Дипломный проект
«Сравнение производительности двух PostgreSQL кластеров на большом объеме данных»

Евгений Жемеркин

Цели проекта

1. Создать отказоустойчивый кластер PostgreSQL на основе Patroni
2. Создать отказоустойчивый кластер PostgreSQL на основе pg_auto_failover
3. Провести сравнительное тестирование производительности кластеров

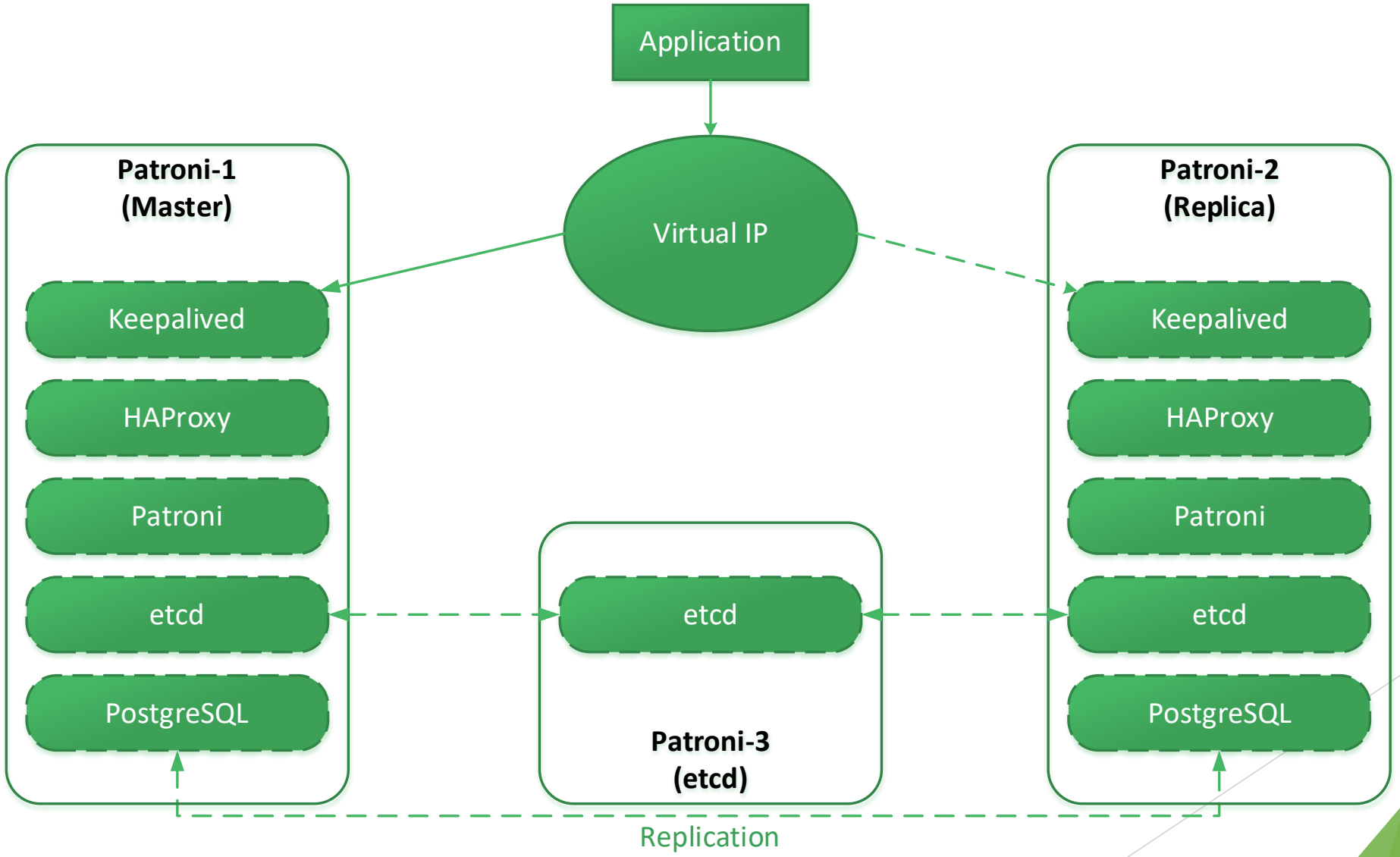
Используемые технологии

- PostgreSQL
- Patroni
- Pg_auto_failover
- HAProxy
- Keepalived
- ETCD

Кластер Patroni

Для начала развернём кластер Patroni на 3-х нодах:
patroni-1 - PostgreSQL Master, etcd
patroni-2 - PostgreSQL Replica, etcd
patroni-3 - etcd

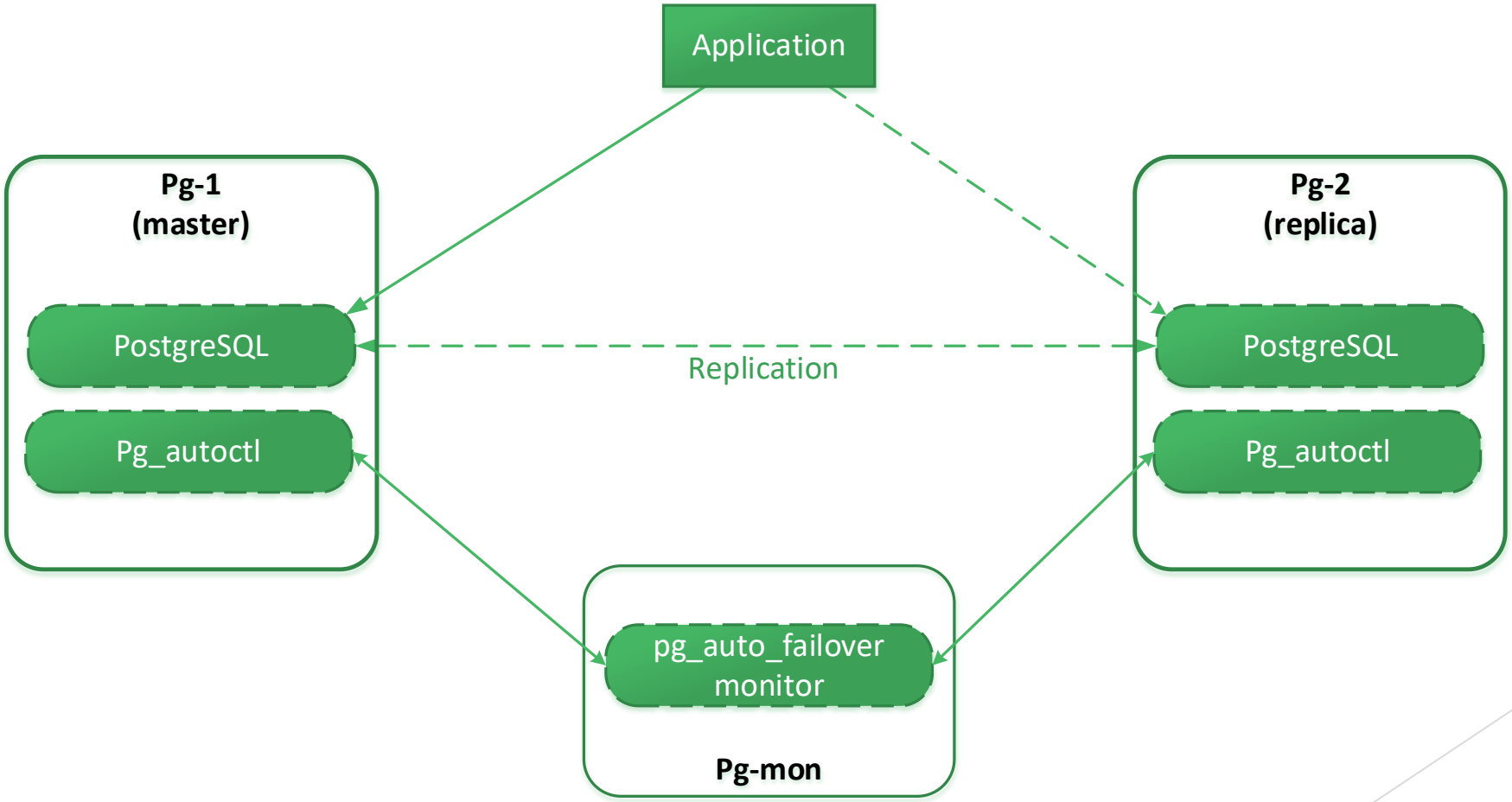
Кластер Patroni



Кластер Pg_auto_failover

Развернём кластер pg_auto_failover на 3-х нодах:
pg-1 - PostgreSQL Master,
pg-2 - PostgreSQL Replica
pg-mon - monitor

Кластер Pg_auto_failover



Преимущества Patroni

- Высокая отказоустойчивость, поскольку кластер состоит из множества компонентов, которые можно расположить на отдельных серверах
- Наличие множества настраиваемых параметров у используемых в кластере инструментов, что даёт возможность гибкой настройки
- Возможность использовать etcd для хранения конфигурации кластера

Недостатки Patroni

- Относительная сложность развёртывания и настройки
- Большое количество компонентов требует дополнительных ресурсов

Преимущества Pg_auto_failover

- Простота настройки и развёртывания
- Малое количество компонентов, что позволяет не задействовать дополнительные ресурсы

Недостатки Pg_auto_failover

- Малое количество компонентов не позволяет задействовать для них большее количество независимых серверов, что может снизить отказоустойчивость
- По умолчанию не предусматривается использование балансировщиков и Virtual IP

Тестирование производительности кластеров

Для тестирования производительности использовался датасет чикагского такси объёмом 10 Гб.

Тестирование производилось сложным SQL-запросом:

```
SELECT payment_type,
    round(sum(tips)/sum(trip_total)*100, 0) + 0 as tips_percent,
    count(*) as c
FROM taxi_trips
group by payment_type
order by 3;
```

Результаты тестирования

В ходе тестирования оба кластера показали практически одинаковое время выполнения запроса:

- Время выполнения запроса для кластера Patroni составило 6 минут 35 секунд.
- Время выполнения запроса для кластера Pg_auto_failover составило 5 минут 55 секунд.

Заключение

Время выполнения одиночного сложного запроса примерно одинаково для кластеров Patroni и pg_auto_failover.

Оба кластера были развёрнуты на серверах с одинаковыми характеристиками. Кластер Patroni развёрнут без балансировки нагрузки запросов на чтение между серверами. В кластере pg_auto_failover балансировка нагрузки стандартными средствами не поддерживается.

Кластер на основе pg_autofailover более прост в настройке и обслуживании, однако имеет меньше гибких настроек. При необходимости повышения производительности кластера понадобится использовать дополнительные решения для балансировки нагрузки.

Для приложений, осуществляющих множество запросов к базе одновременно, кластер Patroni более предпочтителен, поскольку включает возможность балансировки нагрузки.

Кластер на основе pg_auto_failover лучше подойдёт для более простых решений по кластеризации.