OpenSIPS as an IP-PBX replacement in a multi-sites environment

13 May 2015

About Us

About Us

- Be IP
  - Founded in 2008 from NOVACOM (2003)
  - Commercializes an IP PBX product based on OpenSIPS & Asterisk
  - Approximatively 15k users of our products in the BeLux
- Damien Sandras
  - Created FOSDEM in 2000
  - Created Ekiga in 2001 and Ekiga.net in 2005
  - Created NOVACOM in 2003
- Steve Frécinaux
  - Joined NOVACOM in 2007

AWIPH

AWIPH

- Governmental Agency
  - 700 users
  - 1 main site
  - 7 remote offices
- Specific Requirements
  - High-Availability
  - All Offices must be reachable at any time
  - Presence, Instant Messaging and Exchange Integration are important

AWIPH

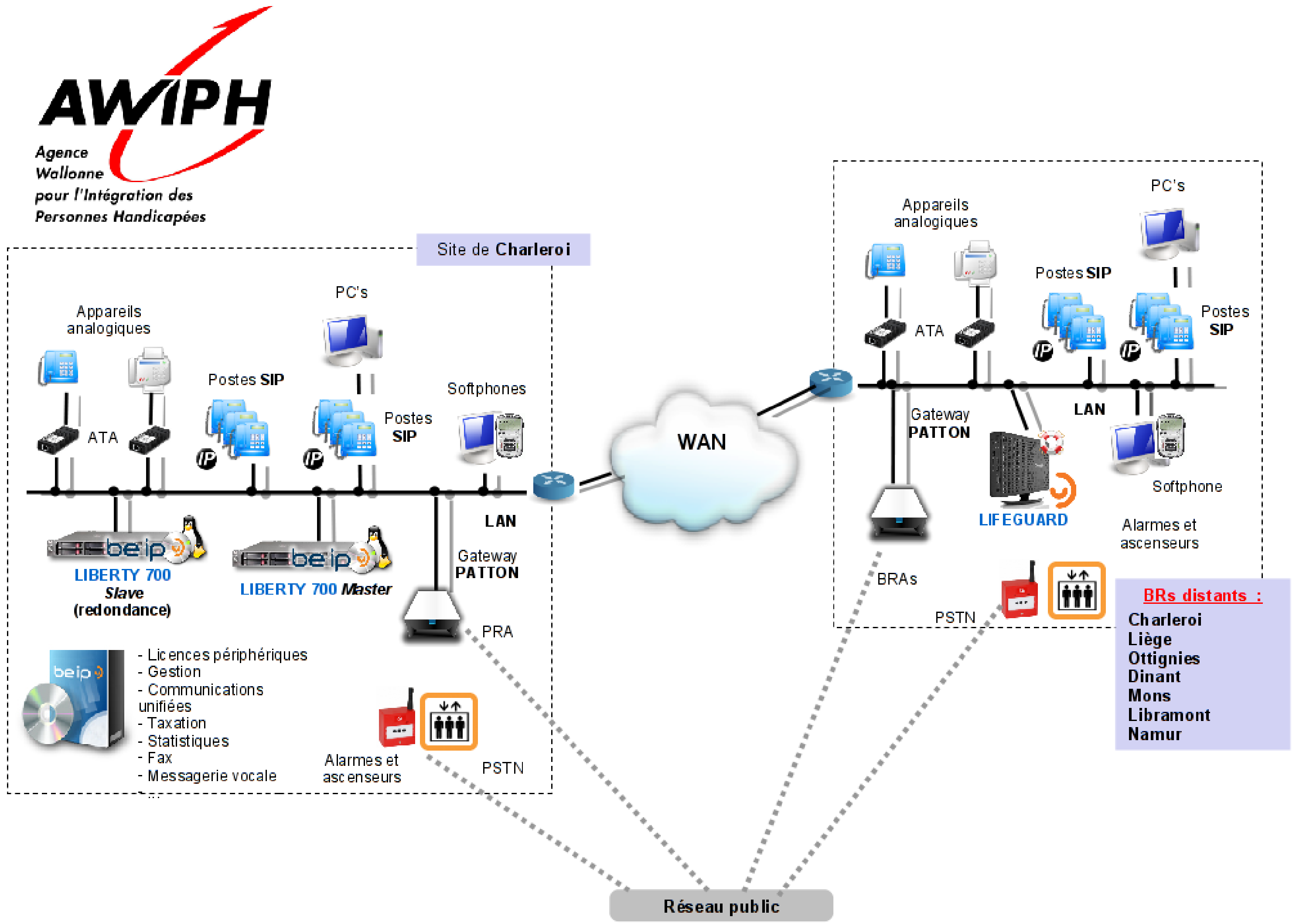

Schéma général projet téléphonie – A.W.I.P.H.
Document établi par Damien Sandras
Date : 16/08/2011

Architecture

Architecture: Requirements

- Our servers are still mostly on premises
- This means we have different constraints than cloud or ITSP operators do:
  - 10-2000 users, but really 10-100 most of the time
  - Hardware is “expensive”
    - We have very few servers available

Architecture: Requirements

- This means we have different constraints than cloud or ITSP operators do (cont'd):
  - Bandwidth is rare
    - 1 Mbps inter-site links are common
    - QoS guarantees are usually lame
  - Maintenance is “expensive”
    - Lots of servers to manage relative to the amount of users
    - Few economies of scale to benefit from

Architecture: Requirements

- This means we have different constraints than cloud or ITSP operators do (cont'd):
  - People expect traditional PBX features to be available
    - Directed Pick-up
    - Group Pick-up
    - Pick-up notifications
    - Boss / Secretary features
    - …
  - Different brands implement different features with different RFC's

Architecture

- A few words about our typical architecture
  - 2 “main servers” with
    - OpenSIPS 1.8 and Asterisk 1.4
    - A shared and redundant MySQL database used by OpenSIPS
    - A bunch of other services
  - Several “satellite servers” with
    - OpenSIPS 1.8 alone
    - A local MySQL database for the OpenSIPS data.
    - The local copy of active registrations is sync'ed every few minutes
    - Nothing else shared with the other servers
  - DNS SRV is doing the rest

Architecture: Asterisk

- Why Asterisk?
  - Historical reasons
    - We come from an Asterisk-only situation (back in 2003)
    - And Asterisk is still handling every single call
  - Some features are currently holding us back
    - Call history and statistics
    - Voice applications
      - Voicemail, IVRs, queues, …
      - Call recording
      - Group pick-up
    - RTP stream management, for trunks and NAT
      - Alleviate routing issues (somewhat like `rtpproxy`)
      - Can make transcoding easier, while making codec management harder

Architecture: Opensips

- Why OpenSIPS?
  - Provides nice extra features
    - Forking across several devices
    - Actually working presence and dialog-infos
    - TCP, SIP MESSAGE / MSRP, Called Number Display
  - Alterations possible at the SIP level
    - Asterisk manages calls, not SIP sessions and messages
    - `failure_route`, `reply_route`
    - e.g.: dynamic, reINVITE-aware call counting
  - Works around Asterisk deficiencies
  - More and more OpenSIPS, less and less Asterisk

Important Features

Important Features: Presence

- In the PBX world, presence means supporting “Busy Lamp Fields”
  - On the Phone, Free, Ringing (with directed pick-up)
- In the UC world, presence means “user availability”
  - Available, Away, Busy, …
- We need to support both modes / both worlds

Important Features: Presence

- SIP defines two types of events
  - Dialog-info
    - Several bug fixes and much debugging required
  - Presence
    - RFC 4480 - RPID/Rich Presence Extensions to PIDF
  - Presence Agent Implementation (or simulation)
  - `mix_dialog_presence`

Important Features: Presence

- Specific requirements
  - One unique presence state for several presence sources
    - Most SIP implementations do not handle aggregated documents very well
    - Most Human Brain implementations do not understand aggregated presence very well
  - If calendar integration is enabled, its presence state must “win”
  - Calls need to be routed according the presence state

Important Features: Presence

- With specific requirements (cont'd)
  - Presence needs to be shared among multiple SIP servers
    - Clients can be split 50/50 across the servers
  - Phones implement different things … differently:
    - SNOM phones support PUBLISH but not RFC 4480 (old im: tag)
    - Polycom phones do not support PUBLISH
    - Sofpthones usually support more things

Important Features: Presence

- Our implementation
  - Uses `pua_usrloc`
    - For SIP UAs that do not support PUBLISH
  - Uses `mi_xmlrpc` and `pua_mi`
    - For web-published or calendar presence status or unaware devices
  - Uses `cachedb_sql`
    - To store the unique presence state
  - Adds 3 settings to OpenSIPS
    - `merge` → Use merge instead of aggregation
    - `im_to_rpidf` → Converts im: into Rich PIDF
    - `merge_primary_source` → Specifies what is the primary source

Important Features: Presence

- How does the merge algorithm work?
  - If the presence document contains
    - *dialog-info* related information → this presence state wins
    - `pua_usrloc` generated information → this presence state is considered as the least important one
  - If the presence document contains
    - presence information identified as originating from the primary source → this presence state wins
  - Otherwise
    - the most recently PUBLISHed presence states wins
  - The result is NOTIFYed when appropriate to SUBSCRIBERs and stored in cacheDB for reuse in the call routing

Important Features: Multi-Sites

- Several sites with several network profiles
- OpenSIPS
  - Rejects registrations from unknown networks
  - Handles call counting and call limits
    - Using dialog profiles
    - From the first initial INVITE to the final BYE, including reINVITEs

Important Features: Redundancy

- DNS-based redundancy
  - Each of our servers embed a DNS server
  - This DNS server is authoritative on a “tel” DNS zone:
    - *; A record to the main server, for dumb endpoints*
      tel.beip.be. IN A 172.30.42.1
    - *; NAPTR record*
      tel.beip.be. IN NAPTR 10 10 "S" "SIP+D2T" "" _sip._tcp.tel.beip.be.
    - *; SRV records to be used by SIP endpoints*
      _sip._tcp.tel.beip.be. IN SRV 10 50 5060 laurel.tel.beip.be.
      _sip._tcp.tel.beip.be. IN SRV 20 50 5060 hardy.tel.beip.be.
    - *; Individual servers*
      laurel.tel.beip.be. IN A 172.30.42.11
      hardy.tel.beip.be. IN A 172.30.42.12

Important Features: Redundancy

- Each remote site has a LifeGUARD server
- LifeGUARD servers are really dumb
  - No Asterisk instance → No local voice applications
    - No call pickup, no queues, no intercom, no music on hold, no voicemail, ...
    - Call transfer are supported.
  - As few dynamic knowledge as possible
    - No presence, pretty much only registrations
    - Configurable redirections are not honored (busy, no answer, etc)

Important Features: Redundancy

- LifeGUARD servers are really dumb (cont'd)
  - Overly simplified call routing
    - Direct desk phone numbers only
    - Any unknown number is redirected to a local operator

      (Unknown means "not a direct desk phone number")
    - Only a single (local) trunk is supported
  - Embeds a DNS server and a redundant DHCP server
    - But still, no provisioning!
    - Please don't reboot your desk phone.

Important Features: Redundancy

- This server is made available through DNS:
  - Extra DNS records for each LifeGUARD server
    - _sip._tcp.tel.beip.be. IN SRV 30 50 5060 lg-mons.tel.beip.be.
    - lg-mons.tel.beip.be. IN A 192.196.203.2
  - We make use of bind9 views.
    - At most one LifeGUARD server shows up in the DNS answer, depending on the source IP of the request.
    - The local server has a local copy of the DNS zone, to avoid timing out on DNS queries.

Important Features: Redundancy

- Sharing data among servers is difficult
  - It must be kept in sync between the servers
  - It consumes bandwidth
  - It can generate conflicts and break
- So we'd really like to share nothing
  - But we need to know about registrations and stuff
  - LifeGUARDs only know the bare minimum
- MySQL replication is prone to failure
  - Custom script synchronizes registrations periodically with the master servers, both ways
  - OpenSIPS Binary Interface looks very promising

Important Features: Redundancy

- Choosing the transport protocol
  - Most of the time, SIP uses UDP as its transport layer
  - We chose to use TCP instead, though
    - With UDP, some phones (Snom) tend to hang forever while waiting for an hypothetical SIP responses in this scenario
    - TCP handshake guarantees a (somewhat) quick failure if a server is unreachable
  - TCP support in OpenSIPS sometimes made our lives difficult
    - Bad performance on slow network lines due to blocking connections
      - we had to increase the number of processes a lot
    - Patch which disables the restriction on shared NOTIFYes
      - we need to be able to open a TCP connection if there isn't one already

More Features

- Other features handled by OpenSIPS
  - Direct Call Pick-up
  - Caller ID Name
  - Call Forwarding (internal vs external calls)
    - On busy (on the phone)
    - On offline
    - On no answer
    - Depending on the presence state
  - Callback-on-busy
  - Distinctive Rings

More Features

- Other features handled by OpenSIPS (cont'd)
  - Cellphone integration (in terms of BLFs)
  - Asterisk related
    - Asterisk failures
    - Problems due to multiple Asterisk instances
      - Consultative transfer with one call on one server, the other call on the other server
      - Group pick-up
  - Instant Messaging
    - MESSAGE
    - MSRP

Conclusion : The Future

The Future

- Main Goal
  - Get rid of Asterisk when & where possible
- Short Term TODO
  - Migrate to a more recent OpenSIPS release
    - Use the new events framework for our Call Events feature
    - Use the new binary interface to get rid of the MySQL redundancy
      - Presence will be one difficult point
    - Implement WebRTC

The Future

- Short TODO (cont'd)
  - Move more features from Asterisk to OpenSIPS
    - CDR handling
    - Call Recording handling
    - Codec management
    - Implement group pickup in OpenSIPS
  - Share more infrastructure among cloud customers
    - We started with an “on premises” solution
    - Multi-domain
    - Routing data partitioning
- See you next year!

Questions?

beip

www.beip.be