SHARKFEST '12

Wireshark Developer and User Conference

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?

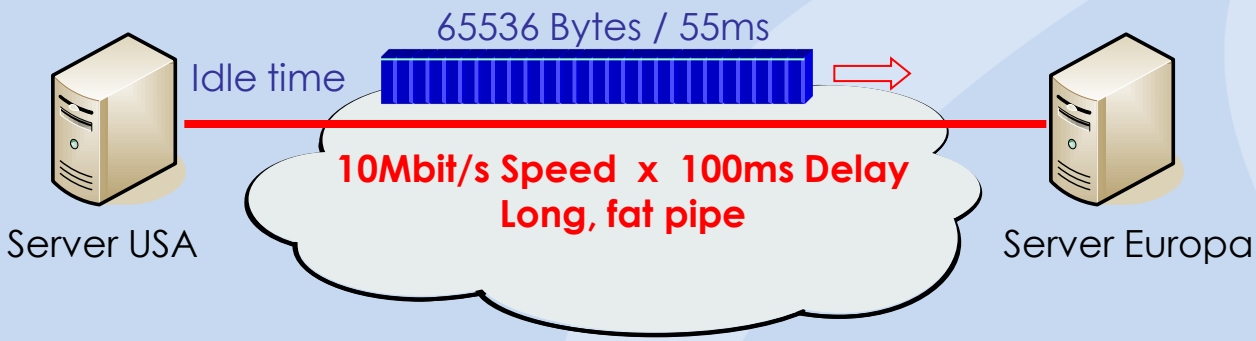
- Analyze the performance of a TCP session using TCP Stream graph

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 Skaling 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 to14.
- Calculation for the scaled Window Size is as follows:

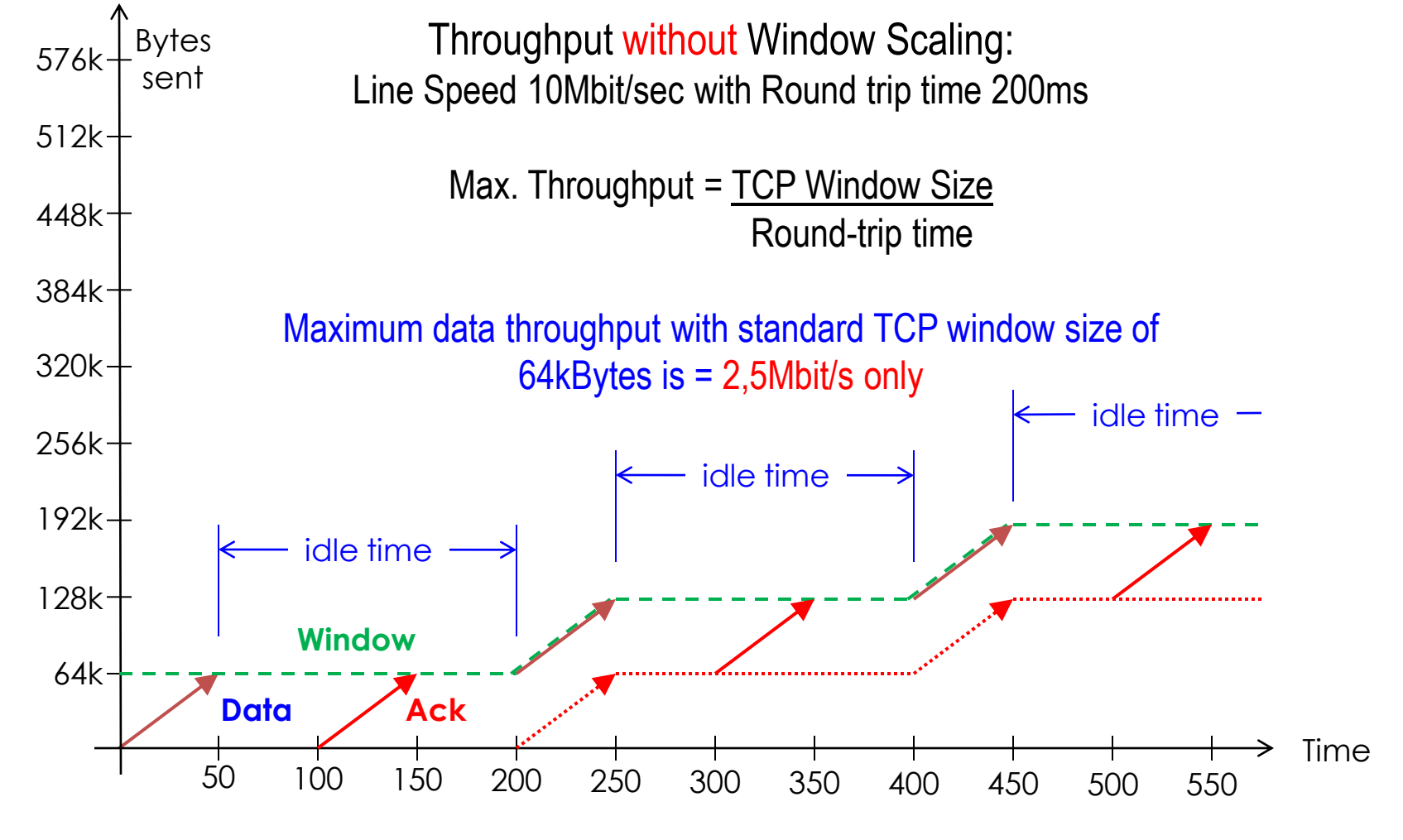
$$\text{Scaled Window Size} = \text{Window Bytes} \times 2^S$$

- Example: Window Size 46 Bytes, Scaling Factor S=7→ $2^7 = 128$

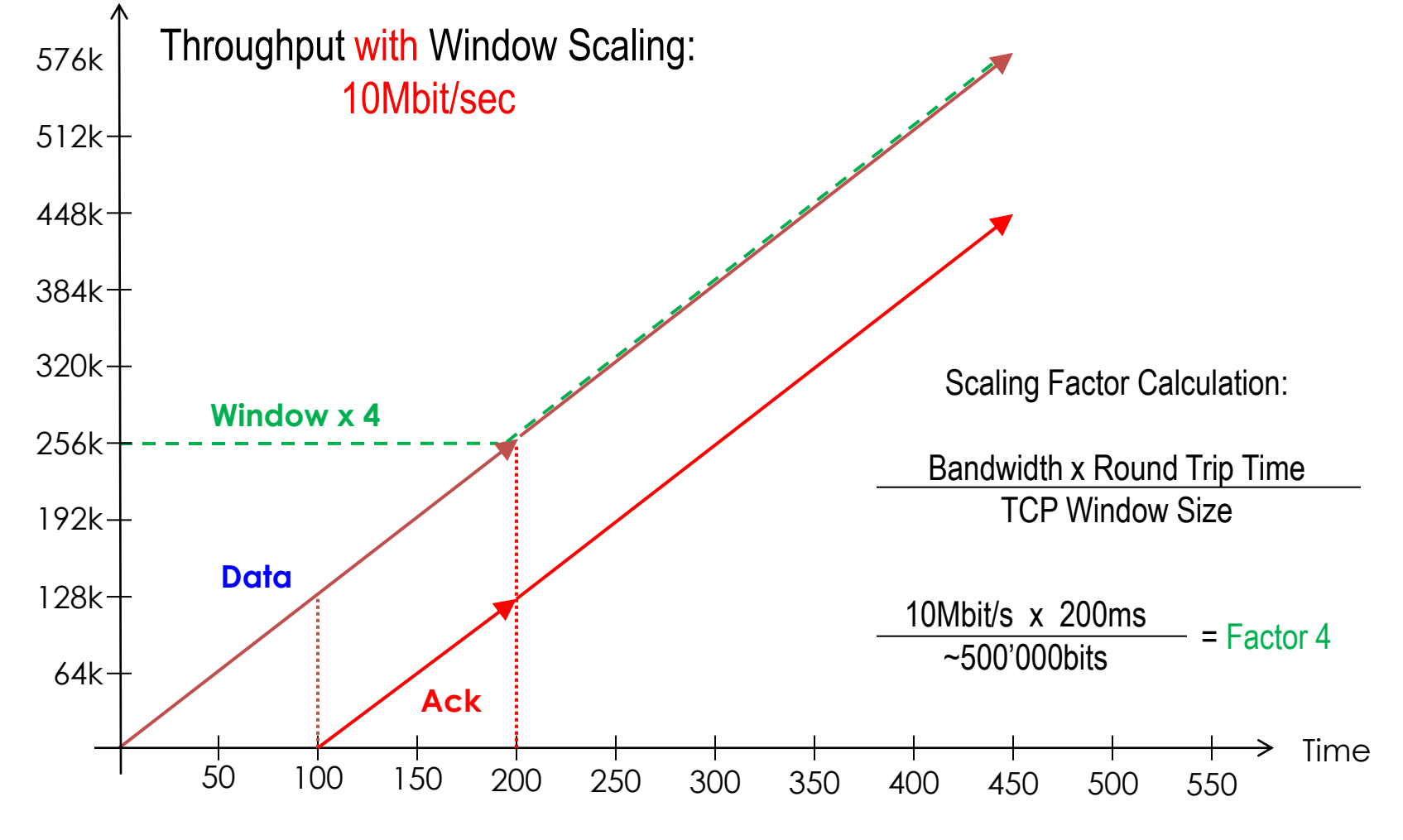
$$46 \text{ Bytes} \times 128 = 5'888 \text{ Bytes}$$

- The maximum Window Size can be 1’073’741’824 Bytes = 1 Gigabyte

TCP ‚Window Scaling' Option



TCP ‚Window Scaling' Option



TCP ‚Window Scaling‘ Option

‚Window Scaling‘ factor from Client

TCP Sessions 01.pcap [Wireshark 1.6.4 (SVN Rev 39941 from /trunk-1.6)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: tcp.stream eq 99 Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
3367	66.349750	192.168.0.203	69.4.231.52	TCP	54889 > http [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
3368	66.459624	69.4.231.52	192.168.0.203	TCP	http > 54889 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1360 SACK_PERM=1 WS=512
3369	66.459716	192.168.0.203	69.4.231.52	TCP	54889 > http [ACK] Seq=1 Ack=1 Win=66640 Len=0
3370	66.460087	192.168.0.203	69.4.231.52	HTTP	GET /wireshark-win32/wireshark-win32-1.2.3.exe HTTP/1.1
3371	66.580604	69.4.231.52	192.168.0.203	TCP	http > [ACK] Seq=1 Ack=825 Win=7680 Len=0
3372	66.583870	69.4.231.52	192.168.0.203	TCP	[TCP segment of a reassembled PDU]
3373	66.586377	69.4.231.52	192.168.0.203	TCP	[TCP segment of a reassembled PDU]
3374	66.586424	192.168.0.203	[illegible]		[illegible] > http [ACK] Seq=825 Ack=2721 Win=66640 Len=0
3375	66.697221	69.4.[illegible]	[illegible]		[illegible] segment of a reassembled PDU]

```
Frame 3370: 878 b[illegible] (7024 bi[illegible]
Ethernet II, Src[illegible] D-Link[illegible]:3e (00:11:95:b7:e0:3e)
Internet Protocol [illegible] 03), [illegible].4.231.52 (69.4.231.52)
Transmission Control [illegible] Dst Port[illegible] (80), Seq: 1, Ack: 1[illegible]
  Source port: 54889 (54889)
  Destination port: http (80)
  [Stream index: 99]
  Sequence number: 1    (relative se[illegible]
  [Next sequence number: 825    ([illegible]
  Acknowledgement number: 1    (re[illegible]
  Header length: 20 bytes
  Flags: 0x18 (PSH, ACK)
  Window size value: 16660
  [Calculated window size: 66640]
  [Window size scaling factor: 4]
  Checksum: 0xf0fe [validation disabled]
```

‚Window Size‘ unscaled

‚Window Size‘ scaled

‚Window Scaling‘ factor from Server

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)

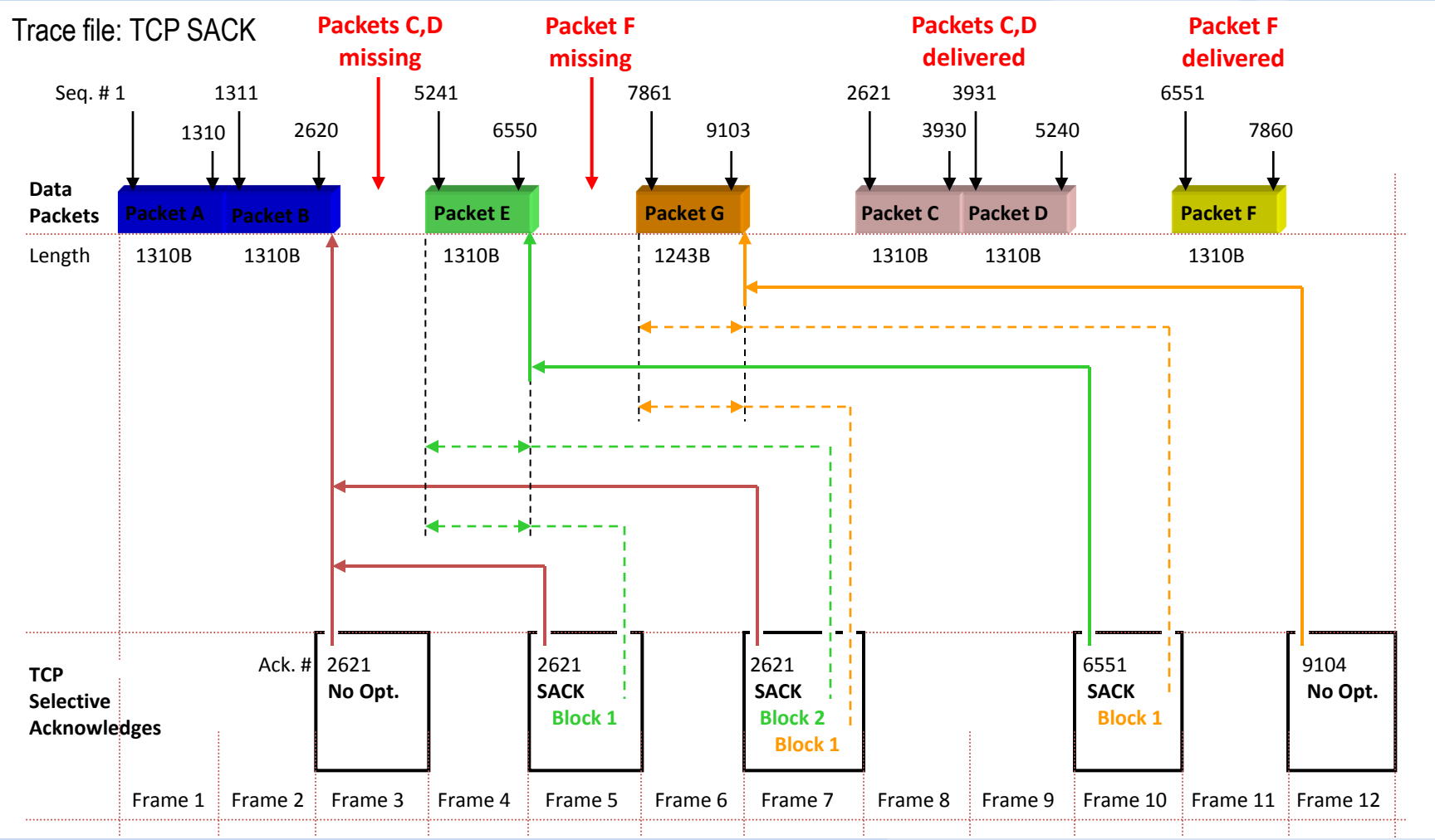
```
⊟ Options: (24 bytes)
    Maximum segment size: 1460 bytes
    NOP
    Window scale: 2 (multiply by 4)
    NOP
    NOP
    Timestamps: TSval 0, TSecr 0
    NOP
    NOP
    SACK permitted

0000  00 90 27 96 a9 2e 00 00  e8 20 20 58 08 00 45 00   ..'..... .  X..E.
0010  00 40 00 79 40 00 80 06  77 fe c0 a8 00 69 c0 a8   .@.y@... w....i..
0020  00 87 04 10 00 8b aa 5a  20 00 00 00 00 00 b0 02   .......Z  .......
0030  eb c0 f9 07 00 00 02 04  05 b4 01 03 03 02 01 01   ........ ........
0040  08 0a 00 00 00 00 00 00  00 00 01 01 04 02         ........ ......
```

TCP 'Selective Acknowledge' Option

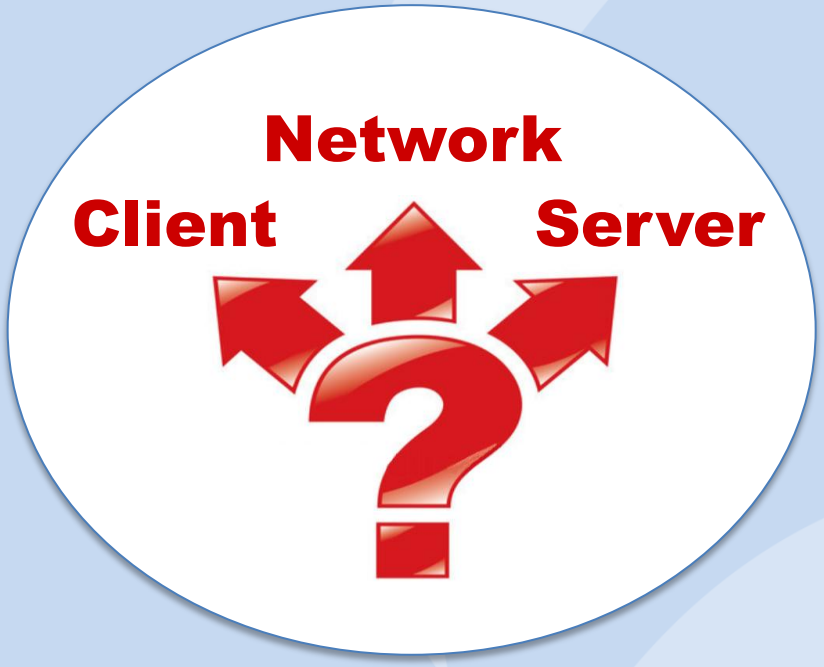
- The usage of the TCP SACK option is negotiated during the 3-Way hand shake.
- The SACK option can be activated from one or both sides.
- Without SACK option, only the last received segment of a contiguous series can be acknowledged.
- The SACK Option allows to acknowledge non-contiguous segments of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

TCP 'Selective Acknowledge' Option



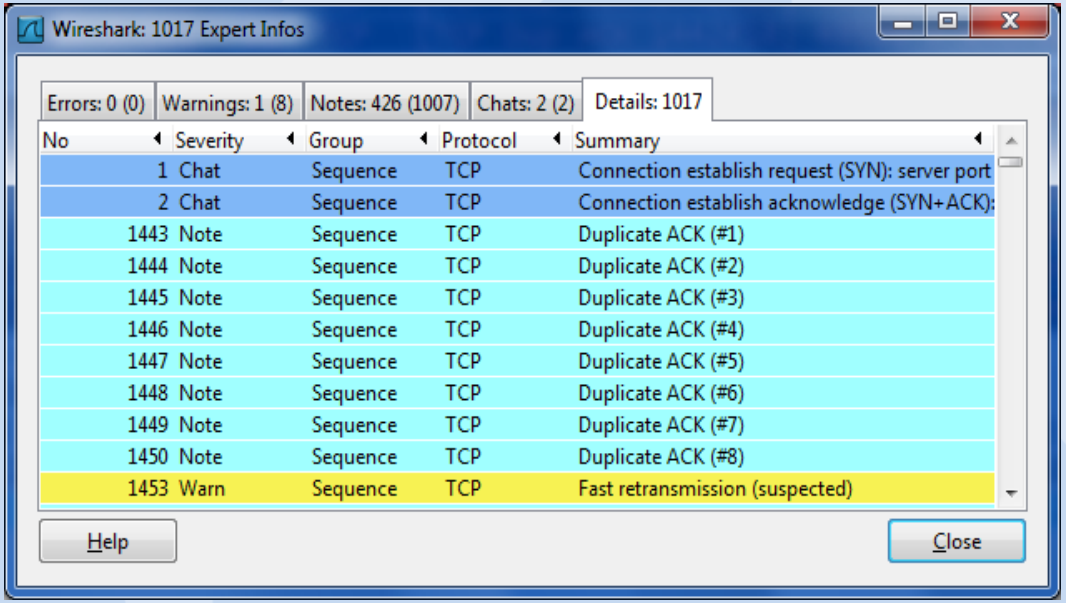
TCP Analysis with Wireshark Expert

- 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…

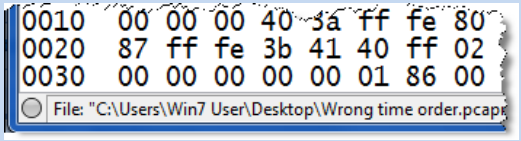


No	Severity	Group	Protocol	Summary
1	Chat	Sequence	TCP	Connection establish request (SYN): server port
2	Chat	Sequence	TCP	Connection establish acknowledge (SYN+ACK):
1443	Note	Sequence	TCP	Duplicate ACK (#1)
1444	Note	Sequence	TCP	Duplicate ACK (#2)
1445	Note	Sequence	TCP	Duplicate ACK (#3)
1446	Note	Sequence	TCP	Duplicate ACK (#4)
1447	Note	Sequence	TCP	Duplicate ACK (#5)
1448	Note	Sequence	TCP	Duplicate ACK (#6)
1449	Note	Sequence	TCP	Duplicate ACK (#7)
1450	Note	Sequence	TCP	Duplicate ACK (#8)
1453	Warn	Sequence	TCP	Fast retransmission (suspected)

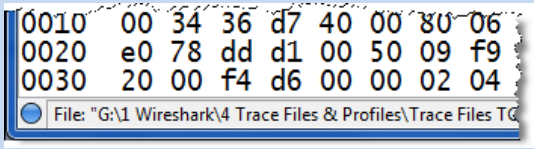
- You still need well-founded TCP knowledge to understand the error messages and to draw the right conclusions.

TCP Analysis with Wireshark Expert

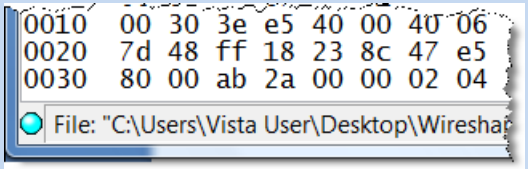
- Click on the colored ‚Expert Button‘ to open the ‚Expert Infos‘ window



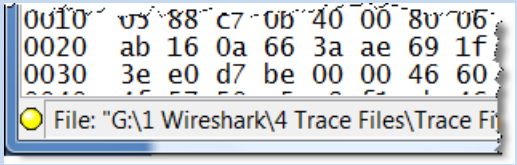
Level 0 = No Expert info available for protocols present in trace file (i.e. for protocols using UDP)



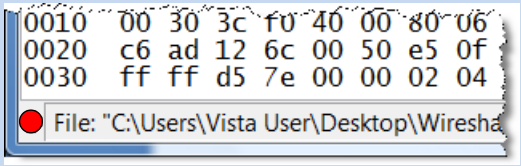
Level 1 = Chats: Information about normal data flow, e.g. TCP session establishment and closing. HTTP Get/OK/404 etc.



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.

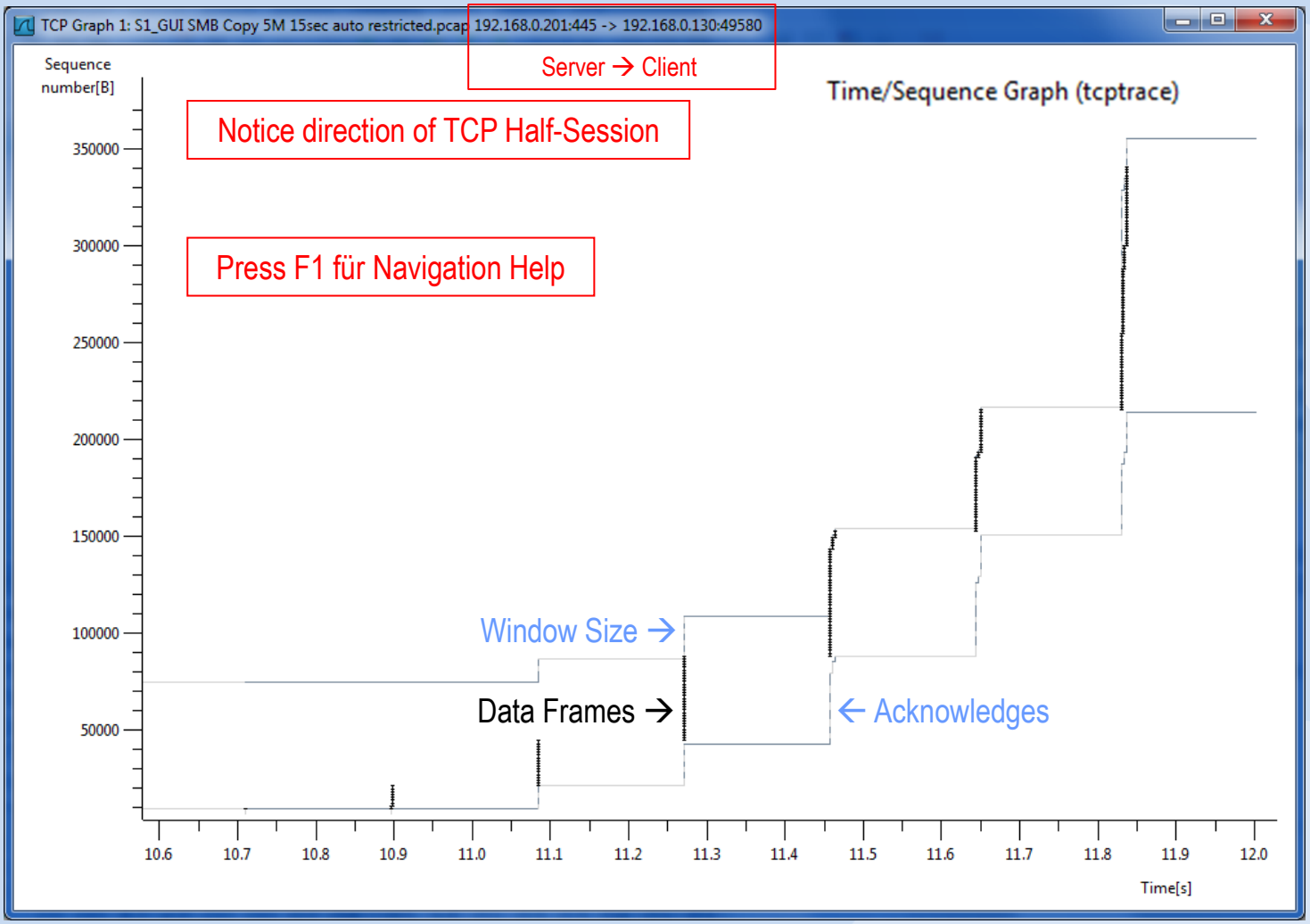


Level 4 = Errors: Messages on serious problems like deformed segments (i.e. missing fields)

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



TCP Analysis with TCP Stream Graph

- Now, let us analyze our customer case using Frame Analysis

S1_GUI SMB Copy 5M 15sec auto restricted.pcap [Wireshark 1.6.8 (SVN Rev 42761 from /trunk-1.6)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

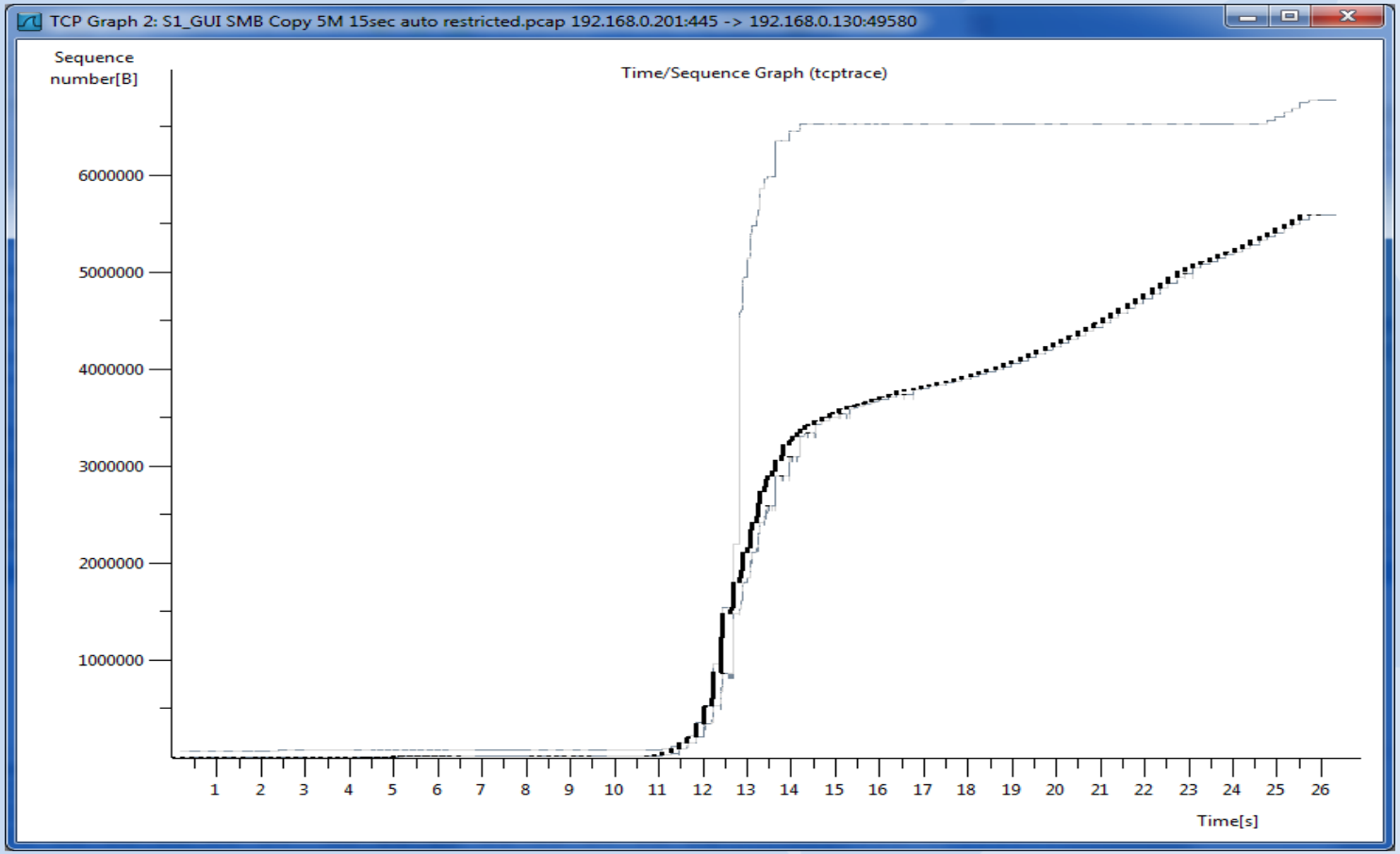
Filter: Expression... Clear Apply

No.	Time	Source	Destination	Length	Protocol	Info
1	0.000000	Client	Server	66	TCP	49580 > microsoft-ds [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
2	0.000334	Server	Client	66	TCP	microsoft-ds > 49580 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1464 WS=16 SACK_PERM
3	0.186347	Client	Server	60	TCP	49580 > microsoft-ds [ACK] Seq=1 Ack=1 Win=65536 Len=0
4	0.000005	Client	Server	213	SMB	Negotiate Protocol Request
5	0.029779	Server	Client	506	SMB2	NegotiateProtocol Response
6	0.186723	Client	Server	162	SMB2	NegotiateProtocol Request
7	0.029456	Server	Client	506	SMB2	NegotiateProtocol Response
8	0.184073	Client	Server	220	SMB2	SessionSetup Request, NTLMSSP_NEGOTIATE
9	0.000711	Server	Client	359	SMB2	SessionSetup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALLENGE
10	0.185778	Client	Server	663	SMB2	SessionSetup Request, NTLMSSP_AUTH, User: WIN7-USER-PC\test, Unknown message type
11	0.000918	Server	Client	159	SMB2	SessionSetup Response, Unknown message type
12	0.185759	Client	Server	170	SMB2	TreeConnect Request Tree: \\192.168.0.201\IPC$
13	0.000321	Server	Client	138	SMB2	TreeConnect Response

```
⊞ Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
⊞ Ethernet II, Src: Wistron_c0:66:fd (00:0a:e4:c0:66:fd), Dst: QuantaCo_6d:6c:e0 (00:23:8b:6d:6c:e0)
⊞ Internet Protocol Version 4, Src: Client (192.168.0.130), Dst: Server (192.168.0.201)
⊞ Transmission Control Protocol, Src Port: 49580 (49580), Dst Port: microsoft-ds (445), Seq: 0, Len: 0
```

TCP Analysis with TCP Stream Graph

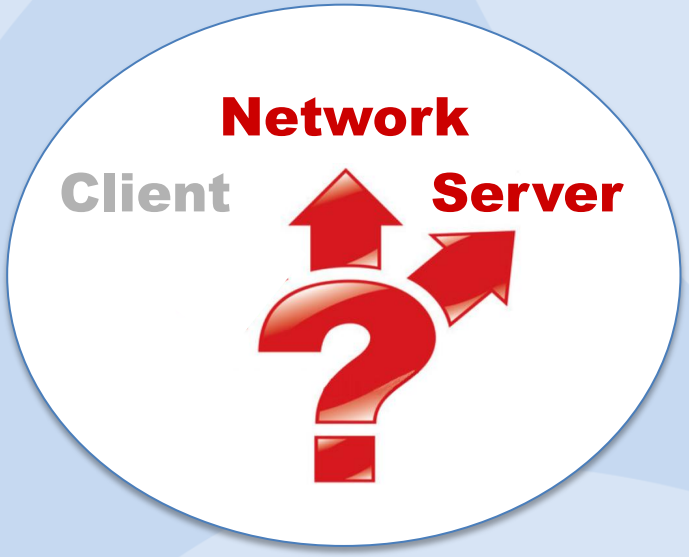
- Now, let us analyze our customer case using TCP Stream Graph



TCP Analysis with 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 !

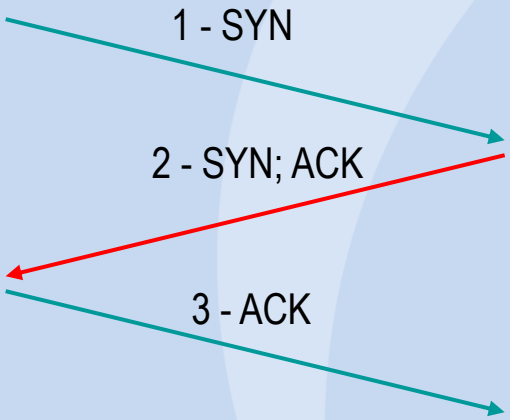
TCP Analysis with TCP Stream Graph

- 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

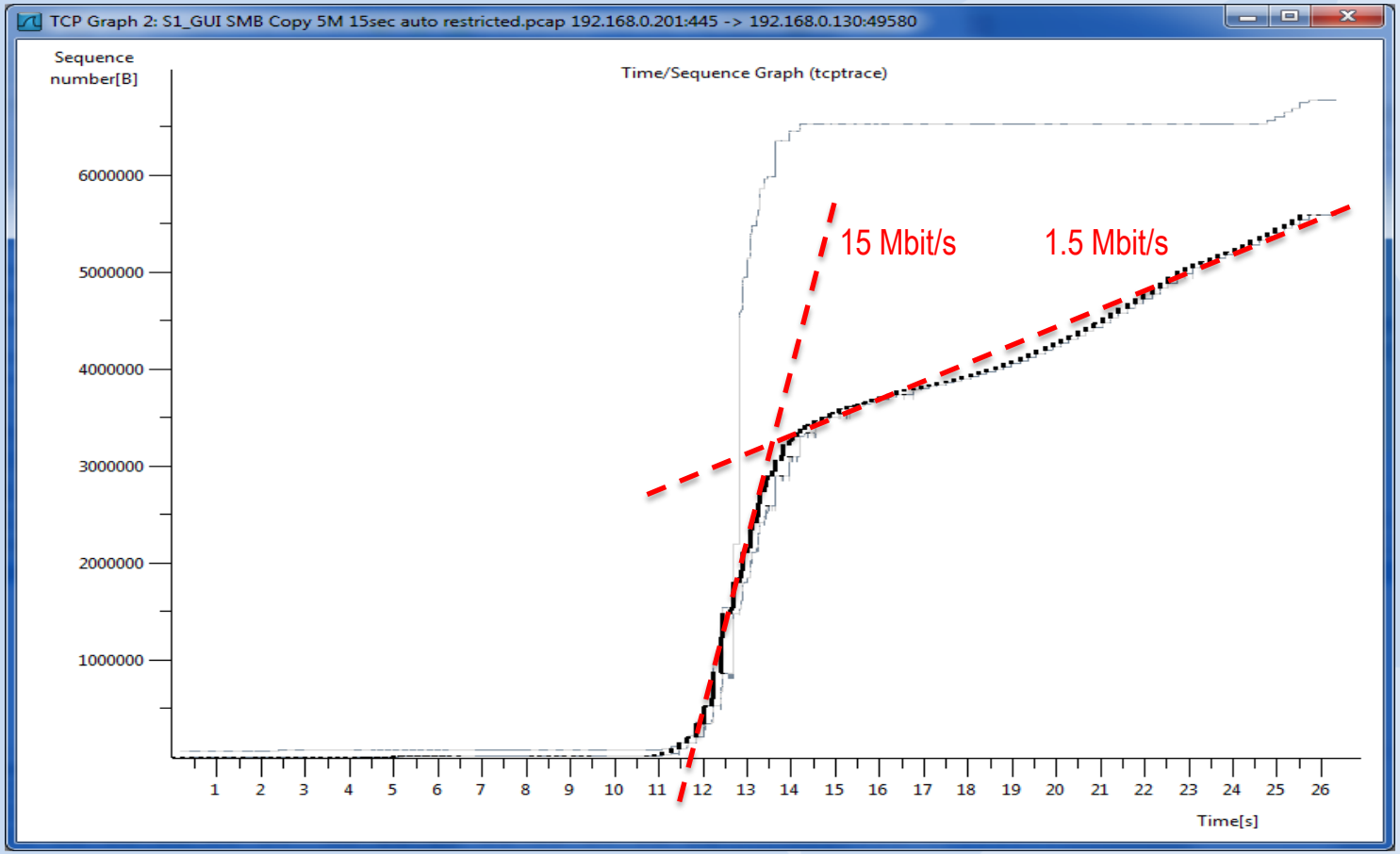


Server SYN; ACK
- Start Sequence Number
- Acknowledge Client Sequence Number
- Window Size

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!

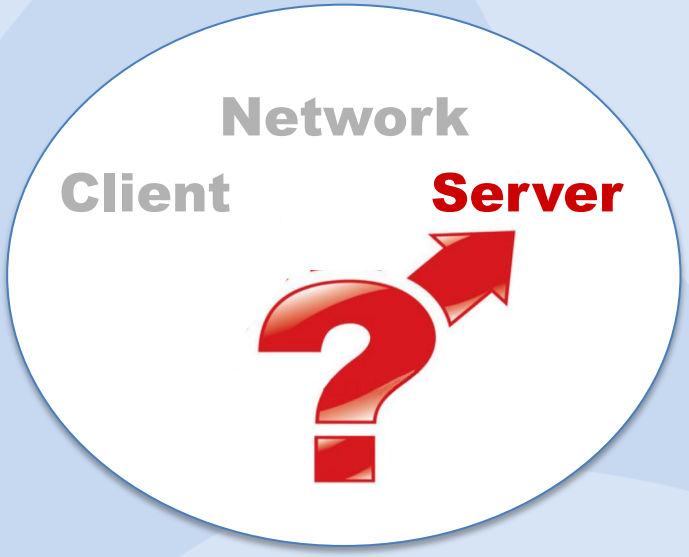


TCP Analysis with TCP Stream Graph

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



MS Windows TCP Autotuning Features

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 !

Administrator: Eingabeaufforderung

```
C:\Windows\system32>netsh interface tcp show global
Querying active state…

TCP Global Parameters
----------------------------------
Receive-Side Scaling State : enabled
Chimney Offload State : enabled
Receive Window Auto-Tuning Level : normal
Add-On Congestion Control Provider : none
ECN Capability : disabled
RFC 1323 Timestamps : disabled
```

MS Windows TCP Autotuning Features

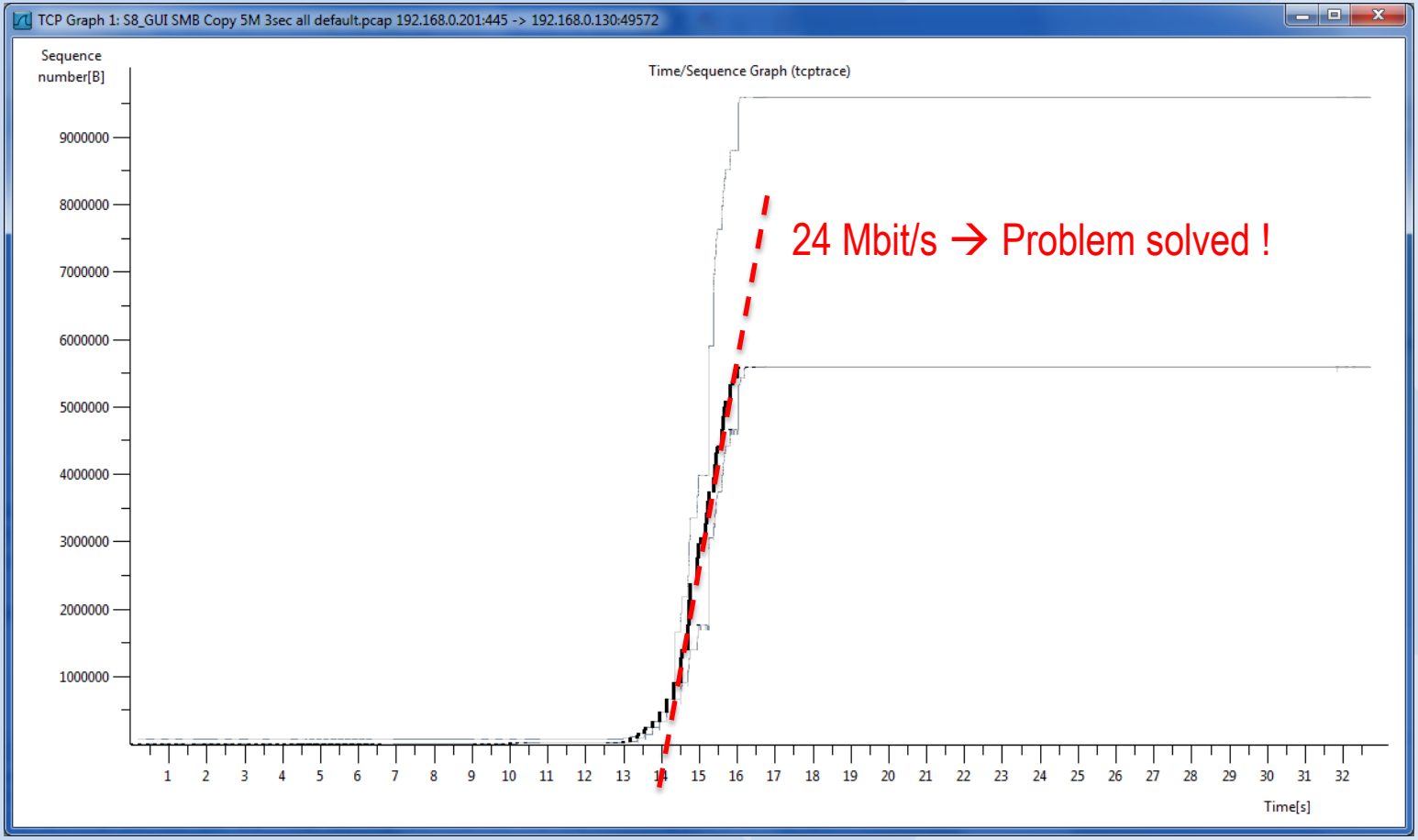
• Autotuning	→ Activate:	netsh interface tcp set global autotuning=normal
	→ Deactivate:	netsh interface tcp set global autotuning=disabled
• Compound TCP	→ Activate:	netsh interface tcp set global congestionprovider=ctcp
	→ Deactivate:	netsh interface tcp set global congestionprovider=none
• ECN Support	→ Activate:	netsh interface tcp set global ecncapability=enabled
	→ Deactivate:	netsh interface tcp set global ecncapability=disabled
• TCP Chimney offloading	→ Activate:	netsh interface tcp set global chimney=enabled
	→ Deactivate:	netsh interface tcp set global chimney=disabled
• Receive-side Scaling (RSS)	→ Activate:	netsh interface tcp set global rss=enabled
	→ Deactivate:	netsh interface tcp set global rss=disabled

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!



Thanks for visiting

Rolf Leutert, Leutert NetServices, www.wireshark.ch