Solución de problemas de rendimiento de TCP en Nexus 9000 (NX-OS)

Contenido

- Introducción
- Prerequisites
  - Requirements
  - Componentes Utilizados
- Antecedentes
  - Qué es TCP
  - Tres ventajas clave
  - Descripción general de la encapsulación TCP/IP
  - Encabezado Ethernet (IEEE 802.3)
  - Encabezado IP (IPv4)
  - Estructura de encabezado TCP
  - Opciones de TCP (Común 10)
  - Comportamiento de confirmación y secuencia de TCP (incluido SYN/FIN)
    - Ejemplo 1: SYN con datos (TCP Fast Open)
    - Ejemplo 2: FIN con datos (terminación de la conexión)
  - MSS y su relación con MTU
    - Cómo Funciona la Negociación MSS en el Protocolo TCP de Intercambio de Contactos Tridireccional
    - Regla clave: MSS es direccional
    - ¿Puede el origen enviar más carga de TCP que el MSS de destino?
    - Perspectiva práctica para la resolución de problemas
  - Tamaño de la ventana (control de flujo)
- Resolución de problemas del plano de datos TCP en Cisco Nexus 9000 (NX-OS)
  - Validación inicial (disponibilidad)
  - Identificación de la ruta de tráfico (interfaces)
    - Configuración de ELAM (escala de nube de Nexus 9300)
    - Referencia
  - Validación de nivel de interfaz
  - Estabilidad de ARP y routing
  - Verificación de que el Tráfico no es Punteado a la CPU
  - Determinación de la latencia de reenvío de paquetes
    - SPAN a CPU (captura de paquetes para el plano de datos)
    - Validación de límite de velocidad del plano de control
    - Validación basada en ICMP antes de TCP
    - Determinación de la latencia de reenvío del switch Nexus mediante captura de paquetes
    - Referencias
- Análisis del Tráfico TCP desde la Captura de Paquetes de Host de Origen
  - Análisis del protocolo de enlace de tres vías TCP
    - Identificación del tráfico

- Análisis del tiempo de recorrido de ida y vuelta (iRTT) inicial
- Identificación de puertos TCP
- Análisis del tamaño de la ventana TCP
- Análisis del rendimiento, el tiempo de transferencia y las condiciones necesarias
- Longitud del encabezado IP y TCP
- Análisis de opciones TCP y TTL

Análisis de RTT de TCP: ACK RTT frente a RTT inicial

Análisis de retransmisiones TCP y retransmisiones falsas

- Retransmisiones TCP en el tiempo
- Retransmisiones falsas de TCP

Análisis de rendimiento eficaz

Análisis de datos en vuelo (ventana TCP)

Análisis de carga útil TCP frente a MSS en el tiempo

Análisis de la causa raíz (RCA): Degradación del rendimiento de TCP

Conclusión

Solución

- Reflexión técnica

Introducción

Este documento describe los fundamentos de TCP, el análisis detallado de paquetes de Wireshark y la resolución de problemas práctica para optimizar el rendimiento de extremo a extremo.

Prerequisites

Requirements

Cisco recomienda que tenga conocimiento sobre estos temas:

- IP/TCP

Componentes Utilizados

La información que contiene este documento se basa en las siguientes versiones de software y hardware.

- Cisco Nexus 9000 Cloud Scale con Cisco NX-OS 10.6(X).

Nota: Cualquier pregunta sobre la configuración y la interoperabilidad del software o hardware de terceros queda fuera del soporte de Cisco. El uso de herramientas de terceros es un esfuerzo para demostrar su configuración y funcionamiento con los equipos de Cisco.

La información que contiene este documento se creó a partir de los dispositivos en un ambiente de laboratorio específico. Todos los dispositivos que se utilizan en este documento se pusieron en funcionamiento con una configuración verificada (predeterminada). Si tiene una red en vivo, asegúrese de entender el posible impacto de cualquier comando.

Antecedentes

Qué es TCP

El protocolo de control de transmisión (TCP) es un protocolo de capa de transporte fundamental que funciona en la capa 4 del modelo OSI y proporciona una entrega fiable, ordenada y verificada por error de un flujo de bytes entre aplicaciones que se comunican a través de una red IP.

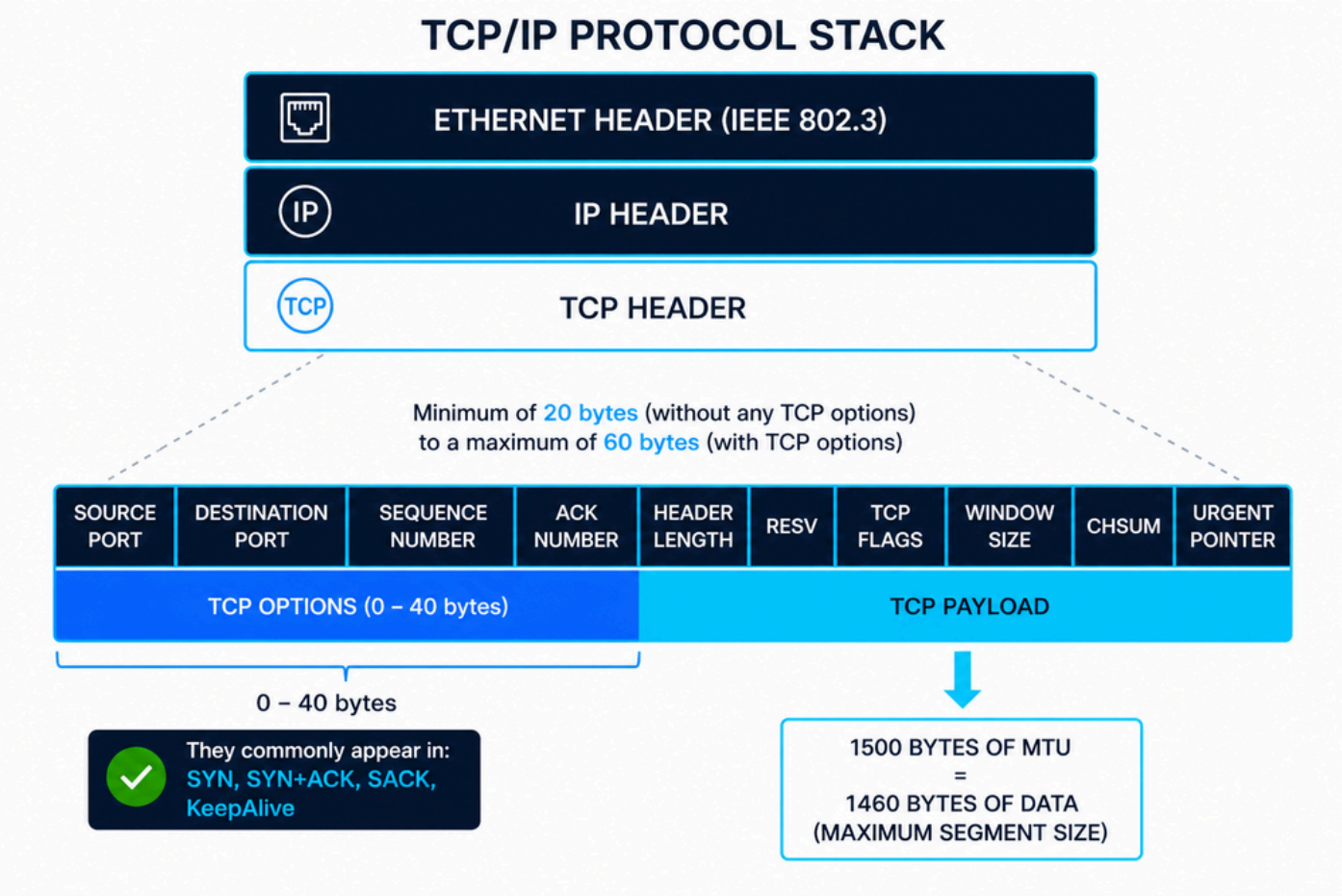
Tres ventajas clave

1. Fiabilidad: TCP está orientado a la conexión y garantiza la entrega al requerir confirmaciones del receptor. Si un paquete se pierde o se daña durante la transmisión, TCP retransmite automáticamente los datos para asegurarse de que lleguen al destino.
2. Entrega ordenada: Debido a que los paquetes de red pueden llegar desordenados, TCP asigna números de secuencia a cada segmento. Esto permite al sistema receptor volver a ensamblar los datos en el orden exacto en que se enviaron originalmente.
3. Control de congestión y flujo: TCP administra dinámicamente la velocidad de transmisión de datos para que coincida con la capacidad de procesamiento de los receptores y las condiciones de red actuales, evitando la pérdida de datos causada por los desbordamientos de búfer o la congestión de la red.

Descripción general de la encapsulación TCP/IP

El diagrama representa la pila TCP/IP en la que un segmento TCP (capa 4) se encapsula dentro de un paquete IP (capa 3) y, a continuación, dentro de una trama Ethernet (capa 2) definida por

IEEE 802.3. Este enfoque por capas garantiza la comunicación modular, en la que cada capa agrega su propia información de control (encabezados) para garantizar la entrega, el enrutamiento y la integridad de los datos.



Encabezado Ethernet (IEEE 802.3)

El encabezado Ethernet es generalmente de 14 bytes, compuesto por:

- Dirección MAC de destino (6 bytes)
- Dirección MAC de origen (6 bytes)
- EtherType/Length (2 bytes)

Además, las tramas Ethernet incluyen una cola de secuencia de comprobación de tramas (FCS) de 4 bytes para la detección de errores en la capa 2. IEEE 802.3 define la trama, los tamaños de trama mínimos y máximos y las restricciones de entrega física que afectan directamente a los protocolos de capa superior como TCP.

Encabezado IP (IPv4)

El encabezado IPv4 tiene un tamaño mínimo de 20 bytes, ampliable hasta 60 bytes con opciones. Los campos clave incluyen:

- Direcciones IP de origen y de destino
- Tiempo de vida (TTL)
- Protocolo (identifica TCP como carga)

La capa IP es responsable del direccionamiento lógico y el routing a través de las redes, pero no garantiza la fiabilidad.

Estructura de encabezado TCP

El encabezado TCP varía de 20 a 60 bytes según las opciones. Los campos clave incluyen:

- Puertos de origen/destino
- Número de secuencia
- Número de confirmación
- Indicadores (SYN, ACK, FIN, RST, etc.)
- Tamaño de ventana
- Checksum

TCP añade entrega fiable, secuenciación adecuada y control de flujo a la comunicación IP.

Opciones de TCP (Común 10)

Las opciones TCP amplían el protocolo base. Los más comunes incluyen:

1. Maximum Segment Size (MSS) (Tamaño máximo de segmento (MSS)): Define la carga TCP más grande que puede aceptar un host.
2. Window Scale - Extiende la ventana de recepción más allá de 65,535 bytes.
3. Reconocimiento selectivo permitido (SACK permitido): habilita la capacidad de reconocimiento selectivo.
4. Confirmación selectiva (SACK): especifica los bloques de datos recibidos para evitar retransmisiones completas.
5. Marcas de tiempo: se utilizan para el cálculo de RTT y la protección contra números de secuencia ajustados (PAWS).
6. Sin funcionamiento (NOP): relleno para la alineación de opciones.
7. End of Option List (EOL): marca el final de las opciones TCP.
8. TCP Fast Open (TFO): permite el intercambio de datos durante el intercambio de señales inicial.

9. Multipath TCP (MPTCP): habilita varias rutas de red para una sola sesión TCP.
10. Opción de tiempo de espera del usuario (UTO): controla el tiempo que los datos transmitidos pueden permanecer sin acuse de recibo.

Comportamiento de confirmación y secuencia de TCP (incluido SYN/FIN)

Tanto los indicadores SYN como FIN consumen cada uno un número de secuencia, incluso cuando no hay carga útil. TCP funciona mediante un modelo de secuenciación orientado a bytes, donde cada byte transmitido (y los indicadores de control específicos) avanzan en el espacio de la secuencia. Este comportamiento es esencial para un análisis preciso de TCP en capturas de paquetes y para diagnosticar inconsistencias de secuenciación o reconocimiento.

ACK = SEQ + Payload Length + (SYN ? 1: 0) + (FIN ? 1: 0)

Where:

- SEQ = Número de secuencia inicial
- Longitud de carga útil = Tamaño de datos en bytes
- SYN ? 1: 0 = Agrega 1 si se establece el indicador SYN; de lo contrario, 0
- FIN? 1: 0 = Agrega 1 si se establece el indicador FIN; de lo contrario, 0
- ACK = Próximo byte esperado

Ejemplo 1: SYN con datos (TCP Fast Open)

- SEQ = 1000
- SYN = 1
- Longitud de carga útil = 200 bytes

Cálculo ACK:

- ACK = 1000 + 200 + 1 + 0 = 1201

Esto refleja un escenario donde los datos se envían durante el intercambio de señales TCP. Tanto la carga útil como el indicador SYN consumen espacio de secuencia.

Ejemplo 2: FIN con datos (terminación de la conexión)

- SEQ = 3000

- FIN = 1
- Longitud de carga útil = 150 bytes

Cálculo ACK:

- ACK = 3000 + 150 + 0 + 1 = 3151

Esto muestra que TCP puede incluir datos durante el desmontaje de la conexión, y tanto la carga útil como el indicador FIN incrementan el número de secuencia.

MSS y su relación con MTU

El tamaño máximo de segmento (MSS) define la carga útil máxima que TCP puede enviar en un segmento.

- MTU Ethernet típica = 1500 bytes
- MSS = MTU − Encabezado IP − Encabezado TCP
- MSS estándar = 1460 bytes (1500 − 20 − 20)

Si hay opciones TCP, MSS se reduce en consecuencia. MSS se negocia durante el protocolo de enlace de tres vías TCP y evita la fragmentación en la capa IP.

Cómo Funciona la Negociación MSS en el Protocolo TCP de Intercambio de Contactos Tridireccional

El tamaño máximo de segmento (MSS) se intercambia durante el protocolo de enlace de tres vías TCP mediante la opción MSS en paquetes SYN:

- Host A → Host B (SYN): anuncia su MSS (por ejemplo, 1460)
- Host B → Host A (SYN-ACK): anuncia su MSS (por ejemplo, 1380)

Cada lado está diciendo efectivamente:

Esta es la carga útil TCP más grande aceptada.

Regla clave: MSS es direccional

MSS no se negocia como un único valor acordado.

En su lugar:

- Cada host utiliza el MSS anunciado por el otro lado.
- Esto crea dos límites independientes, uno por dirección.

Por lo tanto:

- A envía datos mediante MSS de B.
- B envía los datos mediante el MSS de A.

¿Puede el origen enviar más carga de TCP que el MSS de destino?

En una pila TCP que funcione correctamente: No.

- El remitente debe respetar el MSS anunciado por el receptor.
- El envío de segmentos más grandes supondría un riesgo:
  - Fragmentación de IP (si se supera la MTU)
  - Descartes de paquetes (si la fragmentación está bloqueada o no es compatible)
- Esto conduce a:
  - Retransmisiones
  - Degradación del rendimiento
  - Problemas como agujeros negros PMTUD (Path MTU Discovery)

Perspectiva práctica para la resolución de problemas

- Verifique siempre los valores MSS en el protocolo de enlace de tres vías TCP (paquetes SYN/SYN-ACK).
- Compruebe la existencia de discrepancias causadas por:
  - Túneles (VXLAN, GRE, IPsec)
  - Firewalls que modifican MSS (sujeción MSS)
- En plataformas como Cisco NX-OS, el ajuste de MSS se utiliza a menudo para evitar la fragmentación en rutas encapsuladas

Tamaño de la ventana (control de flujo)

El Tamaño de la ventana define la cantidad de datos que el receptor puede aceptar sin reconocimiento.

Qué es:

- Mecanismo de control de flujo para evitar el desbordamiento del búfer.

Propósito:

- Garantiza que el remitente no sobrecargue al receptor.

Dónde obtenerlo:

- Visible en capturas de paquetes (por ejemplo, Wireshark).
- Derivado de la configuración de la pila TCP del SO y del tamaño del búfer.

Variabilidad de proveedor/sistema operativo:

- Las distintas implementaciones (Linux, Windows, Cisco NX-OS) utilizan escalado dinámico y ajuste de búfer, lo que da lugar a distintos tamaños de ventana.

Condición de ventana cero:

- Cuando el tamaño de la ventana = 0, el búfer del receptor está lleno.
- El remitente detiene la transmisión y envía sondeos periódicos.

Mecanismos variables de Windows

- Control de flujo basado en velocidad
    - Asigna al remitente una velocidad de datos fija y garantiza que los datos nunca superen esa asignación.
    - Ideal para aplicaciones de streaming.
    - Entrega de difusión y multidifusión
- Control de flujo basado en ventana
    - El tamaño de la ventana varía con el tiempo.
    - El receptor logra el control de flujo mediante la señalización de la ventana permitida a las actualizaciones de la ventana del remitente.

Uso de Troubleshooting:

- Ventanas pequeñas o nulas → Cuellos de botella en el lado del receptor (CPU, memoria, aplicación).
- Ventanas grandes pero bajo rendimiento → Problemas de red (latencia, congestión).
- El análisis del comportamiento de la ventana es crítico para diagnosticar problemas de rendimiento en sesiones TCP.

Resolución de problemas del plano de datos TCP en Cisco Nexus 9000 (NX-OS)

En esta sección se describe una metodología práctica para diagnosticar si un switch Cisco Nexus con NX-OS está afectando al reenvío de tráfico TCP o si presenta problemas de rendimiento. El enfoque se presenta a través de un escenario hipotético.

Cuando se observa latencia de TCP o degradación del rendimiento, es común sospechar inicialmente que la red lo está causando. Sin embargo, esta suposición debe validarse a través de un análisis controlado por datos. El método autorizado para la resolución de problemas de TCP es la captura de paquetes, que se realiza idealmente:

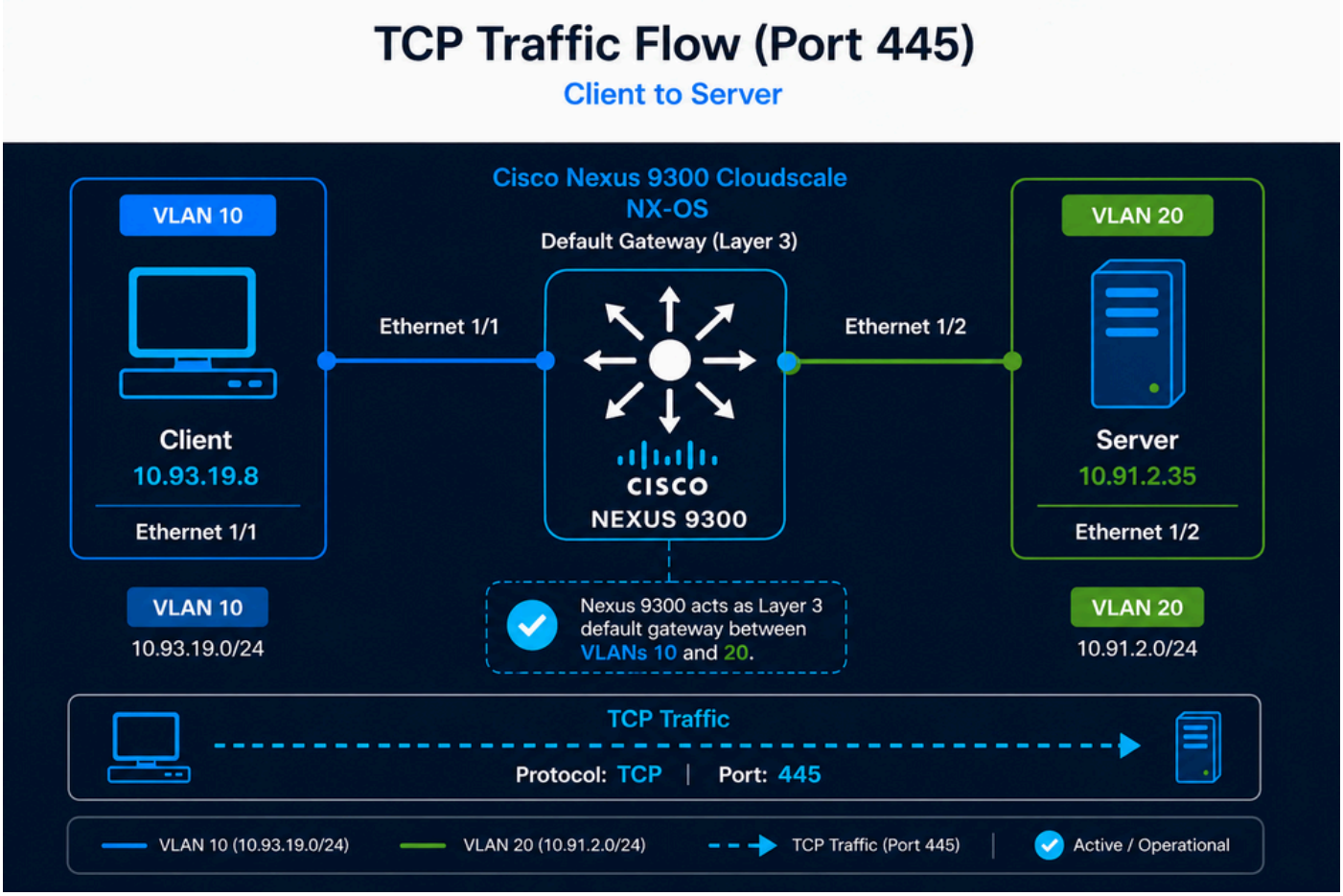
- Simultáneamente en origen y destino
- Antes del inicio del tráfico

Esto garantiza la visibilidad del protocolo de enlace de tres vías TCP, donde se negocian parámetros críticos como MSS, Window Scale y SACK y no se repiten más adelante en la sesión. Si no es posible realizar capturas simultáneas, el análisis puede continuar con una sola captura, pero las conclusiones son limitadas.

Definición de escenario

Un usuario ha identificado que el proceso de copia de seguridad de un conjunto de datos de aplicación de aproximadamente 7,5 TB, que anteriormente se completaba en unas 9 horas, ahora está tardando casi 21 horas. Aunque las sesiones TCP entre el cliente y el servidor se siguen estableciendo correctamente, el aumento significativo en la duración de la copia de seguridad sugiere una degradación potencial en el rendimiento o en el rendimiento general de TCP. Dado que el switch Nexus es el único dispositivo de red en la ruta y también proporciona funcionalidad de gateway de capa 3, el administrador de la red sospecha que el switch Nexus es la causa del problema.

- Cliente: 10.93.19.8 (VLAN 10)
- Servidor: 10.91.2.35 (VLAN 20)
- Nexus 9300 actúa como gateway predeterminado
- Puerto TCP 445



Validación inicial (disponibilidad)

- Estos comandos se utilizan para validar la MTU de ruta (PMTU) entre un origen y un destino mediante el envío de paquetes ICMP con el bit Don’t Fragment (DF) configurado. Esto ayuda a determinar el tamaño máximo de paquete que puede atravesar la red sin fragmentación. Este proceso debe realizarse tanto en el origen como en el destino.
- Verifique siempre la MTU de la interfaz física tanto en el origen como en el destino.
- En este escenario, el acceso está disponible solamente para el host de origen, donde se identificó una MTU de 1500.

```
Linux: ping -c 10 -I 10.93.19.8 -s 1472 -M do 10.91.2.35
```

- -c 10 → Envía 10 solicitudes de eco ICMP
- -I 192.168.10.10 → Utiliza esta IP/interfaz de origen específica
- -s 1472 → Establece el tamaño de carga útil de ICMP en 1472 bytes
- -M do → Establece el bit DF (no fragmentar)
- 192.168.20.20 → IP de destino

```
Windows: ping -n 10 -l 1472 -f 10.91.2.35
```

- -n 10 → Envía 10 solicitudes de eco ICMP
- -l 1472 → Establece el tamaño de carga útil de ICMP en 1472 bytes
- -f → Establece el indicador No fragmentar (DF)
- 192.168.20.20 → IP de destino

¿Por qué 1472 bytes?

- Carga útil ICMP = 1472 bytes
- encabezado IP = 20 bytes
- encabezado ICMP = 8 bytes
- Tamaño total de paquete: 1472 + 20 + 8 = 1500 bytes (MTU estándar)
- Esto prueba si la trayectoria soporta una MTU de 1500 bytes sin fragmentación. Si intenta enviar 1500 bytes de carga útil ICMP, el ping puede fallar porque el tamaño total del paquete excedería la MTU estándar después de agregar los encabezados IP e ICMP.

Qué se puede concluir

- Si el ping es exitoso (sin pérdida de paquetes), la trayectoria soporta al menos una MTU de 1500 bytes y no se requiere fragmentación.
    - Limpiar resultados ICMP → continuar con el análisis de TCP
    - Éxito de ping intermitente → posible pérdida de paquetes, congestión transitoria, limitación de velocidad o un problema de reenvío; continúe con el análisis de pérdida de paquetes, ya que TCP requiere una ruta sin pérdidas para funcionar de manera eficiente.
- Si el ping falla con el error "Fragmentation needed" o se agota el tiempo de espera, hay un link en la trayectoria con MTU inferior a 1500 bytes, el paquete no se puede reenviar debido al bit DF, y esto indica un problema de MTU de trayectoria.

Cómo utilizar esto para solucionar problemas

- Reduzca gradualmente el tamaño de la carga útil (por ejemplo, 1472 → 1400 → 1300) para identificar el tamaño más grande que se consigue.
- Una vez identificada, calcule la MTU mediante la fórmula MTU = carga + 28 bytes (encabezados IP + ICMP).

Relevancia práctica para TCP

- Si la MTU es menor de lo esperado, los segmentos TCP se pueden fragmentar o descartar.

- Esto conlleva retransmisiones, mayor latencia y menor rendimiento, lo que afecta directamente al rendimiento de las aplicaciones.

Identificación de la ruta de tráfico (interfaces)

Para solucionar de forma eficaz los problemas de rendimiento de TCP en un switch Cisco Nexus 9000, es esencial determinar qué interfaces reciben y reenvían el tráfico entre el origen y el destino.

En las topologías simples, esto se puede inferir directamente de las conexiones físicas. Por ejemplo, si el cliente está conectado a Ethernet1/1 y el servidor a Ethernet1/2, la trayectoria del tráfico es directa. Sin embargo, en entornos reales con varias interfaces activas, canales de puerto o configuraciones vPC, esta identificación no siempre es trivial.

En estos casos, el enfoque recomendado es utilizar el módulo analizador de lógica incorporada (ELAM), que proporciona visibilidad en el nivel ASIC (hardware de plano de datos).

ELAM le permite capturar un paquete mientras es procesado por la canalización de reenvío y revela información crítica como:

- Interfaz de entrada
- Interfaz de salida
- Decisión de reenvío (resultado de la búsqueda de capa 2 y 3)

Este método es significativamente más preciso que confiar en las herramientas del plano de control, ya que refleja la ruta de reenvío de hardware real.

Es importante tener en cuenta que ELAM captura sólo un paquete a la vez, por lo que los criterios de filtrado deben definirse con precisión para que coincidan con el tráfico deseado (por ejemplo, IP de origen, IP de destino, puerto TCP). Si los filtros son demasiado amplios, existe el riesgo de capturar tráfico no relacionado como ICMP o UDP en lugar del flujo TCP deseado.

Además, este proceso debe repetirse para ambas direcciones de tráfico:

- Origen → Destino
- Destino → Origen

En entornos que utilizan vPC o ECMP, el tráfico se puede equilibrar en función de la carga a través de varias rutas. Como resultado, el tráfico de reenvío y retorno puede atravesar diferentes switches o interfaces. En estos casos, la ELAM debe ejecutarse en cada switch Nexus relevante

para garantizar una visibilidad completa.

Al identificar con precisión las interfaces de entrada y salida, el alcance de la resolución de problemas se reduce significativamente, lo que permite una validación centrada de los contadores de interfaz, las políticas de QoS, la configuración de MTU y los puntos de congestión potenciales a lo largo de la ruta de reenvío exacta.

Configuración de ELAM (escala de nube de Nexus 9300)

Este ejemplo filtra el tráfico con la IP de origen 10.93.19.8, la IP de destino 10.91.2.35 y el puerto de destino TCP 445.

Configuración de ELAM

<#root>

```
switch#

debug platform internal tah elam


switch(TAH-elam)#

trigger init


Slot 1: param values: start asic 0, start slice 0, lu-a2d 1, in-select 6, out-select 0
switch(TAH-elam-insel6)#

set outer ipv4 src_ip 10.93.19.8


switch(TAH-elam-insel6)#

set outer ipv4 dst_ip 10.91.2.35


switch(TAH-elam-insel6)#

set outer l4 l4-type 0


switch(TAH-elam-insel6)#

set outer l4 dst-port 445
```

```
switch(TAH-elam-insel6)#

start
```

Después de generar el tráfico, recupere el resultado:

```
<#root>

switch(TAH-elam-insel6)#

report
```

Captura de tráfico inversa (obligatoria para una visibilidad completa)

Para validar la ruta de acceso de retorno, repita la configuración intercambiando las direcciones IP de origen y de destino:

```
<#root>

switch#

debug platform internal tah elam


switch(TAH-elam)# trigger init
Slot 1: param values: start asic 0, start slice 0, lu-a2d 1, in-select 6, out-select 0
switch(TAH-elam-insel6)#

set outer ipv4 dst_ip 10.93.19.8


switch(TAH-elam-insel6)#

set outer ipv4 src_ip 10.91.2.35


switch(TAH-elam-insel6)#

set outer l4 l4-type 0


switch(TAH-elam-insel6)#

set outer l4 dst-port 445
```

```
switch(TAH-elam-insel6)#

start
```

Notas operativas

- ELAM captura solo un paquete, así que asegúrese de que el tráfico fluye activamente al iniciar la captura.
- Los filtros deben ser precisos para evitar la captura de tráfico no relacionado.
- En entornos vPC, ejecute ELAM en ambos switches, ya que el tráfico puede hash de forma diferente en cada dirección.
- La salida muestra la interfaz de entrada, la interfaz de salida y la decisión de reenvío en el hardware, lo que proporciona una visibilidad autorizada en el plano de datos.

Referencia

Guía de ELAM ASIC de la escala de nube Cisco Nexus 9000

Validación de nivel de interfaz

La validación a nivel de interfaz garantiza que el switch Nexus no introduzca ninguna restricción o anomalía que afecte al tráfico TCP. El objetivo es confirmar que la configuración, el estado operativo y los contadores de hardware son coherentes con el comportamiento esperado para el reenvío del plano de datos de alto rendimiento.

Validación de configuración

- Verifique que no se apliquen ACL restrictivas a las interfaces:

<#root>

```
switch#

show running-config interface ethernet1/1-2 | include access-group
```

- Valide que ninguna política de QoS no intencionada esté afectando al tráfico (QoS global y

de nivel de interfaz, incluidas las colas, la regulación y el modelado):

```
<#root>
switch#
show running-config interface ethernet1/1-2 | include service-policy

switch#
show policy-map interface ethernet1/1-2
```

```
<#root>
switch#
show policy-map
```

```
<#root>
switch#
show class-map
```

```
<#root>
switch#
show class-map type network-qos
```

```
<#root>
switch#
show policy-map type network-qos
```

<#root>

switch#

show policy-map system type network-qos

<#root>

switch#

show queuing interface ethernet1/1-2

<#root>

switch#

show policy-map type queuing

- Confirme la configuración de Capa 2 o Capa 3 (switchport vs interfaz ruteada), incluida la pertenencia a VLAN, el estado STP y el direccionamiento IP:

<#root>

switch#

show running-config interface ethernet1/1-2

<#root>

switch#

show interface ethernet1/1-2 switchport

<#root>

switch#

show spanning-tree interface ethernet1/1-2

<#root>

switch#

show ip interface ethernet1/1-2

Validación del estado operativo

- Verifique la consistencia de MTU y asegúrese de que coincida con la configuración esperada (por ejemplo, 1500 o 9000 bytes):

<#root>

switch#

show interface ethernet1/1-2 | include MTU

- Confirme la configuración de dúplex y velocidad de la interfaz:

<#root>

switch#

show interface ethernet1/1-2 | include speed|duplex

- Validar la estabilidad de la interfaz (sin inestabilidad o transiciones de link frecuentes):

<#root>

switch#

show interface ethernet1/1-2 | include rate|flap

Validación de contadores de errores

- Borre los contadores antes de la prueba:

<#root>

```
switch#

 clear counters interface all
```

- Supervisar contadores de errores (sólo valores distintos de cero):

<#root>

```
switch#

 show interface counters errors non-zero | include Port|Eth1/1|Eth1/2
```

Validación posterior a la prueba

- Vuelva a ejecutar la prueba de tráfico TCP y observe los contadores de nuevo:

<#root>

```
switch#

 show interface counters errors non-zero | include Port|Eth1/1|Eth1/2
```

- Los contadores no deben incrementarse; cualquier aumento indica posibles problemas de capa 1 o relacionados con el hardware, como errores de enlace físico, errores CRC/FCS o desbordamientos/caídas de búfer.

Estabilidad de ARP y routing

Garantizar el routing y la estabilidad ARP es fundamental para confirmar que el switch Nexus tiene un alcance de capa 3 uniforme y no presenta problemas de resolución intermitentes que puedan afectar al rendimiento de TCP. La inestabilidad en las entradas de ruteo o en la resolución

ARP puede conducir a la pérdida de paquetes, al aumento de la latencia o al bloqueo del tráfico.

Criterios de validación

- Las entradas de enrutamiento de origen y destino deben estar presentes, ser estables y no cambiar con frecuencia.
- Las entradas ARP deben resolverse y no actualizarse continuamente o faltar.

<#root>

switch#

show ip route 10.93.19.8

<#root>

switch#

show ip route 10.91.2.35

<#root>

switch#

show ip arp detail | include 10.93.19.8

<#root>

switch#

show ip arp detail | include 10.91.2.35

Verificación de que el Tráfico no es Punteado a la CPU

En los switches Cisco Nexus 9000, el reenvío se realiza en el hardware (ASIC) y la CPU no participa en las operaciones normales del plano de datos. Por lo tanto, observar el tráfico TCP de host a host en el plano de control es anormal e indica que los paquetes se están impulsando

debido a excepciones o configuraciones erróneas. Una vez que la CPU debe procesar el tráfico, éste queda sujeto a Control Plane Policing, y se espera que se puedan observar caídas si el tráfico excede la velocidad del plano de control permitida.

Método de validación

- Capture el tráfico que llega al plano de control mediante Ethanalyzer:

<#root>

```
switch#

ethanalyzer local interface inband display-filter "ip.addr==10.93.19.8 and ip.addr==10.91.2.35" limit-ca
```

Comportamiento esperado

- No se puede observar tráfico de plano de datos TCP de host a host en la CPU.

Comportamiento inesperado

- Si los paquetes que coinciden con el flujo son visibles, el tráfico está siendo impulsado, lo que puede ser causado por:
  - Gestión excepcional de paquetes (vencimiento de TTL, registro de ACL, redireccionamiento)
  - Configuración incorrecta o funciones no admitidas
  - Programación de hardware incorrecta

Determinación de la latencia de reenvío de paquetes

La latencia de reenvío de paquetes en los switches Nexus 9000 depende del tamaño del paquete, el modo de reenvío y las funciones habilitadas. Las especificaciones de Cisco suelen hacer referencia a la latencia de los paquetes de 64 bytes en el reenvío por conexión directa.

Switch Model	ASIC / Architecture	Ports (example config)	Typical Forwarding Latency (64B packet)
Nexus 93180YC-EX	Cloud Scale (EX)	48x25G + 6x100G	~1.0 – 1.2 microseconds
Nexus 93180YC-FX	Cloud Scale (FX)	48x25G + 6x100G	~0.9 – 1.0 microseconds
Nexus 93180YC-FX2	Cloud Scale (FX2)	48x25G + 6x100G	~0.8 – 0.9 microseconds
Nexus 9364C	Cloud Scale	64x100G	~1.0 microsecond
Nexus 9336C-FX2	Cloud Scale (FX2)	36x100G	~0.8 microseconds

| Nexus 93240YC-FX2 | Cloud Scale (FX2) | 48x25G + 12x100G | ~0.8 – 0.9 microseconds
| Nexus 92300YC | Broadcom Trident II | 48x10/25G + 6x40/100G | ~2 – 3 microseconds
| Nexus 92160YC-X | Broadcom Tomahawk | 48x25G + 6x100G | ~2 microseconds
+---------------------+---------------------+-------------------------+-----------------------------

- Reenvío directo (valor predeterminado en Nexus 9000):
  - Comienza el reenvío antes de que se reciba el paquete completo.
  - Minimiza la latencia (desde menos de microsegundos hasta ~1 µs).
- Almacenamiento y retransmisión:
  - Se debe recibir el paquete completo antes del reenvío.
  - Agrega latencia proporcional al tamaño del paquete.

Las funciones adicionales pueden introducir una latencia incremental:

- Encapsulación/desencapsulación de VXLAN
- Búsquedas de ACL (procesamiento TCAM)
- Clasificación de QoS y colas
- Telemetría (NetFlow, ERSPAN, sFlow)
- Almacenamiento en búfer durante congestión

Sin embargo:

- Estas operaciones se realizan en canalizaciones de hardware.

El único escenario realista en el que la latencia aumenta considerablemente es la congestión:

- Los paquetes se almacenan en búfer en las colas de salida.
- El retraso depende de:
  - Profundidad de la cola
  - Utilización de interfaz
  - Políticas de QoS

Incluso en estos casos:

- La latencia suele estar en el rango de microsegundos a cientos de microsegundos.
- Una demora sostenida de milisegundos implicaría:
  - Congestión grave
  - Sobresuscripción
  - QoS o almacenamiento en búfer mal configurados

SPAN a CPU (captura de paquetes para el plano de datos)

Esto permite duplicar el tráfico del plano de datos en el plano de control para la captura de paquetes y la exportación a un archivo .pcapng, lo que permite un análisis detallado en Wireshark.

Configuración

```
monitor session 1
 source interface ethernet1/1 both
 source interface ethernet1/2 both
 destination interface sup-eth0
 no shut
```

Capturar ejecución

<#root>

```
switch#
```

ethanalyzer local interface inband mirror capture-filter "tcp port 445" limit-capture 0 write bootflash:

Consideraciones técnicas

- El tráfico reflejado en la CPU está sujeto a la política de plano de control (CoPP).
- Si el tráfico excede la CoPP:
  - Los paquetes sólo se pueden descartar en el plano de control.
  - Esto crea falsos positivos durante el análisis.
- SPAN a CPU se recomienda para escenarios de tráfico de bajo a moderado.
- Para entornos de alto rendimiento:
  - Utilizar SPAN local (analizador externo)
  - Utilice ERSPAN para la captura remota

Método	Ventaja	Limitación
TRAMO	Precisa, sin encapsulación	Requiere conexión física.
ERSPAN	Capacidad de captura remota	Susceptible a la congestión de la red.

Validación de límite de velocidad del plano de control

Para garantizar que las capturas de SPAN a CPU sean confiables, es necesario validar que el plano de control no está descartando paquetes reflejados debido a la limitación de velocidad.

Comando de validación

```
switch(config)# show hardware rate-limiter | i Allowed|span

Allowed, Dropped & Total: aggregated bytes since last clear counters

R-L Class     Config Allowed Dropped Total

span          50           0       0     0 <<<

span-egress   disabled     0       0     0
```

Metodología de validación

- Ejecute el comando a intervalos de ~3 segundos.
- Observe los contadores de caídas relacionados con SPAN.

Interpretación

- Ningún incremento en los contadores de caídas para la fila SPAN indica una captura confiable.
- El aumento de los contadores de caídas indica la pérdida de paquetes en el plano de control, lo que hace que la captura no sea fiable.

Si se observan caídas, el método de captura debe cambiarse a SPAN o ERSPAN.

Validación basada en ICMP antes de TCP

Las pruebas de ICMP proporcionan una validación de línea base de la integridad del plano de datos antes de realizar análisis de TCP complejos. Debido a que el ICMP no tiene estado y es más simple, permite una detección rápida de pérdida de paquetes, duplicación o inconsistencias de trayectoria.

Comportamiento esperado en la captura de SPAN

- Cada paquete ICMP puede aparecer dos veces:
    - Una vez en el ingreso

    - Una vez en la salida
- Para un ping estándar:
    - Solicitud de eco → 2 paquetes
    - Respuesta de eco → 2 paquetes

Esto confirma el reenvío correcto y la ausencia de pérdida de paquetes en el plano de datos.

Comportamiento anormal

- La falta de duplicados o recuentos asimétricos de paquetes indica una posible pérdida de paquetes o limitaciones de captura.
- Los tiempos de espera intermitentes sugieren problemas de Capa 1, congestión o problemas ascendentes.

Si el tráfico ICMP se reenvía constantemente sin pérdidas, existe una alta probabilidad de que el tráfico TCP también se reenvíe correctamente en la Capa 2/3.

Determinación de la latencia de reenvío del switch Nexus mediante captura de paquetes

Cuando el tráfico se captura usando SPAN a CPU (o SPAN/ERSPAN), cada paquete se puede observar dos veces: una vez en el ingreso y una vez en el egreso. Esta duplicación se puede utilizar para estimar la latencia de reenvío introducida por el switch Nexus calculando la diferencia de tiempo entre ambas instancias del mismo paquete.

En la práctica, esta latencia se puede medir utilizando el tráfico ICMP capturado anteriormente comparando el delta de tiempo entre los paquetes duplicados de solicitud de eco y respuesta de eco. Esto proporciona una línea de base sencilla y eficaz para el rendimiento de reenvío del switch. Si se requiere un análisis más profundo, se puede aplicar la misma metodología al tráfico TCP capturando el flujo y midiendo la diferencia de tiempo entre los paquetes TCP duplicados.

Metodología

- Identifique un paquete y su duplicado (mismo número de secuencia).
- Mida el delta de tiempo entre las copias de entrada y salida.
- Este delta representa una estimación del límite superior de la latencia de reenvío del switch, ya que puede incluir la sobrecarga de la duplicación y la marca de tiempo.

Configuración de Wireshark

- Habilitar visualización delta de tiempo:

```
View > Time Display Format > Seconds Since Previous Displayed Packet
```

- Agregue una columna personalizada para el delta de tiempo:

```
Right-click on "Time Delta from Previous Displayed Packet" → Apply as Column
```

- Filtrar tráfico relevante (ejemplo):

```
ip.addr==10.93.19.8 and ip.addr==10.91.2.35 and tcp
```

- Ordenar paquetes por número de secuencia o secuencia TCP:

```
Right-click packet → Follow → TCP Stream
```

Interpretación

- El delta de tiempo entre los paquetes duplicados puede estar en el rango de microsegundos.
  - Si este es el caso, el switch Nexus no introduce latencia en el reenvío de paquetes.
- Los bajos deltas constantes confirman el rendimiento de reenvío basado en hardware.
- Los deltas más altos o inconsistentes pueden indicar:
  - Congestión o almacenamiento en búfer

Referencias

- Hojas de datos de Cisco Nexus serie 9000
- Guías de diseño de switches Nexus de Cisco serie 9000
- Informe técnico sobre la gestión inteligente de búferes en los switches Nexus de Cisco serie 9000

Análisis del Tráfico TCP desde la Captura de Paquetes de Host de Origen

Esta sección proporciona una metodología detallada para analizar una captura de paquetes TCP en Wireshark, incluida la configuración del perfil, a través del caso hipotético descrito anteriormente. Las imágenes mostradas fueron tomadas directamente desde Wireshark. A modo de recordatorio, la situación es la siguiente:

Un usuario ha identificado que el proceso de copia de seguridad de un conjunto de datos de aplicación de aproximadamente 6,5 TB, que anteriormente se completaba en unas 9 horas, ahora tarda casi 21 horas. El único dispositivo de red accesible es un switch Cisco Nexus 9300 conectado al servidor de origen (10.93.19.8). La MTU configurada en la interfaz del switch es de 9000 bytes (tramas jumbo), mientras que la MTU en el servidor es desconocida. Hay disponible una captura de paquetes del servidor de origen y todos los pasos de validación de Nexus anteriores ya se han completado sin que se detecten anomalías.

Principales observaciones y limitaciones

- Se ha descartado el switch Nexus:
  - Sin pérdidas de paquetes
  - No hay errores de interfaz
  - Sin impacto de QoS o ACL
  - Reenvío de hardware confirmado
- Configuración de la interfaz:
  - Puerto de acceso
  - MTU: 9000 bytes
- Datos disponibles:
  - Captura de paquetes en origen
  - Conocimiento de MTU de extremo a extremo
    - El ping se completó correctamente sin fragmentación mediante un paquete de 1500 bytes con 1472 bytes de datos.
- Faltan datos:
  - Visibilidad de destino
  - No hay captura de paquetes disponible en el servidor de destino.

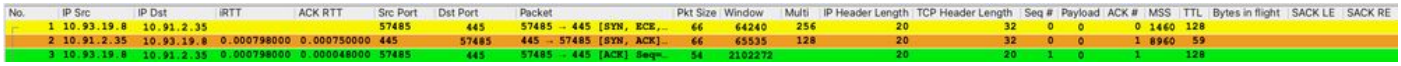
En Wireshark, puede crear perfiles personalizados adaptados al tipo específico de análisis que desee realizar.

Descripción de columna

- tcp.analysis.initial_rtt (iRTT): Calcula el tiempo de ida y vuelta inicial basándose en el protocolo de enlace de tres vías TCP.
- tcp.analysis.ack_rtt (ACK RTT): Mide el tiempo entre un segmento TCP y su correspondiente confirmación.
- tcp.window_size (Ventana): Indica el tamaño de la ventana TCP anunciado del receptor antes de aplicar la escala.
- tcp.options.wscale.multiplier (múltiple): Representa el factor de escala de ventana utilizado para calcular la ventana de recepción efectiva.
- tcp.seq (Nº de secuencia): Muestra el número de secuencia del primer byte del segmento TCP.
- tcp.len (carga útil): Muestra el tamaño de la carga útil TCP en bytes para ese segmento.
- tcp.ack (ACK#): Indica el siguiente byte esperado del remitente (confirmación acumulativa).
- tcp.options.mss_val (MSS): Muestra el tamaño máximo de segmento anunciado durante el protocolo de enlace TCP.
- ip.ttl (TTL): Muestra el valor de Tiempo de vida, útil para identificar el recuento de saltos y el comportamiento de routing.
- tcp.analysis.bytes_in_flight (Bytes en vuelo): Representa la cantidad de datos no reconocidos actualmente en tránsito.

Análisis del protocolo de enlace de tres vías TCP

Es obligatorio capturar el protocolo de enlace de tres vías TCP porque contiene parámetros críticos como MSS, Window Scale y SACK que definen cómo se comporta la sesión.
Sin esta información, cualquier análisis de TCP es incompleto y puede llevar a conclusiones incorrectas sobre el rendimiento o la causa raíz.



No.	IP Src	IP Dst	iRTT	ACK RTT	Src Port	Dst Port	Packet	Pkt Size	Window	Multi	IP Header Length	TCP Header Length	Seq #	Payload	ACK #	MSS	TTL	Bytes in flight	SACK LE	SACK RE
1	10.93.19.8	10.91.2.35			57485	445	57485 → 445 [SYN, ECE,…	66	64240	256	20	32	0	0	0	1460	128			
2	10.91.2.35	10.93.19.8	0.000798000	0.000750000	445	57485	445 → 57485 [SYN, ACK]…	66	65535	128	20	32	0	0	1	8960	59			
3	10.93.19.8	10.91.2.35	0.000798000	0.000048000	57485	445	57485 → 445 [ACK] Seq=…	54	2102272		20	20	1	0	1		128			

Identificación del tráfico

Desde la captura de paquetes:

- Dirección IP de origen: 10.93.19.8
- Dirección IP de destino: 10.91.2.35

Análisis del tiempo de recorrido de ida y vuelta (iRTT) inicial

El RTT inicial (iRTT) se calcula como:

- iRTT = 798 microsegundos

Este valor se deriva de:

- Paquete 2 (SYN-ACK) ACK RTT: 750 µs → Tiempo para que el destino responda al SYN.
- Paquete 3 (ACK) ACK RTT: 48 µs → Tiempo para que la fuente reconozca el SYN-ACK.

La mayor parte de la latencia (~94%) se encuentra en la ruta de reenvío (cliente → servidor → cliente), mientras que el tiempo de respuesta del origen es mínimo, lo que indica que no hay retraso de la CPU o de la aplicación en el cliente.

Identificación de puertos TCP

- Puerto TCP de destino: 445

El puerto 445 corresponde al Bloque de mensajes de Microsoft Server (SMB), que se utiliza habitualmente para compartir archivos, unidades de red y servicios de autenticación de Windows. Este protocolo es sensible tanto a la latencia como al rendimiento, por lo que depende en gran medida de la eficacia de TCP y de la estabilidad de la red.

Análisis del tamaño de la ventana TCP

- Ventana de origen (escalada): 64,240 bytes
- Ventana de destino: 65,535 bytes

La ventana TCP representa la cantidad de datos que se pueden enviar antes de esperar la confirmación. En este caso, el origen es ligeramente más restrictivo que el destino. Estos valores son relativamente pequeños para los entornos modernos y pueden limitar el rendimiento, especialmente a medida que aumenta el RTT.

El rendimiento teórico máximo se puede calcular utilizando:

Rendimiento = Tamaño de la ventana TCP/RTT

Sustitución de los valores observados:

- Tamaño de la ventana TCP = 64.240 bytes
- RTT = 798 microsegundos = 0,000798 segundos

Rendimiento ≈ 64 240 / 0,000798 ≈ 80,5 MB/s (~644 Mbps)

Esto representa el rendimiento del límite superior suponiendo que:

- Sin pérdida de paquetes
- No hay retransmisiones
- Condiciones de red ideales

Análisis del rendimiento, el tiempo de transferencia y las condiciones necesarias

Con el rendimiento actual de 644 Mbps, la transferencia de un archivo de 6,5 TB tarda aproximadamente 23,5 horas, lo que coincide con la degradación observada. Para lograr un intervalo de transferencia de 9 horas, el rendimiento debe aumentar hasta aproximadamente 1,68 Gbps, lo que requiere una ventana TCP más grande (~2,7 veces más) o un RTT significativamente inferior (~291 µs).

Con las condiciones actuales (ventana de 64 KB y ~798 µs RTT), no es posible alcanzar el objetivo de 9 horas, porque el rendimiento de TCP está limitado por el producto de retraso de ancho de banda. Sin aumentar el tamaño de la ventana o reducir la latencia, el protocolo no puede utilizar un ancho de banda disponible más alto, lo que hace que el objetivo sea inalcanzable.

Situación	Rendimiento de procesamiento	Tiempo de transferencia estimado (6,5 TB)	Ventana TCP requerida	RTT requerido
Estado actual	644 Mbps (~80,5 MB/s)	~23,5 horas	64 KB	798 µs
Objetivo (9 horas)	~1683 Mbps (~210 MB/s)	9 horas	~172 KB	~291 µs

Esto funcionaba anteriormente, lo que indica que se ha producido un cambio en la red, la aplicación, el origen o el destino. Es importante señalar que, únicamente sobre la base de este análisis inicial, ya se puede llegar a una conclusión significativa: con el tamaño actual de la ventana TCP y las condiciones de RTT, no es posible alcanzar el objetivo de 9 horas.

Las tablas muestran una comparación de cómo varía el rendimiento a medida que el tamaño de la ventana RTT y TCP aumenta o disminuye.

Impacto de RTT en el rendimiento (tamaño de ventana fijo = 64 240 bytes)

RTT	Rendimiento (MB/s)	Rendimiento (Mbps)
200 µs (0,0002 s)	~321 MB/s	~2.568 Mbps

RTT	Rendimiento (MB/s)	Rendimiento (Mbps)
798 µs (0,000798 s)	~80,5 MB/s	~644 Mbps
2 ms (0,002 s)	~32,1 MB/s	~257 Mbps
10 ms (0,01 s)	~6,4 MB/s	~51 Mbps

Impacto en el tamaño de la ventana TCP (RTT fijo = 798 µs)

Tamaño de la ventana TCP	Rendimiento (MB/s)	Rendimiento (Mbps)
16 KB (16 384 B)	~20,5 MB/s	~164 Mbps
64 KB (64 240 B)	~80,5 MB/s	~644 Mbps
256 KB (262.144 B)	~328 MB/s	~2624 Mbps
1 MB (1.048.576 KB)	~1314 MB/s	~10,5 Gbps

Interpretación técnica

- El rendimiento es inversamente proporcional al RTT → una mayor latencia reduce el rendimiento.
- El rendimiento es directamente proporcional al tamaño de la ventana TCP → ventanas más grandes aumentan la capacidad.
- El tamaño reducido de las ventanas limita considerablemente el rendimiento, incluso en entornos de baja latencia.
- Las redes de alta velocidad (10 G+) requieren la ampliación de la ventana para aprovechar al máximo el ancho de banda.

Esto demuestra que el tamaño de la ventana RTT y TCP son factores críticos en el rendimiento TCP y deben analizarse conjuntamente al resolver problemas de rendimiento.

Longitud del encabezado IP y TCP

- Longitud del encabezado IP: 20 bytes
- Longitud del encabezado TCP: 32 bytes

Un encabezado IP de 20 bytes indica que no hay opciones IP presentes. El encabezado TCP de

32 bytes confirma que se están usando las opciones TCP, agregando 12 bytes más allá del encabezado base. Estas opciones suelen incluir MSS, Windows Scale y SACK Permitted.

Análisis de opciones TCP y TTL

La confirmación selectiva (SACK) está habilitada en ambos terminales. Esto no se ve en la imagen. SACK permite al receptor confirmar bloques de datos no contiguos, informando al remitente exactamente qué segmentos se recibieron con éxito.

Por ejemplo, si se reciben los segmentos 1000-2000 y 3000-4000 pero falta 2000-3000, el receptor puede indicarlo explícitamente. Sin SACK, el remitente retransmitiría todos los datos después de la brecha; con SACK, solo se retransmite la parte que falta. Esto mejora significativamente el rendimiento en entornos con pérdida de paquetes.

Análisis de paquete 1 (SYN)

- Nº de secuencia: 0 (Wireshark normalizado)
- Carga útil: 0 bytes
- Nº DE ACK: 0
- MSS: 1460 bytes
- TTL: 128

Wireshark normaliza el número de secuencia a cero para facilitar la lectura, aunque en la práctica es un valor aleatorio grande. Se espera la ausencia de carga útil durante el establecimiento de la conexión. El valor MSS de 1460 bytes indica una MTU de 1500 bytes (encabezado IP de 20 bytes + encabezado TCP de 20 bytes). Un TTL de 128 puede ser un host basado en Windows, y ver este valor en la captura indica que la captura probablemente se tomó en o muy cerca del origen a través de la Capa 2.

Análisis de paquete 2 (SYN-ACK)

- Nº DE ACK: 1

El valor ACK es 1 porque el indicador SYN consume un número de secuencia, incluso cuando no hay carga útil presente. Por lo tanto, ACK = SEQ + 1.

- TTL: 59

El TTL observado de 59 sugiere un TTL inicial de 64, lo que significa que el paquete atravesó aproximadamente 5 saltos de ruteo (64 − 59 = 5). Cada salto ruteado disminuye el TTL en uno.

Riesgo de fragmentación e impacto en la red

La presencia de aproximadamente cinco saltos de ruteo introduce riesgos potenciales de rendimiento, especialmente relacionados con las discordancias y fragmentación de MTU.

Si cualquier link intermedio tiene una MTU menor que el tamaño del paquete original, puede ocurrir la fragmentación. Esto conlleva varias consecuencias:

- Mayor latencia debido a la sobrecarga de fragmentación y reensamblado.
- Mayor probabilidad de pérdida de paquetes, ya que la pérdida de un solo fragmento requiere la retransmisión de todo el paquete.
- Menor rendimiento, ya que TCP interpreta la pérdida como congestión y reduce su velocidad de envío.
- Mayor utilización de la CPU en dispositivos de red que gestionan la fragmentación.
- Riesgo de errores de detección de MTU de ruta (PMTUD) si se bloquea el ICMP, lo que provoca caídas de paquetes silenciosas.

Teniendo en cuenta estos factores, es fundamental garantizar una MTU uniforme a lo largo de la ruta o implementar una sujeción MSS cuando sea necesario.

Análisis de RTT de TCP: ACK RTT frente a RTT inicial

Cuando ACK RTT es mayor que iRTT, indica que la latencia ha aumentado en comparación con la línea de base establecida durante el intercambio de señales TCP.

Esto significa que la red o los terminales están introduciendo un retraso adicional durante la sesión, debido normalmente a:

- Colas o congestión de red
- Retrasos del receptor o del procesamiento de aplicaciones
- Dispositivos intermedios (firewalls, equilibradores de carga)
- Retransmisiones

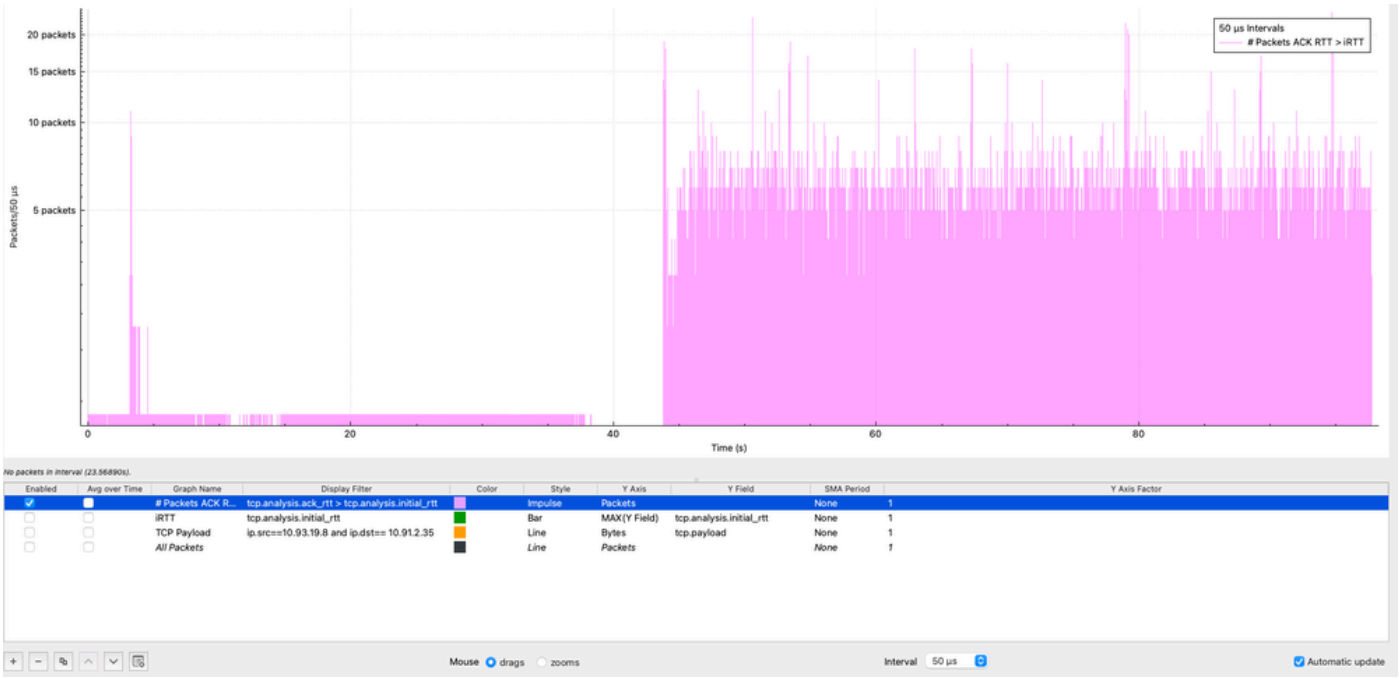
Si esta condición persiste durante la sesión TCP, conduce a:

- Rendimiento TCP reducido
- Utilización ineficiente de la ventana
- Rendimiento de las aplicaciones disminuido

En Wireshark, es posible visualizar la frecuencia con que ocurre la condición ACK RTT > iRTT

mediante la función de gráficos de E/S en: Estadísticas → Gráficos de E/S, aplicación del filtro de visualización (tcp.analysis.ack_rtt > tcp.analysis.initial_rtt), selección del estilo de impulso, configuración del eje Y en paquetes y uso de un intervalo de 50 microsegundos.

En el gráfico, los impulsos morados representan el número de paquetes que cumplen esta condición dentro de cada intervalo de 50 microsegundos. Como se ha observado, esta condición persiste durante toda la captura de paquetes, lo que indica que la latencia durante la sesión es constantemente más alta que la línea de base inicial. Este comportamiento sugiere claramente una degradación del rendimiento sostenida en lugar de una condición transitoria, lo que refuerza la necesidad de investigar fuentes potenciales como la congestión, el almacenamiento en búfer o los retrasos en el procesamiento de terminales a través de la ruta de extremo a extremo.



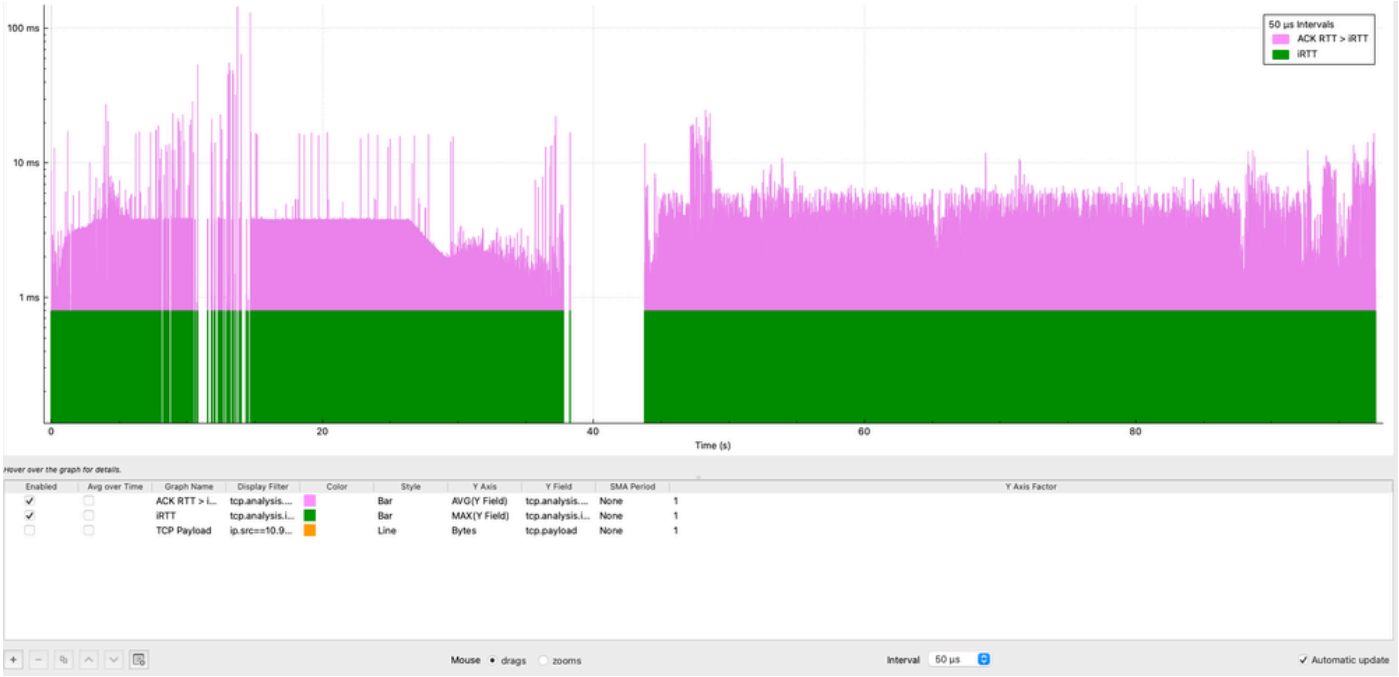
También es importante determinar por cuánto tiempo se excede el iRTT, no solo con qué frecuencia. Aunque Wireshark no permite directamente la resta entre campos, se puede lograr una comparación visual utilizando gráficos de E/S:

- Vaya a Estadísticas → Gráficos de E/S
- Gráfico 1:
  - Filtro de visualización: tcp.analysis.ack_rtt > tcp.analysis.initial_rtt
  - Estilo: Barra
  - Eje Y: PROMEDIO
  - Campo Y: tcp.analysis.ack_rtt
  - Intervalo: 50 microsegundos
- Gráfico 2:
  - Filtro de visualización: tcp.analysis.initial_rtt
  - Estilo: Barra
  - Eje Y: MAX

    - Campo Y: tcp.analysis.initial_rtt
- A continuación, haga clic con el botón derecho en el gráfico y habilite Escala de registro.

En esta visualización, el gráfico morado representa la condición ACK RTT > iRTT, que está presente consistentemente a lo largo de toda la sesión TCP. Los datos muestran una inflación de latencia sostenida, con picos múltiples que alcanzan los 11 milisegundos y un pico máximo de más de 100 milisegundos, lo que representa de 11x a 100x el iRTT de línea de base.

Este comportamiento confirma que el aumento de latencia no es transitorio, sino persistente, lo que indica un problema sistémico que afecta a la sesión a lo largo del tiempo. Esta desviación sostenida sugiere factores como la congestión de la red, el almacenamiento en búfer (bufferbloat) o los retrasos en el procesamiento de los terminales.



Análisis de retransmisiones TCP y retransmisiones falsas

Esta sección evalúa la confiabilidad de TCP analizando las retransmisiones a lo largo del tiempo, permitiendo la validación de si la pérdida de paquetes está contribuyendo a la degradación del rendimiento.

Retransmisiones TCP en el tiempo

El gráfico muestra la distribución de las retransmisiones TCP a lo largo del tiempo. Se observaron un total de 42 retransmisiones, que representaban solo el 0,00125% del tráfico total.

Este nivel de retransmisiones es insignificante e indica claramente que la pérdida de paquetes no es un factor que contribuya en esta situación.

Configuración de Wireshark (retransmisiones TCP)

```
Statistics → I/O Graphs
```

- Filtro de visualización:

```
tcp.analysis.retransmission and !tcp.analysis.spurious_retransmission
```

- Estilo: Impulso o barra
- Eje Y: Paquetes
- Intervalo: 1 seg.

Retransmisiones falsas de TCP

El gráfico muestra el número de retransmisiones falsas de TCP en intervalos de 1 segundo generadas por el origen 10.93.19.8.

En Wireshark, una retransmisión espuria de TCP indica que un host retransmitió un segmento que no se perdió realmente. El paquete original llegó correctamente al receptor, pero el remitente asumió incorrectamente la pérdida debido a una estimación inexacta del tiempo. Este comportamiento no indica una pérdida real de paquetes, sino una lógica de retransmisión ineficiente en el remitente.

En esta captura:

- El origen 10.93.19.8 retransmite los paquetes después de solo ~8 microsegundos.
- Mientras que los temporizadores de retransmisión típicos están en el orden de ~200 milisegundos.

Esto confirma que el comportamiento de retransmisión es controlado enteramente por la pila TCP de origen, no por la red.

El número total de retransmisiones falsas observadas es de 1112, lo que representa el 0,0332% del tráfico total capturado.

Configuración de Wireshark (retransmisiones falsas de TCP)

`Statistics → I/O Graphs`

- Filtro de visualización:

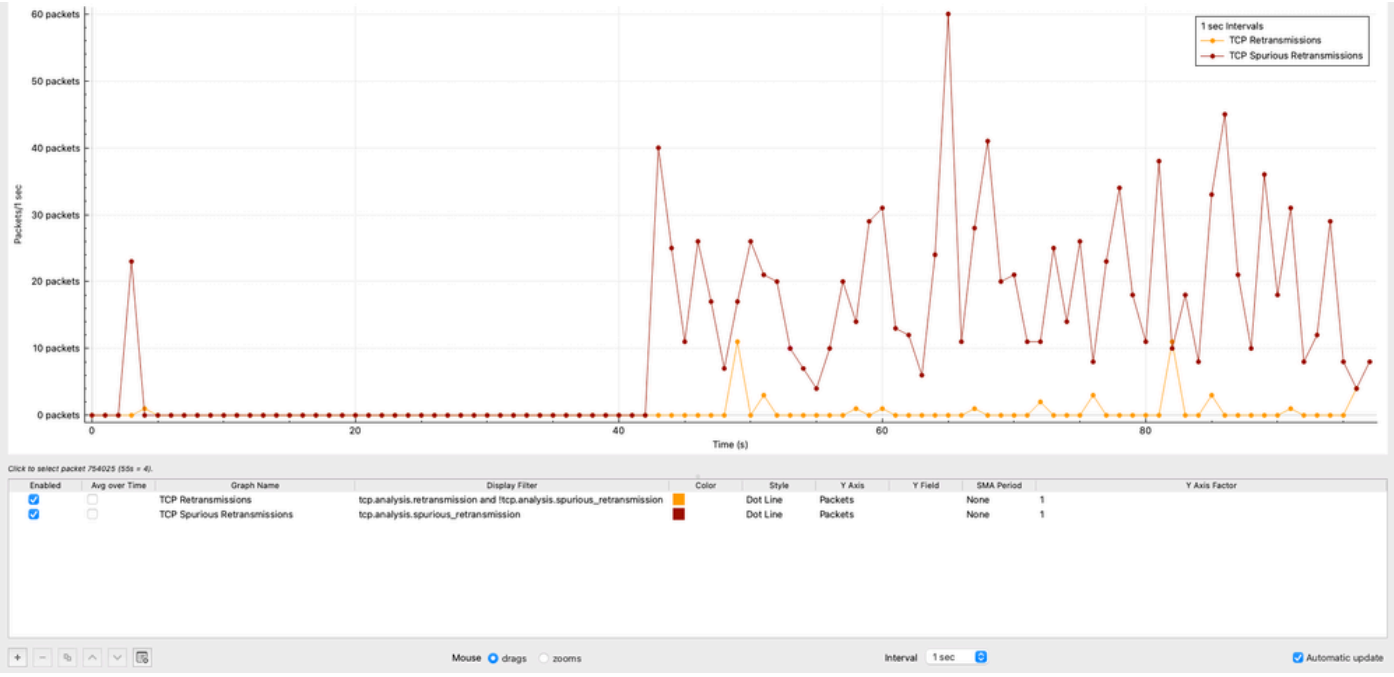
```
tcp.analysis.spurious_retransmission and ip.src==10.93.19.8
```

- Estilo: Impulso o barra
- Eje Y: Paquetes
- Intervalo: 1 seg.

Interpretación técnica

- El porcentaje extremadamente bajo de retransmisiones reales confirma que la pérdida de paquetes no está presente en la red.
- La presencia de retransmisiones falsas indica decisiones de retransmisión prematuras por parte del host de origen.
- Este comportamiento puede afectar ligeramente a la eficacia, pero no es una causa principal de la degradación grave del rendimiento.

Este análisis refuerza aún más que el problema no está relacionado con la fiabilidad de la red, sino con el comportamiento de TCP, la latencia o el rendimiento de los terminales.

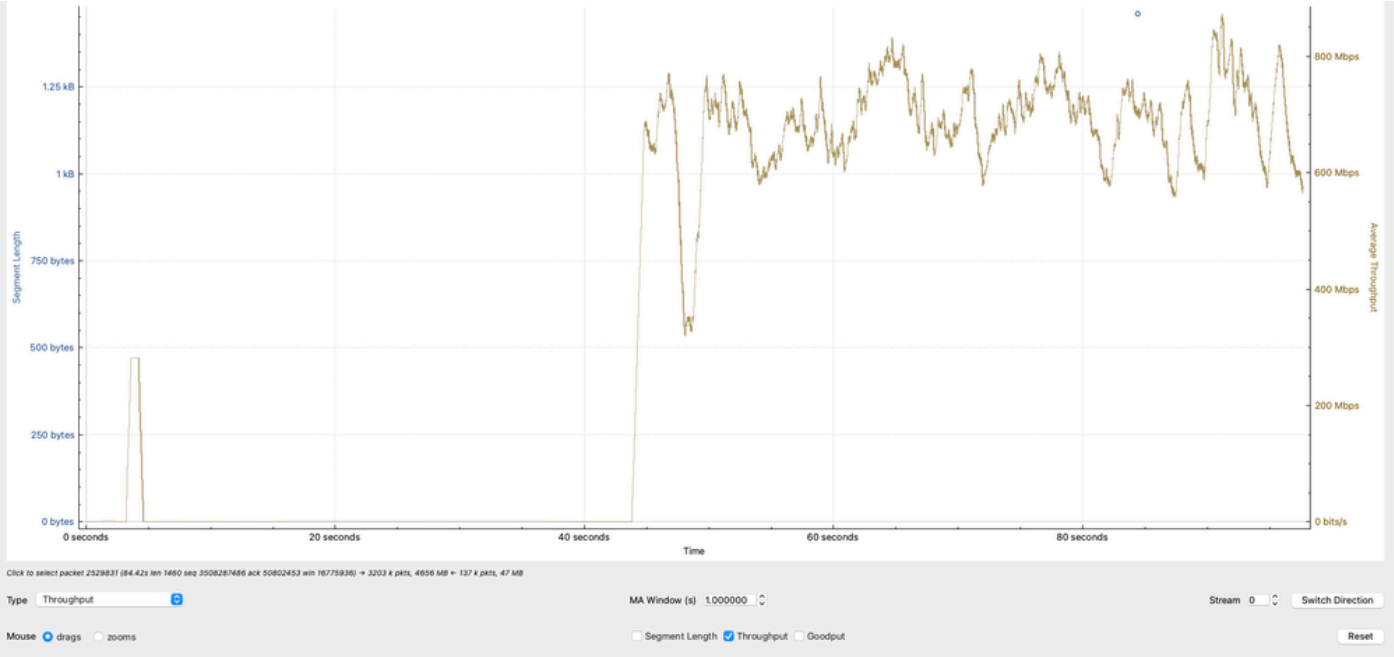


Análisis de rendimiento eficaz

El gráfico muestra el rendimiento efectivo, calculado en función de la carga útil de TCP (datos reales transferidos) en megabits por segundo. El rendimiento observado oscila principalmente entre 600 Mbps y 800 Mbps, lo que indica que, aunque la red transfiere datos de forma activa, no está alcanzando un mayor potencial de ancho de banda.

Configuración De Wireshark (Rendimiento Eficaz)

`Statistics → TCP Streams Graphs → Throughout`



Interpretación técnica

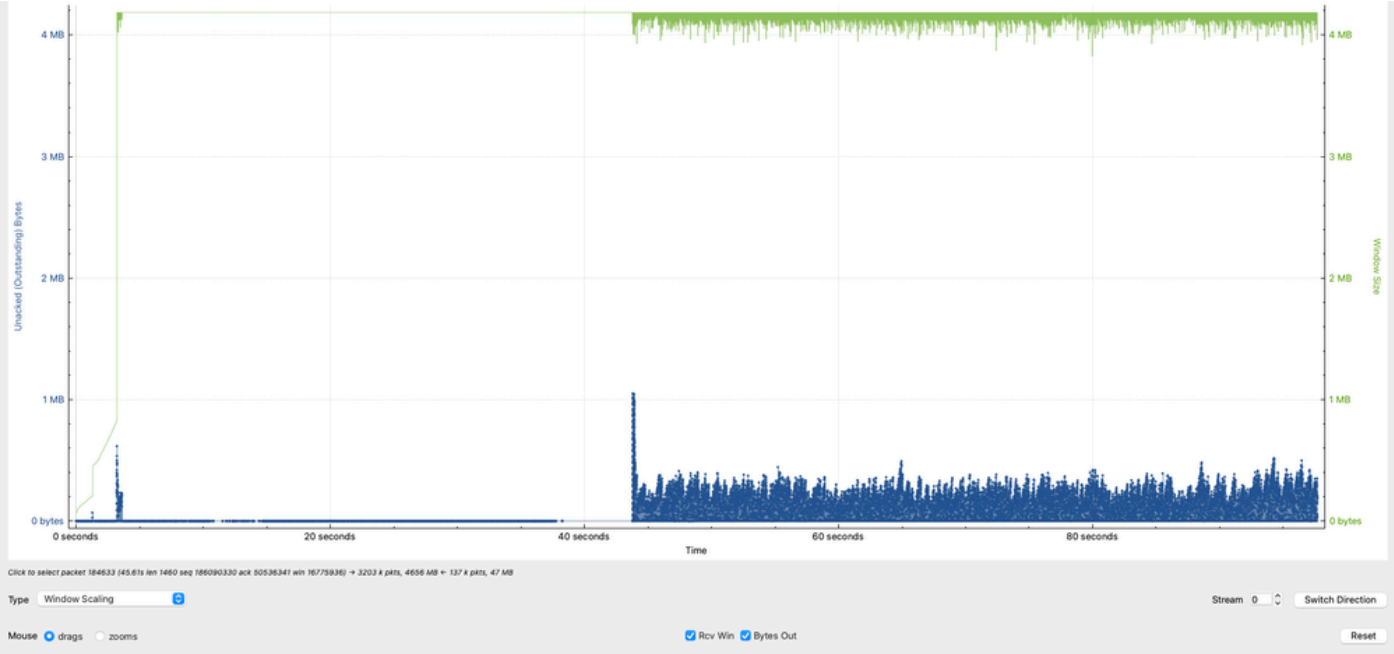
- El rango de rendimiento de 600-800 Mbps se alinea con los cálculos anteriores basados en el tamaño de la ventana TCP y RTT.
- La variabilidad en el rendimiento refleja:
  - Fluctuaciones de RTT
  - Ajustes de control de congestión TCP
  - Ritmo o almacenamiento en búfer de aplicaciones
- Dado que el rendimiento no se aproxima a la velocidad de línea (por ejemplo, 10 G), la limitación no es el ancho de banda físico, sino más bien las restricciones de eficacia de TCP.
- Este análisis confirma que el rendimiento observado es coherente con las limitaciones de TCP (tamaño y latencia de la ventana), lo que refuerza que el cuello de botella no se debe a

la pérdida de paquetes o a la capacidad de la interfaz, sino al comportamiento de la capa de transporte y a las condiciones de los terminales.

Análisis de datos en vuelo (ventana TCP)

El gráfico resalta un comportamiento crítico en la sesión TCP al comparar la capacidad del receptor frente a los datos reales en tránsito (bytes en vuelo).

- La línea verde representa la cantidad de datos TCP que 10.91.2.35 (receptor) puede aceptar (ventana de recepción efectiva).
- La línea azul representa la cantidad de datos TCP actualmente en vuelo desde 10.93.19.8 (remitente).



Los datos observados en vuelo alcanzan picos de aproximadamente 1 MB, con picos adicionales de alrededor de 8 KB y 5 KB, pero se concentran principalmente entre 1 KB y 250 KB.

Esto indica que aunque el receptor es capaz de manejar grandes volúmenes de datos, el remitente no está utilizando consistentemente la ventana disponible.

Configuración de Wireshark (datos en vuelo frente a ventana)

```
Statistics → TCP Streams Graphs → Throughout
```

Interpretación técnica

- El receptor (10.91.2.35) anuncia una ventana significativamente más grande, lo que indica que es capaz de recibir más datos.
- El remitente (10.93.19.8) está infrautilizando la ventana disponible, como lo muestran los datos inferiores e incoherentes en los valores de Vuelo.
    - Idealmente, el remitente puede mantener los valores de Datos en vuelo más cerca de la ventana de los receptores anunciados (~1 MB) para maximizar el rendimiento.
    - La incapacidad de mantener altos niveles de datos en vuelo limita directamente el rendimiento y es un fuerte indicador de la ineficiencia de TCP en el origen, no un problema de capacidad de red.

Análisis de carga útil TCP frente a MSS en el tiempo

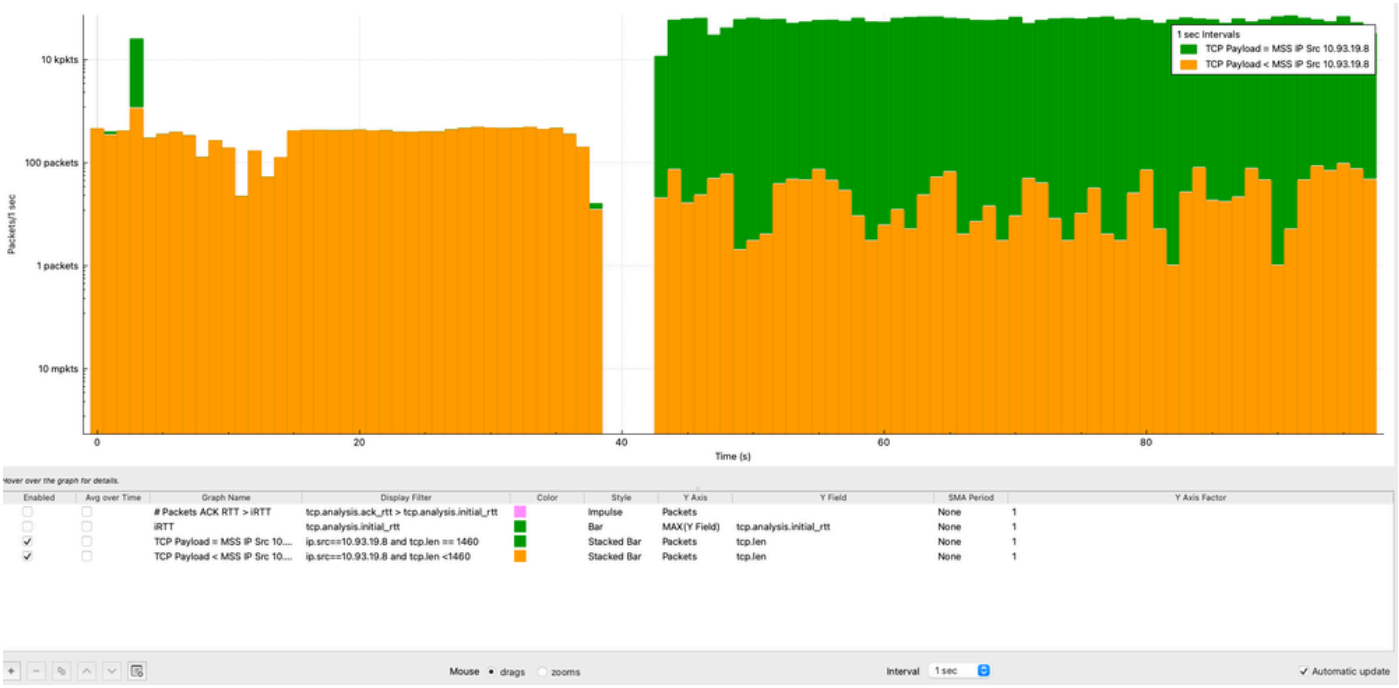
El análisis del tamaño de la carga útil de TCP comparándolo con MSS a lo largo del tiempo ayuda a determinar si el remitente está utilizando eficientemente cada segmento de TCP. Este análisis se realiza desde la perspectiva de la dirección IP de origen (10.93.19.8).

En Wireshark, los gráficos se configuran de la siguiente manera:

- Gráfico 1 (paquetes de tamaño MSS):
    - Filtro de visualización: ip.src==10.93.19.8 y tcp.len == 1460
    - Estilo: Barra apilada
    - Eje Y: Paquetes
    - Intervalo: 1 segundo
- Gráfico 2 (todos los paquetes ≤ MSS):
    - Filtro de visualización: ip.src==10.93.19.8 y tcp.len <= 1460
    - Estilo: Barra apilada
    - Eje Y: Paquetes
    - Intervalo: 1 segundo
- Aplicar escala logarítmica para una mejor visualización

Del análisis:

- La mayoría de los paquetes (>10 000 paquetes por segundo) alcanzan de forma coherente el valor de MSS de 1460 bytes.
- Una parte más pequeña de los paquetes transporta menos carga útil debido al comportamiento normal de TCP (ACK, segmentación o datos de fin de flujo).



Análisis de la causa raíz (RCA): Degradación del rendimiento de TCP

Este análisis demuestra que la identificación de la causa raíz de los problemas de rendimiento de TCP requiere un enfoque integral, en lugar de asumir que la red es la principal fuente de degradación.

Se llevó a cabo una validación exhaustiva en el switch Cisco Nexus 9300, incluidos los contadores de interfaz, las políticas de QoS, el routing y la estabilidad ARP, la verificación de punt de CPU, la captura de paquetes basada en SPAN y la validación de reenvío de nivel ASIC mediante ELAM. Todos los resultados confirmaron consistentemente que el switch estaba funcionando dentro de los parámetros esperados:

- Sin pérdidas de paquetes
- Sin latencia anormal (rango de microsegundos)
- Sin impacto de QoS o plano de control
- Reenvío de hardware correcto

Además, el análisis de TCP reveló:

- Retransmisiones insignificantes (0,00125%)
- Sin evidencia de pérdida de paquetes
- Uso coherente de MSS en la fuente
- Rendimiento alineado con la ventana TCP y las restricciones RTT
- Infrautilización de la ventana TCP disponible (análisis de datos en vuelo)
- La red no es el cuello de botella

- El servidor de origen limita el rendimiento

Conclusión

La degradación del rendimiento se debe a que el servidor de origen funciona con MTU 1500 en un entorno con capacidad Jumbo, lo que impide un uso eficiente de la capacidad de red disponible.

Solución

Aumente la MTU en el servidor de origen de 1500 a 9000 bytes para alinearlo con el destino y la infraestructura de red. Las ventajas:

- Habilitar segmentos TCP más grandes
- Reducir la sobrecarga de paquetes
- Mejorar el rendimiento general

Reflexión técnica

Una conclusión clave de este análisis es la importancia de evitar conclusiones prematuras a la hora de solucionar problemas de rendimiento de red. Si bien es común atribuir inicialmente problemas a la red, este caso demuestra claramente que la red funcionaba correctamente a lo largo de toda la ruta del plano de datos. Solo mediante un análisis profundo de TCP desde las perspectivas de origen y de destino (incluidos los parámetros de entrada en contacto, el comportamiento de RTT, la utilización de la ventana, las retransmisiones y la eficiencia de la carga útil) fue posible identificar con precisión el verdadero cuello de botella.

Dedicar tiempo a analizar detalladamente el comportamiento de TCP evita errores de diagnóstico, reduce los cambios innecesarios en la red y garantiza que los esfuerzos de remediación se dirijan a la causa real.

Acerca de esta traducción
Cisco ha traducido este documento combinando la traducción automática y los recursos humanos a fin de ofrecer a nuestros usuarios en todo el mundo contenido en su propio idioma.

Tenga en cuenta que incluso la mejor traducción automática podría no ser tan precisa como la proporcionada por un traductor profesional.

Cisco Systems, Inc. no asume ninguna responsabilidad por la precisión de estas traducciones y recomienda remitirse siempre al documento original escrito en inglés (insertar vínculo URL).