版本：5.0

Dledger

Dledger快速搭建

前言

DLedger是一套基于Raft协议的分布式日志存储组件，部署 RocketMQ 时可以根据需要选择使用DLeger来替换原生的副本存储机制。本文档主要介绍如何快速构建和部署基于 DLedger 的可以自动容灾切换的 RocketMQ 集群。

1. 源码构建

构建分为两个部分，需要先构建 DLedger，然后 构建 RocketMQ。

1.1 构建 DLedger

```
$ git clone https://github.com/openmessaging/dledger.git
$ cd dledger
$ mvn clean install -DskipTests
```

1.2 构建 RocketMQ

```
$ git clone https://github.com/apache/rocketmq.git
$ cd rocketmq
$ git checkout -b develop origin/develop
$ mvn -Prelease-all -DskipTests clean install -U
```

2. 快速部署

在构建成功后

```
#{rocketmq-version} replace with rocketmq actual version. example: 5.1.0
$ cd distribution/target/rocketmq-{rocketmq-version}/rocketmq-{rocketmq-version}

$ sh bin/dledger/fast-try.sh start
```

如果上面的步骤执行成功，可以通过 mqadmin 运维命令查看集群状态。

```
$ sh bin/mqadmin clusterList -n 127.0.0.1:9876
```

顺利的话，会看到如下内容：

```
Java HotSpot(TM) 64-Bit Server VM warning: ignoring option PermSize=128m; support was removed in 8.0
Java HotSpot(TM) 64-Bit Server VM warning: ignoring option MaxPermSize=128m; support was removed in 8.0
#Cluster Name     #Broker Name            #BID  #Addr                #Version              #InTPS(LOAD)       #OutTPS(LOAD) #PCWait(ms) #Hour #SPACE
RaftCluster       RaftNode00              0     30.5.125.75:30931    V4_3_1                 0.00(0,0ms)         0.00(0,0ms)          0 429417.23 -1.0000
RaftCluster       RaftNode00              1     30.5.125.75:30911    V4_3_1                 0.00(0,0ms)         0.00(0,0ms)          0 429417.23 -1.0000
RaftCluster       RaftNode00              2     30.5.125.75:30921    V4_3_1                 0.00(0,0ms)         0.00(0,0ms)          0 429417.23 -1.0000
```

（BID 为 0 的表示 Master，其余都是 Follower）

启动成功，现在可以向集群收发消息，并进行容灾切换测试了。

关闭快速集群，可以执行：

```
$ sh bin/dledger/fast-try.sh stop
```

快速部署，默认配置在 conf/dledger 里面，默认的存储路径在 /tmp/rmqstore。

3. 容灾切换

部署成功，杀掉 Leader 之后（在上面的例子中，杀掉端口 30931 所在的进程），等待约 10s 左右，用 clusterList 命令查看集群，就会发现 Leader 切换到另一个节点了。

Dledger集群搭建

本部分主要介绍如何部署自动容灾切换的 RocketMQ-on-DLedger Group。

RocketMQ-on-DLedger Group 是指一组**相同名称**的 Broker，至少需要 3 个节点，通过 Raft 自动选举出一个 Leader，其余节点 作为 Follower，并在 Leader 和 Follower 之间复制数据以保证高可用。 RocketMQ-on-DLedger Group 能自动容灾切换，并保证数据一致。 RocketMQ-on-DLedger Group 是可以水平扩展的，也即可以部署任意多个 RocketMQ-on-DLedger Group 同时对外提供服务。

1. 新集群部署

1.1 编写配置

每个 RocketMQ-on-DLedger Group 至少准备三台机器（本文假设为 3）。 编写 3 个配置文件，建议参考 conf/dledger 目录下的配置文件样例。 关键配置介绍：

name	含义	举例
enableDLegerCommitLog	是否启动 DLedger	true
dLegerGroup	DLedger Raft Group的名字，建议和 brokerName 保持一致	RaftNode00
dLegerPeers	DLedger Group 内各节点的端口信息，同一个 Group 内的各个节点配置必须要保证一致	n0-127.0.0.1:40911;n1-127.0.0.1:40912;n2-127.0.0.1:40913
dLegerSelfId	节点 id, 必须属于 dLegerPeers 中的一个；同 Group 内各个节点要唯一	n0
sendMessageThreadPoolNums	发送线程个数，建议配置成 Cpu 核数	16

这里贴出 conf/dledger/broker-n0.conf 的配置举例。

```
brokerClusterName = RaftCluster
brokerName=RaftNode00
listenPort=30911
namesrvAddr=127.0.0.1:9876
storePathRootDir=/tmp/rmqstore/node00
storePathCommitLog=/tmp/rmqstore/node00/commitlog
enableDLegerCommitLog=true
dLegerGroup=RaftNode00
dLegerPeers=n0-127.0.0.1:40911;n1-127.0.0.1:40912;n2-127.0.0.1:40913
## must be unique
dLegerSelfId=n0
sendMessageThreadPoolNums=16
```

1.2 启动 Broker

与老版本的启动方式一致。

```
$ nohup sh bin/mqbroker -c conf/dledger/xxx-n0.conf &
$ nohup sh bin/mqbroker -c conf/dledger/xxx-n1.conf &
$ nohup sh bin/mqbroker -c conf/dledger/xxx-n2.conf &
```

2. 旧集群升级

如果旧集群采用 Master 方式部署，则每个 Master 都需要转换成一个 RocketMQ-on-DLedger Group。
如果旧集群采用 Master-Slave 方式部署，则每个 Master-Slave 组都需要转换成一个 RocketMQ-on-DLedger Group。

2.1 杀掉旧的 Broker

可以通过 kill 命令来完成，也可以调用 `bin/mqshutdown broker`。

2.2 检查旧的 Commitlog

RocketMQ-on-DLedger 组中的每个节点，可以兼容旧的 Commitlog ，但其 Raft 复制过程，只能针对新增加的消息。因此，为了避免出现异常，需要保证 旧的 Commitlog 是一致的。 如果旧的集群是采用 Master-Slave 方式部署，有可能在shutdown时，其数据并不是一致的，建议通过md5sum 的方式，检查最近的最少 2 个 Commitlog 文件，如果发现不一致，则通过拷贝的方式进行对齐。

虽然 RocketMQ-on-DLedger Group 也可以以 2 节点方式部署，但其会丧失容灾切换能力（2n + 1 原则，至少需要3个节点才能容忍其中 1 个宕机）。
所以在对齐了 Master 和 Slave 的 Commitlog 之后，还需要准备第 3 台机器，并把旧的 Commitlog 从 Master 拷贝到 第 3 台机器（记得同时拷贝一下 config 文件夹）。

在 3 台机器准备好了之后，旧 Commitlog 文件也保证一致之后，就可以开始走下一步修改配置了。

2.3 修改配置

参考新集群部署。

2.4 重新启动 Broker

参考新集群部署。