Modern VoIP in Modern Infrastructures

Designing and implementing VoIP architectures in the cloud and container era

Federico Cabiddu - RTC Architect @ Libon
Giacomo Vacca - RTC Architect @ Nexmo

About us

Giacomo Vacca

VoIP since 2001, worked for Truphone, Libon and others.

Currently SIP infrastructure at Nexmo.

User, promoter and contributor of open source projects in the RTC field.

I design and maintain solutions based on Kamailio, FreeSWITCH, Asterisk, RTPEngine, Janus, Homer.

Federico Cabiddu

Working in VoIP since 2001

VoIP/RTC passionate

Libon Voice Team Tech Coordinator

Kamailio developer

Sipcapture team member

Overview

- Evolution of infrastructures used for VoIP
- How VoIP applications are impacted
- Some current workarounds
- A vision for the future

Today’s main focus: signalling. More on media, tools, debugging, QoS, Security in the future.

Why the Cloud?

- Customers/Partners expect it (sales are easier, e.g. with Direct Connect)
- HA more easily achievable
- Scalability
- Smaller upfront investment (pay more as you grow)
- Faster provisioning
- Easier geographic distribution
- Off the shelf advanced tools (HTTP LBs, HA, DNS mgmt, DB/cache, etc)
- New tools (containers, istio, etc)

What challenges come with the Cloud?

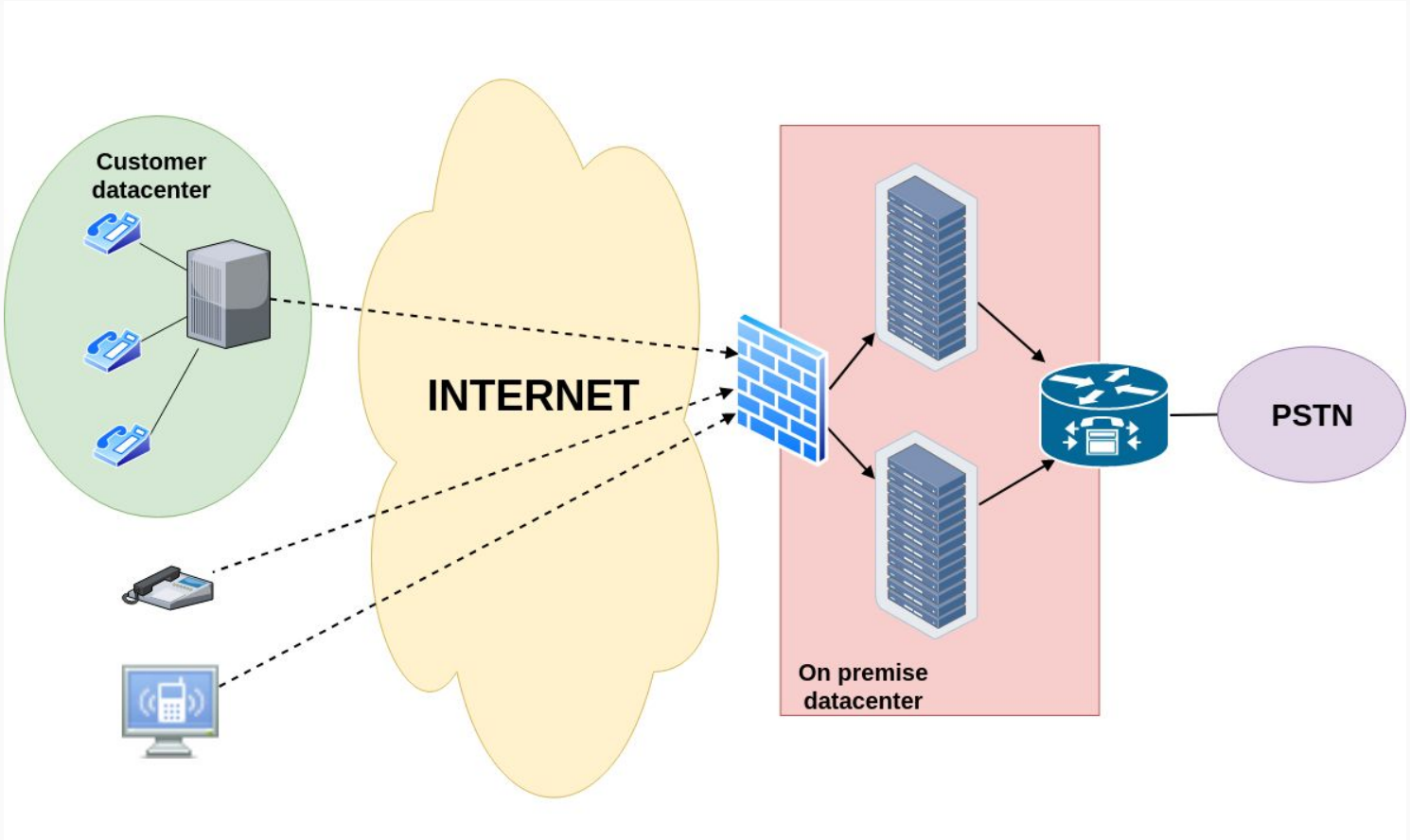
- Moving to a Cloud infrastructure typically requires a re-design
  - Choosing one provider is a critical task for any company
  - You need to take technical decisions that are not easy to change in the future
  - Can be hard to migrate from one provider to another
- Computing resources are not always dedicated
  - Hard to assess impact
- No standard solution for interconnection between providers
  - Network reachability and private addressing
  - VPNs to be maintained, etc
- Provider-specific tools to learn

Paradigm shift

From “Max uptime” to “Max resilience to restarts”.

From “Configuration updates” to “Immutable infrastructure”.

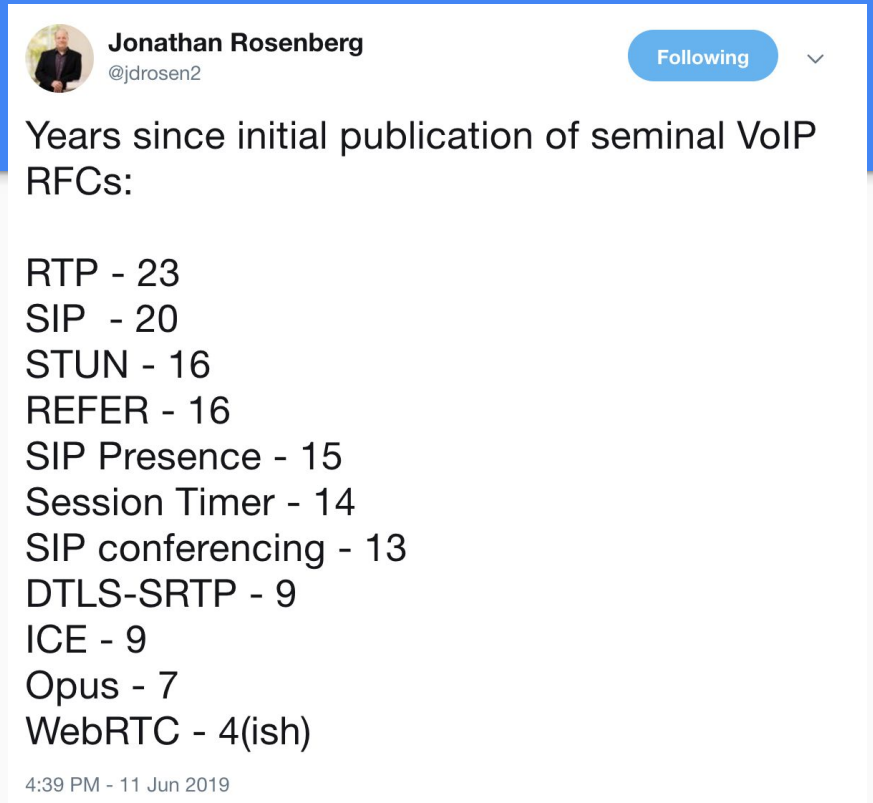
VoIP infrastructure - How it used to be



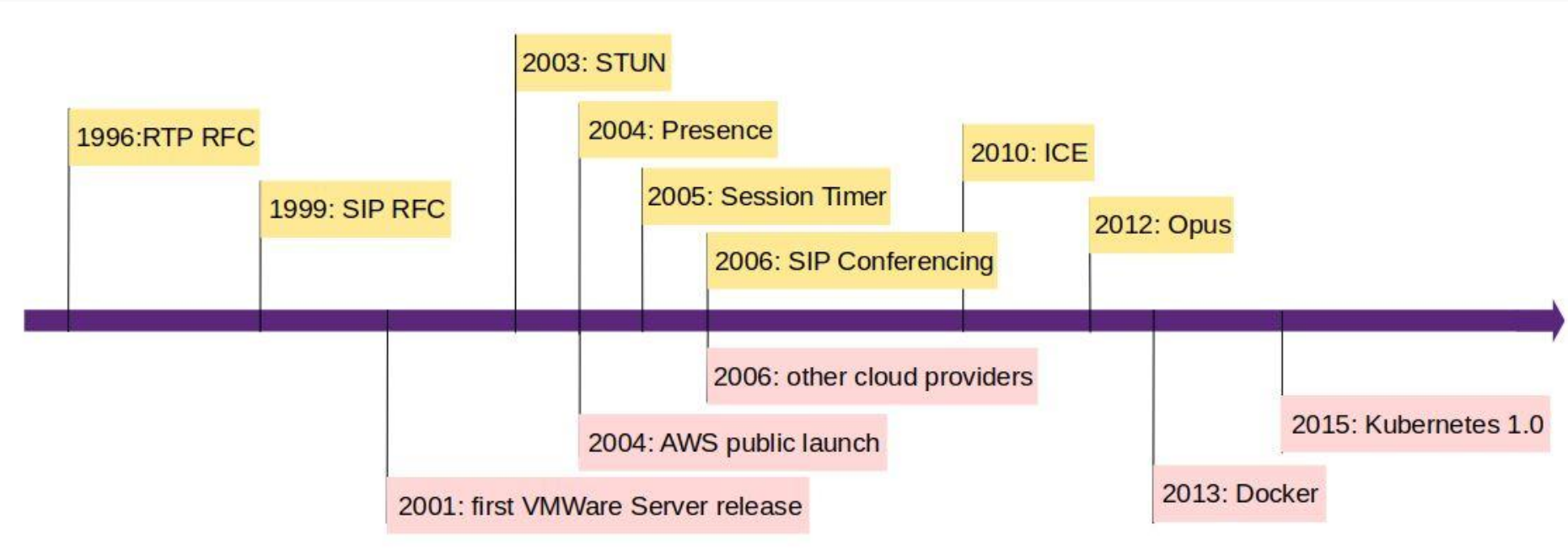
VoIP is “old”

Most of the protocols used in VoIP (e.g. SIP) pre-date modern infrastructures.

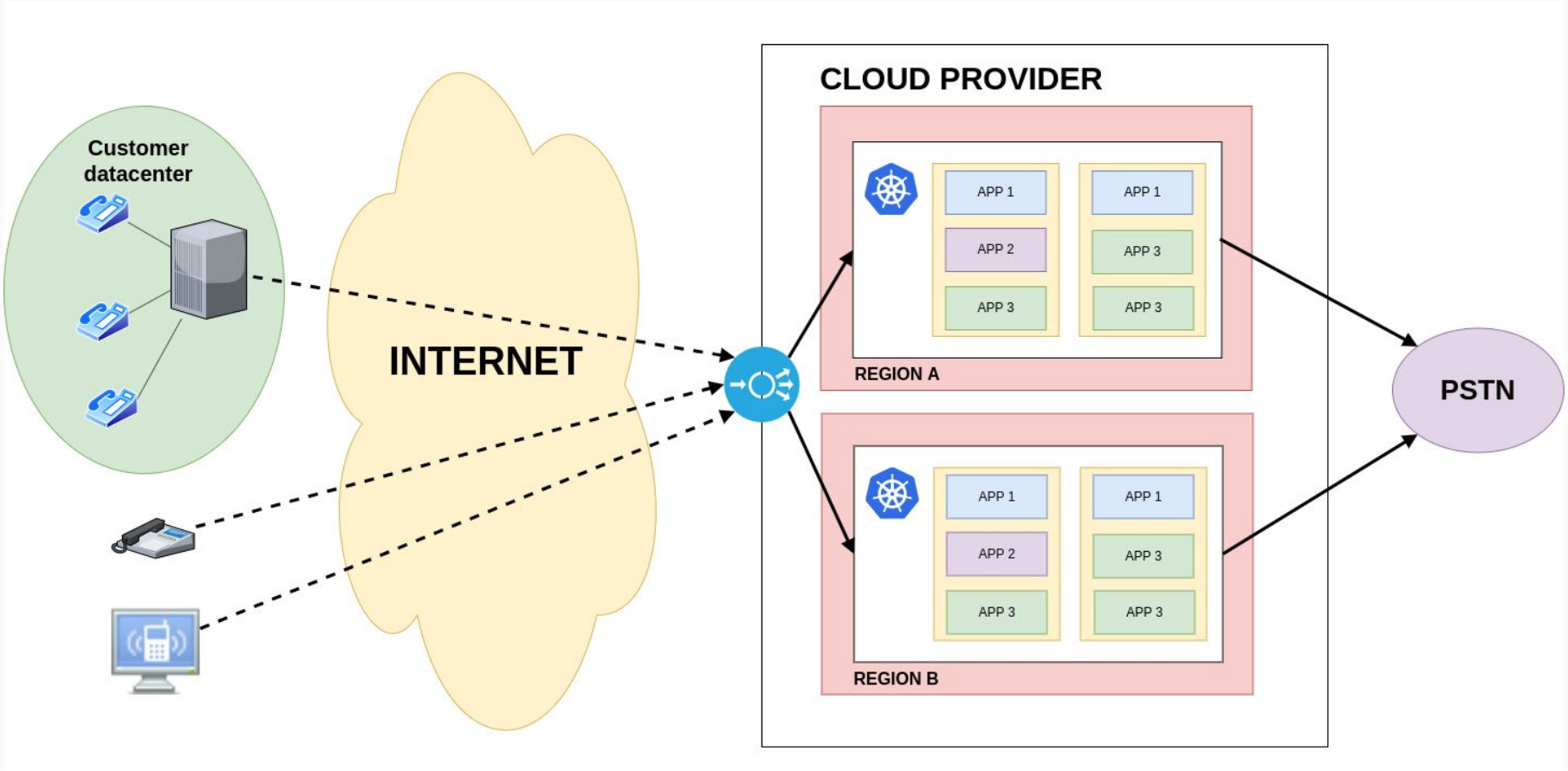
What’s changed throughout the years, but more importantly: how can we design VoIP networks that get the most out of today’s platforms?



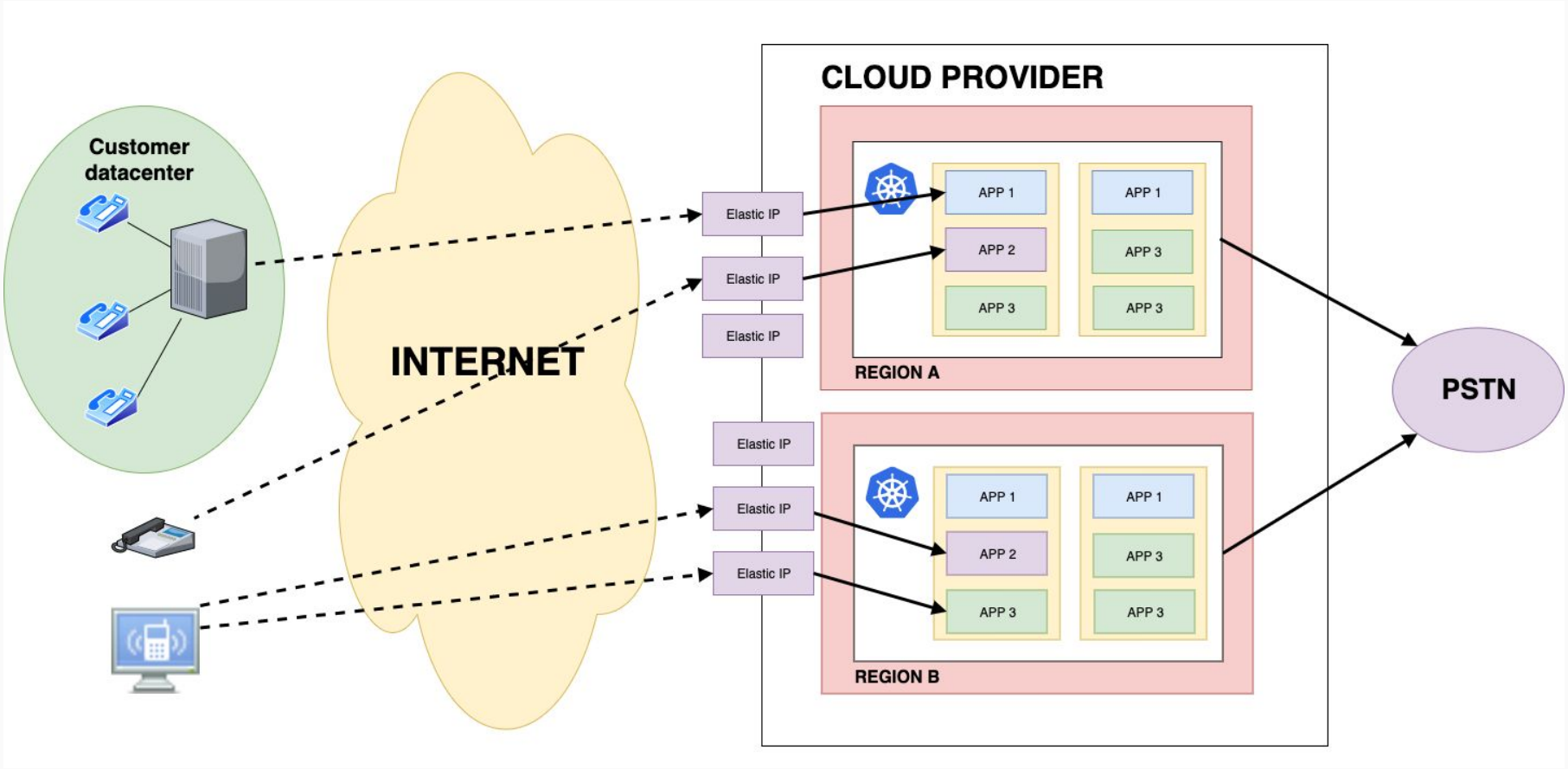
Protocols vs Infrastructure



VoIP infrastructure - How we'd like it to be



VoIP infrastructure - How it is now



What impacts specifically VoIP?

- IP addresses are ephemeral (Both public and private can be floating)
- NAT (no direct use of public IP addresses on hosts)
- No VoIP-aware components available off the shelf
- Packet rate vs Bandwidth (small packets)
- (Shared resources when high load? Transcoding, mixing, etc)
- Instances or containers are intrinsically ephemeral
    - Active calls, logs, traces, etc

VoIP sessions are sticky

vs HTTP request/response models, VoIP requires some sort of state.

Challenges with platforms that are designed to scale up/down continuously.

What’s the right balance between immutability and volatility?

IP addresses are ephemeral

For similar reasons to scaling up and down, modern platforms don’t work well when using IP addresses.

Internal routing benefits from DNS: use it and SRV as much as you can.

But then DNS must be fast and flexible.

No easy and general solution for RTP.

Lack of native components

- No “native SIP balancer”
- Network LB can be used as a partial solution
  - But challenges with UDP
- No easy solution for outbound SNAT/balancing

“OK, then just use a Load Balancer!”

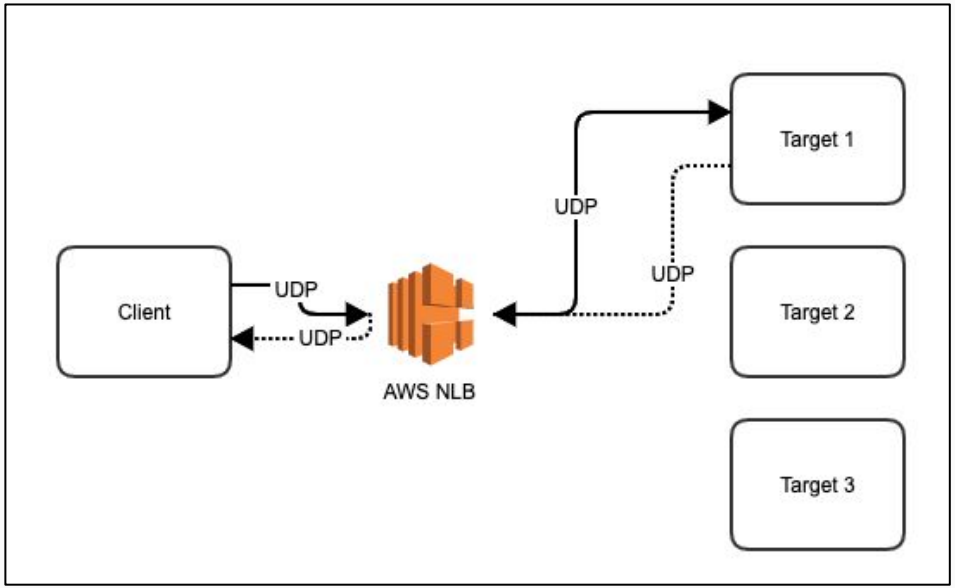
AWS ELB: HTTP only.

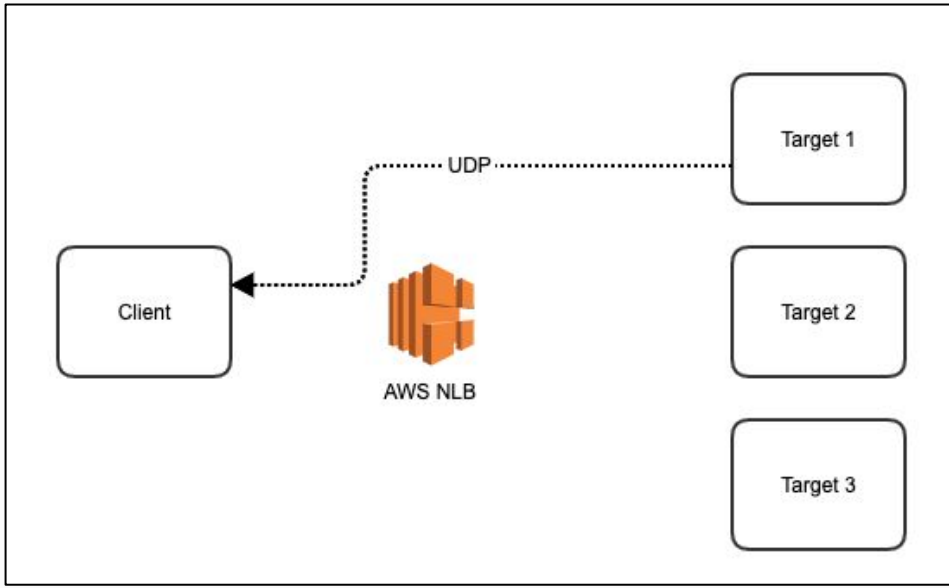
AWS NLB: UDP/TCP/TLS, **but don’t have the concept of “target” replacing another “target”.** UDP tracked connections expire. Balancing is based on source socket and not at SIP call level.

GCP NLB: TCP/TLS can work under certain limitations/constraints. UDP tracked connections expire.

Due to this dependency on the “stream” and not on VoIP calls, both GCP and AWS LB cannot be used for SIP over UDP.

Example: AWS NLB and UDP





Optimised for unidirectional (client to server) scenarios. Can’t work with server-generated messages.

Even AWS doesn’t seem to have a standard solution

“If SIP connections are initiated, another option is to use AWS Marketplace commercial off-the-shelf software (COTS). The AWS Marketplace offers many products that can handle UDP connections, and Layer 4 load balancing. These COTS typically include support for high availability and are commonly integrated with features, such as AWS Auto Scaling, to further enhance availability and scalability”

Reference: “Real-Time Communication in AWS” whitepaper, https://d1.awsstatic.com/whitepapers/Industries/Telco/real-time_communication_aws.pdf

Off topic - Media - “High bandwidth” is not enough

- VoIP uses small packets, but typically bandwidth is estimated with large (jumbo) packets
- Packet rate is critical
- Cloud services must provide tools to monitor packet rate saturation
- Do you think you know the packet rate limit of your infrastructure?

Off topic - Current Clouds don’t know about RTC QoS

- No specific stats on RTP, packet rate etc
- No control over the network
- Need to build your own tools (Homer?)
- Think about webrtc-internals in Chrome, but for servers

Debugging

- Enabling debug often means replacing a container.
- Are traditional tools (TCPdump, wireshark) still enough/useful?
- Hard capturing on a container (https://github.com/eldadru/ksniff)
- Is it still possible to troubleshoot live?
- Should debugging tools be part of the cloud offer?

Interconnecting Clouds

Often you can’t have everything on the same platform.

Interconnecting systems running on separate providers is not easy, and when supported may be expensive.

What do people usually do? VPNs? Special agreements? TLS over public?

Today’s workarounds

- Everybody builds their own SIP Load Balancer
- Direct management of floating IP addresses
  - Even private IPs are set as floating IPs just to keep them known and predictable
- Keepalives at L3 and Active/Stand-by with floating IP reallocation
  - Limitations in cross-AZ structures
- Drain scripts
- For media: dedicated nodes, hostNetwork

The Future?

- QUIC, HTTP/3, ...
- RIPP
  - Consider an extension for server-to-client (push notification + reverse connection)?

https://tools.ietf.org/html/draft-rosenbergjennings-dispatch-ripp-03

The reality

Protocols like SIP and RTP are old and don’t cope well with modern infrastructures.

But they are **not** going to disappear any time soon.

WebRTC has lowered the barrier of entry for new RTC services, which in turn increases demand for “traditional” VoIP solutions.

Questions for the long term

- How to design VoIP networks that get the most out of today's platforms?
- **Should the VoIP community work on a standard, open source solution for a VLB ("VoIP Load Balancer")?**
- Best practices for the cloud?
- How to avoid vendor lock-in?
- Do we need to design new protocols (e.g. RIPP)?

Thanks!

Contact us:

federico.cabiddu@gmail.com

giacomo.vacca@gmail.com