www.kamailio.org

Kamailio

API Based SIP Routing

Daniel-Constantin Mierla
www.asipto.com
@miconda

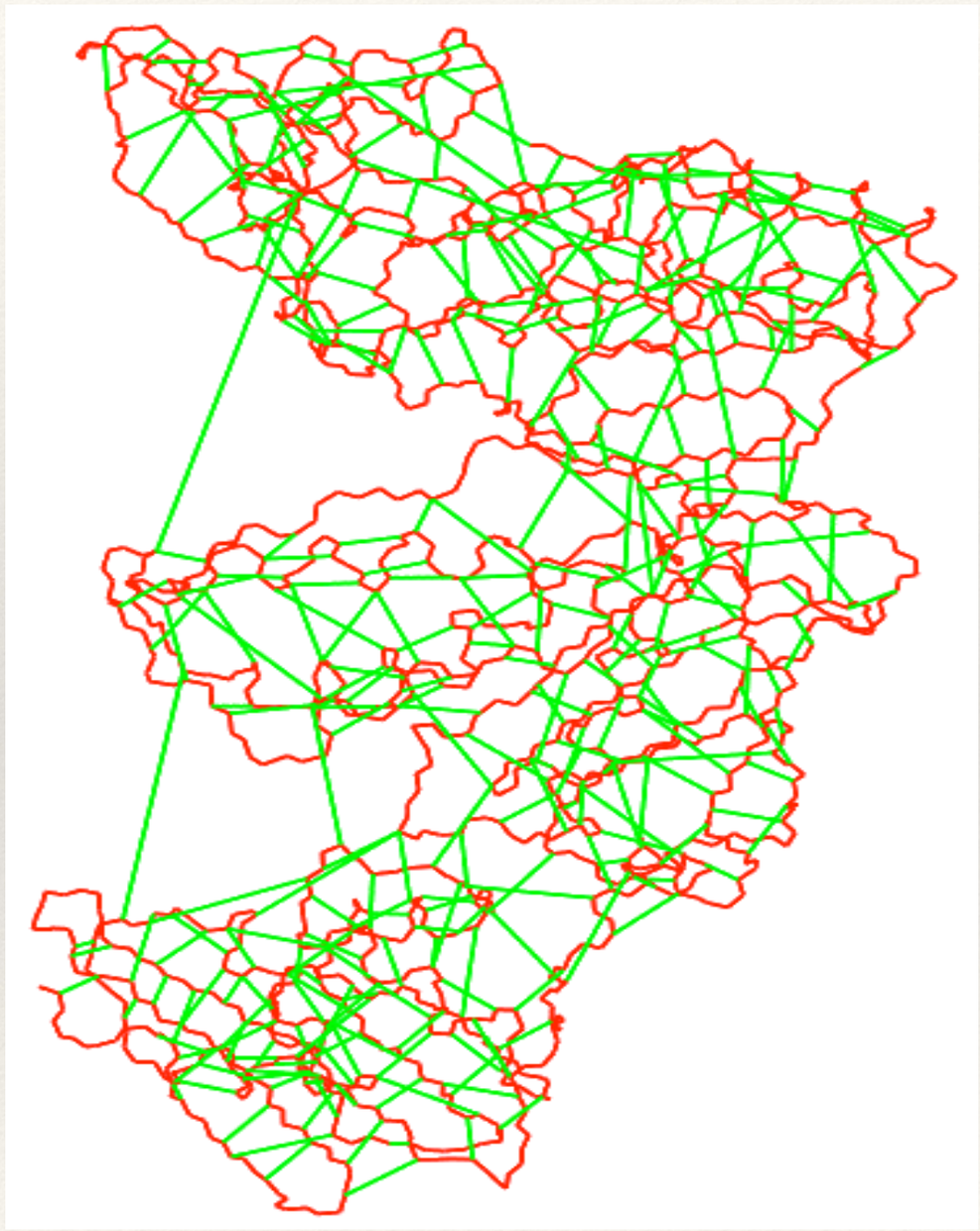
About Kamailio

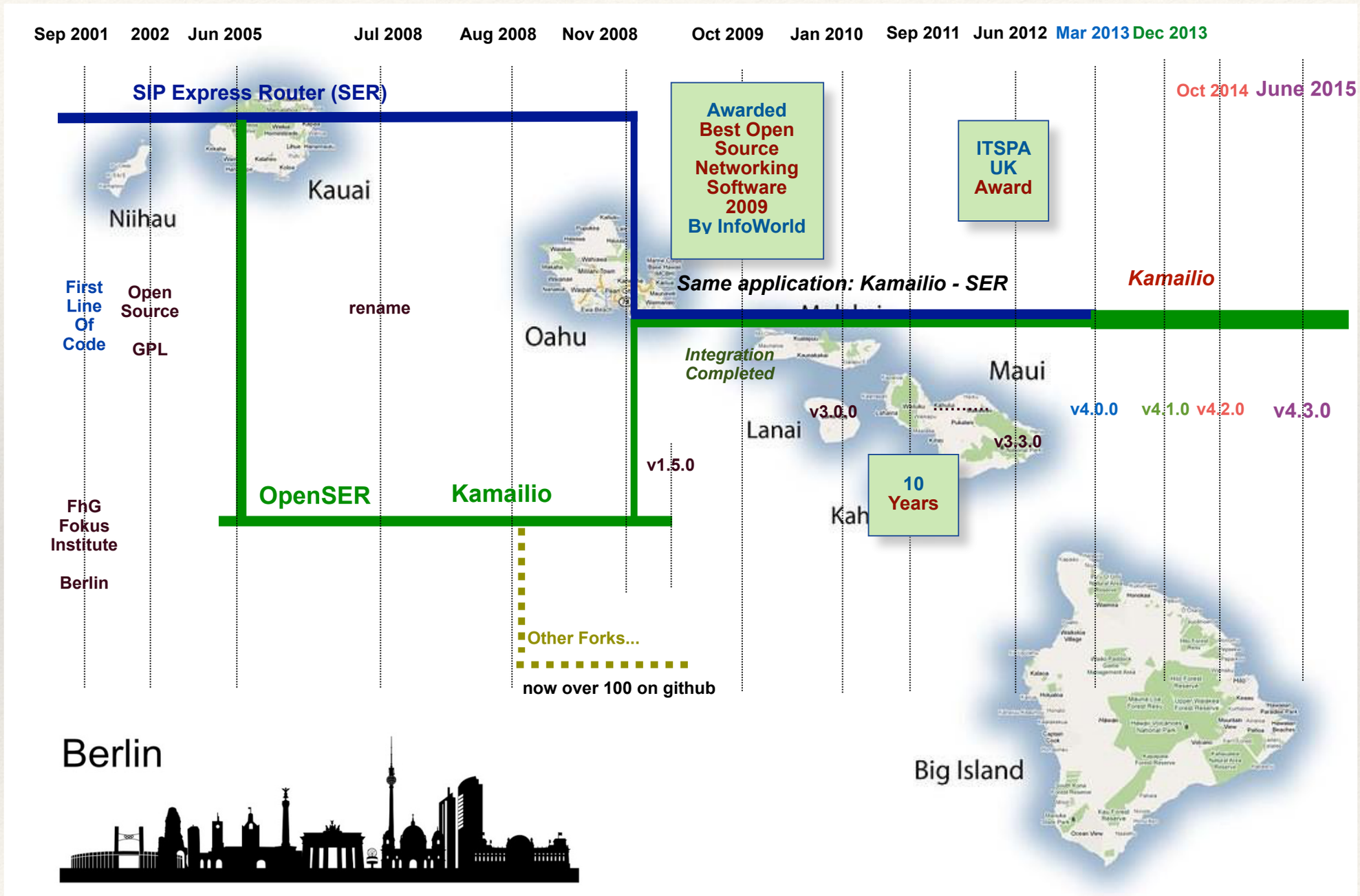
bits about the project

Ground Zero

- SIP signalling routing
  - fast
  - reliable
  - flexible

- In other words
  - not initiating calls
  - not answering calls
  - no audio-video processing




Sep 2001
2002
Jun 2005
Jul 2008
Aug 2008
Nov 2008
Oct 2009
Jan 2010
Sep 2011
Jun 2012
Mar 2013
Dec 2013
Oct 2014
June 2015
SIP Express Router (SER)
Awarded Best Open Source Networking Software 2009 By InfoWorld
ITSPA UK Award
Niihau
Kauai
Oahu
Maui
Lanai
Kah
Big Island
First Line Of Code
Open Source
GPL
rename
Same application: Kamailio - SER
Kamailio
Integration Completed
v3.0.0
v3.3.0
v4.0.0
v4.1.0
v4.2.0
v4.3.0
10 Years
v1.5.0
OpenSER
Kamailio
FhG Fokus Institute
Berlin
Other Forks...
now over 100 on github
Berlin

KAMAILIO WORLD
CONFERENCE & EXHIBITION
BERLIN, GERMANY, MAY 27-29, 2015

http://www.kamailioworld.com

YouTube Kamailio World Channel
https://www.youtube.com/channel/UCElq4JNTPd7bs2vbfAAYVJA

Kamailio World 2016 - Planning a Special Edition

Kamailio Project

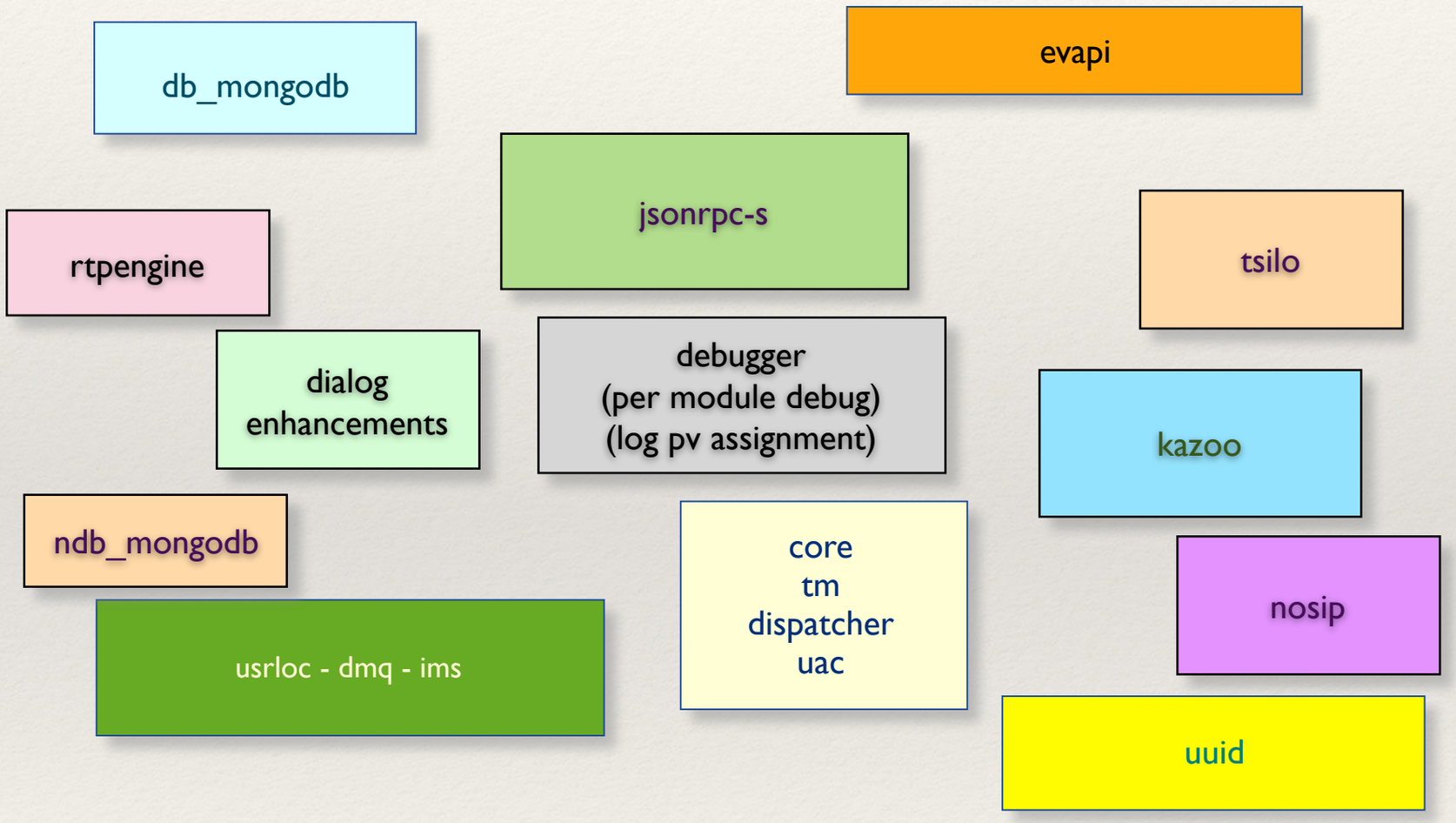
15 YEARS OF DEVELOPMENT

2001-2016

from SER to Kamailio

www.kamailioworld.com

Highlights 2014



http://www.kamailio.org/wiki/features/new-in-4.1.x
http://www.kamailio.org/wiki/features/new-in-4.2.x
http://www.kamailio.org/wiki/features/new-in-devel

Highlights 2015

webrtc

rtjson

statsd

new performat memory manger

janson

tcpops

jansonrpc-c

erlang

dialplan enhancements

ims

auth_xkeys

geoip2

core
pv
rr
uac

ndb_redis
ndb_mongodb

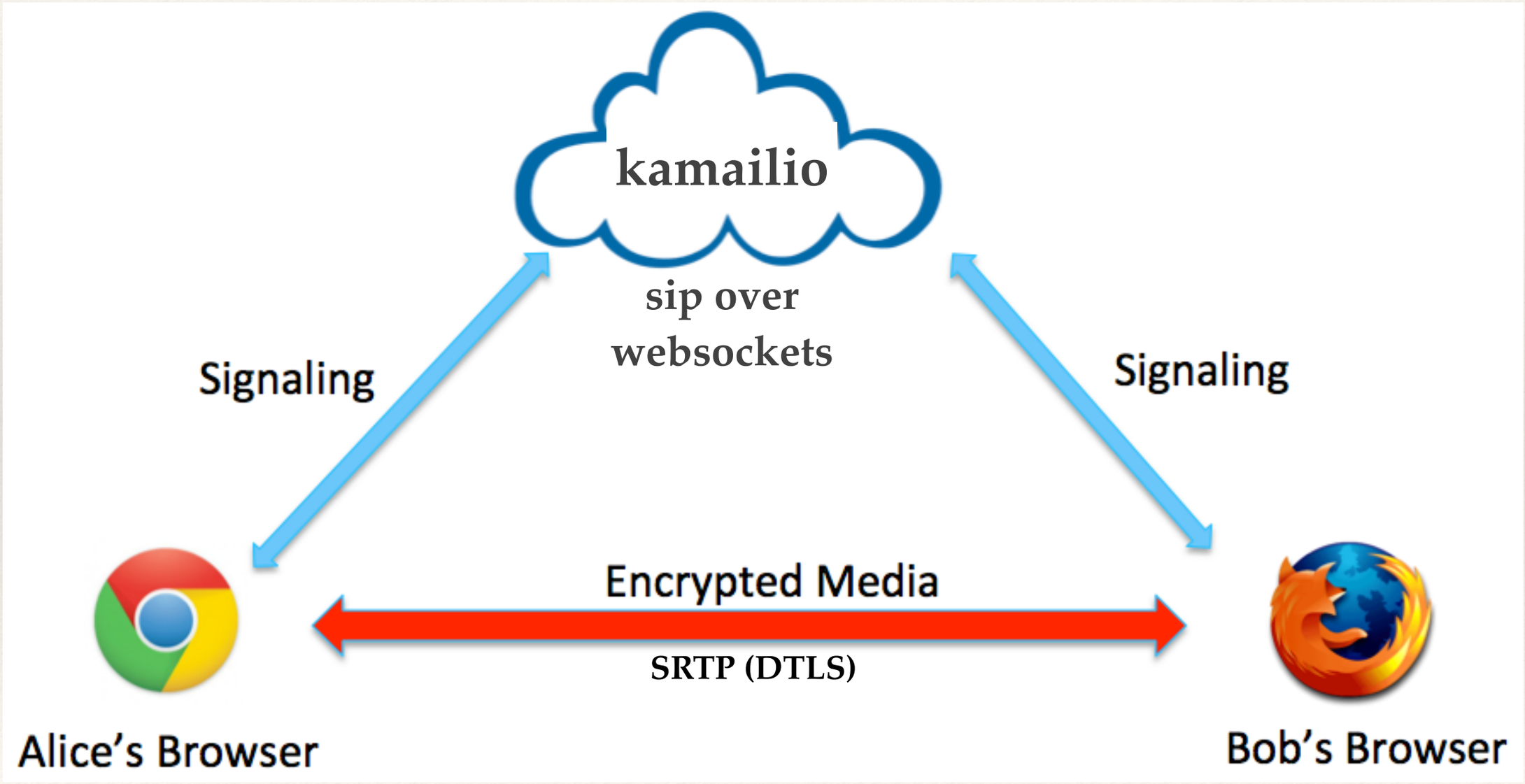
usrloc - dmq - registrar

jsonrpc-s fifo

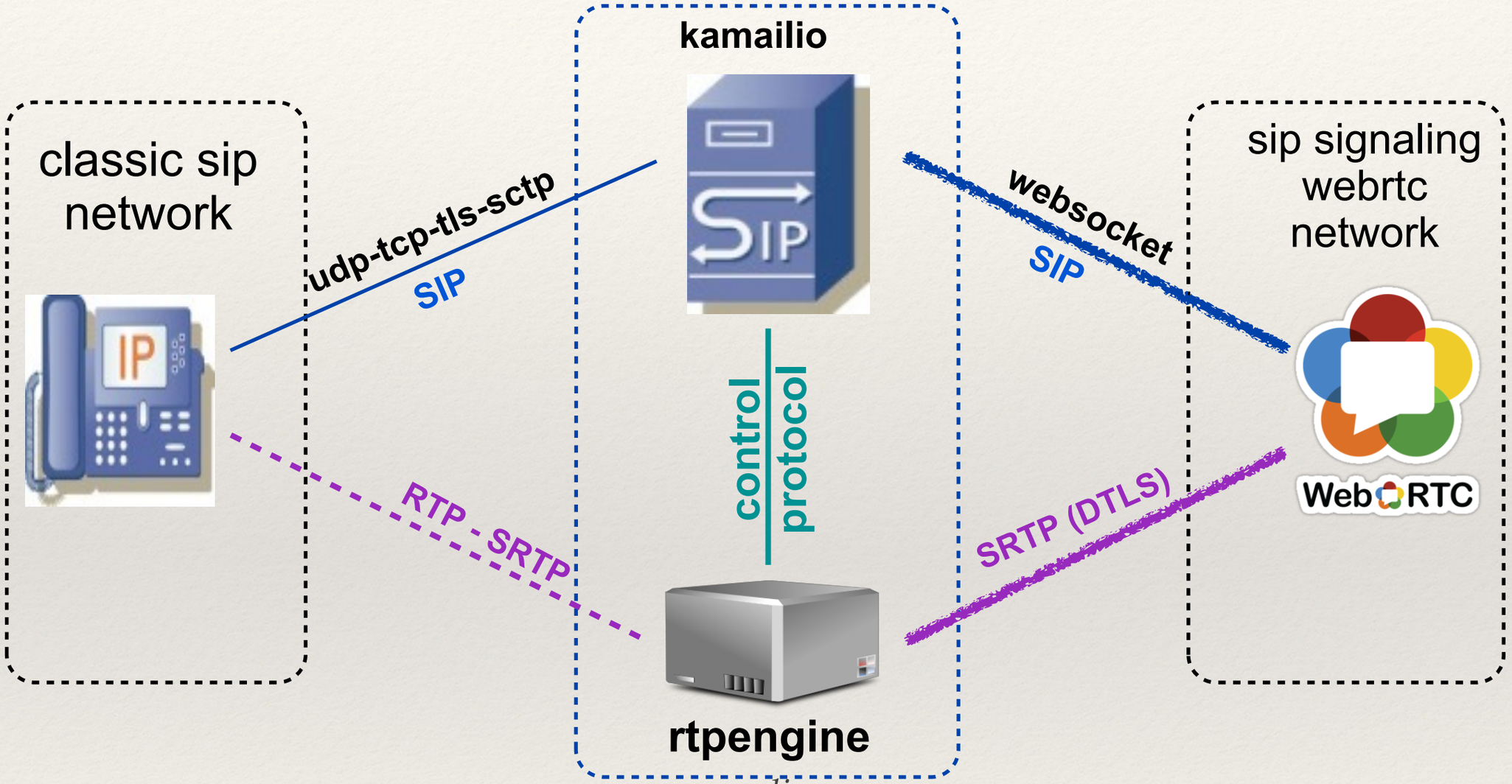
http://www.kamailio.org/wiki/features/new-in-4.1.x
http://www.kamailio.org/wiki/features/new-in-4.2.x
http://www.kamailio.org/wiki/features/new-in-4.3.x
http://www.kamailio.org/wiki/features/new-in-devel

WebRTC

websocket support since 2012



WebRTC SIP Gateway



media encryption/decryption (audio/video/…)

https://github.com/sipwise/rtpengine
- the Sipwise media proxy for Kamailio -

Kamcli

- new Kamailio command line interface tool
  - written in Python
  - extensible via modules
    - easier to handle input values and format output
    - targets using RPC (e.g., JSONRPC) control interface
- https://github.com/asipto/kamcli

```
$ kamcli --help
$ kamcli <command> --help
$ kamcli <command> <subcommand> --help
```

Siremis Web Interface

Code Repository on Github

kamailio / kamailio

Unwatch 71 | Unstar 164 | Fork 112

Kamailio - The Open Source SIP Server http://www.kamailio.org — Edit

21,544 commits | 102 branches | 157 releases | 87 contributors

Branch: master | kamailio / +

registrar: updated the docs for registered(...) function

miconda authored 2 days ago — latest commit b0a776b312

atomic	Fix license in atomic_alpha.h	4 years ago
cfg	Cfg Remove SVN ID, history	7 months ago
doc	doc/stylesheets: xsl template to quote mysql table and column names	3 months ago
docbook	core/docbook: updated the entity for ctl socket file name	3 months ago
etc	etc/kamailio-oob.cfg: remove modules_k from mpath	17 days ago
examples	examples: fix bashism in /bin/sh script	6 months ago
lib	lib/srdb1: dummy string for null values pointed to static buffer	12 days ago
mem	mem: TLSF remove unused function	2 months ago

Code

Issues 18

Pull requests 4

Wiki

Pulse

Graphs

Settings

HTTPS clone URL

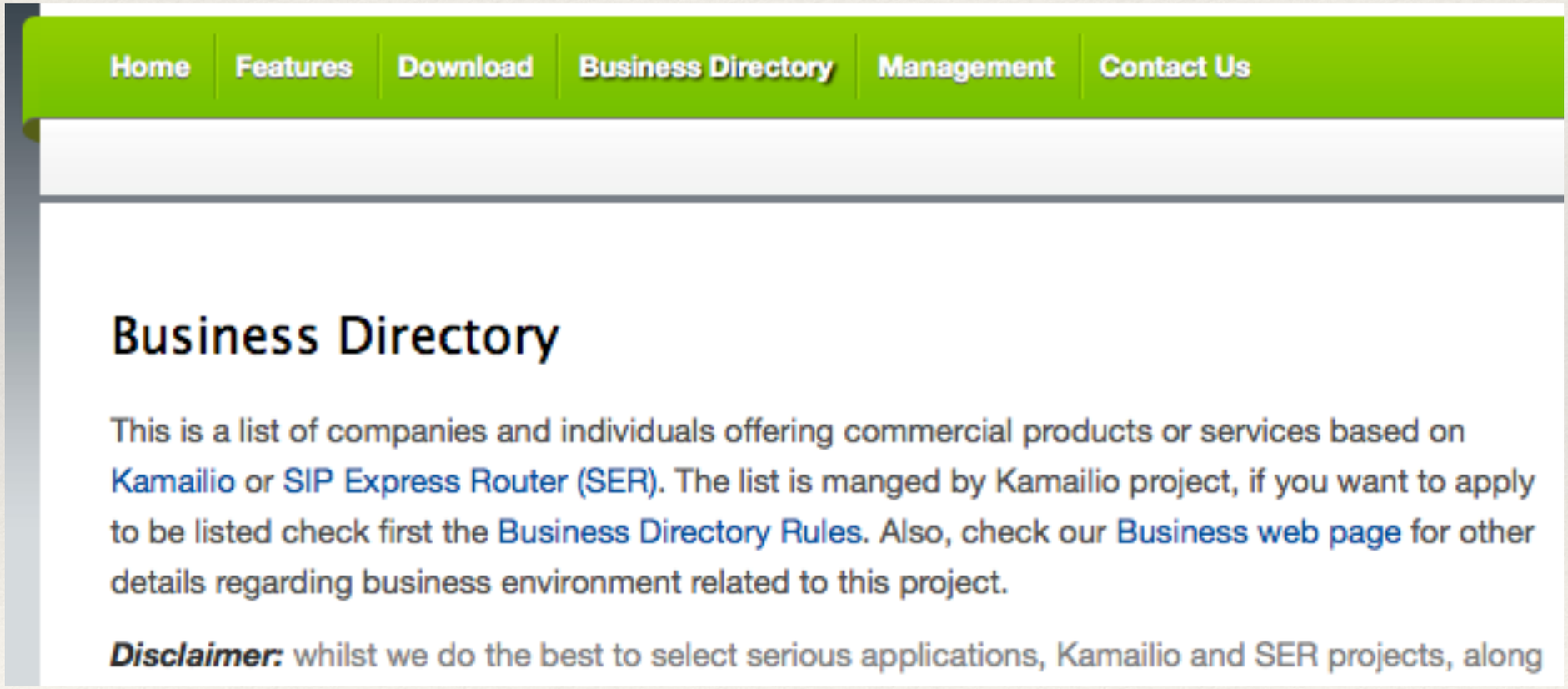
https://github.com/kama

You can clone with HTTPS, SSH, or Subversion.

Kamailio Business Directory

Anyone offering services related to Kamailio can get listed there with:

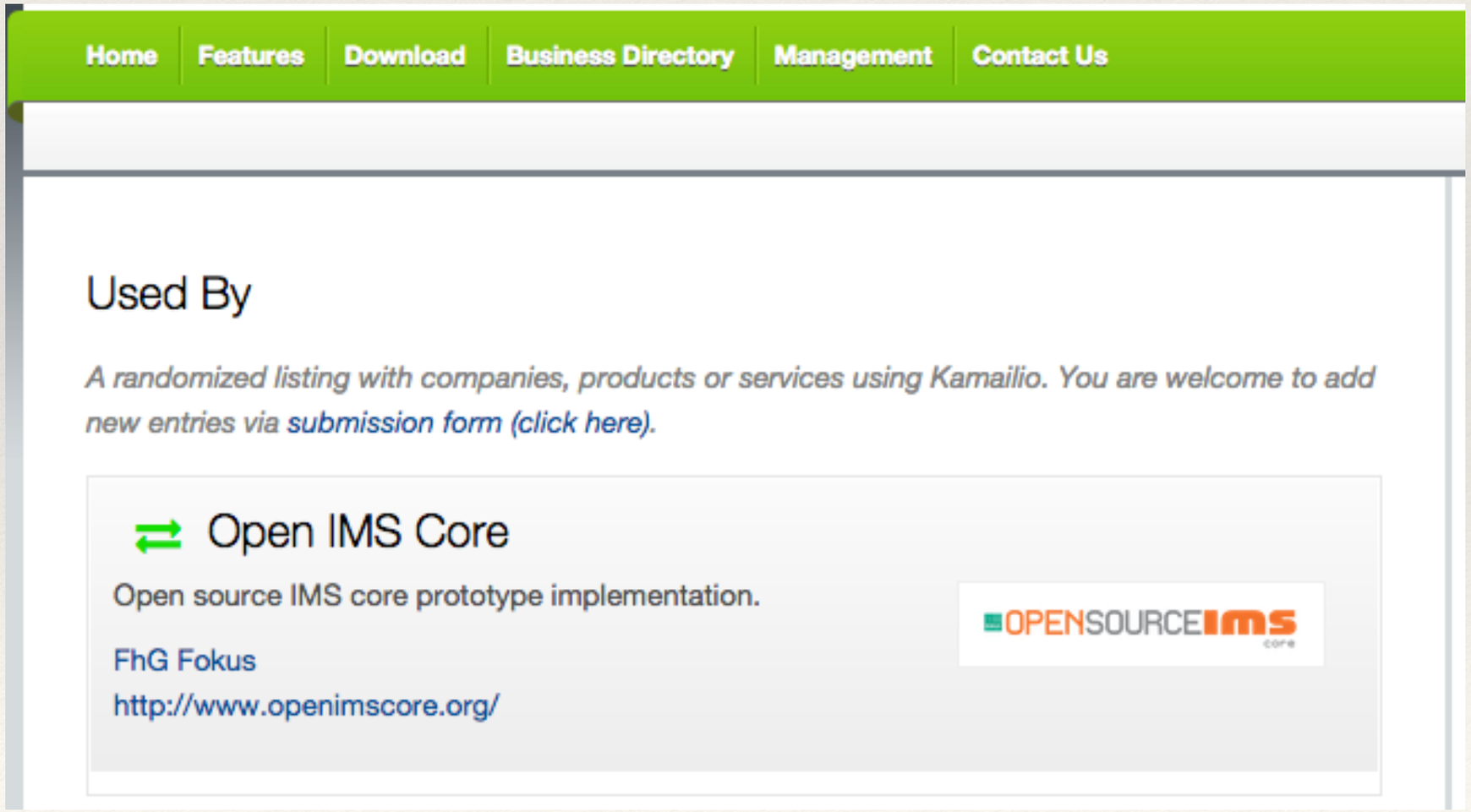
- *name and contact coordinates*
- *short description*



Kamailio Used-By Directory

Anyone using Kamailio can get listed there with:

- *name and contact coordinates*
- *short description*
- *logo*



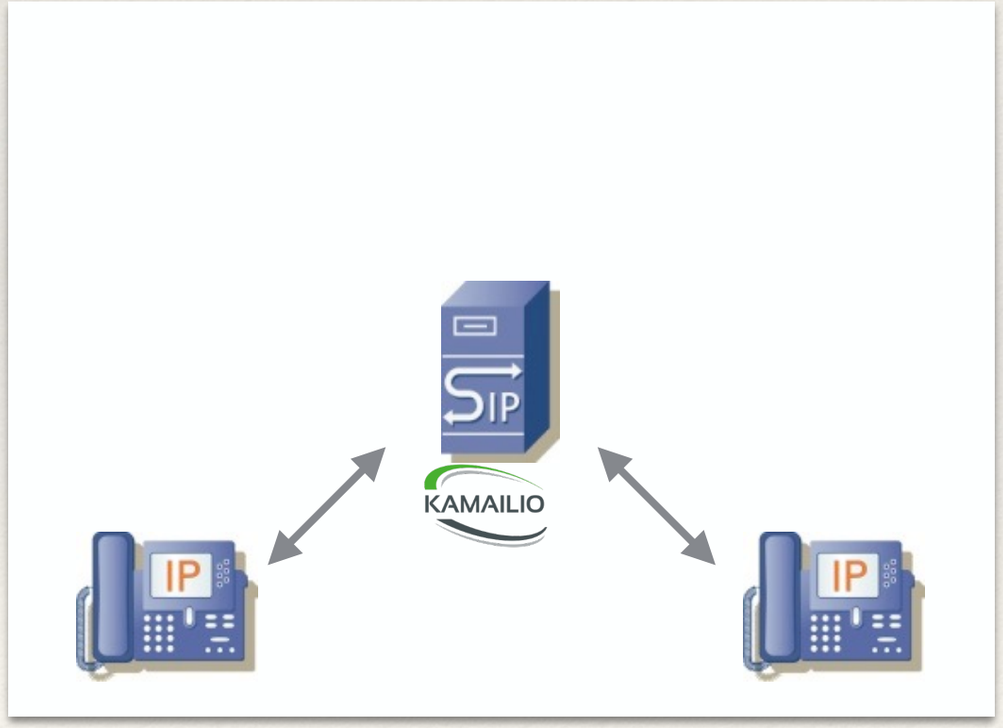
http://www.asipto.com/sw/kamailio-admin-book/

Kamailio

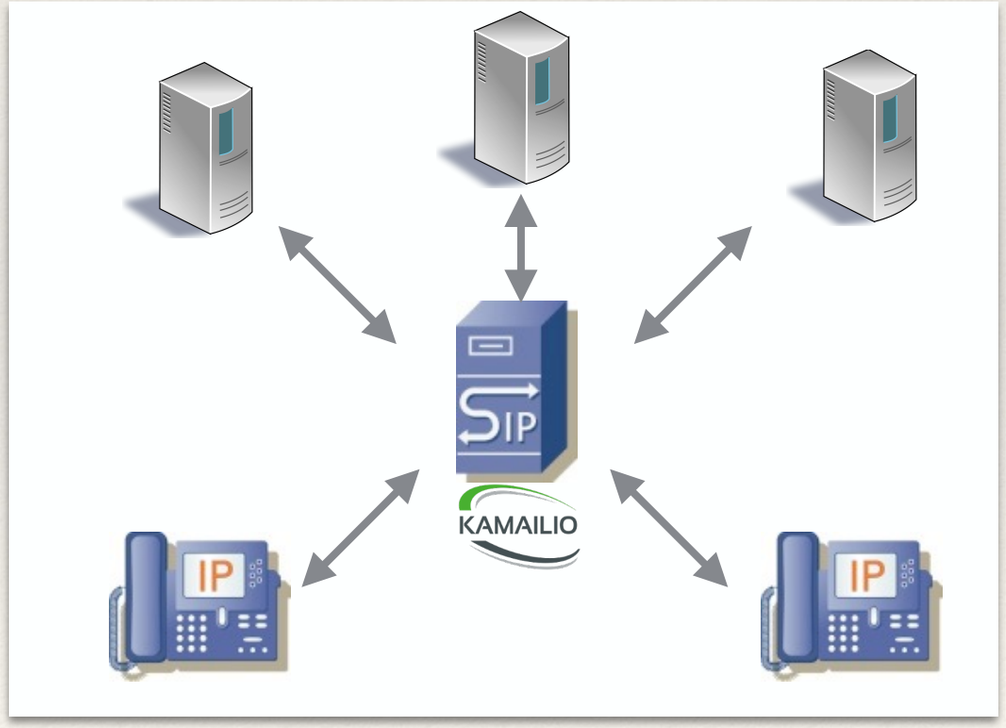
API Based SIP Routing

Two Common Architectures

main signalling server



edge signalling server
(sbc)

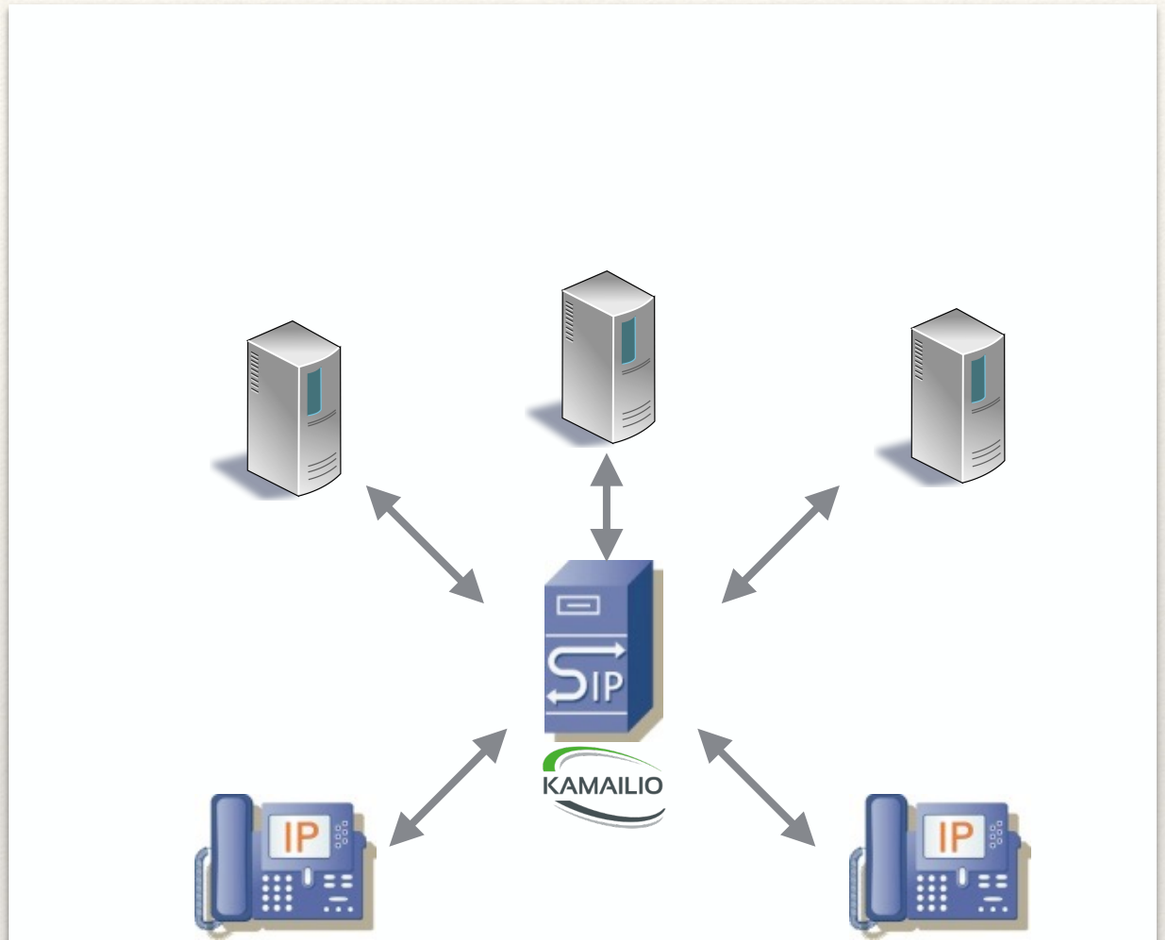


Load Balancer - LCR - ...

dispatcher module

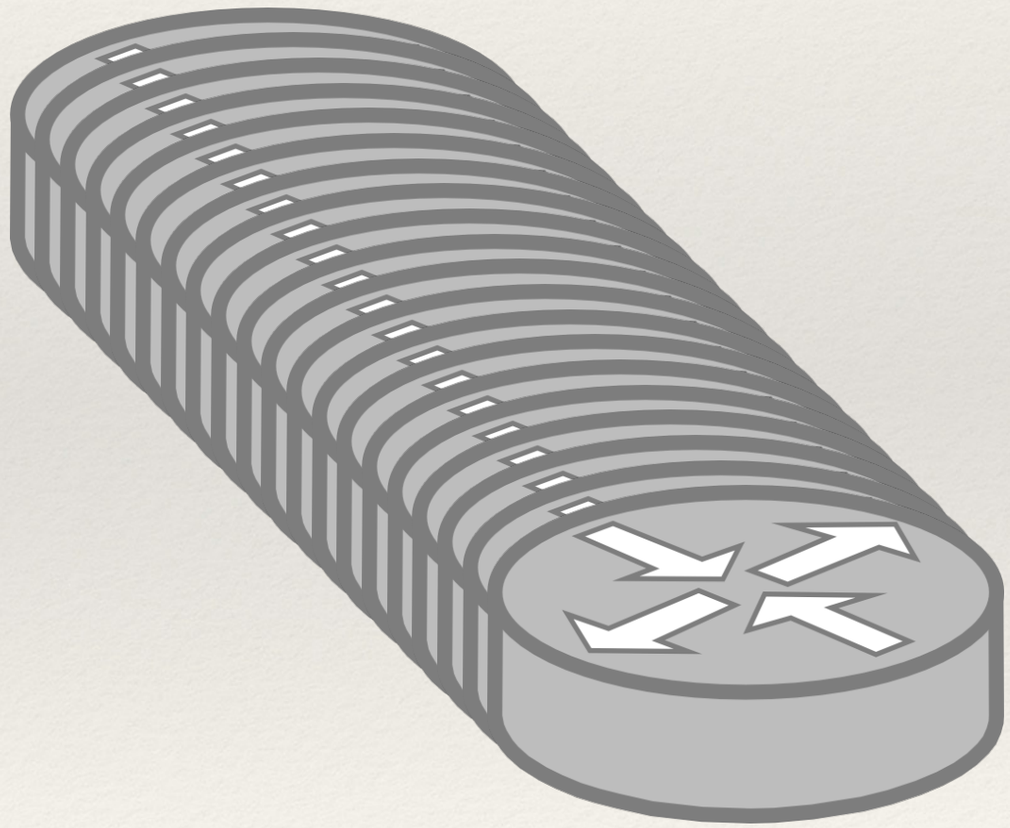
- list of balancing nodes from file or database
- monitoring of nodes (activate/inactivate automatically)
- re-route in case of failure
- various algorithms: hashing, weight distribution, round robin, call load distribution, priority routing
- reload list of nodes without restart

```
# Dispatch requests
route[DISPATCH] {
        # round robin dispatching on gateways group '1'
        if(!ds_select_dst("1",“4")) {
                send_reply("404", "No destination");
                exit;
        }
        xdbg("--- SCRIPT: going to <$ru> via <$du>\n");
        t_on_failure("RTF_DISPATCH");
        route(RELAY);
        exit;
}
# Re-route in case of failure
failure_route[RTF_DISPATCH] {
        if (t_is_canceled()) {
                exit;
        }
        # next node - only for 500 or local timeout
        if (t_check_status(“500") || (t_branch_timeout() && !t_branch_replied())) {
                if(ds_next_dst()) {
                        t_on_failure("RTF_DISPATCH");
                        route(RELAY);
                        exit;
                }
        }
}
```

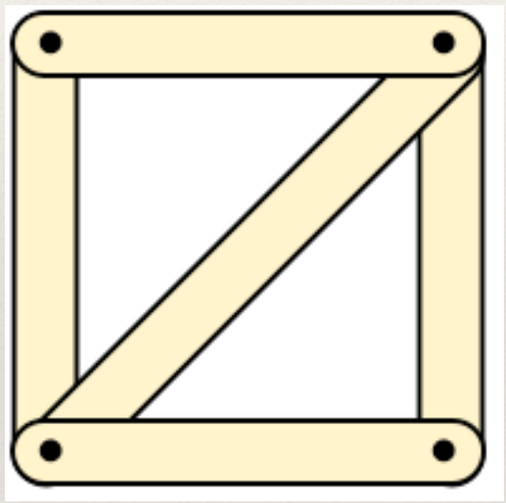


Routing Extensions

- over 200 modules, plenty of them for routing
  - dispatcher
  - lcr
  - carrierroute
  - pdt
  - mtree
  - dialplan
  - prefix route
  - speeddial
  - aliases
  - location
  - …



fast, native implementation



route selection and data model rigidity

Generic Tools

- blending for a better taste
  - configuration script expressions
  - xavp (avp) - per transaction storing data
  - htable, mtree - global in memory storage systems
  - sql - nosql connectors
  - http - jsonrpc clients
  - embedded interpreters (lua, perl, python, …)
  - json - xml tools
  - message queues & event api

The Logic Schism

SIP Standard Handling

sip parsing
authentication
record routing
retransmissions
headers management
failure rerouting
…

KAMAILIO

Business Decision

authorization
billing
next hops
  cheapest rout
  qos
  available resources
…

YOUR API

Basic Concepts

Parallel Forking

```
request_route {
...
    append_branch("sip:103@kamailio.org");
    append_branch("sip:104@kamailio.org");

    $var(newdst) = "sip:105@kamailio.org";
    append_branch("$var(newdst)");

    t_relay();
}
```

Serial Forking

```
request_route {
    ...

    $ru = "sip:102@kamailio.org";
    $xavp(newdst) = "sip:103@kamailio.org";
    $xavp(newdst) = "sip:104@kamailio.org";

    t_on_failure("RETRY");
    t_relay();
}

failure_route[RETRY] {
    if (t_is_canceled()) {
        exit;
    }
    if($xavp(newdst) != $null) {
        $ru = $xavp(newdst);
        $xavp(newdst) = $null;
        t_on_failure("RETRY");
        t_relay();
        exit;
    }
}
```

Building Blocks

- fetching the next routing hops
  - raw data
    - http/rpc results + parsing
  - structured data
    - rtjson
- processing
  - synchronous
    - config wait & act
  - asynchronous
    - evapi, mqueue & rtimer

HTTP & You

the http_query() from utils module

http://kamailio.org/docs/modules/stable/modules/utils.html#utils.f.http_query

```
http_query("http://myapi.local/sip.php?src=$(ru{s.escape.param})&dst=$(fu{s.escape.param})",
 "$var(result)")
```

```
<?php
 echo "sip:" . $_GET["dst"] . "@1.2.3.4;transport=tls"
          . "|"
          . "sip:" . $_GET["dst"] . "@2.3.4.5;transport=sctp";
?>
```

Kamailio Config

Serial Forking

```
request_route {
    ...

    http_query("http://myapi.local/sip.php?src=$(ru{s.escape.param})&dst=$(fu{s.escape.param})",
                "$var(result)");
    $ru = $(var(result){s.select,0,|});
    $xavp(newdst) = $(var(result){s.select,1,|});

    t_on_failure("RETRY");
    t_relay();
}

failure_route[RETRY] {
    if (t_is_canceled()) {
        exit;
    }
    if($xavp(newdst) != $null) {
            $ru = $xavp(newdst);
            $xavp(newdst) = $null;
            t_on_failure("RETRY");
            t_relay();
           exit;
    }
}
```

Your Response Structure

xmlops module to parse the response

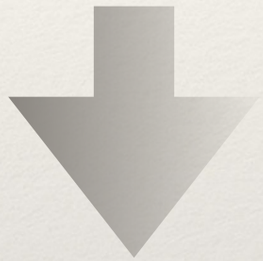
```
<?php
 echo “<route><uri>sip:” . $_GET[“dst”] . “@1.2.3.4;transport=tls”
        . ”</uri><uri>”
        . “sip:” . $_GET[“dst”] . “@2.3.4.5;transport=sctp”
        . “</uri></route>;
?>
```

json or jansson module to parse the response

```
<?php
 echo “{ ‘routes’:  [ ‘sip:” . $_GET[“dst”] . “@1.2.3.4;transport=tls”
        . ”’, ”
        . “‘sip:” . $_GET[“dst”] . “@2.3.4.5;transport=sctp'”
        . “] }”;
?>
```

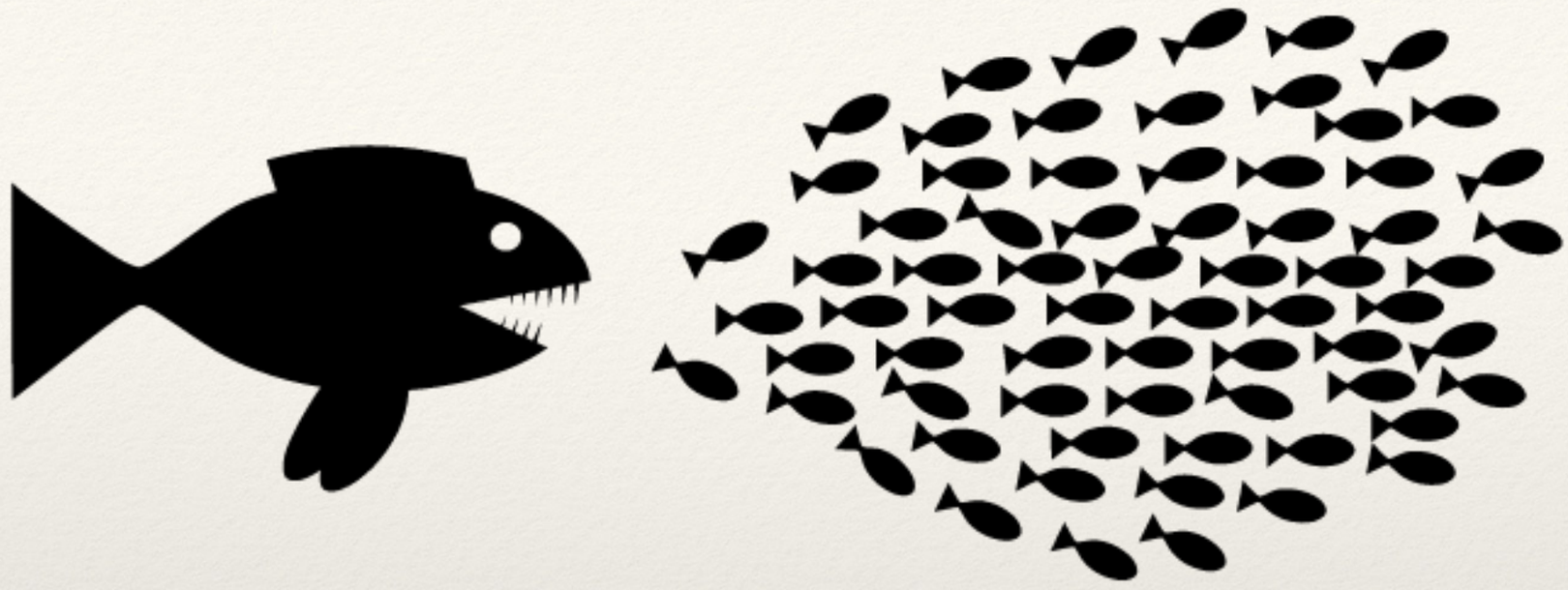
too many options

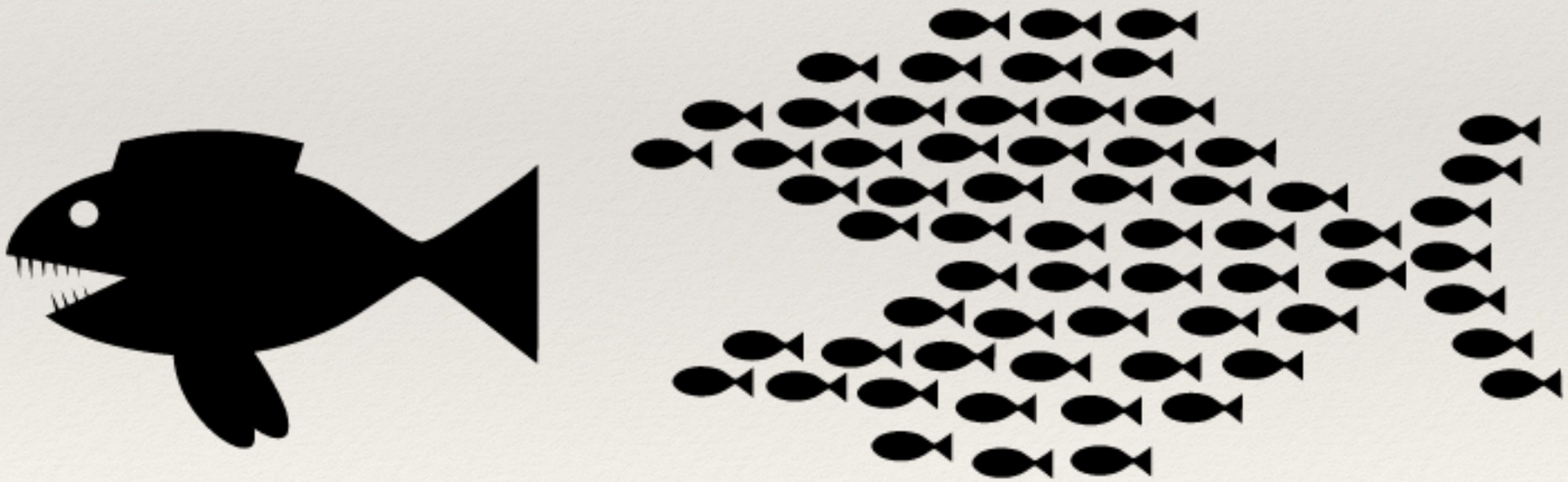
too much config scripting



many like structured, well defined data

“one good option”





Predefined Routing Response

- kamailio config - combine:
  - evapi
  - jansson
  - **rtjson**
- external application
  - node.js
  - json data response
  - predfined structure

evapi

- event based API
- bare data or netstrings serialization
- data format up to you
- asynchronous processing
  - park invite request
  - send event data
  - resuming handling the invite when receiving response on the event

http://kamailio.org/docs/modules/stable/modules/evapi.html

rtjson

- use json input to fill routing attributes
  - request uri
  - outbound proxy
  - intermediary hops
  - local socket
  - caller/callee addresses
  - extra headers
  - ringing timeout
  - flags

http://kamailio.org/docs/modules/stable/modules/rtjson.html

rtjson


```
{
        "version": "1.0",
        "routing": "serial",
        "routes": [
                {
                        "uri": "sip:127.0.0.1:5080",
                        "dst_uri": "sip:127.0.0.1:5082",
                        "path": "<sip:127.0.0.1:5084>, <sip:127.0.0.1:5086>",
                        "socket": "udp:127.0.0.1:5060",
                        "headers": {
                                "from": {
                                        "display": "Alice",
                                        "uri": "sip:alice@127.0.0.1"
                                },
                                "to": {
                                        "display": "Alice",
                                        "uri": "sip:alice@127.0.0.1"
                                },
                                "extra": "X-Hdr-A: abc\r\nX-Hdr-B: bcd\r\n"
                        },
                        "branch_flags": 8,
                        "fr_timer": 5000,
                        "fr_inv_timer": 30000
                },
...
```

kamailio.cfg

```
...
route[TOEVAPI] {
        evapi_async_relay("{\n \"event\": \"sip-routing\",\n"
                " \"tindex\": $T(id_index), \"tlabel\": $T(id_label),"
                " \"caller\": \"$fU\", \"callee\": \"$rU\"\n}");

        xlog("L_INFO", "suspended transaction: $T(id_index) / $T(id_label)\n");
        exit;
}
...
```

node.js

```
...
        var response = {};

        response.version="1.0";
        response.xtra = {};
        response.xtra.tindex=jdoc.tindex;
        response.xtra.tlabel=jdoc.tlabel;
        response.routing = "serial";
        response.routes = [];
        response.routes[0] = {};
        response.routes[0].uri = "sip:127.0.0.1:5080";
        response.routes[0].headers = {};
        response.routes[0].headers.extra = "X-Hdr-A: abc\r\nX-Hdr-B: bcd\r\n";
        response.routes[1] = {};
        response.routes[1].uri = "sip:127.0.0.1:5090";
        response.routes[1].headers = {};
        response.routes[1].headers.extra = "X-Hdr-C: cde\r\nX-Hdr-D: def\r\n";

        return kanapi_send_response(JSON.stringify(response));
...
```

http://kb.asipto.com/kamailio:k43-async-sip-routing-nodejs

node.js

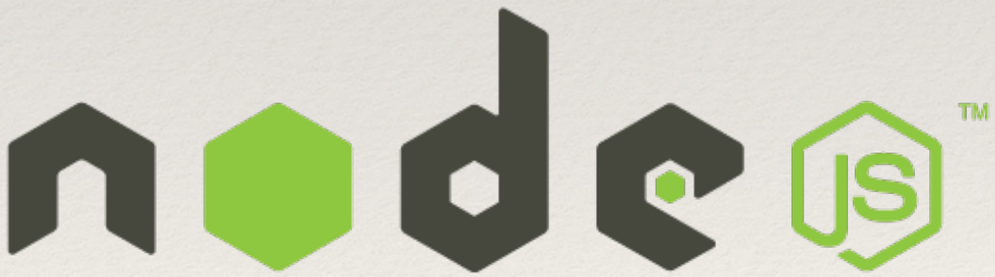
```
…
 function kanapi_send_response(data) {
    var buffer;

    data = new Buffer(data || '', null);

    buffer = Buffer.concat([new Buffer(data.length + ':', null), data, new Buffer(',', null)]);

    // console.log('[' + data + '] :----> [' + buffer + ']');

    client.write(buffer.toString('ascii'));
    return buffer;
}
…
```



http://kb.asipto.com/kamailio:k43-async-sip-routing-nodejs

kamailio.cfg

```
...
event_route[evapi:message-received] {
    xlog("received [$evapi(msg)] from $evapi(srcaddr):$evapi(srcport)\n");
    if($evapi(msg)=~"routing" && $evapi(msg)=~"tindex") {
                jansson_get("xtra.tindex", "$evapi(msg)", "$var(tindex)");
                jansson_get("xtra.tlabel", "$evapi(msg)", "$var(tlabel)");
                $var(evmsg) = $evapi(msg);
                xlog("L_INFO", "preparing to resume transaction for processing: $var(tindex) / $var(tlabel)\n");
                t_continue("$var(tindex)", "$var(tlabel)", "EVAPIRESPONSE");
    }
}
...
```

kamailio.cfg

```
…
route[EVAPIRESPONSE] {
        xlog("L_INFO", "resumed transaction for processing: $T(id_index) / $T(id_label)\n");

        rtjson_init_routes("$var(evmsg)");
        rtjson_push_routes();

        t_on_branch("MANAGE_BRANCH");
        t_on_failure("MANAGE_FAILURE");
        route(RELAY);
        exit;
}
…
failure_route[MANAGE_FAILURE] {
        route(NATMANAGE);
        if (t_is_canceled()) {
                exit;
        }
        if(rtjson_next_route()) {
                t_on_branch("MANAGE_BRANCH");
                t_on_failure("MANAGE_FAILURE");
                route(RELAY);
                exit;
        }
}
…
```

In summary

it's all about you

or sip, kamailio and you

- pick your desired format: json, xml, custom
- pick your desired transport: http, rpc, evapi, or even sip
- pick handling style: sync or async
- pick your external call routing controller: node+js, http+php, standalone app, …

Thank you!

Questions?!?

Daniel-Constantin Mierla
www.asipto.com
@miconda

Kamailio World 2016 - Planning a Special Edition

Kamailio Project

15 YEARS OF DEVELOPMENT

2001-2016
from SER to Kamailio

www.kamailioworld.com