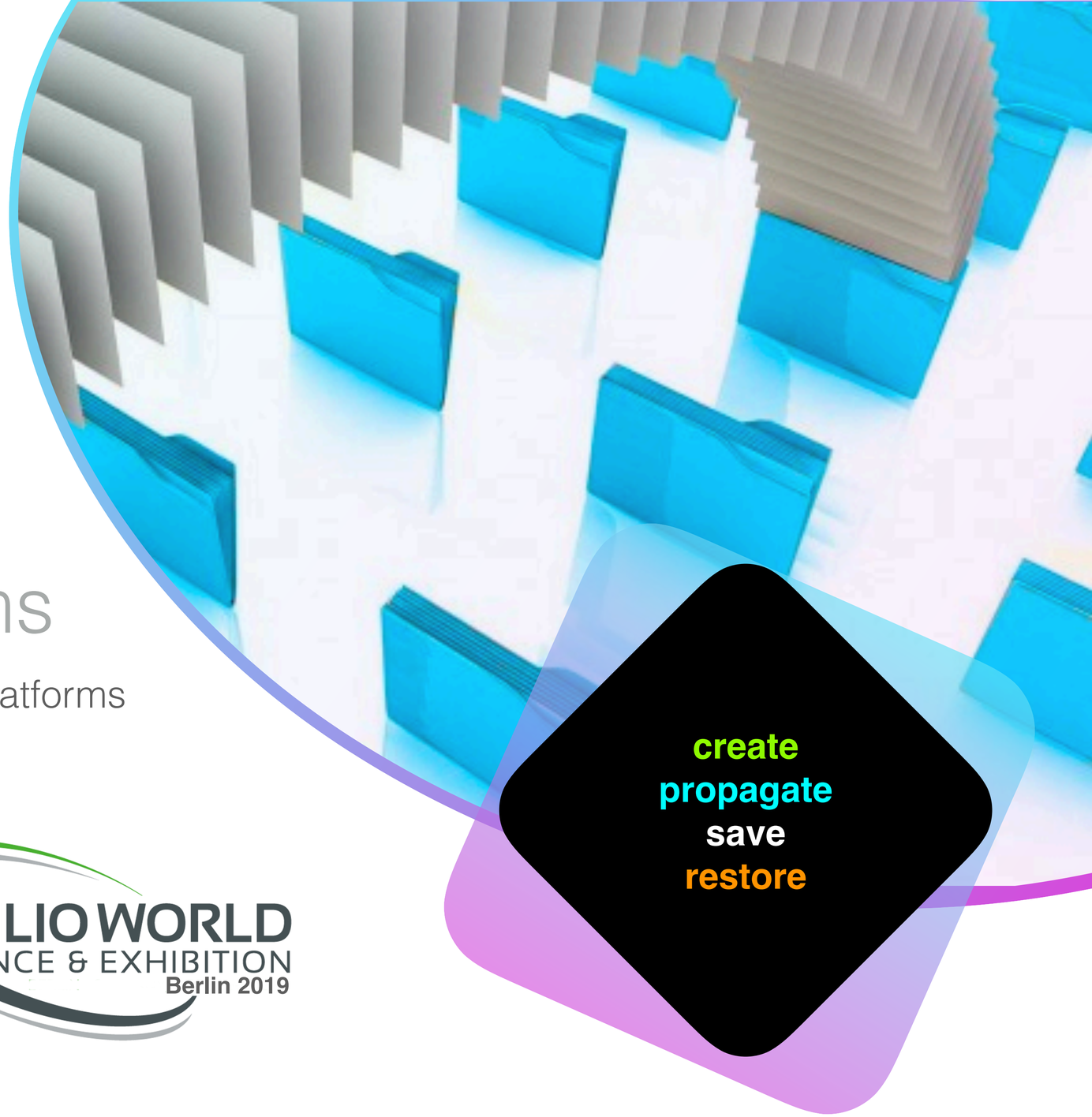
KAMAILIO

Replication Mechanisms

Distributing data in multi-node Kamailio VoIP platforms



KAMAILIO WORLD
CONFERENCE & EXHIBITION
Berlin 2019

Daniel-Constantin Mierla - @miconda
Co-Founder Kamailio Project
asipto.com

Replication in computing involves sharing information so as to ensure consistency between redundant resources, such as software or hardware components, to improve reliability, fault-tolerance, or accessibility.

Replication in computing can refer to:

data replication - where the same data is stored on multiple storage devices

https://en.wikipedia.org/wiki/Replication_(computing)

Replication != Scalability

Replication **=>** **Redundancy**

REDUNDANCY IS IMPORTANT

BUT IT IS A COST GENERATOR

AVOID OVER-ENGINEERING

REPLICATION LAYERS

- **Backed storage**
  - *database backend modules - sql or no-sql*

- **In-memory storage**
  - kamailio specific in-memory data structures

SQL - NoSQL Backend

- their own clustering solutions -

sql modules

- **db_mysql**
- **db_postgress**
- **db_slqite (*)**
- **db_oracle**
- **db_unixodbc**

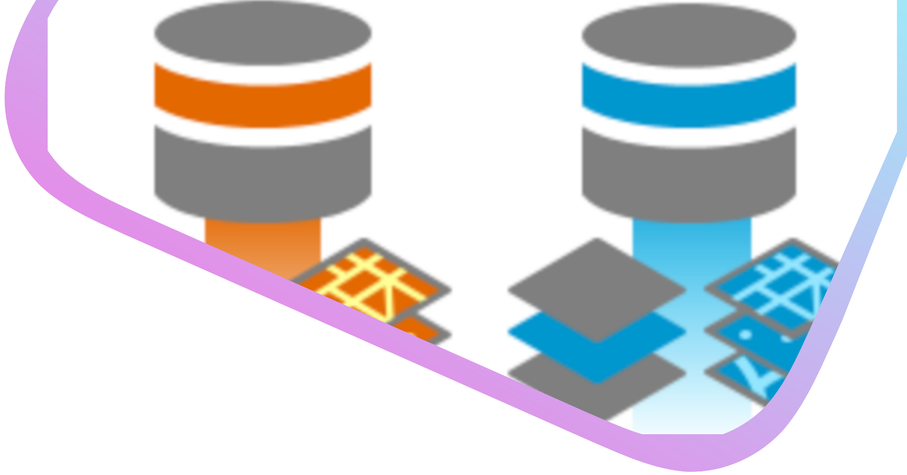
nosql modules

- **ndb_redis**
- **ndb_mongodb**
- **ndb_cassandra**
- memcache

hybrid sql modules

- db_text
- **db_mongodb**
- **db_redis**
- db_flatstore
- db_berkely

CLUSTERING



- **db_cluster module**
  - *basic clustering (more for redundancy purposes)*
  - *application layer clustering*
    - *intermediate layer in front of real db connector modules*
  - *features*
    - *parallel, round robin and serial writing*
    - *round robin and serial reading*
    - *can work with different database modules at the same time*
    - *allows definition of many clusters at the same time*
  - *no function exported to config*

```
...
modparam("db_cluster", "connection",
             "c1=>mysql://kamailio:kamailiorw@localhost/kamailio1")
modparam("db_cluster", "connection",
             "c2=>mysql://kamailio:kamailiorw@localhost/kamailio2")
modparam("db_cluster", "cluster", "k1=>c1=9r9p;c2=9r9p")

modparam("acc", "db_url", "cluster://k1")

modparam("sqlops", "sqlcon", "ca=>cluster://k1")
...
```

SQLOPS

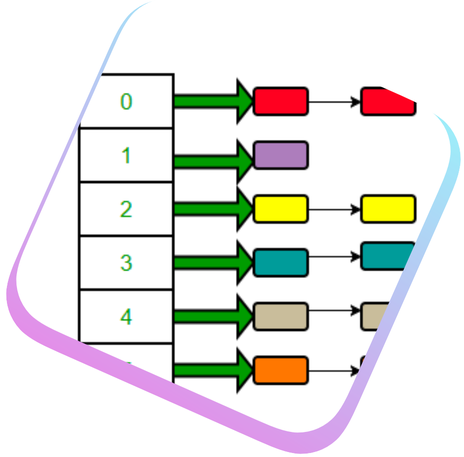
- **sqlops module**
  - *any kind of sql query inside the configuration file*
  - results accessible via configuration file variables
  - could be a solution for ‘partial’ replication

- **Example of usage**
  - Define two connections
  - Do two insert operations

```
modparam("sqlops","sqlcon", “ca=>mysql://user:pass@host1/kamailio“)
modparam("sqlops","sqlcon", “cb=>mysql://user:pass@host2/kamailio")

…

sql_query("ca", "INSERT INTO event_history"
          " (src_username, dst_username, callid, event_type, event_code, event_time)"
          " VALUES ('$var(srcuser)', '$var(dstuser)', '$ci', ‘call’, 408, $Ts)”);

sql_query("cb", "INSERT INTO event_history"
          " (src_username, dst_username, callid, event_type, event_code, event_time)"
          " VALUES ('$var(srcuser)', '$var(dstuser)', '$ci', ‘call’, 408, $Ts)");
```

IN-MEMORY STORAGE REPLICATION

Distributed Message Queues
Hash Tables - Registrations - Calls - Presence States



htable module

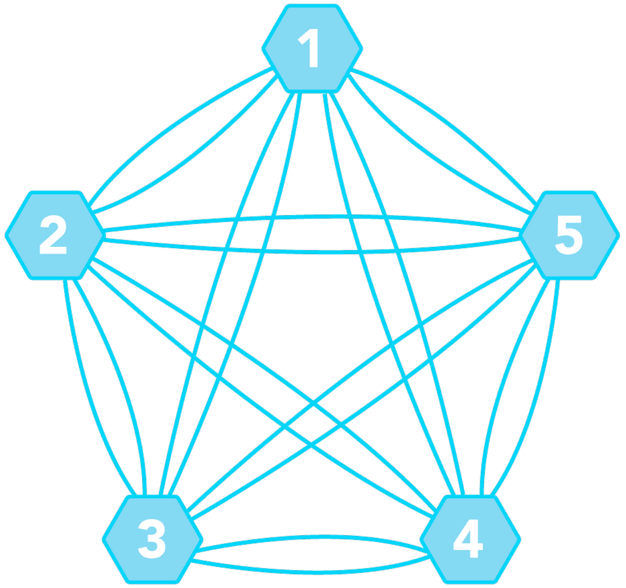
usrloc module

dialog module

presence modules

DMQ: SIP-BASED DISTRIBUTED MESSAGE QUEUE

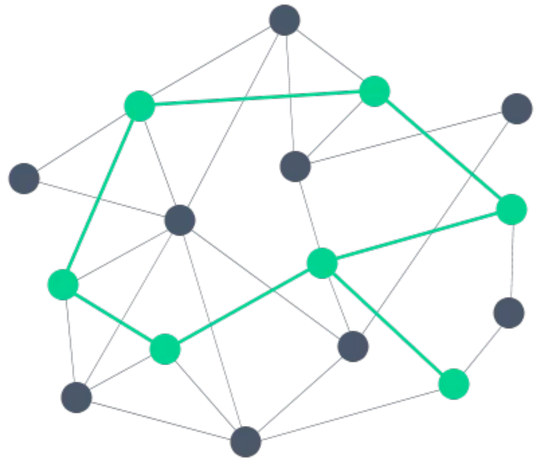
- **dmq module**
  - *Introduced in 2011 (GSoC project)*
  - *Uses SIP as a transport protocol*
    - *Kamailio is very fast at routing SIP*
    - *Full flexibility in processing via kamailio.cfg*
  - *Features*
    - *Network discovery - new nodes announce themselves and become part of the network*
    - *Syncronization of data by notifying all active nodes*
    - *Monitoring of the nodes in the network*
    - *Scripting flexibility to define channels*
  - *Extensions to replicate*
    - *Registration records*
    - *Hash table items*
    - *Dialogs*
    - *Presence*



https://www.kamailio.org/docs/modules/stable/modules/dmq.html

DMQ: NODE DISCOVERY AND HTABLE REPLICATION

```
KDMQ sip:notification_peer@192.168.40.15:5090 SIP/2.0
Via: SIP/2.0/UDP 192.168.40.15;branch=z9hG4bK55e5.423d95110000
To: <sip:notification_peer@192.168.40.15:5090>
From:
<sip:notification_peer@192.168.40.15:5060>;tag=2cdb7a33a7f21abb98fd3a44968e3ffd-5b01
CSeq: 10 KDMQ
Call-ID: 1fe138e07b5d0a7a-50419
Content-Length: 116
User-Agent: kamailio (5.2.2 (x86_64/linux))
Max-Forwards: 1
Content-Type: text/plain

sip:192.168.40.16:5060;status=active
sip:192.168.40.15:5060;status=disabled
sip:192.168.40.17:5060;status=active
```



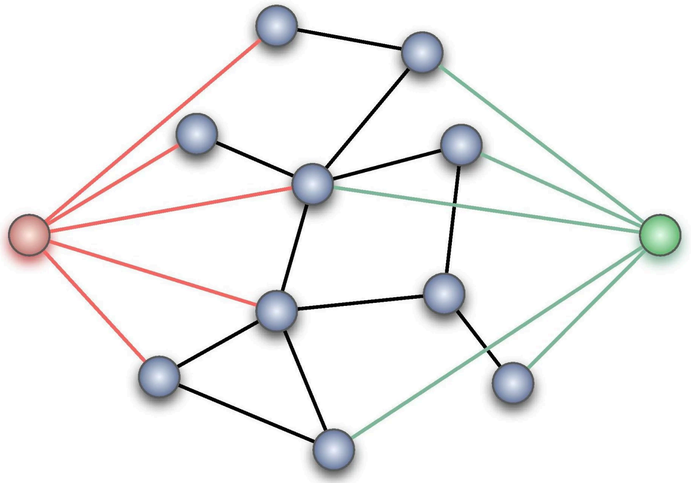
```
modparam("dmq", "server_address", "sip:LOCAL_IP:5060")
modparam("dmq", "notification_address", "sip:PEER1_IP:5060")
modparam("dmq", "notification_address", "sip:PEER2_IP:5060")

modparam("htable", "htable", "ipban=>size=8;autoexpire=300;dmqreplicate=1;")
. . .
request_route {
. . .
    if(is_method("KDMQ")) {
        dmq_handle_message();
    }
. . .
```

https://www.kamailio.org/docs/modules/stable/modules/dmq.html

https://www.kamailio.org/docs/modules/stable/modules/htable.html

DMQ: SIP REQUEST REPLICATION

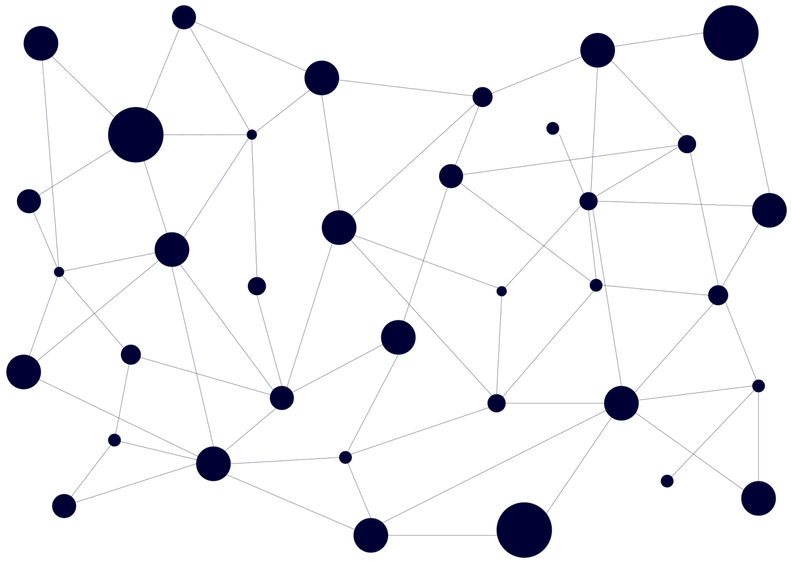
```
. . .
request_route {
. . .
    if(is_method("REGISTER")) {
        if (dmq_is_from_node()) {
            # from peer node
           save("location");
        } else {
            route(AUTH);
            save("location");
            dmq_t_replicate("1");
        }
    }
. . .
```



https://www.kamailio.org/docs/modules/stable/modules/dmq.html

DMQ: USRLOC REPLICATION

```
. . .
modparam("dmq", "server_address", "sip:LOCAL_IP:5060")
modparam("dmq", "notification_address", "sip:PEER1_IP:5060")
modparam("dmq", "notification_address", "sip:PEER2_IP:5060")
modparam("dmq", "num_workers", 6)
modparam("dmq_usrloc", "enable", 1)
modparam("dmq_usrloc", "sync", 1)
modparam("dmq_usrloc", "batch_size", 2000)
modparam("dmq_usrloc", "batch_usleep", 1000)
. . .
request_route {
. . .
    if(is_method("KDMQ")) {
        dmq_handle_message();
    }
. . .
```



https://www.kamailio.org/docs/modules/stable/modules/dmq.html

https://www.kamailio.org/docs/modules/stable/modules/dmq_usrloc.html

Alternative - using presence framework

https://www.kamailio.org/docs/modules/stable/modules/pua.html

https://www.kamailio.org/docs/modules/stable/modules/pua_reginfo.html

DMQ: DIALOGS REPLICATION

```
. . .
modparam("dmq", "server_address", "sip:LOCAL_IP:5060")
modparam("dmq", "notification_address", "sip:PEER1_IP:5060")
modparam("dmq", "notification_address", "sip:PEER2_IP:5060")
modparam("dmq", "num_workers", 6)

modparam("dialog", "enable_dmq", 1)
. . .
request_route {
. . .
    if(is_method("KDMQ")) {
        dmq_handle_message();
    }
. . .
```

https://www.kamailio.org/docs/modules/stable/modules/dmq.html

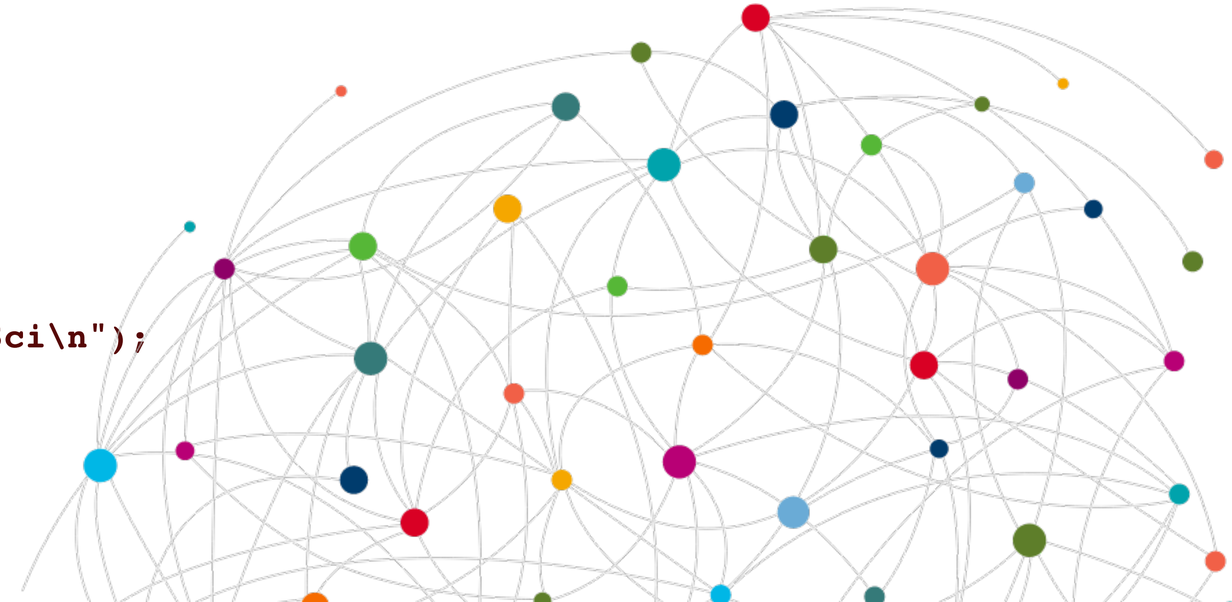
https://www.kamailio.org/docs/modules/stable/modules/dialog.html

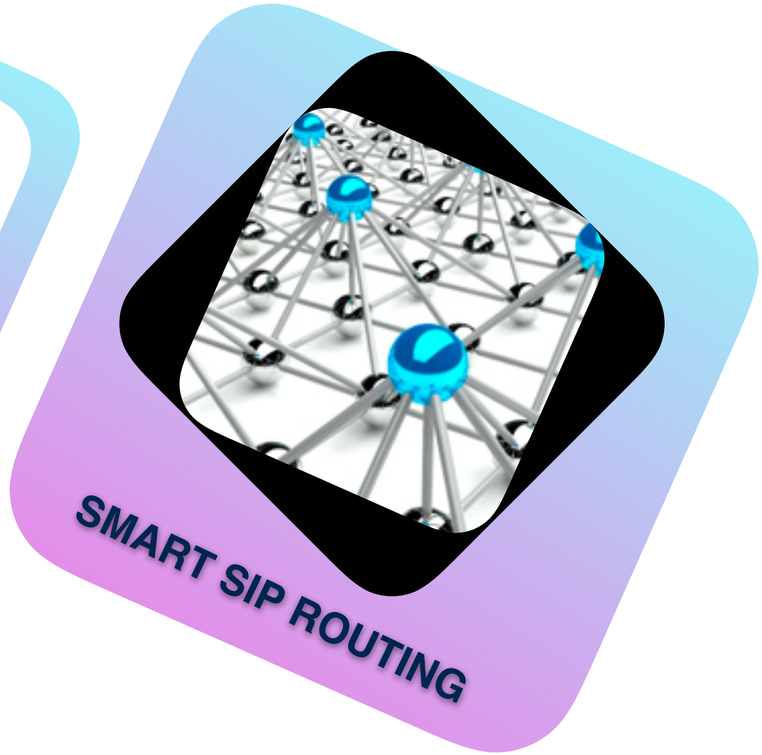
BACKEND REPLICATION: LOAD DIALOGS ON DEMAND

```
. . .
modparam("dialog", "db_mode", 1)
. . .
request_route {
. . .
  if(has_totag()) {
    if(!is_known_dlg()) {
      dlg_db_load_callid("$ci");
      if(!is_known_dlg()) {
         xlog("no dialog found with callid: $ci\n");
      }
    }
  }
. . .
```

https://www.kamailio.org/docs/modules/stable/modules/dialog.html

```
. . .
modparam("dialog", "db_mode", 1)
. . .
request_route {
. . .
  if(has_totag()) {
    if(!is_known_dlg()) {
      dlg_db_load_extra();
      if(!is_known_dlg()) {
        xlog("no dialog found with callid: $ci\n");
      }
    }
  }
. . .
```



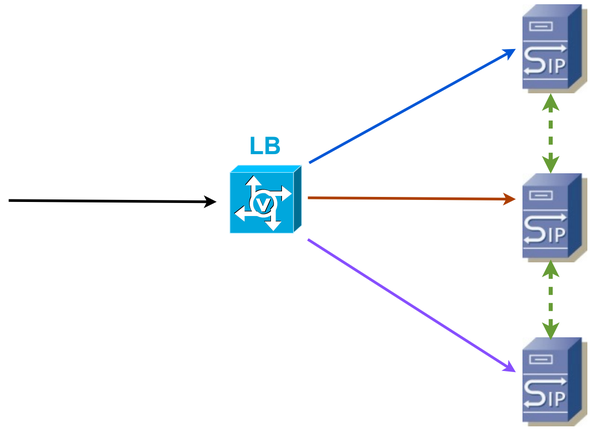


TM-BASED REPLICATION

Simplicity - Reliability

REGISTER REPLICATION

- *load balancer with failover in front of a small farm of SIP servers*




```
modparam("registrar", "sock_flag", FLT_REGSOCKHDR)
modparam("registrar", "sock_hdr_name", “P-Local-Sock")
modparam("registrar", "received_avp", “$avp(received)”)

   route[REGISTRAR] {
      . . .
      if(src_ip != PAIRSRV_IP) {
            if (!save("location")) {
               sl_reply_error();
               exit;
            }

         # replicate the REGISTER
         if (isbflagset(FLB_NATB)) {
            append_hf(“P-SRC-NAT: yes\r\n”);
            append_hf(“P-RCV-ADDR: $avp(received)\r\n");
         }
         add_sock_hdr(“P-Local-Sock");
         force_send_socket(udp:LOCALNET_IP:LOCALNET_PORT);
         if (!t_replicate(“sip:PAIRSRV_IP:PAIRSRV_PORT")) {
            xlog("error - t_replicate failed\n");
         }
         exit;
      }
      . . .
   }
```

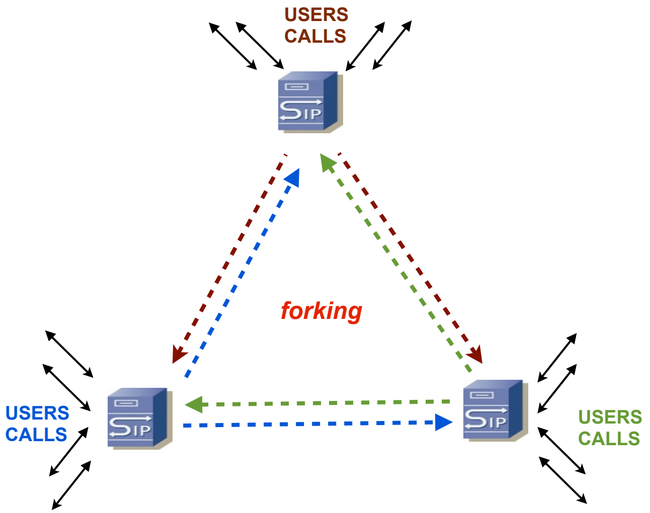
```
if(src_ip == PAIRSRV_IP) {
   # it's a replicated REGISTER
   if (is_present_hf("P-SRC-NAT")) {
      setbflag(FLB_NATB);
      $avp(received) = $hdr(P-RCV-ADDR);
   }
   # set the flag for retrieving socket info from HDR
   setflag(FLT_REGSOCKHDR);
   # just save it into memory
   if (!save("location","0x01")) {
      sl_reply_error();
      exit;
   }
}
```

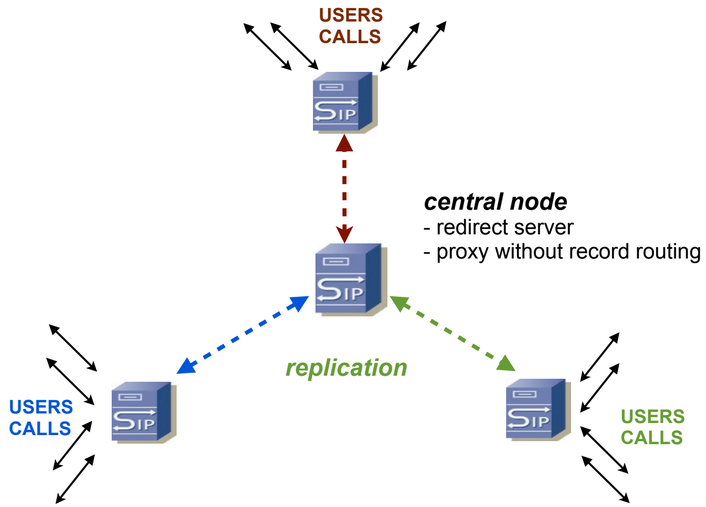
MESSAGE QUEUE CONNECTORS

- **rabbitmq module**
  - *well know messaging queue system*
  - *rabbitmq.com*
  - *https://www.kamailio.org/docs/modules/stable/modules/rabbitmq.html*

AMQP
Advanced Message Queuing Protocol

- **Nsq module**
  - *realtime distributed messaging platform*
  - *nsq.io*
  - *https://www.kamailio.org/docs/modules/stable/modules/nsq.html*


USERS
CALLS
forking
USERS
CALLS
USERS
CALLS




```
modparam(“htable”, “htable”, “kusrloc=>size=10;autoexpire=7200;”)
...
request_route {
    # add here ip authorization, etc…

    # handle location update notification
    if(method=="KUSRLOC" && $rU=="store") {
        $sht(kusrloc=>$rb) = “sip:” + $si + “:” + $sp + “;transport=” + $pr;
        send_reply("200", "Stored");
        exit;
    }
    # handle standard SIP requests
    if($sht(kusrloc=>$rU@$rd)!=$null) {
        $du = $sht(kusrloc=>$rU@$rd);
        t_relay();
        exit;
    }
    send_reply(“404”, “Not found”);
    exit;
}
```

```
# Handle SIP registrations
route[REGISTRAR] {
    if (is_method("REGISTER")) {
        if(isflagset(FLT_NATS)) {
            setbflag(FLB_NATB);
            # uncomment next line to do SIP NAT pinging
            ## setbflag(FLB_NATSIPPING);
        }
        if (!save("location"))
            sl_reply_error();

        $uac_req(method)=“KUSRLOC"
        $uac_req(ruri)="sip:store@centralnode.kamailio.org";
        $uac_req(furi)="sip:server@server1.kamailio.org";
        $uac_req(hdrs)="Content-Type: text/kusrloc\r\n";
        pv_printf(“$uac_req(body)”, "$fu@fd");
        uac_send_req();

        exit;
    }
}
```

```
route[TOMAIN] {
    $du = "sip:CENTRALNODEIP";
    route(RELAY);
    exit;
}

# USER location service
route[LOCATION] {
#!ifdef WITH_SPEEDDIAL
    # search for short dialing - 2-digit extension
    if($rU=~"^[0-9][0-9]$")
        if(sd_lookup("speed_dial"))
            route(SIPOUT);
#!endif
#!ifdef WITH_ALIASDB
    # search in DB-based aliases
    if(alias_db_lookup("dbaliases"))
        route(SIPOUT);
#!endif

    $avp(oexten) = $rU;
    if (!lookup("location")) {
        if(src_ip!=CENTRALNODEIP)
            route(TOMAIN);
...
}
```

SEE YOU AT THE NEXT

KAMAILIO WORLD CONFERENCE
SPRING 2020

KAMAILIO WORLD
CONFERENCE & EXHIBITION

{ REST }

Redis

mong

mbase

THANK YOU!

Daniel-Constantin Mierla
@miconda
Co-Founder Kamailio SIP Server Project
kamailio.org
asipto.com

KAMAILIO