SIPCAPTURE WORK**SHOP**

Written by: Lorenzo Mangani, Alexandr Dubovikov

Contributors: Joseph Jackson, Doug Smith

About **QXIP** and **SIPCAPTURE**

QXIP BV *{QuickSIP}* is an Amsterdam based R&D Company specializing in Open-Source and Commercial Voice Technology Development - Our flagship projects are SIPCAPTURE **HOMER** and **PCAPTURE** based on our mature and open encapsulation protocol **HEP/EEP** *(Extensible Encapsulation Protocol)*

Our Open-Source solutions are deployed and trusted by thousands of businesses worldwide.

Our Customers include large telephony and network operators, voice service carriers, voip service providers, cloud service providers, call center operators and voice equipment vendors.

Our Capture Technologies are natively implemented in all major OSS voip platforms such as *Kamailio, OpenSIPS, FreeSWITCH, Asterisk, RTPEngine* and many tools such as *sipgrep*, *sngrep* and more.

For full details abour our projects and services please visit our website at http://qxip.net

{ SIPCAPTURE }

#STACK

HOMER

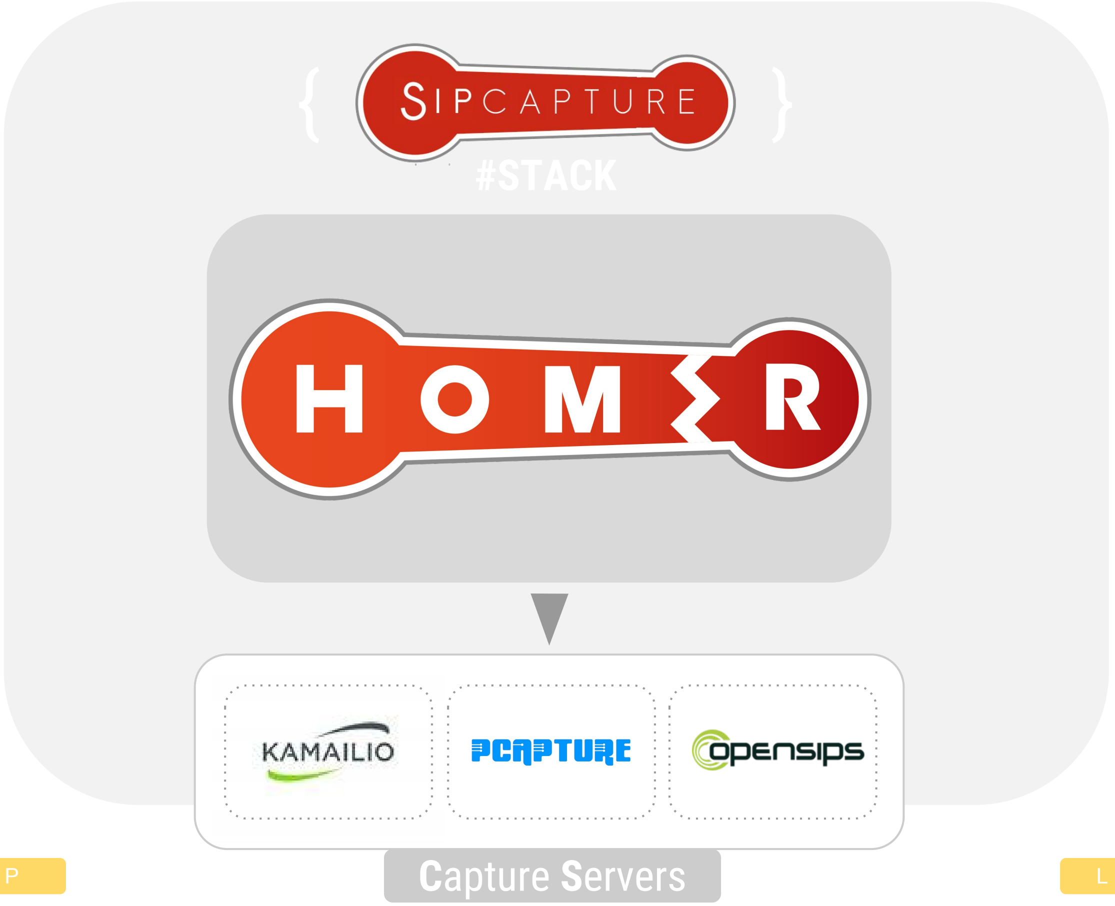
SIP SDP

HEP EEP JSON

LOGS CDR

RTP RTCP QOS

SIPCAPTURE

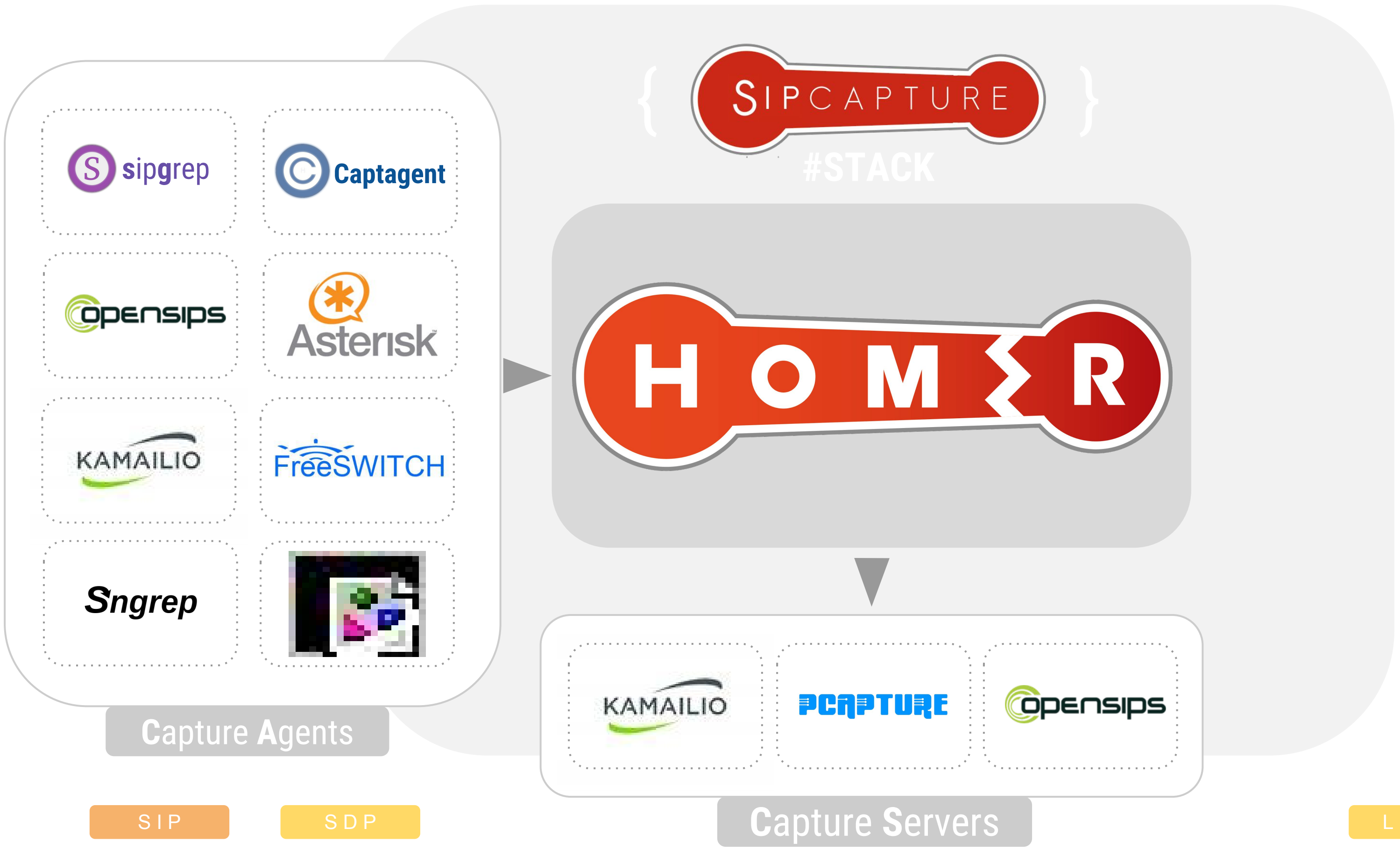


SIP SDP

HEP EEP JSON

LOGS CDR

RTP RTCP QOS

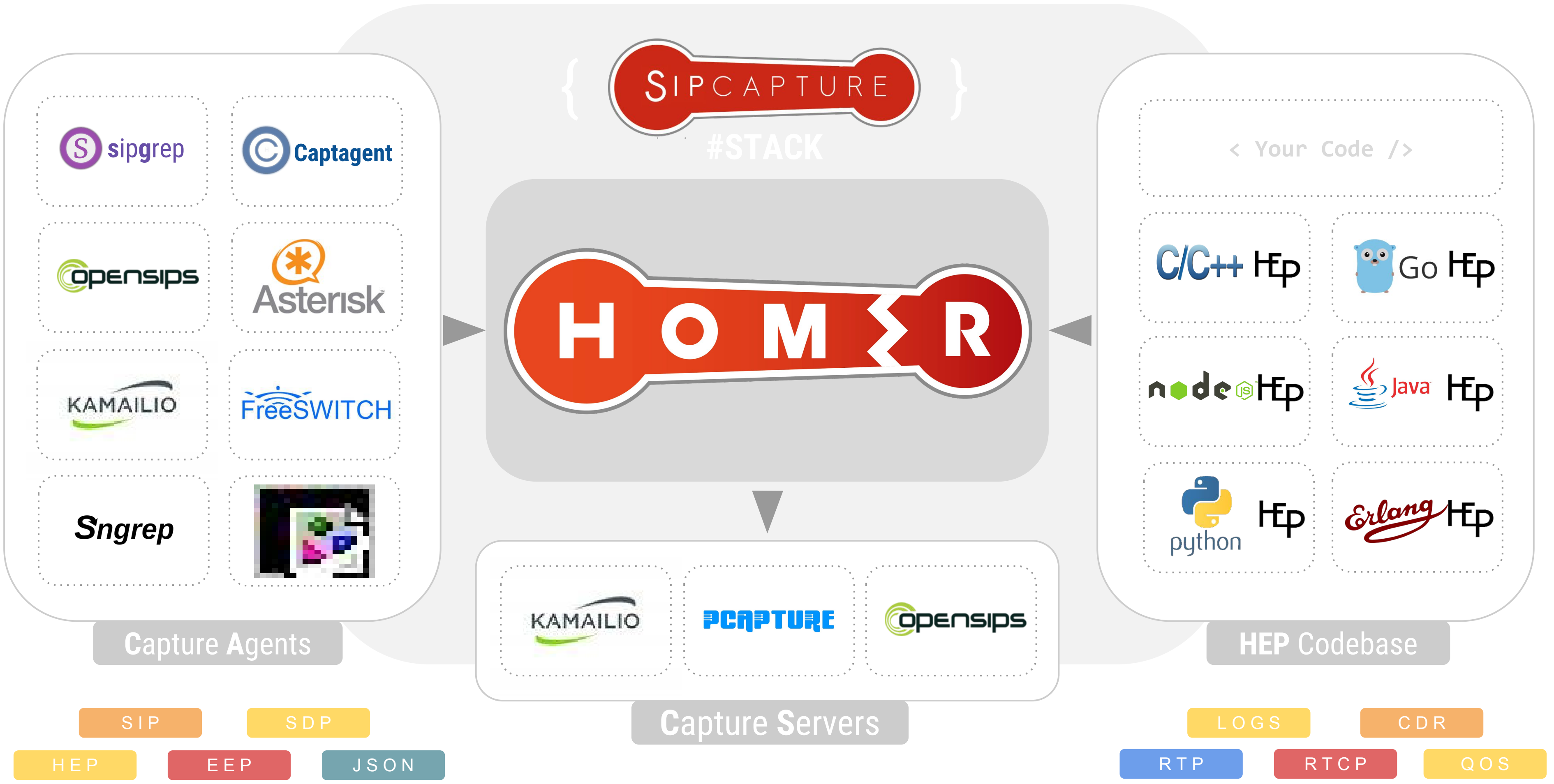


SIP SDP

HEP EEP JSON

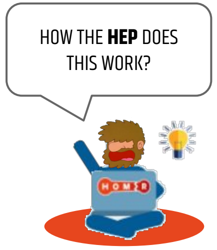
LOGS CDR

RTP RTCP QOS


{ SIPCAPTURE }
#STACK
HOMER
sipgrep
Captagent
opensips
Asterisk
KAMAILIO
FreeSWITCH
Sngrep
Capture Agents
< Your Code />
C/C++ HEP
Go HEP
node HEP
Java HEP
python HEP
Erlang HEP
HEP Codebase
KAMAILIO
PCAPTURE
opensips
Capture Servers
SIP
SDP
HEP
EEP
JSON
LOGS
CDR
RTP
RTCP
QOS

Meet #**HOMER** = VoIP & RTC Time Machine

100% Open Source VoIP Monitoring and Troubleshooting Tools



Meet #**HOMER** = VoIP & RTC Time Machine

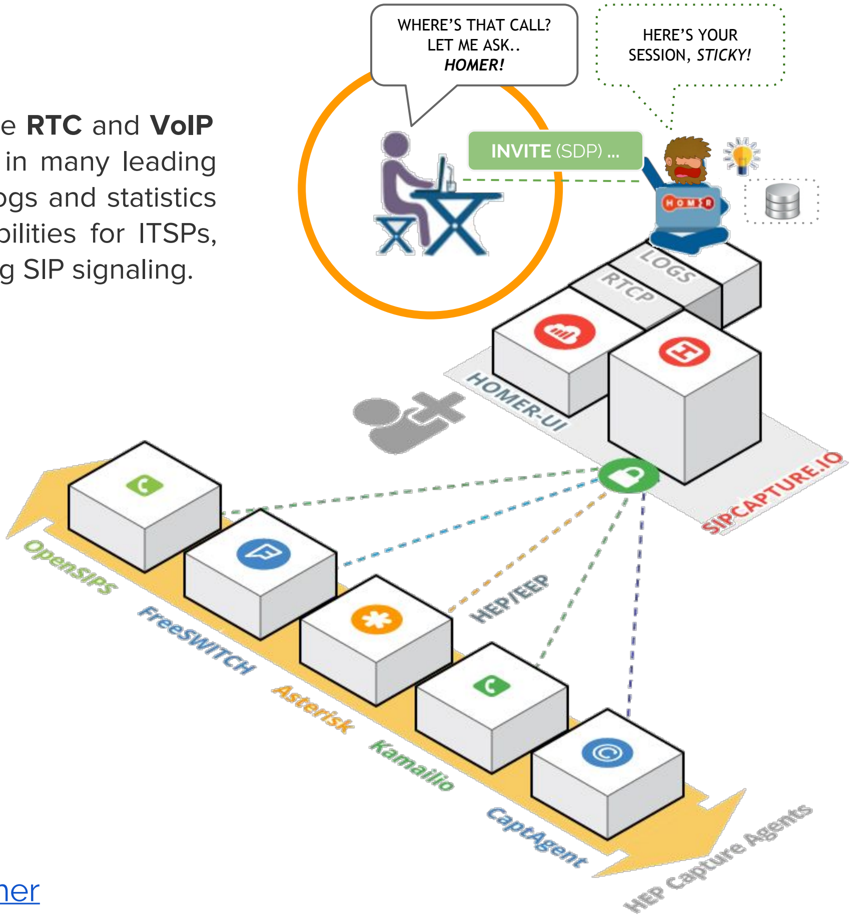
100% Open Source VoIP Monitoring and Troubleshooting Tools

HOMER is part of the **SIPCAPTURE** stack, a robust, carrier-grade, scalable **RTC** and **VoIP** Capture and Monitoring application with built in support out of the box in many leading platforms, ready to process, index & store insane amounts of signaling, logs and statistics and providing instant search, end-to-end analysis and drill-down capabilities for ITSPs, VoIP Providers Trunk Suppliers as well as Enterprises and Developers using SIP signaling.

HOMER provides many features and advantages, including:

- Instant centralized access to present and past signaling & stats
- Full SIP/SDP payload retention with precise timestamping
- Automatic correlation of sessions, logs and reports
- Support for RTP and RTCP Media statistics and analytics
- Visual representation of multi protocol session call-flows
- Fast detection of usage and system anomalies
- System agnostic view of VoIP and RTC traffic flows
- Unlimited plug & play capture agents and HEP custom data feeds
- Multi-User and Customizable UI based on JS/Angular/D3
- PCAP Exporting and Sharing functionality with 3rd parties

... and much more!



FIND ALL ABOUT HOMER: http://github.com/sipcapture/homer

SIPCAPTURE **HOMER** Capture Architecture Elements

SIPCAPTURE **HOMER** is composed of two basic building blocks / elements:

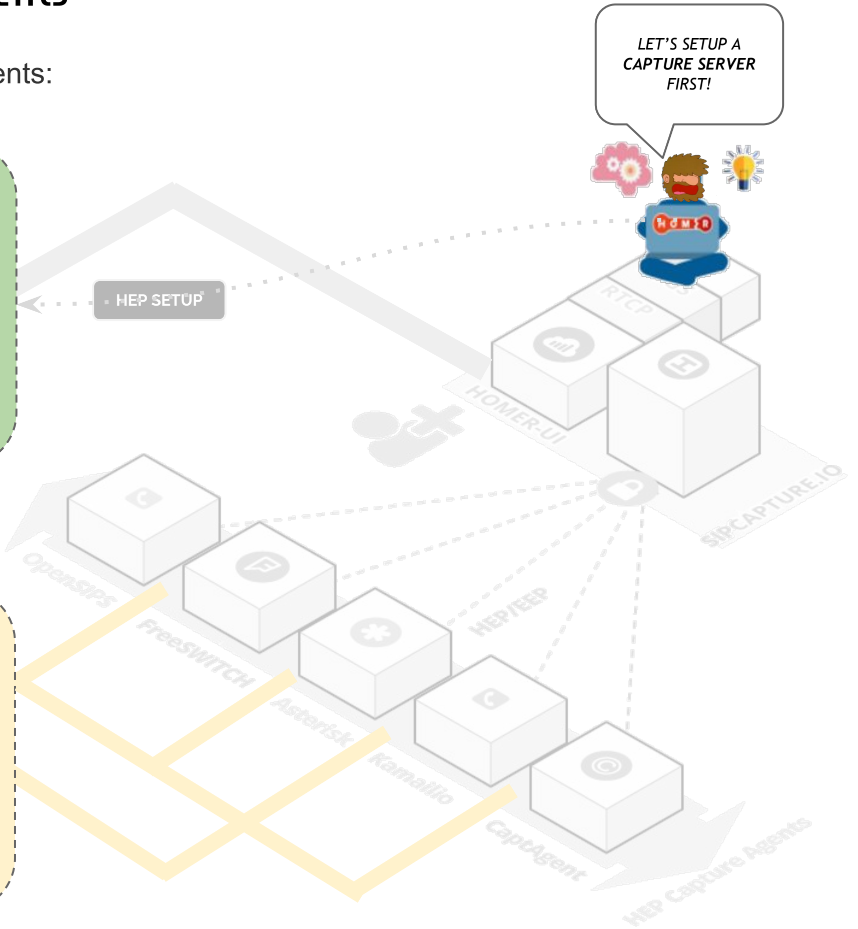
CS:HEP CAPTURE SERVER (Includes API + UI)

The *Capture Server Collects, Indexes and Stores* SIP packets received from *Capture Agents* using *HEP/EEP, IPIP, JSON Payloads* Encapsulation or RAW *SIP* packets captured from Ethernet interfaces and mirrored switch ports, using flexible rules, triggers and arbitrary statistics defined in the powerful, extensible and fully customizable core capture plan *(Kamailio or OpenSIPS)*

CA:HEP CAPTURE AGENT(s)

The Capture Agent captures and sends encapsulated packets or json data to a Capture Server using the *HEP/EEP* Encapsulation protocol via UDP/TCP

The Capture Agent role can be covered by multiple elements or native HEP modules running on different platforms and distributed in a completely modular fashion, easy to scale, grow and expand alongside the monitored infrastructure and systems, allowing flexible support for any network topology including cloud scenarios.



SIPCAPTURE **HOMER** Capture Architecture Elements

SIPCAPTURE **HOMER** is composed of two basic building blocks / elements:

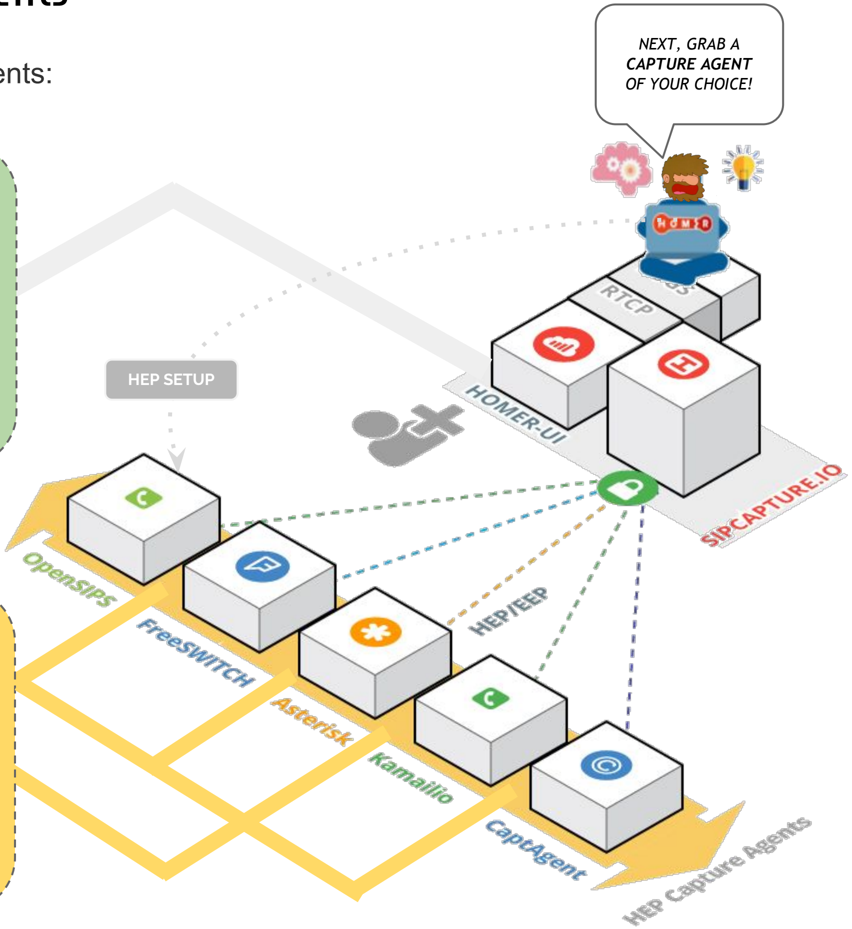
CS:HEP CAPTURE SERVER (Includes API + UI)

The *Capture Server Collects, Indexes and Stores* SIP packets received from *Capture Agents* using *HEP/EEP, IPIP, JSON Payloads* Encapsulation or RAW *SIP* packets captured from Ethernet interfaces and mirrored switch ports, using flexible rules, triggers and arbitrary statistics defined in the powerful, extensible and fully customizable core capture plan *(Kamailio or OpenSIPS)*

CA:HEP CAPTURE AGENT(s)

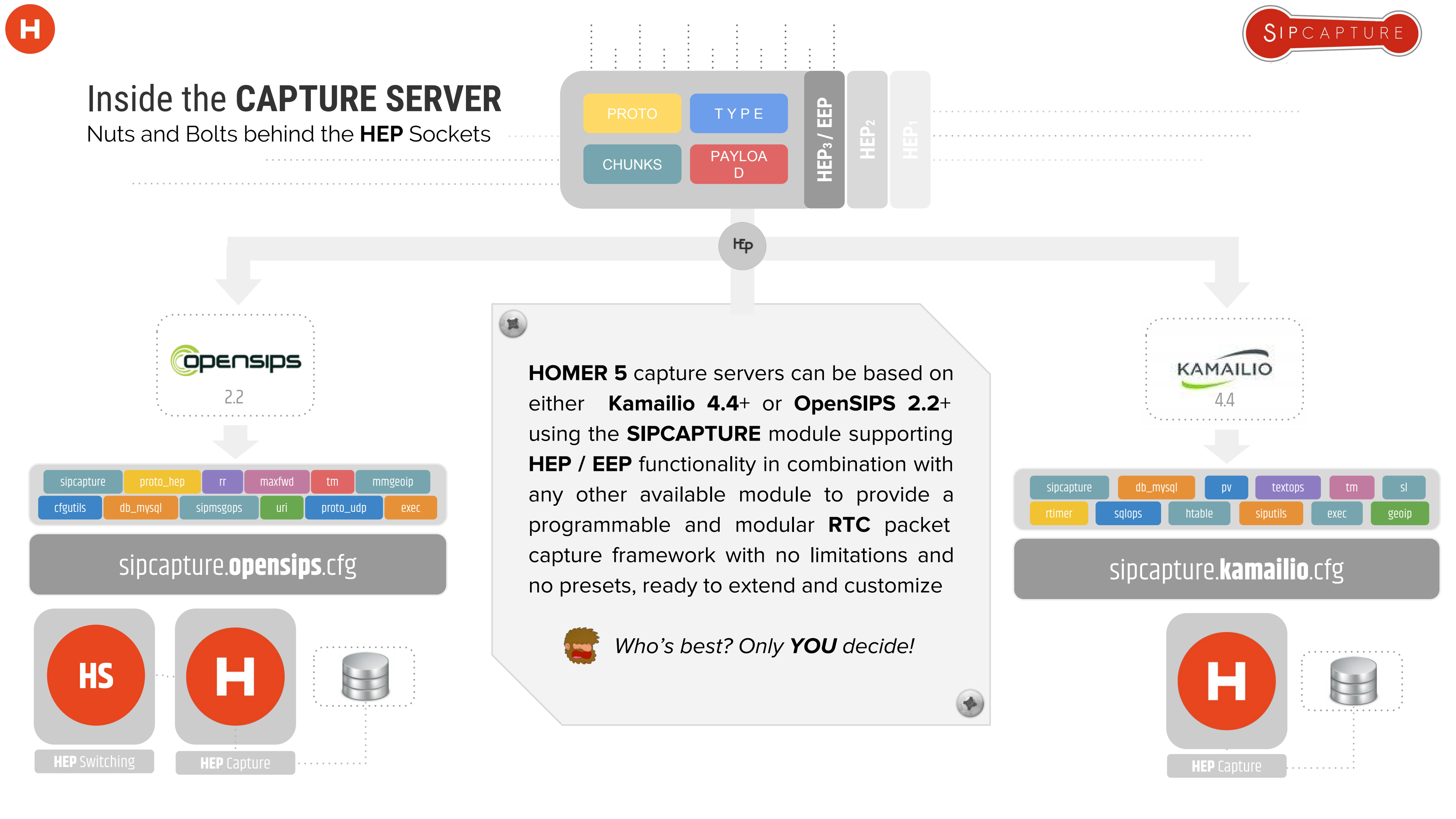
The Capture Agent captures and sends encapsulated packets or json data to a Capture Server using the *HEP/EEP* Encapsulation protocol via UDP/TCP

The Capture Agent role can be covered by multiple elements or native HEP modules running on different platforms and distributed in a completely modular fashion, easy to scale, grow and expand alongside the monitored infrastructure and systems, allowing flexible support for any network topology including cloud scenarios.



Inside the **CAPTURE SERVER**

Nuts and Bolts behind the **HEP** Sockets

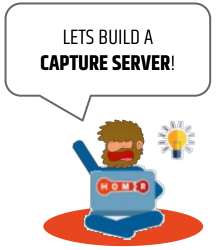


HOMER 5 capture servers can be based on either **Kamailio 4.4**+ or **OpenSIPS 2.2**+ using the **SIPCAPTURE** module supporting **HEP / EEP** functionality in combination with any other available module to provide a programmable and modular **RTC** packet capture framework with no limitations and no presets, ready to extend and customize

*Who's best? Only **YOU** decide!*

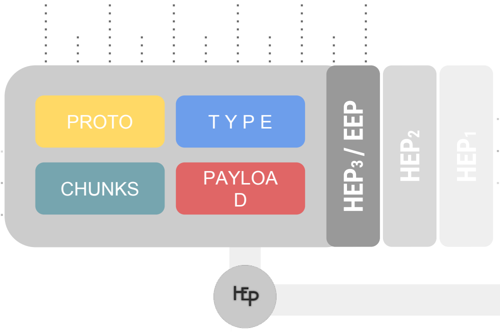
Inside the **CAPTURE SERVER**

Built-in **HEP** functionality in Kamailio 4.4



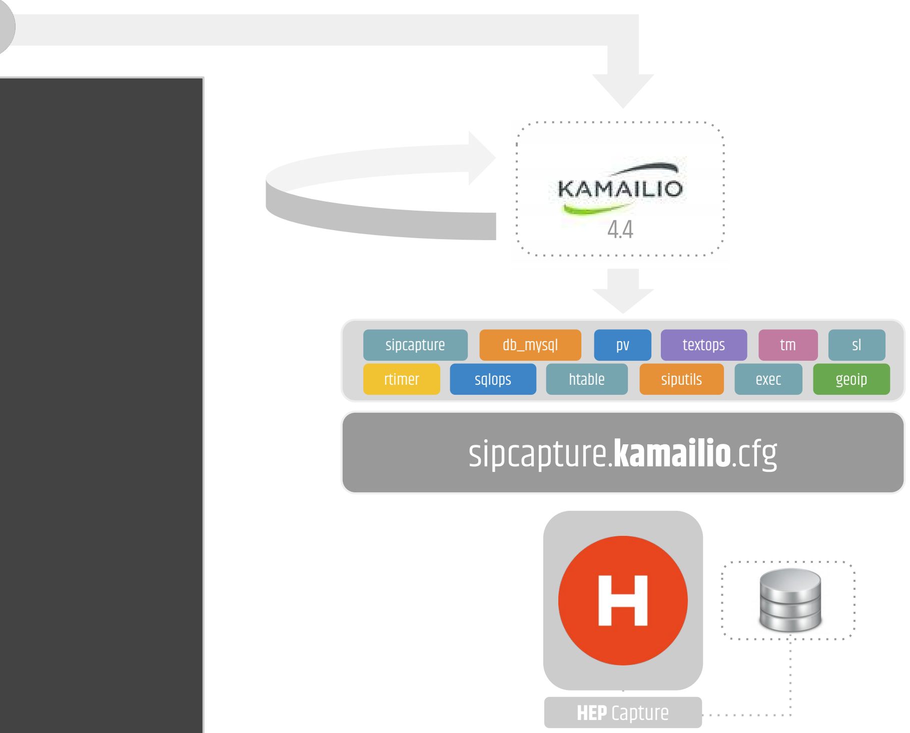
Inside the CAPTURE SERVER

Built-in **HEP** functionality in Kamailio 4.4



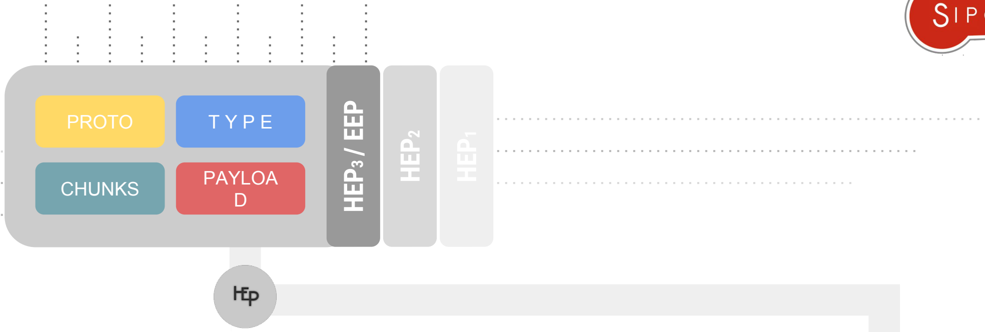
SIPCAPTURE Capture Server: Preferences

```
#!KAMAILIO
# Example configuration file for a sipcapture node
#

####### Global Parameters definitions #########
#
# Please, make all your configuration changes here
#
# *** To enable extra stats
#     - define WITH_STATISTIC_METHOD_EXTRA
#     - define WITH_STATISTIC_INVITE_1XX


#!substdef "!HOMER_DB_USER!homer_user!g"
#!substdef "!HOMER_DB_PASSWORD!homer_password!g"
#!substdef "!HOMER_LISTEN_PROTO!udp!g"
#!substdef "!HOMER_LISTEN_IF!0.0.0.0!g"
#!substdef "!HOMER_LISTEN_PORT!9060!g"
#!substdef "!HOMER_STATS_SERVER!tcp:HOMER_LISTEN_IF:8888!g"
```



Inside the **CAPTURE SERVER**

Built-in **HEP** functionality in Kamailio 4.4

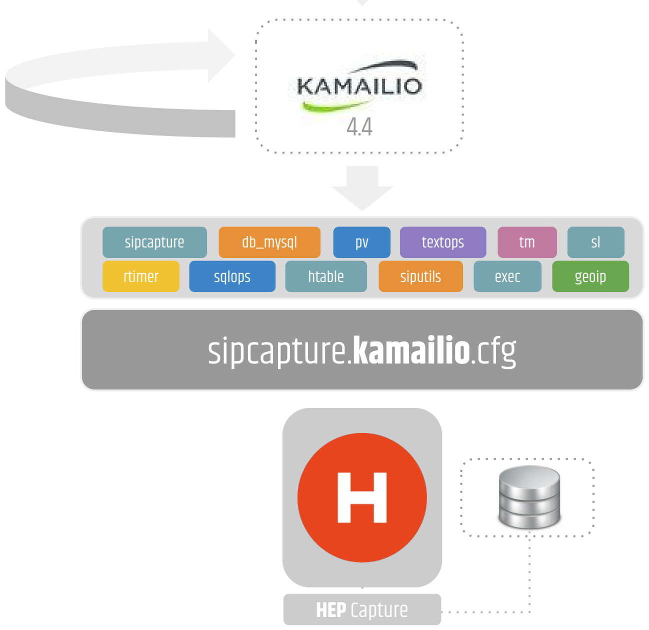


SIPCAPTURE Capture Server: Modules

```
listen=HOMER_LISTEN_PROTO:HOMER_LISTEN_IF:HOMER_LISTEN_PORT

loadmodule "pv.so"
loadmodule "db_mysql.so"
loadmodule "sipcapture.so"
loadmodule "textops.so"
loadmodule "rtimer.so"
loadmodule "xlog.so"
loadmodule "sqlops.so"
loadmodule "htable.so"
loadmodule "tm.so"
loadmodule "sl.so"
loadmodule "siputils.so"
loadmodule "exec.so"
```

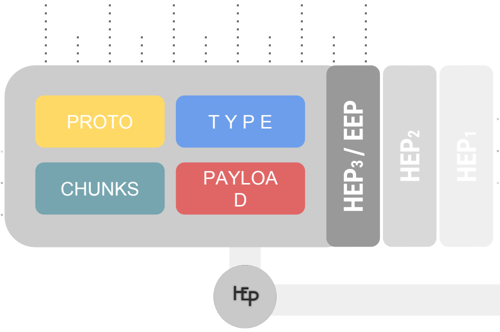
```
#!ifdef WITH_HOMER_GEO
        loadmodule "geoip.so"
#!endif

#!ifdef WITH_HOMER_CUSTOM_STATS
        loadmodule "xhttp.so"
        loadmodule "jansson.so"
        loadmodule "avpops.so"
#!endif
```

```
modparam("htable", "htable", "a=>size=8;autoexpire=400")
modparam("htable", "htable", "b=>size=8;autoexpire=31")
modparam("htable", "htable", "c=>size=8;autoexpire=31")
modparam("rtimer", "timer", "name=ta;interval=60;mode=1;")
modparam("rtimer", "exec", "timer=ta;route=TIMER_STATS")
```



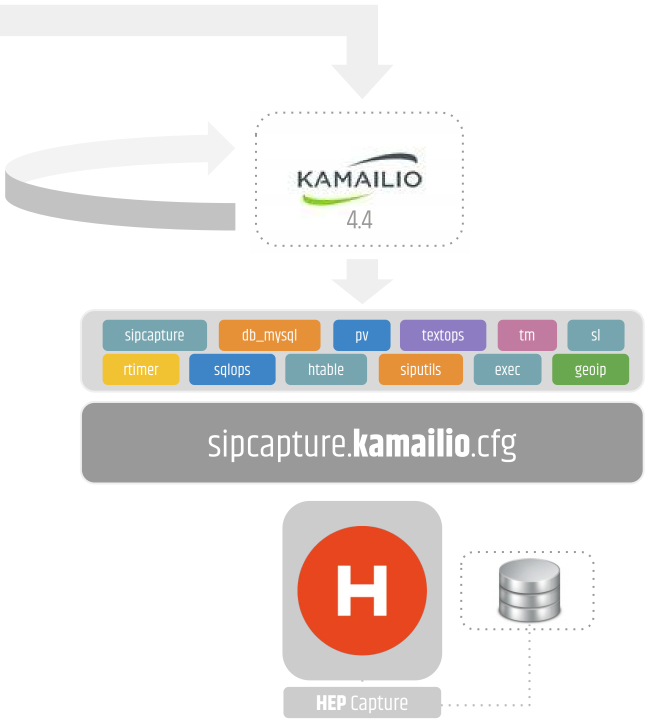
Inside the **CAPTURE SERVER**

Built-in **HEP** functionality in Kamailio 4.4



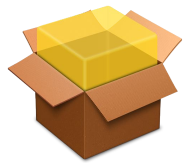
SIPCAPTURE Capture Server: Module Parameters

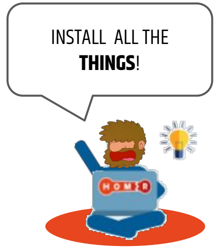
```
####### Capture Logic ########

modparam("sipcapture", "db_url", "mysql://HOMER_DB_USER:HOMER_DB_PASSWORD@127.0.0.1
/homer_data")
modparam("sipcapture", "capture_on", 1)
modparam("sipcapture", "hep_capture_on", 1)
modparam("sipcapture", "insert_retries", 5)
modparam("sipcapture", "insert_retry_timeout", 10)
#modparam("sipcapture", "capture_node", "homer01")

#Stats time
stats.min = 5 desc "My stats TIME min"

Main SIP request routing logic
# - processing of any incoming SIP request starts with this route
route {
        For the full Configuration see:
        github.com/sipcapture/homer-api/blob/master/examples/sipcapture/sipcapture.kamailio

    }
```



Install HOMER 5 in 5 minutes

Learn how to Install and use the SIPCAPTURE Stack





Install HOMER 5 in 5 minutes

SIPCAPTURE basic stack using *Homer-Installer* on supported OSs

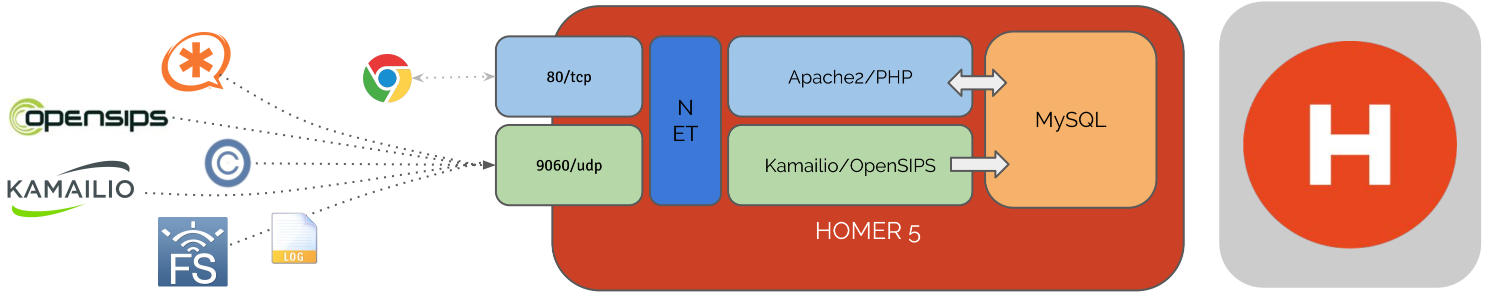
Get started with the latest and greatest **HOMER** version in no time using the semi-automatic installer!

Get a vanilla **Debian** 8 or **CentOS** 7 net-install image up and running with no special settings.
Download and run the **Homer 5** application installer *[Apache2-PHP/MySQL-InnoDB/Kamailio|OpenSIPS/sipcapture]*

```
bash <( curl -s https://cdn.rawgit.com/sipcapture/homer-installer/master/homer_installer.sh
```

Packages + Services will be installed with minimal interaction. Once completed, *login to the UI* using the default settings.

That's all - Easy wasn't it? Here's a quick diagram for the bundle you just installed:



Install HOMER 5 in 5 minutes

SIPCAPTURE public *Homer-Docker* image Single or Multi Container at http://github.com/sipcapture/homer-docker

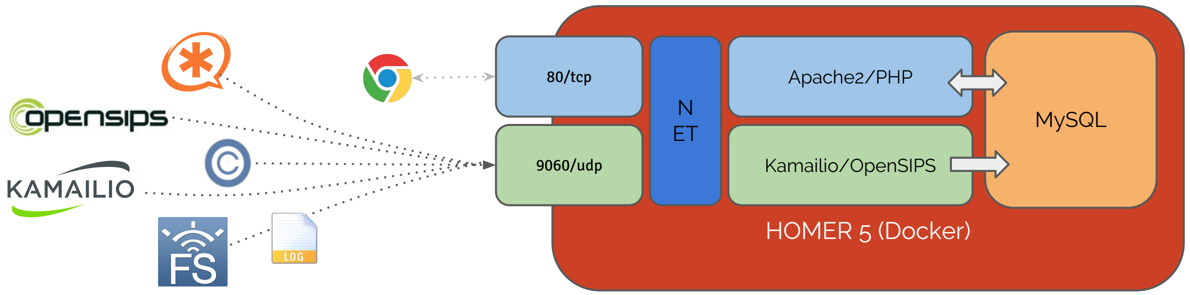
Pull and run the Docker **Homer 5** application bundle *[Apache2-PHP/MySQL-InnoDB/Kamailio-sipcapture]*

```
# docker run -tid --name homer5 -p 80:80 -p 9060:9060/udp -p 9061:9061/tcp sipcapture/homer-docker
4280d228ae472c02eded508bf587fb0bde6bd1604b1fc65c0490d0648f6fbe06
```

Verify the **Homer 5** container is running and all desired ports are published:

```
# docker ps
CONTAINER ID   IMAGE              COMMAND     CREATED        STATUS         PORTS              NAMES
4280d228ae47   qxip/homer-docker  "/run.sh"   1 minute ago   Up 1 minutes   80/tcp,9060/udp    0c0f7939-5ab9-401e-af63-ce8728221d0b-n1/homer5
```

Note down your **IP** for sending HEP traffic to your container using your favourite *HEP/EEP Capture Agent*:

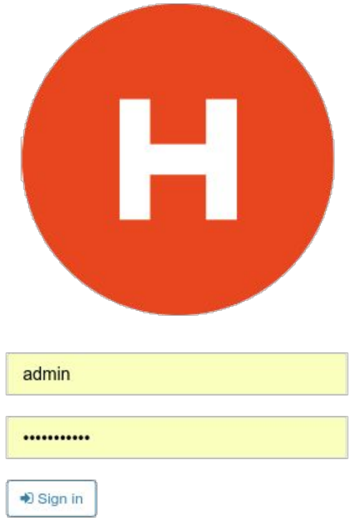


Inside **HOMER** 5

Your very first Login

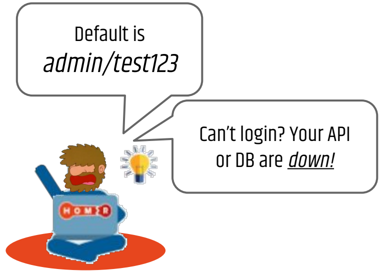
Congratulations! Your very own **HOMER 5** capture server should be now up and running!

It's time to *login*, get familiar with the available *tools* and configure preferences to handle and *correlate* data sessions



admin

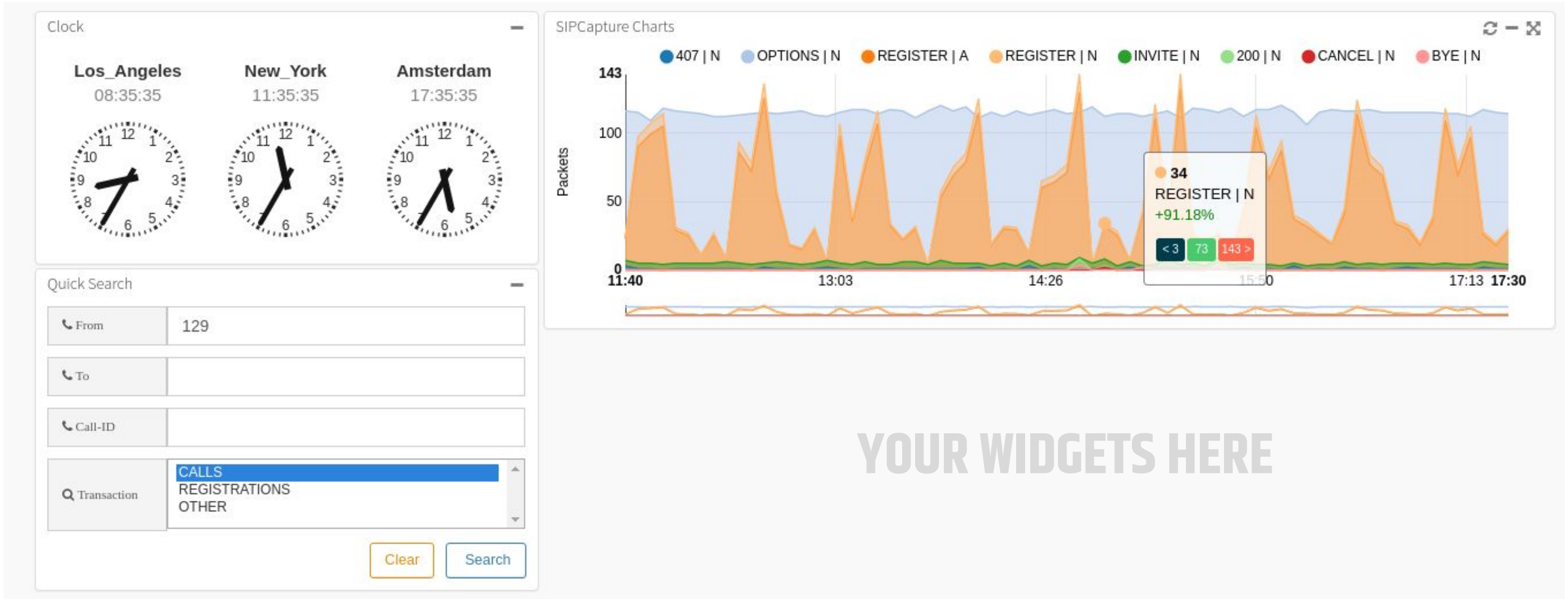
••••••••••••


Default is *admin/test123*
Can't login? Your API or DB are *down!*

Inside **HOMER** 5

Dashboard and Widget management

Homer 5 features a dynamic dashboard/widget system which can easily be extended using standard javascript and AngularJS. All chart and form elements are user-defined and can be assembled based on requirements using the provided examples feed either internal or external data sources, and synchronized to the master Time-Range selector



Clock

Los_Angeles 08:35:35

New_York 11:35:35

Amsterdam 17:35:35

Quick Search

From	129
To	
Call-ID	
Transaction	CALLS REGISTRATIONS OTHER

Clear Search

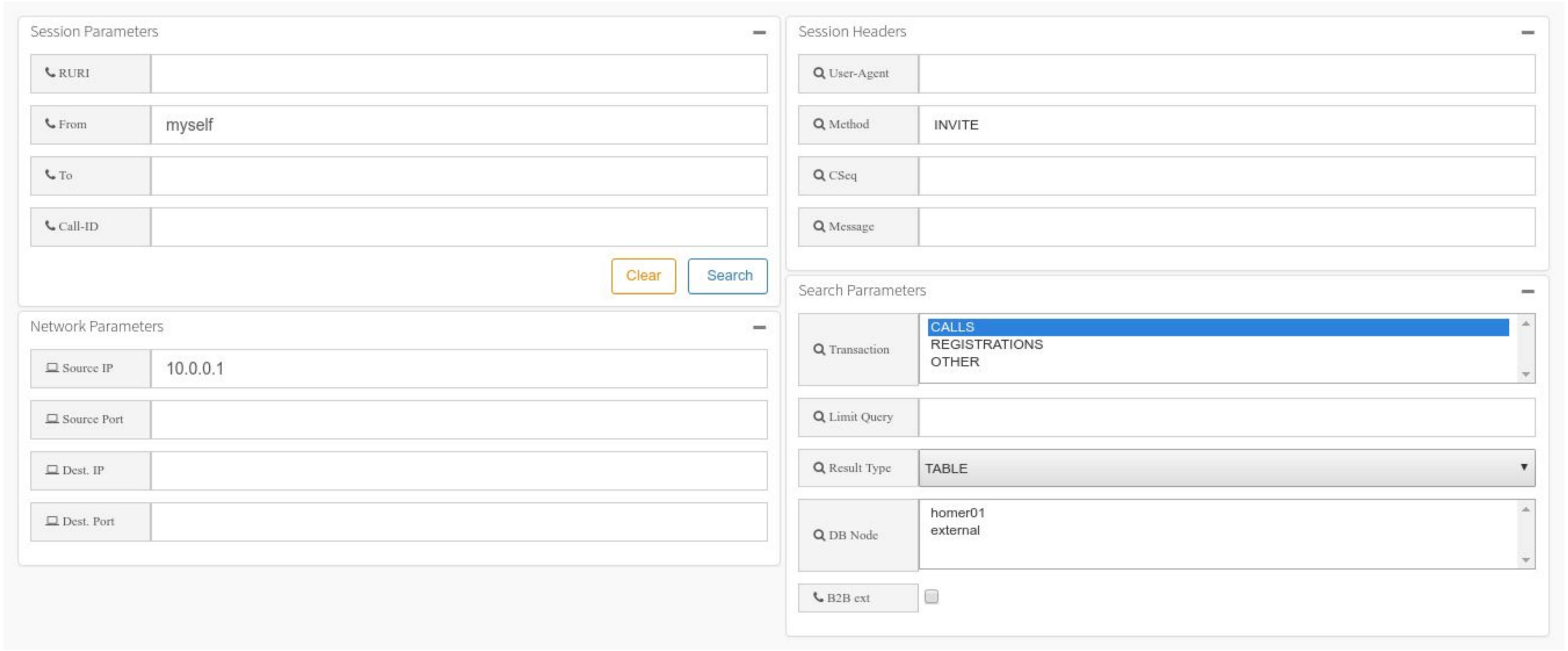
SIPCapture Charts

YOUR WIDGETS HERE

Inside **HOMER** 5

Search Form Widget management

Homer 5 is dynamic all the way to Forms. Shape your Search widgets and any number of custom Panels for your teams:



Session Parameters

RURI	
From	myself
To	
Call-ID	

Network Parameters

Source IP	10.0.0.1
Source Port	
Dest. IP	
Dest. Port	

Session Headers

User-Agent	
Method	INVITE
CSeq	
Message	

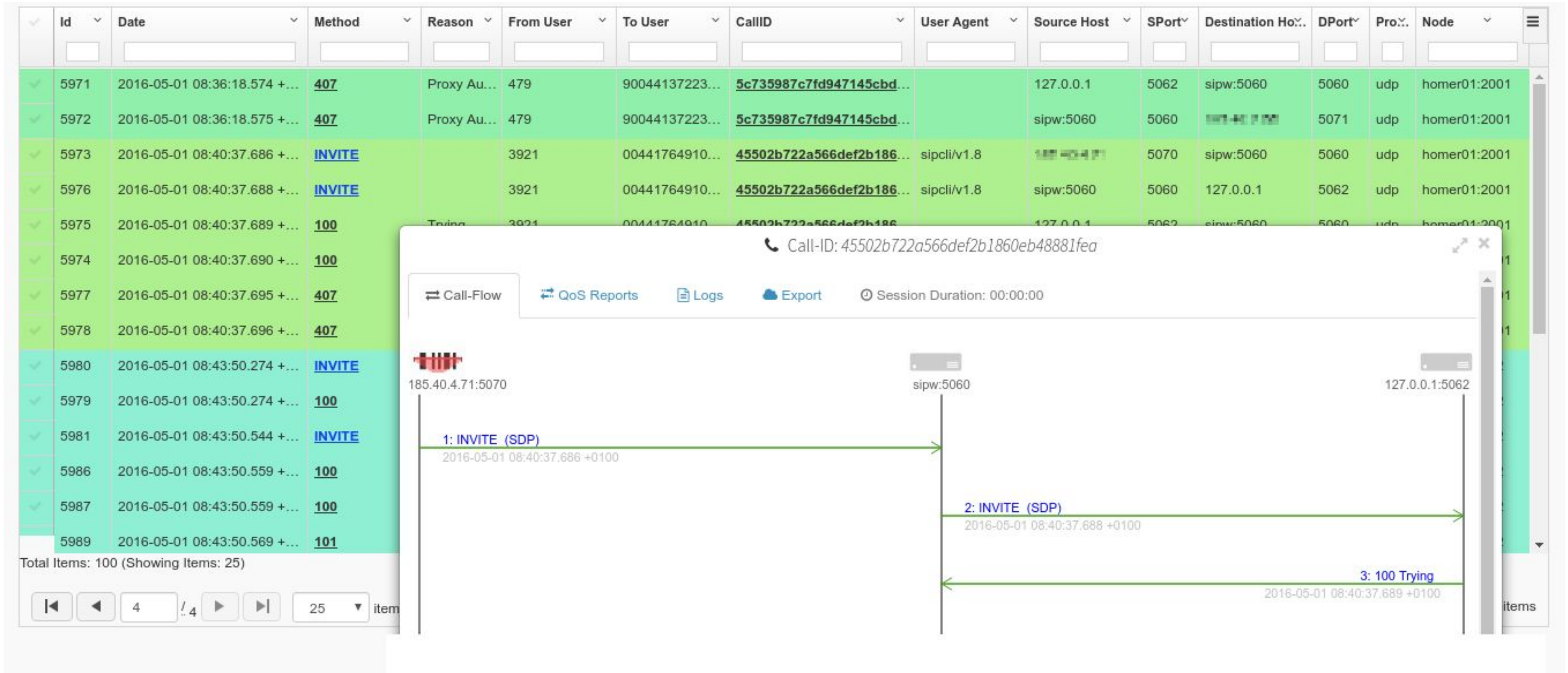
Search Parrameters

Transaction	CALLS (selected), REGISTRATIONS, OTHER
Limit Query	
Result Type	TABLE
DB Node	homer01, external
B2B ext	☐

Inside **HOMER** 5

Search Results and Flow management

Signaling Search results are intuitive, customizable and designed to provide the quickest path for your Troubleshooting:



	Id	Date	Method	Reason	From User	To User	CallID	User Agent	Source Host	SPort	Destination Ho…	DPort	Pro…	Node
✓	5971	2016-05-01 08:36:18.574 +…	407	Proxy Au…	479	90044137223…	5c735987c7fd947145cbd…		127.0.0.1	5062	sipw:5060	5060	udp	homer01:2001
✓	5972	2016-05-01 08:36:18.575 +…	407	Proxy Au…	479	90044137223…	5c735987c7fd947145cbd…		sipw:5060	5060	[redacted]	5071	udp	homer01:2001
✓	5973	2016-05-01 08:40:37.686 +…	INVITE		3921	00441764910…	45502b722a566def2b186…	sipcli/v1.8	[redacted]	5070	sipw:5060	5060	udp	homer01:2001
✓	5976	2016-05-01 08:40:37.688 +…	INVITE		3921	00441764910…	45502b722a566def2b186…	sipcli/v1.8	sipw:5060	5060	127.0.0.1	5062	udp	homer01:2001
✓	5975	2016-05-01 08:40:37.689 +…	100	Trying	3921	00441764910…	45502b722a566def2b186…		127.0.0.1	5062	sipw:5060	5060	udp	homer01:2001
✓	5974	2016-05-01 08:40:37.690 +…	100											
✓	5977	2016-05-01 08:40:37.695 +…	407											
✓	5978	2016-05-01 08:40:37.696 +…	407											
✓	5980	2016-05-01 08:43:50.274 +…	INVITE											
✓	5979	2016-05-01 08:43:50.274 +…	100											
✓	5981	2016-05-01 08:43:50.544 +…	INVITE											
✓	5986	2016-05-01 08:43:50.559 +…	100											
✓	5987	2016-05-01 08:43:50.559 +…	100											
	5989	2016-05-01 08:43:50.569 +…	101											

Total Items: 100 (Showing Items: 25)

4 / 4 — 25 items

Call-ID: *45502b722a566def2b1860eb48881fea*

Call-Flow | QoS Reports | Logs | Export | Session Duration: 00:00:00

185.40.4.71:5070 — sipw:5060 — 127.0.0.1:5062

1: INVITE (SDP) — 2016-05-01 08:40:37.686 +0100

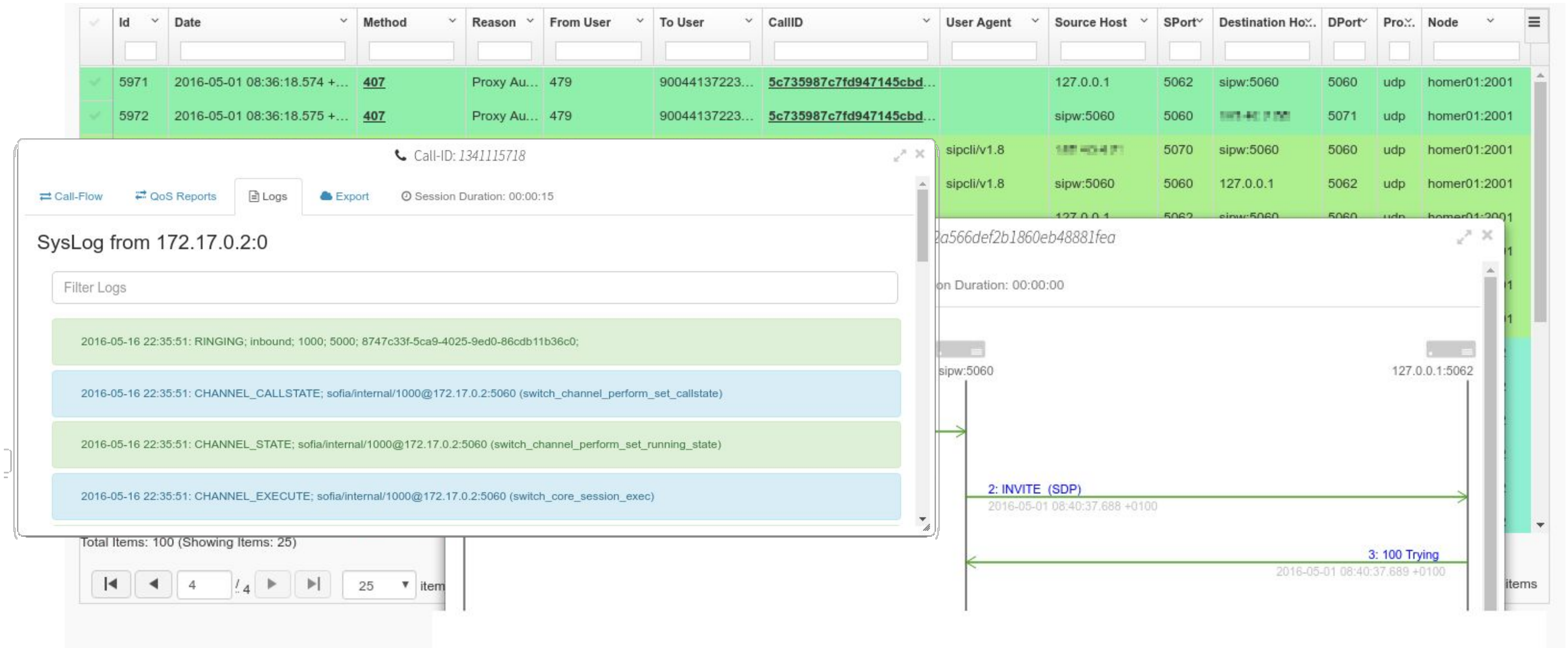
2: INVITE (SDP) — 2016-05-01 08:40:37.688 +0100

3: 100 Trying — 2016-05-01 08:40:37.689 +0100

Inside **HOMER** 5

Search Results and Flow management

HOMER doesn't stop here! Let's add Correlated **Logs** to the mix…



Id	Date	Method	Reason	From User	To User	CallID	User Agent	Source Host	SPort	Destination Ho...	DPort	Pro...	Node
5971	2016-05-01 08:36:18.574 +...	407	Proxy Au...	479	90044137223...	5c735987c7fd947145cbd...		127.0.0.1	5062	sipw:5060	5060	udp	homer01:2001
5972	2016-05-01 08:36:18.575 +...	407	Proxy Au...	479	90044137223...	5c735987c7fd947145cbd...		sipw:5060	5060	[redacted]	5071	udp	homer01:2001
							sipcli/v1.8	[redacted]	5070	sipw:5060	5060	udp	homer01:2001
							sipcli/v1.8	sipw:5060	5060	127.0.0.1	5062	udp	homer01:2001
								127.0.0.1	5062	sipw:5060	5060	udp	homer01:2001

Total Items: 100 (Showing Items: 25)

4 / 4 25 item

Call-ID: 1341115718

Call-Flow | QoS Reports | Logs | Export | Session Duration: 00:00:15

SysLog from 172.17.0.2:0

Filter Logs

```
2016-05-16 22:35:51: RINGING; inbound; 1000; 5000; 8747c33f-5ca9-4025-9ed0-86cdb11b36c0;
2016-05-16 22:35:51: CHANNEL_CALLSTATE; sofia/internal/1000@172.17.0.2:5060 (switch_channel_perform_set_callstate)
2016-05-16 22:35:51: CHANNEL_STATE; sofia/internal/1000@172.17.0.2:5060 (switch_channel_perform_set_running_state)
2016-05-16 22:35:51: CHANNEL_EXECUTE; sofia/internal/1000@172.17.0.2:5060 (switch_core_session_exec)
```

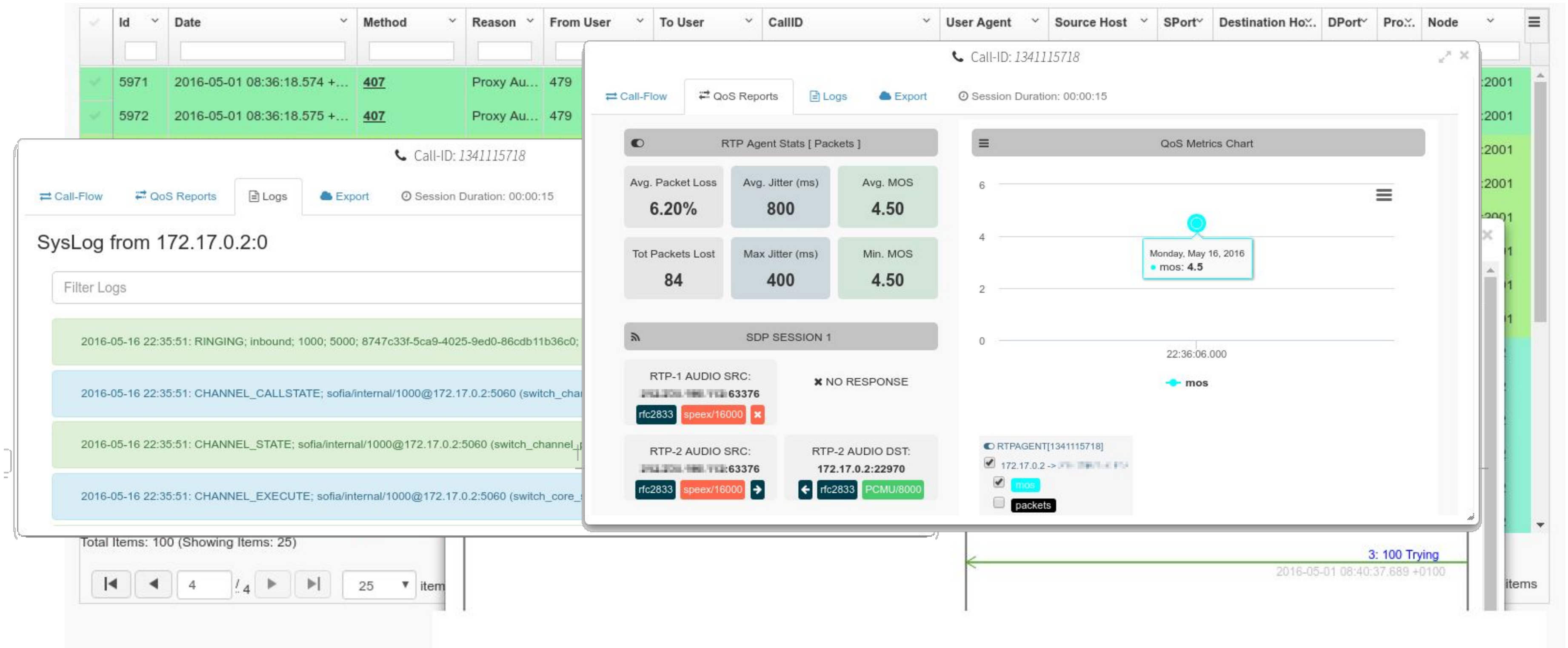
2a566def2b1860eb48881fea

on Duration: 00:00:00

Inside **HOMER** 5

Search Results and Flow management

HOMER doesn't stop here! Let's add Correlated **Logs** to the mix… how about some RTP/RTCP **Media Statistics**, too?



Id	Date	Method	Reason	From User	To User	CallID	User Agent	Source Host	SPort	Destination Ho…	DPort	Pro…	Node
5971	2016-05-01 08:36:18.574 +…	407	Proxy Au…	479									:2001
5972	2016-05-01 08:36:18.575 +…	407	Proxy Au…	479									:2001

Total Items: 100 (Showing Items: 25)

4 / 4 — 25 items

Call-ID: 1341115718

Call-Flow | QoS Reports | Logs | Export | Session Duration: 00:00:15

SysLog from 172.17.0.2:0

Filter Logs

```
2016-05-16 22:35:51: RINGING; inbound; 1000; 5000; 8747c33f-5ca9-4025-9ed0-86cdb11b36c0;
2016-05-16 22:35:51: CHANNEL_CALLSTATE; sofia/internal/1000@172.17.0.2:5060 (switch_cha
2016-05-16 22:35:51: CHANNEL_STATE; sofia/internal/1000@172.17.0.2:5060 (switch_channel_
2016-05-16 22:35:51: CHANNEL_EXECUTE; sofia/internal/1000@172.17.0.2:5060 (switch_core_
```

Call-ID: 1341115718

Call-Flow | QoS Reports | Logs | Export | Session Duration: 00:00:15

RTP Agent Stats [Packets]

Avg. Packet Loss	Avg. Jitter (ms)	Avg. MOS
6.20%	800	4.50
Tot Packets Lost	**Max Jitter (ms)**	**Min. MOS**
84	400	4.50

SDP SESSION 1

RTP-1 AUDIO SRC: [redacted]:63376 — rfc2833 speex/16000 — ✖ NO RESPONSE

RTP-2 AUDIO SRC: [redacted]:63376 — rfc2833 speex/16000

RTP-2 AUDIO DST: 172.17.0.2:22970 — rfc2833 PCMU/8000

Inside **HOMER** 5

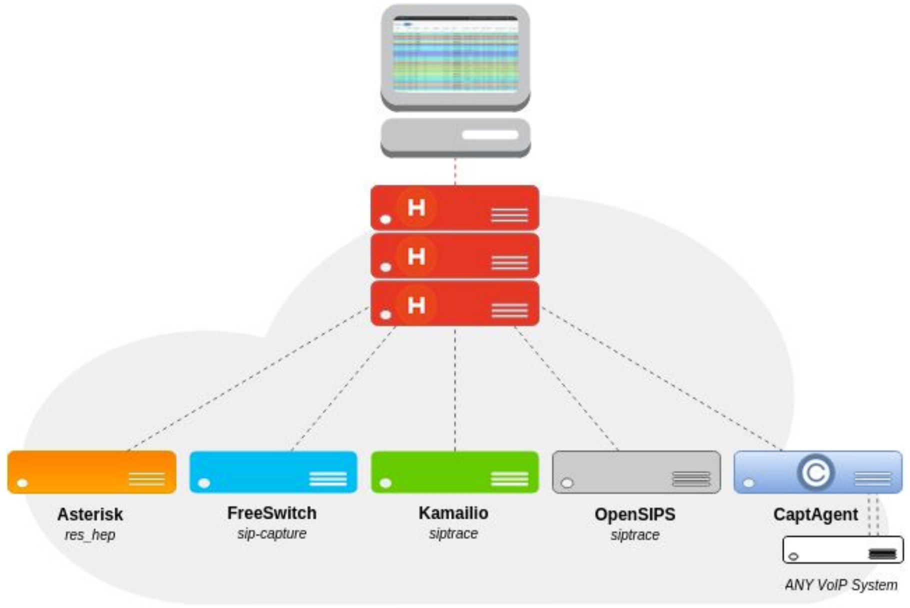
Looks and Sounds **Great**! There’s only one *little* problem . . .

Inside **HOMER** 5

Looks and Sounds **Great**! There's only one *little* problem . . .

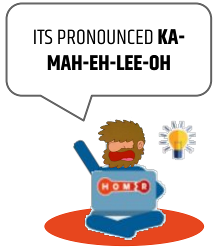
HEP/EEP

FEEDING **HOMER**



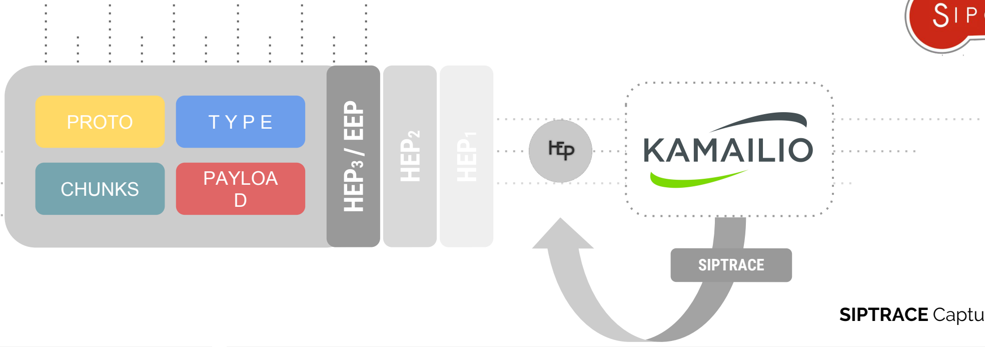
SIPTRACE Packet Capture

Integrated **HEP** functionality in **Kamailio**



SIPTRACE Packet Capture

Integrated **HEP** functionality in Kamailio



SIPTRACE Capture Agent

SIPTRACE Capture Agent

```
#!KAMAILIO
debug=1
log_stderror=no
memdbg=5
memlog=5
log_facility=LOG_LOCAL0
fork=yes
children=4
disable_tcp=yes

listen=udp:192.168.0.1:5060

/* port to listen to
port=5060

####### Modules Section ########
mpath="/usr/local/lib64/kamailio/modules_k/:/usr/local/lib64/kamailio/modules/"

loadmodule "mi_fifo.so"
loadmodule "kex.so"
loadmodule "tm.so"
loadmodule "sl.so"
loadmodule "rr.so"
loadmodule "pv.so"
loadmodule "maxfwd.so"
loadmodule "xlog.so"
loadmodule "textops.so"
loadmodule "siputils.so"
```

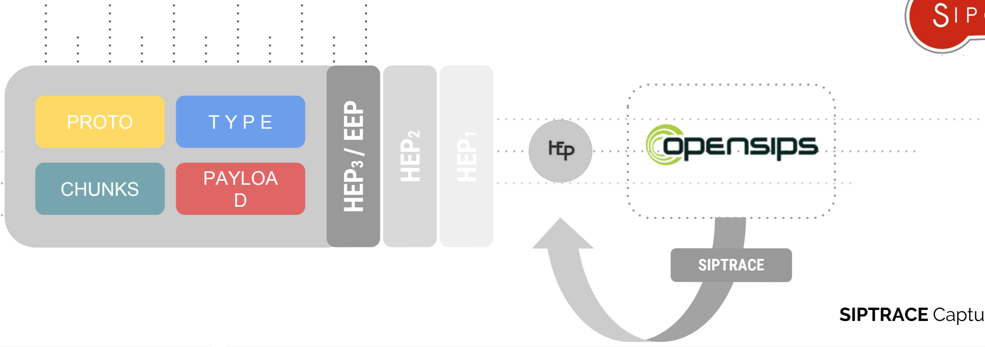
```
#Siptrace
loadmodule "siptrace.so"
modparam("siptrace", "duplicate_uri", "sip:10.0.0.1:9060")
modparam("siptrace", "hep_mode_on", 1)
modparam("siptrace", "trace_to_database", 0)
modparam("siptrace", "trace_flag", 22)
modparam("siptrace", "trace_on", 1)

####### Routing Logic ########
# Main SIP request routing logic
route {
      sip_trace();                    # duplicate all SIP messages
      setflag(22);                    # enable capture by TM/SL
      ....
      route(RELAY);
}

onreply_route {
      sip_trace();                    # duplicate all response SIP messages
      ....
}

route[RELAY] {
      if (!t_relay()) {
            sl_reply_error();
      }
      exit;
}
```

SIPTRACE Packet Capture

Advanced **HEP** functionality in **OpenSIPS**

SIPTRACE Packet Capture

Advanced **HEP** functionality in **OpenSIPS**



SIPTRACE Capture Agent

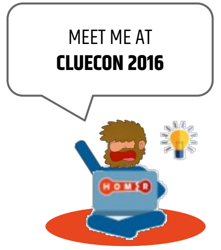
```
loadmodule "proto_hep.so"
loadmodule "siptrace.so"

#Socket to send
listen=hep_udp:10.0.0.1:9060

### a hep uri is in the following form: "hep:[ip]:[port]"
#Default version 3, you can set version 2, 1. And set transport, default udp.
modparam("siptrace", "trace_id", "[hep]uri=hep:192.168.100.6:6161;transport=udp;version=3")

route {

$var(trace_id) = "hep";
#you can define user to trace.
$var(user) = "osip_user@opensips.org";

 ### CHANGEME optional - 'd' is for tracing dialogs(need tm + dialog)
 ###                  't' for tracing transaction(need tm)
 ###                  'm' for tracing only this message

 /* Example 1: Trace a dialog  */

          if (has_totag()) {
                    match_dialog();
          } else {
                    if (is_method("INVITE") {
                              sip_trace("$var(trace_id)", "d", "$var(user)");
                    }
          }
```

SIPTRACE Capture Agent

```
/* Example 2: Trace initial INVITE and BYE */

          if (has_totag()) {
                    if (is_method("BYE")) {
                              sip_trace("$var(trace_id)", "m", "$var(user)")
                    }
          } else {
                    if (is_method("INVITE")) {
                              sip_trace("$var(trace_id)", "m", "$var(user)")
                    }
          }

/* Example 3: Trace initial INVITE transaction */
          if (!has_totag()) {
                    if (is_method("INVITE")) {
                              sip_trace("$var(trace_id)", "t", "$var(user)");
                    }
          }

/* Example 4: stateless transaction aware mode!*/
          /* tm module must not be loaded */
          if (is_method("REGISTER")) {
                    sip_trace("$var(trace_id)", "t", "$var(user)");
                    if (!www_authorize("", "subscriber")) {
                    /* siptrace will also catch the 401 generated by www_challenge() */
                              www_challenge("", "1");
                    }
          }
}
```

FreeSWITCH HEP/EEP Configuration

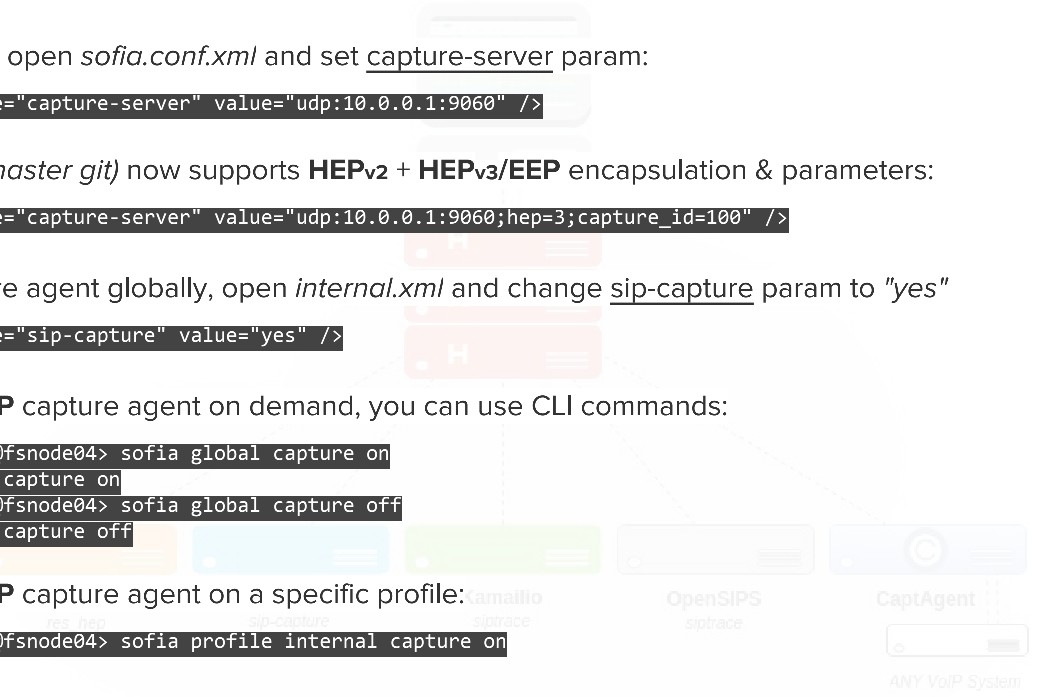
Example Usage of the Integrated Capture Agent for Monitoring



FreeSWITCH HEP/EEP Configuration

Example Usage of the Integrated Capture Agent for Monitoring

FreeSWITCH ships with a built-in HEP agent used to mirror/transfer packets unmodified and carries timestamp and several session key values in its headers, designed for capturing simple and complex scenarios with minimal configuration efforts.



To enable **HEP** capturing, open *sofia.conf.xml* and set capture-server param:

```
<param name="capture-server" value="udp:10.0.0.1:9060" />
```

NEW! Freeswitch v1.6.8 *(master git)* now supports **HEPv2** + **HEPv3/EEP** encapsulation & parameters:

```
<param name="capture-server" value="udp:10.0.0.1:9060;hep=3;capture_id=100" />
```

To enable the **HEP** capture agent globally, open *internal.xml* and change sip-capture param to *"yes"*

```
<param name="sip-capture" value="yes" />
```

To enable/disable the **HEP** capture agent on demand, you can use CLI commands:

```
freeswitch@fsnode04> sofia global capture on
+OK Global capture on
freeswitch@fsnode04> sofia global capture off
+OK Global capture off
```

To enable/disable the **HEP** capture agent on a specific profile:

```
freeswitch@fsnode04> sofia profile internal capture on
```

FreeSWITCH HEP/EEP Configuration + DOCKER

Example Usage of the Integrated Capture Agent for Monitoring

Let's add a Docker container running **FreeSWITCH 1.6.8** with native **HEP3** support to our stack:

```
# Create stateful volume
docker create --name fsdata --volume /usr/local/freeswitch/conf qxip/freeswitch-container:1.6.8 /bin/true

# Start FS using stateful volume for data
docker run -tid --name freeswitch -p 5060:5060/udp -p 5080:5080/udp --expose 16384-32768 --volumes-from fsdata qxip/freeswitch-container:1.6.8

# Bash in
# docker exec -i -t freeswitch /bin/bash
```

FreeSWITCH is up and running - Great!

Let's start by configuring our brand new **capture-server** in */usr/local/freeswitch/conf/autoload_configs/sofia.conf.xml*:

```
    <!-- the new format for HEPv2/v3 and capture ID protocol:host:port;hep=2;capture_id=200;  -->

    <param name="capture-server" value="udp:10.0.0.1:9060;hep=3;capture_id=100"/>
```

The **sip-capture** functionality is controlled at profile level - let's add it to: */usr/local/freeswitch/conf/autoload_configs/internal.xml*:

```
    <param name="sip-capture" value="yes"/>
```

FreeSWITCH HEP/EEP Configuration **+ DOCKER**

Example Usage of the Integrated Capture Agent for Monitoring

Now we can enable packet mirroring for all (or some) of our traffic:

```
freeswitch@fsnode04> reloadxml
freeswitch@fsnode04> reload mod_sofia

freeswitch@fsnode04> sofia global capture on
+OK Global capture on

freeswitch@fsnode04> sofia global capture off
+OK Global capture off
```

In order to correlate internal/external B2BUA session with different UUIDs we can use a **Custom Header** in our master dialplan:

```
…
        <action application="set"><![CDATA[sip_h_X-CID=<sip:${sip_call_id}]]></action>
…
```

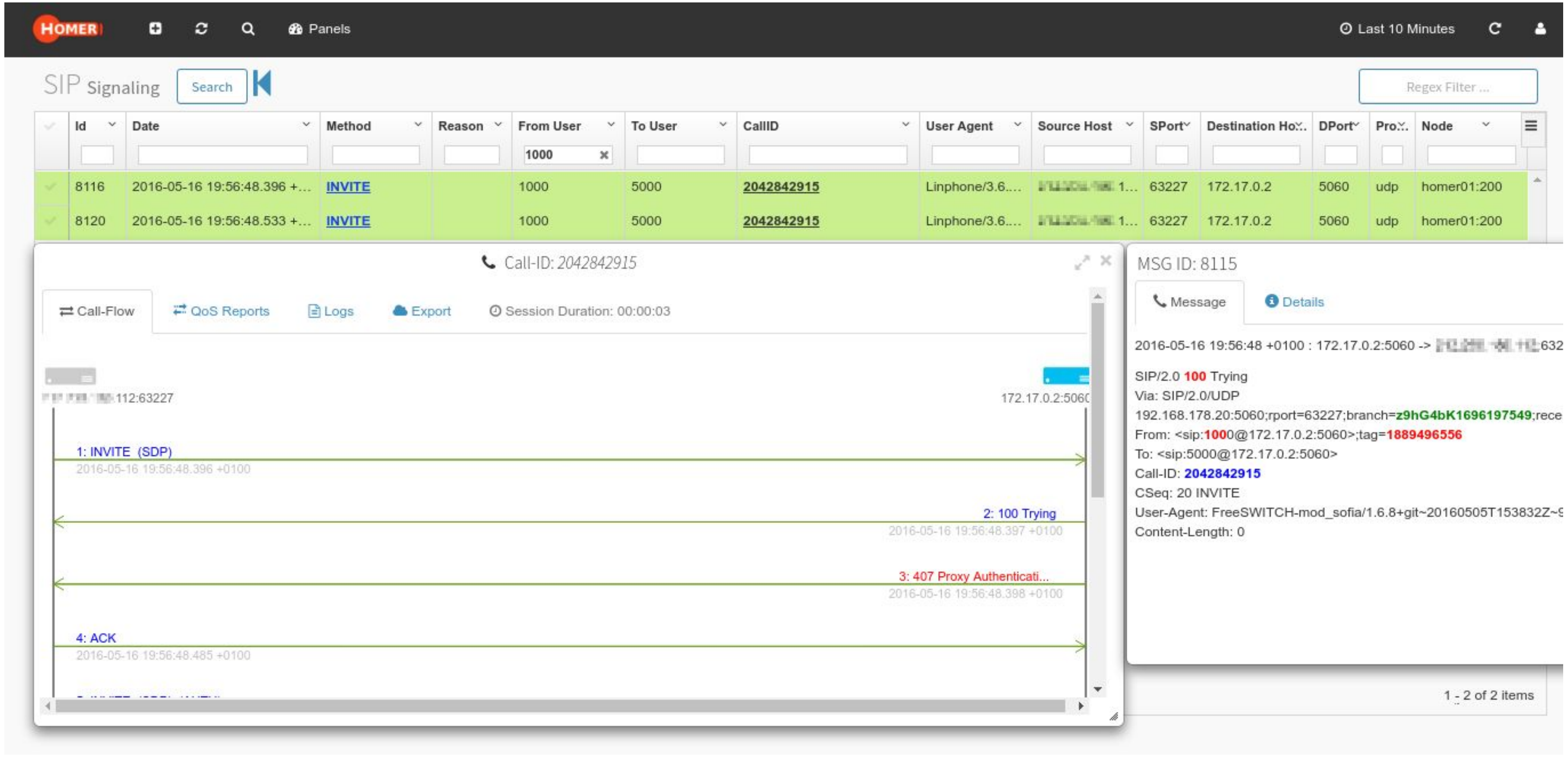
```
INVITE sip:18007654321@1.2.3.4 SIP/2.0
From: "Some Guy" <sip:2132132132@127.0.0.1>;tag=XQKQ322vQF5gK
To: <sip:18007654321@1.2.3.4>
Call-ID: ABC-1234
```

```
INVITE sip:EXT4321@5.6.7.8 SIP/2.0
From: "Anonymous" <sip:anonymous@invalid>;tag=KSJDOKAH678A
To: <sip:EXT4321@5.6.7.8>
Call-ID: DEF-NEW-45678_id
X-CID: ABC-1234
```

FreeSWITCH HEP/EEP Configuration + DOCKER

Example Usage of the Integrated Capture Agent for Monitoring

If you configured everything correctly, you should be ready to search and display your sessions in **Homer**:



Id	Date	Method	Reason	From User	To User	CallID	User Agent	Source Host	SPort	Destination Ho...	DPort	Pro...	Node
				1000									
8116	2016-05-16 19:56:48.396 +...	INVITE		1000	5000	2042842915	Linphone/3.6....	[redacted] 1...	63227	172.17.0.2	5060	udp	homer01:200
8120	2016-05-16 19:56:48.533 +...	INVITE		1000	5000	2042842915	Linphone/3.6....	[redacted] 1...	63227	172.17.0.2	5060	udp	homer01:200

Call-ID: *2042842915*

Call-Flow | QoS Reports | Logs | Export | Session Duration: 00:00:03

```
[redacted].112:63227                                    172.17.0.2:5060
1: INVITE  (SDP)  ->
2016-05-16 19:56:48.396 +0100
                                                  <- 2: 100 Trying
                                        2016-05-16 19:56:48.397 +0100
                                   <- 3: 407 Proxy Authenticati...
                                        2016-05-16 19:56:48.398 +0100
4: ACK  ->
2016-05-16 19:56:48.485 +0100
```

MSG ID: 8115

Message | Details

```
2016-05-16 19:56:48 +0100 : 172.17.0.2:5060 -> [redacted]:632

SIP/2.0 100 Trying
Via: SIP/2.0/UDP
192.168.178.20:5060;rport=63227;branch=z9hG4bK1696197549;rece
From: <sip:1000@172.17.0.2:5060>;tag=1889496556
To: <sip:5000@172.17.0.2:5060>
Call-ID: 2042842915
CSeq: 20 INVITE
User-Agent: FreeSWITCH-mod_sofia/1.6.8+git~20160505T153832Z~9
Content-Length: 0
```

1 - 2 of 2 items

FreeSWITCH HEP/EEP ESL Integration

Example Usage of the External ESL Capture Agent for Monitoring

You want more, *don't you?* *Enter* **HEPIPE-ESL!**

Hepipe-ESL is a nodejs application for harvesting **FreeSWITCH Event Socket** and extracting internal logs, statistics, media reports and much more providing the basics to transform events into arbitrary and correlated **HEP/EEP** Custom Reports sent to **HOMER 5**

```
git clone http://github.com/sipcapture/hepipe.js
cd hepipe.js/esl
npm install
```

Running **Hepipe-ESL** with default settings is as simple as passing two arguments pointing at your Capture server:

```
nodejs hepipe-esl.js -s {homer_ip} -p {homer_port}
```

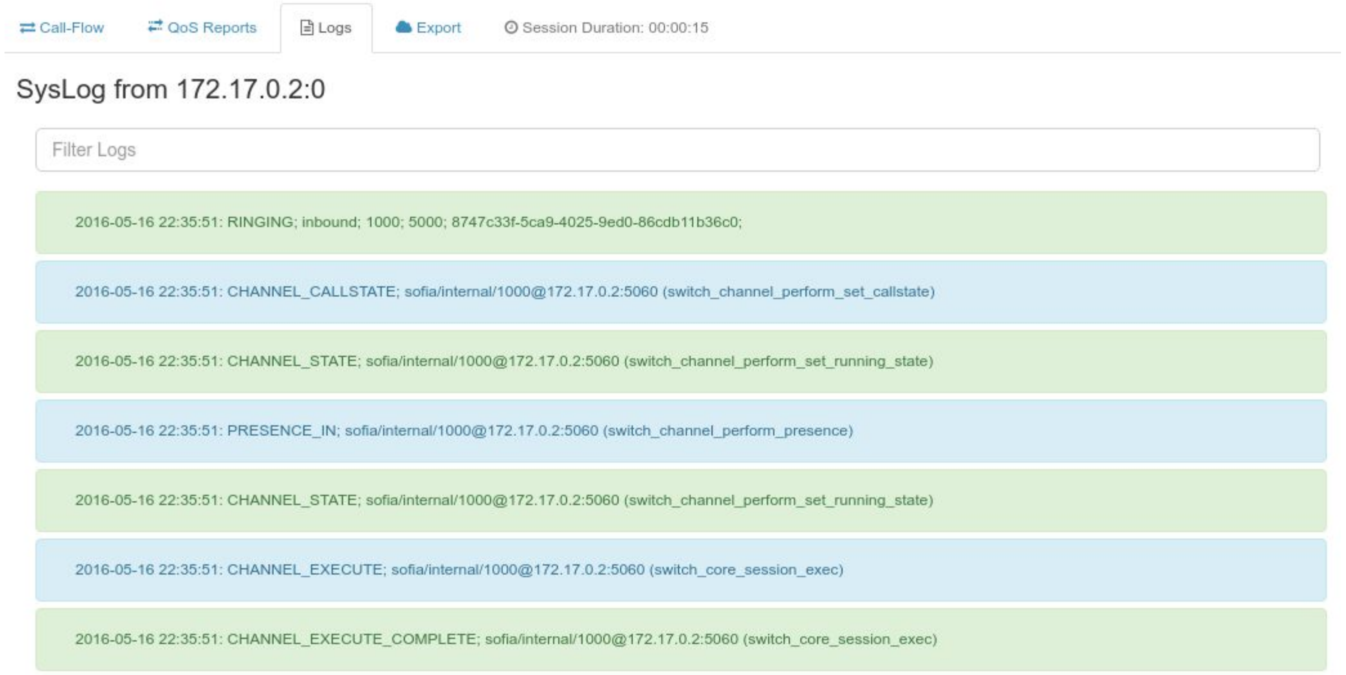
-s	HEP SERVER IP	127.0.0.1
-p	HEP SERVER Port	9060
-es	FS ESL IP	127.0.0.1
-ep	FS ESL Port	8021
-ew	FS ESL Password	ClueCon

FreeSWITCH HEP/EEP ESL Integration

Example Usage of the External ESL Capture Agent for Monitoring

ESL logs are automatically correlated to SIP Sessions by **HEPIPE** and are made available via the **HOMER** "Logs" tab

A QUICK EXAMPLE:

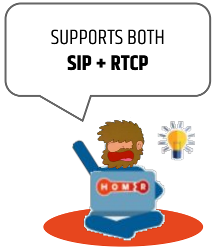


SysLog from 172.17.0.2:0

```
2016-05-16 22:35:51: RINGING; inbound; 1000; 5000; 8747c33f-5ca9-4025-9ed0-86cdb11b36c0;
2016-05-16 22:35:51: CHANNEL_CALLSTATE; sofia/internal/1000@172.17.0.2:5060 (switch_channel_perform_set_callstate)
2016-05-16 22:35:51: CHANNEL_STATE; sofia/internal/1000@172.17.0.2:5060 (switch_channel_perform_set_running_state)
2016-05-16 22:35:51: PRESENCE_IN; sofia/internal/1000@172.17.0.2:5060 (switch_channel_perform_presence)
2016-05-16 22:35:51: CHANNEL_STATE; sofia/internal/1000@172.17.0.2:5060 (switch_channel_perform_set_running_state)
2016-05-16 22:35:51: CHANNEL_EXECUTE; sofia/internal/1000@172.17.0.2:5060 (switch_core_session_exec)
2016-05-16 22:35:51: CHANNEL_EXECUTE_COMPLETE; sofia/internal/1000@172.17.0.2:5060 (switch_core_session_exec)
```

Asterisk + HEP/EEP Configuration **+ DOCKER**

Example Usage of the Integrated Capture Agent for Monitoring



Asterisk + HEP/EEP Configuration **+ DOCKER**

Example Usage of the Integrated Capture Agent for Monitoring

Asterisk 12+ ships with HEP encapsulation support *(res_hep)* and is able to natively mirror its packets to a **SIPCAPTURE** Collector such as **HOMER**. Enabling the HEP/EEP feature is as simple as configuring `/etc/asterisk/hep.conf`

Let's add a Docker container running **Asterisk 13.1** built with **PJSIP** and native **HEP+RTCP** support to our stack:

```
# Create stateful volume
docker create --name asteriskdata --volume /etc/asterisk/ qxip/docker-asterisk-hep  /bin/true

# Start Container
docker run -tid --name asterisk -p 5080:5060 --expose 5060/udp --expose 10000-20000/udp --volumes-from asteriskdata qxip/docker-asterisk-hep

# Attach
docker attach asterisk

# Bash in
Docker exec -ti asterisk /bin/bash
```

The Docker container comes pre-loaded with all **HEP** modules *(res_hep, res_hep_pjsip, res_hep_rtcp)* and can immediately be used:

```
asterisk*CLI> module show like res_hep
Module                         Description                         Use Count  Status      Support Level
res_hep.so                     HEPv3 API                           0          Running          extended
res_hep_pjsip.so               PJSIP HEPv3 Logger                  0          Running          extended
res_hep_rtcp.so                RTCP HEPv3 Logger                   0          Running           unknown
```

Asterisk + HEP/EEP Configuration

Example Usage of the Integrated Capture Agent for Monitoring

Asterisk is up and running - Great!

Enabling the **HEP/EEP** feature globally is as simple as configuring `/etc/asterisk/hep.conf`

```
; res_hep Module configuration for Asterisk
; All settings are currently set in the general section.
[general]
enabled = yes
; Enable/disable forwarding of packets to a
; HEP server. Default is "yes".
capture_address = 10.0.0.1:9060
; The address of the HEP capture server.
capture_password = foo
; If specified, the authorization password for the HEP server. If not specified, no authorization password will be sent.
capture_id = 1234
; A unique integer identifier for this server. This ID will be embedded sent with each packet from this server.
```

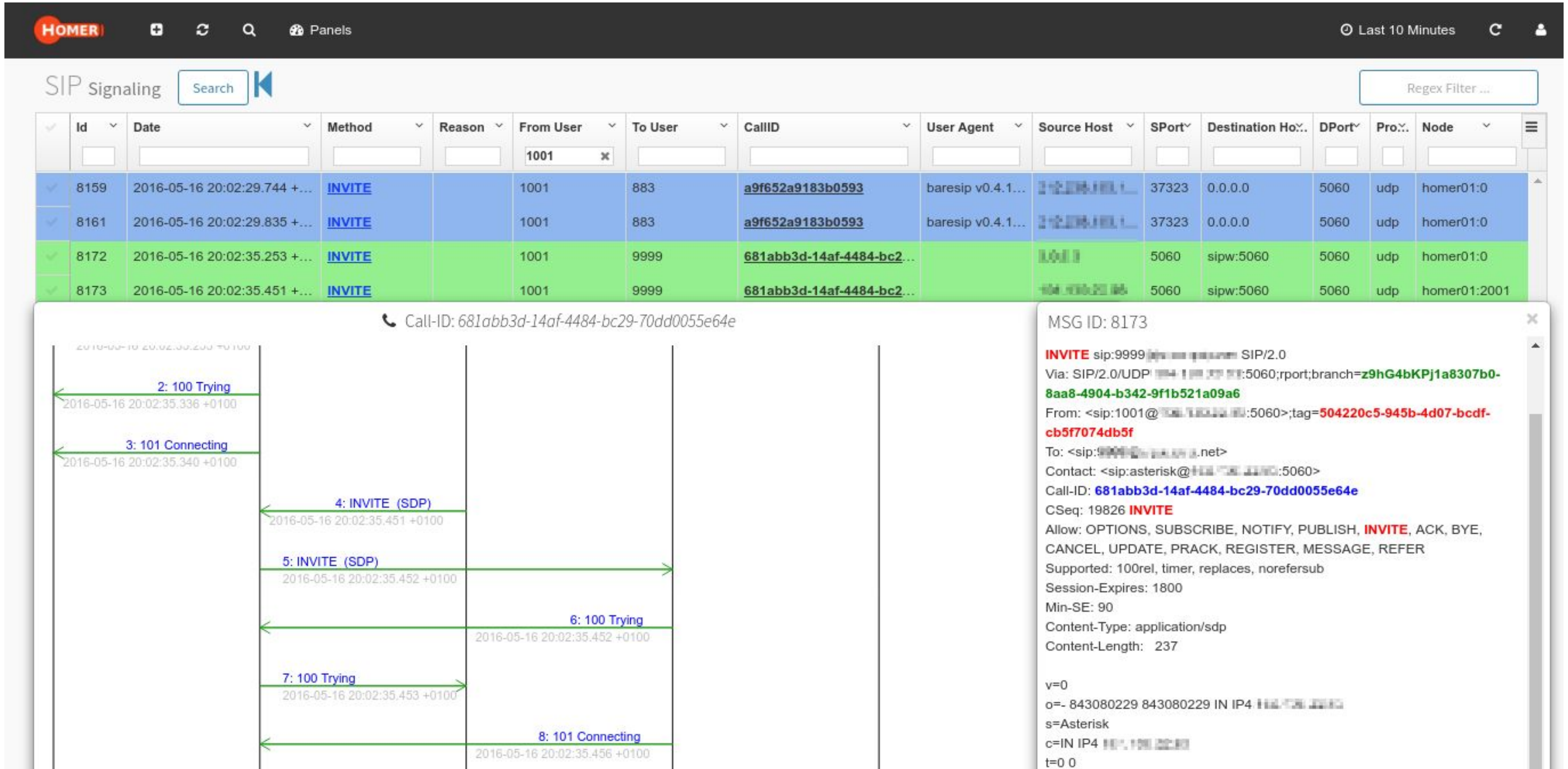
Asterisk 12+ also ships with ***res_hep_rtcp***. The module subscribes to Stasis and receives **RTCP** information back from the message bus, which it encodes into **HEP/EEP** packets and sends to the ***res_hep*** module for transmission. Using this module, Homer users can receive live call quality monitoring for all channels in their **PJSIP** Asterisk 12+ systems.

To enable the functionality, simply load the res_hep_rtcp module alongside the res_hep module (not required for Docker)
Functionality is *only available for chan_pjsip* at this time

Asterisk + HEP/EEP Configuration

Example Usage of the Integrated Capture Agent for Monitoring

If you configured everything correctly, you should be ready to search and display your sessions in **Homer**:



Id	Date	Method	Reason	From User	To User	CallID	User Agent	Source Host	SPort	Destination Ho…	DPort	Pro…	Node
				1001									
8159	2016-05-16 20:02:29.744 +…	INVITE		1001	883	a9f652a9183b0593	baresip v0.4.1…	[redacted]	37323	0.0.0.0	5060	udp	homer01:0
8161	2016-05-16 20:02:29.835 +…	INVITE		1001	883	a9f652a9183b0593	baresip v0.4.1…	[redacted]	37323	0.0.0.0	5060	udp	homer01:0
8172	2016-05-16 20:02:35.253 +…	INVITE		1001	9999	681abb3d-14af-4484-bc2…		[redacted]	5060	sipw:5060	5060	udp	homer01:0
8173	2016-05-16 20:02:35.451 +…	INVITE		1001	9999	681abb3d-14af-4484-bc2…		[redacted]	5060	sipw:5060	5060	udp	homer01:2001

Call-ID: *681abb3d-14af-4484-bc29-70dd0055e64e*

MSG ID: 8173

```
INVITE sip:9999[redacted] SIP/2.0
Via: SIP/2.0/UDP [redacted]:5060;rport;branch=z9hG4bKPj1a8307b0-8aa8-4904-b342-9f1b521a09a6
From: <sip:1001@[redacted]:5060>;tag=504220c5-945b-4d07-bcdf-cb5f7074db5f
To: <sip:[redacted].net>
Contact: <sip:asterisk@[redacted]:5060>
Call-ID: 681abb3d-14af-4484-bc29-70dd0055e64e
CSeq: 19826 INVITE
Allow: OPTIONS, SUBSCRIBE, NOTIFY, PUBLISH, INVITE, ACK, BYE, CANCEL, UPDATE, PRACK, REGISTER, MESSAGE, REFER
Supported: 100rel, timer, replaces, norefersub
Session-Expires: 1800
Min-SE: 90
Content-Type: application/sdp
Content-Length:   237

v=0
o=- 843080229 843080229 IN IP4 [redacted]
s=Asterisk
c=IN IP4 [redacted]
t=0 0
```

Kamailio WSS Monitoring with HEPIPE.js

Example Usage of the External HEP Harvester for WebSocket Log Monitoring

Kamailio WSS Monitoring with HEPIPE.js

Example Usage of the External HEP Harvester for Log Monitoring

Kamailio is great at handling **webSocket** connections, but are you just are great at *troubleshooting* them?
In this simple example, we will configure an external log harvester feeding **Kamailio** logs carrying details about **WSS** socket connections - including the mandatory **SIP** Correlation. First of all, let's create a **custom WSS log** streaming new session details:

```
request_route {

        # per request initial checks
        route(REQINIT);

        if (proto == WS || proto == WSS) {
                 setflag(SRC_WS);
                xlog("L_INFO", "homerwss CID: [$ci], SIP: Method: $rm, CSEQ: $cs, RU: $rU, WSS Request:  RM: $var(wss_rm), RU: $var(wss_ru),
                UAC: $var(wss_uac), Connection: $var(wss_connection), Upgrade: $var(wss_upgrade), Origin: $var(wss_origin),  Host: $var
                (wss_host), Sec_Proto: $var(wss_sec_proto),  Sec-Key: $var(wss_sec_key), WS_VERSION: $var(wss_sec_version)");
        }

        sip_trace();
        setflag(22);
        …
}
```

Next - let's instruct our local **rsyslog.conf** to redirect our new rows (*homerwss*) to a custom file we can use:

```
#### WSS LOG RULE ####
:msg, contains, "homerwss" /var/log/homerwss.log
& ~
```

Kamailio WSS Monitoring with HEPIPE.js

Example Usage of the External HEP Harvester for Log Monitoring

HEPIPE.js is designed to provide a quick and lightweight set of **HEP** functionality to correlate and ship arbitrary user data and the perfect tool for feeding off our brand new custom **WSS** logs. In order to work the node application only needs two key parameters:

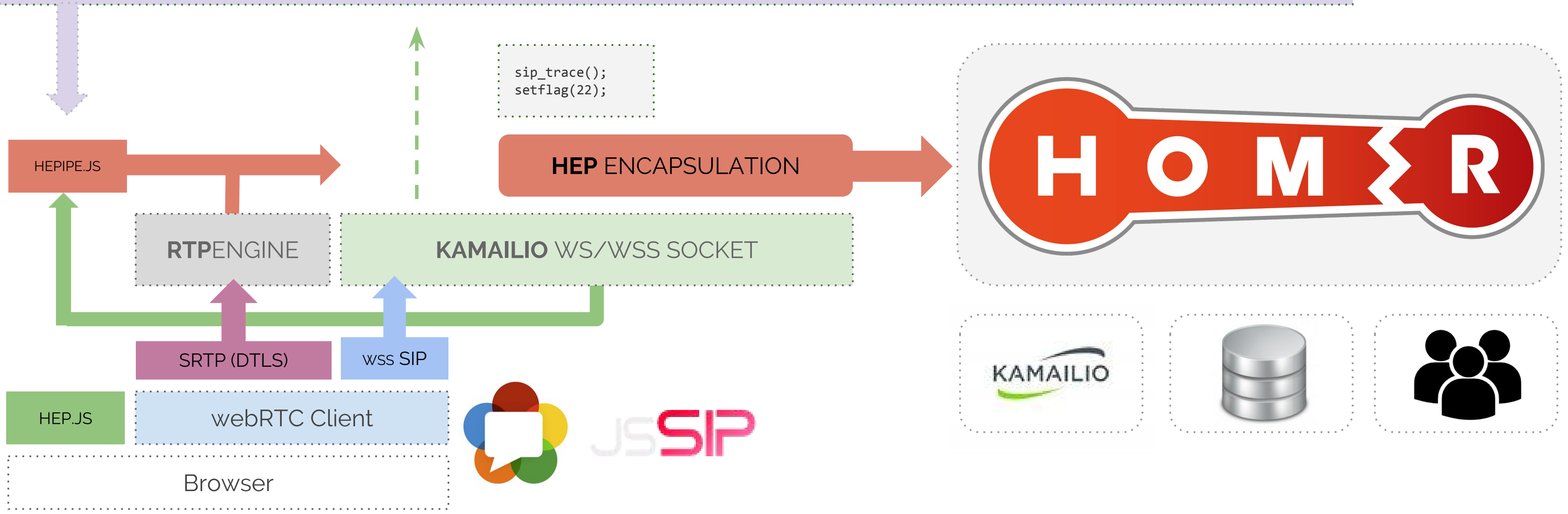
● Path to Log File	*example:*	*/var/log/homerwss.log*
● Regex Filter to extract a Correlation ID	*example:*	*CID: \[(.*)\]*

```
// HEPIPE-JS SETTINGS (please configure)
// ------------------------------------------------------
var config = {
        // Address and Port of your HEP Server
        HEP_SERVER: '10.0.0.1',
        HEP_PORT: 9060,
        // the HEP ID and Authentication for this Agent
        HEP_ID: '2099',
        HEP_AUTH: 'HEProcks',
        // the Logs to monitor
        LOGS: [
                {
                  tag : 'rtc',
                  host : 'WSS',
                   pattern: 'CID: \\[(.*)\\]', // escape backslashes!
                  path : '/var/log/homerwss.log'
                }
              ]
};

module.exports = config;
```

Kamailio WSS Monitoring

http://github.com/sipcapture/wiki

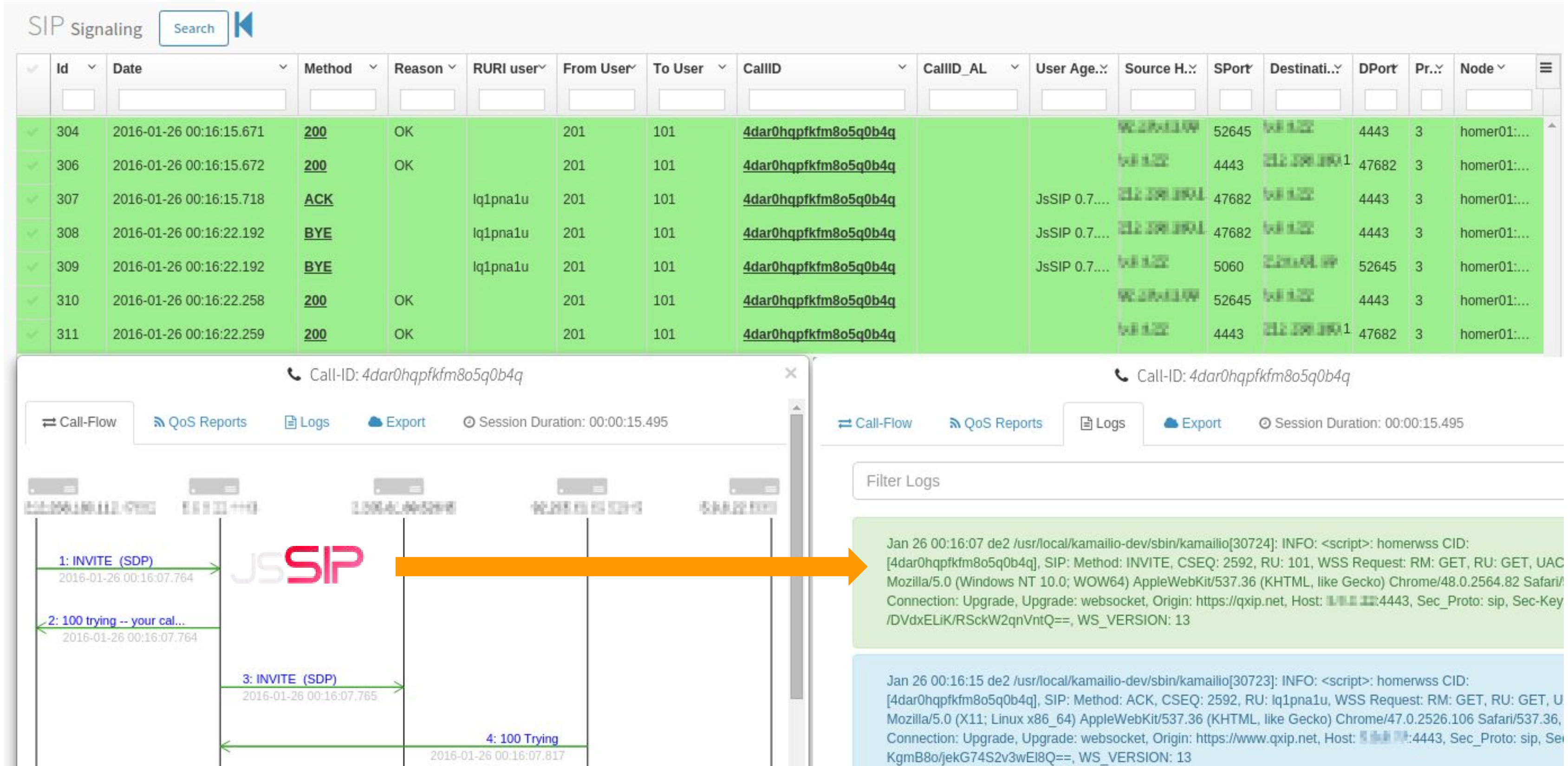
```
if (proto == WS || proto == WSS) {   setflag(SRC_WS);

          xlog("L_INFO", "homerwss CID: [$ci], SIP: Method: $rm, CSEQ: $cs, RU: $rU, WSS Request: RM: $var(wss_rm), RU: $var(wss_ru),
                   UAC: $var(wss_uac), Connection: $var(wss_connection), Upgrade: $var(wss_upgrade), Origin: $var(wss_origin),
                   Host: $var(wss_host), Sec_Proto: $var(wss_sec_proto), Sec-Key: $var(wss_sec_key), WS_VERSION: $var(wss_sec_version)");
}
```



```
sip_trace();
setflag(22);
```

HOMER 5: **WSS** Call Flow

WSS to SIP Call Troubleshooting



SIP Signaling

Id	Date	Method	Reason	RURI user	From User	To User	CallID	CallID_AL	User Age...	Source H...	SPort	Destinati...	DPort	Pr...	Node
304	2016-01-26 00:16:15.671	200	OK		201	101	4dar0hqpfkfm8o5q0b4q			[redacted]	52645	[redacted]	4443	3	homer01:...
306	2016-01-26 00:16:15.672	200	OK		201	101	4dar0hqpfkfm8o5q0b4q			[redacted]	4443	[redacted]	47682	3	homer01:...
307	2016-01-26 00:16:15.718	ACK		lq1pna1u	201	101	4dar0hqpfkfm8o5q0b4q		JsSIP 0.7....	[redacted]	47682	[redacted]	4443	3	homer01:...
308	2016-01-26 00:16:22.192	BYE		lq1pna1u	201	101	4dar0hqpfkfm8o5q0b4q		JsSIP 0.7....	[redacted]	47682	[redacted]	4443	3	homer01:...
309	2016-01-26 00:16:22.192	BYE		lq1pna1u	201	101	4dar0hqpfkfm8o5q0b4q		JsSIP 0.7....	[redacted]	5060	[redacted]	52645	3	homer01:...
310	2016-01-26 00:16:22.258	200	OK		201	101	4dar0hqpfkfm8o5q0b4q			[redacted]	52645	[redacted]	4443	3	homer01:...
311	2016-01-26 00:16:22.259	200	OK		201	101	4dar0hqpfkfm8o5q0b4q			[redacted]	4443	[redacted]	47682	3	homer01:...

Call-ID: *4dar0hqpfkfm8o5q0b4q*

Call-Flow | QoS Reports | Logs | Export | Session Duration: 00:00:15.495

Call-ID: *4dar0hqpfkfm8o5q0b4q*

Call-Flow | QoS Reports | Logs | Export | Session Duration: 00:00:15.495

Filter Logs

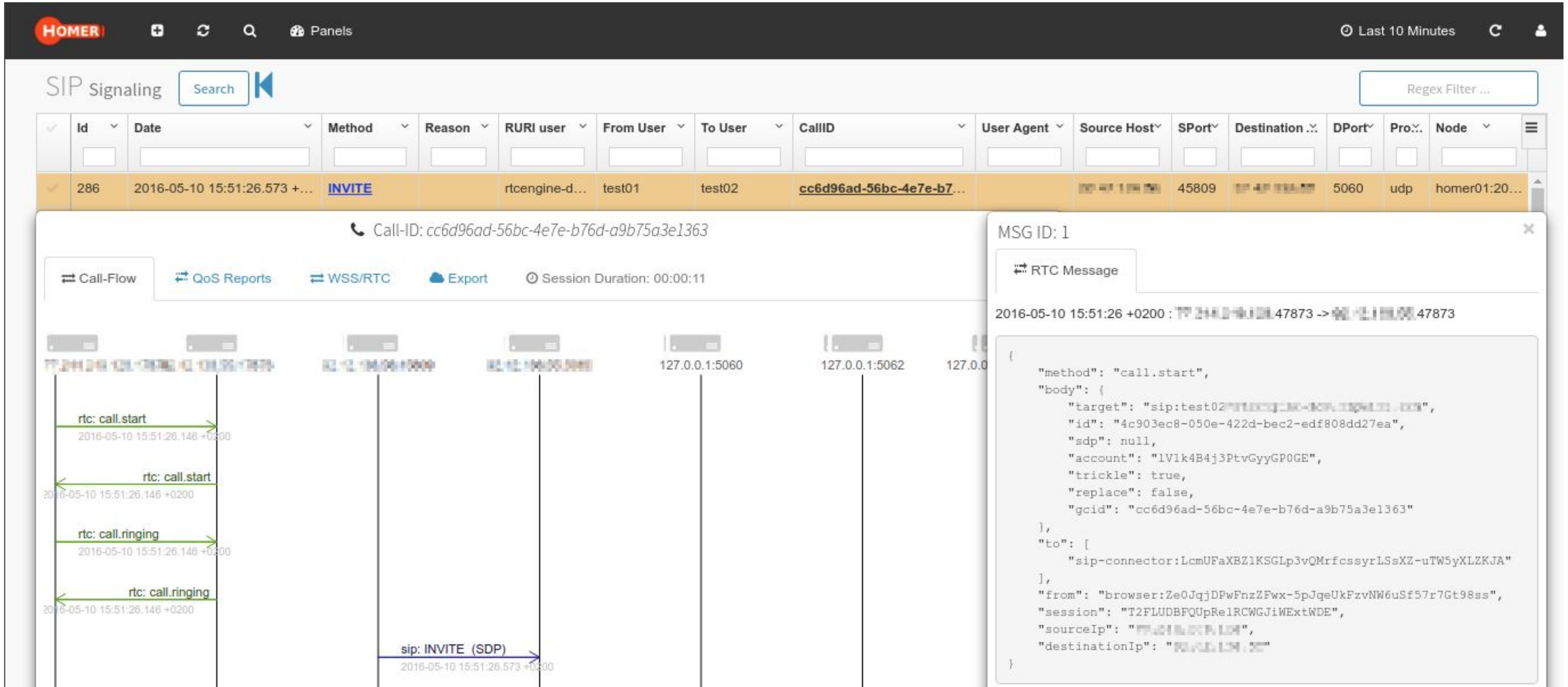
```
Jan 26 00:16:07 de2 /usr/local/kamailio-dev/sbin/kamailio[30724]: INFO: <script>: homerwss CID:
[4dar0hqpfkfm8o5q0b4q], SIP: Method: INVITE, CSEQ: 2592, RU: 101, WSS Request: RM: GET, RU: GET, UAC
Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.82 Safari/
Connection: Upgrade, Upgrade: websocket, Origin: https://qxip.net, Host: [redacted]:4443, Sec_Proto: sip, Sec-Key
/DVdxELiK/RSckW2qnVntQ==, WS_VERSION: 13
```

```
Jan 26 00:16:15 de2 /usr/local/kamailio-dev/sbin/kamailio[30723]: INFO: <script>: homerwss CID:
[4dar0hqpfkfm8o5q0b4q], SIP: Method: ACK, CSEQ: 2592, RU: lq1pna1u, WSS Request: RM: GET, RU: GET, U
Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.106 Safari/537.36,
Connection: Upgrade, Upgrade: websocket, Origin: https://www.qxip.net, Host: [redacted]:4443, Sec_Proto: sip, Se
KgmB8o/jekG74S2v3wEl8Q==, WS_VERSION: 13
```

HOMER 5: **RTC Native** Call Flow

Native webRTC Gateway to SIP Call Troubleshooting via HEP/EEP

A growing number of **RTC** Gateways are being integrated: **RTC:Engine**/Sipwise, **Janus**/Meeteche, **SPiDR**/Genband and more!



Id	Date	Method	Reason	RURI user	From User	To User	CallID	User Agent	Source Host	SPort	Destination ...	DPort	Pro...	Node
286	2016-05-10 15:51:26.573 +...	INVITE		rtcengine-d...	test01	test02	cc6d96ad-56bc-4e7e-b7...		[redacted]	45809	[redacted]	5060	udp	homer01:20...

Call-ID: *cc6d96ad-56bc-4e7e-b76d-a9b75a3e1363*

rtc: call.start 2016-05-10 15:51:26.146 +0200
rtc: call.start 2016-05-10 15:51:26.146 +0200
rtc: call.ringing 2016-05-10 15:51:26.146 +0200
rtc: call.ringing 2016-05-10 15:51:26.146 +0200
sip: INVITE (SDP) 2016-05-10 15:51:26.573 +0200

MSG ID: 1

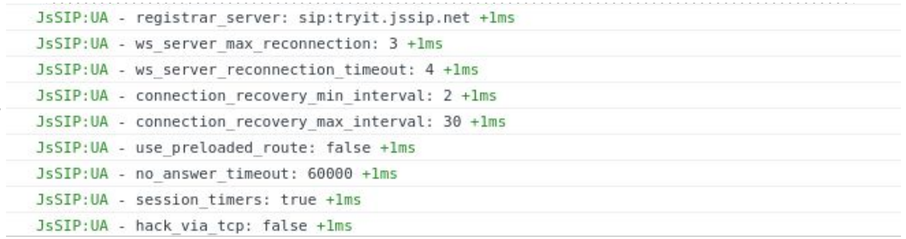
2016-05-10 15:51:26 +0200 : [redacted] 47873 -> [redacted] 47873

```
{
    "method": "call.start",
    "body": {
        "target": "sip:test02[redacted]",
        "id": "4c903ec8-050e-422d-bec2-edf808dd27ea",
        "sdp": null,
        "account": "1V1k4B4j3PtvGyyGP0GE",
        "trickle": true,
        "replace": false,
        "gcid": "cc6d96ad-56bc-4e7e-b76d-a9b75a3e1363"
    },
    "to": [
        "sip-connector:LcmUFaXBZ1KSGLp3vQMrfcssyrLSsXZ-uTW5yXLZKJA"
    ],
    "from": "browser:Ze0JqjDPwFnzZFwx-5pJqeUkFzvNW6uSf57r7Gt98ss",
    "session": "T2FLUDBFQUpRe1RCWGJiWExtWDE",
    "sourceIp": "[redacted]",
    "destinationIp": "[redacted]"
}
```

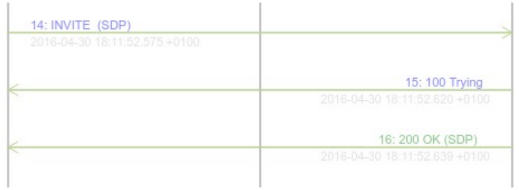
SIPCAPTURE

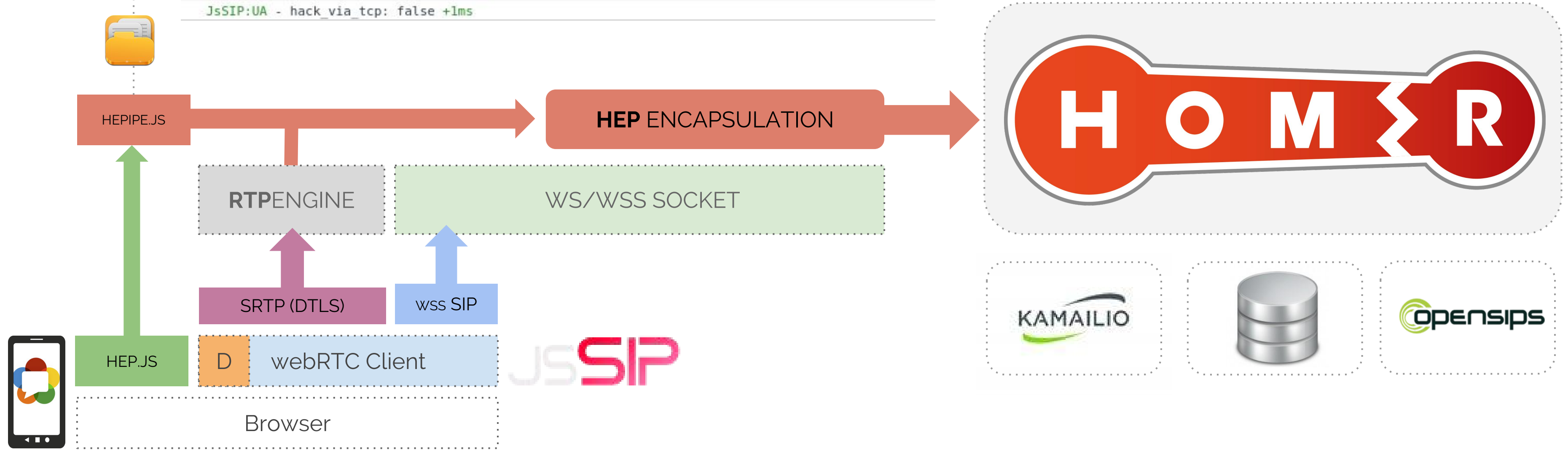
UA Remote Log Monitoring

http://github.com/sipcapture/hepipe-js



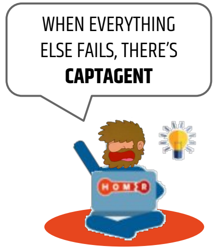
```
JsSIP:UA - registrar_server: sip:tryit.jssip.net +1ms
JsSIP:UA - ws_server_max_reconnection: 3 +1ms
JsSIP:UA - ws_server_reconnection_timeout: 4 +1ms
JsSIP:UA - connection_recovery_min_interval: 2 +1ms
JsSIP:UA - connection_recovery_max_interval: 30 +1ms
JsSIP:UA - use_preloaded_route: false +1ms
JsSIP:UA - no_answer_timeout: 60000 +1ms
JsSIP:UA - session_timers: true +1ms
JsSIP:UA - hack_via_tcp: false +1ms
```





CAPTAGENT 6.1 HEP/EEP Configuration

Example Usage of the Universal Capture Agent for Monitoring



CAPTAGENT 6.1 HEP/EEP Configuration

Example Usage of the Universal Capture Agent for Monitoring

Captagent is a powerful, flexible, completely modular capture agent *framework* ready for virtually any kind of protocol and encapsulation method - past, present *and future*. In this example we will look at a basic standard scenario for passive **SIP** monitoring.

If you are using **Docker** and have access to the *--net=host* option, our **CaptAgent 6** container is ready to use:

```
# Create stateful volume
docker create --name captagentdata --volume /etc/asterisk/ qxip/docker-asterisk-hep  /bin/true

# Start Container
docker run -tid --name captagent --net=host --volumes-from captagentdata qxip/captagent-docker

# Bash in
# docker exec -ti captagent /bin/bash
```

If you are installing on an existing host or system, clone a fresh copy from the main repository:

```
cd /usr/src
git clone https://github.com/sipcapture/captagent.git captagent
cd captagent
./build.sh
./configure
make && make install
```

CAPTAGENT 6.1 HEP/EEP Capture Socket

Example Usage of the Universal Capture Agent for Monitoring

Captagent must be configured before usage. The main configuration file is *captagent.xml*

The first step is to define a **CAPTURE SOCKET** - We will use the default **PCAP** socket and default settings:

- Capture device: *any*
- Capture Portrange: *5060-5091*
- Capture Plan: *sip_capture_plan.cfg*

Let's confirm our configuration in *socket_pcap.xml*

```
<profile name="socketspcap_sip" description="HEP Socket" enable="true" serial="2014010402">
        <settings>
           <param name="dev" value="any"/>
           <param name="promisc" value="true"/>
           <param name="reasm" value="false"/>            // comments here to explain the option?
           <param name="tcpdefrag" value="false"/>        // comments here to explain the option?
           <param name="capture-plan" value="sip_capture_plan.cfg"/>
           <param name="filter">
                      <value>portrange 5060-5091</value>
           </param>
        </settings>
</profile>
```

*NEXT: Let's configure a **Capture Plan** to handle the Procotol*

CAPTAGENT 6.1 HEP/EEP Capture Plans

Example Usage of the Universal Capture Agent for Monitoring

Capture Plans are configurable pipelines handling packets and protocols captured and forwarded by Capture Sockets where additional logic can be defined before sending off the HEP/EEP packet to one or multiple collectors.

In this example we will use the default SIP plan available in: *captureplans/sip_capture_plan.xml*

```
capture[pcap] {
        # Perform checks against source/destination IP/port, message size
        if(msg_check("size", "100")) {
            if(source_ip(“10.0.0.99”)) { drop; }
            # Parse the Message
            if(parse_sip()) {
                # Send using one or multiple profiles defined in transport_hep.xml
                if(!send_hep("hepsocket")) {
                    clog("ERROR", "Error sending HEP!!!!");
                }
            }
        }
        drop;
}
```

*NEXT: Let’s configure a **Transport Socket** to send the HEP/EEP Packet*

CAPTAGENT 6.1 HEP/EEP Transport Socket

Example Usage of the Universal Capture Agent for Monitoring

Transport Sockets are used to deliver the encapsulated packet to a collector.

In this example we will use the default HEP transport module: *transport_hep.xml* and our **HOMER** capture server details:

```
    <profile name="hepsocket" description="Transport HEP" enable="true" serial="201605172204">
             <settings>
                 <param name="version" value="3"/>
                 <param name="capture-host" value="10.0.0.1"/>
                 <param name="capture-port" value="9060"/>
                 <param name="capture-proto" value="udp"/>
                 <param name="capture-id" value="2001"/>
                 <param name="capture-password" value="myhep"/>
                 <param name="payload-compression" value="false"/>
             </settings>
     </profile>
```

*It's **HOMER** Time! Go ahead and **capture** some packets!*

```
# captagent -v
Version: 6.1.0

# captagent -f /usr/local/captagent/etc/captagent/captangent.xml -n
```

CAPTAGENT 6.1 HEP/EEP RTCP + SIP Mirroring

Example Usage of the Universal Capture Agent for Monitoring

Advanced **RTCP** Media Statistics, *You Ask*? ***Pronto!***

Captagent can natively capture and correlate **SIP** and **RTCP** sessions - Just enable the required modules in *captagent.xml*

```
        <load module="transport_hep" register="local"/>
        <load module="database_hash" register="local"/>
        <load module="protocol_sip" register="local"/>
        <load module="protocol_rtcp" register="local"/>
        <load module="socket_pcap" register="local"/>
        <load module="socket_raw" register="local"/>
```

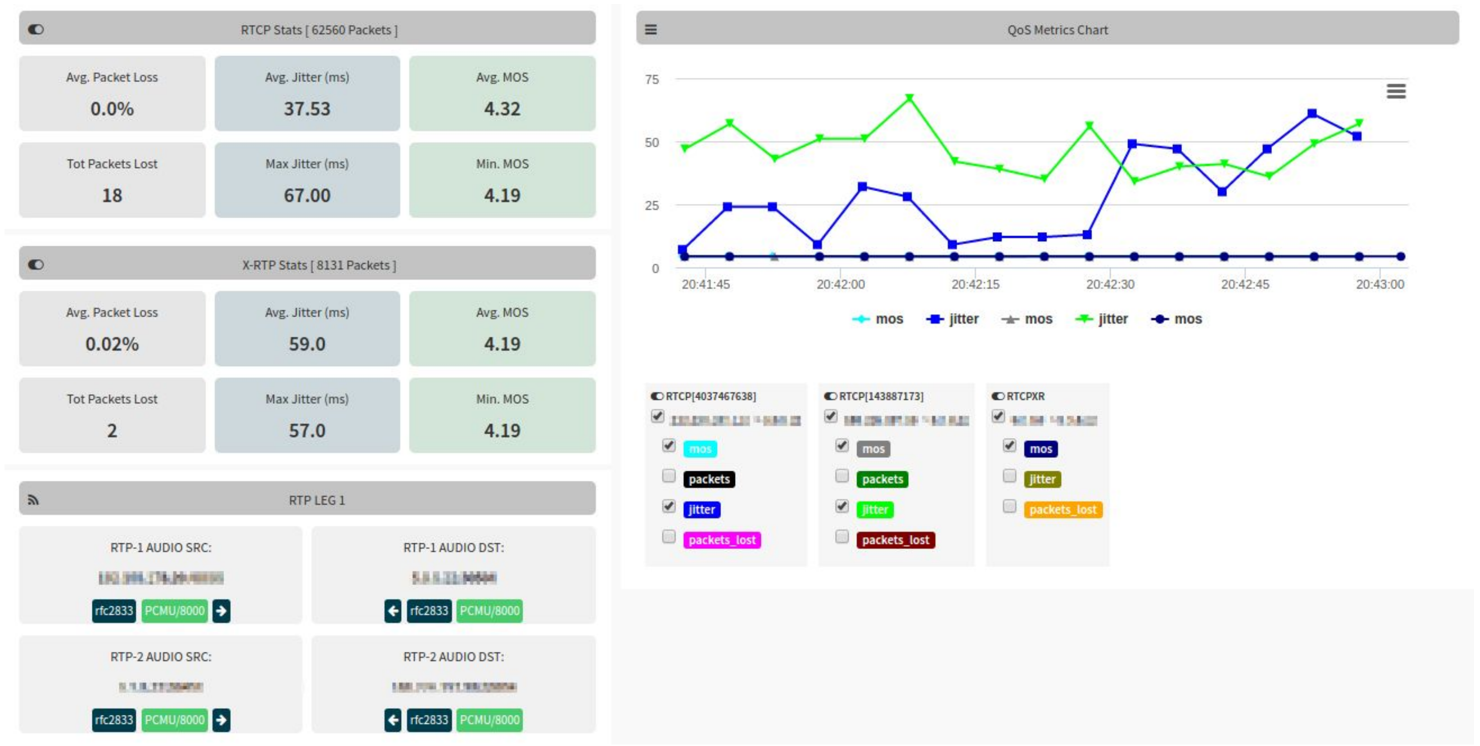
Next, enable the **RTCP Socket** pipeline in *socket_pcap.xml* pointing to your **RTCP** *captureplans/rtcp_capture_plan.xml*

```
    <profile name="socketspcap_rtcp" description="RTCP Socket" enable="true" serial="2014010402">
            <settings>
                 <param name="dev" value="eth0"/>
                 <param name="promisc" value="true"/>
                 <param name="reasm" value="false"/>                          // Enable UDP reassembling
                 <!-- size in MB -->
                 <param name="ring-buffer" value="20"/>                       // Kernel network ring buffer size = RX_RING
                 <!-- for rtp && rtcp < 250 -->
                 <param name="snap-len" value="256"/>                         // for RTP/RTCP packets we should capture maximum 256 bytes
                 <param name="capture-filter" value="rtcp"/>                  // predefined BPF filter - capture only RTCP packets
                 <param name="capture-plan" value="rtcp_capture_plan.cfg"/>
                 <param name="filter">
                     <value>portrange 20000-50000</value>                     // port or portrange filter to use for packet capturing
                 </param>
            </settings>
        </profile>
```

CAPTAGENT 6.1 HEP/EEP RTCP + SIP Mirroring

Example Usage of the Universal Capture Agent for Monitoring

If you configured everything correctly, your HOMER 5 **QoS statistics** will start being populated:



RTCP Stats [62560 Packets]

Avg. Packet Loss	Avg. Jitter (ms)	Avg. MOS
0.0%	37.53	4.32
Tot Packets Lost	**Max Jitter (ms)**	**Min. MOS**
18	67.00	4.19

X-RTP Stats [8131 Packets]

Avg. Packet Loss	Avg. Jitter (ms)	Avg. MOS
0.02%	59.0	4.19
Tot Packets Lost	**Max Jitter (ms)**	**Min. MOS**
2	57.0	4.19

RTP LEG 1

RTP-1 AUDIO SRC:	RTP-1 AUDIO DST:
[redacted]	[redacted]
rfc2833 PCMU/8000	rfc2833 PCMU/8000

RTP-2 AUDIO SRC:	RTP-2 AUDIO DST:
[redacted]	[redacted]
rfc2833 PCMU/8000	rfc2833 PCMU/8000

QoS Metrics Chart

RTCP[4037467638]
[✓] [redacted]
[✓] mos
[ ] packets
[✓] jitter
[ ] packets_lost

RTCP[143887173]
[✓] [redacted]
[✓] mos
[ ] packets
[✓] jitter
[ ] packets_lost

RTCPXR
[✓] [redacted]
[✓] mos
[ ] jitter
[ ] packets_lost

CAPTAGENT+RTPAGENT PRO Modules

Commercial Capture Extensions with Advanced Functionality

RTPAgent is a “privacy-friendly” Analytics and Reporting probe for **HOMER 5** performing wire-speed RTP session and network packet analysis in-transit and in real-time *without storing any data to disk* (unless desired) and delivers granular periodic and final reports with a full stack of dedicated metrics at each interval:

- Source/Destination *IP/PORT/MAC*
- Bytes/Packets Total, Expected
- Packet Loss
- Jitter (min/man/mean)
- RTT Delta/Skew (min/man/mean)
- Codec ID, Clock Rate
- MOS Estimation
- R-Factor Estimation

RTP Reporting frequency can be defined by the integrator or self-adjusted by the probe to send higher number of periodic QoS reports for sessions where suspect quality issues are identified and to automatically reduce the number of reports for those delivering high scores in order to minimize the bandwidth overhead.

RTPAgent is designed to deal with multi-party and multi-codec calls including video sessions and can automatically detect/report a vast number of conditions.

Additional Modules:

★ On-Demand, Filtered Stream Recording to Disk (SIP/RTP/RTCP)
★ Lawful Interception (X1/2/3 ETSI 232)

```
{
"CORRELATION_ID":"56a211936328-fgbtmubkimot",
"RTP_SIP_CALL_ID":"56a211936328-fgbtmubkimot",
"DELTA":19.980,
"JITTER":0.023,
"REPORT_TS":1453461919,
"TL_BYTE":0,
"SKEW":-0.180,
"TOTAL_PK":510,
"EXPECTED_PK":510,
"PACKET_LOSS":0,
"SEQ":0,
"MAX_JITTER":1.892, "MEAN_JITTER":0.126,
"MAX_DELTA":35.547, "MAX_SKEW":-15.615,
"MIN_MOS":4.385, "MEAN_MOS":4.394, "MOS":4.394,
"RFACTOR":92.449, "MIN_RFACTOR":92.013, "MEAN_RFACTOR":92.444,
"SRC_IP":"192.168.178.34", "SRC_PORT":58320, "DST_IP":"192.168.60.70","DST_PORT":
32728,
"SRC_MAC":"00-04-13-29-64-22","DST_MAC":"34-31-C4-38-24-0D",
"CODEC_PT":9,"CLOCK":8000, "CODEC_NAME":"g722", "DIR":1,
"REPORT_NAME": "192.168.178.34:58320", "PARTY":0 ,"TYPE":"PERIODIC"
}
```

SIPGREP 2.x & SNGREP 1.x

Disposable *"on-demand"* console HEP/EEP Agents

CAN'T INSTALL MUCH?
TRY **SIPGREP**

SIPGREP 2.x & SNGREP 1.x

Disposable *"on-demand"* console HEP/EEP Agents

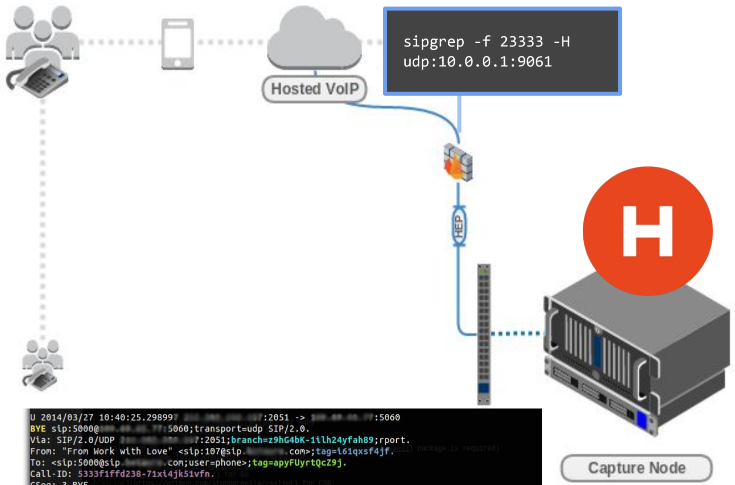
Working and Troubleshooting on Remote system with nothing but a console available? No problem - HEP/EEP has you covered!

sipgrep is SIP console capture and troubleshooting tool able to act as a quick on-demand HEP/EEP capture agent sending packets to a collector to enrich and empower console troubleshooting:

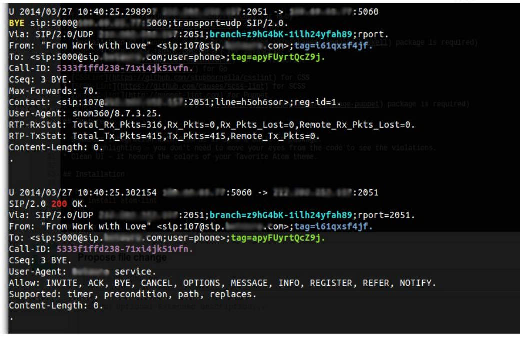
```
sipgrep -f 23333 -H udp:10.0.0.1:9060
```

sngrep 1.x from Irontec/Kaian introduces a HEP/EEP command line option *(-H)* and dedicated settings *(eep.send)* to send capture data in HEP/EEP to Homer and to run headless as a capture agent:

```
sngrep port 5060 -H udp:10.0.0.1:9060 --no-interface -q
```



```
sipgrep -f 23333 -H
udp:10.0.0.1:9061
```



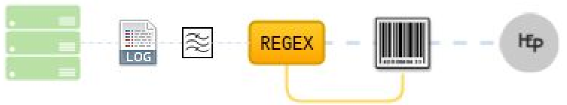
```
U 2014/03/27 10:40:25.29899[redacted]:2051 -> [redacted]:5060
BYE sip:5000@[redacted]5060;transport=udp SIP/2.0.
Via: SIP/2.0/UDP [redacted]:2051;branch=z9hG4bK-1ilh24yfah89;rport.
From: "From Work with Love" <sip:107@sip.[redacted].com>;tag=i61qxsf4jf.
To: <sip:5000@sip.[redacted].com;user=phone>;tag=apyFUyrtQcZ9j.
Call-ID: 5333f1ffd238-71xi4jk51vfn.
CSeq: 3 BYE.
Max-Forwards: 70.
Contact: <sip:107@[redacted]:2051;line=h5oh6sor>;reg-id=1.
User-Agent: snom360/8.7.3.25.
RTP-RxStat: Total_Rx_Pkts=316,Rx_Pkts=0,Rx_Pkts_Lost=0,Remote_Rx_Pkts_Lost=0.
RTP-TxStat: Total_Tx_Pkts=415,Tx_Pkts=415,Remote_Tx_Pkts=0.
Content-Length: 0.
.

U 2014/03/27 10:40:25.302154 [redacted]:5060 -> [redacted]:2051
SIP/2.0 200 OK.
Via: SIP/2.0/UDP [redacted]:2051;branch=z9hG4bK-1ilh24yfah89;rport=2051.
From: "From Work with Love" <sip:107@sip.[redacted].com>;tag=i61qxsf4jf.
To: <sip:5000@sip.[redacted].com;user=phone>;tag=apyFUyrtQcZ9j.
Call-ID: 5333f1ffd238-71xi4jk51vfn.
CSeq: 3 BYE.
User-Agent: [redacted] service.
Allow: INVITE, ACK, BYE, CANCEL, OPTIONS, MESSAGE, INFO, REGISTER, REFER, NOTIFY.
Supported: timer, precondition, path, replaces.
Content-Length: 0.
.
```

HEPIPE.js

Strings Galore!

HEPIPE.js

Strings Galore!



Troubleshooting is not just about network packets - **system logs** will often hold valuable pointers to internal issues not expressed at the protocol level. There are many tools available to forward syslog/rsyslog to notorious collectors but for those looking to build their own voice data collection, we have developed a HEP3 playground utility called **HEPipe**

HEPipe *(pronounced HEP-pipe)* is a NodeJS application designed for monitoring, harvesting and extracting arbitrary data *(from application logs, cdrs, debug lines, syslog, etc)* to a remote *HEP/EEP* capture server such as HOMER or PCAPTURE

This utility can be used to prototype custom HEP/EEP implementations as well as to feed production data into a HEP Collector for real life usage, for instance by using the session Call-ID as correlation parameter for voice system logs

Example Log: NGCP/Kamailio

```
Nov 19 22:05:36 ams2 /usr/sbin/kamailio[1067]: INFO: Sending reply, fs='udp:127.0.0.1:5060' - ID=11876453@127.0.1.1
```

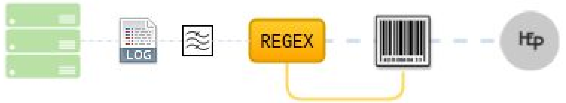
Example HEPIPE.js Config:

Regex Filter:	ID=([^&]\\S*)
Correlation:	11876453@127.0.1.1

```
rcinfo = {
 type: 'HEP',
 version: 3,
 payload_type: '100',
 captureId: '2001',
 capturePass: 'myHep',
  ...
 correlation_id: '11876453@127.0.1.1',
 payload: {
  msg: 'Nov 19 22:05:36 ams2 /usr/sbin/kamailio[1067]: INFO: Sending reply, fs='udp:127.0.0.1:5060' ID=11876453@127.0.1.1'
        }
}
```

HEPIPE.js

Installation & Setup



Setup using Node.JS to mirror and correlate your custom logs:

Step 1: Install HEPIPE from our Github repository on the logging server

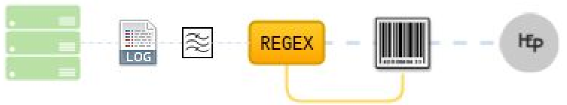
```
git clone http://github.com/sipcapture/hepipe.js
cd hepipe.js
npm install
```

Step 2: Edit the application parameters for HEP and LOGS monitoring in **config.js**
Each LOGS entry defines a log path and a *(regex)* rule to match/extract the proper correlation ID from rows

Example Row:	`Nov 19 22:05:36 ams2 INFO: Sending reply, fs='udp:127.0.0.1:5060' - ID=11876453@127.0.1.1`
Example Regex:	`ID=([^&]\\S*)`
Correlation ID:	`11876453@127.0.1.1`

```
var config = {
        HEP_SERVER: '10.0.0.1',
        HEP_PORT: 9060,
        HEP_ID: '2099',
        HEP_AUTH: 'HEProcks',
        LOGS: [
                {
                   tag : 'NGCP-Logs',
                   host : 'NGCP01',
                   pattern: 'ID=([^&]\\S*)', // escape backslashes!
                   path : '/var/log/syslog.log'
                }
              ]
};
module.exports = config;
```

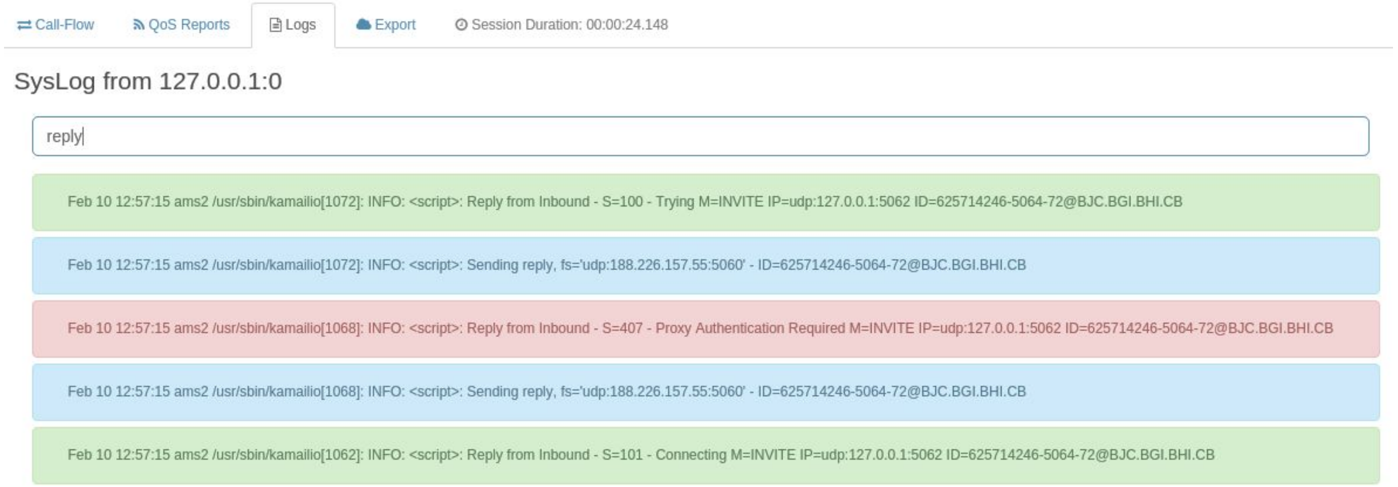
HEPIPE.js



Step 3: *There's not even a step 3* - you are done! It's now time to start sending **HEPIPE** logs to **HOMER**

HEPipe.js logs are automatically correlated to SIP Sessions in **HOMER** and are made available via the "Logs" tab

PRO-TIP: Logs can be filtered directly within the tab using word match or regex rules!

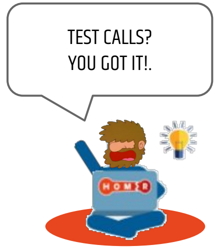


SysLog from 127.0.0.1:0

```
Feb 10 12:57:15 ams2 /usr/sbin/kamailio[1072]: INFO: <script>: Reply from Inbound - S=100 - Trying M=INVITE IP=udp:127.0.0.1:5062 ID=625714246-5064-72@BJC.BGI.BHI.CB
Feb 10 12:57:15 ams2 /usr/sbin/kamailio[1072]: INFO: <script>: Sending reply, fs='udp:188.226.157.55:5060' - ID=625714246-5064-72@BJC.BGI.BHI.CB
Feb 10 12:57:15 ams2 /usr/sbin/kamailio[1068]: INFO: <script>: Reply from Inbound - S=407 - Proxy Authentication Required M=INVITE IP=udp:127.0.0.1:5062 ID=625714246-5064-72@BJC.BGI.BHI.CB
Feb 10 12:57:15 ams2 /usr/sbin/kamailio[1068]: INFO: <script>: Sending reply, fs='udp:188.226.157.55:5060' - ID=625714246-5064-72@BJC.BGI.BHI.CB
Feb 10 12:57:15 ams2 /usr/sbin/kamailio[1062]: INFO: <script>: Reply from Inbound - S=101 - Connecting M=INVITE IP=udp:127.0.0.1:5062 ID=625714246-5064-72@BJC.BGI.BHI.CB
```

BARESIP 0.4.18 LibRE based command-line SIP UA

Example Usage of BareSIP for Call Testing and Quality Probing



BARESIP 0.4.18 LibRE based command-line SIP UA

Example Usage of BareSIP for Call Testing and Quality Probing

How do we test it all? Our favourite FOSS User-Agent is **BareSIP** which features **X-RTP-Stat** and **RTCP-XR** functionality *out of the box!*

Let's fire up our BareSIP Docker container:

```
# Create stateful volume
docker create --name baresipdata --volume /root/.baresip qxip/baresip-docker /bin/true

# Start FS using stateful volume for data
docker run -tid --name baresip -p 5060:5060/udp -p 8080:8080/tcp --expose 10000-20000 --volumes-from baresipdata qxip/baresip-docker

# Bash in
# docker exec -i -t baresip /bin/bash
```

First and foremost, let's enable the QoS reporting options in *`.baresip/config`*

```
# baresip configuration
...
rtp_stats                yes
rtcpxr_stats             yes
rtcpxr_collector         sip:rtcpxr@sip.host.ext:5060
...
```

Before running let's add a SIP account in *`.baresip/accounts`* or just directly in Baresip CLI using the *"R"* command:

```
R
>         sip:username:password@sip.host.ext
```

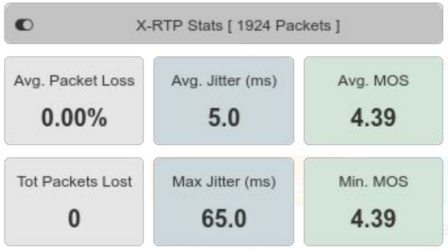
BARESIP 0.4.18 LibRE based command-line SIP UA

Example Usage of BareSIP for Call Testing and Quality Probing

Let's now fire a *test call* and check if we receive the reports - We can use the standard **CLI** or the BareSIP **HTTP API** on port **8080**

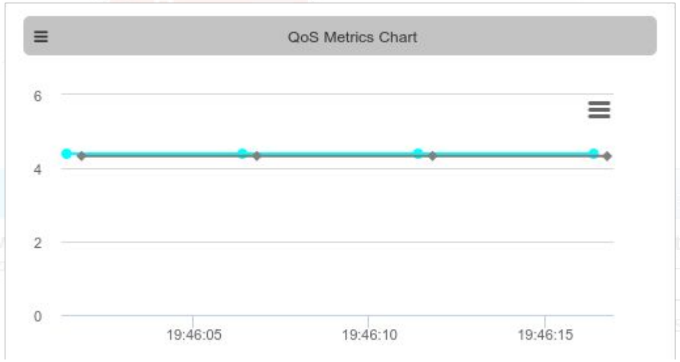
```
baresip is ready.
1001@172: {0/UDP/v4} 200 OK () [1 binding]
All 1 useragent registered successfully! (170 ms)
call: connecting to 'sip:500@172.17.0.3:5080'..
1001@172.17.0.3: Call established: sip:500@172.17.0.3:5080
sip:1001@172.17.0.3:5080: Call with sip:500@172.17.0.3:5080 terminated (duration: 8 secs)
audio           Transmit:     Receive:
packets:            402          298
avg. bitrate:      64.0         48.0  (kbit/s)
errors:               0            0
pkt.report:         221          192
lost:                 0            0
jitter:             7.6          0.1  (ms)
```

Did it work? Open the session in **HOMER 5** and check if the *"QoS Reports"* tabs



X-RTP Stats [1924 Packets]

Avg. Packet Loss	Avg. Jitter (ms)	Avg. MOS
0.00%	5.0	4.39
Tot Packets Lost	Max Jitter (ms)	Min. MOS
0	65.0	4.39

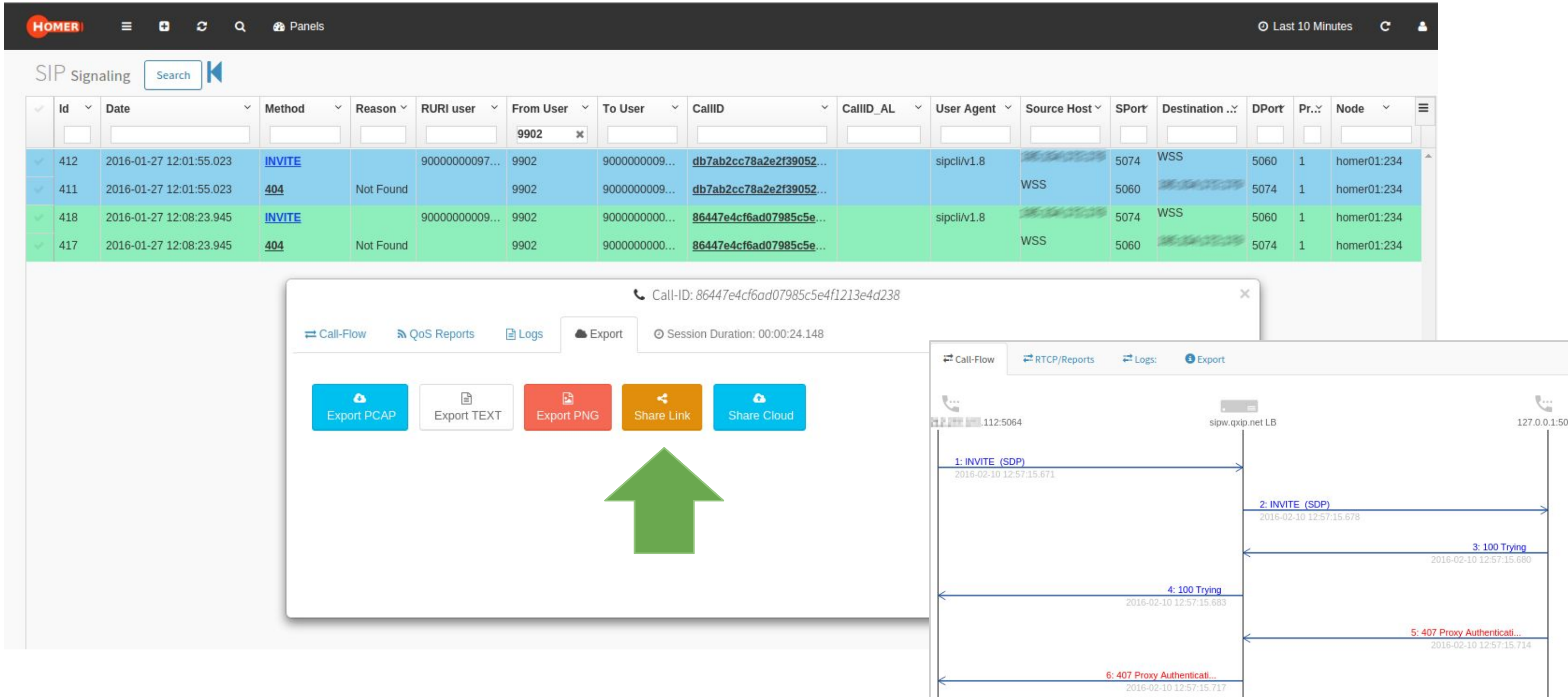


SHARING IS

CUSTOMER CARING

Sharing to **Internal Users & Collaborators**

For trusted entities, **HOMER** provides built-in *“Share Link”* functionality via a secluded part of its web application



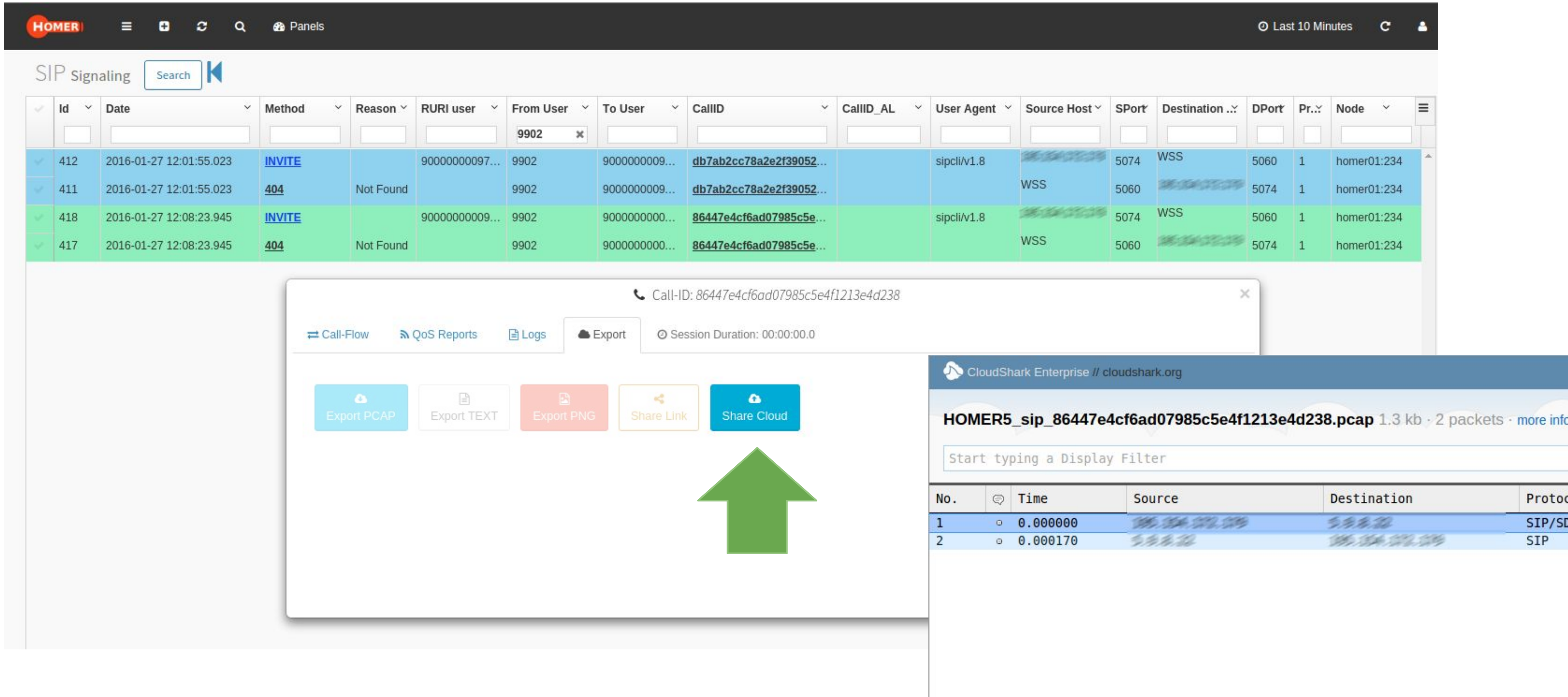
Id	Date	Method	Reason	RURI user	From User	To User	CallID	CallID_AL	User Agent	Source Host	SPort	Destination ...	DPort	Pr...	Node
					9902										
412	2016-01-27 12:01:55.023	INVITE		90000000097...	9902	9000000009...	db7ab2cc78a2e2f39052...		sipcli/v1.8	[redacted]	5074	WSS	5060	1	homer01:234
411	2016-01-27 12:01:55.023	404	Not Found		9902	9000000009...	db7ab2cc78a2e2f39052...			WSS	5060	[redacted]	5074	1	homer01:234
418	2016-01-27 12:08:23.945	INVITE		90000000009...	9902	9000000000...	86447e4cf6ad07985c5e...		sipcli/v1.8	[redacted]	5074	WSS	5060	1	homer01:234
417	2016-01-27 12:08:23.945	404	Not Found		9902	9000000000...	86447e4cf6ad07985c5e...			WSS	5060	[redacted]	5074	1	homer01:234

Call-ID: 86447e4cf6ad07985c5e4f1213e4d238

1: INVITE (SDP) 2016-02-10 12:57:15.671
2: INVITE (SDP) 2016-02-10 12:57:15.678
3: 100 Trying 2016-02-10 12:57:15.680
4: 100 Trying 2016-02-10 12:57:15.683
5: 407 Proxy Authenticati... 2016-02-10 12:57:15.714
6: 407 Proxy Authenticati... 2016-02-10 12:57:15.717

Sharing to **External Parties** and **Partners**

For untrusted entities, **HOMER** provides built-in integration with external applications such as **CloudShark** via “*Share Cloud”*



Id	Date	Method	Reason	RURI user	From User	To User	CallID	CallID_AL	User Agent	Source Host	SPort	Destination ..	DPort	Pr..	Node
					9902										
412	2016-01-27 12:01:55.023	INVITE		90000000097...	9902	9000000009...	db7ab2cc78a2e2f39052...		sipcli/v1.8	[redacted]	5074	WSS	5060	1	homer01:234
411	2016-01-27 12:01:55.023	404	Not Found		9902	9000000009...	db7ab2cc78a2e2f39052...			WSS	5060	[redacted]	5074	1	homer01:234
418	2016-01-27 12:08:23.945	INVITE		90000000009...	9902	9000000000...	86447e4cf6ad07985c5e...		sipcli/v1.8	[redacted]	5074	WSS	5060	1	homer01:234
417	2016-01-27 12:08:23.945	404	Not Found		9902	9000000000...	86447e4cf6ad07985c5e...			WSS	5060	[redacted]	5074	1	homer01:234

Call-ID: *86447e4cf6ad07985c5e4f1213e4d238*

HOMER5_sip_86447e4cf6ad07985c5e4f1213e4d238.pcap 1.3 kb · 2 packets · more info

Start typing a Display Filter

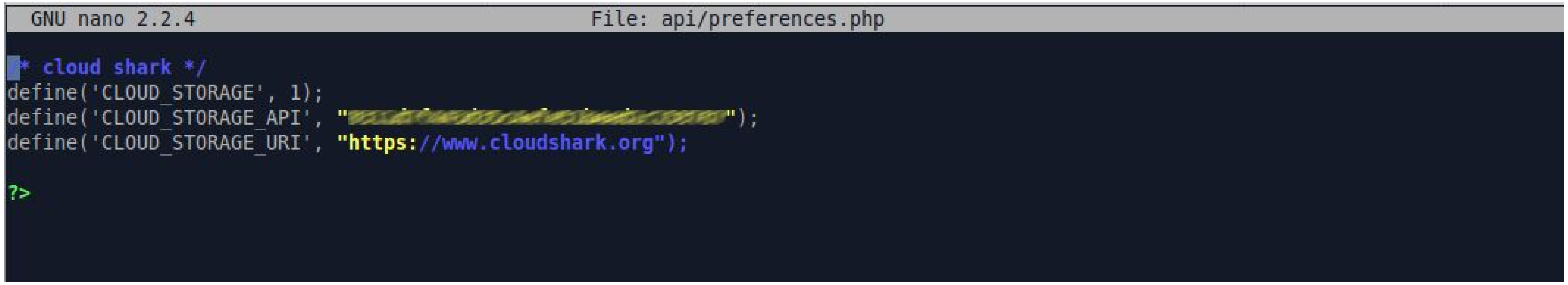
No.		Time	Source	Destination	Protoc
1	○	0.000000	[redacted]	[redacted]	SIP/SD
2	○	0.000170	[redacted]	[redacted]	SIP

CLOUD SHARE configuration

Configure HOMER to Export to CloudShark

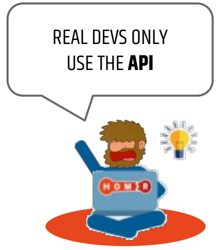
Edit your **HOMER API** Preferences (api/preferences.php):

● **CLOUD_STORAGE** :	Enable Cloud functionality (1)	
● **CLOUD_STORAGE_API** :	Configure using your CloudShark API Key	(upload rights required)
● **CLOUD_STORAGE_URI** :	Configure to point at your Cloudshark URI	(https://www.cloudshark.org)



```
  GNU nano 2.2.4                                    File: api/preferences.php

/* cloud shark */
define('CLOUD_STORAGE', 1);
define('CLOUD_STORAGE_API', "[redacted]");
define('CLOUD_STORAGE_URI', "https://www.cloudshark.org");

?>
```

API
REST INTERFACE


REAL DEVS ONLY USE THE **API**

HOMER 5

API Integration

Homer 5 is 100% based on API functions to provide its features - the same functions used by the UI are available to users and devs to integrate HOMER results and functionality in 3rd party platforms, scripts and monitoring systems.

The **HOMER API** functions are documented within the project itself and being updated as development progresses.

The APIDOC folder is available here: https://github.com/sipcapture/homer-api/tree/master/apidoc

TIP: The best approach towards learning the API is to "spy" on the browser console and network transactions while using the User-Interface features and replicating them by using CURL or other utilities to develop new patterns

Example Integration: SNMP

An example API integration to provide SNMP bridge to Homer internal metrics is available on our repository:

https://github.com/sipcapture/homer-snmp

HOMER 5

Wiki Documentation and Examples

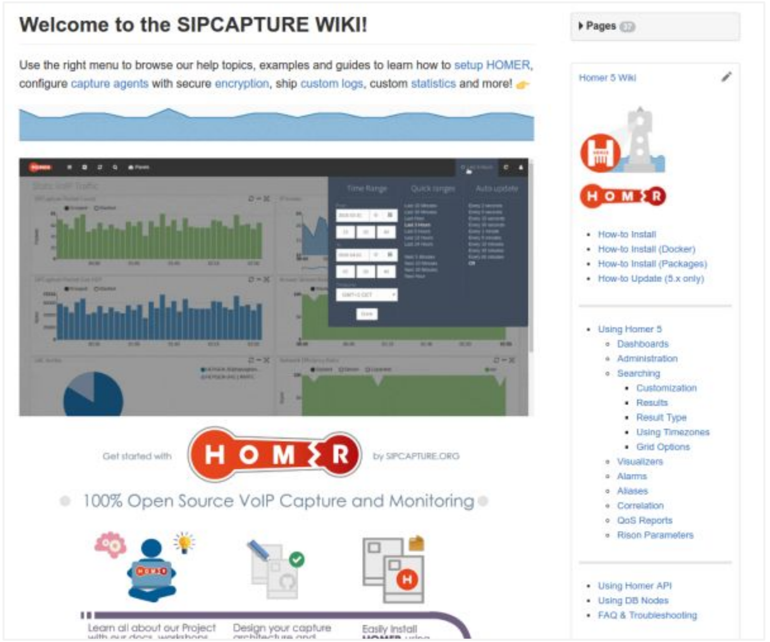
Homer 5 is documented using our **Github Wiki** where all guides, details, example and how-tos are made available. Dive in to get started (or refreshed) with all the available topics updated on a daily basis including:

- ★ How to Install and Update Homer
- ★ How to get started with the User-Interface
- ★ How to customize Panels and Widgets
- ★ How to manage Users and Aliases
- ★ How to configure HEP Capture Agents
- ★ How to configure HEP Custom Agents
- ★ How to correlate Sessions and Reports
- ★ How to make your own Statistics and Widgets

. . . . . *and much more !*

"Just HEP Yourself … "

https://github.com/sipcapture/homer/wiki/



Welcome to the SIPCAPTURE WIKI!

Use the right menu to browse our help topics, examples and guides to learn how to setup HOMER, configure capture agents with secure encryption, ship custom logs, custom statistics and more!

Get started with HOMER by SIPCAPTURE.ORG

100% Open Source VoIP Capture and Monitoring

Pages

Homer 5 Wiki

- How-to Install
- How-to Install (Docker)
- How-to Install (Packages)
- How-to Update (5.x only)

- Using Homer 5
  - Dashboards
  - Administration
  - Searching
    - Customization
    - Results
    - Result Type
    - Using Timezones
    - Grid Options
  - Visualizers
  - Alarms
  - Aliases
  - Correlation
  - QoS Reports
  - Rison Parameters

- Using Homer API
- Using DB Nodes
- FAQ & Troubleshooting

Q & A

Ask us almost Anything

(... 3, 2, 1, MySQL . . .)

LOVE HOMER?
DON't FORGET TO
GET ME A **BEER**
OR A **DONATION**

"That's all Folks!"

Time's UP! Want to go further? "HEP" Yourself!

SIPCAPTURE @GITHUB	http://sipcapture.org + http://sipcapture.io
HOMER @GITHUB	http://github.com/sipcapture/homer
CAPTAGENT @GITHUB	http://github.com/sipcapture/captagent
HEPIPE.JS @GITHUB	http://github.com/sipcapture/hepipe.js
MAILING-LIST @USERS	https://groups.google.com/forum/#!forum/homer-discuss