Yandex Cloud

Как мы делаем облачный Greenplum®

Леонид Борчук,
Team Lead Greenplum, Yandex Cloud

О чём сегодня поговорим

1. Клаудизация GP
2. Доступность
3. Backup/Restore
4. Expand

Клаудизация GP

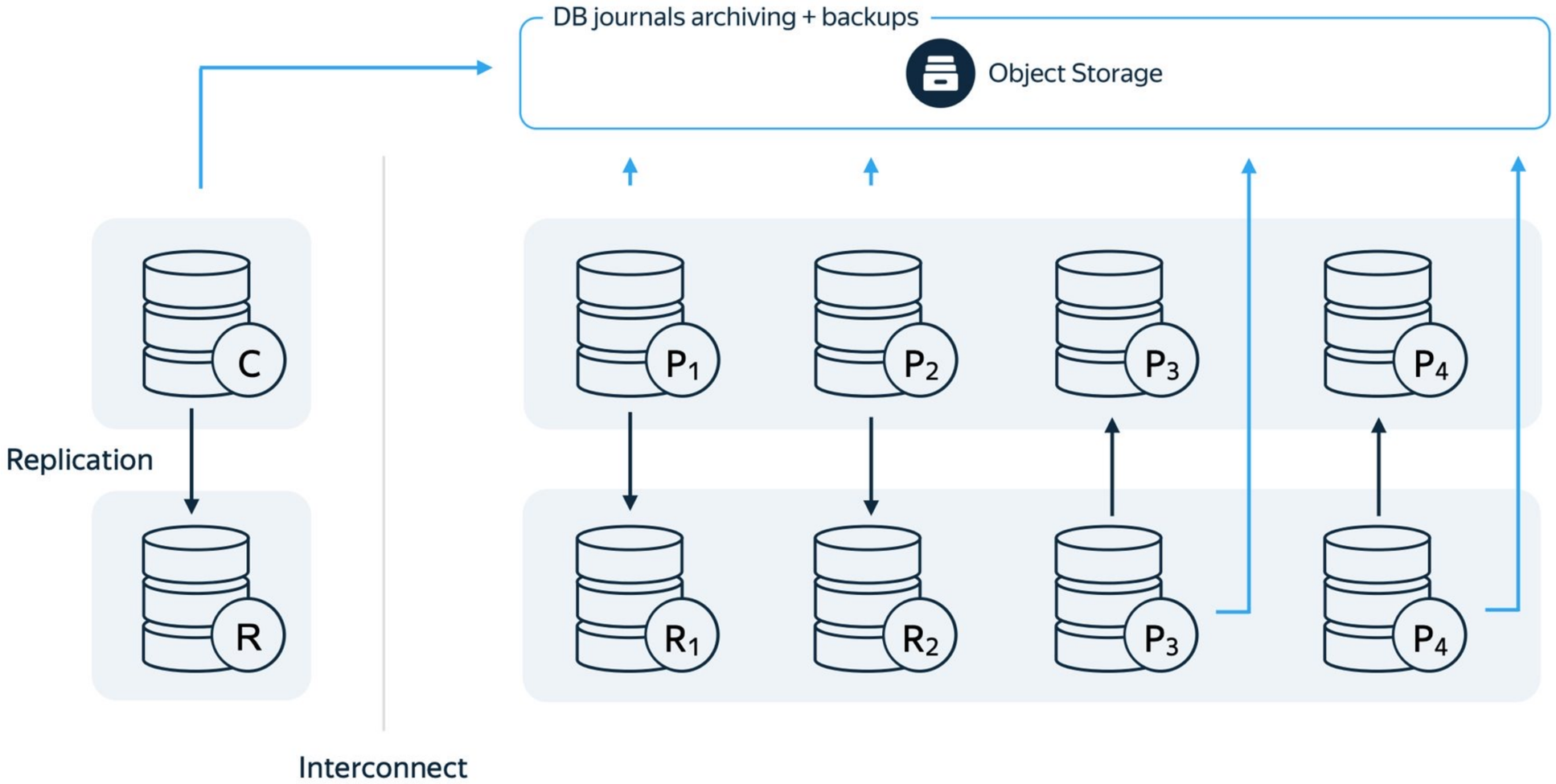
2021

>2 ПБ

> 100

150 ТБ

Кратко о Greenplum



Клаудизация GP

1

Создание
и развёртывание БД

2

Реконфигурация БД
по требованию

3

Мониторинг
потребляемых
ресурсов

4

Создание резервных
копий баз данных

5

Слежение
за доступностью
кластера и сегментов

6

Установка
обновлений ПО

Клаудизация GP

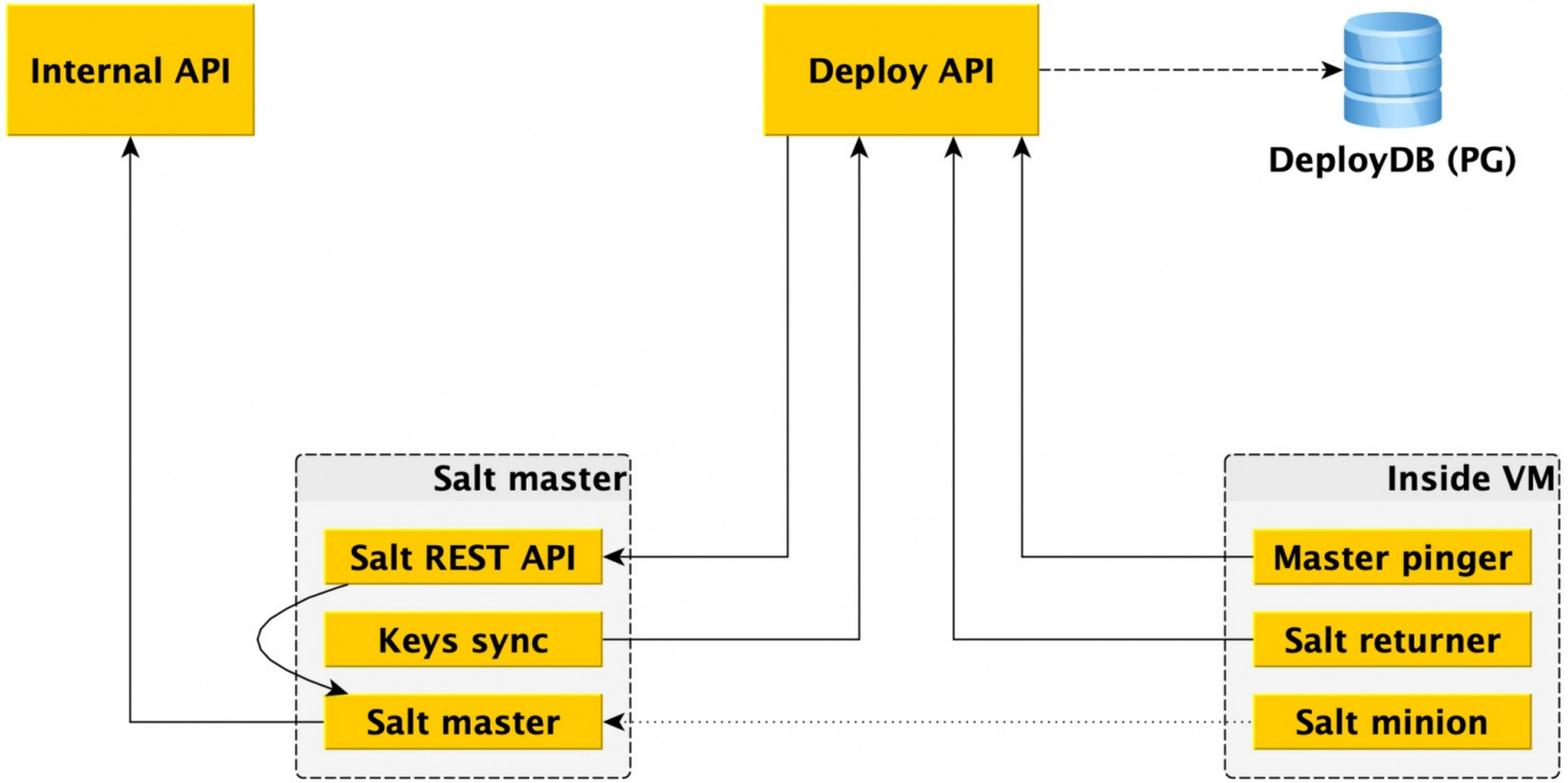
Задачи администрирования инфраструктуры всё равно должен кто-то решать

С ростом количества нод БД и количества разных инсталляций трудоёмкость возрастает

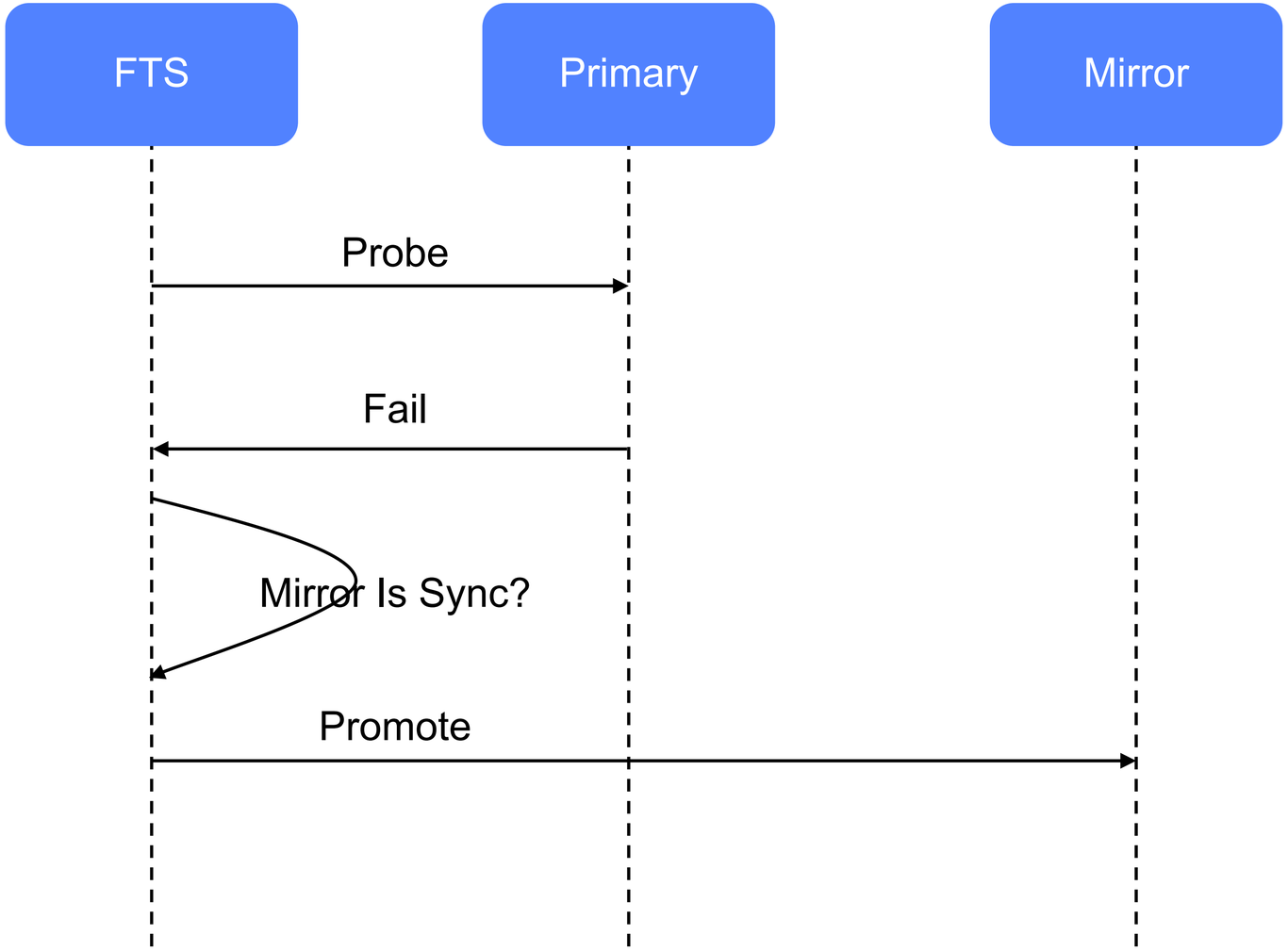
Всё больше нужны автоматизация и мониторинги

1. Клаудизация GP
2. Доступность
3. Backup/Restore
4. Expand

Автоматизируйте



Доступность. Fault Tolerant Service (FTS)



gprecoverseg

```
PATH=/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin
MAILTO=''
MASTER_DATA_DIRECTORY=/var/lib/greenplum/data1/master/gpseg-1

# m h dom mon dow user command
# */N means every N's minute of hour

*/5 * * * * gpadmin flock -n /tmp/gp_autorecovery.lock
/opt/greenplum-db-6/bin/gprecoverseg -a -r -F
```

Доступность. Параметры FTS

gp_fts_probe_interval – how often, in seconds. Default: 60

gp_fts_probe_timeout – probe timeout between master and segment, in seconds. Default: 20

gp_fts_probe_retries – the number of attempts to probe a segment. Default: 5

gp_segment_connect_timeout – the maximum time (in seconds) allowed for a mirror to respond. Default: 600 (10 minutes)

FTS. Причины

1. Проблемы с железом — хост реально недоступен
dmesg

MCE (Machine-check exception)

```
CPU 0: Machine
Check Exception: 0000000000000004
Bank 2: f200200000000863
Kernel panic: CPU context corrupt
```

EDAC (Error detection and correction)

```
CPU 0: Machine
Check Exception: 0000000000000004
Bank 2: f200200000000863
Kernel panic: CPU context corrupt
```

FTS. Причины

1. Проблемы с ресурсами:

На хосте перегружена сеть	Не хватает CPU	Перегружены диски
01	02	03

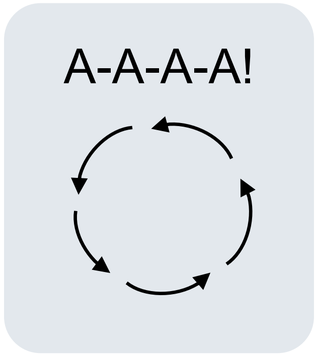
FTS. Причины

```
#0 raise (sig=sig@entry=11) at ../sysdeps/unix/sysv/linux/raise.c:51

#1 0x00005593f8b73d83 in StandardHandlerForSigillSigsegvSigbus_OnMainThread (processName=<optimized out>, postgres_signal_arg=11) at elog.c:5640

#2 <signal handler called>

#3 pg_detoast_datum_packed (datum=0x0) at fmgr.c:2241

#4 0x00005593f8b385e9 in text_right (fcinfo=0x5593fb5e25f0) at varlena.c:4094

#5 0x00005593f88d499e in ExecMakeFunctionResultNoSets (fcache=0x5593fb5e2580, econtext=0x5593fb5e0258, isNull=0x5593fb5e2071 "",
isDone=<optimized out>) at execQual.c:2219

#6 0x00005593f88d493d in ExecMakeFunctionResultNoSets (fcache=0x5593fb5e1cc0, econtext=0x5593fb5e0258, isNull=0x7ffcfe76bd57 "",
isDone=<optimized out>) at execQual.c:2193

#7 0x00005593f88dc01f in ExecQual (qual=qual@entry=0x5593fb5e1350, econtext=econtext@entry=0x5593fb5e0258, resultForNull=resultForNull@entry=0
'\000') at execQual.c:6297

#8 0x00005593f88dcb53 in ExecScan (node=0x5593fb5e0588, accessMtd=0x5593f88f9ea0 <SeqNext>, recheckMtd=0x5593f88f9e90 <SeqRecheck>) at
execScan.c:182

#9 0x00005593f88f9f48 in ExecSeqScan (node=<optimized out>) at nodeSeqscan.c:130

#10 0x00005593f88d333d in ExecProcNode (node=0x5593fb5e0588) at execProcnode.c:1029

#11 0x00005593f88e46c4 in ExecAppend (node=node@entry=0x5593fb5a22a8) at nodeAppend.c:211

#12 0x00005593f88d337d in ExecProcNode (node=node@entry=0x5593fb5a22a8) at execProcnode.c:1006

#13 0x00005593f88e6c90 in agg_retrieve_direct (aggstate=0x5593fb5a1428) at nodeAgg.c:1350

#14 ExecAgg (node=node@entry=0x5593fb5a1428) at nodeAgg.c:1237
```

FTS. Причины

```
2023-11-01 14:29:44.830734 MSK,,,p51984,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","could not fork new process for connection: Cannot allocate
memory",,,,,,,0,,"postmaster.c",45 34,

2023-11-01 14:29:44.919243 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 23877261864, 2424894224, 23877935904, 674040, . ",,,,,,,0,,,,

2023-11-01 14:29:44.919284 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 1040384, 529408, 1040384, 0, smgr relation table
",,,,,,,0,,,,

2023-11-01 14:29:44.919294 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 0, 0, 0, 0, SP-GiST temporary context ",,,,,,,0,,,,

2023-11-01 14:29:44.919302 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 0, 0, 0, 0, GiST temporary context ",,,,,,,0,,,,

2023-11-01 14:29:44.919311 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 0, 0, 0, 0, GIN recovery temporary context ",,,,,,,0,,,,

2023-11-01 14:29:44.919321 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 131072, 63040, 131072, 0, MdSmgr ",,,,,,,0,,,,

2023-11-01 14:29:44.919329 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 8192, 3664, 8192, 0, MdSmgr/Pending Ops Table ",,,,,,,0,,,,

2023-11-01 14:29:44.919338 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 8192, 0, 8192, 0, MdSmgr ",,,,,,,0,,,,

2023-11-01 14:29:44.919348 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 8192, 3664, 8192, 0, MdSmgr/Pending Ops Table ",,,,,,,0,,,,

2023-11-01 14:29:44.919370 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 8192, 0, 8192, 0, MdSmgr ",,,,,,,0,,,,

2023-11-01 14:29:44.919377 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 8192, 3664, 8192, 0, MdSmgr/Pending Ops Table ",,,,,,,0,,,,

2023-11-01 14:29:44.919387 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 8192, 1600, 8192, 0, LOCALLOCK hash ",,,,,,,0,,,,

2023-11-01 14:29:44.919396 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 104112, 3664, 104112, 0, Timezones ",,,,,,,0,,,,

2023-11-01 14:29:44.919406 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 8192, 7360, 8192, 0, MemoryAccountMemoryContext ",,,,,,,0,,,,

2023-11-01 14:29:44.919415 MSK,,,p51986,th1003579264,,,,0,,,seg-1,,,,,"LOG","00000","context: 1, 12336, 7760, 16480, 4144, ErrorContext
```

FTS. Double fault



FTS double fault detected (content=17) primary dbid=19, mirror dbid=37

20220909:09:51:39:3106135 gpstart:rc1b-72ac8bfqle682d3d:gpadmin-[ERROR]:-
No segment started for content: 2.

```
postgres=# select * from gp_segment_configuration where hostname ~'host1' and status='u';
 dbid | content | role | preferred_role | mode | status | port |                 hostname
------+---------+------+----------------+------+--------+------+----------------------------------
   37 |       3 | p    | p              | n    | u      | 6003 | host1
   36 |       2 | p    | p              | n    | u      | 6002 | host1
   47 |      13 | p    | p              | n    | u      | 6013 | host1
(3 rows)

postgres=# select * from gp_segment_configuration where hostname ~'host2' and status='d';
dbid | content | role | preferred_role | mode | status | port |                 hostname
------+---------+------+----------------+------+--------+------+----------------------------------
    5 |       3 | m    | m              | n    | d      | 7003 | host2
    4 |       2 | m    | m              | n    | d      | 7002 | host2
   15 |      13 | m    | m              | n    | d      | 7013 | host2
(3 rows)
```

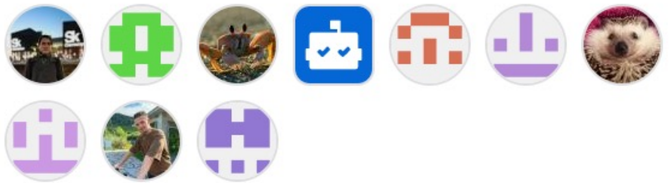
Доступность. Мастер

About

PgConsul is a tool for maintaining High-Availability Postgresql cluster configurations. It is responsible for cluster recovery in case of emergencies.

- Readme
- View license
- Activity
- Custom properties
- **19** stars
- **11** watching
- **5** forks

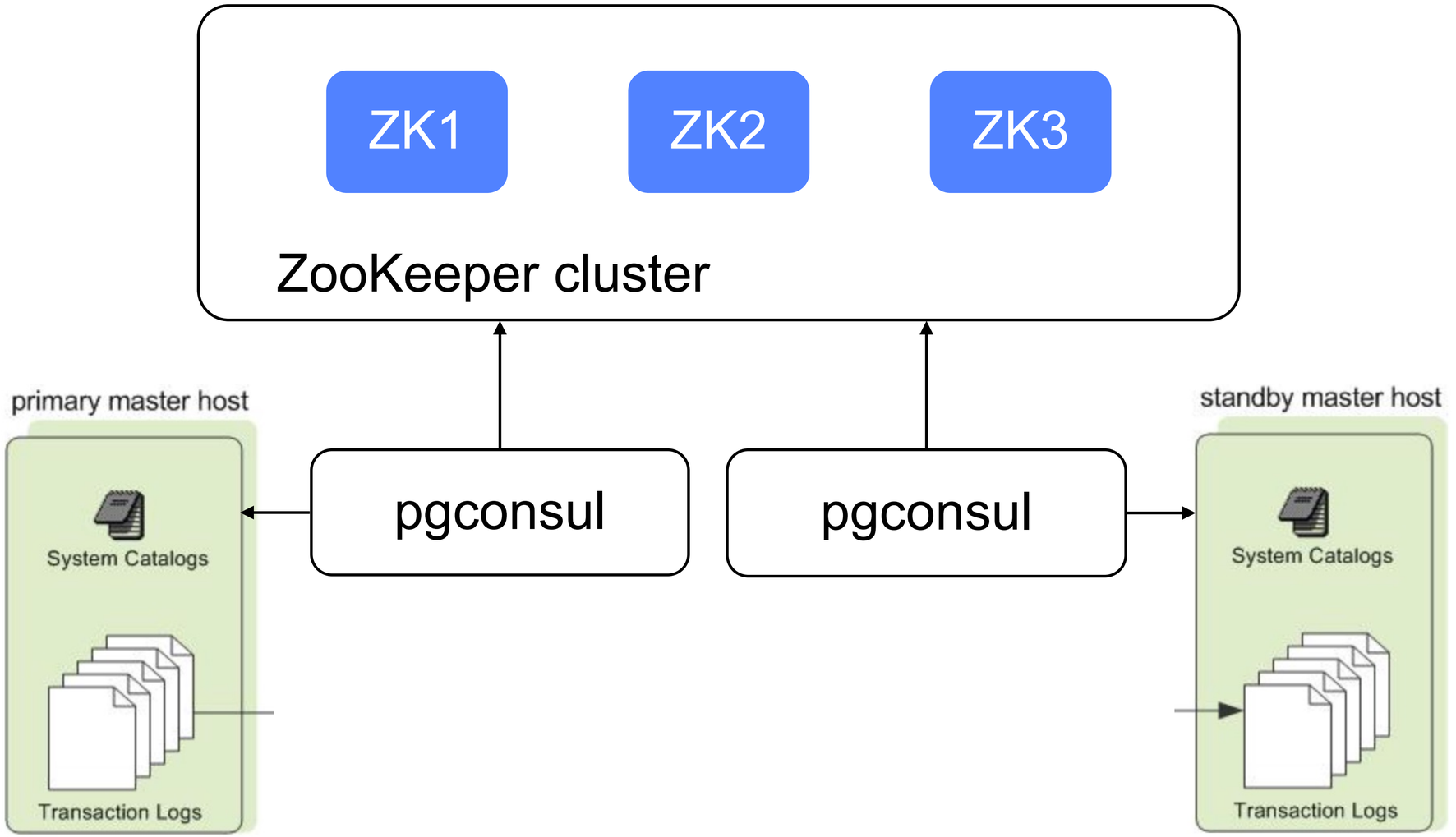
Contributors 10



Languages

- **Python** 51.8%
- **Gherkin** 40.6%
- **Shell** 2.6%
- **Clojure** 2.1%
- **Dockerfile** 1.9%
- **Makefile** 1.0%

PgConsul — архитектура



PgConsul — функции

1. Запускает мастер, если остановлен
2. Switchover
3. Запускает реплику, если остановлена
4. Создаёт новую реплику, если старая сломана

PgConsul — алгоритм мастера

1. Берём блокировку в ZooKeeper
2. Проверяем работу сервиса, если нужно, запускаем
3. Если сервис не стартует, отпускаем блокировку и завершаем работу
4. Проверяем наличие standby
5. Если standby не найден, берём блокировку на gp_segment_configuration
6. Выполняем gpinitstandby

PgConsul — алгоритм реплики

1

Проверяем блокировку в ZooKeeper, если смогли взять — мы мастер, активируем standby и следуем алгоритму мастера

2

Проверяем работу сервиса, запускаем standby

3

Если сервис не стартует либо standby не накатывает логи, останавливаем сервис и удаляем данные

4

Ждём инициализации standby

1. Клаудизация GP
2. Отказоустойчивость
3. Backup/Restore
4. Expand

Резервное копирование

Gpbackup — логический бэкап!

Когда нужен логический бэкап:

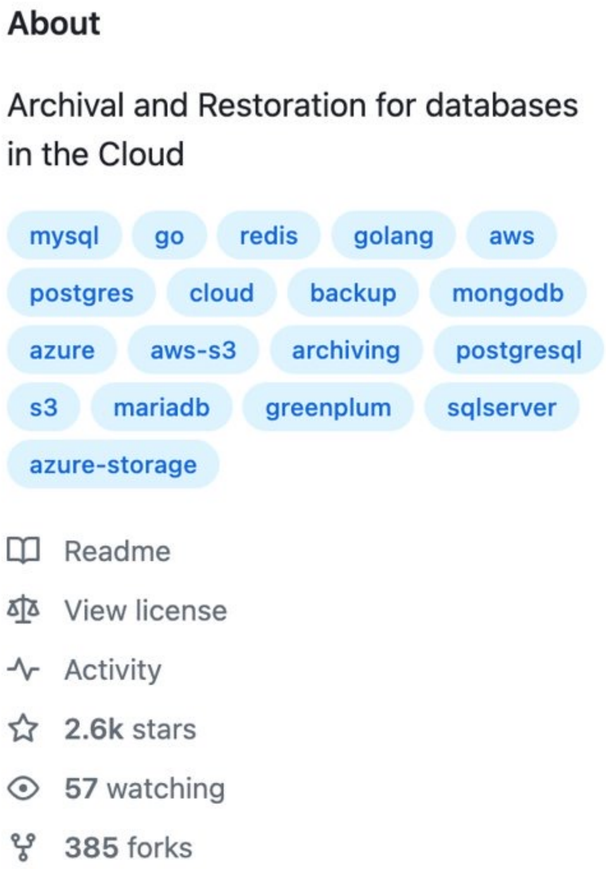
- Меньше размер
- Можно загрузить на другое железо в другом кластер

Недостатки:

- При восстановлении нужно залить **заново всю** БД

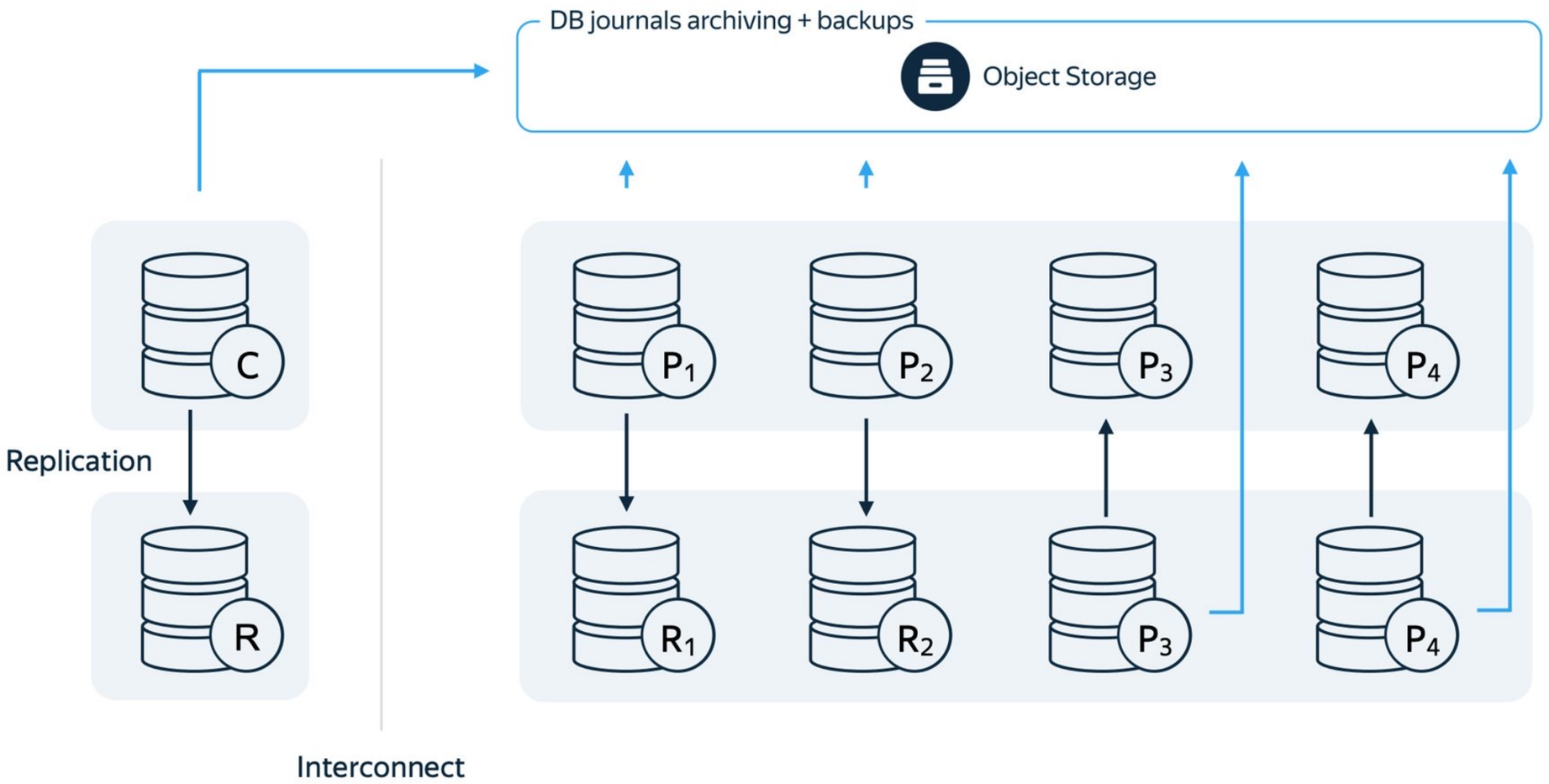
Резервное копирование

WAL-G — https://wal-g.readthedocs.io/

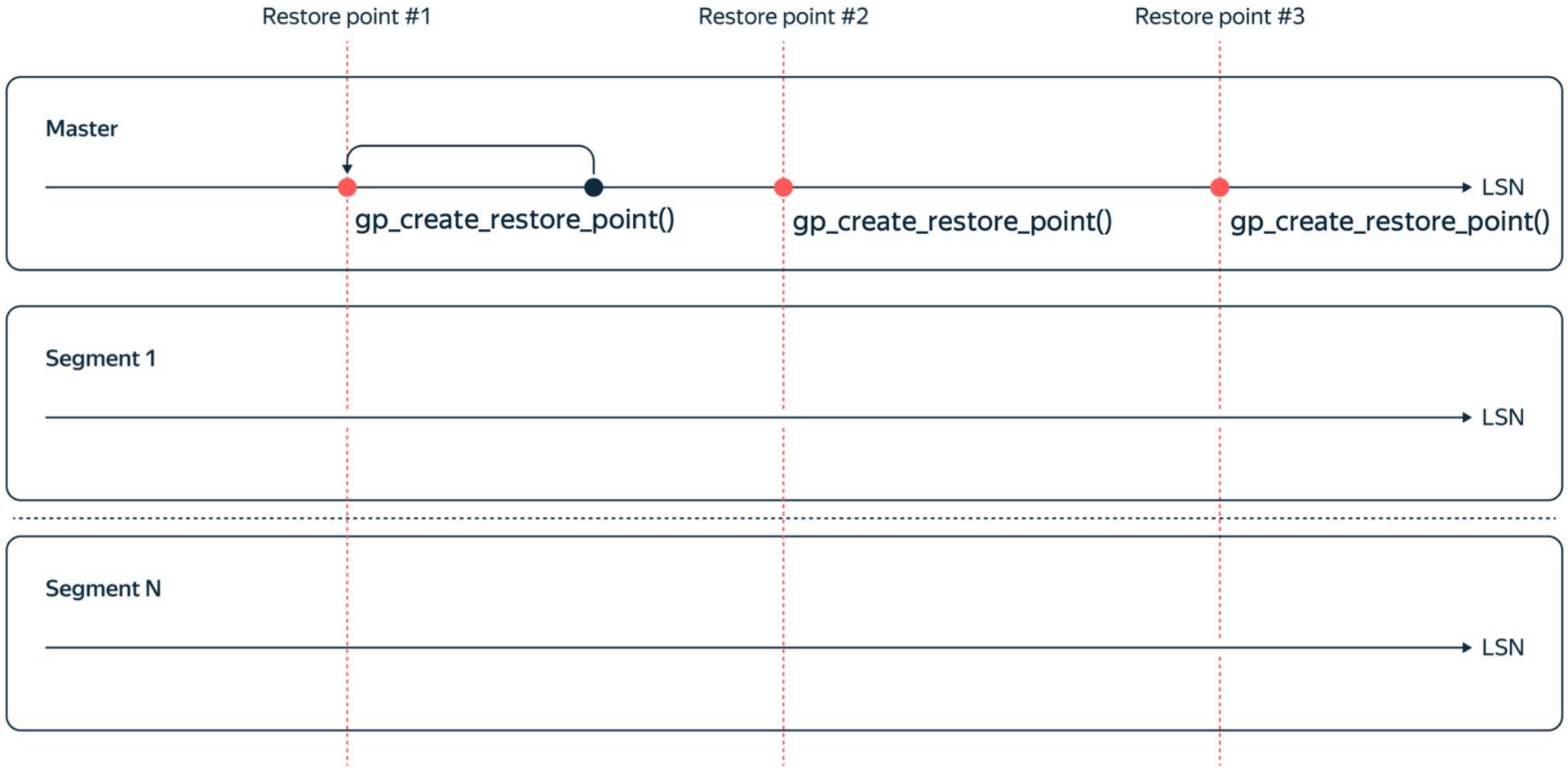


Подробнее
https://highload.rs/2023/abstracts/9752

Как устроен backup Greenplum → S3

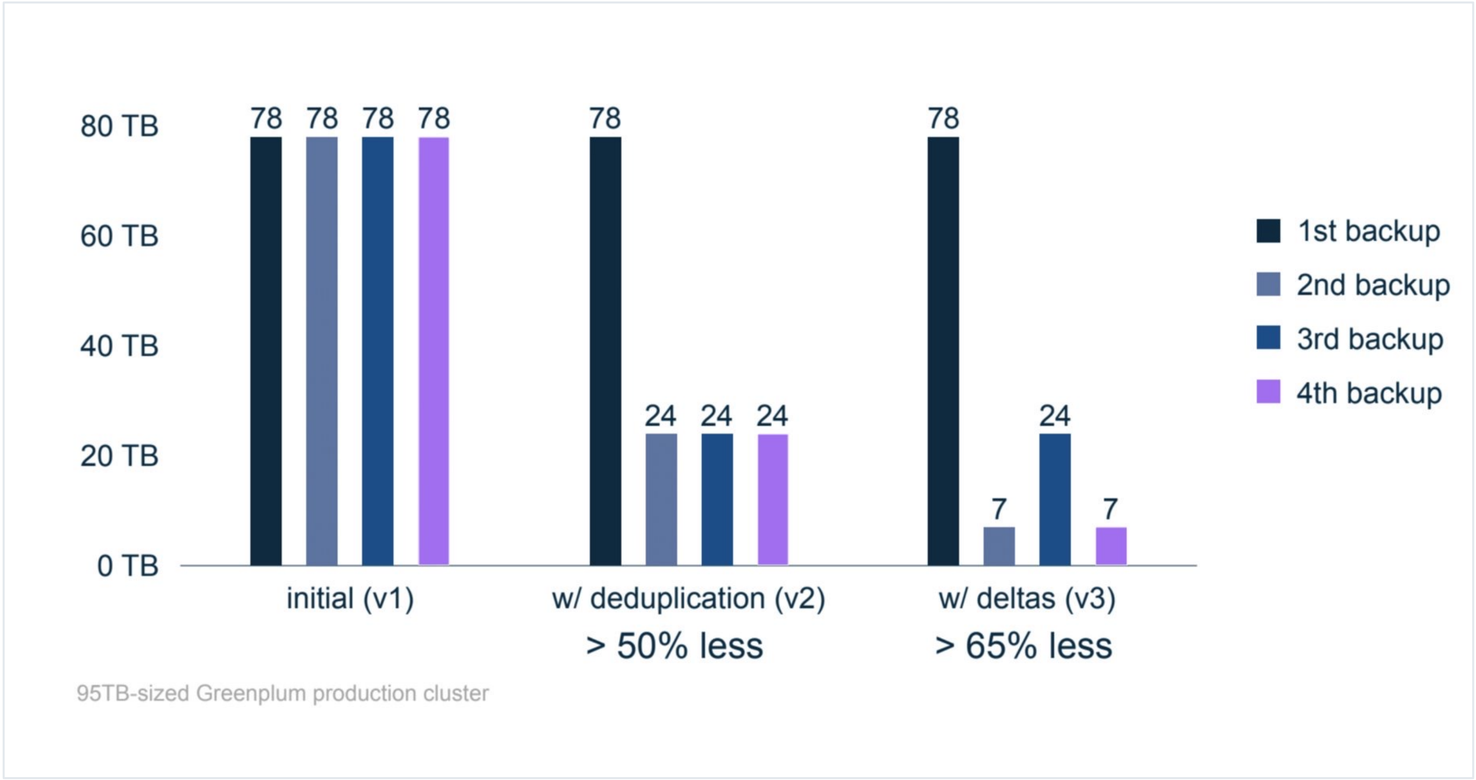


Восстановление на момент в прошлом PITR



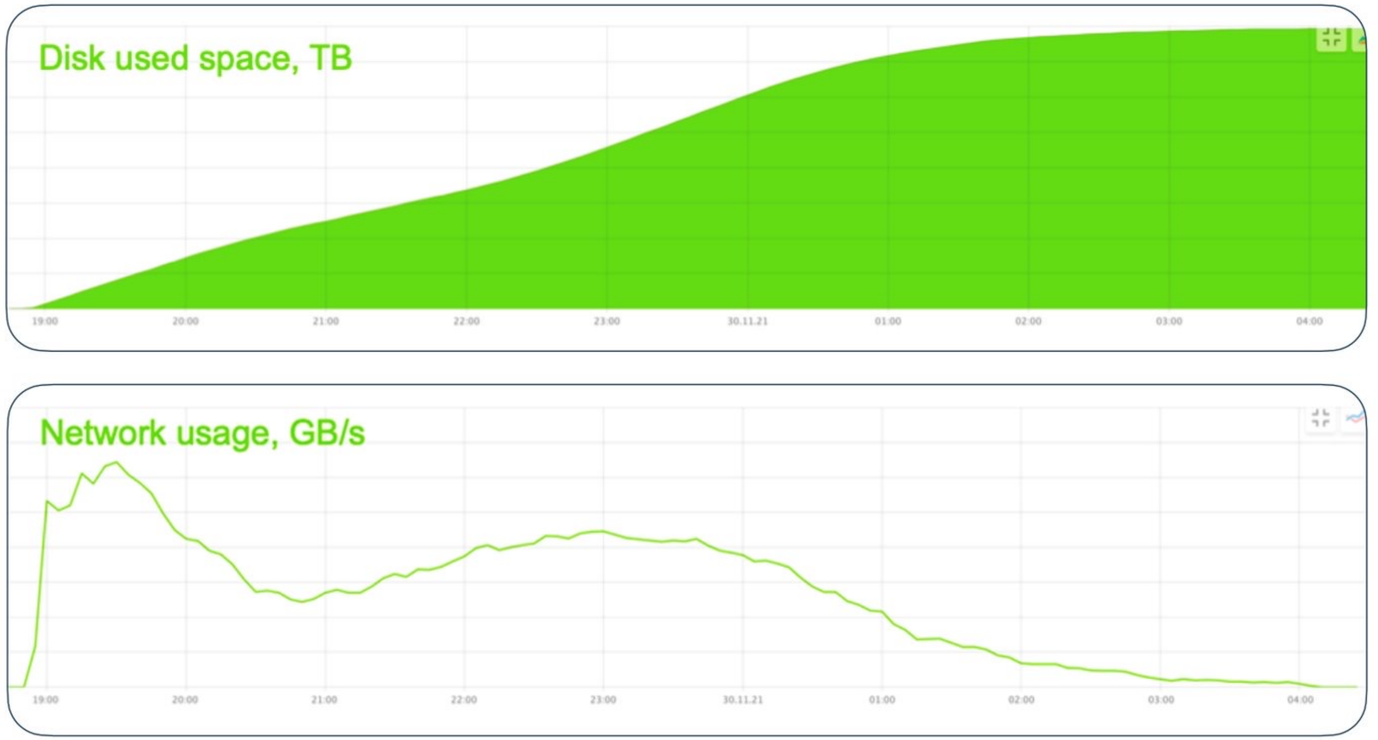
Фичи backup

- Data deduplication (WAL и AO/AOCS)
- Incremental block backup (delta AO/AO CS и Heap)



Восстановление из резервной копии

- 50 ТБ данных (100 ТБ на диске)
- Восстанавливаются за 3 часа
- Restore point можно делать каждые 10 минут



DR для Greenplum в Yandex Cloud

«Быстро поднятое
не считается упавшим»

Быстрое восстановление →
резервный кластер для DR не нужен

Реконфигурация с помощью backup/restore

Сложная задача:
изменить число сегментов на ноду

Ресурсоёмкие расчёты

1

Создаём новый кластер из бэкапа

2

Выполняем сложные вычисления

3

Выгружаем результат / делаем бэкап

4

Удаляем кластер

1. Клаудизация GP
2. Отказоустойчивость
3. Backup/Restore
4. Expand

Expand. Prepare

Характерное время PREPARE — несколько часов. Зависит от количества таблиц в кластере

Greenplum Database System Expansion Checklist

Online Pre-Expansion Tasks
* System is up and available

- ☐ Plan for ordering, building, and networking new hardware platforms, or provisioning cloud resources.
- ☐ Devise a database expansion plan. Map the number of segments per host, schedule the downtime period for testing performance and creating the expansion schema, and schedule the intervals for table redistribution.
- ☐ Perform a complete schema dump.
- ☐ Install Greenplum Database binaries on new hosts.
- ☐ Copy SSH keys to the new hosts (`gpssh-exkeys`).
- ☐ Validate disk I/O and memory bandwidth of the new hardware or cloud resources (`gpcheckperf`).
- ☐ Validate that the coordinator data directory has no extremely large files in the `log` directory.

Offline Pre-Expansion Tasks
* The system is unavailable to all user activity during this process.

- ☐ Validate that there are no catalog issues (`gpcheckcat`).
- ☐ Validate disk I/O and memory bandwidth of the combined existing and new hardware or cloud resources (`gpcheckperf`).

Online Segment Instance Initialization
* System is up and available

- ☐ Prepare an expansion input file (`gpexpand`).
- ☐ Initialize new segments into the system and create an expansion schema (`gpexpand -i input_file`).

Expand. Redistribute

Характерное время REDISTRIBUTE — 1–3 дня.
Зависит от количества таблиц и данных в кластере

Online Expansion and Table Redistribution
* System is up and available

- [ ] Before you start table redistribution, stop any automated snapshot processes or other processes that consume disk space.
- [ ] Redistribute tables through the expanded system (`gpexpand`).
- [ ] Remove expansion schema (`gpexpand -c`).
- [ ] **Important:** Run `analyze` to update distribution statistics.

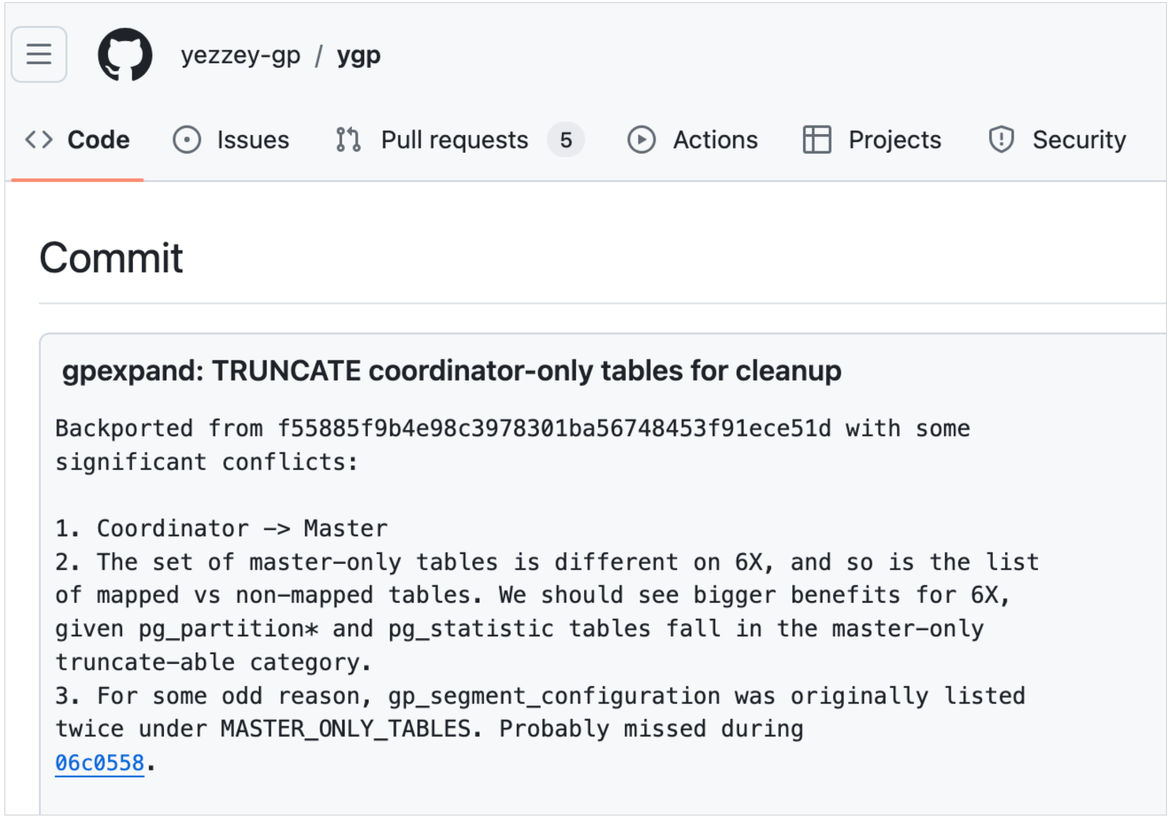
  During the expansion, use `gpexpand -a`, and post-expansion, use `analyze`.

Back Up Databases
* System is up and available

- [ ] Back up databases using the `gpbackup` utility. Backups you created before you began the system expansion cannot be restored to the newly expanded system because the `gprestore` utility can only restore backups to a Greenplum Database system with the same number of segments.

Улучшения

1. Do not expand empty tables
2. Do not expand unlogged table
3. gpexpand: TRUNCATE coordinator-only tables for cleanup
4. Use fast shutdown during expand
5. gpexpand process should skip aleady expanded tables



Стратегии redistribute

- Закрыть кластер
- Запрещаем в pg_hba соединение к БД
- Пользовательская нагрузка *СУЩЕСТВЕННО* замедляет Expand. И наоборот, в процессе Expand произвольные пользовательские запросы могут ждать блокировок и работать *СУЩЕСТВЕННО* медленнее

Стратегии redistribute

1. Выполнять в фоне
2. Распределяем небольшими порциями в окно обслуживания
3. Берут эксклюзивную блокировку
4. Не работает, если используется exchange partition

Советы

1. Автоматизируйте — bash/ansible/salt
2. Используйте cgroup и не допускайте overcommit памяти/CPU
3. Физический бэкап
4. Обновляйтесь
5. Тестируйте изменения

Готов ответить на ваши вопросы

Леонид Борчук
Team Lead Greenplum, Yandex Cloud