# FreeSWITCH High Availability and Scaling

**Giovanni Maruzzelli**
gmaruzz@OpenTelecom.IT

# Agenda

- High Availability and Scalability
- FreeSWITCH specific requirements

- HOW TO:
  - High Available SIP Proxy for Signaling Distribution and NAT Handling
  - High Available RTPProxy for Media Distribution and NAT Handling
  - High Available Database for Status Sharing and Persistence
  - High Available Filesystem for Configuration and Voice Mail Sharing
  - Many FreeSWITCH Servers Acting as One Big FreeSWITCH

# What is your VoIP server doing?

- Signaling
- Media

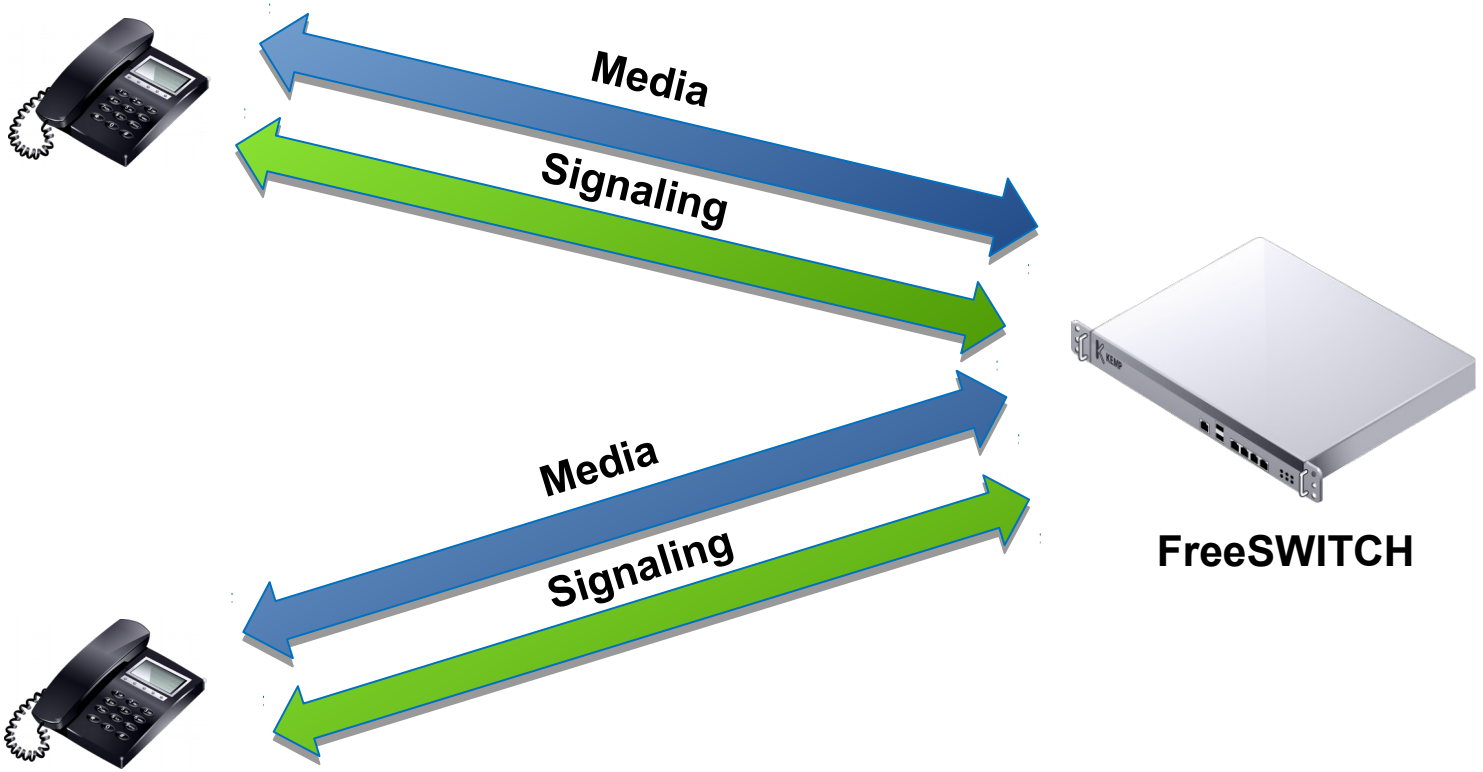
- NAT traversing (eg: relaying audio)

- Registration (Location)
- NAT piercing (eg: OPTIONS)

# One Single FreeSWITCH

- A single FreeSWITCH:
  - Has its own Configuration
  - Keeps its own State
  - Writes and Reads Voice Mail
  - Manages NAT Handling (Media and Signaling)
  - Mixes Conference Participants' Media
  - Parks and Unparks Calls
  - Manages Queues and ACDs

# One Single FreeSWITCH



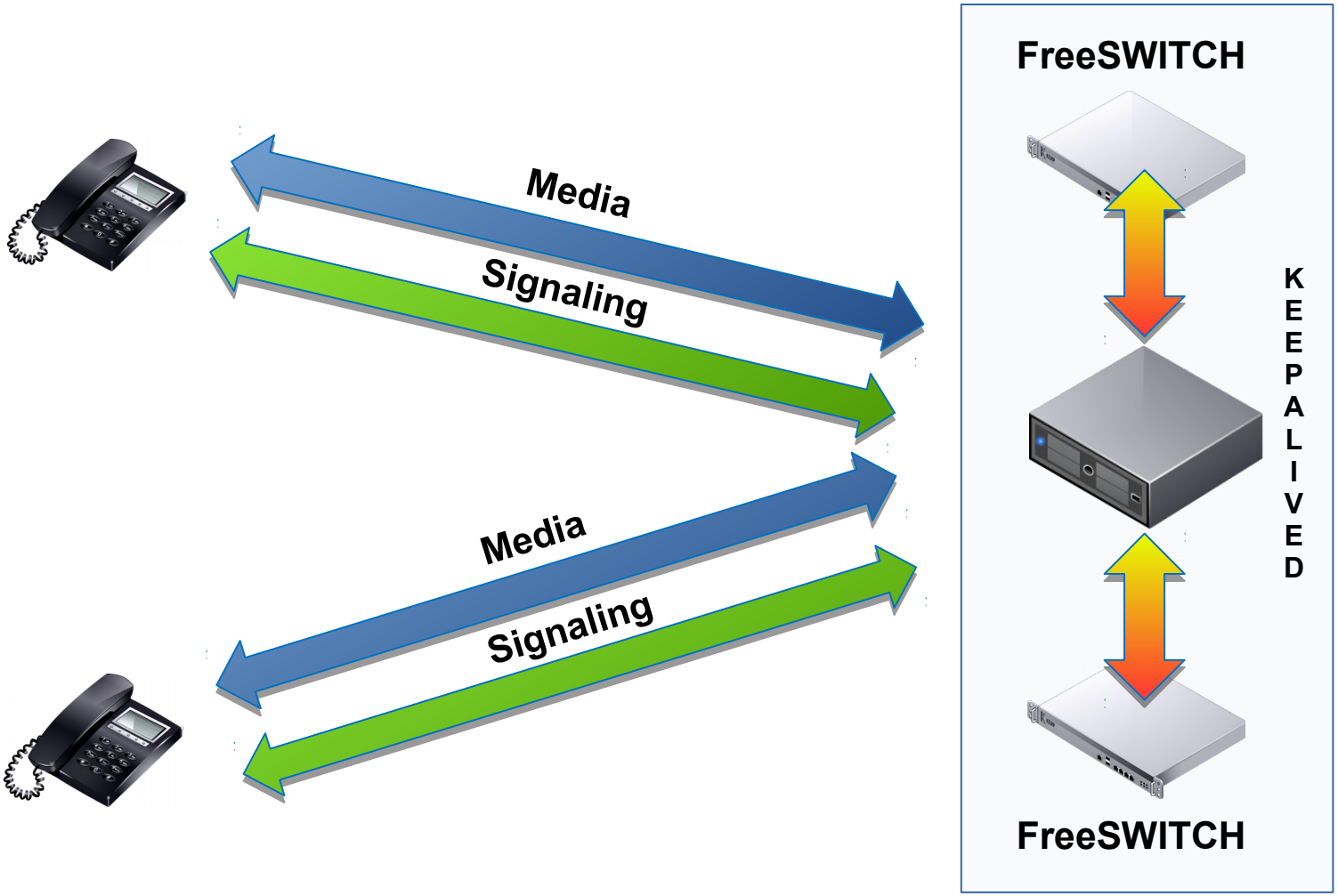
# High Availability: Double it All

- LAN Switch and Cabling
- FreeSWITCH Server
  - Virtual (Floating) IP address
  - HeartBeat, Keepalived, Corosync
- File System
  - DRBD
  - Rsync
  - GlusterFS
- Database
  - On FileSystem (SQLite)
  - Master-Master (Active-Passive)

# Two Single FreeSWITCHes

- Two Single FreeSWITCHes: **ACTIVE - PASSIVE**
  - Rsync or DRBD or GlusterFS:
    - Has its own Configuration
    - Keeps its own State
    - Writes and Reads Voice Mail
  - Manages NAT Handling (Media and Signaling)
  - Mixes Conference Participants' Media
  - Parks and Unparks Calls
  - Manages Queues and ACDs

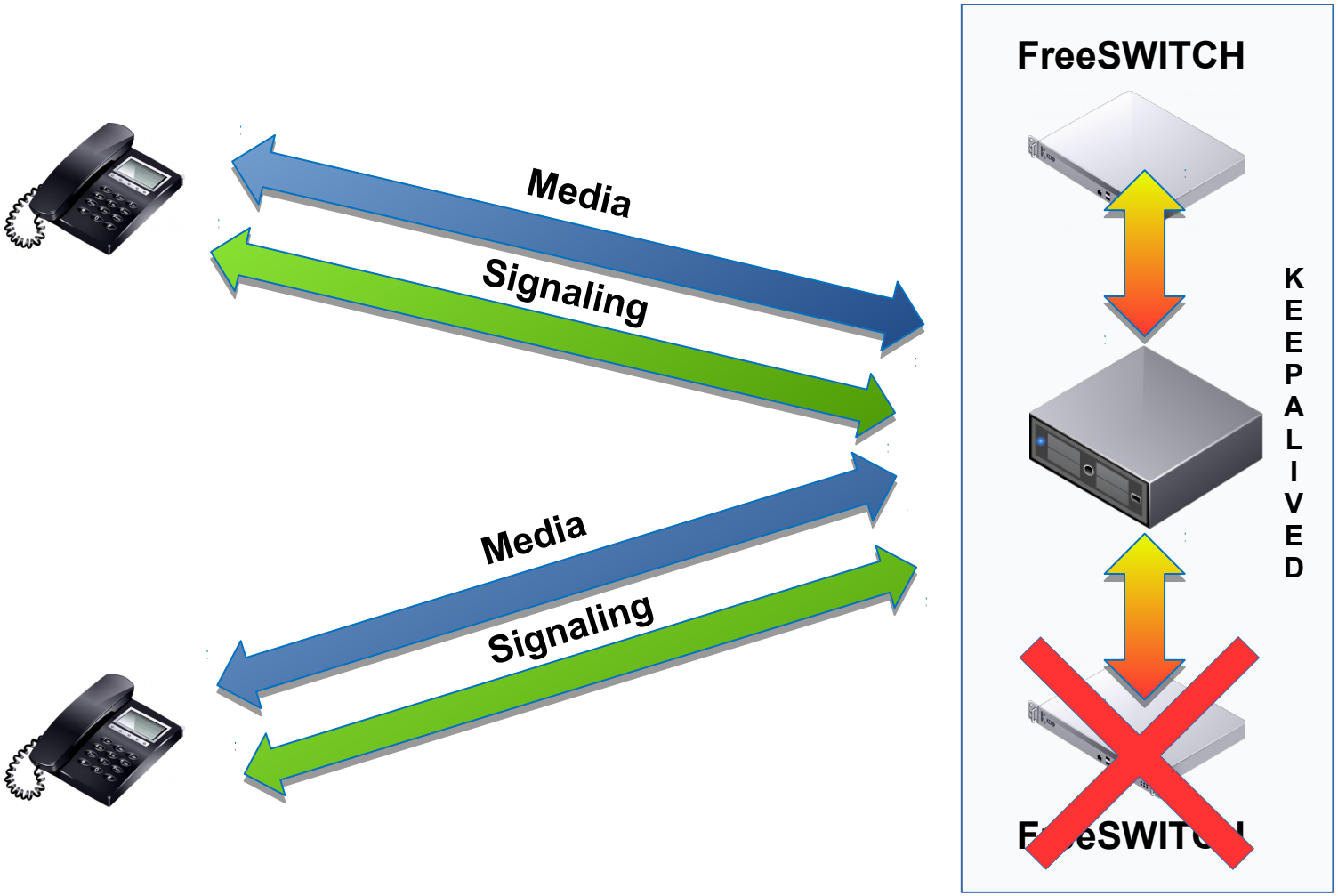
# Two Single FreeSWITCHes



# Two Single FreeSWITCHes

- One FS Machine is **Constantly IDLE**
- BIG FS IDLE = $$$
- After a while you don't know if it will work at all
- You will probably start using the IDLE machine for some small things, and then…
- Scales Only Vertically = $$$$$

# Two Single FreeSWITCHes



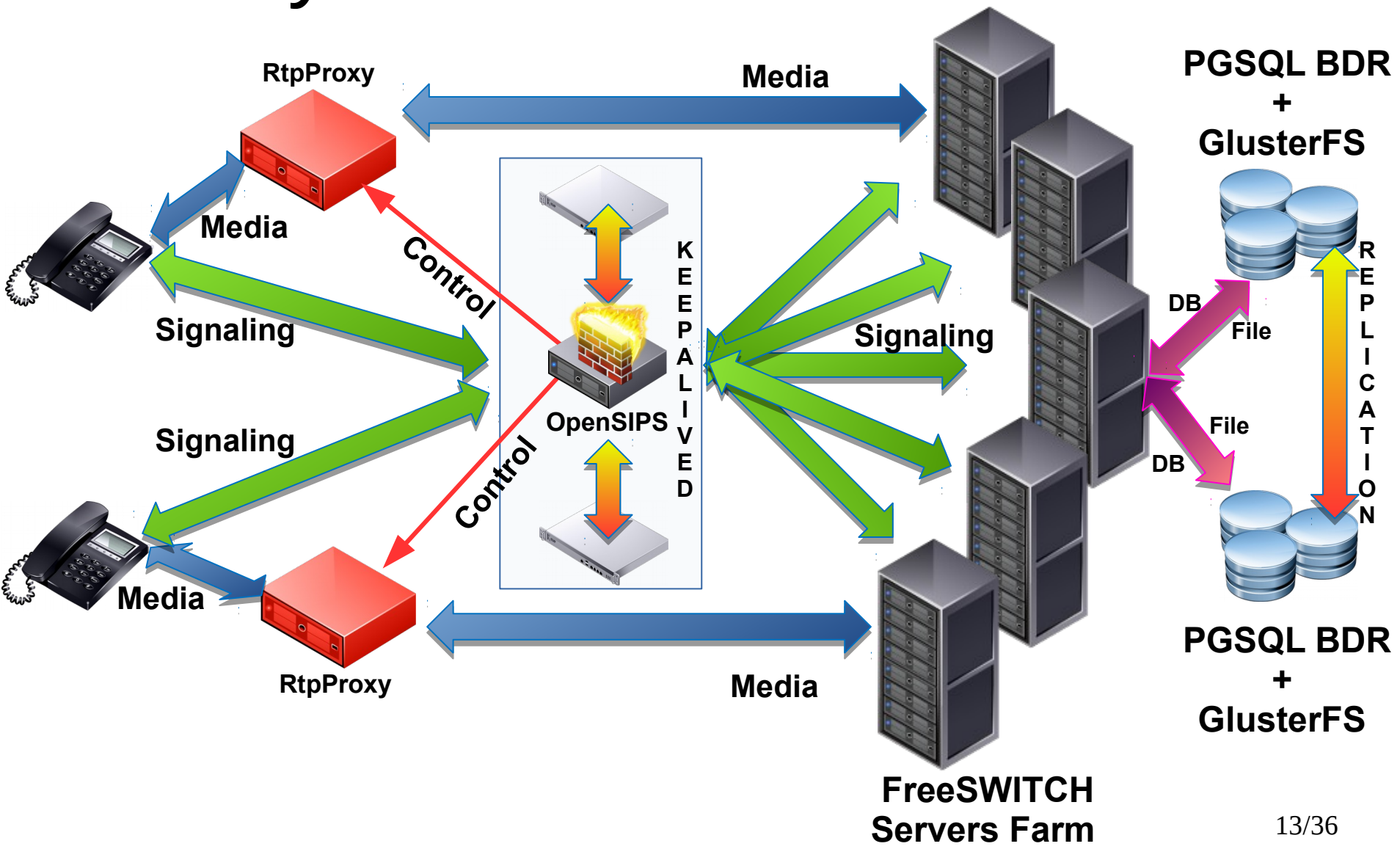
# Horizontal Scalability

- Distribute requests on two or more resources:
  - Registrations
  - Calls
  - Transcoding
  - Voice Mail
  - PSTN Termination and/or Origination
  - Audio Conferencing
  - Video Muxing
  - IVR
  - Web Pages
  - Web Sockets

# Many FreeSWITCHes as ONE

- ALL FreeSWITCH Boxes are **ACTIVE**
  (and most other boxes are too)
  - HA Database:
    - Keeps its own State
  - Distributed FileSystem:
    - Has its own Configuration
    - Writes and Reads Voice Mails
  - HA Load Balancers and Proxies:
    - Manages NAT Handling (RTP Media and SIP Signaling)
  - Partitioning (with Failover):
    - Mixes Conference Participants' Media
    - Parks and Unparks Calls
    - Manages Queues and ACDs

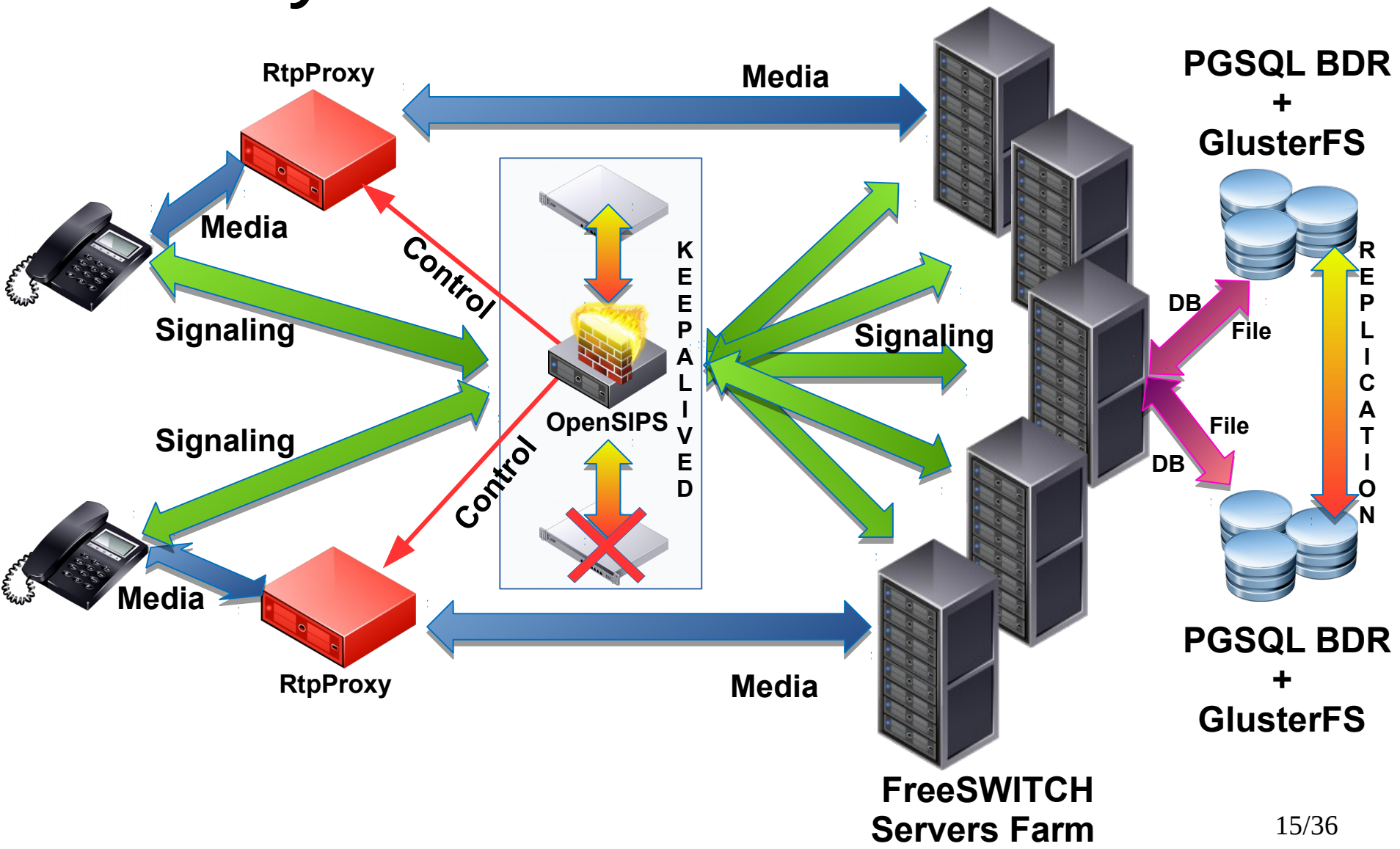
# Many FreeSWITCHes as ONE



# Many FreeSWITCHes as ONE

- One Load Balancer is **Constantly IDLE**
- LITTLE LB IDLE = ¢¢¢
- All is constantly exercised
- Scales Horizontally = ¢¢¢

# Many FreeSWITCHes as ONE



# IPv4 SIP Location and NAT

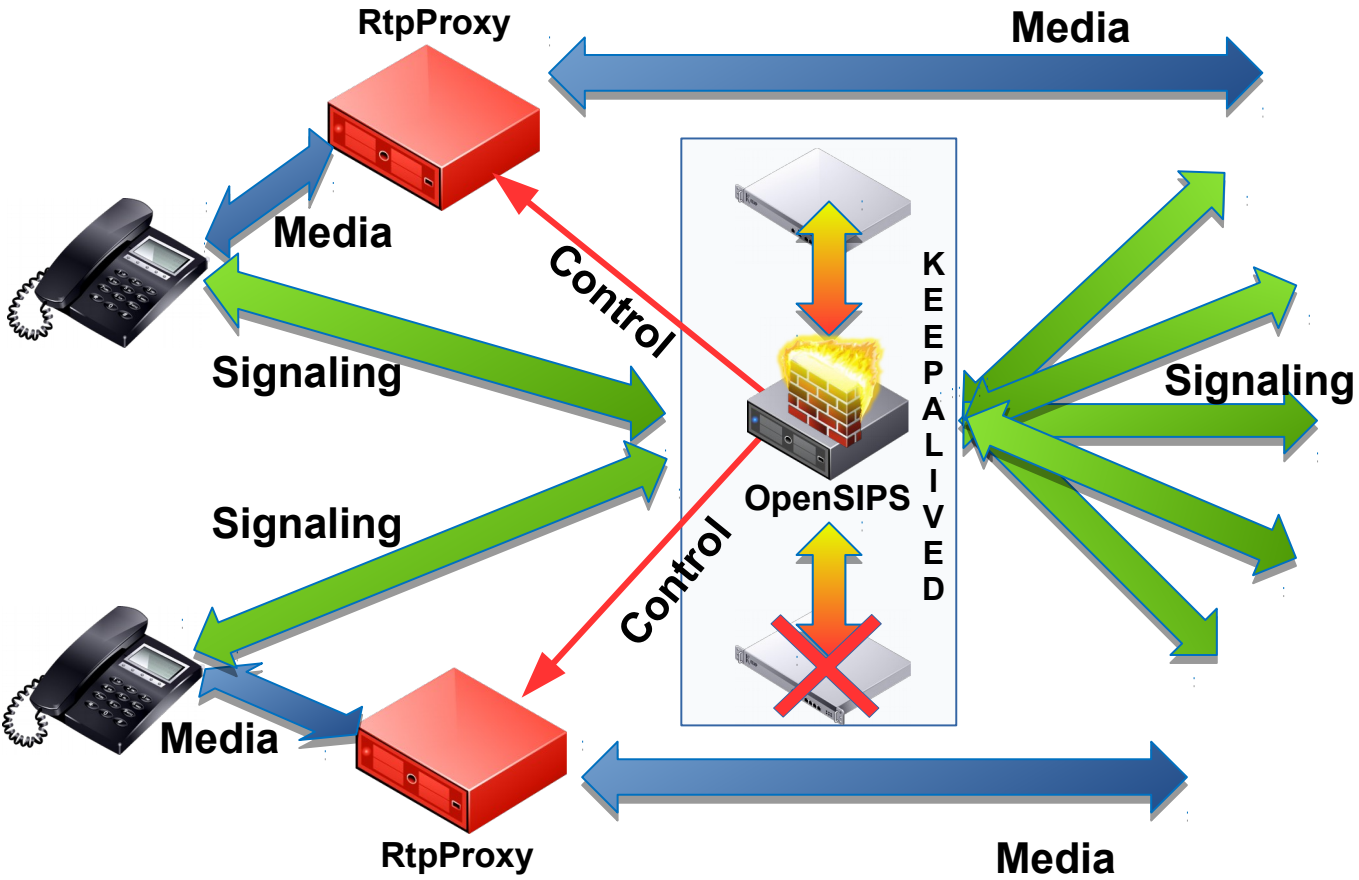
- Client is behind NAT
- Client sends from its own IP:port a REGISTER request to Location Server IP:port, and in doing so it opens a pinhole in the NAT, waiting for server's answer
- NAT pinhole is only able to receive packets from same IP:port couple (Client/Server) it was open by, and for a limited period of time (30 seconds?)
- Location Server sends periodically from same IP:port an OPTIONS message to Client IP:port, Client answers, and in doing so it maintains the pinhole open (FS sends each 23 secs)
- When there is an incoming call for Client, Server sends the INVITE from same IP:port to Client IP:port

# Where to put the REGISTRAR (Location Server)

- **ON LB (OPENSIPS) MACHINE**, directly interacting with Clients
  - REGISTER and NAT Keepalive (OPTIONS, NOTIFY) are high volume, low load transactions
  - One robust box (in active-passive HA) will be able to serve tens of thousands clients

- **ON FREESWITCH MACHINES**, load balanced by LB
  - FreeSWITCHes act as registrars, load balanced, all using the same database
  - This is the closest to ONE FREESWITCH machine
  - This topology scales indefinitely

# Load Balancing and Proxies



# Call Distribution: DISPATCHER & LOAD BALANCER

- OpenSIPS can use Dispatcher module for relaying (or forwarding – stateless mode) requests to multiple boxes using “static” algorithms (eg: round robin, or weighted)
- For “dynamic” algorithms, that take care of actual number of active calls, OpenSIPS uses LoadBalancer module
- Both modules, and all algorithms, are able to “ping” destinations, retry on failed destination, disable the failed box from list, and re-enable it when is back in order

# Call Distribution: KEEPALIVED

- Keepalived is a simple way to move a “Virtual” IP address from one Load Balancer server to another
- Virtual IP address will be the only published and accessed address
- Keepalived will check OpenSIPS is alive and working (eg, with sipsak) on the “primary” load balancer. If primary has failed, Virtual IP address will be moved to “secondary” (or “standby”) load balancer
- All other machines (clients, FS servers, etc) will not perceive any change

# NAT Traversing - Media Relaying

- There are special cases of clients behind NATs that cannot directly sends packets to each other. In those cases ONLY way for them to communicate is via the mediation of a server
- Also, you need to relay media in any case, if you're load balancing servers that are not directly reachable from clients

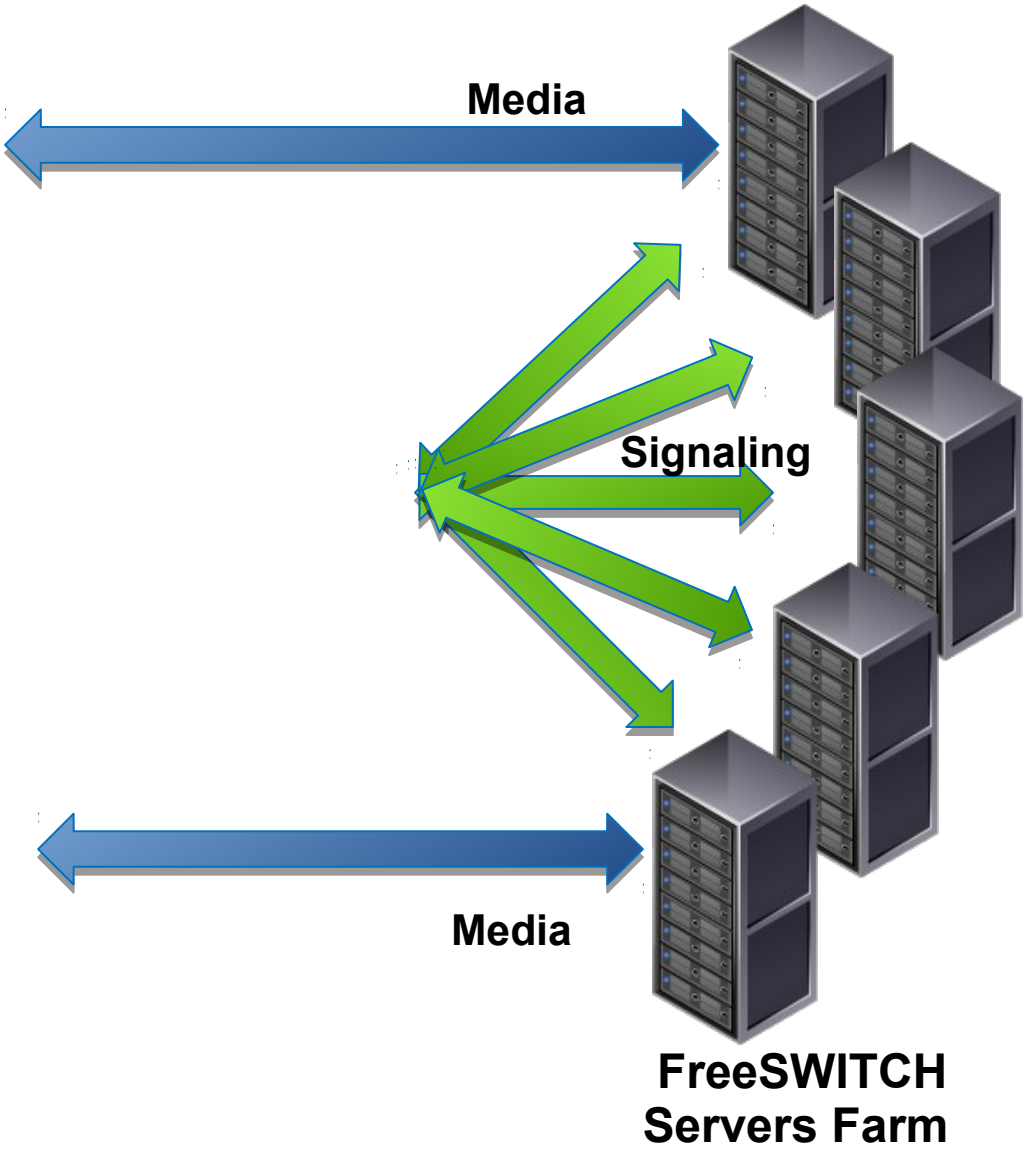
# Media Relaying

- OpenSIPS is a SIP signaling proxy, has nothing to do with media flow, it does not touch RTP
- It can modify SIP headers, and SDP bodies, so clients behind restrictive NATs will use a third party as a relay, and it can pass commands to that relay (eg: so the relay knows which client must be relayed to which)
- Original relay software is “Rtpproxy”
- More recent and advanced (eg: kernel space, etc):
  - Rtpengine (webrtc)
  - Mediaproxy
- All of them can scale indefinitely

Adapted from: https://txlab.wordpress.com/tag/xlab1/ pass-through proxy by Stanislav Sinyagin

```
route {
        force_rport();
        if (!ds_is_in_list("$si", "$sp"))
        {
            # SIP request packet client->backend
            if( !loose_route() )
            {
                if ( !ds_select_dst("1","0") )
                {
                    send_reply("500","No Destination available");
                    exit;
                }
            }
            if (nat_uac_test("19")) {
                if (method=="REGISTER") {
                    fix_nated_register();
                } else {
                    fix_nated_contact();
                }
            }
            add_path_received();
        }
        else
        {
            # SIP request packet backend->client
            loose_route();
        }
        if (method=="INVITE") {
              rtpproxy_engage("cw");
        }
        record_route();
        t_relay();
}
onreply_route {
        if (!ds_is_in_list("$si", "$sp"))
        {
            # SIP reply packet client->backend
            fix_nated_contact();
        }
    return(1);
}
                                                        163,1         Bot
```

# FreeSWITCHes' Farm



# FreeSWITCHes' Farm

- FreeSWITCH uses an internal database to keep state and persistance about SIP registrations, call states, etc
- By default, that database is kept on SQLite files in a local directory
- With PGSQL in CORE, and by setting mod_sofia, all the FS guts will reside in a remote PostgreSQL, shared by many FSes

# FreeSWITCHes' Farm

```
# fix core-dsn
vi /etc/freeswitch/autoload_configs/switch.conf.xml
-------------------------------------------------------------------------
 <param name="core-db-dsn" value="pgsql://hostaddr=127.0.0.1 dbname=free
switch user=postgres port=10001 password='' options='-c client_min_messa
ges=NOTICE'" />

-------------------------------------------------------------------------

# fix sofia-dsn
vi /etc/freeswitch/sip_profiles/internal.xml
-------------------------------------------------------------------------
<param name="odbc-dsn" value="pgsql://hostaddr=127.0.0.1 port=10001 dbna
me=freeswitch user=postgres password='' options='-c client_min_messages=
NOTICE' application_name='freeswitch'" />
-------------------------------------------------------------------------

                                                      638,0-1       31%
```

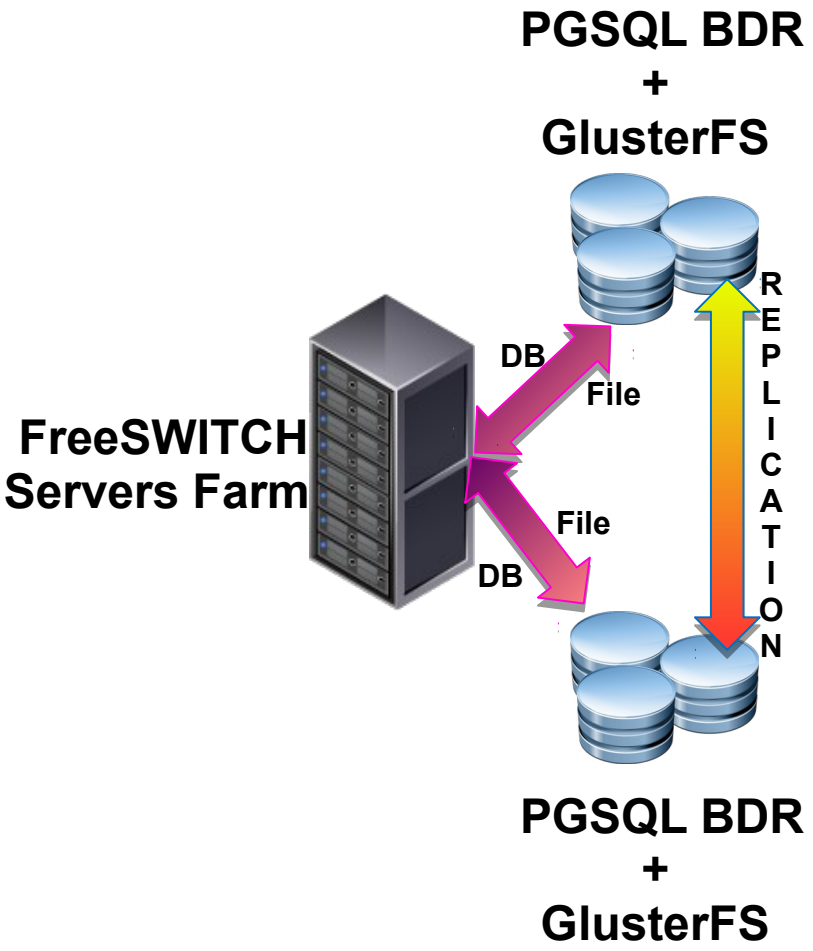
# FreeSWITCHes' Farm

- On each FreeSWITCH machine we put an HAProxy
- PostgreSQL will be accessed by HAProxy
- HAProxy wil automatically balance between PGSQL servers, and failover when needed

# FreeSWITCHes' Farm

- FreeSWITCH gets its own configuration from XML
- By default, that XML is kept in files in a local directory
- GlusterFS client permits to access that directory from many Fses (another way is to use mod_xml_curl to access XML via HTTP)
- VoiceMail metadata resides in DB, while actual audio messages are shared by GlusterFS

# **PERSISTENCE:**
# GlusterFS & PostgreSQL BDR

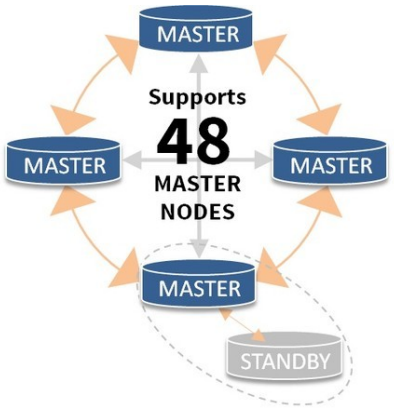


# GlusterFS

- GlusterFS is a distributed filesystem
- Gluster SERVERs “export” local BRICKs
- Gluster CLIENTs “mount” remote BRICKs
- Any modifications made by clients is automatically synched in realtime on all servers and all clients
- If a server fails, clients automatically failover in realtime to another server

# PostgreSQL BDR

- Bi Directional Replication (BDR) is a new addition by 2ndQuadrant to PostgreSQL. Is being integrated into mainline and will be in a future official release
- BDR allows for master-master low latency clustering
- BDR automatically replicate new tables and table modifications
- To use BDR you must have uniq Pks inserted (UUIDs)
- Two ways for doing that from FS



# Special cases

- Load Balancing is predicated on a server farm of equivalent and equipollent (eg: interchangeable) servers

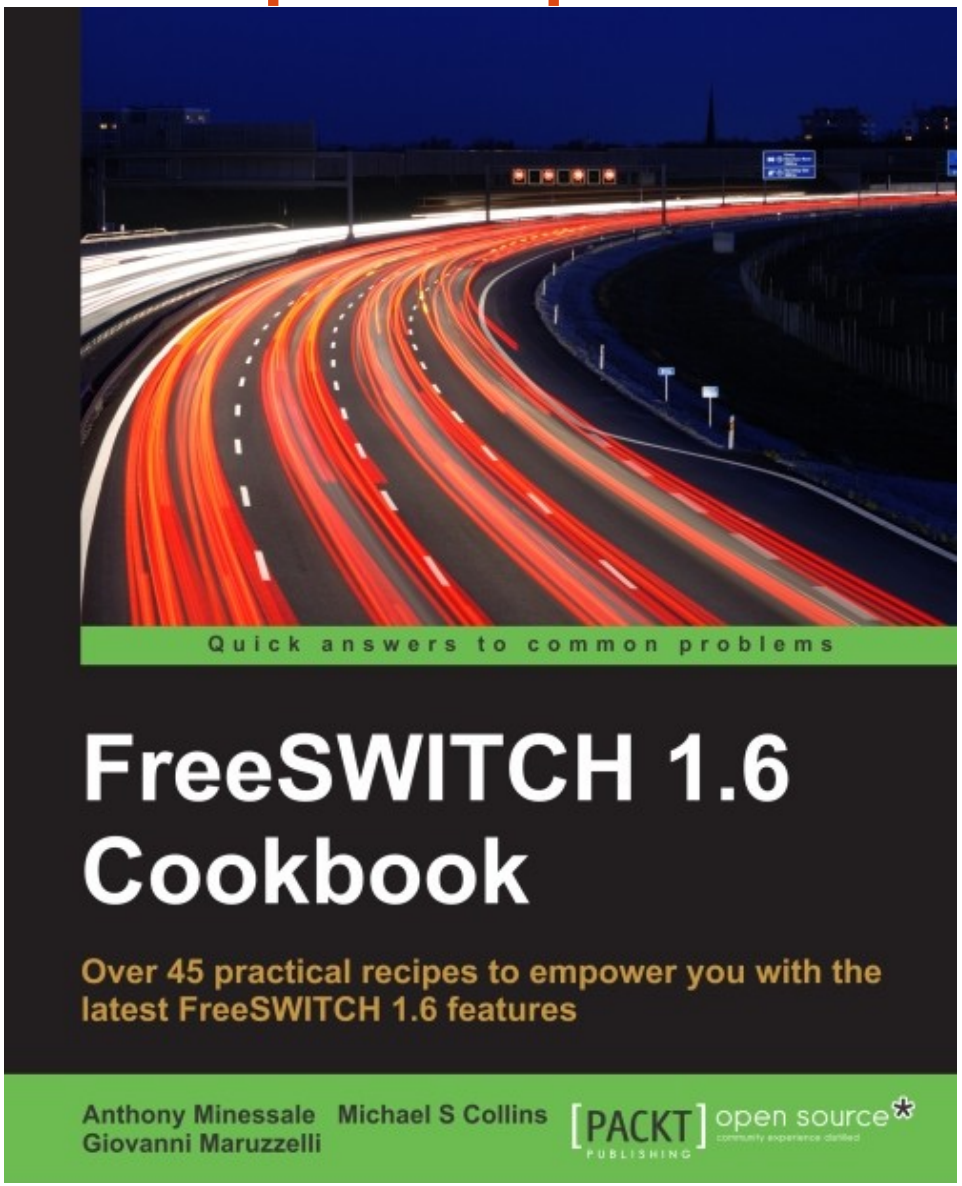
- There are cases for which this is not true:
  - Conferences
  - Call queues
  - Call centers

- **ANSWER IS: Partitioning!**

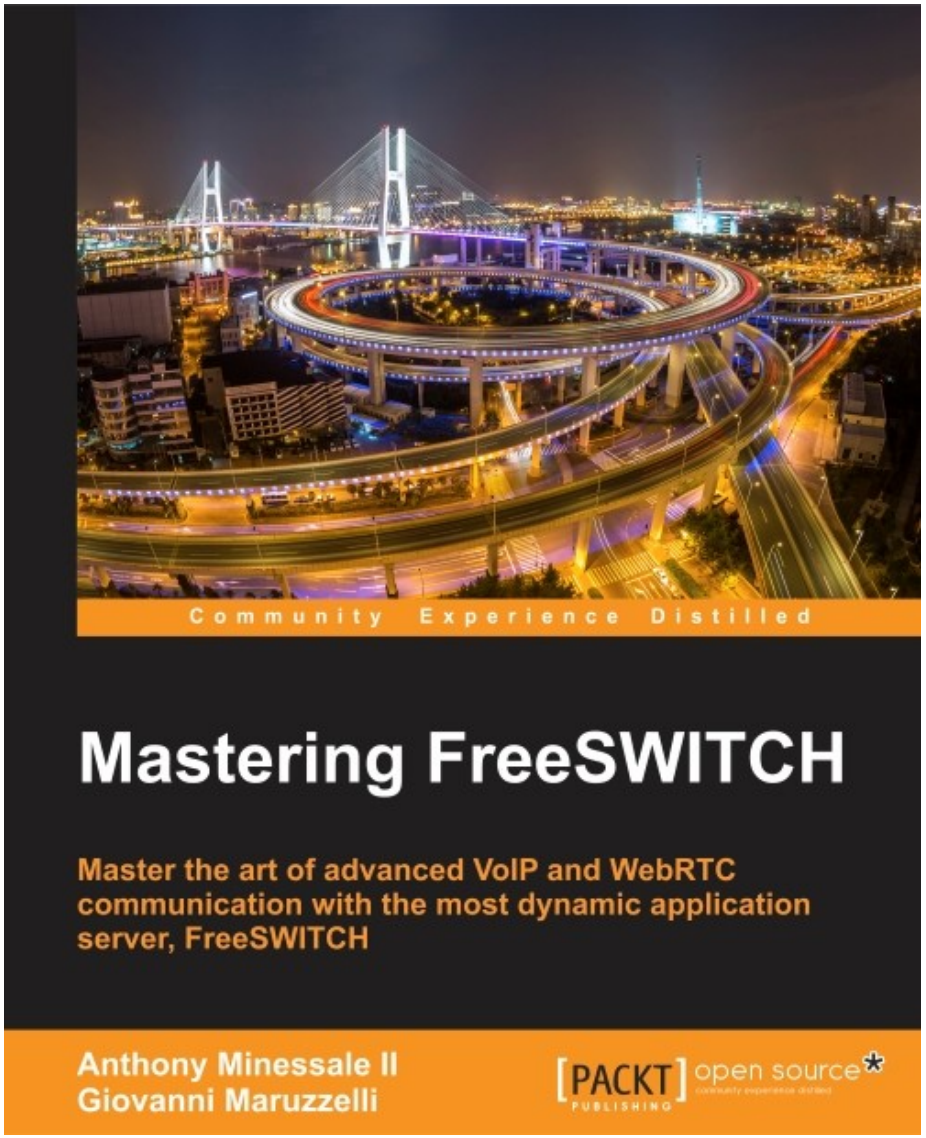
# Special cases

```
# CONFERENCES:
# hash on callid (0) dispatching on FreeSWITCH boxes in group "2"
# lesser priority box will be used only if previous boxes are all down
# eg: failover
# so, let's have only two machines in this group "2"
# first one will get all the conferences traffic, second one is failover
# you can put one of the active machines of group "1" as last in this ("2") group
        if($rU=~"^3[0-9][0-9][0-9]$")
        {
                if(!ds_select_dst("2", "0"))
                {
                        send_reply("403", "No destination");
                        exit;
                }
        }
# EVERYTHING ELSE (eg: NOT CONFERENCES):

# hash on callid (0) dispatching on FreeSWITCH boxes in group "1"
# if WITH_FREESWITCH_HA_CONFERENCES is active then
# lesser priority box will be used only if previous boxes are all down
# you can have the first machine of group "2" as last in this group
        if(!ds_select_dst("1", "0"))
        {
                send_reply("403", "No destination");
                exit;
        }

                                                        1537,1        76%
```

# www.packtpub.com



# www.packtpub.com



# Thank You

# QUESTIONS ?

**Giovanni Maruzzelli**
gmaruzz@OpenTelecom.IT