Installation de Borg

Ressources

- Borg Backup
- Borgomatic
- Docker Borgomatic
- Tuto mise en place de Borg

Mise en place du dépôt Borg

Créer un utilisateur "backups"

- Se connecter au serveur "bkp-srv" en tant qu'admin : '' ssh admin@
- Passer en root : '' sudo -i ''
- Créer l'utilisateur "backups" sans mot de passe (connexion par clé /home/backups : useradd backups --create-home --home-dir /home/backu shell /bin/bash
- Créer le dossier qui contiendra les infos concernant SSH : '' mkdir 700 /home/backups/.ssh ''
- Créer le fichier qui contiendra les clés SSH autorisées : '' touch /home/backups/.ssh/authorized_keys ; chmod 600 /home/backups/.s
- Attribuer la propriété du dossier et de son contenu à l'utilisateur backups : chown -R backups:backups /home/backups/.ssh ''

Autoriser les machines distantes

Afin d'autoriser les machines accédant à l'utilisateur "backups" à accéder à l'utilis backups, nous allons copier le contenu du fichier /home/admin/.ssh/authorized_keys

- En tant que root sur bkp-srv : cat /home/admin/.ssh/authorized_keys >> /home/backups/.ssh/authorized_keys
- Tester la connexion depuis votre machine locale (aucun mot de pa ssh backups@bkp-<region>-sinp ''

Installer Borg sur "bkp-srv"

Sur l'instance "bkp-srv" hébergeant les dépôts Borg, nous installerons aux dernières versions disponibles rapidement :

- Se connecter au serveur "bkp-srv" en tant qu'admin : '' ssh admin@
- Passer en root : '' sudo -i ''
- En tant que root, installer les paquets système nécessaire :
  - Debian 11/12 avec Pyfuse3 : '' apt install python3 python3-pip

libssl-dev libacl1-dev libacl1 build-essential pkg-config fuse3 l
    - Debian 10 avec llfuse: '' apt install python3 python3-pip python
      dev libacl1-dev libacl1 build-essential pkg-config fuse libfuse-
- En tant que backup (sur root), installer l'environnement virtuel : mkvirtualenv --
  python=python3 borg-env
- Activer l'environnement : '' source borg-env/bin/activate ''
- Installer les dépendances : '' pip install -U pip setuptools wheel ''
- Installer Borg Backup avec le support pour Fuse, si le container u
    - Debian 11/12 avec pyfuse3 : '' pip install borgbackup[pyfuse3]
    - Debian 10 avec llfuse : '' pip install borgbackup[llfuse] ''
- Sortir de l'environnement virtuel : deactivate
- Notes : il peut être intéressant de lier un dossier de l'utilisateur borg pour pouvoir accéder
  dossier accessible uniquement depuis le dépôt "db-srv". C'est le cas des ûc
  sauvegarde des bases de données.

Accéder à Borg sans activer l'environnement virtuel

Pour accéder à Borg sans activer l'environnement virtuel :

- Créer un dossier bin avec mkdir ~/bin '' * Modifier les fichiers : * ~/.profile: ajouter le code ci-dessous au début du fichier .profile afin d'autoriser l'accès aux exécutable du dossier ~/bin lors d'un accès par SSH avec shell interactif : '' vi ~/.profile '' * ~/.bashrc: ajouter le code ci-dessous au début du fichier .bashrc avant la ligne de commentaire # If not running interactively, don't do anything (Shimon, l'accès à la commande borg via ssh avec un shell non interactif (ex. container borgmatic) ne fonctionnera pas ! * Remplacer le code exista de création du dossier bin de l'utilisateur par celui-ci :<code bash Set PATH here to include user's private bin for SSH login # It's necessary for using Borg ! if [-d "$HOME/bin"] ; then PATH="${HOME}bin:${PATH}" fi </code> * Pour root, ajouter le code ci-dessous :<code bash> # Se here to include user's private bin for SSH login # It's necessary fo using Borg ! if [-d "/root/bin"] ; then PATH="/root/bin:${PATH}" fi </code> * Vérif que l'environnement virtuelle est bien désactivé : '' deactivate '' * Recharger l'environnement : '' source ~/.bashrc '' * Ajouter le lien symbolique vers l'executable de Borg : '' ln -s ~/borg-env/bin/borg ~/bin/borg '' * Tester avec la version de Borg : '' borg --version

Mettre jour Borg

- En tant que backup, activer l'environnement : '' source borg-env/bin/acti
- Pour mettre à jour Borg :
    - Debian 11 avec pyfuse3 : '' pip install -U borgbackup[pyfuse3]
    - Debian 10 avec llfuse : '' pip install -U borgbackup[llfuse] ''

Sur les machines à sauvegarder

Préparer la connexion SSH

Le dépôt Borg étant distant (hébergé sur l'instance "bkp-srv"), il est n
pour l'utilisateur qui héberge les containers Docker). La clé publique g
copier ans le ~/.ssh/authorized_keys de l'utilisateur backups de l'instance "bkp-srv" afin
d'autoriser l'accès au dépôt depuis l'instance à sauvegarder.

- Se connecter à l'instance concernée : '' ssh admin@<instance>-<re
- Générer une clé SSH : '' ssh-keygen ''
  - Accepter l'emplacement de stockage de la clef par défaut
  - Laisser une phrase de passe vide pour faciliter l'automatisatio
- Copier la clé publique sur le serveur "bkp-srv" dans le fichier ~/.ssh/authorized_keys de l'utilisateur backups afin de permettre une connexion SSH sans demande de mot d backups n'ayant pas de mot de passe, il faut passer par l'utilisateur
  - Copier de la clé publique : '' scp -P <port-ssh-bkp-srv> ~/.ssh/id_rsa.pub admin@bkp-<region>-sinp:~/.ssh/id_rsa.<instance>-srv.pub ''
  - Se connecter au serveur "bkp-srv" avec l'utilisateur admin : '' ssh admin@bkp-<regi sinp ''
  - Passer en root : '' sudo -i ''
  - Ajouter la clé publique de l'instance à sauvegarder au fichier /home/backups/.ssh/authorized_keys de l'utilisateur backups : '' cat /home/geonat/.ssh/id_rsa.<instance>-srv.pub >> /home/backups/.ssh/authorized_keys ''
  - Tester la connexion SSH depuis un autre shell de l'instance à sauvegarder, aucun mot de passe ne doit être demandé : '' ssh backups@bkp-<region>-sinp ''
  - Si la tentative de connexion s'est déroulé sans demande de mo supprimer le fichier de la clé publique de l'instance : /home/geonat/.ssh/id_rsa.<instance>-srv.pub
- Sur l'instance à sauvegarder et pour l'utilisateur admin, créer un f paramètre maintenant la connexion SSH : '' touch ~/.ssh/config; ch ~/.ssh/config ''
  - Y ajouter le contenu suivant :

```
Host *
    ServerAliveInterval 240
```

- Tester à nouveau la connexion SSH depuis admin sur l'instance à sauvegarder, aucun mot de passe ne doit être demandé : '' ssh -p backups@bkp-<region>-sinp ''
- Créer le dossier qui contiendra les clés SSH qui sera par la suite monté dans le conteneur Docker : '' mkdir -p ~/docker/borgmatic/data/.ssh ''
- Copier le dossier .ssh de l'utilisateur admin dans le dossier de Borgmatic : '' sudo c ~/.ssh/ ~/docker/borgmatic/data/ ''
  - l'associer à l'utilisateur root car c'est l'utilisateur qui l'utilisera dans le contai sudo chown root: -R ~/docker/borgmatic/data/.ssh ''
  - ATTENTION : Le docker Borgmatic se connectera à bkp-srv avec les infos contenues dans le dossier /root/.ssh/borgmatic/data/.ssh

Installer le container de sauvegarde

- Au préalable, mettre à jour les fichiers Docker pour l'instance concernée
- Réaliser l'installation du container Docker Borgmatic

Préparer la sauvegarde des bases de données

- Afin de pouvoir accéder au port 5432 sur l'IP "gateaway" du container, il faut éditer le fichier postgresql.conf :
  - Ajouter l'IP 172.18.0.1 au paramètre // listen_addresses //
- Il faut ensuite configurer l'accès aux bases à sauvegarder depuis le container créé a une IP privée 172.18.0.x, nous ajoutons donc 172.18.0.0/16 qui est un ensemble d'IP privé qui devrait convenir dans la plupart des cas. Si ce n'est pas le cas, il faut obtenu par votre container à l'aide de Portainer par exemple. Pour cela, il faut éditer le fichier : vi pg_hba.conf
  - Ajouter :

```
host    geonature2db    geonatadmin     172.18.5.1/32           scram-sha-256
host    gnatlas         geonatadmin     172.18.5.1/32           scram-sha-256
# Voir s'il n'est pas possible d'utiliser l'utilisateur geonatadmin à la place de postgres...
host    gnatlas         postgres        172.18.5.1/32           trust
host    geonature2db    postgres        172.18.5.1/32           trust
```

- Il faut aussi s'assurer que le service Systemd de Postgresql démarre après Docker si l'on veut que Postgresql écoute sur l'IP 172.18.5.1. Pour cela, il faut éditer le service Postgresql : systemctl edit postgresql@.service
  - Ajouter le contenu suivant :

```
[Unit]
After=docker.service
```

  - Vérifier la présence du fichier : /etc/systemd/system/postgresql@.service.d/override.conf

Initialiser des dépôts

- Générer en local un UUID pour la clé de cryptage des dépôts locaux de l'instance concernée :
  - Stocker cette phrase dans Keepass
- Sur l'instance, en tant qu'utilisateur backups, créer le dossier qui contiendra les backups : sudo -i ; su - backups ; mkdir ~/<instance>-srv
- Se connecter à l'instance concernée : '' ssh admin@<instance>-<region>''
  - Modifier le fichier /docker/borgmatic/.env de l'instance en ajoutant la phrase de passe comme valeur du paramètre BORG_PASSPHRASE : sudo nano /docker/borgmatic/.env

- Accéder au conteneur borgmatic pour initialiser le dépôt de l'instance : '' doc
  borgmatic /bin/bash ''
- Établissez une première connexion au serveur du dépôt distant via
  serveur dist ssh -p <port-ssh-bkp-srv> backups@<bkp-srv-private-ip
- À l'aide de borgmatic, initialiser les dépôts de l'instance (présent dans le
  Borgmatic). Borgmatic se charge d exécuter Borg avec les commandes et les options
  encryption repokey-blake2
    - Si besoin, vous pouvez réinitialiser les dépôts en suivant la pr
      dépôt" décrite ci-dessous.
- Pour voir les infos concernant un dépôt :
    - Dépôt de "db-srv" : borg info ssh://backups@10.0.1.30:<ssh-port-bkp-
      srv>/home/backups/db-srv/
    - Dépôt de "web-srv" : borg info ssh://backups@10.0.1.30:<ssh-port-bkp
      srv>/home/backups/web-srv/
    - Le dépôt local à chaque instance : '' borg info /mnt/borg-repos
- Export les clés des dépôts et les stocker dans Keepass :
    - Instance "db-srv" :
        - Dépôt distant : borg key export ssh://backups@10.0.1.30:<ssh-po
          bkp-srv>/home/backups/db-srv/ /root/.config/borg/db-srv-
          dist.repo-conf-file.txt ; cat /root/.config/borg/db-srv-
          dist.repo-conf-file.txt
        - Dépôt local : '' borg key export /mnt/borg-repository /root/
          local.repo-conf-file.txt ; cat /root/.config/borg/db-srv-local
    - Instance "web-srv" :
        - Dépôt distant : borg key export ssh://backups@10.0.1.30:<ssh-po
          bkp-srv>/home/backups/web-srv/ /root/.config/borg/web-srv
          dist.repo-conf-file.txt ; cat /root/.config/borg/web-srv-
          dist.repo-conf-file.txt
        - Dépôt local : '' borg key export /mnt/borg-repository /root/
          local.repo-conf-file.txt ; cat /root/.config/borg/web-srv-loca
- Si l'instance bkp-srv comprend un espace disque supplémentaire monté sur /data, il peut être
  intéressant de déplacer via un lien symbolique les dépôts Borg cré
    - Créer le dossier qui contiendra les dépôts : mkdir /data/borg-repos
    - Donner les droits à l'utilisateur backups : chown backups: /data/borg-repos
    - Déplacer les dépôts : mv /home/backups/*-srv /data/borg-repos/
    - En tant qu'utilisateur backups : '' ln -s /data/borg-repos/web-srv /home/ba
      ln -s /data/borg-repos/db-srv /home/backups/db-srv''
    - Tester l'accès distant depuis borgmatic : borg info
      ssh://backups@10.0.1.30:<ssh-posrt-bkp-srv>/home/backups/we
- Tester une sauvegarde manuellement pour vérifier que tout fonctio
  verbosity 2 --stats --files

Réinitialiser un dépôt

S'il s'avérait nécessaire de réinitialiser un dépôt, la démarche à suivr

- Vérifier les archives actuellement présentes dans les dépôts : '' bo
- Supprimer un dépôt : '' borgmatic borg delete <repo> ''
    - Ex. : '' borgmatic borg delete /mnt/borg-repository/ ''
    - ATTENTION : normalement la commande doit demander confirm

fonctionne pas essayer la commande sans borgmatic et vérifier le dépôt. La commande devrait demander confirmation de la suppression

- Initialiser à nouveau les dépôts borgmatic : '' borgmatic init --encryption repokey-blake2
- Vérifier à nouveau les dépôt : '' borgmatic list ''
- Récupérer et stocker dans Keepass les nouvelles clés des dépôts ("Exporter les clés des dépôts").

Tester le container de sauvegarde

Accéder au container borgmatic pour vérifier et tester son contenu : '' docker exec -it borgmatic /bin/bash ''

Tester l'envoi des emails

- À l'email adminsys : '' echo "THIS IS A TEST EMAIL sended to adminsys at $(date "+%H:%M")" | mail -s "[${HOSTNAME}] Test email" adminsys@<domain> ''
- À un utilisateur spécifique : '' echo "THIS IS A TEST EMAIL sended to root at $(date "+%H:%M")" | mail -s "[${HOSTNAME}] Test email" root ''

Tester l'envoi de message Telegram

- Envoyer un message : '' telegram-send "Test at `date`" ''

Tester Borgmatic

- Tester les fichiers de configuration de Borgmatic : '' borgmatic config validate '' (anciennement '' validate-borgmatic-config '') * ATTENTION il est maintenant nécessaire d'initialiser les dépôts avant de lancer la commande de test ci-dessous. Voir la section "Initialisation des dépôts". * Tester une sauvegarde manuellement pour vérifier que tout fonctionne : '' borgmatic --verbosity 2 --stats --files

Tester le cron de Borgmatic

- Éditer le fichier /etc/cron.d/borgmatic : '' nano /etc/cron.d/borgmatic ''
- Vérifier l'heure de lancement par défaut. Chaque instance devrait avoir sa sauvegarde. Il vaut mieux éviter de sauvegarder sur les dépôts distants (bkp-srv) au même moment.
- Modifier l'heure de lancement par défaut, pour que la sauvegarde commence dans 2 minutes plus tard
- ATTENTION NE PAS Enregistrer les changements avec Control+O car cela écrase les fichiers présents dans le dossier /etc/cron.d/ et ils ne sont gérés indépendamment. Si vous rajouter le fichier /etc/cron.d/borgmatic2 les 2 processus distincts seront lancés en même temps. Par conséquent, des erreurs liées au lock sur les dépôts.
- Si vous avez correctement configurer l'envoie de message via Telegram, vous recevrez des messages indiquant le démarrage de la sauvegarde à l'heure que vous avez définie.

vous devriez pouvoir voir les logs sur l'interface de Portainer.

Tester le dump Postgresql

- Contexte : tester manuellement le dump lancé par Borgmatic dans est installé sur votre instance).
- Lancer le dump. Ex. :

```
PGPASSWORD=<password-geonatadmin> pg_dump --verbose-no-password --
clean--if-exists--host172.18.5.1--port5432--usernamegeonatadmin
--formatcustom geonature2db>/home/geonature2db
```

- ATTENTION : dans les logs du container, le dump est lancer avec PGPASSWORD qui est non visible !
- Pour réaliser le dump de la base, il faut un utilisateur avec des dr il y a des risques que le dump échoue car l'utilisateur n'aura pas certaines tables ou autres éléments de la base.

Utiliser Telegram

Conû gurer Telegram pour N'utiliser le script telegram-send envoyer des alertes l'aide de Borgmatic. Ntfy nécessite d'autoriser manuellement un bot ap d'un container. Ce n'est pas très pratique. Il vaut mieux utiliser le script

- Installer Telegram sur votre Smartphone et/ou machine locale
- Créer un nouveau Bot d'alerte en envoyant un message au contact
  - Indiquer son nom : <region> SINP Alert Bot
  - Indiquer comment l'appeler : '' <region>SinpAlertBot ''
  - Une fois créé, accéder à la liste de vos bot :
    - Choisir le bot '' <region>SinpAlertBot ''
    - Cliquer sur "API Token" pour en créer un nouveau, le copie l'enregistrer sur Keepass.

Conû gurer Telegram avec le script ''telegram-send''

- Le script telegram-send peut être utilisé pour envoyer des messages dire utilisateur ou à un canal (=group)
- Si vous souhaitez créer un "Canal" pour que le bot y envoie les me
  - Dans l'interface de Telegram, via le menu, choisir "Nouveau ca
    - Indiquer le nom du canal : "<REGION> SINP ALERT"
    - Indiquer la description : "Message d'infos et d'alertes du S
    - Cliquer sur "Créer"
    - Choisir un canal privé de type :
  - Ajouter le bot '' <region>SinpAlertBot '' au canal
    - Le promouvoir en tant qu'admin et lui donner seulement les messages
- Pour récupérer l'ID du canal (=group) et ou simplement l'ID d'un u message envoyé sur le canal ou à l'utilisateur au bot getids

    - Il aû che les infos concernant le message dont les id qui vous i du canal.
        - Aû n que tout le monde reçoive bien les notiû cation, utilise
    - Créer des entrées dans Keepass pour ces infos et renseigner l TELEGRAM_BOT_TOKEN TELEGRAM_GROUP_ID
- Redémarrer le container pour prendre en compte les changements : '' docker-compose down ; docker-compose up -d ''
- Tester l'envoie de message :
    - Se connecter au container : '' docker exec -it borgmatic /bin/bash ''
    - Envoyer un message : '' telegram-send "Test at `date`" ''

þ Conû gurer Telegram avec Ntfy

- Sur l'instance de serveur que vous souhaiter sauvegarder, se connecter : '' docker exec -it borgmatic /bin/bash ''
    - ATTENTION : Cette manipulation est à refaire à chaque nouvelle c container !
    - Lancer la commande : '' ntfy -b telegram send "Telegram conû g
        - Coller le token précédemment copier lors de la création du
            - Copier le mot de passe qui s'aû che
        - Sur Telegram, ajouter votre bot en contact et lui démarrer a
            - Coller le mot de passe précédemment copié généré par
            - Un message de succès devrait s'aû cher : Test successfully conû gured for ntfy
    - Tester l'envoie de message sur votre bot : '' ntfy -b telegram - "Borgmatic: test at `date` on ${HOSTNAME} !" ''
        - Si tout fonctionne correctement, un message devrait être r l'interface de Telegram.

Restaurer une sauvegarde

Préparer la restauration

- Se connecter à l'instance : ssh admin@<instance>-<region>-sinp
- Se placer dans le dossier : cd ~/docker/borgomatic
- Arrêter le container de sauvegarde : docker-compose down
- Sur l'hôte, créer un dossier temporaire qui contiendra les fichiers : rm -fR /tmp/restore ; mkdir /tmp/restore
- Lancer un shell interactif avec le dossier précédemment créé monté à f docker-compose.yml -f docker-compose.restore.yml run -v /tmp/restore:/tmp/restore borgmatic
- Vous pouvez alors utiliser les commandes borg

Récupérer des fichiers et/ou dossiers avec Borg

- Se connecter à l'hôte : ssh admin
- Liste les sauvegardes de l'instance :
    - db-srv : borg list /home/backups/db-srv
    - web-srv : borg list /home/backups/web-srv

- Créer un dossier :
  - pour le point de montage du dépôt : `mkdir /tmp/repo`
  - pour le dossier qui contiendra les fichiers à restaurer : `mkdir /tmp/restore`
- Utiliser Fuse pour monter la sauvegarde sur le dossier point de mount <repo> /tmp/repo
  - Ex. avec le dépôt borgmatic : `borg mount /home/backups/db-srv /tmp/repo`
  - Le dossier /tmp/repo contient les dossiers des différentes sauvegardes sauvegardé le 17 mai 2021 à 13h57 : 2021-05-17T13:57:02
- Copier les fichiers à restaurer sur l'hôte en les copiant depuis le dossier lié l'hôte :
  - Ex. pour web-srv : `cp -r "/tmp/repo/web-srv-2021-05-17T13:57:02/mnt/source/etc/cron.d/" /tmp/restore/$(date +'%Y-%m-%d')_gnatlas.custom`
  - Ex. pour db-srv : `cp -r /tmp/repo/db-srv-2022-04-08T01\:07\:55/root/.borgmatic/postgresql_databases/172.18.0.1/geonature2db /tmp/restore/$(date +'%Y-%m-%d')_geonature2db.custom`
  - Vérifier la présence des fichiers sur l'hôte dans /tmp/restore/
- Donner les droits d'accès à l'utilisateur 1654:1654 (admin) : `chown -R 1654:1654 /tmp/restore/*.custom`
- Démonter le point de montage et quitter le shell borg : `umount /tmp/repo`
- Récupérer sur votre machine locale l'archive depuis votre domaine@bkp-données : `scp <region>-sinp:/tmp/restore/*.custom ./`
  - Voir la restauration en local d'une base serveur

Récupérer des fichiers et/ou dossiers avec Borgmatic

- Se placer dans le dossier partagé avec l'hôte pour avoir accès aux fichiers extraits par défaut une archive ou un chemin particulier dans le dossier co
  - Toutefois, il est possible d'utiliser l'option --destination pour préciser l'emplacem de destination.
- Lister les archives (=sauvegardes) disponibles : '' borgmatic list ''
  - ATTENTION : la commande ci-dessus listes les archives de tou le nom de l'archive correspondant au dépôt que vous indiquere d'extraction.
- Extraire l'archive "<archive-name>" du dépôt <repo> dans le dossi borgmatic extract --repository <repo> --archive "<archive-name>" destination /tmp/restore
  - Ex. : `borgmatic extract --repository /mnt/borg-repository --archiv "web-srv-2021-05-17T13:57:02" --destination /tmp/restore`
- Pour extraire seulement un dossier de l'archive, utiliser l'option --path avec le chemin du dossier sans slash au début. Ex : `borgmatic extract --repository /mnt/borg-repository --archive "web-srv-2021-05-17T13:57:02" --path "mnt/source/etc" --destination /tmp/restore`
- Note concernant la localisation des éléments sauvegardés dans le /tmp/restore/ si le dossier d'extraction est /tmp/restore/ :
  - les fichier ou dossiers systèmes sauvegardés se trouvent dans /tmp/restore/mnt/
  - les sauvegardes des bases de données se trouvent dans /tmp/restore/root/.borgmatic/postgresql_databases/172.18.0.1/
- Quitter le container : `exit`

Écraser les fichiers à restaurer par ceux provenant de la sauvegarde s

- Sur l'hôte, écraser les fichiers à restaurer par ceux provenant de l /tmp/restore/cron.d/ /etc/ ''
  - Assurez vous que les dossiers et fichiers sont restaurés avec l propriétaire
- Une fois les fichiers/dossiers restaurés vous pouvez supprimer le contenu : '' rm -fR /tmp/restore ''
- NOTE : si nécessaire, il est possible de monter directement le volume sous forme de répertoire pour remplacer les fichiers de l'hôte directement depuis le shell interac

Restaurer une base de données avec Borgmatic

- Lister les archives disponibles : '' borgmatic list ''
- Restaurer une base de données directement. ATTENTION : Pour qu nécessaire d'avoir utiliser une super utilisateur dans la config Bor

```
borgmatic restore --repository <repo> --archive <archive> --database
<db-name>
```

  - Ex. :

```
borgmatic restore --repository /mnt/borg-repository --archive
"db-srv-2021-05-17T01:00:02" --database geonature2db
```

  - Si la commande échoue, récupérer dans la console la ligne de supprimer l'option --progress et lancer la commande manuellement. D plus détaillées s'afficheront.
- Quitter le container :

Extraire l'archive de la base et la récupérer localement

- Sur l'hôte, copier sur le docker : /tmp/restore/root/.borgmatic/postgresql_databases/172.18.0.1/geon b "/tmp/$(date +'%Y-%m-%d')_geonature2db.custom"
- Changer les droits : ''sudo chown admin: "/tmp/$(date +'%Y-%m-%d
- Depuis votre machine locale récupérer l'archive : '' scp admin@db +'%Y-%m-%d')_geonature2db.custom ./ ''

Finaliser la restauration

- Penser à relancer Bo service : '' docker compose up -d ''
- Au cas où Borg échoue à créer/acquérir un verrou : '' borg break-lock <

Cas particulier de la base GeoNature

- Commencer par extraire une archive dans le dossier /tmp/restore comme indiqué ci-dessous :
  - Ex. :

```
borgmatic extract --repository ssh://backup@10.0.1.30:2022/./port-ssh-bkp-srv/home/backup/db-srv --archive "db-srv-2021-05-19T16:09:01" --destination /tmp/restore
```

- Supprimer les connexions à la base existante :

```
psql -U postgres -h 172.18.0.1 -d gnatlas -c "SELECT pg_terminate_backend(pg_stat_activity.pid) FROM pg_stat_activity WHERE pg_stat_activity.datname = 'geonature2db' ; "
```

- Supprimer la base :

```
psql -U postgres -h 172.18.0.1 -d gnatlas -c "DROP DATABASE geonature2db;"
```

- Recréer une base vide :

```
psql -U postgres -h 172.18.0.1 -d gnatlas -c "CREATE DATABASE geonature2db;"
```

- Restaurer la base :

```
pg_restore --if-exists --exit-on-error --clean --dbname geonature2db -h 172.18.0.1 -p 5432 --username postgres /tmp/restore/root/.borgmatic/postgresql_databases/172.18.0.1/geonature2db
```

Si nécessaire, il est aussi possible de manipuler les éléments à restaurer :

- Créer la liste des éléments à restaurer :

```
pg_restore --list /tmp/restore/root/.borgmatic/postgresql_databases/172.18.0.1/geonature2db > gn2.list
```

  - Éditer le fichier : '' vi gn2.list ''
  - Vous pouvez par exemple commenter les lignes EXTENSION pour ne pas les restaurer et quitter.
- Restaurer la base avec cette liste :

```
pg_restore --if-exists --exit-on-error --clean --dbname geonature2db -h 172.18.0.1 -p 5432 --username postgres --use-list gn2.list /tmp/restore/root/.borgmatic/postgresql_databases/172.18.0.1/geonature2db
```

Cas particulier de la base GeoNature Atlas

En fonction du format de sauvegarde de la base et de l'utilisateur qui
base de données ne pourra pas se faire avec la commande
contient des vues matérialisées qui posent problèmes car elles sont b
"étrangères" (FDW). Or, les informations du Foreign Data Wrapper ne
restaurées (utilisateur, mot de passe) par
Avec la version 13+ de Postgresql ce comportement devrait être améli
par ne fonctionne pas et cela échoue si l'on ne supprime pas les
de rafraîchissement des VM car les tables FDW ne contiennent pas de
des vues échoue.

- Suivre les étapes précédentes visant à extraire une archive dans l
de Borgmatic.
- Le dossier de la sauvegarde dans la base
/tmp/restore/root/.borgmatic/postgresql_databases/172.18.0.1/gnat
- Si le format a été utilisé dans le fichier de
  - Il est simplement nécessaire d'utiliser

```
psql -U postgres -h 172.18.0.1 -p 5432 -d gnatlas -f
"/tmp/restore/root/.borgmatic/postgresql_databases/172.18.0.1
las"
```

- Si le format a été utilisé dans le fichier
de , nous pouvons
  - Créer un fichier qui contient la liste des requêtes a exécuter p
supprimerons de la liste le rafraichissement des vues :

```
pg_restore --list
/tmp/restore/root/.borgmatic/postgresql_databases/172.18.0.1/gnatl
as | grep -v "MATERIALIZED VIEW DATA" > without_refresh.list
```

  - Lancer la restauration de la base en executant seulement les r
fichier

```
pg_restore --if-exists --exit-on-error --clean --dbname gnatlas
--host 172.18.0.1 --port 5432 --username postgres --use-list
without_refresh.list
/tmp/restore/root/.borgmatic/postgresql_databases/172.18.0.1/gnatl
as
```

  - Après avoir restaurer la base, il est nécessaire de reconfigure
FDW puis de peupler les vues matérialisées avec les données
    - Sur l'instance "db-srv", lancer les requêtes suivantes :
      - Ajouter l'utilisateur distant au mapping de l'utilisateur
la base "gnatlas" :

```
sudo -u postgres psql -d gnatlas -c "ALTER USER
MAPPING FOR geonatadmin SERVER geonaturedbserve
(ADD user 'geonatadmin');"
```

      - Ajouter le mot de passe de l'utilisateur distant :

```
sudo -u postgres psql -d gnatlas -c "ALTER USER
```

```
MAPPING FOR geonatadmin SERVER geonaturedbserve
(ADD password '<pwd>');"
```

- Lancer ensuite le rafraichissement des vues :

```
psql -h localhost -U geonatadmin -d gnatlas -f
/home/geonatdata/atlas/data/sql/02_refresh_mv_data.sql
```

Si besoin, pour voir ce qui existe dans la base concernant FDW:

- Se connecter à Postgres avec l'utilisateur postgres sans mot de passe :

```
sudo -u postgres psql -d gnatlas
```

- Lister les serveurs FDW :

```
SELECT * FROM pg_foreign_server;
```

- Lister le USER MAPPING :

```
SELECT um.*, rolname FROM pg_user_mapping um JOIN pg_roles r ON r.oid =
umuser JOIN pg_foreign_server fs ON fs.oid = umserver;
```

- Sortie : exit

Borg/Borgmatic commandes utiles

- Supprimer le cache :

```
borg delete --cache-only <repo>
```

- Au cas ou Borg échoue à créer/acquérir un verrou :

```
borg break-lock <repo>
```

  - Notes : entrer dans le container Docker Borgmatic et lancer la commande borg break-lock /mnt/borg-repository/ ou borg break-lock ssh://<user>@<ip>:<port>/home/backup/pg_dump_bas (vérifier la valeur exacte dans le fichier de config : vi /etc/borgmatic.d/config.yaml)
  - Alternative avec Borgmatic qui enlève les verrous sur tous les repos :

```
borgmatic break-lock
```

- Relancer une sauvegarde manuellement avec Borgmatic :

```
borgmatic --verbosity 2 --stats --files
```