TP – Mise en place d'un architecteur avec réplication avec MongoDB

Tous d'abord, nous devons créer des répertoires qui vont accueillir les données qui vont être sauvegarder (un répertoire pour chaque serveur). Cela va permette de créer plusieurs Serveurs de réplications.

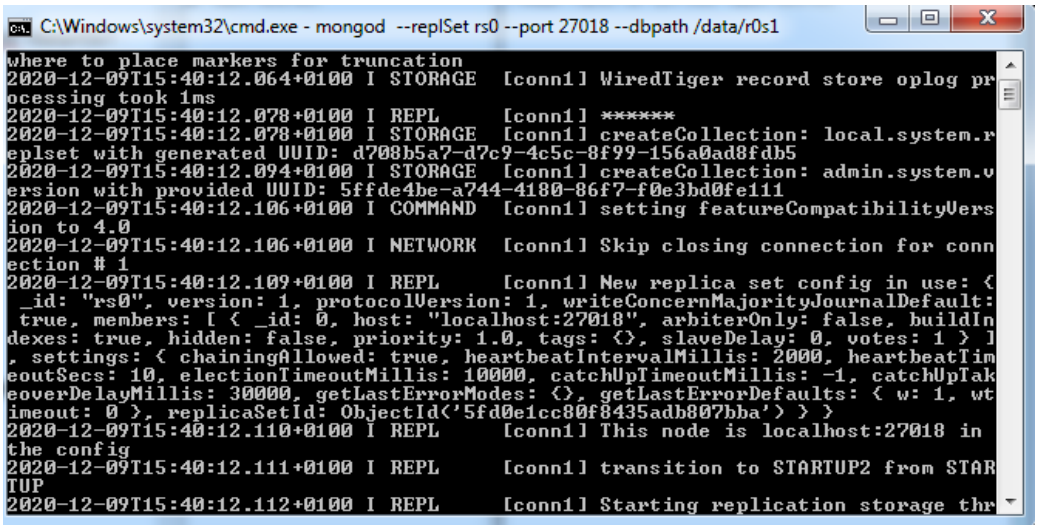
Création des répertoires de sauvegarde pour chaque serveur :

Pour notre TP nous allons créer trois nœuds (ou serveur) nommé de la façon suivante : r0s1 pour réplicatSet 0 serveur 1, r0s2 et r0s3

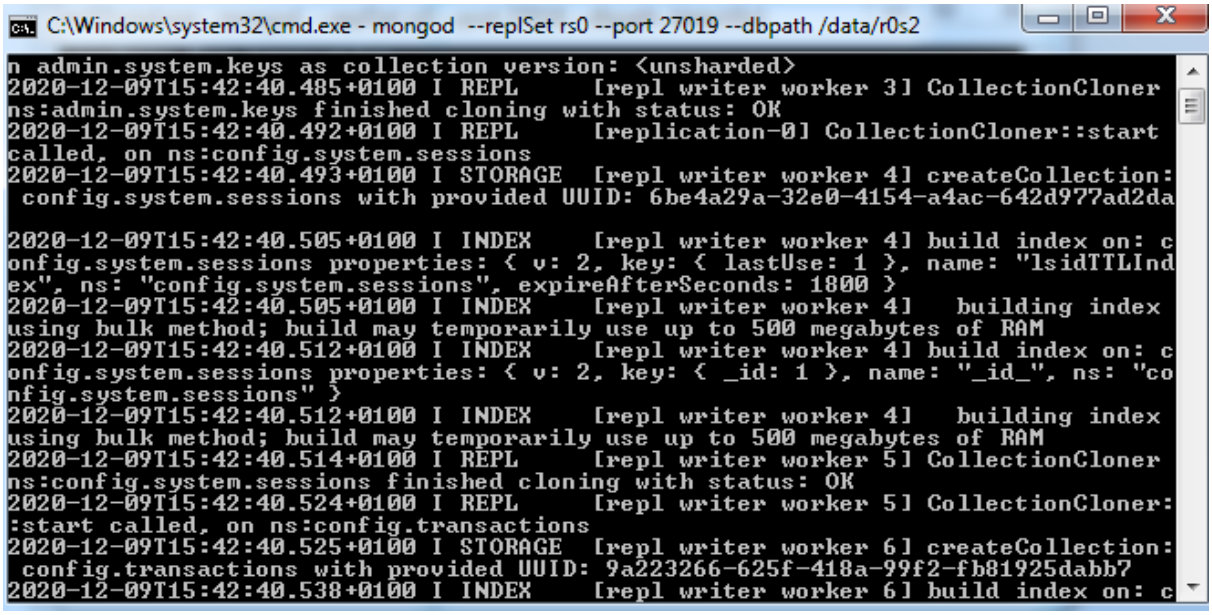
Pour cela je vais créer le répertoire a la racine dans `/data` pour ce faire j'utilise la commande « `mkdir /data/r0s1` »

Nous allons par la suite utiliser `mongod –replSet rs0 –port 27018 –dbpath /data/r0s1`, chaqu'un de c'est serveur va alors devenir des serveurs de stockage en associant un port d'écoute différent (exemple : 21018, 27019).

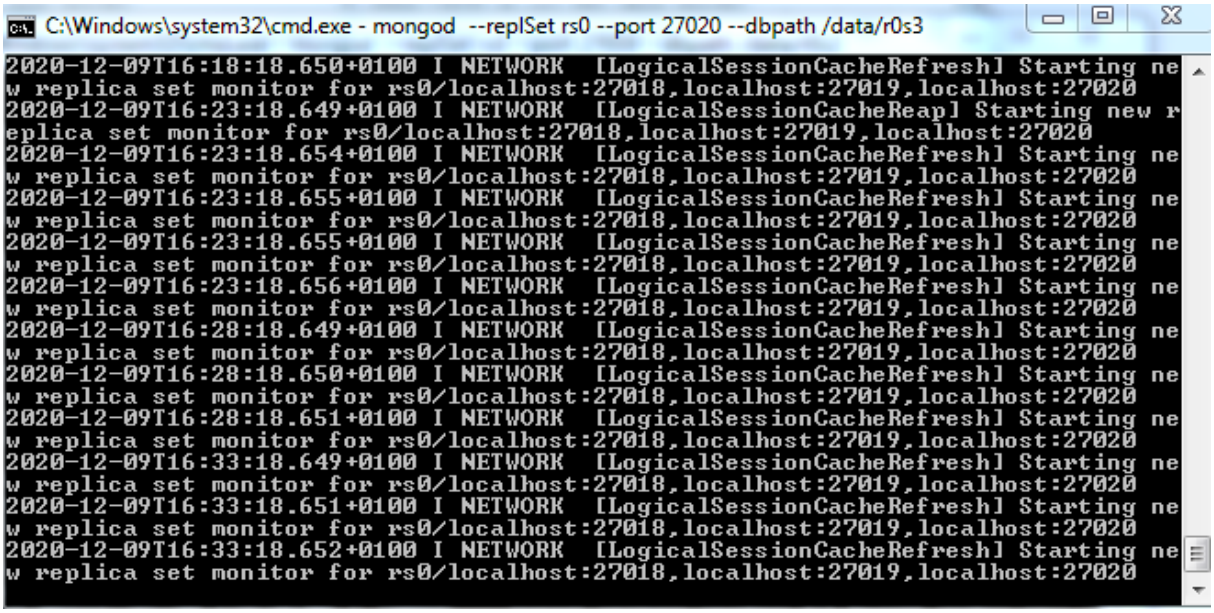
Le serveur r0s1 :



Le serveur r0s2 :



Le serveur r0s3 :



Initialisation replicaSet

On va ensuite connecter les serveurs ensemble et les initialiser

Pour cela je me suis connecté sur le serveur principal à l'aide de la commande `mongo -port <N° port>` après cela j'initialise le replicaSet à l'aide de la commande `rs.initiate`

Ajout des différents serveurs au replicaSet

La ligne `Rs0 :Primary>` apparait

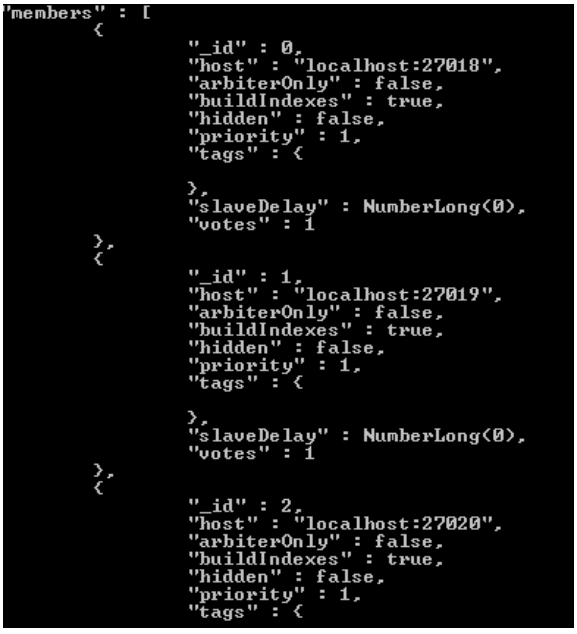
Nous allons maintenant ajouter les serveurs r0s2 et r0s3 avec la commande :

```
rs.add("localhost :27019 ")
```

```
rs.add("localhost :27019 ")
```

Vérification de la configuration du replicaSet :

La commande `rs.conf` permet de vérifier l'ajout des serveurs de réplica



Dans l'onglet host nous allons retrouver où se situe les serveurs définis et le port qu'il utilise.

Vérification du statut du replicaSet :

Pour connaitre le statuts du serveur (primaire ou secondaire) grâce à la commande `rs.status()`

Les rôles qui ont été définis sont utiles lors d'un arrêt du serveur primaire de basculer sur un deux serveur restant actif. Pour notre exemple, si le serveur primaire s'arrête, les autres serveurs vont devoir se mettre d'accord sur le serveur qui va prendre les relais, on parle alors d'élection ce qui peut provoquer un temps de rétablissement plus élever que si nous utilisons une entier arbitre.

Créations d'un arbitre

De la même façon que pour créer un serveur nous allons créer un répertoire dans `/data/arb`

Puis nous allons exécuter la commande dans le réplicaSet : `mongod –port 30000 –dbpath /data/arb –replSet rs0`

Mettre en place l'arbitre replicaSet : `mongod –port 30000 –dbpath /data/arb –replSet rs0`

Ajouter l'arbitre : `Rs.addArb("localhost :30000") ;`

S'assurer de l'ajout de l'arbitre à l'aide de la commande : `rs.status()`