Si Shinken Inférieur à V02.07.00 - Montée de version en Mongodb 3.0 (réalisée automatiquement sous conditions)

Sommaire

Problématique

La version V02.07.00 de Shinken Entreprise apporte un lot de nouveautés, et notamment la mise à jour de la base de données Mongo utilisée pour stocker différents types de données. Cette mise à jour apporte notamment un nouveau système de gestion des données internes à Mongo qui promet une meilleure gestion et performances au niveau de Mongo, ainsi que des améliorations au niveau de la sécurité.

Dans une installation Shinken Entreprise classique, la mise à jour de Mongo est automatisée par le script d'installation.

Lorsque Mongo est configuré pour fonctionner en tant que cluster, le script d'installation/mise à jour de Shinken permet également d'effectuer la mise à jour de cette installation de cluster Mongo de manière automatique. Cependant, compte tenu de la configuration plus complexes, des manipulations supplémentaires sont nécessaires.

Ces manipulations sont décrites de la procédure ci-dessous.

Procédure de mise à jour

Description du fonctionnement de la mise à jour du cluster

Dans un cluster Mongo, la mise à jour de Mongo sur les différents membres du cluster doit être simultanée pour éviter d'avoir des versions de Mongo différentes au sein du cluster, ce qui entraînerait des incompatibilités.

Pour résoudre ce problème, le script de mise à jour de Shinken détecte lorsque la mise à jour se fait sur un membre du cluster Mongo. Dans ce cas, il lance alors la mise à jour de Mongo de manière simultanée sur toutes les machines du cluster Mongo (en s'y connectant via SSH).

Une fois la mise à jour du cluster Mongo terminée, le script de mise à jour Shinken continue la mise à jour de Shinken en V02.07.00 comme sur une installation classique.

La procédure est donc la suivante:

- Lancement de la mise à jour de Shinken sur un noeud du cluster Mongo
- Le script de mise à jour se connecte sur l'ensemble des noeuds du cluster Mongo et met à jour Mongo et sa configuration.
- Mongo maintenant à jour sur tous les noeuds du cluster, la mise à jour de Shinken continue
- Lancement de la mise a jour de Shinken sur les autres noeuds du cluster. Shinken est mis à jour sur ces noeuds. Puisque Mongo est déjà à jour, la partie sur la mise à jour du cluster est ignorée par la mise à jour.
- La mise à jour de Shinken et de Mongo est terminée sur tous les noeuds du cluster Mongo. Shinken est opérationnel.

> Dans la suite de cette documentation, on supposera que le cluster Mongo est composé de 3 nœuds. Dans le cas d'une configuration de cluster Mongo avec davantage de nœuds, il faudra donc adapter les commandes présentées ci-dessous.

Connexion entre machines du cluster

Pour permettre un synchronisation de la mise à jour de Mongo sur toutes les machines du cluster, il faut qu'un des nœuds du cluster Mongo (n'importe lequel, pas forcément le nœud primaire) puisse se connecter via SSH sur les autres nœuds.

Cette connexion doit pouvoir se faire sur l'utilisateur *root* et sans avoir besoin de mot de passe. Une copie de la clé SSH de l'utilisateur *root* de la machine effectuant la mise à jour vers le compte *root* des autres machines du cluster Mongo est donc nécessaire avant le lancement de la mise à jour.

Sur la machine qui effectue la mise à jour du cluster, on effectue alors les commandes suivantes:

- Si aucune clé SSH n'existe pour l'utilisateur root (~/.ssh/id_rsa.pub), en générer une en tant que *root*:

```
ssh-keygen
```

- Copier la clé sur les nœuds du cluster:

```
for i in noeud1 noeud2 noeud3; do ssh-copy-id ~/.ssh/id_rsa.pub root@$i; done
```

- Vérification de la connexion SSH. En tant que root, on vérifie la connexion SSH avec tous les noeuds du cluster:

```
ssh root@noeud1
ssh root@noeud2
ssh root@noeud3
```

  Si les clés ont bien été copiées, chacune des connexion doit s'effectuer avec succès et sans demander de mot de passe.

Déroulement de la mise à jour

Une fois qu'une machine du cluster Mongo a été choisie et peut se connecter avec une clé SSH en root sur les autres machines du cluster, le processus de mise à jour peut être commencé:

1. **Sur la machine choisie, lancer la mise a jour de Shinken**. Le script de mise à jour de Shinken se connecte via SSH sur les autres machines du cluster Mongo et procède à la mise à jour simultanée de Mongo sur tous les membres du cluster.
   Une fois la mise à jour de cette machine terminée, tous les membres du cluster Mongo ont été mis à jour en version v3.0.15 de Mongo. Shinken a été également mis à jour sur cette machine.
2. **Sur les autres machines du cluster Mongo, lancer l'installation de Shinken**. Puisque Mongo a déjà été mis à jour, la partie de mise à jour du cluster Mongo est ignorée et Shinken est mis à jour de manière classique.

Mise à jour des fichiers de configuration

Une des nouveautés de la version 3 de Mongodb est la modification du format des fichiers de configuration. Lors de la mise à jour, ces fichiers ne sont pas modifiés et conservent le format de la version v2.6.9.

Cependant, pour être le plus à jour possible et éviter les problèmes potentiels dus à l'utilisation de fichiers obsolètes, il est conseillé de mettre à jour ces fichiers de configuration dans le nouveau format utilisé par Mongodb.

Dans la suite de cette documentation sont présentés côte à côte ces 2 formats de fichier (ancien à gauche, nouveau à droite), ce qui permet ensuite de faire la modification facilement.

Démon mongod

/etc/mongod.conf (ancien format)

```
# mongod.conf

#where to log
logpath=/var/log/mongodb/mongod.log

logappend=true

# fork and run in background
fork=true

port=27018

dbpath=/var/lib/mongo

# location of pidfile
pidfilepath=/var/run/mongodb/mongod.pid

# Listen to local interface only. Comment out to
listen on all interfaces.
#bind_ip=127.0.0.1

# Disables write-ahead journaling
# nojournal=true

# Enables periodic logging of CPU utilization and I
/O wait
#cpu=true

# Turn on/off security. Off is currently the
default
#noauth=true
#auth=true

# Verbose logging output.
#verbose=true

# Inspect all client data for validity on receipt
(useful for
# developing drivers)
#objcheck=true

# Enable db quota management
#quota=true

# Set oplogging level where n is
# 0=off (default)
# 1=W
# 2=R
# 3=both
# 7=W+some reads
#diaglog=0

# Ignore query hints
#nohints=true

# Enable the HTTP interface (Defaults to port
28017).
#httpinterface=true

# Turns off server-side scripting. This will
result in greatly limited
# functionality
#noscripting=true

# Turns off table scans. Any query that would do a
table scan fails.
#notablescan=true

# Disable data file preallocation.
#noprealloc=true
```

/etc/mongod.conf (nouveau format)

```
# mongod.conf

# for documentation of all options, see:
# http://docs.mongodb.org/manual/reference
/configuration-options/

# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: /var/log/mongodb/mongod.log

# Where and how to store data.
storage:
  dbPath: /var/lib/mongo
  journal:
    enabled: true
  # engine:
  # mmapv1:
  # wiredTiger:

# how the process runs
processManagement:
  fork: true # fork and run in background
  pidFilePath: /var/run/mongodb/mongod.pid #
location of pidfile

# network interfaces
net:
  port: 27017
  bindIp: 127.0.0.1 # Listen to local interface
only, comment to listen on all interfaces.


#security:

#operationProfiling:

replication:
  replSetName: rs-shinken

#sharding:

## Enterprise-Only Options

#auditLog:

#snmp:
net:
 unixDomainSocket:
  enabled: false
```

```
# Specify .ns file size for new databases.
# nssize=<size>

# Replication Options

# in replicated mongo databases, specify the
replica set name here
#replSet=setname
replSet=rs-shinken
# maximum size in megabytes for replication
operation log
#oplogSize=1024
# path to a key file storing authentication info
for connections
# between replica set members
#keyFile=/path/to/keyfile
keyFile=/etc/mongod.keyfile
```

Démon mongos

/etc/mongos.conf (ancien format)

```
logpath=/var/log/mongodb/mongos.log
logappend=true
fork=true
pidfilepath=/var/run/mongodb/mongos.pid

# Port to listen for application connections
port=27017

# Listen to local interface only. Comment out to
listen on all interfaces.
#bind_ip=127.0.0.1

# Change here your real configuration servers
# Better to set names from a /etc/hosts than
# IP that can change in the future
configdb=node1:27019,node2:27019,node3:27019

# Security file, for node authentifications
# IMPORTANT: MUST be the same across ALL your
mongo servers
# and with mongod:mongod 600 rights
keyFile=/etc/mongod.keyfile
```

/etc/mongos.conf (nouveau format)

```
# mongos.conf

# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: /var/log/mongodb/mongos.log

# how the process runs
processManagement:
  fork: true # fork and run in background
  pidFilePath: /var/run/mongodb/mongos.pid #
location of pidfile

# network interfaces
net:
  port: 27017
  #bindIp: 127.0.0.1 # Listen to local interface
only, comment to listen on all interfaces.

sharding:
  configDB: node1:27019,node2:27019,node3:27019
```

Démon mongo-configsrv

/etc/mongo-configsrv.conf (ancien format)

```
# mongo-configsrv.conf

#where to log
logpath=/var/log/mongodb/mongo-configsrv.log

logappend=true

# fork and run in background
fork=true

port=27019

dbpath=/var/lib/mongo/configdb

# location of pidfile
```

```
pidfilepath=/var/run/mongodb/mongo-configsrv.pid

# Listen to local interface only. Comment out to
listen on all interfaces.
#bind_ip=127.0.0.1

# Disables write-ahead journaling
# nojournal=true

# Enables periodic logging of CPU utilization and I
/O wait
#cpu=true

# Turn on/off security.  Off is currently the
default
#noauth=true
#auth=true

# Verbose logging output.
#verbose=true

# Inspect all client data for validity on receipt
(useful for
# developing drivers)
#objcheck=true

# Enable db quota management
#quota=true

# Set oplogging level where n is
#   0=off (default)
#   1=W
#   2=R
#   3=both
#   7=W+some reads
#diaglog=0

# Ignore query hints
#nohints=true

# Enable the HTTP interface (Defaults to port
28017).
#httpinterface=true

# Turns off server-side scripting.  This will
result in greatly limited
# functionality
#noscripting=true

# Turns off table scans.  Any query that would do
a table scan fails.
#notablescan=true

# Disable data file preallocation.
#noprealloc=true

# Specify .ns file size for new databases.
# nssize=<size>

# Replication Options

# in replicated mongo databases, specify the
replica set name here
#replSet=setname
# maximum size in megabytes for replication
operation log
#oplogSize=1024

# path to a key file storing authentication info
for connections
# between replica set members
```

/etc/mongo-configsrv.conf (nouveau format)

```
# mongo-configsrv.conf

# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: /var/log/mongodb/mongo-configsrv.log

# Where and how to store data.
storage:
  dbPath: /var/lib/mongo/configdb
  journal:
    enabled: true

# how the process runs
processManagement:
  fork: true  # fork and run in background
  pidFilePath: /var/run/mongodb/mongo-configsrv.
pid  # location of pidfile

# network interfaces
net:
  port: 27019
  #bindIp: 127.0.0.1  # Listen to local interface
only, comment to listen on all interfaces.
```

```
# IMPORTANT: MUST be the same across ALL your
mongo servers
# and with mongod:mongod 600 rights
keyFile=/etc/mongod.keyfile
```