CYBERTEC DATABASE HIGH AVAILABILITY, DISASTER RECOVERY AND CLUSTERING OPTIONS

Document version: 1.1
Last change: 2024-12-02

TABLE OF CONTENTS

PURPOSE OF THIS DOCUMENT

CYBERTEC provides various options to run **high-availability clusters** based on various technologies including but not limited to:

- PostgreSQL / PGEE
- The Patroni toolchain
- Kubernetes / OpenShift / Rancher

The idea of this document is to outline which **options** are **available** to the customer and which technologies can be used to make **PostgreSQL** even more **redundant** and resilient to failure.

In this document we will describe a variety of solutions:

- Single Patroni cluster
- Standby clusters with Patroni
- Manual data center failover
- Automatic data center failover with bias
- Automatic failover with VM host block replication
- Automatic failover with cloud quorum
- Automatic failover with 3 data centers

AVAILABLE CLUSTER ARCHITECTURES

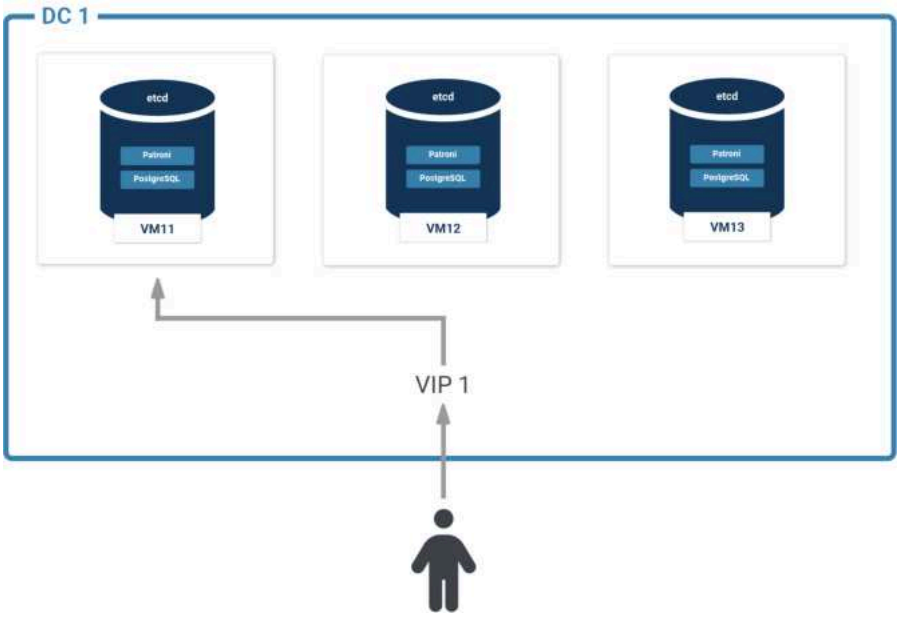
The following section will describe the details of each solution.

OPTION 1: SIMPLE PATRONI CLUSTERS

In the most simplistic scenario a customer wants to run a simple cluster in a single data center. Given a simple setup a Patroni cluster consists of various components:

- Consensus: Through etcd or other RAFT based tools
- Patroni: Automates PostgreSQL handling
- PostgreSQL or PGEE (CYBERTEC PostgreSQL Enterprise Edition)

The architecture is as follows:



Patroni will ensure that there is always **one active node** which can be used to **"read / write"** operations. All other nodes are standby systems and are only available to read data and are therefore suitable for load balancing.

Core requirements:

- An odd number of etcds (3, 5, 7)
- At least 2 database servers
- Bare metal or VMs
- **Optional:**
    - Floating IP connected to the primary (by CYBERTEC vip-manager)

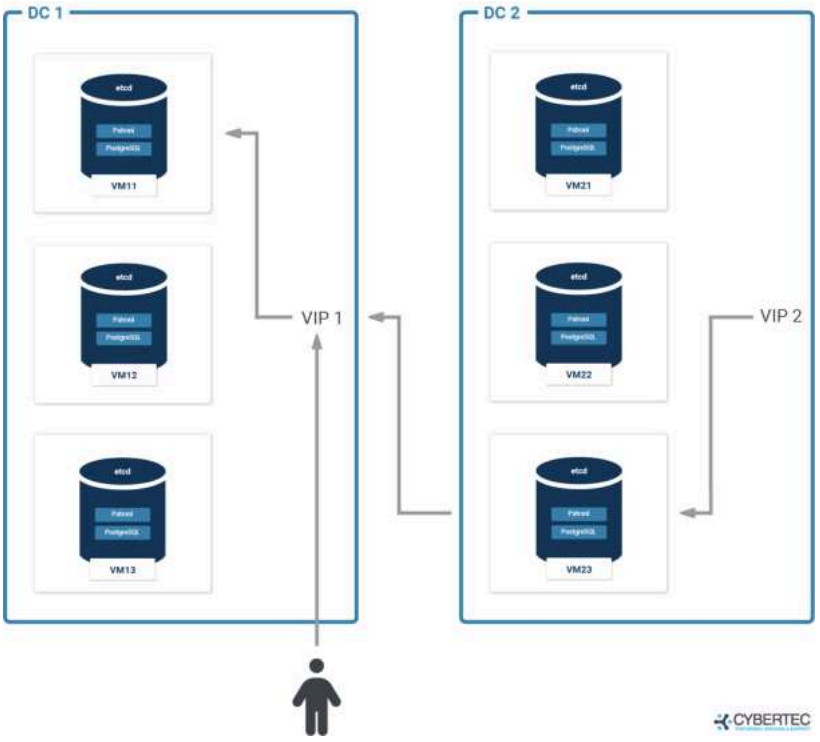
Database nodes and etcds can be in a **single data center** or **spread across multiple sites** (depending on customer requirements and needs). We recommend to keep all systems close to each other

To provide “read / write” access to the database a **majority** of etcds has to be available. Otherwise all services will be degraded to “read only”.

OPTION 2: PATRONI "STANDBY CLUSTERS" FOR SINGLE CLUSTER FAILOVERS

While in a single Patroni cluster all database nodes are created equal this is not always desirable. Often an entire cluster has to follow an entire cluster.

This is especially useful in the following scenario:



In the case of 2 data centers we often want the entire cluster in data center 2 to follow the entire cluster in data center 1. The advantage is that only nodes in data center 1 are valid failover. Why is that desirable:

- We want to keep the application close to the database
- Satisfy a strong preference for one data center
- Have a second cluster for redundancy

Standby clusters (= "a cluster following a cluster") are the way to achieve this goal.

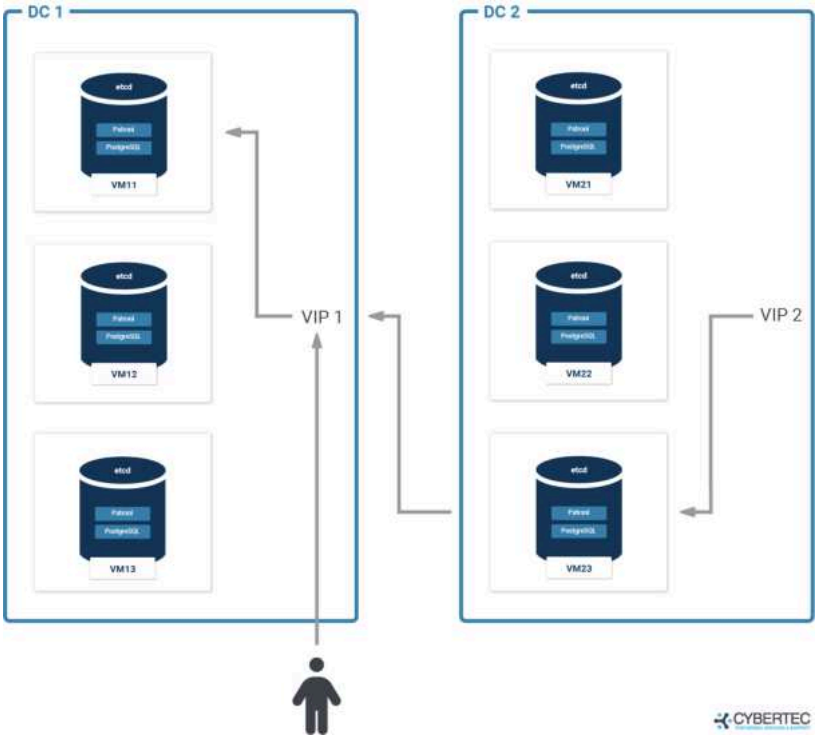
OPTION 3: MANUAL DATA CENTER FAILOVER

Often we want to fail over entire data centers. In the case of two data centers this process is often manual as we cannot ensure “quorum” (= majority vote).

There are two independent **etcd and Patroni clusters** in both data centers, and there is one **virtual IP address (VIP)** in each data center.

The secondary data center is configured as a **Patroni “standby cluster”** and replicates from the Patroni cluster in the primary data center via the VIP.

The user/application always connects to the VIP of the primary data center.
In case of any issues, the Patroni “standby cluster” is promoted to act as a regular cluster (single config change) and the application is configured to connect to the VIP of the secondary data center.



Advantages:

- Very robust
- As many databases as desired
- Easy to extend to more complex setups

Disadvantages:

- Data Center failover not automatic
- Applications have to be moved when there is a DC failover (which has to happen in any architecture)

Requirements:

- Two data centers
- Spare IP addresses
- Sufficient virtual machines

Limitations:

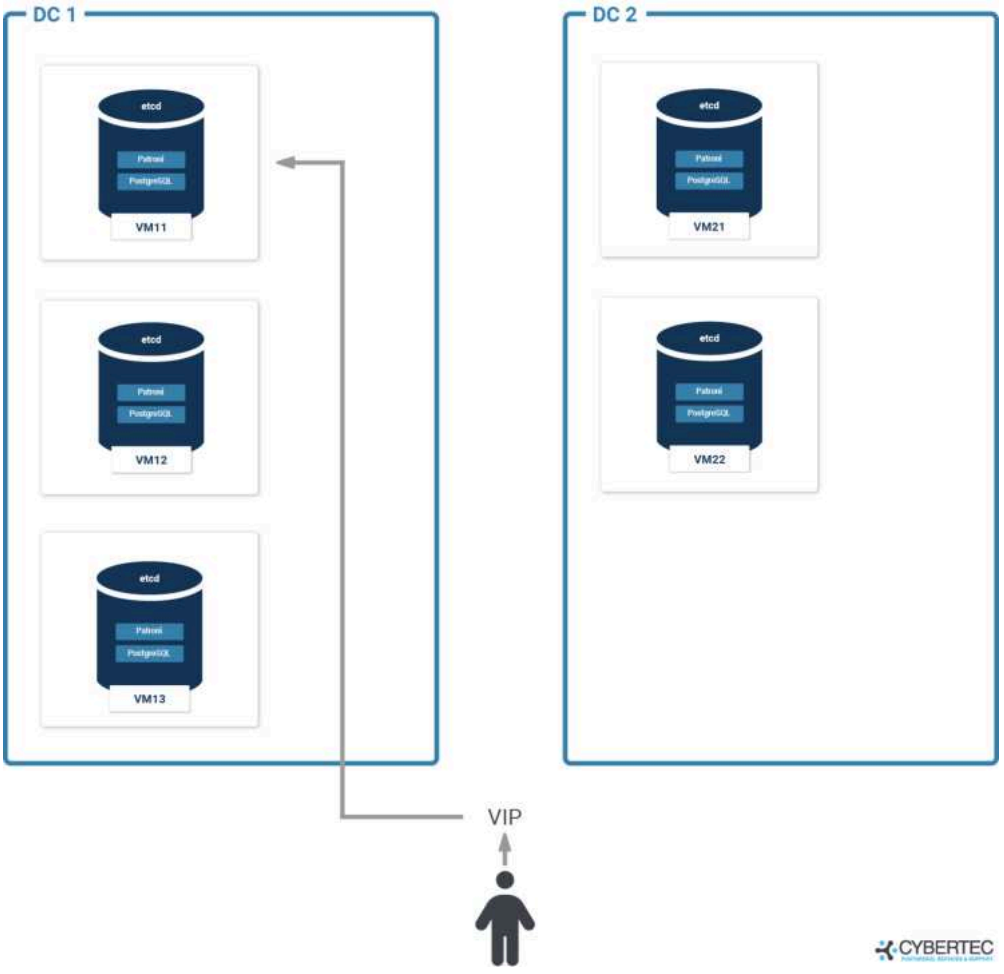
- Identical major versions of PostgreSQL
- Same CPU architecture on all servers (all Intel / AMD, all ARM, etc.)

OPTION 3: AUTOMATIC DATA CENTER FAILOVER WITH BIAS

In this configuration, the resources are more interconnected.

The etcd cluster and Patroni cluster consist of five nodes each, distributed across both data centers, with **a bias on the primary datacenter**.

This means that the primary data center can reach quorum and operate a writable database all on its own, even when the secondary data center might not be reachable.



The secondary data center could be modified to run on its own if a new etcd cluster is created in the secondary data center. This could serve as an emergency fallback but requires a bit of time to configure in case of such an emergency.

Advantages:

- Very robust
- As many databases as desired (odd numbers)
- Easy to extend to more complex setups
- In real world **most customers prefer a primary data center**

Disadvantages:

- Data Center failover not automatic - moving to the secondary needs manual intervention BUT this is needed for the app anyway.
- Applications have to be moved when there is a DC failover (which has to happen in any architecture)

Requirements:

- Two data centers
- Spare IP addresses
- Sufficient virtual machines

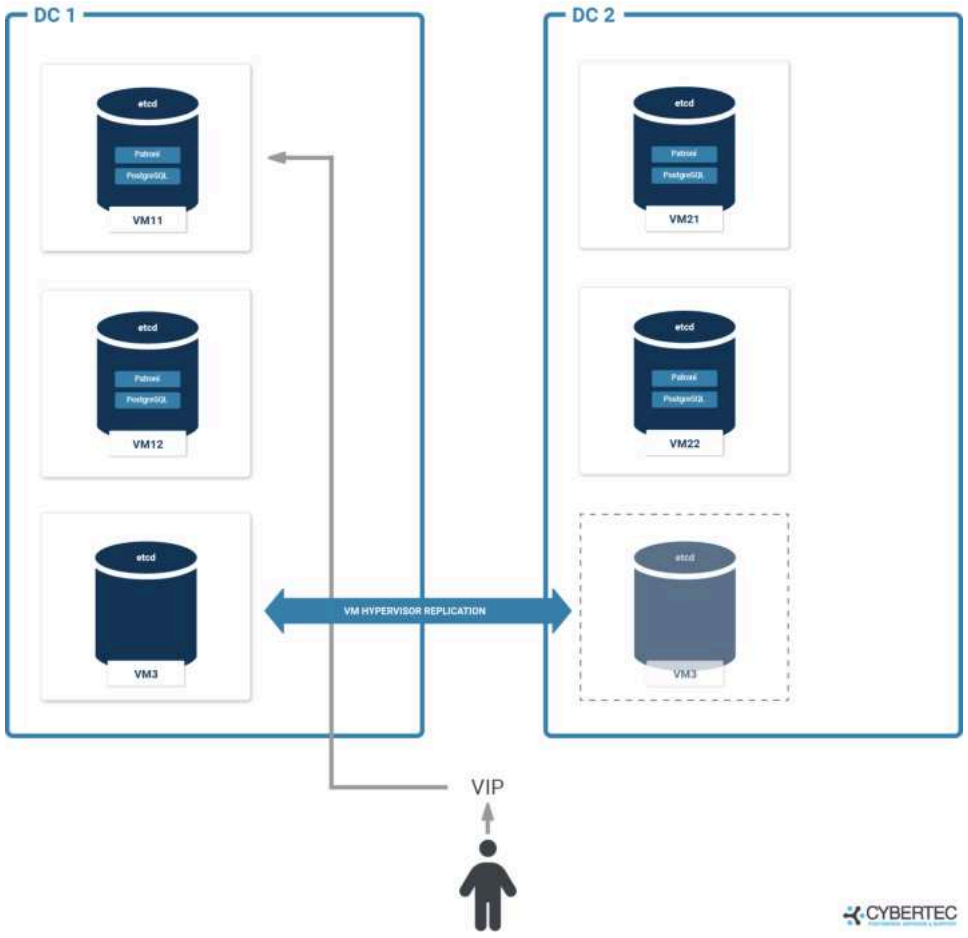
Limitations:

- Identical major versions of PostgreSQL
- Same CPU architecture on all servers (all Intel / AMD, all ARM, etc.)

OPTION 4: AUTOMATIC FAILOVER WITH VM HOST BLOCK REPLICATION

In this configuration, the third machine of the primary cluster is transparently replicated using block based replication from the **VM hypervisor**.

The hypervisor has the ability to move the third machine from one datacenter to another datacenter, moving the “bias” with it.



Subsequently, when the primary datacenter is unreachable, the hypervisor could decide to run the **replicated VM in the secondary datacenter**, thus enabling that secondary datacenter to reach quorum.

Advantages:

- Uses advantages of existing VMware infrastructure (quorum on redundant cables, etc.)
- Easy to extend to more complex setups
- Allows **for full data center failover IF (and only if) data center failover is supported on the VMware level**
- Applications most likely moved by VMware
- HyperVisor very lightweight
- Resilience can be added easily

Disadvantages:

- Requires VMware infrastructure
- Relies on VMware alone

Requirements:

- Two data centers
- **Redundant connections between data centers** (otherwise automatic DC failover is mathematically impossible!)
- Spare IP addresses
- Sufficient virtual machines

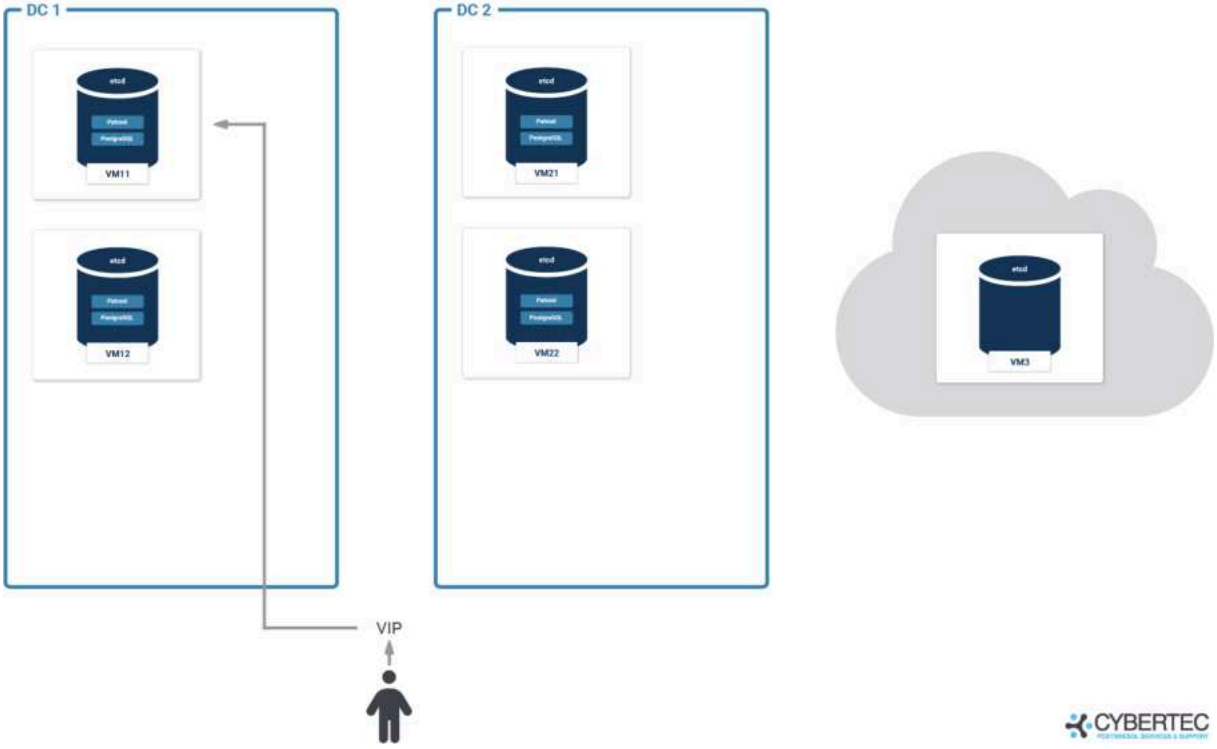
Limitations:

- Identical major versions of PostgreSQL
- Same CPU architecture on all servers (all Intel / AMD, all ARM, etc.)

OPTION 5: AUTOMATIC FAILOVER WITH CLOUD QUORUM

In this configuration, one etcd cluster member is placed in the cloud.

This makes sure that quorum can always be reached, as long as communication between both datacenters is possible.



If communication between both datacenters breaks down, but one of the datacenters is still able to reach the cloud, this cluster member serves as the “tie breaker” and enables quorum, thus ensuring write-ability of the PostgreSQL cluster.

Advantages:

- Most reliable
- Well known setup
- Allows **for fully automatic data center failover**

Disadvantages:

- Requires external infrastructure
- Usually non-existing infrastructure
- One has to discuss app failover

Requirements:

- Two data centers
- Cloud infrastructure
- Spare IP addresses
- Sufficient virtual machines

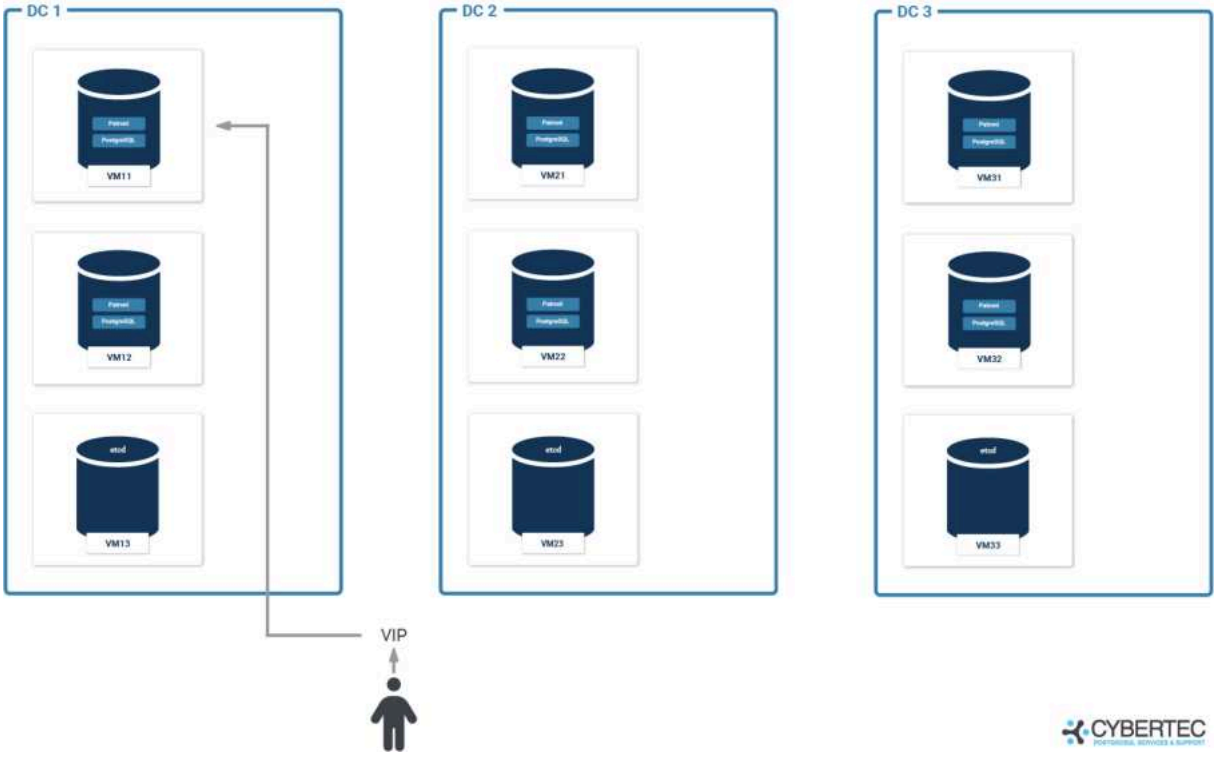
Limitations:

- Identical major versions of PostgreSQL
- Same CPU architecture on all servers (all Intel / AMD, all ARM, etc.)

OPTION 6: AUTOMATIC FAILOVER WITH 3 DATA CENTERS

In this configuration, there is a third data center.

The configuration for all data centers is the same, keeping **complexity low** and making all data centers equally important.



Any data center (or any link between the data centers) can fail; As long as two data centers can still communicate with each other, quorum can still be reached and the PostgreSQL service is still writeable.

Advantages:

- Very reliable
- Well known setup
- Allows **for full data center failover**

Disadvantages:

- Requires 3 data centers
- Usually non-existing infrastructure
- One has to discuss app failover

Requirements:

- Three data centers
- Spare IP addresses
- Sufficient virtual machines

Limitations:

- Identical major versions of PostgreSQL
- Same CPU architecture on all servers (all Intel / AMD, all ARM, etc.)

OPTION 7: DATA CENTER QUORUM AND DISTRIBUTION

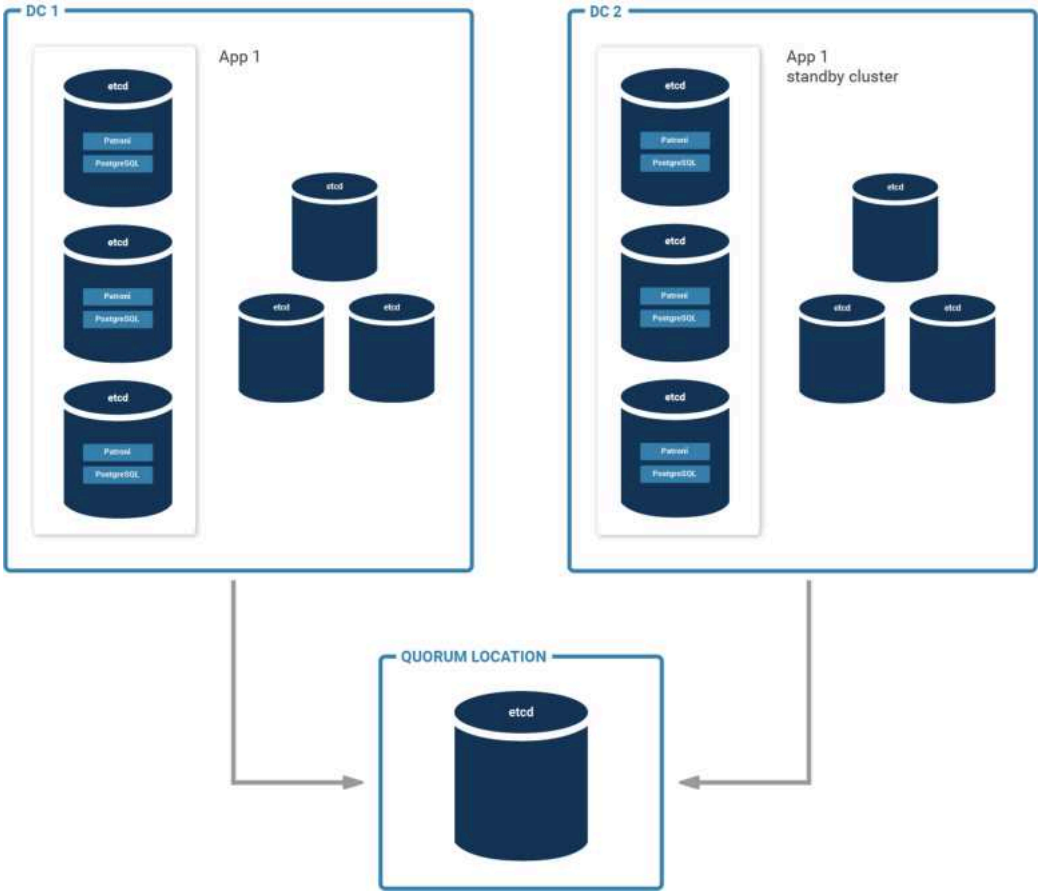
The final scenario is related to how quorum works in a distributed system. In this case we have 3 etcds in each data center forming ONE cluster per data center.

What is important here: If we have 3 x 2 etcds it technically means that we have "2 etcd cluster" which is of course an even number. This leads to a split brain situation in case a data center is lost ("no majority because it 1 : 1 vs 2 : 0).

Therefore we need a 2-layer etcds. There will be 3 etcds per data center plus an additional etcd per data center as well as a space etcd outside.

It would mean:

DC1: 1 cluster (3 etcds) + 1 for etcd quorum
DC2: 1 cluster (3 etcds) + 1 for etcd quorum
Ext: 1 for etcd quorum



Advantages:

- No split brain possible
- Automatic data center failover
- Independent etcd clusters in each DC

Disadvantages:

- Requires 3 locations
- Slightly more complicated setup

Requirements:

- Three locations
- Spare IP addresses
- Sufficient virtual machines

Limitations:

- Identical major versions of PostgreSQL
- Same CPU architecture on all servers (all Intel / AMD, all ARM, etc.)

INTEGRATING WITH KUBERNETES/OPENSHIFT

All architectures are also available on Kubernetes using the following technology:

- Zalando operator + CYBERTEC extensions
  - CYBERTEC multi-data center support
  - CYBERTEC 2-layer RAFT consensus
  - CYBERTEC backup extensions

Our contributions will be released as Open Source to achieve maximum …

> “Digital resilience and independence”

Container orchestration is important. We therefore provide:

- Full automation and high efficiency
- 24x7 Enterprise support for PostgreSQL / PGEE
- High-availability by default
- Enterprise grade audit
- Full customization (if needed)