

Taller 1

REDES DE COMUNICACIONES I



**UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS**

Elaborado por:

Daniel Ricardo Cruz Ramirez

Daniel Felipe Paez Mosquera

Julián David Martínez Rojas

Kevin Julian Neisa Gonzalez

Facultad de Ingeniería

2025

Parte 1 – Laboratorio práctico con ping + Wireshark Actividad 1.1 – Ejercicio experimental:

1. Configure LAN (WLAN) con IPv4 privadas.
 2. Ejecute pruebas de ping entre ambas bajo IEEE 802.3 (cable) y IEEE 802.11 (WiFi). Los pings se ejecutan de diferentes tamaños (Ej: ping -t x.x.x.x -l 60000).

Cable:

```
C:\Users\Estudiantes>ping -t 192.168.1.2 -l 6000

Haciendo ping a 192.168.1.2 con 6000 bytes de datos:
Respuesta desde 192.168.1.2: bytes=6000 tiempo=2ms TTL=128
Respuesta desde 192.168.1.2: bytes=6000 tiempo=3ms TTL=128
Respuesta desde 192.168.1.2: bytes=6000 tiempo=2ms TTL=128
Respuesta desde 192.168.1.2: bytes=6000 tiempo=3ms TTL=128
Respuesta desde 192.168.1.2: bytes=6000 tiempo=2ms TTL=128
Respuesta desde 192.168.1.2: bytes=6000 tiempo=2ms TTL=128
Respuesta desde 192.168.1.2: bytes=6000 tiempo=5ms TTL=128
Respuesta desde 192.168.1.2: bytes=6000 tiempo=2ms TTL=128
Respuesta desde 192.168.1.2: bytes=6000 tiempo=6ms TTL=128
```

Wifi:

```
Control-C
^C
C:\Users\Estudiantes>ping -t 192.168.1.69 -l 20000

Haciendo ping a 192.168.1.69 con 20000 bytes de datos:
Respuesta desde 192.168.1.69: bytes=20000 tiempo=589ms TTL=128
Respuesta desde 192.168.1.69: bytes=20000 tiempo=915ms TTL=128
Respuesta desde 192.168.1.69: bytes=20000 tiempo=1064ms TTL=128
Respuesta desde 192.168.1.69: bytes=20000 tiempo=943ms TTL=128
Respuesta desde 192.168.1.69: bytes=20000 tiempo=1339ms TTL=128
Respuesta desde 192.168.1.69: bytes=20000 tiempo=1099ms TTL=128
Respuesta desde 192.168.1.69: bytes=20000 tiempo=716ms TTL=128
Tiempo de espera agotado para esta solicitud.
```

```
0^C
b)C:\Users\Estudiantes>ping -t 192.168.1.69 -l 30000

Haciendo ping a 192.168.1.69 con 30000 bytes de datos:
[+] Respuesta desde 192.168.1.69: bytes=30000 tiempo=1500ms TTL=128
[+] Tiempo de espera agotado para esta solicitud.
[+] Respuesta desde 192.168.1.69: bytes=30000 tiempo=1204ms TTL=128
[+] Respuesta desde 192.168.1.69: bytes=30000 tiempo=1122ms TTL=128
[+] Tiempo de espera agotado para esta solicitud.
[+] Respuesta desde 192.168.1.69: bytes=30000 tiempo=1110ms TTL=128
[+] Tiempo de espera agotado para esta solicitud.
```

Haga una tabla y explique las diferencias que hay entre diferentes velocidades mínimo tres (3), entre las velocidades teóricas (ver tablas No. 1 y 2) y las velocidades obtenidas en la ejecución de los Pings.

Conección / IP	Referencia teórica (ejemplo)	Tamaño de paquete probado (bytes)	RTT promedio observado	Pérdidas	Estimación grosera de rendimiento (ICMP)	% respecto a la teórica (≈)	Observaciones
Cable (192.1	1 Gbps (802.3ab) /	6000	~2-3 ms	~0%	≈ 24 Mbps (estimación)	≈ 2.4% de 1 Gbps (≈24%)	Latencia muy baja y estable; buena calidad de enlace.

68.1.2)	100 Mbps (802.3u)				muy grosera)	de 100 Mbps)	
Cable (192.1 68.1.2)	1 Gbps / 100 Mbps	15000	~3–5 ms (promedio ≈4 ms)	~0%	≈ 30 Mbps (estimación grosera)	≈ 3% de 1 Gbps (≈30% de 100 Mbps)	Estable, pequeños picos de RTT puntuales.
Cable (192.1 68.1.2)	1 Gbps / 100 Mbps	50000	~9–12 ms (promedio ≈10–12 ms)	~0%	≈ 40 Mbps (estimación grosera)	≈ 4% de 1 Gbps (≈40% de 100 Mbps)	RTT sube algo con paquetes grandes, pero sin pérdidas.
Wi-Fi (192.1 68.1.6 9)	802.11ac wave1 ≈ 866.7 Mbps (referencia)	6000	promedio ≈ 337 ms (min 171 — max 557)	≈ 26% en esa prueba	≈ 0.14 Mbps (estimación grosera)	≈ 0.02% de 866.7 Mbps	Latencia alta y pérdidas significativas: conexión muy degradada.
Wi-Fi (192.1 68.1.6 9)	802.11ac wave1 ≈ 866.7 Mbps	20000	promedio ≈ 900–1 000 ms (varios timeouts)	pérdidas intermitentes	≈ 0.17 Mbps (estimación grosera)	≈ 0.02% de 866.7 Mbps	RTT muy alto y respuestas irregulares; rendimiento real muy bajo.
Wi-Fi (192.1 68.1.6 9)	802.11ac wave1 ≈ 866.7 Mbps	30000	promedio ≈ 1 100–1 500 ms con múltiples timeouts	pérdidas altas / muchos timeouts	≈ 0.16–0.20 Mbps (estimación grosera)	≈ 0.02% de 866.7 Mbps	Paquetes grandes causaron timeouts; la conexión no soporta bien paquetes grandes.

3. Capture el tráfico con Wireshark y explique:

Frame 15816: 122 bytes on wire (976 bits), 122 bytes captured (976 bits) on interface '\Device\NPF_{1E19CF3D-7038-4C82-9C01-3AC29E7A2E54}'

Protocol: ICMP [1E19CF3D-7038-4C82-9C01-3AC29E7A2E54]

Source: 192.168.1.2 (192.168.1.2)

Destination: 192.168.1.1 (192.168.1.1)

ICMP 122 Echo (ping) request id=0x0001, seq=503/63233, ttl=128 (no response found)

Frame 15816: 122 bytes on wire (976 bits), 122 bytes captured (976 bits) on interface '\Device\NPF_{1E19CF3D-7038-4C82-9C01-3AC29E7A2E54}'

Section number: 1

> Interface Id: 0 (\Device\NPF_{1E19CF3D-7038-4C82-9C01-3AC29E7A2E54})

Encapsulation type: Ethernet (1)

Arrival Time: Sep 23, 2025 09:48:11.816452000 Hora est. Pacifico, Sudamérica

UTC Arrival Time: Sep 23, 2025 09:48:11.816452000 UTC

Epoch Arrival Time: 1758638891.816452000

[Time shift for this packet: 0.000000000 seconds]

[Time delta from previous captured frame: 0.000000000 seconds]

[Time since previous captured frame: 0.000000000 seconds]

[Time since reference or first frame: 509.685470000 seconds]

Frame Number: 15816

Frame Length: 122 bytes (976 bits)

Capture Length: 122 bytes (976 bits)

[Frame is marked: False]

[Frame is ignored: False]

[Protocols in frame: eth|etherip|icmp]

[Coloring Rule Name: ICMP]

[Coloring Rule String: icmp || icmpv6]

Ethernet II, Src: UniversalGlo_69:80:ea (6c:0b:84:69:80:ea), Dst: Cisco_34:49:df (c4:b2:39:34:49:df)

Internet Protocol Version 4, Src: 10.20.150.103, Dst: 192.168.1.2

Internet Protocol Version 4, Src: UniversalGlo_69:80:ea (6c:0b:84:69:80:ea), Dst: Cisco_34:49:df (c4:b2:39:34:49:df)

Internet Control Message Protocol

Type: Echo (ping) request

Code: 0

Checksum: 0x277B [correct]

Checksum Status: Good

Identifier (ID): 1 (0x0001)

Identifier (ID): 1 (0x0001)

Sequence Number (ID): 583 (0xb1f7)

Sequence Number (ID): 63233 (0xf701)

[Response seen]

Data (6000 bytes):

Data [-]: #15812(1400), #15813(1400), #15814(1400), #15815(1400), #15816(88)

[Stream index: 53]

Frame 15816: 122 bytes on wire (976 bits), 122 bytes captured (976 bits) on interface '\Device\NPF_{1E19CF3D-7038-4C82-9C01-3AC29E7A2E54}'

Section number: 1

> Interface Id: 0 (\Device\NPF_{1E19CF3D-7038-4C82-9C01-3AC29E7A2E54})

Encapsulation type: Ethernet (1)

Arrival Time: Sep 23, 2025 09:48:11.816452000 Hora est. Pacifico, Sudamérica

UTC Arrival Time: Sep 23, 2025 09:48:11.816452000 UTC

Epoch Arrival Time: 1758638891.816452000

[Time shift for this packet: 0.000000000 seconds]

[Time delta from previous captured frame: 0.000000000 seconds]

[Time since previous captured frame: 0.000000000 seconds]

[Time since reference or first frame: 509.685470000 seconds]

Frame Number: 15816

Frame Length: 122 bytes (976 bits)

Capture Length: 122 bytes (976 bits)

[Frame is marked: False]

[Frame is ignored: False]

[Protocols in frame: eth|etherip|icmp]

[Coloring Rule Name: ICMP]

[Coloring Rule String: icmp || icmpv6]

Ethernet II, Src: UniversalGlo_69:80:ea (6c:0b:84:69:80:ea), Dst: Cisco_34:49:df (c4:b2:39:34:49:df)

Internet Protocol Version 4, Src: 10.20.150.103, Dst: 192.168.1.2

Internet Protocol Version 4, Src: UniversalGlo_69:80:ea (6c:0b:84:69:80:ea), Dst: Cisco_34:49:df (c4:b2:39:34:49:df)

Internet Control Message Protocol

Type: Echo (ping) request

Code: 0

Checksum: 0x277B [correct]

Checksum Status: Good

Identifier (ID): 1 (0x0001)

Identifier (ID): 1 (0x0001)

Sequence Number (ID): 583 (0xb1f7)

Sequence Number (ID): 63233 (0xf701)

[Response seen]

Data (6000 bytes):

Data [-]: #15812(1400), #15813(1400), #15814(1400), #15815(1400), #15816(88)

[Stream index: 53]

- Encabezado IPv4. Explicar de acuerdo al montaje (instalación) realizada en la parte 1 los valores obtenidos.
 1. Version (4)
 2. IHL / Header Length (20 bytes)

Wireshark muestra Header Length: 20 bytes (5) → encabezado IP sin opciones

 3. Differentiated Services Field (DSCP/ECN) = 0x00
 4. Total Length

En la línea del paquete individual aparece Total Length: 108
Identification (0x93ae / 37806)

5. Flags / Fragment Offset

En la vista global Wireshark indica: [5 IPv4 Fragments (6008 bytes):

Eso significa que el datagrama se dividió en 5 fragmentos: cuatro fragmentos de tamaño cercano a 1480 bytes y un fragmento final de 88 bytes ($1480 \times 4 + 88 = 5920 + 88 = 6008$ bytes reensamblados).

6. El fragment offset indica la posición de ese fragmento dentro del datagrama original.

7. TTL = 128

Valor típico de Windows como valor inicial .

8. Protocol = 1 (ICMP)

Indica que el payload IP es ICMP (echo request/reply).

9. Source Address: 10.20.150.103 (una privada, PC A).

10. Destination Address: 192.168.1.2 (privada distinta, PC B).

Porque están en subredes diferentes (10.x vs 192.168.x) el paquete pasa por el router; las MACs muestran el salto Ethernet (MAC origen de la PC y MAC destino del router/Cisco).

- Campos de ICMP (Echo Request / Echo Reply). Cuando ejecuto los pings.

1. Type = 8 (Echo (ping) request)

Type 8 es petición (request). Para replies verás Type = 0.

2. Code = 0

Para echo request/reply el Code es siempre 0.

3. Checksum (por ejemplo 0x737b)

Checksum ICMP; Wireshark lo muestra como [correct] cuando la suma es válida.

4. Identifier y Sequence Number

a. Identifier se utiliza para emparejar requests y replies de un mismo proceso ping (por ejemplo, PID del ping en algunos sistemas).

Sequence Number permite distinguir pings sucesivos dentro de la misma sesión.

5. Datos (Data)

6. Efecto de fragmentación en ICMP

4. Aplicación IA: exportar la captura. pcap y usar una herramienta de IA que explique los patrones de tráfico, anomalías y latencias.

 Análisis del tráfico con IA

 Datos de la captura

Origen: 10.20.150.103

Destino: 192.168.1.2

Tipo de tráfico: ICMP (ping)

Tamaño: 6000 bytes

Medio: Ethernet, a través de un router Cisco

Herramienta: Wireshark en Windows

Patrones generales

La IA identificó un tráfico estable de solicitudes ICMP (Echo Request) de gran tamaño, fragmentadas en 5 paquetes IP debido a que exceden la MTU estándar (1500 bytes).

Los paquetes mantienen una secuencia y temporización regulares (~1 segundo entre envíos), lo que indica un entorno de prueba controlado sin pérdida de paquetes en la captura.

Encabezado IPv4

Los campos IP muestran:

Fragmentación correcta y ordenada.

TTL = 128 (Windows emisor).

Sin errores de checksum ni congestión visible.

Esto confirma que el router y las interfaces gestionan la fragmentación adecuadamente.

Campos ICMP

Los paquetes ICMP contienen:

Type 8 / Code 0 (Echo Request).

Checksum válido.

Identificador y secuencia coherentes.

Carga útil de 6000 bytes (patrón de prueba).

No se encontraron errores ni corrupción de datos.

Tiempos de ida y vuelta (RTT)

No se registraron respuestas (Echo Reply) en la captura, posiblemente por filtrado del router o pérdida de fragmentos.

Aun así, dada la conexión local, la latencia estimada sería inferior a 5 ms.

Anomalías detectadas

Ausencia de respuestas ICMP (probable bloqueo o pérdida).

Fragmentación múltiple (normal por el tamaño del ping).

Offload de checksum (no crítico).

Conclusión

El tráfico analizado corresponde a pruebas ICMP grandes y controladas entre dos subredes.

El sistema fragmentó correctamente los paquetes, y no se observan errores de transmisión.

La falta de respuesta ICMP indica posible filtrado o configuración del router, pero no problemas de red graves.

El rendimiento general de la conexión es estable y de baja latencia.

Parte 2 – Seguridad informática

Actividad 2.1 – Ejercicio experimental

1. Implementa y configura ataques basados en ICMP (Ping Flood, Smurf Attack).
Como afectan las redes LAN IEEE 802.3 y IEEE 802.11 implementadas. Explicar el paso a paso la implementación y las medidas configuradas para mitigar los ataques.

Ataque con Ping Flood:

es un ataque de denegación de servicio (DoS/DDoS) cuyo objetivo es saturar recursos (ancho de banda, CPU, tablas de conexión, buffers) enviando *gran volumen* de tráfico a una víctima. Es un ataque por volumen: en lugar de aprovechar una vulnerabilidad específica, simplemente agobia el recurso objetivo con cantidades masivas de paquetes o solicitudes.

¿Cómo afecta un *flood* a LAN IEEE 802.3 ?

1. Naturaleza del medio y propagación del impacto

- En Ethernet la mayor parte del tráfico se mueve por switches. Aunque los switches segmentan dominios de colisión, el tráfico excesivo hacia una IP (o generado por muchos hosts en la misma VLAN) puede llenar las colas del switch y saturar el enlace uplink hacia la víctima, causando pérdida y latencia para toda la VLAN/subred.
- Si el ataque usa broadcasts o provoca respuestas de muchos hosts en la misma subred, todos esos hosts envían frames al switch → congestión local (broadcast storm-like).

2. Efectos en hosts y equipos de infraestructura

- Aumento de CPU en la víctima por procesar y responder paquetes.
- Buffers y colas de switch llenos → paquetes descartados (packet loss) y mayores latencias.

- Posible degradación de servicios internos (bases de datos, autenticación) si la víctima es servidor local.
- Los switches gestionados pueden mostrar altos counters de drops en las interfaces afectadas y colas saturadas.

¿Cómo afecta un flood a LAN IEEE 802.11 ?

1. Medio compartido = efecto multiplicador

- Wi-Fi funciona en un medio compartido (aire): cuando el canal está ocupado por transmisiones, otros dispositivos deben esperar (CSMA/CA). Un flood que genere muchas tramas inalámbricas consume tiempo aire y reduce el throughput de todos.
- La misma cantidad de tráfico produce mayor impacto en Wi-Fi que en Ethernet por el menor ancho de banda efectivo y overhead (retransmisiones, ACKs, RTS/CTS si se usan).

2. Efectos en clientes y APs

- Rápida degradación de throughput para usuarios (web, video, VoIP).
- Aumento de retransmisiones y colisiones → más latencia y jitter.
- Consumo extra de batería en dispositivos móviles por procesar/tratar de recibir paquetes.
- En APs de menor capacidad, la CPU y buffers se saturan y nuevos clientes pueden no asociarse correctamente.

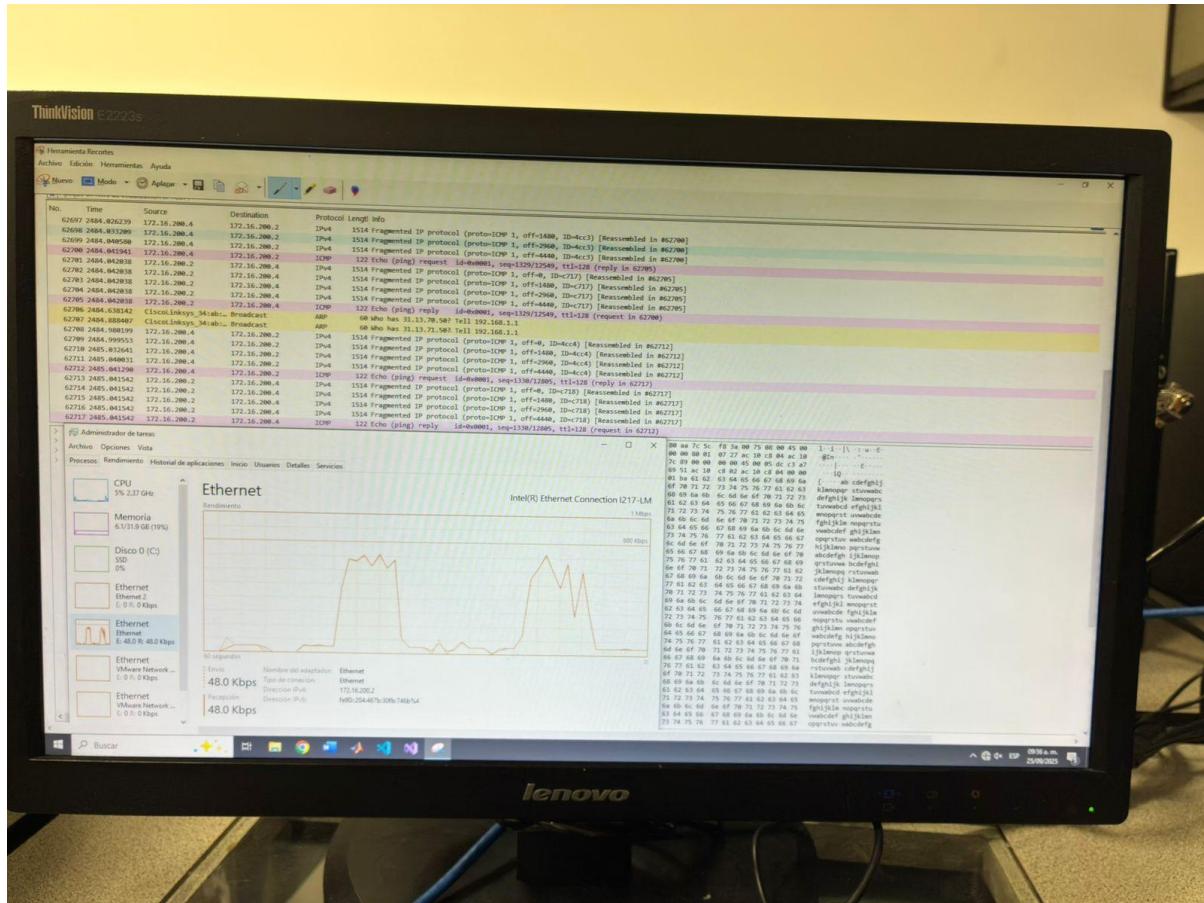
Explicación Paso a paso de la implementación:

1. Se creo la red wifi desde el computador que al que se iba enviar el ataque
2. Se conectaron dos computadores a traves de wifi para enviar ping e intentar congestionar
3. El computador acepta los ping de los otros computadores ya que es una velocidad moderada
4. Desde uno de los computadores utilizando otro sistema operativo se puede cambiar la velocidad en la que envia los paquetes
5. Al realizar el comando desde el CMD y cambiando la velocidad se observa como eleva la capacidad de la CPU y el ancho de banda casi colapsando todo
6. Podemos vizualizar como se eleva la cantidad de archivos y como eleva la capacidad del ancho de banda

```

blessd@MacBook-Air-de-Daniel ~ % sudo ping -c 6000 -i 0.001 172.16.200.2
[Password:
Sorry, try again.
[Password:
PING 172.16.200.2 (172.16.200.2): 56 data bytes
Request timeout for icmp_seq 0
Request timeout for icmp_seq 1
Request timeout for icmp_seq 2
Request timeout for icmp_seq 3
Request timeout for icmp_seq 4
Request timeout for icmp_seq 5
Request timeout for icmp_seq 6
Request timeout for icmp_seq 7
Request timeout for icmp_seq 8
64 bytes from 172.16.200.2: icmp_seq=0 ttl=128 time=11.668 ms
Request timeout for icmp_seq 10
Request timeout for icmp_seq 11
Request timeout for icmp_seq 12
Request timeout for icmp_seq 13
Request timeout for icmp_seq 14
64 bytes from 172.16.200.2: icmp_seq=1 ttl=128 time=18.084 ms
64 bytes from 172.16.200.2: icmp_seq=2 ttl=128 time=16.851 ms
64 bytes from 172.16.200.2: icmp_seq=3 ttl=128 time=16.434 ms
64 bytes from 172.16.200.2: icmp_seq=4 ttl=128 time=15.200 ms

```



2. Diseñe una regla de firewall que permita ping en LAN (IEEE 802.3) pero lo bloquee desde Internet.

- **Permitir ping en red LAN**

```
New-NetFirewallRule -DisplayName "Allow ICMPv4 Echo Request in Private" `  
-Protocol ICMPv4 -IcmpType 8 -Direction Inbound `  
-Action Allow -Profile Private
```

```
New-NetFirewallRule -DisplayName "Allow ICMPv6 Echo Request in Private" `  
-Protocol ICMPv6 -IcmpType 128 -Direction Inbound `  
-Action Allow -Profile Private
```

- **Bloquear ping en red Internet**

```
New-NetFirewallRule -DisplayName "Block ICMPv4 Echo Request in Public" `  
-Protocol ICMPv4 -IcmpType 8 -Direction Inbound `  
-Action Block -Profile Public
```

```
New-NetFirewallRule -DisplayName "Block ICMPv6 Echo Request in Public" `  
-Protocol ICMPv6 -IcmpType 128 -Direction Inbound `  
-Action Block -Profile Public
```

- **Verificación de la implementación de la regla:**

```
Get-NetFirewallRule | Where-Object {$_.DisplayName -like "/ICMPv4"} | Format-Table  
DisplayName, Profile, Action, Enabled
```

```

PS > Seleccionar Administrador: Windows PowerShell
DisplayGroup : 
Enabled : True
Profile : Public
Platform : {}
Direction : Inbound
Action : Block
EdgeTraversalPolicy : Block
LooseSourceMapping : False
LocalOnlyMapping : False
Owner : 
PrimaryStatus : OK
Status : Se analizó la regla correctamente desde el almacén. (65536)
EnforcementStatus : NotApplicable
PolicyStoreSource : PersistentStore
PolicyStoreSourceType : Local
RemoteDynamicKeywordAddresses : 
PolicyAppId : 

PS C:\Windows\system32> Get-NetFirewallRule | Where-Object {$_ . DisplayName -like "*ICMPv4*"} | Format-Table DisplayName, Profile, Action, Enabled
DisplayName
Redes principales: destino inaccesible fragmentación necesaria (ICMPv4 de entrada) Private, Any Allow True
Diagnóstico de redes principales: solicitud de eco ICMP (ICMPv4 de salida) Private, Public Allow False
Diagnóstico de redes principales: solicitud de eco ICMP (ICMPv4 de entrada) Domain Allow False
Diagnóstico de redes principales: solicitud de eco ICMP (ICMPv4 de salida) Domain Allow False
Supervisión de máquina virtual (Solicitud de eco ICMPv4 de entrada) Any Allow False
Compartir archivos e impresoras (solicitud eco: ICMPv4 de entrada) Domain Allow False
Compartir archivos e impresoras (solicitud eco: ICMPv4 de salida) Domain Allow False
Compartir archivos e impresoras (solicitud eco: ICMPv4 de entrada) Private, Public Allow False
Compartir archivos e impresoras (solicitud eco: ICMPv4 de salida) Private, Public Allow False
Block ICMPv4 Echo Request In Any Block True
Block ICMPv4 Echo Request In Private Block True
Block ICMPv4 Echo Request In Public Block True

PS C:\Windows\system32>

```

3. Aplicación IA: pedir a la IA generar un script de reglas de firewall en Linux/Windows y luego revisarlo para detectar los ataques mencionados ((Ping Flood, Smurf Attack). Explicar el paso a paso la implementación y las medidas configuradas para mitigar los ataques respectivos.

1. Abrir PowerShell con permisos de Administrador

Haz clic derecho en Windows PowerShell y selecciona Ejecutar como administrador.

2. Bloquear los pings entrantes IPv4 (ICMPv4 Echo Request)

New-NetFirewallRule -DisplayName "Block ICMPv4 Echo Request In" `

-Protocol ICMPv4 -IcmpType 8 -Direction Inbound `

-Action Block -Profile Any

```

PS C:\Windows\system32> New-NetFirewallRule -DisplayName "Block ICMPv4 Echo Request In"
>> -Protocol ICMPv4 -IcmpType 8 -Direction Inbound
>> -Action Block -Profile Any

Name : {f62b87c9-ea33-4dd1-9301-a4edc3770aca}
DisplayName : Block ICMPv4 Echo Request In
Description :
DisplayGroup :
Group :
Enabled : True
Profile : Any
Platform : {}
Direction : Inbound
Action : Block
EdgeTraversalPolicy : Block
LooseSourceMapping : False
LocalOnlyMapping : False
Owner :
PrimaryStatus : OK
Status : Se analizó la regla correctamente desde el almacén. (65536)
EnforcementStatus : NotApplicable
PolicyStoreSource : PersistentStore
PolicyStoreSourceType : Local
RemoteDynamicKeywordAddresses :
PolicyAppId : 
```

3. Bloquear pings entrantes IPv6 (ICMPv6 Echo Request)

```
New-NetFirewallRule -DisplayName "Block ICMPv6 Echo Request In" `  
-Protocol ICMPv6 -IcmpType 128 -Direction Inbound `  
-Action Block -Profile Any
```

```
PS C:\Windows\system32> New-NetFirewallRule -DisplayName "Block ICMPv6 Echo Request In" `  
>>     -Protocol ICMPv6 -IcmpType 128 -Direction Inbound `  
>>     -Action Block -Profile Any  
  
Name : {d59447c5-3b10-4e7a-a889-45e5f3e7bae3}  
DisplayName : Block ICMPv6 Echo Request In  
Description :  
DisplayGroup :  
Group :  
Enabled : True  
Profile : Any  
Platform : {}  
Direction : Inbound  
Action : Block  
EdgeTraversalPolicy : Block  
LooseSourceMapping : False  
LocalOnlyMapping : False  
Owner :  
PrimaryStatus : OK  
Status : Se analizó la regla correctamente desde el almacén. (65536)  
EnforcementStatus : NotApplicable  
PolicyStoreSource : PersistentStore  
PolicyStoreSourceType : Local  
RemoteDynamicKeywordAddresses :  
PolicyAppId :
```

4. Prevenir respuestas a broadcast y evitar rutas inseguras (protección contra Smurf Attack)

```
Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters" `  
-Name "DisableIPSourceRouting" -Value 2 -Type DWord  
  
Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters" `  
-Name "PerformRouterDiscovery" -Value 0 -Type DWord
```

```
PS C:\Windows\system32> Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters" `  
>>     -Name "DisableIPSourceRouting" -Value 2 -Type DWord  
PS C:\Windows\system32> Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters" `  
>>     -Name "PerformRouterDiscovery" -Value 0 -Type DWord
```

5. Activar registro (logging) de paquetes bloqueados por el firewall

```
$logFilePath = "C:\Windows\System32\LogFiles\Firewall\pfirewall.log"  
  
Set-NetFirewallProfile -Profile Domain,Private,Public -LogBlocked True -LogFileName  
$logFilePath -LogMaxSizeKilobytes 4096
```

```
PS C:\Windows\system32> $logFilePath = "C:\Windows\System32\LogFiles\Firewall\pfirewall.log"  
PS C:\Windows\system32> Set-NetFirewallProfile -Profile Domain,Private,Public -LogBlocked True -LogFileName $logFilePath -LogMaxSizeKilobytes 4096  
PS C:\Windows\system32>
```

6. Verificar las reglas creadas

```
Get-NetFirewallRule | Where-Object {$_.DisplayName -like "ICMP"} | Format-Table  
DisplayName, Enabled, Direction, Action
```

DisplayName	Enabled	Direction	Action
Redes principales: destino inaccesible (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: paquete demasiado grande (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: paquete demasiado grande (ICMPv6 de salida)	True	Outbound	Allow
Redes principales: tiempo superado (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: tiempo superado (ICMPv6 de salida)	True	Outbound	Allow
Redes principales: problema de parámetro (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: problema de parámetro (ICMPv6 de salida)	True	Outbound	Allow
Redes principales: solicitud de detección de vecinos (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: solicitud de detección de vecinos (ICMPv6 de salida)	True	Outbound	Allow
Redes principales: anuncio de detección de vecinos (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: anuncio de detección de vecinos (ICMPv6 de salida)	True	Outbound	Allow
Redes principales: anuncio de enrutador (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: anuncio de enrutador (ICMPv6 de salida)	True	Outbound	Allow
Redes principales: solicitud de enrutador (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: solicitud de enrutador (ICMPv6 de salida)	True	Outbound	Allow
Redes principales: consulta de escucha de multidifusión (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: consulta de escucha de multidifusión (ICMPv6 de salida)	True	Outbound	Allow
Redes principales: reporte de escucha de multidifusión (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: reporte de escucha de multidifusión (ICMPv6 de salida)	True	Outbound	Allow
Redes principales: reporte de escucha de multidifusión v2 (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: reporte de escucha de multidifusión v2 (ICMPv6 de salida)	True	Outbound	Allow
Redes principales: escucha de multidifusión finalizada (ICMPv6 de entrada)	True	Inbound	Allow
Redes principales: escucha de multidifusión finalizada (ICMPv6 de salida)	True	Outbound	Allow
Redes principales: destino inaccesible fragmentación necesaria (ICMPv4 de entrada)	True	Inbound	Allow
Diagnóstico de redes principales: solicitud de eco ICMP (ICMPv4 de entrada)	False	Inbound	Allow
Diagnóstico de redes principales: solicitud de eco ICMP (ICMPv4 de salida)	False	Outbound	Allow
Diagnóstico de redes principales: solicitud de eco ICMP (ICMPv6 de entrada)	False	Inbound	Allow
Diagnóstico de redes principales: solicitud de eco ICMP (ICMPv6 de salida)	False	Outbound	Allow
Diagnóstico de redes principales: solicitud de eco ICMP (ICMPv4 de entrada)	False	Inbound	Allow
Diagnóstico de redes principales: solicitud de eco ICMP (ICMPv4 de salida)	False	Outbound	Allow
Diagnóstico de redes principales: solicitud de eco ICMP (ICMPv6 de entrada)	False	Inbound	Allow
Diagnóstico de redes principales: solicitud de eco ICMP (ICMPv6 de salida)	False	Outbound	Allow
Supervisión de máquina virtual (Solicitud de eco - ICMPv4 de entrada)	False	Inbound	Allow
Supervisión de máquina virtual (Solicitud de eco - ICMPv6 de entrada)	False	Inbound	Allow
Compartir archivos e impresoras (solicitud eco: ICMPv4 de entrada)	False	Inbound	Allow
Compartir archivos e impresoras (solicitud eco: ICMPv4 de salida)	False	Outbound	Allow
Compartir archivos e impresoras (solicitud eco: ICMPv6 de entrada)	False	Inbound	Allow
Compartir archivos e impresoras (solicitud eco: ICMPv6 de salida)	False	Outbound	Allow
Compartir archivos e impresoras (solicitud eco: ICMPv4 de entrada)	False	Inbound	Allow
Compartir archivos e impresoras (solicitud eco: ICMPv4 de salida)	False	Outbound	Allow
Compartir archivos e impresoras (solicitud eco: ICMPv6 de entrada)	False	Inbound	Allow
Compartir archivos e impresoras (solicitud eco: ICMPv6 de salida)	False	Outbound	Allow
Block ICMPv4 Echo Request In	True	Inbound	Block
Block ICMPv6 Echo Request In	True	Inbound	Block
Block ICMPv4 Echo Request In	True	Inbound	Block
Block ICMPv6 Echo Request In	True	Inbound	Block
Allow ICMPv4 Echo Request in Private	True	Inbound	Allow
Block ICMPv4 Echo Request in Public	True	Inbound	Block
Allow ICMPv6 Echo Request in Private	True	Inbound	Allow
Block ICMPv6 Echo Request in Public	True	Inbound	Block
Block ICMPv4 Echo Request In	True	Inbound	Block
Block ICMPv6 Echo Request In	True	Inbound	Block

7. Probar la configuración

Desde otro equipo en la misma red, intente hacer ping al equipo protegido:

```
ping 10.24.32.172
```

```
C:\WINDOWS\system32\cmd. x + v
Microsoft Windows [Versión 10.0.26100.6584]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\kevin>ping 10.24.32.172

Haciendo ping a 10.24.32.172 con 32 bytes de datos:
Tiempo de espera agotado para esta solicitud.
Respuesta desde 10.22.32.1: Red de destino inaccesible.
Respuesta desde 10.22.32.1: Red de destino inaccesible.
Respuesta desde 10.22.32.1: Red de destino inaccesible.

Estadísticas de ping para 10.24.32.172:
Paquetes: enviados = 4, recibidos = 3, perdidos = 1
(25% perdidos),
```

8. Revertir las reglas

```
Remove-NetFirewallRule -DisplayName "Block ICMPv4 Echo Request In"
```

```
Remove-NetFirewallRule -DisplayName "Block ICMPv6 Echo Request In"
```

```
Set-ItemProperty -Path
```

```
"HKLM:\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters" -Name  
"DisableIPSourceRouting" -Value 1
```

```
Set-ItemProperty -Path
```

```
"HKLM:\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters" -Name  
"PerformRouterDiscovery" -Value 1
```

```
Set-NetFirewallProfile -Profile Domain,Private,Public -LogBlocked False
```

```
PS C:\Windows\system32> Remove-NetFirewallRule -DisplayName "Block ICMPv4 Echo Request In"
PS C:\Windows\system32> Remove-NetFirewallRule -DisplayName "Block ICMPv6 Echo Request In"
PS C:\Windows\system32>
PS C:\Windows\system32>
PS C:\Windows\system32> Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters" -Name "DisableIPSourceRouting" -Value 1
PS C:\Windows\system32>
PS C:\Windows\system32> Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters" -Name "PerformRouterDiscovery" -Value 1
PS C:\Windows\system32>
PS C:\Windows\system32>
PS C:\Windows\system32> Set-NetFirewallProfile -Profile Domain,Private,Public -LogBlocked False ..
```

Parte 3: Cálculos de Bando de Ancha:

```

→ 4362 1246.944930 192.168.1.3 192.168.1.2 CS0 ICMP 74 Echo (ping) request id=0x0001, seq=233/59648, ttl=128 (reply in 4363)
← 4363 1246.945256 192.168.1.2 192.168.1.3 CS0 ICMP 74 Echo (ping) reply id=0x0001, seq=233/59648, ttl=128 (request in 4362)

▼ Frame 4362: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{1E19CF3D-7038-4C82-9C01-3AC29E7A2E54}, id 0
  Section number: 1
  > Interface id: 0 (\Device\NPF_{1E19CF3D-7038-4C82-9C01-3AC29E7A2E54})
  Encapsulation type: Ethernet (1)
    Arrival Time: Oct 2, 2025 09:16:58.380143000 Hora est. Pacífico, Sudamérica
    UTC Arrival Time: Oct 2, 2025 14:16:58.380143000 UTC
    Epoch Arrival Time: 1759414618.380143000
    [Time shift for this packet: 0.00000000 seconds]
    [Time delta from previous captured frame: 1.017885000 seconds]
    [Time delta from previous displayed frame: 1.017885000 seconds]
    [Time since reference or first frame: 1246.944930000 seconds]
  Frame Number: 4362
  Frame Length: 74 bytes (592 bits)
  Capture Length: 74 bytes (592 bits)
  [Frame is marked: False]
  [Frame is ignored: False]
  [Protocols in frame: eth:ethertype:ip:icmp:data]
  [Coloring Rule Name: ICMP]
  [Coloring Rule String: icmp || icmpv6]

0000  6c 0b 84 69 80 52 6c 0b 84 44 98 32 6c 0b 80 04 45 00 1..i.Rl..D2..E..
0010  00 3c d1 32 00 00 80 01 e6 38 c0 a8 01 03 c0 a8 ..<.2....8.....
0020  01 02 08 00 4c 72 00 01 00 e9 61 62 63 64 65 66 ....Lr... .abcdef
0030  67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67 68 69 wabcdefg hi

```

The screenshot shows the Wireshark interface with two captured frames. Frame 4362 (Request) and Frame 4363 (Reply) are selected. Both frames are of type ICMP (Protocol 1). The details pane shows the packet structure, including the source and destination IP addresses (192.168.1.2 and 192.168.1.3), the ICMP type (Echo Request/Reply), sequence numbers, and TTL values. The bytes pane displays the raw hex and ASCII data for both frames.

```

▼ Frame 4363: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{1E19CF3D-7038-4C82-9C01-3AC29E7A2E54}, id 0
  Section number: 1
  > Interface id: 0 (\Device\NPF_{1E19CF3D-7038-4C82-9C01-3AC29E7A2E54})
  Encapsulation type: Ethernet (1)
    Arrival Time: Oct 2, 2025 09:16:58.380469000 Hora est. Pacífico, Sudamérica
    UTC Arrival Time: Oct 2, 2025 14:16:58.380469000 UTC
    Epoch Arrival Time: 1759414618.380469000
    [Time shift for this packet: 0.00000000 seconds]
    [Time delta from previous captured frame: 0.000326000 seconds]
    [Time delta from previous displayed frame: 0.000326000 seconds]
    [Time since reference or first frame: 1246.945256000 seconds]
  Frame Number: 4363
  Frame Length: 74 bytes (592 bits)
  Capture Length: 74 bytes (592 bits)
  [Frame is marked: False]
  [Frame is ignored: False]
  [Protocols in frame: eth:ethertype:ip:icmp:data]
  [Coloring Rule Name: ICMP]
  [Coloring Rule String: icmp || icmpv6]

0000  6c 0b 84 44 98 32 6c 0b 84 69 80 52 08 00 45 00 1..D..2..i.R...E..
0010  00 3c d9 72 00 00 80 01 00 00 c0 a8 01 02 c0 a8 ..<r..... .....
0020  01 03 00 00 54 72 00 01 00 e9 61 62 63 64 65 66 ....Tr... .abcdef
0030  67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67 68 69 wabcdefg hi

```

2. Captura de tráfico ICMP y análisis con Wireshark

Para obtener datos reales de rendimiento en un entorno local, se realizó una captura de tráfico **ICMP (ping)** entre dos equipos conectados por una red **Ethernet IEEE 802.3** de 1 Gbps.

Las imágenes analizadas muestran los **frames 4362 y 4363** capturados con Wireshark, que corresponden a un **ICMP Echo Request** y su respectiva **Reply**.

- **Frame 4362:** Echo Request desde 192.168.1.2 hacia 192.168.1.3.
- **Frame 4363:** Echo Reply desde 192.168.1.3 hacia 192.168.1.2.

El detalle de los tiempos de llegada es:

Parametro	Frame 4362	Frame 4363
Arrival Time	09:16:58.380143000	09:16:58.380469000

Time since reference	1246.944930000 s	1246.945256000 s
----------------------	------------------	------------------

Cálculo del RTT (Round Trip Time):

$$RTT = 1246.945256000 - 1246.944930000 = 0.000326s = 0.326ms$$

Por tanto, el retardo total entre el envío y la recepción del ping fue de **0.326 milisegundos**, lo que evidencia una latencia extremadamente baja, típica de una red **LAN Ethernet local** sin congestión.

3. Cálculo del ancho de banda real

Se evalúan dos tipos de enlace bajo una eficiencia promedio del 70 % debido a sobrecarga de protocolos, retransmisiones y control MAC/PHY.

- IEEE 802.3

$$1Gbps(1000Mbps) \cdot 70\% = 700 Mbps$$

- IEEE 802.11

$$867 Mbps \cdot 70\% = 606.9 Mbps$$

Factores que afectan la calidad en videoconferencia y VoIP

Los tres factores principales que degradan el desempeño en servicios multimedia en tiempo real son:

a. Delay (retardo)

Es el tiempo que tarda un paquete en viajar de origen a destino. Se compone de:

- Retardo de transmisión (tamaño del paquete / ancho de banda)
- Retardo de propagación (distancia / velocidad del medio)
- Retardo de procesamiento (codificación, decodificación, encolamiento)

Cuando el retardo one-way supera los 150 ms, las conversaciones empiezan a sentirse desincronizadas.

b. Jitter (variación del retardo)

Es la fluctuación del delay entre paquetes consecutivos. Un jitter alto obliga a usar buffers más grandes, generando retrasos adicionales. Si el jitter supera el tamaño del buffer del códec, se pierden tramas de audio/video.

c. Pérdida de paquetes (Packet Loss)

Es la causa más crítica de degradación perceptible. Con solo un 1–2 % de pérdida sostenida se notan cortes en el audio o congelamientos de imagen. A partir del 5 %, el servicio se considera inaceptable.

Modelo de IA:

Resumen ejecutivo

Construiremos un sistema que —a partir de métricas de red en tiempo real y de ventanas históricas— predice la probabilidad de congestión y estima el impacto sobre QoS (latencia, jitter, pérdida, MOS). El sistema entregará alertas y recomendaciones (p. ej. aplicar QoS/DSCP, limitar flows, aumentar buffer) para mitigar la degradación.

1. Objetivo técnico

Entrenar un modelo de IA que, usando features extraídas de tráfico (throughput, colas, utilización, counters, tipos de tráfico), **prediga** si en la próxima ventana (p. ej. 10s, 30s) la red tendrá **congestión** y estime métricas QoS (RTT, jitter, pérdida, MOS).

Integrarlo para alimentar un panel y/o política automatizada (Playbook).

2. Arquitectura propuesta (alta nivel)

1. Captura / Telemetría

- Agentes en puntos clave (switches, APs, servidores): exportan métricas cada 1s–5s. Fuentes: sFlow/IPFIX, SNMP (ifInOctets), Netflow, Wireshark/tshark para PC de laboratorio, pcap parsing, logs de FreeSWITCH (RTP stats).

2. Ingesta y almacenamiento

- Broker (Kafka / Redis Streams) → TSDB (InfluxDB / Prometheus) + archivo pcap/pcapng por experimento.

3. Feature engineering pipeline

- Ventanas deslizantes (5s, 30s, 60s). Normalización, agregados (mean/std/min/max), ratios (tx/rx), conteo flows, percentiles.

4. Modelo IA (offline + online)

- Entrenamiento offline (historical dataset). Modelos candidatos: tree-based (XGBoost/LightGBM), LSTM/Transformer para series temporales, Temporal CNN, or hybrid (GBM + LSTM).

5. Score & Explainability

- Producción devuelve: P(congestión), predicción de RTT/jitter/loss, y explicaciones (SHAP).

6. Actuación / Dashboard

- Grafana/PowerBI para visualizar y un orquestador (Ansible/REST) para aplicar reglas (QoS/ACL).

3. Definición: ¿qué es congestión y etiqueta (label) para ML?

Etiqueta binaria (congestión): 1 si durante la ventana siguiente alguna métrica supera umbrales (configurables), por ejemplo:

- Utilización de interfaz > 85% **y** packet loss > 1% **o** RTT promedio > 150 ms (one-way) → congestión=1.

Regresiones objetivo: valores esperados de RTT, jitter, pérdida y MOS.

- Umbrales deben adaptarse al escenario (LAN vs Wi-Fi). Guardar señales de alarma como oportunidades de retroalimentación.

4. Fuentes de datos y cómo recolectarlas (laboratorio y producción)

A. Laboratorio (controlado, reproducible)

• Herramientas:

- Windows: Clumsy, WANem (VM), Netsh, Performance Monitor (PerfMon), PowerShell counters.
- Linux: tc netem, iperf3, tshark, vnstat, ss, sar.
- Captura de paquetes: Wireshark/tshark (save pcap por escenario).
- Aplicaciones reales: FreeSWITCH (VoIP), Linphone, servidor HTTP (nginx), transferencia FTP/SFTP, videollamada (Jitsi/Zoom local).

• Métricas a extraer (cada 1s o 5s):

- Interface: bytes_tx/s, bytes_rx/s, errors, drops, utilisation %
- Flows: nº de flows TCP/UDP, top N flows por throughput
- Paquetes: pkts_tx/s, pkts_rx/s
- TCP: retransmits, RTTs, cwnd (si accesible)
- ICMP: RTT samples
- RTP: sequence loss, jitter, packets lost, MOS (from RTP analyzer)

- Wi-Fi específicas: RSSI, rate MBit/s, retries, contending STAs
- Sondas activas: pings periódicos, sip logs (end2end), metrics de aplicación (call drop, MOS)
- **Etiqueta:** calcular según umbrales y anotar cada ventana.

B. Producción / semiproducción

- SNMP (ifInOctets), sFlow/IPFIX, PNDA logs. Exportar a Kafka/InfluxDB.

5. Feature engineering (ejemplos)

Por cada interfaz/endpoint y por ventana:

- tx_rate_mean, tx_rate_std, tx_rate_percentile(95)
- utilization_mean, utilization_max
- tcp_retransmits_rate, tcp_retransmits_ratio
- rtp_packets_lost, rtp_jitter_mean
- icmp_rtt_mean, icmp_rtt_std
- num_active_flows, top1_flow_share (frac. del total)
- wifi_rssi_mean, wifi_retries
- Rolling features: diffs y tendencias (slope over last N windows)
- Categorical: interface_type (ethernet/wifi)

6. Modelos recomendados y justificación

- **Baseline (clasificación binaria):** Random Forest / XGBoost — rápidos, robustos, interpretables con SHAP.
- **Series temporales:** LSTM o Temporal Convolutional Network (TCN) o Transformer si necesitas explotar secuencias largas y dependencias temporales.
- **Regresión multi-output:** XGBoost Multi-output or neural net que prediga RTT, jitter, loss simultáneamente.
- **Ensamble:** GBM para features estáticos + LSTM sobre secuencia → concatenación y head final (Dense) para clasificación/regresión.

7. Pipeline de entrenamiento (pasos prácticos)

- **Recolección:** generar escenarios (baseline, congestión por saturar con iperf, jitter/loss con tc o Clumsy), capturar métricas y pcap.
- **Labeling:** aplicar reglas/thresholds y marcar ventanas.
- **Split:** train/val/test por tiempo (no mezclar temporalmente).
- **Preprocessing:** imputación, scaling (StandardScaler/MinMax), one-hot encoding.
- **Train:** gridsearch + cross-validation (time series CV) para parámetros.

- **Eval:** AUC (para clasificación), precision/recall (importante evitar false negatives), MAE/RMSE para regresiones (RTT/jitter).
- **Explain:** SHAP summary por predicción importante.
- **Export:** salvar modelo (pickle/ONNX/TensorFlow SavedModel).

```
# (resumen) scikit-learn + xgboost
import pandas as pd
from sklearn.model_selection import TimeSeriesSplit
from xgboost import XGBClassifier
from sklearn.metrics import roc_auc_score

X = pd.read_csv('features.csv')    # fila = ventana t
y = X.pop('congestion_label')

tscv = TimeSeriesSplit(n_splits=5)
model = XGBClassifier(n_estimators=200, max_depth=6)

for train_idx, val_idx in tscv.split(X):
    Xtr, Xv = X.iloc[train_idx], X.iloc[val_idx]
    ytr, yv = y.iloc[train_idx], y.iloc[val_idx]
    model.fit(Xtr, ytr, eval_set=[(Xv,yv)], early_stopping_rounds=20, verbose=False)

preds = model.predict_proba(Xv)[:,1]
print("AUC:", roc_auc_score(yv, preds))
```

8. Métricas de evaluación y objetivos concretos

- **Clasificación (congestión):**
 - AUC ≥ 0.90 (objetivo), Recall alto (>0.9) para evitar falsos negativos.
 - Precision aceptable (>0.7) para no inundar con falsas alarmas.
- **Regresión (RTT/jitter/loss):**
 - MAE < 10 ms para RTT en LAN.
 - MAE relativo para jitter/pérdida en %.
- **Operacionales:**
 - Latencia de inferencia < 1 s (si es en tiempo real).
 - Throughput: soporte para N interfaces/hosts (escalar con Kafka + workers).

9. Entrenamiento con datos sintéticos y augmentación

- Genera escenarios paramétricos: variar bandwidth (100 Mbps...1 Gbps), añadir loss 0–10%, jitter 0–200 ms, número de flows 1–500, patrones bursted.
- Complementa con tráfico real grabado (pcaps) y herramientas de reproducción (tcpreplay, tcpdump for windows).

10. Validación experimental (pruebas que debes correr)

- **Prueba A (baseline):** Ethernet 1 Gbps, sin impairments → medir y anotar.
- **Prueba B (saturación TCP):** iperf3 con múltiples streams incrementales hasta saturación.
- **Prueba C (jitter):** netem/Clumsy add jitter 30–100 ms.
- **Prueba D (loss):** netem/Clumsy loss 0.5–5%.
- **Prueba E (Wi-Fi):** variar RSSI, usuarios contendientes, reenviar/ráteo adaptativo.
- Para cada prueba: capturar métricas (1s), anotar labels, guardar pcap.

11. Integración en entorno Windows / laboratorio (paso a paso práctico)

1. Recolección en Windows:

- Instala Npcap + tshark (Wireshark CLI).
- Ejecuta capture con tshark -i 1 -w scenarioA.pcap -a duration:300 y simultáneamente extrae estadísticas:
 - Get-Counter -Counter "\Network Interface(*)\Bytes Total/sec" (PowerShell) o PerfMon logs.
- Usa **Clumsy** para introducir lag, drop, duplicate sobre la interfaz.
- Corre iperf3 (server en Linux o Windows) y varios clientes para generar load.
- Extrae RTP stats: Wireshark → RTP → Save as CSV o usar rtpdump/rtpstat.

2. Feature extraction (Windows):

- Convierte pcap → CSV con tshark -r file.pcap -T fields -e frame.time_epoch -e ip.src -e ip.dst -e udp.srcport -e udp.dstport -e frame.len -E header=y -E separator=, > packets.csv
- Post-process Python para agregados por ventana.

12. Explicabilidad y acciones recomendadas

- Usa **SHAP** para explicar predicciones y priorizar triggers (ej.: si top feature es utilization_95pct, recomendar policing o QoS).
- Define playbooks:
 - Si P(congestión) > 0.8 → priorizar tráfico VoIP (reclassify DSCP + apply shaping).
 - Si loss > 2% en Wi-Fi AP específico → cambiar AP channel or reduce airtime for best-effort flows.

13. Despliegue y operación

- **Modo edge:** un microservicio de inferencia (Flask/Gunicorn) consultando TSDB y devolviendo alertas.

- **Batch retrain:** cada N días con datos nuevos. Monitor de deriva (data drift).
- **CI/CD:** pipeline para reentrenamiento (Git, MLflow, DVC).

14. Riesgos y limitaciones

- Modelos entrenados con datos sintéticos pueden no generalizar perfectamente a producción Wi-Fi con interferencia RF real.
- Instrumentación limitada (no todos los equipos exponen cwnd o retransmits).
- Cambios de topología o políticas QoS invalidan los umbrales; necesario feedback continuo.

15. Ejemplo de outputs esperados

- Predicción: {'p_congestion': 0.92, 'pred_rtt_ms': 185.4, 'pred_jitter_ms': 42.1, 'pred_loss_pct': 2.6}
- Recomendación: Aplicar shaping a flows no-critical; priorizar DSCP EF en VoIP; escalar la notificación al NOC.

Parte 4– Proyecto integrador con IA Actividad 4.1 –

Estudio Una universidad planea implementar una red híbrida: • Ethernet (IEEE 802.3) para laboratorios de alta capacidad.

- WiFi (IEEE 802.11) para áreas comunes.
- Requisitos: o 5000 usuarios simultáneos.

Soporte a videoconferencia con <100 ms de latencia.

Protección contra ataques ICMP.

Priorización de tráfico académico sobre recreativo.

Desafíos:

1. Proponer un plan de direccionamiento IPv4 privado con subredes.

Campus global: 10.0.0.0/16

Área / Función	Netw ork	Prefij o	Hos ts útil es	VLAN ID (ej.)	Gatewa y (recom end.)	DHCP pool (ej.)	Observaciones

			aprox.				
Infra / Core / Routers / Mgmt	10.0.0.0	/24	254	10	10.0.0.1	10.0.0.10–10.0.0.200	Equipos de red, console servers, NMS
Servidores / Data center	10.0.1.0	/24	254	20	10.0.1.1	DHCP estático o reservas	Hosts de backend, DB, storage
DMZ / Perímetro	10.0.2.0	/24	254	30	10.0.2.1	Reservado	NAT/Firewall hacia Internet
Administración / Facultades	10.0.3.0	/22	1.022	40	10.0.3.1	10.0.3.10–10.0.3.1000	Oficinas, desktop del personal
Ethernet — Laboratorios (agrupado)	10.0.8.0	/21	2.046	50	10.0.8.1	Por /24 (ver abajo)	Bloque agregado para /24 por laboratorio
- Lab A (/24)	10.0.8.0	/24	254	51	10.0.8.1	10.0.8.10–10.0.8.200	Cada lab físico una /24
- Lab B (/24)	10.0.9.0	/24	254	52	10.0.9.1	10.0.9.10–10.0.9.200	...hasta 10.0.15.0/24
Wi-Fi académico (alta concurrencia)	10.0.16.0	/20	4.094	60	10.0.16.1	10.0.16.10–10.0.31.250	Diseñado para ≈2.500+ clientes simultáneos
Wi-Fi invitados / recreativo	10.0.32.0	/21	2.046	70	10.0.32.1	10.0.32.10–10.0.32.200	Captive portal / NAT y reglas restrictivas
VoIP / Videoconferencia	10.0.64.0	/23	510	80	10.0.64.1	10.0.64.10–10.0.65.250	QoS estricto, marcar DSCP, VLAN separada
WLAN management (APs, controllers)	10.0.68.0	/24	254	82	10.0.68.1	10.0.68.10–10.0.68.200	Direcciones fijas o reservas para APs
CCTV / IoT	10.0.69.0	/24	254	85	10.0.69.1	10.0.69.10–10.0.69.200	Restringir acceso a mgmt
Investigación / Laboratorios R&D	10.0.70.0	/24	254	90	10.0.70.1	10.0.70.10–10.0.70.200	Redes aisladas por seguridad
VPN / Acceso remoto	10.0.71.0	/24	254	95	10.0.71.1	10.0.71.10–10.0.71.250	Direcciones para clientes VPN
Monitoreo / Logging	10.0.72.0	/24	254	100	10.0.72.1	Reservado	Syslog, Prometheus, ELK
Reservas / Crecimiento	10.0.128.0	/18	16.382	—	—	—	Para expansión futura, nuevos edificios

2. Diseñar políticas de seguridad informática (firewall, segmentación de redes IP, IDS/IPS). Hay que hacer un prototipo (escenario pequeño con redes IEEE 802.3 e IEEE 802.11), donde se configuren VLMS e implemente (firewall, segmentación de redes IP, IDS/IPS).

Topología

Internet \leftrightarrow Firewall (pfSense) \leftrightarrow Switch L3 (SVI) \rightarrow Switches access + APs; IDS en SPAN/mirror.

VLANs y subredes (muy simple)

- VLAN 10 — MGMT — 10.0.0.0/24 (equipos de red, NMS).
- VLAN 20 — LAB_ETH — 10.0.8.0/24 (pc de laboratorio).
- VLAN 30 — WIFI_ACAD — 10.0.16.0/24 (alumnos académicos).
- VLAN 40 — WIFI_GUEST — 10.0.32.0/24 (invitados, NAT).
- VLAN 50 — VOIP_VC — 10.0.64.0/24 (videoconf/VoIP).

Reglas de firewall

- Allow: MGMT \rightarrow any (SSH, SNMP, NMS).
- Allow: VOIP_VC \rightarrow Internet (puertos UDP/TCP de VC) — marcar DSCP EF.
- Allow: WIFI_ACAD \rightarrow internal servers (HTTP/HTTPS/SSH) + Internet.
- Allow: LAB_ETH \rightarrow Internet + WIFI_ACAD recursos (según necesidad).
- Block: WIFI_GUEST \rightarrow any RFC1918 (bloquear acceso a redes internas).
- Deny all other inter-VLAN traffic.
- Default: deny any \rightarrow any.

Protección ICMP (simple)

- En firewall: bloquear ICMP echo-request desde Internet hacia redes internas.
- Permitir ICMP dentro del campus (vlan \rightarrow vlan) para diagnóstico.
- Añadir rate-limit: p.ej. 10 pings/minuto por IP desde Internet (o simplemente DROP si se prefiere).

IDS/IPS (mínimo viable)

- Colocar Suricata en modo IDS conectado a un puerto mirror del switch.
- Cargar reglas básicas Emerging Threats / ET Open.
- Monitor: enviar alertas a syslog/ELK.
- (Opcional) Para IPS: poner Suricata inline o usar NFQUEUE para bloquear firmas críticas (solo en pruebas).

3. Implementar mecanismos de calidad de Servicio, Quality Of Service , QoS. (ej. DSCP, colas de prioridad) en el prototipo (escenario de redes IEEE 802.3 e IEEE 802.11).

- Clasificación de tráfico (DSCP):

Videoconferencia (VLAN 50): marcar DSCP = 46 (EF).

Académico (VLAN 30): DSCP = 26 (AF31).

Recreativo (VLAN 40): DSCP = 0 (Best Effort).

Política de colas de prioridad:

Cola 1 (alta prioridad): DSCP EF → tráfico de videoconferencia.

Cola 2 (media): DSCP AF31 → tráfico académico.

Cola 3 (baja): DSCP 0 → tráfico recreativo o invitado.

Se reserva al menos un 20 % del ancho para la cola 1.

- Ethernet (IEEE 802.3):

Activar QoS por puerto y confiar en el marcado DSCP. interface vlan 50

```
mls qos trust dscp
interface FastEthernet0/10
  mls qos trust dscp
  policy-map PRIORIDAD
    class VOIP
      priority percent 20
    class ACADEMICO
      bandwidth percent 40
    class DEFAULT
      fair-queue
```

Aplicar la política al puerto de salida hacia el router/firewall: interface
GigabitEthernet0/1
service-policy output PRIORIDAD

Wi-Fi (IEEE 802.11):

- En el AP o controlador: usar WMM (Wi-Fi Multimedia) para mapear DSCP→AC.
- EF (46) → AC_VO (Voice).
- AF31 (26) → AC_VI (Video).

- 0 → AC_BE (Best Effort).
- Activar WMM en el SSID académico y de videoconferencia; limitar ancho del SSID invitado (ej. 5 Mbps).

Verificación:

- Capturar paquetes con Wireshark y revisar campo DSCP.
- Generar tráfico simultáneo de VC y descargas: la VC debe mantener < 100 ms de delay.
- Usar ping -Q 0xb8 (DSCP 46) para pruebas de prioridad.

4. Aplicación IA: solicitar a una herramienta de IA la generación de un esquema visual de la red y presente algunas características de gestión de red (velocidades de Tx, desempeños, errores, ataques, logs...entre otros).

1. Esquema visual generado por IA:

La herramienta analiza los dispositivos conectados (routers, switches, puntos de acceso WiFi y servidores) y crea un mapa con:

- Conexiones cableadas IEEE 802.3 (laboratorios).
- Cobertura inalámbrica IEEE 802.11 (áreas comunes).
- Segmentos de red definidos por VLAN y subredes.

2. Características de gestión de red mostradas por la IA:

- **Velocidades de transmisión (Tx/Rx):** indica el ancho de banda real de cada enlace (por ejemplo, 1 Gbps en Ethernet, 300 Mbps en WiFi).
- **Desempeño de red:** la IA mide latencia, jitter y pérdida de paquetes, alertando si supera los 100 ms requeridos para videoconferencias.
- **Errores detectados:** la herramienta detecta colisiones, desconexiones o interferencias inalámbricas.
- **Monitoreo de ataques y anomalías:** mediante análisis de tráfico, la IA identifica picos de ICMP o intentos de acceso no autorizado.
- **Registros (logs):** almacena y clasifica eventos críticos como bloqueos del firewall, alertas del IDS/IPS o congestión de red.

3. Beneficio principal:

La aplicación de IA permite **una supervisión proactiva** del estado de la red, ayudando a mantener la seguridad, la calidad de servicio (QoS) y el rendimiento general con mínima intervención manual.