IMP: MongoDB

Thomas Uyttendaele

22 augustus 2014

1 Algemene inleiding

In deze sectie zal voor de verschillende systemen de automatisering van installatie en configuratie met behulp van IMP uitgelegd worden. Er zal steeds de afhankelijkheden gegeven worden, een domeinmodel, uitleg bij het domeinmodel en voorbeeld configuratie gegeven worden.

De automatisatie van installatie is ontwikkeld en getest met Fedora 18 en 20, op andere distributies en versies is er niet getest. Elk systeem maakt gebruik van *ip::services::Server*, een instantie hiervan is een (virtuele) machine met een IP adres en besturingssysteem.

Bij elke instantie is het verplicht om de firewall uit te zetten en SELinux op permissive te zetten. Dit kan met behulp van de volgende commando's:

```
systemctl stop firewalld.service
systemctl disable firewalld.service
setenforce 0
sed -i "s/SELINUX=enforcing/SELINUX=permissive/g" /etc/
    sysconfig/selinux
sed -i "s/SELINUX=enforcing/SELINUX=permissive/g" /etc/
    selinux/config|
```

2 MongoDB

Link: *https://github.com/thuys/mongodb*

Benodigde IMP modules: std, net, ip, redhat, hosts en yum.

De installatie en configuratie is gebeurd aan de hand van de uitleg en yum-repository van MongoDB[1].

2.1 Domein model en uitleg

Het domeinmodel is te zien in figuur 1.

[1] http://docs.mongodb.org/manual/tutorial/install-mongodb-on-red-hat-centos-or-fedora-linux/, http://docs.mongodb.org/manual/tutorial/deploy-replica-set-for-testing/ en http://docs.mongodb.org/manual/tutorial/deploy-shard-cluster/

MongoDB is een server in het IMP model en is verantwoordelijk voor het installeren van de basis van MongoDB. Hierna zijn basis commando's voor connectie te maken met een MongoDB instantie beschikbaar.

MonogDBServer is een server in het IMP model en is verantwoordelijk voor het installeren van de MongoDB server.

MongoDBNode is de implementatie van een data instantie, maximaal 1 per server. Indien gelinkt met een replica set zal deze als een deel van een replica set worden genitialiseerd, anders als een zelfstandige instantie.

MongoDBReplicaSet is de voorstelling van een replica set, dit wordt niet aan een specifieke server toegewezen.

MongoDBReplicaSetController is verantwoordelijk om de replica set te initialiseren. Belangrijk is dat indien er een uitbreiding is van de set, de node verbonden met de controller een reeds genitialiseerde node is.

MongoDBConfigServer is de implementatie van een configuratie server, 1 of 3 servers zijn nodig per cluster.

MongoDBAccessServer is de implementatie van mongos, minstens 1 is nodig maar meer kunnen gebruikt worden.

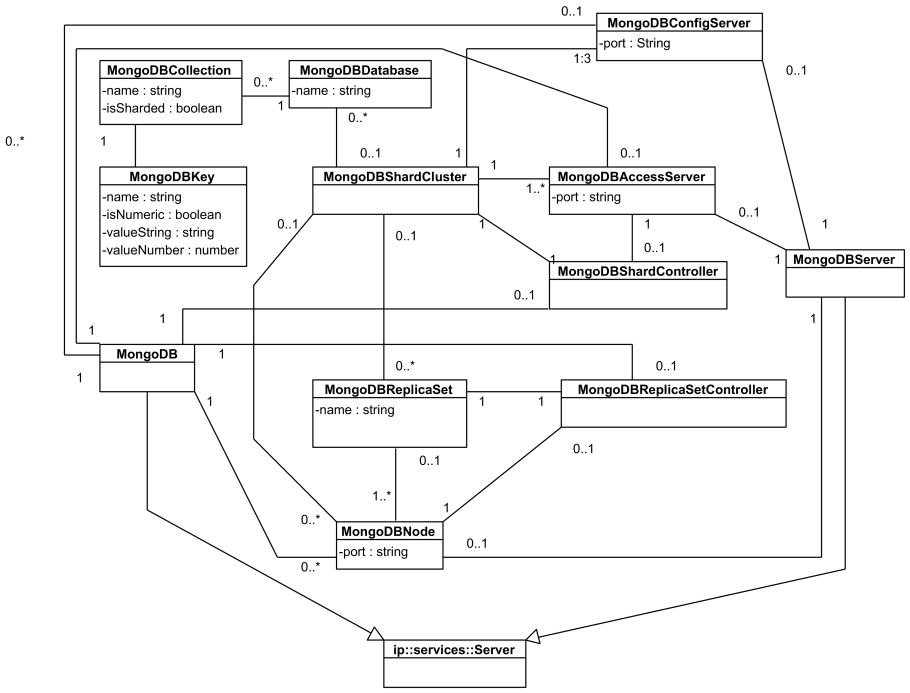
MongoDBShardCluster is de voorstelling van een cluster van shards, er kunnen zowel alleenstaand instanties als replica sets aan toegevoegd worden.

MongoDBShardController is verantwoordelijk om de cluster te initialiseren met de verschillende shards, databases, collecties en keys.

MongoDBDatabase is de voorstelling van een database.

MongoDBCollection is de voorstelling van een collectie, indien verbonden met een cluster via een database zal deze gedeeld worden over de verschillende shards.

MongoDBKey is de wijze waarmee een collectie verdeeld wordt over de verschillende shards.



Figuur 1: MongoDB: Domeinmodel MongoDB in IMP

2.2 Voorbeeld configuratie

De configuratie voor de testomgeving gaat als onderstaand. Bij de uitrol van IMP gaat dit verschillende keren uitgevoerd moeten worden omdat eerst de MongoDBNodes moeten draaien, vervolgens kunnen de replicasets aangemaakt worden, daarna kunnen de replicasets pas toegevoegd worden in de cluster.

In IMP was het nog niet mogelijk om een te zeggen dat x uitgevoerd moet zijn op een andere instantie, vooraleer y kan uitgevoerd worden, ondertussen is dit mogelijk door de thesis van Harm De Weirdt[**thesisHarm**] waar de nieuwe installatie beschikbaar is op https://github.com/Foezjie/mongodb maar hierbij dient ook gebruik gemaakt te worden van zijn IMP installatie.
Met het ontbreken hieraan kan het zijn dat er 3 keer een volledige IMP deploy uitgevoerd moet worden.

```
vmMDB1 = ip::Host(name = "vmmdb1", os = "fedora-18", ip =
    "172.16.32.45")
vmMDB2 = ip::Host(name = "vmmdb2", os = "fedora-18", ip =
    "172.16.32.46")
vmMDB3 = ip::Host(name = "vmmdb3", os = "fedora-18", ip =
    "172.16.32.47")
vmMDB4 = ip::Host(name = "vmmdb4", os = "fedora-18", ip =
```

```
    "172.16.32.48")
vmMDB5 = ip::Host(name = "vmmdb5", os = "fedora-18", ip =
    "172.16.32.49")
vmMDB6 = ip::Host(name = "vmmdb6", os = "fedora-18", ip =
    "172.16.32.50")

mongo1 = mongodb::MongoDB(host = vmMDB1)
mongo2 = mongodb::MongoDB(host = vmMDB2)
mongo3 = mongodb::MongoDB(host = vmMDB3)
mongo4 = mongodb::MongoDB(host = vmMDB4)
mongo5 = mongodb::MongoDB(host = vmMDB5)
mongo6 = mongodb::MongoDB(host = vmMDB6)

mongo1Server = mongodb::MongoDBServer(host=vmMDB1)
mongo2Server = mongodb::MongoDBServer(host=vmMDB2)
mongo3Server = mongodb::MongoDBServer(host=vmMDB3)
mongo4Server = mongodb::MongoDBServer(host=vmMDB4)
mongo5Server = mongodb::MongoDBServer(host=vmMDB5)
mongo6Server = mongodb::MongoDBServer(host=vmMDB6)

mongoN1 = mongodb::MongoDBNode(host=mongo1, server=
    mongo1Server)
mongoN2 = mongodb::MongoDBNode(host=mongo2, server=
    mongo2Server)
mongoN3 = mongodb::MongoDBNode(host=mongo3, server=
    mongo3Server)
mongoN4 = mongodb::MongoDBNode(host=mongo4, server=
    mongo4Server)
mongoN5 = mongodb::MongoDBNode(host=mongo5, server=
    mongo5Server)
mongoN6 = mongodb::MongoDBNode(host=mongo6, server=
    mongo6Server)

set1 = mongodb::MongoDBReplicaSet(name="repl1", nodes = [
    mongoN1, mongoN2, mongoN3])
set2 = mongodb::MongoDBReplicaSet(name="repl2", nodes = [
    mongoN4, mongoN5, mongoN6])

controller1 = mongodb::MongoDBReplicaSetController(host=
    mongo1, replicaSet = set1, connectingNode = mongoN1)
controller2 = mongodb::MongoDBReplicaSetController(host=
    mongo4, replicaSet = set2, connectingNode = mongoN4)

mongoDBCluster = mongodb::MongoDBShardCluster(replicaSets
    = [set1, set2])
shardController = mongodb::MongoDBShardController(host=
```

```
    mongo5, accessServer = access3, shardCluster =
    mongoDBCluster)

access1 = mongodb::MongoDBAccessServer(host=mongo2,
    server= mongo2Server, shardCluster = mongoDBCluster)
access2 = mongodb::MongoDBAccessServer(host=mongo3,
    server= mongo3Server, shardCluster = mongoDBCluster)

config1 = mongodb::MongoDBConfigServer(host=mongo2,
    server= mongo2Server, shardCluster = mongoDBCluster)

databaseYCSB = mongodb::MongoDBDatabase(name="ycsb",
    shardCluster = mongoDBCluster)
collectionYCSB = mongodb::MongoDBCollection(name="
    usertable", database = databaseYCSB)
keyYCSB = mongodb::MongoDBKey(name = "_id", valueString =
     "hashed", collection = collectionYCSB)
```