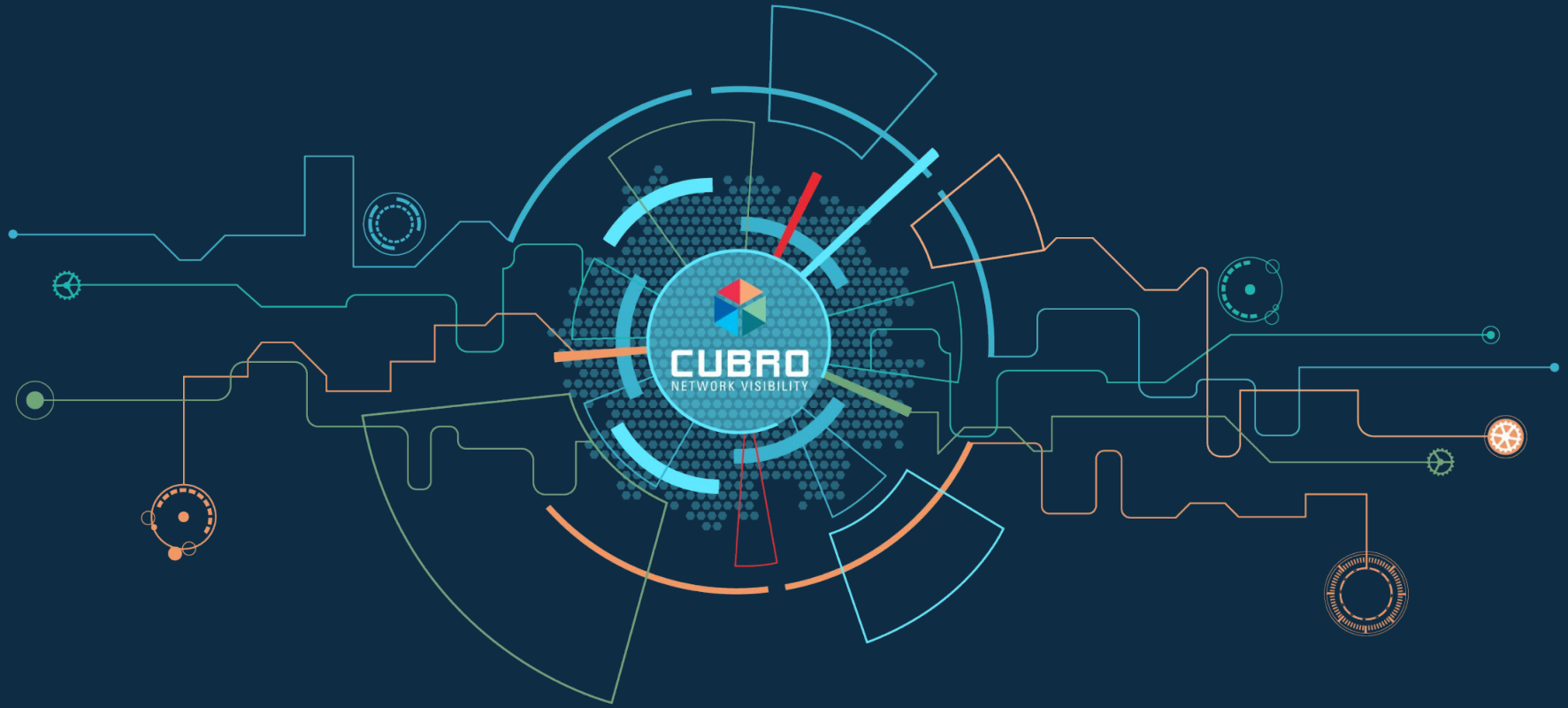


DPI Use Cases for Service Providers

July 2023

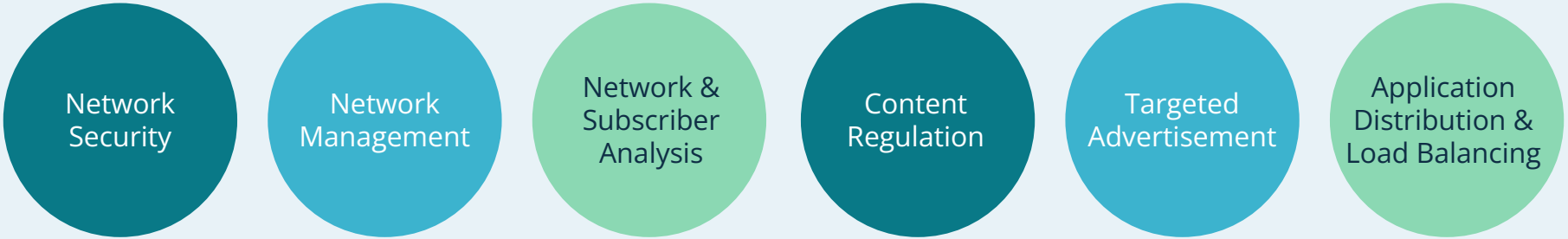
Introduction: DPI in Telecommunication

Deep Packet Inspection (DPI) is a technology that enables the network owner to analyse internet traffic, through the network, in real-time and to differentiate them according to their payload.

DPI is often used for understanding the performance or behaviour of subscribers, which applications they use, how often etc. This helps operators to focus on improving service for the important applications. For instance, video streaming services like Netflix, YouTube, etc. consume a lot of bandwidth. DPI can be used to limit this.

What is Deep Packet Inspection (DPI)

DPI facilitates analyzing and managing IP traffic and securing IP networks in real time by providing network visibility and real-time application awareness. Besides influencing bandwidth and traffic management decisions, DPI can provide insights into:



Why DPI

Deep Packet Inspection (DPI) is used extensively by both enterprises and internet service providers for the following applications.

- Policy Definition and Enforcement
- Buffer Overflow Attack Detection
- Data Leak Prevention (DLP)
- Targeted Advertising
- Quality of Service (QoS)
- Tiered Services Offer
- Copyright Enforcement
- Net Neutrality Prevention
- Lawful Interception
- OTT application monitoring

(see our Video analytic approach)

How is Omnia Metadata output used?

- Find, Identify, Classify, Reroute, and Block Packets with particular data/code payloads.
- Allocate available resources to smoothen traffic flow
- Ameliorate network performance and throughput
- Impose online privacy through sender-receiver identification
- Enable advanced network management, user service, internet data mining, internet censorship, and eavesdropping
- Ensure throttled data transfer, preventing P2P (Peer-to-Peer) misuse

Why is this needed?

Use Case 1: Proactive Service Outage Alerts

Problem

An operator is often held accountable when a major service is unavailable. The support call center is faced with a massive influx of calls, causing a major problem which costs money and creates anger.

Solution

Cubro solution offers real-time information about the service availability, like YouTube, Netflix, all Microsoft services.

How does this help?

- The call center is aware of this situation and can answer accordingly and does not need to generate support tickets.
- Keep subscribers informed: Receive alerts via SMS/Messenger/Website/Mail for service outages and restoration.
- Optimize planning and streamline discussions through service outage monitoring.

Use Case 2: Misuse of Service

In many countries, there is an issue when subscribers act as a sub-service provider. Residents are typically provided with telephone service via a high-powered Wi-Fi hotspot. Subscriptions can also be shared within a building.

Problem

Solution

The Cubro solution is capable of detecting this through unusual traffic/sessions/applications.

How does this help?

To detect misuse of a private plan by companies

Use Case 3: Reselling of geolocation data

Objective

Depending on the regulations in the country, reselling of geolocation data by CSPs can be very profitable.

Problem

Solution

Geolocation intelligence to increase revenue

Cubro can enrich the Geolocation data with application information, even for blocking applications in specific sectors to prevent overloading the network.

Or the SP can sell anonymized geolocation data provided by the Cubro solution.

Use Case 4: Fraud Detection

Law enforcement might require to find SPAM servers and subscribers, Portscanner, Cryptomining, Tor Users/ Servers, SIM Box detection in case of unrecognized activity that might lead to a scam, identity theft or cybercrime.

Problem

Solution

The Cubro solution is capable of detecting this through unusual traffic/sessions/applications and subscribers.

How does this help?

- Detect unusual network and subscriber behaviour
- Detect fraud activities via signature based DPI

Use Case 5: Customer Retention

The rapid change and growing competition in communication service providers makes it tough to retain the existing customers.

Considering the fact that it is pricier to acquire new customers compared to retaining the existing ones, it is one of the main goals of CSPs.

Problem

Solution

Subscribers who frequently use bandwidth test tools are unhappy with the service. It is possible to solve this problem with a proactive call, keeping the customer in the plan.

How does this help?

- More in-depth understanding of user behaviour to proactively contact customers.
- Detailed information about applications used by customers (eg: Speedtest). Extensive speed testing reveals discontent among customers.

Use Case 6: Billing and Volume control

Billing and Billing control are crucial components for success as a Service Provider.

Volume measurements are important, but the volume measurement must be done in different aspects.

Problem

Solution

The Cubro solution can provide very precise “one second resolution” volume measurements.

1. Either aggregated per subscriber or segmented by application.
2. Either aggregated per cell or segmented by subscriber.
3. Especially useful for VMNO roaming and outbound traffic to different AS (Autonomous Systems).

Measurement Metrics based on Cubro Metadata

Subscriber & Roaming Subscriber Metrics

- Monitor real-time total bandwidth usage of each individual subscriber with one-second resolution.
- Calculate volume usage of each individual subscriber for different time frames such as Day/Week/Month/Year.
- Monitor real-time bandwidth usage per application of each individual subscriber with one-second resolution.
- Calculate volume usage per application of each individual subscriber for different time frames such as Day/Week/Month/Year.
- Detect unwanted or forbidden applications per individual subscriber.
- Identify irregular SIM card changes per subscriber, which may indicate possible fraud cases (for mobile networks only).
- Track the location of subscribers in real-time based on IMSI or IMEI to find stolen mobiles (for mobile networks only).
- Perform baselining to detect proactive misuse of the offered service such as reselling or SIM box activities.

Service Metrics

Service Metrics

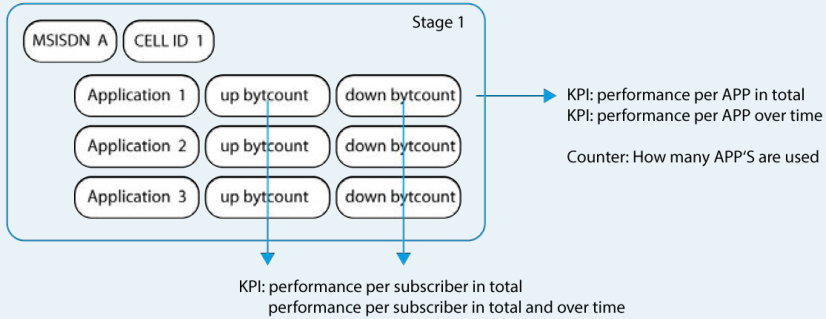
- Receive an alarm in the event of a service outage across the entire network, such as Netflix, Office364, and AWS services.
- Receive an alarm in the event of a service outage at specific sites, such as Netflix, Office364, and AWS services.
- Receive an alarm in the event of a service outage for a specific user or user group.
- Receive information when the service is restored.
- Monitor service usage statistics per individual service.
- Monitor service usage statistics per service groups, such as Social Media and Video Streaming.
- Monitor traffic distribution to Autonomous Systems (AS).
- Receive an alarm for unwanted services and apps. If necessary, inline blocking can be applied to the entire network or specific network sectors.

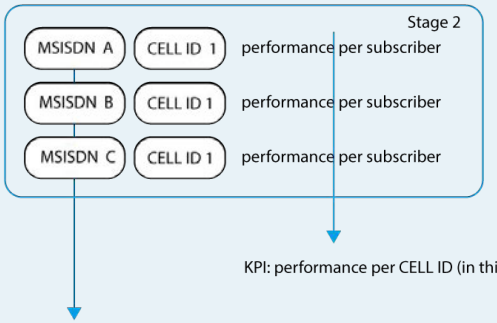
Network Element Metrics

Network Element Metrics

- Monitor near real-time bandwidth usage per cell with one-second resolution.
- Detect congestion near real-time per cell using burst detection.
- Monitor service distribution per cell for network planning purposes.
- Use machine learning to establish baselines and identify abnormal cell behaviour.

From raw metadata → metric → KPI → use case





Counter: amount of subscribers per CELL ID

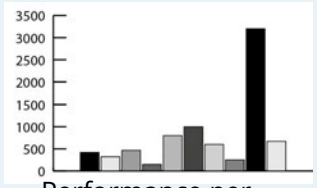
Alarm: When a specific subscriber enters a specific CELL or CELL cluster /region)

KPI: change rate of subscribers per CELL (Cell jumping)

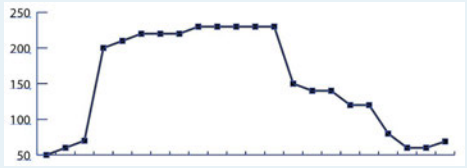
This schema describes how much value our data has:

By combining different fields from the CDR and some simple calculator, it is possible to generate KPIs / Counter and Alarms out of the network.

In this case, it is the performance of the CELLs in the entire network.



Performance per Cell

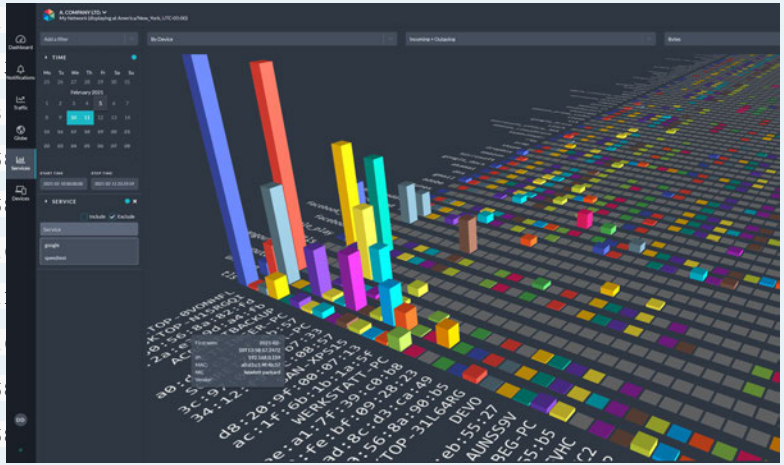


Cell performance over time

Cubro DPI

Deep Packet Inspection (DPI) is detecting traffic type by Signature; beyond port and protocol


```
{ "hash":"7461864e", "service": "whatsapp", "ip_1":"213.143.110.250", "ip_2":"31
{ "hash":"7461864e", "service": "whatsapp", "ip_1":"31.13.84.49", "ip_2":"213.14
{ "hash":"ce931c02", "service": "whatsapp", "ip_1":"192.168.3.83", "ip_2":"31.13
{ "hash":"ce931c02", "service": "whatsapp", "ip_1":"31.13.84.51", "ip_2":"192.16
{ "hash":"70d46209", "service": "whatsapp", "ip_1":"31.13.84.49", "ip_2":"192.16
{ "hash":"b6cd9e62", "service": "whatsapp", "ip_1":"80.110.82.15", "ip_2":"192.1
{ "hash":"6fc2f8ce", "service": "whatsapp", "ip_1":"192.168.3.130", "ip_2":"80.1
{ "hash":"55e8f416", "service": "whatsapp", "ip_1":"192.168.3.44", "ip_2":"192.1
{ "hash":"113d5e32", "service": "whatsapp", "ip_1":"31.13.84.49", "ip_2":"192.16
{ "hash":"a597661a", "service": "whatsapp", "ip_1":"31.13.84.49", "ip_2":"192.16
```

This is the output from our DPI engine, we can find WhatsApp even when it is ciphered.
We can find up to 4000 different applications.

DPI Applications

There are generally two different main applications for DPI

1. Analytics

A: In this application, the DPI engine can decode the full traffic and produce results in DB format for analytics purpose. This is only possible on CPU-based units like Omnia series. Since every packet has to be handled, it is a big effort in terms of CPU load and data output.

B: IPFIX with DPI enriched output. This is also a very common way of analyzing DPI data, but it is not very efficient and produces a lot of overhead. IPFIX on ISP level is very difficult because of the high amount of parallel sessions in the network. This often leads to issues on the installed probe like reaching memory limits.

2. Tagging/filtering/blocking

This application resonates with Cubro approach - remove an unwanted application type from the monitoring. **It is common to remove video streaming services.**

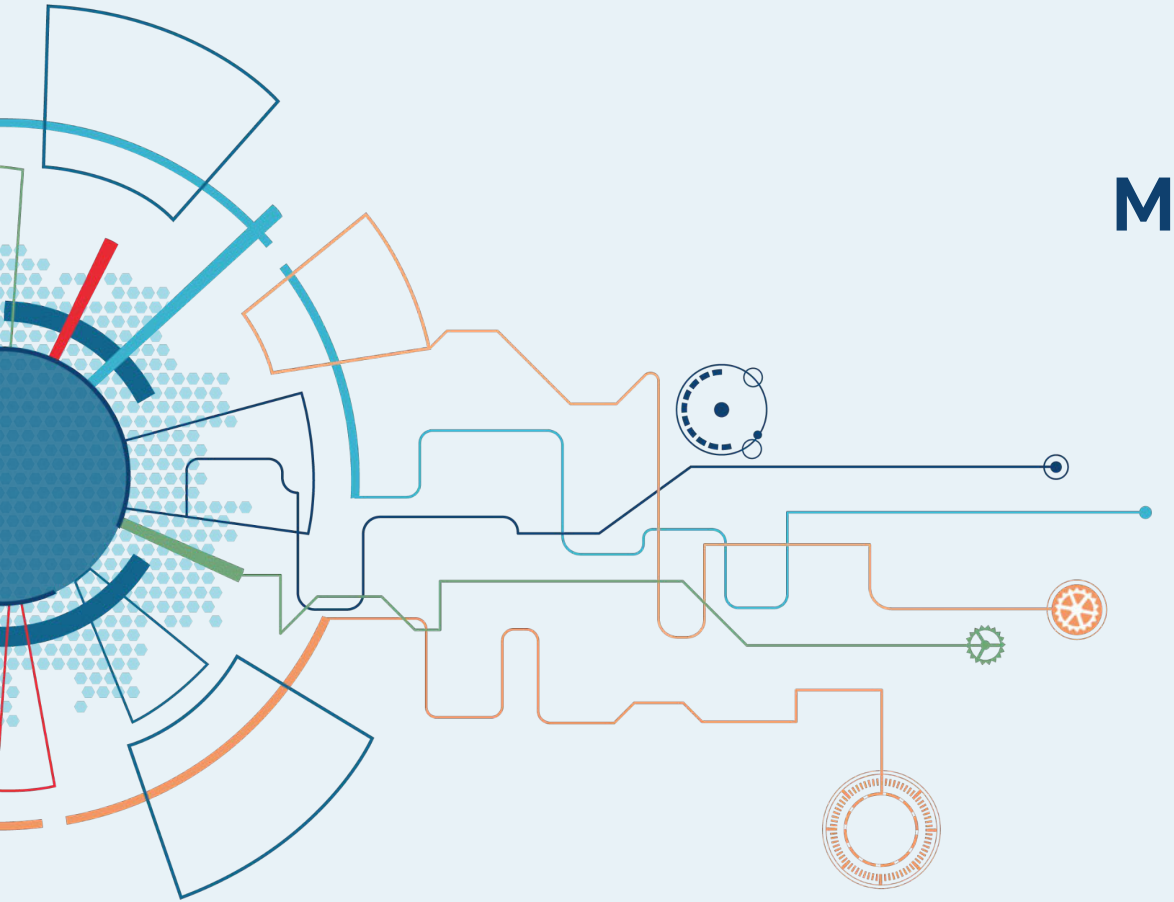
The same application is for blocking certain applications, or sending certain traffic to a special monitoring device. In this case, it is not needed to do a full decode because sampling gives a similar result but with much less effort.

DPI Signatures (Applications & Protocols)

We support up to 4000 signatures. These signatures are divided into two parts:

- 1400 see DPI Services - these are the top signatures which are maintained manually.
- The other signatures are maintained by deep learning and AI.

(The update cycle is between 7 and 10 Days)



More detailed Use Cases

Bandwidth per Service per Subscriber

For a service provider, customer experience is a significant concern, and it directly correlates with bandwidth per application. This is precisely what Cubro can efficiently deliver.

Cubro provides per user/per application bandwidth information with a one-second resolution, ranging from 100 Gbit/sec up to multiple TB/sec.

Additionally, Cubro offers DPI data enrichment with Radius and Mobile data from other sources on request.

This is the XDR Probe output : *MS ISDN, IMSI, IMEI, Cell ID, RAT, APN, USER IP, Application, Bandwidth in 1 sec resolution*

Service Provider Use cases

- If the customer complaints, the Service Provider can demonstrate the bandwidth was adequate or inadequate due to a CELL issue, incorrect RAT (Radio Access Technology) or APN (Access Point Name). This simple method effectively resolves most customer complaints.
- With the same data, you can see the load on a specific CELL, and the application distribution.
- Customer movements can be monitored.
- Customer mobile phone type change.

Bandwidth per Service per Subscriber

LI Use cases:

- Profiling a person by his usage of application over time (base lining)

  Master or Slave because we have a 100% price volume statistic up and down, it is possible to detect if the person is giving orders or receiving orders.

- Activity of individuals (baselining and ML)
- Changing SIM cards (IMEI – MS ISDN correlation)
- Customer mobile phone type change.
- Movement of individuals via CELL ID and RAT changes
- With additional ML learning, it would be possible to identify people with relations
- Filtering on IMSI/MS ISDN for specific person to get full traffic captured.

Omnia QM Solution

Omnia QM Solution using Call Detail Records (CDRs) provides an efficient method of optimizing data and can potentially help CSPs to monitor data with less HW and SW resources.

One user WhatsApp traffic

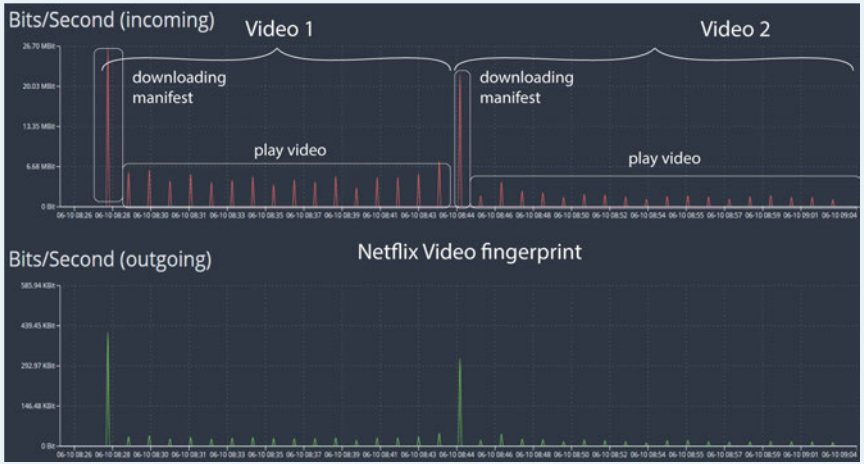
Master or Slave because we have a 100% price volume statistic both up and down, it is possible to detect is the person giving orders or receiving orders.

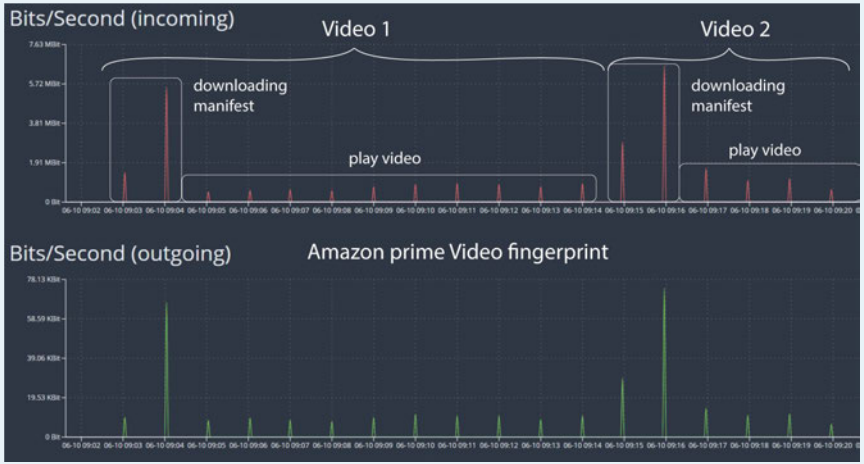
This graph shows that this subscriber has only received messages because the outgoing traffic was in Bit and incoming traffic in kBit.

The outgoing was only management traffic (the acknowledgment).

This simple example shows how high-resolution bandwidth information can be.

Video analytic via Fingerprint

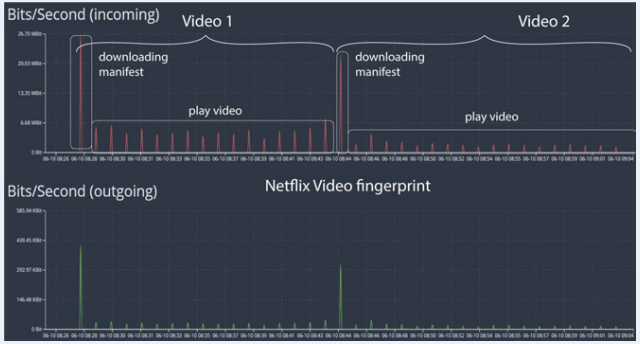




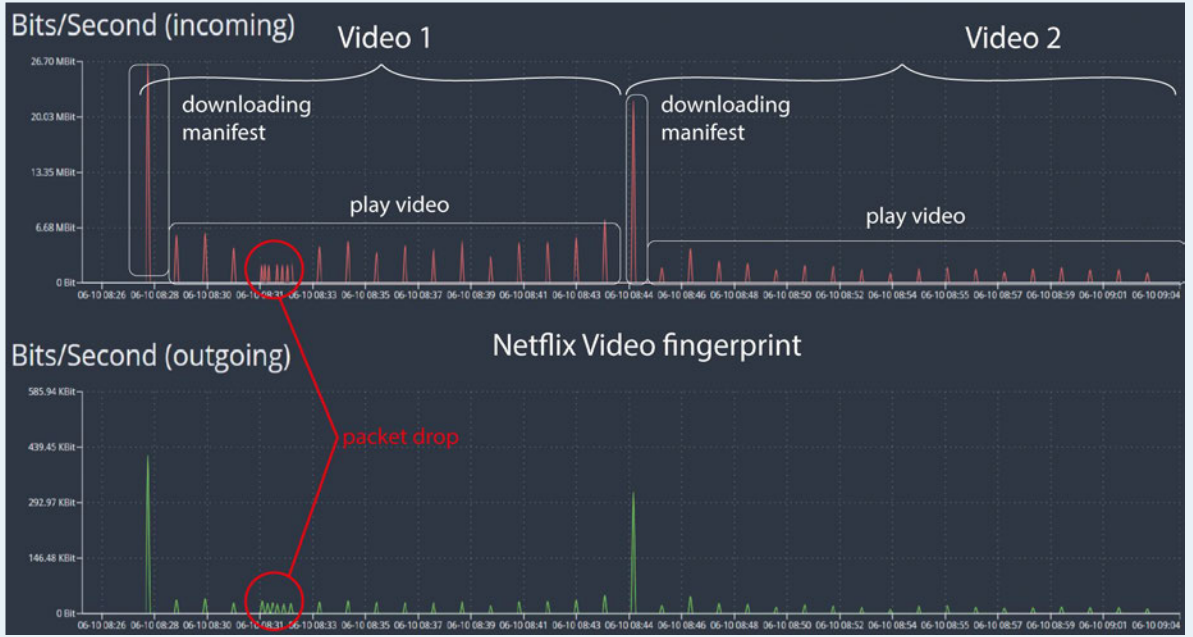
Each streaming application has a unique bandwidth fingerprint. This information can be used to detect good and bad service quality. Additionally, it can be leveraged to determine the frequency of customer channel switching and the number of streams being watched. Such insights are valuable for troubleshooting, or providing counterarguments against customer complaints.

These pictures show good performance, any other behaviour leads to bad quality.

Example of Packet Drop



Packet drop leads to this picture.



This test is done with an impairment analyser to show what is the effect of 3% packet drop!

Time window based XDR
(Cubro XDR)

Time window-based Monitoring (1/2)

CUBRO NETWORK VISIBILITY

As described in the previous slides, there are several disadvantages to performing flow-based computation, mainly due to a lack of resources (CPU, memory, storage).

Aggregating metadata time window based solves most of these problems and allows a much higher computation throughput without losing any important metadata attributes.

The significant metadata cannot be extracted from single flows, as they mainly contain data from a technical perspective. The significant information is, for example, based on a device or user within the network. Which traffic is generated from a specific device, which services are used by a specific device and to what extent, how much traffic is generated by a user over time, which servers are involved, etc.

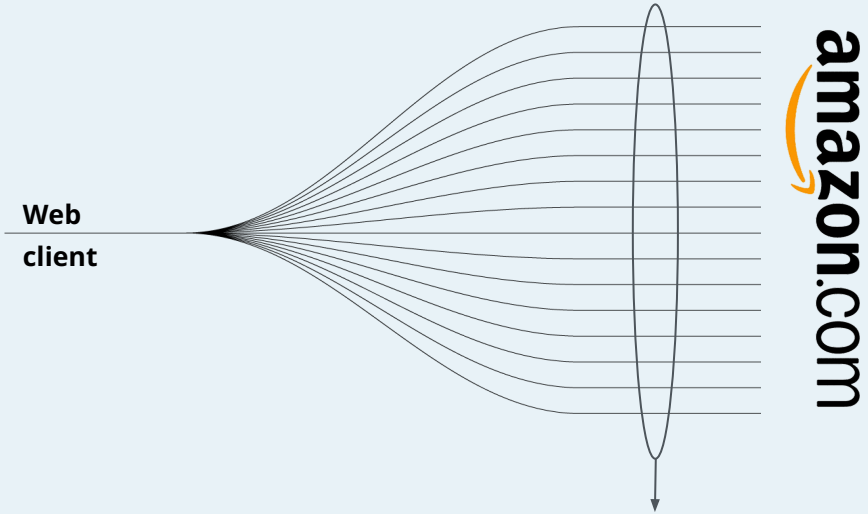
Time window-based Monitoring (2/2)

The time window-based approach focuses on this significant data.

The key is collecting Metadata for uploads, downloads, and internal traffic as well as DPI information from a device/user perspective over time. Extracting the essential information out of big data streams with the benefit of not wasting resources and storage for absolutely meaningless information (contained within single flows) is the important point.
The time window-based approach allows many essential views of the data:

- **Service perspective:** For every Service the amount of traffic, how often it is used over time, how much traffic is uploaded or downloaded, etc.
- **Client/Device perspective:** For every Client (user) the usage of the network by upload and download, which services and locations are used, how often they are used over time.
- **Network perspective:** For every APN, eNodeB Cell, radio technology, how much traffic is uploaded or downloaded, what are the most frequent services, how many clients are connected, etc.

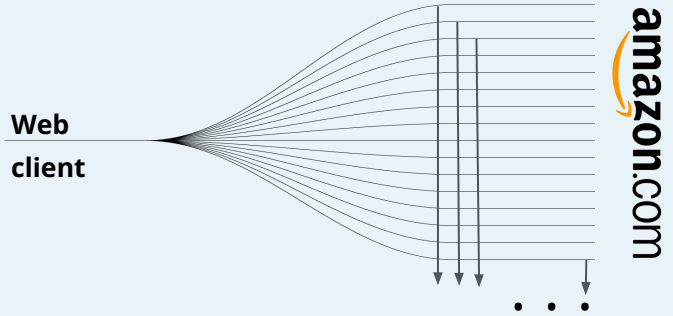
Time window based



For each client, there is a bucket for each application, if needed, and we collect/count all packets for a certain time window (configurable). When the window is closed, an XDR is produced/enriched and sent out. The advantage is, the traffic is reduced on most far points to avoid constraints on the workflow along to the database.

vs

Flow based



For each 5 tuple connection, one flow is produced. A lot of these flows cannot be detected, for instance, Amazon related because the external domain cannot be resolved.

After producing the flow, it is forwarded to the Flow Cache.
A Flow Cache can contain hundreds of thousands of entries, and in some cases, even millions of entries. This costs memory resources.

When the flows expire, they're exported off to the NetFlow Collector, which will constantly analyse and archive the flows for future reference.

Time window based

- We aren't concerned about sessions, (TCP handshake) perpetually open sessions are not an issue.
- Irregularly terminated/established sessions are also not an issue.
- The configurable time window offers the option to balance between performance constraints and granularity of the output.
- 0.1% - 0.5% of the input traffic is the size of the resulting metadata stream (configurable).

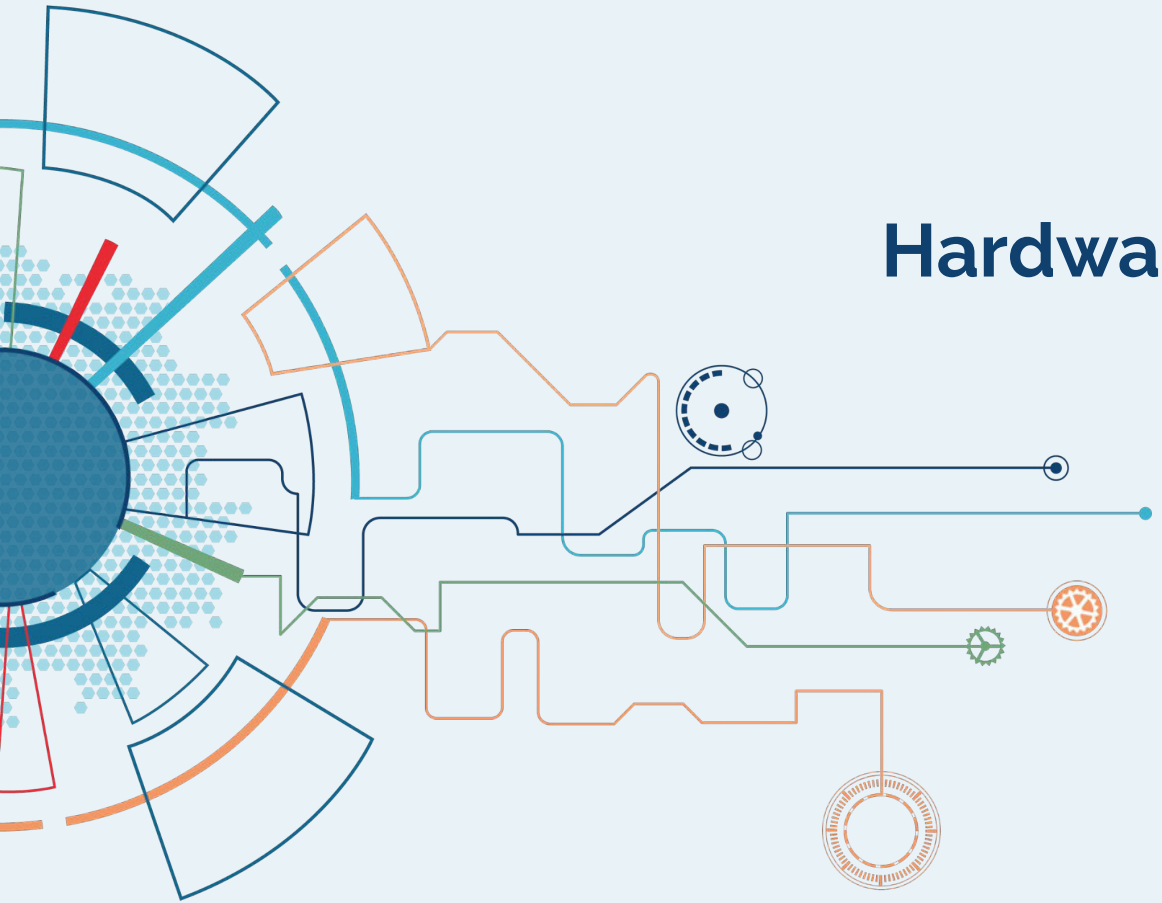
vs

Flow based

- The flow-based solution cares about sessions, (TCP handshake) perpetually open sessions are an issue. Typically, a flow probe has a limitation in terms of number of flows (FPS) not bandwidth.
- Irregularly terminated/established sessions are also an issue because an irregularly terminated session stays open until the timeout and consumes unnecessary resources.

  A session where the initial handshake is not seen for any reason will not be detected. For IoT especially, this could be an issue because such devices talk very rarely; it could be days until a session is detected again.
- 2% - 3% of the input traffic is the size of the resulting metadata stream.

For both solutions, there are pros and cons, but with the dramatic growth of traffic, a time window-based solution is much more efficient and saves Capex and Opex costs.

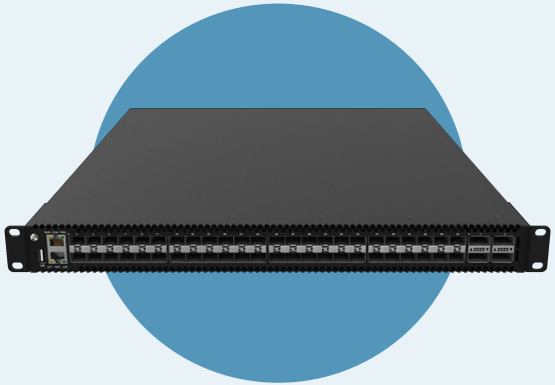


Hardware Appliance

CPU & Switch Omnia models overview

CUBRO NETWORK VISIBILITY

Omnia120



CPU only Omnia models overview

Omnia200

Omnia400

Omnic

Omnics (Cubro Smart NICs)

The implementation of high layer software on X86 server is becoming more and more complex. The performance of network service subsystem is a key factor to simplify application development and deployment.

Cubro i-NIC helps application software to offload network related processing from server CPU to a dedicated SoC, so that the application system can be accelerated with too much modification on existing software.

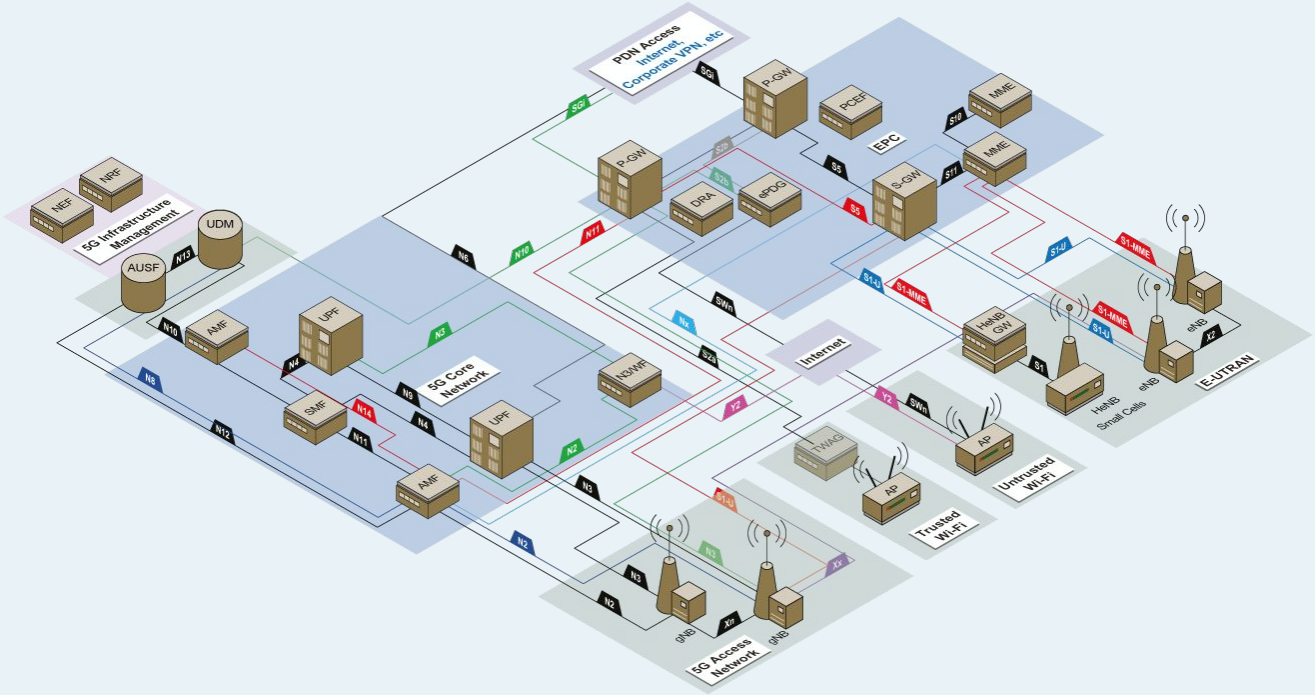
- **4 x 10/25 Gbit SFP+**
- **2 x 100 Gbit QSFP**
- **24 core ARM CPU**
- **64 GB Memory**
- **16 lane PCI connection (v4)**
- **Works also as a stand alone**

4 x 10/ 25 Gbit/s Interface

2 x 100 Gbit/s Interface

Terabit metadata extraction application

How we connect the solution to the network



4G and 5G Interface which we can use

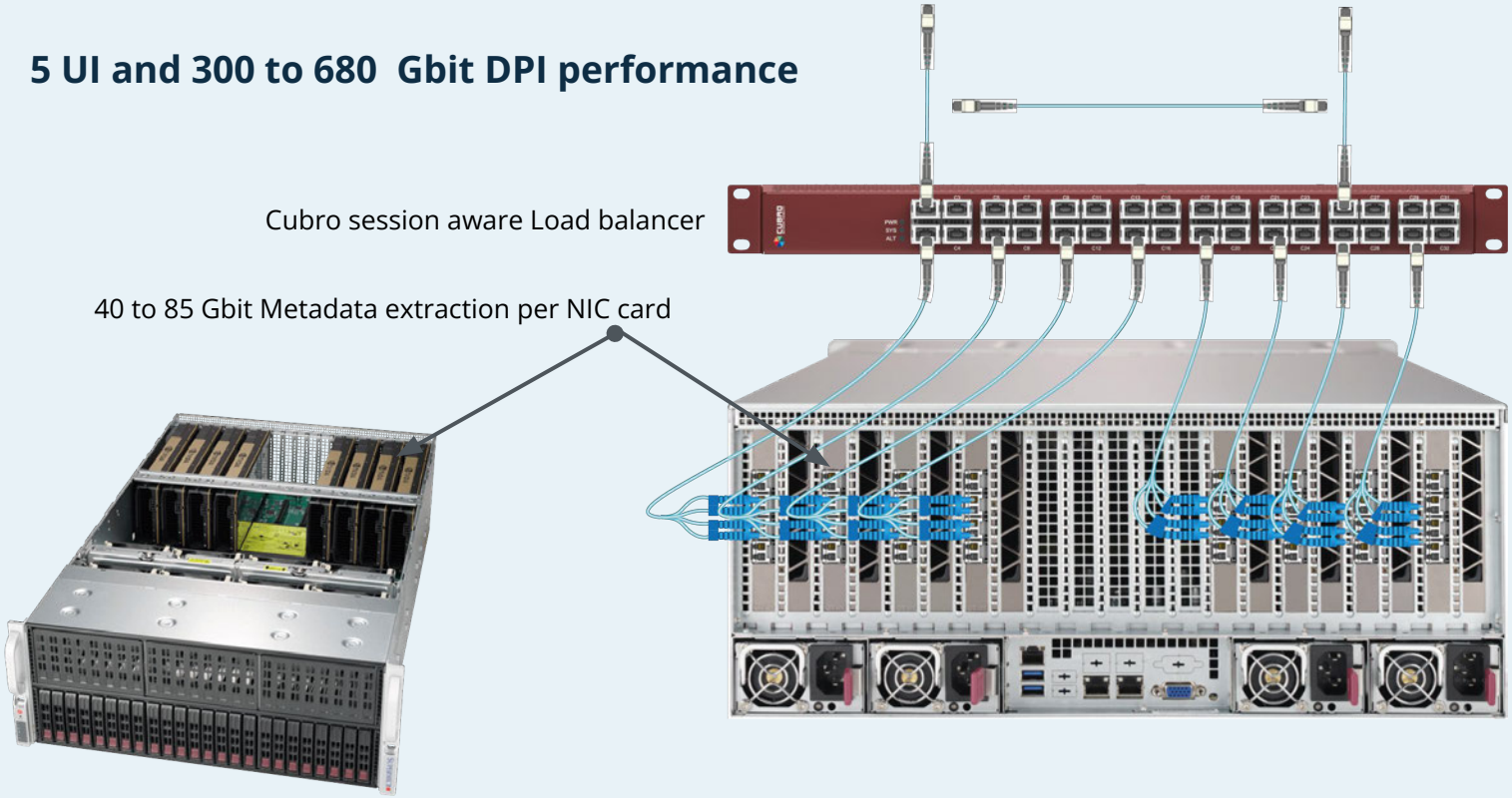
S2b	GTPv2-C/GTPv1-U	ePDG - PDN-GW
S3	GTPv2-C	MME - SGSN
S4	GTPv2-C/GTPv1-U	S-GW - SGSN
S5	GTPv2-C/GTPv1-U	S-GW - PDN-GW
S6a	Diameter	MME - HSS
S6d	Diameter	SGSN - HSS
S8	GTPv2-C/GTPv1-U	S-GW - PDN-GW (Roaming)
S10	GTPv2-C	MME - MME
S11	GTPv2-C	MME - S-GW
S12	GTPv1-U	S-GW - UTRAN
S13	Diameter	MME - EIR
SGi	IP	PDN-GW - PDN
N1	N2-AP/SCTP	gNB - AMF
N2	N2-AP/SCTP	gNB - AMF
N3	GTPv1-U	gNB - UPF
N4U	GTPv1-U	SMF - UPF
N6	GTPv1-U	UPF - Data Network
N9	GTPv1-U	UPF - UPF

Smart NIC in Server

5 UI and 300 to 680 Gbit DPI performance

Cubro session aware Load balancer

40 to 85 Gbit Metadata extraction per NIC card



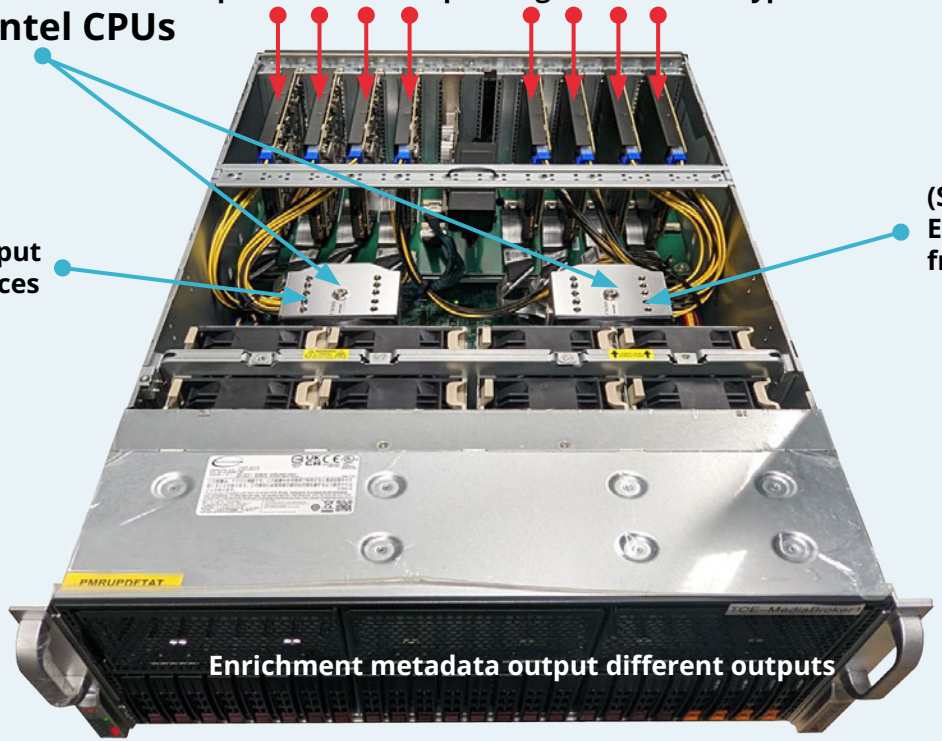
Super Server internal function

Enrichment and data processing on the Intel CPUs

(User plane) Raw packet input up to 40 – 85 Gbit per Smart NIC depending on the traffic type

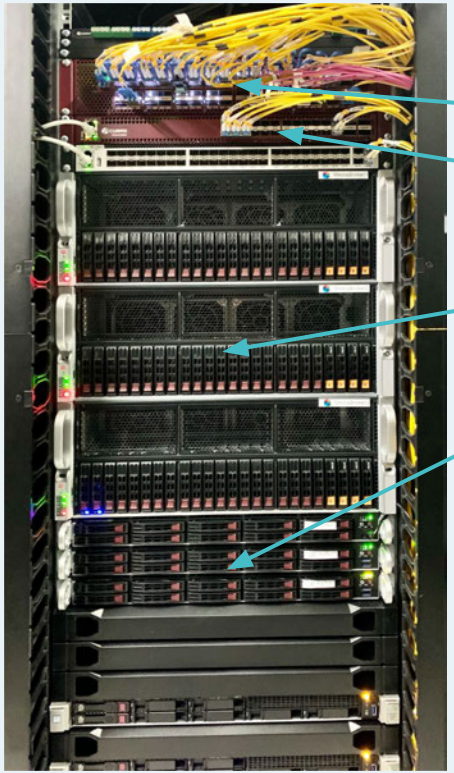
(Signaling traffic) Enrichment data input from different sources

(Signaling traffic) Enrichment data input from different sources



Enrichment metadata output different outputs

1–2 Tbps User Plane monitoring in an MNO
(real picture)

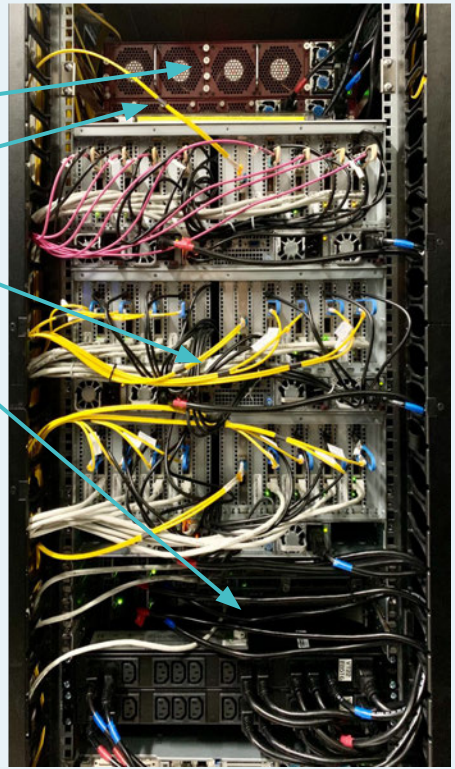


NPB EXA64100

Signalling probe

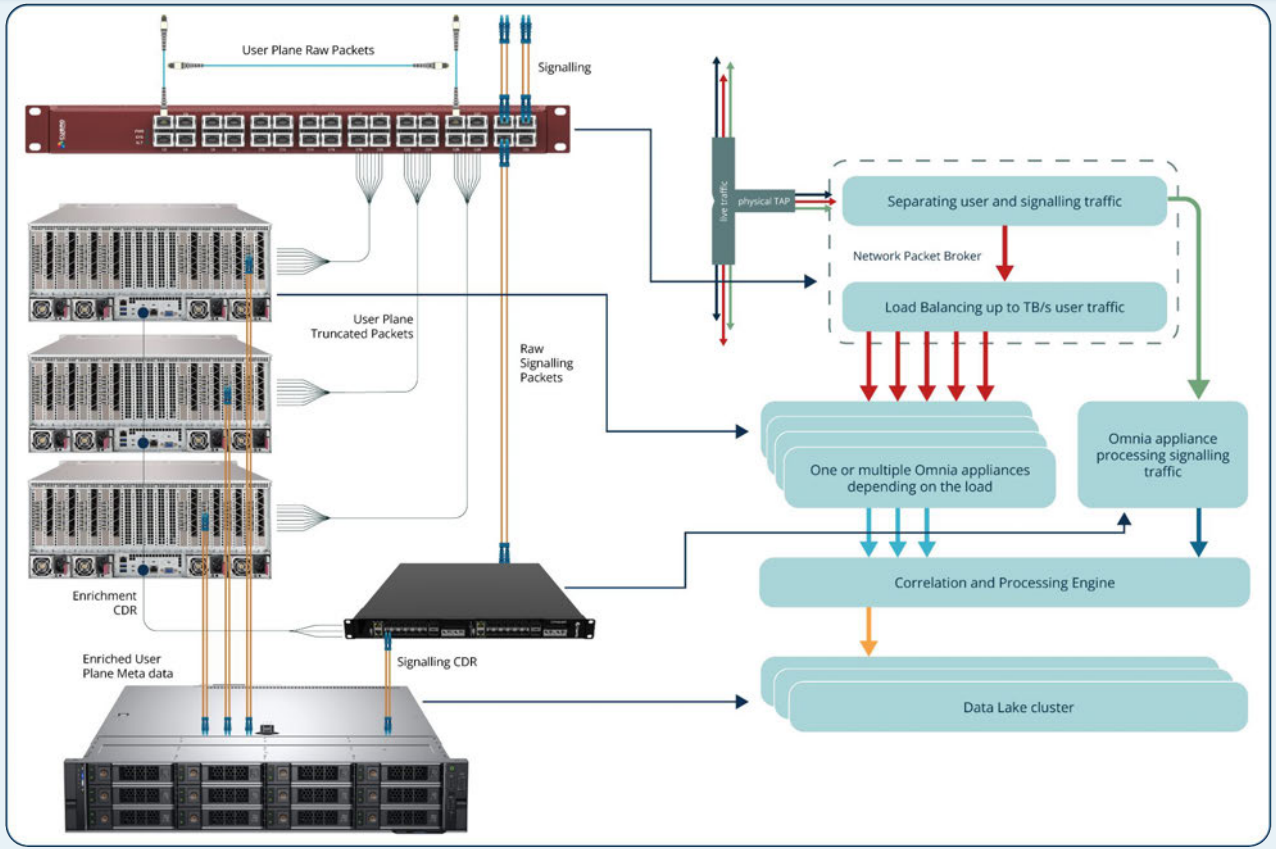
3 Server with
24 NPU smart NIC

3 Kafka server for
Metadata handling

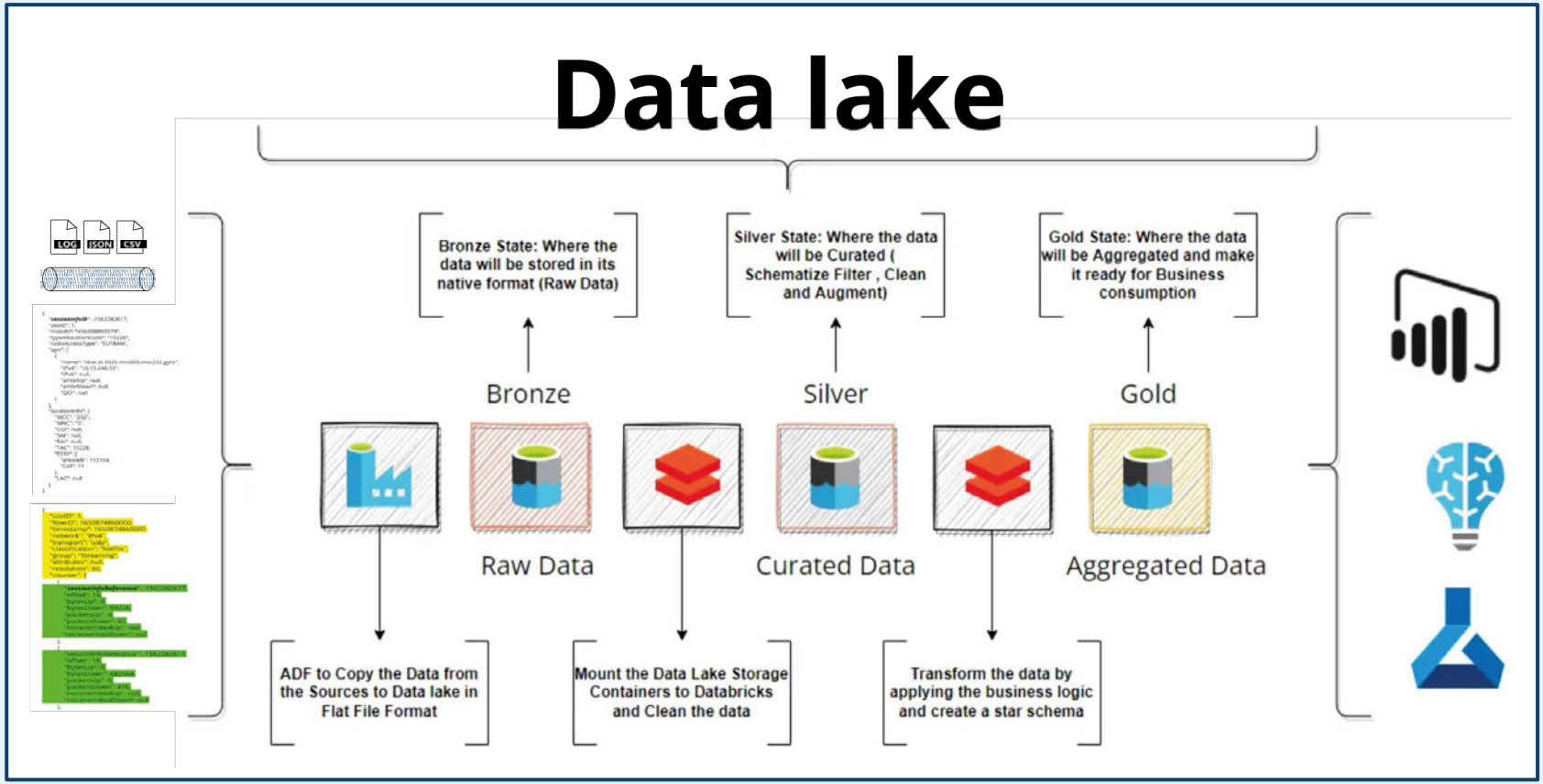


only 19 U
836 mm

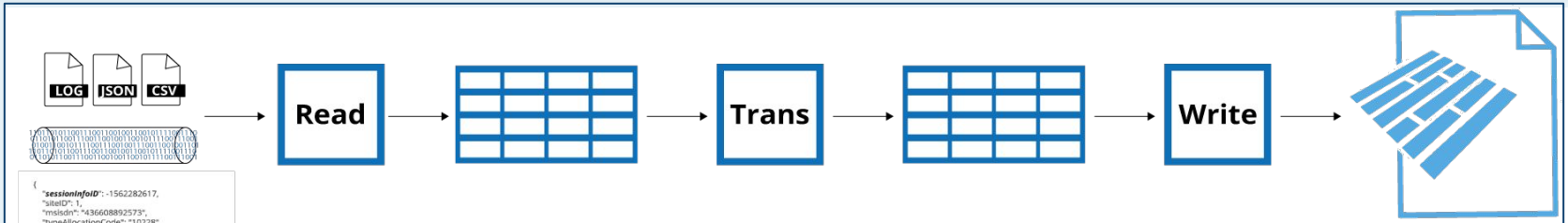
System Schema for 1-1.2 TB of user traffic



Data Lake


Data lake
LOG
JSON
CSV
Bronze State: Where the data will be stored in its native format (Raw Data)
Silver State: Where the data will be Curated (Schematize Filter , Clean and Augment)
Gold State: Where the data will be Aggregated and make it ready for Business consumption
Bronze
Silver
Gold
Raw Data
Curated Data
Aggregated Data
ADF to Copy the Data from the Sources to Data lake in Flat File Format
Mount the Data Lake Storage Containers to Databricks and Clean the data
Transform the data by applying the business logic and create a star schema

Function of the Data Lake




```
{
  "sessioninfoID": -1562282617,
  "siteID": 1,
  "msisdn": "436608892573",
  "typeAllocationCode": "10228",
  "radioAccessType": "EUTRAN",
  "apn": [
    {
      "name": "drei.at.1020.mnc005.mcc232.gprs",
      "IPv4": "10.15.246.51",
      "IPv6": null,
      "ambrUp": null,
      "ambrDown": null,
      "QCI": null
    }
  ],
  "locationInfo": {
    "MCC": "232",
    "MNC": "5",
    "CGI": null,
    "SAI": null,
    "RAI": null,
    "TAC": 10228,
    "ECGI": {
      "eNodeB": 112164,
      "Cell": 11
    },
    "LAC": null
  }
}
```



```
{
  "siteID": 1,
  "flowID": 1650874860000,
  "timestamp": 1650874860000,
  "network": "IPv4",
  "transport": "udp",
  "classification": "Netflix",
  "group": "Streaming",
  "attributes": null,
  "resolution": 60,
  "counter": [
    {
      "sessionInfoReference": -1562282617,
      "offset": 14,
      "bytesUp": 0,
      "bytesDown": 59226,
      "packetsUp": 0,
      "packetsDown": 45,
      "retransmittedUp": null,
      "retransmittedDown": null
    },
    {
      "sessionInfoReference": -1562282617,
      "offset": 18,
      "bytesUp": 0,
      "bytesDown": 682564,
      "packetsUp": 0,
      "packetsDown": 476,
      "retransmittedUp": null,
      "retransmittedDown": null
    },
```


Signaling / Control Plan

site ID	sessioninfoID	msisdn	rat	apn	cell id	MCC	MNC	LAC
1	1562282617	43 660 234 233	LTE	xxx	2	●	●	●
1	1562282618	43 660 234 224	5G	xxx	8	●	●	●
1	1562282619	43 660 565 755	3G	xxx	16	●	●	●
1	1562282620	43 660 862 934	LTE	xxx	23	●	●	●
1	1562282621	43 660 496 946	LTE	xxx	45	●	●	●
1	1562282622	43 660 888 331	5G	xxxxx	78	●	●	●
1	1562282623	43 660 405 413	5G	xxx	876	●	●	●
1	1562282624	43 660 631 179	3G	xxxxxx	3	●	●	●
1	1562282625	43 660 539 629	4G	xxx	2	●	●	●
1	1562282626	43 660 974 038	LTE	xxxxxxx	3	●	●	●
1	1562282627	43 660 642 882	LTE	xx	45	●	●	●

User Plan

site ID	sessioninfoID	User IP	Group	App	bytes up	bytes down	packets up	packets down
1	1562282617	10.25.37.110	Streaming	Netflix	767	454	8	27
1	1562282617	10.25.37.110	Messaging	Whatsapp	188	771	14	23
1	1562282617	10.25.37.110	Social	Facebook	437	553	26	3
1	1562282617	10.25.37.110	Office	Salesforce	688	917	34	20
1	1562282618	10.35.32.200	Streaming	Youtube	291	11	10	7
1	1562282618	10.35.32.200	Social	Instagram	302	128	12	25
1	1562282618	10.35.32.200	Office	Office365	666	723	35	4
1	1562282618	10.35.32.200	Cloud	S3 bucket	410	472	38	27
1	1562282618	10.35.32.200	HTTPS	unknown	722	295	7	11
1	1562282619	10.56.19.157			238	842	19	34
	I	I	I	I	622	347	30	21
1	1562282620	10.77.35.78	I	I	293	497	37	22
	I	I	I	I	604	281	33	6
1	1562282621	10.34.77.148						
1	1562282622	10.99.23.13						
1	1562282623	10.22.19.158						
1	1562282624	10.66.34.26						
1	1562282625	10.87.43.76						
1	1562282626	10.10.123.18						
1	1562282627	10.87.12.99						

Combinde Tabel (silver)

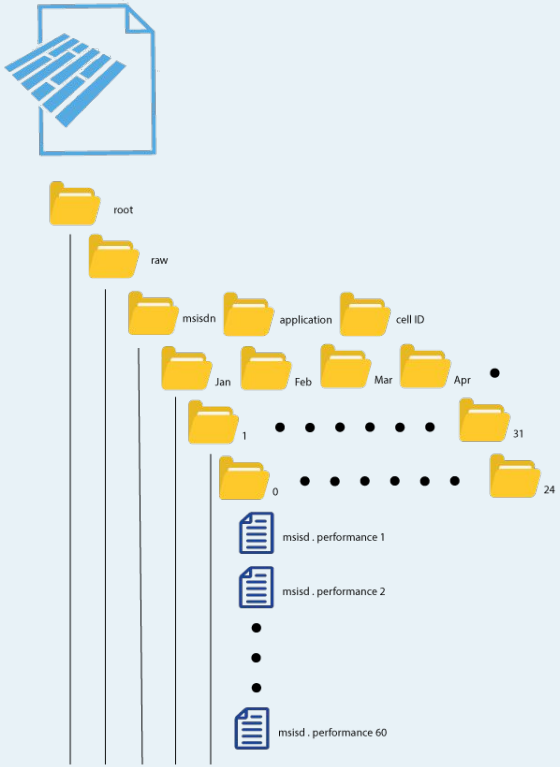
site ID	sessioninfoID	User IP	msisdn	rat	Group	App	bytes up	bytes down	packets up	packets down
1	1562282617	10.25.37.110	43 660 234 233	LTE	Streaming	Netflix	199	414	17	4
1	1562282617	10.25.37.110	43 660 234 233	LTE	Messaging	Whatsapp	431	724	39	34
1	1562282617	10.25.37.110	43 660 234 233	LTE	Social	Facebook	761	699	15	25
1	1562282617	10.25.37.110	43 660 234 233	LTE	Office	Salesforce	155	741	34	15
1	1562282618	10.35.32.200	43 660 234 224	5G	Streaming	Youtube	856	651	26	40
1	1562282618	10.35.32.200	43 660 234 224	5G	Social	Instagram	446	367	2	19
1	1562282618	10.35.32.200	43 660 234 224	5G	Office	Office365	226	829	13	11
1	1562282618	10.35.32.200	43 660 234 224	5G	Cloud	S3 bucket	759	611	1	5
1	1562282618	10.35.32.200	43 660 234 224	5G	HTTPS	unknown	519	400	12	29
1	1562282619	10.56.19.157	43 660 565 755	3G			183	263	39	11
	I	I			I	I	390	354	3	10
1	1562282620	10.77.35.78			I	I	296	47	18	33
	I	I			I	I	636	249	1	28
1	1562282621	10.34.77.148								
1	1562282622	10.99.23.13								
1	1562282623	10.22.19.158								
1	1562282624	10.66.34.26								
1	1562282625	10.87.43.76								
1	1562282626	10.10.123.18								
1	1562282627	10.87.12.99								

reade to use Tabel (gold)

cell id	bytes down	bytes up
2	6554	3267
8	42707	5973
16	38931	6438
23	3344	9941
45	54513	7722
78	22931	5033
876	4629	5184
3	32412	4757
2	46399	4616
3	93629	9860
45	45340	4412

msisdn	bytes down	bytes up
43 660 234 233	369554	4965
43 660 234 224	655738	3156
43 660 565 755	282191	9529
43 660 862 934	114190	4303
43 660 496 946	620065	6384
43 660 888 331	657119	4316
43 660 405 413	512471	8317
43 660 631 179	960819	5276
43 660 539 629	475265	4716
43 660 974 038	459683	4649
43 660 642 882	949917	8568

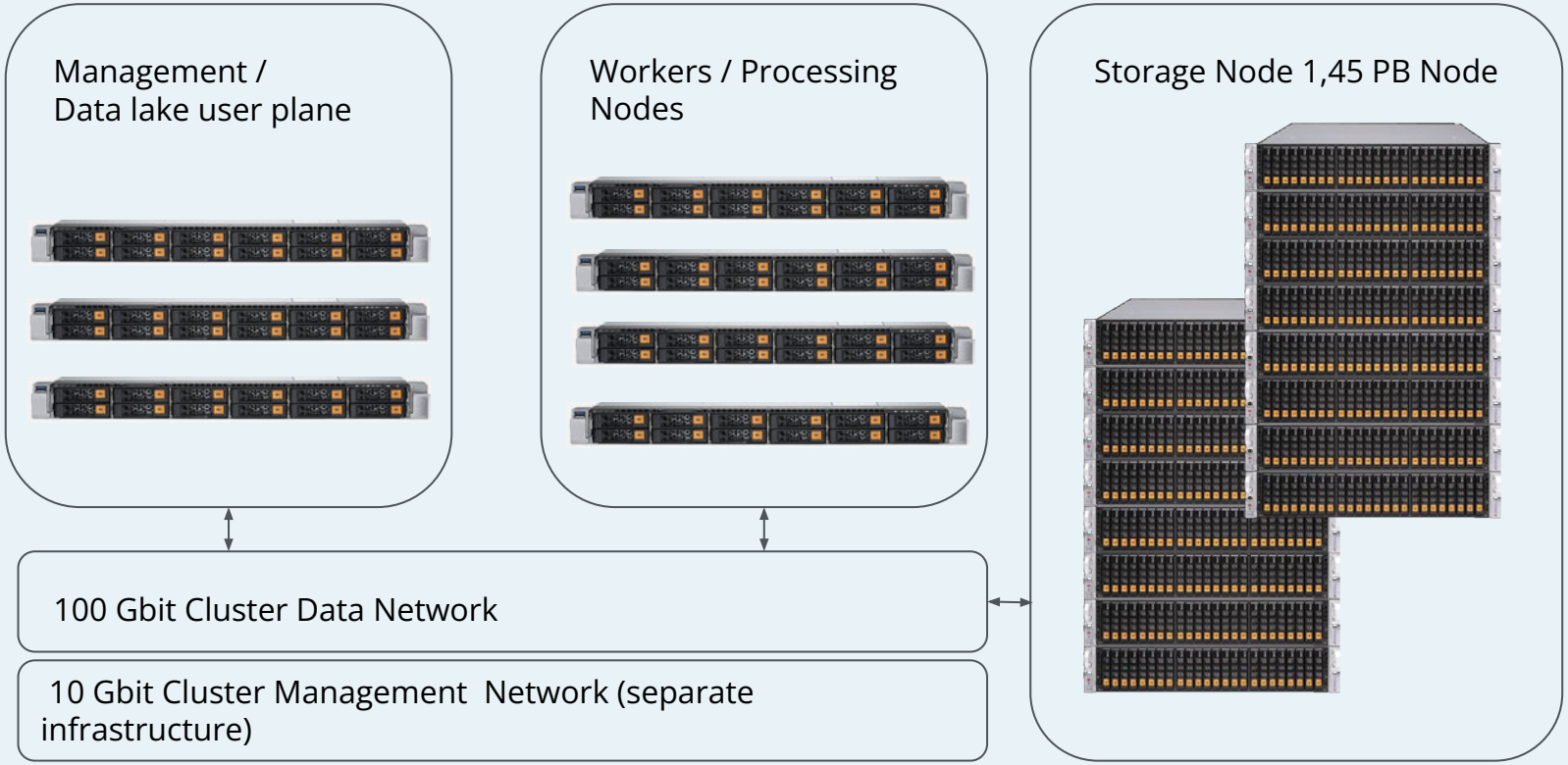
Advantages of a Data Lake compared to a DB



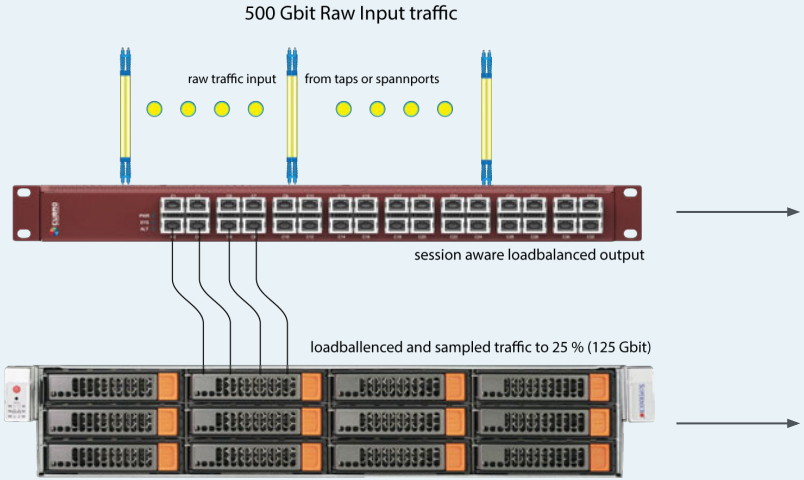
All reports and results are preprocessed, which ensures rapid execution.

However, this approach demands higher storage and CPU resources compared to a database (DB) approach.

Cluster Data Network



The passive DPI solution for sampled traffic 1:4



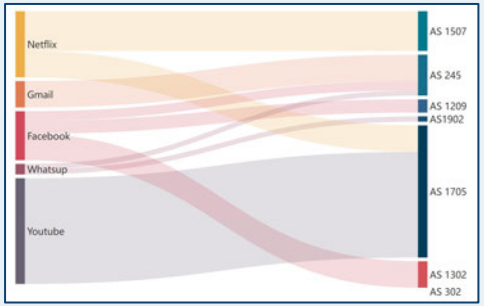
Total Counter from the live link

recalculate to the full bandwidth

DPI output per live link

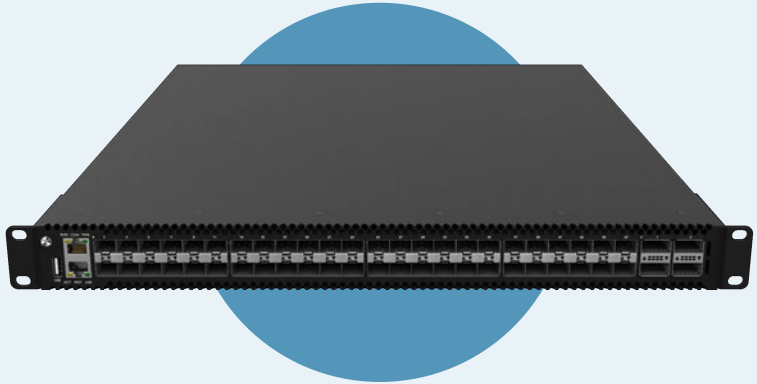
correlating with BGP protocol

For some applications, sampling can help to reduce the cost. This is such an example for routing and peering Analytics, 1 : 4 sampling ratio will work perfect for such application.



3U small site solution up to 180 Gbit

Omnia120

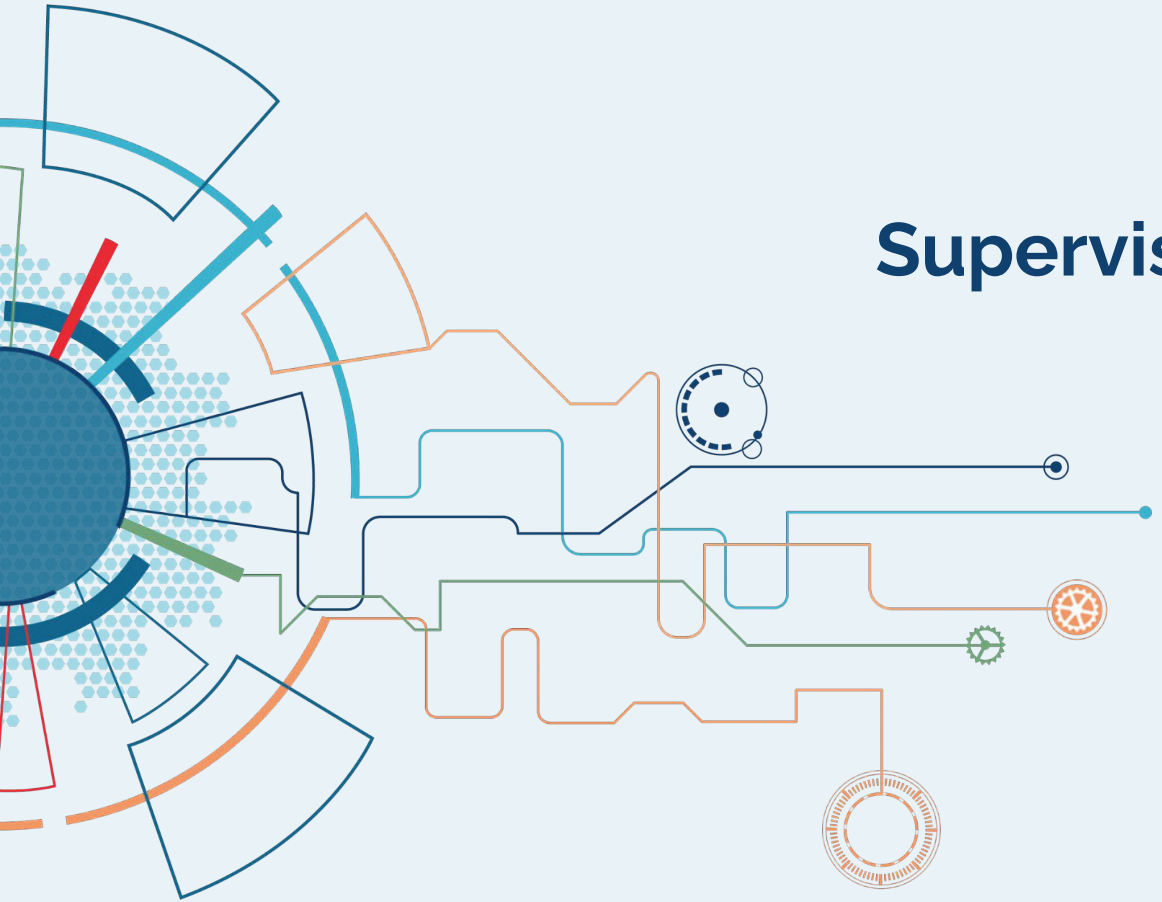


- Load balancer
- Filter
- Signaling Probe

Super server

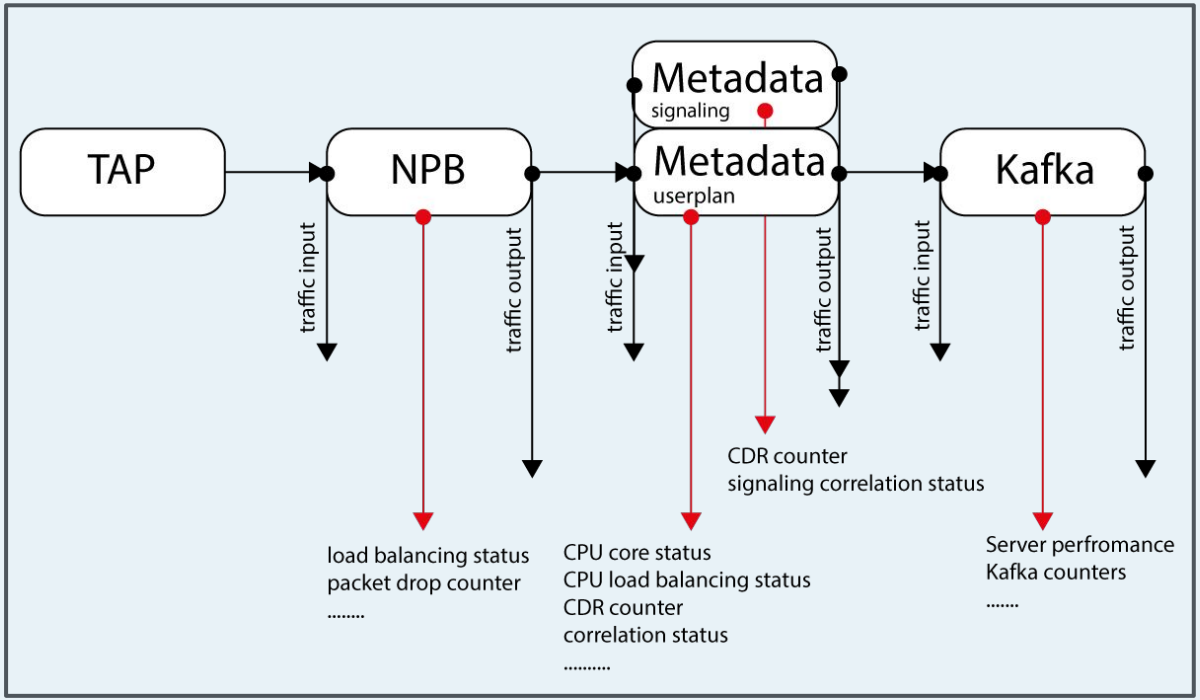
- Up to three Smart NICs for DPI analytics
- Performance up to 150 to 180 Gbit/s

User plane / Signaling correlation is done in the server CPU.



Supervision Solution

Managing and Monitoring Cubro DPI solution



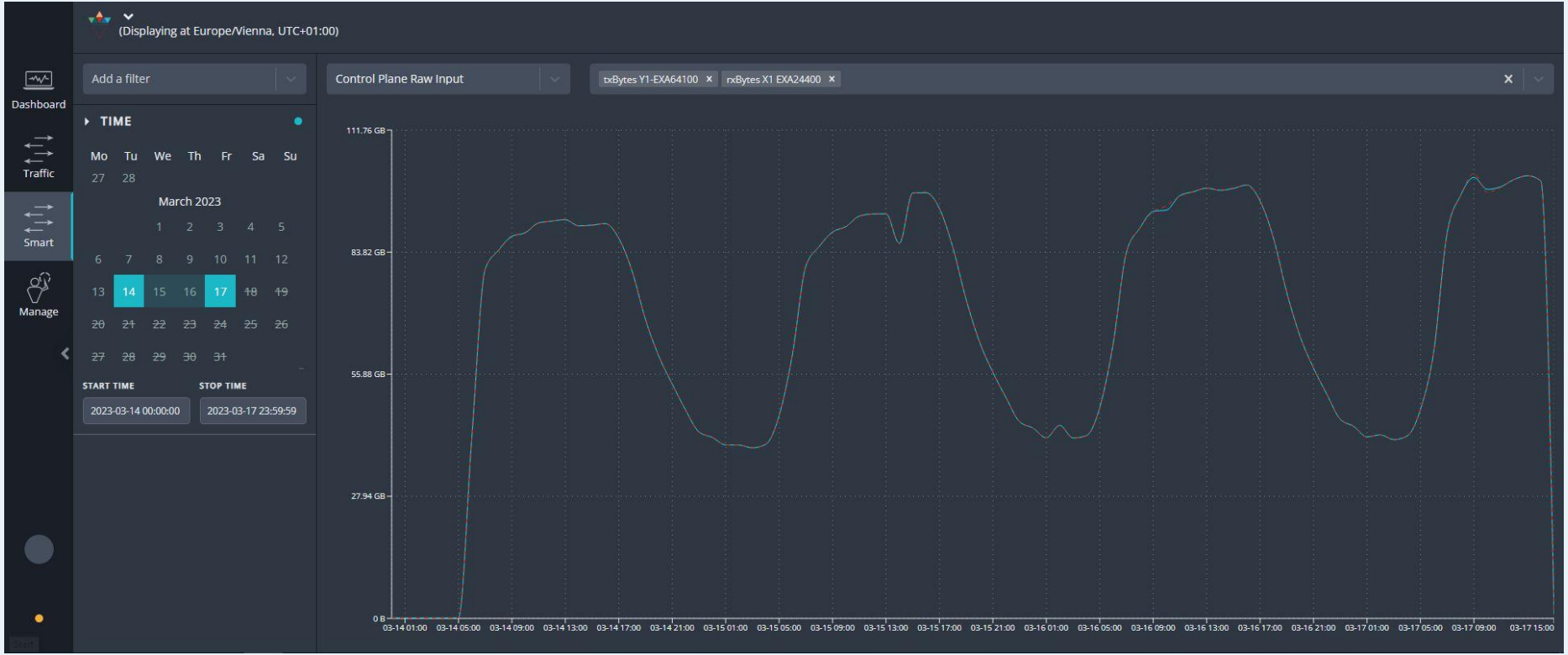
Managing and monitoring such a complex system is a significant issue. With its numerous components, the owner requires comprehensive information spanning the entire chain. The drawing shows the chain and all data collection points. But we are not solely collecting information (counters), we are building KPIs.

For instance, comparing input traffic on the NPB to the output in CDR or input traffic on all ports of the NPB versus output all ports. These KPIs can be configured by the customer themselves.

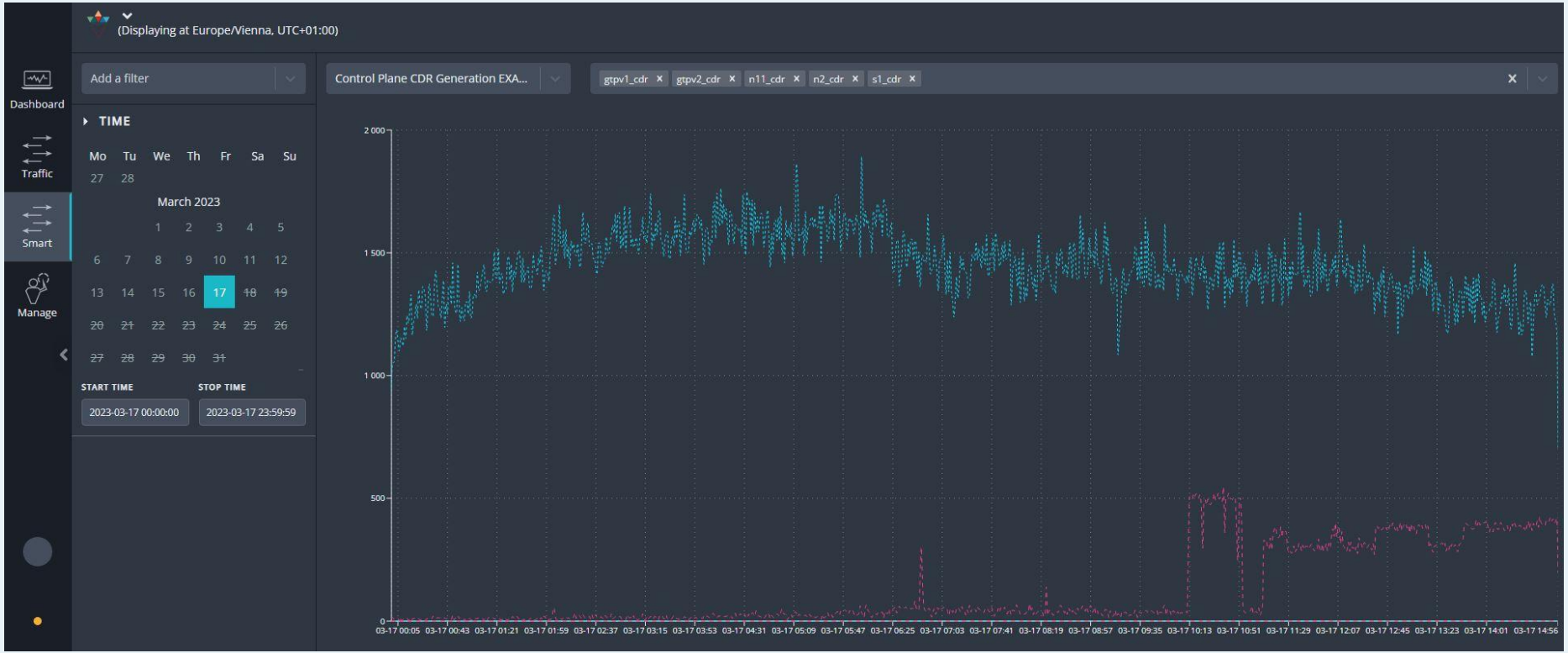
Graphical User Plane Output - Overflowing with data

Kafka Message output to data lake

Signaling Control plane Input



Control Plane CDR output



Only a specific CDR; in this case, N11

Technical Figures

Performance figures

(single use case estimations, can differ by traffic type and other external factors)

Product	IPFIX	IPFIX/DPI	Aggregated Cubro XDR with DPI (user plane)	Capture
Omnia120	20 Gbps on CPU 1	40 Gbps only one CPU 2	40 – 60 Gbps (on CPU 2)	Up to 6 TB / 8 - 10 Gbps performance
Omnia200	60 Gbps	60 Gbps	Up to 85 Gbps depending on the traffic situation	Up to 16 TB / 10 - 15 Gbps performance
Omnia400	60 Gbps (per CPU)	60 Gbps (per CPU)	160 Gbps (85 per CPU)	Up to 16 TB 10 - 15 Gbps performance
Omnic	40 Gbps	40 Gbps	Up to 85 Gbps depending on the traffic situation	No capture on NIC, but endless on the server via PCI
Omnic NG (road map Q1/23)	NA	NA	120+ Gbps depending on the traffic situation	No capture on NIC, but endless on the server via PCI

Raw packet storage calculation

Bandwidth in Gbit/s	1	3	6	12	24	Retention time in hours
0,1	0,08	0,25	0,5	1	2	
0,5	0,25	0,75	1,5	3	6	
1	0,46	1,375	2,75	5,5	11	
5	2,25	6,75	13,5	27	54	
10	4,50	13,5	27	54	108	
50	22,50	67,5	135	270	540	
100	45,00	135	270	540	1.080	in TB storage

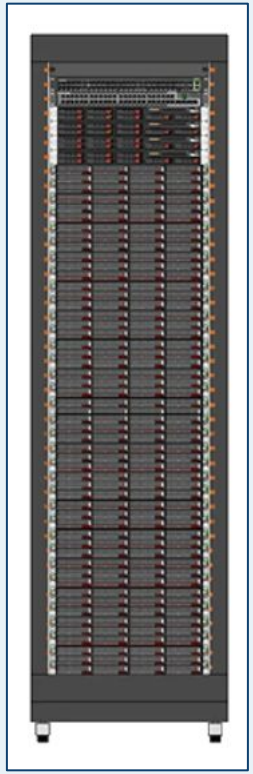
1024 TB = 1 PB Petabyte

Such a 42 Rack can support 3600 TB = 3,54 PB

432 x 8 TB HDD

2x 50A 208 3-Phase Metered PDU

Cost 600 - 890K Euro depending on CPU RAM and HDD type



Bandwidth in Gbit/s	1	5	10	30	60	90	Retention time in days
0,1	2	6	11	33	65	98	
0,5	6	27	54	162	324	468	
1	11	54	108	324	648	972	
5	54	270	540	1.620	3.240	4.860	
10	108	540	1.080	3.240	6.480	9.720	
50	540	2.700	5.400	16.200	32.400	48.600	
100	1.080	5.400	10.800	32.400	64.800	97.200	in TB storage

The huge difference in Volume to store - 2,5 PB to 29 PB

Estimated IPFIX Metadata retention time

Bandwidth in Gbit/s	1	5	10	30	60	90	Retention time in days
0,1	0,060	0,180	0,330	0,990	1,950	2,940	
0,5	0,180	0,810	1,620	4,860	9,720	14,040	
1	0,330	1,62	3,240	9,720	19,440	29,160	
5	1,62	8,10	16,20	48,60	97,20	145,80	
10	3,24	16,20	32,40	97,20	194,40	291,60	
50	16,20	81	162	486	972	1.458	
100	32,40	162	324	972	1.944	2.916	
500	162	810	1.620	4.860	9.720	14.580	
1000	324	1.620	3.240	9.720	19.440	29.160	in TB storage

Estimated Cubro Metadata retention time

Bandwidth in Gbit/s	1	5	10	30	60	90	Retention time in days
0,1	0,00	0,014	0,029	0,086	0,173	0,259	
0,5	0,01	0,072	0,144	0,432	0,864	1,296	
1	0,03	0,14	0,29	0,86	1,73	2,59	
5	0,29	1,44	2,88	8,64	17,28	25,92	
10	0,72	3,60	7,20	21,60	43,20	64,80	
50	1,44	7,20	14,40	43,20	86,40	129,60	
100	2,88	14,40	28,80	86,40	172,80	259,20	
500	14,40	72	144	432	864	1.296	
1000	28,80	144	288	864	1.728	2.592	in TB storage

Cubro aggregated XDR volume calculation

UP Input	CDR/sec	CDR/H	Volume JSON/H Mbyte	Volume Binary/H Mbyte	Volume JSON/H Gbyte	Volume Binary/H Gbyte
25 Mbit	1,70	6.200	25	3		
100 Mbit	6,80	24.800	100	12		
250 Mbit	17	62.000	250	30		
500 Mbit	34	124.000	500	60		
1 Gbit	68	248.000	1.000	120	1	0,12
10 Gbit	680	2.480.000	10.000	1.200	10	1
25 Gbit	1.700	6.200.000	25.000	3.000	25	3
50 Gbit	3.400	12.400.000	50.000	6.000	50	6
100 Gbit	6.800	24.800.000	100.000	12.000	100	12
500 Gbit	34.000	124.000.000	500.000	60.000	500	60
1 Tbit	68.000	248.000.000	1.000.000	120.000	1.000	120

User plane Data (This data is based on the real output from a European Mobile SP)

The output options

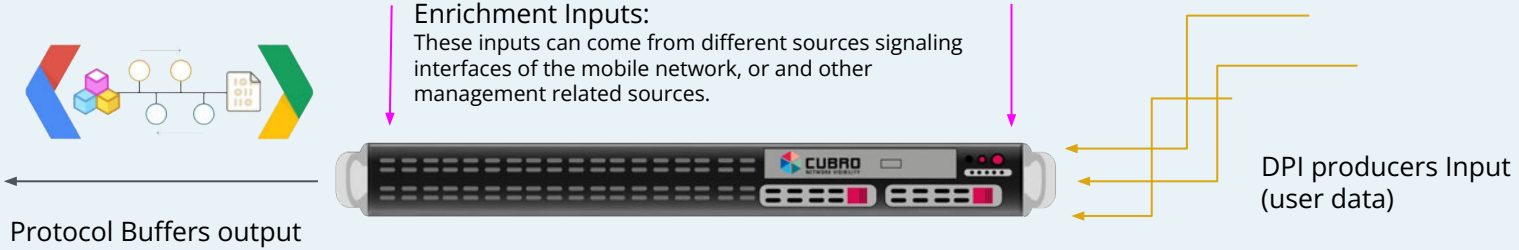
Cubro Metadata Broker

The Cubro Metadata Broker is a server based application available on Intel or ARM which receives the UDP streams from the DPI producers and performs several actions to generate an output stream which fulfills the customers requests.

- Data Enrichment from other sources
- Data Aggregation
- Time Aggregation
- Output Formation

The preferred output format is Google Protocol Buffers because it is a very efficient format. (see next page)

This format is supported by a modern programming language and is a very efficient way to serialize and deserialize data.



Protocol Buffers as Output

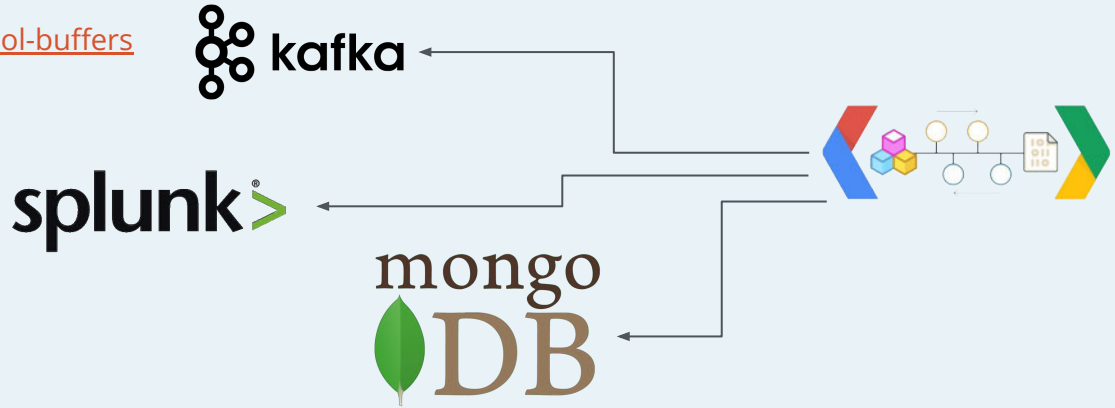
Google developed Protocol Buffers for use in their internal services. It is a binary encoding format that allows you to specify a *schema* for your data using a specification language.

Protocol Buffers offers several compelling advantages over JSON or YAML for sending data over the wire between internal services. While not a wholesale replacement for JSON, especially for services which are directly consumed by a web browser, Protocol Buffers offers very real advantages not only in the ways outlined above, but also typically, in terms of speed of encoding and decoding, size of the data on the wire, and more.

Confluent just updated their Kafka streaming platform with additional support for serializing data with Protocol buffers (or protobuf) and JSON Schema serialization. This makes integration much less difficult.

https://developers.google.com/protocol-buffers

Protocol Buffers are supported by all Big Data platforms



Proto Buffer Cubro XDR output schema and Jason

CUBRO NETWORK VISIBILITY

```
syntax = "proto3";

package com.messages.merlin;

option java_package =
"com.merlin.lib.pp.common.api.tw.protobuf";
option java_multiple_files = true;
option java_outer_classname = "Merlin";
option optimize_for = SPEED;

message TimeWindow {
  int64 timestamp = 1;
  int64 bps = 2;
  int64 incomingPkts = 3;
  int64 outgoingPkts = 4;
  int64 internalPkts = 5;
  int64 incomingBytes = 6;
  int64 outgoingBytes = 7;
  int64 internalBytes = 8;
  int32 connections = 9;
  repeated ServiceData services = 10;
  repeated ClientData clients = 11;
  repeated IpData servers = 12;
  repeated TypeBytesData l3 = 13;
  repeated TypeBytesData l4 = 14;
  repeated PortData ports = 15;
  repeated DnsData domains = 16;
}
message ServiceData {
  ServiceEntry service = 1;
  int64 incomingPkts = 2;
  int64 outgoingPkts = 3;
  int64 internalPkts = 4;
  int64 incomingBytes = 5;
  int64 outgoingBytes = 6;
  int64 internalBytes = 7;
  repeated IpData clients = 8;
  repeated IpData servers = 9;
}
message ClientData {
  IpData ip = 1;
  repeated ServiceEntry services = 2;
  repeated IpData servers = 3;
}
```

```
message PortData {
  TypeBytesData port = 1;
  repeated IpData clients = 2;
}

/* represents a generic entry for an IP, bytes (volume), and geo location
information */
message IpData {
   IpEntry ip = 1;
   int64 incomingPkts = 2;
   int64 outgoingPkts = 3;
   int64 internalPkts = 4;
   int64 incomingBytes = 5;
   int64 outgoingBytes = 6;
   int64 internalBytes = 7;
   GeoData geo = 8;
   oneof oneof_mac {
     bytes mac = 9;
   }

}

message ServiceEntry {
  string name = 1;
  oneof oneof_hash {
    int64 hash = 2;
  }
}

/* represents a generic entry for IPv4 or IPv6 Address */
message IpEntry {
  oneof oneof_ip {
    int32 ip4 = 1;
    bytes ip6 = 2;
  }
}

message GeoData {
  float lon = 1;
  float lat = 2;
}

/* represents a generic entry for some type and bytes mapping */
message TypeBytesData {
   int32 type = 1;
   int64 incomingPkts = 2;
   int64 outgoingPkts = 3;
   int64 internalPkts = 4;
   int64 incomingBytes = 5;
   int64 outgoingBytes = 6;
   int64 internalBytes = 7;
}

message DnsData {
  string name = 1;
  repeated IpData clients = 2;
  repeated IpData servers = 3;
}
```

IPFIX output from Custos

No.	Time	Source	Destination	Protocol	Length	Info
4	4.997546	10.30.0.10	10.30.1.61	CFLOW	252	IPFIX flow (210 bytes) Obs-Domain-ID= 1 [Data:258]
5	9.997553	10.30.0.10	10.30.1.61	CFLOW	442	IPFIX flow (400 bytes) Obs-Domain-ID= 1 [Data:258]
6	14.999700	10.30.0.10	10.30.1.61	CFLOW	537	IPFIX flow (495 bytes) Obs-Domain-ID= 1 [Data:258]
7	14.999704	10.30.0.10	10.30.1.61	CFLOW	118	IPFIX flow (76 bytes) Obs-Domain-ID= 1 [Data-Template:256]
8	19.997696	10.30.0.10	10.30.1.61	CFLOW	134	IPFIX flow (92 bytes) Obs-Domain-ID= 1 [Data-Template:257]
9	19.997698	10.30.0.10	10.30.1.61	CFLOW	134	IPFIX flow (92 bytes) Obs-Domain-ID= 1 [Data-Template:258]
10	25.002232	10.30.0.10	10.30.1.61	CFLOW	252	IPFIX flow (210 bytes) Obs-Domain-ID= 1 [Data:258]
11	29.971536	10.30.0.10	10.30.1.61	CFLOW	1411	IPFIX flow (1369 bytes) Obs-Domain-ID= 1 [Data:257]
12	35.000297	10.30.0.10	10.30.1.61	CFLOW	133	IPFIX flow (91 bytes) Obs-Domain-ID= 1 [Data:257]
13	35.000310	10.30.0.10	10.30.1.61	CFLOW	822	IPFIX flow (780 bytes) Obs-Domain-ID= 1 [Data:258]
14	35.000313	10.30.0.10	10.30.1.61	CFLOW	118	IPFIX flow (76 bytes) Obs-Domain-ID= 1 [Data-Template:256]
15	40.003088	10.30.0.10	10.30.1.61	CFLOW	134	IPFIX flow (92 bytes) Obs-Domain-ID= 1 [Data-Template:257]
16	40.003091	10.30.0.10	10.30.1.61	CFLOW	134	IPFIX flow (92 bytes) Obs-Domain-ID= 1 [Data-Template:258]
17	41.276563	10.30.20.75	10.30.255.255	NBNS	92	Name query NB WPAD<00>
18	42.026401	10.30.20.75	10.30.255.255	NBNS	92	Name query NB WPAD<00>

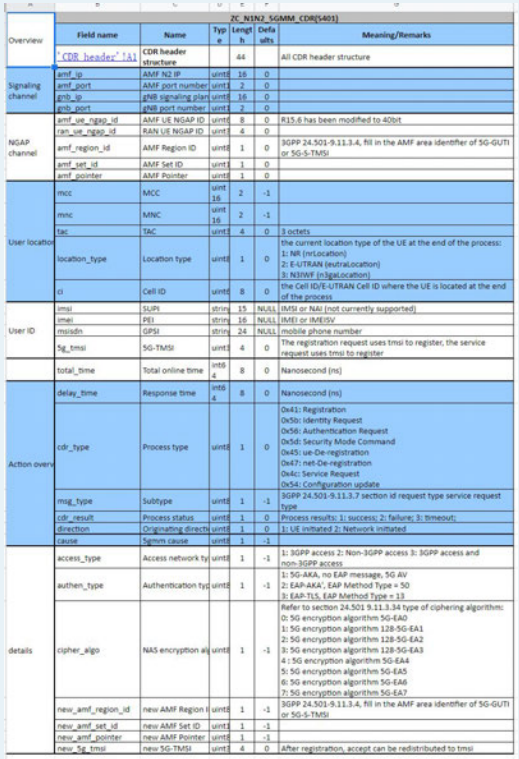
```
> Internet Protocol Version 4, Src: 10.30.0.10, Dst: 10.30.1.61
> User Datagram Protocol, Src Port: 4739, Dst Port: 9995
v Cisco NetFlow/IPFIX
    Version: 10
    Length: 1369
  v Timestamp: Sep 18, 2020 10:57:14.000000000 Mitteleuropäische Sommerzeit
      ExportTime: 1600419434
  > FlowSequence: 224072 (expected 223958)
    Observation Domain Id: 1
  v Set 1 [id=257] (19 flows)
      FlowSet Id: (Data) (257)
      FlowSet Length: 1353
      [Template Frame: 2]
    v Flow 1
        InputInt: 3
        OutputInt: 2
        Packets: 10
      > [Duration: 0.117622944 seconds (nanoseconds)]
        Source Mac Address: Dell_8e:75:8b (a4:4c:c8:8e:75:8b)
        Destination Mac Address: SuperMic_1b:1a:5f (ac:1f:6b:1b:1a:5f)
        Ethernet Type: 8
        SrcAddr: 192.168.3.36
        DstAddr: 99.86.245.33
        Protocol: TCP (6)
        Octets: 929
        SrcPort: 59828 (59828)
        DstPort: 443 (443)
      > TCP Flags: 0x001b, ACK, PSH, SYN, FIN
        Enterprise Private entry: (cubro) Type 1: Value (hex bytes): f3 06
```

1436	pexip	Pexip	Messaging	Meeting and interoperability media platform
1437	phantasy_star_online_2	Phantasy Star Online 2	Gaming	Online role-playing game published by Sega
1438	phantom_fex	Phantom EFX	Gaming	Card, casino and gambling games
1439	phantom_vpn	Phantom VPN	Remoting	Multi-protocol Virtual Private Network
1440	philips	Philips	Remoting	Consumer and business electronics
1441	philips_hue	Philips Hue	Remoting	Wirelessly controlled LED lamps
1442	photomath	Photomath	Education	Mathematical camera calculator
1443	photorank	Photorank	Advertising	Measuring user photo-sharing activity
1444	piccollage	PicCollage	Photo	Photo, slicker and text collages
1445	picsart	PicsArt	Photo	Image editing, collage and drawing service
1446	picturethis	PictureThis	Education	Online plant encyclopedia
1447	pinger	Pinger	Messaging	Texting and calling service
1448	pinger_call	Pinger Call	Messaging	Voice and video calling
1449	pinterest	Pinterest	Social	Discover and save photos
1450	pivotal_tracker	Pivotal Tracker	Productivity	Project planning software
1451	pixalate	Pixalate	Advertising	Advertising fraud intelligence and marketing compliance platform
1452	pixelberry	Pixelberry Studios	Gaming	Video game traffic by Pixelberry
1453	pjl	PJL	Networking	Printer job management service
1454	plarium	Plarium	Gaming	Video game traffic by Plarium
1455	playdemic	Playdemic	Gaming	Video game traffic by Playdemic
1456	playerunknowns_battlegrounds	PlayerUnknown's Battlegrounds	Gaming	Battle royale video game
1457	playfab	PlayFab	Gaming	Backend game platform as a service
1458	playgendary	Playgendary	Gaming	Video game traffic by Playgendary
1459	playhaven	Playhaven	Advertising	Mobile gaming monetization platform
1460	playrix	Playrix	Gaming	Video game traffic by Playrix
1461	playtika	Playtika	Gaming	Casino style video games
1462	plentyoffish	Plentyoffish	Social	Online dating service
1463	plex	Plex	Video	Audio and video media player services
1464	plexvpn	PlexVPN	Remoting	Multi-protocol Virtual Private Network
1465	plume	Plume	Productivity	Plume WiFi services
1466	plustransfer	PlusTransfer	File Transfer	Large file transfer for email
1467	plutotv	PlutoTV	Video	American internet television service
1468	pnrp	PNRP	Networking	Peer-to-peer name resolution service
1469	pocket	Pocket	News	Article reading list management service
1470	pocketcamp	Animal Crossing: Pocket Camp	Gaming	Social simulation video game
1471	pocket_controller	Pocket Controller	Remoting	Remote control help desk software
1472	pokec	Pokec	Social	Online social network
1473	pokemon_cafe_mix	Pokemon Cafe Mix	Gaming	Puzzle video game
1474	pokemon_go	Pokemon GO	Gaming	Augmented reality video game
1475	pokemon_home	Pokemon Home	Gaming	Cloud service for Pokemon gaming
1476	pokemon_lets_go	Pokemon Let's Go	Gaming	Adventure Role-Playing video game
1477	pokemonmasters	Pokemon Masters	Gaming	Pokemon strategy and battling game
1478	pokemon_sword_shield	Pokemon Sword and Shield	Gaming	Role-playing video game
1479	polyup	Polyup	Education	3D playground for learning math
1480	pop3	POP3	Messaging	Standard email protocol
1481	pops_lots	POP! Slots	Gaming	Casino gambling video game
1482	poshmark	Poshmark	Shopping	Social commerce marketplace

Signaling Probe output

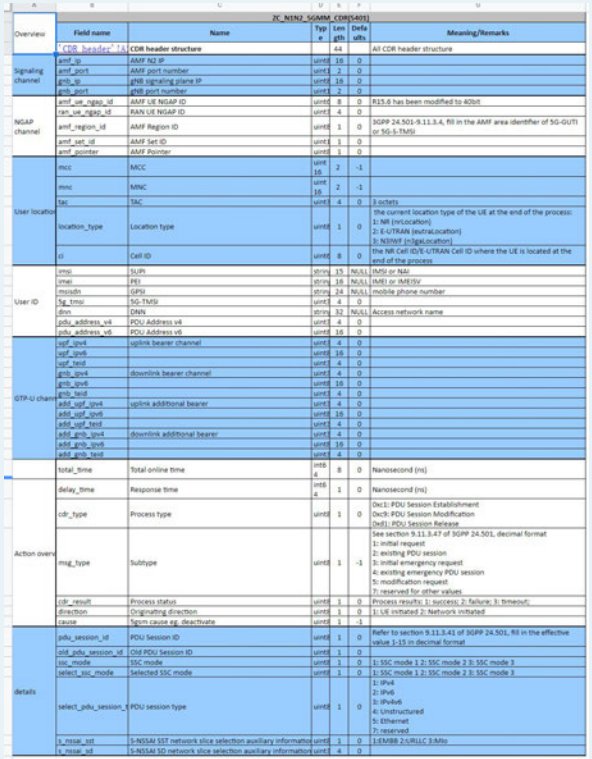
CUBRO NETWORK VISIBILITY

NAS 5GMM



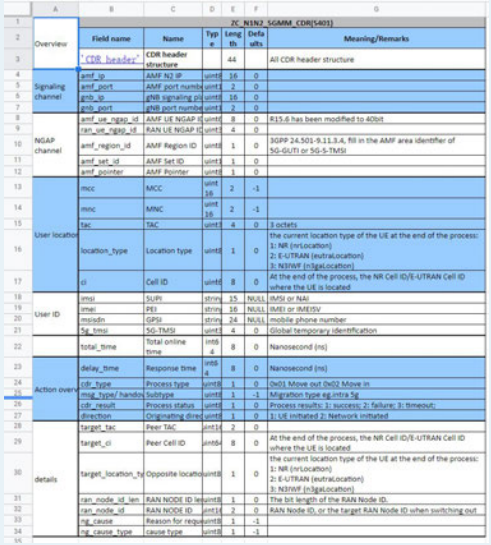
	Field name	Name	Type	Length	Defaults	Meaning/Remarks
	ZC_N1N2_5GMM_CDR(5401)					
Overview	'_CDR_header'!A1	CDR header structure		44		All CDR header structure
Signaling channel	amf_ip	AMF N2 IP	uint16	16	0	
	amf_port	AMF port number	uint16	2	0	
	gnb_ip	gNB signaling plan	uint16	16	0	
	gnb_port	gNB port number	uint16	2	0	
NGAP channel	amf_ue_ngap_id	AMF UE NGAP ID	uint64	8	0	R15.6 has been modified to 40bit
	ran_ue_ngap_id	RAN UE NGAP ID	uint32	4	0	
	amf_region_id	AMF Region ID	uint8	1	0	3GPP 24.501-9.11.3.4, fill in the AMF area identifier of 5G-GUTI or 5G-S-TMSI
	amf_set_id	AMF Set ID	uint16	1	0	
	amf_pointer	AMF Pointer	uint8	1	0	
User location	mcc	MCC	uint16	2	-1	
	mnc	MNC	uint16	2	-1	
	tac	TAC	uint32	4	0	3 octets
	location_type	Location type	uint8	1	0	the current location type of the UE at the end of the process: 1: NR (nrLocation) 2: E-UTRAN (eutraLocation) 3: N3IWF (n3gaLocation)
	ci	Cell ID	uint64	8	0	the Cell ID/E-UTRAN Cell ID where the UE is located at the end of the process
User ID	imsi	SUPI	string	15	NULL	IMSI or NAI (not currently supported)
	imei	PEI	string	16	NULL	IMEI or IMEISV
	msisdn	GPSI	string	24	NULL	mobile phone number
	5g_tmsi	5G-TMSI	uint32	4	0	The registration request uses tmsi to register, the service request uses tmsi to register
	total_time	Total online time	int64	8	0	Nanosecond (ns)
Action overv	delay_time	Response time	int64	8	0	Nanosecond (ns)
	cdr_type	Process type	uint8	1	0	0x41: Registration 0x5b: Identity Request 0x56: Authentication Request 0x5d: Security Mode Command 0x45: ue-De-registration 0x47: net-De-registration 0x4c: Service Request 0x54: Configuration update
	msg_type	Subtype	uint8	1	-1	3GPP 24.501-9.11.3.7 section id request type service request type
	cdr_result	Process status	uint8	1	0	Process results: 1: success; 2: failure; 3: timeout;
	direction	Originating direct	uint8	1	0	1: UE initiated 2: Network initiated
	cause	5gmm cause	uint8	1	-1	
details	access_type	Access network ty	uint8	1	-1	1: 3GPP access 2: Non-3GPP access 3: 3GPP access and non-3GPP access
	authen_type	Authentication tyc	uint8	1	-1	1: 5G-AKA, no EAP message, 5G AV 2: EAP-AKA', EAP Method Type = 50 3: EAP-TLS, EAP Method Type = 13
	cipher_algo	NAS encryption al	uint8	1	-1	Refer to section 24.501 9.11.3.34 type of ciphering algorithm: 0: 5G encryption algorithm 5G-EA0 1: 5G encryption algorithm 128-5G-EA1 2: 5G encryption algorithm 128-5G-EA2 3: 5G encryption algorithm 128-5G-EA3 4 : 5G encryption algorithm 5G-EA4 5: 5G encryption algorithm 5G-EA5 6: 5G encryption algorithm 5G-EA6 7: 5G encryption algorithm 5G-EA7
	new_amf_region_id	new AMF Region I	uint8	1	-1	3GPP 24.501-9.11.3.4, fill in the AMF area identifier of 5G-GUTI or 5G-S-TMSI
	new_amf_set_id	new AMF Set ID	uint16	1	-1	
	new_amf_pointer	new AMF Pointer	uint8	1	-1	
	new_5g_tmsi	new 5G-TMSI	uint32	4	0	After registration, accept can be redistributed to tmsi

NAS 5GSM



	Field name	Name	Type	Length	Defaults	Meaning/Remarks
	ZC_N1N2_5GMM_CDR(5401)					
Overview	'_CDR_header'!A1	CDR header structure		44		All CDR header structure
Signaling channel	amf_ip	AMF N2 IP	uint16	16	0	
	amf_port	AMF port number	uint16	2	0	
	gnb_ip	gNB signaling plane IP	uint16	16	0	
	gnb_port	gNB port number	uint16	2	0	
NGAP channel	amf_ue_ngap_id	AMF UE NGAP ID	uint64	8	0	R15.6 has been modified to 40bit
	ran_ue_ngap_id	RAN UE NGAP ID	uint32	4	0	
	amf_region_id	AMF Region ID	uint8	1	0	3GPP 24.501-9.11.3.4, fill in the AMF area identifier of 5G-GUTI or 5G-S-TMSI
	amf_set_id	AMF Set ID	uint16	1	0	
	amf_pointer	AMF Pointer	uint8	1	0	
User location	mcc	MCC	uint16	2	-1	
	mnc	MNC	uint16	2	-1	
	tac	TAC	uint32	4	0	3 octets
	location_type	Location type	uint8	1	0	the current location type of the UE at the end of the process: 1: NR (nrLocation) 2: E-UTRAN (eutraLocation) 3: N3IWF (n3gaLocation)
	ci	Cell ID	uint64	8	0	the NR Cell ID/E-UTRAN Cell ID where the UE is located at the end of the process
User ID	imsi	SUPI	string	15	NULL	IMSI or NAI
	imei	PEI	string	16	NULL	IMEI or IMEISV
	msisdn	GPSI	string	24	NULL	mobile phone number
	5g_tmsi	5G-TMSI	uint32	4	0	
	dnn	DNN	string	32	NULL	Access network name
	pdu_address_v4	PDU Address v4	uint32	4	0	
	pdu_address_v6	PDU Address v6	uint16	16	0	
GTP-U chanr	upf_ipv4	uplink bearer channel	uint32	4	0	
	upf_ipv6		uint16	16	0	
	upf_teid		uint32	4	0	
	gnb_ipv4	downlink bearer channel	uint32	4	0	
	gnb_ipv6		uint16	16	0	
	gnb_teid		uint32	4	0	
	add_upf_ipv4	uplink additional bearer	uint32	4	0	
	add_upf_ipv6		uint16	16	0	
	add_upf_teid		uint32	4	0	
	add_gnb_ipv4	downlink additional bearer	uint32	4	0	
	add_gnb_ipv6		uint16	16	0	
	add_gnb_teid		uint32	4	0	
	total_time	Total online time	int64	8	0	Nanosecond (ns)
Action overv	delay_time	Response time	int64	1	0	Nanosecond (ns)
	cdr_type	Process type	uint8	1	0	0x1: PDU Session Establishment 0x9: PDU Session Modification 0x41: PDU Session Release
	msg_type	Subtype	uint8	1	-1	See section 9.11.3.47 of 3GPP 24.501, decimal format 1: initial request 2: existing PDU session 3: initial emergency request 4: existing emergency PDU session 5: modification request 7: reserved for other values
	cdr_result	Process status	uint8	1	0	Process results: 1: success; 2: failure; 3: timeout;
	direction	Originating direction	uint8	1	0	1: UE initiated 2: Network initiated
	cause	5gsm cause eg. deactivate	uint8	1	-1	
details	pdu_session_id	PDU Session ID	uint8	1	0	Refer to section 9.11.3.41 of 3GPP 24.501, fill in the effective value 1-15 in decimal format
	old_pdu_session_id	Old PDU Session ID	uint8	1	0	
	ssc_mode	SSC mode	uint8	1	0	1: SSC mode 1 2: SSC mode 2 3: SSC mode 3
	select_ssc_mode	Selected SSC mode	uint8	1	0	1: SSC mode 1 2: SSC mode 2 3: SSC mode 3
	select_pdu_session_t	PDU session type	uint8	1	0	1: IPv4 2: IPv6 3: IPv4v6 4: Unstructured 5: Ethernet 7: reserved
	s_nssai_sst	S-NSSAI SST network slice selection auxiliary information	uint8	1	0	1:eMBB 2:URLLC 3:Mlo
	s_nssai_sd	S-NSSAI SD network slice selection auxiliary information	uint8	4	0	

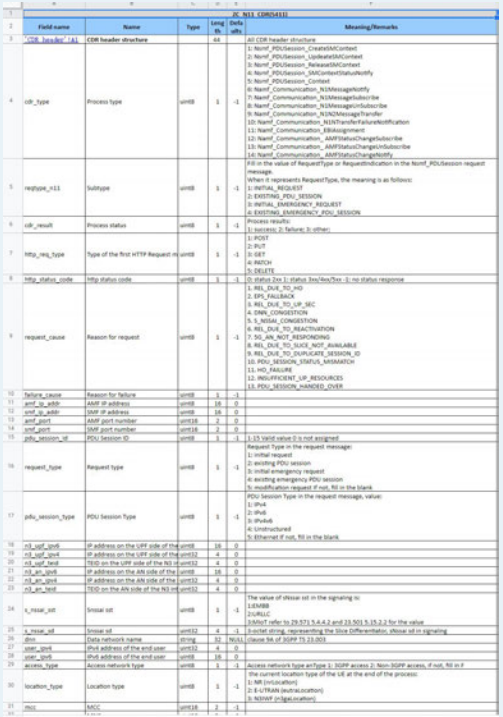
N2 Handover



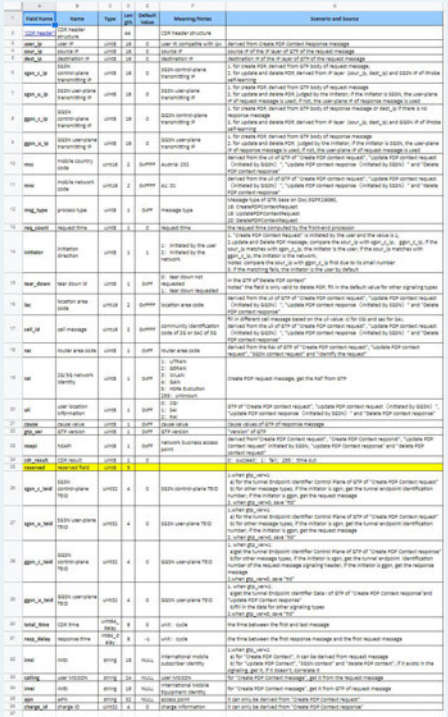
	Field name	Name	Type	Length	Defaults	Meaning/Remarks
	ZC_N1N2_5GMM_CDR(5401)					
Overview	'_CDR_header'	CDR header structure		44		All CDR header structure
Signaling channel	amf_ip	AMF N2 IP	uint16	16	0	
	amf_port	AMF port number	uint16	2	0	
	gnb_ip	gNB signaling plane IP	uint16	16	0	
	gnb_port	gNB port number	uint16	2	0	
NGAP channel	amf_ue_ngap_id	AMF UE NGAP ID	uint64	8	0	R15.6 has been modified to 40bit
	ran_ue_ngap_id	RAN UE NGAP ID	uint32	4	0	
	amf_region_id	AMF Region ID	uint8	1	0	3GPP 24.501-9.11.3.4, fill in the AMF area identifier of 5G-GUTI or 5G-S-TMSI
	amf_set_id	AMF Set ID	uint16	1	0	
	amf_pointer	AMF Pointer	uint8	1	0	
User location	mcc	MCC	uint16	2	-1	
	mnc	MNC	uint16	2	-1	
	tac	TAC	uint32	4	0	3 octets
	location_type	Location type	uint8	1	0	the current location type of the UE at the end of the process: 1: NR (nrLocation) 2: E-UTRAN (eutraLocation) 3: N3IWF (n3gaLocation)
	ci	Cell ID	uint64	8	0	At the end of the process, the NR Cell ID/E-UTRAN Cell ID where the UE is located
User ID	imsi	SUPI	string	15	NULL	IMSI or NAI
	imei	PEI	string	16	NULL	IMEI or IMEISV
	msisdn	GPSI	string	24	NULL	mobile phone number
	5g_tmsi	5G-TMSI	uint32	4	0	Global temporary identification
	total_time	Total online time	int64	8	0	Nanosecond (ns)
Action overv	delay_time	Response time	int64	8	0	Nanosecond (ns)
	cdr_type	Process type	uint8	1	0	0x01 Move out 0x02 Move in
	msg_type/ handov	Subtype	uint8	1	-1	Migration type eg.intra 5g
	cdr_result	Process status	uint8	1	0	Process results: 1: success; 2: failure; 3: timeout;
	direction	Originating direc	uint8	1	0	1: UE initiated 2: Network initiated
details	target_tac	Peer TAC	uint32	2	0	
	target_ci	Peer Cell ID	uint64	8	0	At the end of the process, the NR Cell ID/E-UTRAN Cell ID where the UE is located
	target_location_ty	Opposite locatio	uint8	1	0	the current location type of the UE at the end of the process: 1: NR (nrLocation) 2: E-UTRAN (eutraLocation) 3: N3IWF (n3gaLocation)
	ran_node_id_len	RAN NODE ID le	uint8	1	0	The bit length of the RAN Node ID.
	ran_node_id	RAN NODE ID	uint16	2	0	RAN Node ID, or the target RAN Node ID when switching out
	ng_cause	Reason for requ	uint8	1	-1	
	ng_cause_type	cause type	uint8	1	-1	

Signaling Probe output

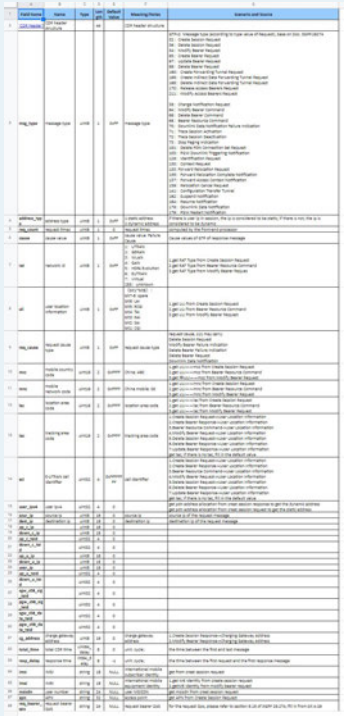
N11 Signaling



GTP



GTPV2



We have operations in all time zones.
Reach us at: support@cubro.com