PostgreSQL - Master ReplicaServer Anleitung

Version	Date	Information	Who
0.10.000	2025-03-23	Dokument erstellt mit Testinstallation	Alexander Goerisch

Information

Diese Anleitung ist ganz einfach gehalten, mit dieser Anleitung kannst du unter Ubuntu Server 24.04.2 LTS und mit PostgreSQL 17 (letzte Version bei der Dokumentenerstellung) einen Master- und Replica-PostgreSQL-Server erstellen. Das bedeutet alles, was auf dem Master geschrieben wird, landet auch auf dem Replicaserver, von dem die Daten lesend genutzt werden koennen. Man kann naetuerlich auch mehrere Replicaserver erstellen. Das haette unter anderem den Vorteil, dass der Master von den lesenden Zugriffen entlastet wird und auf diesem nur geschrieben wird. In den meisten Faellen werden auch mehr Daten gelesen als geschrieben.

Basic Installation

Zuerst muss man auf zwei PC's oder in einem Hypervisor (Vmware, Proxmox, Xen) zwei Ubuntu Server 24.04.2 LTS Server installieren. In meinem Fall mache ich das auf meiner Testumgebung, die ich mit Proxmox betreibe.

Ich lege zwei VMs an mit je 4GB Ram, 4 vCPUs und 60 GB Storge. Die Dimension der Systeme ist von der Menge an Daten in der Datenbank abhaengig. Es gibt Test-Datenbanken im Internet, die auch ueber 60 GB grosz sind, hier muesste man sowohl RAM als auch den Storaespace entsprechend anpassen.

Ich nenne die beiden Server *psqlc1a* und *psqlc1b* und weise ihnen das ISO-Image fuer den Ubuntu Server 2024.02 LTS zu (https://ubuntu.com/download/server).

Die beiden Server bekommen folgende IP-Adressen zugewiesen:

- psqlc1a - 192.168.42.51
- psqlc1b - 192.168.42.52

Bei der Installation vom Server aktiviere ich zum *Ubuntu Server* noch *Search for third-party drivers* . Bei der Netzwerkkonfiguration gebe ich die Konfiguration selbst ein. Hier ist bei Subnet zu beachten, das die Angabe entsprechen `192.168.42.0/24` angegeben werden muss, natuerlich dein eigene Subnetz, das du verwendest. Rest sollte selbsterklaerend sein bzw. vielleicht noch ein kleiner Hinweis zu den Domainname. Wenn man zu Hause keine eigene Domain hat, aber einen DNS eingerichtet hat, kommt es doch oefter vor das man sich der `.local` Domain bedient. Diese sollte man nicht verwenden, die Definition fuer private Netzwerke lautet `.home.arpa`. Also z. B. `meinnetzwerk.home.arpa`. Das aber nur so am Rande.

Die Disk lasse ich auf Standard und vergroeszere spaeter auf die maximale Groesze des angegebenen Storage.

Bei den Userdaten gebe ich fuer dieses Beispiel *dbadmin* als User an, Servername wurde oben schon angesprochen und das Passwort muss sowieso jeder fuer sich entscheiden.

Dann waehle ich noch *Install OpenSSH server* aus damit ich per SSH auf die Systeme komme. Ist etwas einfacher als per Console von Proxmox. Vor allem kann ich auch Copy & Past von dieser Anleitung nutzen.

Der Rest ist dann per default und am Ende gibt es einen Restart. Da muss ich dann unter Hardware noch das ISO-File aushaengen und Enter druecken bis zum 'Reboot Now'.

Das Spiel mit den zu aendernden Daten (Name des Servers, IP-Adresse) mache ich auch auf dem zweiten Server.

Update und Erweitern der Parition

Ich melde mich an der Stelle schon per SSH an den VMs an:

```
ssh dbadmin@192.168.42.51
ssh dbadmin@192.168.42.52
```

Erstmal die Systeme updaten mit dem folgenden Befehl:

```
sudo apt-get update && sudo apt-get upgrade -y
```

Ich starte nach einem Update das System neu, ist meist nicht notwendig, aber an der Stelle spielt das noch keine Rolle, ob ich den Server neustarte oder nicht. Daher ein `sudo reboot`.

Nun noch die Partition auf ihre maximale Groesze erweitern, sofern du das System so installiert hast wie ich in diesem Beispiel:

```
sudo lvextend -l +100%FREE /dev/ubuntu-vg/ubuntu-lv
sudo resize2fs /dev/mapper/ubuntu--vg-ubuntu--lv
```

Ausgabe sieht dann in etwas so aus, die kanpp 60GB der `/` Partition ist zu sehen:

```
dbadmin@psqlc1b:~$ sudo lvextend -l +100%FREE /dev/ubuntu-vg/ubuntu-lv
[sudo] password for dbadmin:
  Size of logical volume ubuntu-vg/ubuntu-lv changed from <29.00 GiB (7423 extents) to
<58.00 GiB (14847 extents).
  Logical volume ubuntu-vg/ubuntu-lv successfully resized.
dbadmin@psqlc1b:~$ sudo resize2fs /dev/mapper/ubuntu--vg-ubuntu--lv
resize2fs 1.47.0 (5-Feb-2023)
Filesystem at /dev/mapper/ubuntu--vg-ubuntu--lv is mounted on /; on-line resizing required
old_desc_blocks = 4, new_desc_blocks = 8
The filesystem on /dev/mapper/ubuntu--vg-ubuntu--lv is now 15203328 (4k) blocks long.

dbadmin@psqlc1b:~$ df -h
Filesystem                         Size  Used Avail Use% Mounted on
tmpfs                              392M  984K  391M   1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv   57G  6.3G   48G  12% /
tmpfs                              2.0G     0  2.0G   0% /dev/shm
tmpfs                              5.0M     0  5.0M   0% /run/lock
/dev/sda2                          2.0G   96M  1.7G   6% /boot
tmpfs                              392M   12K  392M   1% /run/user/1000
```

Was man jetzt machen kann, ist ein Snapshot, falls man irgendetwas versemmelt, so das man zum Snapshot zurueckspringen kann. Den Snapshot sollte man am Ende auch wieder loeschen.

Installation des PostgreSQL Servers

Die Installation des PostgreSQL Server laeuft wie folgt ab inkl. der Konfiguration der `postgresql.conf` und der `pg_hba.conf`.

```
# Automatische Repository konfiguration:
sudo apt install -y postgresql-common
sudo /usr/share/postgresql-common/pgdg/apt.postgresql.org.sh
# Press ENTER

# Updaten der Paketliste:
sudo apt-get update

# Installieren der letzten Version des PostgreSQL-Servers:
sudo apt -y install postgresql

# Firewall Regel updaten
sudo ufw allow 5432/tcp

sudo vim /etc/postgresql/17/main/postgresql.conf
# Unter folgendem:
#------------------------------------------------------------------------------
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------------

# - Connection Settings -

#listen_addresses = 'localhost'         # what IP address(es) to listen on;

# Folgende Zeile hinzufuegen:
listen_addresses = '*'

sudo vim /etc/postgresql/17/main/pg_hba.conf
# Unter:
# IPv4 local connections:
host    all             all             127.0.0.1/32            scram-sha-256

# Aendern auf:
host    all             all             0.0.0.0/0               scram-sha-256

sudo systemctl restart postgresql
```

PostgreSQL User

Auf beiden Server muss man der `postgres` Superuser Rolle ein Passwort vergeben:

```
# Anmelden als postgres User
sudo -u postgres psql
# Passwort aendern
postgres=# \password postgres
Enter new password for user "postgres":
Enter it again:
exit
```

Tuning `postgresql.conf`

Wer will, kann an dieser Stelle schon die `postgresql.conf` einem grundlegenden Tuning unterziehen. Hier gibt es die Webseite https://pgtune.leopard.in.ua/ wo man seine Daten und Anforderungen angeben kann und einen Vorschlag generieren kann. Der sieht fuer das oben angefuehrte System in etwas so aus wie hier drunter angegeben. Die dort ausgegebenen Werte passen in den meisten Situationen und koennen natuerlich auf die jeweilige Applikation, den jeweiligen Anwendungsfall entsprechend angepasst werden. Feintuning kann nie schaden, man sollte aber Wissen was man tut. Eine Standard Config ist keine gute Idee, weil es keine Rolle spiel, wie viel RAM und CPU main dann reinpfeffert, es wird kaum besser.

Editieren der Datei: `sudo vim /etc/postgresql/17/main/postgresql.conf`

```
# DB Version: 17
# OS Type: linux
# DB Type: mixed
# Total Memory (RAM): 4 GB
# CPUs num: 4
# Connections num: 100
# Data Storage: ssd

max_connections = 100
shared_buffers = 1GB
effective_cache_size = 3GB
maintenance_work_mem = 256MB
checkpoint_completion_target = 0.9
wal_buffers = 16MB
default_statistics_target = 100
random_page_cost = 1.1
effective_io_concurrency = 200
work_mem = 2621kB
huge_pages = off
min_wal_size = 1GB
max_wal_size = 4GB
max_worker_processes = 4
max_parallel_workers_per_gather = 2
max_parallel_workers = 4
max_parallel_maintenance_workers = 2
```

Danach den PostgreSQL Service neustarten:

`sudo systemctl restart postgresql`

Konfigureren von Master und Replica

User anlegen

Den User fuer die Replication muss man sowohl am Master- als auch am Replica-Server anlegen. Username und Passwort kann natuerlich frei gewaehlt werden, muss aber ident sein.

```
sudo -u postgres psql
# Name kann auch anders sein statt replicarole, ich habe sie in pgadmin4 angelegt
postgres=# CREATE ROLE replicarole WITH REPLICATION PASSWORD 'mysecretpassword' LOGIN;
```

Wenn die Rolle erfolgreich angelegt wurde, wird ein `CREATE ROLE` ausgegeben.

Man kann sich die Rolle mit `\du` anzeigen lassen.

```
postgres=# \du
                             List of roles
  Role name  |                         Attributes
-------------+------------------------------------------------------------
 postgres    | Superuser, Create role, Create DB, Replication, Bypass RLS
 replicarole | Replication

 # Aussteigen
 \q
```

`pg_hba.conf` bearbeiten

Auf dem Master den Replica-Server eintragen. Wenn es mehrere geben sollte, dann sind die einfach genauso hinzuzufuegen, sprich in der Zeile darunter:

```
sudo vim /etc/postgresql/17/main/pg_hba.conf
# Suche nach replication (/ fuer die Suche), USER und ADDRESS eintragen
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication     all                                     peer
host    replication     replicarole     192.168.42.52/24        md5
...

# PostgreSQL neustarten
sudo systemctl restart postgresql
```

Replica Konfiguration loeschen und neu syncen

Am Replica muss nun die Konfiguration geloescht werden, diese wird danach vom Master gesynct. Um zu sehen wo die Daten liegen kann man den folgenden Befehl in postgresql nutzen: `SHOW data_directory;`

Loeschen des `main`-Verzeichnisses mit dem folgenden Befehl bzw. neu anlegen und berechtigen:

```
sudo -u postgres rm -r /var/lib/postgresql/17/main
sudo -u postgres mkdir /var/lib/postgresql/17/main
sudo -u postgres chmod 700 /var/lib/postgresql/17/main
```

Nun Backupen wir vom Primary/Master zum Secondary/Replica, am Secondary/Replica. Das Passwort ist das PW vom `replicarole`:

```
sudo -u postgres pg_basebackup -h 192.168.42.51 -p 5432 -U replicarole -D
/var/lib/postgresql/17/main/ -Fp -Xs -R

# Service Restart
sudo systemctl restart postgresql
```

Test

Vom Master aus Testen:

```
dbadmin@psqlc1a:~$ sudo -u postgres psql
psql (17.4 (Ubuntu 17.4-1.pgdg24.04+2))
Type "help" for help.

postgres=# SELECT client_addr, state FROM pg_stat_replication;
  client_addr  |   state
---------------+-----------
 192.168.42.52 | streaming
(1 row)
```

Wenn man jetzt eine Datenbank anlegt, wird diese schneller gesynct als man selbst hin und her klicken oder switchen kann und einen refresh machen.

Nicht vergessen, ggf. die Snapshots loeschen.