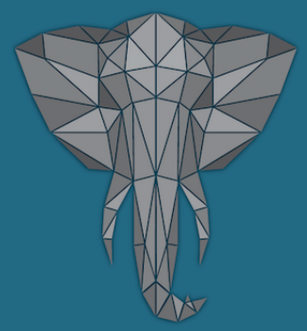
Modern PostgreSQL High Availability



PGConf.Russia 2018

ONGRES

Álvaro Hernandez Tortosa

ALVARO HERNANDEZ

CEO

DBA and Java Software Developer
OnGres and ToroDB Founder
Well-Known member of the PostgreSQL Community
World- Class Database Expert (+30 Talks in last 2 years)

@ahachete

Introduction

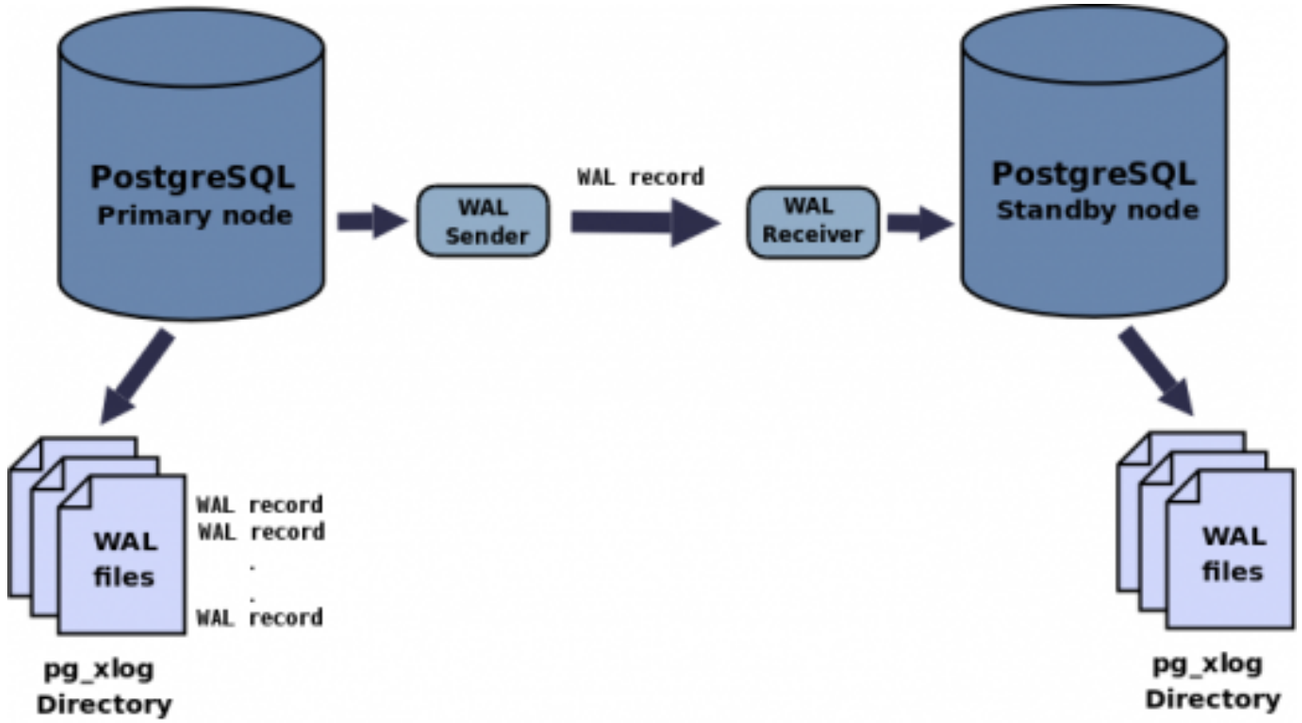
What HA is… NOT

- Easy
- A mechanism to improve the performance
- A risk-free or data-loss free silver-bullet
- Totally Available (in the CAP sense). Either CP or Eventually Consistent with High Availability
- Something integrated into core

What HA is… NOT

SR != HA

(Streaming Replication)



What I don't call "*Modern*" HA

HA built on disk replication or OS virtualization

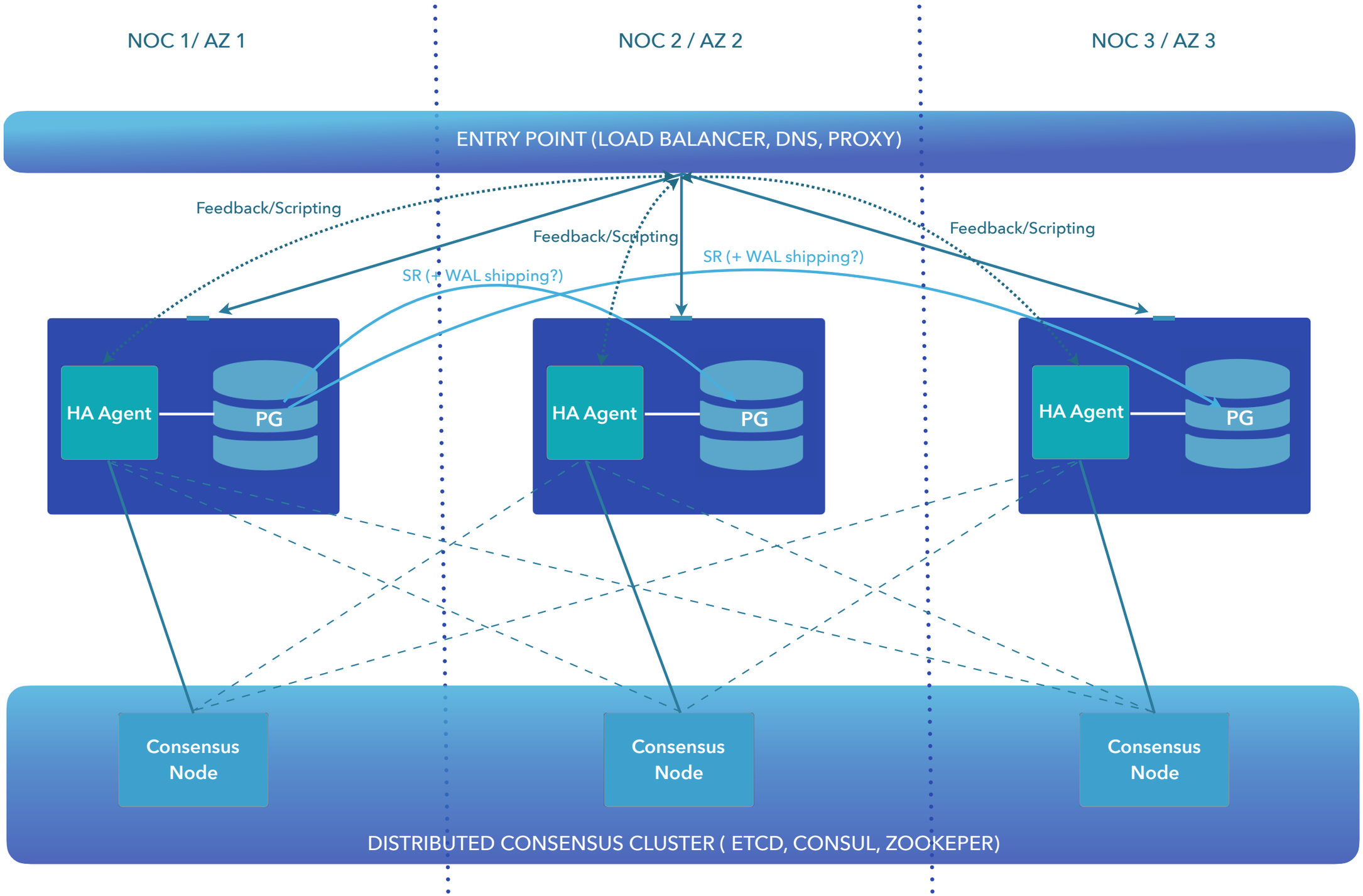
- Aurora is modern, but opaque
- Hardware-assisted, drdb, iSCSI, etc: a lot of limitations
- Typically limited to only 2 nodes
- Hot-standby (waste of resources)

What I don't call "*Modern*" HA

Solutions not based on consensus alg

- HA is a distributed "agreement" problem
- Only well-known consensus algorithms (Paxos, Raft, ZAB) are proven to work
- Solutions based on home-grown algorithms would probably **fail**

What I don't call "*Modern*" HA



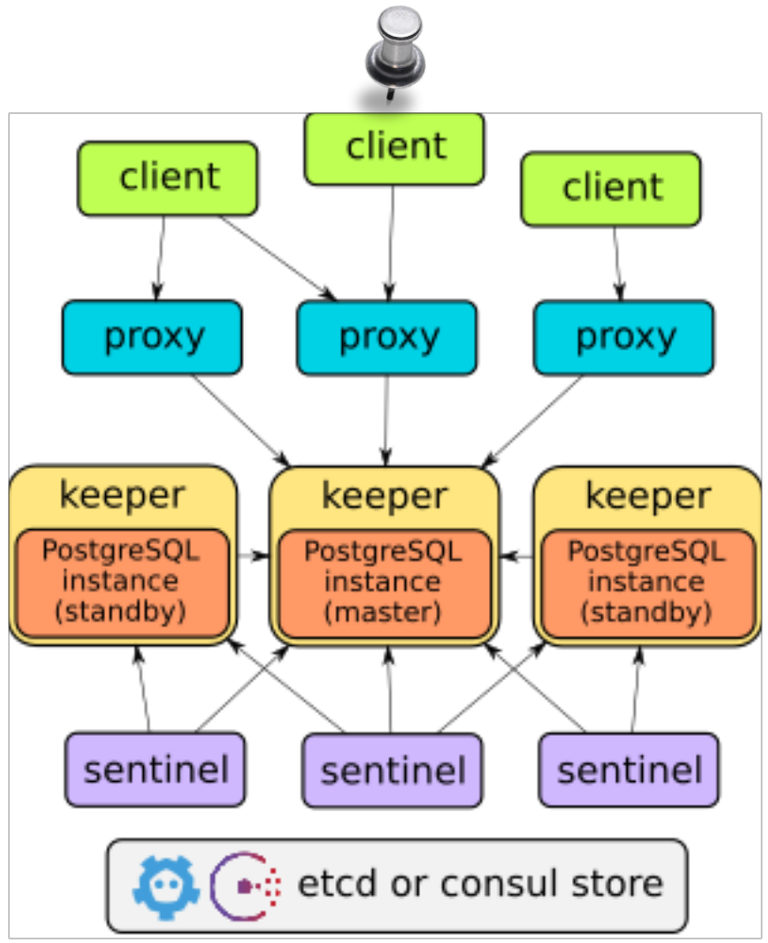
Some open-source "*Modern*" HA solutions

Patroni (forked from Governor)
https://github.com/zalando/patroni
Python

Stolon
https://github.com/sorintlab/stolon
Go

Current limitations / problems

- Quite involved set up
- Too invasive (agent controls PG, even config!)
- No entry-point included (Stolon does; perf?)
- Consensus is external (yet another moving part)
- Hard automatic deployments (complexity)
- How tested are them?



Stolon Architecture

HA caveats / recommendations / best practices

Replication options (I)

- Asynchronous (default):
  - ✓ data-loss risk
  - ✓ may require *pg_rewind*
- Synchronous:
  - ✓ lower performance (pick wisely!)
  - ✓ may lead to cluster unavailability

Replication options (II)

- Synchronous + asynchronous:
  - ✓ Multiple synchronous, PG 9.6:

```
synchronous_standby_names = 'x (node_1,
node_2, …, node_N)’
```

  - ✓ Quorum commit, PG 10:

```
synchronous_standby_names = 'ANY x (node_1,
node_2, …, node_N)'
```


Replication options (III)

- If you include synchronous, you may also want (9.6):

```
synchronous_commit = 'remote_apply'
```

(otherwise, you achieve no data-loss, but only casual reads)

The entry point (I)

How do PostgreSQL clients connect to master?

PostgreSQL **does not** provide support in the protocol for advertising the master and/or propagating cluster topology.

MongoDB does: https://github.com/mongodb/specifications/blob/master/source/server-discovery-and-monitoring/server-discovery-and-

The entry point (II)

- Client libraries in PG 10 add support for multiple `hosts/ports`:

```
host=host1,host2 port=port1,port2
```

postgresql://host1:port2,host2:port2/

- As well as read-write only or any server:

```
postgresql://...?target_session_attrs=read-write|any
```


The entry point (III)

- Similarly, JDBC introduced support for multiple hosts and read-write/read-only host selection (compatible with earlier PG versions too):

```
jdbc:postgresql://node1,node2,node3,...,
nodeN/db?targetServerType=master

jdbc:postgresql://node1,node2,..., nodeN/db?
targetServerType=preferSlave&loadBalanceHosts=t
rue|false
```


The entry point (IV)

- HAProxy, pgbouncer, cloud load balancers, DNS
- Abit DIY: how do entry point knows of changes to master state? Typically scripting required
- Patroni healthcheck URLs return 200|503: GET|OPTIONS http://ip:8008/(master|replica)
- Stolon provides a proxy as entry-point. Performance?

The entry point (V)

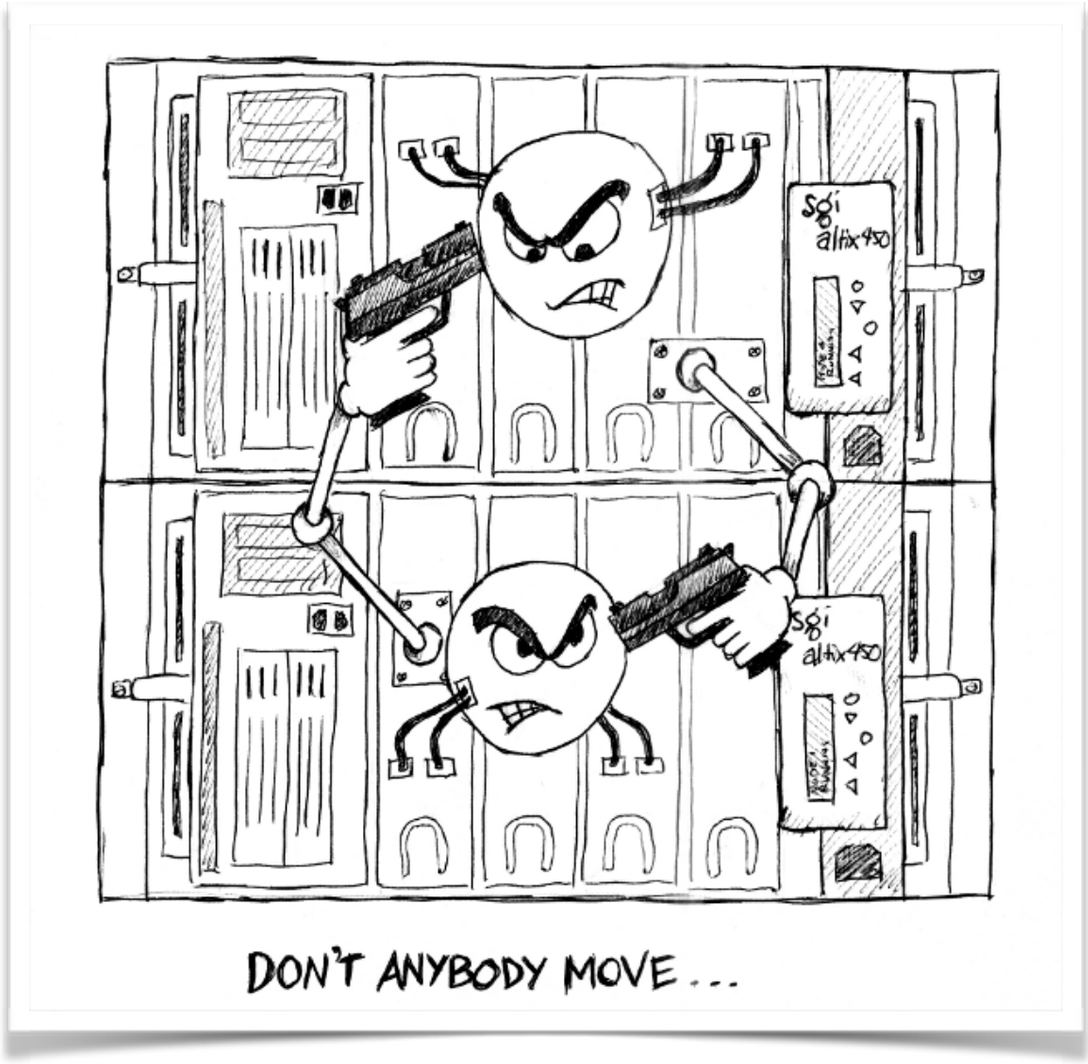
- Consul is a distributed consensus database, that also provides authoritative DNS support
- It could be used to consistently propagate master state changes via DNS
- No changes required on clients
- Consul DNS not used by open source HA solutions

Difficulties in automation / deployment

- You may not be able to use already created automation "roles" for PostgreSQL, as they "take over" it and control even *initdb* process.
- Many moving parts, many automations needed
- Provisioning is done by HA or AutoScaling?
- You may need to rely on fixed private IPs or use internal load balancer

STONITH

- Master gets partitioned away from cluster. But… clients still connected and writing!
- You need to fence/STONITH “old” master.
- Architecture/environment/cloud dependent (scripting again…)
- Possible unreplicated transactions due to asynchronous mode (data loss)
- Best is done through proxy (Stolon)



Application retry logic

- Application needs to be programmed to retry connections, as they may fail and be restarted.
- Transactions may be aborted on connection fails, augment retry logic to also transactions.
- On PG>=10 you may check status of a tx:

```
BEGIN; SELECT txid_current(); …; END;
SELECT txid_status(txid);
```


HA replicas vs scale read-only replicas

- They are not the same!
- Should have the same configuration and same hardware on HA replicas
- Fine-tune, if desired, scale read-only replicas for specific workloads --but never promote them!
- Application should be aware of read-only replicas' stale-reads.

Sneak-peek into the future

HA replicas vs scale read-only replicas

- K8S provides monitoring, master election
- Also shared storage
- And also proxy services

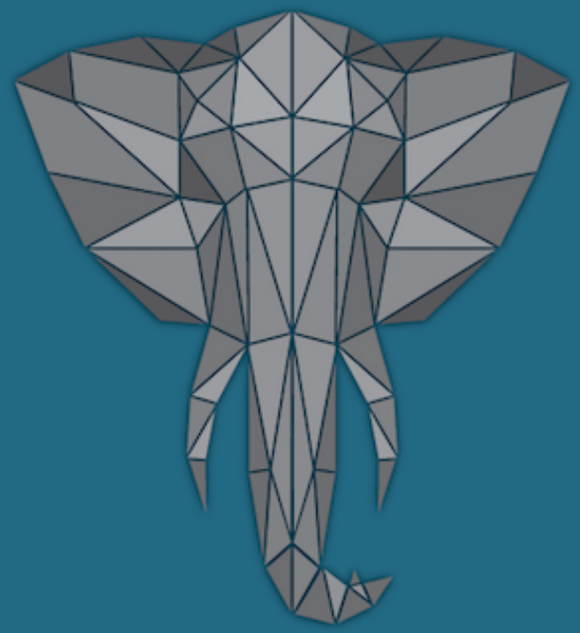
Should be straightforward, right?

- Not sufficiently mature / stable yet
- Generic solution, not PostgreSQL-specific

Multi-datacenter HA: Hydra (OnGres + Groupon)

- Uses Consul KV and DNS Query APIs to provide consistent, multi data center entry points
- Failover in a single data center, followed by instances in other data centers
- Switchover between full data centers
- Automatic election of most recent replica
- Async REST API for management

Questions?



www.ongres.com

1177 Avenue of the Americas. Suite 500
New York, 10036, NY

Phone: +1 (646) 452 7168

Crta. de Fuencarral, 22, Edificio 4B, Local 33
Alcobendas, 28108, MD

Phone: +34 918675554

info@ongres.com