Scaling Location Services with Kamailio

Henning Westerholt, Marius Zbihlei

Kamailio Project

Overview

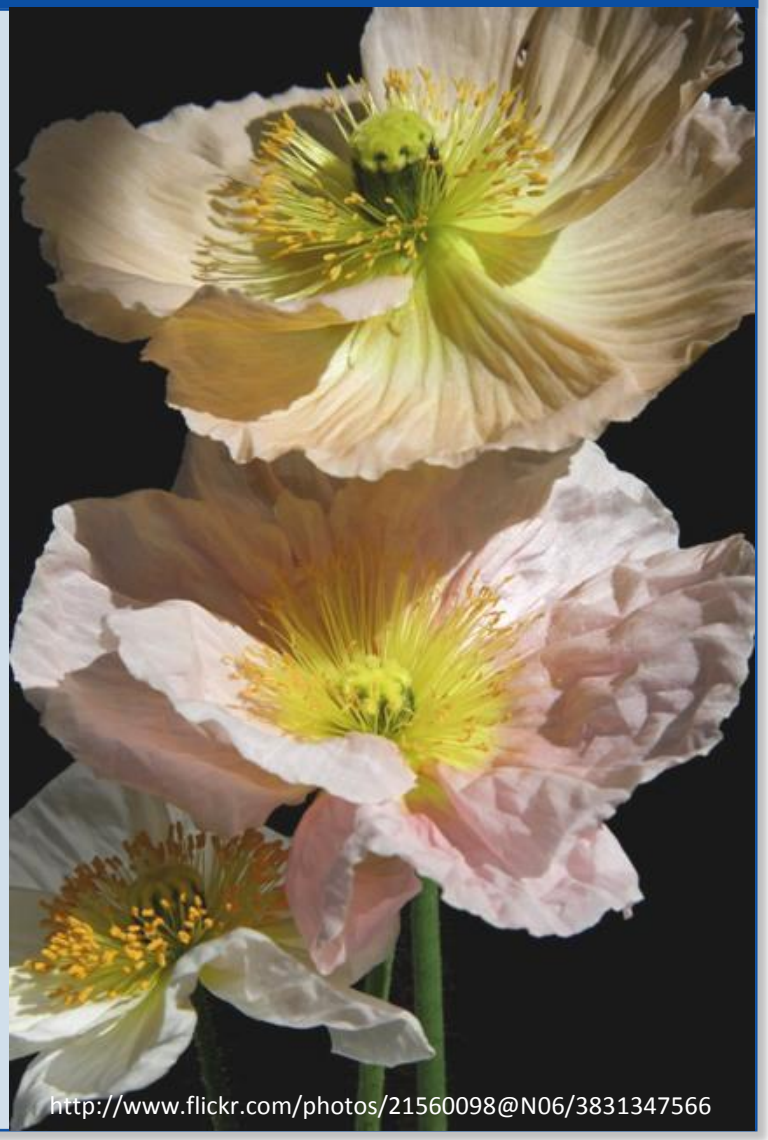
- Introduction
- Kamailio at 1&1
- Scaling Location Services
- Partitioned user location overview
- Upstream status, setup and development

About us

- 1&1 Internet AG
  - Since 2006/2007 with software development
  - Now more involved in IT Operation
  - Development done in Germany and Romania
- Kamailio Open Source project
  - Core Developer
  - Member of management board
- Part of the much bigger group that design, build and also operate the service we'll present in this talk
- Interested in Open Source and Open Systems

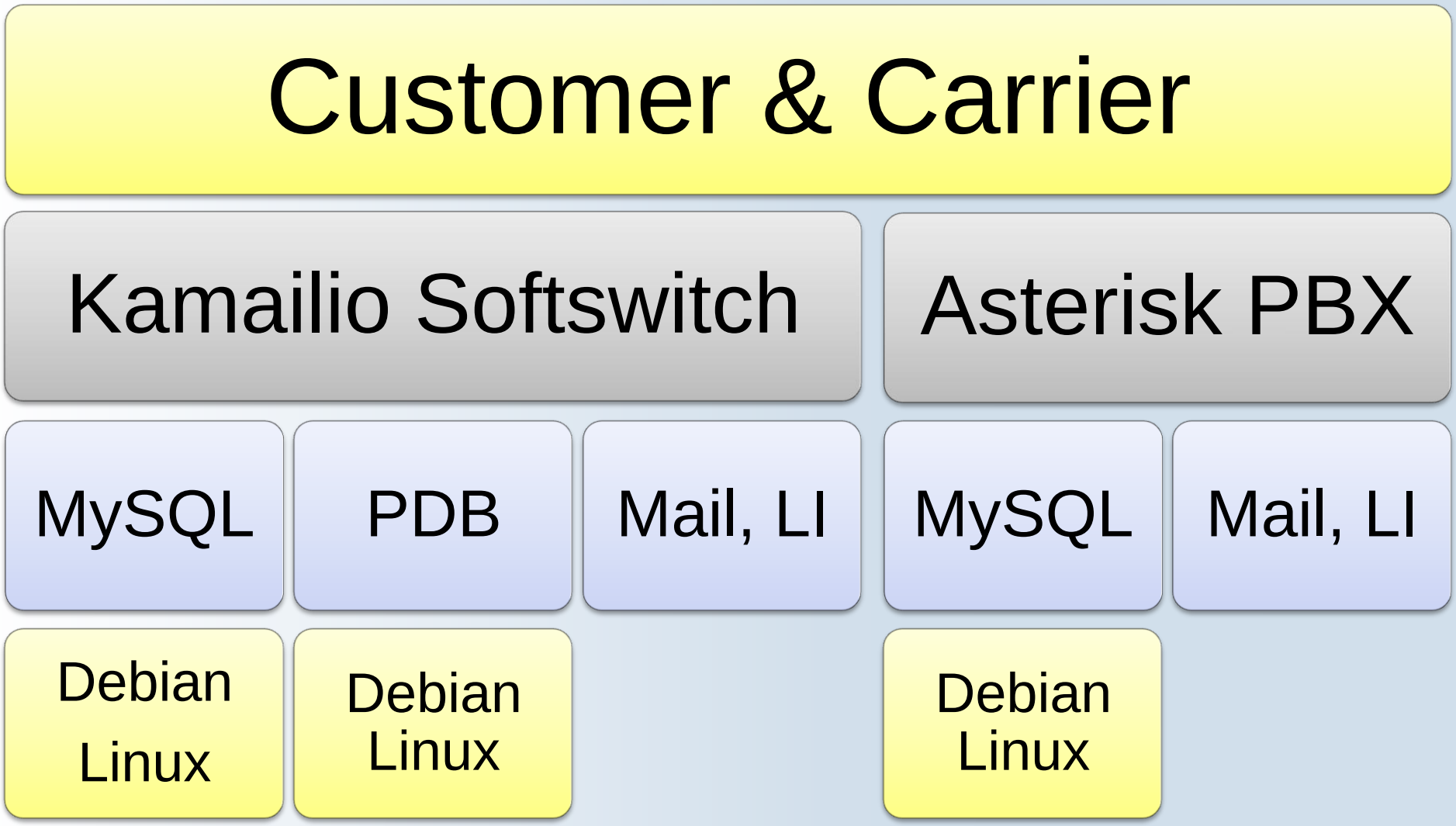
VoIP backend at 1&1

- Operated mainly with Open Source components
- One of the biggest deployments of its kind, started in 2005
- Data
  - Over three million customers on the platform
  - Interconnections to Telefonica, Vodafone and QSC
  - More then 1 billion minutes per Month to the PSTN
- Geographical redundant backend in a load-sharing setup
- Focus towards small businesses and home users
- Current interesting topics are IPv6 and LTE

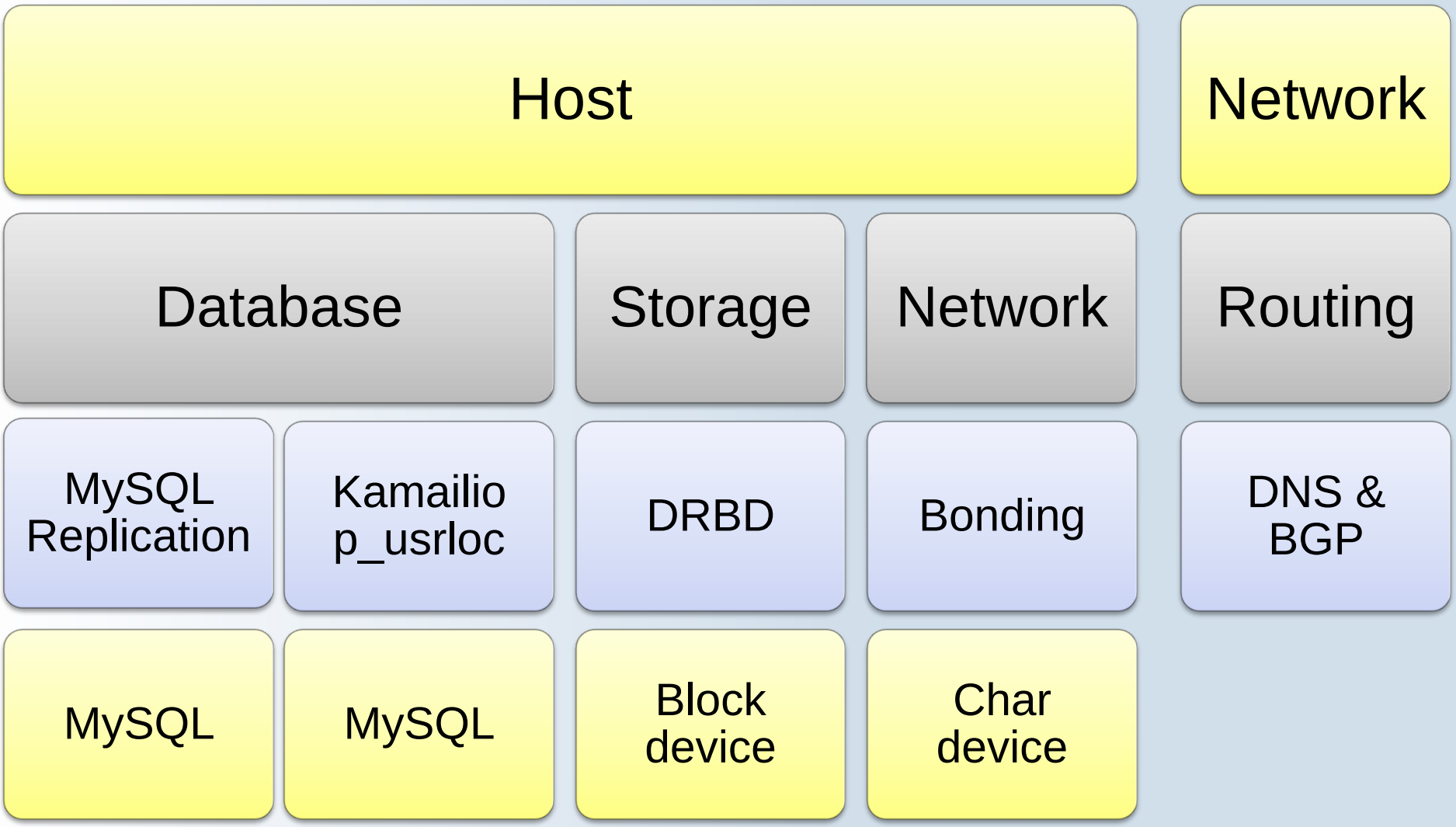


http://www.flickr.com/photos/21560098@N06/3831347566

VoIP stack overwiew



VoIP stack redundancy mechanism



Motivation for partitioned user location

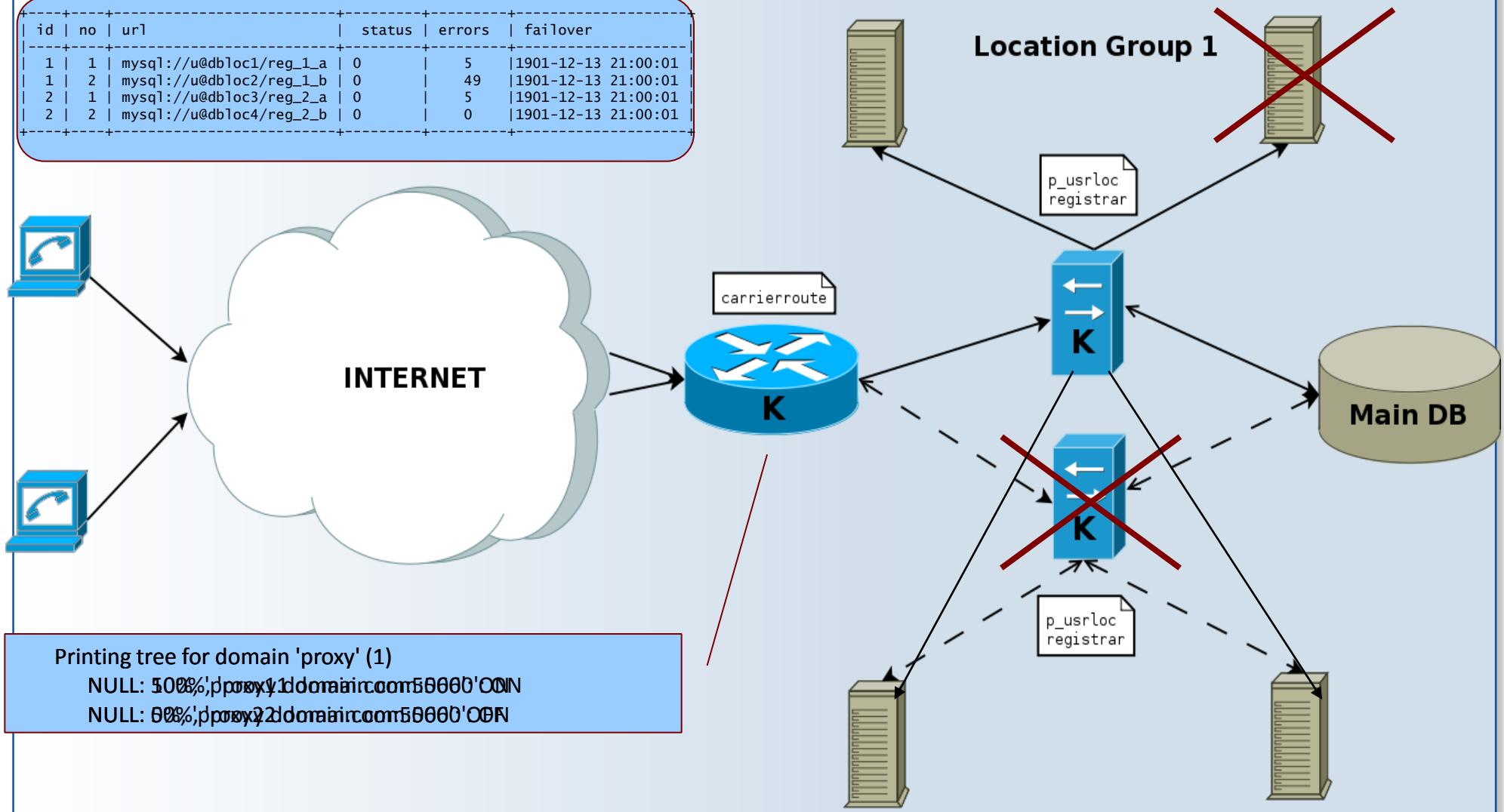
- User location is critical for call-setup
  - Mapping of Phone Number to IP Address
- High-availability for location services necessary
  - Load-balancing
  - Failover
  - Redundancy
- Problem space: n-Proxies writes and reads to m-databases concurrently
  - Without p_usrloc m=1
  - Workload equally distributed between reads, writes and deletes
- Failure of databases can happens everytime
- Maintenance is also necessary
- Different needs for user location and other location searches
- Proprietary or complex clustering solutions not wanted

Features of p_usrloc module

- Exports the usrloc API, so modules can use classic usrloc and p_usrloc with minimal changes to the config file, and no changes to code
- Usable with any SQL backend (MySQL, PostgreSQL, Oracle)
- Can support also non-distributed location service, in parallel with distributed ones
- Can use a faster Read-Only Master Database for configuration data
- The location group is selected by hashing over the either the username or username@domain
- Two failover algorithms that provides additional flexibility to the module
- Cluster Configuration is stored in a database
  - Active/passive state, Failover Time etc..
  - But no customer QoS impact in case of database non-availability

HA location service design

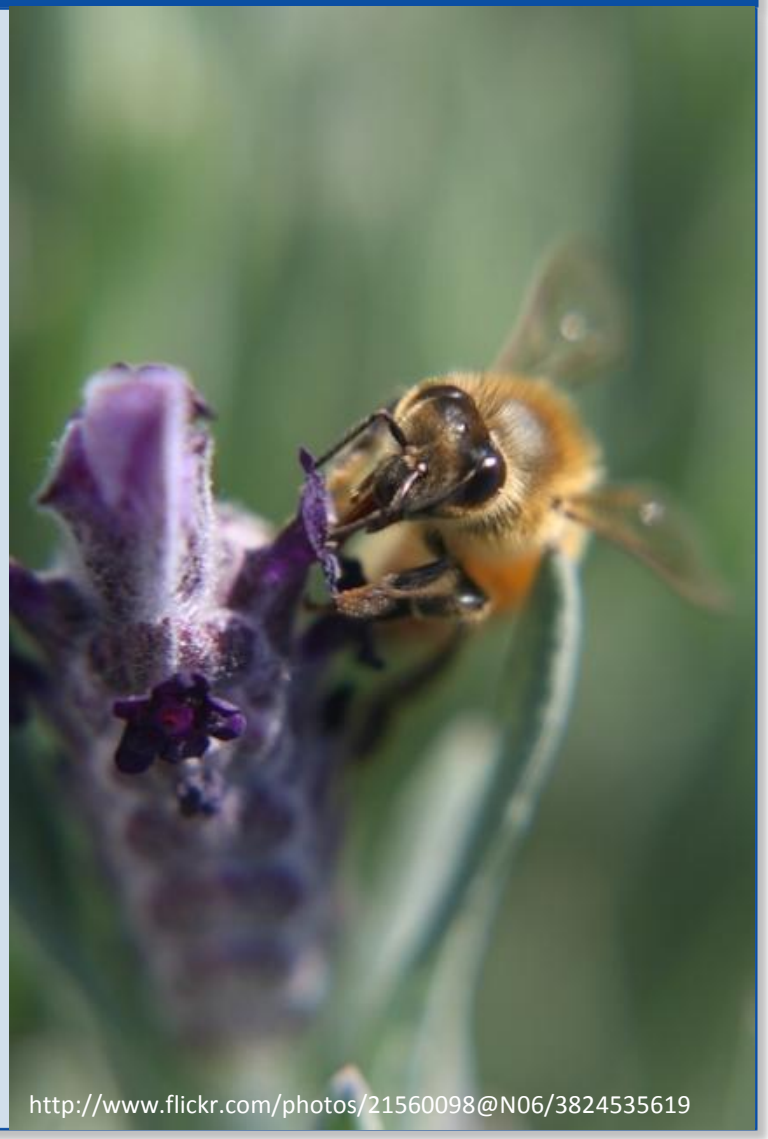
id	no	url	status	errors	failover
1	1	mysql://u@dbloc1/reg_1_a	0	5	1901-12-13 21:00:01
1	2	mysql://u@dbloc2/reg_1_b	0	49	1901-12-13 21:00:01
2	1	mysql://u@dbloc3/reg_2_a	0	5	1901-12-13 21:00:01
2	2	mysql://u@dbloc4/reg_2_b	0	0	1901-12-13 21:00:01



Printing tree for domain 'proxy' (1)
NULL: 100%, 'proxy1.domain.com:5060;lr' CON
NULL: 0%, 'proxy2.domain.com:5060;lr' CON

Upstream status

- Currently in master branch of Kamailio repository
- More information in the module documentation files
- Will be probably included in release version 3.2
- Missing features:
  - Automatic deletion of expired contacts
  - Configurable number of location handlers in a groups (currently changeable at compile time)
  - Spare Databases handling testing
  - General testing
- Internal version since years in production, without any know bugs



http://www.flickr.com/photos/21560098@N06/3824535619

Installation

- Get the source

git clone --depth 1 git://git.sip-router.org/sip-router kamailio

cd kamailio

- Tune the build

make FLAVOUR=kamailio cfg

vim modules.lst

Enable the wanted modules: p_usrloc, db_mysql etc.

- Compile

make all

- Install

make install

Binaries in /usr/local/sbin, modules in /usr/local/lib/kamailio/modules/ and /usr/local/lib/kamailio/modules_k/

Testsetup and Usage

- Configure the databases

  modules_k/p_usrloc/p_usrloc.sql script added to master db
  module_k/p_usrloc/location.sql added to each location db
  setup locdb Table with cluster configuration

- Load the module in the configuration file

  loadmodule “p_usrloc”
  loadmodule “registrar”

- Configure module parameters

  modparam("p_usrloc", "write_db_url", "mysql://ser:ser@localhost/ser")

  modparam("p_usrloc", "domain_db", "location=cluster,cfa=single")

  modparam("p_usrloc", "reg_db_table", "locdb")

- Use the normal lookup/save functions

  if (method=="REGISTER") { save("location"); exit; }

  if (!lookup("location")) { ...}

- Kill one database, do some calls, monitor locdb table and log file for failover

Thanks for your attention!

Questions?

Contact

- Henning Westerholt
  - *henning.westerholt@1und1.de*
- Marius Zbihlei
  - marius.zbihlei@1and1.ro