Risoluzione dei problemi relativi alle prestazioni TCP su Nexus 9000 (NX-OS)

Sommario

Introduzione

In questo documento vengono descritti i concetti fondamentali di TCP, l'analisi approfondita dei pacchetti di Wireshark e la risoluzione pratica dei problemi per ottimizzare le prestazioni end-to-end.

Prerequisiti

Requisiti

Cisco raccomanda la conoscenza dei seguenti argomenti:

- IP/TCP

Componenti usati

Le informazioni fornite in questo documento si basano sulle seguenti versioni software e hardware:

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

Nota: Qualsiasi domanda sulla configurazione e l'interoperabilità di software o hardware di terze parti non rientra nel supporto Cisco. L'uso di strumenti di terze parti è il modo migliore per dimostrare la configurazione e il funzionamento dell'azienda con le apparecchiature Cisco.

Le informazioni discusse in questo documento fanno riferimento a dispositivi usati in uno specifico ambiente di emulazione. Su tutti i dispositivi menzionati nel documento la configurazione è stata ripristinata ai valori predefiniti. Se la rete è operativa, valutare attentamente eventuali conseguenze derivanti dall'uso dei comandi.

Premesse

Che cos'è TCP

Il protocollo TCP (Transmission Control Protocol) è un protocollo fondamentale del livello trasporto che funziona sul livello 4 del modello OSI e fornisce la consegna affidabile, ordinata e controllata dagli errori di un flusso di byte tra le applicazioni che comunicano su una rete IP.

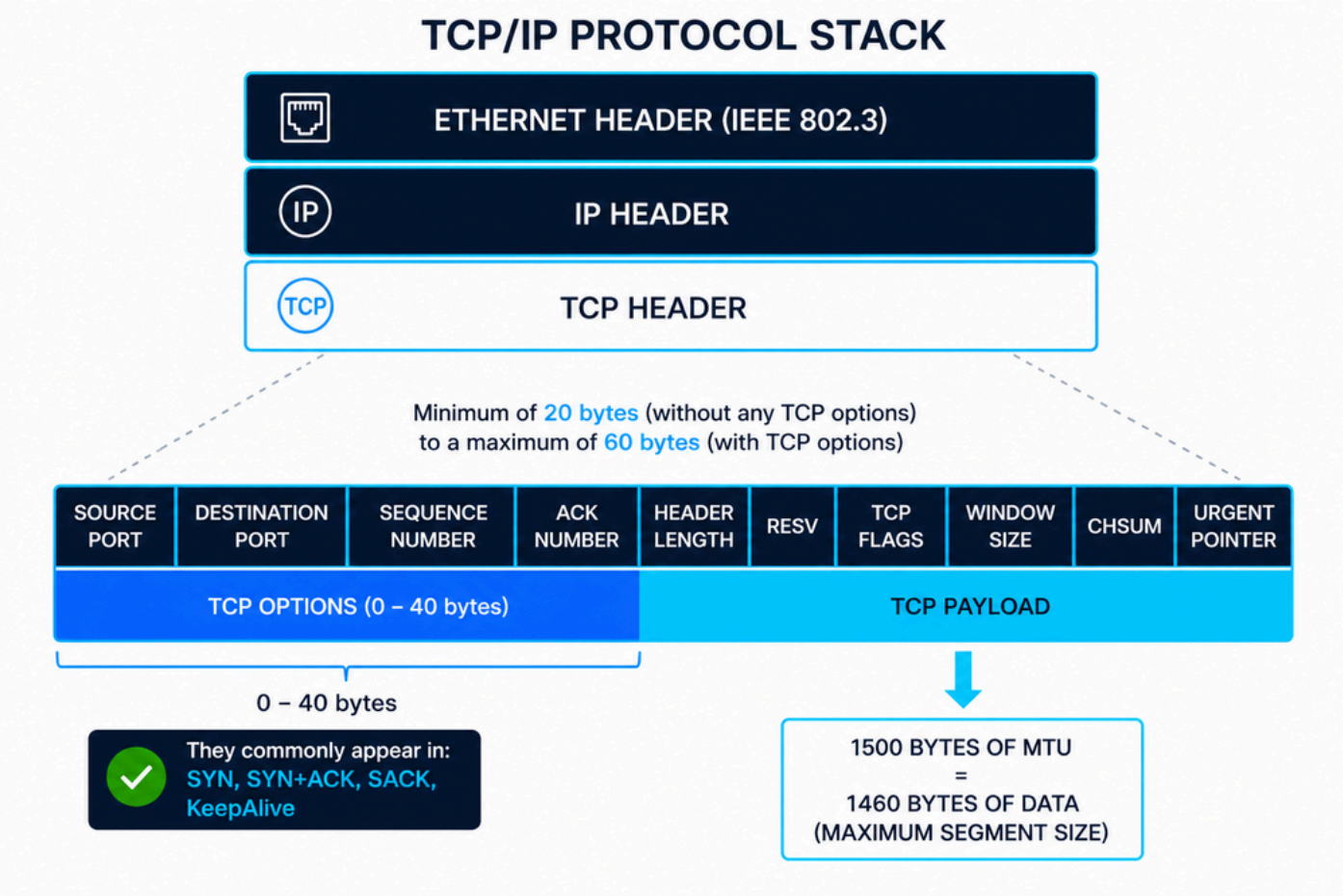
Tre vantaggi principali

1. Affidabilità: il protocollo TCP è orientato alla connessione e garantisce la consegna richiedendo conferme da parte del destinatario. Se un pacchetto viene perso o danneggiato durante la trasmissione, il protocollo TCP trasmette nuovamente i dati per assicurarsi che raggiunga la destinazione.
2. Consegna ordinata: Poiché i pacchetti di rete possono arrivare fuori ordine, il protocollo TCP assegna numeri di sequenza a ciascun segmento. In questo modo, il sistema ricevente può ricomporre i dati nell'esatto ordine in cui sono stati originariamente inviati.
3. Controllo flusso e congestione: Il protocollo TCP gestisce in modo dinamico la velocità di trasmissione dei dati in modo che corrisponda alla capacità di elaborazione dei ricevitori e alle condizioni di rete correnti, impedendo la perdita di dati causata da overflow del buffer o congestione della rete.

Panoramica dell'incapsulamento TCP/IP

Il diagramma rappresenta lo stack TCP/IP dove un segmento TCP (layer 4) è incapsulato in un

pacchetto IP (layer 3) e quindi all'interno di un frame Ethernet (layer 2) definito da IEEE 802.3. Questo approccio su più livelli assicura una comunicazione modulare, in cui ogni layer aggiunge le proprie informazioni di controllo (intestazioni) per garantire la consegna, il routing e l'integrità dei dati.



Intestazione Ethernet (IEEE 802.3)

L'intestazione Ethernet è in genere di 14 byte, composta da:

- Indirizzo MAC di destinazione (6 byte)
- Indirizzo MAC di origine (6 byte)
- EtherType/Length (2 byte)

Inoltre, i frame Ethernet includono un trailer FCS (Frame Check Sequence) da 4 byte per il rilevamento degli errori sul layer 2. IEEE 802.3 definisce il framing, le dimensioni minime e massime dei frame e i vincoli di distribuzione fisica che influiscono direttamente sui protocolli di livello superiore come TCP.

Intestazione IP (IPv4)

L'intestazione IPv4 ha una dimensione minima di 20 byte, estendibile fino a 60 byte con opzioni. I campi chiave includono:

- Indirizzi IP di origine e di destinazione
- Durata (TTL)
- Protocollo (identifica TCP come payload)

Il layer IP è responsabile dell'indirizzamento logico e del routing attraverso le reti, ma non garantisce l'affidabilità.

Struttura intestazione TCP

L'intestazione TCP va da 20 a 60 byte, a seconda delle opzioni. I campi chiave includono:

- Porte di origine/destinazione
- Numero progressivo
- Numero di riconoscimento
- Flag (SYN, ACK, FIN, RST e così via)
- Dimensioni finestra
- Checksum

Il protocollo TCP aggiunge alle comunicazioni IP una modalità affidabile di consegna, una corretta sequenza e un controllo del flusso.

Opzioni TCP (comune 10)

Le opzioni TCP estendono il protocollo base. Le più comuni includono:

1. Maximum Segment Size (MSS): definisce il payload TCP più grande che un host può accettare.
2. Window Scale - Estende la finestra di ricezione oltre 65.535 byte.
3. Riconoscimento selettivo consentito (SACK consentito) - Abilita la funzionalità di riconoscimento selettivo.
4. SACK (Selective Acknowledgment) - Specifica i blocchi di dati ricevuti per evitare ritrasmissioni complete.
5. Timestamp: utilizzato per il calcolo RTT e la protezione dai numeri di sequenza a capo (PAWS).
6. Nessuna operazione (NOP) - Padding per l'allineamento delle opzioni.
7. Fine elenco opzioni (EOL): contrassegna la fine delle opzioni TCP.
8. TCP Fast Open (TFO) - Consente lo scambio di dati durante l'handshake iniziale.

9. Multipath TCP (MPTCP) - Abilita più percorsi di rete per una singola sessione TCP.
10. Opzione di timeout utente (AUTO): controlla per quanto tempo i dati trasmessi possono rimanere non riconosciuti.

Sequenza TCP e comportamento di conferma (incluso SYN/FIN)

Entrambi i flag SYN e FIN utilizzano un numero di sequenza, anche quando non è presente alcun payload. Il protocollo TCP funziona utilizzando un modello di sequenziamento basato su byte, in cui ogni byte trasmesso e specifici flag di controllo fanno avanzare lo spazio della sequenza. Questo comportamento è essenziale per un'analisi TCP accurata nelle acquisizioni dei pacchetti e per la diagnosi di incoerenze nella sequenza o nella conferma.

ACK = SEQ + Lunghezza payload + (SYN ? 1: 0) + (FIN ? 1: 0)

Dove:

- SEQ = numero di sequenza iniziale
- Lunghezza payload = dimensioni dei dati in byte
- SYN? 1: 0 = Aggiunge 1 se è impostato il flag SYN, altrimenti 0
- FIN ? 1: 0 = Aggiunge 1 se è impostato il flag FIN, altrimenti 0
- ACK = byte previsto successivo

Esempio 1: SYN con dati (TCP Fast Open)

- SEQ = 1000
- SYN = 1
- Lunghezza payload = 200 byte

Calcolo ACK:

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

In questo modo si riflette uno scenario in cui i dati vengono inviati durante l'handshake TCP. Sia il payload che il flag SYN utilizzano lo spazio della sequenza.

Esempio 2: FIN con dati (interruzione connessione)

- SEQ = 3000

- FIN = 1
- Lunghezza payload = 150 byte

Calcolo ACK:

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

Ciò indica che il protocollo TCP può includere dati durante il ripristino della connessione e sia il payload che il flag FIN incrementano il numero di sequenza.

MSS e la sua relazione con l'MTU

Il parametro Maximum Segment Size (MSS) definisce il payload massimo che il protocollo TCP può inviare in un segmento.

- MTU Ethernet tipica = 1500 byte
- MSS = MTU - intestazione IP - intestazione TCP
- MSS standard = 1460 byte (1500 - 20 - 20)

Se sono presenti opzioni TCP, il valore MSS viene ridotto di conseguenza. Il valore MSS viene negoziato durante l'handshake TCP a tre vie e impedisce la frammentazione sul layer IP.

Funzionamento della negoziazione MSS nell'handshake a tre vie TCP

Il valore Maximum Segment Size (MSS) viene scambiato durante l'handshake a tre vie TCP utilizzando l'opzione MSS nei pacchetti SYN:

- Host A → Host B (SYN): annuncia il valore MSS (ad esempio, 1460)
- Host B → Host A (SYN-ACK): annuncia il valore MSS (ad esempio, 1380)

Entrambe le parti affermano:

Si tratta del payload TCP più grande accettato.

Regola chiave: Il valore MSS è direzionale

Il valore MSS non è negoziato come un unico valore concordato.

Invece:

- Ciascun host utilizza il valore MSS annunciato dall'altro lato del collegamento.
- In questo modo vengono creati due limiti indipendenti, uno per direzione.

Pertanto:

- A invia i dati utilizzando il valore MSS di B.
- B invia i dati utilizzando il valore MSS di A.

L'origine può inviare un payload TCP superiore al valore MSS di destinazione?

In uno stack TCP correttamente funzionante: No.

- Il mittente deve rispettare il valore MSS pubblicizzato dal destinatario.
- L'invio di segmenti più grandi comporta i seguenti rischi:
  - Frammentazione IP (se si supera l'MTU)
  - Pacchetti scartati (se la frammentazione è bloccata o non supportata)
- Questo comporta:
  - Ritrasmissioni
  - Riduzione delle prestazioni
  - Problemi come i buchi neri del PMTUD (Path MTU Discovery)

Informazioni pratiche per la risoluzione dei problemi

- Verificare sempre i valori MSS nell'handshake a tre vie TCP (pacchetti SYN/SYN-ACK).
- Verifica la presenza di mancata corrispondenza causate da:
  - Tunnel (VXLAN, GRE, IPsec)
  - Firewall che modificano il valore MSS (blocco MSS)
- Sulle piattaforme come Cisco NX-OS, la regolazione MSS viene spesso usata per impedire la frammentazione sui percorsi incapsulati

Dimensioni finestra (controllo flusso)

La finestra Dimensioni definisce la quantità di dati che il ricevitore può accettare senza conferma.

Che cos'è:

- Meccanismo di controllo del flusso per impedire l'overflow del buffer.

Scopo:

- Assicura che il mittente non sovraccarichi il destinatario.

Dove ottenerlo:

- Visibile nelle acquisizioni dei pacchetti (ad esempio, Wireshark).
- Derivato dalla configurazione dello stack TCP del sistema operativo e dalle dimensioni del buffer.

Variabilità fornitore/sistema operativo:

- Le diverse implementazioni (Linux, Windows, Cisco NX-OS) utilizzano la scalabilità dinamica e l'ottimizzazione dei buffer, che portano a dimensioni delle finestre variabili.

Condizione zero finestra:

- Quando Dimensione finestra = 0, il buffer del ricevitore è pieno.
- Il mittente sospende la trasmissione e invia richieste periodiche.

Meccanismi variabili delle finestre

- Controllo del flusso basato sulla velocità
  - Assegna al mittente una velocità dati fissa e assicura che i dati non superino mai tale allocazione.
  - Ideale per applicazioni di streaming.
  - Trasmissione broadcast e multicast
- Controllo del flusso basato su finestra
  - Le dimensioni della finestra variano nel tempo.
  - Il ricevente ottiene il controllo del flusso segnalando la finestra consentita agli aggiornamenti della finestra del mittente.

Uso della risoluzione dei problemi:

- Finestre piccole o zero → Colli di bottiglia sul lato ricevitore (CPU, memoria, applicazione).
- Finestre grandi ma throughput ridotto → Problemi di rete (latenza, congestione).
- L'analisi del comportamento delle finestre è fondamentale per la diagnosi dei problemi di prestazioni nelle sessioni TCP.

Risoluzione dei problemi di TCP Data-Plane su Cisco Nexus 9000 (NX-OS)

In questa sezione viene descritta una metodologia pratica per diagnosticare se uno switch Cisco Nexus con NX-OS influisce sull'inoltro del traffico TCP o introduce problemi di prestazioni. L'approccio è presentato attraverso uno scenario ipotetico.

Quando si osserva una latenza TCP o un peggioramento delle prestazioni, inizialmente si ha il sospetto che sia stata causata dalla rete. Tuttavia, questo presupposto deve essere convalidato tramite l'analisi basata sui dati. Il metodo autorevole per la risoluzione dei problemi TCP è l'acquisizione dei pacchetti, che viene eseguita in modo ottimale:

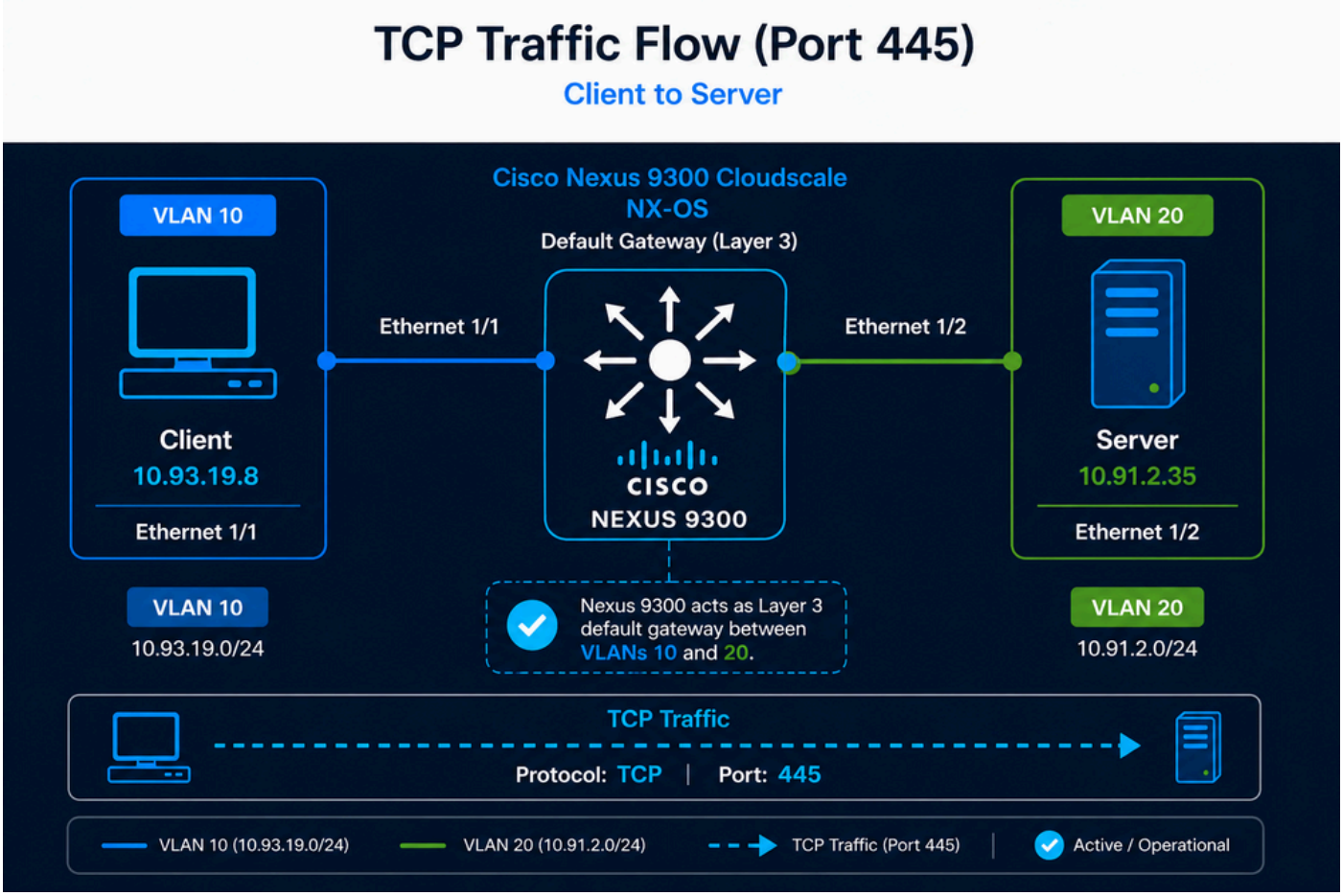
- Contemporaneamente all'origine e alla destinazione
- Prima dell'inizio del traffico

Ciò assicura la visibilità nell'handshake a tre vie TCP, in cui i parametri critici quali MSS, Window Scale e SACK vengono negoziati e non vengono ripetuti in seguito nella sessione. Se non è possibile acquisire simultaneamente, l'analisi può procedere con una singola acquisizione, ma le conclusioni sono limitate.

Definizione scenario

Un utente ha rilevato che il processo di backup per un dataset dell'applicazione di circa 7,5 TB, precedentemente completato in circa 9 ore, richiede ora quasi 21 ore. Sebbene le sessioni TCP tra il client e il server siano ancora stabilite correttamente, l'aumento significativo della durata dei backup suggerisce una potenziale riduzione della velocità di trasmissione o delle prestazioni TCP complessive. Poiché lo switch Nexus è l'unico dispositivo di rete nel percorso e fornisce anche la funzionalità del gateway di layer 3, l'amministratore di rete sospetta che la causa del problema sia lo switch Nexus.

- Client: 10.93.19.8 (VLAN 10)
- Server: 10.91.2.35 (VLAN 20)
- Nexus 9300 con funzione di gateway predefinito
- Porta TCP 445



Convalida iniziale (raggiungibilità)

- Questi comandi sono usati per convalidare la PMTU (Path MTU) tra l'origine e la destinazione inviando pacchetti ICMP con il bit "non frammentare" (DF, Don't Fragment) impostato. Questo aiuta a determinare le dimensioni massime del pacchetto che può attraversare la rete senza frammentazione. Questo processo deve essere eseguito sia sull'origine che sulla destinazione.
- Verificare sempre la MTU dell'interfaccia fisica sia sull'origine che sulla destinazione.
- In questo scenario, l'accesso è disponibile solo per l'host di origine, in cui è stata identificata una MTU di 1500.

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

- -c 10 → Invia 10 richieste echo ICMP
- -I 192.168.10.10 → Utilizza questa specifica origine IP/interfaccia
- -s 1472 → Imposta le dimensioni del payload ICMP su 1472 byte
- -M do → Imposta il bit DF (Non frammentare)
- 192.168.20.20 → IP di destinazione

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

- -n 10 → Invia 10 richieste echo ICMP
- -l 1472 → Imposta le dimensioni del payload ICMP su 1472 byte
- -f → Imposta il flag Non frammentare (DF, Don't Fragment)
- 192.168.20.20 → IP di destinazione

Perché 1472 Byte?

- Payload ICMP = 1472 byte
- Intestazione IP = 20 byte
- Intestazione ICMP = 8 byte
- Dimensioni totali del pacchetto: 1472 + 20 + 8 = 1500 byte (MTU standard)
- Questo comando verifica se il percorso supporta una MTU di 1500 byte senza frammentazione. Se si tenta di inviare 1500 byte di payload ICMP, il ping può avere esito negativo perché le dimensioni totali del pacchetto supererebbero l'MTU standard dopo l'aggiunta delle intestazioni IP e ICMP.

Conclusioni possibili

- Se il ping ha esito positivo (nessuna perdita di pacchetto), il percorso supporta almeno una MTU di 1500 byte e non è richiesta alcuna frammentazione.
    - Clean ICMP results → procedere con l'analisi TCP
    - Riuscita del ping intermittente → possibile perdita di pacchetti, congestione temporanea, limitazione della velocità o problema di inoltro. procedere all'analisi della perdita dei pacchetti, poiché il protocollo TCP richiede un percorso privo di perdite per funzionare in modo efficiente.
- Se il ping ha esito negativo e viene visualizzato l'errore "Frammentazione richiesta" o il timeout, il percorso contiene un collegamento la cui MTU è inferiore a 1500 byte, il pacchetto non può essere inoltrato a causa del bit DF, e questo indica un problema di MTU del percorso.

Modalità d'uso per la risoluzione dei problemi

- Ridurre gradualmente le dimensioni del payload (ad esempio, 1472 → 1400 → 1300) per identificare le dimensioni maggiori che riescono.
- Dopo averla identificata, calcolare l'MTU usando la formula MTU = payload + 28 byte (intestazioni IP + ICMP).

Rilevanza pratica per il TCP

- Se l'MTU è inferiore al previsto, i segmenti TCP possono essere frammentati o eliminati.
- Ciò determina ritrasmissioni, maggiore latenza e throughput ridotto, con un impatto diretto sulle prestazioni delle applicazioni.

Identificazione del percorso del traffico (interfacce)

Per risolvere in modo efficace i problemi di prestazioni TCP su uno switch Cisco Nexus 9000, è essenziale determinare le interfacce che ricevono e inoltrano il traffico tra l'origine e la destinazione.

Nelle topologie semplici, questo può essere dedotto direttamente dalle connessioni fisiche. Ad esempio, se il client è connesso a Ethernet1/1 e il server a Ethernet1/2, il percorso del traffico è semplice. Tuttavia, negli ambienti reali con più interfacce attive, canali di porte o configurazioni vPC, questa identificazione non è sempre banale.

In questi casi, si consiglia di utilizzare il modulo ELAM (Embedded Logic Analyzer Module), che fornisce visibilità a livello ASIC (Data-Plane Hardware).

ELAM consente di acquisire un pacchetto mentre viene elaborato dalla pipeline di inoltro e rivela informazioni critiche quali:

- Interfaccia in ingresso
- Interfaccia in uscita
- Decisione di inoltro (risultato ricerca L2/L3)

Questo metodo è molto più accurato rispetto all'utilizzo degli strumenti del control plane, in quanto riflette l'effettivo percorso di inoltro hardware.

È importante notare che ELAM acquisisce solo un pacchetto alla volta, quindi i criteri di filtro devono essere definiti con precisione per corrispondere al traffico desiderato (ad esempio, IP di origine, IP di destinazione, porta TCP). Se i filtri sono troppo ampi, esiste il rischio di acquisire traffico non correlato, ad esempio ICMP o UDP, anziché il flusso TCP desiderato.

Inoltre, questa procedura deve essere ripetuta per entrambe le direzioni di traffico:

- Origine → Destinazione
- Destinazione → Origine

Negli ambienti che utilizzano vPC o ECMP, è possibile bilanciare il carico del traffico su più percorsi. Di conseguenza, il traffico di inoltro e di ritorno può attraversare diversi switch o interfacce. In questi scenari, ELAM deve essere eseguito su ogni switch Nexus rilevante per garantire la completa visibilità.

Identificando con precisione le interfacce in entrata e in uscita, l'ambito della risoluzione dei problemi viene notevolmente ridotto, consentendo una convalida mirata dei contatori di interfaccia, dei criteri QoS, delle impostazioni MTU e dei potenziali punti di congestione lungo l'esatto percorso di inoltro.

Configurazione ELAM (scala cloud Nexus 9300)

In questo esempio il traffico viene filtrato in base all'indirizzo IP 10.93.19.8 di origine, all'indirizzo IP 10.91.2.35 di destinazione e alla porta TCP 445 di destinazione.

Impostazione 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
```

Dopo aver generato il traffico, recuperare il risultato:

```
<#root>

switch(TAH-elam-insel6)#

report
```

Reverse Traffic Capture (obbligatorio per la completa visibilità)

Per convalidare il percorso di ritorno, ripetere la configurazione scambiando gli indirizzi IP di origine e di destinazione:

```
<#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
```

Note operative

- ELAM acquisisce un solo pacchetto, quindi assicura che il traffico scorra attivamente quando si avvia l'acquisizione.
- I filtri devono essere precisi per evitare di acquisire traffico non correlato.
- Negli ambienti vPC, eseguire ELAM su entrambi gli switch perché l'hash del traffico può avvenire in modo diverso in ciascuna direzione.
- L'interfaccia in entrata del display di uscita, l'interfaccia in uscita e la decisione di inoltro nell'hardware, forniscono una visibilità autorevole sul piano dati.

Riferimento

Guida Cisco Nexus 9000 Cloud Scale ASIC ELAM

Convalida a livello di interfaccia

La convalida a livello di interfaccia garantisce che lo switch Nexus non introduca vincoli o anomalie che influiscano sul traffico TCP. L'obiettivo è verificare che la configurazione, lo stato operativo e i contatori hardware siano coerenti con il comportamento previsto per l'inoltro del piano dati ad alte prestazioni.

Convalida configurazione

- Verificare che non vengano applicati ACL restrittivi alle interfacce:

<#root>

```
switch#

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

- Verificare che non vi siano criteri QoS non intenzionali che influiscono sul traffico (QoS globale e a livello di interfaccia, incluse le code, il policing e il shaping):

```
<#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

- Confermare la configurazione di layer 2 o layer 3 (porta switch rispetto all'interfaccia indirizzata), includendo l'appartenenza della VLAN, lo stato STP e l'indirizzamento 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

Convalida stato operativo

- Verificare la coerenza MTU e assicurarsi che corrisponda alla configurazione prevista (ad esempio, 1500 o 9000 byte):

<#root>

switch#

show interface ethernet1/1-2 | include MTU

- Confermare le impostazioni di velocità interfaccia e duplex:

<#root>

switch#

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

- Convalida stabilità interfaccia (nessuna transizione flapping o collegamento frequente):

<#root>

switch#

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

Convalida contatore errori

- Cancella contatori prima del test:

<#root>

```
switch#
```

```
clear counters interface all
```

- Monitora contatori errori (solo valori diversi da zero):

<#root>

```
switch#
```

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

Convalida post-test

- Rieseguire il test del traffico TCP e osservare di nuovo i contatori:

<#root>

```
switch#
```

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

- I contatori non devono aumentare; qualsiasi aumento indica potenziali problemi relativi al layer 1 o all'hardware, ad esempio errori di collegamento fisico, errori CRC/FCS o sovraccarichi/perdite del buffer.

Stabilità ARP e routing

Verificare la stabilità del routing e dell'ARP è fondamentale per assicurarsi che lo switch Nexus

abbia una raggiungibilità costante sul layer 3 e non introduca problemi di risoluzione intermittenti che potrebbero influire sulle prestazioni del TCP. L'instabilità delle voci di routing o della risoluzione ARP può causare la perdita dei pacchetti, un aumento della latenza o blocchi del traffico.

Criteri di convalida

- Le voci di routing per l'origine e la destinazione devono essere presenti, stabili e non soggette a modifiche frequenti.
- Le voci ARP devono essere risolte e non devono essere continuamente aggiornate o mancanti.

<#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
```

Verifica per determinare se il traffico non è indirizzato alla CPU

Sugli switch Cisco Nexus 9000, l'inoltro viene eseguito nell'hardware (ASIC) e la CPU non è coinvolta nelle normali operazioni del piano dati. Pertanto, osservare il traffico TCP da host a host nel control-plane è anormale e indica che i pacchetti vengono puntualizzati a causa di eccezioni o configurazioni errate. Una volta che il traffico deve essere elaborato dalla CPU, diventa soggetto al Control Plane Policing e si prevede che possano essere osservate delle cadute se il traffico supera la velocità del Control Plane consentita.

Metodo di convalida

- Catturare il traffico che raggiunge il control plane utilizzando l'etanalizzatore:

<#root>

```
switch#

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

Comportamento previsto

- Nella CPU non è possibile osservare alcun traffico di data-plane TCP da host a host.

Comportamento imprevisto

- Se sono visibili pacchetti corrispondenti al flusso, il traffico viene interrotto a causa di:
  - Gestione pacchetti eccezionale (scadenza TTL, registrazione ACL, reindirizzamenti)
  - Configurazione errata o funzionalità non supportate
  - Programmazione hardware non corretta

Determinazione della latenza di inoltro pacchetti

La latenza di inoltro dei pacchetti negli switch Nexus 9000 dipende dalle dimensioni dei pacchetti, dalla modalità di inoltro e dalle funzionalità abilitate. Le specifiche Cisco in genere fanno riferimento alla latenza per i pacchetti da 64 byte con inoltro cut-through.

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

- Inoltro cut-through (predefinito in Nexus 9000):
  - Inizia l'inoltro prima della ricezione del pacchetto completo.
  - Riduce la latenza (da sotto-microsecondi a ~1 µs).
- Storage e inoltro:
  - Ricevere l'intero pacchetto prima di inoltrarlo.
  - Aggiunge una latenza proporzionale alle dimensioni del pacchetto.

Altre caratteristiche possono introdurre una latenza incrementale:

- Incapsulamento/decapsulamento VXLAN
- Ricerche ACL (elaborazione TCAM)
- Classificazione QoS e accodamento
- Telemetria (NetFlow, ERSPAN, sFlow)
- Buffer durante la congestione

Tuttavia:

- Queste operazioni vengono eseguite in pipeline hardware.

L'unico scenario realistico in cui la latenza aumenta sensibilmente è la congestione:

- I pacchetti vengono memorizzati nel buffer nelle code di uscita.
- Il ritardo dipende da:
  - Profondità coda
  - Utilizzo interfaccia
  - Criteri QoS

Anche in questi casi:

- La latenza è generalmente compresa tra i microsecondi e le centinaia di microsecondi.
- Un ritardo di millisecondo sostenuto implica:
  - Grave congestione
  - Sottoscrizione in eccesso
  - QoS o buffering non configurato correttamente

SPAN su CPU (Packet Capture for Data Plane)

Ciò consente il mirroring del traffico del piano dati nel piano di controllo per l'acquisizione dei pacchetti e l'esportazione in un file .pcapng, permettendo un'analisi dettagliata in Wireshark.

Configurazione

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

Esecuzione dell'acquisizione

<#root>

```
switch#
```

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

Considerazioni tecniche

- Il traffico di cui è stato eseguito il mirroring nella CPU è soggetto al Control Plane Policing (CoPP).
- Se il traffico supera il protocollo CoPP:
  - I pacchetti possono essere scartati solo nel control plane.
  - Questo crea falsi positivi durante l'analisi.
- L'uso di SPAN sulla CPU è consigliato in scenari di traffico da basso a moderato.
- Per gli ambienti a throughput elevato:
  - Usa SPAN locale (analizzatore esterno)
  - Usa ERSPAN per l'acquisizione remota

Metodo	Vantaggio	Limitazione
SPAN	Accurato, senza incapsulamento	Richiede una connessione fisica.
ERSPAN	Funzionalità di acquisizione remota	Sensibile a congestione della rete.

Convalida della limitazione della velocità del Control Plane

Per garantire che le acquisizioni da SPAN alla CPU siano affidabili, è necessario verificare che il control-plane non stia eliminando i pacchetti in mirroring a causa della limitazione della velocità.

Comando Validation

```
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
```

Metodologia di convalida

- Eseguire il comando a intervalli di circa 3 secondi.
- Osservare i contatori di rilascio relativi a SPAN.

Interpretazione

- Nessun incremento nei contatori di rilascio per la riga SPAN indica un'acquisizione affidabile.
- L'aumento dei contatori di rilascio indica la perdita di pacchetti sul control plane, rendendo l'acquisizione inaffidabile.

Se si osservano perdite, il metodo di acquisizione deve essere modificato in SPAN o ERSPAN.

Convalida basata su ICMP prima del protocollo TCP

I test ICMP forniscono una convalida di base dell'integrità del piano dati prima di eseguire analisi TCP complesse. Poiché il protocollo ICMP è senza stato e più semplice, permette di rilevare rapidamente le perdite di pacchetti, le duplicazioni o le incoerenze nei percorsi.

Comportamento previsto nell'acquisizione SPAN

- Ogni pacchetto ICMP può essere visualizzato due volte:

    - Una volta in entrata
    - Una volta in uscita
- Per un ping standard:
    - Richiesta echo → 2 pacchetti
    - Risposta echo → 2 pacchetti

Ciò conferma la correttezza dell'inoltro e l'assenza di perdita dei pacchetti nel data-plane.

Comportamento anomalo

- I duplicati mancanti o i conteggi dei pacchetti asimmetrici indicano potenziali perdite di pacchetti o limitazioni di acquisizione.
- I timeout intermittenti suggeriscono problemi di layer 1, congestione o problemi a monte.

Se il traffico ICMP viene inoltrato in modo coerente senza perdita di dati, è probabile che anche il traffico TCP venga inoltrato correttamente sul layer 2/3.

Determinazione della latenza di inoltro dello switch Nexus tramite l'acquisizione dei pacchetti

Quando il traffico viene acquisito usando SPAN per raggiungere la CPU (o SPAN/ERSPAN), ciascun pacchetto può essere osservato due volte: una volta in entrata e una volta in uscita. Questa duplicazione può essere utilizzata per stimare la latenza di inoltro introdotta dallo switch Nexus calcolando la differenza di tempo tra entrambe le istanze dello stesso pacchetto.

In pratica, questa latenza può essere misurata usando il traffico ICMP acquisito in precedenza, confrontando il delta di tempo tra i pacchetti di richiesta echo duplicati e i pacchetti di risposta echo. Questo fornisce una base semplice ed efficace per le prestazioni di inoltro dello switch. Se è necessaria un'analisi più approfondita, la stessa metodologia può essere applicata al traffico TCP acquisendo il flusso e misurando la differenza di tempo tra i pacchetti TCP duplicati.

Metodologia

- Identificare un pacchetto e il relativo duplicato (stesso numero di sequenza).
- Misurare il delta temporale tra le copie in entrata ed in uscita.
- Questo delta rappresenta una stima in alto della latenza di inoltro dello switch, in quanto può includere il mirroring e il sovraccarico dell'indicatore orario.

Configurazione di Wireshark

- Abilita visualizzazione delta ora:

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

- Aggiungere una colonna personalizzata per il delta temporale:

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

- Filtra il traffico rilevante (esempio):

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

- Ordina pacchetti per numero di sequenza o flusso TCP:

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

Interpretazione

- Il delta temporale tra i pacchetti duplicati può essere compreso nell'intervallo di microsecondi.
  - In questo caso, lo switch Nexus non introduce la latenza nell'inoltro dei pacchetti.
- I delta bassi coerenti confermano le prestazioni di inoltro basato su hardware.
- Delta più alti o incoerenti possono indicare:
  - Congestione o buffering

Riferimenti

- Cisco Nexus serie 9000 Data sheet
- Guide alla progettazione di switch Cisco Nexus serie 9000
- White paper sulla gestione intelligente dei buffer sugli switch Cisco Nexus serie 9000

Analisi del traffico TCP dall'acquisizione dei pacchetti dell'host di

origine

In questa sezione viene fornita una metodologia dettagliata per analizzare l'acquisizione di un pacchetto TCP in Wireshark, inclusa la configurazione del profilo, attraverso il caso ipotetico descritto in precedenza. Le immagini mostrate sono state prese direttamente da Wireshark. Si ricorda che lo scenario è il seguente:

Un utente ha rilevato che il processo di backup per un dataset dell'applicazione di circa 6,5 TB, precedentemente completato in circa 9 ore, richiede ora quasi 21 ore. L'unico dispositivo di rete accessibile è uno switch Cisco Nexus 9300 collegato al server di origine (10.93.19.8). L'MTU configurata sull'interfaccia dello switch è di 9000 byte (frame jumbo), mentre l'MTU sul server è sconosciuta. È disponibile l'acquisizione di un pacchetto dal server di origine e tutte le operazioni di convalida Nexus precedenti sono già state completate senza alcuna anomalia rilevata.

Osservazioni e vincoli principali

- L'opzione Nexus è stata esclusa:
  - Nessuna perdita di pacchetti
  - Nessun errore di interfaccia
  - Nessun impatto su QoS o ACL
  - Inoltro hardware confermato
- Configurazione dell'interfaccia:
  - Porta di accesso
  - MTU: 9000 byte
- Dati disponibili:
  - Acquisizione dei pacchetti all'origine
  - Conoscenza dell'MTU end-to-end
    - Il ping è stato completato correttamente senza frammentazione con un pacchetto da 1500 byte con 1.472 byte di dati.
- Dati mancanti:
  - Visibilità destinazione
  - Nessuna acquisizione di pacchetti disponibile nel server di destinazione.

In Wireshark potete creare profili personalizzati personalizzati in base al tipo specifico di analisi che desiderate eseguire.

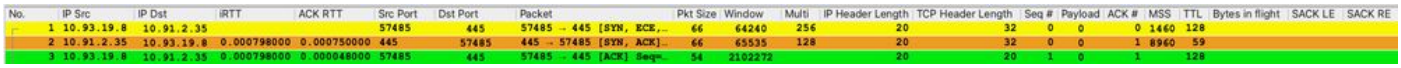
Descrizione colonna

- tcp.analysis.initial_rtt (iRTT): Stima il tempo di andata e ritorno iniziale in base all'handshake a tre vie TCP.
- tcp.analysis.ack_rtt (ACK RTT): Misura l'intervallo di tempo tra un segmento TCP e la

conferma corrispondente.
- tcp.window_size (Finestra): Indica le dimensioni della finestra TCP annunciate dal ricevitore prima dell'applicazione del ridimensionamento.
- tcp.options.wscale.multiplier (multiplo): Rappresenta il fattore di scala della finestra utilizzato per calcolare la finestra ricezione effettiva.
- tcp.seq (Seq#): Visualizza il numero di sequenza del primo byte del segmento TCP.
- tcp.len (Payload): Mostra le dimensioni del payload TCP in byte per il segmento.
- tcp.ack (ACK#): Indica il byte successivo previsto dal mittente (conferma cumulativa).
- tcp.options.mss_val (MSS): Visualizza le dimensioni massime del segmento annunciate durante l'handshake TCP.
- ip.ttl (TTL): Mostra il valore Time To Live, utile per identificare il numero di hop e il comportamento del routing.
- tcp.analysis.bytes_in_flight (byte in volo): Rappresenta la quantità di dati non riconosciuti attualmente in transito.

Analisi dell'handshake a tre vie TCP

L'acquisizione dell'handshake TCP a tre vie è obbligatoria perché contiene parametri critici, ad esempio MSS, Window Scale e SACK, che definiscono il comportamento della sessione. Senza queste informazioni, l'analisi TCP è incompleta e può portare a conclusioni errate sulle prestazioni o sulla causa principale.



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			

Identificazione traffico

Dall'acquisizione del pacchetto:

- Source IP Address: 10.93.19.8
- Indirizzo IP di destinazione: 10.91.2.35

Analisi iRTT (Round-Trip Time) iniziale

L'RTT iniziale (iRTT) viene calcolato nel modo seguente:

- iRTT = 798 microsecondi

Questo valore deriva da:

- RTT ACK pacchetto 2 (SYN-ACK): 750 µs → Tempo per la destinazione di rispondere al SYN.
- ACK RTT pacchetto 3 (ACK): 48 µs → Tempo per la fonte di riconoscere il SYN-ACK.

La maggior parte della latenza (~94%) si trova nel percorso di inoltro (client → server → client), mentre il tempo di risposta dall'origine è minimo, il che indica che non vi è alcun ritardo della CPU o dell'applicazione sul client.

Identificazione porta TCP

- Porta TCP di destinazione: 445

La porta 445 corrisponde a Microsoft Server Message Block (SMB), comunemente utilizzato per la condivisione di file, le unità di rete e i servizi di autenticazione di Windows. Questo protocollo è sensibile sia alla latenza che alla velocità di trasmissione, il che lo rende fortemente dipendente dall'efficienza del TCP e dalla stabilità della rete.

Analisi dimensioni finestra TCP

- Finestra di origine (in scala): 64,240 byte
- Finestra di destinazione: 65,535 byte

La finestra TCP indica la quantità di dati che è possibile inviare prima di attendere la conferma. In questo caso, l'origine è leggermente più restrittiva della destinazione. Questi valori sono relativamente piccoli per gli ambienti moderni e possono limitare il throughput, soprattutto con l'aumento del RTT.

Il throughput teorico massimo può essere stimato utilizzando:

Throughput = Dimensione finestra TCP / RTT

Sostituzione dei valori osservati:

- Dimensione finestra TCP = 64.240 byte
- RTT = 798 microsecondi = 0,000798 secondi

Throughput ≈ 64.240 / 0,000798 ≈ 80,5 MB/s (~644 Mbps)

Questo rappresenta il throughput del limite superiore presupponendo:

- Nessuna perdita di pacchetti
- Nessuna ritrasmissione
- Condizioni di rete ideali

Throughput, tempo di trasferimento e analisi delle condizioni richieste

Con l'attuale throughput di 644 Mbps, il trasferimento di un file da 6,5 TB richiede circa 23,5 ore, il che è in linea con la degradazione osservata. Per ottenere una finestra di trasferimento di 9 ore, la velocità effettiva deve aumentare a circa 1,68 Gb/s, richiedendo una finestra TCP più grande (~2,7x aumento) o un valore RTT significativamente più basso (~291 µs).

Con le condizioni attuali (finestra 64 KB e RTT ~798 µs), non è possibile raggiungere l'obiettivo delle 9 ore, perché la velocità di trasmissione TCP è limitata dal prodotto del ritardo della larghezza di banda. Senza aumentare le dimensioni della finestra o ridurre la latenza, il protocollo non può utilizzare una larghezza di banda maggiore, rendendo la destinazione irraggiungibile.

Scenario	Velocità effettiva	Tempo di trasferimento stimato (6,5 TB)	Finestra TCP necessaria	RTT richiesto
Stato corrente	644 Mbps (~80,5 MB/s)	Circa 23,5 ore	64 KB	798 µs
Target (9 ore)	Circa 1683 Mbps (~210 MB/s)	9 ore	CIRCA 172 KB	~291 µs

Questa operazione ha funzionato in precedenza, indicando che si è verificata una modifica nella rete, nell'applicazione, nell'origine o nella destinazione. È importante notare che, sulla base della sola analisi iniziale, è già possibile trarre una conclusione significativa: con le attuali dimensioni della finestra TCP e le condizioni RTT, non è possibile raggiungere l'obiettivo delle 9 ore.

Le tabelle mostrano un confronto tra le variazioni di velocità di trasmissione quando le dimensioni della finestra RTT e TCP aumentano o diminuiscono.

Impatto RTT sul throughput (dimensioni finestra fisse = 64.240 byte)

RTT	Throughput (MB/s)	Throughput (Mbps)
200 µs (0,0002 s)	Circa 321 MB/s	Circa 2.568 Mbps
798 µ (0,000798 s)	Circa 80,5 MB/s	Circa 644 Mbps

RTT	Throughput (MB/s)	Throughput (Mbps)
2 ms (0,002 s)	Circa 32,1 MB/s	Circa 257 Mbps
10 ms (0,01 s)	Circa 6,4 MB/s	Circa 51 Mbps

Impatto sulle dimensioni della finestra TCP (RTT fisso = 798 µs)

Dimensioni finestra TCP	Throughput (MB/s)	Throughput (Mbps)
16 KB (16.384 KB)	Circa 20,5 MB/s	Circa 164 Mbps
64 KB (64.240 KB)	Circa 80,5 MB/s	Circa 644 Mbps
256 KB (262.144 KB)	Circa 328 MB/s	Circa 2.624 Mbps
1 MB (1.048.576 MB)	Circa 1.314 MB/s	Circa 10,5 Gb/s

Interpretazione tecnica

- Il throughput è inversamente proporzionale a RTT → una latenza più elevata riduce le prestazioni.
- Il throughput è direttamente proporzionale alle dimensioni della finestra TCP → le finestre più grandi aumentano la capacità.
- Le dimensioni ridotte delle finestre limitano notevolmente il throughput, anche in ambienti a bassa latenza.
- Le reti ad alta velocità (10G+) richiedono la scalabilità delle finestre per utilizzare completamente la larghezza di banda.

Questo dimostra che le dimensioni delle finestre RTT e TCP sono fattori critici per le prestazioni TCP e devono essere analizzate insieme per risolvere i problemi di throughput.

Lunghezza header IP e TCP

- Lunghezza intestazione IP: 20 byte
- Lunghezza intestazione TCP: 32 byte

Un'intestazione IP di 20 byte indica che non sono presenti opzioni IP. L'intestazione TCP da 32 byte conferma l'utilizzo delle opzioni TCP, aggiungendo 12 byte oltre l'intestazione base. Queste opzioni in genere includono MSS, Window Scale e SACK Permesso.

Opzioni TCP - Analisi e TTL

La conferma selettiva (SACK) è abilitata su entrambi gli endpoint. Questo non è visibile nell'immagine. SACK consente al destinatario di riconoscere blocchi di dati non contigui, informando il mittente esattamente quali segmenti sono stati ricevuti correttamente.

Ad esempio, se i segmenti 1000-2000 e 3000-4000 vengono ricevuti ma 2000-3000 risulta mancante, il ricevente può indicarlo esplicitamente. Senza lo SACK, il mittente ritrasmetterebbe tutti i dati dopo il gap; con SACK, viene ritrasmessa solo la parte mancante. Ciò migliora in modo significativo le prestazioni negli ambienti con perdita di pacchetti.

Analisi del pacchetto 1 (SYN)

- N. sequenza: 0 (Wireshark normalizzato)
- Payload: 0 byte
- N. ACK: 0
- MSS: 1460 byte
- TTL: 128

Wireshark normalizza il numero di sequenza a zero per la leggibilità, sebbene in pratica si tratti di un valore casuale di grandi dimensioni. Durante la connessione è prevista l'assenza di payload. Il valore MSS di 1460 byte indica una MTU di 1500 byte (20 byte di intestazione IP + 20 byte di intestazione TCP). Un valore TTL di 128 può essere un host basato su Windows e la visualizzazione di questo valore nell'acquisizione indica che l'acquisizione è stata probabilmente eseguita in corrispondenza o in prossimità della sorgente attraverso il layer 2.

Analisi Packet 2 (SYN-ACK)

- N. ACK: 1

Il valore ACK è 1 perché il flag SYN consuma un numero di sequenza, anche quando non è presente alcun payload. ACK = SEQ + 1.

- TTL: 59

Il valore TTL osservato di 59 indica un valore TTL iniziale di 64, quindi il pacchetto ha attraversato circa 5 hop di routing (64 - 59 = 5). Ogni hop indirizzato decrementa il valore TTL di uno.

Rischio di frammentazione e impatto sulla rete

La presenza di circa cinque hop di routing introduce potenziali rischi di prestazioni, correlati in particolare a mancata corrispondenza delle MTU e alla frammentazione.

Se un collegamento intermedio ha una MTU inferiore alle dimensioni del pacchetto originale, può verificarsi la frammentazione. Ne conseguono diverse conseguenze:

- Maggiore latenza dovuta alla frammentazione e al sovraccarico di riassemblaggio.
- Maggiore è la probabilità di perdita del pacchetto, in quanto la perdita di un singolo frammento richiede la ritrasmissione dell'intero pacchetto.
- Throughput ridotto, in quanto il protocollo TCP interpreta la perdita come congestione e riduce la velocità di invio.
- Maggiore utilizzo della CPU sui dispositivi di rete che gestiscono la frammentazione.
- Rischio di errori di rilevamento dell'MTU del percorso (PMTUD) se l'ICMP è bloccato, con conseguenti perdite di pacchetti invisibili all'utente.

Alla luce di questi fattori, è fondamentale garantire un'MTU coerente sul percorso o implementare il blocco MSS, se necessario.

Analisi TCP/RTT: RTT ACK e RTT iniziale

Quando ACK RTT è maggiore di iRTT, indica che la latenza è aumentata rispetto alla linea di base stabilita durante l'handshake TCP.

Ciò significa che la rete o gli endpoint introducono un ulteriore ritardo durante la sessione, in genere dovuto a:

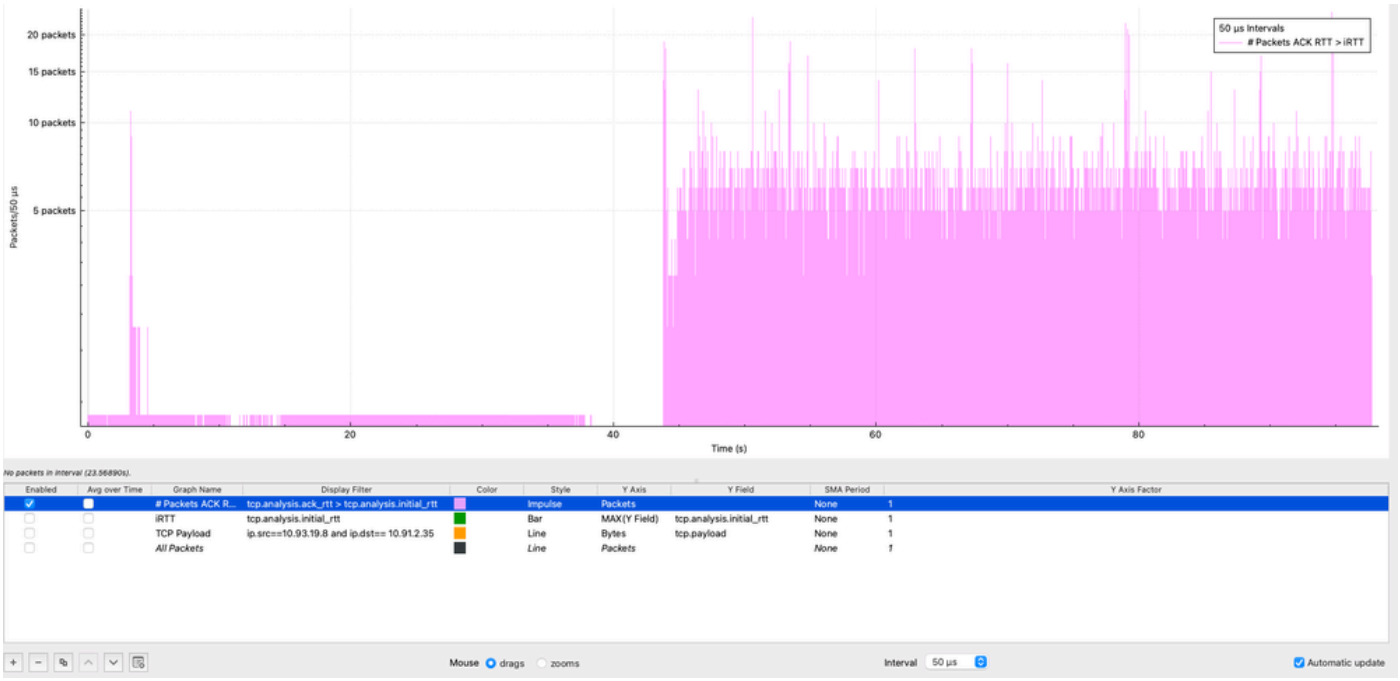
- Congestione o accodamento della rete
- Ritardi di elaborazione del ricevitore o dell'applicazione
- Dispositivi intermedi (firewall, bilanciatori di carico)
- Ritrasmissioni

Se questa condizione persiste durante l'intera sessione TCP, determina:

- Throughput TCP ridotto
- Utilizzo inefficiente delle finestre
- Prestazioni ridotte delle applicazioni

In Wireshark, è possibile visualizzare la frequenza con cui si verifica la condizione ACK RTT > iRTT utilizzando la funzione Grafici di I/O in: Statistiche → Grafici I/O, applicazione del filtro di visualizzazione (tcp.analysis.ack_rtt > tcp.analysis.initial_rtt), selezione dello stile Impulso, impostazione dell'asse Y su Pacchetti e utilizzo di un intervallo di 50 microsecondi.

Nel grafico, gli impulsi viola rappresentano il numero di pacchetti che soddisfano questa condizione entro ogni intervallo di 50 microsecondi. Come osservato, questa condizione persiste nell'intera acquisizione del pacchetto, indicando che la latenza durante la sessione è notevolmente superiore alla linea di base iniziale. Questo comportamento suggerisce fortemente un peggioramento prolungato delle prestazioni piuttosto che una condizione transitoria, rafforzando la necessità di indagare su potenziali cause come congestione, buffering o ritardi di elaborazione dell'endpoint lungo il percorso end-to-end.



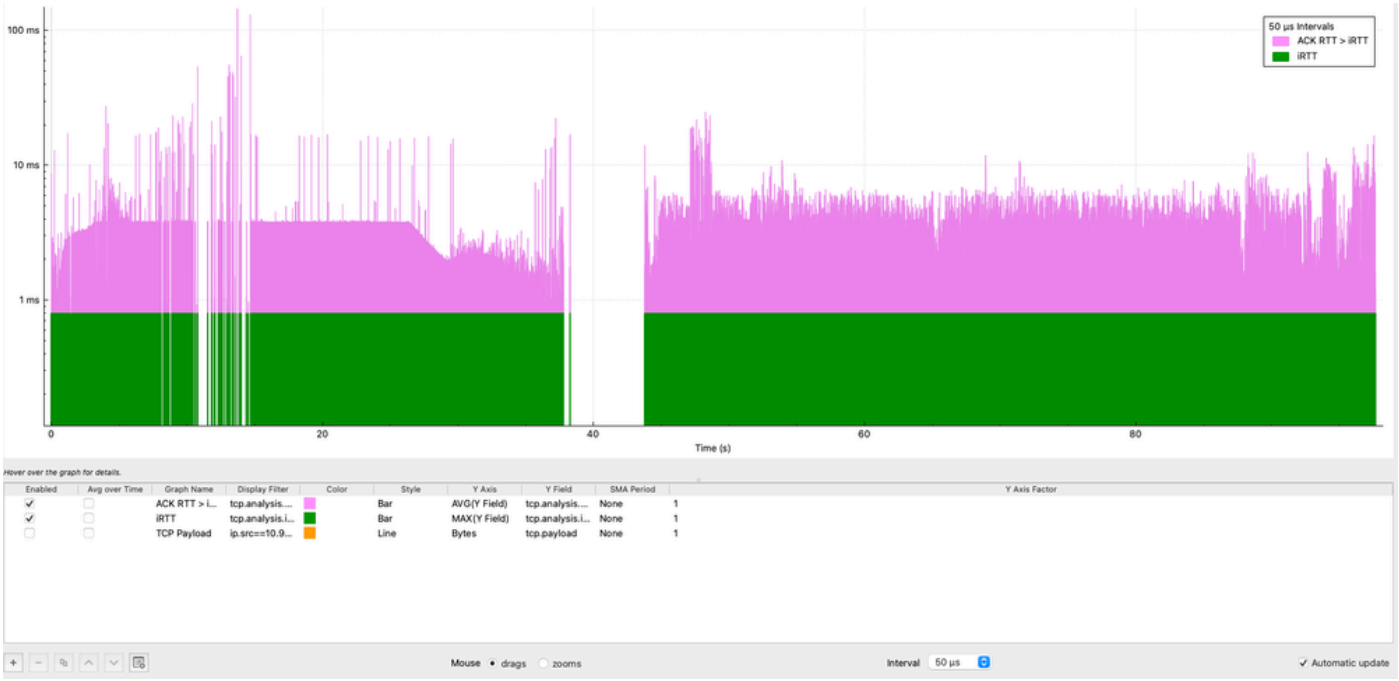
È inoltre importante determinare per quanto tempo l'iRTT viene superato, non solo con quale frequenza. Mentre Wireshark non consente direttamente la sottrazione tra i campi, è possibile ottenere un confronto visivo utilizzando i grafici I/O:

- Passare a Statistiche → Grafici I/O
- Grafico 1:
  - Filtro di visualizzazione: tcp.analysis.ack_rtt > tcp.analysis.initial_rtt
  - Stile: Barre
  - Asse Y: MEDIA
  - Campo Y: tcp.analysis.ack_rtt
  - Intervallo 50 microsecondi
- Grafico 2:
  - Filtro di visualizzazione: tcp.analysis.initial_rtt
  - Stile: Barre
  - Asse Y: MAX
  - Campo Y: tcp.analysis.initial_rtt
- Quindi fate clic con il pulsante destro del mouse sul grafico e attivate Scala logaritmica.

In questa visualizzazione, il grafico viola rappresenta la condizione ACK RTT > iRTT, che è

costantemente presente durante l'intera sessione TCP. I dati mostrano un'inflazione di latenza sostenuta, con picchi multipli che raggiungono 11 millisecondi e un picco massimo di oltre 100 millisecondi, che rappresentano da 11x a 100x l'iRTT di base.

Questo comportamento conferma che l'aumento di latenza non è transitorio ma persistente, indicando un problema sistemico che influisce sulla sessione nel tempo. Questa deviazione prolungata suggerisce fortemente fattori come congestione di rete, buffering (bufferbloat) o ritardi di elaborazione dell'endpoint.



Analisi ritrasmissioni TCP e ritrasmissioni spurie

In questa sezione viene valutata l'affidabilità del TCP analizzando le ritrasmissioni nel tempo, consentendo di verificare se la perdita di pacchetti contribuisce al peggioramento delle prestazioni.

Ritrasmissioni TCP nel tempo

Il grafico mostra la distribuzione delle ritrasmissioni TCP nel tempo. Sono state osservate 42 ritrasmissioni, che rappresentano solo lo 0,00125% del traffico totale.

Questo livello di ritrasmissioni è trascurabile e indica chiaramente che la perdita di pacchetti non è un fattore che contribuisce a questo scenario.

Configurazione Wireshark (ritrasmissioni TCP)

```
Statistics → I/O Graphs
```

- Filtro di visualizzazione:

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

- Stile: Impulso o barra
- Asse Y: Pacchetti
- Intervallo 1 sec.

Ritrasmissioni spurie TCP

Il grafico mostra il numero di ritrasmissioni spurie TCP in intervalli di 1 sec generate dall'origine 10.93.19.8.

In Wireshark, una ritrasmissione spurie TCP indica che un host ha ritrasmesso un segmento che non era stato effettivamente perso. Il pacchetto originale ha raggiunto il destinatario, ma il mittente ha erroneamente presunto una perdita a causa di una stima imprecisa degli intervalli. Questo comportamento non indica una reale perdita di pacchetti, ma piuttosto una logica di ritrasmissione inefficiente del mittente.

In questa clip:

- La sorgente 10.93.19.8 ritrasmette i pacchetti dopo solo circa 8 microsecondi.
- Mentre i timer di ritrasmissione sono nell'ordine di circa 200 millisecondi.

Ciò conferma che il comportamento di ritrasmissione è interamente controllato dallo stack TCP di origine, non dalla rete.

Il numero totale di ritrasmissioni spurie osservate è 1.112, che rappresenta lo 0,0332% del traffico totale catturato.

Configurazione Wireshark (ritrasmissioni spurie TCP)

```
Statistics → I/O Graphs
```

- Filtro di visualizzazione:

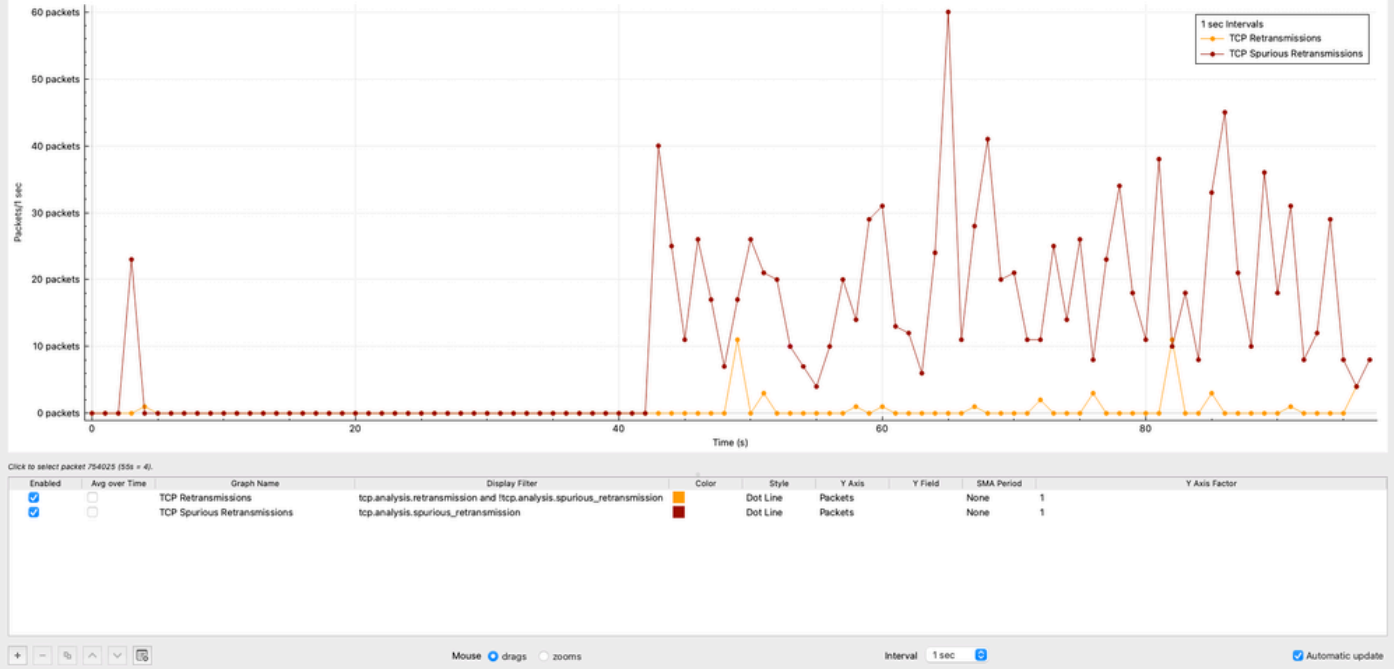
```
tcp.analysis.spurious_retransmission and ip.src==10.93.19.8
```

- Stile: Impulso o barra
- Asse Y: Pacchetti
- Intervallo 1 sec.

Interpretazione tecnica

- La percentuale estremamente bassa di ritrasmissioni reali conferma che la perdita di pacchetti non è presente nella rete.
- La presenza di ritrasmissioni spurie indica decisioni di ritrasmissione premature da parte dell'host di origine.
- Questo comportamento può influire leggermente sull'efficienza, ma non è una causa primaria di grave degrado del throughput.

Questa analisi conferma ulteriormente che il problema non è correlato all'affidabilità della rete, ma piuttosto al comportamento TCP, alla latenza o alle prestazioni dell'endpoint.



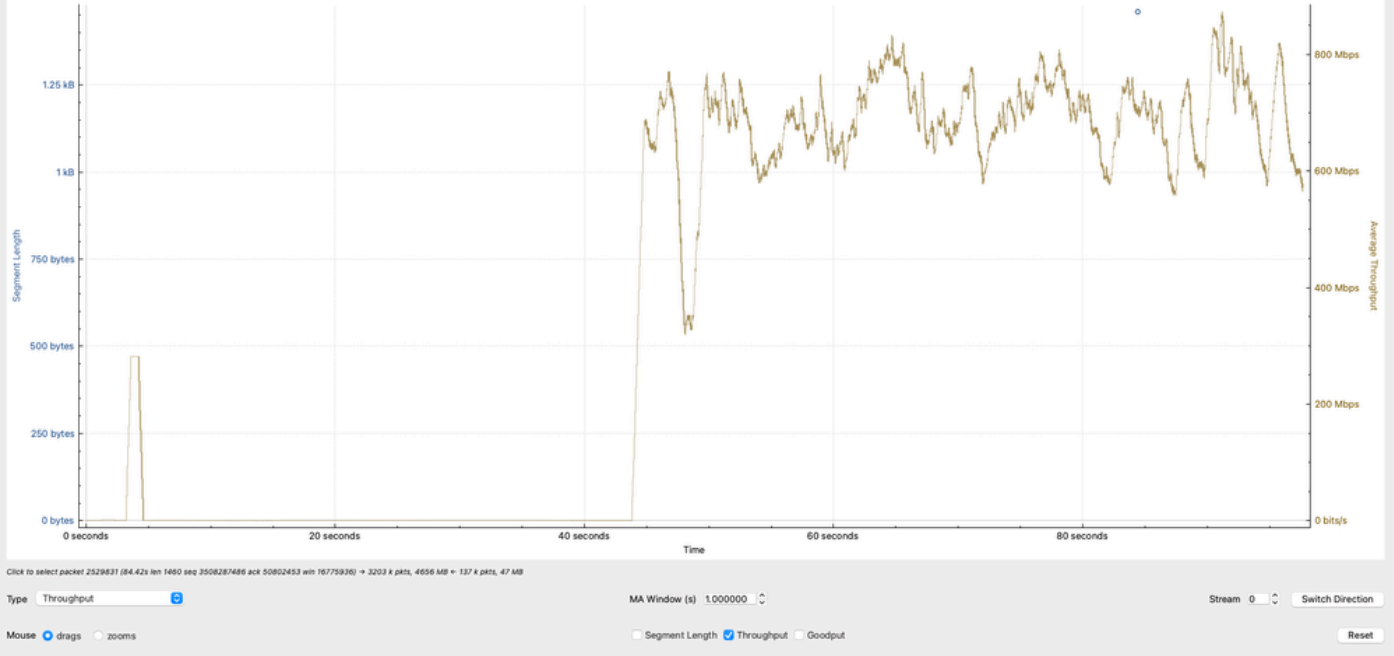
Analisi effettiva del throughput

Il grafico mostra la velocità effettiva, calcolata in base al payload TCP (dati effettivi trasferiti), in

megabit al secondo. Il throughput osservato oscilla principalmente tra 600 Mbps e 800 Mbps, indicando che mentre la rete sta trasferendo attivamente i dati, non sta raggiungendo un potenziale di larghezza di banda maggiore.

Configurazione Wireshark (Throughput Effettivo)

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



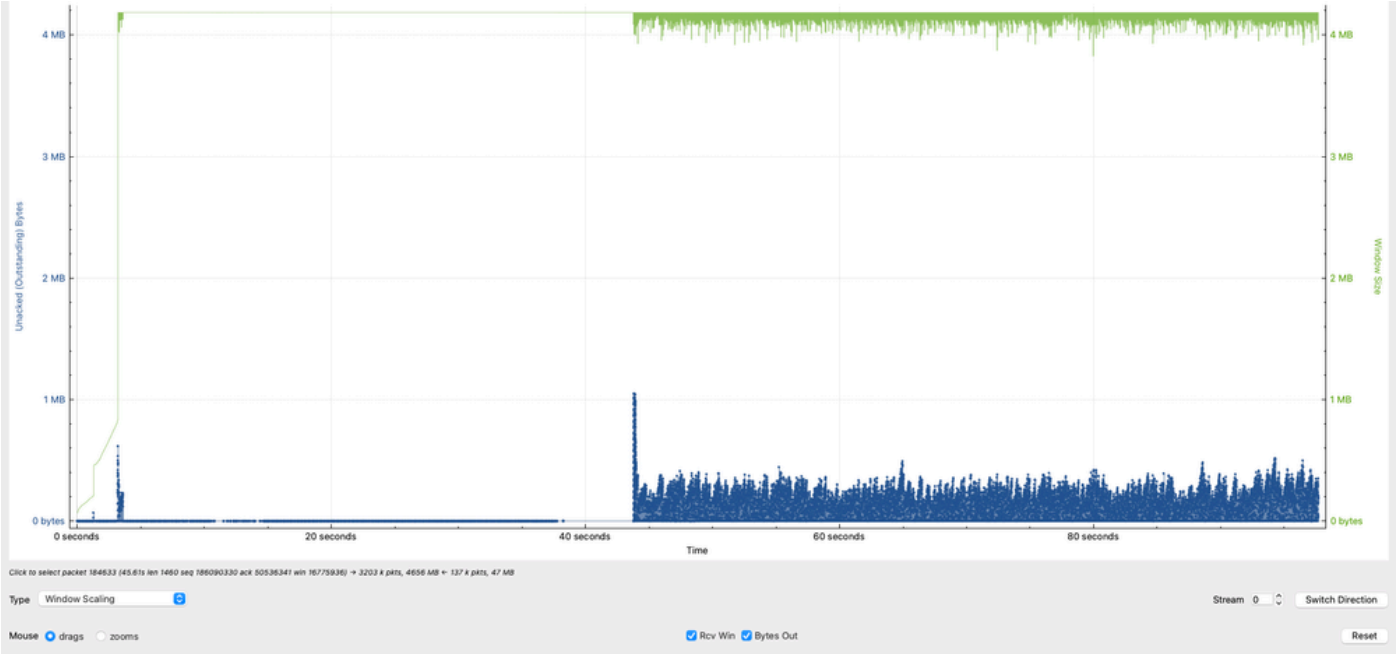
Interpretazione tecnica

- L'intervallo di velocità di trasmissione di 600-800 Mbps è allineato con i calcoli precedenti basati sulle dimensioni della finestra TCP e su RTT.
- La variabilità del throughput riflette:
  - Fluttuazioni RTT
  - Regolazioni controllo congestione TCP
  - Pacing o buffering delle applicazioni
- Poiché il throughput non si avvicina alla velocità della linea (ad esempio, 10G), il limite non è la larghezza di banda fisica, bensì i vincoli di efficienza del TCP.
- Questa analisi conferma che il throughput osservato è coerente con le limitazioni TCP (dimensioni della finestra e latenza), rafforzando il fatto che il collo di bottiglia non è dovuto alla perdita di pacchetti o alla capacità dell'interfaccia, ma al comportamento del livello di trasporto e alle condizioni dell'endpoint.

Analisi dati in volo (finestra TCP)

Il grafico evidenzia un comportamento critico nella sessione TCP confrontando la capacità del ricevitore con i dati effettivi in transito (byte in volo).

- La linea verde rappresenta la quantità di dati TCP che 10.91.2.35 (ricevitore) può accettare (finestra di ricezione effettiva).
- La linea blu rappresenta la quantità di dati TCP attualmente in esecuzione da 10.93.19.8 (mittente).



I dati osservati in volo raggiungono il picco di circa 1 MB, con picchi aggiuntivi di circa 8 KB e 5 KB, ma si concentrano principalmente tra 1 KB e 250 KB.

Ciò indica che, sebbene il destinatario sia in grado di gestire volumi di dati più grandi, il mittente non utilizza in modo coerente la finestra disponibile.

Configurazione di Wireshark (dati in volo vs finestra)

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

Interpretazione tecnica

- Il ricevitore (10.91.2.35) annuncia una finestra significativamente più grande, indicando che è in grado di ricevere più dati.
- Il mittente (10.93.19.8) sta sottoutilizzando la finestra disponibile, come mostrato dai valori più bassi e incoerenti di Data in Flight (Dati in volo).

    - Il mittente può idealmente mantenere i valori di Data in Flight più vicini alla finestra pubblicizzata del destinatario (~1 MB) per massimizzare la velocità di trasmissione.
    - L'incapacità di sostenere elevati livelli di dati in-flight limita direttamente la velocità di trasmissione ed è un forte indicatore dell'inefficienza TCP all'origine, non un problema di capacità di rete.

Payload TCP e MSS durante l'analisi del tempo

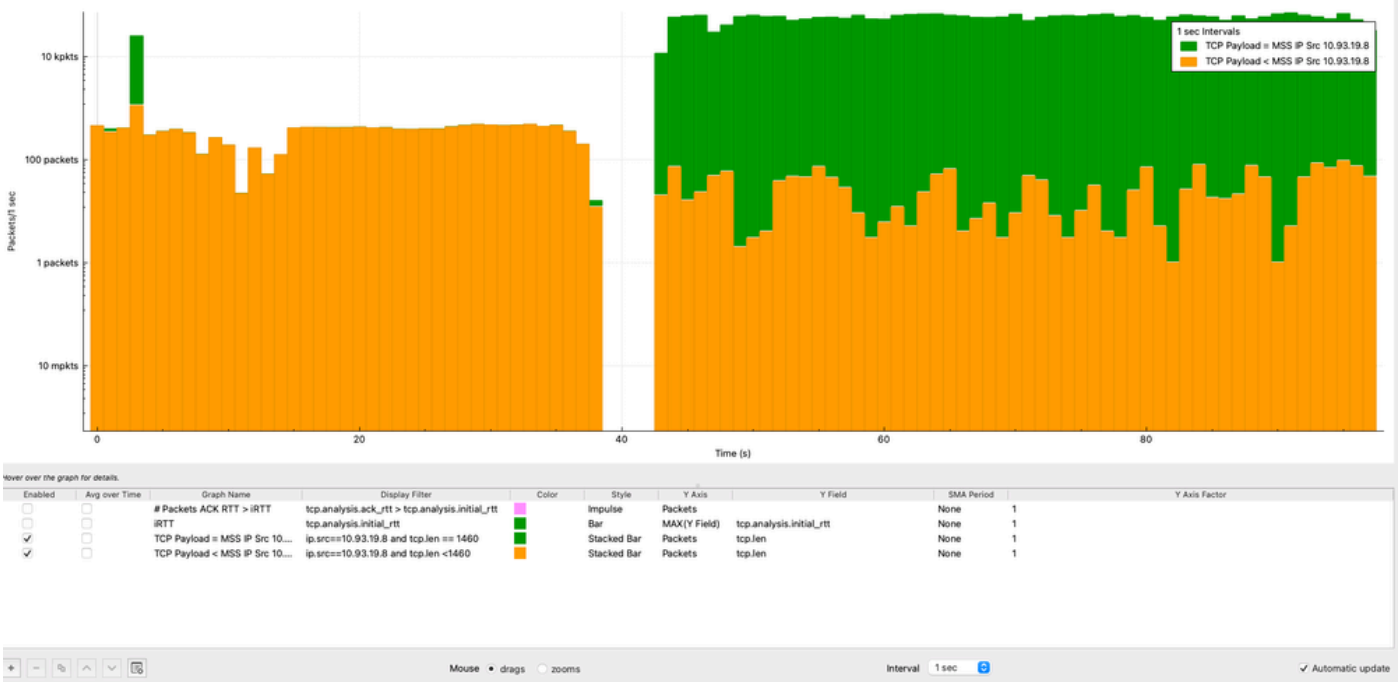
L'analisi delle dimensioni del payload TCP sul valore MSS nel tempo aiuta a determinare se il mittente sta utilizzando in modo efficiente ciascun segmento TCP. L'analisi viene eseguita dalla prospettiva dell'indirizzo IP di origine (10.93.19.8).

In Wireshark, i grafici sono configurati come segue:

- Grafico 1 (pacchetti di dimensioni MSS):
    - Filtro di visualizzazione: ip.src==10.93.19.8 e tcp.len == 1460
    - Stile: Barre in pila
    - Asse Y: Pacchetti
    - Intervallo 1 secondo
- Grafico 2 (tutti i pacchetti ≤ MSS):
    - Filtro di visualizzazione: ip.src==10.93.19.8 e tcp.len <= 1460
    - Stile: Barre in pila
    - Asse Y: Pacchetti
    - Intervallo 1 secondo
- Applicazione della scala logaritmica per una migliore visualizzazione

Dall'analisi:

- La maggior parte dei pacchetti (più di 10.000 pacchetti al secondo) raggiunge coerentemente il valore MSS di 1460 byte.
- Una porzione più piccola di pacchetti supporta un payload inferiore a causa del normale comportamento TCP (ACK, segmentazione o dati di fine flusso).



RCA (Root Cause Analysis): Riduzione prestazioni TCP

Questa analisi dimostra che per identificare la root cause dei problemi di prestazioni del protocollo TCP è necessario un approccio olistico end-to-end, anziché presupporre che la rete sia la principale causa di degradazione.

Sullo switch Cisco Nexus 9300 è stata eseguita una convalida estesa, che include contatori di interfaccia, policy QoS, stabilità di routing e ARP, verifica del punt della CPU, acquisizione di pacchetti basata su SPAN e convalida dell'inoltro a livello ASIC con ELAM. Tutti i risultati hanno confermato che lo switch funzionava secondo i parametri previsti:

- Nessuna perdita di pacchetti
- Nessuna latenza anomala (intervallo microsecondi)
- Nessun impatto QoS o sul piano di controllo
- Correggi inoltro hardware

Inoltre, l'analisi TCP ha rivelato:

- Trasmissioni trascurabili (0,00125%)
- Nessuna prova di perdita di pacchetti
- Utilizzo MSS coerente all'origine
- Throughput allineato con i vincoli della finestra TCP e RTT
- Sottoutilizzo della finestra TCP disponibile (analisi dati in volo)
- La rete non è il collo di bottiglia
- Il server di origine limita le prestazioni

Conclusioni

Il calo delle prestazioni è causato dal server di origine che opera con MTU 1500 in un ambiente con capacità jumbo, impedendo un uso efficiente della capacità di rete disponibile.

Soluzione

Aumentare l'MTU sul server di origine da 1500 a 9000 byte per allinearla alla destinazione e all'infrastruttura di rete. I vantaggi:

- Abilita segmenti TCP di dimensioni maggiori
- Riduzione del sovraccarico dei pacchetti
- Miglioramento del throughput complessivo

Riflessione tecnica

Un aspetto chiave di questa analisi è l'importanza di evitare conclusioni premature durante la risoluzione dei problemi relativi alle prestazioni della rete. Benché inizialmente sia comune attribuire i problemi alla rete, questo caso dimostra chiaramente che la rete stava funzionando correttamente nell'intero percorso del piano dati. Solo eseguendo un'analisi TCP approfondita sia dalla prospettiva di origine che da quella di destinazione, inclusi i parametri di handshake, il comportamento RTT, l'utilizzo delle finestre, le ritrasmissioni e l'efficienza del payload, è stato possibile identificare con precisione il vero collo di bottiglia.

L'analisi dettagliata del comportamento TCP consente di evitare diagnosi errate, di ridurre le modifiche di rete non necessarie e di indirizzare gli sforzi di correzione verso la root cause effettiva.

Informazioni su questa traduzione

Cisco ha tradotto questo documento utilizzando una combinazione di tecnologie automatiche e umane per offrire ai nostri utenti in tutto il mondo contenuti di supporto nella propria lingua. Si noti che anche la migliore traduzione automatica non sarà mai accurata come quella fornita da un traduttore professionista. Cisco Systems, Inc. non si assume alcuna responsabilità per l’accuratezza di queste traduzioni e consiglia di consultare sempre il documento originale in inglese (disponibile al link fornito).