OTUS

Онлайн образование

otus.ru

Проверить, идет ли запись

Меня хорошо видно && слышно?

Patroni on premise

Аристов Евгений

telegram @AEugene

https://aristov.tech

Правила вебинара

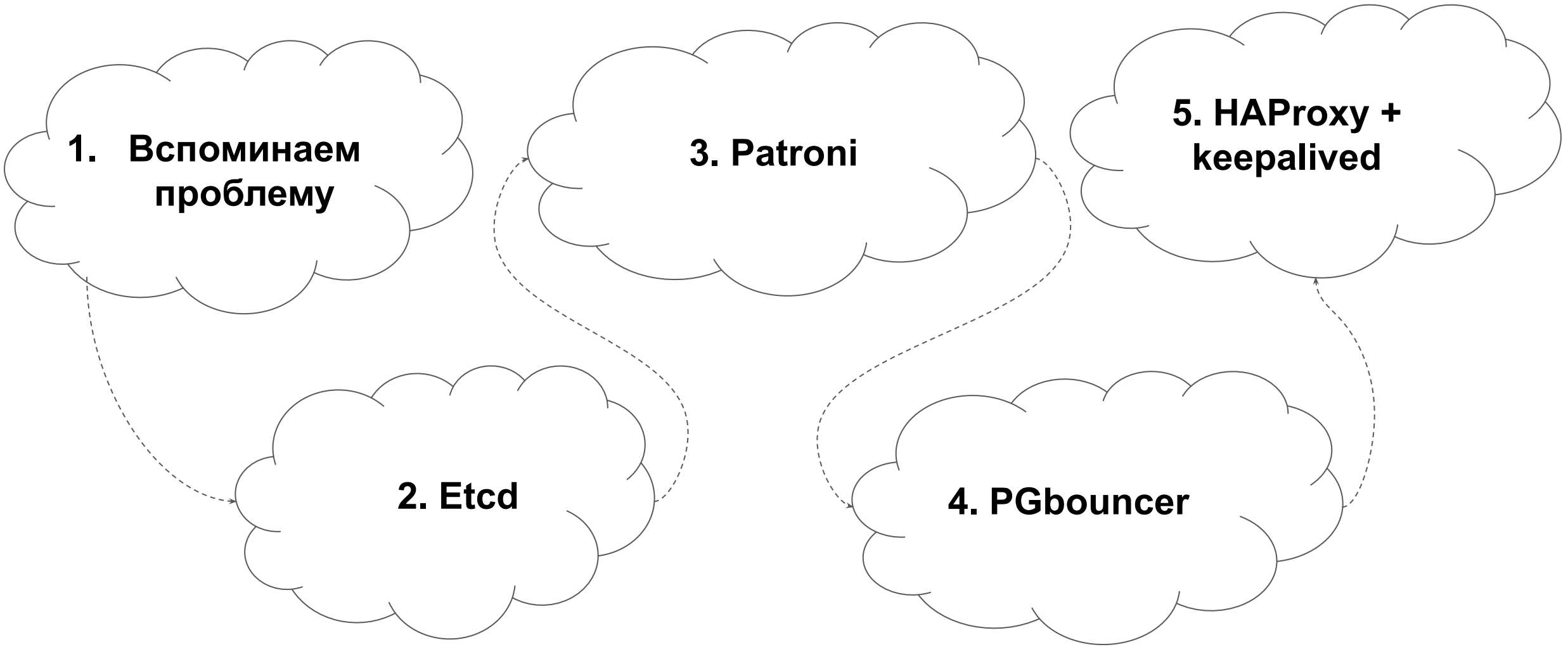
Активно участвуем

Задаем вопрос в чат

Вопросы вижу в чате, могу ответить не сразу

Если остались вопросы/офтопик в канал Slack

Маршрут вебинара



Цели вебинара | После занятия вы сможете

1 Установить и настроить отказоустойчивый кластер Постгреса на базе Патрони

Смысл | Зачем вам это уметь, в результате:

1 Чтобы спать спокойно)

Проблема

Репликация PostgreSQL

master slave

тип

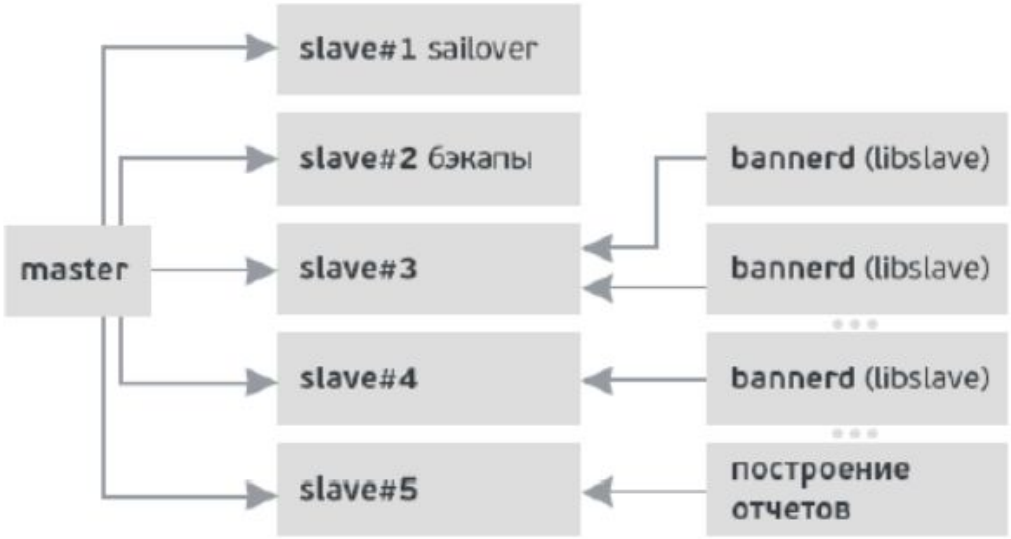
- логическая
- физическая

задержка

- синхронная
- асинхронная

доступность реплики

- warm
- hot



*Структура проекта Mail.Ru Target

Репликация PostgreSQL - чего не хватает?

Репликация PostgreSQL - чего не хватает?

управление трафиком

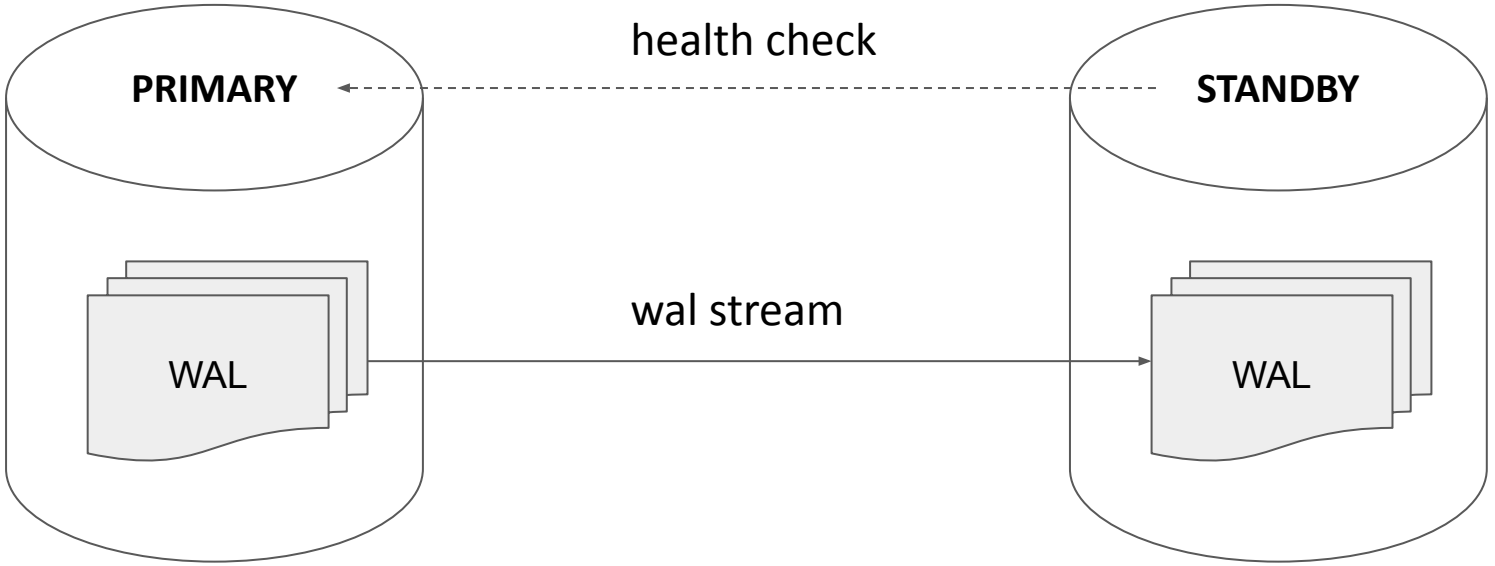
- запросы должны автоматически отправляется на активный сервер
- в случае hot standby запросы на чтение могут автоматически отправляться на standby сервер

управление сервисами

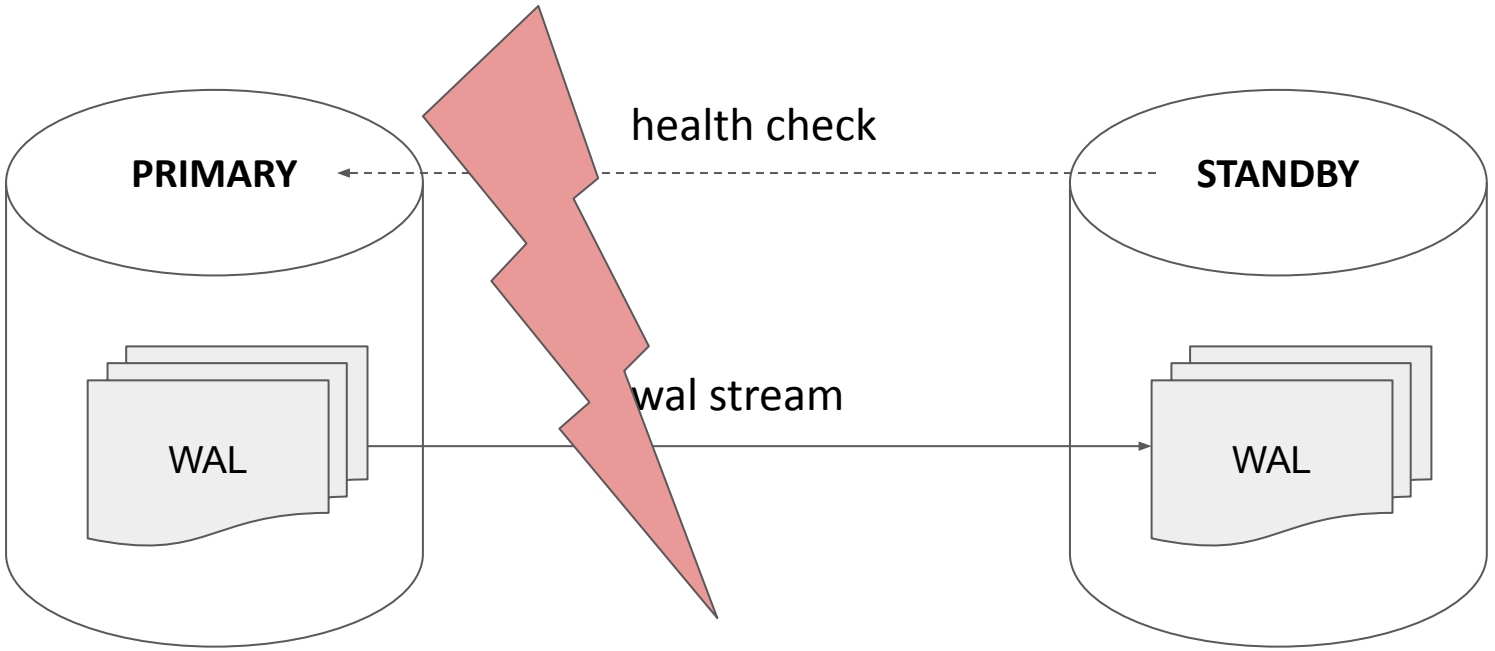
- failover
- failback

Посмотрим в чем проблема

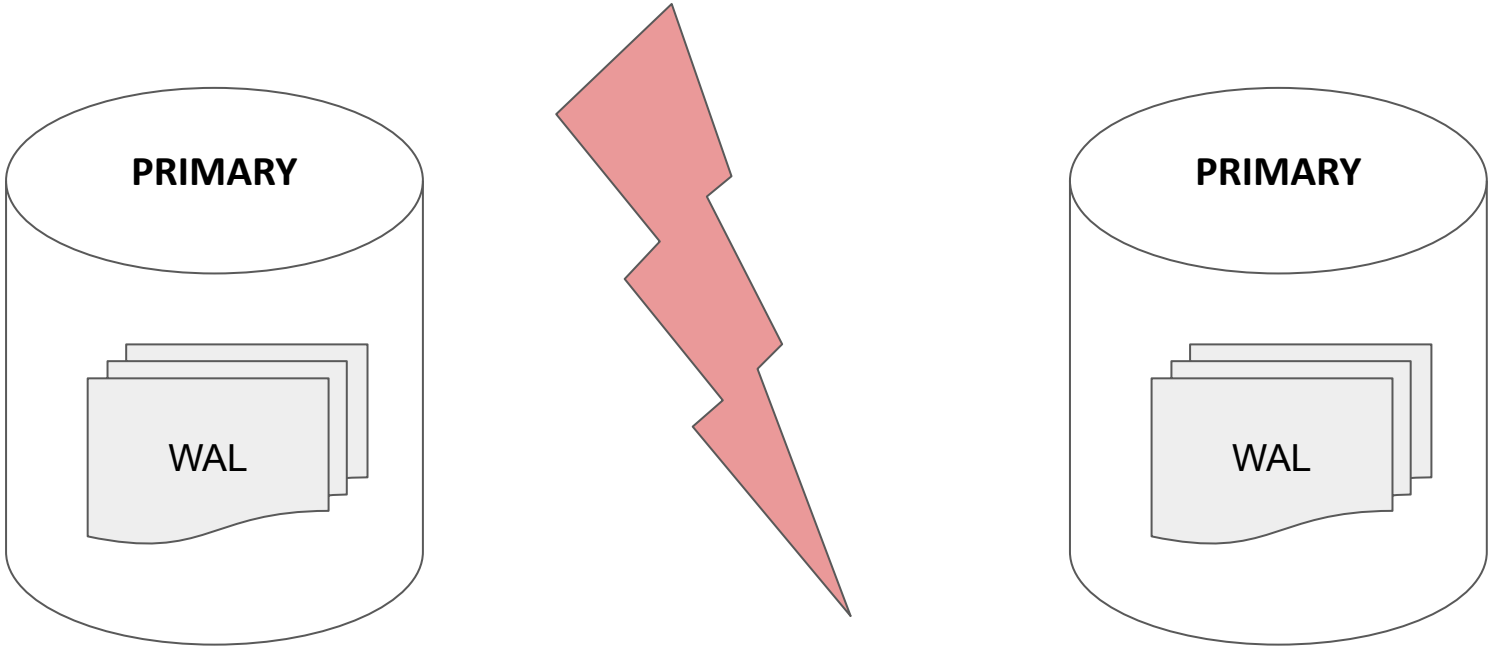
Кластер из 2 нод



Что произойдет при обрыве сети?

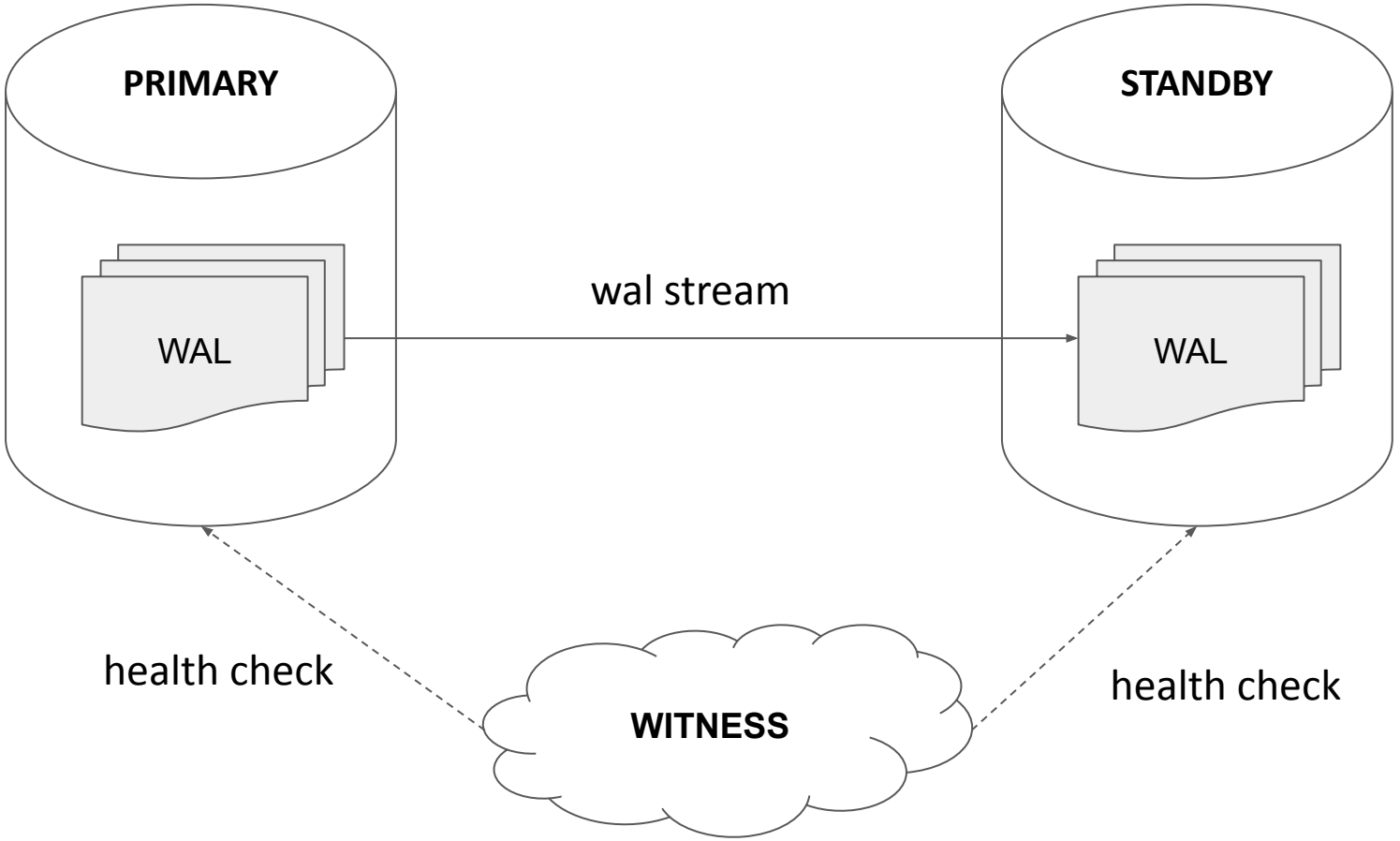


Правильно! Splitbrain



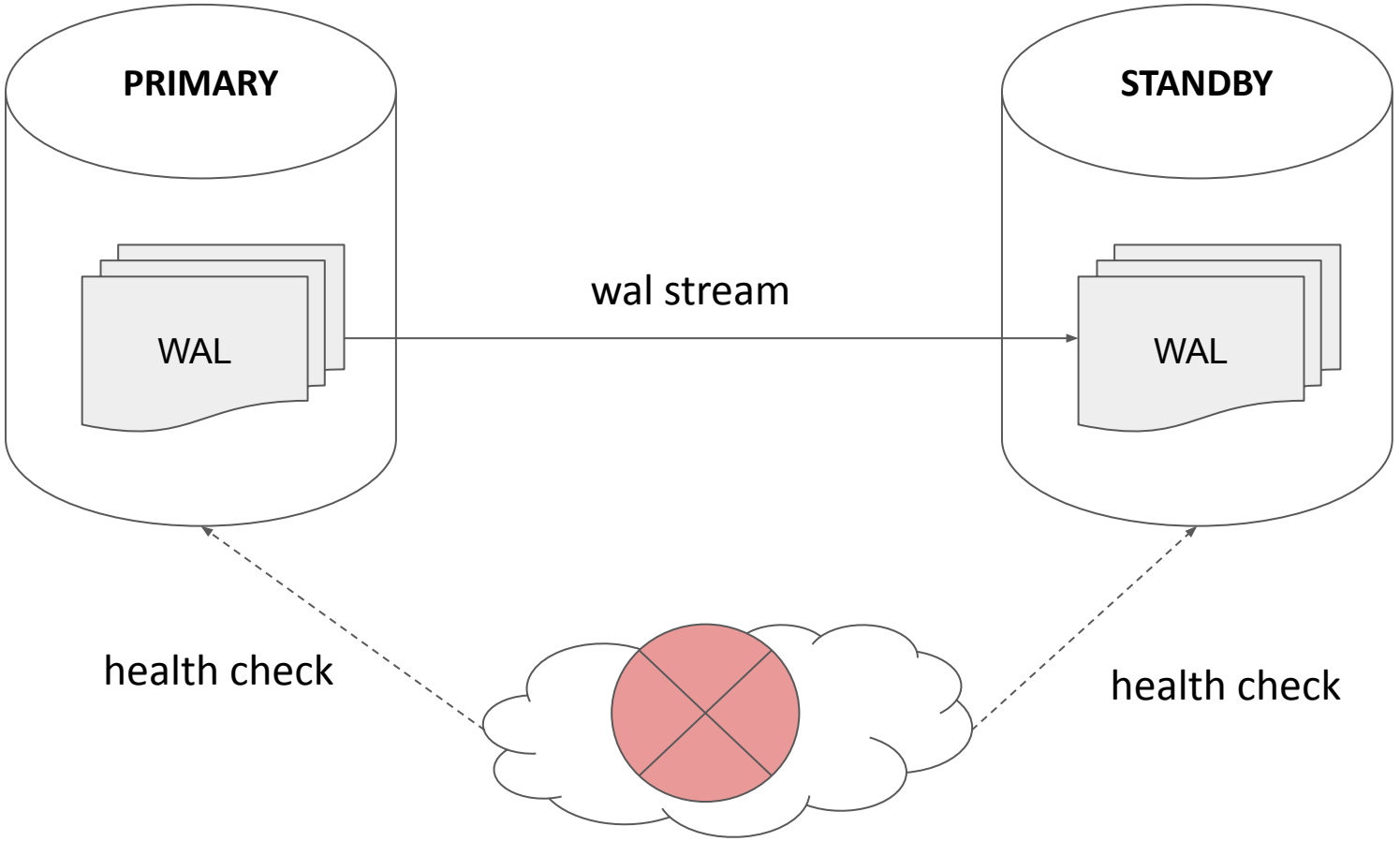
Кластер из 2 нод с Witness

Что может пойти не так?



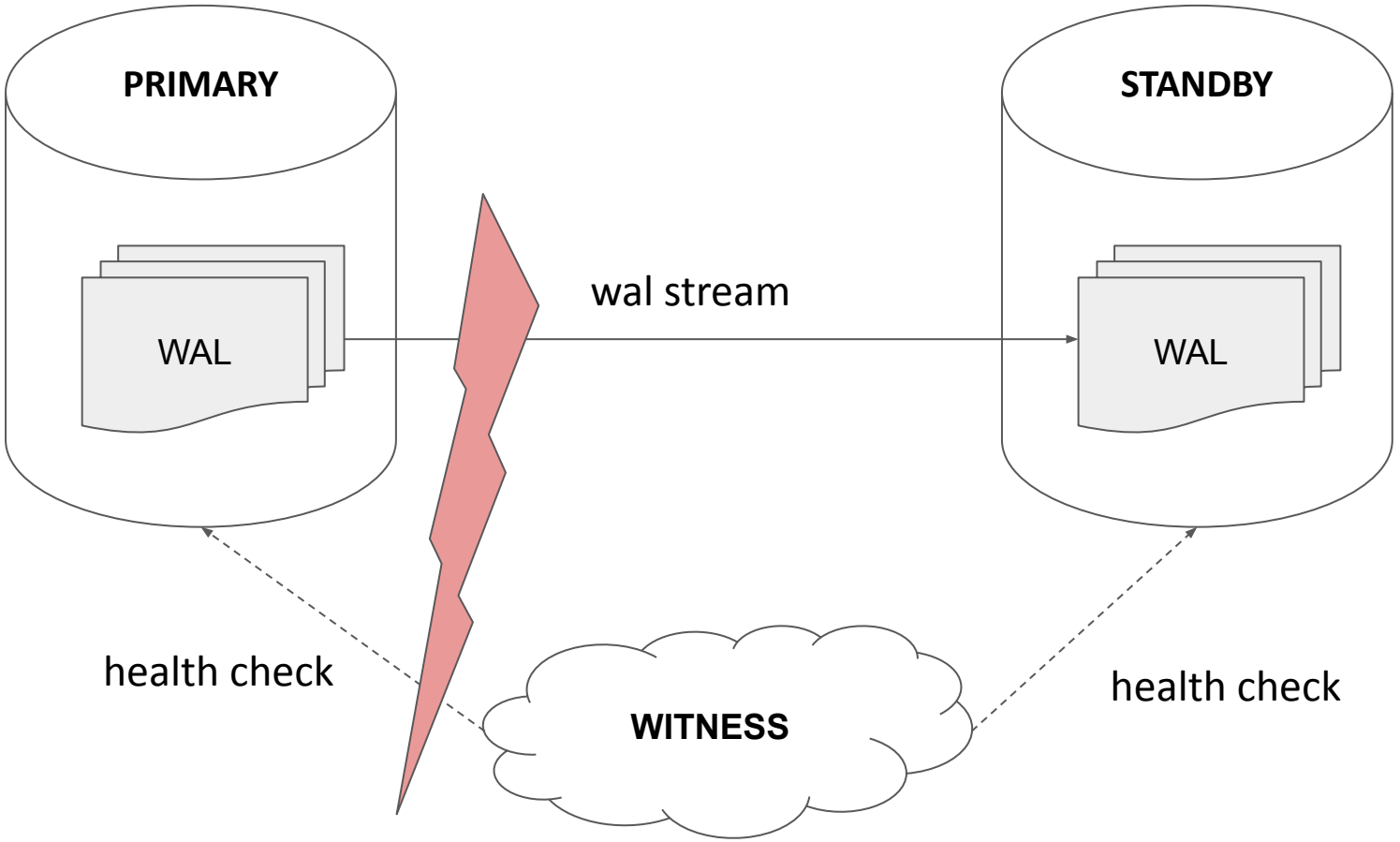
Кластер из 2 нод с Witness

Умрет наблюдатель



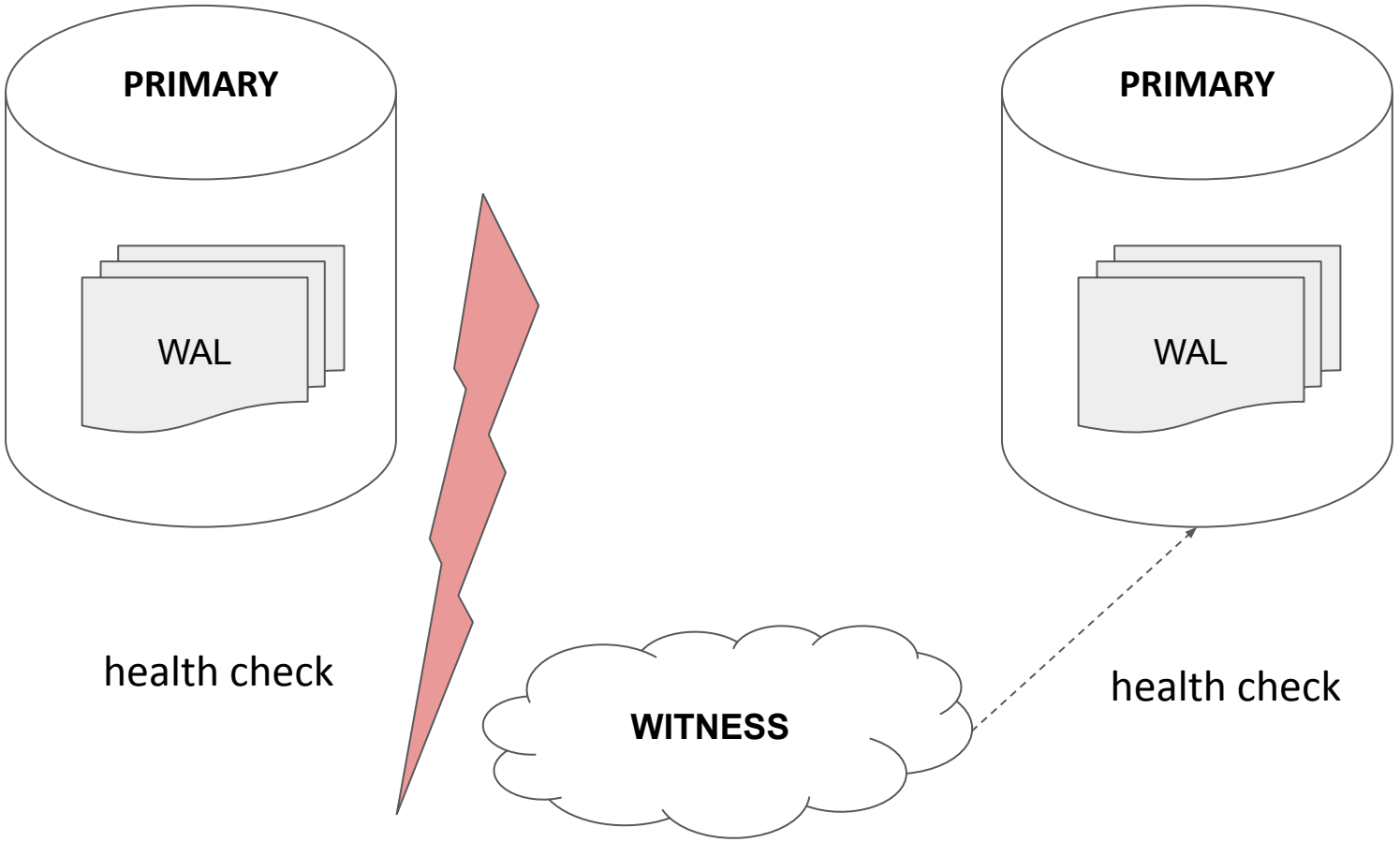
Кластер из 2 нод с Witness

Обрыв соединения с основной нодой



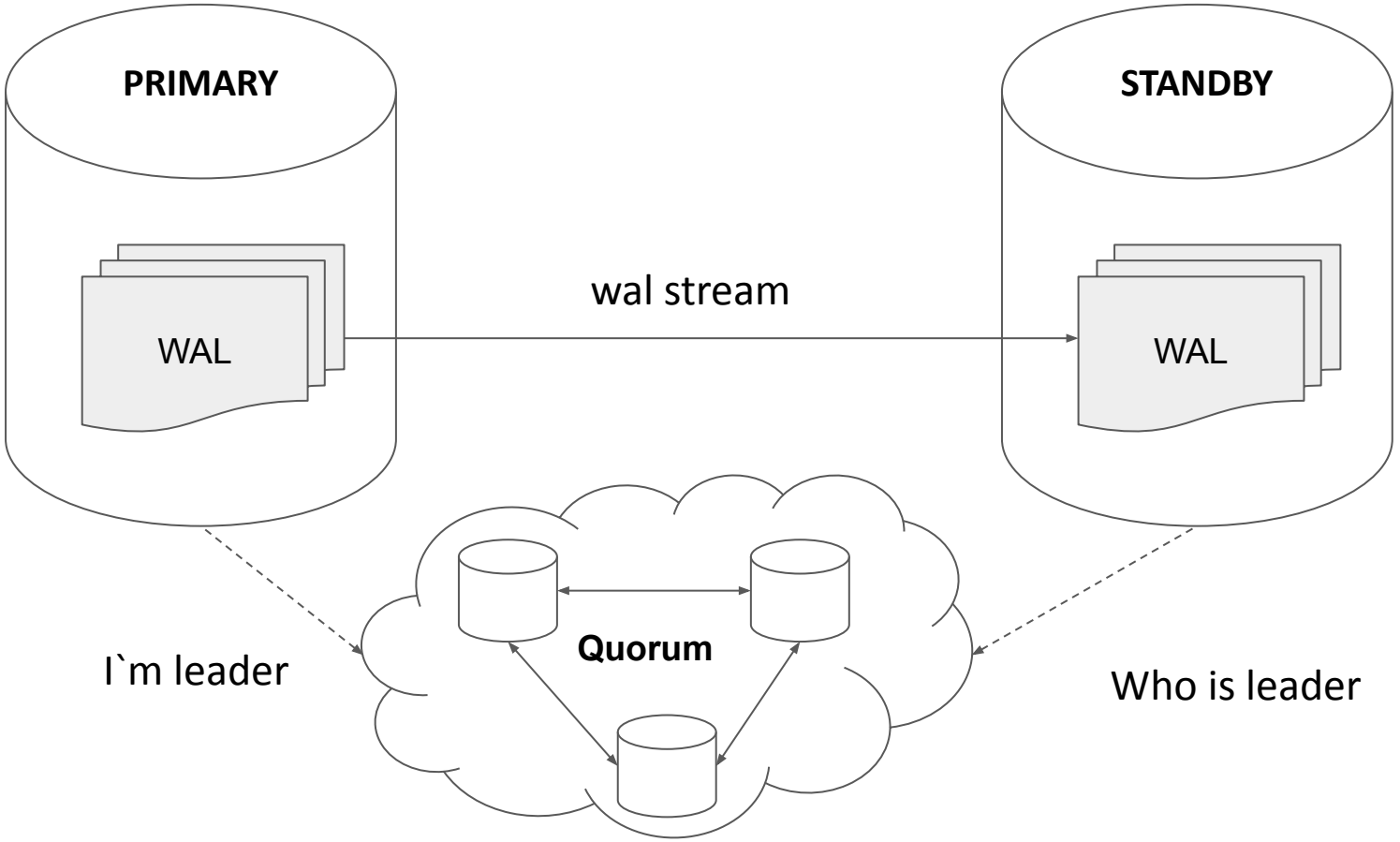
Кластер из 2 нод с Witness

И опять splitbrain

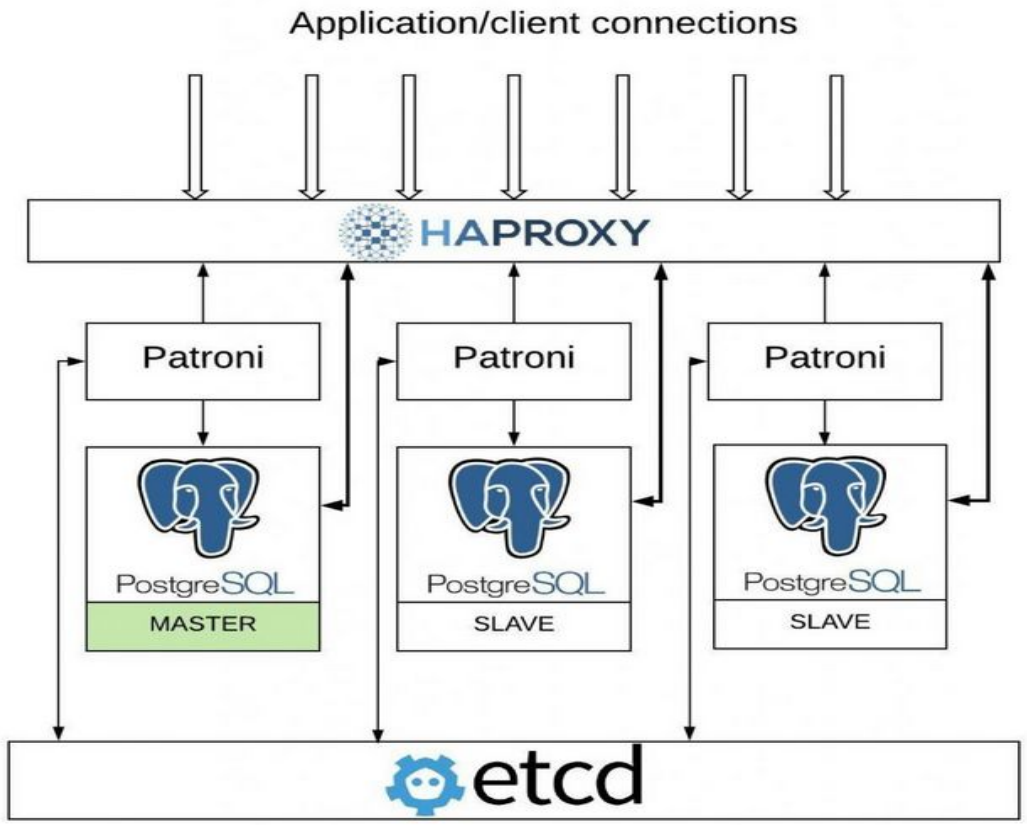


Кластер из 2 нод с ETCD/CONSUL/ZOOKEEPER

Одно из решений (Patroni)



Patroni



можно еще добавить pgbouncer, keepalived+2 HAPROXY для HA

Patroni

Функции DCS (*distributed* control system)

- etcd (или Consul, Zookeeper) хранят информацию о том, кто сейчас лидер
- DCS хранит конфигурацию кластера
- помогает решить проблему с партиционированием сети
- STONITH
- Неплохо бы иметь watchdog (Например, Nomad by HashiCorp)

Patroni

Consul
- Service check
- + Consul templates
- Есть GUI =)
- Есть свой DNS
- Patroni может анонсировать master/replica
- ETCD при большой загрузке замечен в высокой нагрузке на дисковую подсистему (обычно когда кладут на те же ноды с Потсгресом)

с версии 2.0 нативно поддерживает выборы нового мастера по рафт протоколу без использования etcd/consul
https://raft.github.io/

Patroni

Зачем нужен Patroni

- PostgreSQL не умеет взаимодействовать с etcd
- Демон на питоне будет запущен рядом с PostgreSQL
- Демон умеет взаимодействовать с etcd
- Демон принимает решение promotion/demotion

DCS на основе ETCD

Etcd

etcd - это распределенное хранилище данных вида "**ключ-значение**"

Особенности:

- хранение небольших объемов метаданных в виде ключей относительно небольшого размера
- полная репликация между нодами и высокая степень доступности
- **все данные пишутся на диск**, in-memory отсутствует
- использует для работы алгоритм консесуса RAFT
- написан на Go, кроссплатформенный, имеет небольшой размер и большое сообщество

Etcd

Преимущества:

- простой API интерфейс http + json
- иерархическая структура хранения данных по аналогии с файловой системой
- возможность отслеживание изменений (watch) и реакция на них (в этом качестве используется в Kubernetes)
- распределенные блокировки
- транзакции
- B-tree индексы для ключей
- полностью ACID

Etcd

Алгоритм консенсуса Raft
Особенности:

- чёткое разделение фаз (декомпозиция задачи управления кластером на несколько, слабо связанных, подзадач)
- явно выделенный лидер (алгоритм предполагает, что в кластере всегда существует явно выделенный лидер)
- протоколы работы не могут содержать пропусков (записи добавляются строго последовательно)
- изменение размера кластера по количеству нод (Raft позволяет легко менять конфигурацию кластера, не останавливая его работы)

Etcd

Ограничения алгоритма Raft:

- механизм согласования в кластере - консенсус
- количество нод в кластере должно быть равно (n / 2) + 1
- все сообщения на запись отправляются на ноду-лидер
- каждый узел кластера хранит полную копию данных ноды-лидера
- чтение может проходить на любом узле кластера
- прежде чем сохранить данные большинство узлов должны подтвердить вставку
- если умирает нода-лидер, кластер ждет определенное время и начинает голосование за нового лидера, все сообщения в это время помещаются в специальную очередь до выбора нового лидера

Etcd

Алгоритм консенсуса Raft который использует etcd имеет ряд ограничений:

- полностью исключает возможность split-brain/multimaster, так как для консенсуса нужно иметь большинство живых узлов
- это же может привести к ситуации полного развала кластера ввиду недостатка кворума

Etcd

https://raft.github.io/

Кластер Etcd

Кластер Etcd

Особенности:

- минимально отказоустойчивый кластер можно собрать из 3 нод
- допустимое количество вышедших из строя нод можно посмотреть в таблице:

  https://etcd.io/docs/v2/admin_guide/

- теоретически количество нод не ограничено, но надо помнить о том, что любое изменение данных согласуют все ноды кластера
- задержка записи-чтения (так как используется запись на диск) в свою очередь приводит к нестабильной работе кластера и постоянным переизбраниям мастера
- так как плохо переживает нагрузку на дисковую подсистему - не рекомендуется размещать ноды кластера на используемых уже в продакшне ВМ
- *рекомендация размера кластера - 5 нод*

Кластер Etcd

Ограничение на размер запроса:

- etcd спроектирована в расчете на небольшие размеры хранимых ключей.
- большие запросы будут работать, однако это может увеличить задержку ответа
- размер запроса по-умолчанию - 1,5 мб, его можно поменять с помощью флага --max-request-bytes

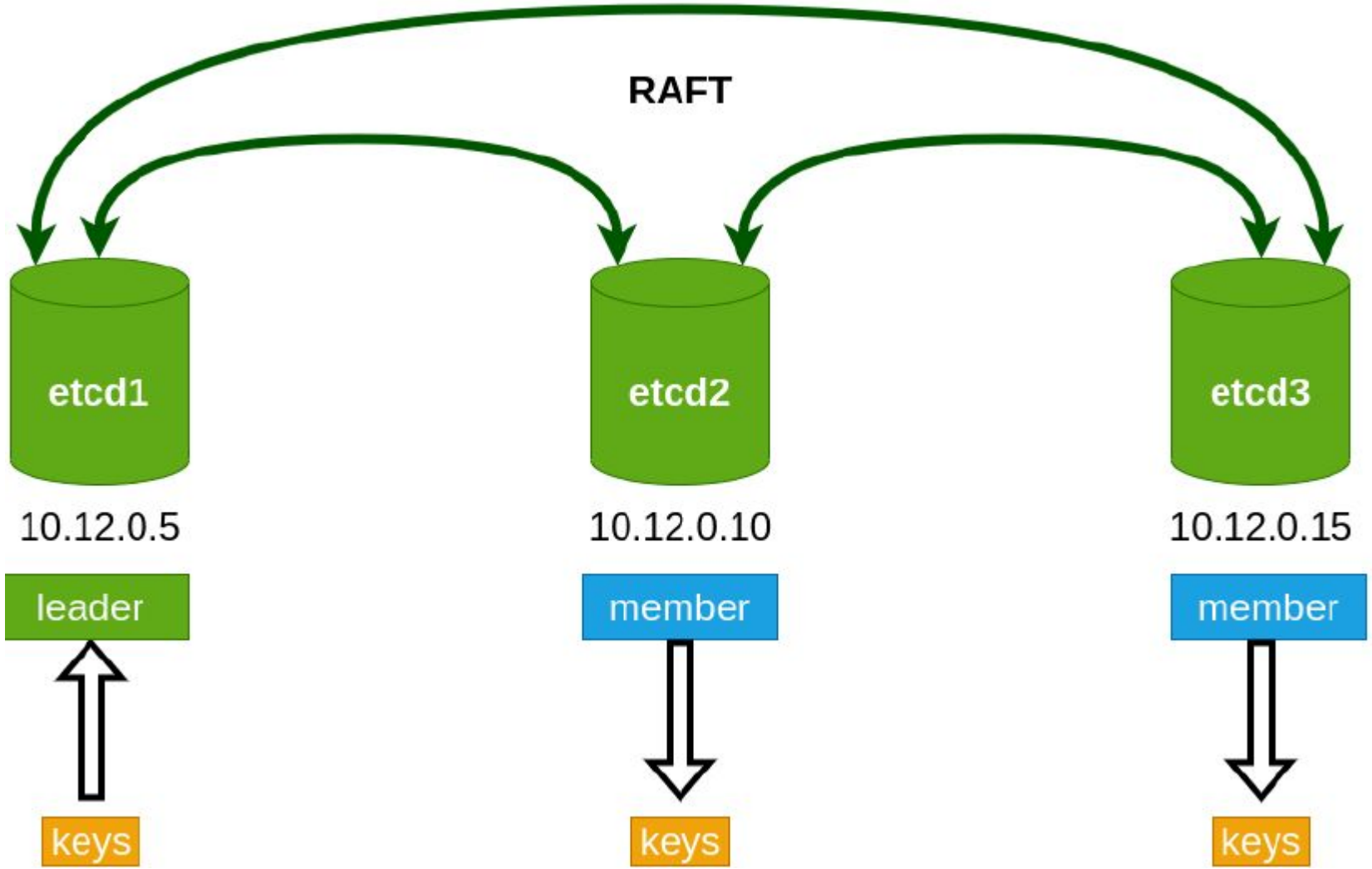
Кластер Etcd

Ограничение на размер хранилища:

- размер хранилища по-умолчанию - 2 Гб
- можно расширить с помощью флага --quota-backend-bytes
- рекомендуемый размер хранилища для нормальной работы кластера - 8 Гб
- больше делать не рекомендуется из-за процессов сжатия и дефрагментации

Кластер Etcd

Схема тестового стенда:



Кластер Etcd

Установка etcd:

sudo apt install etcd

Конфигурация ноды etcd1 /etc/etcd/etcd.conf:

ETCD_NAME="etcd1"

ETCD_LISTEN_CLIENT_URLS="http://0.0.0.0:2379"

ETCD_ADVERTISE_CLIENT_URLS="http://10.12.0.5:2379"

ETCD_LISTEN_PEER_URLS="http://0.0.0.0:2380"

ETCD_INITIAL_ADVERTISE_PEER_URLS="http://10.12.0.5:2380"

ETCD_INITIAL_CLUSTER_TOKEN="**TestCluster**"

ETCD_INITIAL_CLUSTER="etcd1=http://10.12.0.5:2380,etcd2=http://10.12.0.10:2380,etcd3=http://10.12.0.15:2380”

ETCD_INITIAL_CLUSTER_STATE="**new**"

ETCD_DATA_DIR="/var/lib/etcd"

ETCD_ELECTION_TIMEOUT="5000"

ETCD_HEARTBEAT_INTERVAL="1000"

Кластер Etcd

Конфигурация ноды etcd2 /etc/etcd/etcd.conf:

ETCD_NAME="etcd2"

ETCD_LISTEN_CLIENT_URLS="http://0.0.0.0:2379"

ETCD_ADVERTISE_CLIENT_URLS="http://10.12.0.10:2379"

ETCD_LISTEN_PEER_URLS="http://0.0.0.0:2380"

ETCD_INITIAL_ADVERTISE_PEER_URLS="http://10.12.0.10:2380"

ETCD_INITIAL_CLUSTER_TOKEN="TestCluster"

ETCD_INITIAL_CLUSTER="etcd1=http://10.12.0.5:2380,etcd2=http://10.12.0.10:2380,etcd3=http://10.12.0.15:2380”

ETCD_INITIAL_CLUSTER_STATE="new"

ETCD_DATA_DIR="/var/lib/etcd"

ETCD_ELECTION_TIMEOUT="5000"

ETCD_HEARTBEAT_INTERVAL="1000"

Кластер Etcd

Конфигурация ноды etcd3 /etc/etcd/etcd.conf:

ETCD_NAME="etcd3"

ETCD_LISTEN_CLIENT_URLS="http://0.0.0.0:2379"

ETCD_ADVERTISE_CLIENT_URLS="http://10.12.0.15:2379"

ETCD_LISTEN_PEER_URLS="http://0.0.0.0:2380"

ETCD_INITIAL_ADVERTISE_PEER_URLS="http://10.12.0.15:2380"

ETCD_INITIAL_CLUSTER_TOKEN="TestCluster"

ETCD_INITIAL_CLUSTER="etcd1=http://10.12.0.5:2380,etcd2=http://10.12.0.10:2380,ETCD_INITIAL_CLUSTER_STATE="new"

ETCD_DATA_DIR="/var/lib/etcd"

ETCD_ELECTION_TIMEOUT="5000"

ETCD_HEARTBEAT_INTERVAL="1000"

Кластер Etcd

WARNING:

Environment variable ETCDCTL_API is not set; defaults to etcdctl v2.

Set environment variable ETCDCTL_API=3 to use v3 API or ETCDCTL_API=2 to use v2 API.

Экспорт переменной с версией API etcd, чтобы не объявлять ее каждый раз:

```
export ETCDCTL_API=3
```

Посмотреть статус текущей ноды:

```
etcdctl --write-out=table endpoint status
```

Посмотреть статусы нод кластера:

```
etcdctl member list
```

Кластер Etcd

Работа с кластером

После создания кластера есть рекомендация в конфиге поправить new->existing

Добавить участника в кластер:

etcdctl member add etcd4 --peer-urls=http://10.12.0.20:2380

ETCD_NAME="etcd4"

ETCD_INITIAL_CLUSTER="etcd3=http://10.12.0.15:2380,etcd2=http://10.12.0.10:ETCD_INITIAL_ADVERTISE_PEER_URLS="http://10.12.0.20:2380"

ETCD_INITIAL_CLUSTER_STATE="existing"

Удалить участника:

etcdctl member remove <member_id>

Patroni

Patroni

Создать свой первый кластер Patroni:
patroni /etc/patroni.yml OR **systemctl start patroni.service**
INFO: Selected new etcd server http://10.128.0.48:2379
INFO: Lock owner: None; I am pg01
trying to bootstrap a new cluster§
LOG: listening on IPv4 address "10.128.0.49", port 5432
INFO: establishing a new patroni connection to the postgres cluster
INFO: Lock owner: pg01; I am pg01
INFO: no action. i am the leader with the lock

Patroni

Состояние кластера

- patronictl - утилита для управления кластером
- patronictl -c /etc/patroni.yml list

```
[root@pg01 ~]# patronictl -c /etc/patroni.yml list
+----------+--------+-------------+--------+---------+----+-----------+
| Cluster  | Member |     Host    |  Role  |  State  | TL | Lag in MB |
+----------+--------+-------------+--------+---------+----+-----------+
| postgres |  pg01  | 10.128.0.47 |        | running |  7 |       0.0 |
| postgres |  pg02  | 10.128.0.46 | Leader | running |  7 |       0.0 |
| postgres |  pg03  | 10.128.0.45 |        | running |  7 |       0.0 |
+----------+--------+-------------+--------+---------+----+-----------+
```

Patroni

Автоматический Failover

systemctl stop patroni - любой другой способ протестировать failover =)

• 30 секунд по умолчанию на истечение ключа в DCS

• После чего Patroni стучится на каждую ноду в кластере и спрашивает, не мастер ли ты, проверяет WAL логи, насколько близки они к мастеру. В итоге если WAL логи у всех одинаковые то, промоутится следующий по порядку

• Опрос нод идёт параллельно

Patroni

Важные параметры

Обновление данных в DCS идет циклично:

- **loop_wait** - промежуток в секундах между попытками обновить ключ лидера.
- **retry-timeout** - через сколько будем ретраить попытку обновить ключ
- **ttl** - время жизни ключа лидера. Рекомендация: как минимум loop_wait + retry_timeout, но вообще таким комфортным, чтобы избежать нескольких медленных/неудавшихся вызовов к DCS
- **maximum_lag_on_failover** - максимальное отставание ноды от лидера для того, чтобы участвовать в выборах
- **synchronous_mode** - вкл/выкл синхронной реплики
- **synchronous_mode_strict** - вкл/выкл строго синхронного режима, чтобы мастер останавливался при смерти синхронной реплики

https://patroni.readthedocs.io/en/latest/replication_modes.html

Patroni

Редактирование конфигурации

patronictl -c /etc/patroni.yml edit-config

Ручной Switchover:

patronictl -c /etc/patroni.yml switchover

Перезагрузка

- patronictl -c /etc/patroni.yml restart postgres pg02
- Применение новых параметров требующих обязательной перезагрузки

Реинициализация

- patronictl -c /etc/patroni.yml reinit postgres pg03
- Реинициализирует ноду в кластере. Т.е. по сути удаляет дата директорию и делает pg_basebackup, если это поведение не изменено параметром create_replica_method

Patroni

Локальная конфигурация

Что делать если нужно поменять конфигурацию PostgreSQL только локально:

- patroni.yml
- postgresql.base.conf
- ALTER SYSTEM SET - имеет наивысший приоритет

Некоторые параметры, такие как: max_connections, max_locks_per_transaction, wal_level, max_wal_senders, max_prepared_transactions, max_replication_slots, max_worker_processes **не могут быть переопределены локально - Patroni их перезаписывает**.

https://patroni.readthedocs.io/en/latest/SETTINGS.html?highlight=custom_conf#postgresql

Patroni

Monitoring

Проверка запущен ли PostgreSQL мастер:

• GET /master - должно возвращать 200 ТОЛЬКО для одной ноды, остальные 503 и наоборот GET /replica

Проверка работают ли реплики

• GET /patroni с мастера должно возвращать replication:[{state: streaming}] для всех реплик

Запущен ли сам PostgreSQL:

• GET /patroni должен возвращать state:running для каждой ноды

Отставание реплики:

• GET /patroni - xlog: location с реплик не должен быть далеко от этого же параметра на мастере

Patroni

Роутинг трафика

- HAProxy, TCP Proxy (NGINX)
- Pgbouncer (pgPool, Odyssey)

Patroni

Tags

- nofailover (true/false) - в положении true нода никогда не станет мастером
- noloadbalance (true/false) - /replica всегда возвращает код 503
- clonefrom (true/false) - patronictl выберет предпочтительную ноду для pgbasebackup
- nosync (true/false) - нода никогда не станет синхронной репликой
- replicatefrom (node name) - указать реплику с которой снимать реплику

Patroni

Switchover vs failover

- Switchover
    - Переключение роли Мастера на новую ноду. Делается вручную, по сути плановые работы
- Failover
    - Экстренное переключение Мастера на новую ноду
    - Происходит автоматически
    - Ручной вариант - manual failover - только когда не система не может решить на кого переключать

Patroni

Режим паузы

- Отключается автоматический failover
- Ставиться глобальная пауза на все ноды
- Проведение плановых работ, например с etcd или обновление PostgreSQL

Тем не менее:

- Можно создавать реплики
- Ручной switchover возможен
- patronictl -c /etc/patroni.yml pause|resume

Patroni

Истории аварий с Patroni, или Как уронить PostgreSQL-кластер

Patroni

Параметры в Патрони:
https://patroni.readthedocs.io/en/latest/SETTINGS.html?highlight=custom_conf#postgresql

минитест

???

2-3 минуты

PGBouncer

PGbouncer

PgBouncer

PgBouncer - пул соединений:

- легковесный - 2 Кб на соединение
- можно выбрать тип соединения: на сессию, транзакцию или каждую операцию
- онлайн-реконфигурация без сброса подключений

Сравнение с PGpool2

https://scalegrid.io/blog/postgresql-connection-pooling-part-4-pgbouncer-vs-pgpool/

Конфигурация

http://www.pgbouncer.org/usage.html

HAproxy

HAproxy

Для балансинга нагрузки

https://www.percona.com/blog/2018/10/02/scaling-postgresql-using-connection-poolers-and-load-balancers-for-an-enterprise-grade-environment/

keepalived

VRRP

https://ru.wikipedia.org/wiki/VRRP

Протокол VRRP предназначен для увеличения доступности маршрутизаторов выполняющих роль шлюза по умолчанию.

Для группы маршрутизаторов настраивается их принадлежность **виртуальному маршрутизатору**. Фактически, виртуальный маршрутизатор — это группа интерфейсов маршрутизаторов, которые находятся в одной сети и разделяют **Virtual Router Identifier (VRID)** и **виртуальный IP-адрес**.

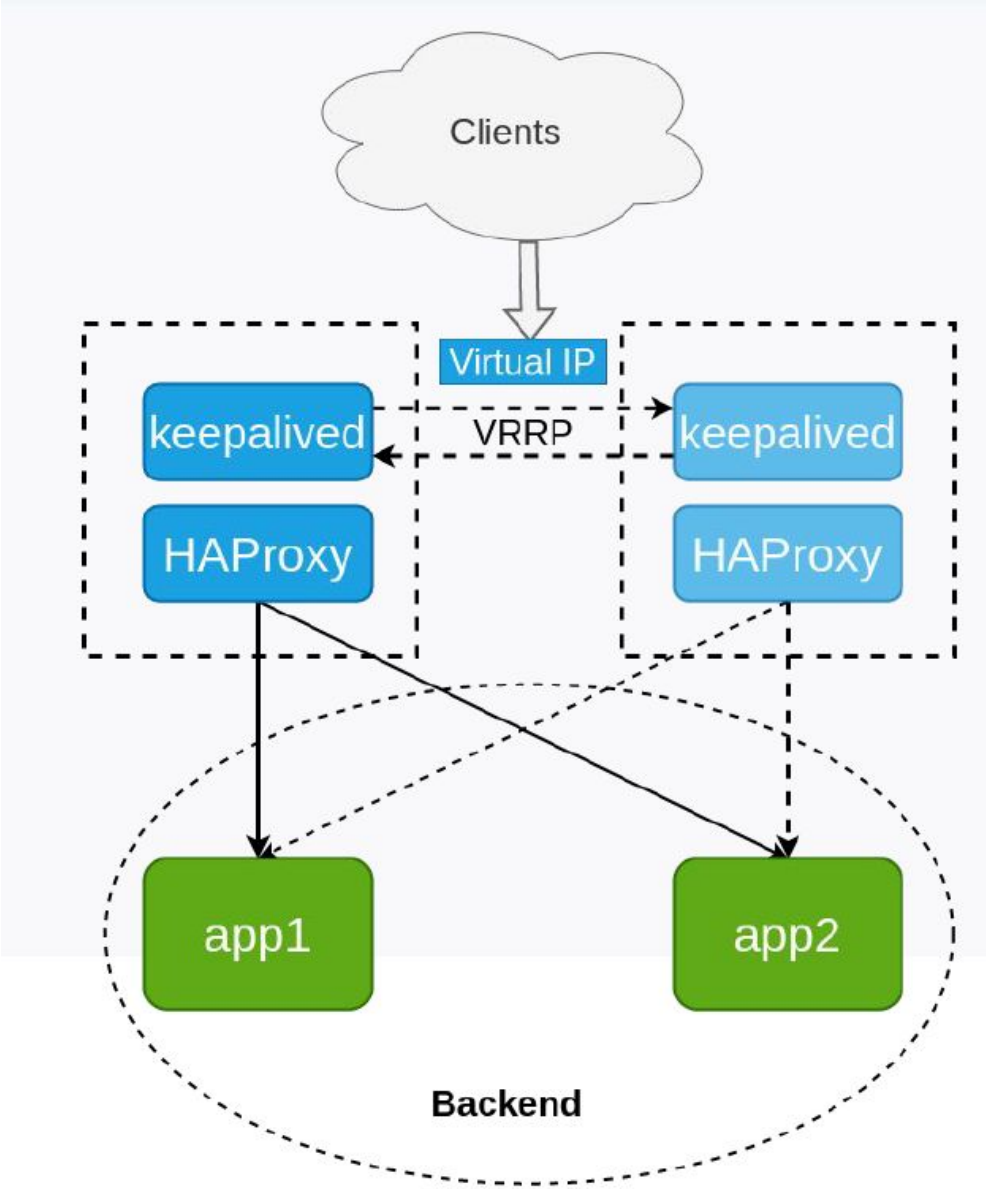
keepalived

https://dasunhegoda.com/how-to-setup-haproxy-with-keepalived/833/

Keepalived is a routing software written in C. The main goal of this project is to provide simple and robust facilities for loadbalancing and high-availability to Linux system and Linux based infrastructures. Loadbalancing framework relies on well-known and widely used Linux Virtual Server (IPVS) kernel module providing Layer4 loadbalancing

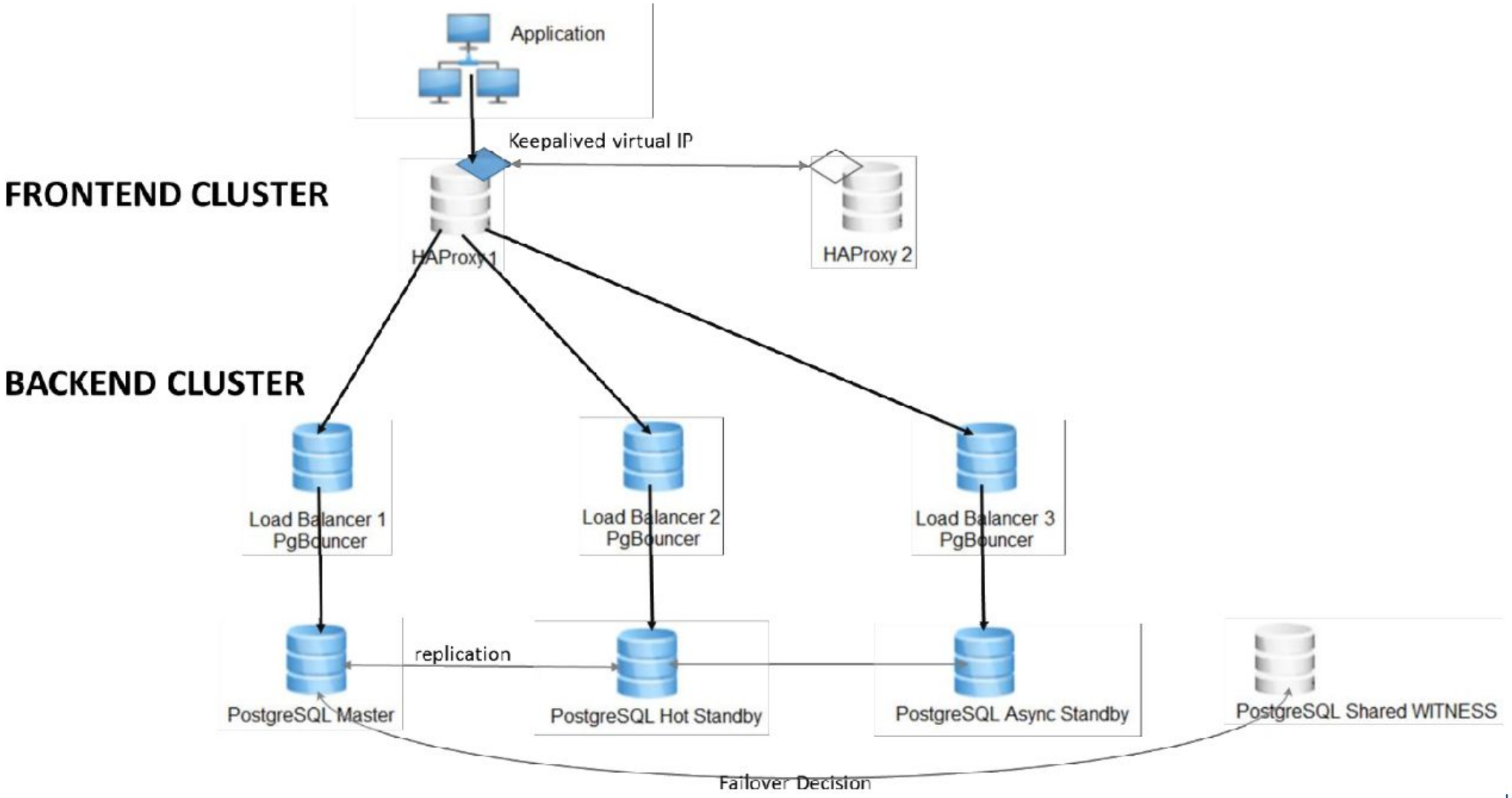
https://keepalived.readthedocs.io/en/latest/software_design.html

keepalived


Clients
Virtual IP
keepalived
VRRP
keepalived
HAProxy
HAProxy
app1
app2
Backend

Неплохой вариант

Как бы вы доработали этот варинат



ДЗ

Домашнее задание

Создаем 3 ВМ для etcd + 3 ВМ для Patroni +1 HA proxy (при проблемах можно на 1 хосте развернуть)

Инициализируем кластер

Проверяем отказоустойсивость

*настраиваем бэкапы через wal-g или pg_probackup

Критерии оценивания:

Выполнение ДЗ: 10 баллов

+5 баллов за задание со *

плюс 2 балла за красивое решение

минус 2 балла за рабочее решение, и недостатки указанные преподавателем не устранены

Рефлексия

Рефлексия

Как вам занятие?

Заполните, пожалуйста,
опрос о занятии по ссылке в чате
https://otus.ru/polls/43962/
https://otus.ru/polls/43424/

Спасибо за внимание!
Приходите на следующие вебинары

Аристов Евгений