PostgreSQL Cloud Solutions

otus.ru

Проверить, идет ли запись

Меня хорошо видно && слышно?

Тема вебинара

Работа с кластером высокой доступности для PostgreSQL

Алексей Железной

Data Engineer

- 4 года опыта Инженером данных/аналитиком (Python, Airflow, Clickhouse, Greenplum)
- 2 года опыта преподавания в OTUS (курсы DWH Analyst/DE/PostgreSQL Cloud)

LinkedIn

Правила вебинара

Активно
участвуем

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения

Индивидуально

Время, необходимое
на активность

Пишем в чат

Говорим голосом

Документ

Ответьте себе или
задайте вопрос

Маршрут вебинара

- Postgres Operator
- Helm чарт Patroni
- Patroni on-premise
- Практика
- Рефлексия

Цели вебинара

К концу занятия вы

1. Научитесь настраивать кластер Patroni через чарт helm
2. Изучите Postgres Operator
3. Узнаете, как настраивать классический кластер Patroni

Смысл

Зачем вам это уметь

1. Выбрать оптимальный вариант высокой доступности для PostgreSQL
2. Уметь настроить высокодоступный кластер PostgreSQL своими руками

Повторение, практика

Высокая доступность

High Availability, или HA - показатель устойчивости системы к сбоям инфраструктуры:

- гарантирует отсутствие длительного эффекта от сбоя сервера или системы, позволяет контролировать и поддерживать работоспособность внутренних серверов
- измеряется в 9-ках
- все хотят 99,999 - 5 минут простоя в год
- все предлагают 99,99 - час простоя в год
- метод измерения и критерий доступности - тот еще вопрос

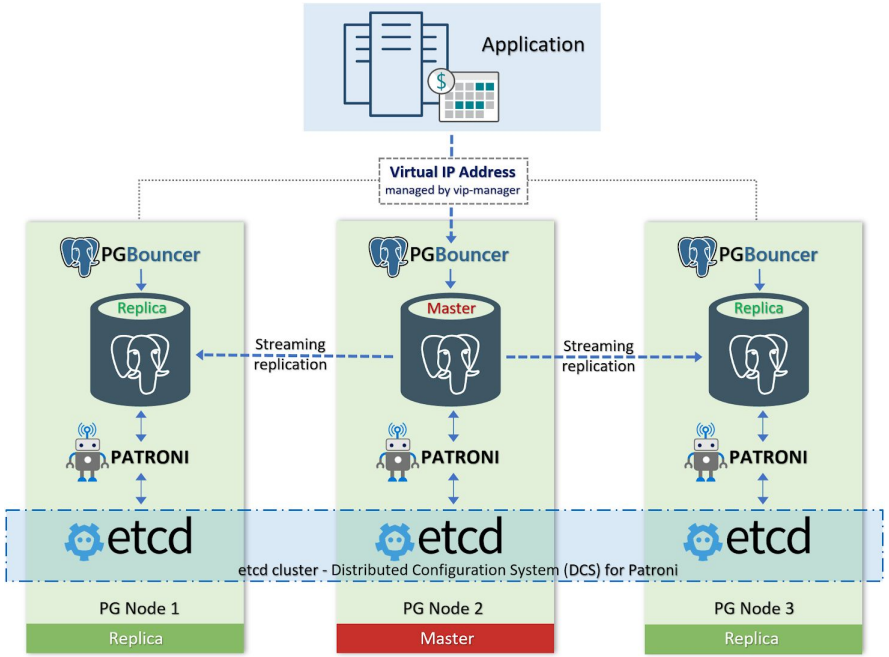
[Калькулятор SLA: 99.9% аптайм]

HA кластеры PostgreSQL

- Классические
  - Pgpool II
  - pg_bouncer + haproxy
  - pg-auto-failover
  - repmgr
- k8s
- Cloud native
  - patroni
  - stolon
  - slony
  - ClusterControl

Patroni

Patroni - это шаблон для создания собственного специализированного решения высокой доступности с использованием Python и - для максимальной доступности - распределенного хранилища конфигурации, такого как ZooKeeper, etcd, Consul или Kubernetes.



Зачем нужен Patroni

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

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

Создать свой первый кластер Patroni:

`patroni /etc/patroni.yml` OR `systemctl start patroni.service`

Состояние кластера через `patronictl` - утилита для управления кластером

`patronictl -c /etc/patroni.yml list`

Bitnami

Установка PostgreSQL с архитектурой HA в кластере Kubernetes

- Helm chart на основе схемы bitnami/postgresql, включает некоторые изменения для обеспечения высокой доступности, такие как:
  - Добавлено новое развертывание, сервис был добавлен для развертывания Pgpool-II, чтобы действовать как прокси для PostgreSQL бэкенда.
  - Замена bitnami/postgresql на bitnami/postgresql-repmgr, который включает и настраивает repmgr.

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Postgres Operator

Что же такое RDS?

Relational Database Service - по опыту, управляемый (managed) сервис СУБД, которая:

- легко настраивается
- имеет возможность работы со снапшотами и восстанавливаться из них
- позволяет создавать топологии master-slave
- имеет богатый список расширений
- предоставляет аудит и управление пользователями/доступами.

Именно операторы — общепринятый подход для решения подобных задач в экосистеме Kubernetes.

Выбор оператора

Ожидания:

- деплой из Git и с Custom Resources *(CRD нужен, поскольку все больше пользователей хотят вносить данные в Kubernetes. Формат данных может быть разным, а CRD не определены в Kubernetes «из коробки». В итоге требуется создавать собственные таблицы, устанавливая имена и типы столбцов, как в базе данных)*.
- поддержку pod anti-affinity
- установку node affinity или node selector
- установку tolerations
- наличие возможностей тюнинга
- понятные технологии и даже команды.

Postgres Operator

Управляет кластерами PostgreSQL на Kubernetes (K8s):

- Следит за добавлением, обновлением и удалением манифестов кластеров PostgreSQL и соответствующим образом изменяет запущенные кластеры.
- Следит за обновлениями собственной конфигурации и при необходимости изменяет запущенные кластеры Postgres.
- Периодически синхронизирует фактическое состояние каждого кластера Postgres с желаемым состоянием, определенным в манифесте кластера.

Postgres Operator

- Helm чарт
- Кубер в GKE, yandex cloud
- На базе патрони

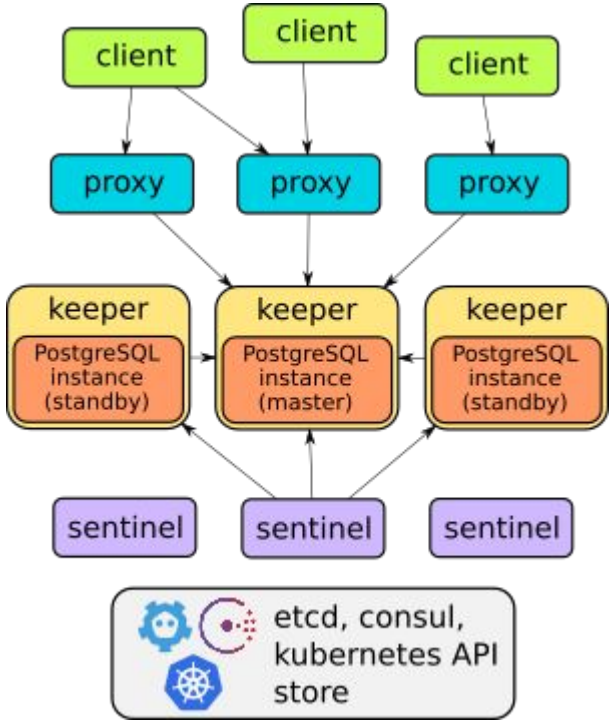
Для PostgreSQL существует несколько популярных K8s-операторов:

- Stolon
- Crunchy Data PostgreSQL Operator
- Zalando Postgres Operator

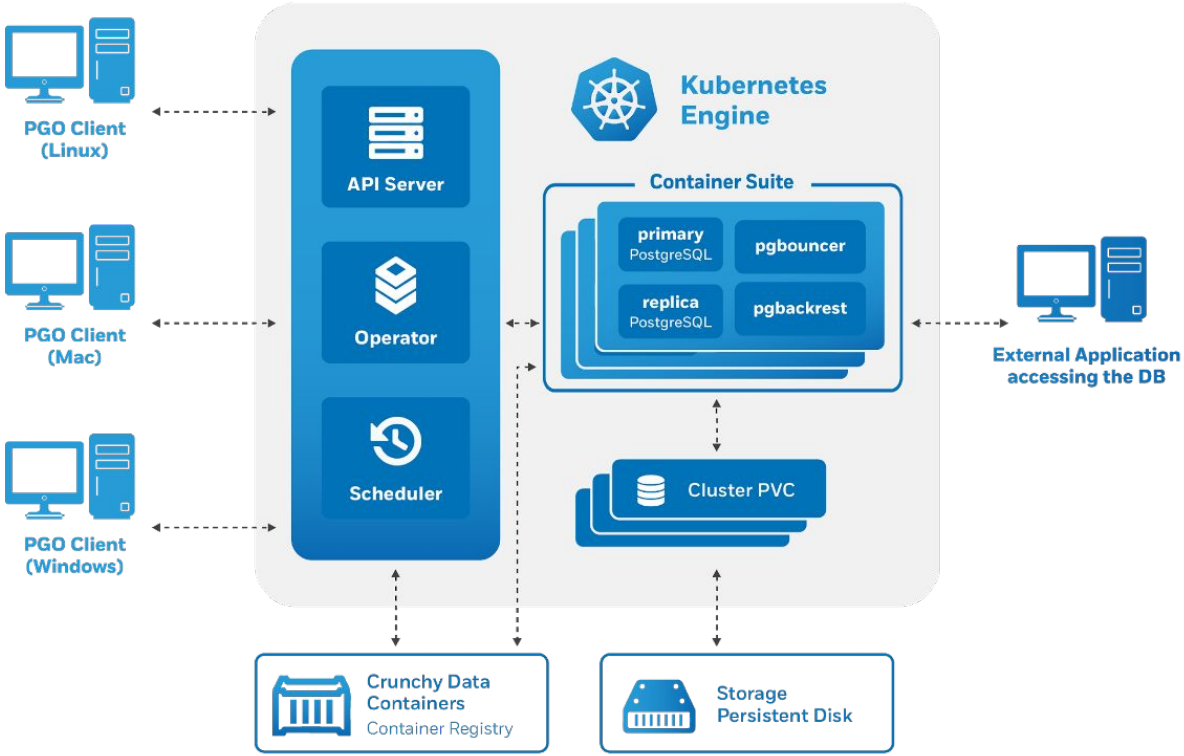
Stolon

Умеет всё (почти) описанное: failover, прокси для прозрачного доступа клиентов (через один сервис endpoint), бэкапы.

Нет Custom Resources (по сути, не совсем себе и оператор) - требуется заводить чарт под каждую БД



Crunchy Data PostgreSQL Operator



Crunchy Data PostgreSQL Operator

Плюсы:

- есть управление через CRD
- удобное управление пользователями (тоже через CRD)
- интеграция с другими компонентами Crunchy Data Container Suite — специализированной коллекции образов контейнеров для PostgreSQL и утилит для работы с ней

Проблемы:

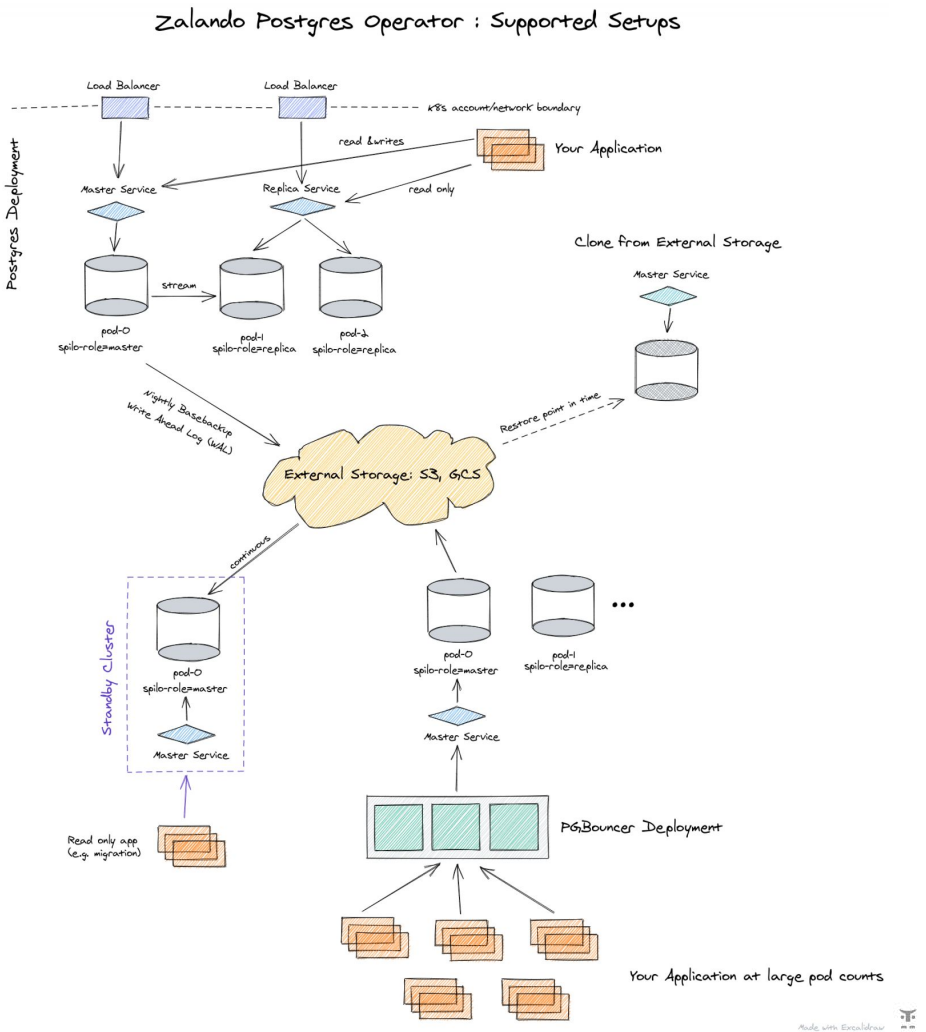
- Не оказалось возможности tolerations — предусмотрен только nodeSelector.
- Создаваемые pod’ы были частью Deployment’a. В отличие от StatefulSet, Deployment’ы не умеют создавать диски.

Zalando Postgres Operator

«Под капотом» этого оператора используются решения, проверенные временем:

- Patroni и Spilo для управления,
- WAL-E — для бэкапов,
- PgBouncer — в качестве пула подключений.

Оператор полностью управляется через Custom Resources, автоматически создает StatefulSet из контейнеров, которые затем можно кастомизировать, добавляя в pod различные sidecar'ы.



Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Patroni

Patroni

Helm чарт

кубер в GKE, yandex cloud

Полезные ссылки:

- https://patroni.readthedocs.io/en/latest/
- https://github.com/helm/charts/tree/master/incubator/patroni - архив
- https://ruzickap.github.io/k8s-postgresql/part-02/

Patroni on-premise

Полуручное раскатывание https://github.com/zalando/patroni

Ручное раскатывание

Построение кластера PostgreSQL высокой доступности с использованием Patroni, etcd, HAProxy

Заряжай Patroni. Тестируем Patroni + Zookeeper кластер (Часть первая) / Блог компании VS Robotics / Хабр

Заряжай Patroni. Тестируем Patroni + Zookeeper кластер (Часть вторая) / Блог компании VS Robotics / Хабр

Практика

ДЗ

Домашнее задание

- Вариант 1 How to Deploy PostgreSQL for High Availability
- Вариант 2 Introducing pg_auto_failover: Open source extension for automated failover and high-availability in PostgreSQL
- Для гурманов Настройка Active/Passive PostgreSQL Cluster с использованием Pacemaker, Corosync, и DRBD (CentOS 5,5)
- Задачка со звездочкой:
  - Создать два кластера GKE в разных регионах
  - Установить на первом Patroni HA кластер
  - Установить на втором Patroni Standby кластер
  - Настроить TCP LB между регионами
  - Сделать в каждом регионе по клиентской ВМ
  - Проверить как ходит трафик с клиентской ВМ
  - Описать что и как делали и с какими проблемами столкнулись

Список материалов для изучения

1. How to Achieve PostgreSQL High Availability with pgBouncer
2. PgBouncer config
3. Crunchy Postgres Operator v5
4. Обзор операторов PostgreSQL для Kubernetes. Часть 1: наш выбор и опыт

Рефлексия

Цели вебинара

К концу занятия вы

1. Научитесь настраивать кластер Patroni через чарт helm
2. Изучите Postgres Operator
3. Узнаете, как настраивать классический кластер Patroni

Рефлексия

С какими впечатлениями уходите с вебинара?

Как будете применять на практике то,
что узнали на вебинаре?

Заполните, пожалуйста, опрос о занятии по ссылке в чате

Спасибо за внимание!

Приходите на следующие вебинары

Алексей Железной

Data Engineer

- 4 года опыта Инженером данных/аналитиком (Python, Airflow, Clickhouse, Greenplum)
- 2 года опыта преподавания в OTUS (курсы DWH Analyst/DE/PostgreSQL Cloud)

LinkedIn