#sf21veu

Chasing application performance with Wireshark

Analyzing Database Applications with Wireshark

Matthias Kaiser
ExperTeach GmbH,
Germany

#sf21veu

Hello!

I am Matthias Kaiser

I am here because I love packet analysis with Wireshark and I love to present

You can find me at Twitter: @wiresharky

You can contact me via Email: matthias.kaiser@experteach.de

About Myself

- Sniffer University Instructor at Network General /NAI
- Freelancer with own analysis courses
- Trainer and Consultant at ExperTeach
  - Wireshark Training and more
  - Consulting Services for Packet Analysis
- Motto:
  - „Every trace hides a story. Uncover and tell it.“

Files and Downloads

- Presentation covers Real-Life Cases
- Trace Files and Wireshark Profiles:
  - https://tinyurl.com/8nmc59c2
- Trace Files have been anonymized and sanitized with TraceWrangler, © by Jasper Bongertz

#sf21veu

Agenda

- Database applications
- Before we start…
- Sample Database Flow
- Case Study 1
- Case Study 3
- Application metrics for Wireshark
- Lessons learned
- Q&A

Database Applications

- Applications drive processes …. Everywhere
- Database applications are all over the place.
  - E-Commerce
  - ERP, like Warehousing or Finance and HR
  - Automation
  - …..
- All applications will be IP-based
- Software Defined Networking
  - Controllers will tell servers and network, what to do.
- So, we better understand, how applications work … in order to analyze them with Wireshark

Before we start … looking at packets

Have a plan

- Set your goals for the analysis.
- Describe your problem.
- Find out who is affected?
  - Locations, Users, entire PCs, just applications
  - Check the severity of your problem
- Identify the application(s)
- Find out when the problem occurs
  - Permanent
  - Sporadic / intermittent
- Do not just capture some traffic!
- Do not just look at trace files!
- And please … stop guessing!

Before we start… II

Capture

- What are the traffic flows for your application?
- Capture Location: Where do I see interesting traffic?
- Define the user activity to be analysed.
  - Permanent problem: Pick one typical action
  - Intermitting problem: Long-Term analysis

Analyze

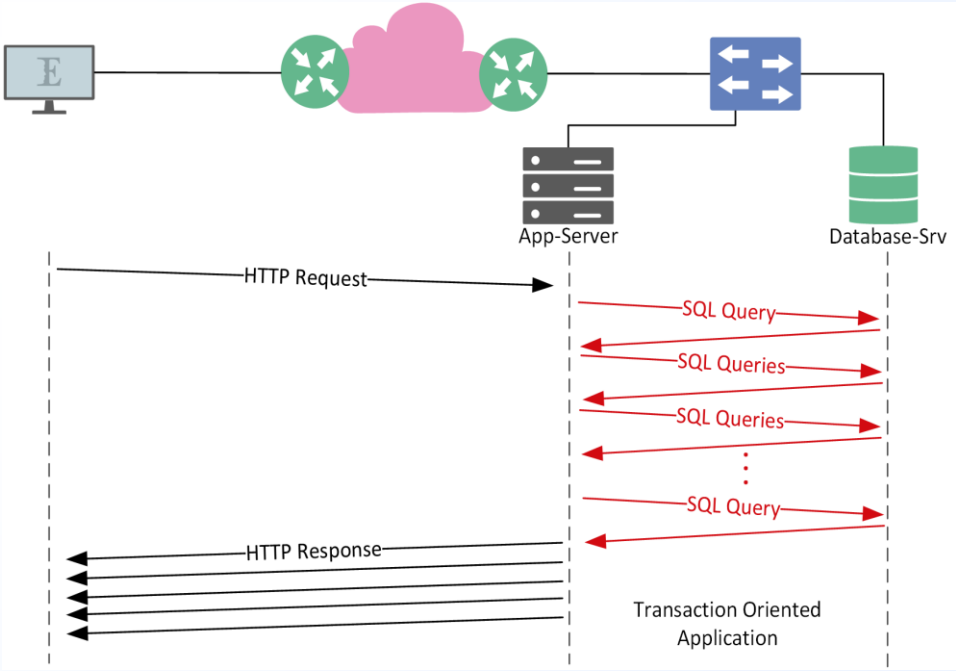
- Prepare your Wireshark (Profiles)
- Filter your trace file
  - IP addresses, Ports
- Identify traffic for User actions
- Know the key metrics for the application?

And then

- Do the analysis

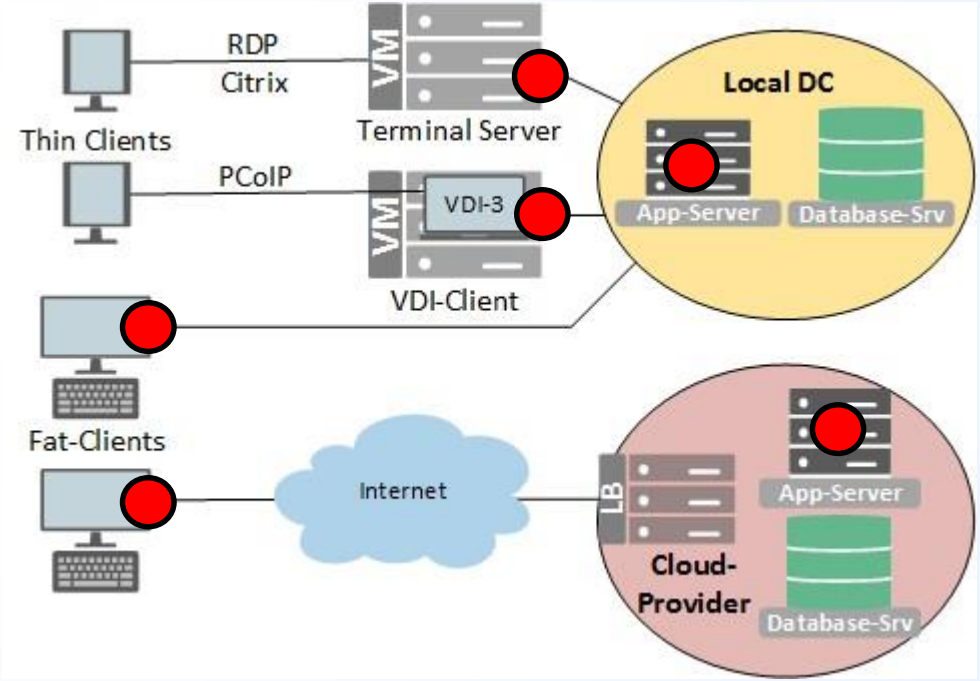
Sample Database Flow

- Front end Process
  - HTTP(S) or specific TCP
- Back end Process
  - Many Requests - Responses (Application Turns)
  - Small amount of data
- Back End sensitive to
  - Round Trip Time
  - Number of Turns
  - Application Response Time of Database Server
  - Delays at App-Server



Traffic Flows and Client Server Architecture

- Client-Server architecture
  - Fat Clients
  - Terminal Server
  - Virtual Clients (VDI)
  - Cloud environment
- Traffic flows
  - Client – Servers – DB-Srv
- Which Users are affected?
- Capture location
  - Client Session
  - Application



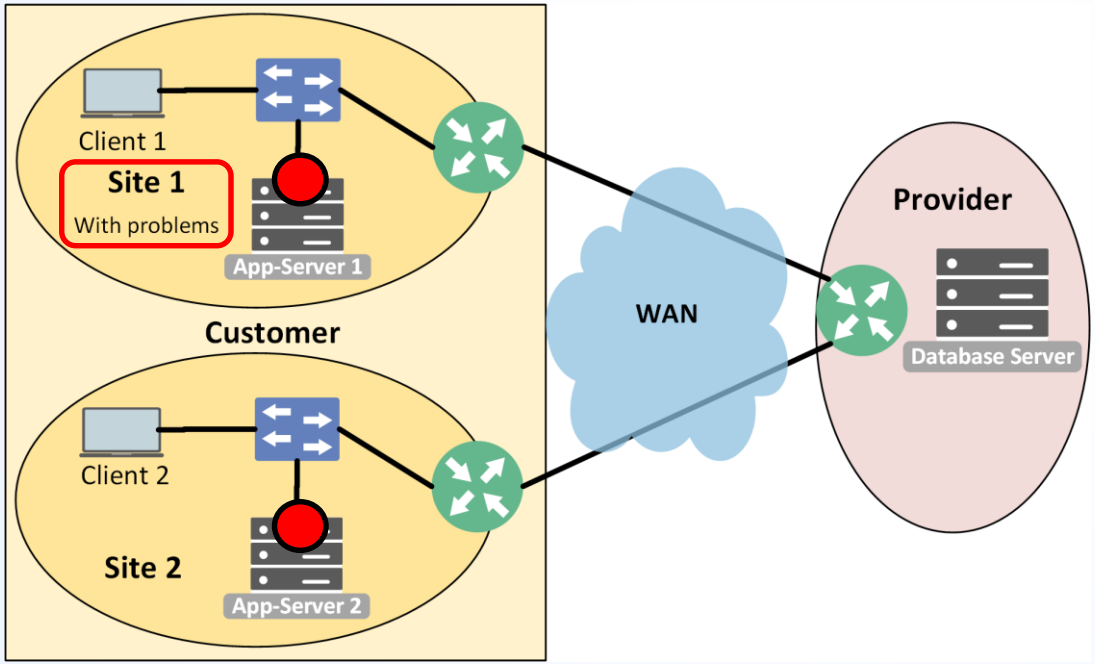
#sf21veu

Case Study 1

Real-Life Case #1

- Case: Weighing process of palettes (steel)
- Problem: High transaction times for database transaction → Permanent
- User activity: Weighing process (repeatable)
- Trace Files: 1-Site1-before.pcapng
  1-Site2-reference.pcapng
- Wireshark Profile: App-Analysis-I
- Suspect: Network
- Questions: Where is the problem?

Real-Life Case #1 - network map



- Network map for Case #1
- Traces taken at App-Servers.
- Front-End and Back-End Flows visible

Real-Life Case #1 - Analysis

ip.addr eq 192.0.2.19 and ip.addr eq 192.0.2.108

No.	Delta Time	Rel. Time	Source	Destination	Protocol	Length	Info
1	0.000000	0.000000	Client-Site1	App-SRV-Site-1	TCP	193	4813 → 2048
2	0.213349	0.213349	App-SRV-Site-1	Client-Site1	TCP	60	2048 → 4813
52	6.733710	6.947059	App-SRV-Site-1	Client-Site1	TCP	116	2048 → 4813
53	0.197722	7.144781	Client-Site1	App-SRV-Site-1	TCP	60	4813 → 2048
54	1.239990	8.384771	Client-Site1	App-SRV-Site-1	TCP	221	4813 → 2048
55	0.140996	8.525767	App-SRV-Site-1	Client-Site1	TCP	60	2048 → 4813
207	31.921329	40.447096	App-SRV-Site-1	Client-Site1	TCP	146	2048 → 4813
237	0.166188	40.613284	Client-Site1	App-SRV-Site-1	TCP	60	4813 → 2048

ip.addr eq 198.51.100.41 and ip.addr eq 198.51.100.90

No.	Delta Time	Rel. Time	Source	Destination	Protocol	Length	Info
1	0.000000	0.000000	Client-Site-2	APP-SRV-Site-2	TCP	60	1035 → 2048 [
2	0.111118	0.111118	APP-SRV-Site-2	Client-Site-2	TCP	54	2048 → 1035 [
3	0.004341	0.115459	Client-Site-2	APP-SRV-Site-2	TCP	112	1035 → 2048 [
4	0.214422	0.329881	APP-SRV-Site-2	Client-Site-2	TCP	54	2048 → 1035 [
52	2.392365	2.722246	APP-SRV-Site-2	Client-Site-2	TCP	116	2048 → 1035 [
53	1.435792	4.158038	APP-SRV-Site-2	Client-Site-2	TCP	116	[TCP Retransm
54	2.832130	6.990168	Client-Site-2	APP-SRV-Site-2	TCP	60	1035 → 2048 [
57	0.209241	7.329956	APP-SRV-Site-2	Client-Site-2	TCP	54	2048 → 1035 [
58	0.006218	7.336174	Client-Site-2	APP-SRV-Site-2	TCP	141	1035 → 2048 [
59	0.212528	7.548702	APP-SRV-Site-2	Client-Site-2	TCP	54	2048 → 1035 [
180	5.063001	12.611703	APP-SRV-Site-2	Client-Site-2	TCP	178	2048 → 1035 [
181	0.179710	12.791413	Client-Site-2	APP-SRV-Site-2	TCP	60	1035 → 2048 [
195	2.469500	15.260913	APP-SRV-Site-2	Client-Site-2	TCP	146	2048 → 1035 [

- Front End - Site 1:
  - Transaction Time: **40.6 s**
  - High ACK-Times at App-Srv: app. 150 - 200 ms
  - TCP Retransmissions
- Front End – Site 2: Reference Site
  - Transaction Time: **15.2 s**
  - High ACK-Times at App-Srv: app. 100 - 150 ms
  - TCP Retransmissions

Real-Life Case #1 - Analysis

tcp.len > 1 | Expression... | + | Just Application Data

No.	Delta Time	Rel. Time	Source	Destination	Protocol	Length	Info
1	0.000000	0.000000	Client-Site1	App-SRV-Site-1	TCP	193	4813 → 2048 [PSH, ACK]
3	2.795672	2.795672	App-SRV-Site-1	DB-Server-1	TCP	153	4152 → 3231 [PSH, ACK]
4	0.017785	2.813457	DB-Server-1	App-SRV-Site-1	TCP	633	3231 → 4152 [PSH, ACK]
6	3.154290	5.967747	App-SRV-Site-1	DB-Server-1	TCP	177	4152 → 3231 [PSH, ACK]
7	0.015088	5.982835	DB-Server-1	App-SRV-Site-1	TCP	221	3231 → 4152 [PSH, ACK]

tcp.len > 1 | Expression... | + | Just Application Data

No.	Delta Time	Rel. Time	Source	Destination	Protocol	Length	Info
56	6.952157	15.336928	App-SRV-Site-1	DB-Server-1	TCP	153	4152 → 3231 [PSH, ACK]
169	3.919706	33.623444	App-SRV-Site-1	DB-Server-1	TCP	386	4152 → 3231 [PSH, ACK]
59	3.533441	18.886262	App-SRV-Site-1	DB-Server-1	TCP	177	4152 → 3231 [PSH, ACK]
88	3.370261	23.756958	App-SRV-Site-1	DB-Server-1	TCP	153	4153 → 3231 [PSH, ACK]
6	3.154290	5.967747	App-SRV-Site-1	DB-Server-1	TCP	177	4152 → 3231 [PSH, ACK]
91	3.145083	26.919965	App-SRV-Site-1	DB-Server-1	TCP	174	4153 → 3231 [PSH, ACK]
184	2.898743	40.189704	App-SRV-Site-1	DB-Server-1	TCP	174	4153 → 3231 [PSH, ACK]
3	2.795672	2.795672	App-SRV-Site-1	DB-Server-1	TCP	153	4152 → 3231 [PSH, ACK]
163	2.422311	29.687232	DB-Server-1	App-SRV-Site-1	TCP	1514	3231 → 4152 [ACK] Seq=
171	2.132969	35.756413	DB-Server-1	App-SRV-Site-1	TCP	221	3231 → 4152 [PSH, ACK]
54	1.437712	8.384771	Client-Site1	App-SRV-Site-1	TCP	221	4813 → 2048 [PSH, ACK]
71	1.388123	20.325455	App-SRV-Site-1	DB-Server-1	TCP	473	4152 → 3231 [PSH, ACK]
173	1.255924	37.012337	App-SRV-Site-1	DB-Server-1	TCP	153	4153 → 3231 [PSH, ACK]
52	0.441456	6.947059	App-SRV-Site-1	Client-Site1	TCP	116	2048 → 4813 [PSH, ACK]
38	0.232304	6.411953	DB-Server-1	App-SRV-Site-1	TCP	221	3231 → 4152 [PSH, ACK]
176	0.224049	37.252566	DB-Server-1	App-SRV-Site-1	TCP	1514	3231 → 4152 [ACK] Seq=

- Filter on just application data
  - tcp.len > 1
  - Sort by Delta Time
  - Large Delta times can be easily spotted.
- Sort by High Delta Times
  - From App-SRV:
    - 10 * High Delta Times
    - Total: 32.6 seconds
  - From Database-SRV:
    - 2 * High SRT
    - Total: 4.55 seconds

#sf21veu

Real-Life Case #1 - Analysis

- Comparing Site 1 to Site 2

tcp.len > 1

No.	Delta Time	Rel. Time	Source	Destination	Protocol
56	6.952157	15.336928	App-SRV-Site-1	DB-Server-1	TCP
169	3.919706	33.623444	App-SRV-Site-1	DB-Server-1	TCP
59	3.533441	18.886262	App-SRV-Site-1	DB-Server-1	TCP
88	3.370261	23.756958	App-SRV-Site-1	DB-Server-1	TCP
6	3.154290	5.967747	App-SRV-Site-1	DB-Server-1	TCP
91	3.145083	26.919965	App-SRV-Site-1	DB-Server-1	TCP
184	2.898743	40.189704	App-SRV-Site-1	DB-Server-1	TCP
3	2.795672	2.795672	App-SRV-Site-1	DB-Server-1	TCP
163	2.422311	29.687232	DB-Server-1	App-SRV-Site-1	TCP
171	2.132969	35.756413	DB-Server-1	App-SRV-Site-1	TCP
54	1.437712	8.384771	Client-Site1	App-SRV-Site-1	TCP
71	1.388123	20.325455	App-SRV-Site-1	DB-Server-1	TCP
173	1.255924	37.012337	App-SRV-Site-1	DB-Server-1	TCP
52	0.441456	6.947059	App-SRV-Site-1	Client-Site1	TCP
38	0.232304	6.411953	DB-Server-1	App-SRV-Site-1	TCP
176	0.224049	37.252566	DB-Server-1	App-SRV-Site-1	TCP

tcp.len > 1

No.	Delta Time	Rel. Time	Source	Destination	Protocol
56	2.962677	7.120715	Client-Site-2	APP-SRV-Site-2	TCP
53	1.435792	4.158038	APP-SRV-Site-2	Client-Site-2	TCP
76	1.413021	9.826197	APP-SRV-Site-2	DB-Server-1	TCP
184	1.307916	15.075653	DB-Server-1	APP-SRV-Site-2	TCP
182	1.156034	13.767737	DB-Server-1	APP-SRV-Site-2	TCP
176	0.662046	12.477821	APP-SRV-Site-2	DB-Server-1	TCP
5	0.638570	0.754029	APP-SRV-Site-2	DB-Server-1	TCP
20	0.563156	1.896660	APP-SRV-Site-2	DB-Server-1	TCP
127	0.551369	11.298349	APP-SRV-Site-2	DB-Server-1	TCP
63	0.539112	8.360383	APP-SRV-Site-2	DB-Server-1	TCP
8	0.498658	1.270346	APP-SRV-Site-2	DB-Server-1	TCP
60	0.469608	7.805782	APP-SRV-Site-2	DB-Server-1	TCP

Real-Life Case #1 - Solution

- Results from the analysis
  - App-Server seems to take many long breaks!
  - App-Server shows high ACK time.
  - Also a few retransmissions at both sites.
- Next steps
  - Check App-Server health!
  - Check App-Server application!
  - Take care of retransmissions later

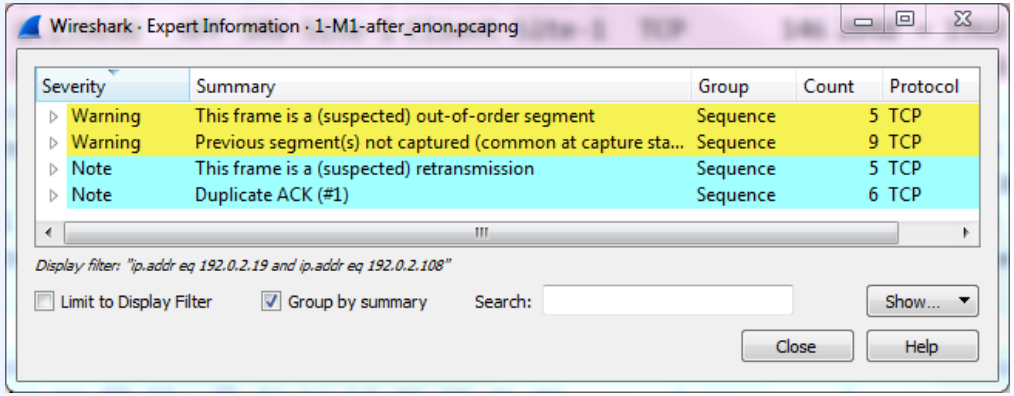
Real-Life Case #1 - Solution

- What we found on the „Application Server“:
- Application:
  - MS-Access „database“ with 1.2 GBytes in size.
  - Had not been reorganized for months
- Machine itself:
  - Just 256 MBytes of RAM, high level of disk swapping
  - Machine was heavily overloaded (corresponding with high RTT)
- Fix
  - Reorganize the DB on App-Server → 1-Site1-after.pcapng
  - Add more RAM to the machine → scheduled for later

Real-Life Case #1 - Solution

ip.addr eq 192.0.2.19 and ip.addr eq 192.0.2.108 | Expression... | + | Just Application Data | Frontend-Site1 | Backend-Site1 | Fro

No.	Delta Time	Rel. Time	Source	Destination	Protocol	Length	Info
1	0.000000	0.000000	Client-Site-1	APP-SRV-Site-1	TCP	113	1968 → 2048
2	0.190374	0.190374	APP-SRV-Site-1	Client-Site-1	TCP	60	2048 → 1968
51	1.189452	1.379826	APP-SRV-Site-1	Client-Site-1	TCP	116	2048 → 1968
52	0.154333	1.534159	Client-Site-1	APP-SRV-Site-1	TCP	60	1968 → 2048
53	1.186427	2.720586	Client-Site-1	APP-SRV-Site-1	TCP	141	1968 → 2048
54	0.204285	2.924871	APP-SRV-Site-1	Client-Site-1	TCP	60	2048 → 1968
133	5.214819	8.139690	APP-SRV-Site-1	Client-Site-1	TCP	146	2048 → 1968
134	0.175923	8.315613	Client-Site-1	APP-SRV-Site-1	TCP	60	1968 → 2048



- Transaction after fix #1
  - Transaction time: 8.3 s
- Still present:
  - High ACK-times at App-Srv: 200 ms
  - Still overloaded machine (RAM to be added)
- Retransmissions due to packet loss
  - Caused by **Duplex Mismatch** between APP-Server and DB-Server

Lessons learned - Case 1

- An overloaded App-Server caused high delays.
  - Filter out TCP-ACKs (tcp.len > 1)
  - Look at large Delta Times
  - Check Flow Graph
- Reorganizing the database helped
- Adding RAM helped as well
- Duplex mismatch between Switch and Router caused packet reordering and retransmission
- Important: Don‘t stop after you identified the first problem.

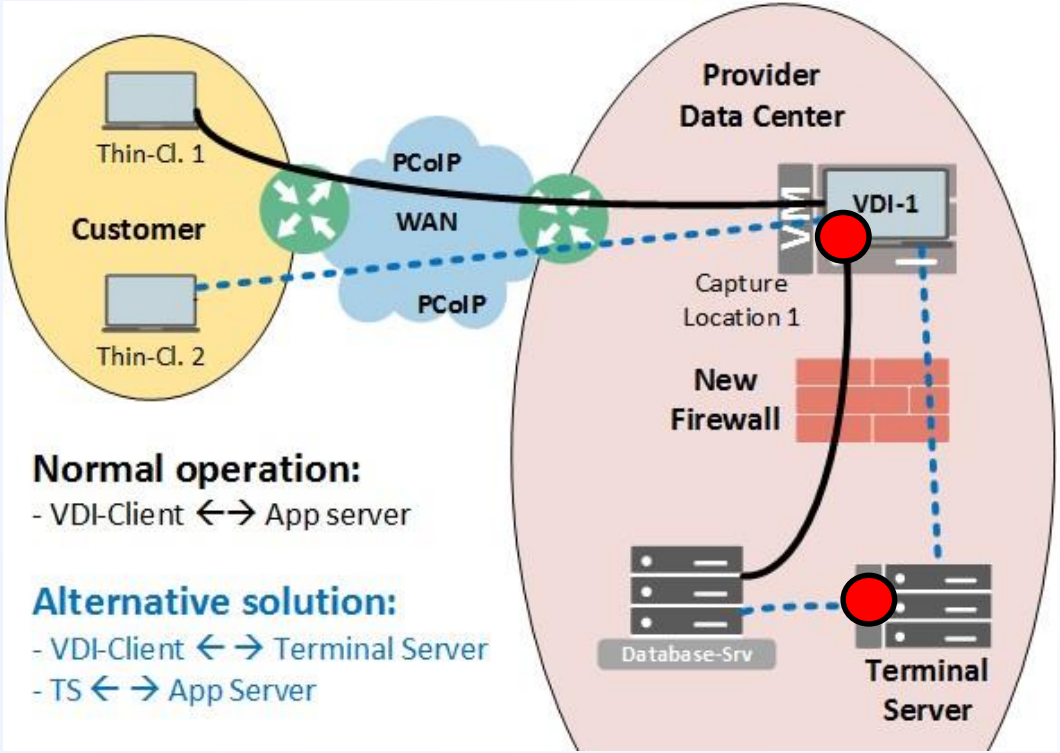
#sf21veu

Case Study 3

Real-Life Case #3

- Case: Accounting software usage after moving Data Center to a new Service Provider
- Customer: Agency for temporary work.
- Problem: High transaction times, higher than baseline → Permanent
- Transaction: Login-Process for user
- Trace File: 3-VDI.pcapng
  Reference: 3-TS.pcapng
- Wireshark Profile: App-Analysis-III
- Suspects: „Check out everything!“

Case #3 - network map



- Normal client
  - VDI Client accesses app server directly
  - Problem: Longer transaction times
- Intermediate test
  - VDI-Client via Terminal Server
  - Result: Better transaction times
- Let‘s start the analysis.

Case #3 - Capture Preps

- Problem affected all users using the main application
  - Replicable process
  - We selected a typical task where a baseline existed.
    - User Login
  - Traces were taken - data was filtered and isolated
    - 3 client traces showed similar figures
    - So it was really a repeatable process
  - Traces were taken for
    - Pure VDI users
    - Terminal Server test users

Case #3 - Analysis

- Transaction: User Login
  - VDI: 31 seconds, 18056 packets
  - TS: 10 seconds, 18035 packets
  - Baseline: 8 seconds, # of packets unknown
- Questions
  - What makes VDI so slow?
  - Why is TS much faster?
  - Where is the bottleneck?
  - How can we improve the performance?

#sf21veu

Case #3 - Analysis

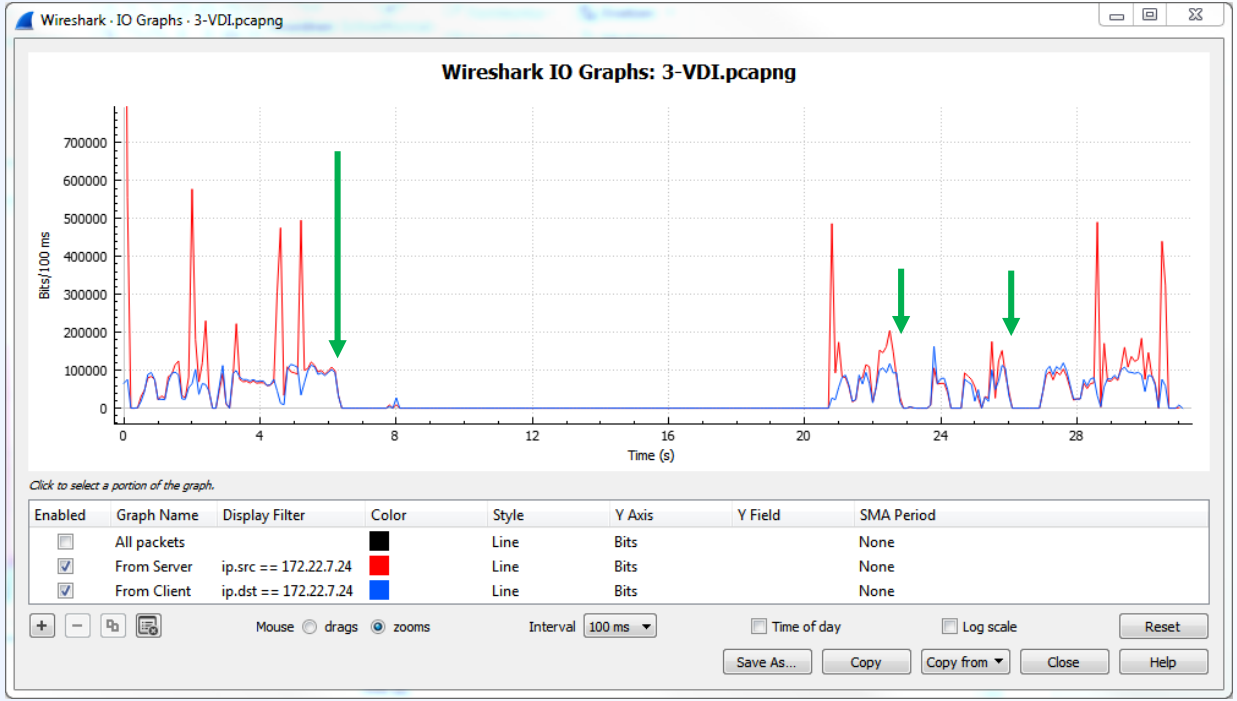
- VDI-Client
  - Filter out ACKs, then sort by Delta Times
  - Note the three high Delta Times, all from VDI Client

tcp.len > 1 — Expression...

No.	Delta Time	Rel. Time	Time Delta abs	Source	Destination	Protocol	Length	Info
8299	12.763792	20.843198	12.7498...	VDI-Client	Database-Srv	TDS	99	Unknown
8283	1.490410	7.818979	0.001012	VDI-Client	Database-Srv	TDS	148	Unknown
12733	0.951646	27.018337	0.951605	VDI-Client	Database-Srv	TDS	1441	Unknown
10947	0.565906	23.773320	0.565857	VDI-Client	Database-Srv	TDS	403	Unknown
11574	0.446685	24.716186	0.435278	VDI-Client	Database-Srv	TDS	607	Unknown
18052	0.420182	31.083026	0.411542	VDI-Client	Database-Srv	TDS	99	Unknown
10935	0.329163	23.161573	0.329079	VDI-Client	Database-Srv	TDS	99	Unknown
400	0.304443	0.491266	0.288595	VDI-Client	Database-Srv	TDS	99	Unknown
2746	0.267188	2.823361	0.261315	VDI-Client	Database-Srv	TDS	291	Unknown
8288	0.216659	8.038074	0.210400	VDI-Client	Database-Srv	TDS	244	Unknown

Case #3 - Analysis

#sf21veu



- VDI-Client
  - Large gaps on client side
  - Longest is 12 seconds

#sf21veu

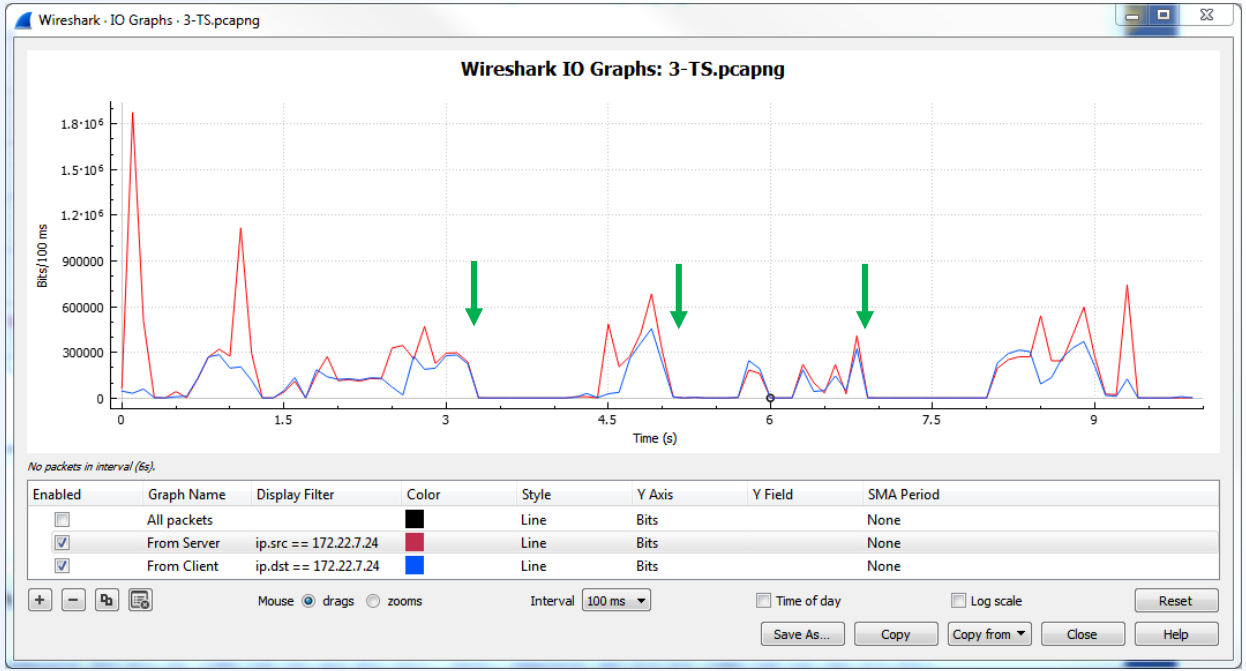
Case #3 - Analysis

- Terminal Server
  - Lower values for Large Delta Times, all from Terminal Server

tcp.len > 1 | Expression... | + | Backer

No.	Delta Time	Rel. Time	Time Delta abs	Source	Destination	Protocol	Length	Info
12708	1.203484	8.114523	1.203472	Terminal-Srv	Database-Srv	TDS	1514	Unknown Pac
8268	0.939714	4.272634	0.000678	Terminal-Srv	Database-Srv	TDS	148	Unknown Pac
18030	0.483126	9.894180	0.483113	Terminal-Srv	Database-Srv	TDS	99	Unknown Pac
10928	0.425522	5.789749	0.425494	Terminal-Srv	Database-Srv	TDS	403	Unknown Pac
11549	0.323179	6.312351	0.323153	Terminal-Srv	Database-Srv	TDS	607	Unknown Pac
2775	0.298503	1.584523	0.295125	Terminal-Srv	Database-Srv	TDS	291	Unknown Pac
390	0.230776	0.548034	0.230769	Terminal-Srv	Database-Srv	TDS	99	Unknown Pac
10917	0.193351	5.316234	0.182085	Terminal-Srv	Database-Srv	TDS	99	Unknown Pac
3146	0.179343	1.841506	0.130231	Terminal-Srv	Database-Srv	TDS	305	Unknown Pac
17723	0.172489	9.296061	0.124133	Terminal-Srv	Database-Srv	TDS	519	Unknown Pac

Case #3 - Analysis

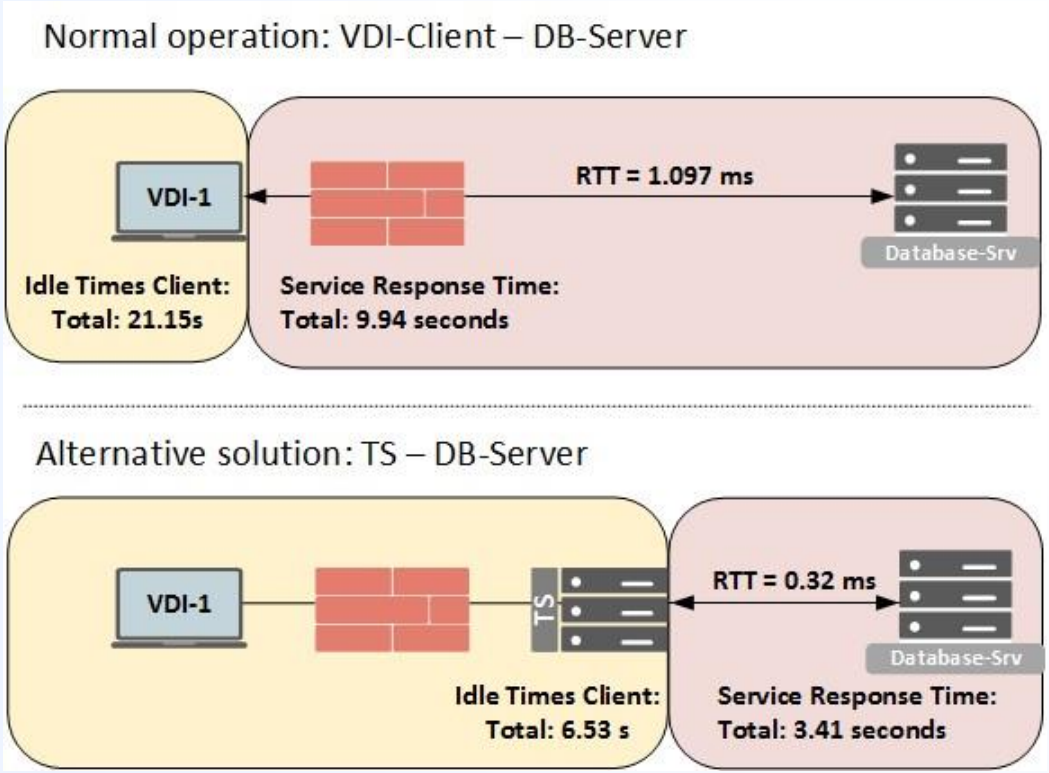


- Terminal Server
  - Gaps on client side
  - Longest is 1.4 seconds

Case #3 - Resolution - Who‘s to blame?

- Who is responsible for most of the increase in transaction time?
  - VDI-Client, Database Server, Terminal Server?
- Answer: It must be the …
  - VDI-Client
  - → Check for reasons for the „high think times“ on VDI Client
- Action:
  - Improve the hardware of VDI host and the VDI software platform
- Findings
  - Improved the transaction time, but still worse than baseline.
- Is there still room for improvement?
  - Database Server?
  - Firewall?

Case #3 - Extras: Summarizing the Response Times



- Isolated Client and Server times
- Client Idle Times:
  - VDI: 21.15 s
  - TS: 6.53 s
- Server Response Times
  - VDI: 9.94 s
  - TS: 3.41 s

#sf21veu

Case #3 - Extras: How this was done

Total		
Sum Requests + Responses	17169	Packets
Transaction Time	**31,09**	Seconds

Client Idle Times (Client Think Time)		
Number of Requests (Turns)	8607	Requests
Idle Time SUM	**21,15**	Seconds
Idle Time Average	2,46	Milliseconds
Idle Time Median	0,06	Milliseconds
Idle Time 90% Percentile	0,35	Milliseconds
Idle Time Max	**12763,79**	Milliseconds
Sum 10% worst	20,51	Seconds

Network		
Average Round Trip Time	1,097	Milliseconds
Retransmissions	**17**	

Server Response Times (Server Think Time)		
Number of responses	8562	Responses
Sum SRT	**9,94**	Seconds
SRT Average	1,16	Milliseconds
SRT Median	0,82	Milliseconds
SRT 90% Percentile	1,45	Milliseconds
SRT Max	**141,33**	Milliseconds
Sum 10% worst	3,41	Seconds

Effect of RTT and Turns		
Application Turns	8562	Requests
RTT	1,097	Milliseconds
Total: (Turns * RTT)	**9,39**	**Seconds**

Total Server w/o Network	**0,55**	**Seconds**

- VDI Client Metrics
- Total Time spent: 31 seconds
  - At VDI-Client: **21.15 s**
  - Network + Server + Application: **9.94 s**

#sf21veu

Case #3 - Extras: How this was done!

Total		
Total Number of Packets	17175	Packets
Transaction Time	**9,94**	Seconds

Client Idle Times (Client Think Time)		
Number of Requests	8599	Requests
Idle Time SUM	**6,53**	Seconds
Idle Time Average	0,76	Milliseconds
Idle Time Median	0,04	Milliseconds
Idle Time 90% Percentile	0,15	Milliseconds
Idle Time Max	**1203,48**	Milliseconds
Sum 10% worst	6,21	Seconds

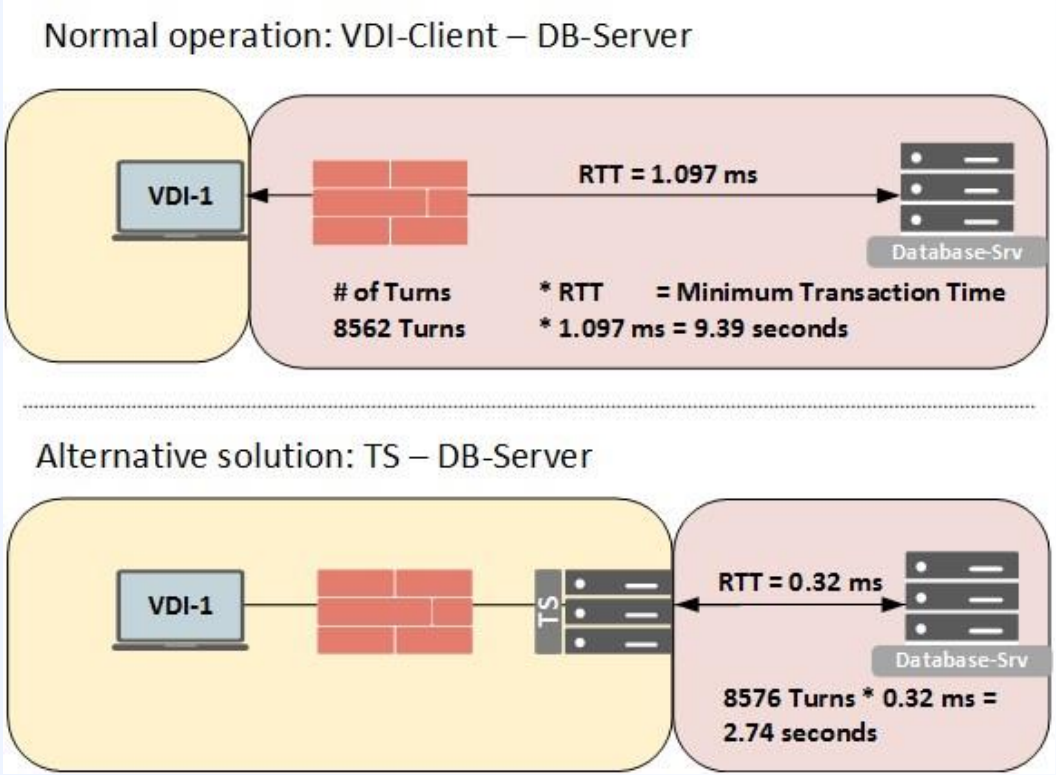
Network		
Average Round Trip Time	0,32	Milliseconds
Retransmissions	**22**	

Server Response Times (Server Think Time)		
Number of responses	8576	Responses
Sum SRT	**3,41**	Seconds
SRT Average	0,39	Milliseconds
SRT Median	0,21	Milliseconds
SRT 90% Percentile	0,51	Milliseconds
SRT Max	**142,60**	Milliseconds
Sum 10% worst	1,68	Seconds

Effect of RTT and Turns		
Application Turns	8576	Requests
RTT	0,32	Milliseconds
Total: (Turns * RTT)	**2,74**	**Seconds**
Total Server w/o Network	**0,67**	**Seconds**

- Terminal Server Metrics
- Total Time spent:
  - At Terminal Server: **6.53 s**
  - Network + Server + Application: **3.41 s**
    → which is 6,55 s less than SRT at VDI

Case#3 - Effect of Round Trip Time



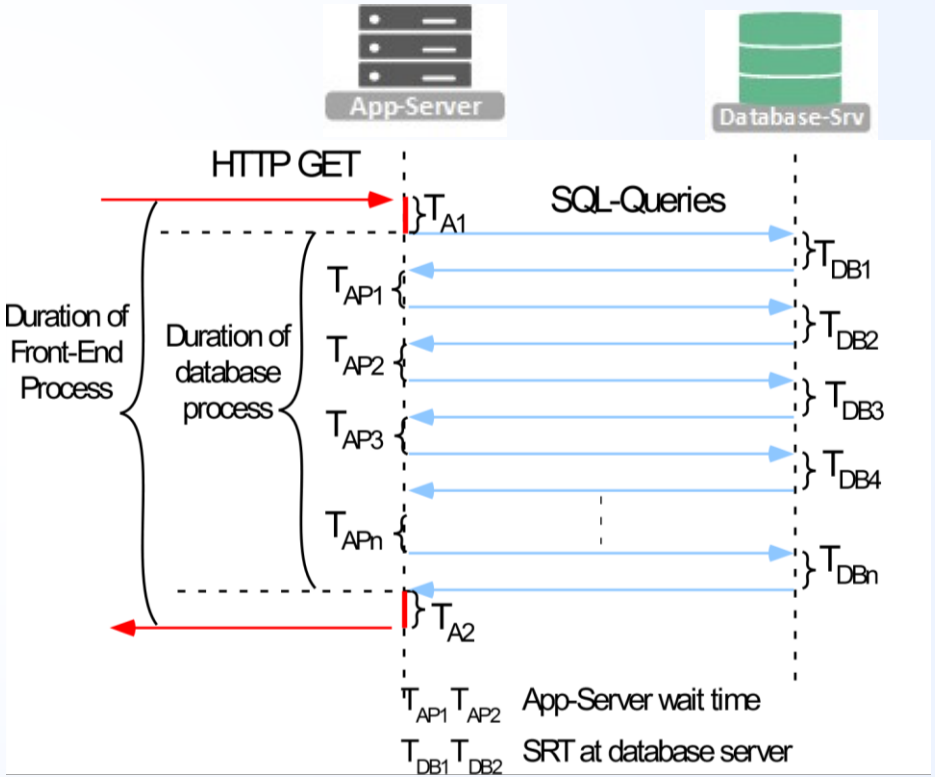
- One Round Trip:
  - Difference: **0.777 ms**
- ~ 8560 turns
  - Difference: **6,65 seconds**
  - Added by the firewall
- Guess what the customer responded, when we told him…
- **Let‘s remove the firewall!**
- **Was he right?**

Lessons Learned - Case 3

- VDI Clients have to carry the load of the application!
  - Need enough performance!
  - Applications often show a tendency towards Terminal Services or to VDI
- Firewalls add extra delay
  - But was the firewall the real problem?
- A huge number of application turns has great impact on app performance
  - Small increase on RTT, but huge overall delay
- Would you remove the firewall or have the application rewritten?

Application Metrics for Database Applications

- Key Metrics
  - SRT at Front End vs.
  - Duration of database process
- Additional Metrics
  - SRTs at Database (T_{DB})
  - App-Server „think“ times (T_{AP})
  - Number of application turns
  - Round Trip Time
  - RTT * Number of Turns = Minimum Transaction Time



Lessons Learned

Database Applications are sensitive to

- High Server (DB) response times -> Slow database
- Long Client wait times -> Slow Application on Client or App-Srv
- Very sensitive to Round Trip Time (RTT)
  - Many application turns should be avoided
  - Programming Techniques: „Row by Row is slow by slow“

Q & A

- Questions from the chat?

Thank You!

- Thank you for listening!
- Please leave your feedback in the feedback portal.
  - https://forms.gle/vELKPFgDobAMVC8n7
  - Link also in Chat and published on SharkFest documents.
- For further questions meet me on Discord Server
  - Voice Channel: zoom 1 discussion
  - Starts in 5 minutes after this presentation ends
- Contact me
  - Matthias.Kaiser@experteach.de
  - Twitter: @wiresharky

#sf21veu

End of Presentation

#sf21veu

#sf21veu

Case Study 2

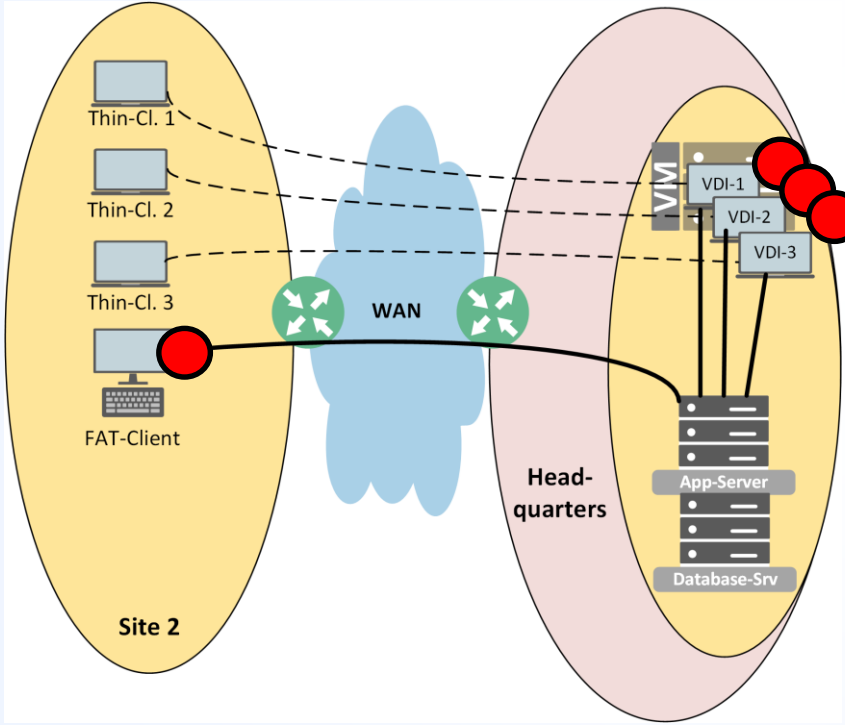
#sf21veu

Real-Life Case #2

- Case: New application in emergency room of a hospital
- Real-Life Case #2: Application freeze for users → Intermittent
- Trace File: 2-Before-oneclient.pcapng
  2-Before-allclients.pcapng
- Wireshark Profile: App-Analysis-II
- Suspects: Network
- Questions: Is it the network?
  If not, where is the problem?

Real-Life Case #2 - network map

#sf21veu



- Sample Client machines
  - 3 Virtual-Clients at Site 2, VDI clients at HQ.
  - 1 Fat-Client at Site 2
- All four users reported problems
  - „Application freeze“
  - Freezes > 10 seconds
  - App freezes, not the client.
- Intermittent Problem

Real-Life Case #2 - Methodology

- Identify communication pattern
- Methodology
  - Capture 4 clients simultaneously
    Traces were captured on fat client and on VDI clients.
  - Ask users to note application freeze times
  - Try to correlate noted freeze times to packets in the trace files
- Analysis
  - Check network performance (RTT and TCP errors)
    - ➔ RTT: 4ms, No Errors
  - Then check Server Response Times
  - And check Server Performance

Real-Life Case #2 - Analysis

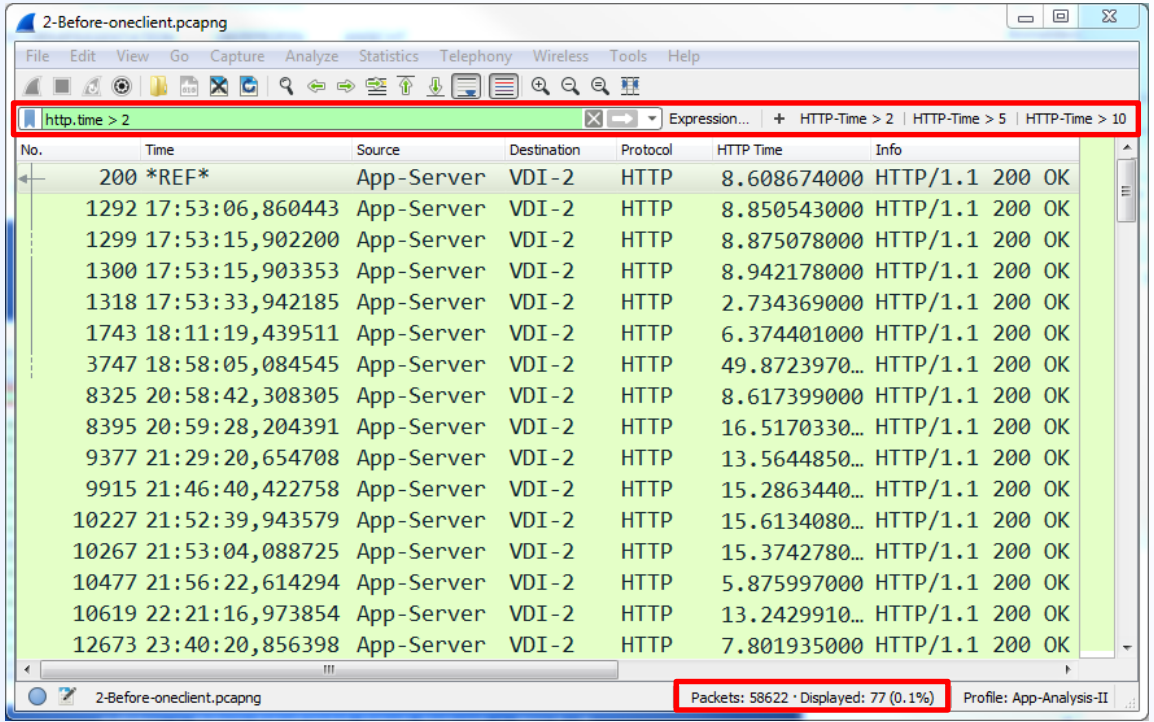
No.	Time	Source	Destination	Protocol	Info
1	17:28:13,064071	VDI-2	App-Server	HTTP	POST /transaction01
2	17:28:13,066549	App-Server	VDI-2	HTTP	HTTP/1.1 200 OK [Pa
3	17:28:25,689058	VDI-2	App-Server	HTTP	POST /transaction01
4	17:28:25,690381	App-Server	VDI-2	HTTP	HTTP/1.1 200 OK [Pa
5	17:28:25,691127	VDI-2	App-Server	HTTP	POST /transaction01
6	17:28:25,692921	App-Server	VDI-2	HTTP	HTTP/1.1 200 OK [Pa

No.	Time	Source	Destination	Protocol	HTTP Time	Info
1	17:28:13,064071	VDI-2	App-Server	HTTP		POST /transactior
2	17:28:13,066549	App-Server	VDI-2	HTTP	0.002478000	HTTP/1.1 200 OK
3	17:28:25,689058	VDI-2	App-Server	HTTP		POST /transactior
4	17:28:25,690381	App-Server	VDI-2	HTTP	0.001323000	HTTP/1.1 200 OK
5	17:28:25,691127	VDI-2	App-Server	HTTP		POST /transactior
6	17:28:25,692921	App-Server	VDI-2	HTTP	0.001794000	HTTP/1.1 200 OK
7	17:28:25,693980	VDI-2	App-Server	HTTP		POST /transactior
8	17:28:25,698063	App-Server	VDI-2	HTTP	0.004083000	HTTP/1.1 200 OK

- Communication Pattern
  - HTTP POST →
  - ← HTTP/1.1 200 OK
  - http.time shows Application Response Time
- Task
  - Identify high values for SRT for HTTP
  - Correlate with times, when users noted an application freeze

Real-Life Case #2 - Analysis

#sf21veu



- First check with one client
  - ➔ High values for SRT.
  - High SRT values correlated with application freeze.
  - High SRT values showed random timing
- Next step
  - Long term capture on 4 clients

#sf21veu

Setting up the long term capture

- Long term capture with tshark
  - Batch file to start tshark for 1 day
    - `tshark -i 2 -w file.pcapng -B 200 -a duration:86400 -b filesize:200000`
  - Batch file put into Windows Task Scheduler
    - Starting after Login with SYSTEM rights (not interactive)
- First steps
  - Automated trace file processing with tshark and mergecap
    - Merge related files with mergecap
    - Filter by ip address and http packet with tshark
    - `tshark -2 -r infile.pcapng -Y „filter-expr“ -w outfile.pcapng“`

Case#2 - Analysis

- Evaluate the file via tshark script

```
echo ******************************************************
echo "Analyzing high HTTP Response Times > 2 seconds"
echo ******************************************************
echo Dumping to file
@call "C:\Program Files\Wireshark"\tshark.exe -2 -r filtered-b.pcapng -Y "http.time > 2" -T fields
-e frame.number -e frame.time -e ip.src -e ip.dst -e http.time > before.txt
echo Done
```

- Result

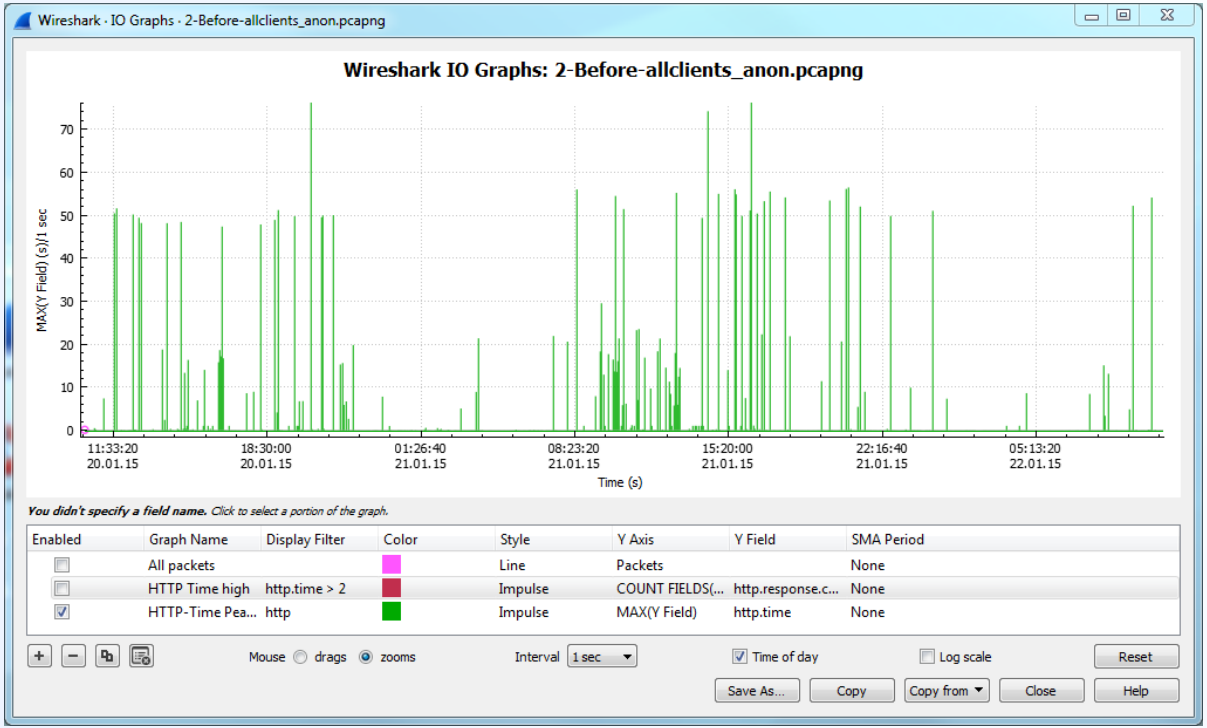
before.txt - Editor

Datei Bearbeiten Format Ansicht ?

6475	Jan 20, 2015 11:06:35.934341000 Mitteleuropäische Zeit	172.20.100.51	172.20.10.130	3.659518000
6490	Jan 20, 2015 11:06:38.167815000 Mitteleuropäische Zeit	172.20.100.51	172.20.10.128	7.385331000
9246	Jan 20, 2015 11:33:44.942576000 Mitteleuropäische Zeit	172.20.100.51	172.20.10.130	15.627170000
9271	Jan 20, 2015 11:33:47.209755000 Mitteleuropäische Zeit	172.20.100.51	172.20.10.128	7.529331000
9304	Jan 20, 2015 11:33:49.743762000 Mitteleuropäische Zeit	172.20.100.51	172.32.10.16	13.134020000
9387	Jan 20, 2015 11:34:14.157619000 Mitteleuropäische Zeit	172.20.100.51	172.32.10.16	50.425335000
9750	Jan 20, 2015 11:36:17.714457000 Mitteleuropäische Zeit	172.20.100.51	172.20.10.130	17.224207000
9772	Jan 20, 2015 11:36:21.668372000 Mitteleuropäische Zeit	172.20.100.51	172.20.10.130	3.253279000
9783	Jan 20, 2015 11:36:22.525544000 Mitteleuropäische Zeit	172.20.100.51	172.32.10.16	10.223137000
9795	Jan 20, 2015 11:36:26.490959000 Mitteleuropäische Zeit	172.20.100.51	172.32.10.16	3.712146000
9797	Jan 20, 2015 11:36:36.557460000 Mitteleuropäische Zeit	172.20.100.51	172.20.10.130	9.896393000
9816	Jan 20, 2015 11:36:46.762004000 Mitteleuropäische Zeit	172.20.100.51	172.20.10.130	5.069643000

Case#2 - Analysis

#sf21veu



- Evaluate via i/O graph
  - HTTP peak SRT values
  - MAX(http.time)

Case#2 - Analysis

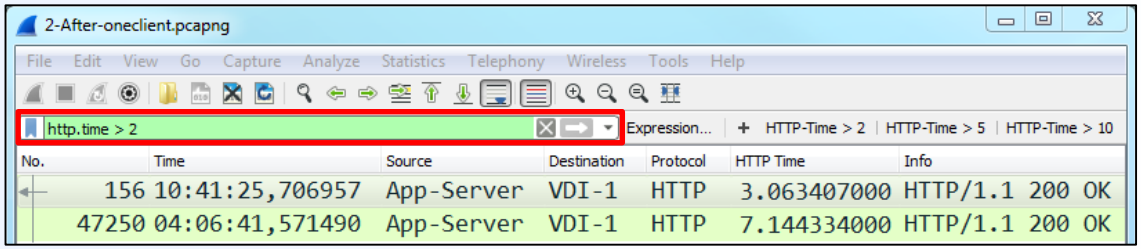
- Findings:
  - Server Related Problems
- At Server
  - Processes for IIS and MS-SQL went up to 98% CPU utilization every now and then (always together)
- From trace file
  - High response time were only seen when one specific transaction was issued from the client.
- This was reported to the company who wrote this application …

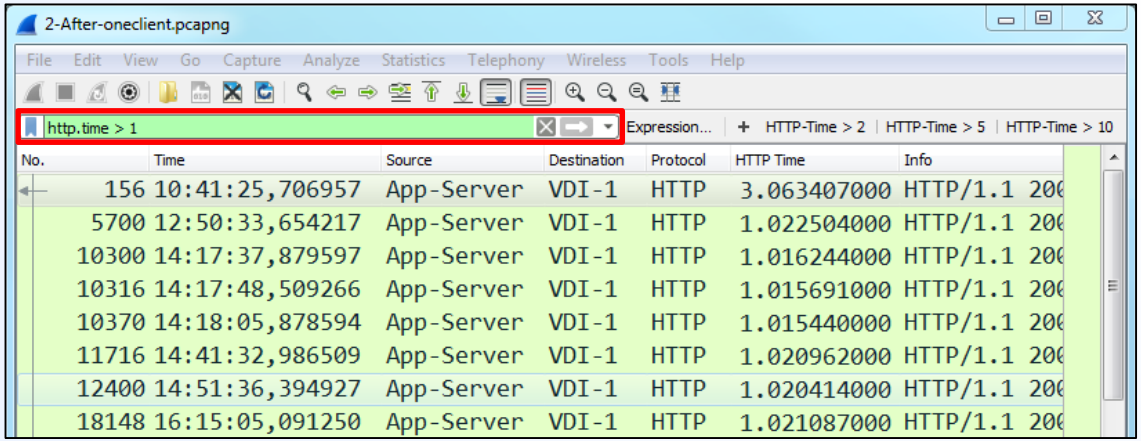
Case#2 - Solution

- Surprise: They listened and found a problem
- Software Update: We were asked to check performance again
- Trace files: 2-After-oneclient.pcapng
  2-After-allclients.pcapng
- Wireshark Profile: App-Analysis-II

Real-Life Case #2 - Solution

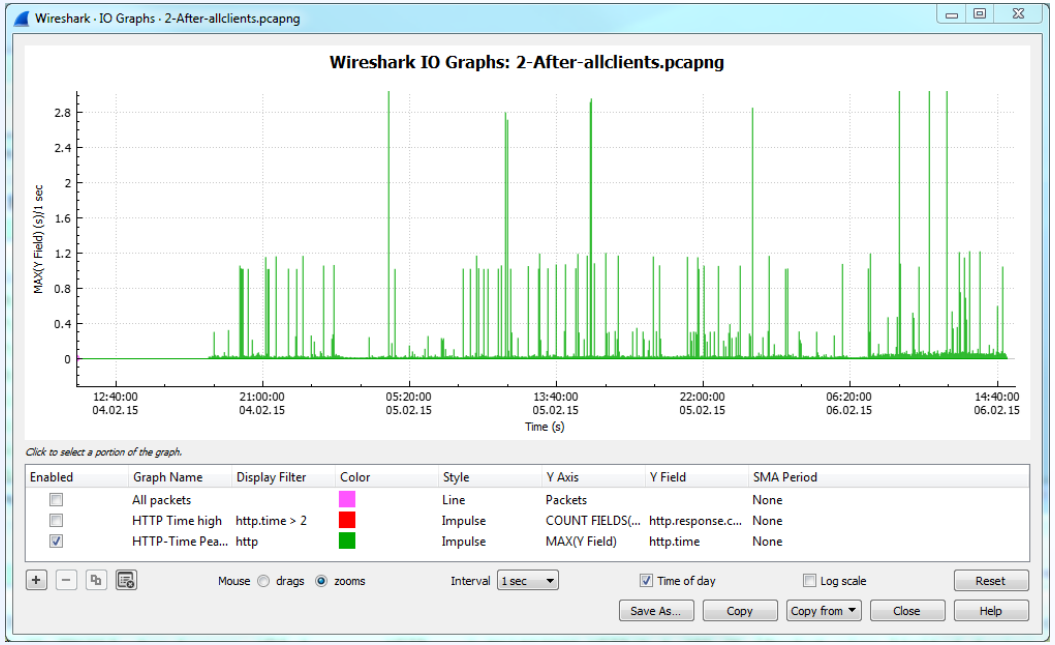
#sf21veu





- Result
  - Most of the high values for SRT were gone.
  - No app freeze noted by users any more.
- Still open
  - Response times of 1 s

Case#2 - Solution



- Evaluate via i/O graph
  - HTTP peak SRT values
  - http.time -> Max

Lessons Learned - Case 2

- Simple application Pattern
- No network problems
- High response times at Appserver
  - High load on Database Service
  - Timeout at Webserver
  - Specific application calls hung
- If it is not the network, check on the server side.

#sf21veu