阿里云

DB GEEK IT大咖说 知识分享平台

Greenplum备份恢复浅析

姓名：张文杰
邮箱：zhuodao.zwj@alibaba-inc.com
公司：阿里云

PostgreSQL

Greenplum数据备份恢复：

1. 数据量较大

2. 不能完全使用Xlog日志备份

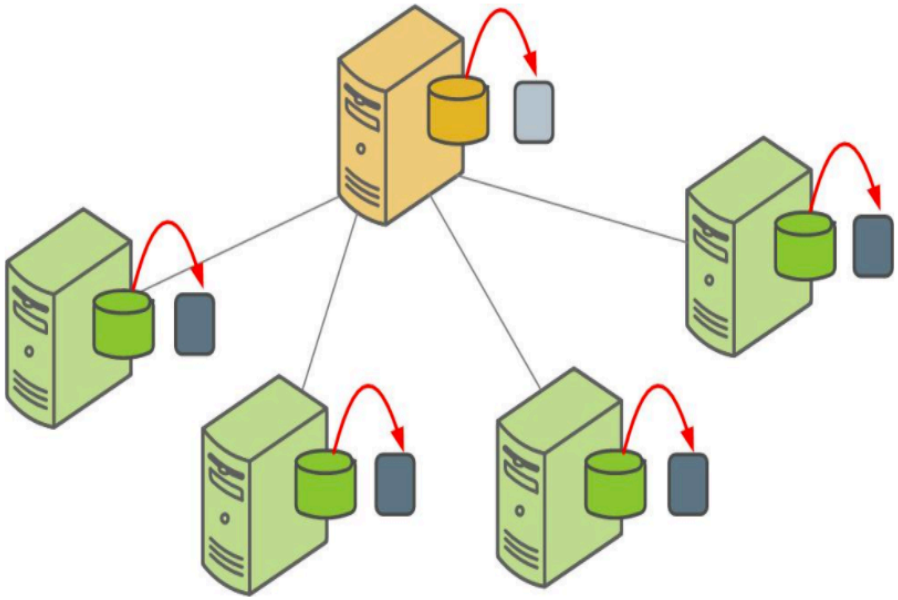
3. 需要保证数据完整性和一致性

Greenplum提供了：

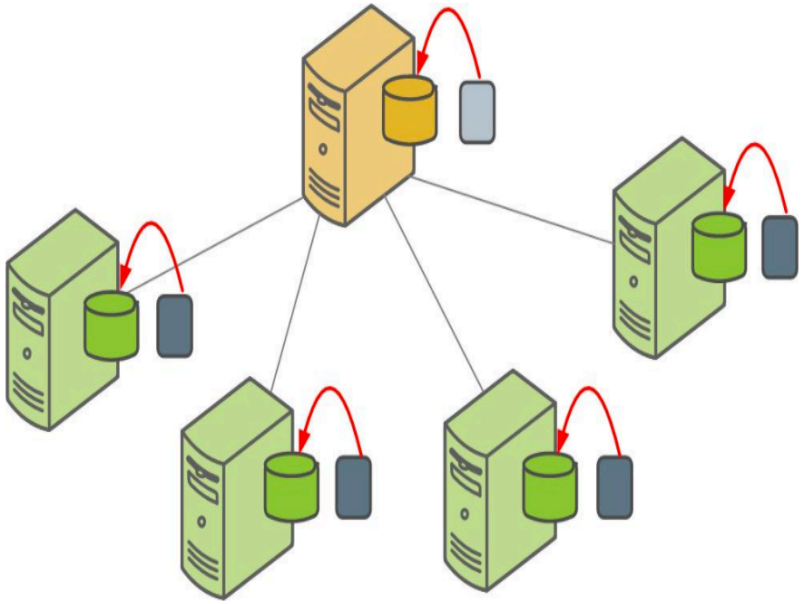
1. 非并行备份和恢复：
   --pg_dump和pg_dumpall（pg_restore）
   --copy、psql

2. 并行备份和恢复
   --gpcrondump（gpdbrestore）

并行备份和恢复



gpcrondump



gpdbrestore

gpcrondump用法(1/3)

gpcrondump -x database_name
[-s <schema> | -S <schema> | -t <schema>.<table> | -T <schema>.<table>]
[--table-file=<filename> | --exclude-table-file=<filename>]
[--schema-file=<filename> | --exclude-schema-file=<filename>]
[-u backup_directory] [-R post_dump_script] [--incremental]
[-K <timestamp> [--list-backup-files]]
[--prefix <prefix_string> [--list-filter-tables]]
[-c] [-z] [-r] [-f <free_space_percent>] [-b] [-h] [-j | -k]
[-g] [-G] [-C] [-d <master_data_directory>] [-B <parallel_processes>]
[-a] [-q] [-y <reportfile>] [-l <logfile_directory>]
[--email-file <path_to_file>] [-v]
{ [-E encoding] [--inserts | --column-inserts] [--oids]
[--no-owner | --use-set-session-authorization]
[--no-privileges] [--rsyncable]

gpcrondump用法(2/3)

gpcrondump命令使用-K <timestamp>来指定唯一时间戳来标示某个备份集文件，其中如果当前备份目录中存在更未来的备份集，则备份报错。

gpcrondump命令使用-t 或者--table-file，-T或者--exclusive-table-file，-s或者--schema-file，-S或者--exclusive-schema-file灵活指定需要全量备份的某个table或者某个schema，其中-s和-t选项不能同时使用

gpcrondump命令使用选项--incremental和--prefix执行增量备份，但是这里的增量备份实际上只对有如下操作的表进行备份；

ALTER TABLE
DELETE
INSERT
TRUNCATE
UPDATE
DROP and then re-create the table

gpcrondump用法(3/3)

gpcrondump命令使用--list-backup-files可以将备份产生的所有文件都列举出来，分为两类，分别是：

1. Pipes files 包括每个segment的数据文件（可以流式输出）、master产生的post_data文件包含indexes, triggers, primary key constraints等数据库对象，master产生的全局对象包含角色和表空间等

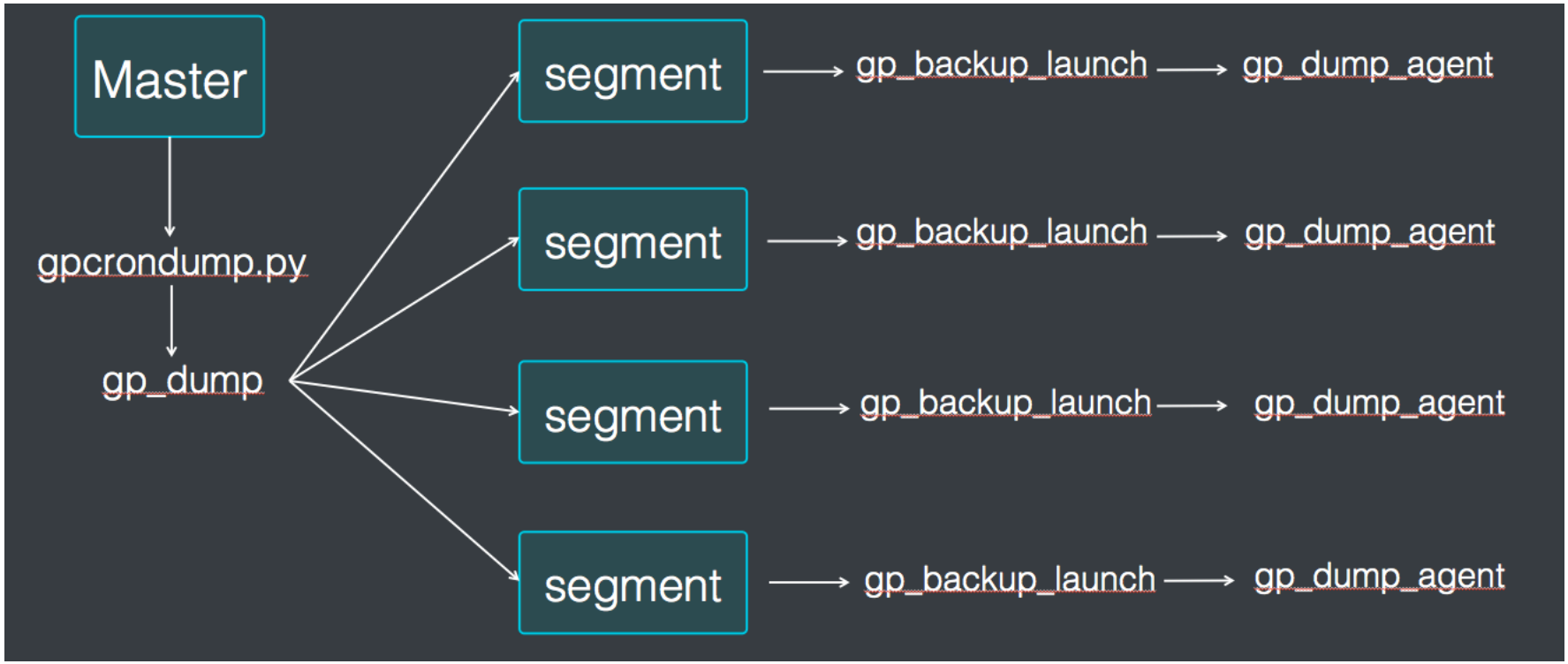
2. Regular files 包括各类辅助文件，例如存储着create database语句的文件，备份状态报告文件等

其中最核心的每个segment的数据文件，命名格式如下：

xxx_gp_dump_0_2_20170206160253.gz

其中XXX表示用户定义的文件前缀，0代表是非master节点，2代表该文件产生的segment对应dbid，20170206160253是前面所说的时间戳。在恢复时，会根据这个命名规则，找到对应的文件。

gpcrondump具体实现(1/2)

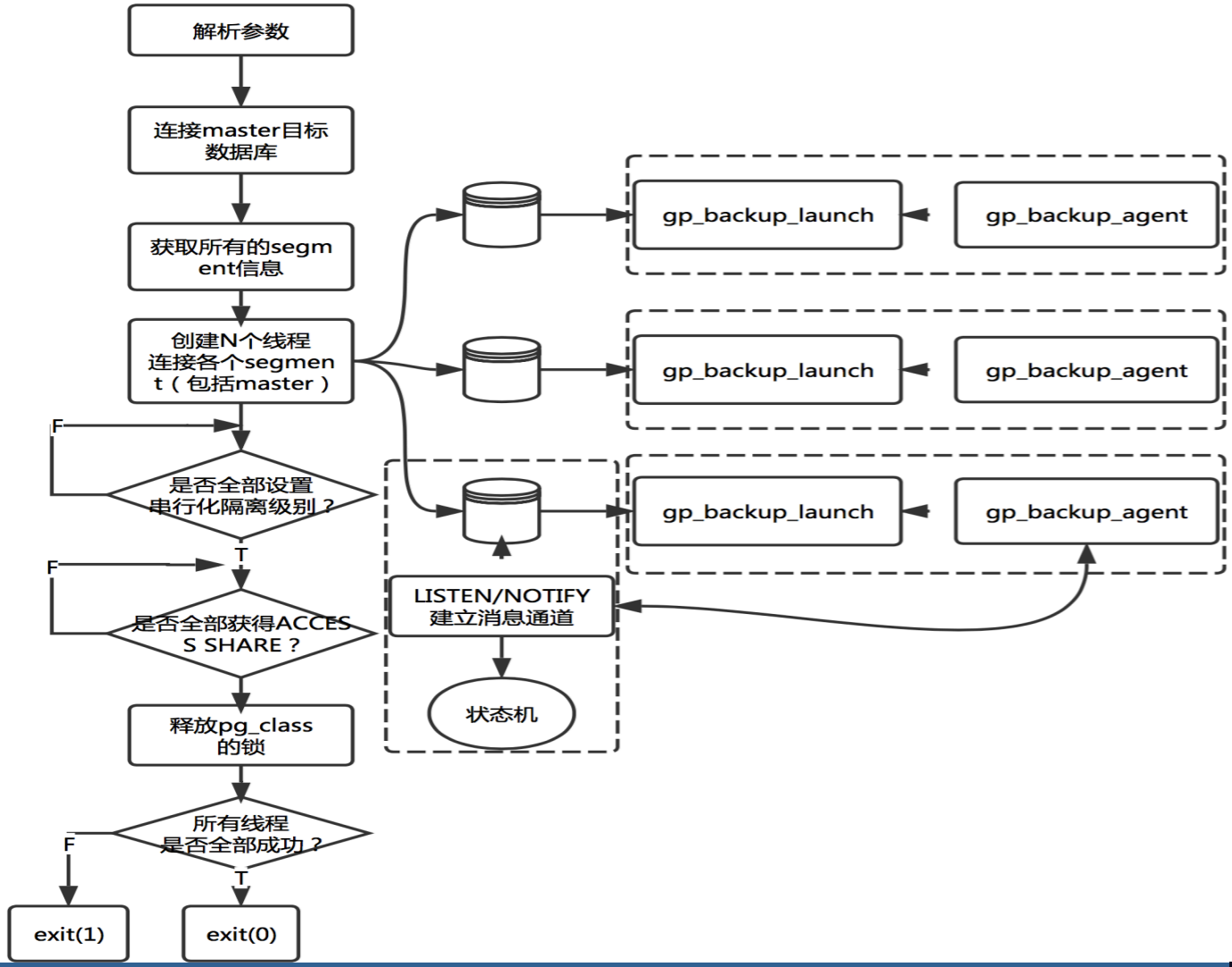


gpcrondump具体实现(2/2)

gpcrondump实际是对gp_dump的封装，具体步骤如下：

1. 读取参数，检测合理性
2. master执行对pg_class加锁操作
3. 封装并执行gp_dump命令
4. 检测每个segment备份状态
5. 其他操作，例如备份全局对象（角色和表空间）、备份config文件、清理旧备份集以及VACCUM等

gp_dump具体实现


解析参数
连接master目标数据库
获取所有的segment信息
创建N个线程连接各个segment（包括master）
是否全部设置串行化隔离级别？
F
T
是否全部获得ACCESS SHARE？
F
T
释放pg_class的锁
所有线程是否全部成功？
F
T
exit(1)
exit(0)
LISTEN/NOTIFY建立消息通道
状态机
gp_backup_launch
gp_backup_agent
gp_backup_launch
gp_backup_agent
gp_backup_launch
gp_backup_agent

数据恢复(1/2)

非并行数据恢复

如果恢复前后的数据库节点个数不同，则推荐使用非并行数据恢复，不过需要保证备份集完整，而且都位于master所在的机器上，具体执行步骤如下

1.createdb database_name

2.psql database_name -f /gpdb/backups/gp_dump_1_1_20120714

3.$ psql database_name -f /gpdb/backups/gp_dump_0_2_20120714

$ psql database_name -f /gpdb/backups/gp_dump_0_3_20120714

$ psql database_name -f /gpdb/backups/gp_dump_0_4_20120714

$ psql database_name -f /gpdb/backups/gp_dump_0_5_20120714

4.psql database_name -f /gpdb/backups/gp_dump_0_5_20120714_post_data

5. gunzip -c /data/gpdb/master/gpseg-1/db_dumps/20150112/gp_dump_1_1_20150112140316.gz | egrep “SET search_path|SELECT pg_catalog.setval" > schema_path_and_seq_next_val

psql test_restore -f schema_path_and_seq_next_val

数据恢复(2/2)

并行恢复

如果恢复前面实例的节点个数相同，并且备份文件在对应所属的segment host上，可以使用gpdbrestore并发恢复提高恢复的速度，其使用方法如下：

- gpdbrestore { -t <timestamp_key> { [-L]
- | [--netbackup-service-host <netbackup_server>
- [--netbackup-block-size <size>]] }
- | -b <YYYYMMDD>
- | -R <hostname>:<path_to_dumpset>
- | -s <database_name> }
- [--noplan] [--noanalyze] [-u <backup_directory>] [--list-backup]
- [--prefix <prefix_string>] [--report-status-dir <report_directory>]
- [-T <schema>.<table> [,...]] [--table-file <file_name>]
- [--truncate] [-e] [-G]
- [-B <parallel_processes>]
- [-d <master_data_directory>] [-a] [-q] [-l <logfile_directory>]
- [-v] [--ddboost]
- [--redirect <database_name>]

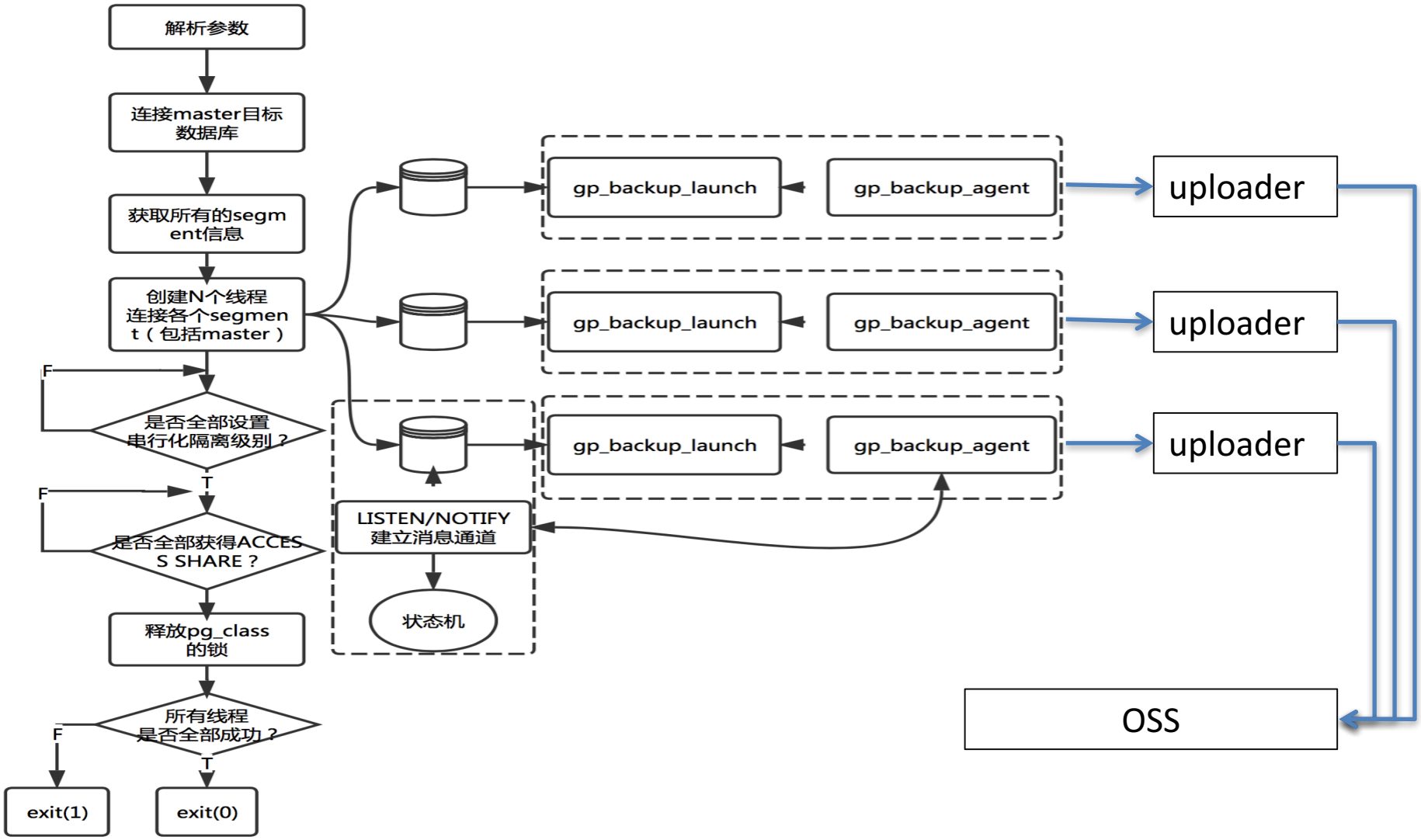
并行备份恢复存在的问题

虽然并行备份和恢复大大提高了备份和恢复的速度，但是仍然存在很多问题：

1. 大量数据需要落盘
2. 使用dbid作为备份文件命名规则，在主备切换或者primary和mirror后会出现问题
3. 可能会出现各个segment数据不一致的情况

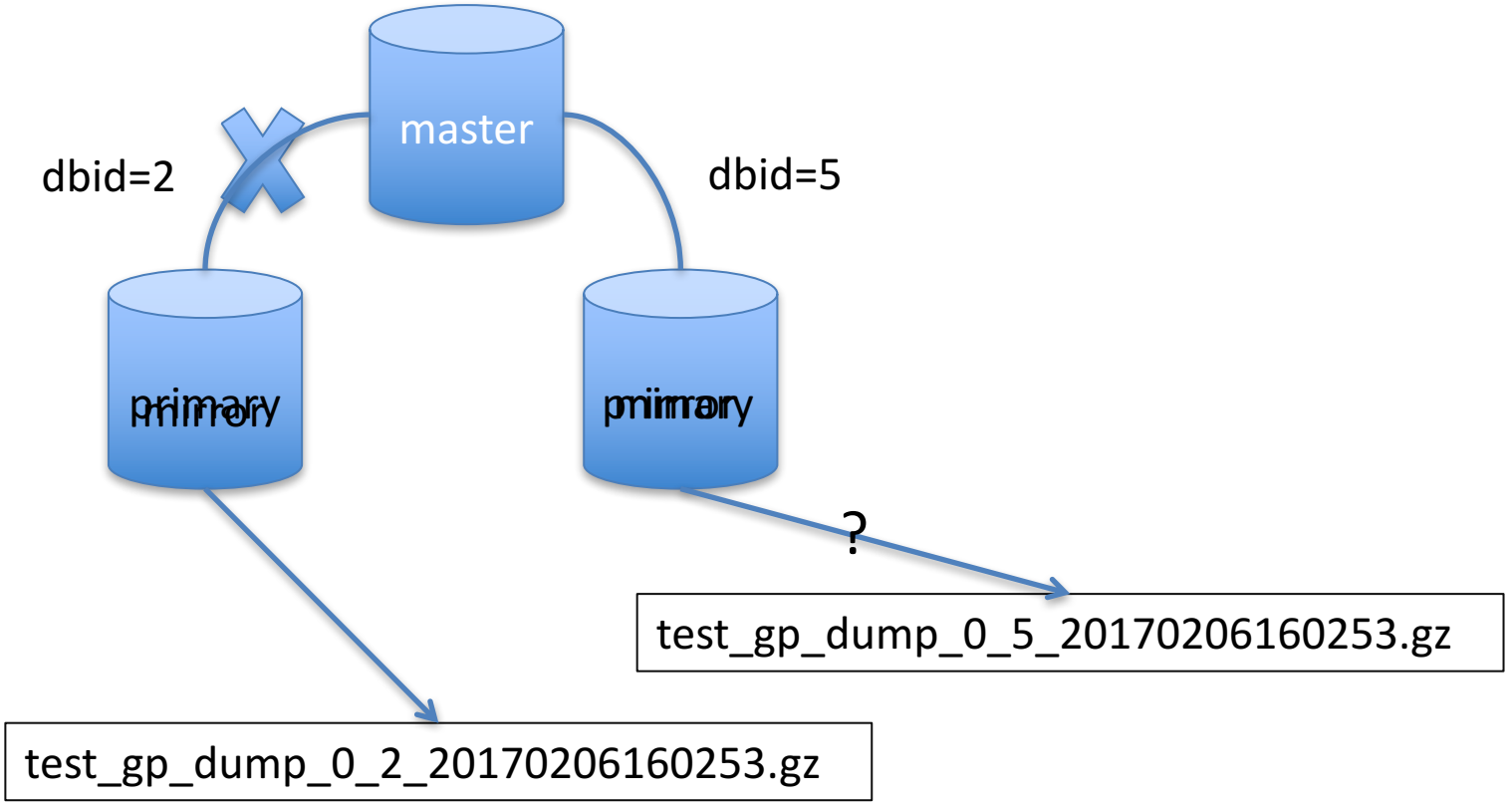
……

并行备份恢复优化(1/3)


解析参数
连接master目标数据库
获取所有的segment信息
创建N个线程连接各个segment（包括master）
是否全部设置串行化隔离级别？
是否全部获得ACCESS SHARE？
释放pg_class的锁
所有线程是否全部成功？
exit(1)
exit(0)
F
T
LISTEN/NOTIFY建立消息通道
状态机
gp_backup_launch
gp_backup_agent
gp_backup_launch
gp_backup_agent
gp_backup_launch
gp_backup_agent
uploader
uploader
uploader
OSS

并行备份恢复优化(2/3)

使用dbid作为备份文件命名规则，在主备切换或者primary和mirror后会出现问题，例如：



并行备份恢复优化(3/3)

目前GP的并行备份，为了保证数据的一致性，有以下2步：

1. 给pg_class 加排他锁
2. 每个segment备份时设置隔离级别为串行化，保证每个segment的数据一致性

但是，各个segment的数据设置隔离级别的动作存在时间差，而master仍然接受新的事务，从而导致各个segment上的数据不一致。我们可以通过实现barrier机制来避免这种情况：

1. 使数据库只读
2. 等待所有的事务全部提交，开始备份
3. 给pg_class加锁，等待每个segment备份时设置隔离级别为串行化，恢复数据库为可读可写

Thanks!

Q & A