PostgreSQL Cluster

Цели

Какие у нас есть варианты

Как делать/не делать Failover

Архитектура Patroni

Создание кластера

Как менять конфигурацию кластера

Немного про ETCD

Перенаправление клиентов на Master

Создание реплик и их реинициализация

Высокая доступность

- Распределенное хранилище
  - NFS NAS/SAN
  - DRBD
  - ISCSI (+ LVM)
- Мульти-мастер
  - BDR, Bucardo
- Логическая репликация
  - pglogical, slony, встроенная фича в postgresql 10
- Физическая репликация
  - В postgresql начиная с 9.0
- Облака: Azure, Amazon: Aurora/RDS

Варианты

- Встроенные решения
- Patroni
- Stolon:
  - Проксирует все запросы в мастер ноду. Нельзя давать нагрузку на реплики
  - Мастер выбирается самостоятельно при switchover-е
- repmgr:
  - Нет фэнсинга из коробки (защита от двойного мастера)
  - Нет нужды в DCS - на мой взгляд это минус

Slony

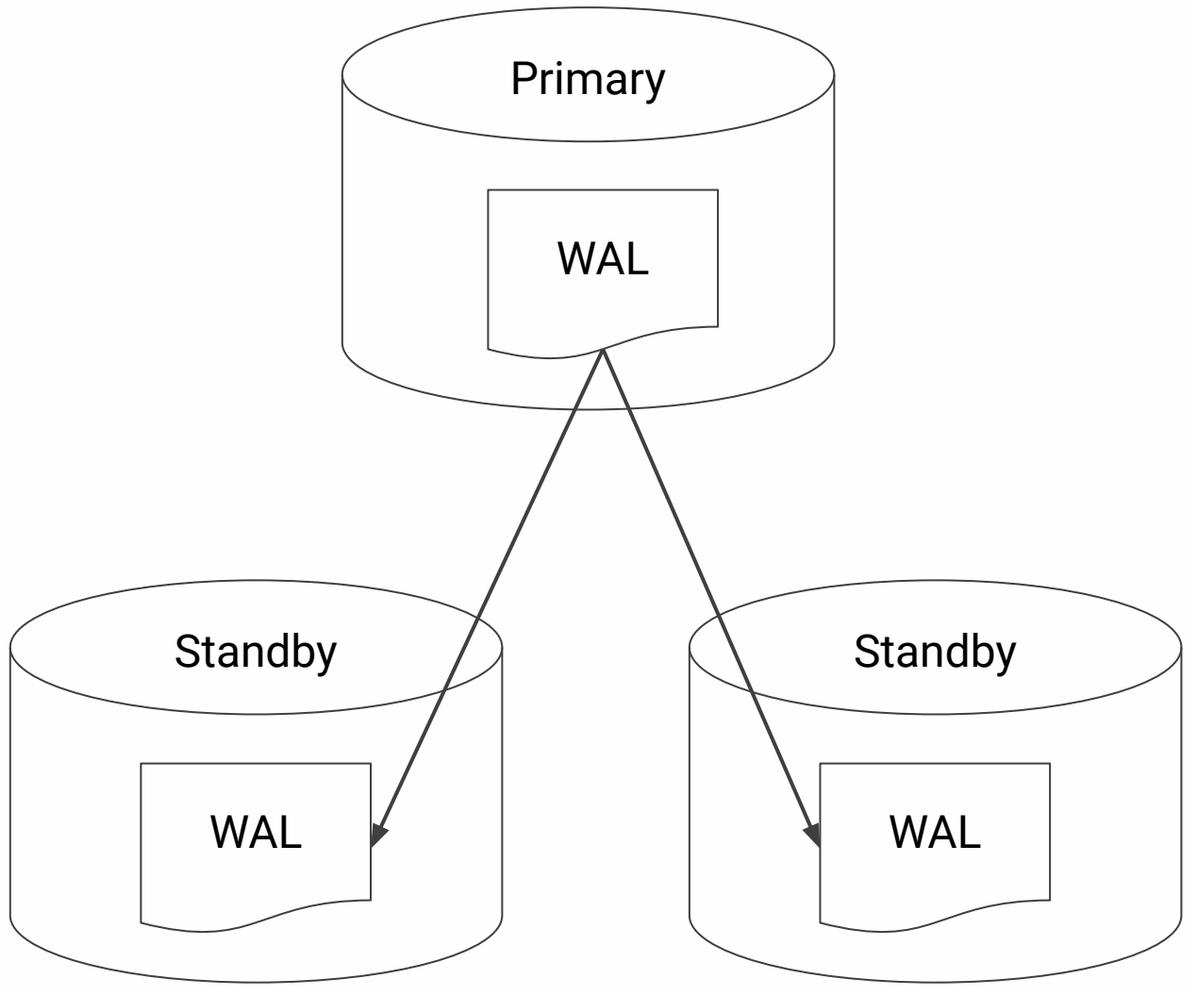
Физическая репликация

Плюсы:

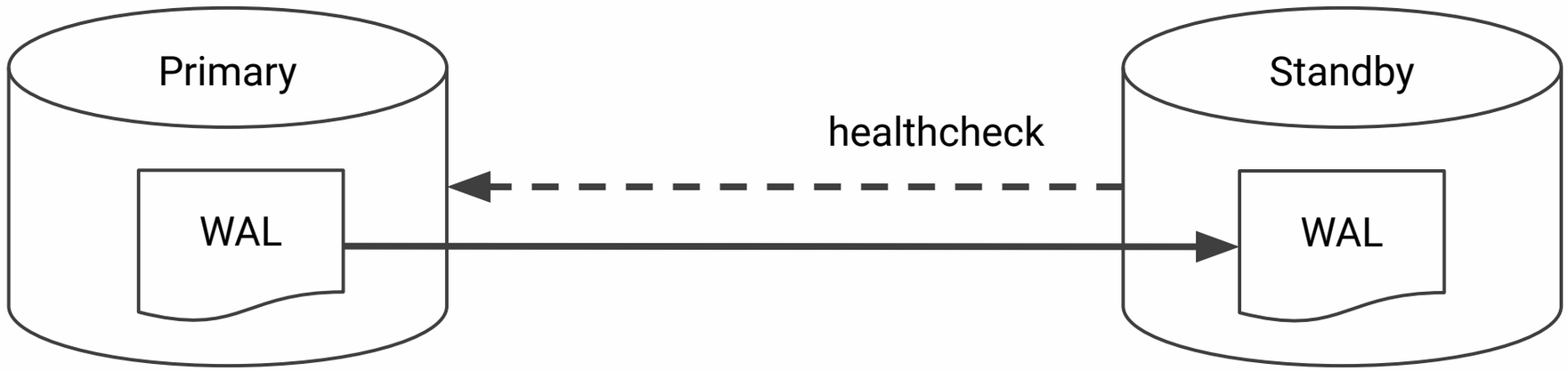
- Встроенная фича
- Минимальная задержка
- Идентичные копии

Минусы:

- Нужны одинаковый мажорные версии
- Нет автоматического failover

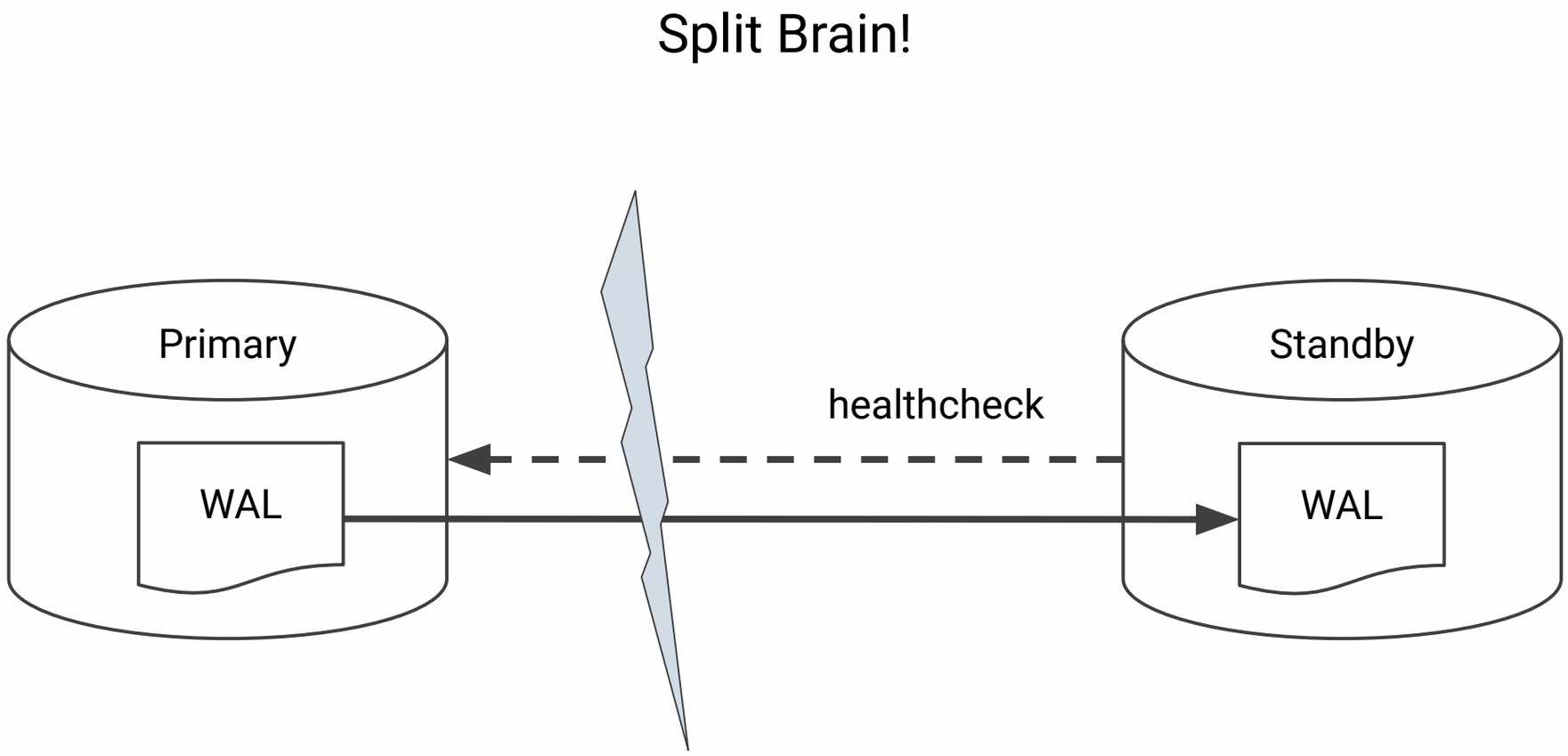


Автоматический Failover

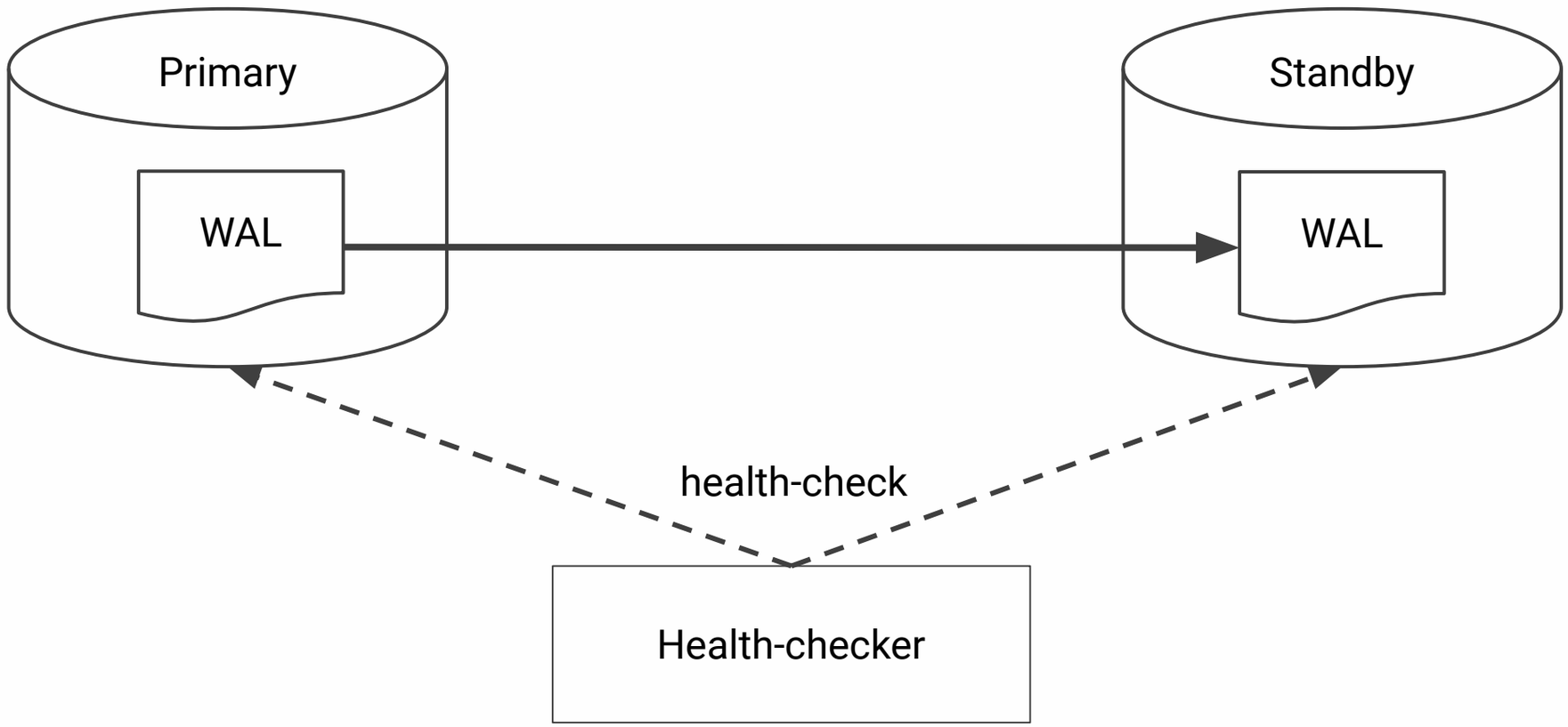


Запускаем healthcheck со стендбая и при отрицательном ответе продвигаем (promote) его до Мастера

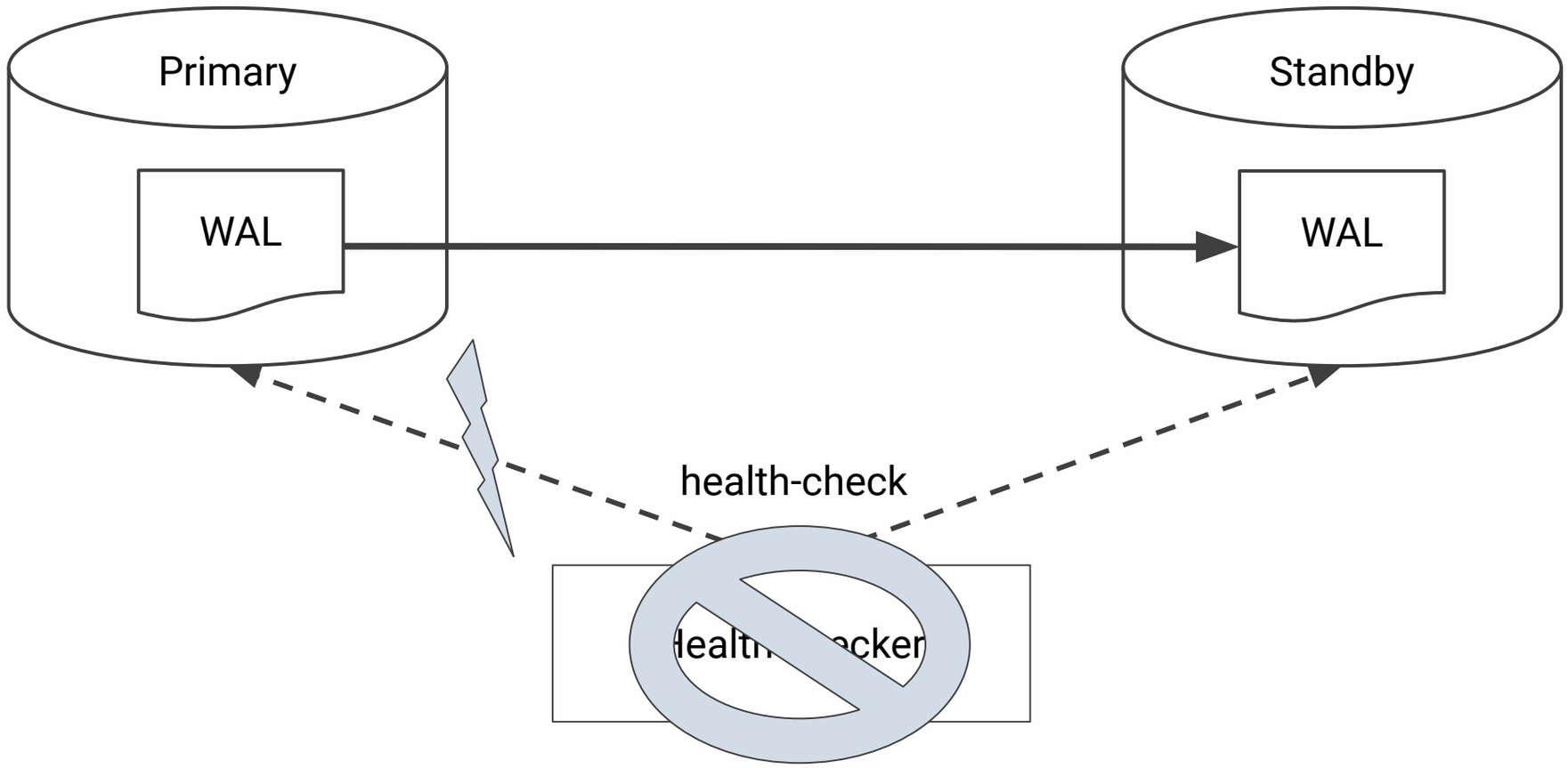
Автоматический Failover



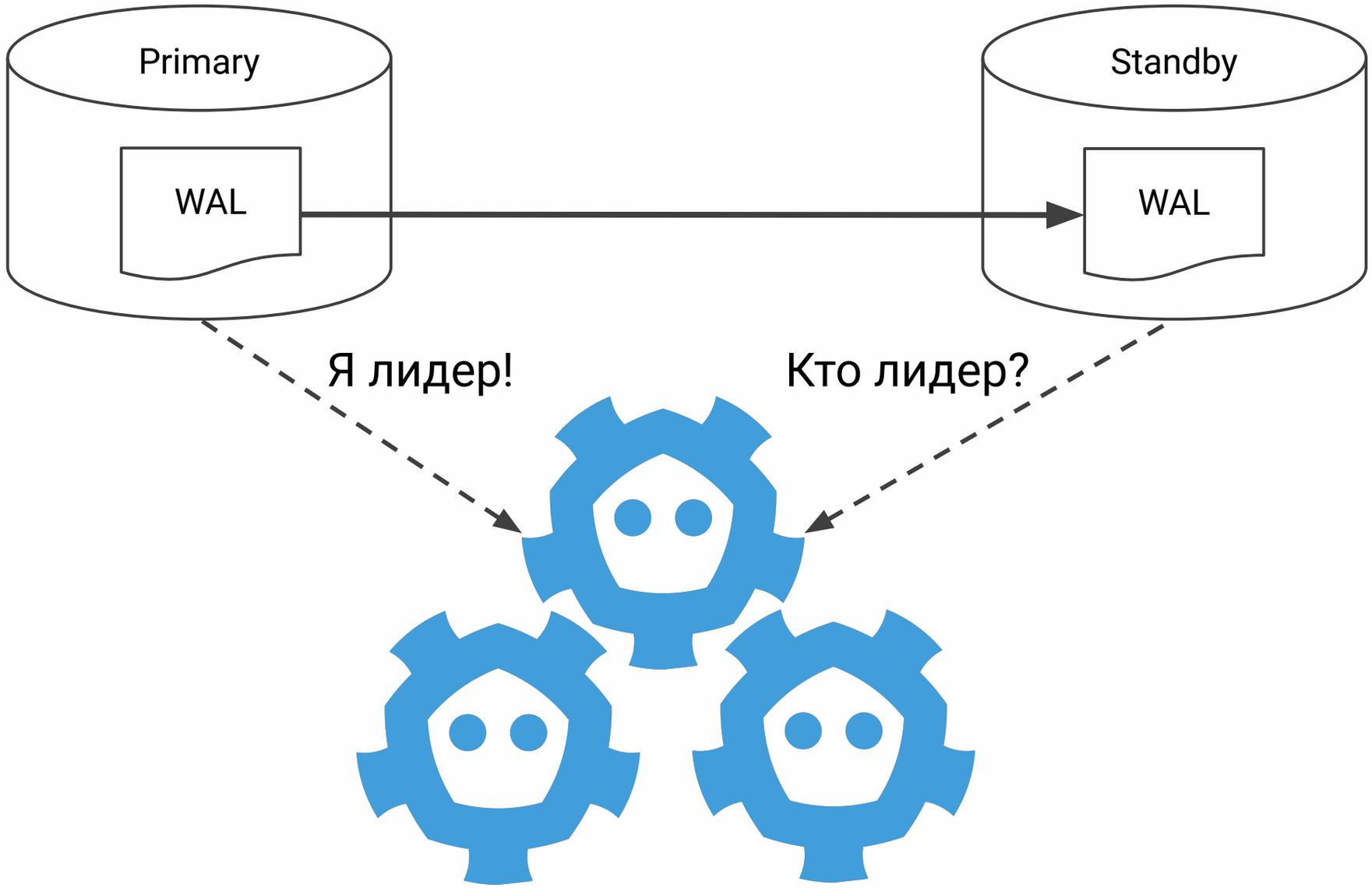
Автоматический Failover



Автоматический Failover



Автоматический Failover



Автоматический Failover

У постгреса нет
какого либо решения
по автоматическому
фейловеру из коробки

Функции DCS

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

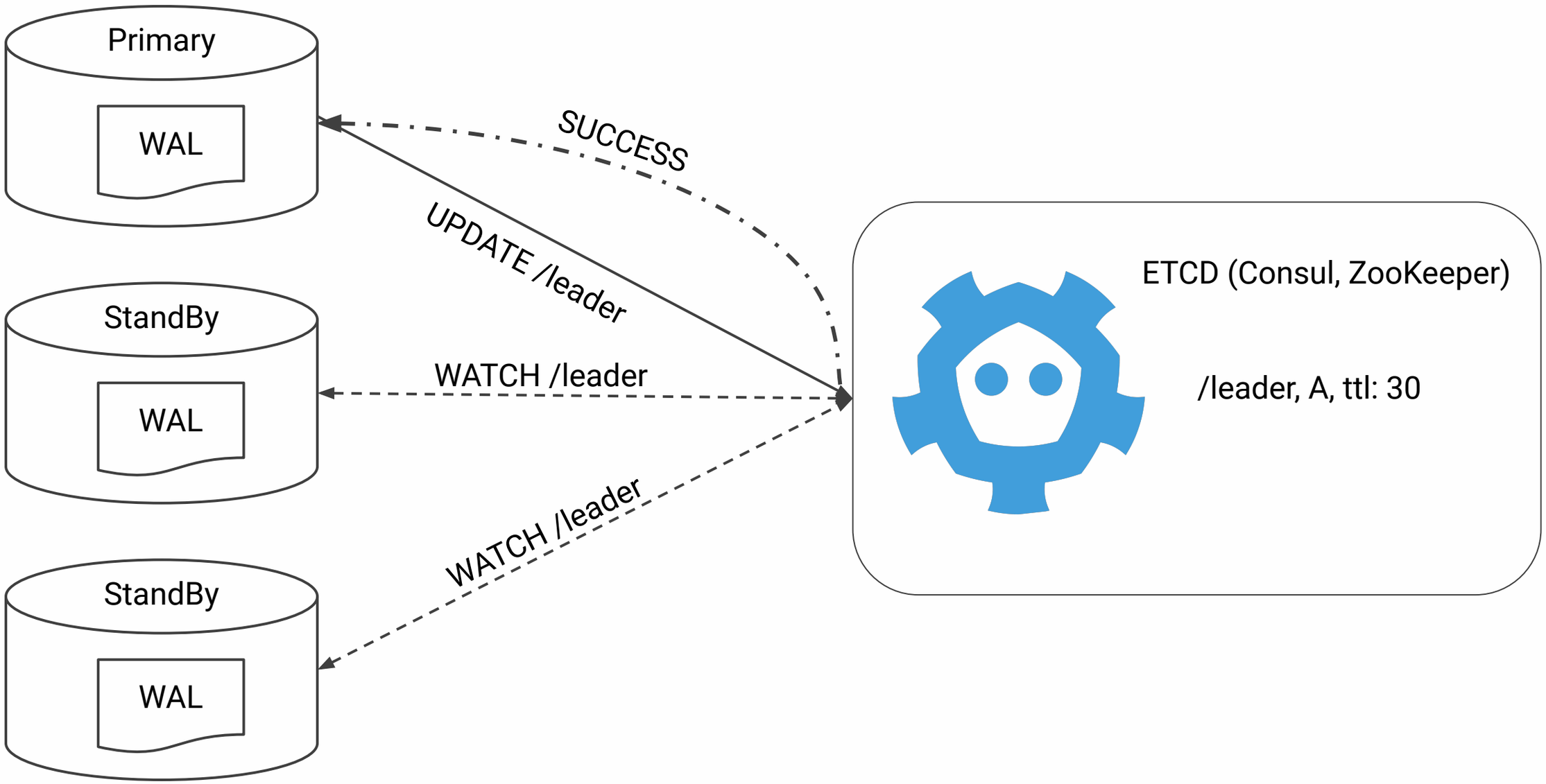
Почему Consul

- 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

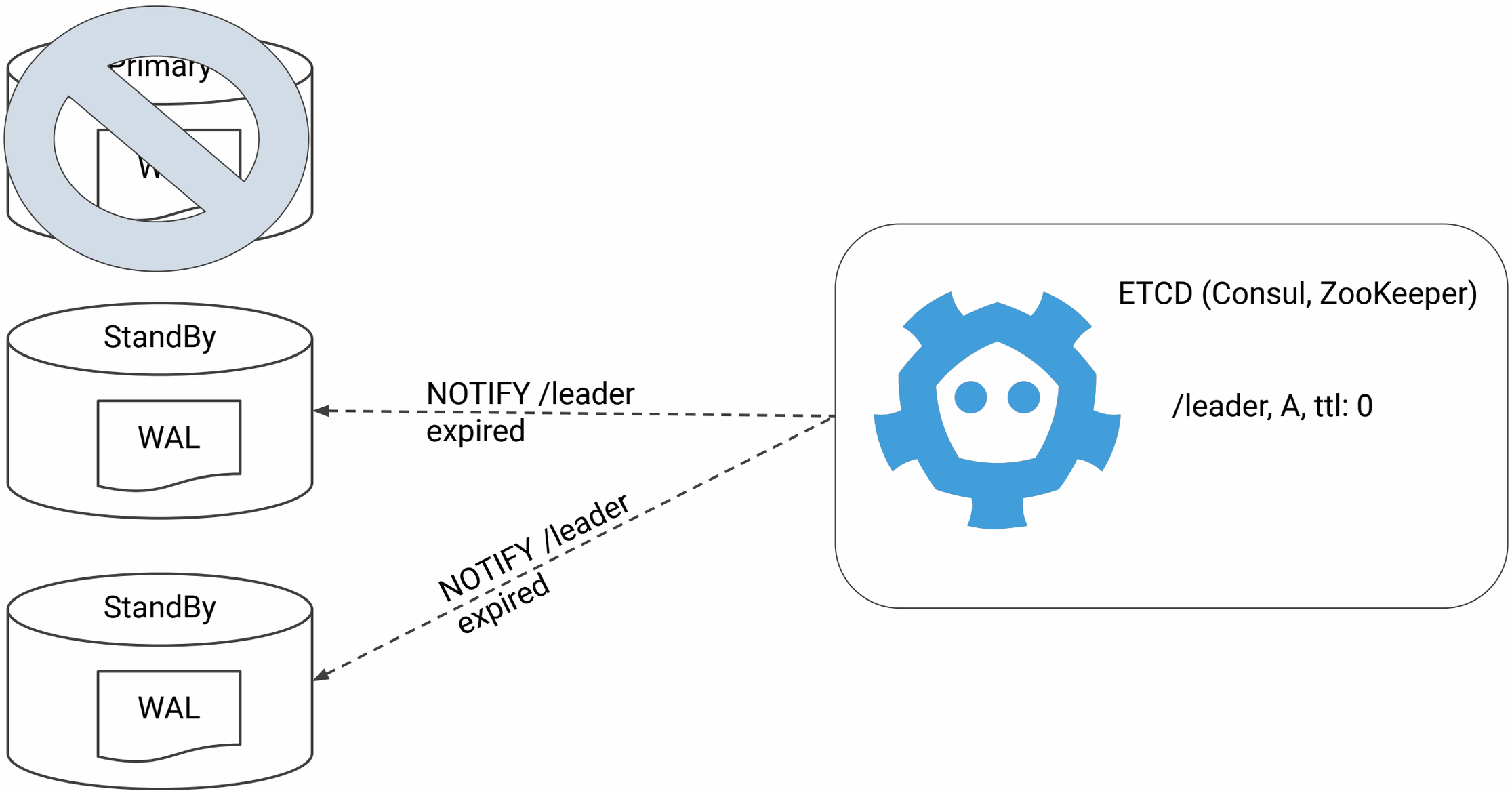
Patroni

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

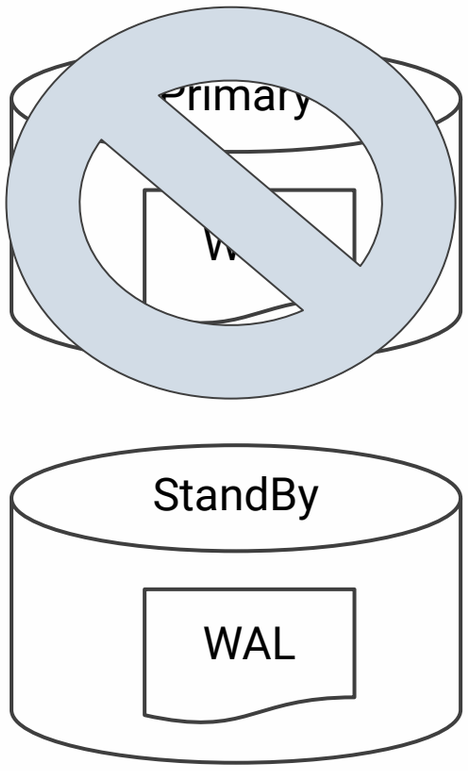
Автоматическая репликация



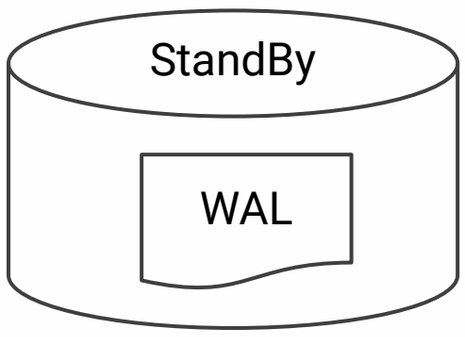
Автоматическая репликация



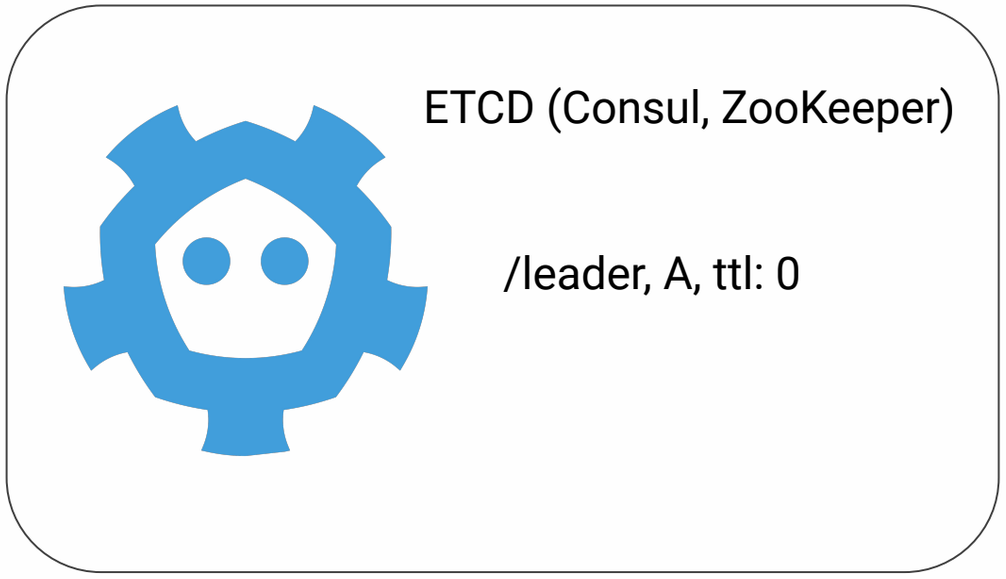
Автоматическая репликация



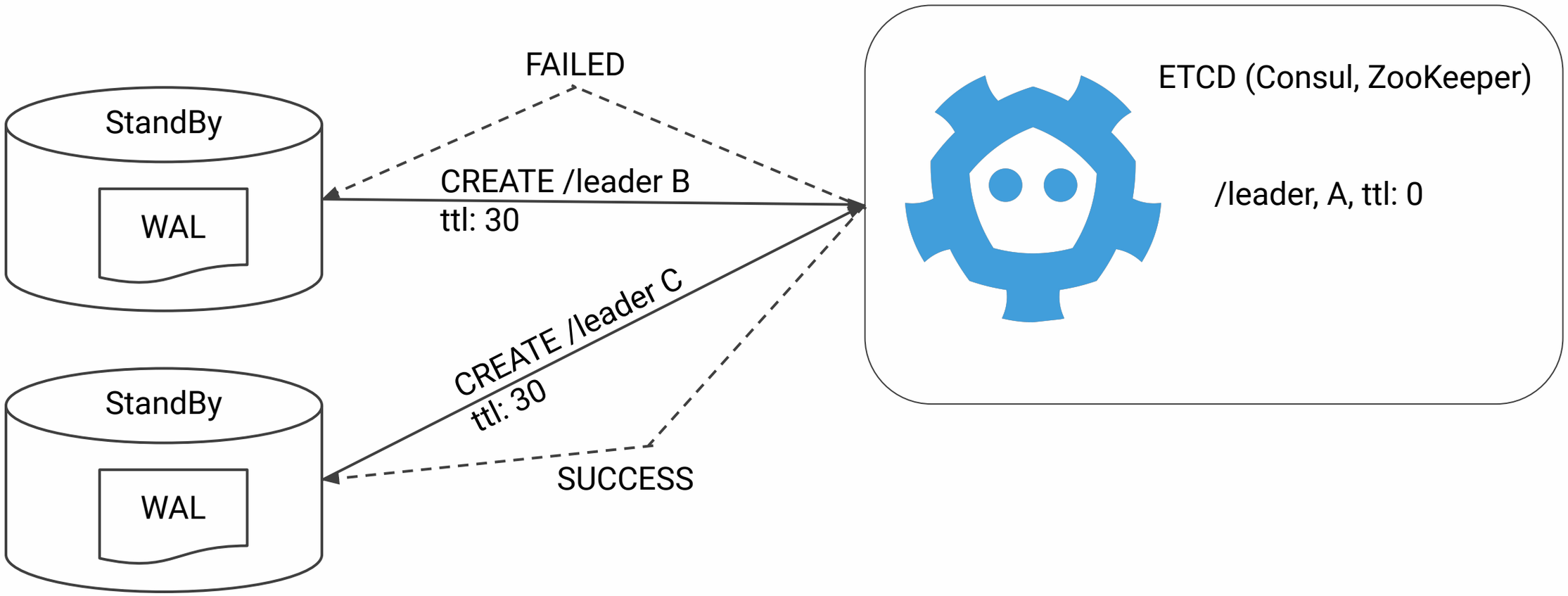
Node B:
GET hostA:patroni -> Timeout
GET hostB:patroni -> wal_position: 200
GET hostC:patroni -> wal_position: 100



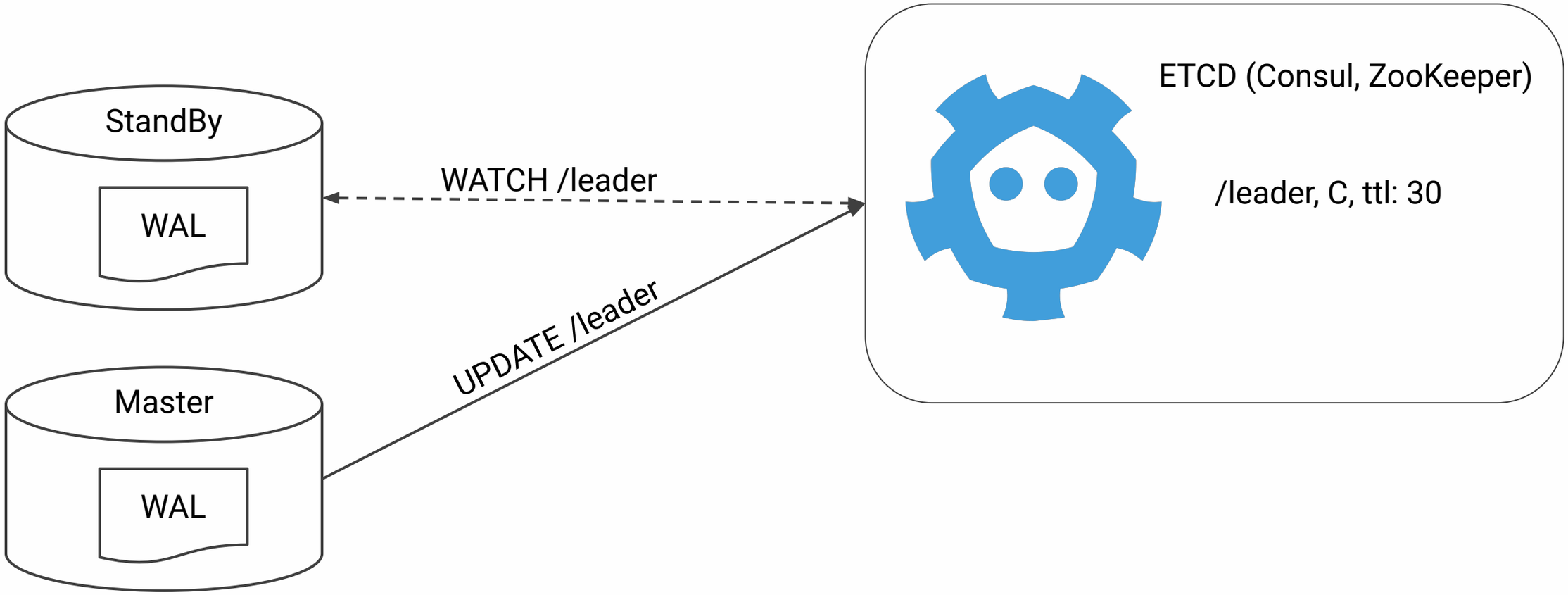
Node C:
GET hostA:patroni -> Timeout
GET hostB:patroni -> wal_position: 200
GET hostC:patroni -> wal_position: 100



Автоматическая репликация



Автоматическая репликация



Свой первый кластер Patroni

ETCD

HAProxy

pgbouncer (pgpool-II)

Нода Postgresql:

- Postgresql{9,10,11}-server
- pip install patroni и зависимости
- конфигурационный файл patroni.yml
- Дата директория - с правами для пользователя postgres

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

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 |
+----------+--------+-------------+--------+---------+----+-----------+
```

Переменные окружения

- **PATRONI_CONFIGURATION -** путь до конфигурационного файла
- **PATRONI_NAME -** имя текущей ноды. Должно быть уникально в контексте кластера
- **PATRONI_SCOPE** - имя кластера
- **PATRONI_LOG_*** - все что связано с логами

export PATRONI_CONSUL_HOST=’192.168.11.100:8500’
export PATRONI_CONSUL_TOKEN=aabbccddeeff

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

Важные параметры

Обновление данных в DCS идет циклично:

- loop_wait - минимальный промежуток в секундах между попытками обновить ключ лидера.
- ttl - время жизни ключа лидера. Рекомендация: как минимум loop_wait + retry_timeout, но вообще таким комфортным, чтобы избежать нескольких медленных/неудавшихся вызовов к DCS
- retry-timeout - общее время всех попыток внутри одной операции

- maximun_lag_on_failover - максимальное отставание ноды от лидера для того, чтобы учавствовать в выборах

- synchronous_mode: - вкл/выкл синхронной реплики
- synchronous_mode_strict: - вкл/выко строго синхронного режима

Fun Fact (NO)

Patroni делит TTL пополам, потому-что (барабанная дробь) Consul умножает его на два:

```
$ consul kv get service/pg-ha-cluster/config | jq -c '. | { ttl }'
{"ttl":30}

$ curl -s http://127.0.0.1:8500/v1/session/node/$( hostname -s ) | jq -c '.[] | { TTL }'
{"TTL":"15.0s"}
```

Редактирование конфигурации

```
[root@pg02]# patronictl -c /etc/patroni.yml edit-config
---
+++
@@ -2,5 +2,6 @@
 maximum_lag_on_failover: 1048576
 postgresql:
   use_pg_rewind: true
+  parameters:
+    maintenance_work_mem: 256MB
 retry_timeout: 10
 ttl: 30
Apply these changes? [y/N]:
```

Mar 21 09:59:50 pg03 patroni: 2019-03-21 09:59:50,666 INFO: Changed maintenance_work_mem from 65536 to 256MB

Mar 21 09:59:50 pg03 patroni: 2019-03-21 09:59:50,667 INFO: PostgreSQL configuration items changed, reloading configuration.

Редактирование конфигурации

Попробуем поменять параметр требующий перезагрузки: **max_connections**

Mar 21 10:04:10 pg03 patroni: 2019-03-21 10:04:10,665 INFO: Changed max_connections from 100 to 200 (restart required)

[root@pg02] http http://10.128.0.45:8008

Ручной Switchover:

patronictl -c /etc/patroni.yml switchover --master pg03 --candidate pg01

Локальная конфигурация

Что делать если нужно поменять конфигурацию PostgreSQL только локально.

- etcd
- 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 могу быть переопределены локально - Patroni их перезаписывает.

Monitoring

Проверка запущен ли PostgreSQL мастер:

- GET /master - должно возвращать 200 ТОЛЬКО для одной ноды

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

Направление клиентов

- HAProxy
- Pgbouncer - решит проблему с дисконнектом у клиентов
- KeepaliveD
- TCP Proxy (NGINX)

Пользовательские скрипты. ХУКИ!

```
postgresql:
  callbacks:
    on_start: /opt/pgsql/pg_start.sh
    on_stop: /opt/pgsql/pg_stop.sh
    on_role_change: /opt/pgsql/pg_role_change.sh
```

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

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, или не настроен автомат

switchover

- patronictl switchover cluster_name
- Отложенный switchover
- Смена мастера для работы с ним

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

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

Синхронная репликация

- **synchronous_mode**: true/false - не делает failover ни на какую реплику кроме синхронной
- **synchronous_mode_strict:** true/false - если синхронная реплика пропала, то мастер не принимает новые записи пока она не вернется

Бэкап кластера

Полные и инкрементные бэкапы создаются кастомными скриптами по плану (cron)/barman/wal-g/wal-e/etc

- Роль узла в кластере можно узнать запросом к DCS
- Архивные транзакционные логи (WAL):
  - сегментами в 16 Мб с мастер узла (**archive_command=on**)
  - потоком по протоколу физической репликации (**pg_receive_wal**)

Восстановление кластера из бэкапа

- Возможность восстановиться из бэкапа на любую точку по:
  - времени
  - id транзакции (xid)
  - lsn транзакционной записи в журнале
  - именной записи в журнале

bootstrap:
method: probackup
probackup:
    command: “pg_probackup restore -B /path/to/backup --instance <scope> -D <datadir> --time=’2019-09-08 00:00:00 UTC’ \ --recovery-target-action=promote”
    recovery_conf:
        recovery_target_timeline: latest
        restore_command: pg_probackup-11 archive-get -B /var/backup --instance <scope> --remote-user=dbbackup --wal-file-path %p --wal-file-name %f --remote-host=AA.BB.CC.DD

Восстановление кластера из бэкапа

: pg_probackup archive-get from /var/backup/wal/db-mt/0000001300000000000000076 to /var/data/base/pg_wal/RECOVERYXLOG
ERROR: Source WAL file "/var/backup/wal/db-mt/0000001300000000000000076" doesn't exist
2019-08-28 10:06:57.782 UTC [23] LOG: redo done at 0/75000198
INFO: pg_probackup archive-get from /var/backup/wal/db-mt/0000001300000000000000075 to /var/data/base/pg_wal/RECOVERYXLOG
INFO: pg_probackup archive-get completed successfully
2019-08-28 10:07:01.015 UTC [23] LOG: restored log file "0000001300000000000000075" from archive
INFO: pg_probackup archive-get from /var/backup/wal/db-mt/00000014.history to /var/data/base/pg_wal/RECOVERYHISTORY
ERROR: Source WAL file "/var/backup/wal/db-mt/00000014.history" doesn't exist
2019-08-28 10:07:01.639 UTC [23] LOG: selected new timeline ID: 20
2019-08-28 10:07:01.677 UTC [23] LOG: archive recovery complete
INFO: pg_probackup archive-get from /var/backup/wal/db-mt/00000013.history to /var/data/base/pg_wal/RECOVERYHISTORY
INFO: pg_probackup archive-get completed successfully
2019-08-28 10:07:02.280 UTC [23] LOG: restored log file "00000013.history" from archive
2019-08-28 10:07:02.389 UTC [21] LOG: database system is ready to accept connections

Создание реплики из бекапа

- По умолчанию реплика создается с помощью утилиты pg_basebackup
- Это поведение можно переопределить параметром create_replica_methods
- Важно, обязательно нужно указать basebackup, иначе если из бекапа не получится, то реплика не заведется.

```
postgresql:
  create_replica_methods:
    - probackup
    - basebackup
  probackup:
    command: "ssh dbbackup@10.23.2.163 ‘bash /var/backup/pg_restore.sh"'
    no_params: True
  basebackup:
    max-rate: '100M'
```

Создание реплики из бекапа

2019-08-20 14:17:51,986 INFO: Removing data directory: /var/data/base
INFO: Validating backup PWJ0PZ
INFO: Backup PWJ0PZ data files are valid
INFO: Backup PWJ0PZ WAL segments are valid
INFO: Backup PWJ0PZ is valid.
INFO: Restore of backup PWJ0PZ completed.
2019-08-20 14:17:56,150 INFO: replica has been created using probackup
2019-08-20 14:17:56,153 INFO: bootstrapped from leader 'AA.BB.CC.DD'

Валидация бекапа

- Docker образ для минимального запуска
- Скрипт с восстановлением из бекапа
    - Минимальный конфиг для старта
    - pg_hba.conf с trust доступами (для упрощения)
- docker exec pgvalid pg_dump -h localhost -U postgres > /dev/null
- Amcheck
    - CREATE EXTENSION amcheck;
    - pg_probackup checkdb --amcheck --heapallindexed ...

Отстающая реплика

Custom recovery.conf:

- recovery_min_apply_delay = '12h'

Tags:

- nosync: True
- nofailover: True
- nobalance: True

Практическое задание

- Развернуть кластер PostgreSQL из трех нод. Создать тестовую базу - проверить статус репликации
- Сделать switchover/failover
- Поменять конфигурацию PostgreSQL + с параметром требующим перезагрузки

Ваши вопросы?