流复制完成, 现在我们需要使用pgtool-ii +pg实现HA主备切换

简言之: 提供对外统一ip(虚拟) , 容灾自动切换, 在这里完成.

# 免密设置

两台机器均在 /etc/hosts 添加：

10.242.111.204 master

10.242.111.207 slave

10.242.111.203 vip

两台机器均声称ssh密钥：

su – postgres

-bash-4.2$ ssh-keygen

分别将master的公钥复制到slave的authorized\_keys，slave的公钥复制到master的authorized\_keys

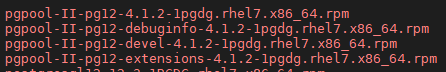
# 安装pgpoll-ii（两台机器均执行）

1. 使用rpm离线安装

下载好相应的离线包上传至服务器，需要注意的版本问题， pgpool的每个版本理论上都会出pg每个版本对应的包。从ftp路径就能反映出来：

https://www.pgpool.net/yum/rpms/**3.7**/redhat/rhel-7-x86\_64/

https://www.pgpool.net/yum/rpms/**4.1**/redhat/rhel-7-x86\_64/

比如上面两个地址，第一个是pgpool-ii3.7版本对应的目录，下面是pgpool-ii4.1版本对应的目录， 每个目录中都会有不同版本postgresql的关联包， postgre的版本号具体体现在文件名中的pgXX这个数字中。

安装命令

#rpm -ivh pg\*.rpm

注：源码编译安装参照官方指定文档：<https://www.pgpool.net/docs/pgpool-II-3.2.1/tutorial-zh_cn.html#install>

2、这里稳妥起见需要配置一个系统权限命令，让postgres用户执行切换脚本需要，可以让其他普通用户执行：

chmod u+x /usr/sbin/ip

chmod u+s /usr/sbin/arping

chmod u+s /sbin/ip

chmod u+s /sbin/ifconfig

3、设置pid目录权限和创建日志目录（后续我们把日志放在这）

chown postgres.postgres /var/run/pgpool

mkdir -p /var/log/pgpool/

touch /var/log/pgpool/pgpool\_status

chown -R postgres.postgres /var/log/pgpool/

# 配置自动切换

## 配置pool\_hba.conf（主备相同）

pool\_hba.conf是对登录用户进行验证的，要和pg的pg\_hba.conf保持一致，要么都是trust，要么都是md5验证方式，这里采用了md5验证方式如下设置：

[root@localhost ~]# su - postgres

-bash-4.2$ cd /etc/pgpool-II/

-bash-4.2$ vim pool\_hba.conf

#编辑内容如下(这里和postgressql设置一样, trust/md5保持一致)

# IPv4 local connections:

host all all 0.0.0.0/0 md5

host all all 0/0 md5

## 配置pcp.conf（主备相同）

-bash-4.2$ pg\_md5 postgres

e8a48653851e28c69d0506508fb27fc5

-bash-4.2$ vim pcp.conf

#编辑内容如下(pcp.conf是pgpool管理器自己的用户名和密码，用于管理集群。):

# USERID:MD5PASSWD

postgres: e8a48653851e28c69d0506508fb27fc5

#在pgpool中添加pg数据库的用户名和密码

[postgres@etc~]$ pg\_md5 -p -m -u postgres pool\_passwd

#数据库登录用户是postgres,这里输入登录密码，不能出错

#输入密码后，在/etc/pgpoll-II目录下会生成一个pool\_passwd文件

## 配置pgpool.conf

### 主库(10.242.111.204)-pgpool.conf 配置

#基础配置

pid\_file\_name = '/var/run/pgpool/pgpool.pid'

listen\_addresses = '\*'

port = 9999

pcp\_listen\_addresses = '\*'

pcp\_port = 9898

backend\_hostname0 = 'master'

backend\_port0 = 5432

backend\_weight0 = 5

backend\_data\_directory0 = '/var/lib/pgsql/12/data'

backend\_flag0 = 'ALLOW\_TO\_FAILOVER'

backend\_hostname1 = 'slave'

backend\_port1 = 5432

backend\_weight1 = 5

backend\_data\_directory1 = '/var/lib/pgsql/12/data'

backend\_flag1 = 'ALLOW\_TO\_FAILOVER'

enable\_pool\_hba = on

pool\_passwd = 'pool\_passwd'

replication\_mode = off

load\_balance\_mode = on

master\_slave\_mode = on

master\_slave\_sub\_mode = 'stream'

sr\_check\_period = 5

sr\_check\_user = 'repuser'

sr\_check\_password = 'repuser'

sr\_check\_database = 'postgres'

delay\_threshold = 1024

health\_check\_period = 10

health\_check\_timeout = 20

health\_check\_user = 'postgres'

health\_check\_password = 'postgres'

health\_check\_database = 'postgres'

health\_check\_max\_retries = 0

health\_check\_retry\_delay = 1

#故障转移,follow\_master\_command多台需要配置

failover\_command = '/etc/pgpool-II/failover.sh %d %h %p %D %m %H %M %P %r %R %N %S'

fail\_over\_on\_backend\_error = off

use\_watchdog = on

trusted\_servers = ''

ping\_path = '/bin'

wd\_hostname = 'master'

wd\_port = 9000

wd\_priority = 1

wd\_authkey = ''

wd\_ipc\_socket\_dir = '/var/run/postgresql'

delegate\_IP = 'vip'

if\_cmd\_path = '/sbin'

# 网卡名字需要正确配置

if\_up\_cmd = '/usr/bin/sudo /sbin/ip addr add $\_IP\_$/24 dev ens160 label ens160:0'

if\_down\_cmd = '/usr/bin/sudo /sbin/ip addr del $\_IP\_$/24 dev ens160'

arping\_cmd = '/usr/bin/sudo /usr/sbin/arping -U $\_IP\_$ -w 1 -I ens160'

wd\_lifecheck\_method = 'heartbeat' # rtm 看门狗健康检测方法

wd\_heartbeat\_port = 9694

wd\_heartbeat\_keepalive = 2

wd\_heartbeat\_deadtime = 30

heartbeat\_destination0 = 'slave'

heartbeat\_destination\_port0 = 9694

heartbeat\_device0 = 'ens160'

wd\_life\_point = 3 # rtm 生命检测失败后重试次数

wd\_lifecheck\_query = 'SELECT 1' # rtm 用于检查 pgpool-II 的查询语句。

wd\_lifecheck\_dbname = 'postgres' # rtm 检查健康状态的数据库名称

wd\_lifecheck\_user = 'postgres' # rtm 检查数据库的用户，该用户需要在Postgres数据库存在，且有查询权限

wd\_lifecheck\_password = 'postgres' # rtm 看门狗健康检查用户密码

other\_pgpool\_hostname0 = 'slave'

other\_pgpool\_port0 = 9999

other\_wd\_port0 = 9000

#配置故障库恢复后变更为备库

#多pgpool将不可用

#recovery\_user = 'postgres' # Online recovery user

#recovery\_password = 'postgres' # Online recovery password

#recovery\_1st\_stage\_command = 'recovery\_1st\_stage'

参数含义参照：<https://www.pgpool.net/docs/latest/en/html/runtime-config.html>

### 备库(10.242.111.207)-pgpool.conf 配置

将主库的pgpool.conf拷贝过来

修改项：

wd\_hostname = 'slave'

heartbeat\_destination0 = 'master'

other\_pgpool\_hostname0 = 'master'

### 此配置文件中的脚本

此处一共需要四个脚本分别为（脚本中的路径请注意修改）:

failover.sh 目录为: /usr/local/pgpool/etc/下

follow\_master.sh 目录为: /usr/local/pgpool/etc/下

recovery\_1st\_stage 目录为：PostgreSQL数据库数据目录下

pgpool\_remote\_start目录为：PostgreSQL数据库数据目录下

见文件夹中“pgpool脚本”文件夹

### 额外关注:

如果出现pgpool挂掉诸侯其他节点不自动执行切换策略，请查看pgpool.conf中配置项：

故障转移策略分两种：

第一种是数据库故障之后，转移

第二种是pgpool-ii节点故障后，转移

# - Watchdog consensus settings for failover -

failover\_when\_quorum\_exists = on

failover\_require\_consensus = on

allow\_multiple\_failover\_requests\_from\_node = on

enable\_consensus\_with\_half\_votes = on

## 启动验证

此之前请确保两台机器上的postgresql服务正常

使用命令启动pgpool 服务

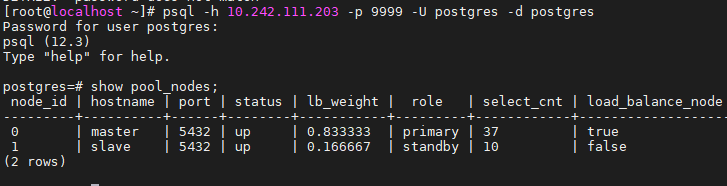
无debug日志：pgpool -n -D > /var/log/pgpool/pgpool.log 2>&1 &

有debug日志：pgpool -n -d -D > /var/log/pgpool/pgpool.log 2>&1 &

两台机器启动完，检查完日志，如果没有问题：

可使用 psql -h vip -p9999 -Upostgres -d postgres 登陆， 注意是那个**虚拟IP**，

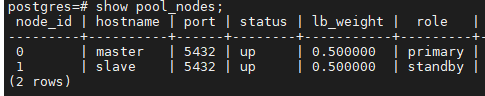
登陆之后查询集群节点信息,发现主备节点都是up正常状态：

show pool\_nodes;

#在slave节点上也是同样登陆方式(链接虚拟IP)

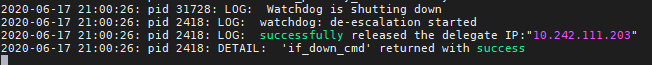
这个时候你可以使用虚拟ip 进行查询，建表，插入数据等操作来验证当前环境，是否支持：通过虚拟IP操作时的数据同步问题

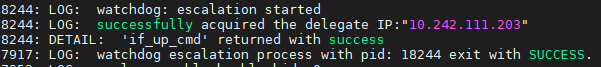
# 宕机测试

现在我们的已经完成安装的情况是：

## 模拟master端pgpool宕机



关掉的主机器上释放虚拟IP：

备机在检测完成之后接管虚拟IP：

这种方式整个流程等于就做了两件事：

主机上pgpool-ii节点挂了，就没办法维持这个虚拟IP了，备机上watchdog检测到了，就把这个虚拟IP接管过来，底层的数据库主备角色没有任何变化；

虚拟IP会在一定时间后被切换到备机上，从而保证虚拟IP一直可用。

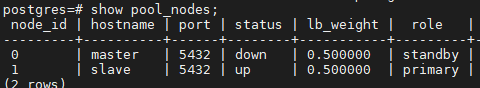
但这个检测-切换过程 需要时间，所以外界数据库链接可能需要重新链接。

重启master的pgpool之后，会自动将此pgpool-ii节点加入集群管理。

## master-postgresql宕机故障转移

1.模拟主机master postgresql 挂掉

systemctl stop postgresql-12

看一下集群状态：

**master的数据库挂掉了，同时变成了standby，slave正在作为新的主数据库提供服务。**

但是：此时通过203的访问操作等应该一切正常(此时虚拟IP仍然在204的机器上，没有转移，因为pgpool是正常存存活的)

### master恢复

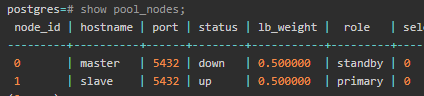
通过我们配置的在线恢复功能:

recovery\_1st\_stage\_command = 'recovery\_1st\_stage'

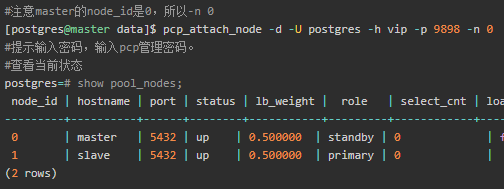
(要确保脚本在当前主服务器的数据库目录中，即现在主服务器slave的数据库集群目录中存在recovery\_1st\_stage和pgpool\_remote\_start脚本)

**pcp\_recovery\_node -h vip -p 9898 -U postgres -n 0**

这个时候的流程和手动恢复是一致的，就是说他会把master(现备机)上的data清除掉，然后从slave(现主机)使用命令将data同步过来， 然后启动服务。

启动完成之后：

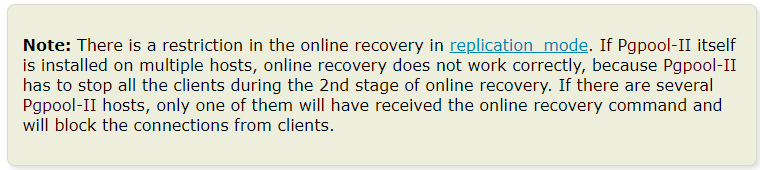
这个时候虽然数据库修复了，启动了，能在主库（207）查询到备机连接了，但是在集群中的状态仍然是down， 这个时候需要我们手动将此pgpool-ii节点加入集群

**pcp\_attach\_node -d -U postgres -h vip -p 9898 -n 0**

现在的slave为书数据库能够读写， master(standby) 为从库，只读。

同时，也可通过虚拟IP访问。

#注：通过show pool\_nodes; 查询结果可能会有缓存，重启pgpool 可以解决；

#注： 在流复制的模式下，如果 Pgpool-II本身安装在多个主机上，则pcp\_recovery\_node将无法正常使用，因为Pgpool-II必须在pcp\_recovery\_node的第二阶段停止所有客户端。

## 主机直接宕机

接上面，现在主机是slave207, 虚拟IP也挂在207的机器上， 现在207机器关机了：

备机上的pgpool-ii节点会做两件事：

* + - 1. 依照我们配置的策略，经过一段时间的重复健康检测之后
      2. 确保原主机上的postgresql-12数据库确实是连不上了，访问不了了，会执行配置的主备切换脚本failover\_command.sh脚本来将当前机器上的数据切换为主库
      3. 确保原维护虚拟IP的机器确实访问不了了，同时虚拟IP也访问不了了，就会在这太机器上重新维护虚拟IP；

主机直接宕机的情况类似于上面两种情况的综合

第二步骤类似于数据库down了,故障转移仅限于下层postgresql-12数据库

第三步骤类似于pgpool-ii节点挂了，即使切换-维护虚拟IP以保证可用，不影响下层数据库；

# 关于数据恢复的几点不足：

1. 如果pgpool-ii节点挂了，虚拟IP的切换需要时间，虽然之后的链接地址是一致的，但这个过程中，数据库链接将断开，需要重新链接（解决方案：连接池技术）
2. 只要牵扯到postgresql数据库挂掉了，那么在恢复的时候必定是要通过**pg\_basebackup**命令， 先同步数据，后启动服务，在有pgpool-ii的情况，还需要手动将这个数据库加入集群中。
3. 如果我们的集群中仅有一个pgpool节点，那么我们可以使用pgpool提供的在线恢复脚本来恢复挂掉的postgresql数据库库， 否则只能是使用命令手动恢复服务然后将节点重新加入集群；因为那个步骤当中第一步需要断开所有备库链接，但是多个pgpool节点会相互影响，所以其他pgpool-ii节点就直接拒绝了。这个无解，只能手动恢复。

pgpool在线恢复功能，也是通过：

清空备机节点数据目录 > 使用**pg\_basebackup**命令同步数据 > 启动数据库服务 > 将数据库重新加入集群

几个步骤来实现的。