Introduction to WAN Optimization Traffic

*** Updated Session ***

Using Wireshark to assess the effectiveness of your WAN OPT features & deployment

John Pittle
Services CTO
Global Customer Experience
Riverbed Technologies
john.pittle@riverbed.com
@end2endViz
www.linkedin.com/in/john-pittle
Updated Session

- This is an update to the US v2020 session of same title
- Due to time constraints, we’re going to skip some of the background and intro material so we can get straight into Wireshark
- You can find the additional background and concepts in the US v20 session (Links on next slide)
Links to v20 US

• https://sharkfestvirtual.wireshark.org/sf20v

• https://www.youtube.com/watch?v=IyvlvmdbvZM
About me?

• SharkFest Instructor since 2017
• Practicing Performance Engineering since 1980
• Protocol Analysis since 1991
• Professional Services with OPNET / Riverbed since 2005
• Love the mystery of a complicated performance issue
• Shaved off beard in 2003...
Why this session...?

• WAN OPT technologies modify / enhance protocol behavior

• You will see protocol behavior in Wireshark that might look confusing / questionable

• The more background you have, the more effective you will be interpreting Wireshark to determine the benefits of your WAN OPT deployment
Why this session…?

- Some of this behavior is similar to other tunnelling and proxy technologies
- You will gain knowledge that will help you in a variety of special technology situations
Agenda

• Why WAN Optimization
• Overview of Features (Subset)
• Wireshark Capture & Analysis Examples
• Wrap-up with Q & A
IT’S GO TIME
Why WAN-OPT?
Benefits of WAN OPT

• Improve User Productivity

• Reduce WAN bandwidth usage
Benefits of WAN OPT

• Improve User Productivity

• Reduce WAN bandwidth usage

• Reduce Cloud Egress Costs
Concepts to Baseline / Level Set
Application Performance

• Application networking performance is primarily dependent on...
  • Latency – due to distance
  • End to End Network Health (Packet Loss / Protocol Effects)
  • Bandwidth - smallest link rate (physical or subscribed)
  • Congestion - busy devices, congested links, QoS Policies
Focus on Latency

Latency has a direct relationship with physics and distance.

**Network is soooo slow!**

2563 miles x 3way TCP handshake

**Wow! Network is fast!**

48 miles x 3way TCP handshake
Round Trip Time

- Time required to send packets between two hosts (request from A to B, followed by response from B back to A)

- Function of Latency + Congestion + Protocol Delay

- More / Faster Bandwidth will **not** improve latency
Related Wireshark Metrics

- `tcp.analysis.initial_rtt`
  - Time from SYN to SYN+ACK (plus ‘x’ factor)
  - Static value for the life of a connection
- `tcp.analysis.ack_rtt`
  - Time to ACK a particular segment
- `tcp.analysis.acks_frame`
  - The frame being acknowledged
### Sample from Decode Summary

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Delta Time</th>
<th>RTT</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>582621</td>
<td>863.00822</td>
<td>0.00000000</td>
<td>192.168.2.127</td>
<td>13.107.136.9</td>
<td>TCP</td>
<td>66</td>
<td>63442 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1</td>
<td></td>
</tr>
<tr>
<td>582622</td>
<td>863.089362</td>
<td>0.08003900</td>
<td>0.00000000</td>
<td>192.168.2.127</td>
<td>TCP</td>
<td>66</td>
<td>443 → 63442 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1</td>
<td></td>
</tr>
<tr>
<td>582623</td>
<td>863.089427</td>
<td>0.60006600</td>
<td>0.08940500</td>
<td>192.168.2.127</td>
<td>TCP</td>
<td>54</td>
<td>63442 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0</td>
<td></td>
</tr>
</tbody>
</table>
Q. What is the RTT2ACK for the Client Hello Message?
Q. What is the RTT2ACK for the Client Hello Message?
Q. What is the RTT2ACK for the Client Hello Message?
A. 83.7ms
Key Concept Ahead
Application Turn

- Request Message / Response Message Pair
  - Request Message can be multiple packets
  - Response Message can be multiple packets
- Request/Response Pair == 1 Turn
- Time duration is at least 1 RTT
- 10 Turns @ 100ms RTT >= 1 Second Duration
HTTP Example

User Input
San Rafael
Branch Office

HTTP Get

New York
Data Center

Turn

HTTP 200
With Response
SMB2 Example

User Input
San Rafael
Branch Office

SMB2 Read 16KB Block at file Offset 0

Turn

New York
Data Center

SMB2 Response
With the requested
data block
SQL Example

SQL: Select * from customer table

User Input
San Rafael
Branch Office

Status and rows from DB

Turn

New York
Data Center
One more useful metric

- Some decodes measure delta between request / response
What about TCP? (Transport)

• Is it sensitive to Latency?

• Does it have Turns too?
What about TCP?

• Is it sensitive to Latency?

• Does it have Turns too?

• Consider Congestion Window Mechanisms, Slow-Start, Delayed-ACK, Retransmits, etc.

• You pay a RTT Penalty for some of these
If you remember only one key point from this entire session...
Key Point

• Latency * Turn Rate == User Pain
Key Point

- Latency * Turn Rate == User Pain
- Reduce Turn Rate == Reduced User Pain
- Reduce Latency == Reduced User Pain
Review

- Application protocol inefficiencies
- Latency is the secret killer!
- Transport protocol chattiness
- Not enough bandwidth

You have to solve all three to see performance benefits
SteelHead Features Overview
Optimization Features

- Transport Optimization
  - TCP Proxy / ACK Spoofing
  - Intelligent Caching
  - Compression / Deduplication
  - WAN Connection Pooling
  - Overrides for sub-optimal TCP Options
  - Enhanced WAN Packet Loss Recovery Mechanisms
  - High Latency Detection / Optimizations
Optimization Features

- Application Protocol Specific Optimizations
  - Override sub-optimal settings / behavior
  - Pre-Fetch
  - Read Ahead / Immediate Write
  - Object Caching
- Policy and QoS based Traffic Shaping
- App Recognition to Drive Traffic Mgmt Policies
- Secure traffic between sites
Common Themes

✓ Reduce Turns
✓ Reduce Payload
✓ Reduce User Pain
Wireshark Analysis & Timing Samples
Scenarios

- Scenario #1 – Virtual Lab Environment
  - Enhanced Auto Discovery
  - Transport Optimization

- Scenario #2 - Client Accelerator (John’s Laptop)
  - Transport Optimization
  - Improve Response Times
Scenario #1

- Riverbed Training - Lab Environment
- Virtual Everything (SH, Client, Server, WAN, etc.)
- Explore Enhanced Auto Discovery & Transport Optimization
Transactions, all HTTP, of note (from cfe LAN perspective):

- Open 10.1.31.130
- Click public folder (srcprt: 51902)
- Click High-Res Images folder (srcprt: 51904)
- R-click > save as: wallpaper-1871712.png (srcprt: 51907)
- R-click > save as: wallpaper-1985738.png (srcprt: 51910)
- R-click > save as: wallpaper-1985738 (1).png (srcprt: 51915)
Four Capture Files

- TCPDumps controlled from the SteelHead Web UI

<table>
<thead>
<tr>
<th>Name</th>
<th>Date modified</th>
<th>Type</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>n120-sh1_lan0_0_ead.pcap</td>
<td>10/16/2020 6:12 PM</td>
<td>ACE Capture File</td>
<td>6,395 KB</td>
</tr>
<tr>
<td>n120-sh1_wan0_0_ead.pcap</td>
<td>10/16/2020 6:12 PM</td>
<td>ACE Capture File</td>
<td>4,837 KB</td>
</tr>
<tr>
<td>n130-sh1_lan0_0_ead.pcap</td>
<td>10/16/2020 6:12 PM</td>
<td>ACE Capture File</td>
<td>7,163 KB</td>
</tr>
<tr>
<td>n130-sh1_wan0_0_ead.pcap</td>
<td>10/16/2020 6:12 PM</td>
<td>ACE Capture File</td>
<td>4,863 KB</td>
</tr>
</tbody>
</table>
Purpose of Auto Discovery

• Initiated by Client side SteelHead
• Discover possible SteelHeads in the path that are closest to Server
• If an appropriate SH is discovered, then client SH will establish peering relationship if one does not already exist
• Transparent to both the client and the server end points
EAD - Bounce Chart

- Probe result is cached for 10 sec
- Connect result is cached until failure
- We are still using 0x4c but we now use two of them (back-to-back)
- Notification is being sent to SH1

Listening on Service Port 7800
Be on the “lookout”

- Decode Labels: SYN+, SYN++, SYN+*
- SYN-ACK Retransmission
- iRTT higher than expected
- TCP Options being modified
Journey of SYN
**SYN-LAN_0**

Client: 10.1.21.110

N120 – C-SH: 10.1.120.21

Web Server: 10.1.31.130

TCP Options:
- Maximum segment size: 1460 bytes
- No-Operation (NOP)
- Window scale: 8 (multiply by 256)
- No-Operation (NOP)
- No-Operation (NOP)
- SACK permitted
TCP Option - Maximum segment size: 1460 bytes
TCP Option - No-Operation (NOP)
TCP Option - Window scale: 8 (multiply by 256)
TCP Option - No-Operation (NOP)
TCP Option - No-Operation (NOP)
TCP Option - SACK permitted
TCP Option - Riverbed Probe: Probe Query, CSM IP: 10.1.120.21
TCP Option - Riverbed Probe: Probe Query Info
TCP Option - No-Operation (NOP)
TCP Option - End of Option List (EOL)
TCP Option - Maximum segment size: 1450 bytes
TCP Option - No-Operation (NOP)
TCP Option - Window scale: 8 (multiply by 256)
TCP Option - No-Operation (NOP)
TCP Option - No-Operation (NOP)
TCP Option - SACK permitted
TCP Option - Riverbed Probe: Probe Query, CSH IP: 10.1.120.21
TCP Option - No-Operation (NOP)
TCP Option - End of Option List (EOL)
<table>
<thead>
<tr>
<th>Time</th>
<th>Delta Time</th>
<th>RTT</th>
<th>RTTACK</th>
<th>ACK4</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>SACK CT</th>
<th>Seq</th>
<th>ACK</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>Delta Time</td>
<td>RTT</td>
<td>RTTACK</td>
<td>ACK4</td>
<td>Source</td>
<td>Destination</td>
<td>Protocol</td>
<td>Length</td>
<td>SACK CT</td>
<td>Seq</td>
<td>ACK</td>
<td>Info</td>
</tr>
</tbody>
</table>

**C-SH LAN**

**C-SH WAN**

**S-SH WAN**

**S-SH LAN**

#sf21veu • Online • June 14-18
Enhanced SYN Decodes

**Issue**

WireShark shows like this. What are they?

<table>
<thead>
<tr>
<th>No.</th>
<th>SourceIP</th>
<th>DestIP</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>54</td>
<td>12</td>
<td>10.0.0.3</td>
<td>S+</td>
</tr>
<tr>
<td>55</td>
<td>3</td>
<td>10.0.0.3</td>
<td>SA++</td>
</tr>
<tr>
<td>56</td>
<td>12</td>
<td>10.0.0.3</td>
<td>S*</td>
</tr>
<tr>
<td>57</td>
<td>12</td>
<td>10.0.0.3</td>
<td>S+*</td>
</tr>
<tr>
<td>58</td>
<td>3</td>
<td>10.0.0.3</td>
<td>SA+</td>
</tr>
</tbody>
</table>

**Solution**

These are SteelHead probe and probe response:

- **S+**: Probe
- **SA+**: Probe Response
- **S++**: Auto Peering (EAD, seen on MFE/SFE LAN)
- **SA++**: Probe Response (EAD)
- **S#**: Probe Trace, sent by Mobile Client if fixed target rule is defined.
- **S##**: Probe Trace, sent by Mobile Client if fixed target rule is not defined.
- **SA#**: Probe Trace response sent by CFE. Used when Mobile Client is installed.
- **SA##**: Probe Trace response sent by CFE. Used when Mobile Client is installed.
- **S-**: Cloud.
Questions you might have...

• Why does S-SH SYN to server have SYN+*?
• Why don’t we see any HTTP traffic on the WAN interface captures?
• Why did the S-SH change the Scaling Factor?
• Why did the S-SH introduce TCP Timestamps?
• Why is iRTT greater than expected latency?
Why iRTT can be higher...

- We are still using 0x4c but we now use two of them (back-to-back)
- Notification is being sent to SH1

Listening on Service Port 7800

20x
Why iRTT can be higher...

Because it includes the time needed to create the control and setup connection.
Journey of SYN+ACK
SYN+ACK (#1) - WAN_0 DC

Client

10.1.21.110

lan_0

N120 – C-SH
10.1.120.21

wan_0

N130 – S-SH
10.1.130.31

lan_0

Web Server

10.1.31.130

TCP Option - Riverbed Probe: Probe Query Info
TCP Option - No-Operation (NOP)
TCP Option - No-Operation (NOP)
TCP Option - No-Operation (NOP)
TCP Option - End of Option List (EOL)
SYN+ACK-LAN_0 DC

Client
10.1.21.110

N120 – C-SH
10.1.120.21

N130 – S-SH
10.1.130.31

Web Server
10.1.31.130

10.1.120.21

10.1.130.31

Client

Web Server

TCP Option - Maximum segment size: 1460 bytes
TCP Option - SACK permitted
TCP Option - Timestamps: T5val 5984269, T5secr 18699350
TCP Option - No-Operation (NOP)
TCP Option - Window scale: 7 (multiply by 128)
SYN+ACK (#2) - WAN_0 DC

Client
10.1.21.110

N120 – C-SH
10.1.120.21

10.1.120.21

wan_0

N130 – S-SH
10.1.130.31

10.1.130.31

lan_0

Web Server

10.1.31.130

TCP Option - Maximum segment size: 1460 bytes
TCP Option - SACK permitted
TCP Option - Timestamps: TSval 5984269, TSecr 18699350
TCP Option - No-Operation (NOP)
TCP Option - Window scale: 7 (multiply by 128)
SYN+ACK-LAN_0

Client

10.1.21.110

N120 – C-SH
10.1.120.21

lan_0

wan_0

N130 – S-SH
10.1.130.31

wan_0

lan_0

Web Server

10.1.31.130

10.1.21.110

10.1.120.21

10.1.130.31

TCP Option - Maximum segment size: 1460 bytes
TCP Option - No-Operation (NOP)
TCP Option - SACK permitted
TCP Option - No-Operation (NOP)
TCP Option - Window scale: 2 (multiply by 4)

TCP Option - Riverbed Probe: Probe Response, Server Steelhead: 10.1.130.31:7800
TCP Option - Riverbed Probe: Probe Response Info
TCP Option - No-Operation (NOP)
TCP Option - End of Option List (EOL)

TCP Option - Riverbed Probe: Probe Response, Server Steelhead: 10.1.130.31:7800
TCP Option - Riverbed Probe: Probe Response Info
TCP Option - No-Operation (NOP)
TCP Option - End of Option List (EOL)

[Expert Info (Note/Protocol): The SYN packet does not contain a MSS option]
Q++

• Why are there two SYN+ACKs on the WAN interface captures?
• Why did S-SH change the server’s scaling factor?
Discovery Complete

Time now to create Connection Pool for Inner Channel

- We are still using 0x4c but we now use two of them (back-to-back)
- Notification is being sent to SH1

Setup Information
28 Connections on Port 7800

<table>
<thead>
<tr>
<th>Address A</th>
<th>Port A</th>
<th>Address B</th>
<th>Port B</th>
<th>Packets</th>
<th>Bytes</th>
<th>Packets A - B</th>
<th>Bytes A - B</th>
<th>Packets B - A</th>
<th>Bytes B - A</th>
<th>Ref Start</th>
<th>Duration</th>
<th>Bits/s A - B</th>
<th>Bits/s B - A</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1.21.10</td>
<td>80</td>
<td>10.1.31.30</td>
<td>80</td>
<td>3</td>
<td>218</td>
<td>1</td>
<td>82</td>
<td>2</td>
<td>1</td>
<td>136</td>
<td>6.640348</td>
<td>0.0995</td>
<td>6590</td>
</tr>
<tr>
<td>10.1.21.11</td>
<td>80</td>
<td>10.1.31.30</td>
<td>80</td>
<td>3</td>
<td>218</td>
<td>1</td>
<td>82</td>
<td>2</td>
<td>1</td>
<td>136</td>
<td>6.669691</td>
<td>0.1038</td>
<td>6590</td>
</tr>
<tr>
<td>10.1.21.10</td>
<td>80</td>
<td>10.1.31.30</td>
<td>80</td>
<td>3</td>
<td>218</td>
<td>1</td>
<td>82</td>
<td>2</td>
<td>1</td>
<td>136</td>
<td>7.272921</td>
<td>0.1014</td>
<td>6548</td>
</tr>
<tr>
<td>10.1.21.11</td>
<td>80</td>
<td>10.1.31.30</td>
<td>80</td>
<td>3</td>
<td>218</td>
<td>1</td>
<td>82</td>
<td>2</td>
<td>1</td>
<td>136</td>
<td>21.226332</td>
<td>0.1002</td>
<td>6548</td>
</tr>
<tr>
<td>10.1.21.10</td>
<td>80</td>
<td>10.1.31.30</td>
<td>80</td>
<td>3</td>
<td>218</td>
<td>1</td>
<td>82</td>
<td>2</td>
<td>1</td>
<td>136</td>
<td>31.904544</td>
<td>0.0986</td>
<td>6535</td>
</tr>
<tr>
<td>10.1.21.11</td>
<td>80</td>
<td>10.1.31.30</td>
<td>80</td>
<td>3</td>
<td>218</td>
<td>1</td>
<td>82</td>
<td>2</td>
<td>1</td>
<td>136</td>
<td>67.879833</td>
<td>0.0973</td>
<td>6540</td>
</tr>
<tr>
<td>10.1.21.10</td>
<td>80</td>
<td>10.1.31.30</td>
<td>80</td>
<td>3</td>
<td>218</td>
<td>1</td>
<td>82</td>
<td>2</td>
<td>1</td>
<td>136</td>
<td>87.2034</td>
<td>0.0973</td>
<td>6540</td>
</tr>
<tr>
<td>10.1.21.11</td>
<td>80</td>
<td>10.1.31.30</td>
<td>80</td>
<td>3</td>
<td>218</td>
<td>1</td>
<td>82</td>
<td>2</td>
<td>1</td>
<td>136</td>
<td>93.061777</td>
<td>0.0911</td>
<td>6525</td>
</tr>
</tbody>
</table>

#sf21veu  •  Online  •  June 14-18
Transport Optimization

✓ Override TCP Options

✓ Connection Pooling

• ACK Spoofing
RTT2ACK is sub-ms vs. 400ms of iRTT

- RTT2ACK for GET in #11 is < 1 ms
- RTT2ACK for GET in #14 is < 1 ms
Scenario #2 – SaaS Accelerator
Help for my SaaS Apps?

Work from Anywhere

Internet
Microsoft Backbone

Service Instance

Service Cluster

Azure
Load Balancer

Coffee Shop
Client Accelerator

Hotel Room
Client Accelerator

Home Office
Client Accelerator

Office 365
box
salesforce
Veeva
ServiceNow
Scenario #2

- Client Accelerator on my laptop in Orlando
- SaaS Accelerator provisioned for Rvbd O365
- O365 Apps in the cloud (likely to be West Coast)
- Cloud SteelHeads running in a “Service Cluster” behind an Azure Load Balancer
Scenario #2 – Test Plan

- Test script planned out in advance
- Multiple copies of 56MB PPT test files with different file names
- Before and After Test Runs
- Packet Captures and Screen Video Captures
Scenario #2 - Actions

- Copy 56MB PPT from Desktop to local OneDrive via File Explorer, watch Synch
- Copy 56MB PPT to OneDrive via Browser
- Edit PPT on local OneDrive and watch Synch
- Edit PPT with SharePoint Online and watch Synch
- Capture packets for all steps
Why this activity was chosen

- Demonstrate time savings
- Caching based on byte patterns, not file names
- Measure optimization time savings for both local copy / edit, as well as browser-based copy / edit / synch
Test Script

SaaS Accelerator Test Plan

1. Enable WAN Opt
2. Start SH packet capture for 20 minutes
3. Ping 13.107.136.9
4. Copy ppt file #1 into test folder
5. Wait for synch
6. Ping 13.107.136.9
7. Copy ppt file #2 into test folder
8. Wait for synch
9. Ping 13.107.136.9
10. View folder online
11. Drag Copy #3 into test folder
12. Open local oneDrive folder
13. Wait for synch
14. Ping 13.107.136.9
15. Edit ppt #1, duplicate slide 2
16. Exit-save
17. Wait for synch
18. Ping 13.107.136.9
19. View ppt#2 online
20. Wait for it to fully open
21. ping
22. Duplicate slide 2
23. Wait for it to save
24. Ping
25. Close browser
26. Wait for file explorer to show synch'd
27. Ping
28. Open test folder in browser
29. Ping
30. Drag file #3 into the folder
31. Wait for upload to complete
32. Ping
33. Drag file #4 into the folder
34. Wait for the upload to complete
35. ping
36. Stop capture
1st Test - WAN OPT Disabled
Topology: WAN OPT Disabled

11Mbps – Upload
100Mbps – Download
Wifi 24G
Capture from WiFi Interface

Home Office
192.168.2.127

Internet

OneDrive
13.107.136.9

SharePoint
13.107.6.171

Office 365

13Mbps – Upload
100Mbps – Download
Evidence of RTT Cost

- Client ACK says “ready for stream bytes @ 7537”
- 498ms later that segment arrives
The cost of retransmissions
The cost of retransmissions +350ms of delay in this screenshot
Turning Optimization On

• Now we’ll run the script navigation with Optimization Turned on
OneDrive & SharePoint

Work from Home

Fluctuating Ping Latency
80ms – 130ms

Home Office

Client Accelerator

192.168.2.127

Internet

Microsoft Backbone

Service Instance

Azure

Load Balancer

13.107.136.9

Service Cluster

Fluctuating Ping Latency
80ms – 130ms

Internet

Microsoft Backbone

Service Instance

Azure

Load Balancer

13.107.136.9

Service Cluster

11Mbps – Upload
100Mbps – Download
Wifi 24G
Capture from WiFi Interface

Office 365

#sf21veu • Online • June 14-18
Be on the “lookout”

- Phantom TCP Connections in LAN_0 TCPDUMP
- RTT Timing evidence that incoming data is served from “nearby” cache
- No retransmissions
- Faster transfer times
With Client Accelerator Enabled

These “phantom” connections on port 63xxx occur only within the laptop. Part of internal WAN-Opt processing.
If you look closely, you’ll notice connection pairs that transfer roughly the same amount of traffic. Again, this is internal processing.

We’ll use display filter and “export specified packets…” to create a new pcap with tcp.port==443 only.
New PCAP with 443 Only

### Wireshark - Conversations: shm_1602706667-Jan_0_443Only.cap

<table>
<thead>
<tr>
<th>Ethernet 1</th>
<th>IPv4 2</th>
<th>IPv6</th>
<th>TCP</th>
<th>UDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address A</td>
<td>Port A</td>
<td>Port B</td>
<td>Packets</td>
<td>Bytes</td>
</tr>
<tr>
<td>192.168.2.127</td>
<td>63174</td>
<td>13.107.136.9</td>
<td>443</td>
<td>156,067</td>
</tr>
<tr>
<td>192.168.2.127</td>
<td>63072</td>
<td>13.107.136.9</td>
<td>443</td>
<td>159,819</td>
</tr>
<tr>
<td>192.168.2.127</td>
<td>63202</td>
<td>13.107.136.9</td>
<td>443</td>
<td>114,041</td>
</tr>
<tr>
<td>192.168.2.127</td>
<td>63442</td>
<td>13.107.136.9</td>
<td>443</td>
<td>65,907</td>
</tr>
<tr>
<td>192.168.2.127</td>
<td>63239</td>
<td>13.107.6.171</td>
<td>443</td>
<td>64,436</td>
</tr>
<tr>
<td>192.168.2.127</td>
<td>63114</td>
<td>13.107.136.9</td>
<td>443</td>
<td>64,203</td>
</tr>
<tr>
<td>192.168.2.127</td>
<td>62426</td>
<td>13.107.136.9</td>
<td>443</td>
<td>39,662</td>
</tr>
<tr>
<td>192.168.2.127</td>
<td>63077</td>
<td>13.107.136.9</td>
<td>443</td>
<td>1,117</td>
</tr>
</tbody>
</table>

#sf21veu • Online • June 14-18
SSL Handshake Acceleration

- TLS Server Hello arrives 15ms after Client Hello, but server iRTT == 89ms?
Segment from Local Cache

- Segment arrives immediately after ACK, no RTT delay from me to O365
RiOS: Scalable Data Reference (SDR)

Binary representation

1\textsuperscript{st} level references
- Ref[9z34]
- Ref[55k1]

2\textsuperscript{nd} level references
- Ref[vs5q6]

3\textsuperscript{rd} level references
- Ref[vv7a2]
- Ref[qk7j9]
- Ref[4u244]
- Ref[j8s]

16-Byte references communicate megabytes of existing data (128Byte average chunk size)
No Retransmissions

- Stream Bytes Served Locally
Significant Increase in User Productivity

- Over 30 hours since running initial script test
- Cache is still warm on laptop and in the cloud
- Video captures for upload and download scenarios
- Upload 16s vs. 64s
- Download 9s vs. 24s
Session Recap

- Expect your captures to contain some unusual side effects if WAN Optimization is in path
- Client accelerator running inside laptop provides significant user productivity improvements to support WFA employees
- Remediates latency, retransmissions, home WiFi issues
WAN Optimization Features

- Eliminate Application Latency
- De-Duplicate Data
- Optimize Network Transport
- Inspect, Report & Capture
- Shape, Direct & Protect
Your Feedback is Important

• Please take a moment to complete the session feedback form

• Help us to keep SharkFest relevant and interesting

• https://forms.gle/GGRAzkJcEuDkx5r36
End of Session