SHARKFEST'14

WIRESHARK DEVELOPER AND USER CONFERENCE
JUNE 16-20 2014 · DOMINICAN UNIVERSITY

B10 - Understanding Wireshark's Reassembly Features

Christian Landström, Senior Consultant
Airbus Defence and Space

Agenda

- Introduction to Reassembly Features
- Use cases where Reassembly is used
- Side effects of the feature stack
- Best practices and recommendations

Introduction to Reassembly Features

- Reassembly works within:
  - IP
  - TCP
  - SSL
- Can be toggled via different ways
- Default: All features turned **ON**

Hands-on time!

- Fire up your Wireshark and capture your traffic (highly recommended)
- Go to:

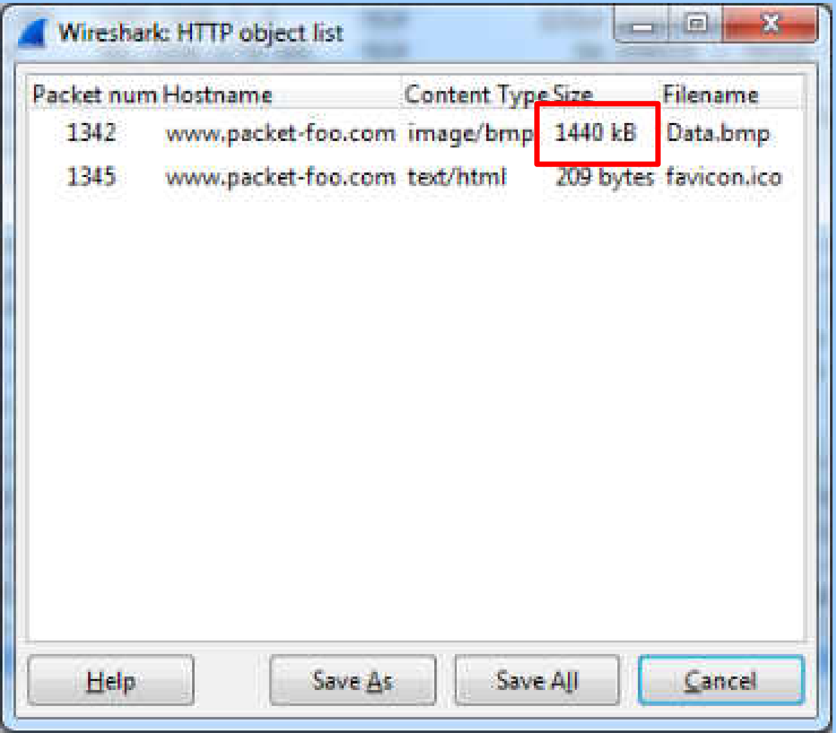
www.packet-foo.com/SF14/Data.bmp

- Alternatively click along using the sample captures:

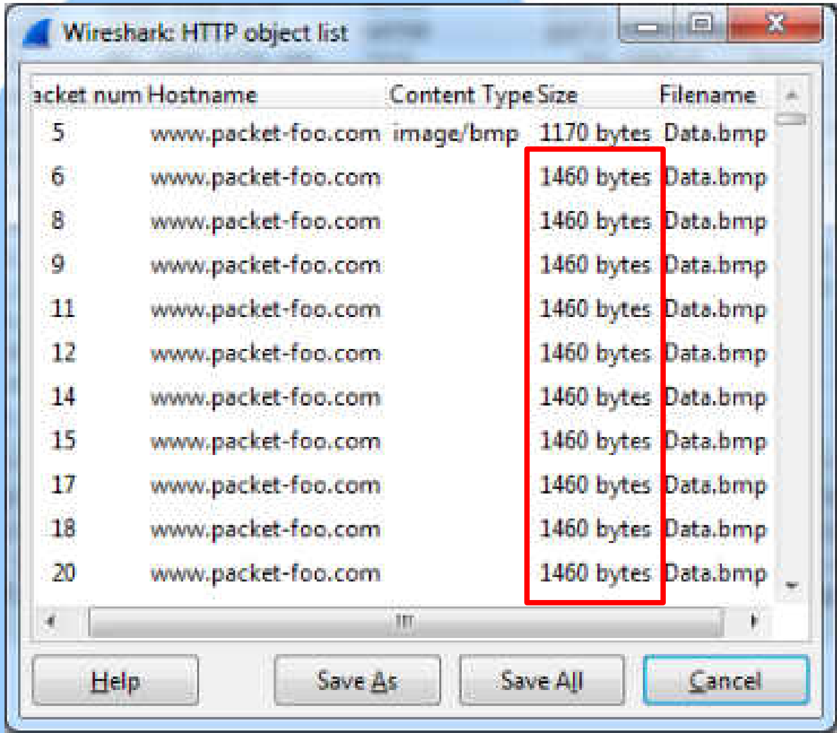
www.packet-foo.com/SF14/B10.zip

Focus: TCP Stream Reassembly

- Regularly used withing network analysis
- Enables reconstruction of segmented payload



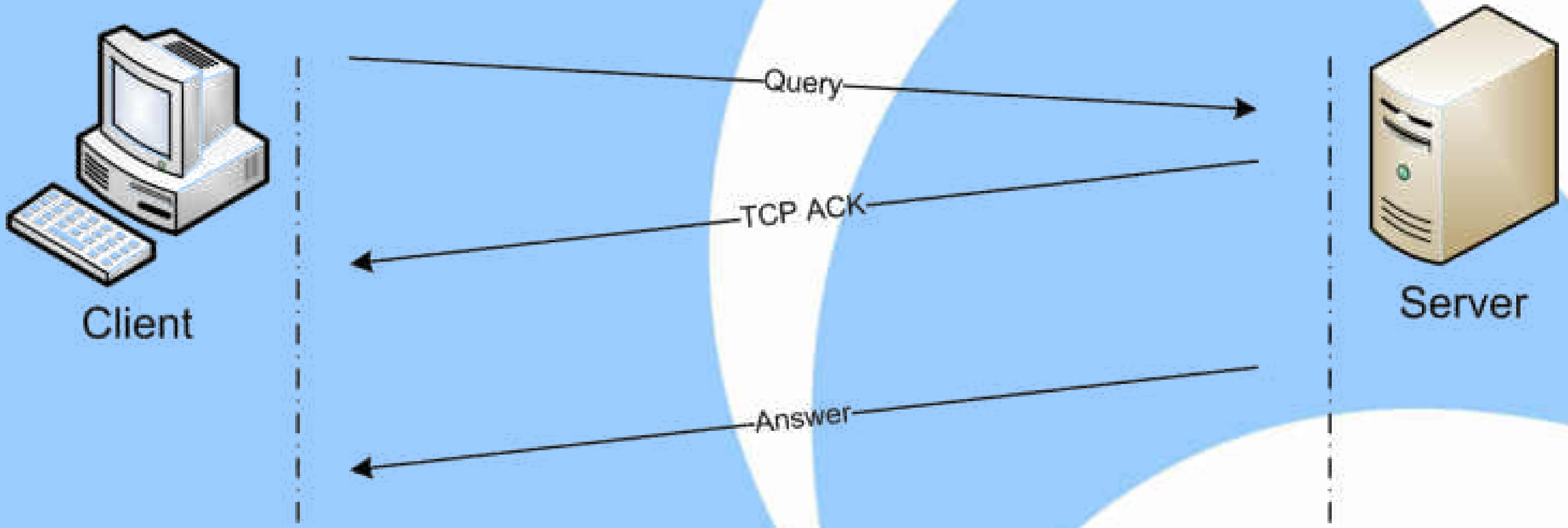
Packet num	Hostname	Content Type	Size	Filename
1342	www.packet-foo.com	image/bmp	1440 kB	Data.bmp
1345	www.packet-foo.com	text/html	209 bytes	favicon.ico



acket num	Hostname	Content Type	Size	Filename
5	www.packet-foo.com	image/bmp	1170 bytes	Data.bmp
6	www.packet-foo.com		1460 bytes	Data.bmp
8	www.packet-foo.com		1460 bytes	Data.bmp
9	www.packet-foo.com		1460 bytes	Data.bmp
11	www.packet-foo.com		1460 bytes	Data.bmp
12	www.packet-foo.com		1460 bytes	Data.bmp
14	www.packet-foo.com		1460 bytes	Data.bmp
15	www.packet-foo.com		1460 bytes	Data.bmp
17	www.packet-foo.com		1460 bytes	Data.bmp
18	www.packet-foo.com		1460 bytes	Data.bmp
20	www.packet-foo.com		1460 bytes	Data.bmp

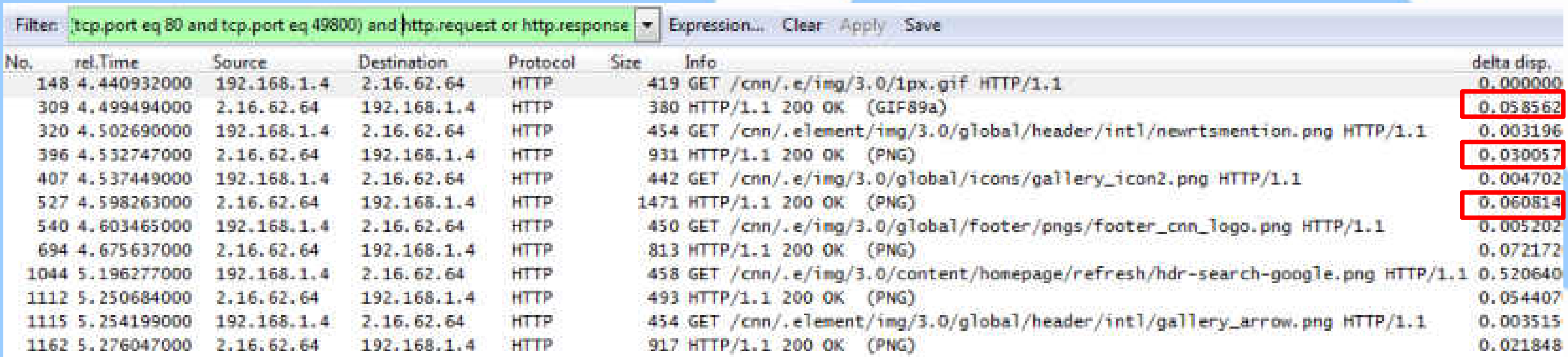
Let‘s do some network analysis

- Use case: Application Server Analysis
  - To be analyzed: Application response times
  - Simple with HTTP: delta time Request <> Response



Going from request to response

- Simple with delta displayed
  - Remember to filter for single TCP sessions before
  - Refer to Round-Trip-Time (RTT) for real application response time, depending where the capture was taken

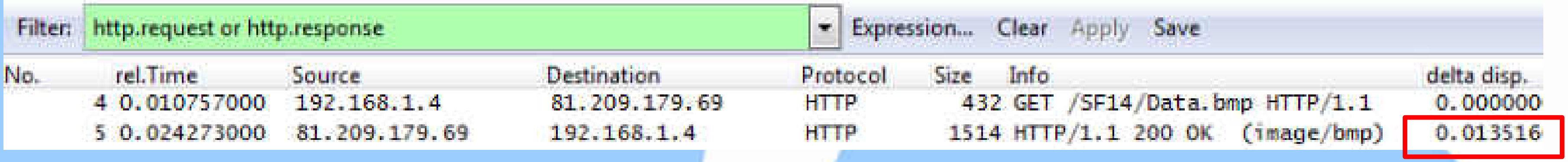


Filter: (tcp.port eq 80 and tcp.port eq 49800) and http.request or http.response

No.	rel.Time	Source	Destination	Protocol	Size	Info	delta disp.
148	4.440932000	192.168.1.4	2.16.62.64	HTTP	419	GET /cnn/.e/img/3.0/1px.gif HTTP/1.1	0.000000
309	4.499494000	2.16.62.64	192.168.1.4	HTTP	380	HTTP/1.1 200 OK (GIF89a)	0.058562
320	4.502690000	192.168.1.4	2.16.62.64	HTTP	454	GET /cnn/.element/img/3.0/global/header/intl/newrtsmention.png HTTP/1.1	0.003196
396	4.532747000	2.16.62.64	192.168.1.4	HTTP	931	HTTP/1.1 200 OK (PNG)	0.030057
407	4.537449000	192.168.1.4	2.16.62.64	HTTP	442	GET /cnn/.e/img/3.0/global/icons/gallery_icon2.png HTTP/1.1	0.004702
527	4.598263000	2.16.62.64	192.168.1.4	HTTP	1471	HTTP/1.1 200 OK (PNG)	0.060814
540	4.603465000	192.168.1.4	2.16.62.64	HTTP	450	GET /cnn/.e/img/3.0/global/footer/pngs/footer_cnn_logo.png HTTP/1.1	0.005202
694	4.675637000	2.16.62.64	192.168.1.4	HTTP	813	HTTP/1.1 200 OK (PNG)	0.072172
1044	5.196277000	192.168.1.4	2.16.62.64	HTTP	458	GET /cnn/.e/img/3.0/content/homepage/refresh/hdr-search-google.png HTTP/1.1	0.520640
1112	5.250684000	2.16.62.64	192.168.1.4	HTTP	493	HTTP/1.1 200 OK (PNG)	0.054407
1115	5.254199000	192.168.1.4	2.16.62.64	HTTP	454	GET /cnn/.element/img/3.0/global/header/intl/gallery_arrow.png HTTP/1.1	0.003515
1162	5.276047000	2.16.62.64	192.168.1.4	HTTP	917	HTTP/1.1 200 OK (PNG)	0.021848

How about our important data?

- Check webserver application response time



Filter: http.request or http.response

```
No.      rel.Time      Source          Destination     Protocol  Size  Info                                delta disp.
      4 0.010757000  192.168.1.4     81.209.179.69    HTTP       432 GET /SF14/Data.bmp HTTP/1.1        0.000000
      5 0.024273000  81.209.179.69   192.168.1.4      HTTP      1514 HTTP/1.1 200 OK  (image/bmp)       0.013516
```

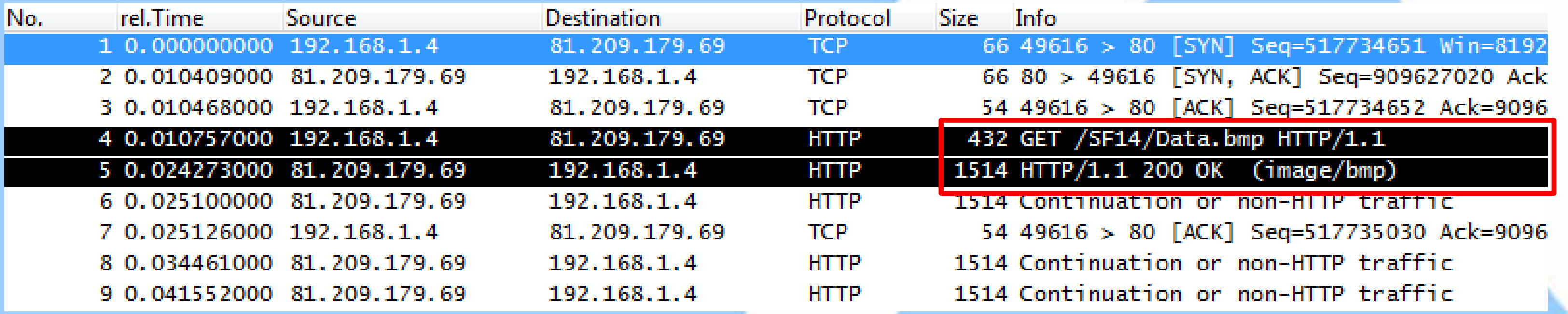
That‘s a fast one !!

Questions up to here?

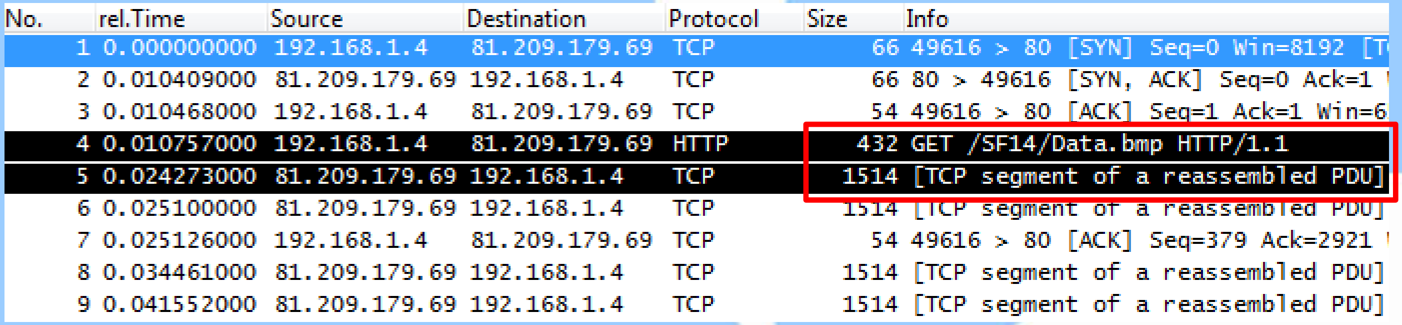
- Everybody agrees on the timings? (roughly if captured by yourself)
- Anyone having strange behavior with his/her Wireshark version?

That‘s where reassembly kicks in

- Watch the difference:



No.	rel.Time	Source	Destination	Protocol	Size	Info
1	0.000000000	192.168.1.4	81.209.179.69	TCP	66	49616 > 80 [SYN] Seq=517734651 Win=8192
2	0.010409000	81.209.179.69	192.168.1.4	TCP	66	80 > 49616 [SYN, ACK] Seq=909627020 Ack
3	0.010468000	192.168.1.4	81.209.179.69	TCP	54	49616 > 80 [ACK] Seq=517734652 Ack=9096
4	0.010757000	192.168.1.4	81.209.179.69	HTTP	432	GET /SF14/Data.bmp HTTP/1.1
5	0.024273000	81.209.179.69	192.168.1.4	HTTP	1514	HTTP/1.1 200 OK (image/bmp)
6	0.025100000	81.209.179.69	192.168.1.4	HTTP	1514	Continuation or non-HTTP traffic
7	0.025126000	192.168.1.4	81.209.179.69	TCP	54	49616 > 80 [ACK] Seq=517735030 Ack=9096
8	0.034461000	81.209.179.69	192.168.1.4	HTTP	1514	Continuation or non-HTTP traffic
9	0.041552000	81.209.179.69	192.168.1.4	HTTP	1514	Continuation or non-HTTP traffic



No.	rel.Time	Source	Destination	Protocol	Size	Info
1	0.000000000	192.168.1.4	81.209.179.69	TCP	66	49616 > 80 [SYN] Seq=0 Win=8192 [T
2	0.010409000	81.209.179.69	192.168.1.4	TCP	66	80 > 49616 [SYN, ACK] Seq=0 Ack=1
3	0.010468000	192.168.1.4	81.209.179.69	TCP	54	49616 > 80 [ACK] Seq=1 Ack=1 Win=6
4	0.010757000	192.168.1.4	81.209.179.69	HTTP	432	GET /SF14/Data.bmp HTTP/1.1
5	0.024273000	81.209.179.69	192.168.1.4	TCP	1514	[TCP segment of a reassembled PDU]
6	0.025100000	81.209.179.69	192.168.1.4	TCP	1514	[TCP segment of a reassembled PDU]
7	0.025126000	192.168.1.4	81.209.179.69	TCP	54	49616 > 80 [ACK] Seq=379 Ack=2921
8	0.034461000	81.209.179.69	192.168.1.4	TCP	1514	[TCP segment of a reassembled PDU]
9	0.041552000	81.209.179.69	192.168.1.4	TCP	1514	[TCP segment of a reassembled PDU]

Side-Effects within TCP Reassembly

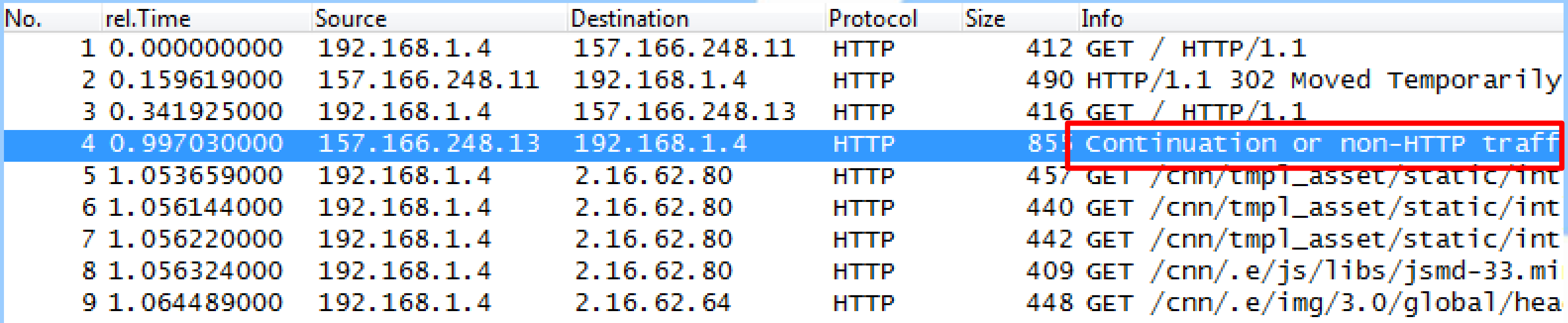
- Possible Re-Ordering of INFO-Column statements within the packet list
- Affects display filters too (e.g. http.response)
- Changes to the labeling of the „protocol“ column within Wireshark

→ Also possibly affects display filters, statistics etc.

Side-Note: Wireshark Bugs #1?

- Filter for all HTTP request and HTTP responses

  → GUI export or tshark
- Save into new capture file and open for analysis



No.	rel.Time	Source	Destination	Protocol	Size	Info
1	0.000000000	192.168.1.4	157.166.248.11	HTTP	412	GET / HTTP/1.1
2	0.159619000	157.166.248.11	192.168.1.4	HTTP	490	HTTP/1.1 302 Moved Temporarily
3	0.341925000	192.168.1.4	157.166.248.13	HTTP	416	GET / HTTP/1.1
4	0.997030000	157.166.248.13	192.168.1.4	HTTP	855	Continuation or non-HTTP traff
5	1.053659000	192.168.1.4	2.16.62.80	HTTP	457	GET /cnn/tmpl_asset/static/int
6	1.056144000	192.168.1.4	2.16.62.80	HTTP	440	GET /cnn/tmpl_asset/static/int
7	1.056220000	192.168.1.4	2.16.62.80	HTTP	442	GET /cnn/tmpl_asset/static/int
8	1.056324000	192.168.1.4	2.16.62.80	HTTP	409	GET /cnn/.e/js/libs/jsmd-33.mi
9	1.064489000	192.168.1.4	2.16.62.64	HTTP	448	GET /cnn/.e/img/3.0/global/hea

Side-Note: Wireshark Bugs #2?

- Check the protocol hierarchy statistics
- Watch for HTTP percentage
- Try to explain the different results based on reassembly setting

No bugs of course!

- Yet more side-effects of reassembly
- Valid output, but strongly dependent on the question you ask:
  - Time until start OR end of data stream delivery
  - Statistics of ALL HTTP-related packets, meaning tcp.port==80

    OR

    All HTTP-related packets containg data (without ACKs, Handshake etc.)

    OR

    Just the Requests and Response packets

Best practices

- Watch carefully !
- Use separate Profiles
  - Turn off reassembly for any timing / statistics based analysis tasks
  - Turn on reassembly for content analysis / forensics
- Check your default profile, since it is the base setting for tshark on command line level

!! Thank you for your attention !!

Q / A