
UNIVERSIDAD DEL ISTMO

FACULTAD DE INGENIERÍA

# Parcial 3: Traceroute Manual en Python

NAJAR VENAMENTE, JOSE HUMBERTO

Guatemala, 11 de abril de 2025

# Índice

# Introducción

Este proyecto consiste en la implementación de un script en Python que simula el funcionamiento de la herramienta traceroute, utilizando el protocolo ICMP para identificar los saltos de red entre un dispositivo local y un servidor de destino. Se listan las direcciones IP intermedias, medir el tiempo de respuesta de cada salto y resolver manualmente los nombres de host, todo sin utilizar herramientas externas como nmap. Además, se analizan las repercusiones de tener ICMP habilitado,  se presentan métodos de prevención ante posibles riesgos de seguridad y un caso de estudio.

# Objetivos

**Objetivo específico**

- Desarrollar un script en Python que emule el funcionamiento de traceroute, permitiendo identificar los saltos de red hacia un destino, calcular el tiempo de respuesta por hop y resolver manualmente los nombres de host asociados a cada IP utilizando ICMP.

**Objetivos generales:**

- Comprender el funcionamiento del protocolo ICMP y su rol en el diagnóstico de red mediante herramientas como ping y traceroute.
- Aplicar conceptos de programación en Python para el envío y recepción de paquetes ICMP con control de TTL (Time To Live).
- Analizar las implicaciones de seguridad asociadas al uso de ICMP en redes abiertas, incluyendo métodos de prevención y casos reales de explotación.

## Descripción de librerías utilizadas

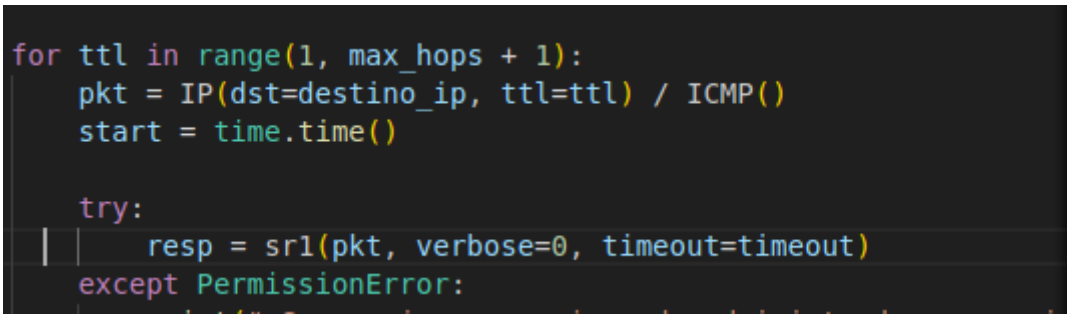
**Scapy**

Es una librería de Python que se especializa tanto en construcción, manipulación, envío y captura de paquetes de red, esto lo hace ya sea a nivel bajo (raw) como en protocolos como TCP, UDP, ICMP, ARP.

Se usa esta librería debido a que nos permite crear paquetes ICMP personalizados, donde podemos modificar manualmente el TTL (Time To Live).

En mi código lo que hace es lo siguiente:

- Crea un paquete ICMP (ICMP()).
- Lo envuelve en una capa IP con TTL controlado: IP(dst=destino_ip, ttl=ttl)
- Envía este paquete con sr1() y recibe la respuesta del hop correspondiente.



```
for ttl in range(1, max_hops + 1):
    pkt = IP(dst=destino_ip, ttl=ttl) / ICMP()
    start = time.time()

    try:
        resp = sr1(pkt, verbose=0, timeout=timeout)
    except PermissionError:
```

Aquí se crea se crea un paquete con destino a la IP deseada y con un TTL limitado. Además, sr1() lo envía y espera la respuesta del router donde expiró.

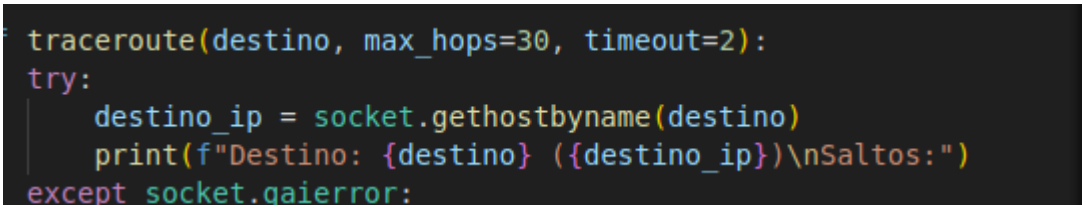
Y ya con se empieza el start, y se mide cuánto tiempo tardó en recibir respuesta desde que se envió el paquete, en milisegundos.

**Socket**

Es un módulo de Python que va a permitir poder trabajar con conexiones de red (como DNS, IPs, puertos). Por lo que nos ayudará a recibir y resolver conexiones.

Esto lo usamos para dos tareas específicas relacionadas con DNS:

- Obtener la IP desde un dominio



```
traceroute(destino, max_hops=30, timeout=2):
try:
    destino_ip = socket.gethostbyname(destino)
    print(f"Destino: {destino} ({destino_ip})\nSaltos:")
except socket.gaierror:
```

Aquí traduce el dominio escrito por el usuario a IP (gethostbyname).

- Obtener el nombre del host desde una IP

```
    tiempo = round((end - start) * 1000, 2)
    try:
        host = socket.gethostbyaddr(ip)[0]
    except:
        host = "Desconocido"
```

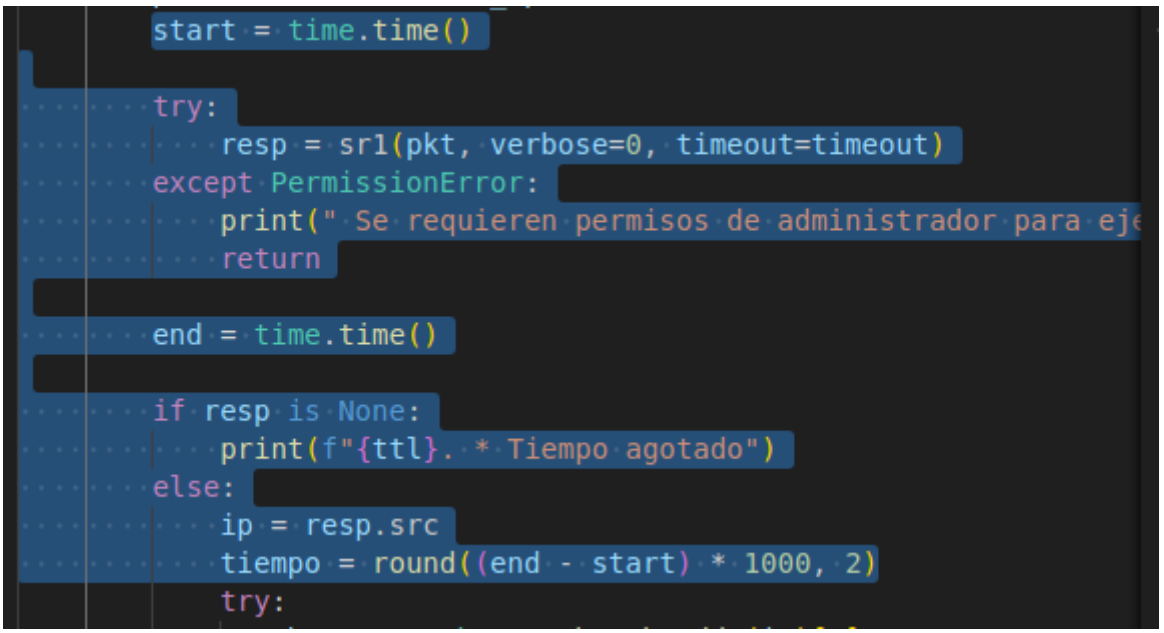
Aquí Intenta traducir la IP de cada hop a nombre DNS (gethostbyaddr).

**Time**

Es un módulo de Python que sirve para trabajar con tiempos, pero lo usamos para medir tiempos de ejecución (latencia).

Por lo que lo usamos para calcular el tiempo de ida y vuelta (RTT) entre el envío del paquete ICMP y la llegada de su respuesta.

En mi código hace lo siguiente:




- Guarda el momento justo antes de enviar el paquete en start = time.time()
- Guarda el momento cuando se recibe la respuesta en end = time.time()
- La diferencia (end - start) els el tiempo que tardó en responder ese hop.

# Métodos de prevención contra escaneos ICMP / traceroute

El protocolo ICMP (Internet Control Message Protocol) es esencial para herramientas de diagnóstico como ping y traceroute, ya que permite verificar la conectividad y el tiempo de respuesta entre dispositivos en una red.

Sin embargo, cuando ICMP está habilitado sin restricciones, también puede ser aprovechado por atacantes para realizar tareas de reconocimiento, como mapeo de red y descubrimiento de dispositivos vulnerables.

Por eso, existen métodos de prevención:

1. **Filtrado de ICMP en firewall**

Se pueden permitir que los firewalls filtren tipos específicos de mensajes ICMP. Por ejemplo:

- echo-request (utilizado por ping).

- time-exceeded (utilizado por traceroute).

Por lo que al configurar el firewall para permitir únicamente los tipos de ICMP necesarios, o restringirlos a ciertas direcciones IP autorizadas. Ayudando a reducir la visibilidad de la red ante herramientas de escaneo y evita la detección innecesaria de infraestructura.

2. **Rate limiting (limitación de frecuencia)**

Consiste en establecer un límite en la cantidad de respuestas ICMP que un dispositivo puede emitir por segundo. Esto nos ayuda a controlar el tráfico excesivo, protege los recursos del sistema y dificulta los ataques de reconocimiento automatizado.

3. **Deshabilitar ICMP en dispositivos internos**

Cuando hablamos en ambientes corporativos, no todos los dispositivos deben responder a paquetes ICMP, por lo que desactivar la respuesta ICMP en routers, servidores o equipos que no deban ser accesibles desde otros segmentos de red. Esto impide que dichos dispositivos aparezcan en un traceroute, dificultando el movimiento de un atacante.

4. **Monitoreo y detección de tráfico ICMP**

Los sistemas de detección/previsión de intrusos (IDS/IPS) permiten identificar patrones anómalos de tráfico ICMP. Por lo que se puede detectar intentos de escaneo mediante alertas o bloqueos automáticos, utilizando herramientas como Snort. Esto ayudará a reaccionar en tiempo real ante actividades sospechosas y proteger la red de posibles ataques y amenazas.

# Repercusiones de tener ICMP habilitado

Debemos recordar que el protocolo ICMP cumple una función vital en la administración y diagnósticos de redes, un ejemplo claro es que puede comprobar si un host está activo o rastrear la ruta que toma un paquete, su uso en manos intencionadas pueden representar un riesgo significativo en entornos de red sensibles o expuestos a internet. Estas son algunas de las repercusiones de mantener ICMP habilitado sin medidas de seguridad:

1. **Reconocimiento no autorizado**

Los atacantes pueden utilizar herramientas como ping sweep para escanear rangos de direcciones IP dentro de una red y detectar qué dispositivos están activos. Este tipo de reconocimiento inicial es clave para un atacante antes de lanzar un ataque más avanzado.

Un ejemplo puede ser un escaneo tipo **nmap -sn** usa ICMP para determinar qué dispositivos están vivos sin hacer ruido evidente.

2. **Detección de infraestructura crítica**

ICMP permite descubrir dispositivos intermedios en la red como routers, firewalls o balanceadores de carga a través de herramientas como traceroute. Esto ayuda a un atacante a mapear la topología de la red, algo crucial para realizar movimiento lateral dentro de una organización.

El riesgo recorre a que un atacante puede identificar rutas internas no protegidas o dispositivos mal configurados que no deberían ser accesibles desde redes externas.

3. **Ataques de DoS con ICMP**

ICMP también puede ser usado en ataques para interrumpir la disponibilidad de servicios. Como por ejemplo con el caso de **Ping flood** que fue el de saturar a un host con paquetes ICMP para agotar sus recursos.

Estos ataques pueden bloquear servicios, afectar servidores o incluso provocar reinicios si el sistema no está correctamente configurado.

4. **ICPM spoofing**

ICMP permite enviar mensajes de error como **Destination Unreachable** o **Redirect**. Un atacante puede falsificar estos mensajes ICMP para interrumpir o desviar el tráfico legítimo de una red. Un ejemplo claro puede ser que un atacante puede hacer que un equipo piense que un destino está caído o que debe re-enviar su tráfico por una ruta maliciosa.

## Caso real: Ataque “Smurf”

Uno de los ejemplos más conocidos del uso malicioso del protocolo ICMP fue el ataque Smurf, que alcanzó notoriedad en los años 90 y principios de los 2000 como una forma efectiva de realizar ataques de denegación de servicio distribuido (DDoS).

Este ataque aprovechaba la capacidad de amplificación del tráfico ICMP mediante el uso de direcciones de broadcast

El atacante enviaba paquetes ICMP echo-request (como los de un ping) a la dirección de broadcast de una red, por ejemplo, 192.168.0.255, utilizando como dirección de origen una IP falsificada (spoofed) que pertenecía a la víctima.

Todos los dispositivos dentro de esa red respondían simultáneamente al echo-request, enviando sus respuestas ICMP al remitente falsificado. El resultado era una inundación masiva de tráfico ICMP dirigido al objetivo, provocando saturación de ancho de banda, caída de servicios o bloqueo completo del sistema.

Sin embargo, este ataque es menos común hoy en día gracias a mejores configuraciones de red como desactivar la respuesta a broadcast ICMP.

## Referencias

Cloudflare. (n.d.). *What is a smurf attack?*. Cloudflare. Recuperado el 11 de abril de 2025, de https://www.cloudflare.com/learning/ddos/smurf-ddos-attack/

Fortinet. (n.d.). *Smurf attack*. Fortinet CyberGlossary. Recuperado el 11 de abril de 2025, de https://www.fortinet.com/resources/cyberglossary/smurf-attack