Asterisk 12 and PJSIP
Asterisk and PJSIP

Asterisk’s PJSIP channel driver: a SIP architecture for the future

The future is now!
Asterisk and SIP: A History

- Why write a new SIP stack?
- RFC 3261 – SIP: Session Initiation Protocol
  - June 2002
- chan_sip:
  - r472 | markster | 2002-06-28 15:34:46 -0500 (Fri, 28 Jun 2002) | 2 lines
    Version 0.1.12 from FTP
- That's 12 years ago!
No Facebook, Twitter, or even MySpace

- Social network: friendster
  - This is now a gaming site?

Linux landscape was changing

- RedHat releases first version of RHEL
  - May 6 2002: RHEL 2.1 AS (Pensacola)

https://fedoraproject.org/wiki/History_of_Red_Hat_Linux
http://redhat.com
We still cared about Blackberry

- Blackberry 5810 (March 4, 2002)

http://www.techhive.com/article/172837/the_mobile_phone_a_history_in_pictures.html
Asterisk was very new

- Two asterisk-users mailing list emails still exist
- One is unsubscribe

---

January 2002 Archives by thread

- Messages sorted by: [subject] [author] [date]
- More info on this list...

Starting: Wed Jan 16 01:02:31 MST 2002
Ending: Wed Jan 16 01:02:31 MST 2002
Messages: 1

- [Asterisk-Users] unsubscribe jave

Last message date: Wed Jan 16 01:02:31 MST 2002
Archived on: Tue Sep 5 15:25:22 MST 2006

- Messages sorted by: [subject] [author] [date]
- More info on this list...

This archive was generated by Pipermail 0.09 (Mailman edition).
chan_sip: a long and venerable life

- Architecture was never designed for its current size
  - 0.1.12 – 1950 lines
  - Trunk – 34570 lines

- Current structure limits change
  - No stack
  - Large monolithic architecture

- Venerable, yet time to retire
Example: Inbound INVITE request
Example: Inbound INVITE request

Put the request in a threadpool for processing.
Example: Inbound INVITE request

```
res_pjsip => res_pjsip_endpoint_identifier_user
res_pjsip::distributor
PJSIP

Identify the endpoint
```
Example: Inbound INVITE request

Application

Dialog

UA/Proxy Layer

Transaction Layer

res_pjsip_nat

res_pjsip => res_pjsip_endpoint_identifier_user

res_pjsip::distributor

PJSIP

See if we need to change anything due to NAT settings
Example: Inbound INVITE request

If this is a re-INVITE, update media state (it's not, move on)
Example: Inbound INVITE request

Application

res_pjsip =>
res_pjsip_authenticator_digest

Authenticate the request

Dialog

res_pjsip_session

UA/Proxy Layer

res_pjsip_nat

res_pjsip =>
res_pjsip_endpoint_identifier_user

Transaction Layer

res_pjsip::distributor

PJSIP
Example: Inbound INVITE request

Transaction Layer

res_pjsip::distributor
res_pjsip_endpoint_identifier_user
res_pjsip_nat
res_pjsip_authenticator_digest

UA/Proxy Layer

res_pjsip_session
res_pjsip_nat
res_pjsip_endpoint_identifier_user
res_pjsip::distributor
PJSIP

Application

Make a new session
Example: Inbound INVITE request

**Application**
- `res_pjsip_sdp_rtp`
- `res_pjsip_session`
- `res_pjsip => res_pjsip_endpoint_identifier_user`
- `res_pjsip_nat`
- `res_pjsip_authenticator_digest`

**Dialog**
- `res_pjsip_session`

**UA/Proxy Layer**
- `res_pjsip_nat`
- `res_pjsip => res_pjsip_endpoint_identifier_user`

**Transaction Layer**
- `res_pjsip::distributor`
- `PJSIP`

Process the offer (if we have one)
Example: Inbound INVITE request

**Application**

- res_pjsip_caller_id
- res_pjsip_sdp_rtp
- res_pjsip_session
- res_pjsip => res_pjsip_authenticator_digest

**Dialog**

- res_pjsip_session

**UA/Proxy Layer**

- res_pjsip_nat
- res_pjsip => res_pjsip_endpoint_identifier_user

**Transaction Layer**

- res_pjsip::distributor
- PJSIP

Extract caller ID and store it
Example: Inbound INVITE request

Make the ast_channel object

```
chan_pjsip
res_pjsip_caller_id
res_pjsip_sdp_rtp
res_pjsip_session
res_pjsip =>
res_pjsip_authenticator_digest
```

```
res_pjsip_session
```

```
res_pjsip_nat
res_pjsip =>
res_pjsip_endpoint_identifier_user
```

```
res_pjsip::distributor
PJSIP
```
### Example: Inbound INVITE request

**Transaction Layer**
- `res_pjsip::distributor`
- `res_pjsip_nat`
- `res_pjsip_endpoint_identifier_user`
- `res_pjsip::distributor`

**UA/Proxy Layer**
- `res_pjsip_session`
- `res_pjsip_sdp_rtp`
- `res_pjsip_caller_id`
- `res_pjsip_authenticator_digest`

**Application**
- `res_pjsip_t38`
- `chan_pjsip`
- `res_pjsip_caller_id`
- `res_pjsip_sdp_rtp`
- `res_pjsip_session`
- `res_pjsip => res_pjsip_authenticator_digest`

**Dialog**
- `res_pjsip_session`

See if we need to do anything with T.38 fax state (nope!)
Example: Inbound INVITE request

Start the PBX!

Transaction Layer

PJSIP

res_pjsip::distributor

UA/Proxy Layer

res_pjsip_endpoint_identifier_user

res_pjsip_authenticator_digest

Dialog

res_pjsip_session

res_pjsip_sdp_rtp

res_pjsip_caller_id

res_pjsip_t38

Application

chan_pjsip

chan_pjsip

res_pjsip

res_pjsipNat

res_pjsip_session

res_pjsip =>
res_pjsip_authenticator_digest

res_pjsip_session
Configuration

- **Sorcery: Data Abstraction Layer**
  - ORM (for some values of O and R)
  - Supports CRUD operations
  - Well defined lifetime, thread-safe, reload-safe
  - Prune realtime peers/reloads

- **Smaller objects**
  - Takes advantage of templating/databases
  - Can change storage location of each object
    - Example: Configuration data in database; contacts in AstDB
  - Simple mapping to in-memory representation
    - Defer higher level concepts to systems on top of Asterisk
Configuration Example

[alice]
type=endpoint
context=internal
allow=!all,g722,alaw
auth=alice_auth
aors=alice_aors

[alice_auth]
type=auth
auth_type=userpass
username=alice
password=as8918hd!@8hs19a1m

[alice_aors]
type=aor
max_contacts=10
http://www.markthalle-in-hannover.de/p_73_bistro
http://shop.brewforia.com/browse-by/style/german-pilsner
Asterisk 12 and Kamailio

environment

; sorcery.conf
[res_pjsip]
endpoint=realtime,ps_endpoints
aor=realtime,ps_aors
; extconfig.conf
ps_endpoints => odbc,asterisk
ps_aors => odbc,asterisk
Asterisk 12 and Kamailio

Alice

Bob

Internet

Kamailio 1

Kamailio 2

Database

Asterisk (IVR)

Asterisk (Queues)

Asterisk (VM)

noload => res_pjsip_authenticator_digest.so
noload => res_pjsip_registrar.so
What's Next?

- Distributing Presence
Asterisk 12 and Kamailio: Next Steps

- Alice
- Bob
- Internet
- Kamailio 1
- Kamailio 2
- Database
- Asterisk (IVR)
- Asterisk (Queues)
- Asterisk (VM)

MWI State
Presence State
Asterisk 12 and Kamailio: Next Steps

Kamailio 1

Kamailio 2

Asterisk (IVR)

Asterisk (Queues)

Asterisk (VM)

Database

Internet

MWI State

Presence State
Asterisk 12 and Kamailio: Next Steps

Asterisk (IVR) -> Kamailio 1 -> Kamailio 2 -> Asterisk (VM)

Alice
Bob

Internet

MWI State
Presence State

Database
What's Next?

- What happens if we need more of some resource?
  - Not always easy to change purpose of an Asterisk server
  - Major constraining point: dialplan

- Alternative: ARI
  - Treat Asterisk as an application engine
  - Push all dialplan logic out of the Asterisk instances
Asterisk: Things on the horizon

- **Testing**
  - 390 Unit Tests
  - 433 Functional Tests
  - 2.1x more tests than Asterisk 11
  - Lots more to go

- **PJSIP: Enhance and Extend**

- **Publish/Subscribe**
  - Highly desired
  - RLS

- **ARI: enable application logic outside of Asterisk**