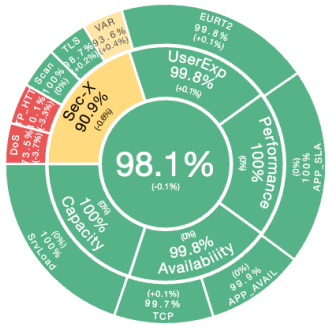
Sharkmon

Monitor your PCAP Data

- anything and everywhere

Andreas Diedrich

Packetfest'25



InterView

network solutions

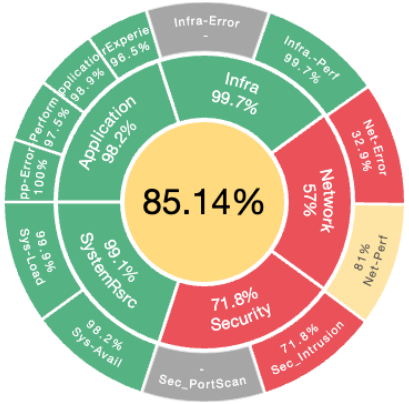
since 1996

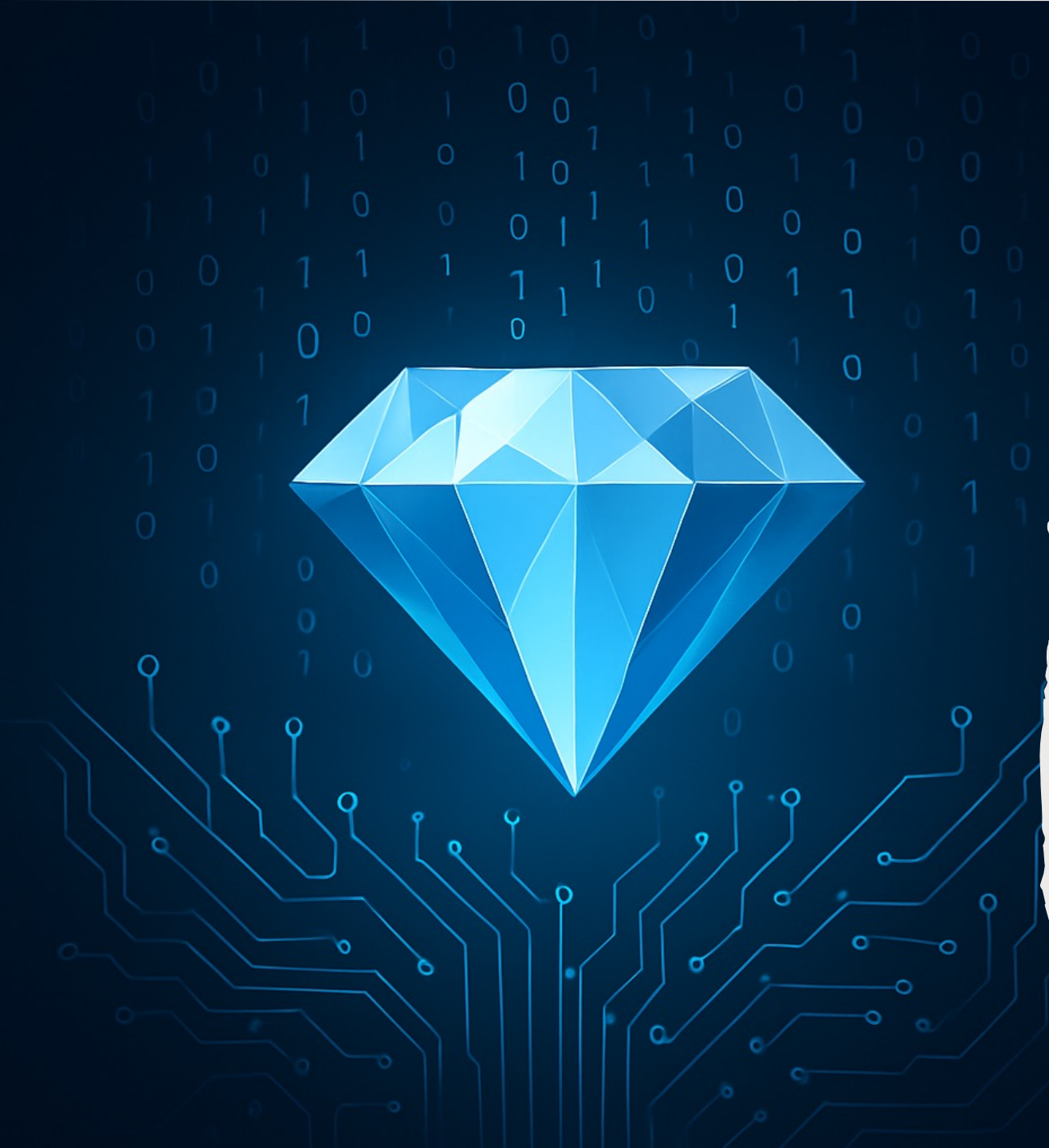
DPA
network packet analysis

IT Service management

Since 2015

Working on Integration





Why we are here

- **Wireshark / Tshark**
  - **Wireshark / Tshark is not just a tool**
    - **it is the diamond of packet data**
    - **The global reference for packet analysis**
  - 3.000 protocols
  - 317.000 display filter = 317.000 metrics !
  - Each of these fields including data and/or information
- **Usecases**
  - Network, infrastructure , application, performance, incidents, security, industry, finance, health, Telco...
  - Constantly new protocols
    - (last 4 years - UIC, HTTP/3, MASQUE, STUN, TURN, ICE, SCTP, WebTransport,RIFT, Flowspec v2 amo.)
  - And daily new attack types (ca. 30.000 per year)

Not using tshark for packet analysis monitoring does create immediately gaps

Lets start – tshark for Monitoring

anyone can do it !

Tshark Data delivery

- Each of **317.000 filter fields** can be selected
- **RingBuffer capture** – eg. every minute one file – can cover days, weeks, months, years
- 100s of different data fields in a single command
- various data aggregation schemes (COUNT, MAX, AVG…)
- analysis/inter-frame calculation
  - Retransmisisons
  - Timing (RTO,iRTT, DNS Time…)
  - experts
- export to standard formats (json, CSV…)
  - Every minute one
  - 1000 Pcap files -> 1.000 Json Files
    - Export DB

Protocol	# Fields
Arp	55
BGP	910
BootP	530
DHCP	591
DLSW	71
DNS	378
ETH	37
FTP	38
H323	45
Hsrp	44
http	107
http2	162
Icmp	108
Imap	18
IP	140
Ipv6	384
Mpls	24
mySQL	292
Nbns	52
Netbios	53
Nfs	691
NTP	346
OSFP	337
QUIC	322
…	…

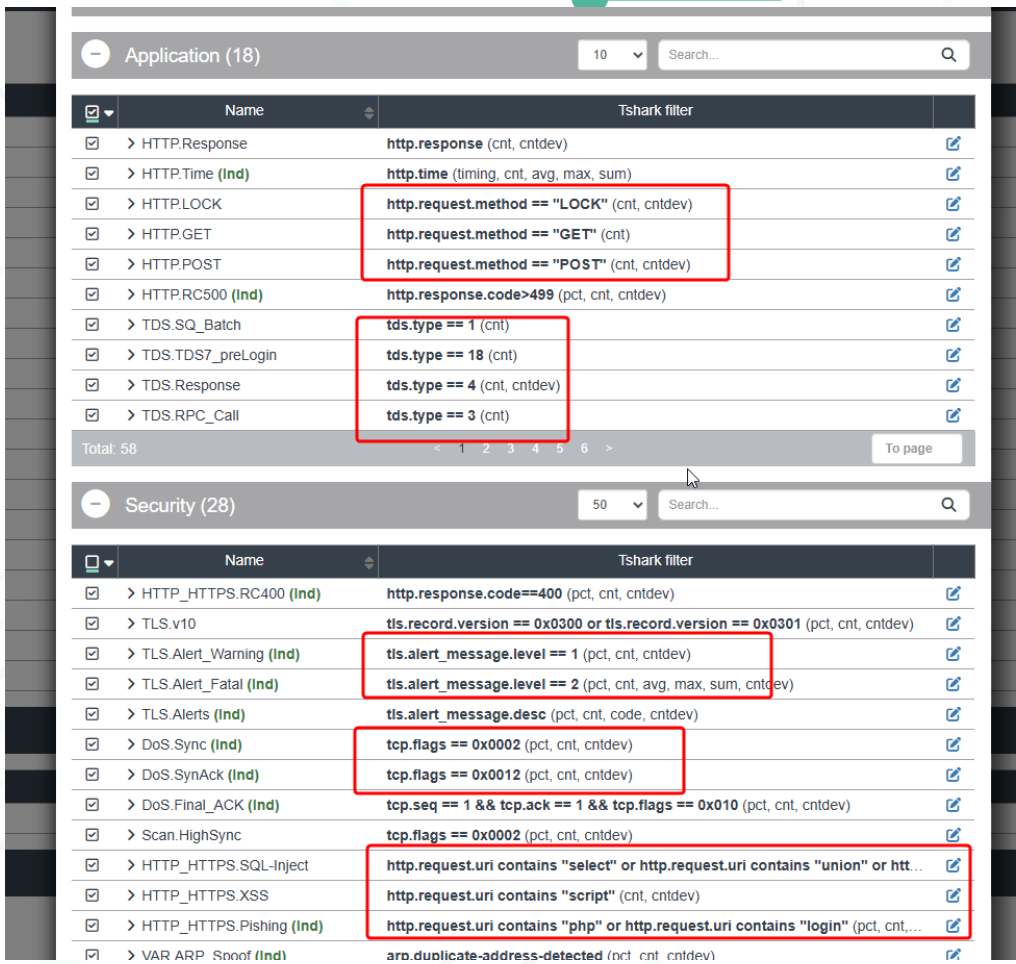
Standard Requirements for DPA Monitoring

- Potentially all Protocols (Tshark logic filter) – user decides !!
- All fields per protocol – user decides !!
- Database for longtime data store
- Incident detection
- Alerting
- Anomaly
- Reporting
- **The rest is “private” decision**
  - Plattform
  - Indexing
  - Dashboards
  - Value

Tshark Commands

1. **Pcap file Summay** - capinfos -TRB trace.pcap
2. **Conversations** - tshark -qr trace.pcap -z conv,ip
3. **Protocols -** - tshark -qr trace.pcap -z io,phs
4. **--z io,stat, data statistisc**
    1. **incl. free selectable granularity, aggregation (avg(MIN/MAX/SUM)**
    2. tshark -qr **trace.pcap** –z io,stat,**1**,"COUNT(tcp)tcp","AVG(tcp.analysis.rto)tcp.analysis.rto“ ….
5. **„-T fields –e“ every packet data field**
    1. tshark -2 --enable-protocol transum -qnlr "YOUR_PCAP_TO_REPLACE.pcap" -t u -Tfields -E header=y -e frame.protocols -e frame.time_epoch -e frame.len -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport -e udp.srcport -e udp.dstport -e http.response -e tls.record.version -e tls.record.content_type -e tls.handshake.type -e transum.art -e dns.time -e dns.flags.rcode -e http.time -e http.request.method -e http.response.code -e tcp.flags -e tcp.analysis.zero_window -e tcp.analysis.window_full -e tcp.flags.reset -e ip.ttl -e tcp.flags.fin -e tds.type -e tls.alert_message.level -e tls.alert_message.desc -e tls.app_data_proto -e icmp.code -e smb2.nt_status -e tcp.analysis.rto -e tcp.analysis.retransmission -e smb2.time -e ntp.flags.mode -e ntp.flags.li -e tcp.analysis.duplicate_ack -e tds.status.reset_conn -e sip.Status-Code -e http.request.uri -e quic.long.packet_type -e quic.ack.gap -e tcp.flags.syn -e tcp.flags.ack -e tcp.analysis.fast_retransmission -e arp.duplicate-address-detected -e dns.qry.name -e tcp.port -e http.request.full_uri -e tls.record.length -e smtp.req.parameter -e ftp.request.command -e tcp.payload -e tcp.analysis.initial_rtt -e tcp.seq -e tcp.ack -e tcp.time_relative -e http.response.code.desc -e http.request -e icmp.type > "YOUR_SCV_TO_REPLACE.csv

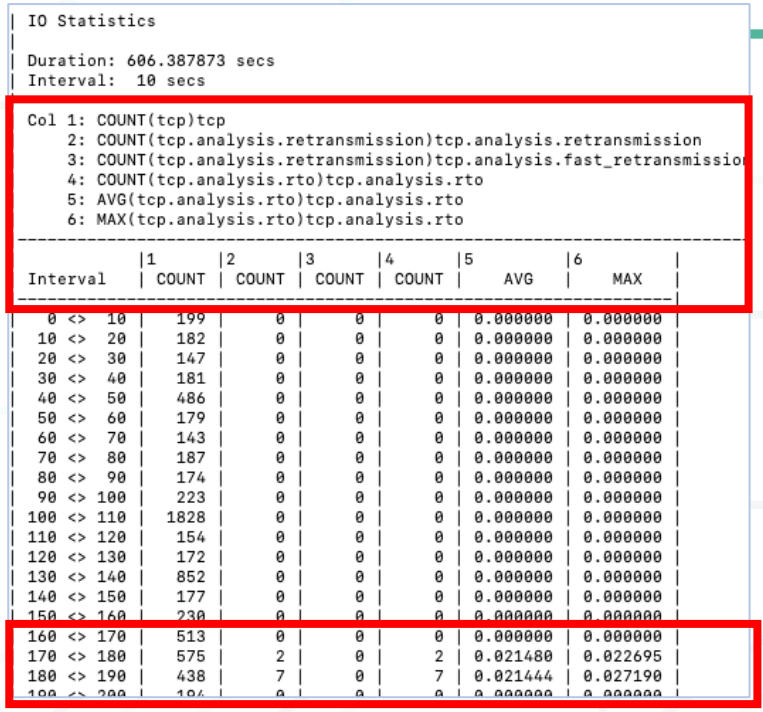
- example 125 Metrics
- creates a record for each packet – huge amount of data !
- for us biggest challenge / last way



Tshark Data commands

What –z io,stat,

- Tshark -qr trace.pcap -z io, stat, 10, "COUNT(tcp)tcp","COUNT(tcp.analysis.retransmission)tcp.analysis.retransmission","COUNT(tcp.analysis.retransmission)tcp.analysis.fast_retransmission","COUNT(tcp.analysis.rto)tcp.analysis.rto","AVG(tcp.analysis.rto)tcp.analysis.rto","MAX(tcp.analysis.rto)tcp.analysis.rto“
- Benefits : very clear data, data volume is small / each time span one line
- Challenge : no IP addresses / Ports assigment



```
| IO Statistics
|
| Duration: 606.387873 secs
| Interval:  10 secs

Col 1: COUNT(tcp)tcp
    2: COUNT(tcp.analysis.retransmission)tcp.analysis.retransmission
    3: COUNT(tcp.analysis.retransmission)tcp.analysis.fast_retransmissio
    4: COUNT(tcp.analysis.rto)tcp.analysis.rto
    5: AVG(tcp.analysis.rto)tcp.analysis.rto
    6: MAX(tcp.analysis.rto)tcp.analysis.rto
-------------------------------------------------------------------------------
              |1       |2       |3       |4       |5          |6          |
Interval      | COUNT  | COUNT  | COUNT  | COUNT  |    AVG    |    MAX    |
-------------------------------------------------------------------------------
  0 <=  10    |   199  |     0  |     0  |     0  | 0.000000  | 0.000000  |
 10 <=  20    |   182  |     0  |     0  |     0  | 0.000000  | 0.000000  |
 20 <=  30    |   147  |     0  |     0  |     0  | 0.000000  | 0.000000  |
 30 <=  40    |   181  |     0  |     0  |     0  | 0.000000  | 0.000000  |
 40 <=  50    |   486  |     0  |     0  |     0  | 0.000000  | 0.000000  |
 50 <=  60    |   179  |     0  |     0  |     0  | 0.000000  | 0.000000  |
 60 <=  70    |   143  |     0  |     0  |     0  | 0.000000  | 0.000000  |
 70 <=  80    |   187  |     0  |     0  |     0  | 0.000000  | 0.000000  |
 80 <=  90    |   174  |     0  |     0  |     0  | 0.000000  | 0.000000  |
 90 <= 100    |   223  |     0  |     0  |     0  | 0.000000  | 0.000000  |
100 <= 110    |  1828  |     0  |     0  |     0  | 0.000000  | 0.000000  |
110 <= 120    |   154  |     0  |     0  |     0  | 0.000000  | 0.000000  |
120 <= 130    |   172  |     0  |     0  |     0  | 0.000000  | 0.000000  |
130 <= 140    |   852  |     0  |     0  |     0  | 0.000000  | 0.000000  |
140 <= 150    |   177  |     0  |     0  |     0  | 0.000000  | 0.000000  |
150 <= 160    |   230  |     0  |     0  |     0  | 0.000000  | 0.000000  |
160 <= 170    |   513  |     0  |     0  |     0  | 0.000000  | 0.000000  |
170 <= 180    |   575  |     2  |     0  |     2  | 0.021480  | 0.022695  |
180 <= 190    |   438  |     7  |     0  |     7  | 0.021444  | 0.027190  |
190 <= 200    |   194  |     0  |     0  |     0  | 0.000000  | 0.000000  |
```

Who and when requires –T fields

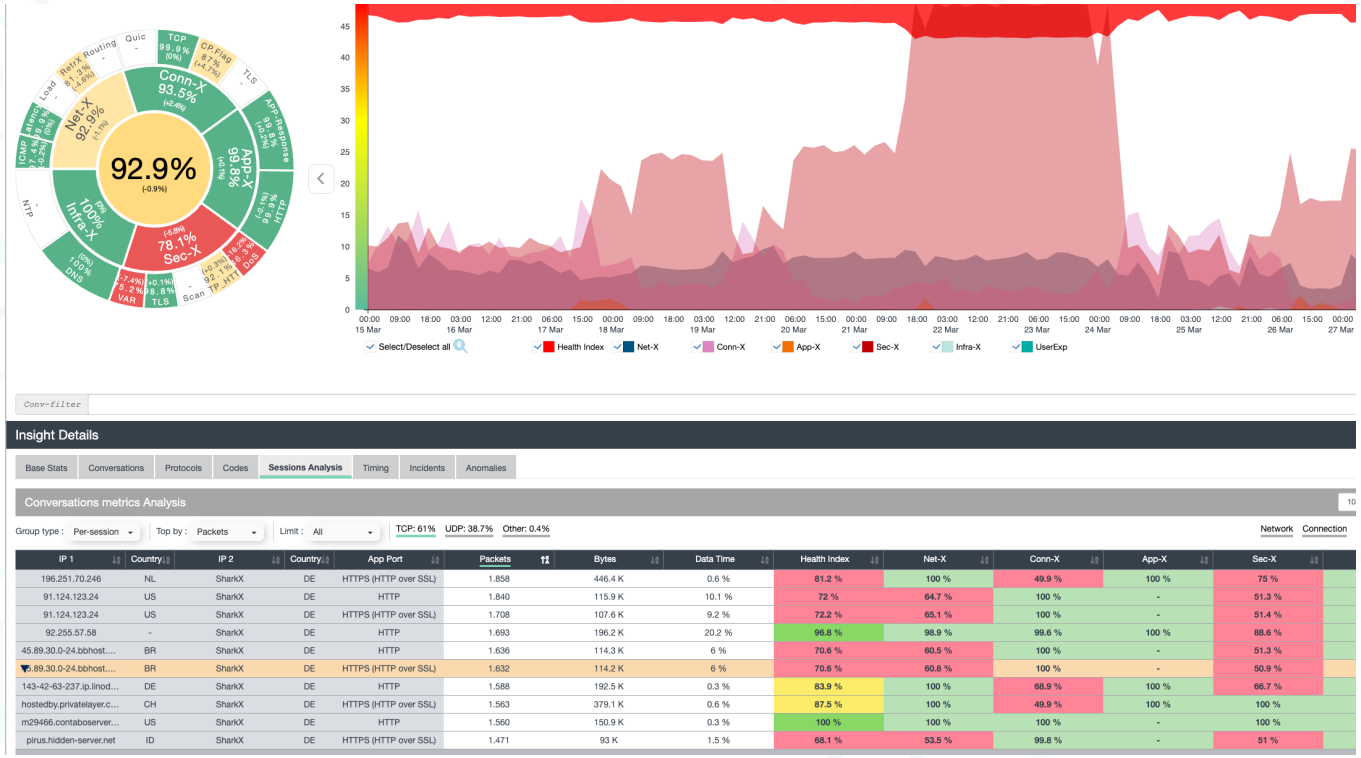
- Tshark -qr trace2.pcap -Y "tcp.analysis.retransmission" -T fields -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport -e tcp.analysis.retransmission -e tcp.analysis.fast_retransmission -e tcp.analysis.rto
- **Each packet = one line in export file**
- **10 Gbps = 1 GB/sec = 1 mio pps !!! = 1 Mio lines in export file**

```
192.168.178.37  17.248.209.72   64256  443    1                0.022825000
13.107.213.45   192.168.178.37  443    64202  1
192.168.178.37  148.251.120.247 64290  443    1                0.052445000
192.168.178.37  148.251.120.247 64290  443    1                0.092634000
192.168.178.37  148.251.120.247 64290  443    1                0.131608000
148.251.120.247 192.168.178.37  443    64290  1                0.000001000
192.168.178.37  148.251.120.247 64290  443    1                0.079033000
192.168.178.37  148.251.120.247 64306  443    1                0.039949000
192.168.178.37  148.251.120.247 64306  443    1                0.079401000
192.168.178.37  148.251.120.247 64306  443    1                0.067025000
148.251.120.247 192.168.178.37  443    64306  1                0.135705000
192.168.178.37  108.138.36.22   64409  443    1                0.031376000
192.168.178.37  108.138.36.34   64408  443    1                0.044714000
199.232.191.6   192.168.178.37  443    64436  1
```

t(shark)mon
data and results

- Tshark is the core of analysis
- Sharkmon is the „big data“ organizer – around tshark data

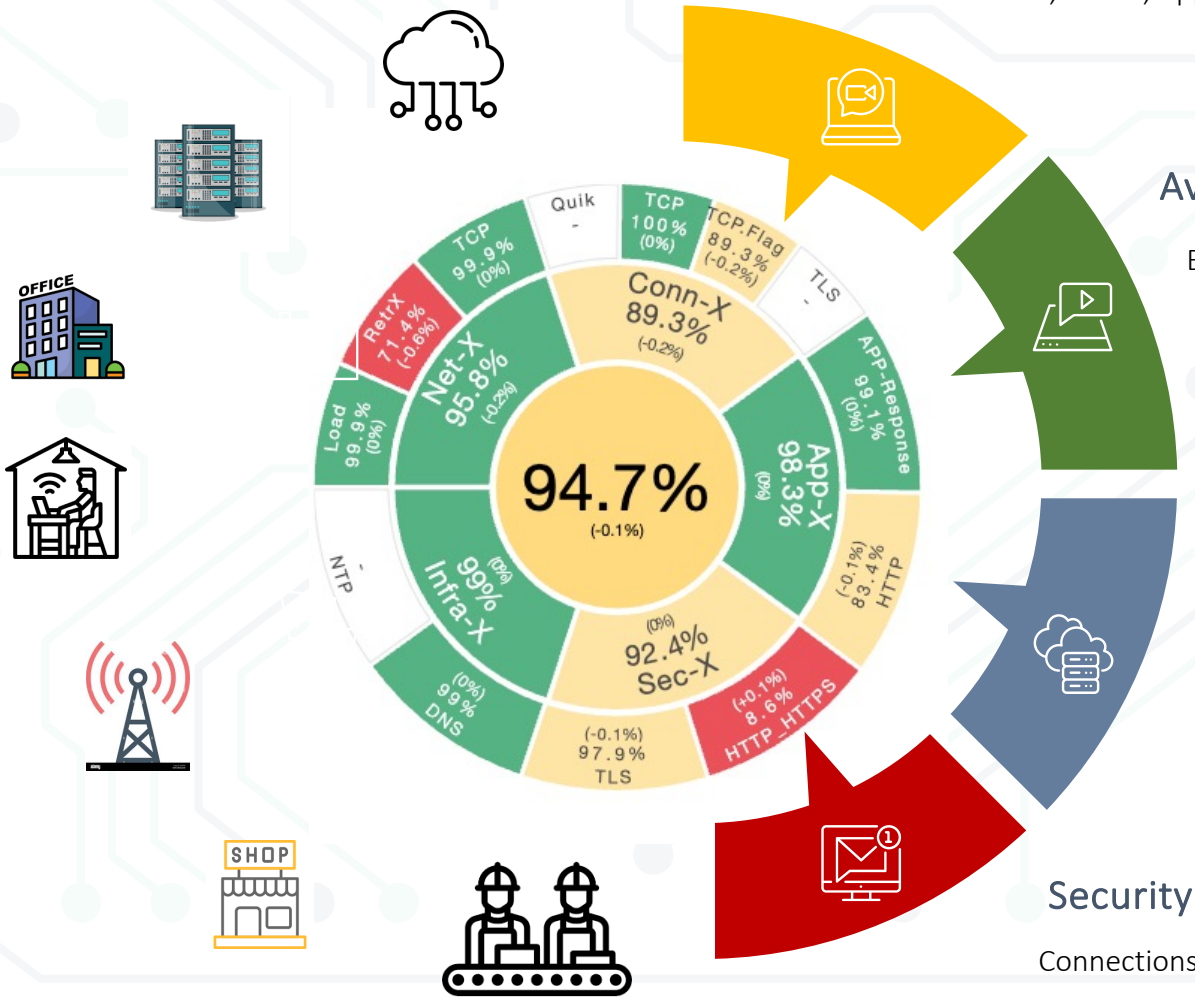
tshark -2 --enable-protocol transum -qnlr "YOUR_PCAP_TO_REPLACE.pcap" -t u –T fields -E header=y -e frame.protocols -e frame.time_epoch -e frame.len -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport -e udp.srcport -e udp.dstport -e http.response -e tls.record.version -e tls.record.content_type -e tls.handshake.type -e transum.art -e dns.time -e dns.flags.rcode -e http.time -e http.request.method -e http.response.code -e tcp.flags -e tcp.analysis.zero_window -e tcp.analysis.window_full -e tcp.flags.reset -e ip.ttl -e tcp.flags.fin -e tds.type -e tls.alert_message.level -e tls.alert_message.desc -e tls.app_data_proto -e icmp.code -e smb2.nt_status -e tcp.analysis.rto -e tcp.analysis.retransmission -e smb2.time -e ntp.flags.mode -e ntp.flags.li -e tcp.analysis.duplicate_ack -e tds.status.reset_conn -e sip.Status-Code -e http.request.uri -e quic.long.packet_type -e quic.ack.gap -e tcp.flags.syn -e tcp.flags.ack -e tcp.analysis.fast_retransmission -e arp.duplicate-address-detected -e dns.qry.name -e tcp.port -e http.request.full_uri -e tls.record.length -e smtp.req.parameter -e ftp.request.command -e tcp.payload -e tcp.analysis.initial_rtt -e tcp.seq -e tcp.ack -e tcp.time_relative -e http.response.code.desc -e http.request -e icmp.type > "YOUR_SCV_TO_REPLACE.csv"



Sharkmon

anything – everywhere

- All known protocols
- Customizable fields
- Customizable indexing



Performance

Network, Server, Application, Infrastructure

Availability

Elements, services

Correctness

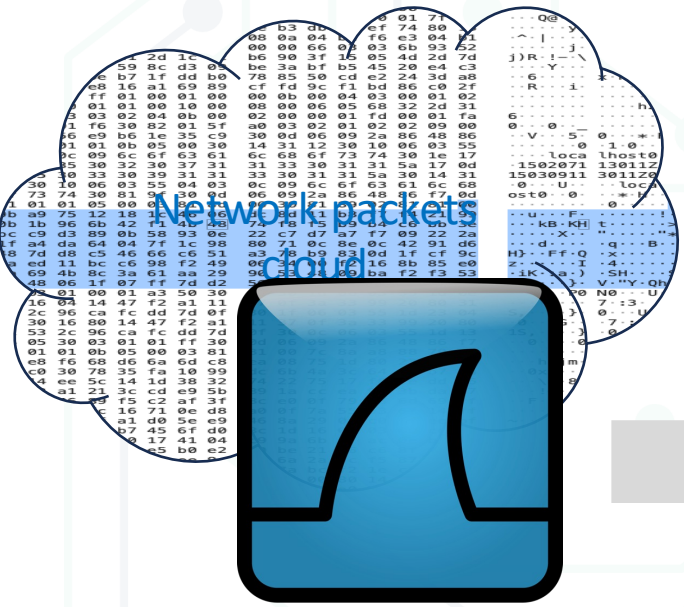
Codes, response, behaviour

Security

Connections, requests, anomaly

The sharkmon

from data - to results



Tshark Packet Processing / Analysis

- Calculation
- Identification
- Filter
- Data export

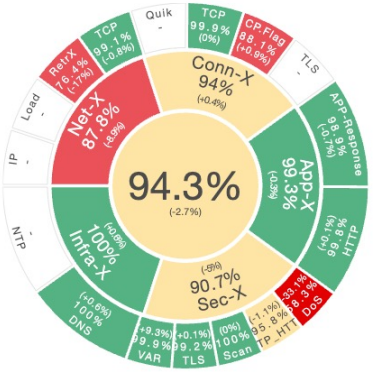
Sharkmon Processing

- Data Import
- Indexing
- alerting
- Database

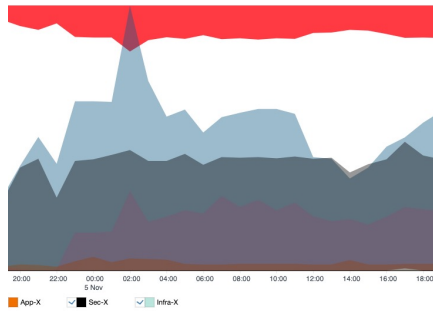
Where

What

When

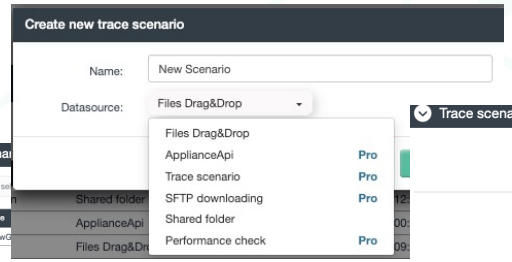


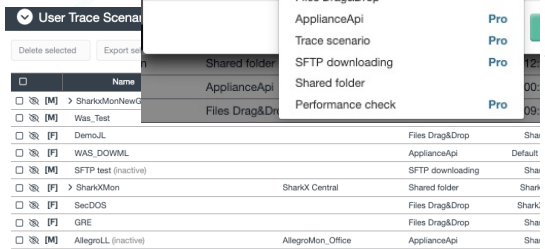
How

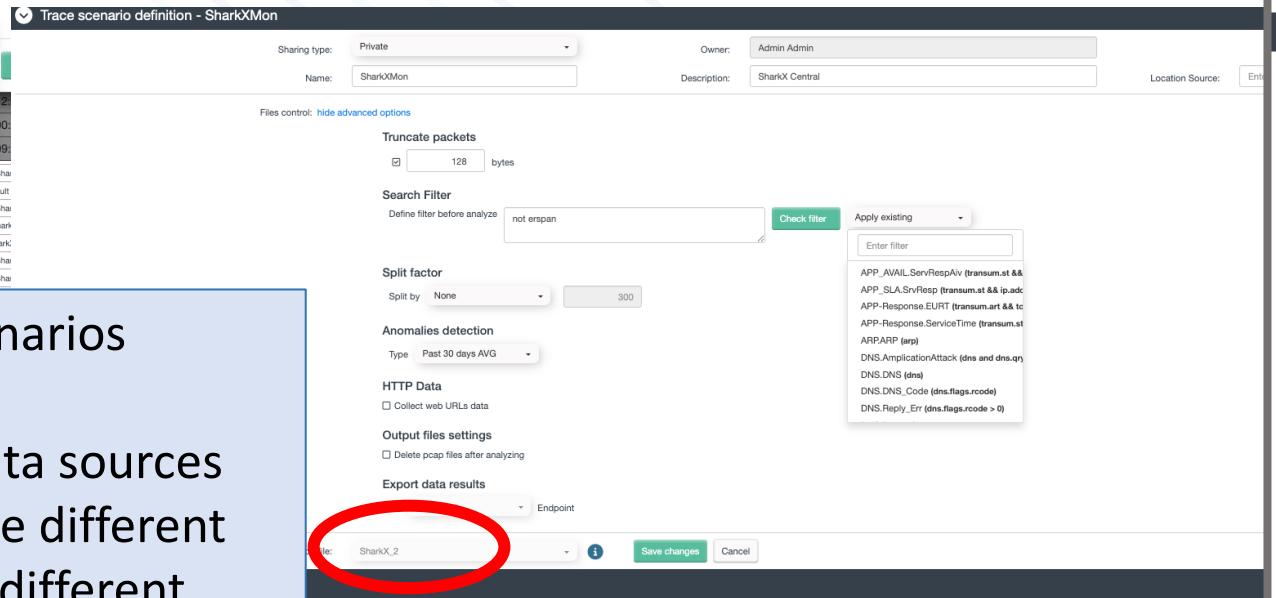


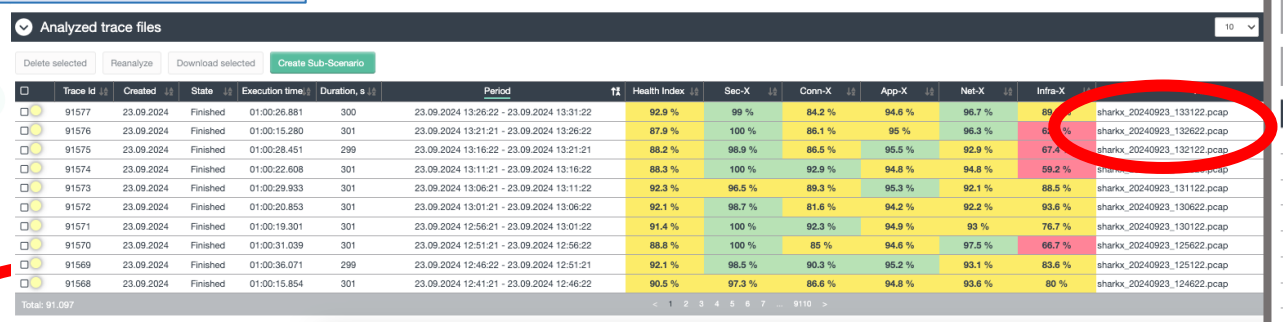
Scenario / Profile

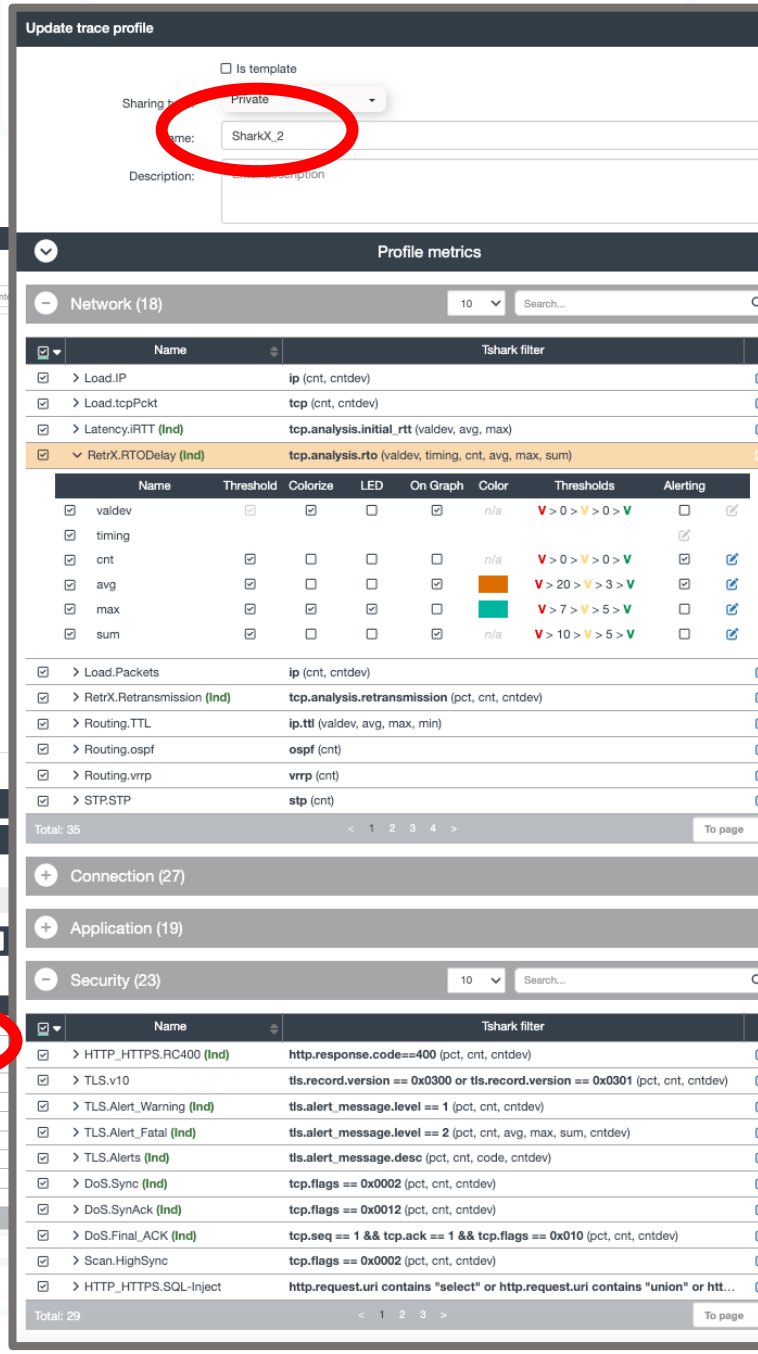
- Multiple parallel scenarios possible
- Multiple different data sources
- Each scenario can use different analysis profile with different categories / classes / thresholds











Distributed Data Import

Monitoring Auto-Import

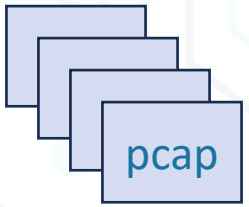
Capture Appliance API

Server OS Tcpdump/ringbuffer

sFTP auto Import

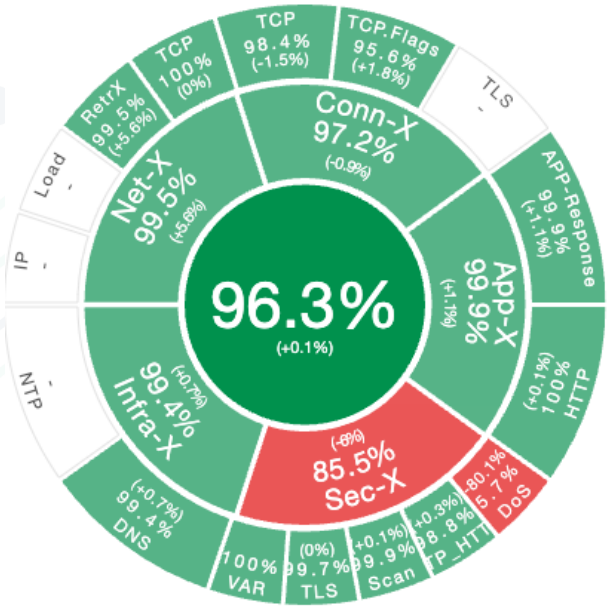
Manual upload

sFTP Put



Drag & drop

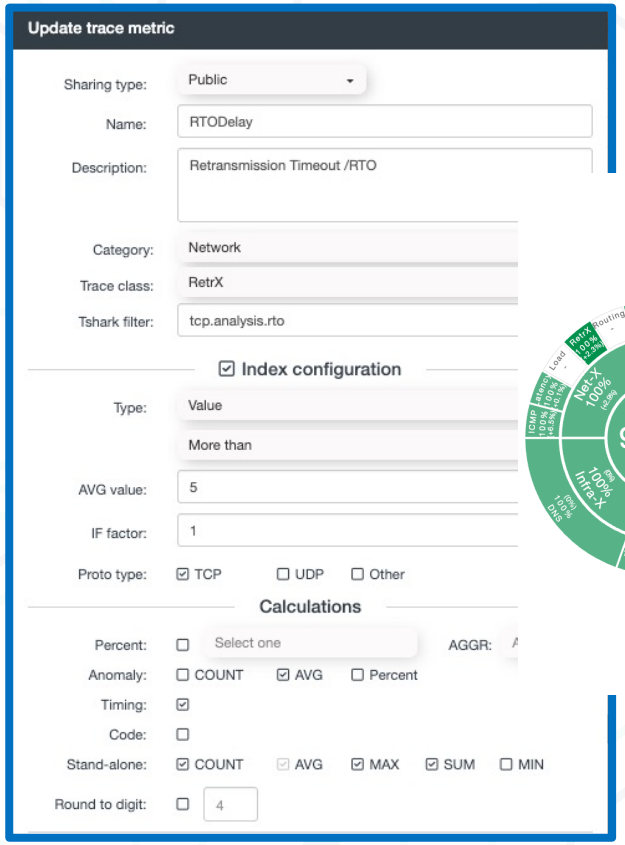
- Parallel data import
- Multiple datasources
- Multiple locations

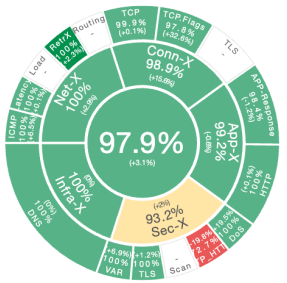


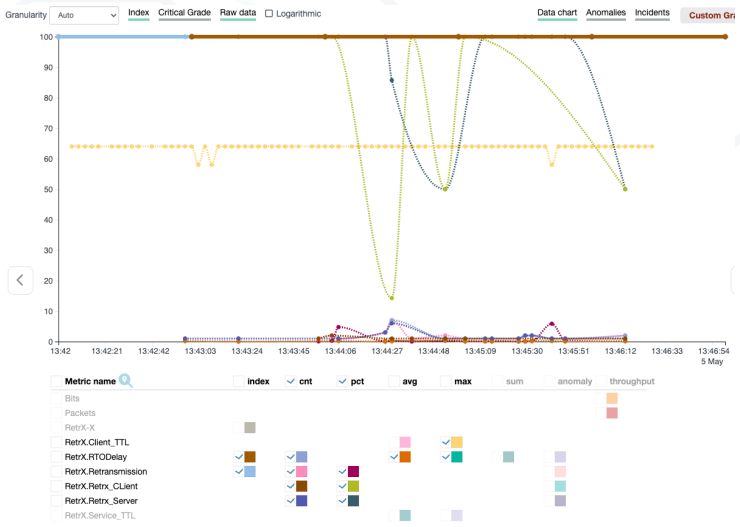
t(Shark)mon data

Sharkmon Profile Metrics

- Tshark filter field
- Tresholds
- Naming
- **Category / Class assignment**
- **Indexing**
- **Anomaly**
- MIN/AVG/MAX Values per sec
- COUNT / pSec
- Percent – can use any reference (retrx Per TLS.Sync)
- Codes
  - Tls alert, icmp type, dns response, HTTP reponse – all Codes use
- Timing





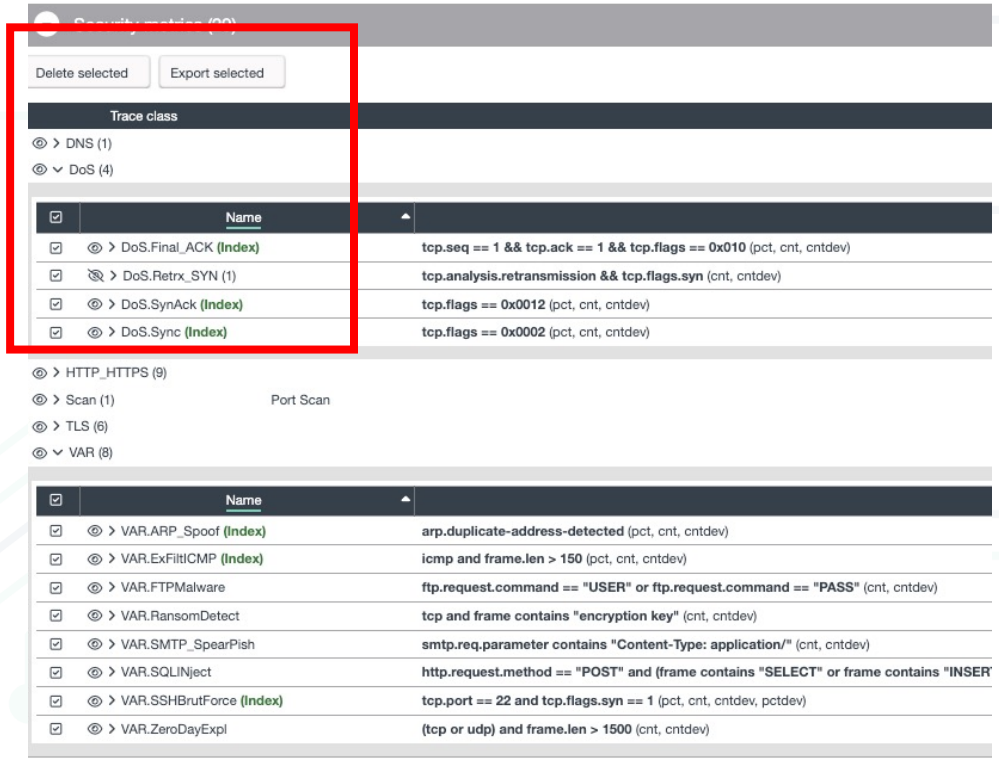


Metric what and how

- 317.000 possible fields – nobody can know all !
- So lets use - what We know – and what others know

Metrics can be shared:
- Export / import

Metric author
- Default set
- User defined own extension
- ChatGPT assistance
- External consultants
- In Future may be reference site

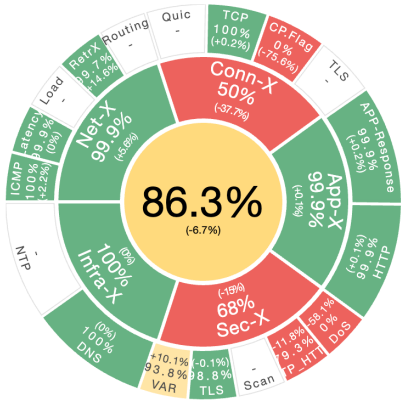


Indexing – why and How AND WHAT !

What is the TOP JOB of Monitoring / analysis ?

- Find the BAD ! – its effects – and ist causes !

- DPA does cover all levels of IT – from most basic transport till final transaction = 7 layers
- **Each layer has often different support team, different skills – and different responsibility** !
- And layers are connected by its effects ! Packetloss & latency may causing bad App responsetime
- Indexing can help indicate
  - what of the much data is critical : the category, the class – and the metric
  - can show clear – who / what was affected !
  - Categories are defined by ist own usecase

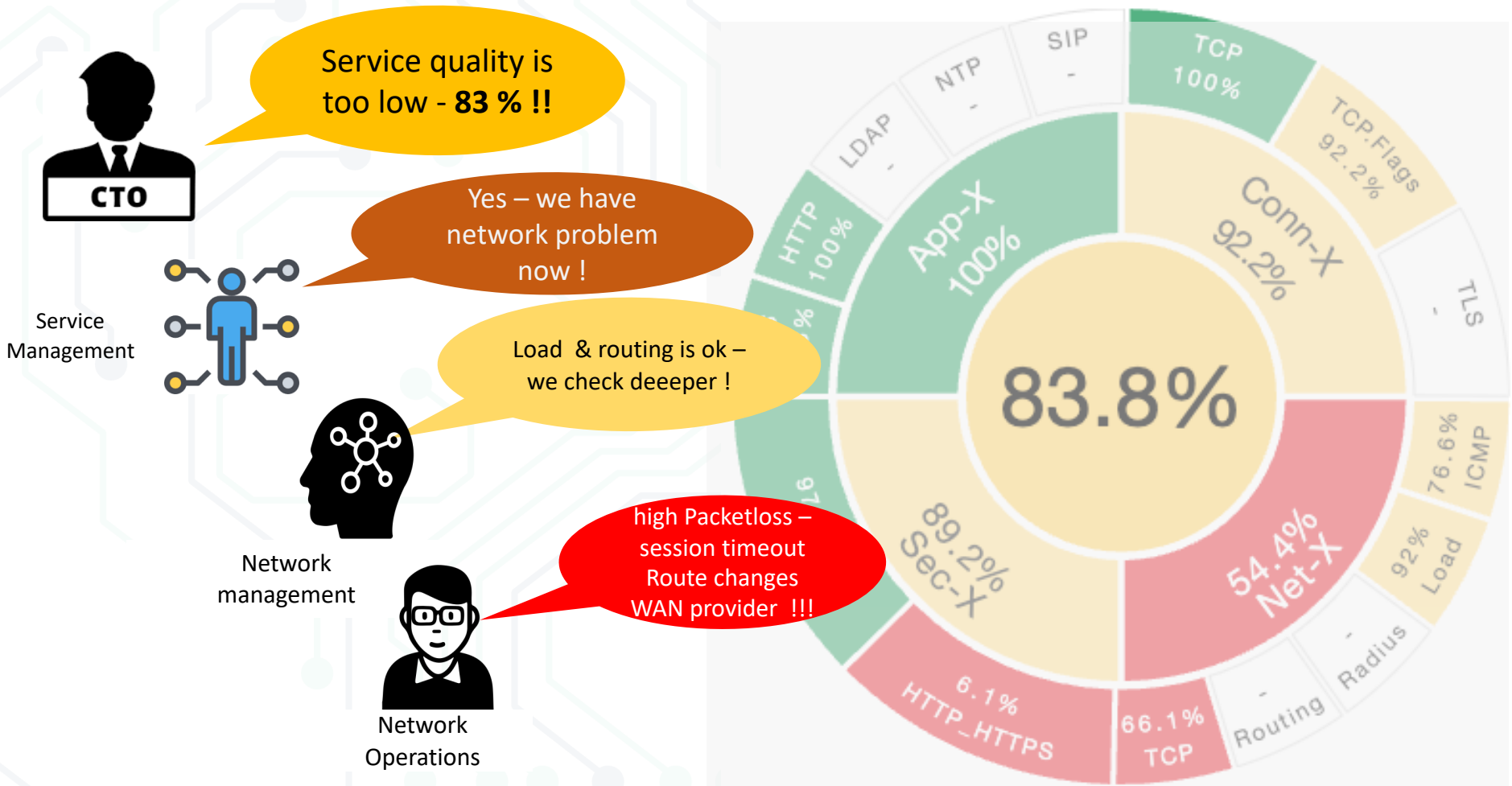


86 % – is this good or bad ?

Indexing value by itself is limited

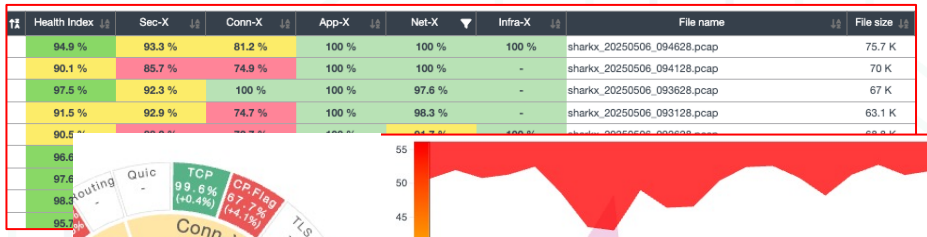
- 92% today - how was yesterday ? Oh 75% ? We are good !
  - Oh 98 % ? We are bad !
- it needs history – comparison
- And user definable thresholds

Indexing

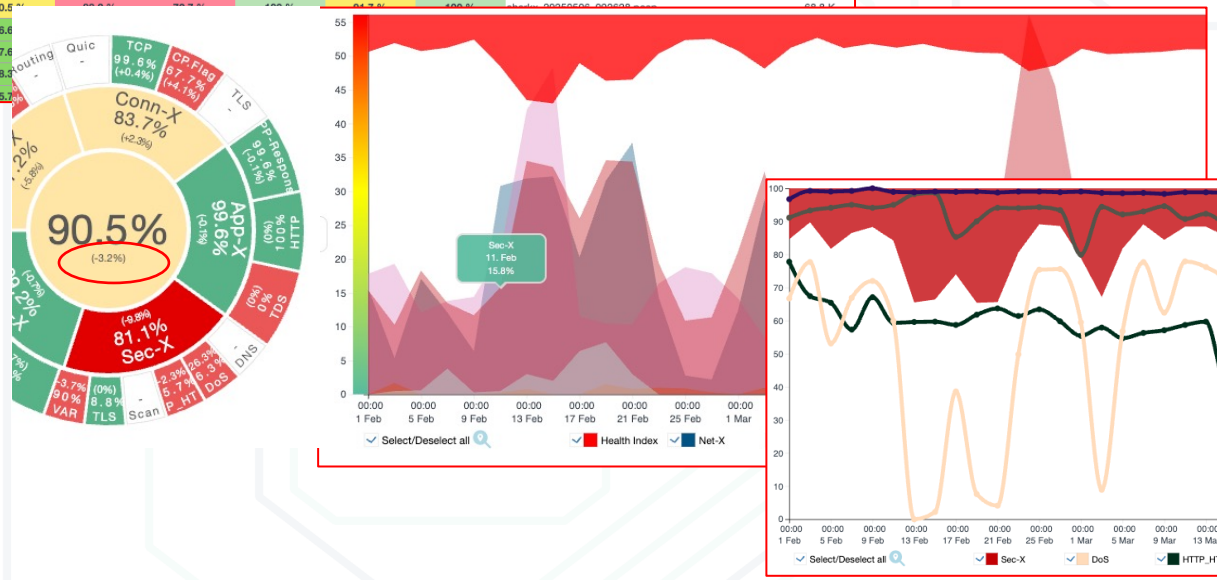

Service quality is too low - **83 % !!**
CTO
Yes – we have network problem now !
Service Management
Load & routing is ok – we check deeeper !
Network management
high Packetloss – session timeout Route changes WAN provider !!!
Network Operations
83.8%
App-X 100%
Conn-X 92.2%
Net-X 54.4%
Sec-X 89.2%
TCP 100%
TCP.Flags 92.2%
TLS -
ICMP 76.6%
Load 92%
Radius -
Routing -
TCP 66.1%
HTTP_HTTPS 6.1%
HTTP 100%
LDAP -
NTP -
SIP -

Indexing objects

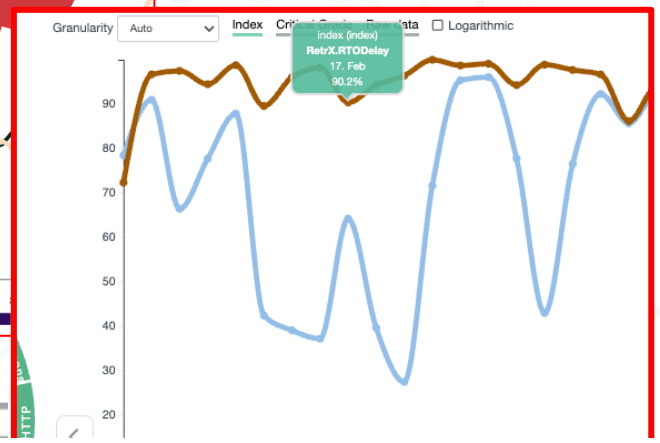
Per PCAP File



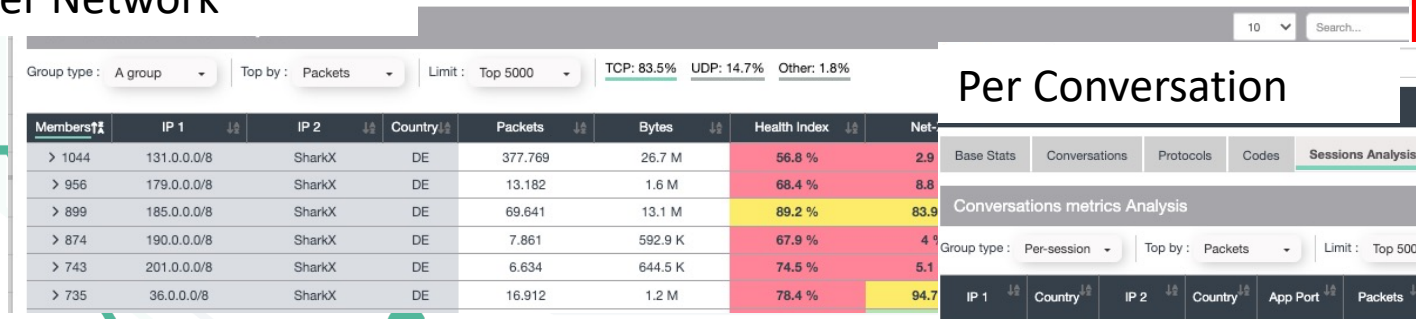
Per category / Class



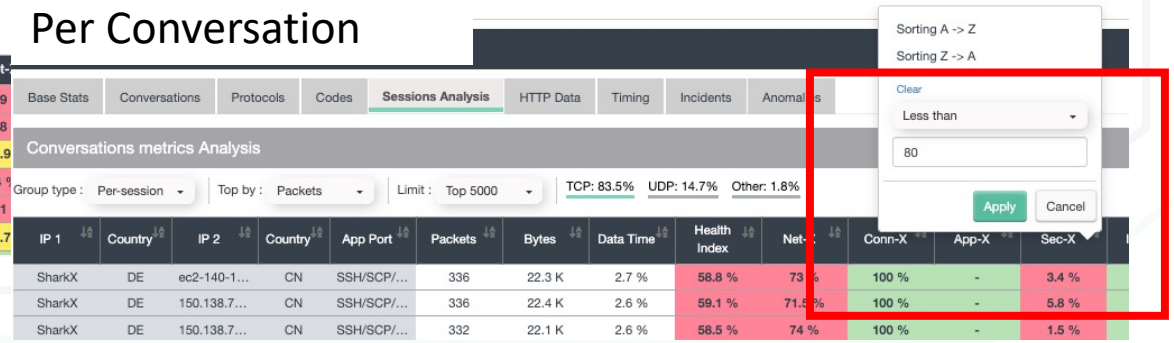
Per Metric



Per Network



Per Conversation

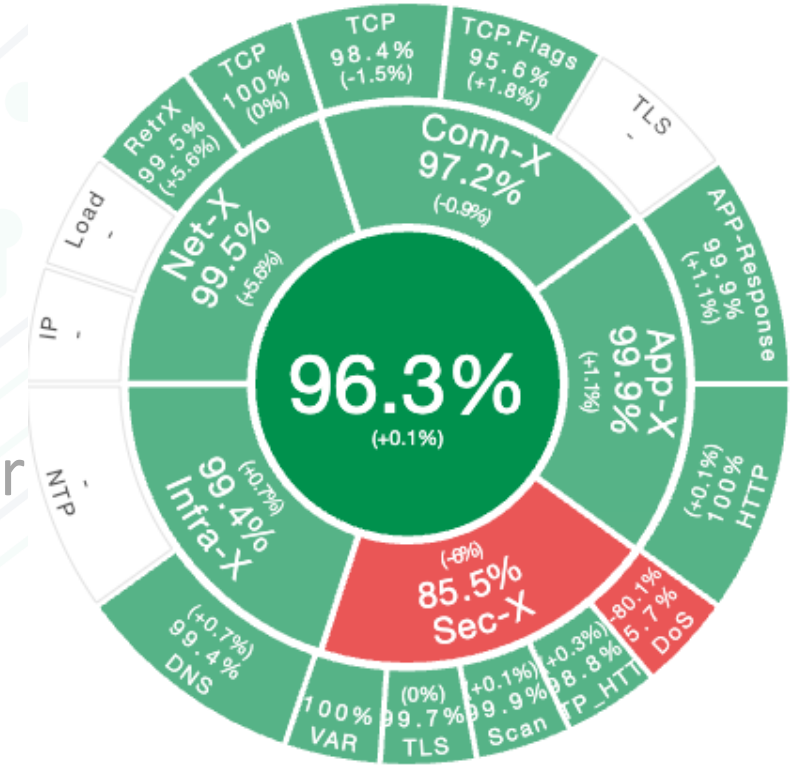


Different Perspectives

the round table view

- Network
- Server
- Application
- Infrastructure
- Security

WAN
LAN
Cloud
Provider
SaaS



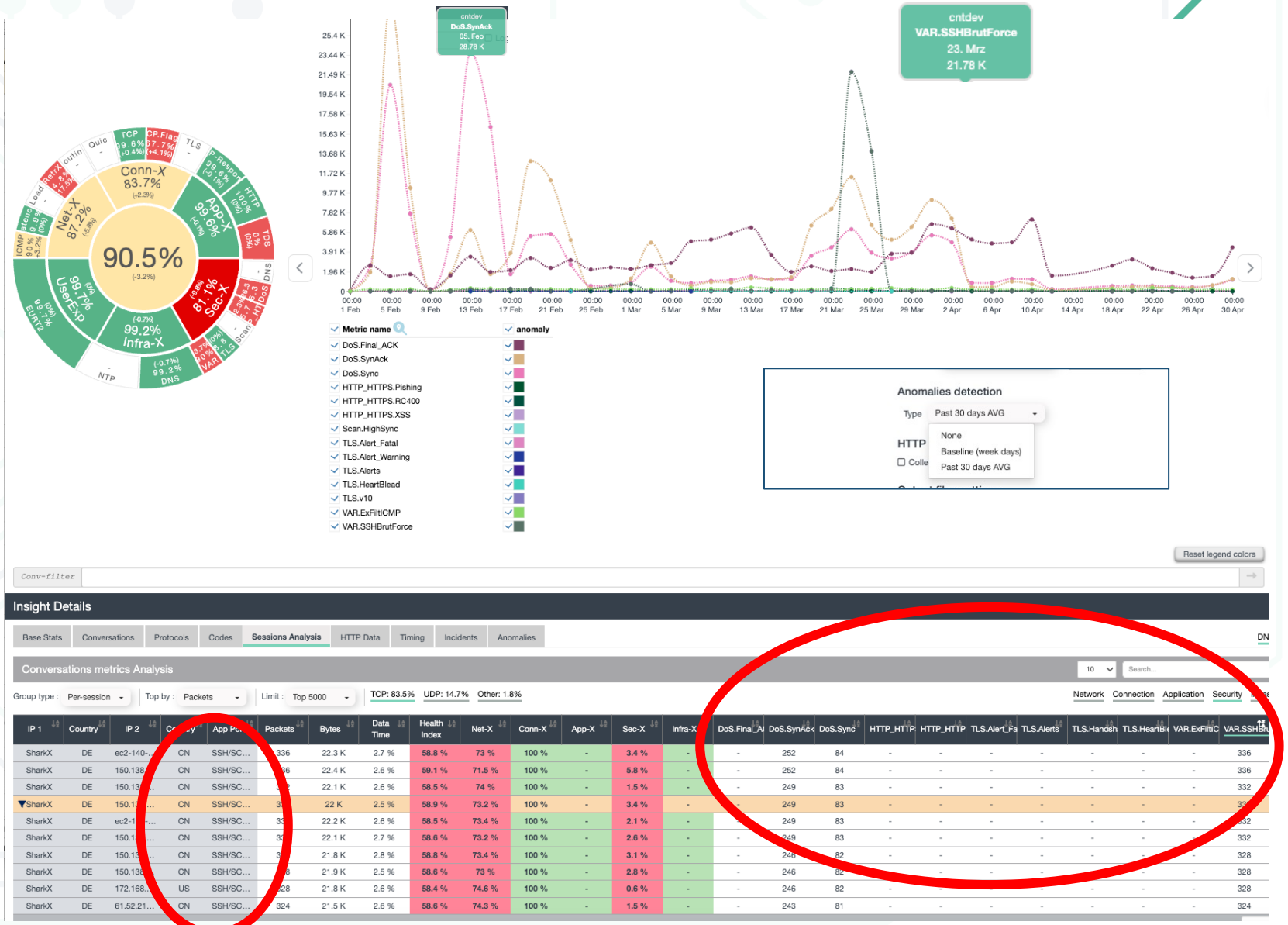
Anomaly – why and how

Problems

- User can not KNOW all metrics
- Fix thresholds often not helpfull
- Cause & Effect often technology-crossing

… and solving

- The anomaly factor – used for many metrics – can help to show heavy changes **/ abnormal behaviour**
  - Example: Brutforce : 23.000!! higher than normal
  - 23.000 remote IPs
- Can show cause & effect – even over different technologies
- We calc constantly AVG over weekday baseline and compare the last seen values with this AVG value



Sharkport

- Easy check features
- Manual upload files
- Create / change profiles
- Invite customers to upload pcaps

- Limits
  - no nomaly dete ction
  - Just manual upload

  - Contact me at ad@inets.de

What do we want here ?

- partners
- Supporters
- Feedback