Fehlerbehebung bei TCP-Leistung auf Nexus 9000 (NX-OS)

Inhalt

Einleitung

Dieses Dokument beschreibt die TCP-Grundlagen, Wireshark Deep Packet Analysis und die praktische Fehlerbehebung zur Optimierung der End-to-End-Leistung.

Voraussetzungen

Anforderungen

Cisco empfiehlt, dass Sie über Kenntnisse in folgenden Bereichen verfügen:

- IP/TCP

Verwendete Komponenten

Die Informationen in diesem Dokument basierend auf folgenden Software- und Hardware-Versionen:

- Cisco Nexus 9000: Cloud-Skalierbarkeit mit Cisco NX-OS 10.6(X)

Anmerkung: Fragen zur Konfiguration und Interoperabilität von Software oder Hardware von Drittanbietern liegen außerhalb des Cisco Supports. Die Verwendung von Tools von Drittanbietern ist eine gute Möglichkeit, Ihre Konfiguration und Ihren Betrieb mit Cisco Geräten zu demonstrieren.

Die Informationen in diesem Dokument beziehen sich auf Geräte in einer speziell eingerichteten Testumgebung. Alle Geräte, die in diesem Dokument benutzt wurden, begannen mit einer gelöschten (Nichterfüllungs) Konfiguration. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die möglichen Auswirkungen aller Befehle kennen.

Hintergrundinformationen

Was ist TCP

Das Transmission Control Protocol (TCP) ist ein grundlegendes Transportschichtprotokoll, das auf Schicht 4 des OSI-Modells arbeitet und eine zuverlässige, geordnete und fehlerüberprüfte Bereitstellung eines Bytestroms zwischen Anwendungen ermöglicht, die über ein IP-Netzwerk kommunizieren.

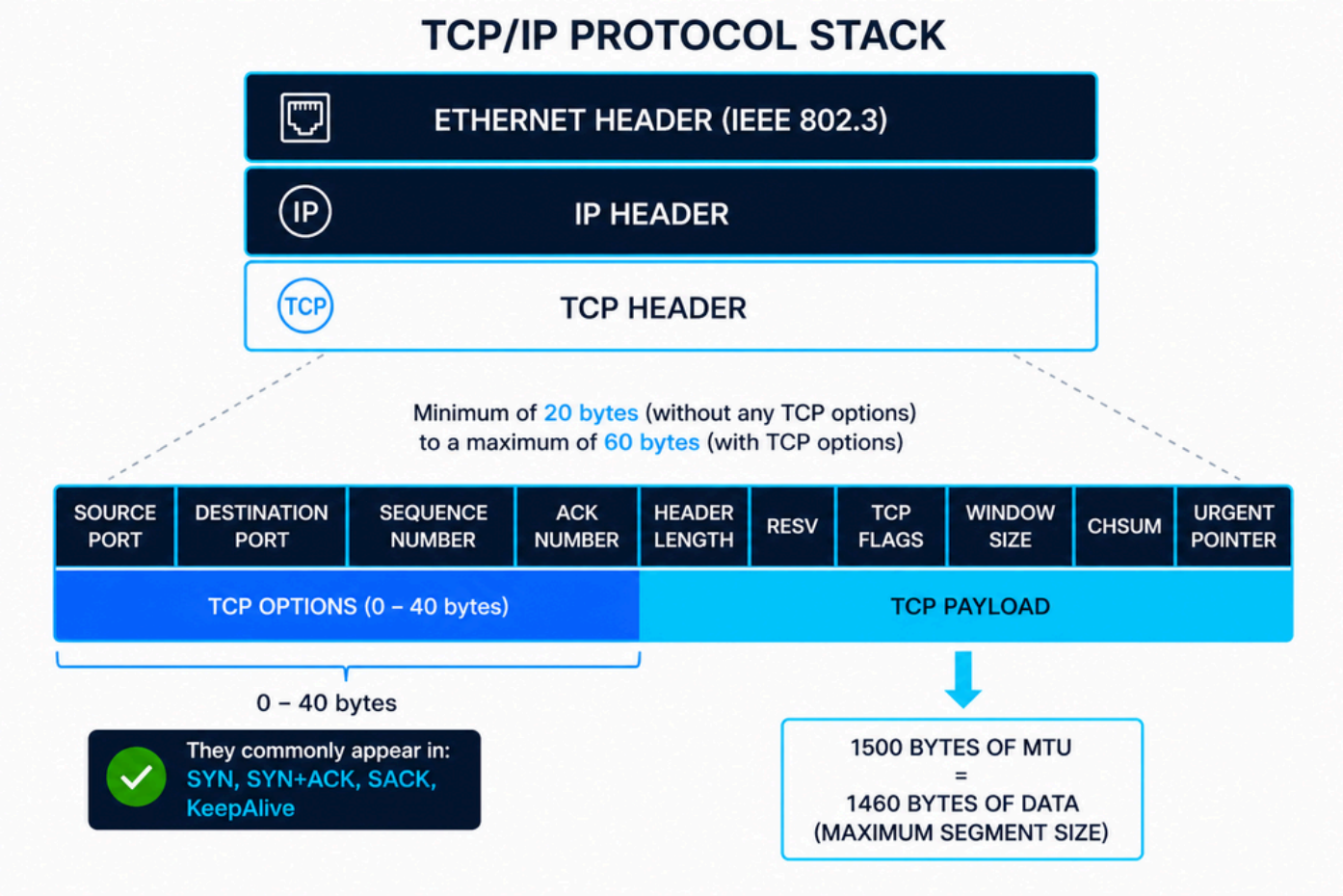
Drei wichtige Vorteile

1. Zuverlässigkeit: TCP ist verbindungsorientiert und garantiert die Übermittlung, da es Bestätigungen vom Empfänger erfordert. Wenn ein Paket während der Übertragung verloren geht oder beschädigt wird, überträgt TCP die Daten automatisch erneut, um sicherzustellen, dass es das Ziel erreicht.
2. Bestellte Lieferung: Da Netzwerkpakete nicht in der richtigen Reihenfolge ankommen können, weist TCP jedem Segment Sequenznummern zu. Auf diese Weise kann das empfangende System die Daten in der exakten Reihenfolge wiederherstellen, in der sie ursprünglich gesendet wurden.
3. Fluss- und Überlastungskontrolle: TCP verwaltet die Datenübertragungsrate dynamisch, um sie an die Verarbeitungskapazität des Empfängers und die aktuellen Netzwerkbedingungen anzupassen. So werden Datenverluste durch Pufferüberläufe oder Netzwerküberlastungen vermieden.

TCP/IP-Kapselung - Übersicht

Das Diagramm stellt den TCP/IP-Stack dar, in dem ein TCP-Segment (Layer 4) in ein IP-Paket

(Layer 3) und dann in einen Ethernet-Frame (Layer 2) gemäß IEEE 802.3 eingekapselt wird. Dieser Layer-Ansatz stellt eine modulare Kommunikation sicher, bei der jede Layer eigene Steuerinformationen (Header) hinzufügt, um die Bereitstellung, das Routing und die Datenintegrität zu gewährleisten.



Ethernet-Header (IEEE 802.3)

Der Ethernet-Header umfasst in der Regel 14 Byte und setzt sich aus folgenden Komponenten zusammen:

- Ziel-MAC-Adresse (6 Byte)
- Quell-MAC-Adresse (6 Byte)
- EtherType/Length (2 Byte)

Ethernet-Frames enthalten außerdem einen 4 Byte langen Frame Check Sequence (FCS) Trailer zur Fehlererkennung auf Layer 2. IEEE 802.3 definiert Framing, minimale/maximale Frame-Größen und physische Bereitstellungseinschränkungen, die sich direkt auf Protokolle höherer Layer wie TCP auswirken.

IP-Header (IPv4)

Der IPv4-Header hat eine Mindestgröße von 20 Byte und kann mit Optionen auf bis zu 60 Byte erweitert werden. Zu den Schlüsselfeldern gehören:

- Quell- und Ziel-IP-Adressen
- Time To Live (TTL)
- Protokoll (identifiziert TCP als Nutzlast)

Die IP-Schicht ist für die logische Adressierung und das Routing über Netzwerke hinweg zuständig, gewährleistet jedoch keine Zuverlässigkeit.

TCP-Header-Struktur

Der TCP-Header reicht je nach Optionen von 20 bis 60 Byte. Zu den Schlüsselfeldern gehören:

- Quell-/Ziel-Ports
- Sequenznummer
- Bestätigungsnummer
- Flags (SYN, ACK, FIN, RST usw.)
- Fenstergröße
- Prüfsumme

TCP fügt der IP-Kommunikation zuverlässige Bereitstellung, korrekte Sequenzierung und Flusskontrolle hinzu.

TCP-Optionen (allgemein 10)

TCP-Optionen erweitern das Basisprotokoll. Die häufigsten sind:

1. Maximum Segment Size (MSS) (Maximale Segmentgröße (MSS)) - Definiert die größte TCP-Nutzlast, die ein Host akzeptieren kann.
2. Fensterskala - Erweitert das Empfangsfenster über 65.535 Byte hinaus.
3. Selektive Bestätigung zulässig (SACK zulässig) - Aktiviert die Funktion zur selektiven Bestätigung.
4. Selective Acknowledgment (SACK) - Gibt empfangene Datenblöcke an, um vollständige Neuübertragungen zu vermeiden.
5. Timestamps: Diese werden für die RTT-Berechnung und den Schutz vor umschlossenen Sequenznummern (PAWS) verwendet.

6. No-Operation (NOP) - Padding für die Ausrichtung der Optionen.
7. End of Option List (EOL) (Ende der Optionsliste) - Markiert das Ende der TCP-Optionen.
8. TCP Fast Open (TFO) - Ermöglicht den Datenaustausch während des ersten Handshakes.
9. Multipath TCP (MPTCP) - Aktiviert mehrere Netzwerkpfade für eine einzelne TCP-Sitzung.
10. User Timeout Option (UTO) (Benutzerzeitüberschreitungsoption) - Steuert, wie lange übertragene Daten unbestätigt bleiben können.

TCP-Sequenz und Bestätigungsverhalten (einschließlich SYN/FIN)

Sowohl SYN- als auch FIN-Flags verwenden jeweils eine Sequenznummer, auch wenn keine Payload vorhanden ist. TCP arbeitet mit einem byteorientierten Sequenzierungsmodell, bei dem jedes übertragene Byte - und bestimmte Kontrollflags - den Sequenzraum vorverlegt. Dieses Verhalten ist für eine genaue TCP-Analyse bei der Paketerfassung und für die Diagnose von Sequenzierungs- oder Quittierungsinkonsistenzen unerlässlich.

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

Dabei gilt:

- SEQ = Initial Sequence Number
- Nutzlastlänge = Datengröße in Byte
- SYN? 1: 0 = Fügt 1 hinzu, wenn das SYN-Flag gesetzt ist, andernfalls 0
- FIN? 1: 0 = Fügt 1 hinzu, wenn das FIN-Flag gesetzt ist, andernfalls 0
- ACK = Nächstes erwartetes Byte

Beispiel 1: SYN mit Daten (TCP Fast Open)

- SEQ = 1000
- SYN = 1
- Nutzlastlänge = 200 Byte

ACK-Berechnung:

- ACK = 1.000 + 200 + 1 + 0 = 1.201

Dies zeigt ein Szenario an, in dem Daten während des TCP-Handshakes gesendet werden. Sowohl die Nutzlast als auch das SYN-Flag belegen Sequenzräume.

Beispiel 2: FIN mit Daten (Verbindungsabschluss)

- SEQ = 3000
- FIN = 1
- Nutzlastlänge = 150 Byte

ACK-Berechnung:

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

Dies zeigt, dass TCP Daten während des Verbindungsabbruchs enthalten kann, und dass sowohl das Nutzdaten- als auch das FIN-Flag die Sequenznummer inkrementieren.

MSS und seine Beziehung zur MTU

Die maximale Segmentgröße (Maximum Segment Size, MSS) definiert die maximale Nutzlast, die TCP in einem Segment senden kann.

- Typische Ethernet-MTU = 1.500 Byte
- MSS = MTU − IP-Header − TCP-Header
- Standard-MSS = 1.460 Byte (1.500 − 20 − 20)

Wenn TCP-Optionen vorhanden sind, wird die MSS entsprechend reduziert. MSS wird während des TCP-Drei-Wege-Handshakes ausgehandelt und verhindert die Fragmentierung auf der IP-Ebene.

Funktionsweise der MSS-Aushandlung beim Drei-Wege-TCP-Handshake

Die maximale Segmentgröße (Maximum Segment Size, MSS) wird während des TCP-Drei-Wege-Handshakes mithilfe der MSS-Option in SYN-Paketen ausgetauscht:

- Host A → Host B (SYN): meldet seine MSS (z. B. 1460)
- Host B → Host A (SYN-ACK): meldet seine MSS (zum Beispiel 1380)

Jede Seite sagt:

Dies ist die größte akzeptierte TCP-Nutzlast.

Hauptregel: MSS ist gerichtet

Die MSS wird nicht als einheitlicher vereinbarter Wert ausgehandelt.

Stattdessen:

- Jeder Host verwendet die von der anderen Seite angekündigte MSS.
- Dadurch entstehen zwei unabhängige Grenzen, eine pro Richtung.

Daher:

- A sendet Daten mithilfe der MSS von B.
- B sendet Daten mithilfe der MSS von A.

Kann die Quelle mehr TCP-Payload senden als die Ziel-MSS?

In einem richtig funktionierenden TCP-Stapel: Nein.

- Der Sender muss die vom Empfänger angekündigte MSS respektieren.
- Das Senden größerer Segmente birgt folgende Risiken:
    - IP-Fragmentierung (bei Überschreitung der MTU)
    - Paketverluste (wenn die Fragmentierung blockiert oder nicht unterstützt wird)
- Dies führt zu:
    - Erneute Übertragungen
    - Leistungsabfall
    - Probleme wie die PMTUD (Path MTU Discovery) mit schwarzen Löchern

Praktische Hinweise zur Fehlerbehebung

- Überprüfen Sie die MSS-Werte immer im TCP-Drei-Wege-Handshake (SYN/SYN-ACK-Pakete).
- Überprüfen Sie auf Diskrepanzen, die verursacht werden durch:
    - Tunnel (VXLAN, GRE, IPsec)
    - Firewalls ändern MSS (MSS-Klemmung)
- Auf Plattformen wie Cisco NX-OS wird die MSS-Anpassung häufig verwendet, um eine Fragmentierung über gekapselte Pfade hinweg zu verhindern.

Fenstergröße (Flusssteuerung)

Die Fenstergröße legt fest, wie viele Daten der Empfänger ohne Bestätigung akzeptieren kann.

Worum handelt es sich?

- Ein Flusssteuermechanismus, um einen Pufferüberlauf zu verhindern.

Zweck:

- Stellt sicher, dass der Absender den Empfänger nicht überwältigt.

Wo kann sie bezogen werden:

- Sichtbar bei der Paketerfassung (z. B. Wireshark).
- Abgeleitet von der TCP-Stapelkonfiguration und der Puffergröße des Betriebssystems.

Variabilität von Anbieter/Betriebssystem:

- Verschiedene Implementierungen (Linux, Windows, Cisco NX-OS) verwenden dynamische Skalierung und Puffer-Tuning, was zu unterschiedlichen Fenstergrößen führt.

Bedingung für Nullfenster:

- Wenn Fenstergröße = 0 ist, ist der Empfängerpuffer voll.
- Absender unterbricht die Übertragung und sendet regelmäßige Tests.

Variable Windows-Mechanismen

- ratenbasierte Flusssteuerung
  - Er weist dem Absender eine feste Datenrate zu und stellt sicher, dass die Daten diese Zuordnung nie überschreiten.
  - Ideal für Streaming-Anwendungen.
  - Broadcast- und Multicast-Bereitstellung
- Fensterbasierte Flusssteuerung
  - Die Fenstergröße variiert mit der Zeit.
  - Der Empfänger erreicht die Flusskontrolle, indem er das zulässige Fenster an die Aktualisierung des Absenderfensters signalisiert.

Problembehandlung:

- Kleine oder gar keine Fenster → Receiver-seitiger Engpass (CPU, Arbeitsspeicher, Anwendung).
- Große Fenster, aber geringer Durchsatz → Netzwerkprobleme (Latenz, Überlastung).
- Die Analyse des Fensterverhaltens ist für die Diagnose von Leistungsproblemen in TCP-Sitzungen von entscheidender Bedeutung.

Fehlerbehebung für TCP-Datenebene auf Cisco Nexus 9000 (NX-OS)

In diesem Abschnitt wird eine praktische Methode beschrieben, mit der festgestellt werden kann, ob sich ein Cisco Nexus-Switch mit NX-OS auf die Weiterleitung des TCP-Datenverkehrs auswirkt oder ob Leistungsprobleme auftreten. Der Ansatz wird anhand eines hypothetischen Szenarios vorgestellt.

Wenn eine TCP-Latenz oder ein Leistungsabfall festgestellt werden, ist es üblich, zunächst zu vermuten, dass das Netzwerk diese Latenz verursacht. Diese Annahme muss jedoch durch datengesteuerte Analysen validiert werden. Die maßgebliche Methode für die TCP-Fehlerbehebung ist die Paketerfassung, die idealerweise durchgeführt wird:

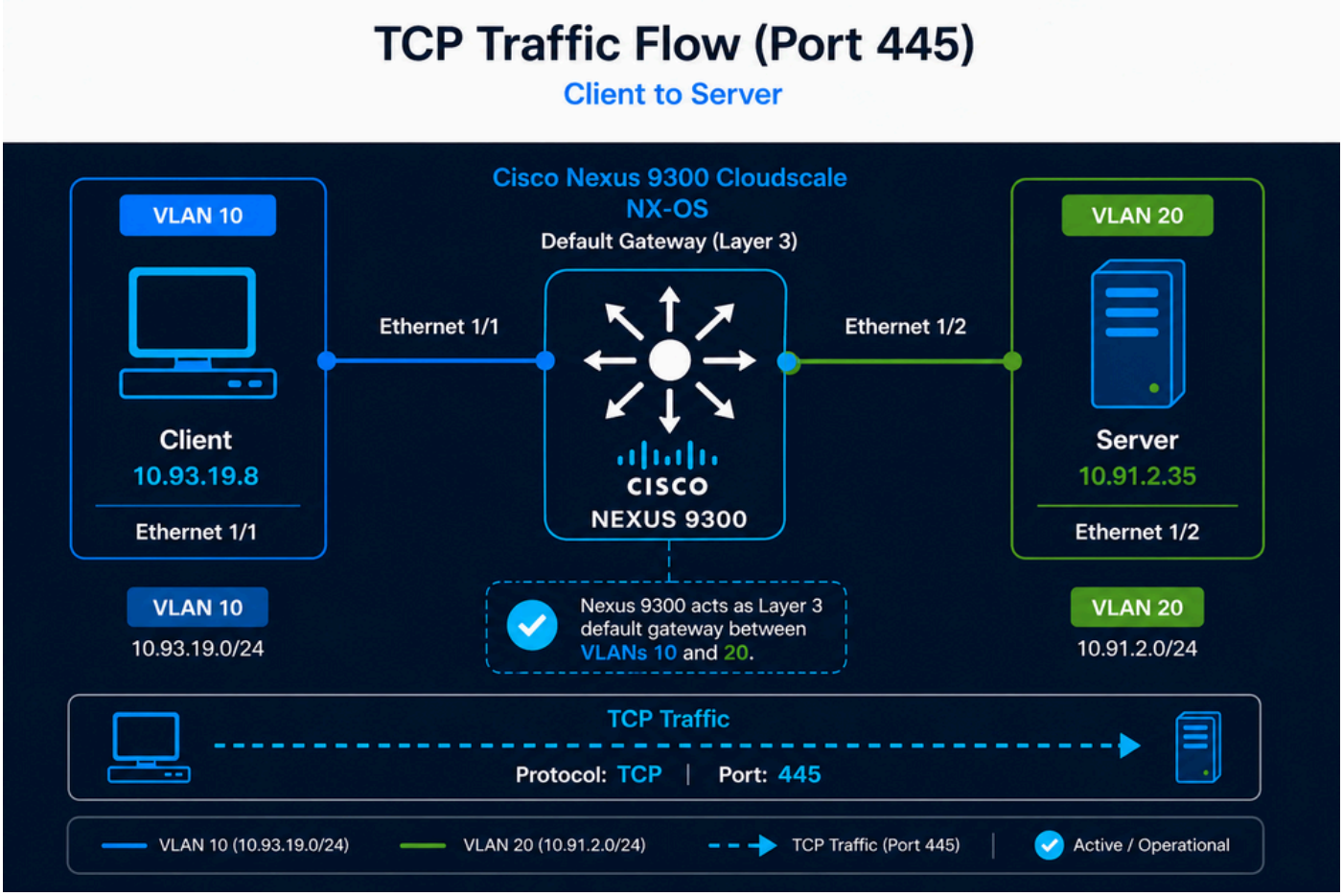
- Gleichzeitig an Quelle und Ziel
- Vor der Datenverkehrsinitiierung

Dadurch wird die Transparenz des TCP-Drei-Wege-Handshakes sichergestellt, bei dem wichtige Parameter wie MSS, Fensterskala und SACK ausgehandelt und später in der Sitzung nicht wiederholt werden. Wenn keine gleichzeitigen Erfassungen möglich sind, kann die Analyse mit einer einzigen Erfassung fortgesetzt werden, die Schlussfolgerungen sind jedoch begrenzt.

Szenariodefinition

Ein Benutzer hat festgestellt, dass der Backup-Prozess für einen Anwendungsdatensatz von ca. 7,5 TB, der zuvor in ca. 9 Stunden abgeschlossen wurde, jetzt fast 21 Stunden dauert. Obwohl TCP-Sitzungen zwischen dem Client und dem Server weiterhin erfolgreich hergestellt werden, deutet die deutliche Verlängerung der Backup-Dauer auf eine mögliche Verschlechterung des Durchsatzes oder der TCP-Gesamtleistung hin. Da der Nexus-Switch das einzige Netzwerkgerät im Pfad ist und zudem Layer-3-Gateway-Funktionen bereitstellt, vermutet der Netzwerkadministrator, dass der Nexus-Switch die Ursache des Problems ist.

- Kunde: 10.93.19.8 (VLAN 10)
- Server: 10.91.2.35 (VLAN 20)
- Nexus 9300 als Standard-Gateway
- TCP-Port 445



Anfängliche Validierung (Erreichbarkeit)

- Diese Befehle werden verwendet, um die Path-MTU (PMTU) zwischen Quelle und Ziel zu validieren, indem ICMP-Pakete mit dem Don’t Fragment (DF)-Bit gesendet werden. Dies erleichtert die Bestimmung der maximalen Paketgröße, die das Netzwerk ohne Fragmentierung durchlaufen kann. Dieser Prozess muss sowohl für die Quelle als auch für das Ziel ausgeführt werden.
- Überprüfen Sie stets die MTU der physischen Schnittstelle an der Quelle und am Ziel.
- In diesem Szenario steht der Zugriff nur für den Quell-Host zur Verfügung, auf dem eine MTU von 1500 identifiziert wurde.

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

- -c 10 → Sendet 10 ICMP-Echoanfragen
- -I 192.168.10.10 → Verwendet diese spezifische Quell-IP/Schnittstelle.
- -s 1472 → Setzt die ICMP-Nutzlastgröße auf 1472 Byte
- -M do → Legt das DF-Bit (Nicht fragmentieren) fest
- 192.168.20.20 → Ziel-IP

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

- -n 10 → Sendet 10 ICMP-Echoanfragen
- -l 1472 → Setzt die ICMP-Nutzlastgröße auf 1472 Byte
- -f → Legt die Markierung "Nicht fragmentieren" (DF) fest
- 192.168.20.20 → Ziel-IP

Warum 1472 Byte?

- ICMP-Nutzlast = 1.472 Byte
- IP-Header = 20 Byte
- ICMP-Header = 8 Bytes
- Gesamtpaketgröße: 1.472 + 20 + 8 = 1.500 Byte (Standard-MTU)
- Dabei wird getestet, ob der Pfad eine MTU mit 1500 Byte ohne Fragmentierung unterstützt. Wenn Sie versuchen, 1.500 Byte ICMP-Nutzlast zu senden, kann der Ping fehlschlagen, da die Gesamtpaketgröße nach dem Hinzufügen der IP- und ICMP-Header die Standard-MTU überschreiten würde.

Schlussfolgerungen

- Wenn der Ping-Befehl erfolgreich ausgeführt wird (ohne Paketverlust), unterstützt der Pfad eine MTU von mindestens 1500 Byte, und es ist keine Fragmentierung erforderlich.
    - ICMP-Ergebnisse löschen → mit TCP-Analyse fortfahren
    - Intermittierender Ping-Erfolg → möglicher Paketverlust, vorübergehende Überlastung, Ratenbegrenzung oder ein Weiterleitungsproblem; mit der Paketverlustanalyse fortfahren, da TCP einen verlustfreien Pfad benötigt, um effizient zu arbeiten.
- Wenn der Ping fehlschlägt und die Fehlermeldung "Fragmentation needed" angezeigt wird oder das Zeitlimit überschritten wird, gibt es eine Verbindung im Pfad mit einer MTU von weniger als 1500 Byte. Das Paket kann aufgrund des DF-Bits nicht weitergeleitet werden, und dies weist auf ein MTU-Problem im Pfad hin.

Verwendung dieses Handbuchs zur Fehlerbehebung

- Reduzieren Sie schrittweise die Nutzlastgröße (z. B. 1472 → 1400 → 1300), um die größte erfolgreiche Nutzlastgröße zu ermitteln.
- Berechnet die MTU nach ihrer Identifizierung mit der Formel MTU = Payload + 28 Byte (IP + ICMP-Header).

Praktische Relevanz für TCP

- Wenn die MTU kleiner ist als erwartet, können TCP-Segmente fragmentiert oder verworfen werden.
- Dies führt zu Neuübertragungen, erhöhter Latenz und verringertem Durchsatz und wirkt sich direkt auf die Anwendungsleistung aus.

Identifizieren des Datenverkehrspfads (Schnittstellen)

Um die TCP-Leistung auf einem Cisco Nexus 9000-Switch effektiv zu überprüfen, muss ermittelt werden, welche Schnittstellen den Datenverkehr zwischen Quelle und Ziel empfangen und weiterleiten.

In einfachen Topologien kann dies direkt aus den physischen Verbindungen abgeleitet werden. Wenn der Client beispielsweise mit Ethernet1/1 und der Server mit Ethernet1/2 verbunden ist, ist der Datenverkehrspfad unkompliziert. In realen Umgebungen mit mehreren aktiven Schnittstellen, Port-Channels oder vPC-Konfigurationen ist diese Identifizierung jedoch nicht immer trivial.

In diesen Fällen wird empfohlen, das Embedded Logic Analyzer Module (ELAM) zu verwenden, das Transparenz auf ASIC-Ebene (Data-Plane Hardware) bietet.

Mit ELAM können Sie ein Paket erfassen, während es von der Weiterleitungspipeline verarbeitet wird, und wichtige Informationen wie die folgenden offenlegen:

- Eingangsschnittstelle
- Ausgangs-Schnittstelle
- Weiterleitungsentscheidung (L2/L3-Suchergebnis)

Diese Methode ist wesentlich genauer als die Verwendung von Tools auf Kontrollebene, da sie den tatsächlichen Weiterleitungspfad der Hardware widerspiegelt.

Es ist wichtig zu beachten, dass ELAM jeweils nur ein Paket erfasst, sodass die Filterkriterien genau definiert werden müssen, um dem gewünschten Datenverkehr (z. B. Quell-IP, Ziel-IP, TCP-Port) gerecht zu werden. Wenn die Filter zu breit gefasst sind, besteht das Risiko, dass nicht zusammenhängender Datenverkehr wie ICMP oder UDP anstatt des beabsichtigten TCP-Datenflusses erfasst wird.

Außerdem muss dieser Vorgang für beide Verkehrsrichtungen wiederholt werden:

- Quelle → Ziel
- Ziel → Quelle

In Umgebungen, die vPC oder ECMP verwenden, kann die Datenverkehrslast auf mehrere Pfade verteilt werden. So kann der vor- und zurückfließende Datenverkehr über verschiedene Switches oder Schnittstellen geleitet werden. In diesen Szenarien muss ELAM auf jedem relevanten Nexus-Switch ausgeführt werden, um vollständige Transparenz zu gewährleisten.

Durch die genaue Identifizierung von Eingangs- und Ausgangsschnittstellen wird der Umfang der Fehlerbehebung erheblich reduziert, sodass eine gezielte Validierung der Schnittstellenzähler, QoS-Richtlinien, MTU-Einstellungen und potenziellen Überlastungspunkten entlang des genauen Weiterleitungspfads möglich ist.

ELAM-Konfiguration (Nexus 9300 Cloud-Skalierung)

In diesem Beispiel wird der Datenverkehr mit der Quell-IP 10.93.19.8, der Ziel-IP 10.91.2.35 und dem TCP-Zielport 445 gefiltert.

ELAM-Einrichtung

<#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
```

Nachdem Sie den Datenverkehr generiert haben, rufen Sie das Ergebnis ab:

<#root>

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

report
```

Umgekehrte Datenverkehrserfassung (erforderlich für volle Transparenz)

Um den Rückgabepfad zu validieren, wiederholen Sie die Konfiguration, indem Sie die Quell- und Ziel-IP-Adresse austauschen:

<#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
```

Betriebshinweise

- ELAM erfasst nur ein Paket, um sicherzustellen, dass der Datenverkehr aktiv fließt, wenn die Erfassung gestartet wird.
- Filter müssen präzise sein, um die Erfassung von nicht in Beziehung stehendem Datenverkehr zu vermeiden.
- In vPC-Umgebungen führen Sie ELAM auf beiden Switches aus, da der Datenverkehr in jeder Richtung unterschiedlich gehasst werden kann.
- Die Ausgabe zeigt die Eingangsschnittstelle, die Ausgangsschnittstelle und die Weiterleitungsentscheidung in der Hardware an und bietet eine autoritative Transparenz der Datenebene.

Referenz

Cisco Nexus 9000 Cloud Scale ASIC ELAM-Leitfaden

Validierung auf Schnittstellenebene

Die Validierung auf Schnittstellenebene stellt sicher, dass der Nexus-Switch keine Einschränkungen oder Anomalien mit Auswirkungen auf den TCP-Datenverkehr einführt. Im Mittelpunkt steht die Bestätigung, dass Konfiguration, Betriebsstatus und Hardware-Zähler mit dem erwarteten Verhalten für eine leistungsstarke Weiterleitung auf Datenebene übereinstimmen.

Validierung der Konfiguration

- Vergewissern Sie sich, dass keine restriktiven ACLs auf die Schnittstellen angewendet werden:

<#root>

```
switch#

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

- Überprüfen Sie, ob sich unbeabsichtigte QoS-Richtlinien auf den Datenverkehr auswirken (Schnittstellenebene und globale QoS, einschließlich Warteschlangen, Richtlinien und 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
```

- Bestätigung der Layer 2- oder Layer 3-Konfiguration (Switch-Port vs. geroutete Schnittstelle), einschließlich VLAN-Mitgliedschaft, STP-Status und IP-Adressierung:

<#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
```

Überprüfung des Betriebsstatus

- Überprüfen Sie die MTU-Konsistenz, und stellen Sie sicher, dass sie der erwarteten Konfiguration entspricht (z. B. 1500 oder 9000 Byte):

```
<#root>

switch#

 show interface ethernet1/1-2 | include MTU
```

- Schnittstellengeschwindigkeit und Duplexeinstellungen bestätigen:

```
<#root>

switch#

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

- Überprüfen Sie die Schnittstellenstabilität (kein Flapping und häufige Verbindungsübergänge):

<#root>

switch#

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

Überprüfung des Fehlerzählers

- Zähler vor dem Testen löschen:

<#root>

switch#

clear counters interface all

- Fehlerzähler überwachen (nur Werte ungleich null):

<#root>

switch#

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

Validierung nach dem Test

- Führen Sie den TCP-Datenverkehrstest erneut aus, und beobachten Sie die Zähler erneut:

<#root>

switch#

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

- Zähler dürfen nicht inkrementiert werden; Jede Erhöhung weist auf potenzielle Layer-1- oder

Hardware-bezogene Probleme hin, wie z. B. physische Verbindungsfehler, CRC-/FCS-Fehler oder Pufferüberläufe/-verluste.

Routing und ARP-Stabilität

Die Sicherstellung von Routing- und ARP-Stabilität ist von entscheidender Bedeutung, um sicherzustellen, dass der Nexus-Switch konsistent auf Layer 3 erreichbar ist und keine intermittierenden Probleme bei der Auflösung verursacht, die die TCP-Leistung beeinträchtigen könnten. Instabilität bei Routing-Einträgen oder eine ARP-Auflösung können zu Paketverlusten, erhöhter Latenz oder Blackholing von Datenverkehr führen.

Validierungskriterien

- Routingeinträge für Quelle und Ziel müssen vorhanden, stabil und dürfen sich nicht häufig ändern.
- ARP-Einträge müssen aufgelöst werden und dürfen nicht fortlaufend aktualisiert werden oder fehlen.

<#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
```

Überprüfen, ob Datenverkehr nicht an CPU weitergeleitet wird

Bei Cisco Nexus Switches der Serie 9000 erfolgt die Weiterleitung über die Hardware (ASIC), und die CPU ist nicht am normalen Betrieb der Datenebene beteiligt. Daher ist die Beobachtung von Host-zu-Host-TCP-Datenverkehr auf der Steuerungsebene ungewöhnlich und weist darauf hin, dass Pakete aufgrund von Ausnahmen oder Fehlkonfigurationen blockiert werden. Sobald der Datenverkehr von der CPU verarbeitet werden muss, unterliegt er dem Control Plane Policing, und es wird erwartet, dass Datenverluste auftreten können, wenn der Datenverkehr die zulässige Kontrollebenenrate überschreitet.

Validierungsmethode

- Erfassung von Datenverkehr, der die Steuerungsebene erreicht, mit Ethanalyzer:

<#root>

```
switch#
```

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

Erwartetes Verhalten

- In der CPU kann kein Datenverkehr von Host zu Host-TCP auf Datenebene beobachtet werden.

Unerwartetes Verhalten

- Wenn Pakete, die dem Datenfluss entsprechen, sichtbar sind, wird Datenverkehr blockiert, was folgende Ursachen haben kann:
  - Herausragende Paketverarbeitung (TTL-Ablauf, ACL-Protokollierung, Umleitungen)
  - Falsche Konfiguration oder nicht unterstützte Funktionen
  - Falsche Hardware-Programmierung

Bestimmen der Latenz bei der Paketweiterleitung

Die Paketweiterleitungslatenz in Nexus 9000-Switches hängt von der Paketgröße, dem Weiterleitungsmodus und den aktivierten Funktionen ab. Die Spezifikationen von Cisco beziehen sich bei der Cut-Through-Weiterleitung für 64-Byte-Pakete in der Regel auf die Latenz.

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

- Cut-Through-Weiterleitung (Standard in Nexus 9000):
  - Beginnt mit der Weiterleitung, bevor das vollständige Paket empfangen wird.
  - Minimiert Latenz (unter Mikrosekunden bis ~1 µs).
- Store-and-forward:
  - Das gesamte Paket muss vor der Weiterleitung empfangen werden.
  - Fügt die Latenz proportional zur Paketgröße hinzu

Zusätzliche Funktionen können zu inkrementeller Latenz führen:

- VXLAN-Kapselung/-Entkapselung
- ACL-Suchen (TCAM-Verarbeitung)
- QoS-Klassifizierung und Warteschlangenverwaltung
- Telemetrie (NetFlow, ERSPAN, sFlow)
- Pufferung bei Überlastung

Allerdings:

- Diese Vorgänge werden in Hardware-Pipelines durchgeführt.

Das einzig realistische Szenario, in dem die Latenz merklich zunimmt, sind Überlastungen:

- Pakete werden in Ausgangswarteschlangen gepuffert.
- Die Verzögerung hängt ab von:
  - Warteschlangentiefe
  - Schnittstellenauslastung
  - QoS-Richtlinien

Selbst in diesen Fällen:

- Die Latenz liegt normalerweise im Mikrosekunden- bis niedrigen Hundert-Mikrosekunden-Bereich.
- Eine anhaltende Verzögerung im Millisekundenbereich würde Folgendes bedeuten:
  - Starke Überlastung
  - Überbelegung
  - Falsch konfigurierte QoS oder Pufferung

SPAN zu CPU (Paketerfassung für Datenebene)

Dies ermöglicht die Spiegelung des Datenverkehrs auf Datenebene in die Steuerungsebene zur Paketerfassung und den Export in eine .pcapng-Datei, sodass detaillierte Analysen in Wireshark durchgeführt werden können.

Konfiguration

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

Erfassungsausführung

<#root>

```
switch#
```

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

Technische Aspekte

- An die CPU gespiegelter Datenverkehr unterliegt dem Control Plane Policing (CoPP).
- Wenn der Datenverkehr die CoPP überschreitet:
  - Pakete können nur auf der Kontrollebene verworfen werden.
  - Dies führt zu Fehlalarmen während der Analyse.
- SPAN zu CPU wird für Szenarien mit geringem bis mittlerem Datenverkehr empfohlen.
- Für Umgebungen mit hohem Durchsatz:
  - Lokales SPAN verwenden (externer Analyzer)

- Verwendung von ERSPAN für die Remote-Erfassung

Methode	Vorteil	Einschränkung
SPAN	Präzise, keine Kapselung	Erfordert physische Verbindung.
ERSPAN	Remote-Erfassungsfunktion	Anfällig für Netzwerküberlastungen.

Validierung der Durchsatzbegrenzung für die Kontrollebene

Um die Zuverlässigkeit der SPAN-zu-CPU-Daten sicherzustellen, muss überprüft werden, ob die Kontrollebene aufgrund der Ratenbegrenzung gespiegelte Pakete verwirft.

Validierungsbefehl

```
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
```

Validierungsmethode

- Führen Sie den Befehl in Intervallen von ca. 3 Sekunden aus.
- Beobachten Sie die SPAN-bezogenen Zähler zum Verwerfen.

Dolmetschen

- Keine Erhöhung der Verwerfungszähler für die SPAN-Zeile weist auf eine zuverlässige Erfassung hin.
- Zunehmende Zähler für das Verwerfen weisen auf einen Paketverlust auf Kontrollebene hin, wodurch die Erfassung unzuverlässig wird.

Wenn Verwerfungen festgestellt werden, muss die Fangmethode auf SPAN oder ERSPAN geändert werden.

ICMP-basierte Validierung vor TCP

ICMP-Tests bieten eine Baseline-Validierung der Integrität der Datenebene vor der Durchführung komplexer TCP-Analysen. Da ICMP Stateless und einfacher ist, ermöglicht es die schnelle Erkennung von Paketverlusten, Duplizierungen oder Pfadinkonsistenzen.

Erwartetes Verhalten bei der SPAN-Erfassung

- Jedes ICMP-Paket kann zweimal vorkommen:
  - Einmal eingehend
  - Einmal ausgegangen
- Für einen Standard-Ping:
  - Echoanforderung → 2 Pakete
  - Echo Reply → 2 Pakete

Dies bestätigt die korrekte Weiterleitung und das Fehlen von Paketverlusten auf Datenebene.

Ungewöhnliches Verhalten

- Fehlende Duplikate oder asymmetrische Paketzähler weisen auf potenziellen Paketverlust oder Erfassungseinschränkungen hin.
- Zeitüberschreitungen weisen auf Layer-1-Probleme, Überlastungen oder Upstream-Probleme hin.

Wenn ICMP-Datenverkehr konsistent und verlustfrei weitergeleitet wird, besteht eine hohe Wahrscheinlichkeit, dass der TCP-Datenverkehr auch auf Layer 2/3 korrekt weitergeleitet wird.

Bestimmen der Nexus-Switch-Weiterleitungslatenz mithilfe der Paketerfassung

Wenn der Datenverkehr über SPAN zur CPU (oder SPAN/ERSPAN) erfasst wird, kann jedes Paket zweimal beobachtet werden: einmal am Eingang und einmal am Ausgang. Diese Duplizierung kann verwendet werden, um die vom Nexus-Switch verursachte Weiterleitungslatenz zu schätzen, indem die Zeitdifferenz zwischen beiden Instanzen desselben Pakets berechnet wird.

In der Praxis kann diese Latenz mithilfe des zuvor erfassten ICMP-Verkehrs gemessen werden, indem das Zeitdelta zwischen duplizierten Echo Request- und Echo Reply-Paketen verglichen wird. Dies bietet eine einfache und effektive Grundlage für die Switch-Weiterleitungsleistung. Wenn eine tiefere Analyse erforderlich ist, kann dieselbe Methode auf den TCP-Datenverkehr angewendet werden, indem der Datenfluss erfasst und die Zeitdifferenz zwischen duplizierten

TCP-Paketen gemessen wird.

Methodik

- Identifizieren Sie ein Paket und dessen Duplikat (gleiche Sequenznummer).
- Messen Sie das Zeitdelta zwischen den Eingangs- und Ausgangskopien.
- Dieses Delta stellt eine Obergrenze für die geschätzte Weiterleitungslatenz des Switches dar, da es Spiegelung und Zeitstempel für den Overhead umfassen kann.

Wireshark-Konfiguration

- Zeitdelta-Anzeige aktivieren:

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

- Hinzufügen einer benutzerdefinierten Spalte für das Zeitdelta:

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

- Filtern von relevantem Datenverkehr (Beispiel):

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

- Pakete nach Sequenznummer oder TCP-Stream sortieren:

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

Dolmetschen

- Das Zeitdelta zwischen duplizierten Paketen kann im Mikrosekundenbereich liegen.
  - In diesem Fall führt der Nexus-Switch keine Latenz für die Paketweiterleitung ein.
- Konsistent niedrige Deltas bestätigen die hardwarebasierte Weiterleitungsleistung.
- Höhere oder inkonsistente Deltas können auf Folgendes hinweisen:
  - Überlastung oder Pufferung

Referenzen

- Datenblätter zur Cisco Nexus Serie 9000
- Cisco Nexus Switches der Serie 9000 - Designleitfäden
- Whitepaper: Intelligentes Puffer-Management für Cisco Nexus Switches der Serie 9000

TCP-Datenverkehrsanalyse von der Paketerfassung des Quellhosts

Dieser Abschnitt enthält eine detaillierte Methodik für die Analyse einer TCP-Paketerfassung in Wireshark, einschließlich der Profilkonfiguration, anhand des oben beschriebenen hypothetischen Falls. Die gezeigten Bilder stammen direkt aus Wireshark. Zur Erinnerung:

Ein Benutzer hat festgestellt, dass der Backup-Prozess für einen Anwendungsdatensatz von ca. 6,5 TB, der zuvor in ca. 9 Stunden abgeschlossen wurde, jetzt fast 21 Stunden dauert. Das einzig zugängliche Netzwerkgerät ist ein mit dem Quellserver (10.93.19.8) verbundener Cisco Nexus 9300-Switch. Die auf der Switch-Schnittstelle konfigurierte MTU beträgt 9.000 Byte (Jumbo Frames), während die MTU auf dem Server unbekannt ist. Eine Paketerfassung vom Quellserver ist verfügbar, und alle vorherigen Nexus-Validierungsschritte wurden bereits abgeschlossen, ohne dass Anomalien erkannt wurden.

Wichtigste Beobachtungen und Einschränkungen

- Nexus-Switch wurde ausgeschlossen:
  - Keine Paketverluste
  - Keine Schnittstellenfehler
  - Keine Auswirkungen auf QoS oder ACL
  - Hardware-Weiterleitung bestätigt
- Schnittstellenkonfiguration:
  - Zugangsport
  - MTU: 9000 Byte
- Verfügbare Daten:
  - Paketerfassung an der Quelle
  - Umfassende MTU-Kenntnisse
    - Der Ping-Test wurde mit einem 1500-Byte-Paket mit 1.472 Byte Daten ohne Fragmentierung erfolgreich abgeschlossen.
- Fehlende Daten:
  - Zieltransparenz
  - Auf dem Zielserver ist keine Paketerfassung verfügbar.

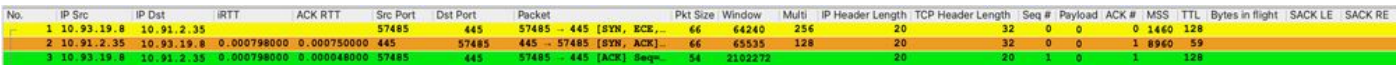
In Wireshark können Sie benutzerdefinierte Profile erstellen, die auf den jeweiligen Analysetyp, den Sie ausführen möchten, zugeschnitten sind.

Spaltenbeschreibung

- tcp.analysis.initial_rtt (iRTT): Schätzt die anfängliche Round-Trip-Zeit basierend auf dem TCP-Drei-Wege-Handshake.
- tcp.analysis.ack_rtt (ACK-RTT): Misst die Zeit zwischen einem TCP-Segment und seiner entsprechenden Bestätigung.
- tcp.window_size (Fenster): Zeigt die vom Empfänger angegebene TCP-Fenstergröße an, bevor die Skalierung angewendet wird.
- tcp.options.wscale.multiplier (Multi): Stellt den Fensterskalierungsfaktor dar, der zum Berechnen des effektiven Empfangsfensters verwendet wird.
- tcp.seq (Seq#): Zeigt die Sequenznummer des ersten Bytes im TCP-Segment an.
- tcp.len (Nutzlast): Zeigt die Größe der TCP-Nutzlast in Byte für dieses Segment an.
- tcp.ack (ACK-Nr.) Zeigt das nächste erwartete Byte des Absenders an (kumulative Bestätigung).
- tcp.options.mss_val (MSS): Zeigt die maximale Segmentgröße an, die während des TCP-Handshakes angegeben wurde.
- ip.ttl (TTL): Zeigt den Time To Live-Wert an, der zum Identifizieren der Hop-Anzahl und des Routingverhaltens nützlich ist.
- tcp.analysis.bytes_in_flight (Bytes im Flug): Stellt die Menge der unbestätigten Daten dar, die derzeit übertragen werden.

Analyse des TCP-Drei-Wege-Handshakes

Das Erfassen des TCP-Drei-Wege-Handshakes ist erforderlich, da es wichtige Parameter wie MSS, Window Scale und SACK enthält, die das Sitzungsverhalten festlegen.
Ohne diese Informationen ist jede TCP-Analyse unvollständig und kann zu falschen Schlussfolgerungen in Bezug auf die Leistung oder die Ursache führen.



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			

Identifizierung des Datenverkehrs

Aus der Paketerfassung:

- IP-Quelladresse: 10.93.19.8
- Ziel-IP-Adresse: 10.91.2.35

Analyse der anfänglichen Round-Trip-Zeit (iRTT)

Der anfängliche RTT (iRTT) wird wie folgt berechnet:

- iRTT = 798 Mikrosekunden

Dieser Wert wird abgeleitet von:

- Paket 2 (SYN-ACK) ACK RTT: 750 µs → Zeit, bis das Ziel auf die SYN reagiert.
- Paket 3 (ACK) ACK RTT: 48 µs → Zeit für die Quellenbestätigung der SYN-ACK.

Der Großteil der Latenz (~94 %) liegt im Weiterleitungspfad (Client → Server → Client), während die Reaktionszeit von der Quelle minimal ist, was darauf hindeutet, dass der Client keine CPU- oder Anwendungsverzögerung aufweist.

TCP-Port-Identifizierung

- Ziel-TCP-Port: 445

Port 445 entspricht dem Microsoft Server Message Block (SMB), der häufig für die Dateifreigabe, Netzwerklaufwerke und Windows-Authentifizierungsdienste verwendet wird. Da dieses Protokoll sowohl auf Latenz als auch auf Durchsatz empfindlich ist, ist es stark von der TCP-Effizienz und der Netzwerkstabilität abhängig.

TCP-Fenstergrößenanalyse

- Quellfenster (skaliert): 64,240 Byte
- Zielfenster: 65,535 Byte

Das TCP-Fenster gibt an, wie viele Daten gesendet werden können, bevor auf die Bestätigung gewartet wird. In diesem Fall ist die Quelle etwas restriktiver als das Ziel. Diese Werte sind in modernen Umgebungen relativ gering und können den Durchsatz insbesondere bei zunehmender RTT begrenzen.

Der maximale theoretische Durchsatz kann wie folgt geschätzt werden:

Durchsatz = TCP-Fenstergröße / RTT

Ersetzen der beobachteten Werte:

- TCP-Fenstergröße = 64.240 Byte
- RTT = 798 Mikrosekunden = 0,000798 Sekunden

Durchsatz ≈ 64.240 / 0,000798 ≈ 80,5 MB/s (~644 Mbit/s)

Dieser Wert stellt den oberen Grenzwert für den Durchsatz dar. Dabei wird Folgendes vorausgesetzt:

- Kein Paketverlust
- Keine Neuübertragungen
- Ideale Netzwerkbedingungen

Analyse von Durchsatz, Übertragungszeit und erforderlichen Bedingungen

Bei einem aktuellen Durchsatz von 644 Mbit/s dauert die Übertragung einer Datei mit 6,5 TB ungefähr 23,5 Stunden, was mit der beobachteten Verschlechterung übereinstimmt. Um ein Übertragungsfenster von 9 Stunden zu erreichen, muss der Durchsatz auf ca. 1,68 Gbit/s erhöht werden. Dies erfordert entweder ein größeres TCP-Fenster (~2,7x Erhöhung) oder eine deutlich niedrigere RTT (~291 µs).

Unter den aktuellen Bedingungen (64 KB Fenster und ~798 µs RTT) ist es nicht möglich, das 9-Stunden-Ziel zu erreichen, da der TCP-Durchsatz durch das Bandbreiten-Verzögerungsprodukt eingeschränkt ist. Ohne die Fenstergröße zu erhöhen oder die Latenz zu reduzieren, kann das Protokoll keine höhere verfügbare Bandbreite nutzen, wodurch das Ziel unerreichbar wird.

Szenario	Durchsatz	Geschätzte Übertragungszeit (6,5 TB)	Erforderliches TCP-Fenster	Erforderliche RTT
Aktueller Status	644 Mbit/s (~80,5 MB/s)	~23,5 Stunden	64 KB	798 µs
Ziel (9 Stunden)	~1683 Mbit/s (~210 MB/s)	9 Stunden	~172 KB	~291 µs

Dies funktionierte bereits, was darauf hinweist, dass eine Änderung im Netzwerk, in der Anwendung, in der Quelle oder im Ziel aufgetreten ist. Es ist wichtig, darauf hinzuweisen, dass allein auf der Grundlage dieser ersten Analyse bereits eine wesentliche Schlussfolgerung gezogen werden kann: Unter den aktuellen TCP-Fenstergrößen und RTT-Bedingungen ist das Erreichen des 9-Stunden-Ziels nicht möglich.

In den Tabellen wird verglichen, wie sich der Durchsatz bei einer Vergrößerung oder Verkleinerung des RTT- und TCP-Fensters ändert.

RTT-Auswirkung auf den Durchsatz (feste Fenstergröße = 64.240 Byte)

RTT	Durchsatz (MB/s)	Durchsatz (Mbit/s)
200 µs (0,0002 s)	~321 MB/s	~ 2.568 Mbit/s
798 µs (0,000798 s)	~80,5 MB/s	~644 Mbit/s
2 ms (0,002 s)	~32,1 MB/s	~257 Mbit/s
10 ms (0,01 s)	~6,4 MB/s	~51 Mbit/s

Auswirkungen auf die TCP-Fenstergröße (Fixed RTT = 798 µs)

TCP-Fenstergröße	Durchsatz (MB/s)	Durchsatz (Mbit/s)
16 KB (16 384 B)	~20,5 MB/s	~ 164 Mbit/s
64 KB (64 240 B)	~80,5 MB/s	~644 Mbit/s
256 KB (262 144 B)	~328 MB/s	~ 2.624 Mbit/s
1 MB (1.048.576 B)	~1.314 MB/s	~10,5 Gbit/s

Technische Interpretation

- Der Durchsatz ist umgekehrt proportional zum RTT →. Eine höhere Latenz reduziert die Leistung.
- Der Durchsatz ist direkt proportional zur TCP-Fenstergröße → größere Fenster erhöhen die Kapazität.
- Kleine Fenstergrößen schränken den Durchsatz selbst in Umgebungen mit niedriger Latenz erheblich ein.
- Hochgeschwindigkeitsnetzwerke (10 G+) erfordern eine Fensterskalierung, um die Bandbreite voll auszunutzen.

Dies zeigt, dass sowohl die RTT- als auch die TCP-Fenstergröße wichtige Faktoren für die TCP-Leistung sind und bei der Behebung von Durchsatzproblemen zusammen analysiert werden

müssen.

IP- und TCP-Header-Länge

- IP-Header-Länge: 20 Byte
- TCP-Headerlänge: 32 Byte

Ein 20-Byte-IP-Header gibt an, dass keine IP-Optionen vorhanden sind. Der 32-Byte-TCP-Header bestätigt, dass TCP-Optionen verwendet werden, und fügt 12 Byte über den Basis-Header hinzu. Zu diesen Optionen gehören in der Regel MSS, Fensterskalierung und SACK Permitted.

Analyse der TCP-Optionen und TTL

Die selektive Bestätigung (SACK) ist auf beiden Endpunkten aktiviert. Dies ist auf dem Bild nicht sichtbar. Mit SACK kann der Empfänger nicht zusammenhängende Datenblöcke bestätigen und dem Absender genau mitteilen, welche Segmente erfolgreich empfangen wurden.

Wenn beispielsweise die Segmente 1000-2000 und 3000-4000 empfangen werden, aber 2000-3000 fehlt, kann der Empfänger dies explizit angeben. Ohne SACK würde der Absender alle Daten nach der Lücke erneut übermitteln; mit SACK wird nur der fehlende Teil erneut übertragen. Dadurch wird die Leistung in Umgebungen mit Paketverlusten erheblich verbessert.

Paket 1 (SYN)-Analyse

- Folgenummer: 0 (Wireshark normalisiert)
- Nutzlast: 0 Byte
- ACK-Nr.: 0
- MSS: 1460 Byte
- TTL: 128

Wireshark normalisiert aus Gründen der Lesbarkeit die Sequenznummer auf Null, obwohl es sich in der Praxis um einen großen Zufallswert handelt. Während des Verbindungsaufbaus wird erwartet, dass keine Nutzlast vorhanden ist. Der MSS-Wert von 1.460 Byte gibt eine MTU von 1.500 Byte an (20 Byte IP-Header + 20 Byte TCP-Header). Ein TTL von 128 kann ein Windows-basierter Host sein. Wenn dieser Wert bei der Erfassung angezeigt wird, bedeutet dies, dass die Erfassung wahrscheinlich über Layer 2 an oder in unmittelbarer Nähe der Quelle durchgeführt wurde.

Paket-2-Analyse (SYN-ACK)

- ACK-Nr.: 1

Der ACK-Wert ist 1, da das SYN-Flag eine Sequenznummer belegt, auch wenn keine Nutzlast vorhanden ist. Daher ist ACK = SEQ + 1.

- TTL: 59

Der beobachtete TTL-Wert von 59 deutet auf einen anfänglichen TTL-Wert von 64 hin, d. h., das Paket durchlief etwa 5 Routing-Hops (64 − 59 = 5). Jeder geroutete Hop dekrementiert den TTL um eins.

Fragmentierungsrisiken und Auswirkungen auf Netzwerke

Das Vorhandensein von ca. fünf Routing-Hops birgt potenzielle Leistungsrisiken, insbesondere im Zusammenhang mit MTU-Diskrepanzen und der Fragmentierung.

Wenn eine zwischengeschaltete Verbindung eine geringere MTU als die ursprüngliche Paketgröße hat, kann es zu einer Fragmentierung kommen. Dies hat mehrere Konsequenzen:

- Höhere Latenz aufgrund von Fragmentierungs- und Reassemblierungsaufwand
- Höhere Wahrscheinlichkeit eines Paketverlusts, da der Verlust eines einzelnen Fragments die erneute Übertragung des gesamten Pakets erfordert.
- Reduzierter Durchsatz, da TCP Verluste als Überlastung interpretiert und die Senderate reduziert.
- Verbesserte CPU-Auslastung auf Netzwerkgeräten, die die Fragmentierung handhaben.
- Das Risiko von PMTUD-Ausfällen (Path MTU Discovery), wenn ICMP blockiert wird, führt zu automatischen Paketverlusten.

Angesichts dieser Faktoren ist es wichtig, eine konsistente MTU über den Pfad sicherzustellen oder ggf. eine MSS-Klemmung zu implementieren.

TCP-RTT-Analyse: ACK RTT und anfängliche RTT

Wenn ACK RTT größer als iRTT ist, weist dies darauf hin, dass die Latenz im Vergleich zur Baseline, die während des TCP-Handshakes festgelegt wurde, erhöht wurde.

Das bedeutet, dass das Netzwerk oder die Endgeräte zusätzliche Verzögerungen während der Sitzung verursachen, was in der Regel auf Folgendes zurückzuführen ist:

- Netzwerküberlastung oder Warteschlange

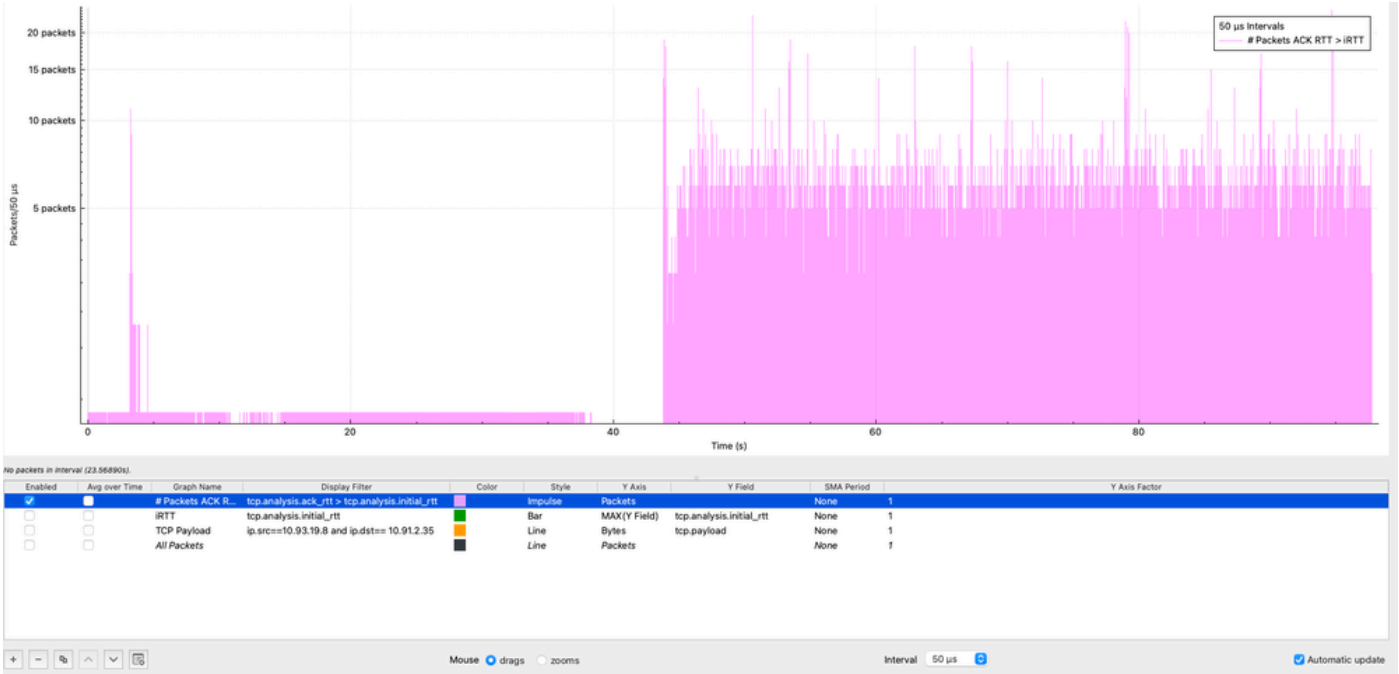
- Verzögerungen bei der Empfänger- oder Anwendungsverarbeitung
- Zwischengeräte (Firewalls, Load Balancer)
- Erneute Übertragungen

Wenn diese Bedingung während der TCP-Sitzung beibehalten wird, führt dies zu:

- Reduzierter TCP-Durchsatz
- Ineffiziente Fensterauslastung
- Beeinträchtigte Anwendungsleistung

In Wireshark können Sie mithilfe der Funktion "I/O Graphs" (E/A-Diagramme) anzeigen, wie oft die Bedingung ACK RTT > iRTT auftritt: Statistiken → E/A-Diagramme, Anwenden des Anzeigefilters (tcp.analysis.ack_rtt > tcp.analysis.initial_rtt), Auswählen des Impulsstils, Festlegen der Y-Achse auf Pakete und Verwenden eines Intervalls von 50 Mikrosekunden.

Im Diagramm stellen die violetten Impulse die Anzahl der Pakete dar, die diese Bedingung innerhalb eines Intervalls von jeweils 50 Mikrosekunden erfüllen. Wie beobachtet, besteht dieser Zustand während der gesamten Paketerfassung fort, was darauf hinweist, dass die Latenz während der Sitzung durchweg höher ist als die ursprüngliche Baseline. Dieses Verhalten deutet stark auf eine nachhaltige Leistungsminderung statt auf einen vorübergehenden Zustand hin und verstärkt die Notwendigkeit, potenzielle Quellen wie Überlastung, Pufferung oder Verzögerungen bei der Endpunktverarbeitung über den End-to-End-Pfad zu untersuchen.

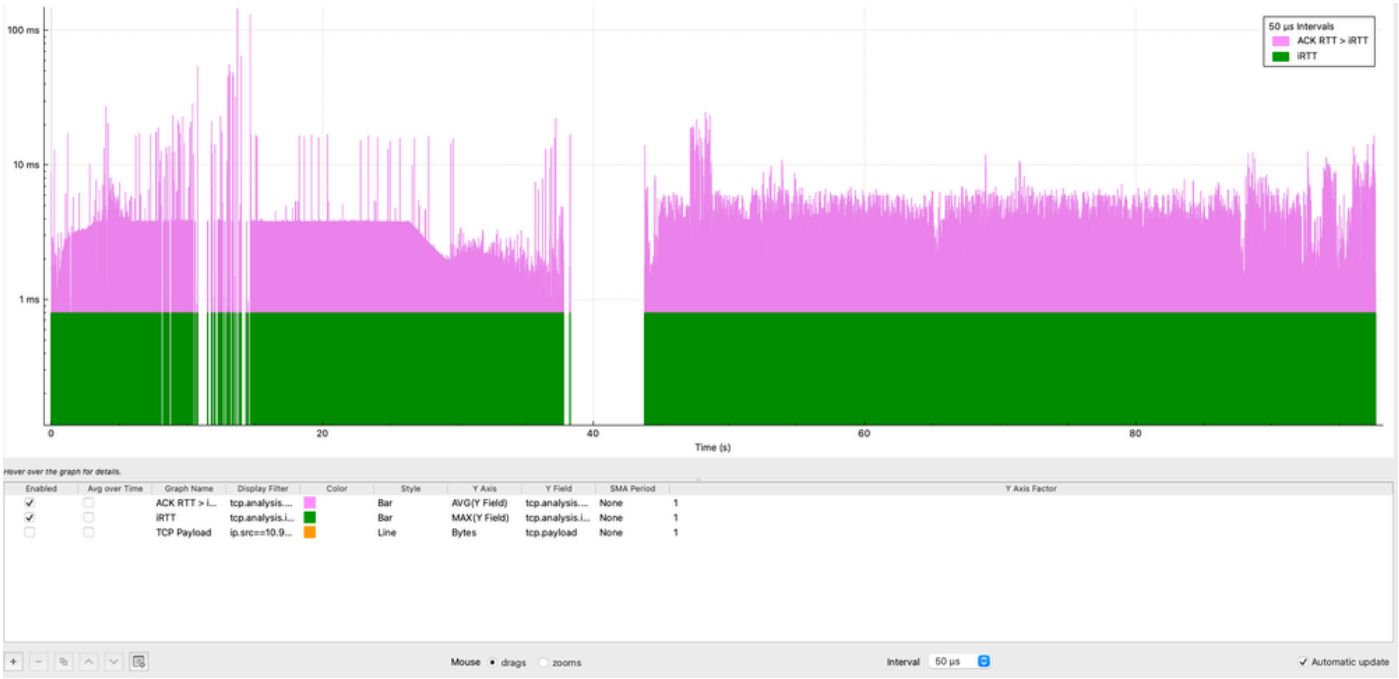


Es ist auch wichtig festzustellen, wie lange die iRTT überschritten wird, nicht nur wie oft. Obwohl Wireshark keine direkte Subtraktion zwischen Feldern zulässt, kann mithilfe von E/A-Diagrammen ein visueller Vergleich erzielt werden:

- Navigieren zu Statistiken → E/A-Diagramme
- Schaubild 1:
  - Anzeigefilter: tcp.analysis.ack_rtt > tcp.analysis.initial_rtt
  - Stil: Leiste
  - Y-Achse: Durchschnitt
  - Y-Feld: tcp.analysis.ack_rtt
  - Intervall: 50 Mikrosekunden
- Schaubild 2:
  - Anzeigefilter: tcp.analysis.initial_rtt
  - Stil: Leiste
  - Y-Achse: MAX.
  - Y-Feld: tcp.analysis.initial_rtt
- Klicken Sie dann mit der rechten Maustaste auf das Diagramm, und aktivieren Sie die Protokollskala.

In dieser Visualisierung stellt der lila Graph die Bedingung ACK RTT > iRTT dar, die während der gesamten TCP-Sitzung konsistent vorhanden ist. Die Daten zeigen eine anhaltende Latenzinflation, bei der mehrere Spitzen 11 Millisekunden erreichen und ein maximaler Spitzenwert von über 100 Millisekunden beträgt, was dem 11- bis 100-fachen des iRTT-Basiswerts entspricht.

Dieses Verhalten bestätigt, dass die Latenzerhöhung nicht vorübergehend, sondern dauerhaft ist, was auf ein systemisches Problem hinweist, das die Sitzung im Laufe der Zeit beeinträchtigt. Diese anhaltende Abweichung deutet stark auf Faktoren wie Netzwerküberlastung, Pufferung (Bufferbloat) oder Verzögerungen bei der Endpunktverarbeitung hin.



Analyse von TCP-Neuübertragungen und unbegründeten Neuübertragungen

In diesem Abschnitt wird die TCP-Zuverlässigkeit bewertet, indem die Übertragungswiederholung analysiert wird. Auf diese Weise kann geprüft werden, ob der Paketverlust zu einer Leistungsverschlechterung beiträgt.

TCP-Neuübertragungen im Laufe der Zeit

Das Diagramm zeigt die Verteilung der TCP-Neuübertragungen über die Zeit. Insgesamt wurden 42 Neuübertragungen beobachtet, was nur 0,00125 % des gesamten Datenverkehrs ausmacht.

Dieses Ausmaß an wiederholten Übertragungen ist vernachlässigbar und zeigt deutlich an, dass der Paketverlust in diesem Szenario keinen Beitrag leistet.

Wireshark-Konfiguration (TCP-Neuübertragungen)

```
Statistics → I/O Graphs
```

- Anzeigefilter:

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

- Stil: Impuls oder Balken
- Y-Achse: Pakete
- Intervall: 1 Sekunde

TCP-Neuübertragungen

Das Diagramm zeigt die Anzahl der TCP-Neuübertragungen in Intervallen von 1 Sekunde, die von der Quelle 10.93.19.8 generiert wurden.

In Wireshark gibt eine TCP Spurious Retransmission an, dass ein Host ein Segment erneut übertragen hat, das noch nicht verloren gegangen ist. Das ursprüngliche Paket erreichte den Empfänger erfolgreich, aber der Sender nahm aufgrund ungenauer Zeitschätzung fälschlicherweise einen Verlust an. Dieses Verhalten weist nicht auf einen echten Paketverlust hin, sondern auf eine ineffiziente Weiterleitungslogik beim Sender.

In dieser Aufzeichnung:

- Die Quelle 10.93.19.8 überträgt Pakete nach nur ~8 Mikrosekunden erneut.
- Die typischen Timer für die Neuübertragung liegen bei ca. 200 Millisekunden.

Dies bestätigt, dass das Übertragungsverhalten vollständig vom Quell-TCP-Stack und nicht vom Netzwerk gesteuert wird.

Insgesamt wurden 1.112 unberechtigte Neuübertragungen beobachtet, was 0,0332 % des gesamten erfassten Datenverkehrs entspricht.

Wireshark-Konfiguration (TCP Spurious Retransmissions)

```
Statistics → I/O Graphs
```

- Anzeigefilter:

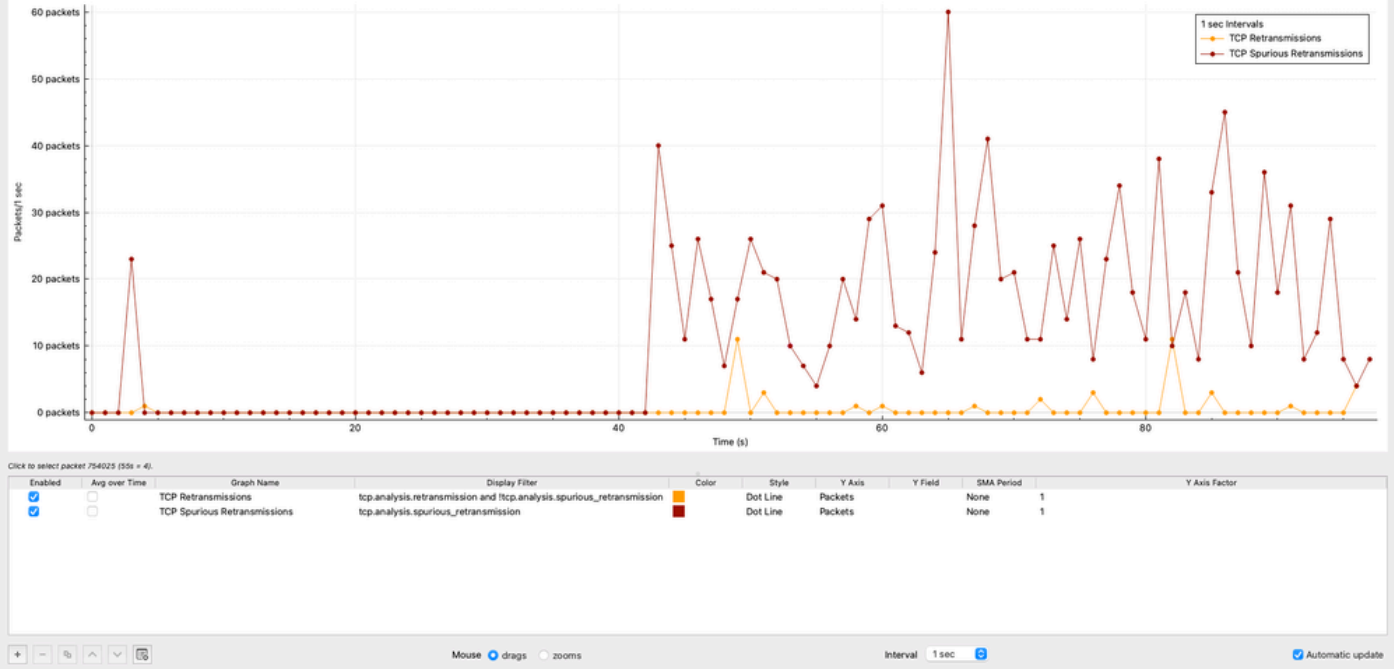
```
tcp.analysis.spurious_retransmission and ip.src==10.93.19.8
```

- Stil: Impuls oder Balken
- Y-Achse: Pakete
- Intervall: 1 Sekunde

Technische Interpretation

- Der extrem niedrige Prozentsatz echter Neuübertragungen bestätigt, dass im Netzwerk kein Paketverlust auftritt.
- Das Vorhandensein falscher Übertragungen deutet auf vorzeitige Weiterleitungsentscheidungen des Quell-Hosts hin.
- Dieses Verhalten kann sich geringfügig auf die Effizienz auswirken, ist jedoch keine Hauptursache für eine schwerwiegende Beeinträchtigung des Durchsatzes.

Diese Analyse bestätigt außerdem, dass das Problem nicht mit der Zuverlässigkeit des Netzwerks, sondern mit dem TCP-Verhalten, der Latenz oder der Leistung der Endgeräte zusammenhängt.

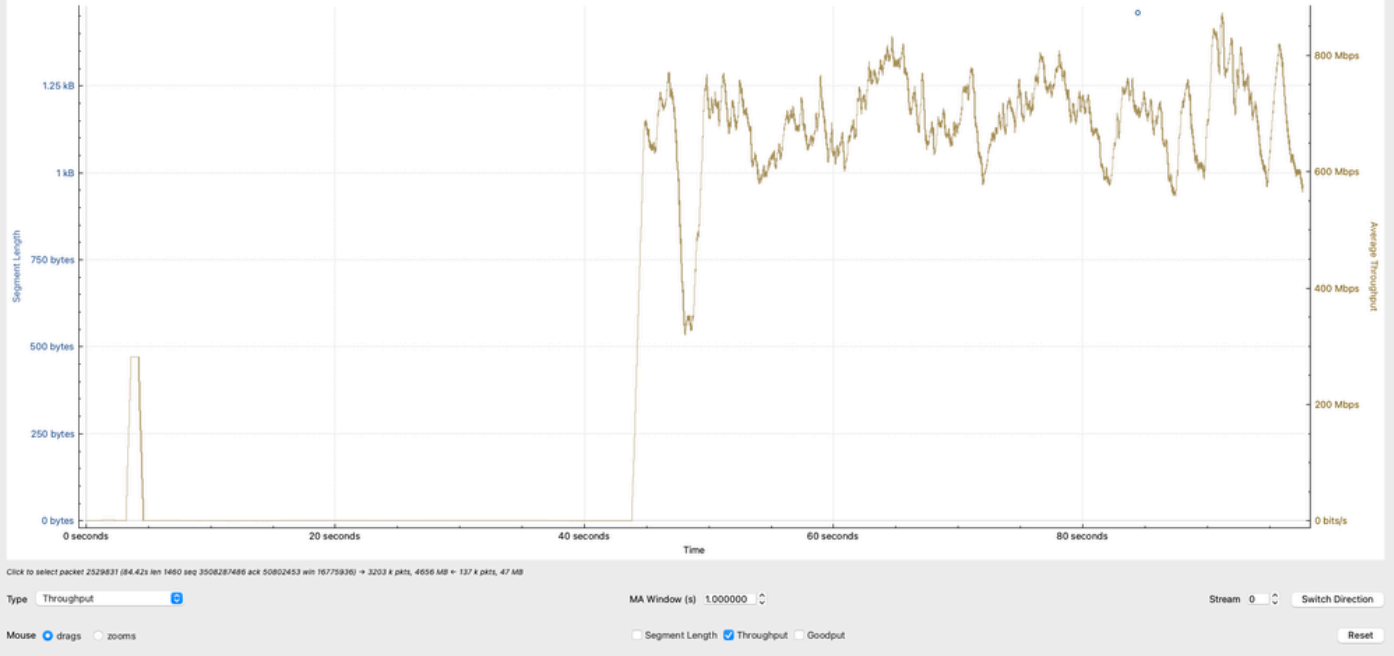


Effektive Durchsatzanalyse

Das Diagramm zeigt den effektiven Durchsatz, berechnet auf Basis der TCP-Nutzlast (tatsächlich übertragene Daten) in Megabit pro Sekunde. Der beobachtete Durchsatz schwankt primär zwischen 600 Mbit/s und 800 Mbit/s, was darauf hinweist, dass das Netzwerk während der aktiven Datenübertragung nicht das höhere Bandbreitenpotenzial erreicht.

Wireshark-Konfiguration (effektiver Durchsatz)

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



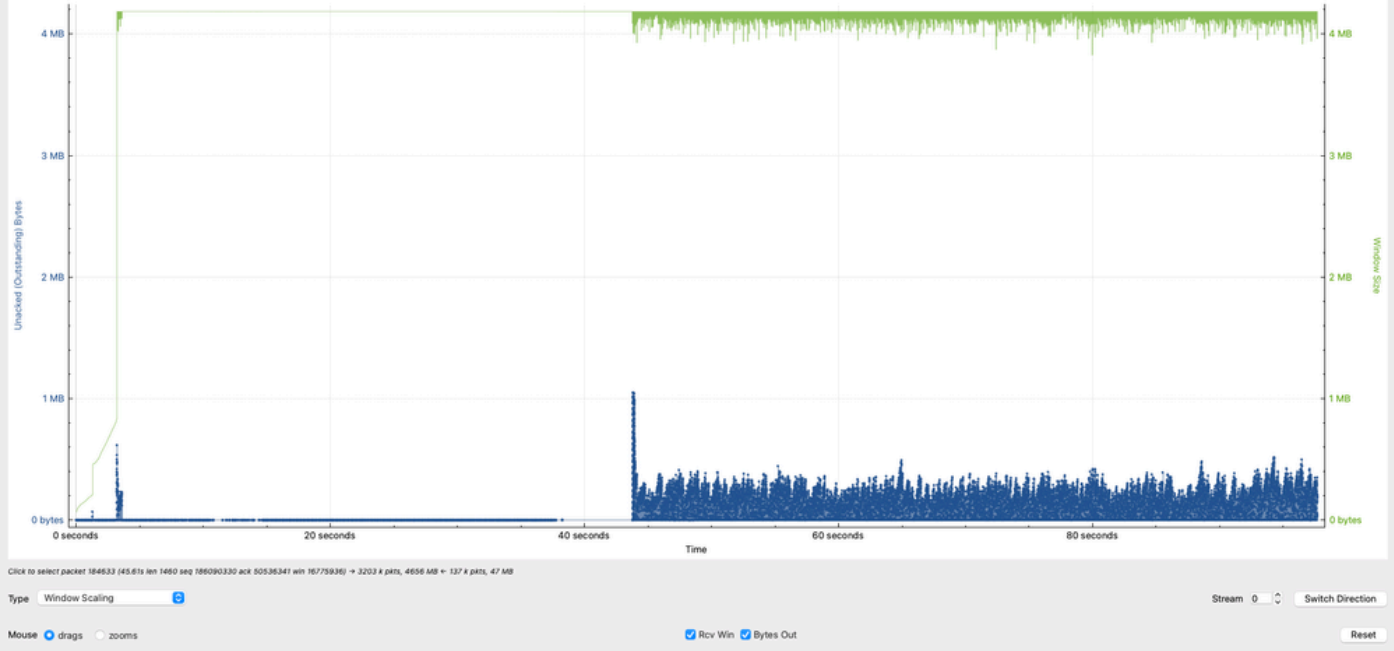
Technische Interpretation

- Der Durchsatzbereich von 600-800 Mbit/s entspricht den vorherigen Berechnungen, die auf der TCP-Fenstergröße und RTT basieren.
- Unterschiede im Durchsatz spiegeln Folgendes wider:
  - RTT-Schwankungen
  - Anpassungen der TCP-Überlastungskontrolle
  - Anwendungssteuerung oder -pufferung
- Da sich der Durchsatz der Leitungsgeschwindigkeit nicht nähert (z. B. 10G), besteht die Einschränkung nicht in physischer Bandbreite, sondern in TCP-Effizienzeinschränkungen.
- Diese Analyse bestätigt, dass der beobachtete Durchsatz mit den TCP-Einschränkungen (Fenstergröße und Latenz) übereinstimmt, und bestätigt, dass der Engpass nicht auf Paketverlust oder Schnittstellenkapazität zurückzuführen ist, sondern auf das Verhalten auf der Transportschicht und auf Endpunktbedingungen.

Analyse von übertragenen Daten (TCP-Fenster)

Das Diagramm zeigt ein kritisches Verhalten in der TCP-Sitzung, indem die Empfängerkapazität mit den tatsächlich übertragenen Daten (Bytes im Flug) verglichen wird.

- Die grüne Zeile steht für die Menge an TCP-Daten, die 10.91.2.35 (Empfänger) akzeptieren kann (effektives Empfangsfenster).
- Die blaue Linie steht für die Menge der TCP-Daten, die derzeit von 10.93.19.8 (Absender) übertragen werden.



Die beobachteten Flugdaten erreichen Spitzenwerte von ca. 1 MB, mit zusätzlichen Spitzenwerten von ca. 8 KB und 5 KB, sind aber primär zwischen 1 KB und 250 KB konzentriert.

Dies deutet darauf hin, dass der Empfänger zwar in der Lage ist, größere Datenmengen zu verarbeiten, jedoch das verfügbare Fenster nicht konsequent ausnutzt.

Wireshark-Konfiguration (Daten im Flug vs. Fenster)

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

Technische Interpretation

- Der Empfänger (10.91.2.35) kündigt ein wesentlich größeres Fenster an, das anzeigt, dass er mehr Daten empfangen kann.
- Der Absender (10.93.19.8) nutzt das verfügbare Fenster nicht voll aus, wie die niedrigeren und inkonsistenten Werte für "Daten im Flug" zeigen.
    - Der Absender kann die Werte für "Data in Flight" idealerweise näher am Fenster für die vom Empfänger gemeldeten Daten (~1 MB) verwalten, um den Durchsatz zu maximieren.
    - Die Unfähigkeit, hohe In-Flight-Datenniveaus aufrechtzuerhalten, schränkt den Durchsatz direkt ein und ist ein starker Indikator für TCP-Ineffizienz an der Quelle und kein Problem mit der Netzwerkkapazität.

Analyse von TCP-Payload und MSS-Over-Time

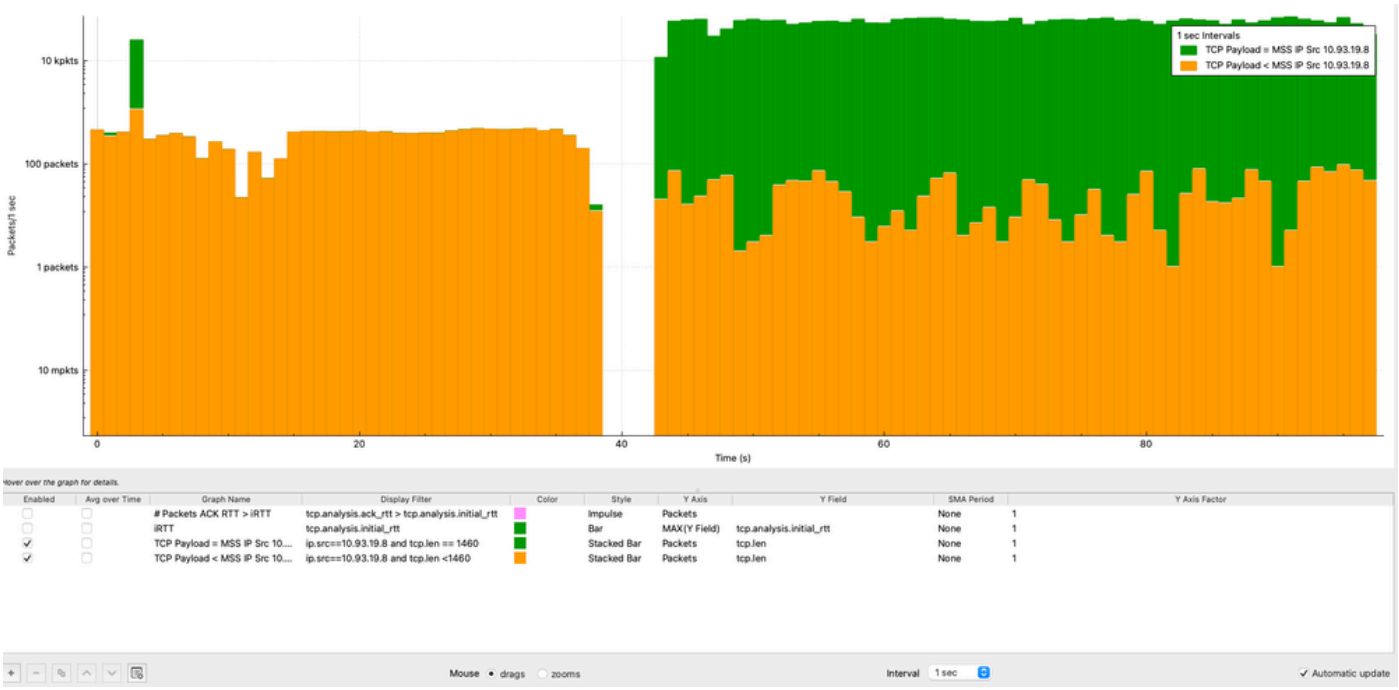
Durch die Analyse der TCP-Nutzlastgröße im Vergleich zur MSS im Zeitverlauf kann festgestellt werden, ob der Sender jedes TCP-Segment effizient nutzt. Diese Analyse wird aus der Perspektive der Quell-IP-Adresse (10.93.19.8) durchgeführt.

In Wireshark werden die Diagramme wie folgt konfiguriert:

- Grafik 1 (MSS-große Pakete):
  - Anzeigefilter: ip.src==10.93.19.8 und tcp.len == 1460
  - Stil: Gestapelte Leiste
  - Y-Achse: Pakete
  - Intervall: 1 Sekunde
- Abbildung 2 (alle Pakete ≤ MSS):
  - Anzeigefilter: ip.src==10.93.19.8 und tcp.len <= 1460
  - Stil: Gestapelte Leiste
  - Y-Achse: Pakete
  - Intervall: 1 Sekunde
- Logarithmische Skalierung für bessere Visualisierung anwenden

Aus der Analyse:

- Die Mehrheit der Pakete (>10.000 Pakete pro Sekunde) erreicht konsistent den MSS-Wert von 1460 Byte.
- Ein kleinerer Teil der Pakete überträgt aufgrund des normalen TCP-Verhaltens (ACKs, Segmentierung oder End-of-Stream-Daten) weniger Nutzlast.



Ursachenanalyse: Leistungsabfall bei TCP

Diese Analyse zeigt, dass die Identifizierung der Ursache von TCP-Leistungsproblemen einen ganzheitlichen End-to-End-Ansatz erfordert, anstatt davon auszugehen, dass das Netzwerk die Hauptursache für die Beeinträchtigung ist.

Der Cisco Nexus 9300-Switch wurde umfassend validiert. Dies umfasste Schnittstellenzähler, QoS-Richtlinien, Routing- und ARP-Stabilität, CPU-Punktverifizierung, SPAN-basierte Paketerfassung und die Validierung der Weiterleitung auf ASIC-Ebene mithilfe von ELAM. Alle Ergebnisse bestätigten konsistent, dass der Switch innerhalb der erwarteten Parameter funktionierte:

- Keine Paketverluste
- Keine ungewöhnliche Latenz (Mikrosekundenbereich)
- Keine Auswirkungen auf QoS oder Kontrollebene
- Richtige Hardware-Weiterleitung

Zusätzlich ergab die TCP-Analyse Folgendes:

- Unbedeutende Neuübertragungen (0,00125 %)
- Kein Hinweis auf Paketverlust
- Konsistente MSS-Nutzung an der Quelle
- Durchsatz abgestimmt auf TCP-Fenster- und RTT-Einschränkungen
- Unterauslastung des verfügbaren TCP-Fensters (Analyse von Daten im Flug)
- Das Netzwerk ist nicht der Engpass
- Der Quellserver schränkt die Leistung ein

Schlussfolgerung

Die Leistungseinbußen werden durch den Quellserver verursacht, der mit einer MTU von 1500 in einer Jumbo-fähigen Umgebung arbeitet, wodurch eine effiziente Nutzung der verfügbaren Netzwerkkapazität verhindert wird.

Lösung

Erhöhung der MTU auf dem Quellserver von 1500 auf 9000 Byte entsprechend der Ziel- und Netzwerkinfrastruktur Die Vorteile:

- Größere TCP-Segmente aktivieren

- Reduzierung des Paket-Overheads
- Verbesserung des Gesamtdurchsatzes

Technische Überlegungen

Eine wichtige Erkenntnis aus dieser Analyse ist, dass frühzeitige Schlussfolgerungen bei der Behebung von Problemen mit der Netzwerkleistung vermieden werden müssen. Obwohl es üblich ist, Probleme zunächst dem Netzwerk zuzuweisen, zeigt dieser Fall deutlich, dass das Netzwerk auf dem gesamten Datenebenenpfad ordnungsgemäß funktionierte. Nur durch eine umfassende TCP-Analyse sowohl aus Quell- als auch aus Zielperspektive - einschließlich Handshake-Parametern, RTT-Verhalten, Fensterauslastung, Neuübertragungen und Nutzlasteffizienz - konnte der tatsächliche Engpass genau identifiziert werden.

Indem Sie sich die Zeit nehmen, das TCP-Verhalten detailliert zu analysieren, verhindern Sie Fehldiagnosen, reduzieren unnötige Netzwerkänderungen und stellen sicher, dass die Beseitigung auf die tatsächliche Ursache ausgerichtet ist.

Informationen zu dieser Übersetzung

Cisco hat dieses Dokument maschinell übersetzen und von einem menschlichen Übersetzer editieren und korrigieren lassen, um unseren Benutzern auf der ganzen Welt Support-Inhalte in ihrer eigenen Sprache zu bieten. Bitte beachten Sie, dass selbst die beste maschinelle Übersetzung nicht so genau ist wie eine von einem professionellen Übersetzer angefertigte. Cisco Systems, Inc. übernimmt keine Haftung für die Richtigkeit dieser Übersetzungen und empfiehlt, immer das englische Originaldokument (siehe bereitgestellter Link) heranzuziehen.