Dirección IP correspondiente.

⚠️ Ataques relacionados al DNS: transferencia de zona

Un atacante puede explotar una mala configuración del servidor DNS mediante la técnica de **transferencia de zona (Zone Transfer)**. ¿Qué sucede?

- El atacante se hace pasar por un servidor autorizado y solicita la **descarga de los registros DNS completos**.

- Si el servidor está mal configurado y **no valida quién solicita la información**, entonces le entrega todos los registros.
- Esto puede exponer:
    - Todos los **subdominios**.
    - Direcciones IP internas.
    - Estructura de red.

Por eso, **los servidores DNS deben configurarse adecuadamente** para impedir este tipo de solicitudes no autorizadas.

🔎 Herramientas para analizar DNS

Desde una red local puedes consultar registros DNS con herramientas como:

- nslookup
- dig
- host
- Herramientas web como MXToolbox

Y puedes encontrar:

- Registros **A** (direcciones IPv4)
- Registros **AAAA** (IPv6)
- **MX** (correo)
- **CNAME**, **TXT**, entre otros.

📧 Análisis de registros DNS y configuración de correos institucionales

Aquí podemos observar un ejemplo aplicado a **nuestro propio dominio institucional**. Estos servidores que se muestran corresponden a los **servidores DNS** configurados para el dominio de nuestra institución.

Por ejemplo: gretarunas.edu.pe

Esto es algo que ya hemos venido comentando en clases: **cuando se expone información en línea**, hay herramientas automatizadas que pueden leer todo el contenido de una página web, buscar patrones como “@” y **extraer direcciones de correo electrónico** fácilmente.

⚠️ Riesgo de registrar correos personales en dominios públicos

Un problema más grave es cuando en los **registros DNS públicos** de la institución se incluyen **correos personales** como, por ejemplo:

plaintext

CopyEdit

nombre@gmai.com

En este caso, alguien que consulta el registro DNS público verá que está vinculado un correo de Gmail a un dominio institucional. Este tipo de configuración **es peligrosa y poco profesional**, ya que:

- Expone correos personales al público.
- Puede ser explotado para suplantaciones de identidad o phishing.
- Indica **una mala configuración** en los registros DNS.

🌐 ¿Qué información se puede obtener desde los DNS públicos?

Al consultar los registros DNS de un dominio, **puedes obtener mucha información**, entre ella:

- **Registros MX**: indican **qué servicio de correo electrónico** se está utilizando (Microsoft, Google, etc.).
- **Registros A y AAAA**: muestran las direcciones IP asociadas.
- **Registros CNAME, TXT**: contienen configuraciones adicionales, como SPF, DKIM o subdominios.

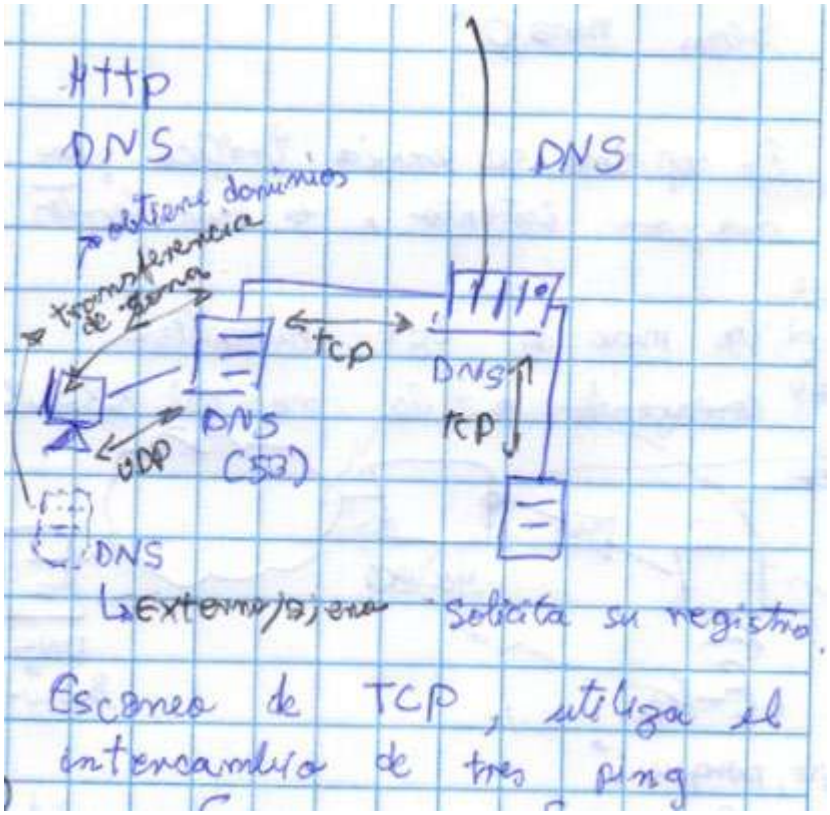
En algunos casos, también es posible extraer:

- **Subdominios**
- **Servidores de correo alternativos**
- **Registros de autenticación y seguridad**

🔒 Transferencia de zona DNS (Zone Transfer)

Una técnica muy usada por atacantes para obtener información del dominio es la **transferencia de zona**. Si un servidor DNS está **mal configurado**, podría permitir que cualquiera descargue **todos sus registros DNS completos**, incluyendo subdominios, configuraciones internas y detalles sensibles.

Esto ocurre cuando no se restringe adecuadamente el acceso a las transferencias de zona (AXFR).



⚙️ **Conflicto entre proveedores de correo: Microsoft vs Google**

En nuestro caso particular, se dio una situación especial:

- Hace años, la institución habilitó cuentas con **Google (G Suite / Workspace)**.
- Paralelamente, también se configuró **Microsoft Office 365** como correo oficial.
- Ambas plataformas usaron el **mismo dominio institucional** (@gretarunas.edu.pe).

Esto generó **conflictos de configuración**, ya que:

- Los registros DNS apuntaban a ambos servicios.
- Las validaciones SPF y MX podían entrar en conflicto.
- Algunos correos enviados desde Gmail no llegaban correctamente o eran redirigidos.

La recomendación técnica en estos casos es:

- **Separar los dominios** (por ejemplo, @gretarunas.edu.pe para Microsoft y @correo.gretarunas.edu.pe para Google).
- O hacer una **configuración avanzada de enrutamiento**, cosa que no se logró implementar completamente.

Como resultado, **el sistema de correo institucional presenta problemas intermitentes** entre plataformas.

📩 Problemas con el envío desde Gmail

Actualmente, si envías un correo desde tu cuenta de Gmail institucional, **puede que no funcione como debería**, debido a esa configuración cruzada entre dos proveedores de correo. Esto afecta:

- Entregabilidad.
- Autenticación.
- Seguridad (SPF, DKIM, DMARC).

¿Te gustaría que compile este contenido como parte de una guía técnica sobre buenas prácticas en DNS y correo institucional? También puedo armarte un infográfico o presentación para capacitar a personal o estudiantes.

📧 Problemas reales por mala configuración de correo institucional

Uno de los casos más serios que se han presentado en nuestra institución fue cuando los correos enviados desde cuentas de **Microsoft Office 365 nunca llegaban a las cuentas configuradas con Google (Gmail)**, o viceversa.

Esto generó incluso conflictos entre estudiantes y docentes. Por ejemplo:

*Un estudiante afirmaba haber enviado todos sus trabajos al correo del profesor, pero este nunca los recibió. El profesor, al no ver evidencia, lo desaprobó. Al revisar el historial, se comprobó que los correos **sí fueron enviados**, pero **nunca llegaron** debido a un conflicto entre plataformas de correo mal configuradas.*

Este problema se origina cuando se configura un **mismo dominio institucional** (como @unas.edu.pe) **tanto para Microsoft como para Google**, lo que genera un **conflicto de registros MX y SPF en los DNS**.

🔍 ¿Qué información se puede obtener desde los registros DNS?

Desde los **registros DNS públicos** se puede conocer:

- Qué proveedor de correo utiliza una institución (Google, Microsoft, Zimbra, etc.).
- Qué tipo de registros están configurados: **MX, A, TXT, SPF, DKIM**.

- En algunos casos, si está mal configurado, se pueden obtener incluso **registros internos y subdominios**.

Pero no todo está disponible a simple vista.

⚠️ ¿Cómo se puede descubrir subdominios?

Aunque los subdominios **no siempre aparecen directamente** en una consulta DNS simple, existen técnicas para descubrirlos:

1. Transferencia de zona (Zone Transfer)

Si el servidor DNS está mal configurado y **no restringe las transferencias de zona**, se puede obtener **todo el contenido de los registros DNS**, incluyendo:

- Subdominios
- Registros MX, TXT, SPF
- Equipos internos

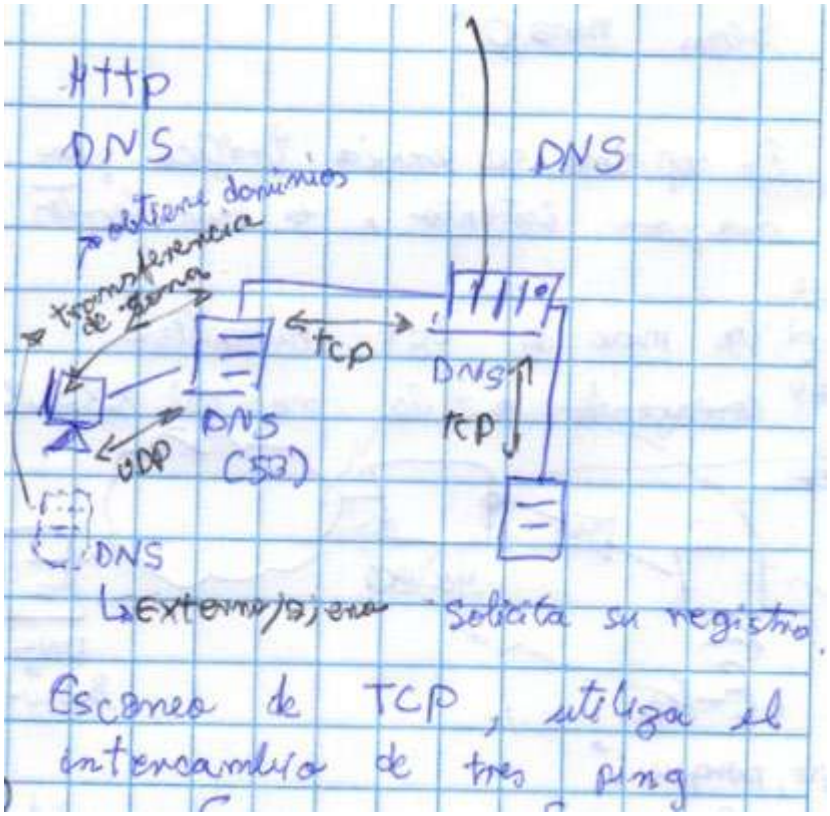
Esto se logra con herramientas como dig, host, dnsenum, etc.

2. Fuerza bruta o diccionario de subdominios

Si la transferencia de zona no está permitida, se puede usar una técnica llamada **"DNS brute force"**:

- Se emplea un **diccionario de palabras comunes** como ftp, mail, campus, biblioteca, api, etc.
- Se hacen consultas automáticas al servidor DNS para verificar si esas palabras están configuradas como subdominios.
- Si alguna respuesta es positiva, se detecta el subdominio.

Herramientas como **Sublist3r, amass, dnsrecon, theHarvester** permiten automatizar este proceso.



🌐 **Ejemplo práctico con el dominio unas.edu.pe**

Si analizamos el dominio institucional unas.edu.pe, y queremos detectar sus subdominios, podemos usar motores de búsqueda o herramientas específicas para observar resultados como:

- trabajo.unas.edu.pe
- gestionambiental.unas.edu.pe
- campus.unas.edu.pe
- biblioteca.unas.edu.pe

Algunos de ellos podrían:

- Tener **sitios web activos**.
- Mostrar **formularios de inicio de sesión (login)**.
- Tener **rutas vulnerables o accesibles públicamente**.

⚠️ **Riesgo: ¿Qué puede hacer un atacante con un subdominio activo?**

Una vez descubierto un subdominio, un atacante puede:

- Comprobar si hay **servicios activos** (FTP, HTTP, correo, etc.).

- Intentar **acceder a paneles de administración expuestos**.
- Buscar **archivos indexados** o **errores de configuración**.
- Realizar **fuzzing o escaneo de directorios**.

Por eso es crucial que las instituciones:

- Monitoreen sus subdominios.
- Cuiden las configuraciones DNS.
- Apliquen medidas de seguridad como firewall y autenticación segura.

✅ Conclusión

- La exposición de registros DNS mal configurados **puede generar filtraciones de información sensible**.
- Problemas de interoperabilidad entre servicios como Gmail y Office pueden provocar **fallos graves de comunicación institucional**.
- Herramientas de descubrimiento de subdominios pueden ser usadas tanto por administradores como por atacantes.
- Tener un dominio bajo control implica **asegurar no solo el correo**, sino también todos los subdominios y servicios asociados.

Además, al identificar subdominios, es posible buscar también **metadatos** dentro de documentos expuestos públicamente, como PDFs, Word o Excel alojados en sitios institucionales. Esta técnica se conoce como **búsqueda de metadatos (metadatalooping)**.

⚠️ ¿Qué podrían hacer los atacantes?

Un atacante, al encontrar subdominios activos, podría:

- **Intentar acceso con credenciales por defecto.**
- Hacer ataques de **fuerza bruta** o uso de **diccionarios**.
- Detectar **vulnerabilidades conocidas** en servicios activos.
- Buscar portales de login expuestos para realizar **ataques de phishing** o **ataques automatizados**.

🔴 **Importante**: Aunque este tipo de exploración puede parecer útil o “interesante” como ejercicio académico, **no debe realizarse sin autorización expresa**.

Incluso si se trata de tu propia universidad, **realizar pruebas sin permiso es antiético y puede ser ilegal**.

🌐 TCP, UDP y el fundamento del escaneo de puertos

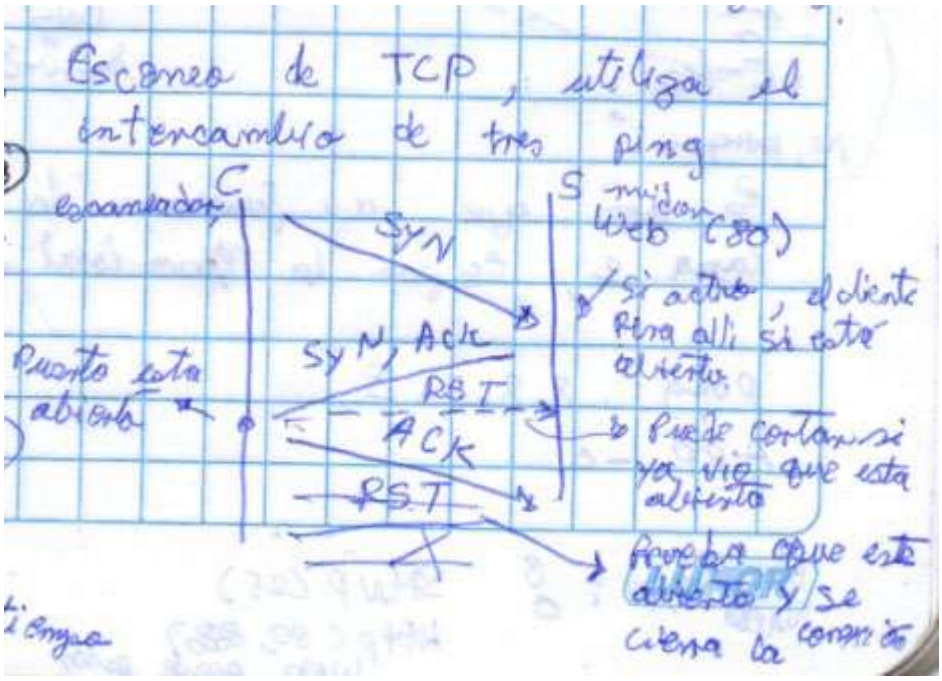
Volviendo al enfoque técnico: para comprender cómo funcionan los escaneos en redes, es esencial tener claro cómo operan los protocolos **TCP** y **UDP**.

🔷 TCP: Conexión confiable

TCP utiliza un **proceso de tres pasos** conocido como el **"three-way handshake"** para establecer una conexión entre cliente y servidor.

1. **SYN**: El cliente envía un paquete de sincronización al servidor (SYN).
2. **SYN-ACK**: El servidor responde con un paquete de sincronización + confirmación (SYN-ACK) si está disponible.
3. **ACK**: El cliente responde con un paquete de confirmación (ACK) y se establece la conexión.

Una vez completado este proceso, se pueden empezar a **intercambiar datos** de forma segura y ordenada.



🤔 ¿Cómo se relaciona esto con la seguridad?

El protocolo TCP es también la **base de la mayoría de los escaneos de puertos** que se realizan para detectar servicios activos en un servidor.

Existen distintas **técnicas de escaneo TCP**:

- **Full connect scan** (conexión completa)
- **Half-open scan** (también conocido como SYN scan)
- **Stealth scan**
- **NULL, FIN, XMAS scan**, entre otros

Cada uno tiene un comportamiento distinto y **puede evadir ciertos sistemas de detección**, por eso es fundamental conocer **cómo funciona TCP internamente** para entender las implicancias de seguridad.

✅ Conclusión

- Aunque la búsqueda de subdominios y metadatos es técnicamente accesible, **no debe hacerse sin autorización**.
- Los **escaneos de red**, especialmente los que usan TCP, son parte esencial del reconocimiento en ciberseguridad.
- Entender el **three-way handshake** es clave para saber cómo operan tanto las conexiones legítimas como los ataques.
- Toda actividad relacionada con análisis de red o pruebas de seguridad **debe realizarse con responsabilidad, ética y permisos formales.**

🔍 ¿Qué ocurre durante un escaneo de puertos?

Supongamos que un cliente desea conectarse a un servidor web. Para ello, intenta conectarse al **puerto 80**, que es el puerto estándar para servicios HTTP.

El cliente envía un **mensaje de sincronización (SYN)** al servidor, preguntando en esencia:

“¿Tienes el servicio del puerto 80 disponible? ¿Puedo conectarme contigo?”

Si el servidor **tiene ese puerto abierto**, responderá con un **SYN-ACK**, indicando que sí, el servicio está activo y disponible para conectarse. En ese momento, el cliente ya **sabe que el puerto 80 está abierto**.

Luego, el cliente responde con un **ACK** para completar el **three-way handshake** y así se inicia una conexión TCP normal.

🕵️ ¿Qué pasa durante un escaneo de puertos?

Un **escaneo de puertos** no busca establecer una conexión real para intercambiar datos, sino **solo detectar qué puertos están abiertos** en un sistema remoto. Es decir:

"¿Qué puertas están abiertas en este sistema que podrían usarse para entrar?"

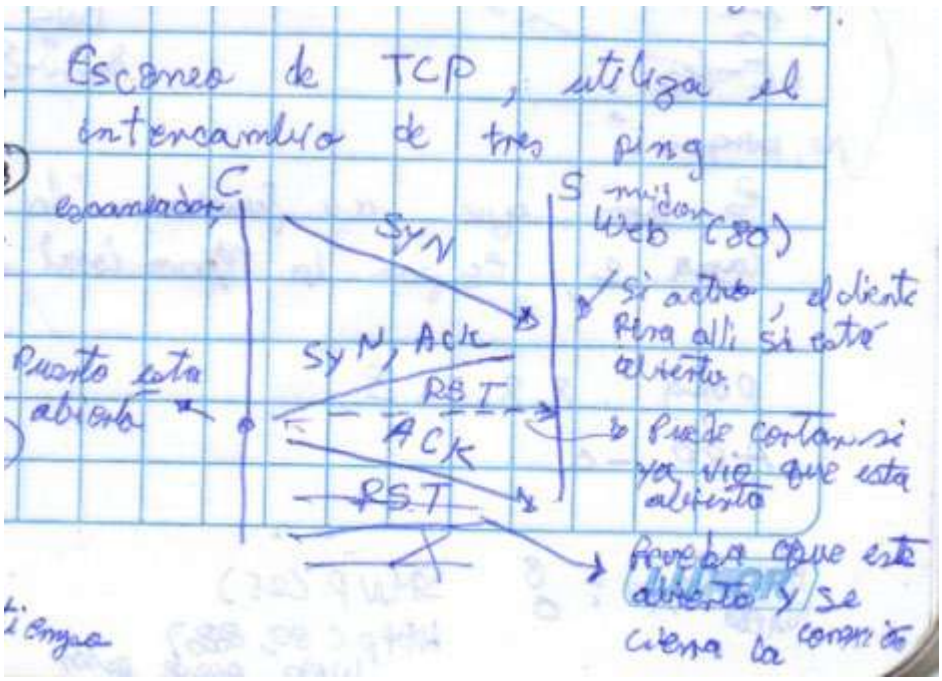
Para eso, un escáner (como **Nmap**) simula el inicio de una conexión enviando un paquete **SYN** a varios puertos, desde el **1 hasta el 65535** si se desea hacer un escaneo completo.

🔓 ¿Cómo detecta si un puerto está abierto?

El procedimiento es muy similar a lo explicado anteriormente con TCP:

1. El escáner envía un **SYN** al puerto 80 (por ejemplo).
2. Si el servidor **responde con un SYN-ACK**, significa que **el puerto está abierto**.
3. El escáner puede entonces **cancelar la conexión** enviando un **RST (reset)** en lugar de completar el handshake (para evitar ser detectado como una conexión real).

Este tipo de escaneo se llama **TCP SYN scan** (también conocido como *half-open scan*), y es muy utilizado porque es rápido y discreto.



🧠 Ejemplo visual (analógico)

Imagina que el sistema tiene muchas puertas (puertos), y el escáner actúa como una persona que **toca cada puerta del edificio**, una por una:

- Si una puerta **se abre** o alguien responde: esa “puerta” (puerto) **está abierta**.
- Si nadie responde o se rechaza la llamada, se interpreta como **cerrado o filtrado**.

❓ Pregunta de reflexión para estudiantes

¿Cómo cree un atacante o un analista de seguridad que puede saber si un puerto específico (como el 100, el 2020 o el 8080) está abierto?

→ La respuesta es: **intentando conectarse**. Y dependiendo de la **respuesta que reciba**, el escáner determina el estado del puerto:

- **Abierto**
- **Cerrado**
- **Filtrado** (cuando un firewall o IDS bloquea la respuesta)

✅ Conclusión

- Un escaneo de puertos **simula conexiones reales** para detectar servicios activos.
- Se basa en el funcionamiento del protocolo **TCP** y su **three-way handshake**.
- Existen **múltiples tipos de escaneo**, pero todos buscan **identificar puntos de entrada** potenciales en un sistema.

🔍 ¿Cómo funciona el escaneo de puertos TCP?

Cuando un escáner quiere saber si un **puerto específico** está abierto (por ejemplo, el **2040**), lo que hace es **simular una solicitud de conexión**.

Ejemplo:

El escáner envía un paquete **SYN** al puerto 2040 del servidor diciendo:

“¿Puedo conectarme contigo en el puerto 2040?”

🟢 Si el puerto está abierto, el servidor responde con:

- Un paquete **SYN-ACK** (sincronización + confirmación).
- Esto significa que el puerto está escuchando conexiones y el escáner **registra que el puerto 2040 está abierto**.

🔴 Si el puerto está cerrado, el servidor responde con:

- Un paquete **RST (reset)**, indicando:

“Ese puerto está cerrado, no acepto conexiones”.

En ese momento, el escáner **marca el puerto como cerrado**.

⚠️ ¿Y si no hay respuesta?

Cuando el escáner **no recibe ninguna respuesta**, puede significar dos cosas:

1. El dispositivo está **apagado**.
2. Un **firewall está filtrando** los paquetes y **bloqueando la respuesta**.

En estos casos, el escáner lo clasifica como: **puerto filtrado**.

Esto es común cuando se usan firewalls como **iptables**, **pfSense** o **firewalld**, que descartan silenciosamente los paquetes entrantes no autorizados.

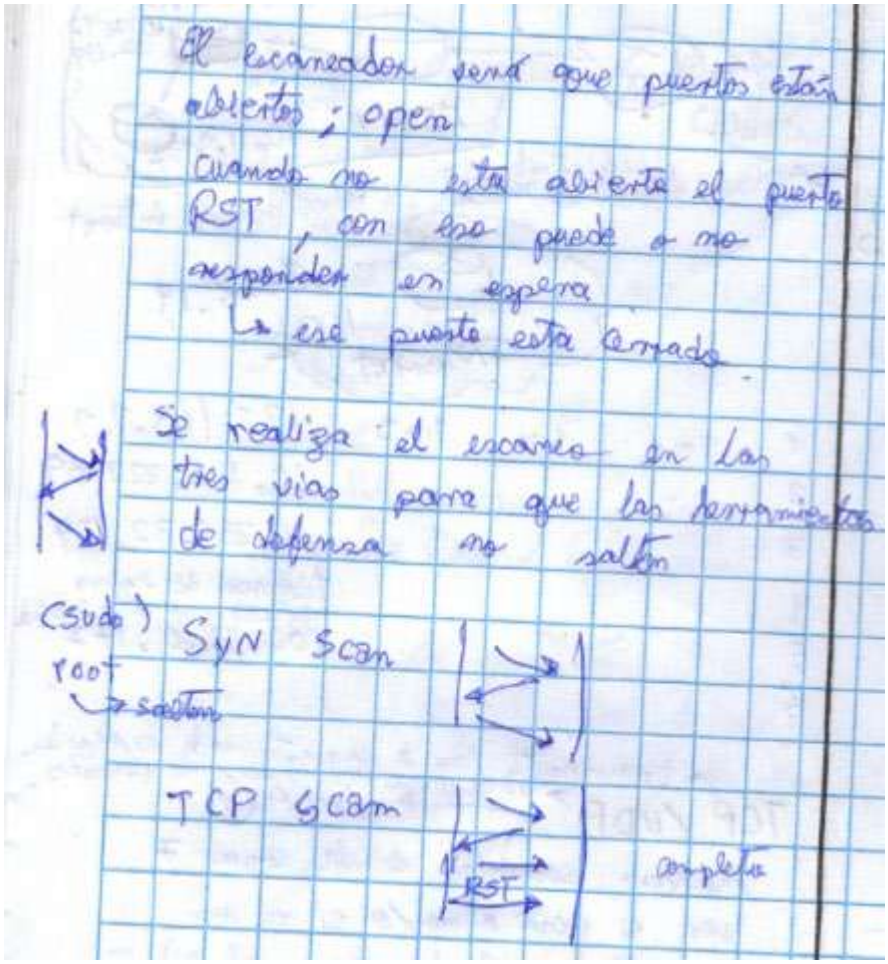
🧪 ¿Qué hace un escáner tras detectar un puerto abierto?

Supongamos que el escáner prueba el **puerto 8080**, y el servidor responde afirmativamente (SYN-ACK), lo que indica que hay un **servicio web alternativo** activo.

Pero el escáner **no completa la conexión**, es decir, **no envía el paquete ACK final**.

¿Por qué? Porque no necesita comunicarse realmente con el servicio, solo **necesita saber que está abierto**.

En este tipo de escaneo (conocido como **SYN scan** o *half-open scan*), el escáner **aborta la conexión enviando un paquete RST (reset)** o simplemente **no responde más**.



🧠 ¿Qué ocurre si el escáner no envía el reset?

Si el escáner **no cancela la conexión**, el servidor **queda esperando** el ACK del cliente para completar la conexión.

Esto puede provocar:

- Que el puerto quede **temporalmente ocupado**.
- Que aparezca una entrada en la tabla de conexiones del servidor (visible con netstat, ss, etc.), indicando un estado como SYN_RECEIVED.

Por eso, en prácticas profesionales o éticas, es recomendable **cerrar correctamente** la conexión simulada para no alterar el comportamiento del sistema.

✅ Conclusión

- El escaneo de puertos se basa en la respuesta al envío de paquetes TCP:
    - **SYN-ACK** → puerto **abierto**

    - **RST** → puerto **cerrado**
    - **Sin respuesta** → puerto **filtrado**
- El escaneo puede ser activo o pasivo, y puede generar evidencia en los logs del servidor.
- Comprender este proceso es esencial tanto para **defender una red** como para realizar **pruebas de penetración éticas**.

🛠️ ¿Qué es un escáner de puertos?

Un **escáner de puertos** es una herramienta que permite detectar qu puertos están **abiertos, cerrados o filtrados** en un sistema.
Pero más allá de usar herramientas listas como Nmap, también puedes **crear tu propio escáner** con:

- Python (usando sockets)
- Bash (con nc, telnet o timeout)
- PowerShell o scripts en Windows

Lo importante es que **el escáner en esencia solo necesita probar si un puerto responde o no a una conexión**.

⚙️ ¿Qué sucede si un escáner no cierra la conexión?

Durante un escaneo **TCP tipo SYN**, lo usual es:

1. El escáner envía un **SYN** al puerto destino.
2. Si el puerto está **abierto**, el servidor responde con un **SYN-ACK**.
3. En lugar de completar la conexión con un ACK, **el escáner envía un RST (reset)** para cortar la conexión y **no dejar huella** en la tabla de conexiones del servidor.

🔴 ¿Qué pasa si el escáner *no* envía el RST?

Si no se envía el reset, la conexión queda "incompleta", y el servidor mantiene el estado SYN_RECEIVED en su tabla de conexiones.

Esto tiene consecuencias:

- Puede **ocupar recursos del servidor** (sobrecargarlo).
- Puede ser **detectado fácilmente** por herramientas de monitoreo.

- Puede dejar trazas en los **logs del sistema**.

✅ **Comportamiento ideal de un escáner eficiente**

- **Detectar puertos abiertos.**
- Enviar un **RST inmediatamente** al recibir el SYN-ACK.
- Continuar con e siguiente puerto.

Esto:

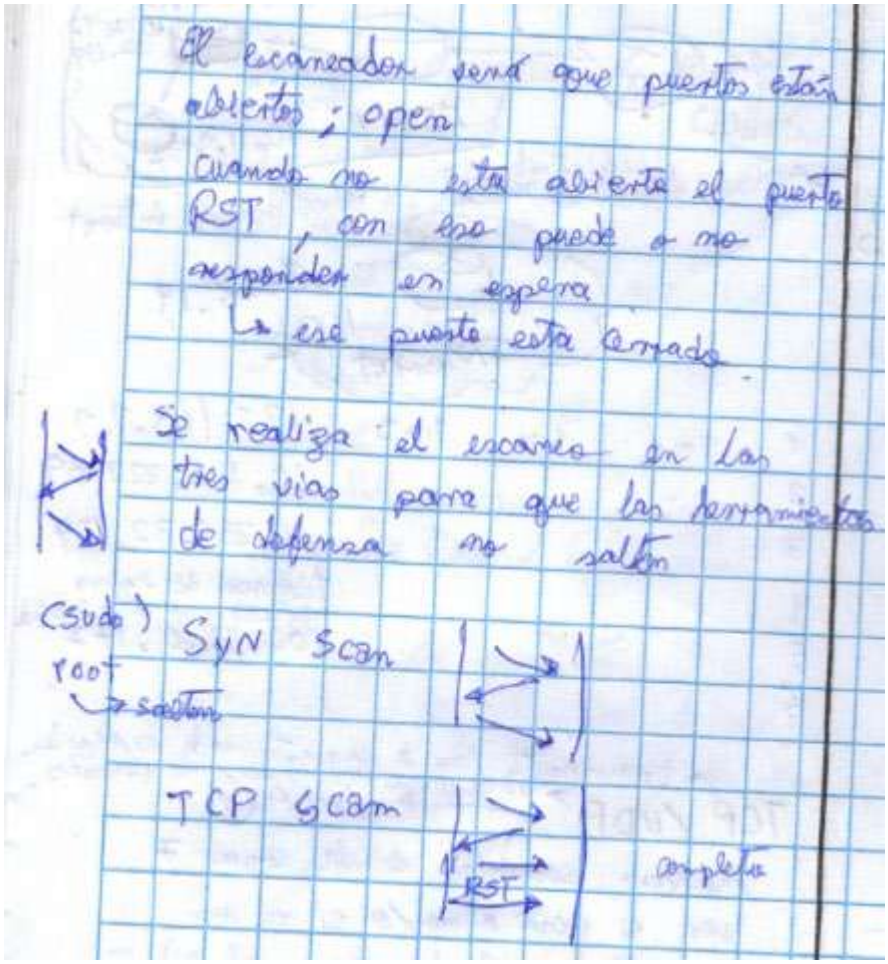
- Libera el recurso en el servidor.
- No termina el *three-way handshake*.
- Minimiza el riesgo de detección en redes bien monitoreadas.

⚠️ **¿Por qué es importante saber esto?**

Porque muchas herramientas de escaneo, como **Nmap**, permiten **ajustar el tipo de escaneo y su comportamiento**, por ejemplo:

- -sS: SYN scan (half-open, común y discreto).
- -sT: Full TCP connect (más ruidoso y detectable).
- --scan-delay, --max-rate: Para escaneos lentos y menos sospechosos.

Si no conoces **cómo funciona tu herramienta por defecto**, puedes estar generando un **tráfico fácilmente detectable**.
En seguridad ofensiva y defensiva, **entender el comportamiento por debajo es fundamental**.



🧠 **Conclusión**

- Los escáneres de puertos pueden ser simples scripts o poentes suites completas.
- El comportamiento predeterminado de muchos escáneres es **detectar un puerto abierto y enviar un RST**.
- **No cerrar adecuadamente las conexiones** puede hacer que un escaneo sea **más ruidoso y visible**.
- Conocer **cómo funciona la herramienta y cómo modificar su comportamiento** es clave para evitar detección y optimizar los análisis.

🕵️ **Escaneos TCP: Comportamiento, detección y buenas prácticas**

Realizar un escaneo de puertos es algo común en pruebas de seguridad, pero **el cómo se hace marca la diferencia** entre pasar desapercibido y ser detectado o incluso baneado.

🔄 **Escaneo TCP "normal" (full connect scan)**

Este tipo de escaneo:

1. Realiza el **three-way handshake completo** (SYN → SYN-ACK → ACK).
2. Luego, **envía un RST (reset)** para cerrar la conexión.

🟢 **Ventaja**: El tráfico simula una conexión legítima (como si un navegador o aplicación se conectara y luego se desconectara), por lo que puede pasar **más desapercibido**.

🔴 **Desventaja**: Es **más lento** y genera **más trazas de conexión** en los registros del sistema destino (como logs o firewalls).

🚨 Escaneo TCP SYN (half-open scan)

Este escaneo envía:

- Solo el **SYN**.
- Espera un **SYN-ACK**.
- Luego envía un **RST** sin completar la conexión.

Este método es más rápido y **no deja conexiones activas**, pero:

⚠️ **¡Cuidado!**: Si haces esto **desde una red cloud (como una VPS)** o contra una red protegida, los sistemas de detección (IDS/IPS) lo reconocerán como **comportamiento hostil o sospechoso**.
→ Resultado: **te bloquean** o **banean tu IP** automáticamente.

🧠 ¿Por qué pasa esto?

Porque los sistemas defensivos como **Snort, Suricata, Fail2Ban, WAFs, etc.**, están configurados para detectar patrones típicos de:

- Escaneos rápidos
- Peticiones SYN sin completar handshake
- Conexiones simultáneas a muchos puertos

Y al identificar ese comportamiento, se asume que **es un ataque de reconocimiento**.

📌 ¿Cómo evitar ser detectado?

Para reducir la probabilidad de detección:

- Utiliza **escaneos completos (TCP connect)** cuando estés autorizado y busques precisión.

- Usa **flags personalizados** o **modos sigilosos** en herramientas como Nmap:
    - -sS (SYN scan)
    - -sT (TCP connect scan)
    - --scan-delay y --max-rate para limitar velocidad
- Siempre **como root**, entiendes lo que estás haciendo. Algunas opciones de Nmap **requieren privilegios root** para funcionar correctamente (por ejemplo, -sS).

🧪 **Ejemplo práctico con Nmap**

- Si ejecutas nmap como **usuario normal**:
    - Hará un **escaneo TCP completo (three-way handshake)** (-sT).
- Si ejecutas nmap como **root**:
    - Permitirá hacer un **escaneo SYN (half-open)** (-sS), que puede ser más sigiloso, pero también **más detectable si mal usado**.

⚠️ Escanear sin saber el modo en que estás actuando puede terminar con tu IP **baneada de toda una red**, especialmente si estás desde VPS.

🧠 Escaneo de puertos en entornos reales: buenas prácticas y riesgos

El escaneo de puertos es una técnica muy común y poderosa, pero también **altamente sensible** en entornos reales. Todo depende de **cómo se hace, desde dónde se hace, y con qué permisos**.

🛠️ ¿La herramienta importa?

Sí, pero **también importa cómo la usas**. Por ejemplo:

- Si usas herramientas como **Nmap** o **Zmap**:
    - El tipo de escaneo **puede cambiar según los permisos** (root vs. usuario normal).
    - Escaneos como -sS (SYN scan) solo funcionan **con privilegios de administrador/root**.

- Si no sabes qué tipo de escaneo estás ejecutando, **podrías terminar generando tráfico sospechoso sin quererlo**.

⚠️ ¿Qué pasa si haces un escaneo sin cuidado?

En un entorno real:

- Podrías ser **detectado rápidamente** por sistemas de seguridad.
- **Tu IP podría ser bloqueada o baneada** (especialmente si estás en la nube).
- Si el sistema objetivo pertenece a una entidad crítica o privada (como universidades, instituciones del Estado o empresas), podrías ser **reportado, rastreado o incluso sancionado**.

Ejemplo: Si intentas escanear los sistemas del **UNAS** o del **IFIP** sin autorización, podrían detectar tu actividad como **maliciosa**, incluso si lo hiciste con fines de aprendizaje.

🧪 ¿Dónde y cómo practicar?

- No realices escaneos en **infraestructura real o de producción** sin permisos.
- Utiliza:
  - **Entornos de laboratorio locales**
  - Máquinas virtuales
  - Plataformas como **TryHackMe**, **Hack The Box**, **Metasploitable**, etc.
- Puedes grabar la práctica con herramientas como **Wireshark** para analizar el tráfico en detalle.

🌐 Cualquier IP activa puede tener puertos abiertos

Todo dispositivo con IP activa (computadora, impresora, servidor, router) **tiene puertos abiertos o cerrados**.

¿Por qué? Porque necesita comunicarse con otros servicios o dispositivos.

Por eso, **el escaneo permite detectar posibles servicios en ejecución**, como por ejemplo:

Tipo de servicio	Puerto estándar	Protocolo
Web (HTTP)	80	TCP
Web seguro (HTTPS)	443	TCP
Web alternativo	8080	TCP

Escritorio remoto (RDP)	3389	TCP
FTP	21	TCP
SMTP (correo saliente)	25	TCP
MySQL	3306	TCP
MS SQL Server	1433	TCP

⚠️ *Un servicio también puede estar corriendo en un puerto no estándar para "ocultarse", por eso es importante entender el contexto del escaneo.*

📚 Tema adicional: números de secuencia TCP

Más allá del escaneo, es importante conocer **cómo se establecen las conexiones TCP**, especialmente mediante:

- El uso de **números de secuencia** (SEQ)
- El **reconocimiento de respuestas** (ACK)

Estos valores permiten a TCP controlar el flujo de datos y garantizar que los segmentos lleguen completos y en orden.

👉 Este tema forma parte del **nivel intermedio/avanzado** del análisis de red y se relaciona con:

- Análisis con **Wireshark**
- **Spoofing** de paquetes
- Detección de **anomalías** o ataques

✅ Conclusión

- El escaneo es útil y legítimo **siempre que se haga de forma ética y controlada**.
- El desconocimiento de cómo funciona la herramienta puede **generarte consecuencias reales**.
- Toda práctica debe ser en **entornos preparados para ello**.
- Conocer los **puertos y protocolos comunes** es base para entender la comunicación en redes.

🌐 Escaneo de puertos UDP: características y dificultades

A diferencia del protocolo TCP, **UDP no tiene un sistema de conexión basado en el three-way handshake**, por lo tanto:

🔶 **No hay confirmación de recepción.**

🔶 **No se envía ningún "SYN", "ACK" ni "RST".**

Esto hace que **el escaneo de puertos UDP sea mucho más incierto**. ¿Por qué?

Porque **si no hay respuesta**, **no sabes con certeza** si el puerto:

- Está cerrado
- Está filtrado
- O simplemente **el servicio no respondió**

🧪 ¿Cómo se realiza el escaneo UDP?

- Se envían paquetes UDP a un puerto.
- Si el puerto está **cerrado**, a veces se recibe un **ICMP "Port Unreachable"**.
- Si está **abierto**, puede que **no haya respuesta en absoluto** (por diseño).
- Por eso se dice que **es un escaneo lento, impreciso y con alta tasa de falsos negativos**.

🔍 Puertos UDP comunes que pueden responder

Algunos puertos UDP suelen responder si el servicio está activo:

Servicio	Puerto UDP
DNS	53
RPCbind (Remote Procedure Call)	111
NFS (Network File System)	2049
SSDP (UPnP discovery)	1900

Estos servicios pueden ser útiles para:

- Detectar **compartición de archivos o recursos**.
- Montar recursos remotos desde sistemas como **NFS en entornos Unix/Linux**.

Ejemplo: Si el puerto 111 está abierto, podría indicar que hay **archivos compartidos en red**, y un atacante podría intentar montarlos localmente y **acceder a información sin autenticación** si está mal configurado.

⏱️ ¿Qué problemas presenta el escaneo UDP?

- **Alto tiempo de espera** por puerto (timeouts).
- Alta probabilidad de **no obtener respuesta** incluso si el servicio está activo.
- Puede ser **bloqueado fácilmente por firewalls**.
- **Mucho más lento** que un escaneo TCP.

Por eso, el escaneo UDP debe hacerse con:

- **Múltiples intentos**.
- **Tiempo de espera adecuado** (ej. --max-retries, --host-timeout en Nmap).
- Paciencia.

📚 ¿Cuándo lo veremos a fondo?

Este tema se explora con mayor detalle en la **Unidad 3**, donde abordaremos:

- **Detección de servicios**
- **Identificación de versiones**
- **Reconocimiento de vulnerabilidades**
- **Escaneos más avanzados (scripts, NSE, fingerprinting)**

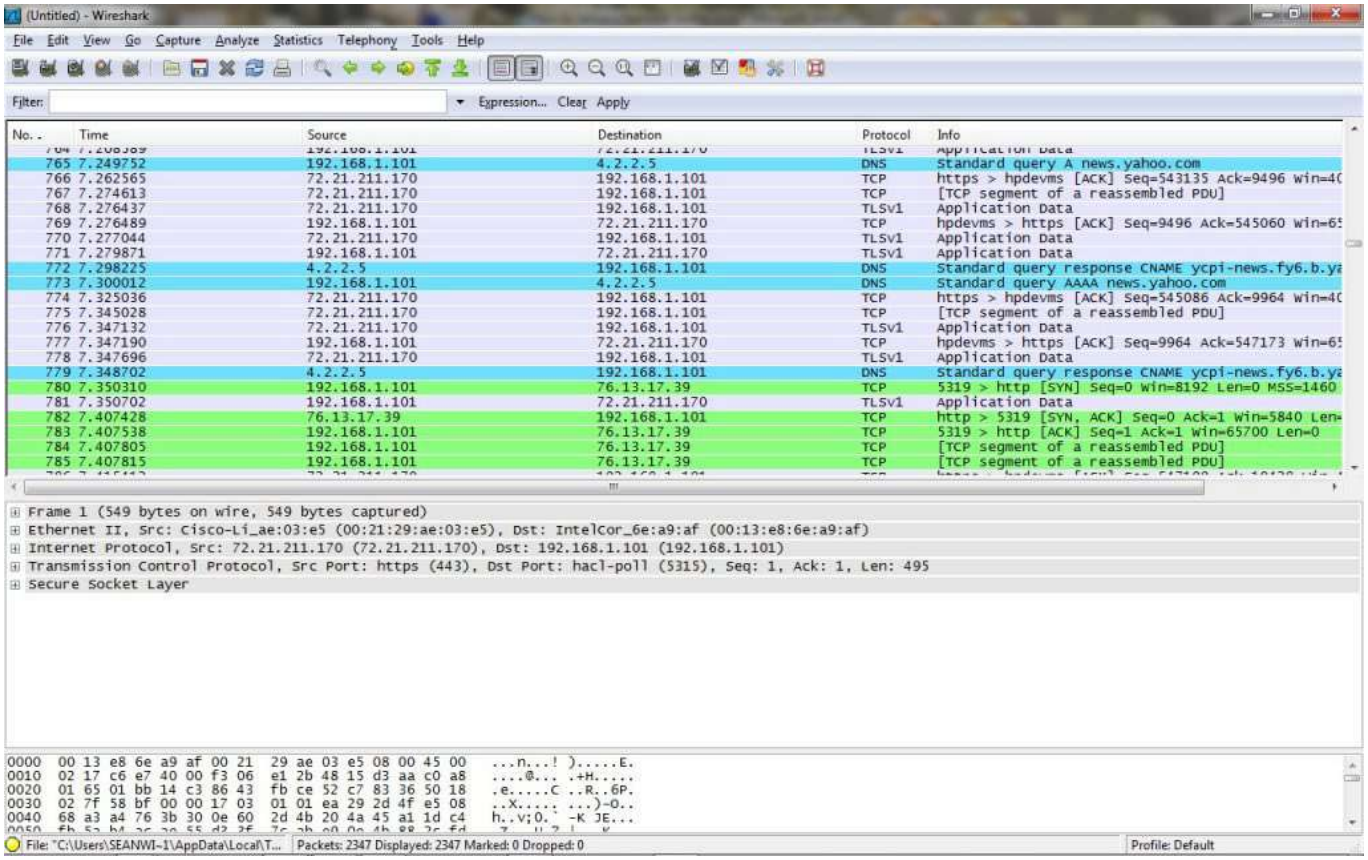
🩹 Sniffer: herramienta esencial para analizar tráfico

🛠️ ¿Qué es un sniffer?

Un **sniffer** (analizador de tráfico) es una herramienta que:

- **Captura paquetes que entran y salen** de una interfaz de red.
- Permite analizar:
    - Protocolos utilizados
    - Cabeceras IP, TCP/UDP
    - Cadenas de datos (incluso contraseñas en texto plano)

Ejemplos de sniffers: **Wireshark**, **tcpdump**, **TShark**, **Ettercap**



🤔 ¿Por qué no siempre vemos el tráfico de toda la red?

Un sniffer **solo ve el tráfico que pasa por su interfaz de red**, por eso:

- En una red **conmutada (switch)**, cada PC solo recibe el tráfico destinado a ella.
- Por eso **no puedes ver la comunicación entre otras dos máquinas** directamente, salvo que:
    - Estés en modo **promiscuo** + tengas configuraciones especiales (como port mirroring).
    - Uses técnicas como **ARP spoofing** o **MiTM** (solo con fines educativos y autorizados).

✅ Conclusión general

- El escaneo UDP es más difícil que el TCP por la falta de confirmación.
- Algunos puertos como 53, 111, 1900 pueden revelar información útil si están abiertos.
- Usar herramientas como **Nmap** con las opciones adecuadas ayuda a minimizar los falsos negativos.

- Los **sniffers** permiten capturar y analizar el tráfico de red, pero no todo el tráfico es visible por defecto.
-

🧠 ¿Por qué no veo todo el tráfico en mi sniffer?

Esta es una pregunta fundamental al usar herramientas como **Wireshark** o **tcpdump**.

Cuando un sniffer está capturando tráfico, **no verá la comunicación entre otros dos equipos** de la red, **a menos que seas uno de los extremos** de esa conversación.

🔍 ¿Por qué sucede esto?

Esto ocurre por **cómo funcionan los dispositivos de capa 2**, en particular **los switches y puntos de acceso Wi-Fi**:

- Un **switch** solo reenvía los paquetes **al puerto donde está conectado el destinatario**.
- No inunda el tráfico a todos los puertos como lo haría un **hub**.
- Esto se llama **microsegmentación** o **aislamiento por puerto**, y es una medida de eficiencia y seguridad.

Por lo tanto:

👉 Si estás en un puerto diferente al de los dispositivos que se comunican, no verás su tráfico.

📡 ¿Qué tipo de tráfico sí puedes ver?

Con tu sniffer podrás ver:

Tipo de tráfico	¿Se captura?	Motivo
Tráfico dirigido a tu IP	✅ Sí	Porque está destinado a ti directamente.
Tráfico broadcast (ARP, DHCP, etc.)	✅ Sí	Se envía a todos los nodos de la red.
Tráfico multicast	✅ Parcial	Dependiendo del grupo y configuración del switch.

Tipo de tráfico	¿Se captura?	Motivo
Tráfico entre otros dos equipos	❌ No	Es enviado directamente entre ellos y **no te llega**.

🎯 Análisis de paquetes capturados: direcciones MAC

Cuando capturas un paquete, puedes analizar:

- **Capa 2 (Ethernet):**
  - Dirección **MAC origen**
  - Dirección **MAC destino**
- **Capa 3 (IP):**
  - Dirección IP origen y destino
- **Capa 4 (TCP/UDP):**
  - Puerto origen y destino

Ejemplo de análisis:

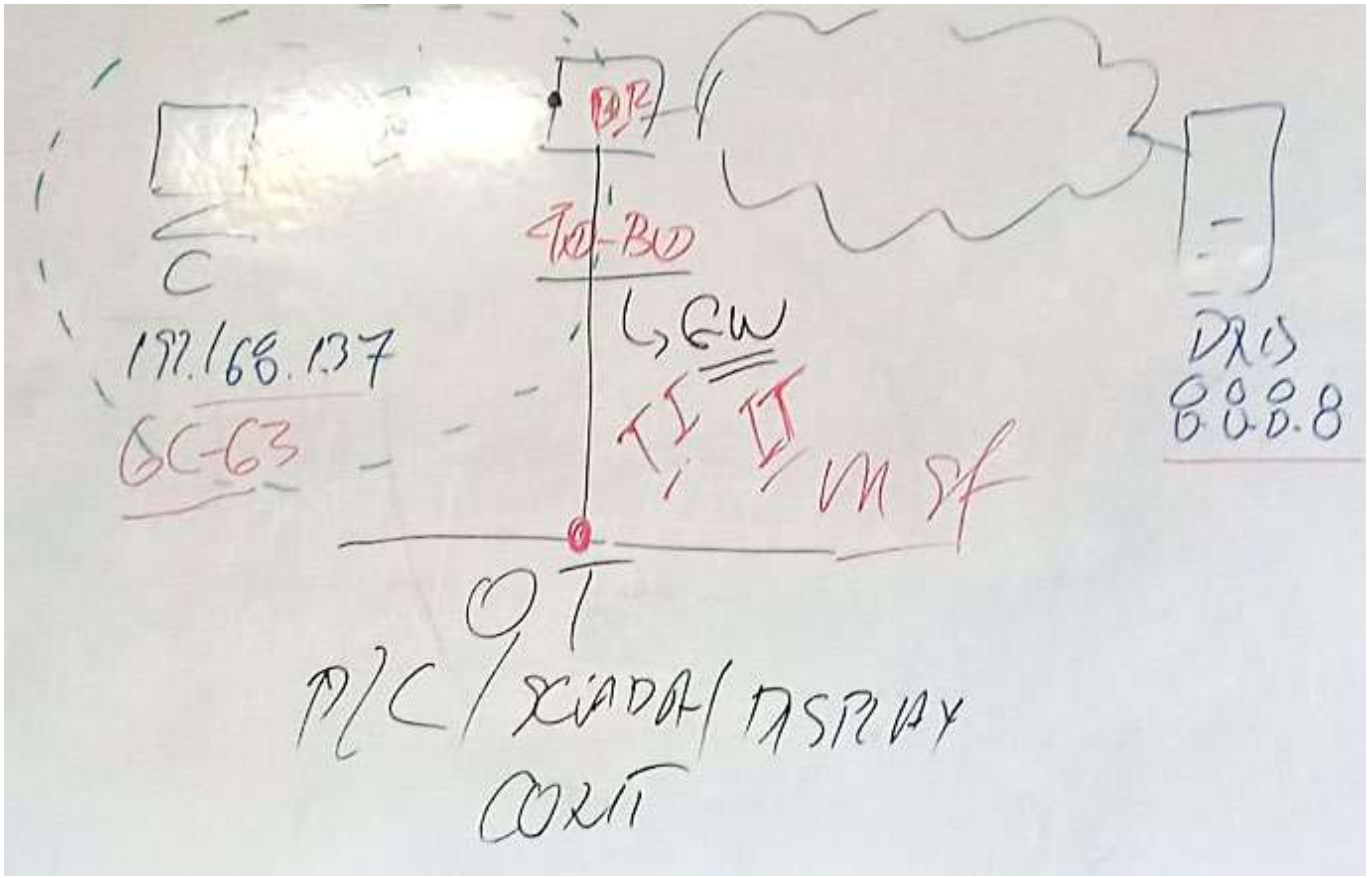
Supongamos que ves un paquete con esta información:

- **MAC origen**: 6C:63:XX:XX:XX:XX
- **MAC destino**: FF:FF:FF:FF:FF:FF (broadcast)

Entonces puedes deducir:

- El tráfico fue generado por el dispositivo con MAC 6C:63..., probablemente tu laptop o tu adaptador Wi-Fi.
- La dirección MAC destino indica que el mensaje fue **enviado a todos los dispositivos de la red** (broadcast).

Puedes incluso buscar a qué fabricante pertenece una MAC en sitios como macvendors.com.



🤔 **¿Cómo saber qué dispositivo es cuál?**

Si ves una **IP y una MAC**, puedes hacer una relación usando herramientas como:

- arp -a (en Windows o Linux)
- Tabla ARP en routers o switches
- Análisis cruzado con paquetes ARP

Así puedes saber, por ejemplo, que:

- La IP 192.168.1.88 puede estar asociada a una MAC que pertenece al **gateway** o **servidor DNS**.
- La IP 192.168.1.109 puede ser tu propia estación de trabajo.

✅ **Conclusión**

- **No verás el tráfico ajeno** en redes conmutadas (switches) debido a la **segmentación de la capa 2**.
- Puedes ver tu tráfico y el tráfico broadcast/multicast.
- Puedes usar sniffers para analizar en detalle las direcciones MAC y protocolos involucrados.

- Para ver más allá (por ejemplo, capturar tráfico entre otros dos hosts), se requerirían técnicas como:
    - **Port mirroring (en switches administrables)**
    - **ARP spoofing/MiTM** (solo en entornos de laboratorio y con permisos)

📣 ¿Por qué no puedo ver la dirección MAC de un servidor externo?

Esta es una de las dudas más frecuentes al analizar tráfico con un sniffer como **Wireshark**. Veamos por qué:

⚙️ Capa 2 (enlace de datos) y el alcance local de las direcciones MAC

Las **direcciones MAC** pertenecen a la **capa 2 del modelo OSI** (capa de enlace de datos), y tienen una **característica muy importante**:

✅ **Las direcciones MAC solo son válidas dentro de la red local (LAN).**

🔎 Entonces, ¿por qué no veo la MAC del servidor de Google?

Porque **Google está fuera de tu red local**, y las direcciones MAC **no cruzan routers**.

📌 Lo que realmente sucede:

- Tu equipo genera un paquete dirigido a la IP pública de Google (ej. 8.8.8.8).
- El paquete es encapsulado con:
    - **Tu MAC como origen**.
    - **La MAC de tu gateway o router local como destino.**
- El router recibe el paquete, **reemplaza las MACs** por las correspondientes a su siguiente salto (otro router), y así sucesivamente.

Por eso, aunque te conectes con 8.8.8.8, **la dirección MAC destino siempre será la del router de tu red local**, no la de Google.

🧠 ¿Y el protocolo ARP?

El protocolo **ARP (Address Resolution Protocol) solo funciona dentro de la red local** y su función es:

🔄 Traducir una dirección IP a su correspondiente dirección MAC **en la misma red**.

Por eso:

- Si estás en 192.168.1.109 y quieres hablar con 192.168.1.1, tu equipo usa ARP para obtener la MAC del gateway.
- Pero si quieres comunicarte con 8.8.8.8, **no puedes hacer ARP directo** a Google, porque está **fuera de tu dominio de broadcast**.
-

🖧 **Entonces, ¿qué dirección MAC ves en el sniffer?**

Verás algo así:

- **MAC de origen**: tu dispositivo (ej. 6C:63:XX:XX:XX:XX)
- **MAC de destino**: la interfaz del router local o AP WiFi (ej. 40:00:XX:XX:XX:XX)

Esto confirma que:

- Estás conectado vía WiFi o cable a un **dispositivo intermediario** (como un router o antena).
- El tráfico IP que capturas puede ser externo, pero **las MACs siempre serán locales.**

✅ **Conclusión**

- La **capa 2 (direcciones MAC)** tiene **alcance limitado a la red local**.
- **No se puede conocer la MAC** de un servidor fuera de tu red (como Google o cualquier otro) **porque está más allá del alcance de capa 2.**
- **ARP** solo se utiliza para obtener direcciones MAC de dispositivos **en la misma subred**.
- Si ves una MAC como destino, **pertenece al gateway**, no al servidor remoto.

🧠 **¿Cómo identificar direcciones MAC e IP en análisis de red?**

Cuando haces un **ping (ICMP)**, puedes usar herramientas como **arp -a** o **Wireshark** para analizar:

1. La IP de destino (por ejemplo, tu puerta de enlace o Google).
2. La **dirección MAC real del dispositivo local** con el que tu equipo está comunicándose en ese momento.

📍 Ejemplo práctico: identificar la MAC del gateway

- Haces un ping a una IP pública (ej. 8.8.8.8).
- Aunque el destino es Google, **la MAC de destino será la del router/gateway local**.
- Ejecutas el comando arp -a y ves:
    - IP: 192.168.1.1 (puerta de enlace)
    - MAC: 40:B0:... → Corresponde al router (ej. Movistar).

Esto confirma que **tu equipo no conoce ni necesita la MAC de Google**, solo la de su salida hacia internet: el router.

⚙️ Protocolo ICMP: tipos de mensaje

Cuando haces un ping, se utiliza el **protocolo ICMP (Internet Control Message Protocol)**. Este no trabaja con MAC directamente, pero se comporta así:

Acción	Tipo ICMP	Código	Descripción
Echo Request	8	0	Petición de ping
Echo Reply	0	0	Respuesta del ping

🧪 En Wireshark puedes ver claramente estos valores en los paquetes ICMP capturados.

📡 ¿Qué más puedes analizar con Wireshark?

1. Consultas DNS (puerto 53)

- Puedes aplicar el filtro: dns o udp.port == 53.
- Verás las consultas tipo **query** (petición) y sus respuestas.
- Por ejemplo:
    - Consulta: teams.microsoft.com
    - Servidor DNS: 192.168.1.225 (puede ser tu proveedor o un servidor propio)
    - Respuesta: IPs como 40.90.142.200, etc.

2. Análisis de tráfico en redes compartidas

Si configuras tu laptop como **puente (bridge)** o **hotspot compartido**, entonces:

- Todo el tráfico que **pase por tu máquina** (entrada y salida) **puede ser capturado**.

- Esto permite capturar tráfico desde y hacia otros dispositivos conectados a través tuyo.

Por eso, **un atacante que logra posicionarse como intermediario (MitM)** puede observar tráfico de otros si logra redirigirlo.

✅ Conclusiones clave

- Las **direcciones MAC solo existen en la red local**.
- Hacer ping te permite detectar la **MAC del gateway** (no del servidor externo).
- Puedes usar arp -a para consultar la tabla ARP y ver qué MAC se asocia a qué IP.
- **ICMP y DNS** son protocolos esenciales para diagnóstico y análisis de red, y se pueden observar detalladamente en Wireshark.
- Si tu laptop **actúa como puente de red**, puedes capturar **todo el tráfico que la atraviesa**.

🌐 Análisis del Three-Way Handshake (conexión TCP)

Cuando analizamos tráfico de red, uno de los aspectos más importantes es detectar **una conexión TCP completa**, lo que se conoce como el **three-way handshake**.

⚙️ ¿Qué es el Three-Way Handshake?

Es el proceso de inicio de una conexión TCP entre un **cliente** y un **servidor**, compuesto por tres pasos:

FASE	CLIENTE ENVÍA	SERVIDOR RESPONDE CON	CLIENTE CONFIRMA
1. INICIO	SYN		
2. ACEPTACIÓN		SYN + ACK	
3. CONFIRMACIÓN			ACK

🔍 ¿Cómo identificarlo en Wireshark o en un archivo .pcap?

1. **Filtrar por TCP**:
   Puedes usar el filtro tcp para centrarte en paquetes relevantes.
2. **Buscar flags SY SYN-ACK, ACK**:
   - El cliente inicia con **SYN** (Flags: 0x02).
   - El servidor responde con **SYN, ACK** (Flags: 0x12).

- El cliente finaliza con **ACK** (Flags: 0x10).

3. **Observar las IPs y puertos**:
   - IP origen y destino: indican cliente y servidor.
   - Puerto destino: indica el **servicio** solicitado (por ejemplo, 80, 443, 53, etc.).

🧪 Ejemplo de análisis

En la traza:

- **IP cliente**: 192.168.1.137
- **IP servidor**: 200.25.130.X
- **Puerto destino**: 53 (TCP)

Entonces, el cliente está intentando conectarse al **puerto 53** del servidor (DNS sobre TCP).

Detalles del intercambio:

Paso	IP origen	IP destino	Flags	Descripción
1	192.168.1.137	200.25.130.X	SYN	Inicio de conexión
2	200.25.130.X	192.168.1.137	SYN, ACK	Aceptación
3	192.168.1.137	200.25.130.X	ACK	Confirmación final

🔢 ¿Qué más analizar?

- **Número de secuencia** (Sequence Number) y de **acknowledgment** (ACK Number)
- **Puertos**:
  - **Origen**: usualmente aleatorio (ephemeral)
  - **Destino**: indica el **servicio solicitado**
- **Interpretación de capas**:
  - Capa 2: MAC origen/destino
  - Capa 3: IPs
  - Capa 4: TCP y sus flags

🧠 Pregunta típica de examen

"Dado un archivo .pcap, identifique una conexión TCP completa, indicando: IP cliente, IP servidor, puerto destino, flags y número de secuencia inicial".

📌 Conclusión

- Un **three-way handshake completo** es indicio de que un cliente y un servidor han establecido una conexión TCP exitosa.
- Puedes identificarlo fácilmente en Wireshark observando **los flags SYN, SYN-ACK y ACK** en secuencia.
- Esta estructura es **clave en exámenes, pruebas prácticas y auditorías de tráfico de red**.

🌐 Análisis de tráfico HTTP, HTTPS y detección de amenazas

Al analizar paquetes en Wireshark, muchas veces la clave está en **interpretar correctamente los puertos y protocolos**, especialmente los conocidos y los cifrados.

📌 ¿Cómo saber qué servicio se está usando?

✅ Claves para identificar servicios:

Puerto	Protocolo	Servicio común
53	UDP/TCP	DNS
80	TCP	HTTP (no cifrado)
443	TCP	HTTPS (con TLS/SSL)
21	TCP	FTP
25	TCP	SMTP (correo)

⚠️ Si ves un puerto **alto y aleatorio** como origen (ej. 49753), no te confundas. **El puerto destino** es el que determina el servicio solicitado.

🔍 ¿Por qué no se ve "HTTPS" en Wireshark?

Cuando visitas una página segura, ves tráfico al puerto 443, **pero no verás "HTTPS" directamente**. En su lugar, verás:

🔐 **TLS** o **SSL** → estos son los protocolos que cifran la comunicación.

Por eso:

- **HTTP (puerto 80)**: muestra contenido legible (métodos GET, POST, cabeceras, cuerpo del mensaje).
- **HTTPS (puerto 443)**: muestra tráfico cifrado → Wireshark lo etiqueta como **TLSv1.2**, **TLSv1.3**, etc.

🕵️ Análisis de comportamiento sospechoso

En la traza mencionada:

- Se observa tráfico **HTTP** que accede a un archivo llamado connect.txt.
- El contenido del tráfico hace referencia a msf, msfm, MSF, etc.

Esto es **sospechoso**, ya que:

⚠️ msf y msfm suelen estar relacionados con **Metasploit Framework**, una suite usada para pruebas de penetración, pero también por atacantes.

Indicadores:

- URL extrañas o genéricas: connect.txt, shell.php, etc.
- IPs no reconocidas o geográficamente lejanas.
- Tráfico HTTP sin cifrado con comandos embebidos o *payloads*.
- Patrón de conexión automatizada o repetitiva.

🧪 ¿Qué hacer en un examen?

Si te entregan un .pcap y te piden:

"Identifica una conexión sospechosa y di qué servicio se está usando."

Debes:

1. Filtrar por http, tcp.port == 80 o tls si buscas HTTPS.
2. Ver si hay:
    - Texto legible en HTTP.
    - URLs sospechosas.
    - Peticiones POST/GET que acceden a scripts o archivos maliciosos.
3. Analizar direcciones IP y puertos de destino.
4. Correlacionar con herramientas o firmas conocidas (como msf).

✅ Conclusión

- **HTTP** te muestra contenido visible: puedes ver directamente qué archivos o rutas se acceden.
- **HTTPS** no muestra contenido, solo el protocolo TLS, porque todo está cifrado.
- Si ves términos como msf, framework, shell, etc., en tráfico HTTP, puedes estar ante una **posible intrusión**.
- Saber identificar el servicio correcto por el **puerto de destino** es **clave en pruebas y auditorías**.

🔍 Análisis de una conexión HTTP sospechosa

Durante la inspección de paquetes se detectó una solicitud hacia un archivo .txt descargable, lo cual encendió alertas iniciales por su posible relación con **Metasploit Framework** o alguna actividad automatizada.

📌 ¿Por qué parece sospechoso?

- El archivo connect.txt fue solicitado vía HTTP (no cifrado).
- Aparecieron menciones a rutas y nombres que suelen asociarse con actividades ofensivas o automatizadas.
- Se sospechó inicialmente de alguna herramienta como **Metasploit**, por términos relacionados como msf o por el tipo de URL.

🧠 ¿Qué reveló el análisis más profundo?

1. **Inspección del proceso**:
    - Se detectó que el tráfico parece originarse de **Microsoft Teams**.
    - Específicamente, desde módulos de integración como msintegra o servicios relacionados al **perfilamiento de usuario**.
2. **Verificación de la IP de destino**:
    - La dirección IP pertenece a **Microsoft Azure (cloud)**.
    - Aunque esté alojado en Azure, **eso no garantiza que el servicio sea propiedad directa de Microsoft**.
    - Azure permite a cualquier organización alojar dominios en su infraestructura.
3. **Revisión del dominio**:

- Se utilizó un verificador de dominios y se confirmó que el dominio es legítimo y público.
- Aun así, es importante mantener la vigilancia, porque **atacantes pueden utilizar Azure o AWS** para camuflar sus servicios maliciosos.

⚠️ Lección clave

El hecho de que una IP pertenezca a Microsoft Azure no significa que el tráfico sea seguro. Siempre hay que **verificar el dominio, proceso asociado, tipo de archivo y patrón de tráfico**.

✅ Buenas prácticas

- Siempre inspeccionar procesos asociados a conexiones establecidas.
- Verificar si las IP pertenecen a rangos oficiales usando herramientas como:
  - whois
  - ipinfo.io
  - Shodan
- Revisar el contenido de los archivos .txt, .php, .sh u otros descargados o solicitados.
- **No asumir** que todo lo alojado en Azure, AWS o GCP es seguro.

🧪 Conclusión

Aunque en un inicio pareció una actividad ofensiva por el tipo de archivo y el tráfico HTTP sin cifrado, el análisis más detallado sugiere que:

- Se trata de tráfico **legítimo de Microsoft Teams**.
- El archivo y el dominio están alojados en **Microsoft Azure**, probablemente para fines de configuración o telemetría.
- Aun así, se debe mantener el criterio y la sospecha cuando se analizan este tipo de comportamientos automáticos en red.

¿Deseas que prepare una plantilla de análisis de incidentes para llenar cada vez que encuentres tráfico sospechoso, con campos como: IP, dominio, puerto, proceso asociado, verificación WHOIS, y recomendación final? Podría ayudarte a sistematizar tus reportes.

🔍 Análisis de tráfico HTTP sospechoso: ¿es un ataque o un comportamiento normal de Windows?

Durante el monitoreo de red con Wireshark, se detectó una conexión HTTP hacia una URL poco común que solicitaba un archivo .txt. Este tráfico salía desde la máquina local hacia un dominio de Microsoft Azure.

Debido al tipo de archivo (connect.txt) y la URL sospechosa, se pensó inicialmente en una posible actividad maliciosa:

- Descarga no autorizada de archivos.
- Uso de herramientas de explotación como **Metasploit**.
- Proceso oculto intentando establecer conexión.

Tras una inspección más detallada:

- Se confirmó que el tráfico HTTP **no entraba** a la máquina, sino **salía** de ella.
- Revisando los procesos asociados y la documentación oficial de Microsoft, se determinó que:

🔎 **El propio sistema operativo Windows realiza esta conexión como parte de su prueba de conectividad de red**.

  - Este comportamiento es parte del sistema para verificar si el equipo tiene acceso a Internet.

✅ Lecciones aprendidas

1. **Es mejor confirmar con evidencia que suponer**.
   - Lo que parecía malicioso (conexión HTTP y archivo .txt) terminó siendo una función legítima de Windows.
2. **Herramientas como Wireshark permiten ver todo**:
   - Tanto tráfico normal como tráfico anómalo.
   - Esto incluye procesos del sistema, navegadores, apps, herramientas internas y más.
3. **El contenido en HTTP es visible en texto plano**:
   - Puedes ver claramente cabeceras, rutas, archivos solicitados, etc.
   - Esto **no ocurre con HTTPS**, donde todo está cifrado a través de TLS/SSL.

🛡️ Recomendación práctica

Siempre que veas tráfico extraño, sigue este proceso:

- 📤 Determina si el tráfico entra o sale.
- 🔍 Investiga el dominio o IP (usando WHOIS, IPinfo, etc.).
- 🧩 Relaciona el proceso con el sistema o aplicaciones legítimas.
- 📚 Consulta documentación oficial (Microsoft, Linux, etc.).
- 🚫 Si aún hay dudas, aíslalo y analiza el comportamiento del proceso o servicio.

🧪 Conclusión final

Aunque el tráfico resultó ser legítimo, **el ejercicio fue valioso** porque:

- Demuestra cómo detectar posibles indicios de amenazas.
- Muestra la utilidad del análisis de streams TCP o seguimiento de conversaciones en HTTP.
- Refuerza la importancia de tener criterio analítico y no depender de primeras impresiones.

🔍 Seguimiento de conversaciones HTTP y reflexiones sobre privacidad

Durante el análisis de tráfico de red, hemos observado cómo a través de herramientas como **Wireshark**, es posible visualizar conversaciones completas en protocolos como **HTTP** y seguir cada paso del intercambio con precisión.

🌐 ¿Qué se puede ver en HTTP?

- El tráfico **no cifrado** de HTTP permite ver **todo**:
    - Métodos GET, POST, HEAD, etc.
    - URLs completas
    - Cabeceras (headers)
    - Respuestas del servidor (ej. 304 Not Modified, 200 OK, 404 Not Found)
- En el análisis realizado, se evidenció:

- Redirecciones (304) hacia nuevas URLs.
- Conversaciones sostenidas con servidores de **Google** y **Microsoft**, probablemente por servicios del sistema y aplicaciones activas (como Teams).

🔐 ¿Y qué ocurre con HTTPS?

- En **HTTPS** no verás este detalle.
- Solo verás paquetes cifrados bajo el protocolo **TLS/SSL**.
- Es posible que se filtre alguna URL parcial o nombre de dominio, pero **nunca verás el contenido** como en HTTP.

🔁 Seguimiento de secuencia TCP

- Al hacer un "TCP Stream Follow", se puede visualizar la **conversación completa** entre cliente y servidor.
- También se puede identificar:
  - Las tres fases del **three-way handshake**: SYN, SYN-ACK, ACK
  - El flujo de datos posterior
  - Las direcciones IP de origen y destino
  - Quién inicia la comunicación (según los flags TCP y la lógica del protocolo)

🧠 Reflexión crítica: ¿Qué está enviando mi máquina?

❓ *"¿Qué datos está enviando mi máquina realmente cuando se conecta a servidores como Microsoft, Google o Amazon?"*

Es una **pregunta legítima y necesaria**. Aunque muchas conexiones parecen inofensivas (diagnóstico de red, sincronización de tiempo, comprobación de conectividad), otras podrían:

- Enviar **telemetría sin consentimiento claro**
- Compartir **información del sistema, software, ubicaciones o comportamiento de usuario**
- Establecer conexiones **constantes** a múltiples servicios en segundo plano

---

🛡️ Recomendación para un análisis más profundo

1. **Monitorea constantemente** tu tráfico con herramientas como:
    - Wireshark
    - Tcpdump
    - Sysmon + SIEM
2. **Verifica destinos**:
    - ¿A qué servidores se conecta tu máquina automáticamente?
    - ¿Qué dominios y qué rangos IP son frecuentes?
3. **Evalúa el contenido (si no está cifrado)**:
    - ¿Qué tipo de datos se están transmitiendo?
    - ¿Hay algo que no debería salir de tu equipo?
4. **Pregúntate si confías en tu sistema operativo**.
    - No por paranoia, sino por auditoría consciente.

✅ **Conclusión final**

Este ejercicio demuestra que:

- **HTTP expone información legible** útil para el análisis, pero también para atacantes.
- **HTTPS protege la información**, pero también puede ocultar prácticas de telemetría invasiva.
- Las herramientas de captura permiten al analista ir más allá de lo evidente y cuestionar incluso lo "oficial".
- Tener una actitud **crítica, informada y técnica** es vital en seguridad informática.

🎙️ **Transcripción mejorada — Seguridad Informática (fragmento de clase)**

Tema: Uso de herramientas de análisis de red y preparación para unidad de criptografía
Duración estimada: [fragmento corto, cierre de sesión]

Docente:
Bueno, eso sería una introducción rápida sobre cómo utilizar esta herramienta —como Wireshark— para análisis de red.
Hemos hecho filtros, identificado protocolos, y lo más importante: **revisado el contenido** de los paquetes.

No se trata de que tengan que entender **cada campo técnico** del paquete, sino que sepan identificar y **entender los elementos clave**, como:

- Códigos de respuesta (HTTP)
- Direcciones IP de origen y destino
- Qué servicio podría estar funcionando detrás de ese tráfico
- Qué está ocurriendo realmente en la comunicación

Este enfoque les permitirá **interpretar** correctamente lo que ven y **no asumir** sin fundamentos.

🔍 Recomendación para las prácticas

Les recuerdo que esto es solo una **introducción práctica**. Ustedes deben reforzarla haciendo sus propios ejercicios:

- Usen el escáner
- Capturen tráfico
- Revisen detenidamente lo que están viendo
- No se queden en lo superficial

🧪 Próximo tema: Unidad 3 - Seguridad en Infraestructura

Vamos a comenzar con **la Unidad 3**, que trata sobre:

- **Seguridad en infraestructura**
- Escaneo de redes a un nivel más avanzado
- Evaluación de superficies de ataque

🔐 Unidad 7: Criptografía

Más adelante, en la **Unidad 7**, veremos el tema de **criptografía**, así que también iremos preparando el terreno para eso.

Docente:
Bueno, eso sería todo por hoy. Si tienen preguntas, este es el momento...

(*Pausa*. *Sin preguntas.*)

Muy bien, gracias a todos.

🏭 Seguridad en Infraestructuras Críticas (Redes OT vs Redes TI)

Fragmento de clase sobre control industrial, protocolos OT y vectores de ataque

---

🔌 ¿Qué son las redes OT?

Las redes **OT (Operational Technology)** están diseñadas para controlar **infraestructuras críticas**, como:

- Plantas de fabricación
- Sistemas eléctricos
- Procesos industriales
- Automatización de maquinaria

Estas redes conectan **sensores, actuadores, PLCs (controladores lógicos programables)** y sistemas SCADA (control y supervisión).

🧩 Diferencias entre redes OT y TI

- Las redes **TI (Tecnologías de la Información)** están orientadas a la gestión de datos (correo, bases de datos, servidores).
- Las redes **OT**, en cambio, están enfocadas en la **operación física de procesos**.

Un punto crucial es la **interfaz** que conecta ambos mundos (TI y OT), generalmente ubicada en el **Nivel 3 del modelo de arquitectura industrial**.

🔗 Protocolos comunes en OT

- **Modbus** (o Modbus TCP): muy utilizado para comunicación entre dispositivos industriales.
- **MQTT**: protocolo ligero para sistemas IoT, utilizado en monitoreo de sensores y telemetría.
- Otros protocolos específicos: **DNP3**, **Profinet**, **OPC-UA**, etc.

⚠️ *Estos protocolos no fueron diseñados originalmente con seguridad en mente.*

🏗️ Modelo de Arquitectura Industrial (Capas)

1. **Nivel 0**: Dispositivos físicos (sensores, actuadores)

2. **Nivel 1**: Controladores (PLCs, RTUs)
3. **Nivel 2**: Supervisión (SCADA, HMI)
4. **Nivel 3**: Red de control / Interfaz OT-TI
5. **Nivel 4**: Red corporativa (TI)
6. **Nivel 5**: Nube / Acceso externo

🧠 Este modelo permite identificar dónde aplicar seguridad y cómo se puede comprometer cada capa.

🔒 Vectores de ataque comunes en redes OT

1. **Accesos físicos no autorizados**
2. **Conexiones expuestas a internet (HMI o displays mal configurados)**
3. **Intrusión desde la red TI hacia la red OT**
4. **Ransomware industrial**: uno de los ataques más frecuentes y peligrosos

---

🔬 MITRE ATT&CK for ICS (Industrial Control Systems)

- Existe una versión específica del **framework MITRE ATT&CK** enfocada en sistemas industriales.
- Identifica tácticas y técnicas específicas utilizadas para comprometer redes OT.
- Puedes consultarla aquí: https://attack.mitre.org/matrices/ics/

📌 Recomendaciones

- Estudiar las **diferencias entre protocolos de red TI y OT**.
- Revisar el **modelo de arquitectura industrial**.
- Explorar vectores de ataque más comunes, especialmente en entornos críticos como energía, transporte o manufactura.