Federated VoIP Switching

Beyond traditional VoIP Switching

Presented by: Nir Simionovich, GreenfieldTech

SIP long and prosper…

1931 - 2015

General Information

- **Name:** Nir Simionovich
- **Location:** Israel
- **Age:** 40
- **Involved in Open Source:** Since 1995
- **Involved in Open Source VoIP:** Since 2001
- **Key Interests:** Asterisk, Kamailio, VoIP Security, Fraud Analysis, Single Malts and other alcoholic beverages
- **My Projects:**

phpapi

Humbug
TELECOM LABS

VoIP On Tap
Meetup
Where VoIP goes social

VoIP Starfleet

Asterisk

FreeSwitch

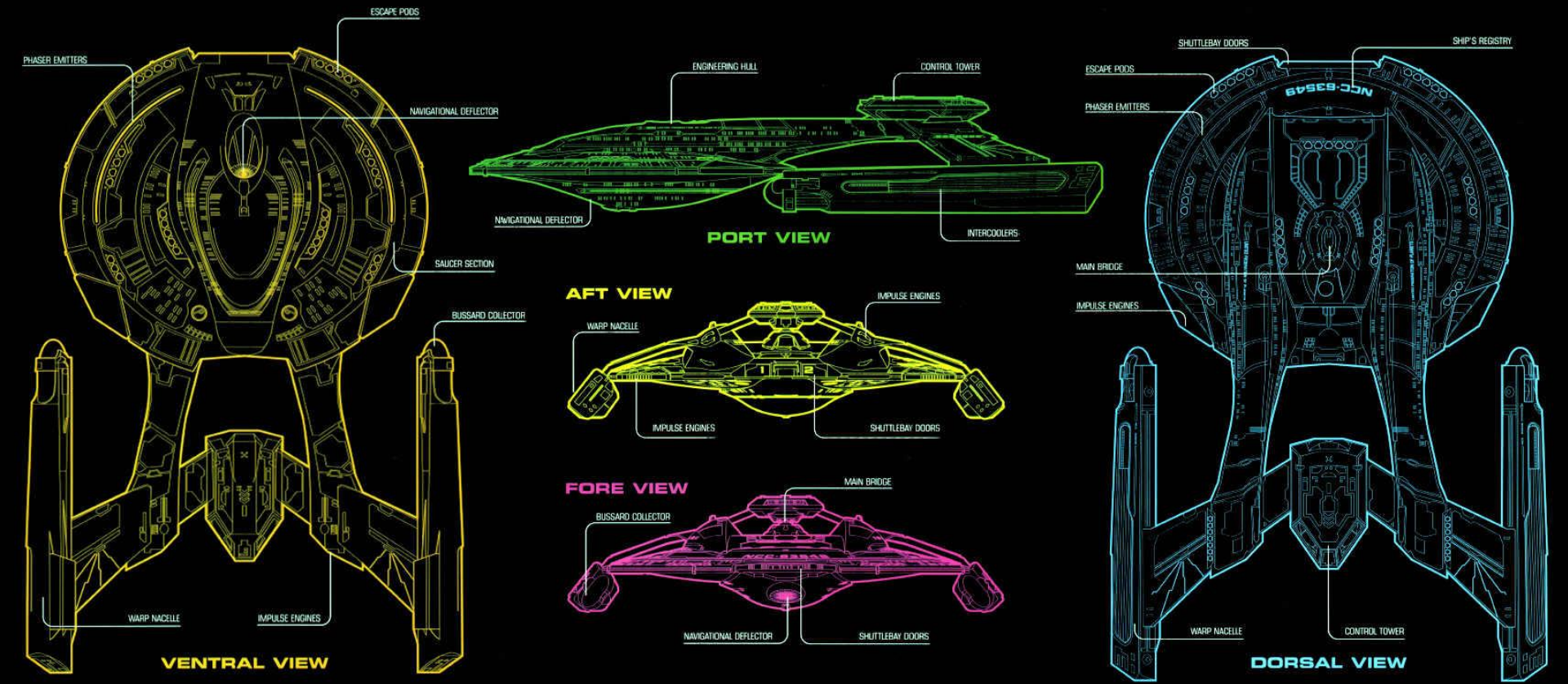
Yate

Kamailio

OpenSIPS

Mobicents

Each has it's own strengths



What does this have to do with VoIP?

- Everything !
- Each engine is very much like it's own starship:
  - Powerful
  - Feature Rich
  - Flexible
  - Had it's own charismatic captain
- And they are all capable of working together – to a degree

Do these guys work well in teams?

Different UI – Similar Functionality

The Issues

VoIP Scalability is mostly Vertical

- More Channels
- More Calls per second
- More servers
- More Session Border Controllers
- More Databases
- More …
- More …
- More …

The more you grow…

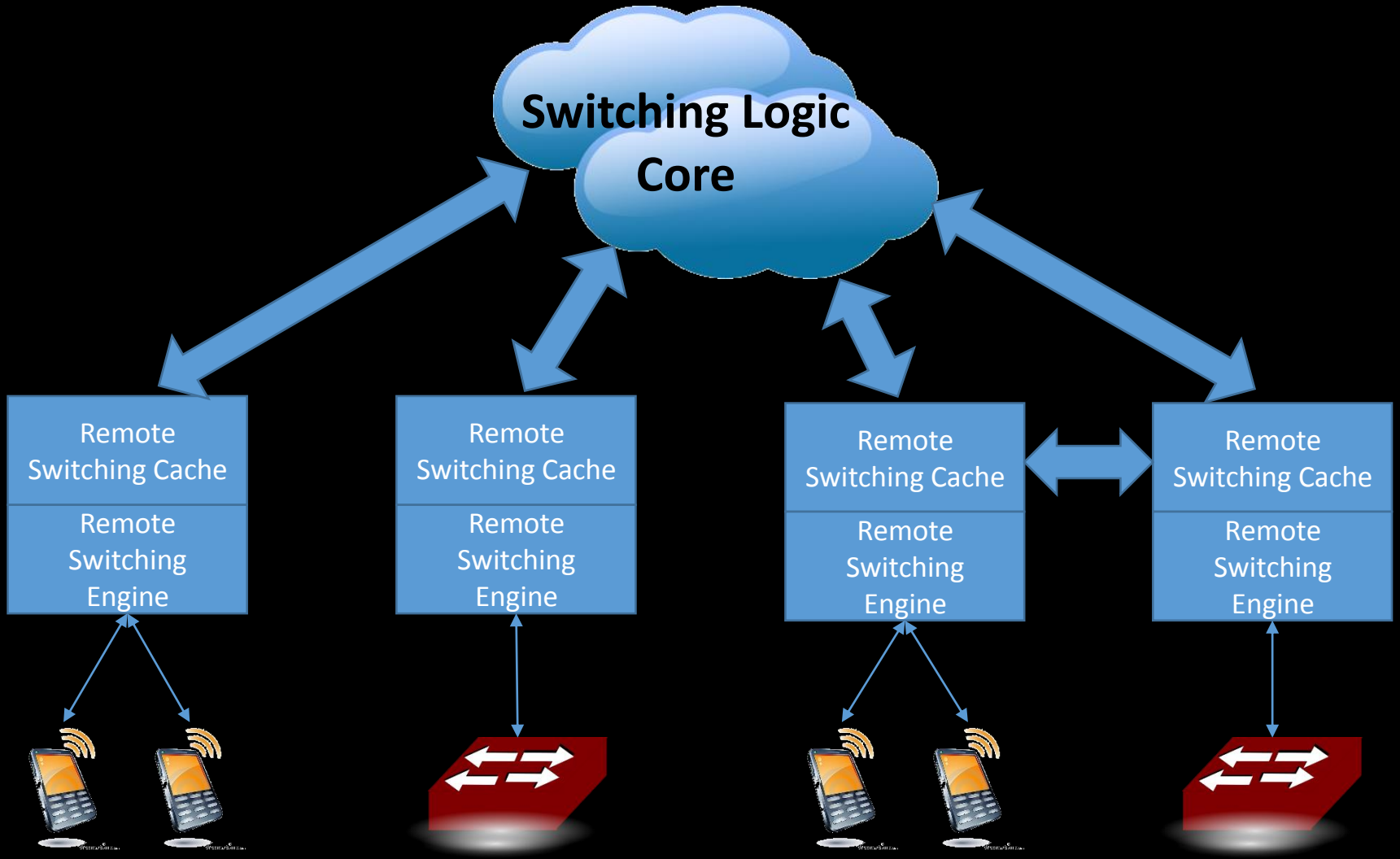
- More management
- More work
- Increased points of failures
- More personnel
- More money
- More money
- Much more money

There is a saying in telecoms…

What is Federated VoIP Switching?

- Cloud based switching logic core
- Discrete functional units
- UA Centric, not Switch Centric
- Distribution via provisioning, not via sharding
- No single point of failure
- “Starship” Agnostic

What does it look like?



Step 1: Provisioning

- User Agents have no knowledge as to their designated serving switch
- Server Designation is determined ad-hoc
- Upon a server failure, clients will re-distribute to other servers
- Re-distribution is performed using the same provisional interfaces
- Data sharding is not required – sharding occurs naturally

Step 2: Routing Logic

- No Static Routing !
- Seriously, no static routing what so ever !
- Routing is determined ad-hoc, created and removed as required
- All routing decisions are made outside of the switching fabric
- Inbound and outbound routing is done from singular locations (for PSTN purposes only)

Step 3: Transcoding and Media

- In an ideal implementation, Transcoding is performed at the PSTN edges only!
- Media services (voicemail, conferencing, IVR, etc.) are performed where the call is – not where the call goes to
- One codec to rule them all – and I truly don't mean g711 (personal favorite, Opus)

Step 4: Fault Tolerance and DRP

- "Your system is only as good as your infrastructure" – Russell Bryant, Astricon 2014
- A federated system doesn't require DRP or High availability
- A federated system is fault tolerant to failures of hardware and software – simply by its design
- A federated system is PaaS/IaaS ready, thus, you don't really need to deploy your own

How did we build ours?

Routing Logic, Provisioning, Billing, CRM

Kamailio, Asterisk, STUN, TURN

Data Warehousing

Analytics and Ops

Taking care of Business

Things you need to build…

- Client Provisioning
  - Integrated into the UA itself
  - Via normal provisioning interfaces
  - JSON API
- Server provisioning
  - Ad-Hoc server bootstrapping
  - Ad-Hoc server scrubbing
- Routing and Authorization
  - Ad-Hoc Route construction and distruction
  - Ad-Hoc Route decision making
  - Ad-Hoc service creation

Questions ?

Please stick around for the Birthday Capella Sing-along