



Καλό απόγευμα!

H O M E R

Lorenzo Mangani

- Co-Founder and CEO @ QXIP BV w/ CTO Alexandr Dubovikov
- Proud father of 4 kids and a hundreds of odd github repositories
- Boring into payloads, events, statistics and logs since 2007

qxip BV

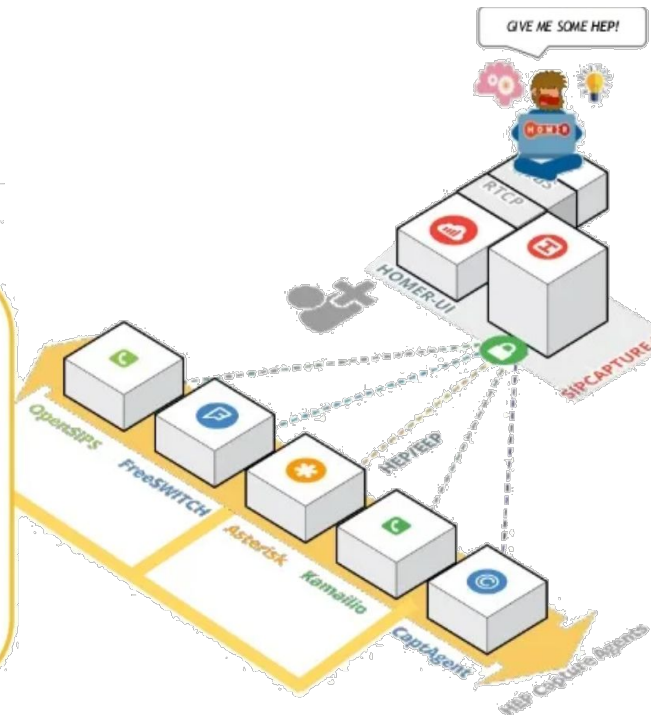
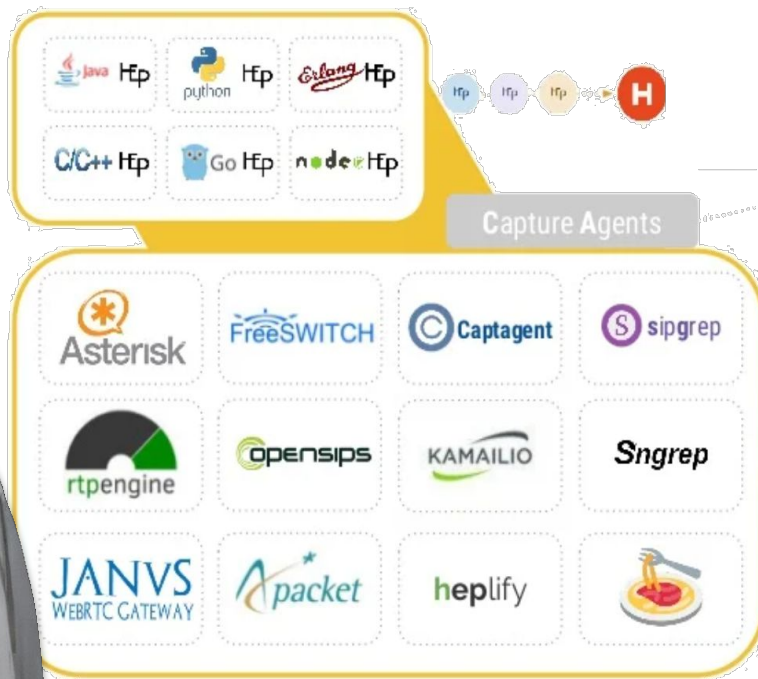
- Research & Development of OSS & Commercial Monitoring Technologies
- Headquarters in Amsterdam (NL) w/ Operations Center in Valencia (ES)
- Worldwide Deployments - from tiny Startups up to Fortune Corporations
- Running a Healthy, Self-Sustainable Business Model w/ great Partnerships
- Robin-Hood Open-Source Model, Commercial Customers sponsoring OSS Devel





ΨΟΥ ΚΝΩΩ

HOMER





H O M E R

Our mission at **qxip** is simple and complex at once

> *Crafting great monitoring solutions for Humans*

The **HOMER** stack has always been focused on *open source integrations through HEP* - and we were **right!**

Every single week dozens of new users and companies worldwide turn to our solutions to resolve challenges. Why? *Our tools save them time, and time is valuable.*

(we have zero donations, but hepic sponsors everything)



2022

Asterisk	87
Kamailio	66
OpenSIPS	65
Homer	63
FreeSWITCH	61
rtengine	61
SIPp	43
ICI Dial	38



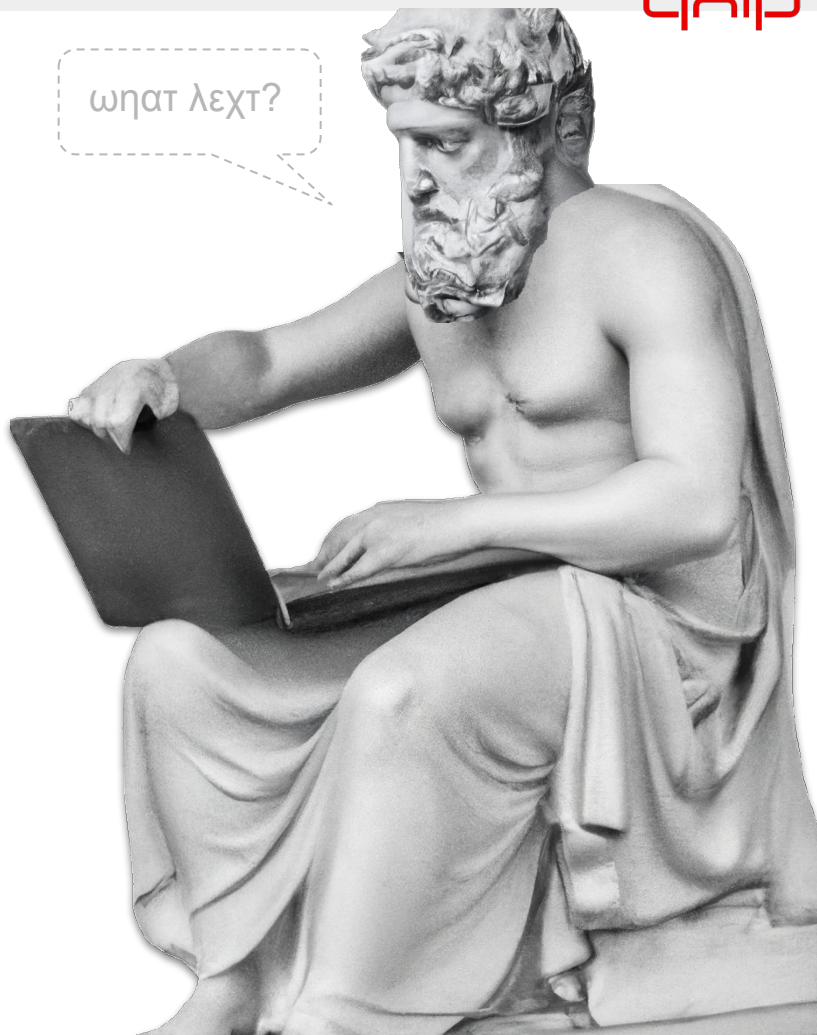
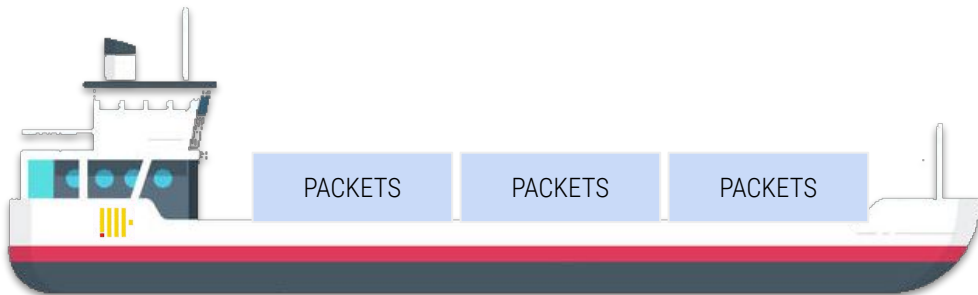
HOMER

qkio

This confirms our **ideas** and **solutions** are still very *actual and extremely useful - but is that enough?*

We're quite good (*some even say great*) at the art of monitoring **VoIP** and **RTC** but *nothing lasts forever...*

Thousands of deployments later, *we know the job is no longer limited to sniffing packets and protocols*

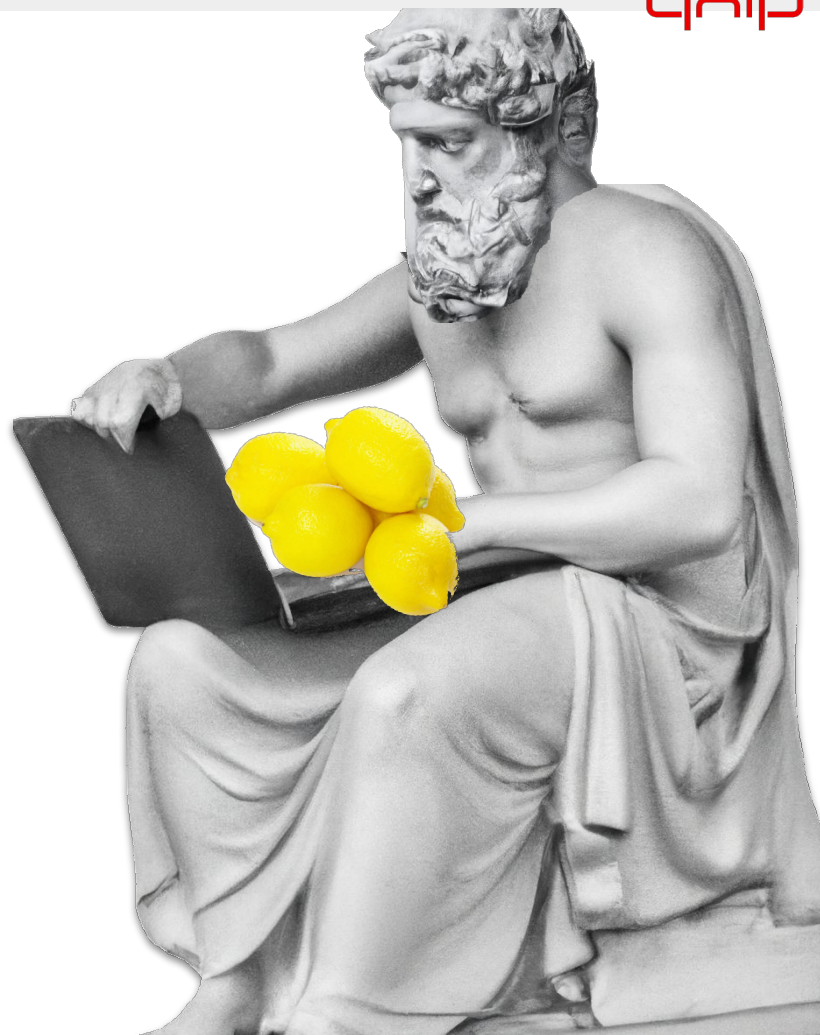
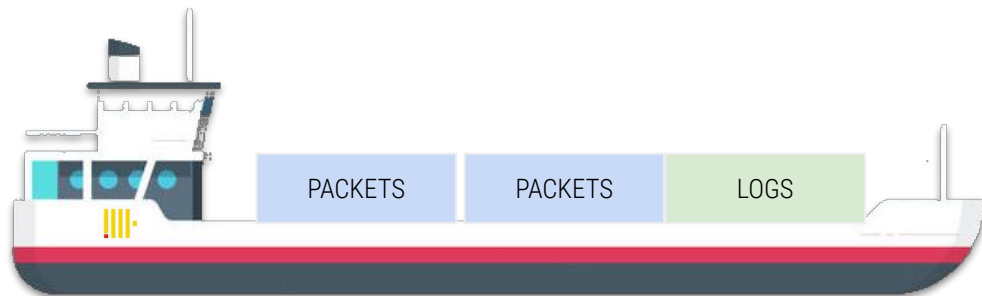




First it was Logs

What a great idea. Let's add some integrations!

+ *Influx, Elastic, Loki, CDRs, SBC logs and more...*

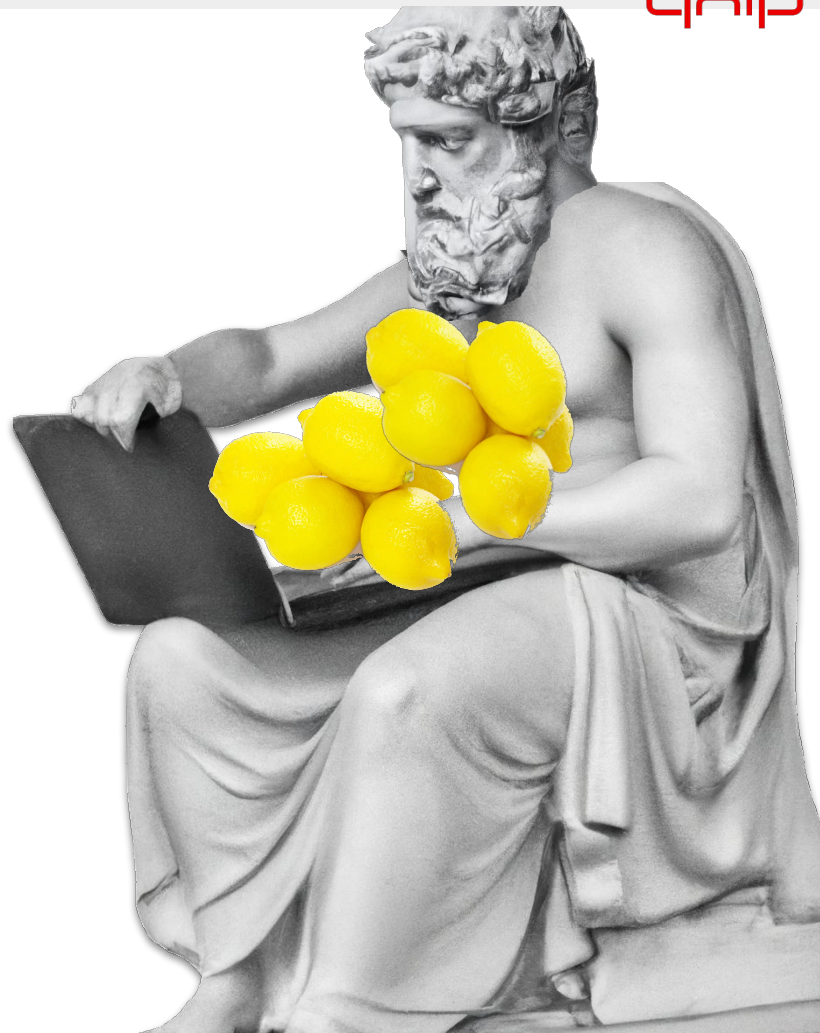
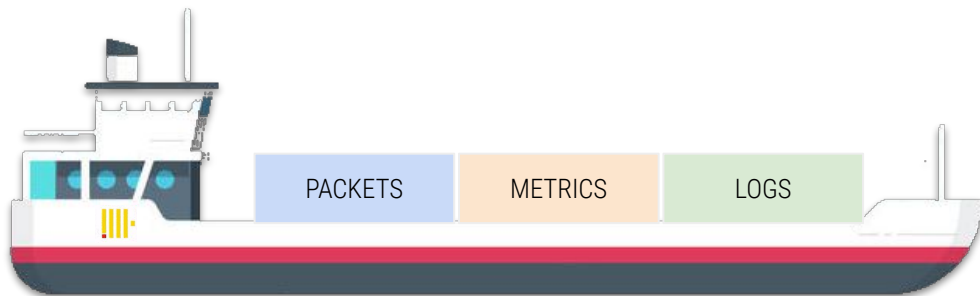




Then it was Metrics.

How could we live without this. *Let's export them!*

- + *Influx, Elastic, Loki, CDRs, SBC logs and more...*
- + *Prometheus, Influxdb, Timescale and others....*

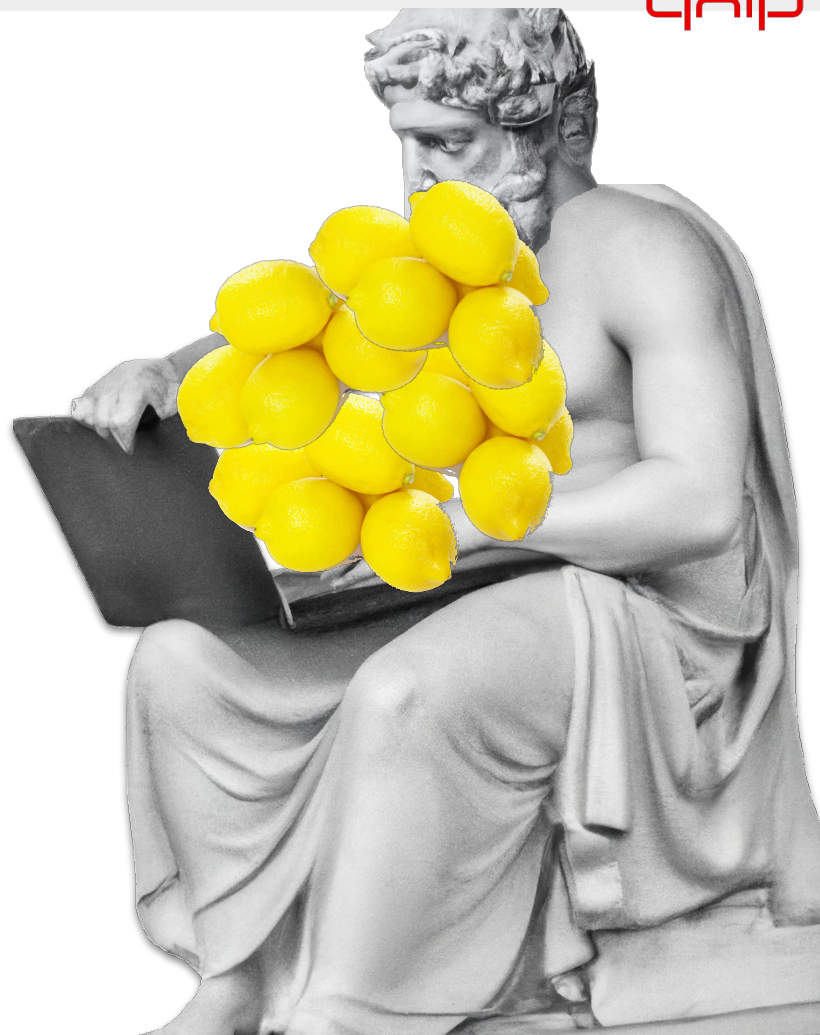
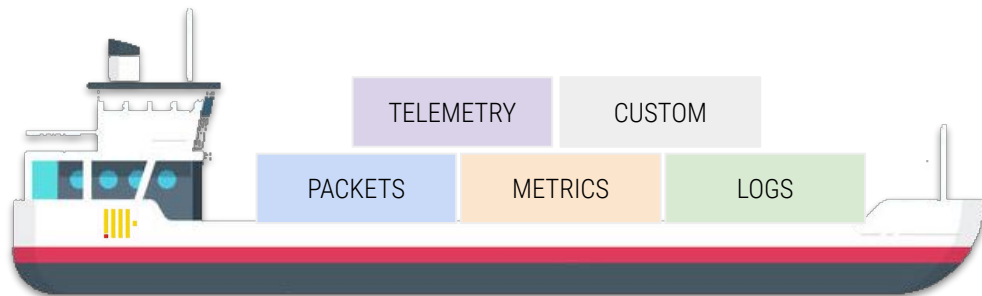




Then Telemetry became cool.

Nice addiction to the mix. *Let's trace stuff!*

- + *Influx, Elastic, Loki, CDRs, SBC logs and more...*
- + *Prometheus, Influxdb, Timescale and others....*
- + *Open-Telemetry, Jaeger, Zipkin, Tempo...*

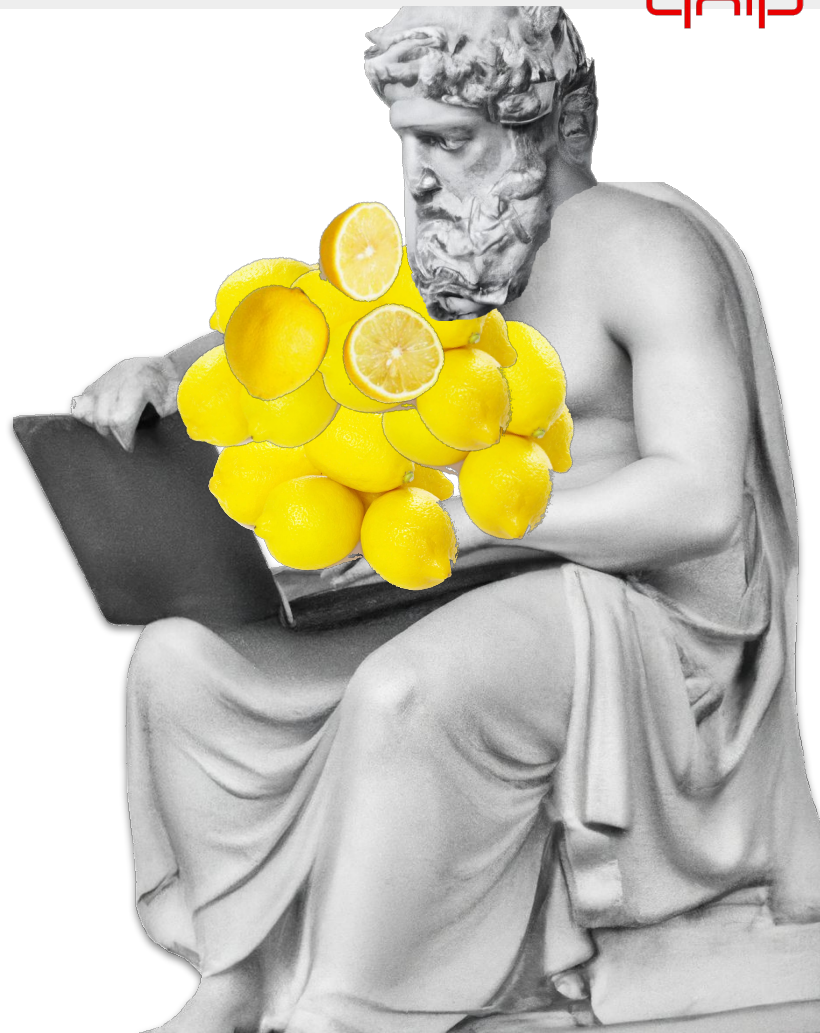
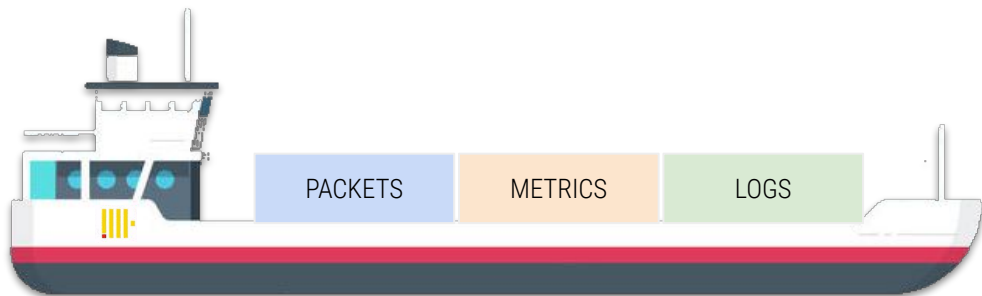




HOMER

QHIP

Some users became confused. "Too many options!"



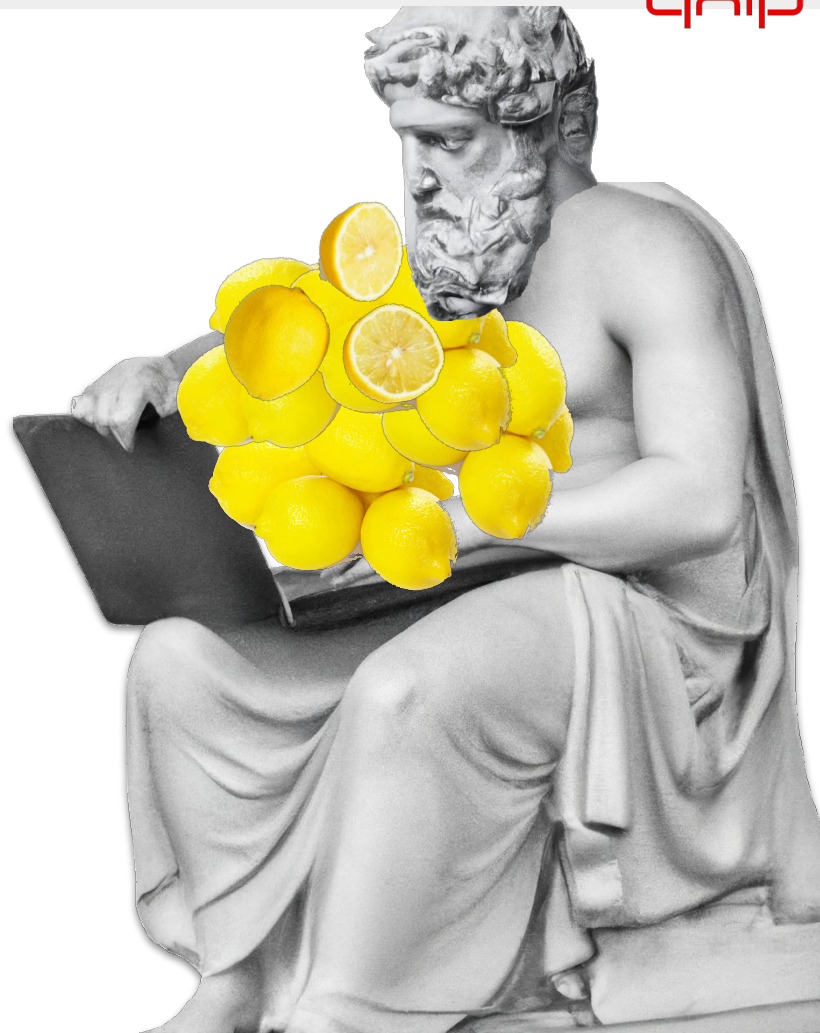
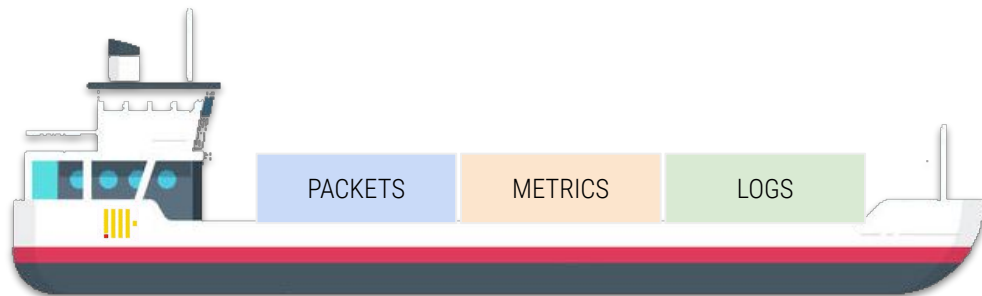


HOMER

qhip

Some users became confused. "Too many options!"

Right. So we turned to standard integrations.





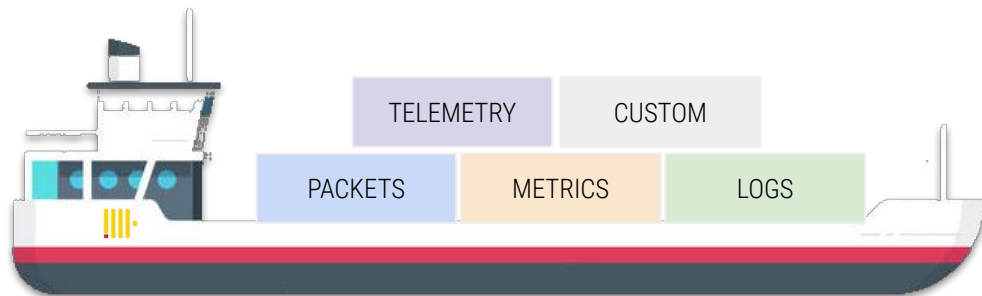
HOMER

qhip

Some users became confused. "Too many options!"

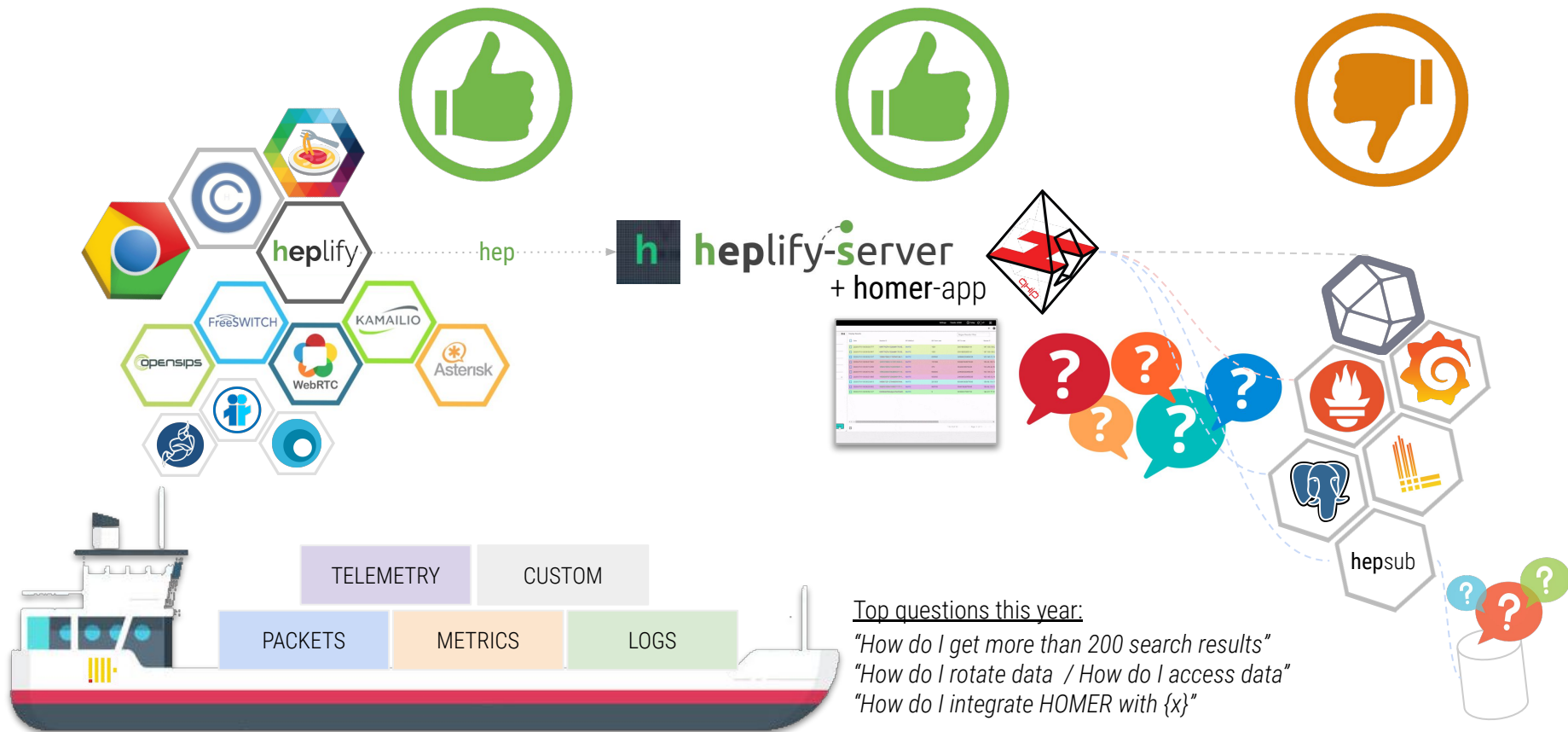
Right. So we turned to standard integrations.

Other users became upset. "Too many integrations!"





HOMER





Time to **reset**. So what do users (!= developers) really want?

- **Everything** preferably without any type of work involved
- **No complex choices** and decision to make - things just work
- **No new standards** or languages to learn, zero learning curve
- **No special protocols** required to ingest data into the system
- **No limits** when it comes to integrations with other tools

*“SOME FAMOUS WORDS
WOULD FIT GREAT HERE”*





Time to **reset**. So what do users (!= developers) really want?

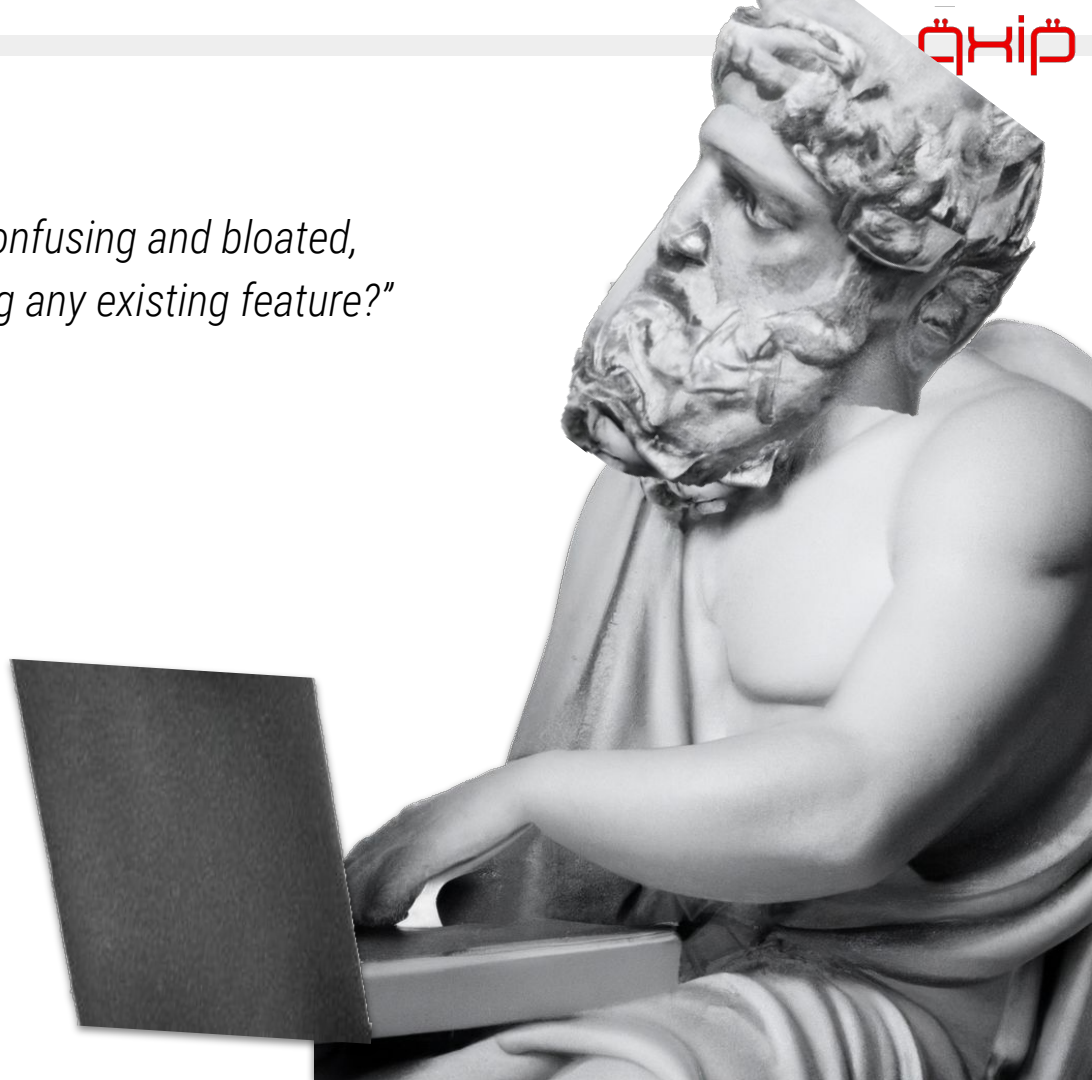
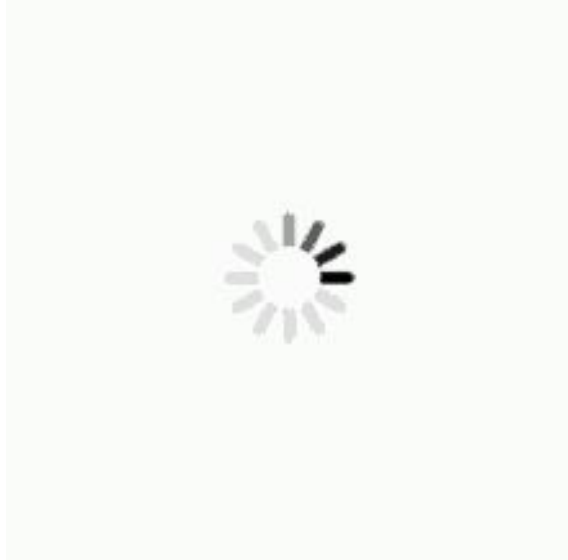
- **Everything** preferably without any type of work involved
- **No complex choices** and decision to make - things just work
- **No new standards** and languages to learn, zero learning curve
- **No special expertise** required to ingest data into the system
- **No limits** when it comes to integrations with other tools

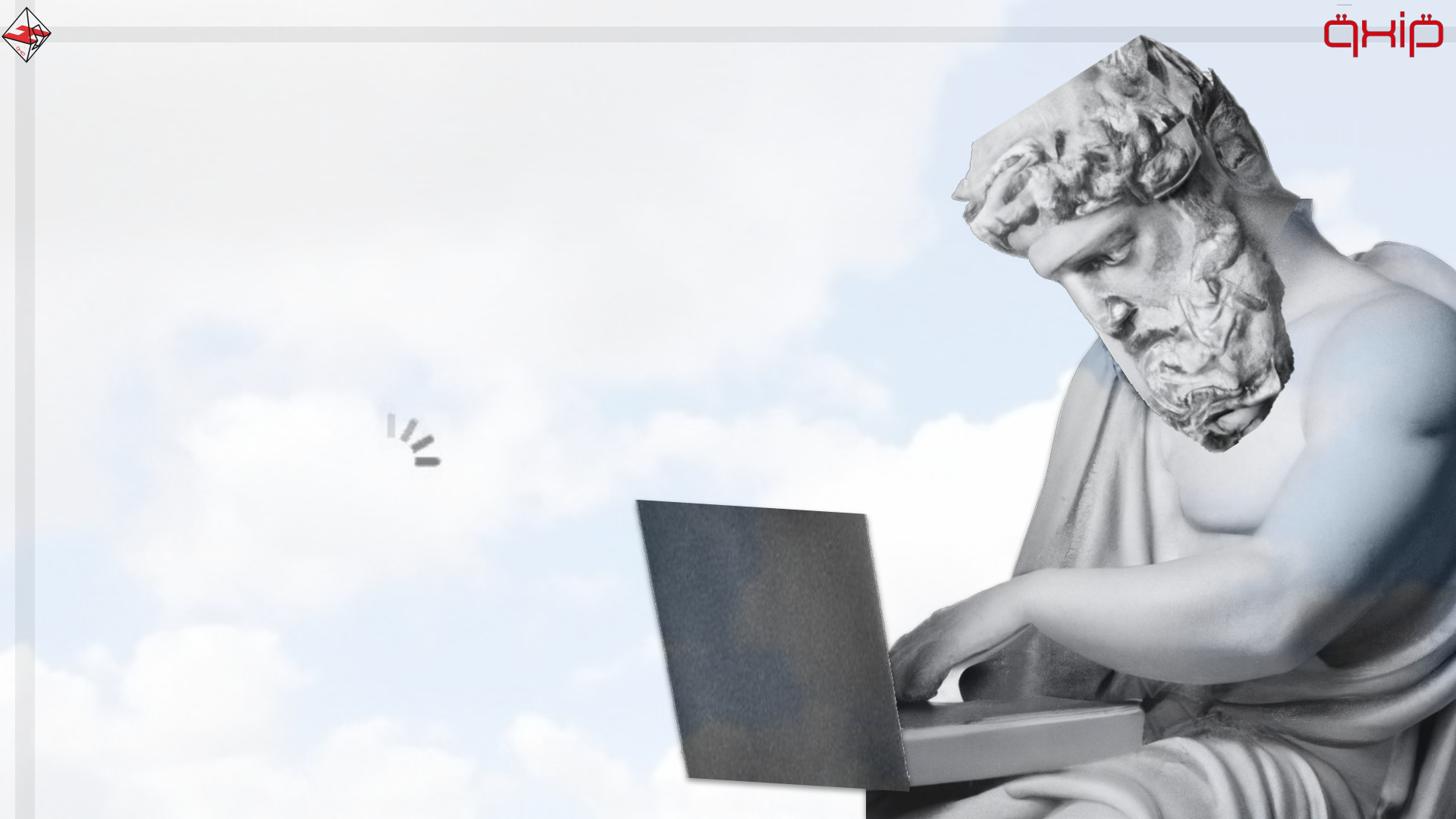
*“SOME FAMOUS WORDS
WOULD FIT GREAT HERE”*





*"How could we get rid of everything that's confusing and bloated,
and still expand our horizons without losing any existing feature?"*





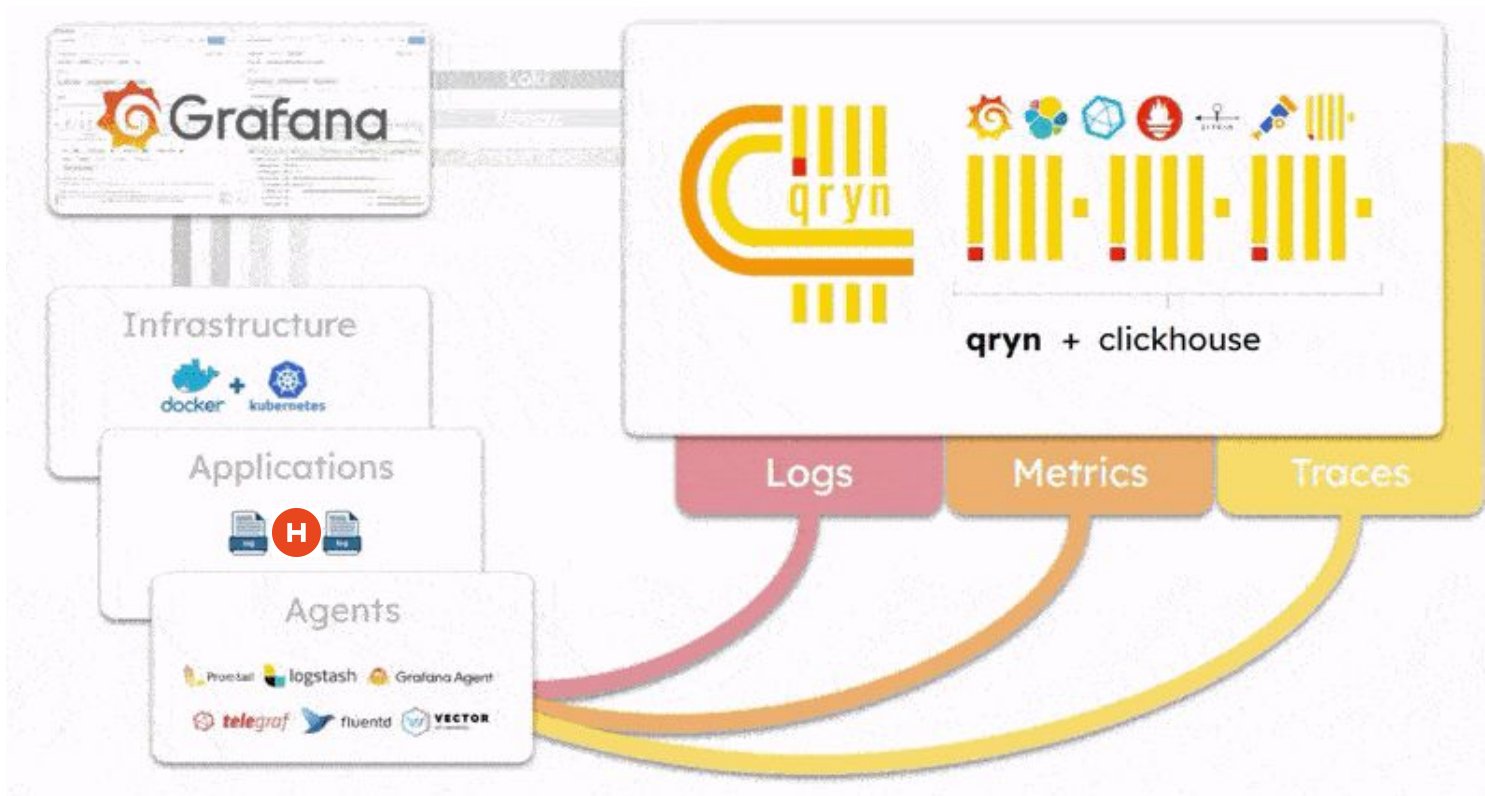




OUR FINAL SOLUTION:
EVERYTHING BECOMES STANDARD *OBSERVABILITY*



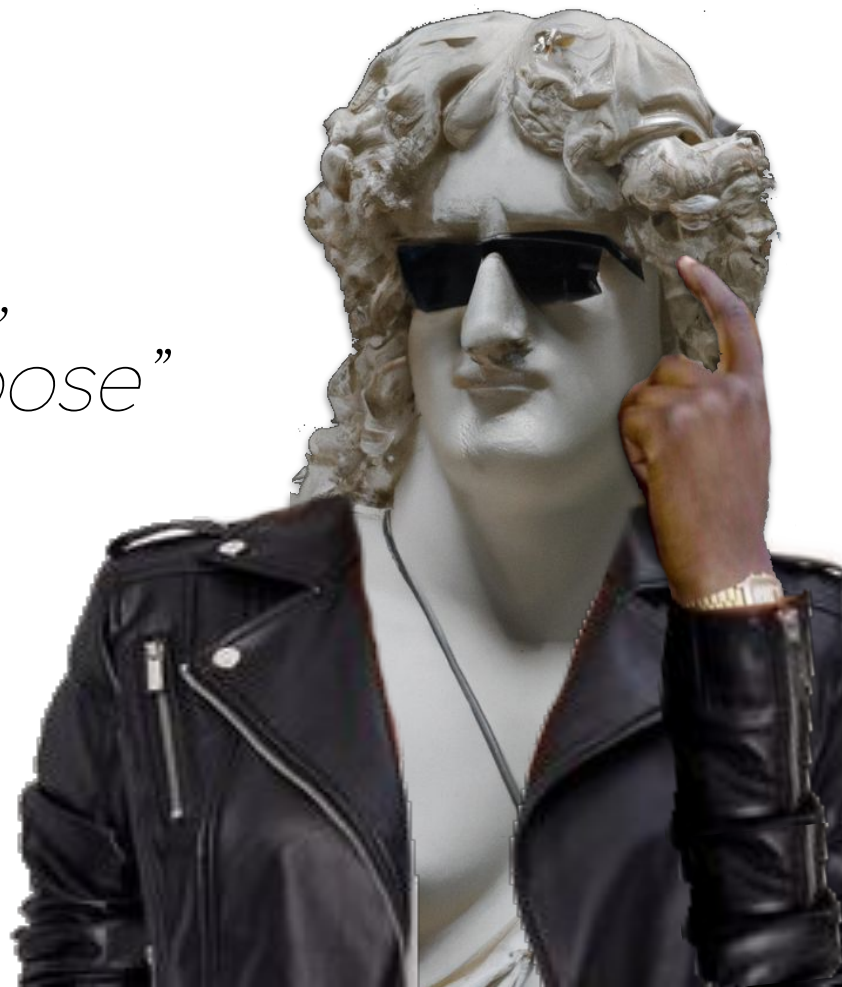
qryn: polyglot monitoring, built on top of Clickhouse





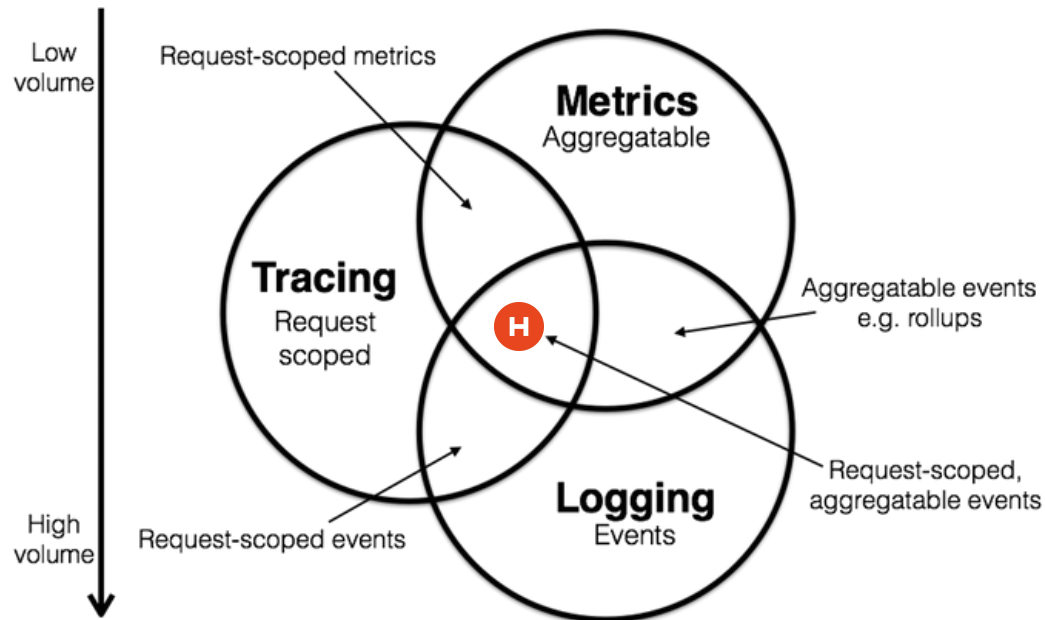
qryn: polyglot monitoring

*“You can’t be wrong,
if you don’t have to choose”*





qryn: polyglot monitoring





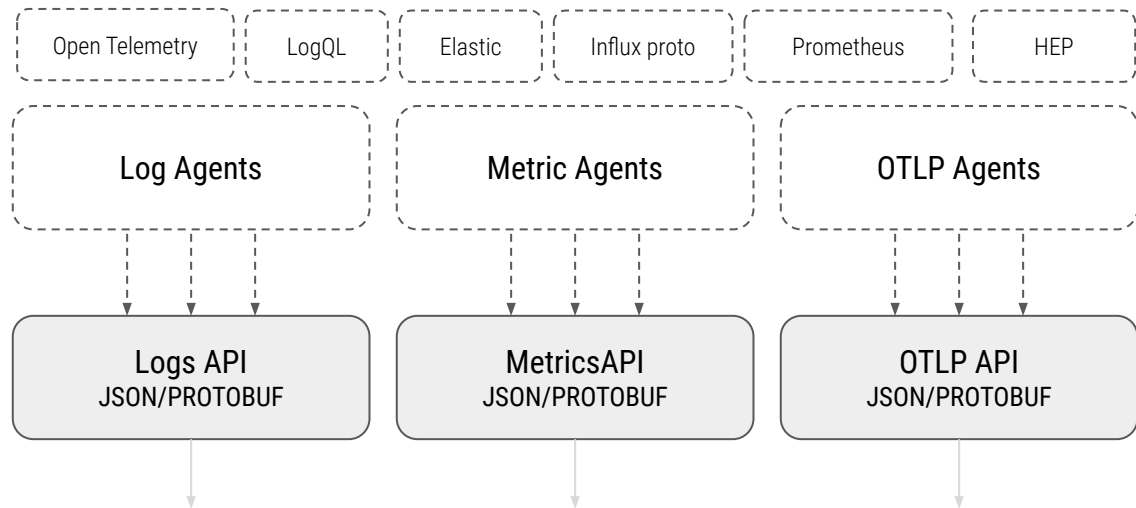
qryn: polyglot monitoring



HEP VoIP/RTC Agents

Industry Standard Protocols

Industry Standard Agents



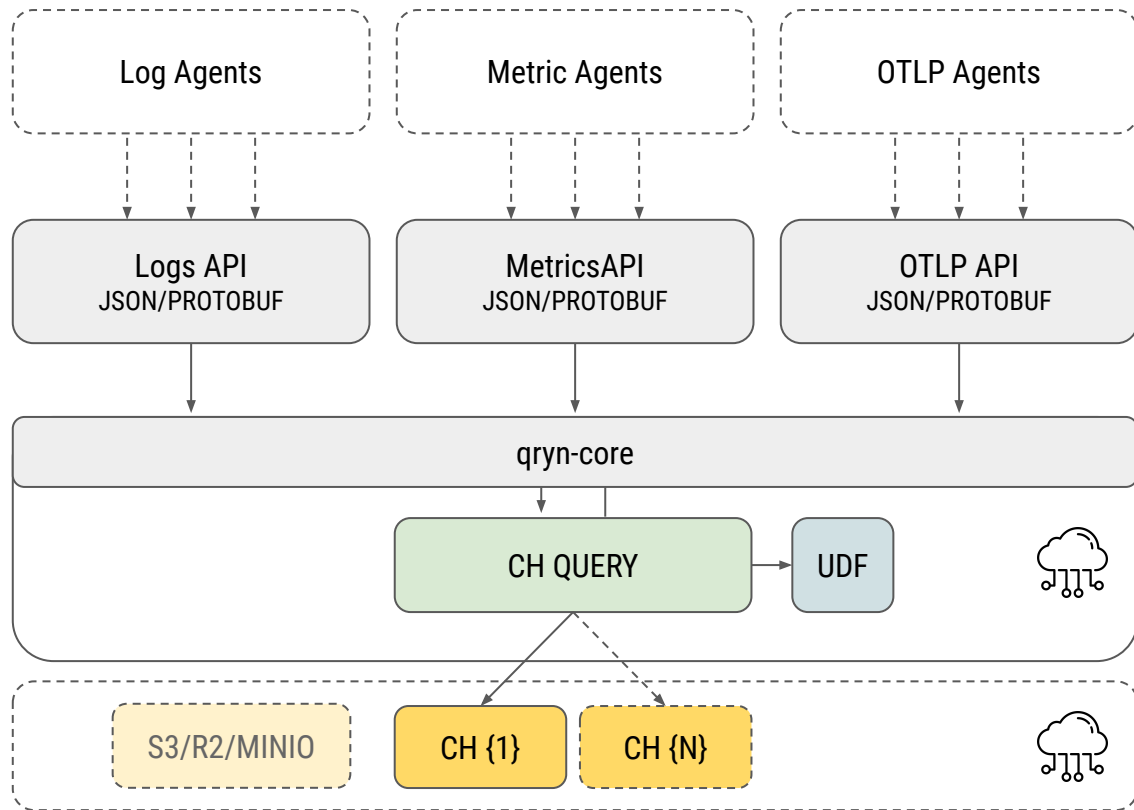
API endpoints

Transparent inputs

QRYN APIs



qryn: polyglot monitoring



QRYN Cloud / On Premise

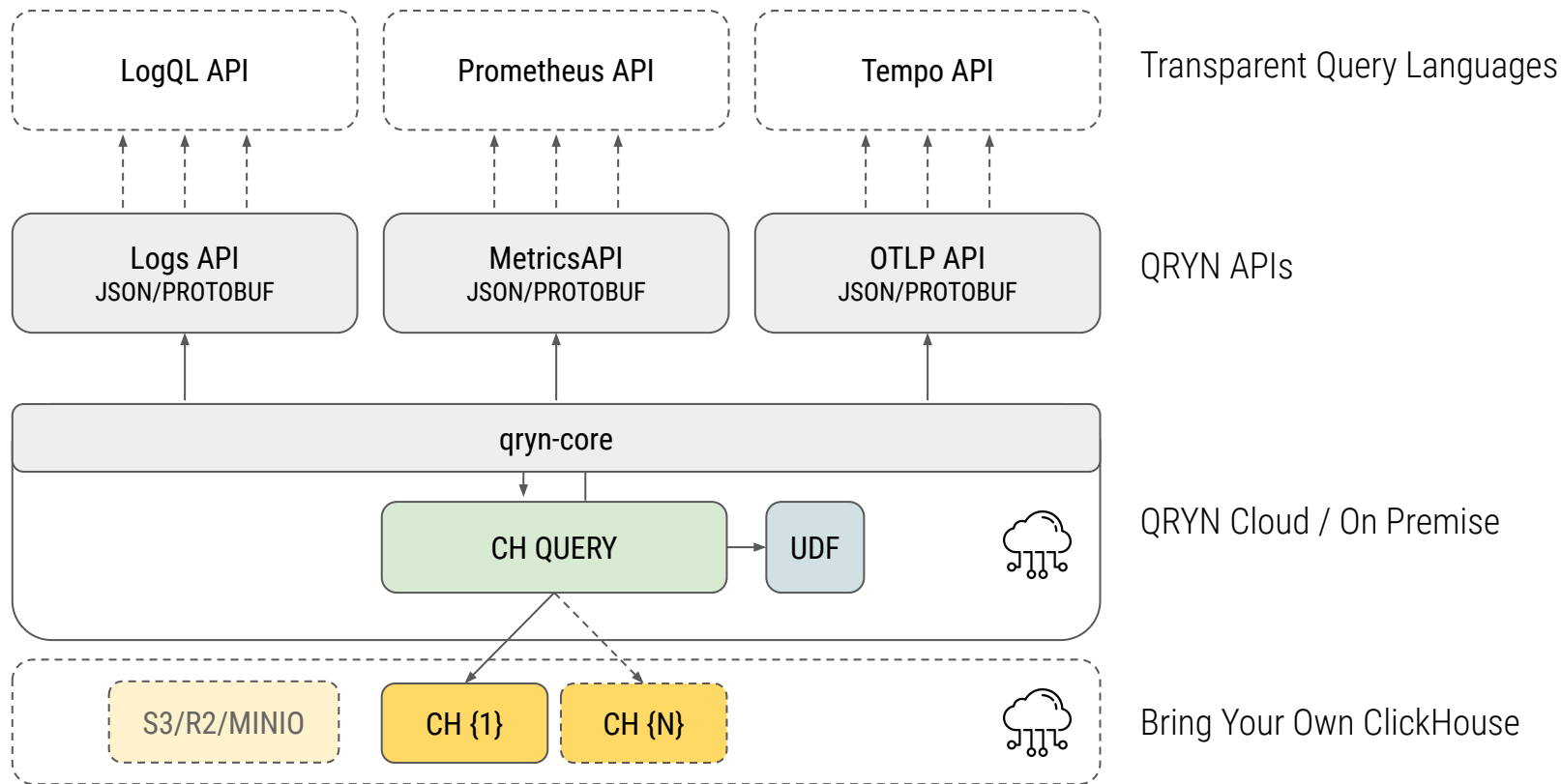
~2M/inserts/second on 8 cores

~500k/inserts/thread/second

Bring Your Own ClickHouse



qryn: polyglot monitoring





EXAMPLES PLEASE



qryn: sip search and filtering with logql

[Log browser >](#)`{job="heplify-server"} |= "rport"`

> Options Type: Range

Common labels: f96a6912a55a heplify-server 2001 sip Line limit: 1000 (36 returned)

▼ 2022-09-27 23:31:58 SIP/2.0 200 OK

Via: SIP/2.0/UDP 127.0.0.1:5080;branch=z9hG4bK3.dYFaLV;rport=5080

Record-Route: <sip:112.111.251.155;lr;ftag=06DE7CEB-56E458BB000864AD-B855F700>

Record-Route: <sip:204.250.2.55;r2=on;lr=on;ftag=06DE7CEB-56E458BB000864AD-B855F700;lb=yes>

Record-Route: <sip:127.0.0.1;r2=on;lr=on;ftag=06DE7CEB-56E458BB000864AD-B855F700;lb=yes>

From: <sip:+9555884468@sipcapture.org>;tag=06DE7CEB-56E458BB000864AD-B855F700

To: <sip:148@112.111.251.155>;tag=as6db2fc4d

Call-ID: aor911@127.0.0.1_b2b-1

CSeq: 11 BYE

X-RTP-Stat: CS=0;PS=1433;ES=1525;OS=229280;SP=0/0;SO=0;QS=-;PR=1522;ER=1525;OR=243520;CR=0;SR=0;QR=-;PL=0,0;BL=12

X-RTP-Stat-Add: DQ=31;DSS=0;DS=0;PLCS=288;JS=1

User-Agent: HEPeer

Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY

Supported: replaces

Contact: <sip:148@112.111.251.155:7070>



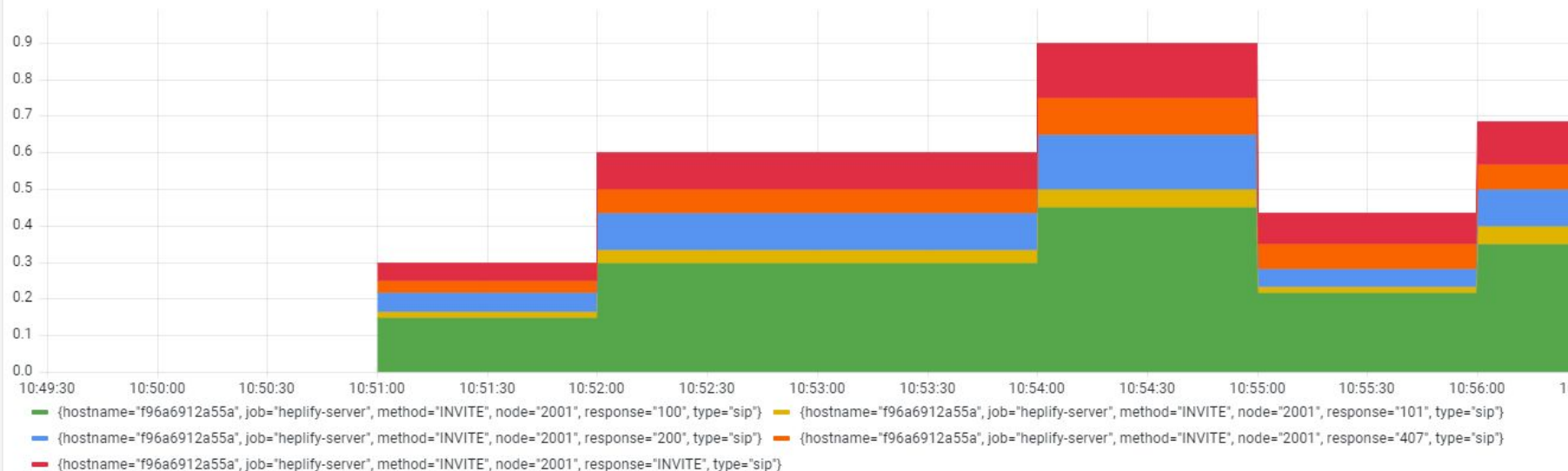
qryn: on the fly statistics from any data

Log browser >

```
rate({job="heplify-server", method="INVITE"} [1m])
```

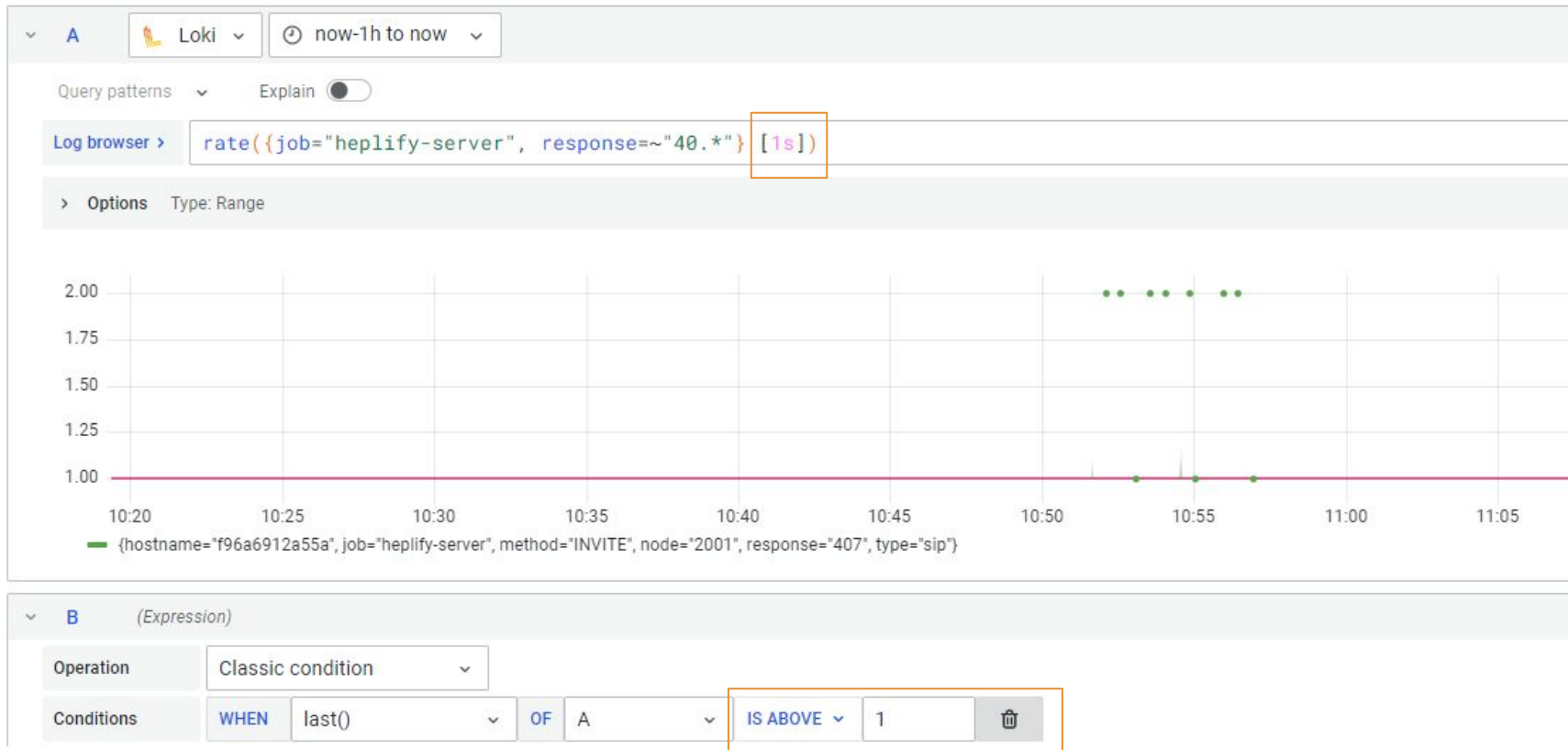
> Options Type: Range

Graph





qryn: alert on any condition, any protocol, in a few clicks





qryn: event search and filtering with Janus

[Log browser >](#)

```
{emitter="MyJanusInstance"}
```

> Options Type: Range

```
2021-10-09 03:36:44 {"emitter":"MyJanusInstance","type":32,"subtype":3,"timestamp":1633743404143327,"session_id":5821783973368110,"handle_id":136799945385545,"opaque_id":"streamingtest-xSv9I
xU4ELEZ","event":{"mid":"v2","mindex":2,"media":"video","base":90000,"rtt":0,"lost":0,"lost-by-remote":0,"jitter-local":0,"jitter-remote":0,"in-link-quality":0,"in-media-link-quality":0,"out-link-quality":100,"out-media-link-quality":100,"packets-received":0,"packets-sent":347,"bytes-received":0,"bytes-sent":376131,"bytes-received-lastsec":0,"bytes-sent-lastsec":32757,"nacks-received":0,"nacks-sent":0,"retransmissions-received":0}}
```

Log labels

emitter	MyJanusInstance
event_base	90000
event_bytes_received	0
event_bytes_received_lastsec	0
event_bytes_sent	376131
event_bytes_sent_lastsec	32757
event_in_link_quality	0
event_in_media_link_quality	0
event_jitter_local	0
event_jitter_remote	0
event_lost	0
event_lost_by_remote	0
event_media	video
event_mid	v2
event_mindex	2



qryn: on the fly statistics and extraction from Janus events

[Log browser >](#)

```
derivative({emitter="MyJanusInstance", type="32"} | json | unwrap event_bytes_sent_lastsec[10s] by (handle_id, opaque_id)
```

> Options Type: Range

2021-10-09 03:36:44 {"emitter": "MyJanusInstance", "type": "32", "subtype": "3", "timestamp": 1633743404143327, "session_id": "5821783973368110", "handle_id": "136799945385545", "opaque_id": "streamingtest-xSv9"} |

Graph





qryn: tracing applications with opentelemetry

▼ A (Tempo)

Query type Search TraceID JSON file Service Graph Loki Search

Service Name dummy-server ×

Span Name Select a span

Tags http.status_code=200 error=true

Min Duration e.g. 1.2s, 100ms

Max Duration e.g. 1.2s, 100ms

Limit

+ Add query Inspector

Table

Trace ID	Trace name	Start time	Duration
B57D816B6990C6E021...	dummy-server request_...	less than 5 seconds ago	20 ms
5EC81D066D213191B7...	dummy-server request_...	less than 5 seconds ago	24 ms
B57D816B6990C6E021...	dummy-server request_...	less than 5 seconds ago	263 ms
B57D816B6990C6E021...	dummy-server cache_l...	less than 5 seconds ago	50 ms
B57D816B6990C6E021...	dummy-server data_req...	less than 5 seconds ago	315 ms
B57D816B6990C6E021...	dummy-server request_...	less than 5 seconds ago	3 ms

▼ A (Tempo)

Query type Search TraceID JSON file Service Graph Loki Search

Trace ID B57D816B6990C6E0215FEDFB8B6FBF87

+ Add query Inspector

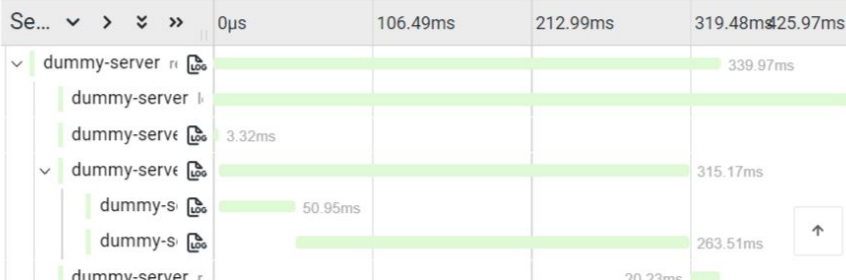
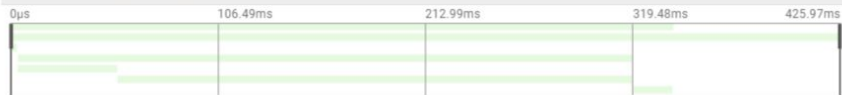
Node graph Beta

Trace View

Find...

dummy-server: request_received b57d816b6990c6e0215fedfb8b6fbf87

Trace Start: 2022-09-28 10:48:49.558 Duration: 425.97ms Services: 1 Depth: 3 Total Spans: 7





qryn: tracing databases with opentelemetry

▼ A (Tempo-Uptrace)

Query type

Search

TraceID

JSON file

Trace ID

e12ff53a-3171-973b-4818-e6bc2a6da639

+ Add query

🕒 Query history

🔍 Inspector

Trace View

Find...

serve: flush-spans e12ff53a3171973b4818e6bc2a6da639

Trace Start: 2022-05-07 16:42:08.374 Duration: 264.52ms Services: 1 Depth: 2 Total Spans: 3

0µs

66.13ms

132.26ms

198.39ms

264.52ms

Service & Operation

▼ > 🔍 ⌵ ⌶

0µs

66.13ms

132.26ms

198.39ms

264.52ms

▼ serve flush-spans | 264.52ms

flush-spans

Service: serve Duration: 264.52ms Start Time: 0µs Child Count: 2

> Tags: otel.library.name = github.com/uptrace/uptrace | otel.library.version = semver:0.31.0 | span.kind = internal | status.code = 1 | telemetry.sdk.language = go | telemetry.sdk.n...

> Process: host.name = 5865aebd049c | service.name = serve

🔗 SpanID: 2ff84687a32c6064

serve INSERT INTO 'spans_data_buffer' ('trace_id','id','pa...

224.95ms

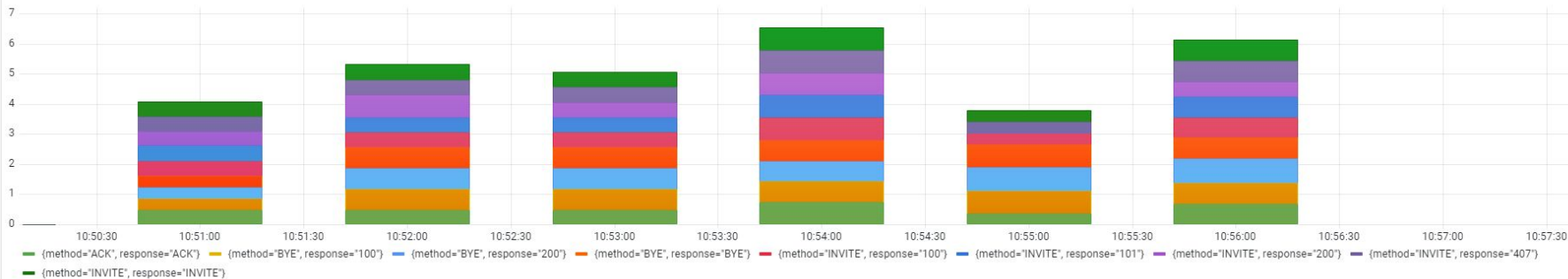
serve INSERT INTO 'spans_index_buffer' ('project_id','spa...

3.69ms

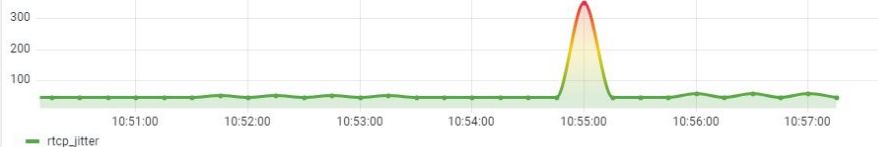


qryn: built-in prometheus/promql and logql metrics

SIP Methods & Responses



RTCP Jitter



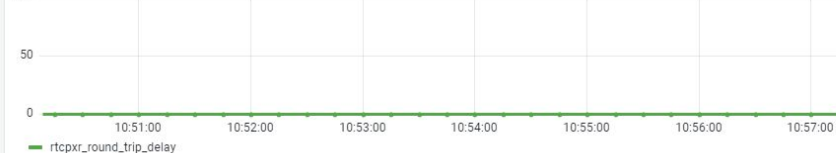
RTCP Fraction Loss



RTCP Packet Loss

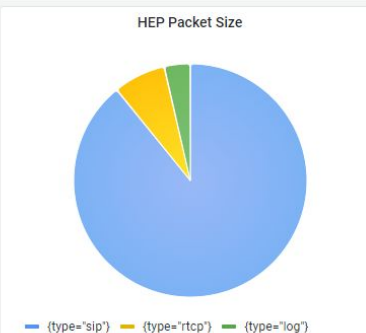
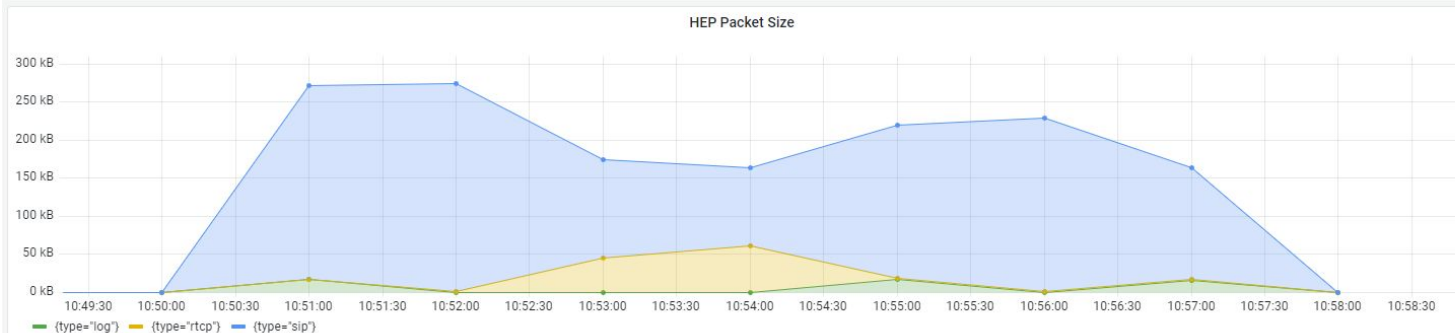
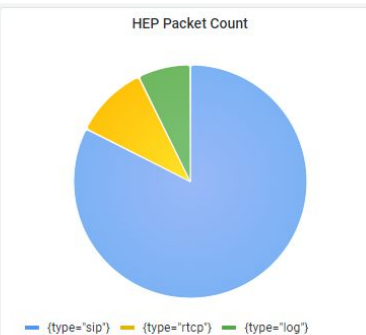
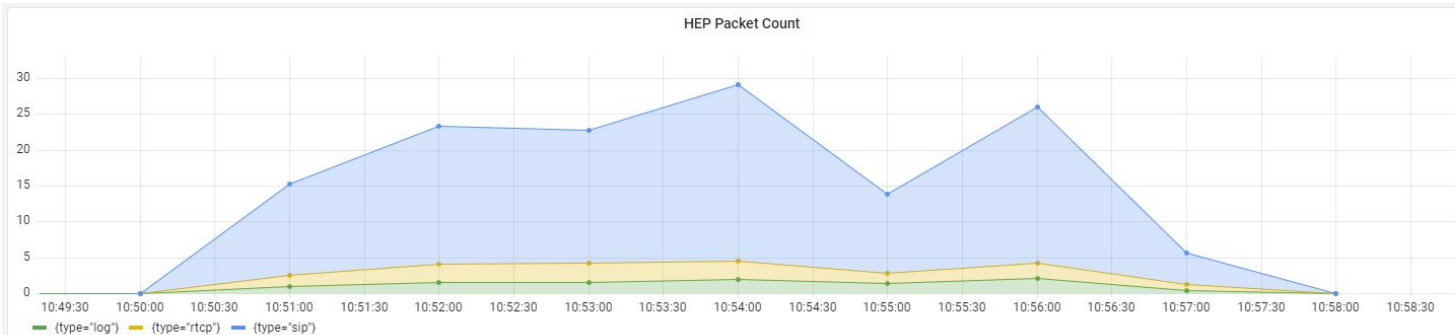


RTCP DLSR



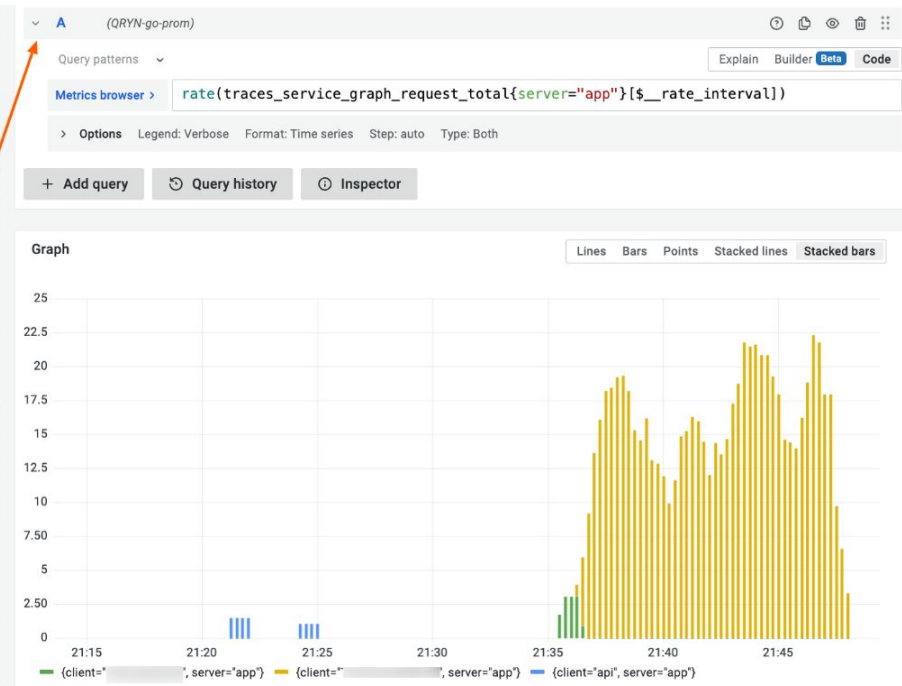
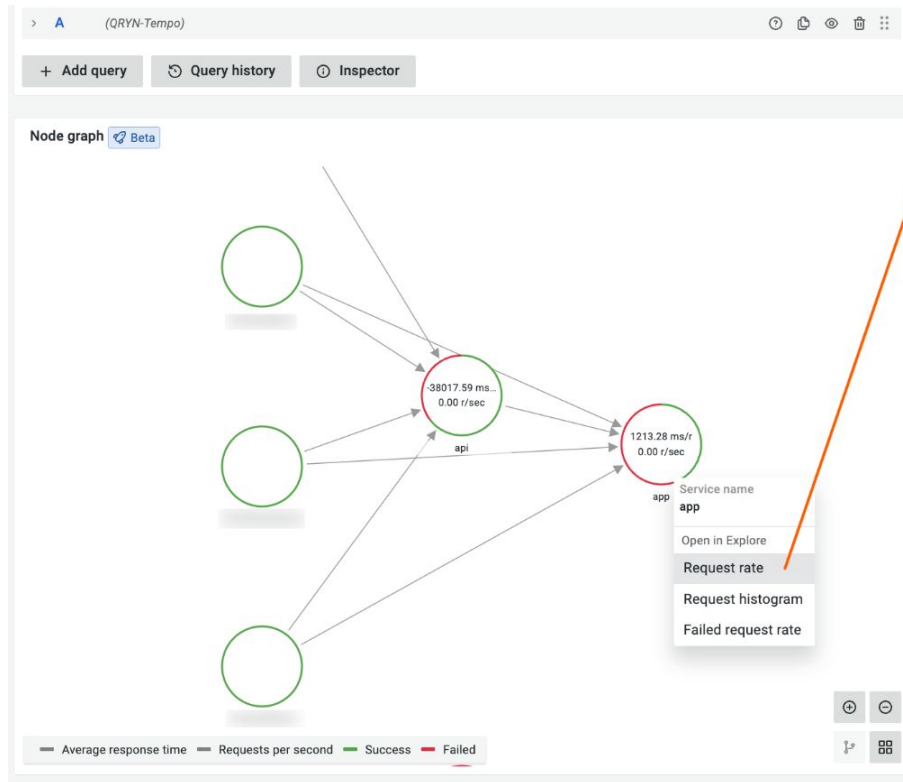


qryn: built-in prometheus/promql and logql statistics



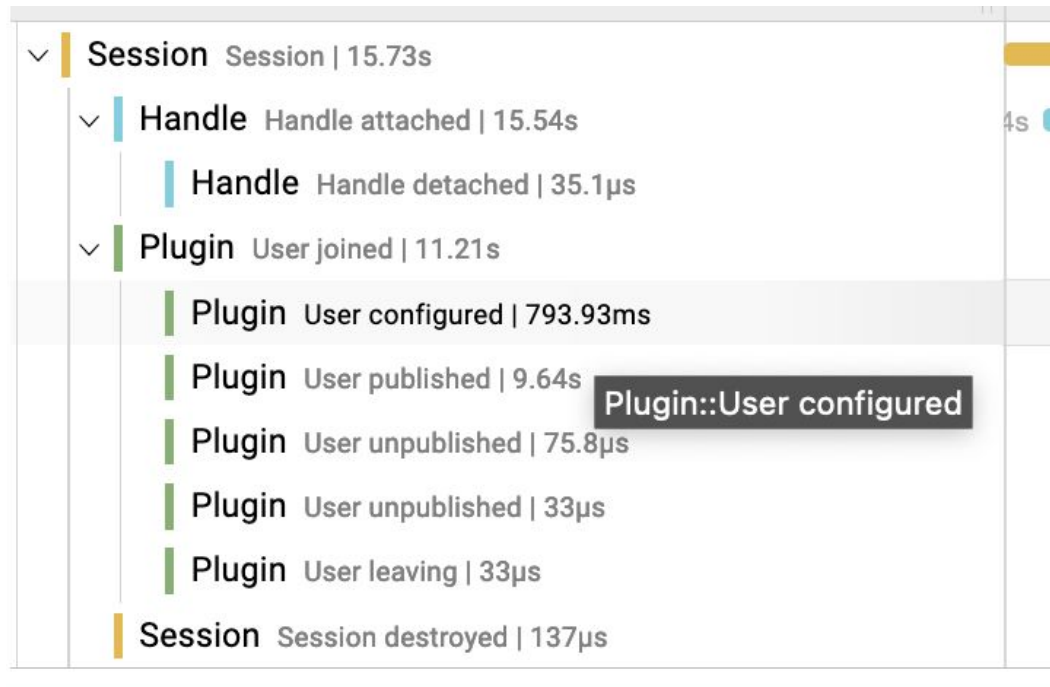


qryn: graph map of services with built-in correlation





qryn: tracing events with Janus



Session Overview

- Session Created / Destroyed
- Handle attached / detached
- **Plugin (User) joins**
- ICE (Connection State)
- Plugin (Media Reporting)
- JSEP (SDP)



qryn: tracing events with Janus and turning them into anything you want!

Service & Operation	0µs	1m	2m	3m	5m
Session Session (JanusInstance.0)	1µs	19s	37s	56s	15s
Session Session created (JanusInstar	1µs				
Handle Handle (JanusInstance.0)	1µs				
Handle Handle attached (JanusIn	1µs				

Handle attached

Service: **Handle** | Duration: **1µs** | Start Time: **545.1ms (06:58:41.330)**

[Logs for this span](#)

Attributes

_type	"Tags_Type2"	
emitter	"janusInstance.0"	
event_name	"attached"	
event_type	"2"	
handle_id	"0"	
localEndpoint.serviceName	"Handle"	
opaque_id	"videoroom_4862"	
plugin	"janus.plugin.videoroom"	
service.name	"Handle"	
session_id	"892016479776831"	
span.kind	"server"	
status.code	0	

> **Resource:** service.name = Handle



qryn: tracing events with Janus and turning them into anything you want!

Audio Media Report

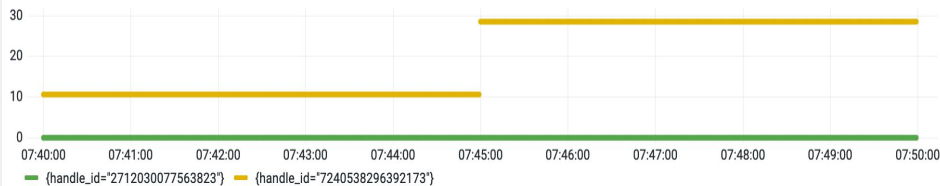
Tags

base	48000
bytes-received	137901
bytes-received-lastsec	9050
bytes-sent	0
bytes-sent-lastsec	0
in-link-quality	100
in-media-link-quality	100
jitter-local	5
jitter-remote	0
lost	0
lost-by-remote	0
media	"audio"
mid	"0"
mindex	0
nacks-received	0

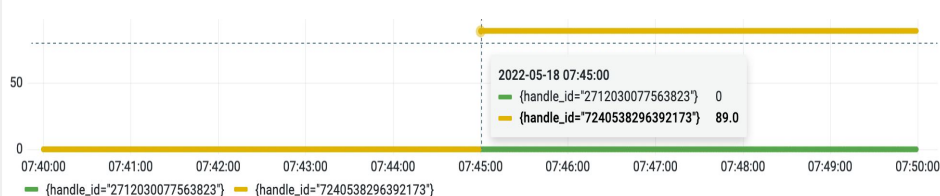
> **Process:** service.name = Media | host.name = unknown_host

Session ID 1932851385915593

Local Jitter by "handle_id"




Local Packet Loss by "handle_id" ▾





qryn: it's all documented, CI tested, and ready to use!

Search...



Home

Introduction

Installation

▼ Data Ingestion

Logs

Metrics

Telemetry

▼ Querying

Logs

Metrics

Telemetry

▼ Advanced

LogQL

APIs

[contribute](#)

Installation

Let's install **qryn** in a snap. All you need is `NodeJS 14.x-16.x` or `Docker` installed.

Configuration is performed using `ENV` parameters passed to the process or container.

NPM

PM2

GIT

Docker

Kubernetes

Bun

RPi4/aarch64



Install `qryn` as global package on your system using `npm` and `nodejs (14.x-16.x)`

```
sudo npm install -g qryn
```

bash

Start `qryn` using `ENV` variables for its settings:



qryn: polyglot monitoring -> completed!

... and that's how we spent the last year, folks!

- **HEP** API 100% backwards compatible
- **LogQL** API ingest and query any log
- **Prometheus** API ingest and query metrics
- **Tempo** API ingest and query telemetry traces





qryn: polyglot monitoring -> completed!

F.A.Q:

- Does QRYN kill HOMER?** No! It's our new backend
- Does it work with HEP?** Yes, all HEP3 clients are OK
- Does it need the HOMER UI?** Optional. Grafana+View
- Is it OSS?** Yes AGPLv3 + Commercial for integrators
- Is it out? Can I install it?** Yes, go to <https://qryn.dev>
- Does it Docker?** Yes. New compose files are ready
- Works with any log/event?** Yes. Any log, any format.
- Can I integrate in my project?** Yes. Core, Edge or both
- Does it take CI?** Yes, includes a github action agent
- Is there a Cloud version?** Yes. Launching by Q422



Got more questions? *Let's Q&A*



qryn: are you ready to make the leap?

Interested in trying out, helping or participating? We're looking for Users, Contributors, Testers and Contesters!
Our community is non exclusive welcomes experts just as much as newcomers, students and general rtc hackers.

homer + hep
qryn/metrico
other tools

<https://sipcapture.org>

<https://qryn.dev>

<https://metrico.in>

Interested in our services and experience to "hep up" your services or projects? Give us a call anytime!

qxip team
qxip projects

info@qxip.net

sales@qxip.net



SUPPORT **OPEN-SOURCE**

BEFORE YOU ASK **OPEN-SOURCE** TO SUPPORT YOU

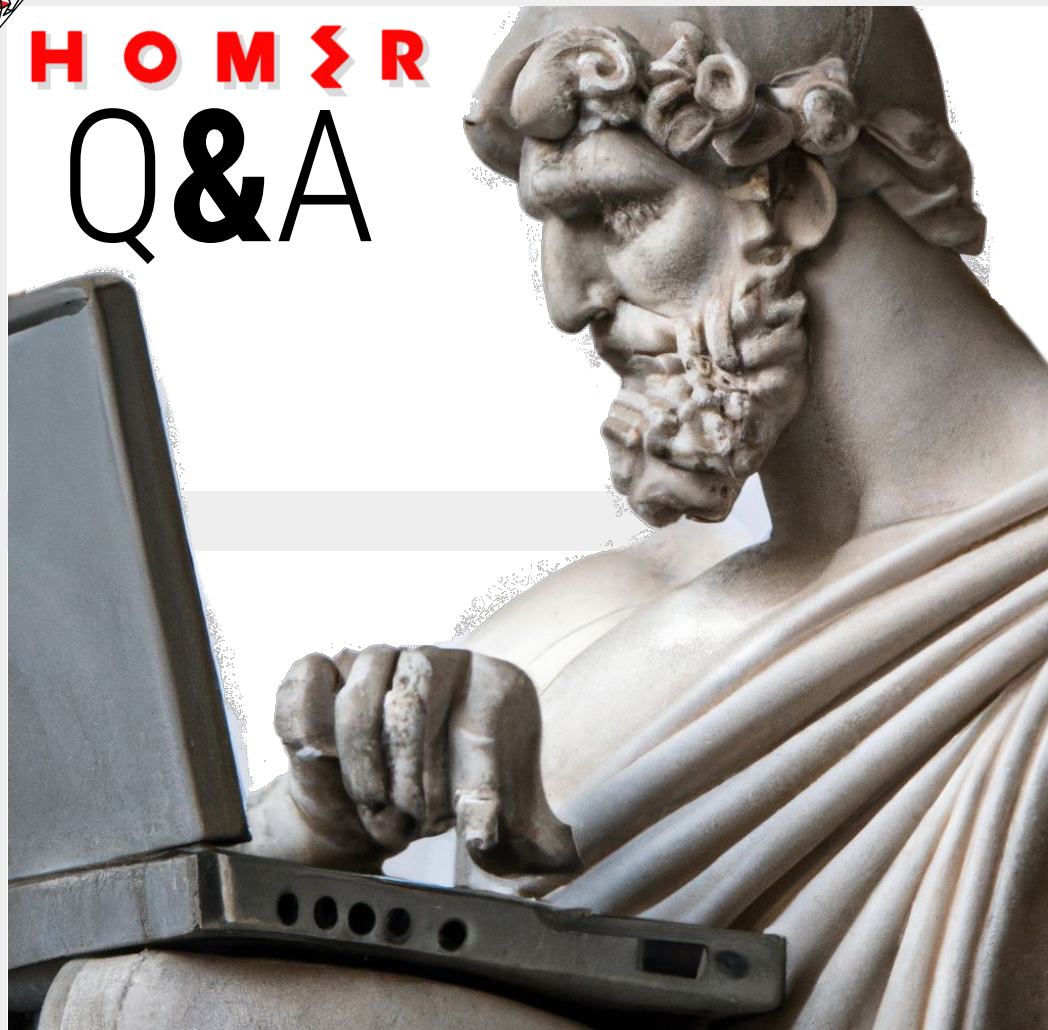


Give respect **Get** respect

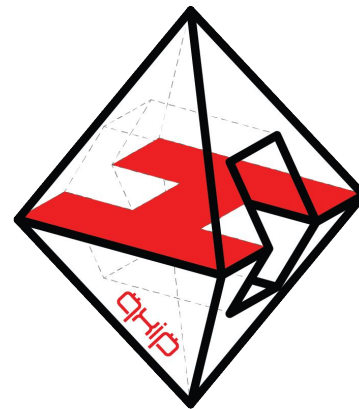




HOMER Q&A



QXIP



info@qxip.net

Next presentation: @Cluecon Oct, 2022

QXIP

