Architectures for PostgreSQL High Availability and Disaster Recovery (HA/DR)

postgrespro.com

Alexey Shishkin
a.shishkin@postgrespro.com

Postgres Professional

- Postgres Professional, established in 2015, is a key contributor to PostgreSQL community
- At Postgres Professional we develop Postgres Pro database, a private PostgreSQL fork
- Postgres Professional also specializes in 24x7 technical support and other professional services (database migration, audit and performance tuning) for PostgreSQL

HA/DR for PostgreSQL databases

- When do you need to consider an HA/DR architecture ?
- HA/DR for Postgres in a nutshell (how to make it work)
- Introduction to commonly used HA/DR architectures
- Pros and cons of various HA/DR architectures for PostgreSQL
- HA/DR field experience

SLA: RTO and RPO

- RTO (Recovery Time Objective) - how long an application can be unavailable for business users

- 99,99% - 52,56 minutes of downtime per year (~0,9 hrs.)
- 99,9% - 525,6 minutes of downtime per year (~9 hrs.)
- 99% - 5256 minutes of downtime per year (~90 hrs.)

- RPO (Recovery Point Objective) - how much business data can be lost

A sample customer’s SLA

- Mission critical application - RTO - 99,99%, RPO - 0
- Business critical application - RTO - 99,9%, RPO - 0
- Business operational application - RTO - 98%, RPO - 1 hr.
- Office operational application - RTO - 90%, RPO - 12 hrs.

HA/DR technologies (1/4)

- Manual switchover/failover

- the fewer moving parts in the technology stack the better
- monitoring and alerting systems work fine to inform the Operations team about the database issues
- switchover/failover scenarios are well documented and both day and night shifts of the Operations team have adequate expertise to cope with database availability issues
- RTO is not very strict (up to 5 minutes for switchover/failover tasks)
- the number of databases is relatively small (up to 50)

HA/DR technologies (2/4)

- HA-cluster (don’t confuse this with Postgres database cluster, which is a collection of databases that is managed by a single instance of a running database server)

- RTO is strict (within a minute for switchover/failover tasks)
- the number of databases is big (up to 100+)

HA/DR technologies (3/4)

- Replication
- logical (database transaction)
- streaming (database block)
- file system (block device)
- disk/LUN (raw device)
- Each kind of replication can be synchronous or asynchronous

HA/DR technologies (4/4)

- Backup
  - online or offline
  - full or incremental
  - physical or logical
  - data files or WAL files
  - via database tools or via disk-array snapshots

HA-cluster

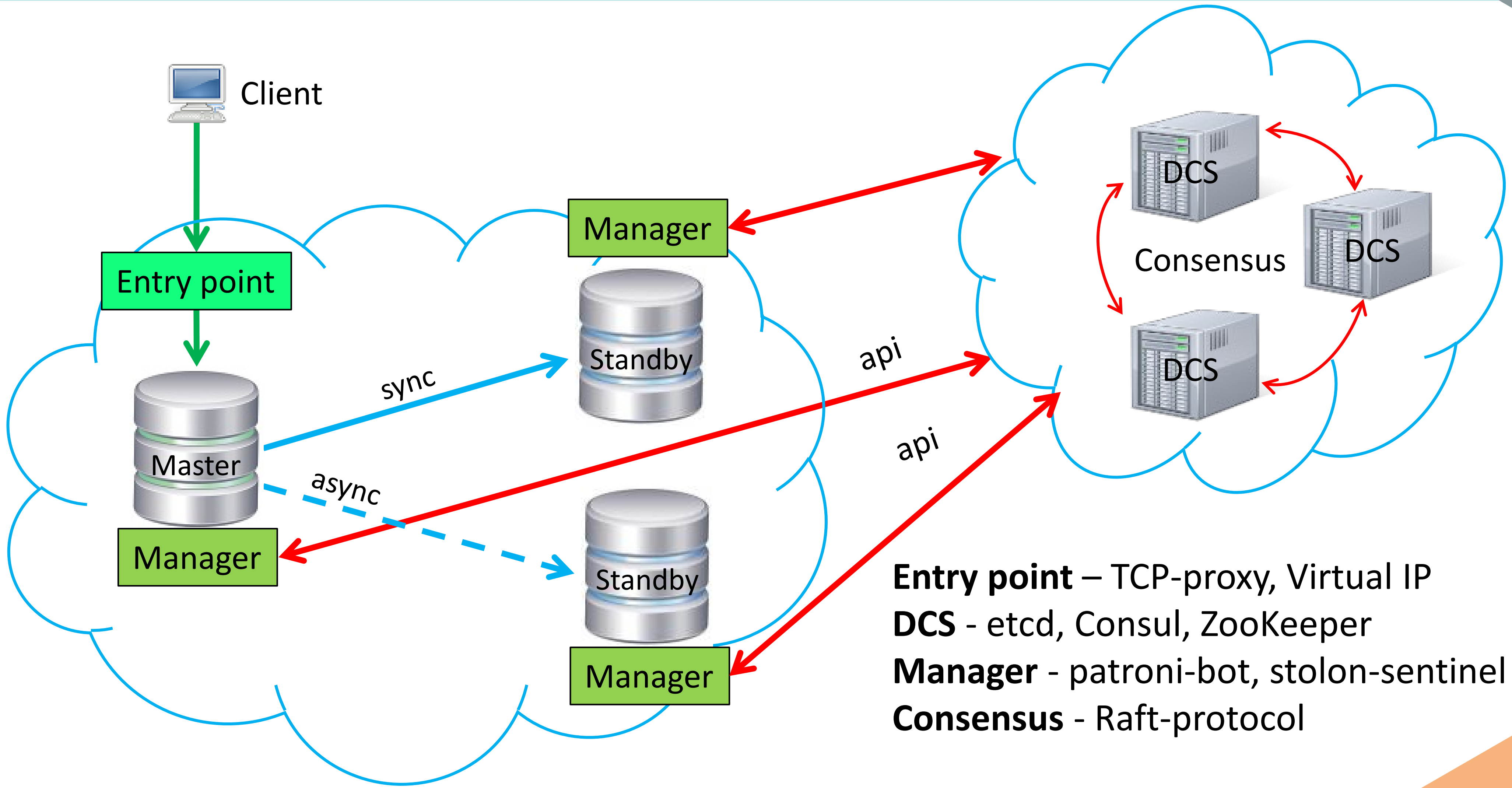
- The list of HA-clusters (sorted by popularity among our customers)

- Patroni - https://github.com/zalando/patroni
- Corosync/Pacemaker - https://github.com/ClusterLabs
- Stolon - https://github.com/sorintlab/stolon
- Postgres Pro Multimaster - https://github.com/postgrespro/mmts
- Veritas - https://www.veritas.com/availability/infoscale

Patroni and Stolon (1/2)

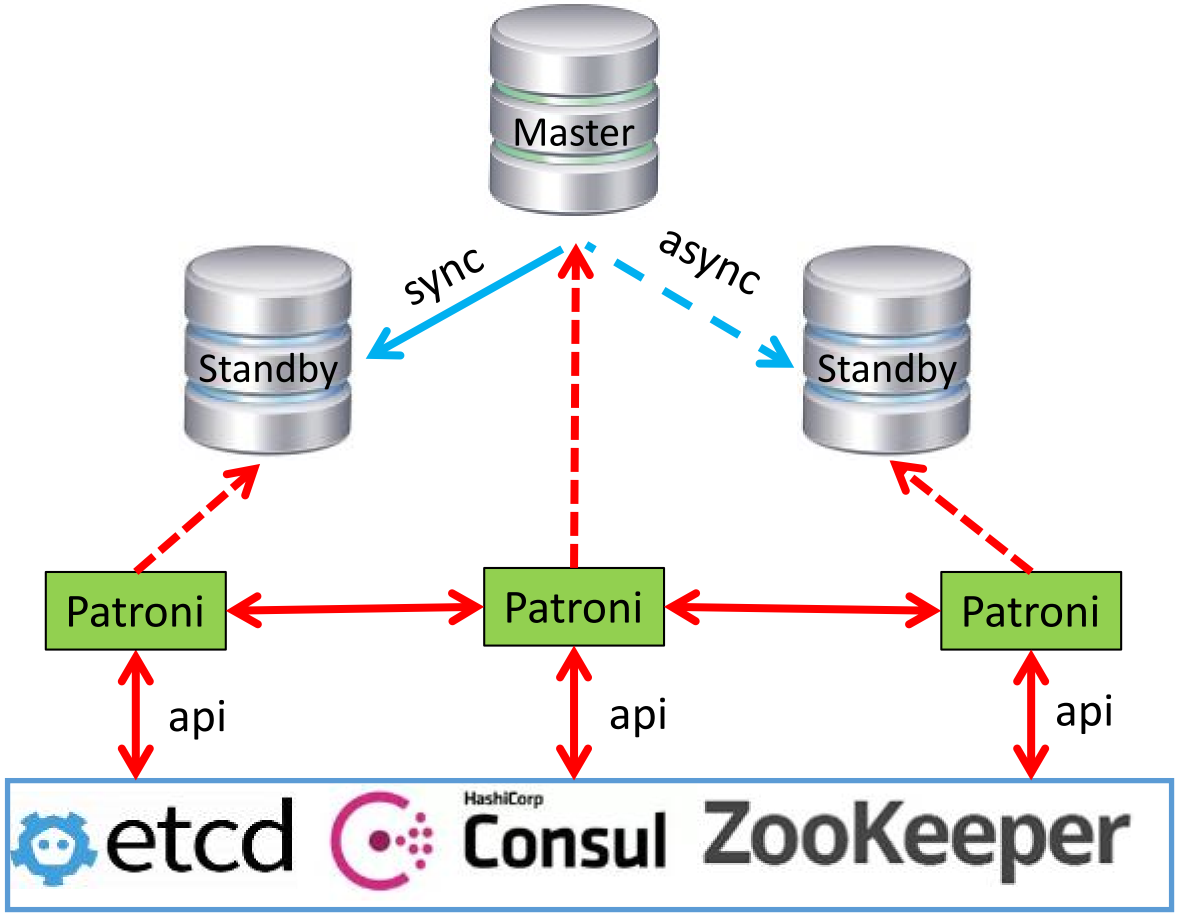
- Patroni and Stolon are similar in functionality and architecture
- depend on DCS (Distributed Configuration Store)
- require Postgres streaming replication
- suitable for physical servers and virtual machines (VMs)
- open-source and free of charge
- Patroni uses external TCP-proxy to connect to master or standby(s)
- Stolon has built-in TCP-proxy to connect to master or standby(s)

Patroni and Stolon (2/2)

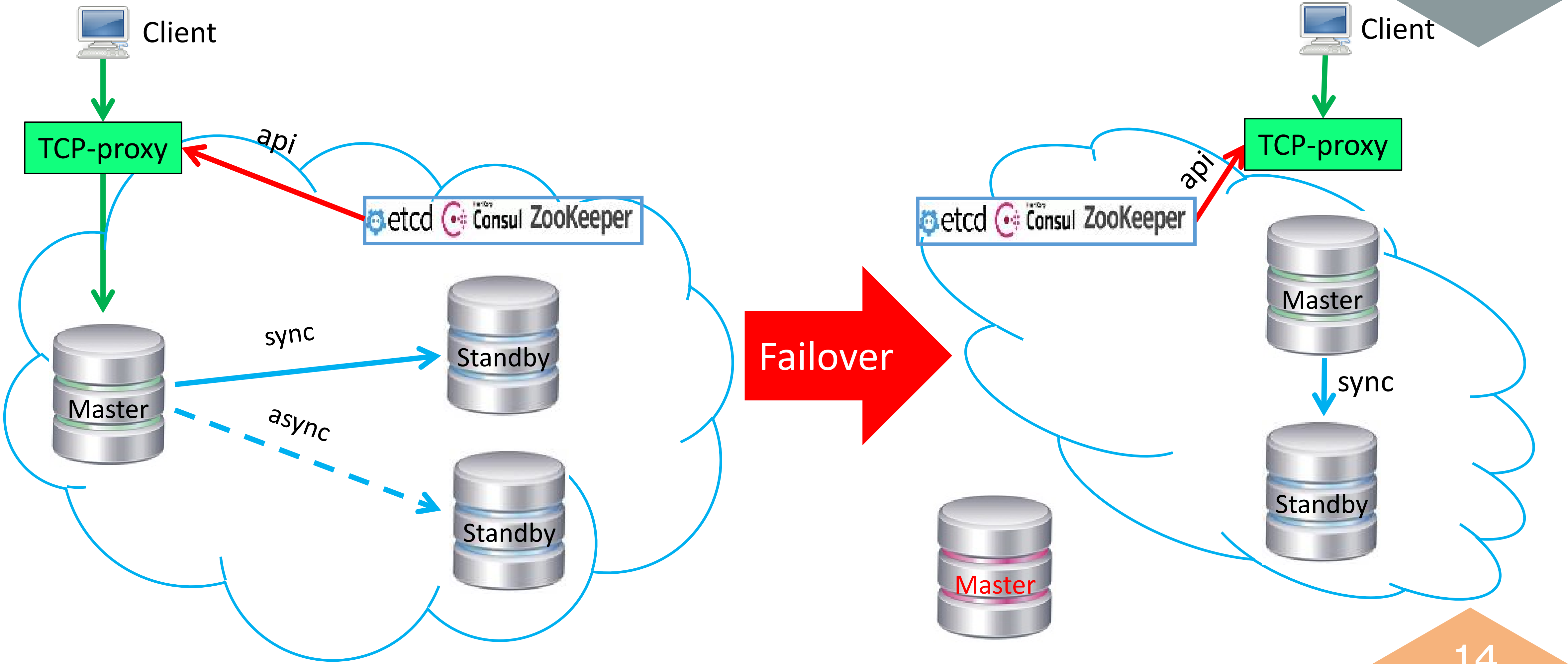


Entry point – TCP-proxy, Virtual IP
DCS - etcd, Consul, ZooKeeper
Manager - patroni-bot, stolon-sentinel
Consensus - Raft-protocol

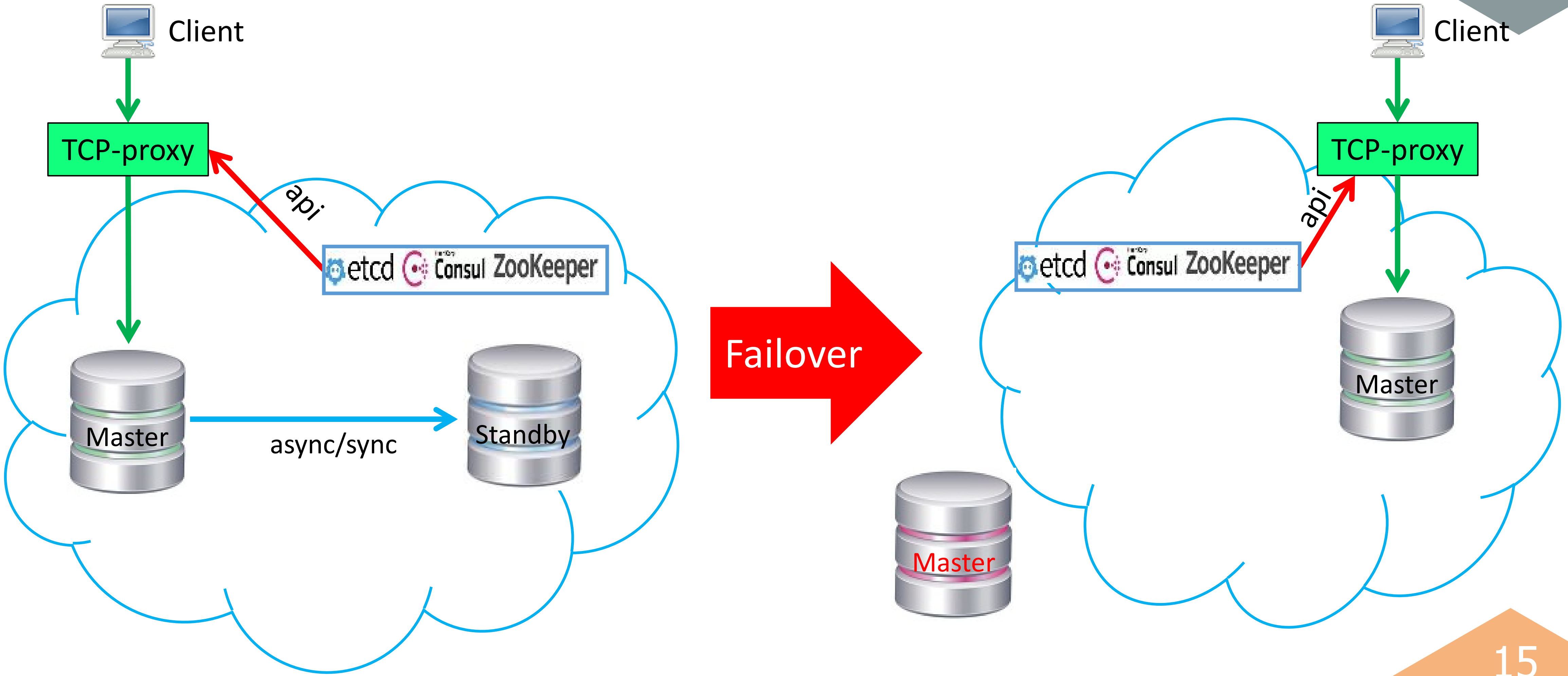
Patroni architecture



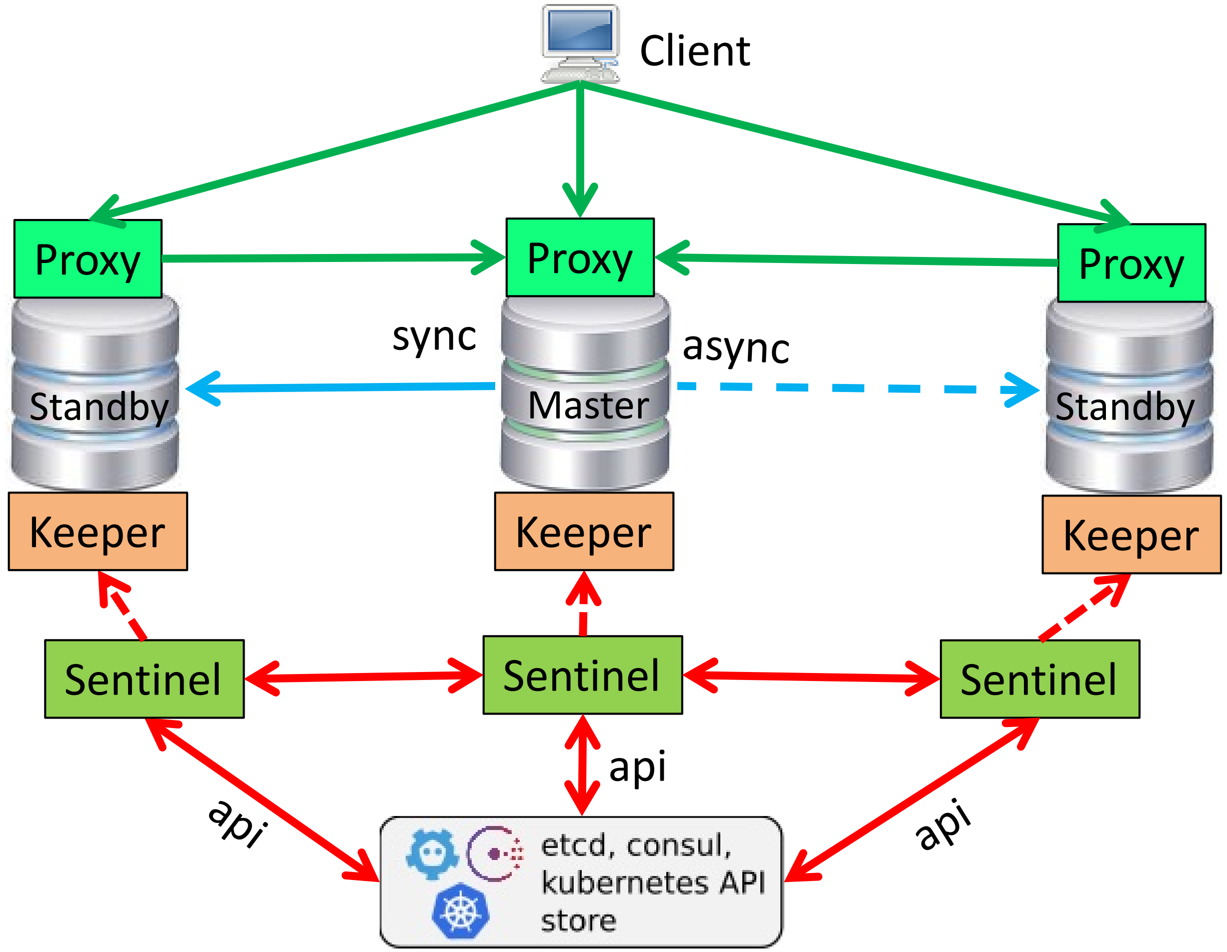
Patroni (3-node HA-cluster)



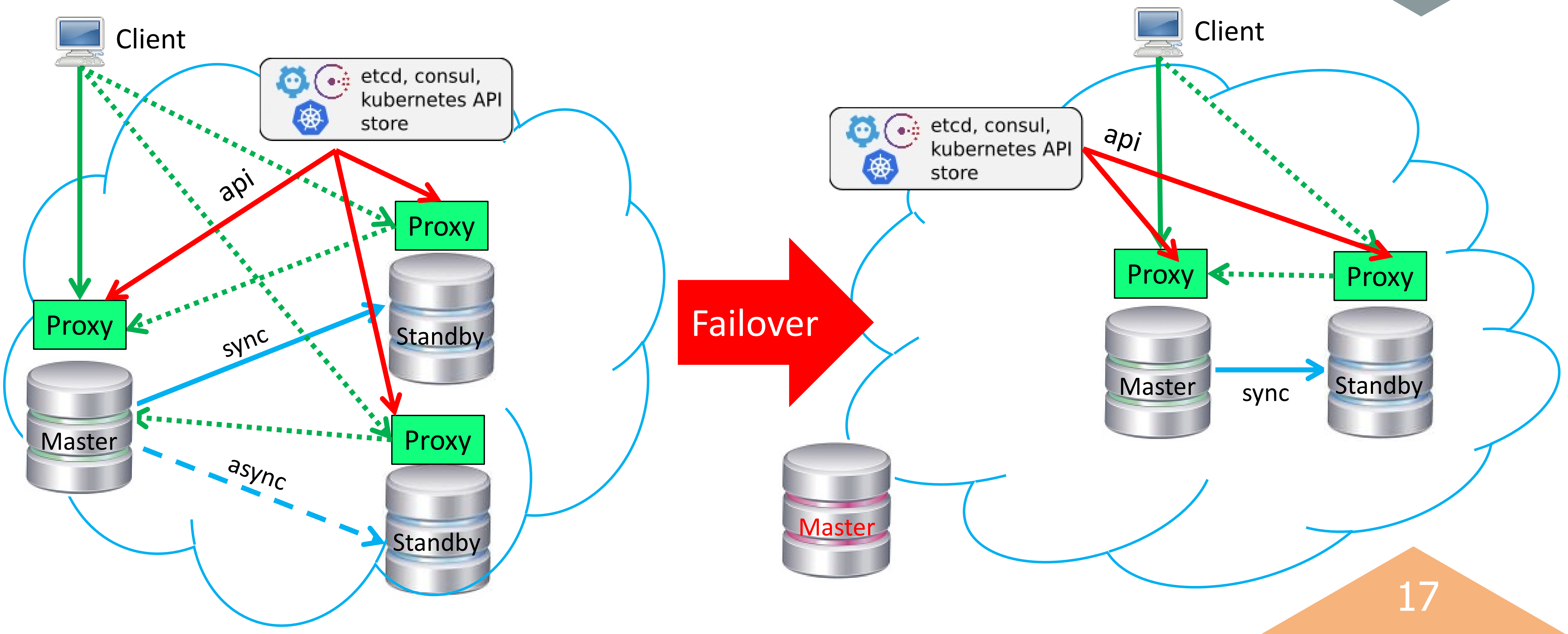
Patroni (2+1 HA-cluster)



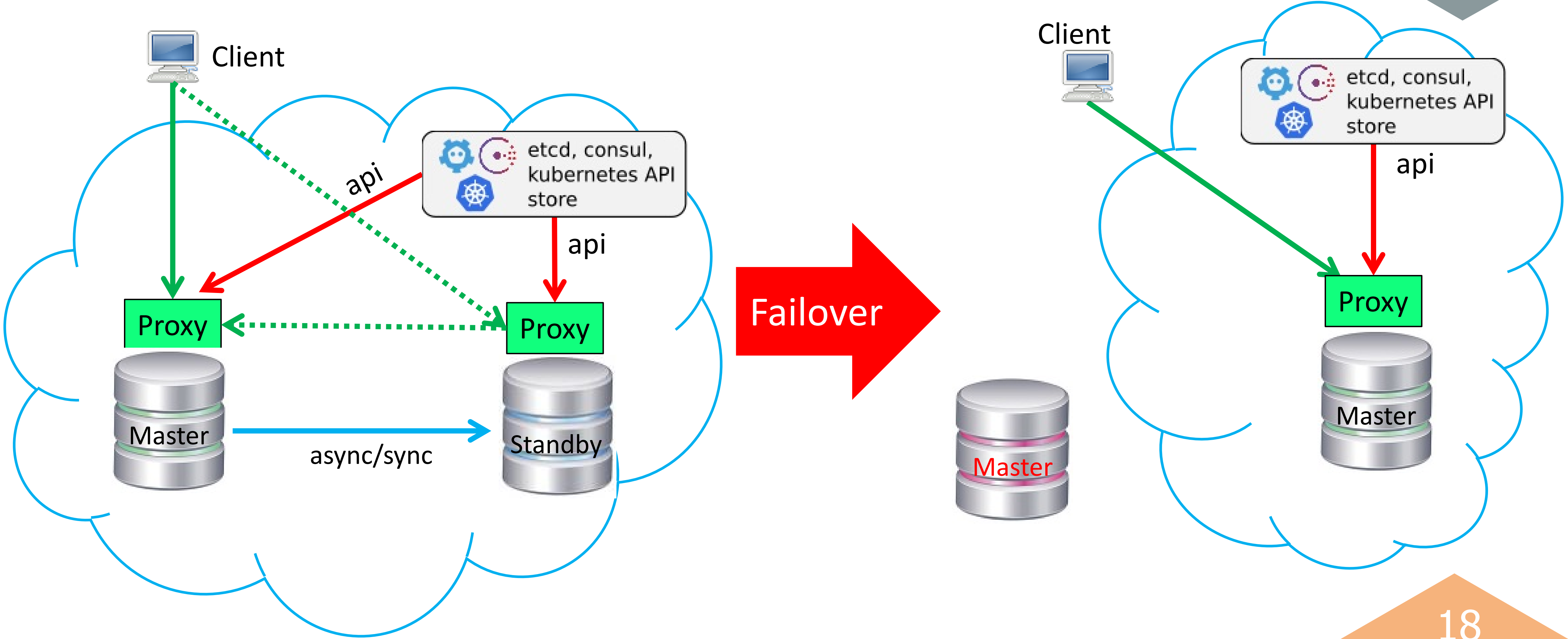
Stolon architecture



Stolon (3-node HA-cluster)



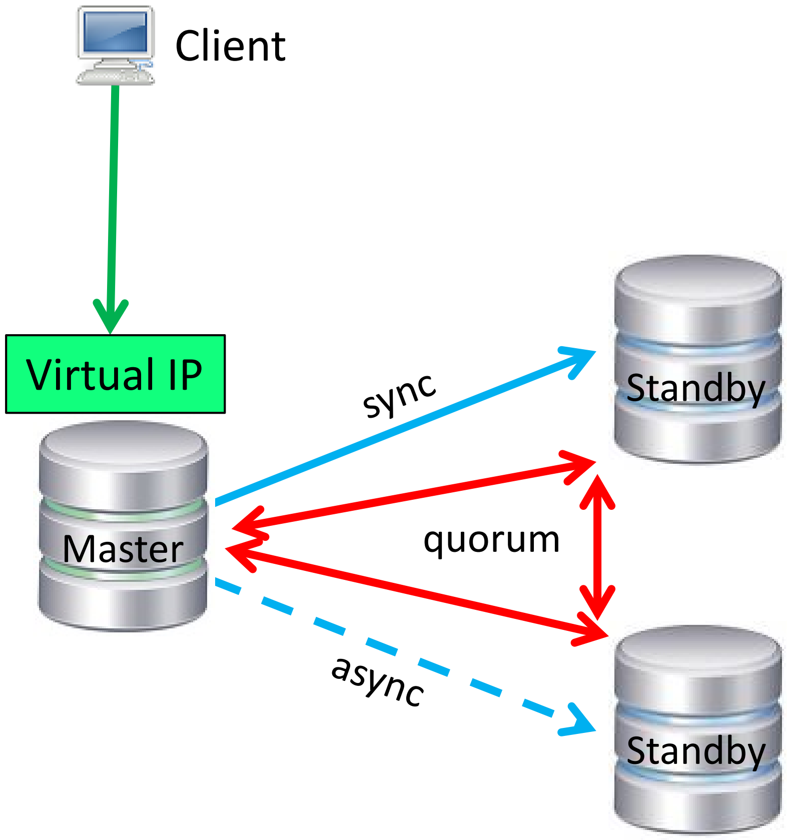
Stolon (2+1 HA-cluster)

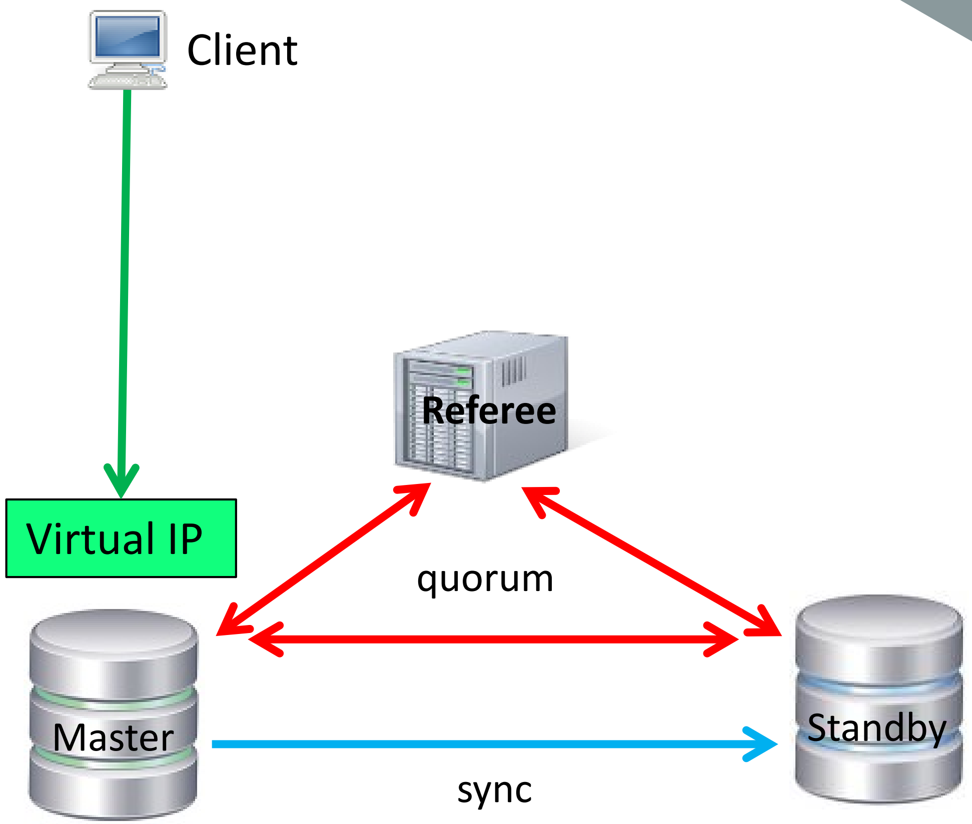


Corosync/Pacemaker and Veritas

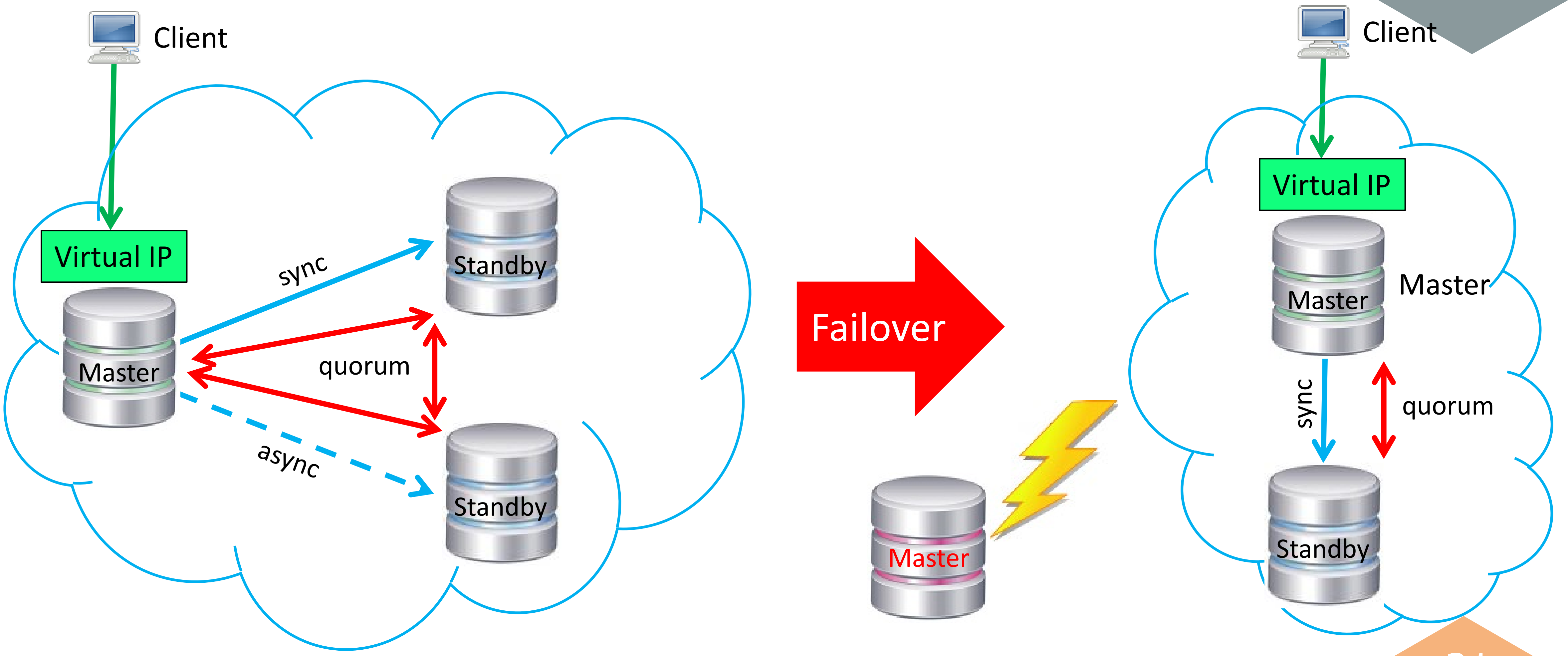
- Corosync/Pacemaker and Veritas are similar in functionality and architecture
  - use resource agents (disk volume, file system, IP-address, Postgres)
  - use Virtual IP-address (VIP) to connect to master or standby(s)
  - can be used with streaming replication and shared disk configuration
  - mostly used with physical servers
  - can be applied to build geo-clusters
- Corosync/Pacemaker is open-source and free of charge
- Veritas is proprietary and requires license (the only HA-cluster which integrates with disk replication)

Corosync/Pacemaker and Veritas architecture

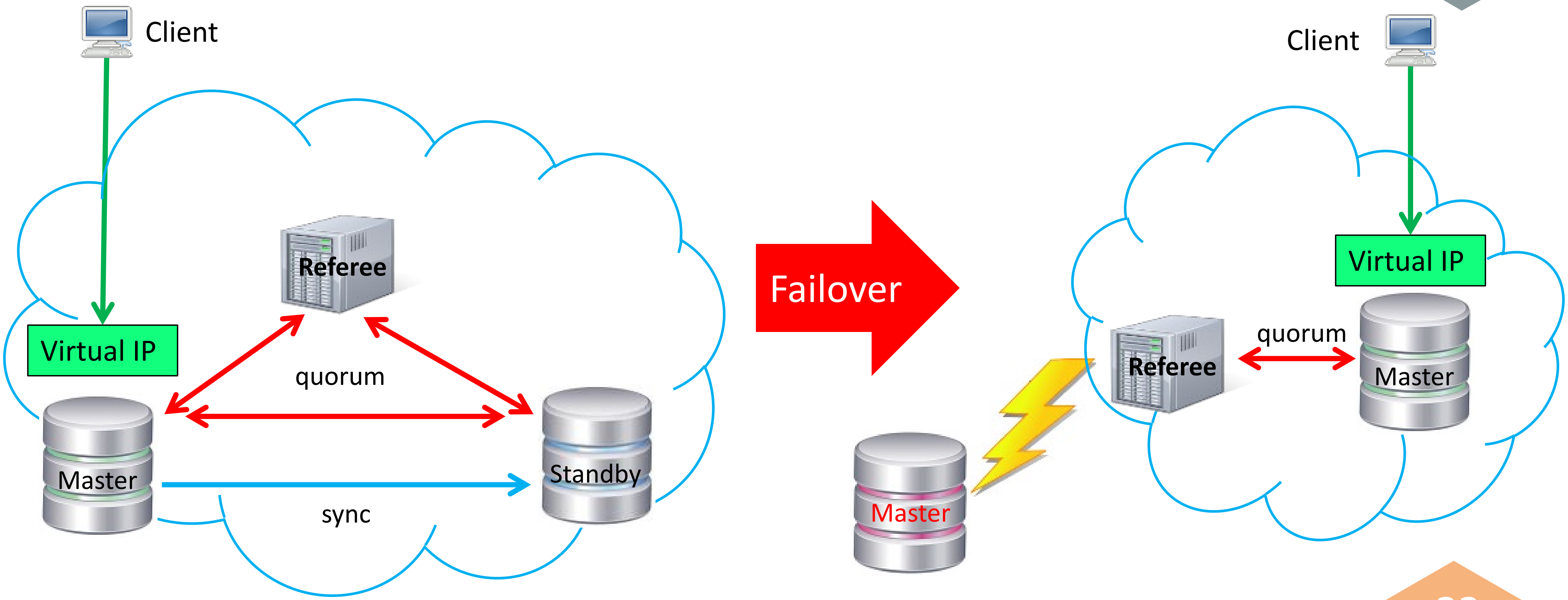




Corosync/Pacemaker (3-node HA-cluster)



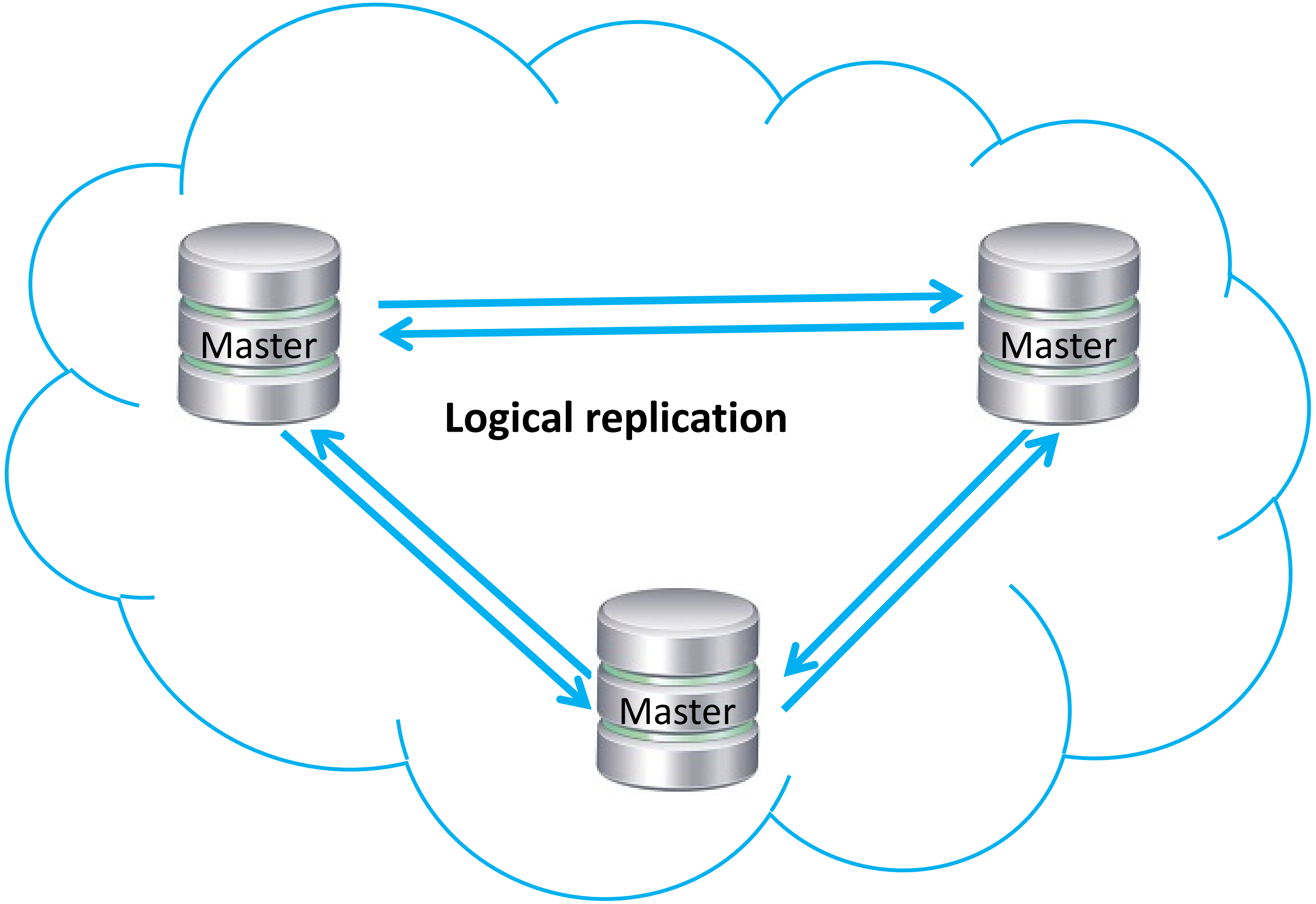
Corosync/Pacemaker (2+1 HA-cluster)



Postgres Pro Multimaster

- Postgres Pro Multimaster differs from other HA-clusters
- uses logical replication
- all nodes can process read-write requests (all nodes are masters)
- delivers minimal possible switchover/failover time (single digit seconds)
- open-source, but requires license to run in a production environment

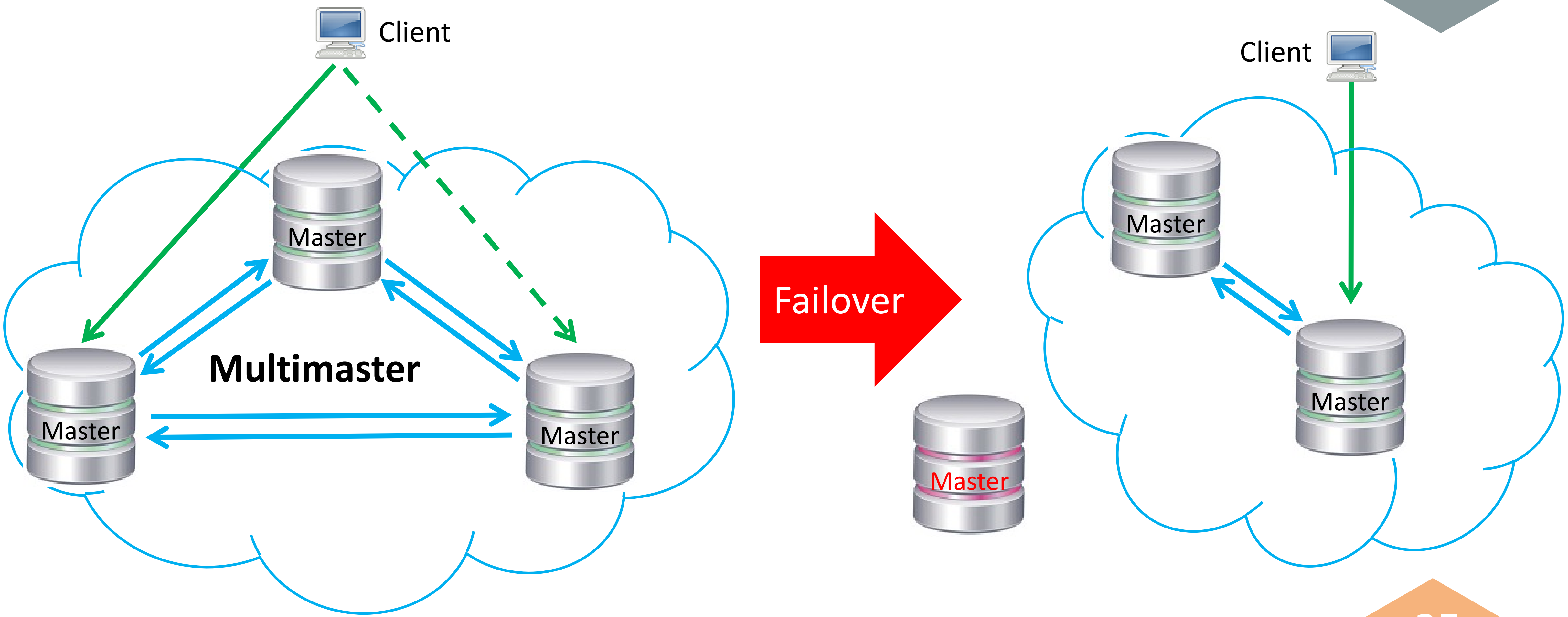
Postgres Pro Multimaster architecture



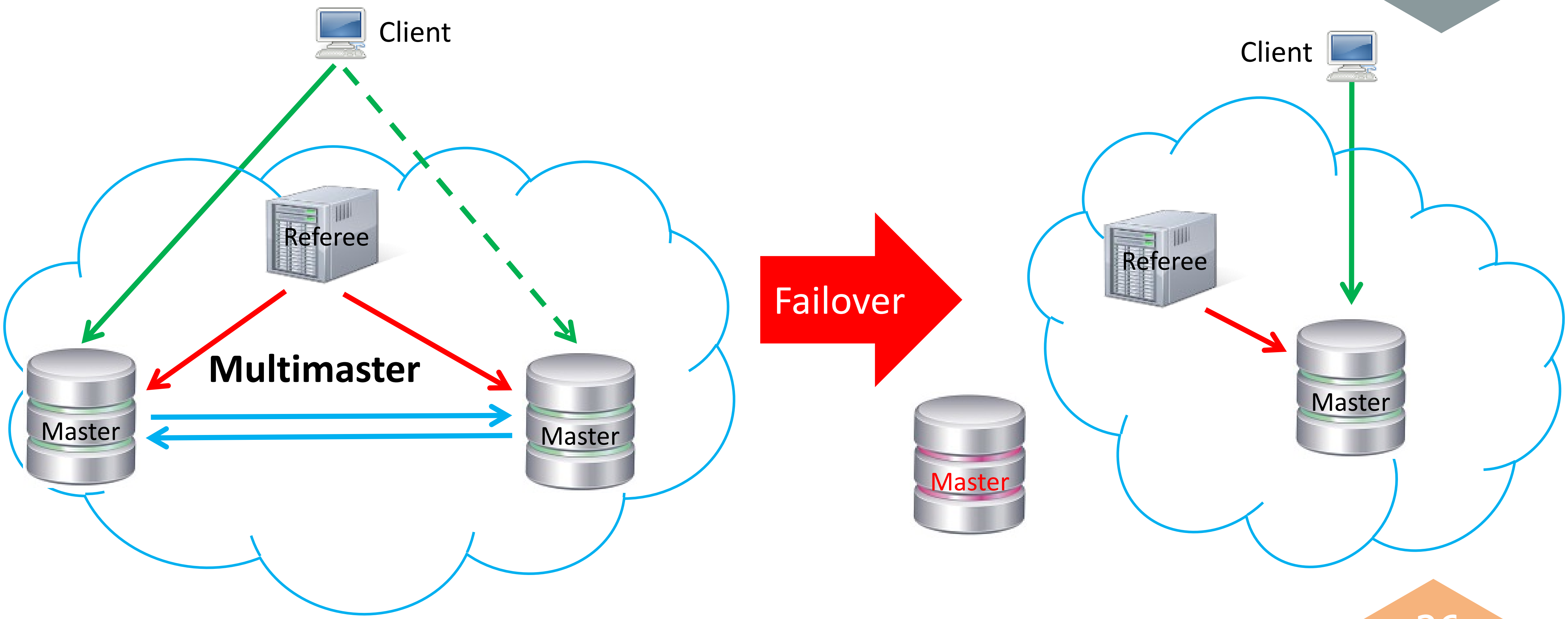
3-phase commit:

1. PREPARE
2. PRECOMMIT
3. COMMIT

Postgres Pro Multimaster (3-node HA-cluster)



Postgres Pro Multimaster (2+1 HA-cluster)



Streaming replication

- Replication is used for disaster recovery purposes (main site lost)
- If the required RPO is strictly zero, synchronous replication has to be used between main and DR sites, otherwise asynchronous replication is enough
- Postgres streaming replication is the most popular solution among the customers, it’s included both in PostgreSQL and Postgres Pro database
- Streaming replication is integrated with HA-clusters (Patroni, Corosync/Pacemaker, Stolon) and uses master/standby(s) configuration, where master is available for read/write requests, while standby(s) can only be used for read-only requests

Logical replication

- Postgres logical replication is slower than streaming replication by design
- Logical replication is more flexible
- replication of only some database objects instead of whole database
- replication between two databases of different major versions
- bi-directional replication between two databases
- Postgres Pro Multimaster uses logical replication to set up an HA-cluster where all nodes can handle read/write requests

Disk/LUN replication

- Replication is on the disk-array level (usually requires a license)
- Disk-arrays have to support this replication on both main and DR sites (have to be of the same type)
- Delivers maximum performance for write intensive load profiles
- Veritas HA-cluster is integrated with disk/LUN replication

Backup (1/2)

- The most popular tool for database backup among our customers is ‘pg_probackup’ - https://github.com/postgrespro/pg_probackup

- supports both full and incremental backup/restore
- supports point-in-time-recovery (PITR)
- provides backup catalog
- supports backup compression
- backup validation without actual data restore
- parallelism of backup/restore tasks
- and many more

Backup (2/2)

- The other popular way for backup/restore employs disk-array snapshots, which allows to do backup/restore of the database very fast (seconds to single digit minutes) regardless of its size

- Built-in PostgreSQL - ‘pg_basebackup’

- full backups only (no incremental backups)

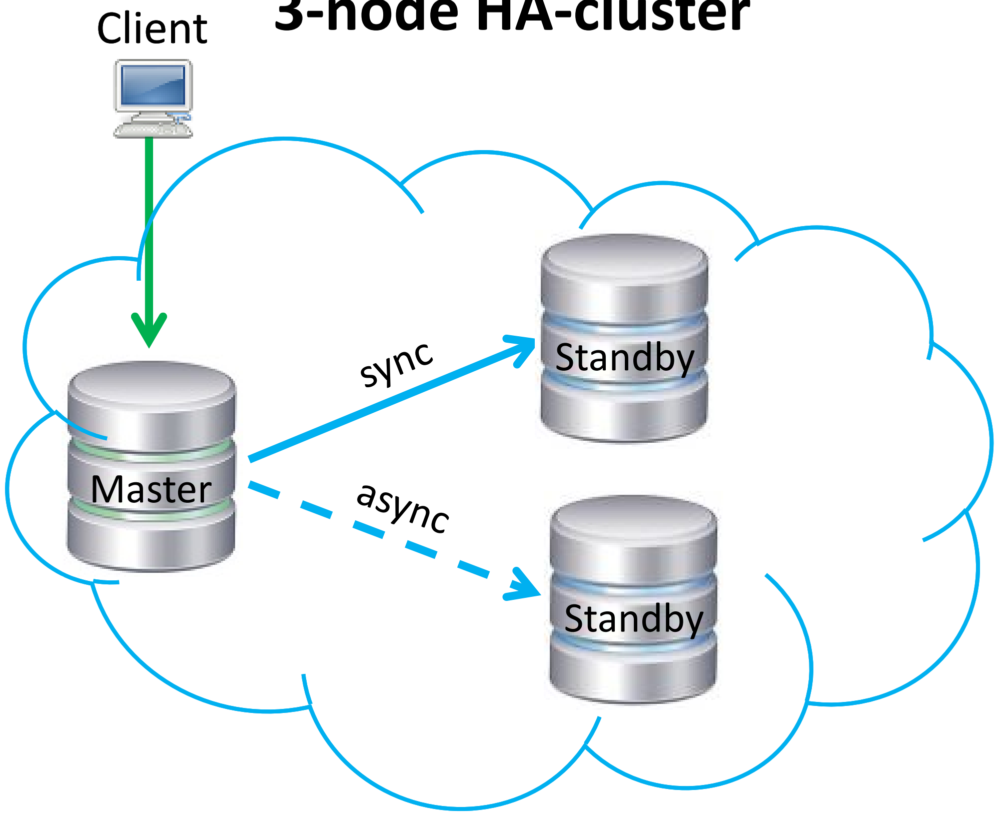
- no parallelism of backup/restore tasks

- Built-in PostgreSQL - ‘pg_dump’ and ‘pg_dumpall’

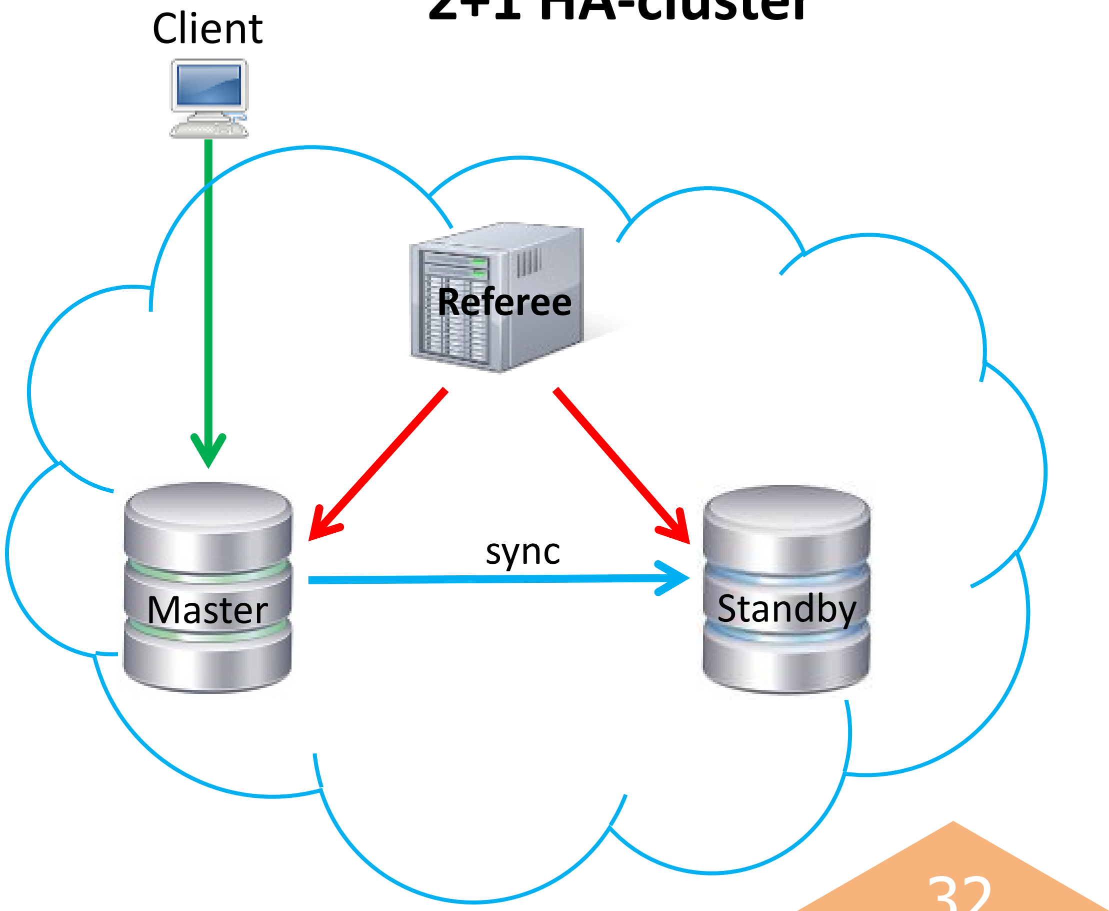
- logical backup (no PITR)

Typical HA/DR architectures use HA-clusters, replication and backup:
- local HA-cluster (all nodes within one site)
- stretched HA-cluster (between two or three sites, up to 30 km to each other)

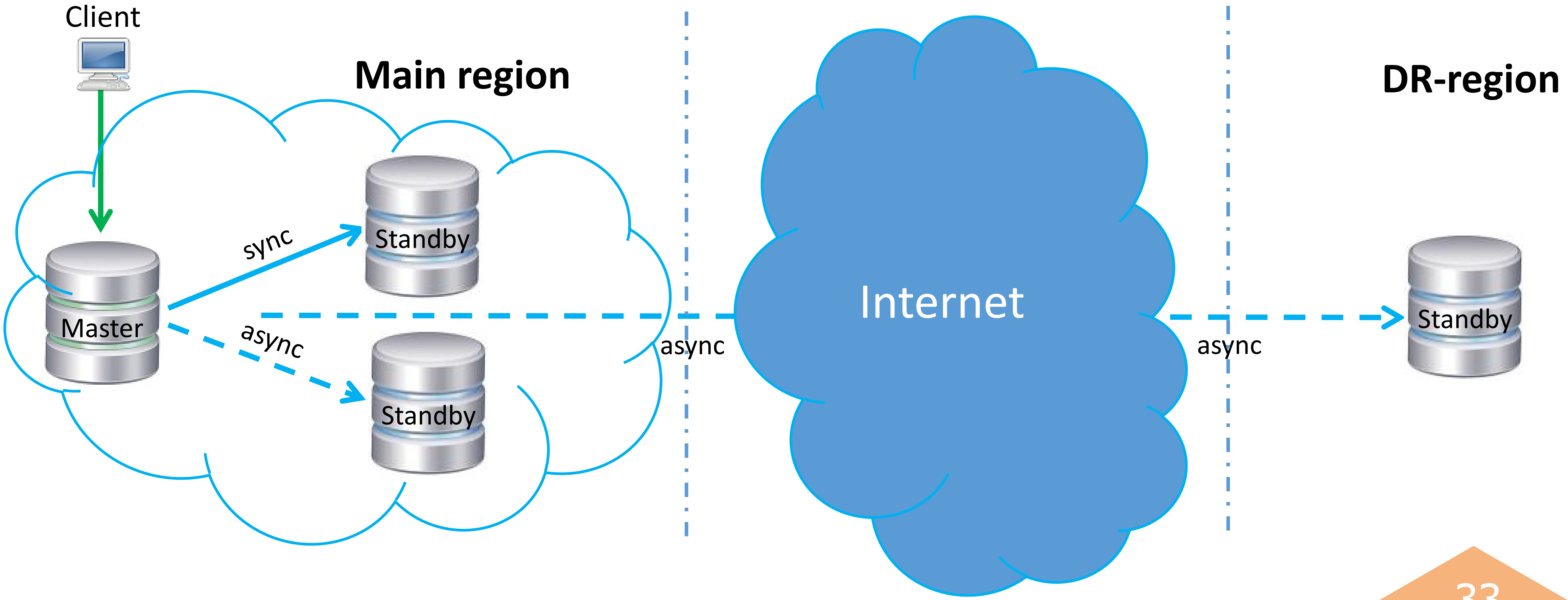
3-node HA-cluster



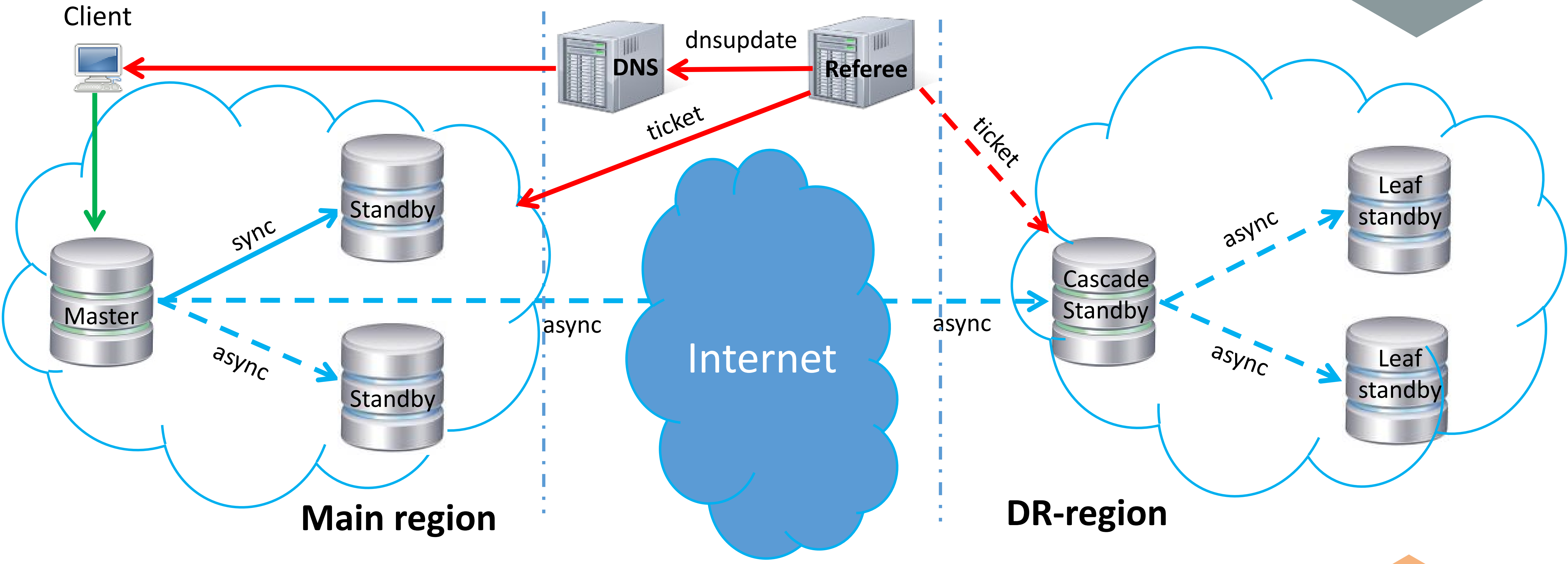
2+1 HA-cluster



Geo HA-cluster (50+ km between regions) (1/2)



Geo HA-cluster (50+ km between regions) (2/2)



Postgres Pro enhancements (1/2)

- Support for relaxed synchronous replication restrictions, which allows the master server to continue running while some of the standbys are temporarily unavailable:

https://postgrespro.com/docs/enterprise/13/runtime-config-replication#GUC-SYNCHRONOUS-STANDBY-GAP

- Automatic database block repair via streaming replication from standby in case of data corruption:

https://postgrespro.com/docs/enterprise/13/warm-standby#REPAIR-PAGE-FROM-STANDBY

Postgres Pro enhancements (2/2)

- Corrupted WAL data restore from in-memory WAL buffers:

https://postgrespro.com/docs/enterprise/13/wal-restoration

- Support for database minor version upgrades without a database instance restart:

https://postgrespro.com/docs/enterprise/13/release-proee-13-2-1

- Compressed file system (CFS) offers database compression at the database block level:

https://postgrespro.com/docs/enterprise/13/cfs

Documentation links

- PostgreSQL ‘High Availability, Load Balancing, and Replication’ documentation:

https://www.postgresql.org/docs/14/high-availability.html

- PostgreSQL ‘Backup and Restore’ documentation:

https://www.postgresql.org/docs/14/backup.html

- Postgres Pro Enterprise documentation:

https://postgrespro.com/docs/enterprise/13/index

Architectures for PostgreSQL High Availability and Disaster Recovery (HA/DR)

Q & A

postgrespro.com

Postgres Professional

https://postgrespro.com/

info@postgrespro.com

postgrespro.com