SIPCAPTURE

KAMAILIO WORLD
CONFERENCE & EXHIBITION

KAMAILIO WORLD
CONFERENCE & EXHIBITION

Hello There!

Alexandr Dubovikov

Sr. Voice Architect at **QSC AG**, one of the major German voice and data providers

Co-Founder at **QXIP** a recognized and innovative Research & Development company specialized in open source and commercial passive packet capture and realtime monitoring solutions

Our flagships include HEP, HOMER, HEPIC, SENTINL, PASTASH and many more tiny tools

QXIP and **HOMER** are 100% open source and powered by actual **HUMANS**

Alexandr, Lorenzo, Celeste, Eugen, Federico, Giacomo, Michele, Sergey, Dario, Gaetano, Joseph

Project Goals

HOMER 7 is a the new major release of our VoIP and RTC Troubleshooting platform and our first step in a new direction reflecting modern architecture requirements.

<u>*Release Highlights:*</u>

- New Capture Servers & Agents
  - Independent, ready to run, portable *(thanks **Negbie** for your massive contribs!)*
- Easy to extend with new searchable protocols
  - Indexing IP Protocols, RTC Events, CDRs, JSON Objects and more!
- Full Scale Indexing and Timeseries storage
  - Data and Timeseries are now split to maximize utilization patterns
- Integration with non-HEP platforms
  - Data can be received using UDP/TCP, HTTP, Protobuf, Queues
- Stable and Documented Backend API
  - Completely redesigned API developed in NodeJS (no more PHP/Apache)
- New UI and Improved user experience
  - Completely redesigned Angular UI with modular and extensible elements

Major Changes

→ Capture Servers

HEPlify-Server	developed in **GO** for *high-performance and net protocols*
HEPop	developed in **NodeJS** for *high-flexibility and event streams*

→ Web Services

HOMER-UI	new framework inherited from the **HEPIC** platform
HOMER-API	developed in **NodeJS**, easy to extend and self-serving

→ Database

Postgres *or* **MySQL**	leveraging native **JSON/JSONB** indexing and search
InfluxDB *or* **Prometheus**	leveraging native Aggregation and **Alerting** features

NEW

New Components

heplify-server

heplify

hepop

HEPlify-server is a stand-alone **HOMER** *Capture Server* developed in **Go**, optimized for speed and simplicity. Distributed as a single binary ready to capture TLS and UDP HEP encapsulated packets from any HEP agent.

HEPlify is captagents little brother, optimized for speed and simplicity. It's a single binary which you can run on Linux, ARM, MIPS, Windows to capture IPv4 or IPv6 packets and send SIP, correlated RTCP, RTCP-XR, DNS, Logs into HOMER, handling fragmented and duplicate packets out of the box.

HEPop is a stand-alone **HOMER** *Capture Server* developed in **NodeJS**, optimized for streams, flexibility and fast prototyping. Distributed via **npm**, it ships ready to capture TLS and UDP HEP encapsulated packets and events from **Janus**, **Mediasoup, Kamailio, OpenSIPS** and other RTC Gateways

Native JSON & Timeseries

NATIVE JSON:

The next-generation Capture Servers are designed to leverage the native **JSON Indexing** and **Search** functionality provided by **Postgres**, **Mysql**, **MongoDB** and already offers experimental insert support for RethinkDB, Elasticsearch and other backends, ready with solid Bulk processors to maximize resource usage & performance

NATIVE CORRELATION:

The latest database schema design in HOMER Seven allows developers and integrators to easily define and map new searchable data types with native support for multiple correlation rules defining "virtual join" vectors between HEP Types, Events, Reports and Logs.

NATIVE TIMESERIES:

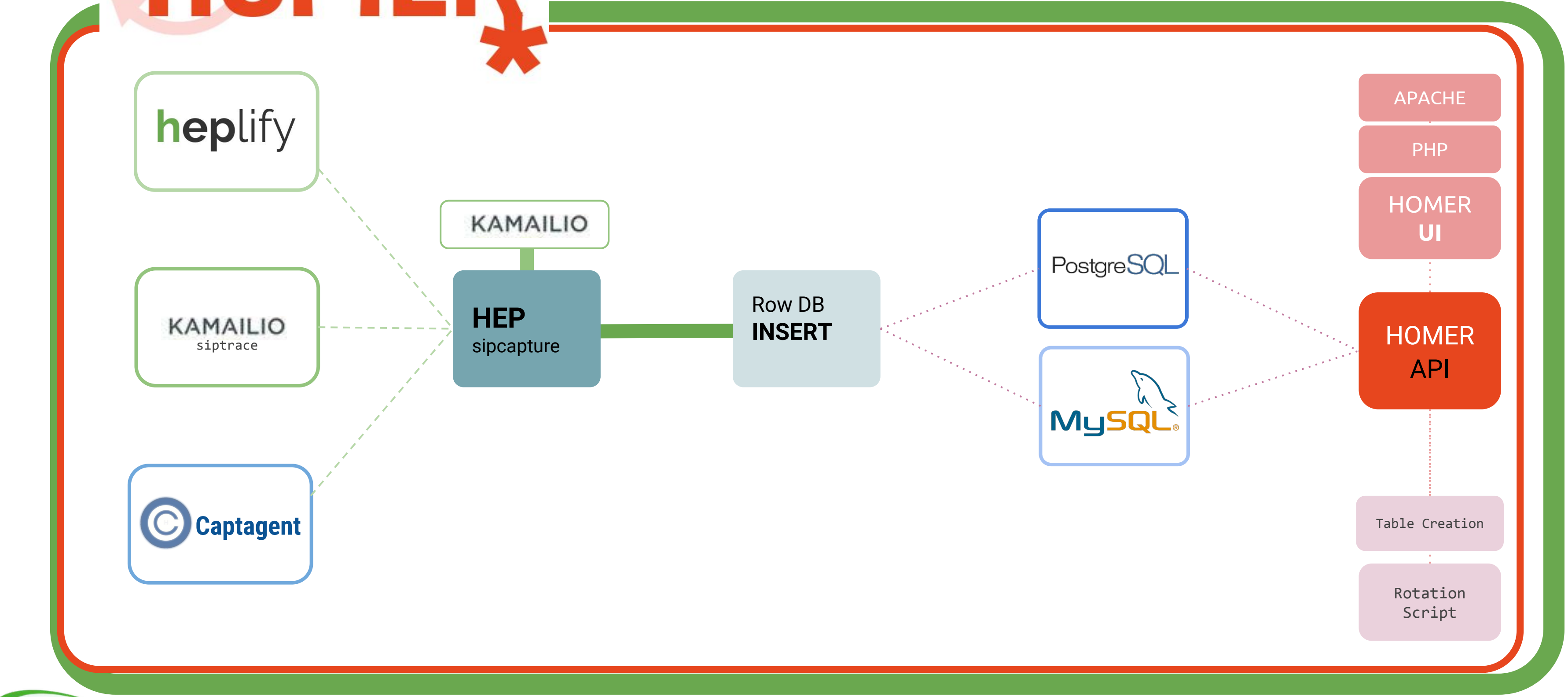
The next-generation Capture Servers are designed to convert specific events into tagged timeseries natively shipped to **InfluxDB**, **Prometheus** or **Elasticsearch**

The new User-Interface can directly fetch data from the connected timeseries backends of choice, providing basic visualization including any data generated by 3rd parties

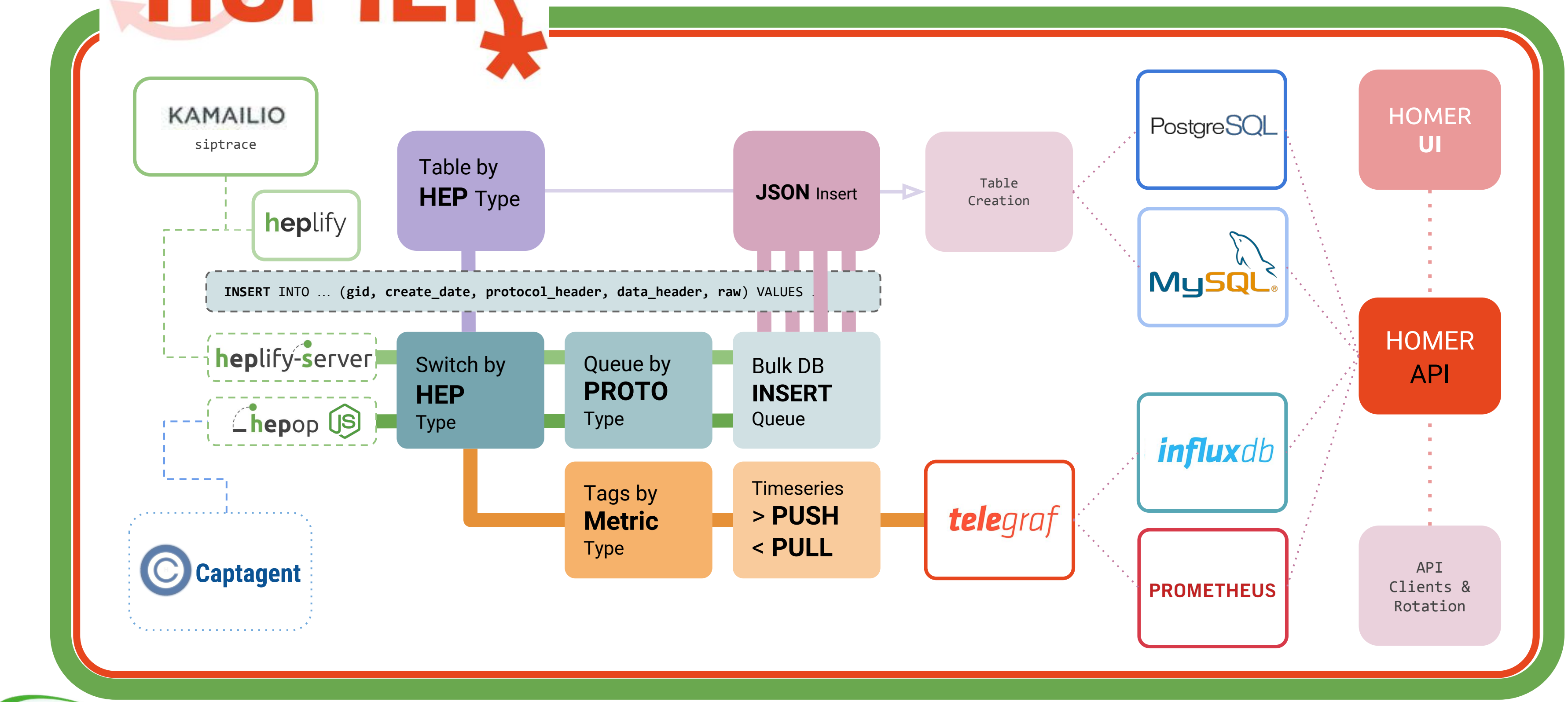
NATIVE ALERTING:

Native integrations allow users to unleash the full power of the Alerting and Reporting capabilities provided by either **Kapacitor** for **InfluxDB**, **Alertmanager** for **Prometheus** and our **SENTINL** for **Elasticsearch** and more in the future

Before: Homer 5.x

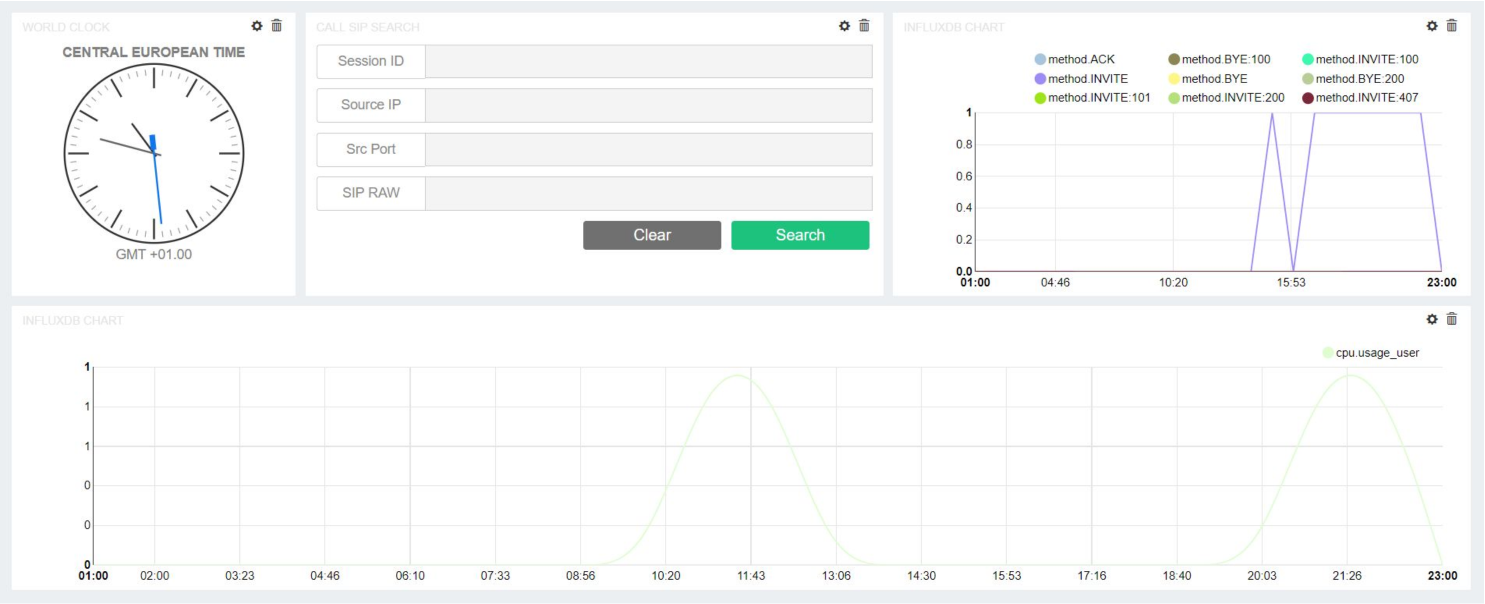


After: Homer 7.x



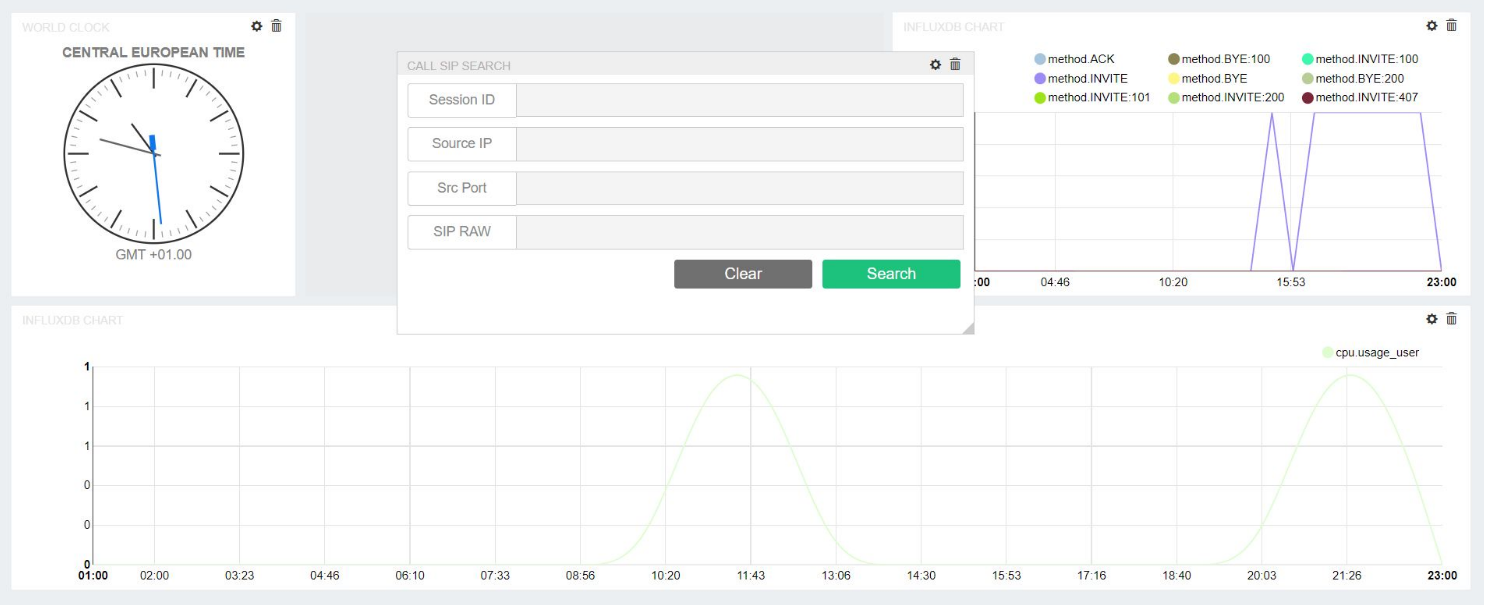
USER INTERFACE

User-Interface

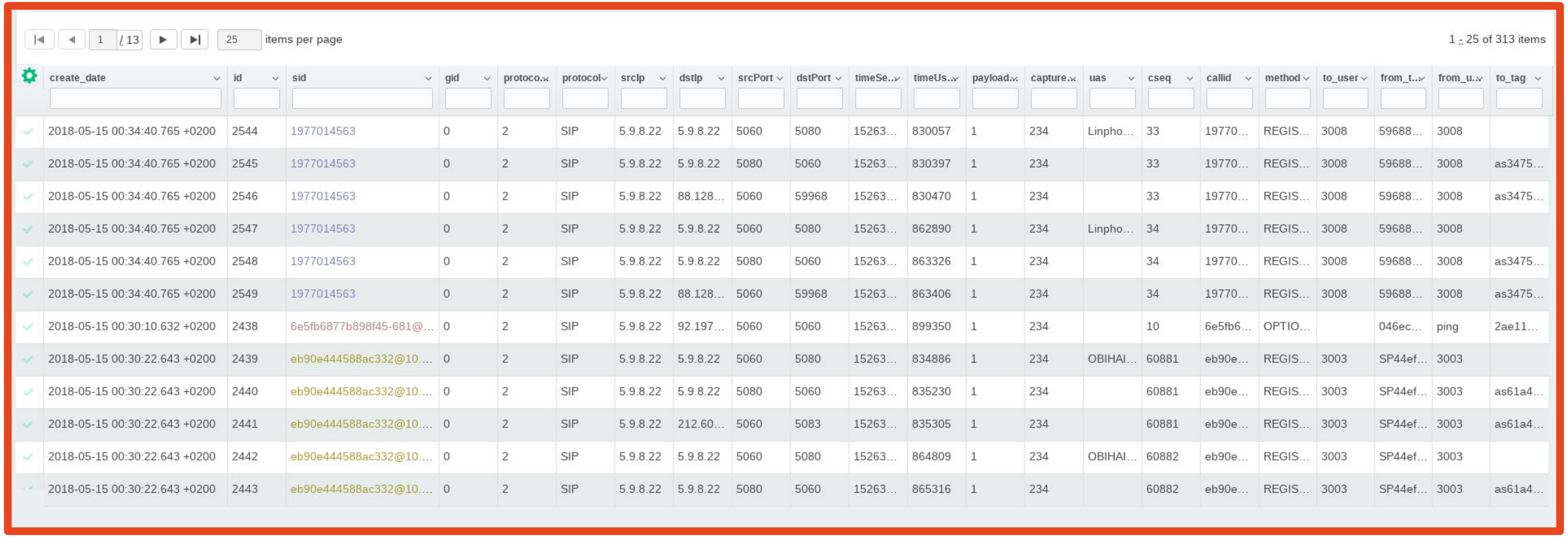


HOMER

User-Interface



User-Interface

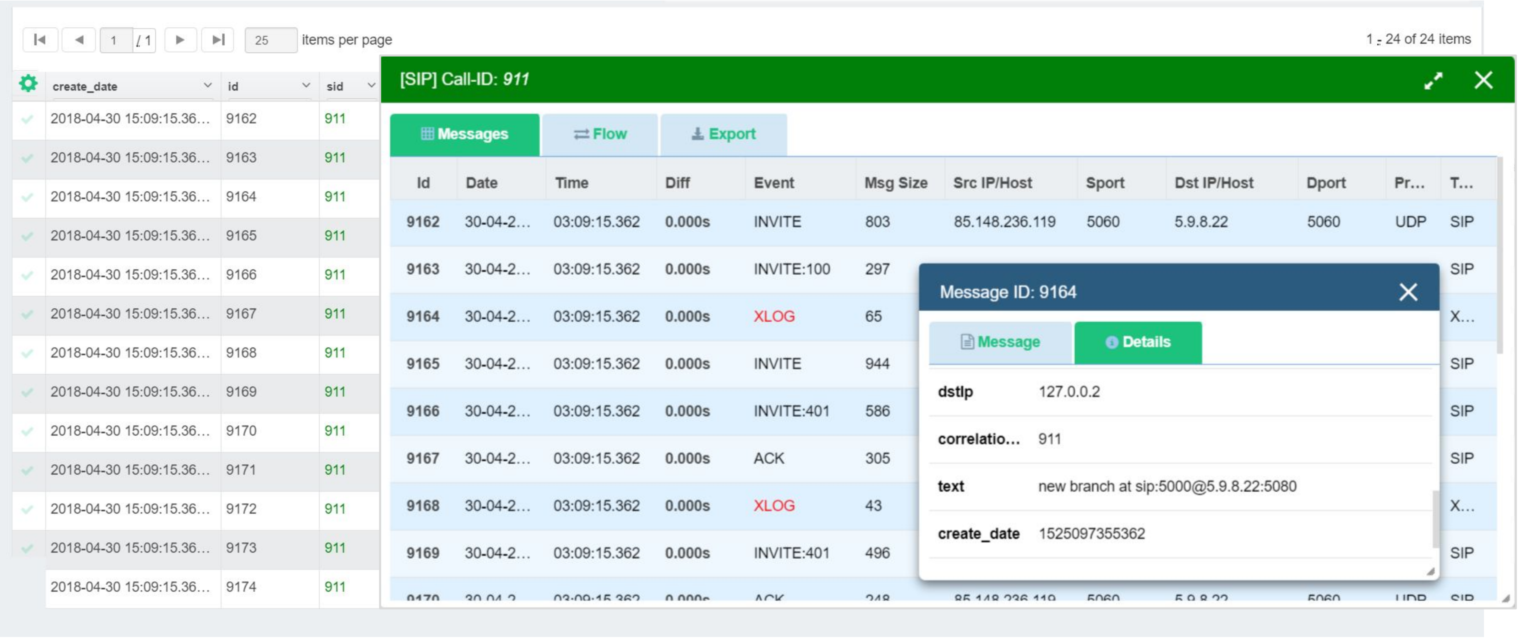


|◀ ◀ 1 / 13 ▶ ▶| 25 items per page

1 - 25 of 313 items

	create_date	id	sid	gid	protoco…	protocol	srcIp	dstIp	srcPort	dstPort	timeSe…	timeUs…	payload…	capture…	uas	cseq	callid	method	to_user	from_t…	from_u…	to_tag
✓	2018-05-15 00:34:40.765 +0200	2544	1977014563	0	2	SIP	5.9.8.22	5.9.8.22	5060	5080	15263…	830057	1	234	Linpho…	33	19770…	REGIS…	3008	59688…	3008	
✓	2018-05-15 00:34:40.765 +0200	2545	1977014563	0	2	SIP	5.9.8.22	5.9.8.22	5080	5060	15263…	830397	1	234		33	19770…	REGIS…	3008	59688…	3008	as3475…
✓	2018-05-15 00:34:40.765 +0200	2546	1977014563	0	2	SIP	5.9.8.22	88.128…	5060	59968	15263…	830470	1	234		33	19770…	REGIS…	3008	59688…	3008	as3475…
✓	2018-05-15 00:34:40.765 +0200	2547	1977014563	0	2	SIP	5.9.8.22	5.9.8.22	5060	5080	15263…	862890	1	234	Linpho…	34	19770…	REGIS…	3008	59688…	3008	
✓	2018-05-15 00:34:40.765 +0200	2548	1977014563	0	2	SIP	5.9.8.22	5.9.8.22	5080	5060	15263…	863326	1	234		34	19770…	REGIS…	3008	59688…	3008	as3475…
✓	2018-05-15 00:34:40.765 +0200	2549	1977014563	0	2	SIP	5.9.8.22	88.128…	5060	59968	15263…	863406	1	234		34	19770…	REGIS…	3008	59688…	3008	as3475…
✓	2018-05-15 00:30:10.632 +0200	2438	6e5fb6877b898f45-681@…	0	2	SIP	5.9.8.22	92.197…	5060	5060	15263…	899350	1	234		10	6e5fb6…	OPTIO…		046ec…	ping	2ae11…
✓	2018-05-15 00:30:22.643 +0200	2439	eb90e444588ac332@10.…	0	2	SIP	5.9.8.22	5.9.8.22	5060	5080	15263…	834886	1	234	OBIHAI…	60881	eb90e…	REGIS…	3003	SP44ef…	3003	
✓	2018-05-15 00:30:22.643 +0200	2440	eb90e444588ac332@10.…	0	2	SIP	5.9.8.22	5.9.8.22	5080	5060	15263…	835230	1	234		60881	eb90e…	REGIS…	3003	SP44ef…	3003	as61a4…
✓	2018-05-15 00:30:22.643 +0200	2441	eb90e444588ac332@10.…	0	2	SIP	5.9.8.22	212.60…	5060	5083	15263…	835305	1	234		60881	eb90e…	REGIS…	3003	SP44ef…	3003	as61a4…
✓	2018-05-15 00:30:22.643 +0200	2442	eb90e444588ac332@10.…	0	2	SIP	5.9.8.22	5.9.8.22	5060	5080	15263…	864809	1	234	OBIHAI…	60882	eb90e…	REGIS…	3003	SP44ef…	3003	
✓	2018-05-15 00:30:22.643 +0200	2443	eb90e444588ac332@10.…	0	2	SIP	5.9.8.22	5.9.8.22	5080	5060	15263…	865316	1	234		60882	eb90e…	REGIS…	3003	SP44ef…	3003	as61a4…

User-Interface



1 / 1 | 25 items per page | 1 - 24 of 24 items

create_date	id	sid
2018-04-30 15:09:15.36…	9162	911
2018-04-30 15:09:15.36…	9163	911
2018-04-30 15:09:15.36…	9164	911
2018-04-30 15:09:15.36…	9165	911
2018-04-30 15:09:15.36…	9166	911
2018-04-30 15:09:15.36…	9167	911
2018-04-30 15:09:15.36…	9168	911
2018-04-30 15:09:15.36…	9169	911
2018-04-30 15:09:15.36…	9170	911
2018-04-30 15:09:15.36…	9171	911
2018-04-30 15:09:15.36…	9172	911
2018-04-30 15:09:15.36…	9173	911
2018-04-30 15:09:15.36…	9174	911

[SIP] Call-ID: *911*

Id	Date	Time	Diff	Event	Msg Size	Src IP/Host	Sport	Dst IP/Host	Dport	Pr…	T…
9162	30-04-2…	03:09:15.362	0.000s	INVITE	803	85.148.236.119	5060	5.9.8.22	5060	UDP	SIP
9163	30-04-2…	03:09:15.362	0.000s	INVITE:100	297						SIP
9164	30-04-2…	03:09:15.362	0.000s	XLOG	65						X…
9165	30-04-2…	03:09:15.362	0.000s	INVITE	944						SIP
9166	30-04-2…	03:09:15.362	0.000s	INVITE:401	586						SIP
9167	30-04-2…	03:09:15.362	0.000s	ACK	305						SIP
9168	30-04-2…	03:09:15.362	0.000s	XLOG	43						X…
9169	30-04-2…	03:09:15.362	0.000s	INVITE:401	496						SIP
9170	30-04-2…	03:09:15.362	0.000s	ACK	248	85.148.236.119	5060	5.9.8.22	5060	UDP	SIP

Message ID: 9164

dstIp	127.0.0.2
correlatio…	911
text	new branch at sip:5000@5.9.8.22:5080
create_date	1525097355362

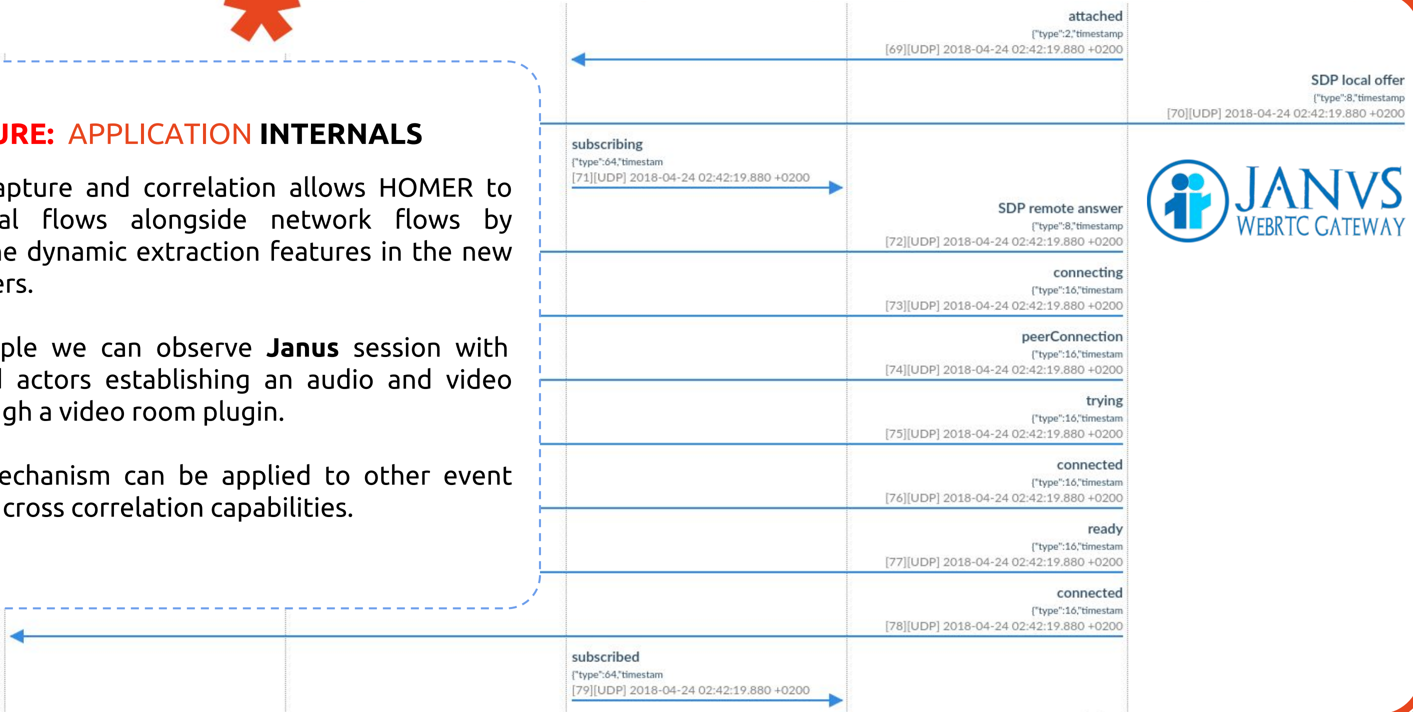
User-Interface

NEW FEATURE: APPLICATION **INTERNALS**

The event capture and correlation allows HOMER to track internal flows alongside network flows by leveraging the dynamic extraction features in the new capture servers.

In this example we can observe **Janus** session with handlers and actors establishing an audio and video session through a video room plugin.

The same mechanism can be applied to other event streams with cross correlation capabilities.



attached
{"type":2,"timestamp
[69][UDP] 2018-04-24 02:42:19.880 +0200

SDP local offer
{"type":8,"timestamp
[70][UDP] 2018-04-24 02:42:19.880 +0200

subscribing
{"type":64,"timestam
[71][UDP] 2018-04-24 02:42:19.880 +0200

SDP remote answer
{"type":8,"timestamp
[72][UDP] 2018-04-24 02:42:19.880 +0200

connecting
{"type":16,"timestam
[73][UDP] 2018-04-24 02:42:19.880 +0200

peerConnection
{"type":16,"timestam
[74][UDP] 2018-04-24 02:42:19.880 +0200

trying
{"type":16,"timestam
[75][UDP] 2018-04-24 02:42:19.880 +0200

connected
{"type":16,"timestam
[76][UDP] 2018-04-24 02:42:19.880 +0200

ready
{"type":16,"timestam
[77][UDP] 2018-04-24 02:42:19.880 +0200

connected
{"type":16,"timestam
[78][UDP] 2018-04-24 02:42:19.880 +0200

subscribed
{"type":64,"timestam
[79][UDP] 2018-04-24 02:42:19.880 +0200

Roadmap & Notes

The core components of **HOMER 7.x** are already available on Github for **beta** testers and will keep on expanding and growing steadily over the next months. ***Join us to help*** *test, debug & release faster!*

- Project
  - API Documentation
  - UI Framework Documentation
  - Installers and Containers
- Capture Servers
  - Additional database support
  - Protobuf and Queuing
- User-Interface
  - Media Charts & Reporting Tabs
  - Preference Panels
  - Graph Visualization

HOW TO

ADD A NEW PROTOCOL

Adding a Protocol 1/3

Table names are composed using the **HEP ID** *(proto type)* and **PROFILE ID** *(default)* used to distribute and shard packets in the database by transaction types. *IE: call, registration, default*

```
hep_proto_{id}_{type}
```

```
CREATE TABLE hep_proto_1000_default (
    id bigint NOT NULL,
    sid character varying(256),
    create_date timestamp with time zone DEFAULT CURRENT_TIMESTAMP NOT NULL,
    protocol_header jsonb NOT NULL,
    data_header jsonb NOT NULL,
    raw character varying(5000) NOT NULL
) PARTITION RANGE(create_date);
```

Adding a Protocol 2/3

```
     Column          |           Type           | Nullable |                Default
---------------------+--------------------------+-----------+----------+-------------------------------------
 id                  | integer                  | not null | nextval('mapping_schema_id_seq'::regclass)
 guid                | uuid                     |          |
 profile             | character varying(100)   | not null | 'default'::character varying
 hepid               | integer                  | not null |
 hep_alias           | character varying(100)   |          |
 version             | integer                  | not null |
 retention           | integer                  | not null | 10
 partition_step      | integer                  | not null | 3600
 create_index        | json                     |          |
 create_table        | text                     |          |
 correlation_mapping | json                     |          |
 fields_mapping      | json                     |          |
 mapping_settings    | json                     |          |
 schema_mapping      | json                     |          |
 schema_settings     | json                     |          |
 create_date         | timestamp with time zone | not null | CURRENT_TIMESTAMP
```

Adding a Protocol 3/3

- Allocate **ID** and **TYPE** for your new protocol or event type
- Create Protocol Mapping
  - Create Protocol Mapping with Retention policy 10 Days @3600 minutes

```
INSERT into mapping_settings(id,guid,profile,hepid,hep_alias,version,retention,partition_step)
VALUES(1,UUID,’calls’,1086,’Kamailio CDRs’,1, 10, 3600)
```

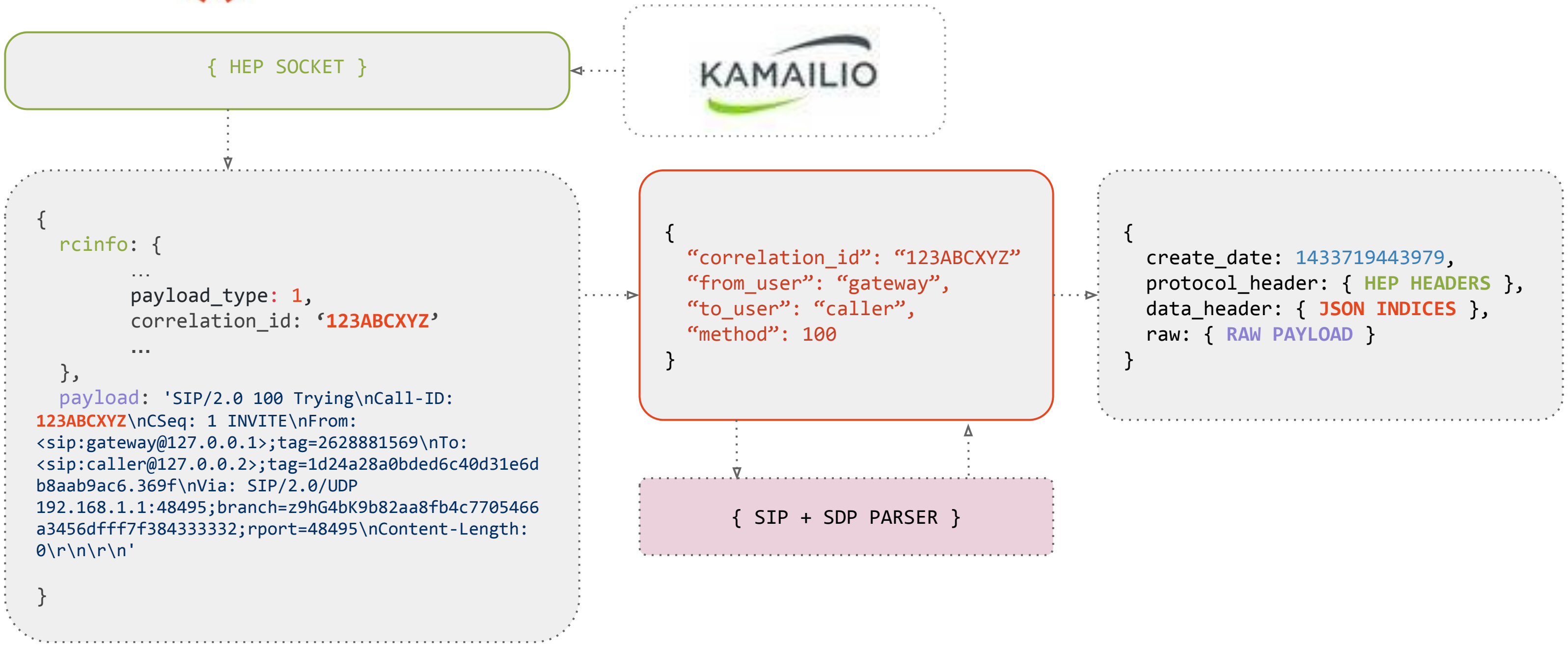
  - Derived table name hep_proto_1087_calls will be created automatically by the Capture Server

    The Table schema is same for all HEP protocols using JSON/JSONB types
  - Add correlation policy to correlation_mapping column of the Protocol Mapping:

```
[{ "source_field": "data_header.callid",
   "lookup_id": 1,
   "lookup_profile": "call",
   "lookup_field": "sid",
   "lookup_range":[ -300, 200 ]
}]
SELECT * FROM hep_proto_${lookup_id}_${lookup_profile} WHERE ${lookup_field}= ‘${source_field}’
```

- Start sending JSON CDRs over HEP type 1087

Sip Indexing



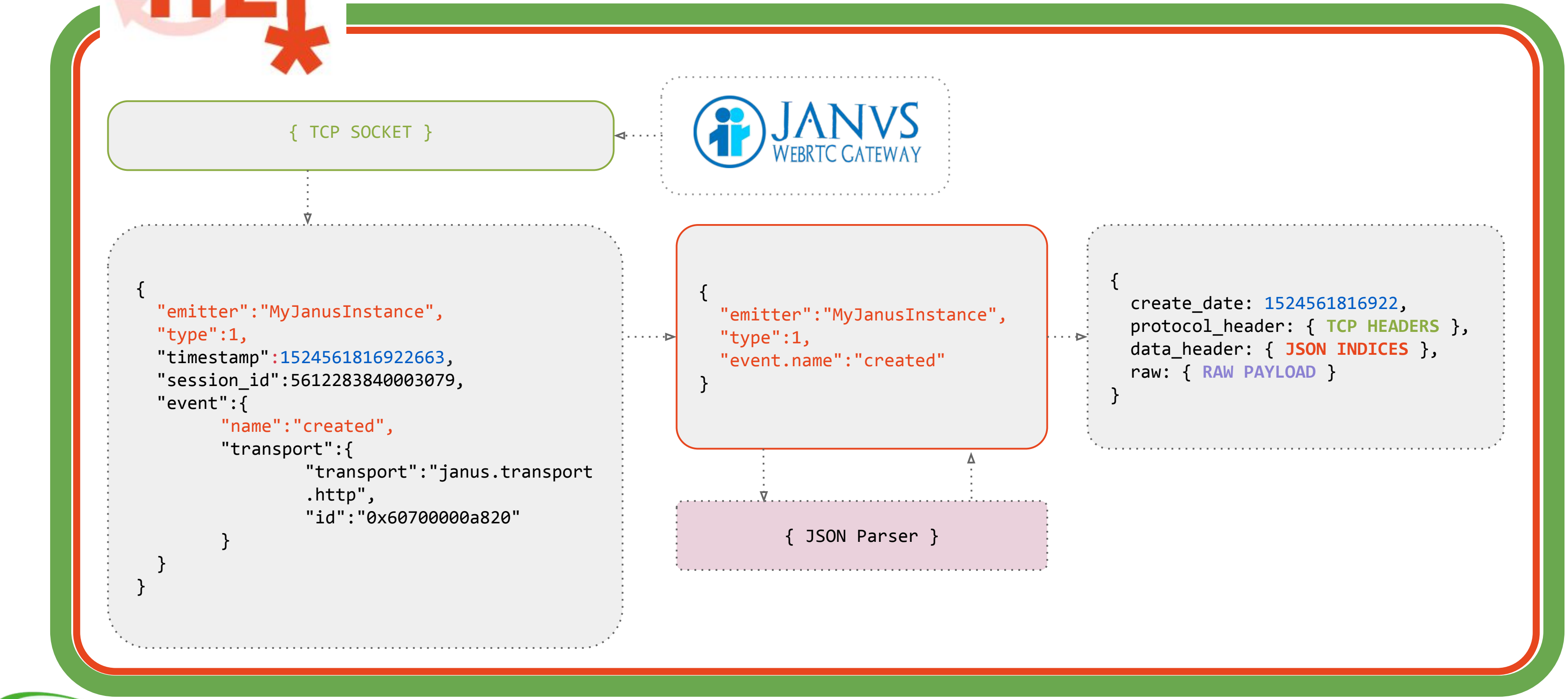
```
{
  rcinfo: {
         …
         payload_type: 1,
         correlation_id: ‘123ABCXYZ’
         …
  },
  payload: 'SIP/2.0 100 Trying\nCall-ID:
123ABCXYZ\nCSeq: 1 INVITE\nFrom:
<sip:gateway@127.0.0.1>;tag=2628881569\nTo:
<sip:caller@127.0.0.2>;tag=1d24a28a0bded6c40d31e6d
b8aab9ac6.369f\nVia: SIP/2.0/UDP
192.168.1.1:48495;branch=z9hG4bK9b82aa8fb4c7705466
a3456dfff7f384333332;rport=48495\nContent-Length:
0\r\n\r\n'

}
```

```
{
  “correlation_id”: “123ABCXYZ”
  “from_user”: “gateway”,
  “to_user”: “caller”,
  “method”: 100
}
```

```
{
  create_date: 1433719443979,
  protocol_header: { HEP HEADERS },
  data_header: { JSON INDICES },
  raw: { RAW PAYLOAD }
}
```

Event Indexing

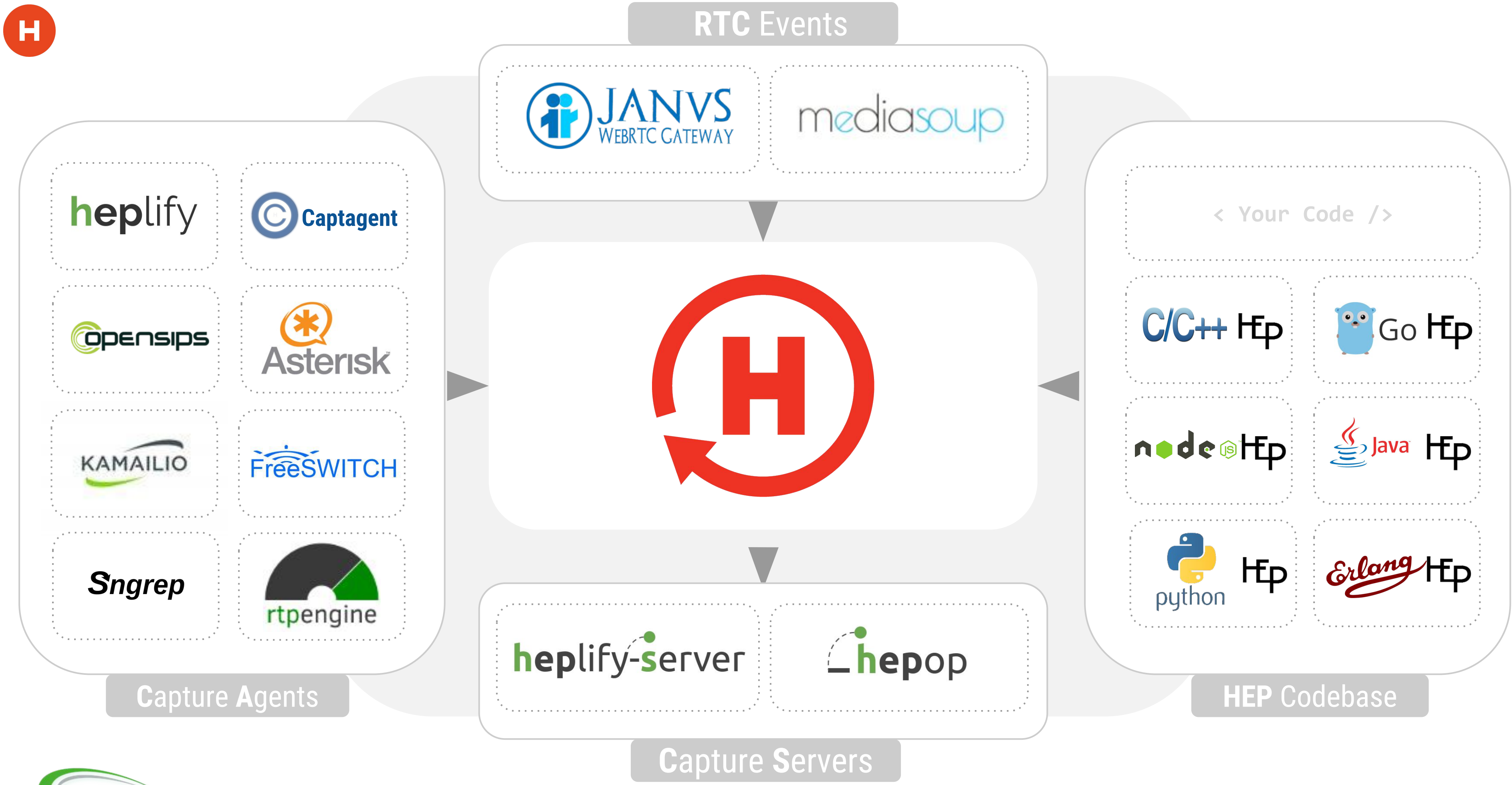


```
{
  "emitter":"MyJanusInstance",
  "type":1,
  "timestamp":1524561816922663,
  "session_id":5612283840003079,
  "event":{
        "name":"created",
        "transport":{
                "transport":"janus.transport
                .http",
                "id":"0x60700000a820"
        }
  }
}
```

```
{
  "emitter":"MyJanusInstance",
  "type":1,
  "event.name":"created"
}
```

```
{
  create_date: 1524561816922,
  protocol_header: { TCP HEADERS },
  data_header: { JSON INDICES },
  raw: { RAW PAYLOAD }
}
```

Got Questions?



RTC Events

Capture Agents

Capture Servers

HEP Codebase

< Your Code />