Глеб Воронов | TradeNarK LLC
IT — искусство XXI века. И мы знаем в нём толк

MongoDB и адекватная репликация с Replica Set

Автор: Воронов Глеб | 08-06-2015 5 комментариев

MongoBD документо-ориентированная СУБД с JSON-подобными схемами данных
Для установки первым делом необходимо добавить официальный репозиторий продукта

```
# vim /etc/yum.repos.d/mongodb.repo

[mongodb]
name=MongoDB Repository
baseurl=http://downloads-distro.mongodb.org/repo/redhat/os/x86_64/
gpgcheck=0
enabled=1
```

Устанавливаем, запускаем и проверяем работоспособность сервера

```
# yum install mongo-10gen mongo-10gen-server
# systemctl start mongod
# netstat -lntpu | grep mongo
tcp        0      0 127.0.0.1:27017         0.0.0.0:*               LISTEN      13709/mong
```

Основной задачей в данный момент является настройка репликации.
MongoDB в отличии от MySQL поддерживает 2 формы репликации:
— реплисеты (Replica Sets)

— ведущий-ведомый (Master-Slave)
Подробности можно узнать на официальном сайте

Обыкновенная настройка Master-Slave не интересна да и не рекомендуется разработчиками. Я хочу проверить реплисеты. Для этого указываю имя реплики для всего сервера, раскоментровав отвечающую за это строку

```
# vim /etc/mongod.conf
replSet = FirstReplica
```

Перезапускаю сервер для применения изменений.

```
# systemctl restart mongod
```

Есть вероятность, что этот шаг можно было пропустить, так как при создании инстансов вручную указывается одно и тоже имя реплики для каждого процесса. Если кто то попробует — напишите в комментариях о результате

В нашем тестовом окружении необходимо запустить 3 экземпляра сервера. 3 это минимум, необходимый для «правильной» реплики: один из серверов выступает исключительно в роли арбитра и не принимает на себя никакие данные.
Он всего лишь помогает выбрать сервер, который будет **PRIMARY**. Это его свойство позволяет использовать в качестве арбитра сервер с минимальными ресурсами.
Для имитирования ситуации с 3 разными серверами я создам 3 процесса mongod с отдельным портом и базой:
Создание самих баз и установка необходимого владельца:

```
# mkdir /var/lib/mongo/{db1,db2,db3}
# chown mongod:mongod /var/lib/mongo/{db1,db2,db3}
```

Этот сервер будет PRIMARY (Мастер)

```
# mongod --dbpath /var/lib/mongo/db1 --port 27001 --replSet FirstReplica --fork --logpath
about to fork child process, waiting until server is ready for connections.
forked process: 18042
child process started successfully, parent exiting
```

Этот сервер будет SECONDARY (слейв)

```
# mongod --dbpath /var/lib/mongo/db2 --port 27002 --replSet FirstReplica --fork --logpath
about to fork child process, waiting until server is ready for connections.
forked process: 18115
child process started successfully, parent exiting
```

Этот сервер будет арбитром, не принимающим данных

```
# mongod --dbpath /var/lib/mongo/db3 --port 27003 --replSet FirstReplica --fork --logpath
about to fork child process, waiting until server is ready for connections.
forked process: 18226
child process started successfully, parent exiting
```

```
--dbpath        параметр, указывающий дирикторию для файлов БД
--port          номер корта, к которому смогут подключаться клиенты. В данном случае испол
--replSet       название набора реплик. Должно быть одинаково на всех реплицируемых сервер
--fork          параметр, отвечающий за запуск mongod в режиме демона.
--logpath       файл для перенаправления вывода
```

Подключаемся к основному серверу и настраиваем реплику из консоли mongo

```
# mongo --port 27001
MongoDB shell version: 2.6.10
connecting to: 127.0.0.1:27001/test
Server has startup warnings:
2015-06-08T11:43:29.342+0300 [initandlisten]
2015-06-08T11:43:29.342+0300 [initandlisten] ** WARNING: Readahead for /var/lib/mongo/db1
2015-06-08T11:43:29.342+0300 [initandlisten] **          We suggest setting it to 256KB (5
2015-06-08T11:43:29.342+0300 [initandlisten] **          http://dochub.mongodb.org/core/re
> rs.status()
{
        "startupStatus" : 3,
        "info" : "run rs.initiate(...) if not yet done for the set",
        "ok" : 0,
        "errmsg" : "can't get local.system.replset config from self or any seed (EMPTYCONF
}
> rs.initiate({"_id" : "FirstReplica", members : [
... {"_id" : 0, priority : 3, host : "127.0.0.1:27001"},
...  {"_id" : 1, host : "127.0.0.1:27002"},
...  {"_id" : 2, host : "127.0.0.1:27003", arbiterOnly : true}
... ]
... });
{
        "info" : "Config now saved locally.  Should come online in about a minute.",
        "ok" : 1
```

```
}
> rs.status()
{
        "set" : "FirstReplica",
        "date" : ISODate("2015-06-08T08:45:57Z"),
        "myState" : 1,
        "members" : [
                {
                        "_id" : 0,
                        "name" : "127.0.0.1:27001",
                        "health" : 1,
                        "state" : 1,
                        "stateStr" : "PRIMARY",
                        "uptime" : 148,
                        "optime" : Timestamp(1433753125, 1),
                        "optimeDate" : ISODate("2015-06-08T08:45:25Z"),
                        "electionTime" : Timestamp(1433753133, 1),
                        "electionDate" : ISODate("2015-06-08T08:45:33Z"),
                        "self" : true
                },
                {
                        "_id" : 1,
                        "name" : "127.0.0.1:27002",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 32,
                        "optime" : Timestamp(1433753125, 1),
                        "optimeDate" : ISODate("2015-06-08T08:45:25Z"),
                        "lastHeartbeat" : ISODate("2015-06-08T08:45:57Z"),
                        "lastHeartbeatRecv" : ISODate("2015-06-08T08:45:55Z"),
                        "pingMs" : 0,
                        "syncingTo" : "127.0.0.1:27001"
                },
                {
                        "_id" : 2,
                        "name" : "127.0.0.1:27003",
                        "health" : 1,
                        "state" : 7,
                        "stateStr" : "ARBITER",
                        "uptime" : 32,
                        "lastHeartbeat" : ISODate("2015-06-08T08:45:57Z"),
                        "lastHeartbeatRecv" : ISODate("2015-06-08T08:45:56Z"),
                        "pingMs" : 0
                }
        ],
        "ok" : 1
}
FirstReplica:PRIMARY> quit()
```

Для проверки подключимся к оставшимся серверам и посмотрим их статус

```
# mongo --port 27002
```

```
MongoDB shell version: 2.6.10
connecting to: 127.0.0.1:27002/test
Server has startup warnings:
2015-06-08T11:43:40.736+0300 [initandlisten]
2015-06-08T11:43:40.736+0300 [initandlisten] ** WARNING: Readahead for /var/lib/mongo/db2
2015-06-08T11:43:40.736+0300 [initandlisten] **          We suggest setting it to 256KB (5
2015-06-08T11:43:40.736+0300 [initandlisten] **          http://dochub.mongodb.org/core/re
FirstReplica:SECONDARY> quit()
```

Видим, что сервер видит реплику и его статус SECONDARY

```
# mongo --port 27003
MongoDB shell version: 2.6.10
connecting to: 127.0.0.1:27003/test
Server has startup warnings:
2015-06-08T11:43:49.369+0300 [initandlisten]
2015-06-08T11:43:49.369+0300 [initandlisten] ** WARNING: Readahead for /var/lib/mongo/db3
2015-06-08T11:43:49.369+0300 [initandlisten] **          We suggest setting it to 256KB (5
2015-06-08T11:43:49.369+0300 [initandlisten] **          http://dochub.mongodb.org/core/re
FirstReplica:ARBITER> quit()
```

Аналогично видим, что этот инстанс является частью реплики FirstReplica и выступает в качестве арбитра

Вроде как завелось. Теперь попробую создать реальные условия:

Инициируем «падение» PRIMARY сервера:

```
# netstat -lntpu | grep mongod
tcp        0      0 127.0.0.1:27017         0.0.0.0:*               LISTEN      15726/mong
tcp        0      0 0.0.0.0:27001           0.0.0.0:*               LISTEN      18042/mong
tcp        0      0 0.0.0.0:27002           0.0.0.0:*               LISTEN      18115/mong

tcp        0      0 0.0.0.0:27003           0.0.0.0:*               LISTEN      18226/mong
# kill -HUP 18042
```

Я прибил процесс, на котором висел PRIMARY сервер. Нет процесса — нет доступа к серверу.

Вернёмся к арбитру и проверим состояние реплики:

```
# mongo --port 27003
MongoDB shell version: 2.6.10
connecting to: 127.0.0.1:27003/test
Server has startup warnings:
2015-06-08T11:43:49.369+0300 [initandlisten]
2015-06-08T11:43:49.369+0300 [initandlisten] ** WARNING: Readahead for /var/lib/mongo/db3
2015-06-08T11:43:49.369+0300 [initandlisten] **          We suggest setting it to 256KB (5
```

```
2015-06-08T11:43:49.369+0300 [initandlisten] **          http://dochub.mongodb.org/core/re
FirstReplica:ARBITER> rs.status()
{
        "set" : "FirstReplica",
        "date" : ISODate("2015-06-08T09:00:10Z"),
        "myState" : 7,
        "members" : [
                {
                        "_id" : 0,
                        "name" : "127.0.0.1:27001",
                        "health" : 0,
                        "state" : 8,
                        "stateStr" : "(not reachable/healthy)",
                        "uptime" : 0,
                        "optime" : Timestamp(1433753125, 1),
                        "optimeDate" : ISODate("2015-06-08T08:45:25Z"),
                        "lastHeartbeat" : ISODate("2015-06-08T09:00:05Z"),
                        "lastHeartbeatRecv" : ISODate("2015-06-08T08:59:35Z"),
                        "pingMs" : 0
                },
                {
                        "_id" : 1,
                        "name" : "127.0.0.1:27002",
                        "health" : 1,

                        "state" : 1,
                        "stateStr" : "PRIMARY",
                        "uptime" : 882,
                        "optime" : Timestamp(1433753125, 1),
                        "optimeDate" : ISODate("2015-06-08T08:45:25Z"),
                        "lastHeartbeat" : ISODate("2015-06-08T09:00:08Z"),
                        "lastHeartbeatRecv" : ISODate("2015-06-08T09:00:10Z"),
                        "pingMs" : 0,
                        "electionTime" : Timestamp(1433753984, 1),
                        "electionDate" : ISODate("2015-06-08T08:59:44Z")
                },
                {
                        "_id" : 2,
                        "name" : "127.0.0.1:27003",
                        "health" : 1,
                        "state" : 7,
                        "stateStr" : "ARBITER",
                        "uptime" : 981,
                        "self" : true
                }
        ],
        "ok" : 1
}
FirstReplica:ARBITER>
```

Видно, что главный сервер недоступен **(«stateStr» : «(not reachable/healthy)»)**, а сервер с id 1 стал **PRIMARY**

Теперь инициируем «поднятие» основного мастера:

```
# mongod --dbpath /var/lib/mongo/db1 --port 27001 --replSet FirstReplica --fork --logpath
about to fork child process, waiting until server is ready for connections.
forked process: 27144
child process started successfully, parent exiting
```

И снова интересуемся у арбитра как там дела у сервачков:

```
# mongo --port 27003
MongoDB shell version: 2.6.10
connecting to: 127.0.0.1:27003/test
Server has startup warnings:
2015-06-08T11:43:49.369+0300 [initandlisten]
2015-06-08T11:43:49.369+0300 [initandlisten] ** WARNING: Readahead for /var/lib/mongo/db3
2015-06-08T11:43:49.369+0300 [initandlisten] **          We suggest setting it to 256KB (5
2015-06-08T11:43:49.369+0300 [initandlisten] **          http://dochub.mongodb.org/core/re
FirstReplica:ARBITER> rs.status()
{
        "set" : "FirstReplica",
        "date" : ISODate("2015-06-08T09:04:42Z"),
        "myState" : 7,
        "members" : [
                {
                        "_id" : 0,
                        "name" : "127.0.0.1:27001",
                        "health" : 1,
                        "state" : 1,
                        "stateStr" : "PRIMARY",
                        "uptime" : 49,
                        "optime" : Timestamp(1433753125, 1),
                        "optimeDate" : ISODate("2015-06-08T08:45:25Z"),
                        "lastHeartbeat" : ISODate("2015-06-08T09:04:41Z"),
                        "lastHeartbeatRecv" : ISODate("2015-06-08T09:04:40Z"),
                        "pingMs" : 0,
                        "electionTime" : Timestamp(1433754240, 1),
                        "electionDate" : ISODate("2015-06-08T09:04:00Z")
                },
                {
                        "_id" : 1,
                        "name" : "127.0.0.1:27002",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 1154,
                        "optime" : Timestamp(1433753125, 1),
                        "optimeDate" : ISODate("2015-06-08T08:45:25Z"),
                        "lastHeartbeat" : ISODate("2015-06-08T09:04:40Z"),
                        "lastHeartbeatRecv" : ISODate("2015-06-08T09:04:40Z"),
                        "pingMs" : 0,
                        "lastHeartbeatMessage" : "syncing to: 127.0.0.1:27001",
                        "syncingTo" : "127.0.0.1:27001"
                },
                {
                        "_id" : 2,
```

```
                        "name" : "127.0.0.1:27003",
                        "health" : 1,
                        "state" : 7,
                        "stateStr" : "ARBITER",
                        "uptime" : 1253,

                        "self" : true
                }
        ],
        "ok" : 1
}
FirstReplica:ARBITER> rs.config()
{
        "_id" : "FirstReplica",
        "version" : 1,
        "members" : [
                {
                        "_id" : 0,
                        "host" : "127.0.0.1:27001",
                        "priority" : 3
                },
                {
                        "_id" : 1,
                        "host" : "127.0.0.1:27002"
                },
                {
                        "_id" : 2,
                        "host" : "127.0.0.1:27003",
                        "arbiterOnly" : true
                }
        ]
}
```

Как видно выше сервер после возвращения из даунтайма снова принялся за свои обязанности, став мастером всей реплики. В этом ему помогает арбитр.
На этом простом примере мы рассмотрели создание простой репликации с особенностями, присущими программному продукту MongoDB. Следует учесть, что в данном случае реплика была создана до начала работ с базой данных.
Если имеется уже настроенный продакшн-сервер, то метод создания реплики может отличаться. Обязательно потренеруйтесь в тестовом окружении.
В заключении — очистка всего нашего хозяйства:

```
# ps aux | grep mongod
root      7823  0.0  0.0 112640   960 pts/1    R+   13:36   0:00 grep --color=auto mongod
root     22820  0.1  2.8 4985748 29460 ?       Sl   11:43   0:07 mongod --dbpath /var/lib/
root     22866  0.1  2.9 744312 30432 ?        Sl   11:43   0:07 mongod --dbpath /var/lib/
root     27144  0.1  3.0 4989860 31240 ?       Sl   12:03   0:06 mongod --dbpath /var/lib/
```

Так как я запускал процессы вручную, то и останавливать их нужно самостоятельно.
Для этого воспользуемся **db.shutdownServer()**

```
# mongo --port 27001
MongoDB shell version: 2.6.10
connecting to: 127.0.0.1:27001/test
Server has startup warnings:
2015-06-08T12:03:52.468+0300 [initandlisten]
2015-06-08T12:03:52.468+0300 [initandlisten] ** WARNING: Readahead for /var/lib/mongo/db1
2015-06-08T12:03:52.468+0300 [initandlisten] **          We suggest setting it to 256KB (5
2015-06-08T12:03:52.468+0300 [initandlisten] **          http://dochub.mongodb.org/core/re
FirstReplica:PRIMARY> use admin
switched to db admin
FirstReplica:PRIMARY> db.shutdownServer()
2015-06-08T13:41:18.996+0300 DBClientCursor::init call() failed
server should be down...
2015-06-08T13:41:18.999+0300 trying reconnect to 127.0.0.1:27001 (127.0.0.1) failed
2015-06-08T13:41:18.999+0300 reconnect 127.0.0.1:27001 (127.0.0.1) ok
FirstReplica:SECONDARY> quit()
```

Видим что сервер выключился и в репликации он переключился на **SECONDARY**
Аналогично останавливаем и остальные, но сначала арбитра, а потом наш **SECONDARY**
Проверяем

```
# ps aux | grep mongod
root     12528  0.0  0.0 112640   956 pts/1    R+   13:50   0:00 grep --color=auto mongod
```

Нет процессов mongod, только наш, который их ищет 🙂
Если базы, которые мы создали для теста более не нужны — их тоже можно удалить. Более того, они не так уж мало занимают

```
# du -h --max-depth=1 /var/lib/mongo/
0       /var/lib/mongo/journal
2.1G    /var/lib/mongo/db1
2.1G    /var/lib/mongo/db2
81M     /var/lib/mongo/db3
4.4G    /var/lib/mongo/
```

Удаляем и снова проверяем

```
# rm -rf /var/lib/mongo/{db1,db2,db3}
# du -h --max-depth=1 /var/lib/mongo/
```

```
0       /var/lib/mongo/journal
81M     /var/lib/mongo/
```

Так лучше. Люблю порядок на сервере

Раздел: CentOS Метки: arbiter, instance, json, mongo, mongod, mongodb, replication, СУБД

Так же будет интересно:

FTP сервер на CentOS 7 под ProFTPd с бекендом в MySQL

Мультисессионный VNC на CentOs 7 или делаем из Linux почти терминальник

Защита SSH от брутфорса с помощью iptables

CMS Odoo 10 на CentOS 7 с NGinx-ом и сертификацией

Открытие порта firewalld CentOS 7

MongoDB и адекватная репликация с Replica Set: 5 комментариев

GFB
03-07-2019

Доброго времени суток!
Подскажите, обязательно ли делать отдельный узел-арбитр?
Судя по этому туториалу http://37yonub.ru/articles/mongo-replica-set-docker-localhost арбитр и primary могут быть на одном узле. Или это ошибка?
Еще такой вопрос — как правильно общаться с кластером? Я могу писать только в primary и читать из любого, так?
DNS для записи это mongodb://localhost:27117/?replicaSet=TestMongoReplicaSet&connect=direct
DNS для чтения может быть такой mongodb://localhost:27117,localhost:27217,localhost:27317/?replicaSet=TestMongoReplicaSet&connect=direct

Воронов Глеб Автор записи
03-07-2019

Добрый день. Начнем с того что приведённый вами туториал тоже написан каким то пользователем.
Опираться в данном случае необходимо на официальную документацию
Эта документация гласит что арбитр сам по себе не обязателен. Реплику можно построить и без него.
Далее про чтение/запись.
Запись производится только на Primary.
Чтение так же производится с Primary, но при определённых конфигурациях реплики читать можно напрямую с Secondary. Необходимо выполнить разрешение на мастере с указание ID того секондари, с которого хотите читать.

По поводу «общения» с кластером как указал выше всё проходит через Primary до тех пор пока не настроете иначе

GFB
03-07-2019

Да, с этим разобрался. Арбитр не принимает трафик, только выбирает кому быть PRIMARY.
А как приложение, использующее кластер монго, работает со сменой PRIMARY сервера?
mongodb://localhost:27117,localhost:27217/?
replicaSet=TestMongoReplicaSet&connect=direct&readPreference=secondary
используем DNS со всеми перечисленными репликами и драйвер сам подбирает что есть PRIMARY и пишет туда? Я проверил, вроде работает. Правда есть небольшой промежутой времени (5-10 секунд) с ошибкой «NotMaster» — наверно арбитр назначает primary не сразу.

Воронов Глеб Автор записи
03-07-2019

Вопрос о смене Primary/Secondary немного не по адресу. Это к прогерам которые работают с драйверами
На практике смена Primary происходит меньше секунды в реплике из трёх инстансов Primary/Secondary/Arbiter
О каком DNS идёт речь если у вас везде используется **localhost**? Не понял вопроса, уточните

GFB
03-07-2019

Опечатка, DSN (connesion string).
Спасибо за подсказки)) дальше буду разбираться с драйверами)