Dépôts de sauvegarde multiples avec pgBackRest

pgSession 14

Stefan FERCOT

17 novembre 2021

Qui suis-je ?

- Stefan Fercot
- aka. pgstef
- https://pgstef.github.io
- utilise PostgreSQL depuis 2010
- fan et contributeur de pgBackRest
- Database Backup Architect @EDB

Agenda

- bref rappel des bases
- description de la fonctionnalité
  - commande par commande
  - impact de l’archivage asynchrone

pgBackRest

- outils de gestion de sauvegardes et restaurations
- version actuelle : 2.36 (1er novembre 2021)
- opère en local ou à distance (via SSH ou *bientôt* via serveur TLS)
- multi-processus et opérations asynchrones !
- support stockage S3, Azure et GCP
- …

Chiffrement

- chiffrement côté client

```
repo1-cipher-pass=xxx
repo1-cipher-type=aes-256-cbc
```

- pour générer une passphrase aléatoire :

```
$ openssl rand -base64 48
```

Compression des fichiers

- `compress-type`
  - none
  - bz2
  - gz (par défaut)
  - lz4
  - zst

Installation

- *Use the PGDG repository, Luke!*
  - yum / dnf / apt-get install pgbackrest

Configuration

- `/etc/pgbackrest.conf`, exemple :

```
[global]
repo1-path=/var/lib/pgsql/14/backups
repo1-retention-full=1
log-level-console=info

[ma_stanza]
pg1-path=/var/lib/pgsql/14/data
```

- configuration principale dans la partie `[global]`
- chaque cluster PostgreSQL à sauvegarder a sa propre configuration, appelée `stanza`

Mise en place - archivage

```
# postgresql.conf
archive_mode = on
archive_command = 'pgbackrest --stanza=ma_stanza --log-level-console=debug archive-push %p'
```

Initialisation

```
$ pgbackrest --stanza=ma_stanza stanza-create
P00   INFO: stanza-create command begin 2.36: ...
P00   INFO: stanza-create for stanza 'ma_stanza' on repo1
P00   INFO: stanza-create command end: completed successfully

$ pgbackrest --stanza=ma_stanza check
P00   INFO: check command begin 2.36: ...
P00   INFO: check repo1 configuration (primary)
P00   INFO: check repo1 archive for WAL (primary)
P00   INFO: WAL segment 000000010000000000000001 successfully archived to '...' on repo1
P00   INFO: check command end: completed successfully
```

Sauvegarde complète

```
$ pgbackrest --stanza=ma_stanza --type=full backup
P00   INFO: backup command begin 2.36: ...
P00   INFO: execute non-exclusive pg_start_backup():
backup begins after the next regular checkpoint completes
P00   INFO: backup start archive = 000000010000000000000003, lsn = 0/3000028
P00   INFO: execute non-exclusive pg_stop_backup() and wait for all WAL segments to archive
P00   INFO: backup stop archive = 000000010000000000000003, lsn = 0/3000138
P00   INFO: check archive for segment(s) 000000010000000000000003:000000010000000000000003
P00   INFO: new backup label = 20211102-072629F
P00   INFO: full backup size = 25MB, file total = 951
P00   INFO: backup command end: completed successfully

P00   INFO: expire command begin 2.36: ...
P00   INFO: repo1: 14-1 remove archive,
start = 000000010000000000000001, stop = 000000010000000000000002
P00   INFO: expire command end: completed successfully
```

Types de sauvegarde

- `full`
  - copie tous les fichiers de l'instance
  - sauvegarde indépendante
- `incr`
  - incrémentale depuis la dernière sauvegarde réussie
- `diff`
  - comme `incr` mais toujours basée sur la dernière sauvegarde **full**

Dépôts de sauvegardes multiples

- fonctionnalité introduite en 2.33 (5 avril 2021)
  - redondance
  - différents paramètres de rétention
  - …

```
# exemple
repo1-path=.../repo1
repo1-retention-full=2
repo2-path=.../repo2
repo2-retention-full=1
```

Option `--repo`

- rétrocompatibilité
  - pas requis quand seulement `repo1` est configuré
- lorsqu'un seul dépôt est configuré
  - il est recommandé d'utiliser `repo1`

Commande `stanza-create`

- s’applique directement sur tous les dépôts configurés

```
$ pgbackrest --stanza=ma_stanza stanza-create
P00   INFO: stanza-create command begin 2.36: ...
P00   INFO: stanza-create for stanza 'ma_stanza' on repo1
P00   INFO: stanza-create for stanza 'ma_stanza' on repo2
P00   INFO: stanza-create command end: completed successfully
```

Commande `check`

- déclenche l’archivage d’un nouveau segment WAL
- essaie de le pousser vers tous les dépôts existants

```
$ pgbackrest --stanza=ma_stanza check
P00   INFO: check command begin 2.36: ...
P00   INFO: check repo1 configuration (primary)
P00   INFO: check repo2 configuration (primary)
P00   INFO: check repo1 archive for WAL (primary)
P00   INFO: WAL segment ... successfully archived to '...' on repo1
P00   INFO: check repo2 archive for WAL (primary)
P00   INFO: WAL segment ... successfully archived to '...' on repo2
P00   INFO: check command end: completed successfully
```

Commande `archive-push`

- essaie de pousser les archives WAL vers tous les dépôts accessibles
  - une erreur empêche PostgreSQL de supprimer/recycler le segment WAL !
  - `archive-async=y` apporte la tolérance aux pannes

```
P00   DEBUG:     storage/storage::storageNewWrite: => {
  type: posix, name: {".../repo1/archive/ma_stanza/14-1/0000000100000000/
              000000010000000000000005-ec8283cdf138c7a6052284686bfbf6c01532c2ce.gz"},
...
P00   DEBUG:     storage/storage::storageNewWrite: => {
  type: posix, name: {".../repo2/archive/ma_stanza/14-1/0000000100000000/
              000000010000000000000005-ec8283cdf138c7a6052284686bfbf6c01532c2ce.gz"},
...
P00    INFO: pushed WAL file '000000010000000000000005' to the archive
```

Archivage asynchrone

- `archive-async=y`
  - données temporaires stockées dans le `spool-path`
  - archivage anticipé en utilisant plusieurs processus (`process-max`)
- `archive-push-queue-max`
  - taille maximale de la file d'attente d'archivage de PostgreSQL
  - évite le remplissage d'espace disque (WAL) et que PostgreSQL finisse par s'éteindre complètement…
  - … mais génère des **archives manquantes** !
- très important de surveiller l'archivage pour s'assurer qu'il fonctionne correctement

Sauvegardes

- planifiées individuellement pour chaque dépôt
- sans `--repo`, utilisation d'un dépôt par ordre de priorité
  - (`repo1` > `repo2` > ...)

```
$ pgbackrest backup --stanza=ma_stanza --type=full
P00   INFO: backup command begin 2.36: ...
P00   INFO: repo option not specified, defaulting to repo1
P00   INFO: execute non-exclusive pg_start_backup():
 backup begins after the next regular checkpoint completes
P00   INFO: backup start archive = 000000010000000000000007, lsn = 0/7000028
P00   INFO: execute non-exclusive pg_stop_backup() and wait for all WAL segments to archive
P00   INFO: backup stop archive = 000000010000000000000007, lsn = 0/7000138
P00   INFO: check archive for segment(s) 000000010000000000000007:000000010000000000000007
P00   INFO: new backup label = 20211102-073323F
P00   INFO: full backup size = 25MB, file total = 951
P00   INFO: backup command end: completed successfully
```

Affichage d'informations

- trie par défaut les sauvegardes par dates, mélangeant les dépôts
  - peut être difficile de trouver les sauvegardes interdépendantes

```
$ pgbackrest info --stanza=ma_stanza
stanza: ma_stanza
    status: ok
    cipher: none

    db (current)
        wal archive min/max (14): 000000010000000000000007/000000010000000000000009

        full backup: 20211102-073323F
            timestamp start/stop: 2021-11-02 07:33:23 / 2021-11-02 07:33:32
            wal start/stop: 000000010000000000000007 / 000000010000000000000007
            database size: 25MB, database backup size: 25MB
            repo1: backup set size: 3.2MB, backup size: 3.2MB

        full backup: 20211102-073718F
            timestamp start/stop: 2021-11-02 07:37:18 / 2021-11-02 07:37:26
            wal start/stop: 000000010000000000000009 / 000000010000000000000009
            database size: 25MB, database backup size: 25MB
            repo2: backup set size: 3.2MB, backup size: 3.2MB
```

Affichage d’informations par dépôt

```
$ pgbackrest info --stanza=ma_stanza --repo=2
stanza: ma_stanza
    status: ok
    cipher: none

    db (current)
        wal archive min/max (14): 000000010000000000000009/000000010000000000000009

        full backup: 20211102-073718F
            timestamp start/stop: 2021-11-02 07:37:18 / 2021-11-02 07:37:26
            wal start/stop: 000000010000000000000009 / 000000010000000000000009
            database size: 25MB, database backup size: 25MB
            repo2: backup set size: 3.2MB, backup size: 3.2MB
```

Récupération

```
restore_command = 'pgbackrest --stanza=ma_stanza archive-get %f "%p"'
```

- `archive-get` cherchera dans chaque dépôt par ordre de priorité
  - (`repo1` > `repo2` > …)
- tolérance aux archives manquantes !

Récupération asynchrone des archives

- `archive-get` avec `archive-async=y`
  - récupération anticipée d'archives (`archive-get-queue-max`) pour accélérer la restauration
  - en utilisant plusieurs processus (`process-max`)
  - stockage dans le `spool-path`

Où ?

- site web officiel : https://pgbackrest.org
- guides utilisateurs : https://pgbackrest.org/user-guide.html
- code source : https://github.com/pgbackrest/pgbackrest
- EDB docs : https://www.enterprisedb.com/docs/supported-open-source/pgbackrest

Conclusion

- pgBackRest est un outil complet et fiable
  - avec de nombreuses fonctionnalités et possibilités
- ne pas oublier de tester ses sauvegardes !
  - *Schrödinger's Law of Backups*
  - *l'état de toute sauvegarde est inconnu jusqu'à ce qu'une restauration soit tentée*

Questions ?

Merci pour votre attention !
#pgsession14