Utiliser Barman pour sauvegarder la base PostgreSQL d'un Gitlab Omnibus

Barman est un super logiciel de sauvegarde d'un *cluster* PostgreSQL au fil de l'eau.

Attention : ça ne sauvegarde pas les bases de données une à une, ça sauvegarde **tout** le *cluster* PostgreSQL. C'est un peu embêtant de devoir remonter un *cluster* entier pour récupérer une base ou juste quelques données mais comme c'est un outil surpuissant qui permet de récupérer ses données à la milliseconde près, il est facile de passer outre cet inconvénient.

Pour le côté « au fil de l'eau », ça veut dire que les modifications sont répliquées du *cluster* PostgreSQL à Barman en temps quasi réél par le biais des WAL.

Il est fort simple de mettre en place la sauvegarde d'un *cluster* PostgreSQL par Barman. Je vous laisse lire la documentation officielle.

Ce tutoriel vise le cas particulier de la sauvegarde du *cluster* PostgreSQL d'un serveur Gitlab installé via les paquets Omnibus. Avec cette méthode d'installation, c'est Gitlab qui installe sa version de PostgreSQL, à l'endroit qu'il a choisi, et qui le configure. Toute modification directe des fichiers de configuration de PostgreSQL serait supprimée à la mise à jour suivante. Ma méthode configure proprement PostgreSQL de façon à conserver les modifications par-delà les mises à jour.

Création des utilisateurs

Pas d'utilisateur `postgres` pour Gitlab, mais `gitlab-psql`, et les chemins habituels des outils ont changé.

On se logue :

```
su gitlab-psql -s /bin/bash
```

Et on crée les utilisateurs :

```
/opt/gitlab/embedded/bin/createuser -h /var/opt/gitlab/postgresql/ -s -P barman
/opt/gitlab/embedded/bin/createuser -h /var/opt/gitlab/postgresql/ -P --replication
streaming_barman
```

Modification de la configuration

Il faut modifier le fichier `/etc/gitlab/gitlab.rb` pour que Gitlab configure PostgreSQL pour nous.

De façon un peu bête, dès qu'on fait écouter PostgreSQL sur une interface réseau, Gitlab n'essaye plus de se connecter en *socket* unix mais par le réseau… donc on va le forcer à utiliser la *socket* :

```
gitlab_rails['db_host'] = "/var/opt/gitlab/postgresql/"
```

Ensuite, c'est l'équivalent de la documentation officielle de Barman :

```
postgresql['listen_address'] = '0.0.0.0'
postgresql['wal_level'] = "replica"
postgresql['max_wal_senders'] = 3
postgresql['max_replication_slots'] = 3
```

À l'exception de la façon de créer des entrées dans

```
postgresql['custom_pg_hba_entries'] = {
  'barman': [{
    type: 'hostssl',
    database: 'all',
    user: 'barman',
    cidr: '203.0.113.42/32',
    method: 'md5'
  }],
  'streaming_barman': [{
    type: 'hostssl',
```

```
    database: 'replication',
    user: 'streaming_barman',
    cidr: '203.0.113.42/32',
    method: 'md5'
  }]
}
```

Puis il suffit de lancer la commande suivante pour que Gitlab reconfigure PostgreSQL (et tout le reste de Gitlab, mais ce n'est pas ce qui nous intéresse) :

```
gitlab-ctl reconfigure
```

Révision #5
Créé 2020-03-11 14:57:33 CET par Luc
Mis à jour 2024-05-13 15:13:11 CEST par Luc