Rolf Leutert

Network Expert & Trainer | Leutert NetServices | Switzerland

Tuning Win7 Using Wireshark’s TCP Stream Graph
Case Study

- Customer is distributing Software over night to remote office in Asia
- But the process does not finish before local business hours starts
- Customer is paying for a WAN bandwidth of 45 Mbps
- He calculates an available throughput of only around 2 Mbps

- Does the bandwidth provider limit the rate?
- Is the server or the client not performing?

- Analyze the performance of a TCP session using TCP Stream graph
TCP Extension for High performance

- TCP was designed to operate in the range 100bps to 10Mbps and delays of 1ms to 100sec.
- The introduction of fiber optics is resulting in ever higher transmission speeds paths and are moving out of the domain for which TCP was originally engineered.
- TCP performance depends not upon the transfer rate itself, but rather upon the product of the transfer rate and the round-trip delay. If the bandwidth x delay product is large, TCP throughput will be limited.
- Internet path operating in this region are called "long, fat pipe", and a network containing this path as an "LFN" (pronounced "elephan(t)").
'Long - Fat - Pipe' Problems

- Maximum standard TCP window size is 65536 Bytes ($=2^{16}$)
'Long - Fat - Pipe' Problems

- High-capacity packet satellite channels are LFN's. Delay 4 x 35‘800 km = 470ms Round Trip Time
- Terrestrial fiber-optical paths will also fall into the LFN class
- There are three fundamental performance problems with the current TCP over LFN paths:
  - Window Size Limit (max 65k bytes) → Remedy: TCP option „Window scale“
  - Recovery from Segment Losses → Remedy: TCP option „Selective acknowledges“
  - Round-Trip Measurement → Remedy: TCP option „Time stamp“
TCP 'Window Scaling' Option

- TCP Window Size of 65,535 Bytes is too small.
- A multiplier Scaling Factor resolves this limitation.
- Scaling Factor $S$ is negotiated at TCP setup.
- Each end can offer an individual Scaling Factor.
- The value for the Scaling Factors can vary from 0 to 14.
- Calculation for the scaled Window Size is as follows:
  \[ \text{Scaled Window Size} = \text{Window Bytes} \times 2^S \]
- Example: Window Size 46 Bytes, Scaling Factor $S=7 \implies 2^7 = 128$
  \[ 46 \text{ Bytes} \times 128 = 5,888 \text{ Bytes} \]
- The maximum Window Size can be 1,073,741,824 Bytes = 1 Gigabyte
TCP 'Window Scaling' Option

Throughput without Window Scaling:
Line Speed 10Mbit/sec with Round trip time 200ms

Max. Throughput = \(\frac{\text{TCP Window Size}}{\text{Round-trip time}}\)

Maximum data throughput with standard TCP window size of 64kBytes is = 2,5Mbit/s only

Idle time

Data

Ack

Window
TCP 'Window Scaling' Option

Scaling Factor Calculation:

\[
\frac{\text{Bandwidth} \times \text{Round Trip Time}}{\text{TCP Window Size}} = \text{Factor 4}
\]

Throughput with Window Scaling:

10Mbit/sec

Window x 4

Data

Ack

~500’000bits

© 2012 Leutert NetServices
After the two TCP SYN frames, the window size is announced in the scaled format and Wireshark displays the scaled value.
TCP Extensions for High Performance

The following TCP options are defined in RFC1323:

- 01 No operation (for padding)
- 02 Max. Window size (SYN)
- 03 Window scale (SYN)
- 04 SACK permitted (SYN)
- 05 SACK option (Acknowledges)
- 08 Time stamp (SYN and Acknowledges)
TCP ‘Selective Acknowledge’ Option

- The usage of the **TCP SACK option** is negotiated during the 3-Way hand shake.
- The SACK option can be activated from one or both sides.
- Without SACK option, only the last received segment of a contiguous series can be acknowledged.
- The SACK Option allows to **acknowledge non-contiguous** segments of a series and can request for specific segments.
- The SACK Option can **improve** the throughput of LFN’s significantly.
TCP ‘Selective Acknowledge’ Option

Trace file: TCP SACK

- Seq. # 1
- Ack. # 2621
- SACK Block 1

- Packet A: 1310B
- Packet B: 1310B
- Packet C: 1310B
- Packet D: 1310B
- Packet E: 1310B
- Packet F: 1310B
- Packet G: 1310B
- Packet C, D missing
- Packet F missing
- Packets C, D delivered
- Packet F delivered
- Data Packets
- Length 1310B

TCP Selective Acknowledges

- Frame 1: 1310B
- Frame 2: 1310B
- Frame 3: 1310B
- Frame 4: 1243B
- Frame 5: 1310B
- Frame 6: 1310B
- Frame 7: 1243B
- Frame 8: 1310B
- Frame 9: 1310B
- Frame 10: 1310B
- Frame 11: 1310B
- Frame 12: 1310B
TCP Analysis with Wireshark Expert

• TCP performance can be influenced by these three main components
• The Wireshark Expert is offering great support in analyzing TCP sessions
• Understanding TCP and Expert Messages helps isolating problems
TCP Analysis with Wireshark Expert

• The Wireshark Expert System recognizes many abnormalities or errors and creates a list sorted by severities:
  • Segment Lost
  • Duplicate ACK
  • Retransmissions
  • Fast Retransmissions
  • Zero Window
  • Window Full
  • and many more…

• You still need well-founded TCP knowledge to understand the error messages and to draw the right conclusions.
TCP Analysis with Wireshark Expert

- Click on the colored 'Expert Button' to open the 'Expert Infos' window

**Level 0** = No Expert info available for protocols present in trace file (i.e. for protocols using UDP)

**Level 1** = Chats: Information about normal data flow, e.g. TCP session establishment and closing. HTTP Get/OK/404 etc.

**Level 2** = Notes: Reference to slight abnormalities like 'Duplicate ACK', 'Retransmissions' etc.

**Level 3** = Warnings: Informs about abnormalities like 'Segment lost', 'Segments out of order' etc.

**Level 4** = Errors: Messages on serious problems like deformed segments (i.e. missing fields)
TCP Analysis with TCP Stream Graph

- Sometimes, a graphic tells us more than a thousand frames
- Wireshark offers excellent graphical TCP session presentations
- **TCP Stream Graph** allows to recognize all the following abnormalities:
  - Lost Frames
  - Duplicate Frames
  - Out of order Frames
  - TCP Sequence number and Segment Sizes
  - Acknowledges, Delayed Acknowledges
  - Duplicate and Selective Acknowledges
  - Retransmissions and Fast Retransmissions
  - Windows Sizes, sliding Window, exceeded und frozen Windows Size
  - Window Scaling, Zero Window and Window Full Situation
  - Slow Start, full Flow rate and Flow throttling
TCP Analysis with TCP Stream Graph

Notice direction of TCP Half-Session

Press F1 für Navigation Help

Server → Client

Window Size →

Data Frames →

Acknowledges
TCP Analysis with TCP Stream Graph

- Now, let us analyze our customer case using Frame Analysis
TCP Analysis with TCP Stream Graph

- Now, let us analyze our customer case using TCP Stream Graph
What can be read out of the trace file and the TCP Stream graph:

- Client and Server are both using **Window Scaling and Selective ACKs**
- The trace file has been captured on the **server side**
- The Round-Trip-Time is **186ms**
- The receiver (Client) window is **wide open**
- The network is **dropping** frames
- The server is **retransmitting** frames

- At this stage, we can **exclude the client**!
TCP Analysis with TCP Stream Graph

• TCP 'Three-way Handshake'

**Client SYN**
• Start Sequence Number
• Window Size

**Options:**
• Maximum Segment Size
• Window Scaling
• Selective Acknowledges
• Timestamp
• PAWS (Protection against wrapped sequence #)

**Client ACK**
• Acknowledge Server Sequence Number

**Server SYN; ACK**
• Start Sequence Number
• Acknowledge Client Sequence Number
• Window Size

**Options:**
• Maximum Segment Size
• Window Scaling
• Selective Acknowledges
• Timestamp
• PAWS (Protection against wrapped sequence #)
TCP Analysis with TCP Stream Graph

- Let us have a closer look at the servers behavior!
What can be read out of the TCP Stream graph:

- The server is starting with **15Mbit/s** transmission rate
- The network is **dropping** some frames (pretty normal on WAN)
- Server is throttling down to **1.5 Mbit/s**
- Server is **not** trying to speed up again
- But why?
- At this stage we can **exclude the network**
Microsoft has implemented new autotuning in Vista, Win7, Server2008

- These features should improve TCP throughput and are ON by default
- However, this is not always the case, and may cause some Internet related issues and problems!
## MS Windows TCP Autotuning Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Activate</th>
<th>Deactivate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autotuning</td>
<td>netsh interface tcp set global autotuning=normal</td>
<td>netsh interface tcp set global autotuning=disabled</td>
</tr>
<tr>
<td>Compound TCP</td>
<td>netsh interface tcp set global congestionprovider=ctcp</td>
<td>netsh interface tcp set global congestionprovider=none</td>
</tr>
<tr>
<td>ECN Support</td>
<td>netsh interface tcp set global ecncapability=enabled</td>
<td>netsh interface tcp set global ecncapability=disabled</td>
</tr>
<tr>
<td>TCP Chimney offloading</td>
<td>netsh interface tcp set global chimney=enabled</td>
<td>netsh interface tcp set global chimney=disabled</td>
</tr>
<tr>
<td>Receive-side Scaling (RSS)</td>
<td>netsh interface tcp set global rss=enabled</td>
<td>netsh interface tcp set global rss=disabled</td>
</tr>
</tbody>
</table>

This command did **solve the issue** in our case:

- Windows Scaling heuristics  
  - Deactivate: netsh int tcp set heuristics disabled  
  - Activate: netsh int tcp set heuristics enabled
TCP Analysis with TCP Stream Graph

• Now let us have a closer look at the servers behavior again!

24 Mbit/s → Problem solved!
Thanks for visiting

Hope you learned something useful

Rolf Leutert, Leutert NetServices, www.wireshark.ch