Replicación WAL y Streaming en PostgreSQL 9.1

Objetivo

- Implementar mecanismo de replicación basado en una combinación de "log shipping" y "streaming replication" entre dos servidores PostgreSQL 9.1[2]

Requisitos

- Lectura y comprensión de los apuntes [2], [5]
- Comprender el concepto de log de transacciones, técnicas de backup y recuperación.
- Comprender tecnologías implementadas en PG tales como WAL y PITR.
- Que el alumno cuente con conocimientos básicos de sistemas operativos Unix / Linux: file system, network file system, shell scripts, etc.
- Contar con acceso administrativo a dos servidores PG 9.1 instalados sobre Linux Debian amd64 y funcionando, ambos servidores deben estar en el mismo entorno de red que les permita su comunicación.

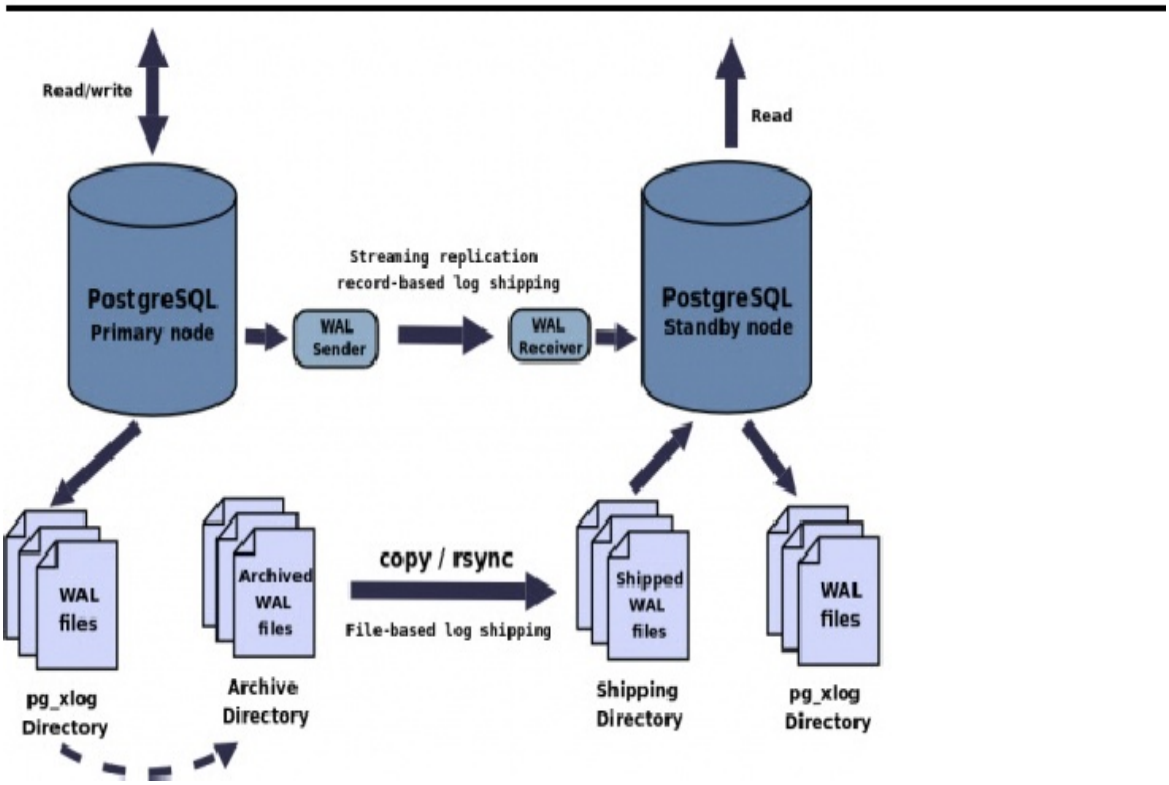
Introducción

Log shipping y streaming son las técnicas más utilizadas en la replicación de base de datos, en especial cuando se trata de una arquitectura de tipo Maestro-Esclavo, en donde los servidores esclavos resolverán consultas de todo tipo para bajar la carga de trabajo del servidor maestro, mientras que éste se dedica al procesamiento de transacciones. Aquí se pretende una implementación de la técnica descripta en la sección "2.3 Warm and Hot Standby Using Point-In-Time Recovery (PITR)" del apunte "Soluciones de Replicación en PostgreSQL 9.1".

Los servidores esclavos se mantendrán actualizados a través del envío de segmentos WAL desde el servidor maestro. A medida que se van completando los segmentos con las transacciones que ha ejecutado el servidor maestro, esta información se transmite a los servidores esclavos, quienes están en modo de recuperación permanente, haciendo REDO de los segmentos WAL que reciben; al mismo tiempo que admiten queries de consulta (esto es lo que se denomina servidor "hot standby"). Es algo así como una actualización "incremental" (y por lo tanto, no tan costosa como transferir toda la base de datos en forma periódica), se transmiten las actualizaciones del servidor maestro a los servidores esclavos. Es una solución asincrónica que también aporta a la alta disponibilidad del sistema, pues, ante la caída del servidor maestro, cualquier esclavo puede ser promovido para convertirse en maestro, pues se asume que todos están prácticamente "al día" (es posible la pérdida de datos, pero, dependiendo como se configure la replicación, del ancho de banda de la red, de la velocidad de transferencia, etc. la pérdida puede ser mínima).

La granularidad de esta replicación es de grano grueso, pues se replica todo el cluster completo del servidor, no es posible utilizar esta técnica para replicar sólo una tabla en particular.

El esquema de replicación puede apreciarse en el siguiente gráfico:



A continuación se detallan los pasos a seguir para realizar esta implementación, cuando se indica tip, se refiere a descripto en [5] y se indica el número correspondiente.

1. Preparación de los Servidores

-Verificar que cumple con todos los requisitos:

-Dos o más servidores con PG 9.1 instalado y funcionando; cada uno con pgAdmin III instalado, contar con usuario Linux postgres administrador PG y usuario root administrador Linux; todos los servidores deben ser del mismo tipo (32bits o 64bits, recomendado 64bits) y con la misma versión de PG instalado.

-Hacer backup de los archivos de configuración PG indicados en tip 3

-Hacer tip 18, instalación de ssh y rsync en cada uno de los servidores involucrados (tanto servidor maestro como esclavos). Se debe configurar ssh debido a que la transmisión del WAL debe hacerse a través de ssh, de forma segura, entre ambos servidores, sin requerir ninguna contraseña (ya que el envío de los segmentos WAL se hace a través de la ejecución de un simple comando o script y se asume que el mismo es un proceso batch, no interactivo, en donde no se espera una interacción en cuanto al tipeo o envío de una contraseña).

2. Configuración de WAL shipping

-Crear directorio de archive[1] en servidor esclavo, usando el usuario postgres:

```
$ mkdir -p /var/lib/postgresql/wal_archive/
```

verificar que no exista previamente y que el mismo se encuentre vacío.

-Activar el almacenamiento WAL y setear el comando de archive de Wal que se indica en parámetro archive_command para permitir el envío de los archivos WAL hacia el o los servidores esclavos. En servidor maestro, editar archivo de configuración del servidor, con usuario root, hacer:

$ nano /etc/postgresql/9.1/main/postgresql.conf

modificar los siguientes parámetros con los siguientes valores:

```
listen_addresses = '*'
wal_level = hot_standby
max_wal_senders = 3
checkpoint_segments = 8 # [3]
wal_keep_segments = 8  #servidor carga liviana, sino aumentar [3]
archive_mode = on
archive_command = 'rsync -av %p postgres@<<IP SERVIDOR ESCLAVO>>:
/var/lib/postgresql/wal_archive/%f'
archive_timeout = 60
```

donde <<IP SERVIDOR ESCLAVO>> reemplazar por la IP correspondiente[2]; %p es el path al segmento WAL que se ha completado y debe ser transferido, este valor es reemplazado automáticamente por PG; %f es el nombre del segmento a transferir y funciona de igual forma que %p.

-Re-arrancar el servidor PG maestro, con usuario root, para que tome los cambios realizados en la configuración:

```
$ /etc/init.d/postgresql restart
```

previamente, asegúrese de que el servidor esclavo esta ejecutándose .

-Verificar que los segmentos WAL están siendo enviados al servidor esclavo, con usuario postgres, en servidor maestro, hacer:

1 Directorio en donde se recepcionarán los segmentos WAL enviados desde el servidor maestro.
2 En caso de tratarse de varios servidores esclavos, se puede implementar un script y reemplazar el comando por la ejecución del script. Otra opción es utilizar el signo “;” para indicar más de un comando rsync.

```
$ tail -f /var/log/postgresql/postgresql-9.1-main.log
```

es probable que tenga que esperar varios minutos hasta ver algo como esto:

```
...
2014-08-26 02:01:28 ART LOG:  el sistema de bases de datos est? listo para
aceptar conexiones
2014-08-26 02:01:28 ART LOG:  el paquete de inicio est? incompleto
sending incremental file list
000000010000000000000000F

sent 16779372 bytes  received 31 bytes  11186268.67 bytes/sec
total size is 16777216  speedup is 1.00
...
```

luego verificar en servidor esclavo, en directorio wal_archive creado previamente, la recepción del archivo WAL de 16MB enviado desde servidor maestro.

-Realizar un “base backup” del servidor maestro, para clonar el servidor maestro en el servidor esclavo. Esto sirve como “punto de partida” a partir del cual comenzar la replicación. Este procedimiento se indica en tip 17, hacerlo con usuario postgres, en servidor maestro (en este caso, comprimimos el backup del cluster utilizando tar):

```
$ cd ~
$ psql -c "select pg_start_backup('base_backup');"

 pg_start_backup
-----------------
 0/18000020
(1 fila)

$ tar -czvf base_backup.tar.gz /var/lib/postgresql/9.1/main/
```

tar lista los nombres de los archivos de ../main comprimidos en base_backup.tar.gz
la siguiente instrucción puede tardar unos mnutos:

```
$ psql -c "select pg_stop_backup();"

 pg_stop_backup
----------------
 0/19000048
(1 fila)
```

transferir el backup comprimido del cluster maestro al servidor esclavo usando rsync, desde servidor maestro con usuario postgres, hacer:

```
$ rsync -av base_backup.tar.gz postgres@<<IP SERVIDOR ESCLAVO>>:
~/base_backup.tar.gz
```

```
sending incremental file list
base_backup.tar.gz

sent 17009297 bytes  received 31 bytes  11339552.00 bytes/sec
total size is 17007115  speedup is 1.00
```

-Detener servidor esclavo, con usuario root, en maquina esclava, hacer:

```
$ /etc/init.d/postgresql stop
```

-Extraer el backup transferido sobre el directorio de datos del cluster del servidor esclavo, en servidor esclavo, usando usuario postgres, hacer:

```
$ cd ~
$ rm -rf ~/base_backup      (# en caso de que ya exista)
$ mkdir -p ~/base_backup
$ cd ~/base_backup
$ tar -xvf ~/base_backup.tar.gz
```

ahora utilizamos rsync para copiar el cluster recibido en servidor esclavo:

```
$ rsync -av ~/base_backup/var/lib/postgresql/9.1/main/
/var/lib/postgresql/9.1/main/
```

rsync lista los archivos copiados y emite error al final, el cual debe ser ignorado:

```
pg_xlog/archive_status/000000010000000000000016.ready
pg_xlog/archive_status/000000010000000000000017.ready
pg_xlog/archive_status/000000010000000000000018.ready

sent 249752275 bytes  received 35540 bytes  29386801.76 bytes/sec
total size is 249623311  speedup is 1.00
rsync error: some files/attrs were not transferred (see previous errors)
(code 23) at main.c(1070) [sender=3.0.9]
```

Ignorar el error de rsync "rsync error: some files/attrs were not transferred ..."

-Borrar el backup comprimido previamente transferido:

```
$ rm -r ~/base_backup/
$ rm ~/base_backup.tar.gz
```

3. Configuración del servidor esclavo

-El servidor esclavo se configura como servidor “hot standby”, en servidor esclavo, con usuario postgres, hacer:

```
$ nano /etc/postgresql/9.1/main/postgresql.conf
```

cambiar los siguientes parámetros con estos valores:

```
wal_level = hot_standby
hot_standby = on
```

-Crear archivo recovery.conf en servidor esclavo para indicarle a PG que opere como servidor standby , en servidor esclavo con usuario postgres, hacer:

```
$ cd /var/lib/postgresql/9.1/main
$ nano recovery.conf
```

crear el archivo recovery.conf y tipear lo siguiente:

```
standby_mode = 'on'

# Specifies a connection string which is used for the standby server to
connect
# with the primary.
primary_conninfo = 'host=<<IP SERVIDOR MAESTRO>> port=5432 user=postgres'

# Specifies a trigger file whose presence should cause streaming
replication to
# end (i.e., failover).
trigger_file = '/tmp/postgres-failover.trigger'

# Specifies a command to load archive segments from the WAL archive. If
# wal_keep_segments is a high enough number to retain the WAL segments
# required for the standby server, this may not be necessary. But
# a large workload can cause segments to be recycled before the standby
# is fully synchronized, requiring you to start again from a new base
backup.
restore_command = 'cp /var/lib/postgresql/wal_archive/%f \"%p\"'

archive_cleanup_command = '/usr/lib/postgresql/9.1/bin/pg_archivecleanup
/var/lib/postgresql/wal_archive/ %r'
```

reemplazar <<IP SERVIDOR MAESTRO>> por la IP que corresponda. Verificar que el contenido del archivo es correcto:

```
$ cat recovery.conf
```

-Permitir que el servidor esclavo se pueda conectarse al servidor maestro, en servidor maestro, utilizando usuario root, modificar el archivo de configuración:

```
$ nano /etc/postgresql/9.1/main/postgresql.conf
```

modificar los siguientes parámetros con los valores indicados:

```
listen_addresses = '<<IP SERVIDOR MAESTRO>>, localhost'
```

-Modificar archivo de configuración cliente en servidor maestro, para permitir el envío de replicación desde servidor maestro, agregar linea en la zona del archivo indicada para configurar las transferencias de replicación; en servidor maestro, con usuario root, hacer:

```
$ nano /etc/postgresql/9.1/main/pg_hba.conf
```

agregar la siguiente linea, una por cada servidor esclavo al cual hay que replicar:

```
host    replication    postgres        <<IP SERVIDOR ESCLAVO>>/32        trust
```

4. Poner en marcha la replicación

-Re-arrancar servidor maestro para que tome los cambios en la configuración; en servidor maestro, con usuario root, hacer:

```
$ /etc/init.d/postgresql restart
```

-Iniciar servidor esclavo; en servidor esclavo, con usuario root, hacer:

```
$ /etc/init.d/postgresql start
```

5. Verificar la replicación

-Chequear que la replicación funciona. En servidor esclavo, con usuario postgres, verificar que no haya errores:

```
$ tail -f /var/log/postgresql/postgresql-9.1-main.log
```

-Chequear que los procesos que envían y reciben segmentos WAL están funcionando; en servidor maestro, con usuario root, hacer:

```
$ ps -ef | grep sender
```

Se debería ver algo similar a esto:

```
postgres   8114   8080   0 00:39 ?           00:00:00 postgres: wal sender
process postgres <<IP SERVIDOR ESCLAVO>>(59341) streaming 0/28000000
```

-En pgAdmin III hacer cambios en la base de datos maestro y verificar la replicación del cambio en servidor esclavo.

6. Promover esclavo

En caso de fallo en el envío o recepción del streaming proveniente del servidor maestro o caída del servidor maestro, se debe elegir un servidor esclavo para promoverlo como nuevo servidor maestro:

-En servidor esclavo, usando usuario postgres, abrir una nueva terminal y hacer:

```
$ tail -f /var/log/postgresql/postgresql-9.1-main.log
```

para ver cualquier cambio que se produzca en el log del sistema.

-En servidor esclavo, usando usuario postgres, en otra terminal distinta a la anterior, hacer:

```
$ touch /tmp/postgres-failover.trigger
```

esta acción alertará al servidor esclavo para detener la recepción del streaming y pasar a un modo de operación normal, si verificamos el directorio `/tmp` deberíamos ver que el archivo `postgres-failover.trigger` fue borrado y que han aparecido nuevos mensajes en el log del sistema, tomando nota de esta situación. Por último, si verificamos en `/var/lib/postgresql/9.1/main` el archivo `recovery.conf` fue renombrado a `recovery.done`.

-Se deberá parar la ejecución de los otros servidores esclavos y configurarlos para que trabajen con el nuevo servidor maestro (repetir pasos anteriores de este tutorial).

-Se deberá configurar el nuevo servidor maestro para el envío de los archivos WAL a los servidores esclavos (repetir pasos anteriores de este tutorial).

-Se deberá limpiar el directorio de archive utilizado en el nuevo servidor maestro (que antes era esclavo), con usuario postgres, ahora, sobre el servidor que será el nuevo maestro, hacer:

```
rm -rf ~/wal_archive
rm /var/lib/postgresql/9.1/main/recovery.done
```

Referencias

[1] "PostgreSQL 9.1 Manual, Chapter 25. High Availability, Load Balancing, and Replication", disponible en http://www.postgresql.org/docs/9.1/static/high-availability.html
[2] Cherencio, G., "Soluciones de Replicación en PostgreSQL 9.1", apunte de asignatura 11078 Base de Datos II, UNLu, disponible en http://www.grch.com.ar/docs/bdd/apuntes/unidad.iii/11078-Soluciones%20de%20Replicacion.pdf
[3] "Binary Replication Tutorial", disponible en https://wiki.postgresql.org/wiki/Binary_Replication_Tutorial
[4] "PostgreSQL Streaming Replication (on Ubuntu 10.04)", disponible en http://technology.trapeze.com/journal/postgresql-streaming-replication-ubuntu-1004
[5] Cherencio, G. "Procedimientos habituales y tips en PostgreSQL 9.1", apunte de asignatura 11078 Base de Datos II, UNLu, disponible en http://www.grch.com.ar/docs/bdd/apuntes/unidad.iii/11078-Procedimientos%20y%20tips.pdf

Atte. Guillermo Cherencio
11078 Base de Datos II
11077 Base de Datos I
División Sistemas
Departamento de Ciencias Básicas
UNLu