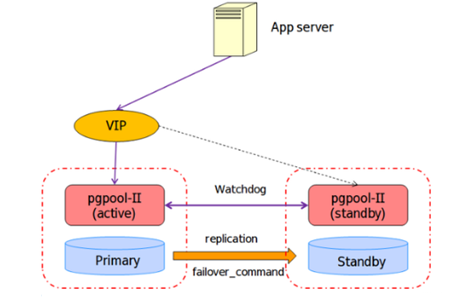
1. 原理图：



2、机器软件

10.253.43.19（postgresql9.6.2，pgpoolii3.6）(primary)

10.253.43.20（postgresql9.6.2，pgpoolii3.6）(standby)

3、步骤：

A、先将两台postgresql安装好，一台初始化启动好，另一台不用初始化（为流复制做准备），用安装用户(假设安装用户为postgres)实现在两台机器无密钥登录，ssh配置

19：

ssh-keygen

ssh-copy-id [postgres@10.253.43.19](mailto:postgres@10.253.43.19)

ssh [postgres@10.253.43.19](mailto:postgres@10.253.43.19)

20:

sh-keygen

ssh-copy-id [postgres@10.253.43.20](mailto:postgres@10.253.43.20)

ssh [postgres@10.253.43.19](mailto:postgres@10.253.43.19)

B、流复制配置：

1)、10.253.43.19 postgresql配置：

wal\_level = replica

checkpoint\_timeout = 5min

archive\_mode = on

max\_wal\_senders = 6

wal\_keep\_segments = 16

2)、在19(primary)这台机器上创建一个具有replication权限的用户

create user rep replication login encrypted password “abc”

3)、19(primary)修改认证配置文件pg\_hba.conf，添加rep的replication认证信息

# Allow replication connections from localhost, by a user with the

# replication privilege.

host all all 0.0.0.0/0 trust

host all all ::/0 trust

host replication rep 10.253.43.0/24 trust

4)、在20(standby)上测试无密码登录19(primary)

psql -h 192.168.3.201 -p 5431 -U rep -d postgres

5)、在20(standby)使用pg\_basebackup进行数据库备份恢复

pg\_basebackup -D $PGDATA -F p -X stream -v -P -h 10.253.43.19 -p 5432 -U rep

验证：查看$PGDATA目录，19的$PGDATA被拷贝到了20

6)、20(standby)配置文件postgresql.conf修改配置

hot\_standby = on

7)、配置20(standby)的recovery.conf配置文件

cp $PGHOME/share/recovery.conf.sample $PGDATA/recovery.conf

vi $PGDATA/recovery.conf

standby\_mode = on

primary\_conninfo = 'host=10.253.43.19 port=5432 user=rep'

trigger\_file = '/pgdata/9.6/postgresql.trigger.5432'

8)、启动20(standby)，查看服务进程，请注意有一个wal receiver process进程

pg\_ctl start

ps –ef | grep postgres

在19(primary)查看，ps –ef | grep postgres 可以看到多了一个wal sender process

9)、测试流复制

19(primary)上新建一张表，并插入数据，可以在20(standby)上看到同样的表和数据，并且查询

C、pgpoolii安装配置（19,20配置类似）

1)、在19和20上源码安装 ,安装完成后，需要进入pgpoolii源码包的src目录，make && make install安装，主要是安装一些pgpoolii函数和数据库表到postgresql中

2)、配置pgpool.conf

# - pgpool Connection Settings -

listen\_addresses = '\*'

port = 9999

socket\_dir = '/var/runs'

listen\_backlog\_multiplier = 2

serialize\_accept = off

# - pgpool Communication Manager Connection Settings -

pcp\_listen\_addresses = '\*'

pcp\_port = 9898

pcp\_socket\_dir = '/var/runs'

# - Backend Connection Settings -

backend\_hostname0 = '10.253.43.19'

backend\_port0 = 5432

backend\_weight0 = 1

backend\_data\_directory0 = '/pgdata/9.6'

backend\_flag0 = 'ALLOW\_TO\_FAILOVER'

backend\_hostname1 = '10.253.43.20'

backend\_port1 = 5432

backend\_weight1 = 1

backend\_data\_directory1 = '/pgdata/9.6'

backend\_flag1 = 'ALLOW\_TO\_FAILOVER'

# - Authentication -

enable\_pool\_hba = off

pool\_passwd = 'pool\_passwd'

authentication\_timeout = 60

# - SSL Connections -

ssl = off

# - Concurrent session and pool size -

num\_init\_children = 32

max\_pool = 4

# - Life time -

child\_life\_time = 300

child\_max\_connections = 0

connection\_life\_time = 0

client\_idle\_limit =

# - Where to log -

log\_destination = 'syslog'

# - What to log -

log\_line\_prefix = '%t: pid %p: '

log\_connections = on

log\_hostname = on

log\_statement = on

log\_per\_node\_statement = off

log\_standby\_delay = 'none'

# - Syslog specific -

syslog\_facility = 'LOCAL0'

syslog\_ident = 'pgpool'

# FILE LOCATIONS

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

logdir = '/var/runs'

# CONNECTION POOLING

connection\_cache = on

reset\_query\_list = 'ABORT; DISCARD ALL'

# REPLICATION MODE

replication\_mode = off

replicate\_select = off

insert\_lock = on

lobj\_lock\_table = ''

# - Degenerate handling –

replication\_stop\_on\_mismatch = off

failover\_if\_affected\_tuples\_mismatch = off

# LOAD BALANCING MODE

load\_balance\_mode = on

ignore\_leading\_white\_space = on

allow\_sql\_comments = off

# MASTER/SLAVE MODE

master\_slave\_mode = on

master\_slave\_sub\_mode = 'stream'

# - Streaming -

sr\_check\_period = 5

sr\_check\_user = 'rep'

sr\_check\_password = 'Rep!234'

sr\_check\_database = 'postgres'

delay\_threshold = 16000

health\_check\_period = 5

health\_check\_timeout = 20

health\_check\_user = 'postgres

health\_check\_password = 'Postgres!234'

health\_check\_database = 'postgres'

health\_check\_max\_retries = 3

health\_check\_retry\_delay = 1

connect\_timeout = 10000

# FAILOVER AND FAILBACK

failover\_command = '/pgpool/failover\_stream.sh %H'

fail\_over\_on\_backend\_error = on

search\_primary\_node\_timeout = 10

recovery\_timeout = 90

client\_idle\_limit\_in\_recovery = 0

# WATCHDOG

use\_watchdog = on

# - Watchdog communication Settings -

wd\_hostname = '10.253.43.19

wd\_port = 9000

wd\_priority = 1

wd\_ipc\_socket\_dir = '/var/runs'

# - Virtual IP control Setting -

delegate\_IP = '10.253.43.100'

if\_cmd\_path = '/sbin'

#if\_up\_cmd = 'ip addr add $\_IP\_$/24 dev eth0 label eth0:0' #

if\_up\_cmd = 'ifconfig eth0:0 inet $\_IP\_$ netmask 255.255.255.128'

#if\_down\_cmd = 'ip addr del $\_IP\_$/24 dev eth0'

if\_down\_cmd = 'ifconfig eth0:0 dow

arping\_path = '/usr/sbin'

arping\_cmd = 'arping -U $\_IP\_$ -w 1'

# - Lifecheck Setting -

wd\_lifecheck\_method = 'heartbeat'

wd\_interval = 10

# -- heartbeat mode --

wd\_heartbeat\_port = 9694

wd\_heartbeat\_keepalive = 2

wd\_heartbeat\_deadtime = 30

heartbeat\_destination0 = '10.253.43.20'

heartbeat\_destination\_port0 = 9694

heartbeat\_device0 = 'eth0'

# -- query mode --

wd\_life\_point = 3

# - Other pgpool Connection Settings -

other\_pgpool\_hostname0 = '10.253.43.20'

other\_pgpool\_port0 = 9999

other\_wd\_port0 = 9000

# OTHERS

relcache\_expire = 0

relcache\_size = 256

check\_temp\_table = on

check\_unlogged\_table = on

memqcache\_oiddir = '/var/log/pgpool/oiddir

3)、配置pcp.conf（pcp接口，可以查看、管理pgpool的状态，并且可以远程操作pgpool，pcp.conf 用来对pcp相关命令认证的文件）

$ pg\_md5 -u 用户名 –p Enter后接着输入密码，就会生成一个md5密钥，将 用户名: (md5密钥) 写入pcp.conf

# USERID:MD5PASSWD

用户名: (md5密钥)

4)、配置pool\_hba.conf

# IPv4 local connections:

host all all 127.0.0.1/32 trust

host all all ::1/128 trust

host all all 0.0.0.0/0 trust

host all all 10.253.34.0/24 trust

5)、配置好后，启动pgpool

$ pgpool （停止pgpool: pgpool –m fast stop）

主备转换脚本：failover\_stream.sh

#!/bin/sh

# Failover command for streaming replication

# args: $1: new master hostname

new\_master=$1

trigger\_command="$PGHOME/bin/pg\_ctl -D $PGDATA promote -m fast"

# Promote standby database

/usr/bin/ssh -T $new\_master $trigger\_command

exit 0;

D、测试主备转换

1)、在任意一台机器登录postgresql

psql -h 10.253.43.100 -p 9999 –U postgres –d postgres

postgres# show pool\_nodes; //如果成功，会出现primary节点和standby节点相关信息

$ pg\_ctl stop -m fast //停掉primary

$ pg\_controldata | grep cluster

//出现Database cluster state:   in production，且配置的recovery.conf变成了recovery.done

再次进入psql -h 10.253.43.100 -p 9999 postgres postgres，并且show pool\_nodes，结果是standby变成了primary，并且之前的primary为down状态

新建表，插入数据进行测试

2)、恢复down掉的postgresql

$ mv recovery.done recovery.conf

$ pg\_ctl start

$ pcp\_attach\_node -d -U pgpool -h 10.253.43.19 -p 9898 -n 0

#添加down掉的节点0信息

$ psql -h 10.253.43.100 -p 9999 –U postgres –d postgres

postgres=# show pool\_nodes;