Problemen oplossen met TCP-prestaties op de Nexus 9000 (NX-OS)

Inhoud

Inleiding

In dit document worden de TCP-fundamentals, de diepgaande pakketanalyse van Wireshark en praktische probleemoplossing beschreven om end-to-end prestaties te optimaliseren.

Voorwaarden

Vereisten

Cisco raadt kennis van de volgende onderwerpen aan:

- IP/TCP

Gebruikte componenten

De informatie in dit document is gebaseerd op de volgende software- en hardware-versies:

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

Opmerking: vragen over de configuratie en interoperabiliteit van software of hardware van derden vallen buiten de ondersteuning van Cisco. Het gebruik van tools van derden is de beste manier om uw configuratie en werking met Cisco-apparatuur aan te tonen.

De informatie in dit document is gebaseerd op de apparaten in een specifieke laboratoriumomgeving. Alle apparaten die in dit document worden beschreven, hadden een opgeschoonde (standaard)configuratie. Als uw netwerk live is, moet u zorgen dat u de potentiële impact van elke opdracht begrijpt.

Achtergrondinformatie

Wat is TCP

Transmission Control Protocol (TCP) is een fundamenteel transportlaagprotocol dat werkt op laag 4 van het OSI-model en biedt een betrouwbare, geordende en foutgecontroleerde levering van een stroom bytes tussen toepassingen die via een IP-netwerk communiceren.

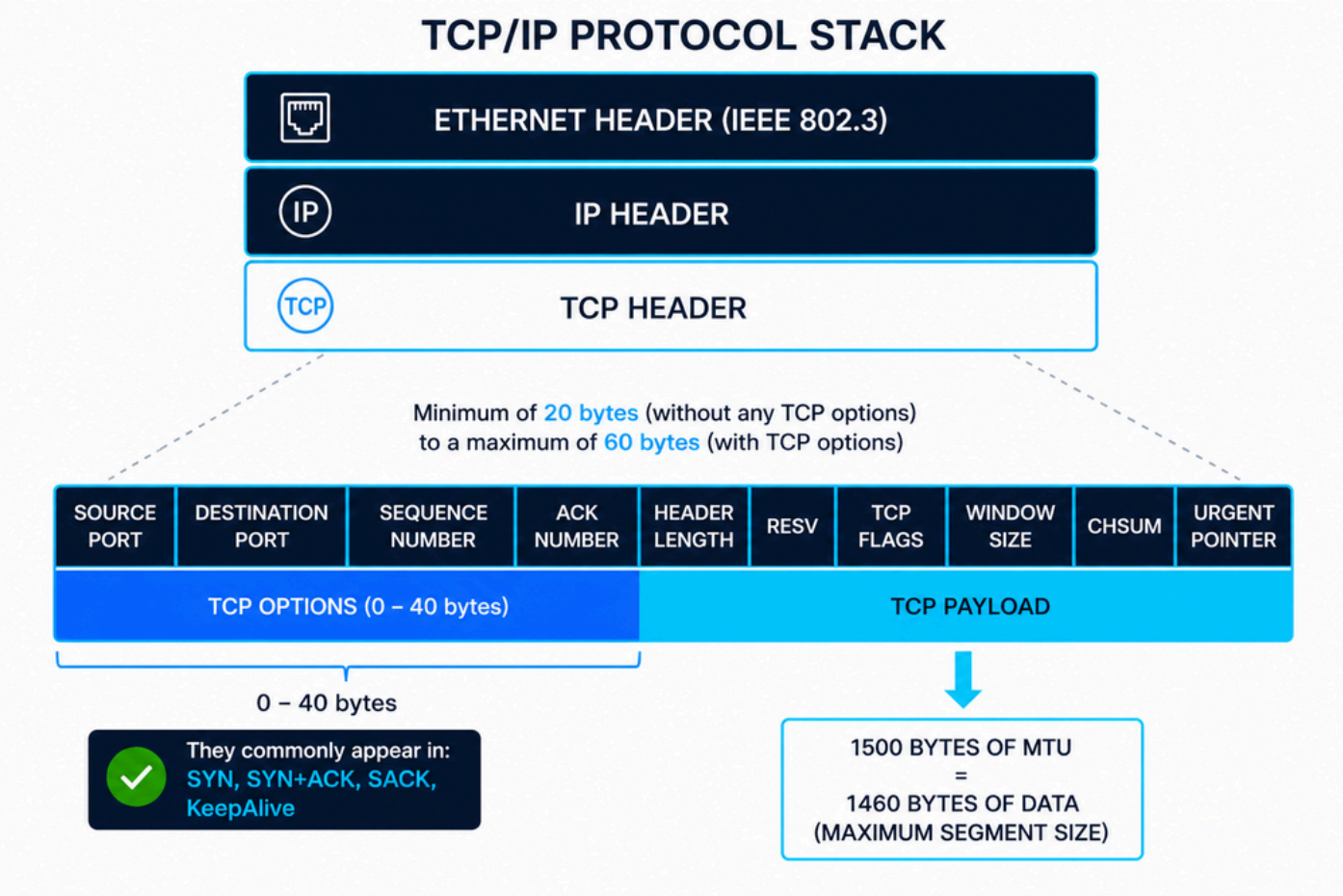
Drie belangrijke voordelen

1. Betrouwbaarheid: TCP is connectiegericht en garandeert levering door bevestigingen van de ontvanger te eisen. Als een pakket verloren gaat of beschadigd raakt tijdens de verzending, verzendt TCP de gegevens automatisch opnieuw om ervoor te zorgen dat het de bestemming bereikt.
2. Bestelde levering: Omdat netwerkpakketten buiten bestelling kunnen komen, kent TCP volgnummers toe aan elk segment. Hierdoor kan het ontvangende systeem de gegevens opnieuw samenstellen in de exacte volgorde waarin deze oorspronkelijk zijn verzonden.
3. Stroom- en congestiecontrole: TCP beheert dynamisch de snelheid van gegevensoverdracht om de verwerkingscapaciteit van de ontvangers en de huidige netwerkomstandigheden te matchen, waardoor gegevensverlies door bufferoverflows of netwerkcongestie wordt voorkomen.

Overzicht van TCP/IP-inkapseling

Het diagram vertegenwoordigt de TCP/IP-stack waarin een TCP-segment (Layer 4) is ingekapseld in een IP-pakket (Layer 3) en vervolgens in een Ethernet-frame (Layer 2) gedefinieerd door IEEE 802.3. Deze gelaagde aanpak zorgt voor modulaire communicatie, waarbij elke laag zijn eigen besturingsinformatie (headers) toevoegt om levering, routering en gegevensintegriteit te

garanderen.



Ethernet-header (IEEE 802.3)

De Ethernet-header is meestal 14 bytes, bestaande uit:

- MAC-adres bestemming (6 bytes)
- MAC-bronadres (6 bytes)
- EtherType/lengte (2 bytes)

Daarnaast bevatten Ethernet-frames een 4-byte Frame Check Sequence (FCS) trailer voor foutdetectie bij Layer 2. IEEE 802.3 definieert framing, minimale/maximale framematen en fysieke leveringsbeperkingen die rechtstreeks van invloed zijn op protocollen met de bovenste laag, zoals TCP.

IP-header (IPv4)

De IPv4-header heeft een minimale grootte van 20 bytes, uitbreidbaar tot 60 bytes met opties.

Belangrijke velden zijn onder meer:

- IP-adressen van bron en bestemming
- Tijd om te leven (TTL)
- Protocol (identificeert TCP als payload)

De IP-laag is verantwoordelijk voor logische adressering en routering over netwerken, maar garandeert geen betrouwbaarheid.

Structuur van TCP-koptekst

De TCP-header varieert van 20 tot 60 bytes, afhankelijk van de opties. Belangrijke velden zijn onder meer:

- Bron-/bestemmingspoorten
- volgnummer
- erkenningsnummer
- Vlaggen (SYN, ACK, FIN, RST, enzovoort)
- Venstergrootte
- controlesom

TCP voegt betrouwbare levering, juiste sequencing en flow control toe aan IP-communicatie.

TCP-opties (algemeen 10)

TCP-opties breiden het basisprotocol uit. De meest voorkomende zijn:

1. Maximale segmentgrootte (MSS) – Definieert de grootste TCP-payload die een host kan accepteren.
2. Vensterschaal – Verlengt het ontvangstvenster tot meer dan 65.535 bytes.
3. Selectieve erkenning toegestaan (SACK toegestaan) – maakt selectieve erkenning mogelijk.
4. Selectieve erkenning (SACK) – Specificeert ontvangen gegevensblokken om volledige hertransmissies te voorkomen.
5. Tijdstempels - Gebruikt voor RTT-berekening en bescherming tegen ingepakte volgnummers (PAWS).
6. No-Operation (NOP) – Padding voor uitlijning van opties.
7. Einde van optielijst (EOL) – markeert het einde van TCP-opties.
8. TCP Fast Open (TFO) – maakt gegevensuitwisseling mogelijk tijdens de eerste handshake.
9. Multipath TCP (MPTCP) – Maakt meerdere netwerkpaden mogelijk voor één TCP-sessie.
10. User Timeout Option (UTO) – Hiermee bepaalt u hoelang verzonden gegevens niet kunnen

worden herkend.

TCP-sequentie en erkenningsgedrag (inclusief SYN/FIN)

Zowel SYN- als FIN-vlaggen gebruiken elk één volgnummer, zelfs als er geen lading aanwezig is. TCP werkt met een byte-georiënteerd sequencingmodel, waarbij elke verzonden byte en specifieke besturingsvlaggen de sequentieruimte bevorderen. Dit gedrag is essentieel voor nauwkeurige TCP-analyse bij pakketopnames en voor het diagnosticeren van sequencing- of bevestigingsinconsistenties.

ACK = SEQ + lengte nuttige lading + (SYN? 1: 0) + (FIN? 1: 0)

Waarbij:

- SEQ = initieel volgnummer
- Lengte payload = gegevensgrootte in bytes
- SYN? 1: 0 = Voegt 1 toe als SYN-markering is ingesteld, anders 0
- FIN? 1: 0 = Voegt 1 toe als FIN-markering is ingesteld, anders 0
- ACK = Volgende verwachte byte

Voorbeeld 1: SYN met gegevens (TCP Fast Open)

- SEQ = 1000
- SYN = 1
- Lengte payload = 200 bytes

ACK-berekening:

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

Dit weerspiegelt een scenario waarbij gegevens worden verzonden tijdens de TCP-handshake. Zowel de payload als de SYN-vlag verbruiken sequentieruimte.

Voorbeeld 2: FIN met gegevens (Verbindingsbeëindiging)

- SEQ = 3000
- FIN = 1
- Lengte payload = 150 bytes

ACK-berekening:

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

Hieruit blijkt dat TCP gegevens kan opnemen tijdens het afbreken van de verbinding en dat zowel de payload als de FIN-markering het volgnummer verhogen.

MSS en zijn relatie met MTU

De maximale segmentgrootte (MSS) definieert de maximale payload die TCP in een segment kan verzenden.

- Typische Ethernet MTU = 1500 bytes
- MSS = MTU − IP-header − TCP-header
- Standaard MSS = 1460 bytes (1500 − 20 − 20)

Als TCP-opties aanwezig zijn, wordt MSS dienovereenkomstig verminderd. MSS wordt onderhandeld tijdens de TCP drieweg handshake en voorkomt fragmentatie op de IP-laag.

Hoe MSS-onderhandeling werkt in de TCP-handdruk

De maximale segmentgrootte (MSS) wordt uitgewisseld tijdens de TCP-handshake in drie richtingen met behulp van de MSS-optie in SYN-pakketten:

- Host A → Host B (SYN): adverteert zijn MSS (bijvoorbeeld 1460)
- Host B → Host A (SYN-ACK): adverteert zijn MSS (bijvoorbeeld 1380)

Elke partij zegt effectief:

Dit is de grootste TCP-payload die wordt geaccepteerd.

Belangrijkste regel: MSS is directioneel

Over MSS wordt niet onderhandeld als één overeengekomen waarde.

In plaats daarvan:

- Elke host gebruikt de MSS die door de andere kant wordt geadverteerd.
- Dit creëert twee onafhankelijke limieten, één per richting.

Daarom:

- A verzendt gegevens met behulp van de MSS van B.
- B verzendt gegevens met behulp van A's MSS.

Kan de bron meer TCP-payload verzenden dan de bestemmings-MSS

In een correct functionerende TCP-stack: Nee.

- De afzender moet de door de ontvanger geadverteerde MSS respecteren.
- Het verzenden van grotere segmenten zou het risico lopen:
  - IP-fragmentatie (als MTU wordt overschreden)
  - Pakketdruppels (als fragmentatie wordt geblokkeerd of niet wordt ondersteund)
- Dit leidt tot:
  - heruitzendingen
  - Verslechtering van prestaties
  - Problemen zoals PMTUD (Path MTU Discovery) zwarte gaten

Praktisch inzicht voor probleemoplossing

- Controleer altijd de MSS-waarden in de TCP-driewegshandshake (SYN/SYN-ACK-pakketten).
- Controleren op mismatches veroorzaakt door:
  - Tunnels (VXLAN, GRE, IPsec)
  - Firewalls die MSS wijzigen (MSS-klemmen)
- Op platforms zoals Cisco NX-OS wordt MSS-aanpassing vaak gebruikt om fragmentatie over ingekapselde paden te voorkomen

Venstergrootte (stroomregeling)

De venstergrootte bepaalt hoeveel gegevens de ontvanger kan accepteren zonder bevestiging.

Wat het is:

- Een stroomregelmechanisme om bufferoverloop te voorkomen.

Doel:

- Zorgt ervoor dat de afzender de ontvanger niet overweldigt.

Waar het te verkrijgen:

- Zichtbaar in pakketopnames (bijvoorbeeld Wireshark).
- Afgeleid van TCP-stackconfiguratie en buffergrootte van het besturingssysteem.

Variabiliteit leverancier/besturingssysteem:

- Verschillende implementaties (Linux, Windows, Cisco NX-OS) maken gebruik van dynamische schaling en bufferafstemming, wat leidt tot verschillende venstergroottes.

Nulvenstervoorwaarde:

- Wanneer Window Size = 0, de buffer van de ontvanger is vol.
- De afzender pauzeert de verzending en stuurt periodieke sondes.

Variabele Windows-mechanismen

- snelheidsafhankelijke stromingsregeling
    - Het wijst de afzender een vaste datasnelheid toe en zorgt ervoor dat de gegevens nooit die toewijzing overschrijden.
    - Ideaal voor streaming toepassingen.
    - Uitzending en levering via multicast
- venstergebaseerde stroomregeling
    - De grootte van het venster varieert met de tijd.
    - De ontvanger bereikt flow control door het toegestane venster te signaleren voor de updates van het zendervenster.

Gebruik voor probleemoplossing:

- Klein of nul Windows → knelpunt aan de ontvanger (CPU, geheugen, toepassing).
- Grote vensters maar lage doorvoer → Problemen met het netwerk (latentie, congestie).
- Het analyseren van venstergedrag is van cruciaal belang voor het diagnosticeren van prestatieproblemen in TCP-sessies.

Problemen met TCP Data-Plane oplossen op de Cisco Nexus

9000 (NX-OS)

In dit gedeelte wordt een praktische methode beschreven voor het vaststellen of een Cisco Nexus-switch met NX-OS het doorsturen van TCP-verkeer beïnvloedt of prestatieproblemen veroorzaakt. De aanpak wordt gepresenteerd door middel van een hypothetisch scenario.

Wanneer TCP-latentie of prestatievermindering wordt waargenomen, is het gebruikelijk om in eerste instantie te vermoeden dat het netwerk dit veroorzaakt. Deze veronderstelling moet echter worden gevalideerd door middel van gegevensgestuurde analyse. De gezaghebbende methode voor het oplossen van TCP-problemen is packet capture, idealiter uitgevoerd:

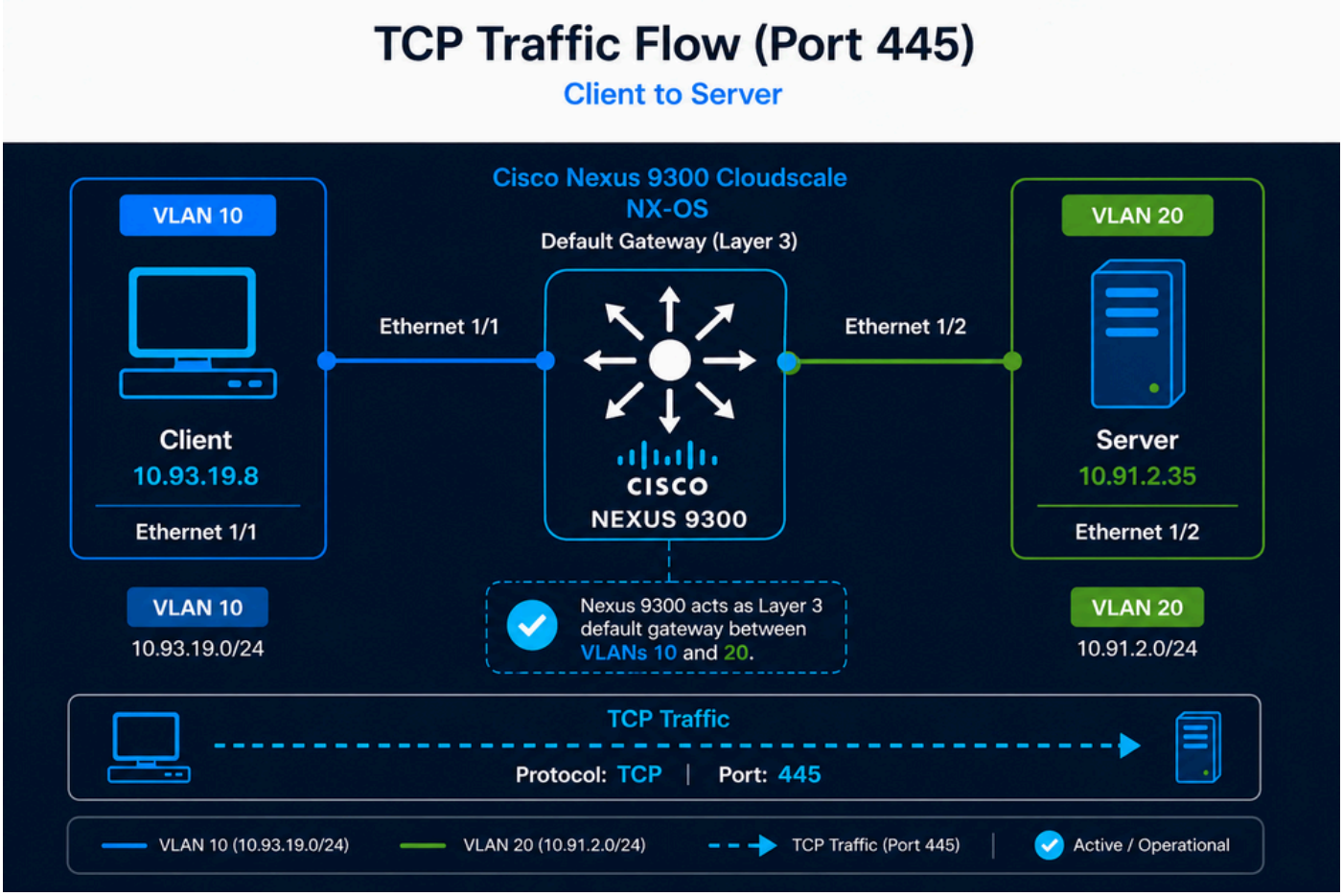
- Gelijktijdig aan de bron en de bestemming
- Voordat het verkeer wordt gestart

Dit zorgt voor zichtbaarheid in de TCP drieweg handshake, waar kritische parameters zoals MSS, Window Scale en SACK worden onderhandeld en niet later in de sessie worden herhaald. Als gelijktijdige vastlegging niet mogelijk is, kan de analyse worden voortgezet met één vastlegging, maar de conclusies zijn beperkt.

scenariodefinitie

Een gebruiker heeft vastgesteld dat het back-upproces voor een toepassingsdataset van ongeveer 7,5 TB, dat eerder in ongeveer 9 uur was voltooid, nu bijna 21 uur in beslag neemt. Hoewel TCP-sessies tussen de client en de server nog steeds met succes zijn ingesteld, wijst de aanzienlijke toename van de back-upduur op een mogelijke verslechtering van de doorvoer of de algehele TCP-prestaties. Omdat de Nexus-switch het enige netwerkapparaat in het pad is en ook Layer 3-gatewayfunctionaliteit biedt, vermoedt de netwerkbeheerder dat de Nexus-switch de oorzaak van het probleem is.

- Client: 10.93.19.8 (VLAN 10)
- Server: 10.91.2.35 (VLAN 20)
- Nexus 9300 fungeert als standaardgateway
- TCP-poort 445



Eerste validatie (bereikbaarheid)

- Deze opdrachten worden gebruikt om het pad MTU (PMTU) tussen een bron en bestemming te valideren door ICMP-pakketten te verzenden met de niet-fragmentatieset (DF). Dit helpt bij het bepalen van de maximale pakketgrootte die het netwerk kan doorkruisen zonder fragmentatie. Dit proces moet zowel op de bron als op de bestemming worden uitgevoerd.
- Controleer altijd de MTU van de fysieke interface op zowel de bron als de bestemming.
- In dit scenario is toegang alleen beschikbaar voor de bronhost, waar een MTU van 1500 is geïdentificeerd.

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

- -c 10 → Verzendt 10 ICMP-echoverzoeken
- -I 192.168.10.10 → Gebruikt deze specifieke bron/IP-interface
- -s 1472 → Hiermee wordt de grootte van de ICMP-payload ingesteld op 1472 bytes
- -M do → Stelt de DF-bit (Don't Fragment) in
- 192.168.20.20 → IP Bestemming

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

- -n 10 → Verzendt 10 ICMP-echoverzoeken
- -l 1472 → Hiermee wordt de grootte van de ICMP-payload ingesteld op 1472 bytes
- -f → Hiermee stelt u de markering Niet fragmenteren (DF) in
- 192.168.20.20 → IP Bestemming

Waarom 1472 bytes?

- ICMP-payload = 1472 bytes
- IP-header = 20 bytes
- ICMP-header = 8 bytes
- Totale pakketgrootte: 1472 + 20 + 8 = 1500 bytes (standaard MTU)
- Hiermee wordt getest of het pad een 1500-byte MTU ondersteunt zonder fragmentatie. Als u probeert 1500 bytes aan ICMP-payload te verzenden, kan de ping mislukken omdat de totale pakketgrootte de standaard MTU zou overschrijden na het toevoegen van de IP- en ICMP-headers.

Wat kan worden geconcludeerd

- Als de ping slaagt (geen pakketverlies), ondersteunt het pad ten minste een MTU van 1500 bytes en is er geen fragmentatie vereist.
    - ICMP-resultaten reinigen → doorgaan naar TCP-analyse
    - Intermitterend ping-succes → mogelijk pakketverlies, voorbijgaande congestie, snelheidsbeperking of een probleem met doorsturen; ga verder met analyse van pakketverlies, omdat TCP een verliesvrij pad vereist om efficiënt te werken.
- Als de ping mislukt met de fout "Fragmentatie vereist" of time-out, is er een koppeling in het pad met MTU lager dan 1500 bytes, het pakket kan niet worden doorgestuurd vanwege de DF-bit, en dit duidt op een Path MTU probleem.

Hoe dit te gebruiken voor probleemoplossing

- Verklein geleidelijk de payloadgrootte (bijvoorbeeld 1472 → 1400 → 1300) om de grootste grootte te identificeren die slaagt.
- Bereken na identificatie de MTU met behulp van de formule MTU = payload + 28 bytes (IP + ICMP-headers).

Praktische relevantie voor TCP

- Als MTU kleiner is dan verwacht, kunnen TCP-segmenten worden gefragmenteerd of

weggelaten.
- Dit leidt tot heruitzendingen, verhoogde latentie en verminderde doorvoer, wat direct van invloed is op de prestaties van toepassingen.

Verkeerspad identificeren (interfaces)

Om de TCP-prestaties op een Cisco Nexus 9000-switch effectief op te lossen, is het van essentieel belang om te bepalen welke interfaces het verkeer tussen de bron en de bestemming ontvangen en doorsturen.

In eenvoudige topologieën kan dit direct worden afgeleid uit de fysieke verbindingen. Als de client bijvoorbeeld is verbonden met Ethernet1/1 en de server met Ethernet1/2, is het verkeerspad eenvoudig. In echte omgevingen met meerdere actieve interfaces, poortkanalen of vPC-configuraties is deze identificatie echter niet altijd triviaal.

In dergelijke gevallen is de aanbevolen aanpak om Embedded Logic Analyzer Module (ELAM) te gebruiken, die zichtbaarheid biedt op het niveau van ASIC (data-plane hardware).

Met ELAM kunt u een pakket vastleggen terwijl het wordt verwerkt door de doorstuurpijplijn en kritieke informatie onthult, zoals:

- Ingress-interface
- Uitganginterface
- Beslissing tot doorsturen (zoekresultaat L2/L3)

Deze methode is aanzienlijk nauwkeuriger dan vertrouwen op control-plane tools, omdat het de werkelijke hardware forwarding pad weerspiegelt.

Het is belangrijk op te merken dat ELAM slechts één pakket tegelijk vastlegt, dus de filtercriteria moeten nauwkeurig worden gedefinieerd om overeen te komen met het gewenste verkeer (bijvoorbeeld bron-IP, bestemming-IP, TCP-poort). Als filters te breed zijn, bestaat het risico dat niet-gerelateerd verkeer zoals ICMP of UDP wordt vastgelegd in plaats van de beoogde TCP-stroom.

Bovendien moet dit proces voor beide verkeersrichtingen worden herhaald:

- Bron → Bestemming
- Bestemming → Bron

In omgevingen waarin vPC of ECMP wordt gebruikt, kan het verkeer over meerdere paden

worden verdeeld. Als gevolg hiervan kan voorwaarts- en terugkeerverkeer verschillende switches of interfaces doorlopen. In deze scenario's moet ELAM op elke desbetreffende Nexus-switch worden uitgevoerd om volledige zichtbaarheid te garanderen.

Door de interfaces voor ingangen en uitgangen nauwkeurig te identificeren, wordt de omvang van het oplossen van problemen aanzienlijk beperkt, waardoor gerichte validatie van interfacetellers, QoS-beleid, MTU-instellingen en potentiële congestiepunten langs het exacte doorstuurpad mogelijk wordt.

ELAM-configuratie (Nexus 9300 Cloud Scale)

Dit voorbeeld filtert verkeer met bron IP 10.93.19.8, bestemming IP 10.91.2.35 en TCP-bestemmingspoort 445.

ELAM-installatie

<#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
```

Nadat u het verkeer hebt gegenereerd, haalt u het resultaat op:

<#root>

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

report
```

Reverse Traffic Capture (verplicht voor volledig zicht)

Als u het retourpad wilt valideren, herhaalt u de configuratie door de IP-adressen van de bron en de bestemming te verwisselen:

<#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
```

Operationele opmerkingen

- ELAM vangt slechts één pakket op, dus zorg ervoor dat het verkeer actief stroomt wanneer u de opname start.
- Filters moeten nauwkeurig zijn om te voorkomen dat niet-gerelateerd verkeer wordt vastgelegd.
- In vPC-omgevingen voert u ELAM op beide switches uit, omdat het verkeer in elke richting anders kan hash.
- De ingangsinterface van het uitgangsscherm, de uitgang-interface en de beslissing tot doorsturen in de hardware bieden een gezaghebbende zichtbaarheid in het gegevensvlak.

referentie

Cisco Nexus 9000 Cloud Scale ASIC ELAM Guide

validatie op interfaceniveau

De validatie op interfaceniveau zorgt ervoor dat de Nexus-switch geen restricties of anomalieën invoert die het TCP-verkeer beïnvloeden. De focus ligt op het bevestigen dat configuratie-, operationele status- en hardwaretellers consistent zijn met het verwachte gedrag voor het doorsturen van high-performance data-plane.

Configuratievalidatie

- Controleer of er geen restrictieve ACL's op de interfaces worden toegepast:

<#root>

```
switch#
```

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

- Valideren dat geen onbedoeld QoS-beleid van invloed is op het verkeer (interface-niveau en wereldwijde QoS, inclusief wachtrijen, bewaking en vormgeving):

<#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

- Bevestig de configuratie van Layer 2 of Layer 3 (switchport versus gerouteerde interface), inclusief VLAN-lidmaatschap, STP-status en IP-adres:

<#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

validering van de operationele toestand

- Controleer de consistentie van de MTU en zorg ervoor dat deze overeenkomt met de verwachte configuratie (bijvoorbeeld 1500 of 9000 bytes):

<#root>

switch#

show interface ethernet1/1-2 | include MTU

- Bevestig de interfacesnelheid en duplexinstellingen:

<#root>

switch#

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

- Valideer de stabiliteit van de interface (geen flapperen of frequente linkovergangen):

<#root>

switch#

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

Validatie van foutteller

- Tellers wissen voordat u test:

<#root>

```
switch#
```

```
clear counters interface all
```

- Fouttellers bewaken (alleen niet-nulwaarden):

<#root>

```
switch#
```

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

validering na de test

- Voer de TCP-verkeerstest opnieuw uit en observeer de tellers opnieuw:

<#root>

```
switch#
```

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

- De tellers mogen niet worden verhoogd; elke toename duidt op potentiële Layer 1- of hardwaregerelateerde problemen zoals fouten in fysieke koppelingen, CRC/FCS-fouten of bufferoverschrijdingen/-dalingen.

Routering en ARP-stabiliteit

Het garanderen van routering en ARP-stabiliteit is van cruciaal belang om te bevestigen dat de

Nexus-switch een consistente Layer 3-bereikbaarheid heeft en geen intermitterende oplossingsproblemen invoert die van invloed kunnen zijn op de TCP-prestaties. Instabiliteit in routeringsitems of ARP-resolutie kan leiden tot pakketverlies, verhoogde latentie of blokkering van het verkeer.

Validatiecriteria

- Routeringsitems voor bron en bestemming moeten aanwezig, stabiel en niet vaak gewijzigd zijn.
- ARP-items moeten worden opgelost en niet voortdurend worden vernieuwd of ontbreken.

<#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
```

Controleren of het verkeer niet op de CPU is ingesteld

In Cisco Nexus 9000-switches wordt forwarding uitgevoerd in hardware (ASIC) en de CPU is niet betrokken bij normale activiteiten van het gegevensplatform. Daarom is het observeren van host-to-host TCP-verkeer in het besturingsvlak abnormaal en geeft aan dat pakketten worden gepunteerd vanwege uitzonderingen of misconfiguraties. Zodra het verkeer door de CPU moet worden verwerkt, wordt het onderworpen aan Control Plane Policing en wordt verwacht dat er dalingen kunnen worden waargenomen als het verkeer de toegestane control-plane rate overschrijdt.

validatiemethode

- Verkeer vastleggen dat het controlevlak bereikt met behulp van Ethanalyzer:

<#root>

```
switch#

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

verwacht gedrag

- In de CPU kan geen TCP-dataverkeer van host naar host worden waargenomen.

Onverwacht gedrag

- Als pakketten die overeenkomen met de stroom zichtbaar zijn, wordt het verkeer gepunteerd, wat kan worden veroorzaakt door:
  - Uitzonderlijke pakketverwerking (TTL-expiratie, ACL-logboekregistratie, omleidingen)
  - Misconfiguratie of niet-ondersteunde functies
  - Onjuiste hardwareprogrammering

Packet Forwarding Latency bepalen

Packet forwarding latency in Nexus 9000-switches is afhankelijk van pakketgrootte, doorstuurmodus en functies die zijn ingeschakeld. Cisco-specificaties verwijzen doorgaans naar latentie voor pakketten van 64 bytes onder doorsnijding.

```
+----------------------+----------------------+--------------------------+------------------------------
| 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

- Doorsnijden doorsturen (standaard in Nexus 9000):
  - Begint met doorsturen voordat het volledige pakket is ontvangen.
  - Minimaliseert latentie (submicroseconde tot ~1 µs).
- Store-and-forward:
  - Het volledige pakket moet worden ontvangen voordat het wordt doorgestuurd.
  - Voegt latentie toe in verhouding tot de pakketgrootte.

Extra functies kunnen incrementele latentie introduceren:

- VXLAN-inkapseling/decapsulatie
- ACL-opzoeken (TCAM-verwerking)
- QoS-classificatie en wachtrij
- Telemetrie (NetFlow, ERSPAN, sFlow)
- Buffering tijdens congestie

Echter:

- Deze handelingen worden uitgevoerd in hardwarepijpleidingen.

Het enige realistische scenario waarbij de latentie merkbaar toeneemt, is congestie:

- Pakketten worden gebufferd in egress wachtrijen.
- Vertraging is afhankelijk van:
  - Wachtrijdiepte
  - Interfacegebruik
  - QoS-beleid

Zelfs in deze gevallen:

- Latentie is meestal in de microseconden tot laag honderden microseconden bereik.
- Aanhoudende vertraging op millisecondeniveau zou het volgende inhouden:
  - Ernstige congestie
  - overinschrijving
  - Verkeerde QoS of buffering

SPAN naar CPU (Packet Capture for Data Plane)

Dit maakt het spiegelen van data-plane verkeer in de control-plane voor packet capture en exporteren naar een .pcaping bestand, waardoor gedetailleerde analyse in Wireshark.

Configuratie

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

Capture-uitvoering

<#root>

```
switch#
```

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

Technische overwegingen

- Het verkeer dat wordt gespiegeld naar de CPU is onderworpen aan Control Plane Policing (CoPP).
- Als het verkeer de CoPP overschrijdt:
  - Pakketten kunnen alleen in control-plane worden gedropt.
  - Dit zorgt voor false positives tijdens de analyse.
- SPAN naar CPU wordt aanbevolen voor scenario's met weinig tot matig verkeer.
- Voor omgevingen met hoge doorvoer:
  - Gebruik lokale SPAN (externe analysator)
  - Gebruik ERSPAN voor externe vastlegging

methode	voordeel	beperking
OVERSPANNEN	Nauwkeurig, geen inkapseling	Hiervoor is een fysieke verbinding nodig.
RESPAN	Opnamemogelijkheid op afstand	Gevoelig voor netwerkcongestie.

snelheidsbegrenzende validatie van het regelvlak

Om ervoor te zorgen dat SPAN-naar-CPU-opnamen betrouwbaar zijn, is het noodzakelijk om te valideren dat het controlevlak geen gespiegelde pakketten laat vallen vanwege snelheidsbeperkingen.

Validatiecommando

```
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
```

Validatiemethode

- Voer de opdracht uit met intervallen van ~3 seconden.
- Let op de SPAN-gerelateerde valtellers.

interpretatie

- Geen toename in valtellers voor SPAN-rij duidt op een betrouwbare opname.
- Toenemende valtellers wijzen op pakketverlies in het controlevlak, waardoor de opname onbetrouwbaar is.

Als er druppels worden waargenomen, moet de vangmethode worden gewijzigd in SPAN of ERSPAN.

ICMP-gebaseerde validatie vóór TCP

De ICMP-test biedt een basisvalidatie van de integriteit van het gegevensvlak voordat een complexe TCP-analyse wordt uitgevoerd. Omdat ICMP stateloos en eenvoudiger is, kan hiermee snel pakketverlies, duplicatie of inconsistenties in het pad worden gedetecteerd.

Verwacht gedrag in SPAN-opname

- Elk ICMP-pakket kan twee keer worden weergegeven:
  - Eenmaal bij binnenkomst
  - Eenmaal op uitgang
- Voor een standaard ping:
  - Echo Request → 2 pakketten
  - Echo Reply → 2 pakketten

Dit bevestigt correct doorsturen en afwezigheid van pakketverlies in het gegevensvlak.

Abnormaal gedrag

- Ontbrekende duplicaten of asymmetrische pakkettellingen wijzen op mogelijk verlies van pakketten of beperkingen voor het vastleggen van pakketten.
- Intermitterende time-outs suggereren Layer 1-problemen, congestie of stroomopwaartse problemen.

Als ICMP-verkeer consequent zonder verlies wordt doorgestuurd, is de kans groot dat TCP-verkeer ook correct wordt doorgestuurd op Layer 2/3.

Nexus Switch Forwarding Latency bepalen met behulp van Packet Capture

Wanneer verkeer wordt vastgelegd met behulp van SPAN naar CPU (of SPAN / ERSPAN), kan elk pakket twee keer worden waargenomen: één keer bij ingress en één keer bij egress. Deze duplicatie kan worden gebruikt om de doorstuurlatentie te schatten die door de Nexus-switch wordt geïntroduceerd door het tijdverschil tussen beide instanties van hetzelfde pakket te berekenen.

In de praktijk kan deze latentie worden gemeten met behulp van het eerder vastgelegde ICMP-verkeer door de tijddelta tussen gedupliceerde Echo Request- en Echo Reply-pakketten te vergelijken. Dit biedt een eenvoudige en effectieve basislijn voor het doorsturen van switches. Als een diepere analyse nodig is, kan dezelfde methodologie worden toegepast op TCP-verkeer door de stroom vast te leggen en het tijdsverschil tussen gedupliceerde TCP-pakketten te meten.

methodologie

- Identificeer een pakket en het duplicaat ervan (hetzelfde volgnummer).
- Meet de tijddelta tussen de ingress- en egress-kopieën.
- Deze delta is een bovengrens schatting van de switch forwarding latency, omdat het kan spiegelen en tijdstempelen overhead bevatten.

Wireshark-configuratie

- Tijdsdelta-weergave inschakelen:

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

- Een aangepaste kolom toevoegen voor tijddelta:

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

- Filter relevant verkeer (voorbeeld):

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

- Pakketten sorteren op volgnummer of TCP-stream:

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

interpretatie

- De tijdsdelta tussen gedupliceerde pakketten kan in het microseconde-bereik liggen.
  - Als dit het geval is, introduceert Nexus switch geen latentie voor het doorsturen van pakketten.
- Consistente lage delta's bevestigen hardwarematige forwardingsprestaties.
- Hogere of inconsistente delta's kunnen wijzen op:
  - Congestie of buffering

Referenties

- Cisco Nexus 9000-reeks — Gegevensbladen
- Cisco Nexus 9000-serie Switches Design Guides
- Intelligent bufferbeheer voor Cisco Nexus 9000-serie Switches Whitepaper

TCP-verkeersanalyse van het vastleggen van het bronhostpakket

Deze sectie biedt een gedetailleerde methodologie voor het analyseren van een TCP-pakketopname in Wireshark, inclusief de profielconfiguratie, door middel van de hypothetische casus die eerder is beschreven. De getoonde afbeeldingen zijn rechtstreeks afkomstig van Wireshark. Ter herinnering: het scenario is:

Een gebruiker heeft vastgesteld dat het back-upproces voor een toepassingsdataset van ongeveer 6,5 TB, dat eerder in ongeveer 9 uur was voltooid, nu bijna 21 uur in beslag neemt. Het enige toegankelijke netwerkapparaat is een Cisco Nexus 9300-switch die is aangesloten op de bronserver (10.93.19.8). De MTU geconfigureerd op de switch-interface is 9000 bytes (jumbo frames), terwijl de MTU op de server is onbekend. Er is een pakketopname van de bronserver beschikbaar en alle eerdere Nexus-validatiestappen zijn al voltooid zonder dat er afwijkingen zijn gedetecteerd.

Belangrijkste opmerkingen en beperkingen

- Nexus-switch is uitgesloten:
    - Geen pakketdruppels
    - Geen interfacefouten
    - Geen impact op QoS of ACL
    - Hardware forwarding bevestigd
- Interface-configuratie:
    - Toegangspoort
    - MTU: 9000 bytes
- Beschikbare gegevens:
    - Packet capture aan de bron
    - End-to-end MTU-kennis
        - De ping is succesvol voltooid zonder fragmentatie met behulp van een 1500-byte pakket met 1.472 bytes aan gegevens.
- Ontbrekende gegevens:
    - Zichtbaarheid van bestemming
    - Er is geen pakketopname beschikbaar op de bestemmingsserver.

In Wireshark kunt u aangepaste profielen maken die zijn afgestemd op het specifieke type analyse dat u wilt uitvoeren.

kolombeschrijving

- tcp.analysis.initial_rtt (iRTT): Schat de initiële retourtijd op basis van de TCP-drieweghandshake.

- tcp.analysis.ack_rtt (ACK RTT): meet de tijd tussen een TCP-segment en de bijbehorende erkenning.
- tcp.window_size (Window): Geeft de door de ontvanger geadverteerde grootte van het TCP-venster aan voordat de schaal wordt toegepast.
- tcp.options.wscale.multiplier (Multi): geeft de vensterschaalfactor weer die wordt gebruikt om het effectieve ontvangstvenster te berekenen.
- tcp.seq (Seq#): Geeft het volgnummer van de eerste byte in het TCP-segment weer.
- tcp.len (Payload): toont de grootte van de TCP-payload in bytes voor dat segment.
- tcp.ack (ACK#): Geeft de volgende verwachte byte van de afzender aan (cumulatieve bevestiging).
- tcp.options.mss_val (MSS): geeft de maximale segmentgrootte weer die tijdens de TCP-handshake wordt geadverteerd.
- ip.ttl (TTL): Toont de waarde Time To Live, handig voor het identificeren van het aantal hop en het routeringsgedrag.
- tcp.analysis.bytes_in_flight (Bytes in Flight): geeft de hoeveelheid niet-erkende gegevens weer die momenteel worden verzonden.

Analyse van de TCP drieweg handshake

Het vastleggen van de TCP-handshake in drie richtingen is verplicht omdat deze kritieke parameters bevat zoals MSS, Window Scale en SACK die bepalen hoe de sessie zich gedraagt. Zonder deze informatie is elke TCP-analyse onvolledig en kan deze leiden tot onjuiste conclusies over prestaties of onderliggende oorzaken.

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			

verkeersidentificatie

Uit de pakketopname:

- Bron IP-adres: 10.93.19.8
- IP-adres bestemming: 10.91.2.35

Initiële analyse van de rondreistijd (iRTT)

De initiële RTT (iRTT) wordt berekend als:

- iRTT = 798 microseconden

Deze waarde is afgeleid van:

- Pakket 2 (SYN-ACK) ACK RTT: 750 µs → Tijd voor de bestemming om te reageren op de SYN.
- Pakket 3 (ACK) ACK RTT: 48 µs → Tijd voor de bron om de SYN-ACK te bevestigen.

Het grootste deel van de latentie (~94%) bevindt zich in het voorwaartse pad (client → server → client), terwijl de responstijd van de bron minimaal is, wat aangeeft dat er geen CPU- of toepassingsvertraging is op de client.

TCP-poortidentificatie

- TCP-poort van bestemming: 445

Poort 445 komt overeen met Microsoft Server Message Block (SMB), dat vaak wordt gebruikt voor het delen van bestanden, netwerkstations en Windows-verificatieservices. Dit protocol is gevoelig voor zowel latentie als doorvoer, waardoor het sterk afhankelijk is van TCP-efficiëntie en netwerkstabiliteit.

Analyse van de grootte van het TCP-venster

- Bronvenster (geschaald): 64.240 bytes
- Bestemmingsvenster: 65.535 bytes

Het TCP-venster geeft aan hoeveel gegevens kunnen worden verzonden voordat wordt gewacht op bevestiging. In dit geval is de bron iets restrictiever dan de bestemming. Deze waarden zijn relatief klein voor moderne omgevingen en kunnen de doorvoer beperken, vooral als RTT toeneemt.

De maximale theoretische doorvoer kan worden geschat met behulp van:

Doorvoer = TCP-venstergrootte / RTT

Vervanging van de waargenomen waarden:

- Grootte TCP-venster = 64.240 bytes
- RTT = 798 microseconden = 0,000798 seconden

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

Dit is de bovengrens van de doorvoer, uitgaande van:

- Geen pakketverlies
- Geen heruitzendingen
- Ideale netwerkomstandigheden

Doorvoersnelheid, overdrachttijd en analyse van vereiste voorwaarden

Bij de huidige doorvoer van 644 Mbps duurt het overbrengen van een 6,5 TB-bestand ongeveer 23,5 uur, wat overeenkomt met de waargenomen degradatie. Om een 9-uurs overdrachtsvenster te bereiken, moet de doorvoer toenemen tot ongeveer 1,68 Gbps, waarbij een groter TCP-venster (~2,7x toename) of een aanzienlijk lagere RTT (~291 µs) nodig is.

Met de huidige omstandigheden (64 KB venster en ~798 µs RTT), is het niet mogelijk om de 9-uur doelstelling te bereiken, omdat de TCP doorvoer wordt beperkt door de bandbreedte-delay product. Zonder de venstergrootte te vergroten of de latentie te verminderen, kan het protocol geen hogere beschikbare bandbreedte gebruiken, waardoor het doel onbereikbaar wordt.

Scenario	doorvoer	Geschatte overdrachttijd (6,5 TB)	Vereist TCP-venster	Vereiste RTT
Huidige staat	644 Mbps (~80,5 MB/s)	~23,5 uur	64 KB	798 µs
Doel (9 uur)	~1683 Mbps (~210 MB/s)	9 uur	~172 KB	~291 µs

Dit werkte eerder, wat aangeeft dat er een wijziging is opgetreden in het netwerk, de toepassing, de bron of de bestemming. Het is belangrijk op te merken dat, alleen al op basis van deze eerste analyse, een belangrijke conclusie al kan worden vastgesteld: onder de huidige TCP-venstergrootte en RTT-voorwaarden is het niet mogelijk om de 9-urendoelstelling te bereiken.

De tabellen laten een vergelijking zien van hoe de doorvoer varieert naarmate de RTT- en TCP-venstergrootte groter of kleiner worden.

RTT-impact op doorvoer (vaste venstergrootte = 64.240 bytes)

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

RTT	Doorvoer (MB/s)	Doorvoer (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

Impact van TCP-venstergrootte (vaste RTT = 798 µs)

TCP-venstergrootte	Doorvoer (MB/s)	Doorvoer (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	~2.624 Mbps
1 MB (1.048.576 B)	~1.314 MB/s	~10,5 Gbps

technische interpretatie

- De doorvoer is omgekeerd evenredig met RTT → hogere latentie vermindert de prestaties.
- De doorvoer is recht evenredig met de grootte van het TCP-venster → grotere vensters verhogen de capaciteit.
- Kleine venstergroottes beperken de doorvoer aanzienlijk, zelfs in omgevingen met lage latentie.
- Hogesnelheidsnetwerken (10G+) vereisen vensterschaling om volledig gebruik te maken van bandbreedte.

Dit toont aan dat zowel de RTT- als de TCP-venstergrootte cruciale factoren zijn voor de TCP-prestaties en samen moeten worden geanalyseerd bij het oplossen van doorvoerproblemen.

Lengte van IP- en TCP-header

- Lengte IP-header: 20 bytes
- Lengte TCP-header: 32 bytes

Een 20-byte IP-header geeft aan dat er geen IP-opties aanwezig zijn. De 32-byte TCP-header bevestigt dat TCP-opties worden gebruikt en voegt 12 bytes toe buiten de basisheader. Deze

opties omvatten meestal MSS, Vensterschaal en TOEGESTANE SACK.

Analyse van TCP-opties en TTL

Selectieve erkenning (SACK) is ingeschakeld op beide eindpunten. Dit is niet zichtbaar op de foto. Met SACK kan de ontvanger niet-aaneengesloten gegevensblokken herkennen en de afzender precies vertellen welke segmenten met succes zijn ontvangen.

Als bijvoorbeeld de segmenten 1000-2000 en 3000-4000 worden ontvangen, maar 2000-3000 ontbreekt, kan de ontvanger dit expliciet aangeven. Zonder SACK zou de afzender alle gegevens na de gap opnieuw verzenden; met SACK wordt alleen het ontbrekende deel opnieuw verzonden. Dit verbetert de prestaties aanzienlijk in omgevingen met pakketverlies.

Analyse van pakket 1 (SYN)

- Volgnummer: 0 (Wireshark genormaliseerd)
- Payload: 0 bytes
- ACK#: 0
- MSS: 1460 bytes
- TTL: 128

Wireshark normaliseert het volgnummer tot nul voor leesbaarheid, hoewel het in de praktijk een grote willekeurige waarde is. De afwezigheid van nuttige lading wordt verwacht tijdens het instellen van de verbinding. De MSS-waarde van 1460 bytes geeft een MTU van 1500 bytes aan (20 bytes IP-header + 20 bytes TCP-header). Een TTL van 128 kan een Windows-gebaseerde host zijn en het zien van deze waarde in de opname geeft aan dat de opname waarschijnlijk op of in de buurt van de bron is gemaakt via Layer 2.

Analyse van pakket 2 (SYN-ACK)

- ACK#: 1

De ACK-waarde is 1 omdat de SYN-vlag één volgnummer verbruikt, zelfs als er geen lading aanwezig is. Daarom is ACK = SEQ + 1.

- TTL: 59

De waargenomen TTL van 59 suggereert een initiële TTL van 64, wat betekent dat het pakket ongeveer 5 routerende hop doorkruiste (64 - 59 = 5). Elke gerouteerde hop verlaagt de TTL met één.

Risico van fragmentatie en impact op het netwerk

De aanwezigheid van ongeveer vijf routinghops brengt potentiële prestatierisico's met zich mee, met name in verband met MTU-mismatches en fragmentatie.

Als een tussenliggende link een lagere MTU heeft dan de oorspronkelijke pakketgrootte, kan fragmentatie optreden. Dit heeft verschillende gevolgen:

- Verhoogde latentie als gevolg van fragmentatie en hermontage overhead.
- Hogere kans op pakketverlies, omdat het verliezen van een enkel fragment hertransmissie van het hele pakket vereist.
- Verminderde doorvoer, omdat TCP verlies interpreteert als congestie en de verzendsnelheid verlaagt.
- Verhoogd CPU-gebruik op netwerkapparaten die fragmentatie verwerken.
- Risico op fouten in Path MTU Discovery (PMTUD) als ICMP wordt geblokkeerd, wat resulteert in stille pakketdalingen.

Gezien deze factoren is het van cruciaal belang om consistente MTU over het pad te garanderen of waar nodig MSS-klemmen te implementeren.

TCP RTT-analyse: ACK RTT vs initiële RTT

Wanneer ACK RTT groter is dan iRTT, geeft dit aan dat de latentie is toegenomen in vergelijking met de baseline die is vastgesteld tijdens de TCP-handshake.

Dit betekent dat het netwerk of de eindpunten tijdens de sessie extra vertraging veroorzaken, meestal als gevolg van:

- Netwerkcongestie of wachtrij
- Vertragingen bij ontvangst of verwerking van aanvraag
- Tussenliggende apparaten (firewalls, load balancers)
- heruitzendingen

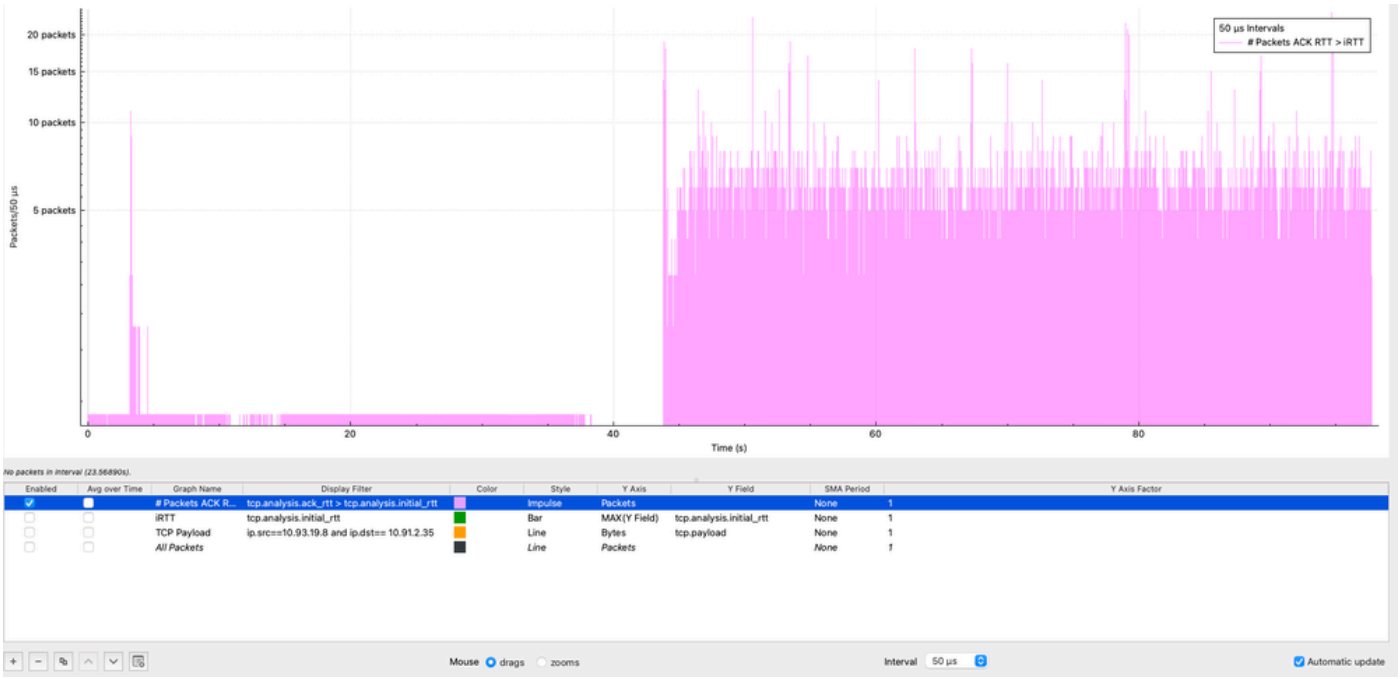
Als deze voorwaarde tijdens de TCP-sessie blijft bestaan, leidt dit tot:

- Verminderde TCP-doorvoer
- Inefficiënt venstergebruik
- Verslechterde toepassingsprestaties

In Wireshark is het mogelijk om te visualiseren hoe vaak de conditie ACK RTT > iRTT optreedt

door gebruik te maken van de I/O Graphs functie onder: Statistics → I/O Graphs, het toepassen van het display filter (tcp.analysis.ack_rtt > tcp.analysis.initial_rtt), het selecteren van Impulse stijl, het instellen van de Y-as naar Packets, en het gebruik van een interval van 50 microseconden.

In de grafiek geven de paarse impulsen het aantal pakketten weer dat aan deze voorwaarde voldoet binnen elk interval van 50 microseconden. Zoals opgemerkt, blijft deze voorwaarde gedurende de gehele pakketopname bestaan, wat aangeeft dat de latentie tijdens de sessie consistent hoger is dan de initiële basislijn. Dit gedrag wijst sterk op een aanhoudende verslechtering van de prestaties in plaats van een voorbijgaande aandoening, wat de noodzaak versterkt om potentiële bronnen zoals congestie, buffering of vertragingen in de verwerking van eindpunten over het end-to-end pad te onderzoeken.



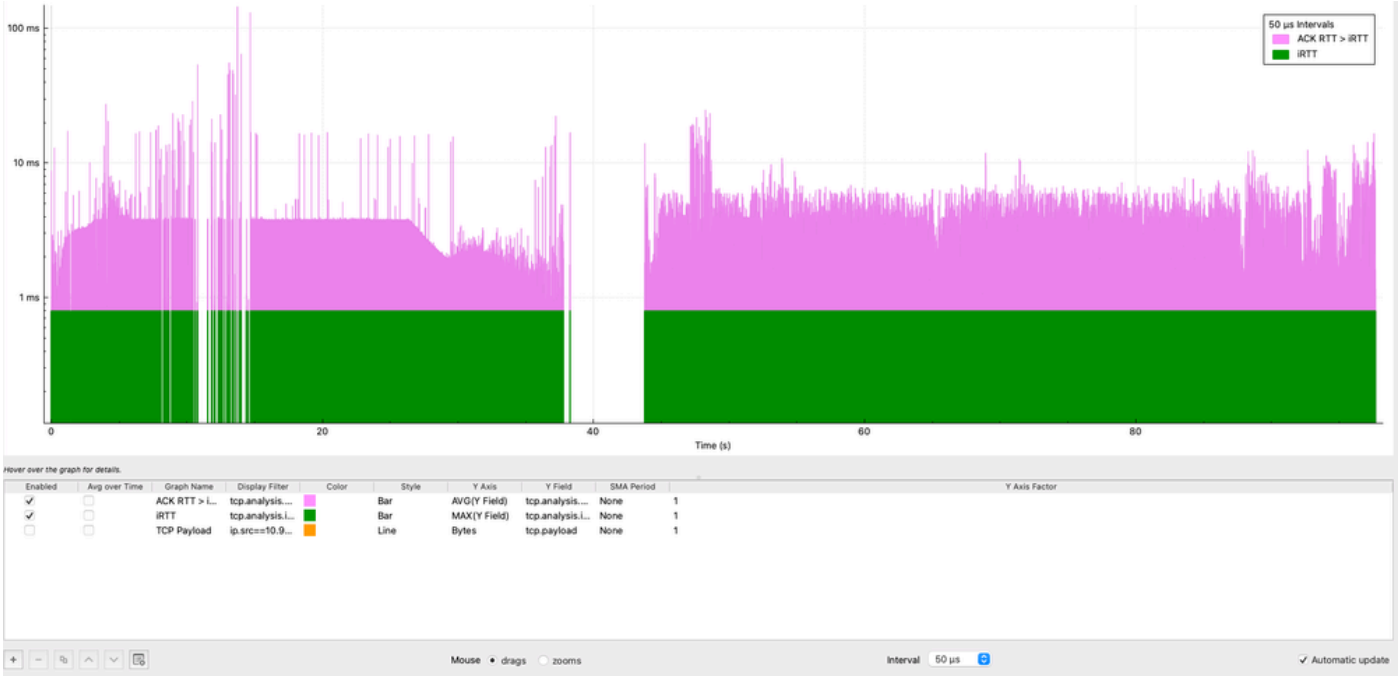
Het is ook belangrijk om te bepalen hoe lang de iRTT wordt overschreden, niet alleen hoe vaak. Hoewel Wireshark niet direct aftrekken tussen velden toestaat, kan een visuele vergelijking worden bereikt met behulp van I / O-grafieken:

- Navigeer naar Statistieken → I/O Grafieken
- Grafiek 1:
  - Weergavefilter: tcp.analysis.ack_rtt > tcp.analysis.initial_rtt
  - Stijl: balk
  - Y-as: AVG
  - Y Veld: tcp.analysis.ack_rtt
  - Interval: 50 microseconden
- Grafiek 2:
  - Weergavefilter: tcp.analysis.initial_rtt
  - Stijl: balk
  - Y-as: MAX

    - Y Veld: tcp.analysis.initial_rtt
- Klik vervolgens met de rechtermuisknop op de grafiek en schakel de logschaal in.

In deze visualisatie vertegenwoordigt de paarse grafiek de toestand ACK RTT > iRTT, die consistent aanwezig is tijdens de hele TCP-sessie. De gegevens tonen aanhoudende latentie-inflatie, met meerdere pieken bereiken van 11 milliseconden en een maximale piek van meer dan 100 milliseconden, wat neerkomt op 11x tot 100x de baseline iRTT.

Dit gedrag bevestigt dat de latentieverhoging niet van voorbijgaande aard is, maar aanhoudend, wat wijst op een systemisch probleem dat de sessie in de loop van de tijd beïnvloedt. Een dergelijke aanhoudende afwijking duidt sterk op factoren zoals netwerkcongestie, buffering (bufferbloat) of vertragingen in de verwerking van eindpunten.



TCP-hertransmissies en analyse van valse hertransmissies

In dit gedeelte wordt de betrouwbaarheid van TCP geëvalueerd door hertransmissies in de loop van de tijd te analyseren, zodat kan worden gevalideerd of pakketverlies bijdraagt aan de verslechtering van de prestaties.

TCP-hertransmissies in de tijd

De grafiek toont de verdeling van TCP-heruitzendingen in de tijd. In totaal werden 42 heruitzendingen waargenomen, die slechts 0,00125% van het totale verkeer vertegenwoordigen.

Dit niveau van heruitzendingen is verwaarloosbaar en geeft duidelijk aan dat pakketverlies geen bijdragende factor is in dit scenario.

Wireshark-configuratie (TCP-heruitzendingen)

```
Statistics → I/O Graphs
```

- Weergavefilter:

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

- Stijl: Impuls of Bar
- Y-as: pakketten
- Interval: 1 sec

TCP Spurious Retransmissions

De grafiek toont het aantal TCP Spurious Retransmissions in 1 sec intervallen gegenereerd door de bron 10.93.19.8.

In Wireshark geeft een TCP Spurious Retransmissie aan dat een host een segment dat niet daadwerkelijk verloren is gegaan, opnieuw heeft verzonden. Het oorspronkelijke pakket bereikte de ontvanger met succes, maar de afzender ging ten onrechte uit van verlies als gevolg van een onjuiste schatting van de timing. Dit gedrag duidt niet op echt pakketverlies, maar eerder op inefficiënte retransmissielogica bij de afzender.

In deze opname:

- De bron 10.93.19.8 verzendt pakketten na slechts ~8 microseconden opnieuw.
- Terwijl typische retransmissietimers in de orde van ~ 200 milliseconden zijn.

Dit bevestigt dat het doorgiftegedrag volledig wordt gecontroleerd door de TCP-bronstack, niet door het netwerk.

Het totale aantal waargenomen valse heruitzendingen is 1.112, wat overeenkomt met 0,0332% van het totale gevangen verkeer.

Wireshark-configuratie (TCP Spurious Retransmissions)

`Statistics → I/O Graphs`

- Weergavefilter:

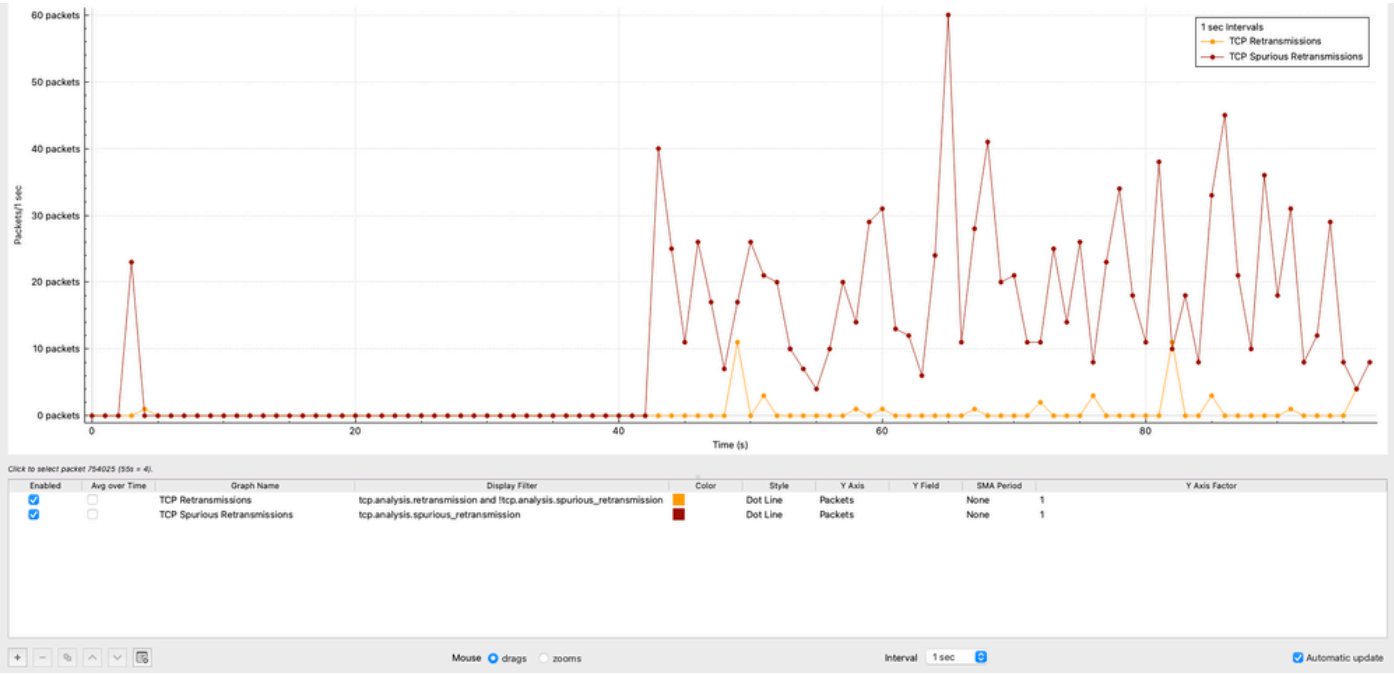
```
tcp.analysis.spurious_retransmission and ip.src==10.93.19.8
```

- Stijl: Impuls of Bar
- Y-as: pakketten
- Interval: 1 sec

technische interpretatie

- Het extreem lage percentage echte heruitzendingen bevestigt dat pakketverlies niet aanwezig is in het netwerk.
- De aanwezigheid van valse heruitzendingen duidt op voortijdige beslissingen over heruitzending door de bronhost.
- Dit gedrag kan de efficiëntie enigszins beïnvloeden, maar is geen primaire oorzaak van ernstige achteruitgang van de doorvoer.

Deze analyse versterkt verder dat het probleem niet gerelateerd is aan netwerkbetrouwbaarheid, maar eerder aan TCP-gedrag, latentie of eindpuntprestaties.

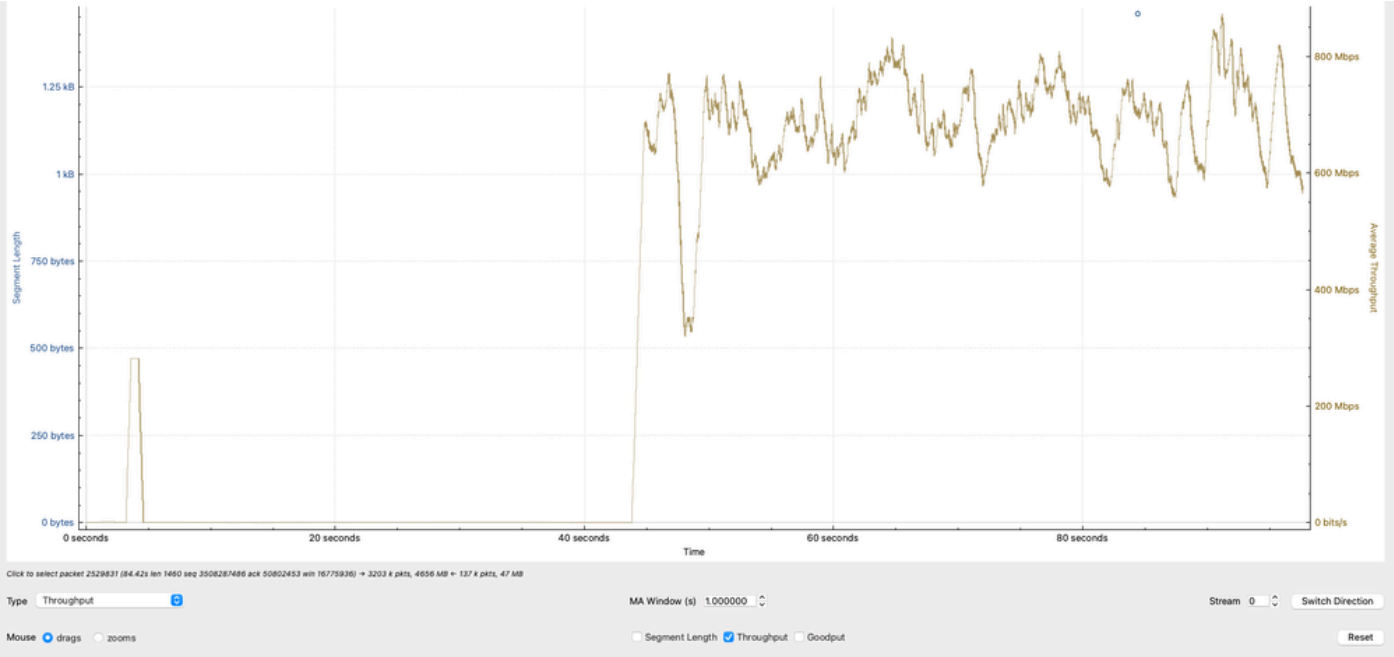


Effectieve doorvoeranalyse

De grafiek toont de effectieve doorvoer, berekend op basis van TCP-payload (daadwerkelijk overgedragen gegevens) in megabits per seconde. De waargenomen doorvoer schommelt voornamelijk tussen 600 Mbps en 800 Mbps, wat aangeeft dat het netwerk weliswaar actief gegevens overdraagt, maar geen hoger bandbreedtepotentieel bereikt.

Configuratie van Wireshark (effectieve doorvoer)

`Statistics → TCP Streams Graphs → Throughout`



technische interpretatie

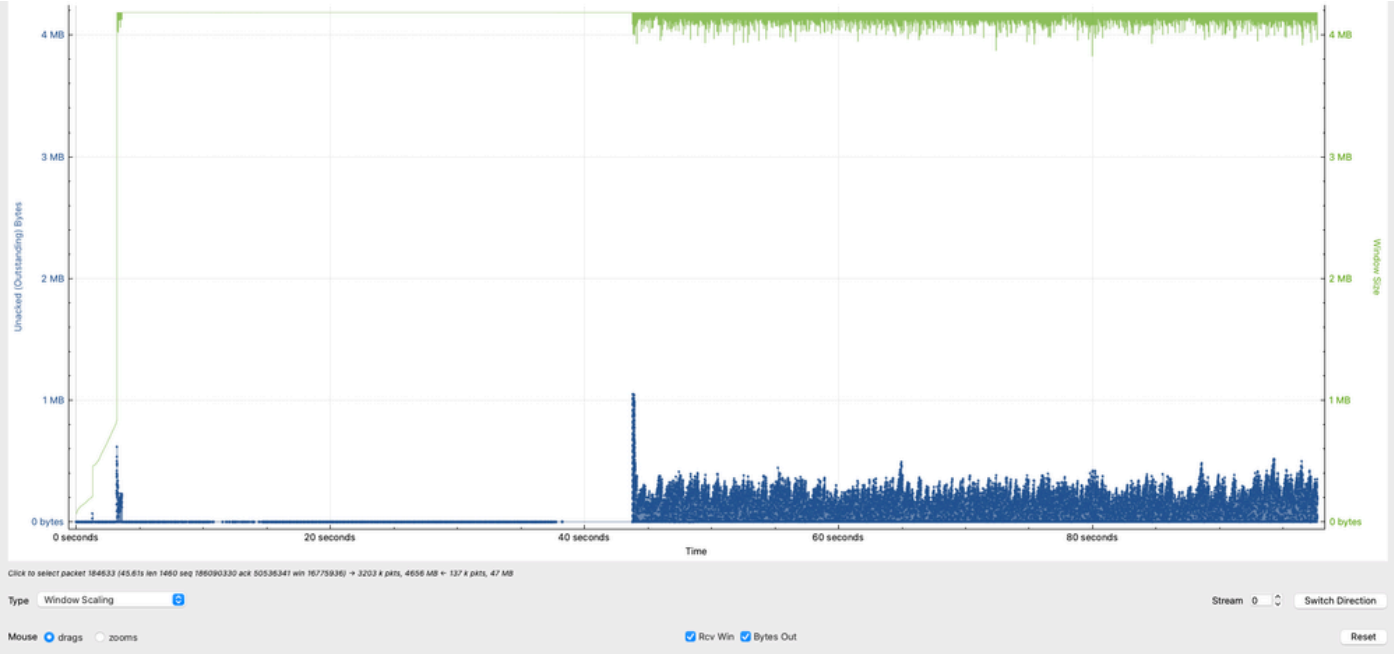
- Het doorvoerbereik van 600-800 Mbps komt overeen met eerdere berekeningen op basis van de grootte van het TCP-venster en RTT.
- Variabiliteit in doorvoer weerspiegelt:
  - RTT-fluctuaties
  - Aanpassingen voor TCP-congestiecontrole
  - Toepassingspacing of buffering
- Aangezien de doorvoer niet de lijnsnelheid benadert (bijvoorbeeld 10G), is de beperking geen fysieke bandbreedte, maar eerder TCP-efficiëntiebeperkingen.
- Deze analyse bevestigt dat de waargenomen doorvoer consistent is met TCP-beperkingen (venstergrootte en latentie), wat versterkt dat de bottleneck niet te wijten is aan pakketverlies of interfacecapaciteit, maar aan het gedrag van de transportlaag en de

eindpuntvoorwaarden.

Data in Flight (TCP Window) Analyse

De grafiek benadrukt een kritisch gedrag in de TCP-sessie door de capaciteit van de ontvanger te vergelijken met de werkelijke gegevens in transit (bytes tijdens de vlucht).

- De groene lijn geeft de hoeveelheid TCP-gegevens weer die 10.91.2.35 (ontvanger) kan accepteren (effectief ontvangstvenster).
- De blauwe lijn geeft de hoeveelheid TCP-gegevens weer die momenteel in vlucht zijn vanaf 10.93.19.8 (afzender).



De waargenomen gegevens in vlucht pieken bij ongeveer 1 MB, met extra pieken rond 8 KB en 5 KB, maar het is voornamelijk geconcentreerd tussen 1 KB en 250 KB.

Dit geeft aan dat hoewel de ontvanger in staat is om grotere hoeveelheden gegevens te verwerken, de afzender niet consequent het beschikbare venster gebruikt.

Configuratie Wireshark (gegevens in vlucht versus venster)

`Statistics → TCP Streams Graphs → Throughout`

technische interpretatie

- De ontvanger (10.91.2.35) adverteert een aanzienlijk groter venster, wat aangeeft dat het in staat is om meer gegevens te ontvangen.
- De afzender (10.93.19.8) maakt onvoldoende gebruik van het beschikbare venster, zoals blijkt uit de lagere en inconsistente gegevens in vluchtwaarden.
    - De afzender kan idealiter gegevens in vluchtwaarden dichter bij het geadverteerde venster van de ontvanger (~1 MB) houden om de doorvoer te maximaliseren.
    - Het onvermogen om hoge gegevensniveaus tijdens de vlucht te handhaven, beperkt de doorvoer rechtstreeks en is een sterke indicator van TCP-inefficiëntie aan de bron, geen probleem met de netwerkcapaciteit.

TCP-payload versus MSS-analyse in de tijd

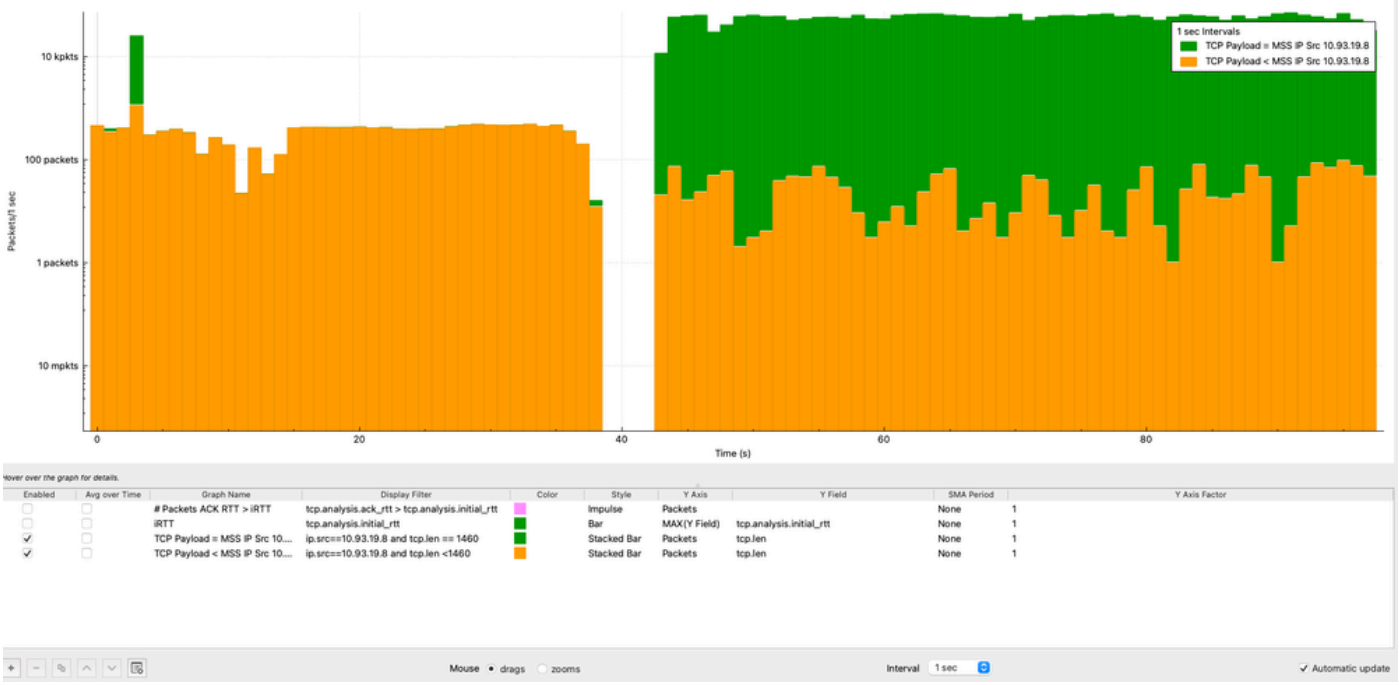
Door de omvang van de TCP-payload in de loop van de tijd te analyseren ten opzichte van MSS, kan worden bepaald of de afzender elk TCP-segment efficiënt gebruikt. Deze analyse wordt uitgevoerd vanuit het perspectief van het IP-adres van de bron (10.93.19.8).

In Wireshark zijn de grafieken als volgt geconfigureerd:

- Grafiek 1 (pakketten ter grootte van een MSS):
    - Weergavefilter: ip.src==10.93.19.8 en tcp.len == 1460
    - Stijl: gestapelde balk
    - Y-as: pakketten
    - Interval: 1 seconde
- Grafiek 2 (alle pakketten ≤ MSS):
    - Weergavefilter: ip.src==10.93.19.8 en tcp.len <= 1460
    - Stijl: gestapelde balk
    - Y-as: pakketten
    - Interval: 1 seconde
- Logaritmische schaal toepassen voor betere visualisatie

Uit de analyse:

- De meeste pakketten (>10.000 pakketten per seconde) bereiken consequent de MSS-waarde van 1460 bytes.
- Een kleiner deel van de pakketten draagt minder lading door normaal TCP-gedrag (ACK's, segmentatie of end-of-stream gegevens).



Root Cause Analysis (RCA): verslechtering van de TCP-prestaties

Deze analyse toont aan dat het identificeren van de hoofdoorzaak van TCP-prestatieproblemen een holistische, end-to-end benadering vereist, in plaats van aan te nemen dat het netwerk de primaire bron van degradatie is.

Uitgebreide validatie werd uitgevoerd op de Cisco Nexus 9300-switch, inclusief interfacetellers, QoS-beleid, routering en ARP-stabiliteit, CPU-puntverificatie, SPAN-gebaseerde pakketregistratie en ASIC-niveau forwarding-validatie met behulp van ELAM. Alle resultaten bevestigden consequent dat de switch binnen de verwachte parameters werkte:

- Geen pakketdruppels
- Geen abnormale latentie (microseconde bereik)
- Geen QoS- of regelvlak-impact
- Juiste hardware forwarding

Bovendien onthulde TCP-analyse:

- Verwaarloosbare heruitzendingen (0,00125 %)
- Geen bewijs van pakketverlies
- Consistent MSS-gebruik aan de bron
- Doorvoer afgestemd op TCP-venster- en RTT-beperkingen
- Onderbenutting van het beschikbare TCP-venster (Data in Flight analysis)
- Het netwerk is niet het knelpunt
- De bronserver beperkt de prestaties

Conclusie

De verslechtering van de prestaties wordt veroorzaakt door de bronserver die werkt met MTU 1500 in een omgeving die geschikt is voor jumbo, waardoor efficiënt gebruik van de beschikbare netwerkcapaciteit wordt voorkomen.

Oplossing

Verhoog de MTU op de bronserver van 1500 naar 9000 bytes om af te stemmen op de bestemming en de netwerkinfrastructuur. De voordelen:

- Grotere TCP-segmenten inschakelen
- Pakketoverhead verminderen
- Verbeter de algehele doorvoer

technische reflectie

Een belangrijke conclusie uit deze analyse is het belang van het vermijden van voorbarige conclusies bij het oplossen van problemen met netwerkprestaties. Hoewel het gebruikelijk is om problemen in eerste instantie toe te schrijven aan het netwerk, toont dit geval duidelijk aan dat het netwerk correct functioneerde gedurende het hele pad van het gegevensvlak. Alleen door diepgaande TCP-analyse uit te voeren vanuit zowel het bron- als het bestemmingsperspectief - inclusief handshake-parameters, RTT-gedrag, venstergebruik, hertransmissies en payload-efficiëntie - was het mogelijk om de echte bottleneck nauwkeurig te identificeren.

Door de tijd te nemen om TCP-gedrag in detail te analyseren, voorkomt u verkeerde diagnoses, vermindert u onnodige netwerkwijzigingen en zorgt u ervoor dat de herstelinspanningen gericht zijn op de werkelijke oorzaak.

Over deze vertaling

Cisco heeft dit document vertaald via een combinatie van machine- en menselijke technologie om onze gebruikers wereldwijd ondersteuningscontent te bieden in hun eigen taal. Houd er rekening mee dat zelfs de beste machinevertaling niet net zo nauwkeurig is als die van een professionele vertaler. Cisco Systems, Inc. is niet aansprakelijk voor de nauwkeurigheid van deze vertalingen en raadt aan altijd het oorspronkelijke Engelstalige document (link) te raadplegen.