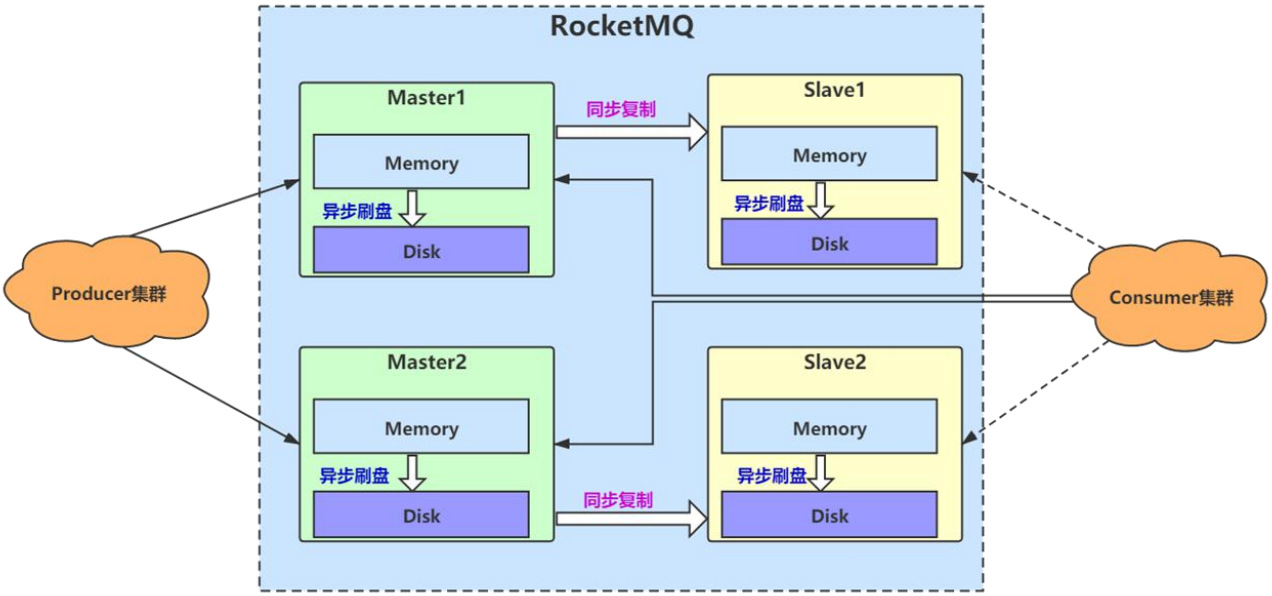
RocketMQ 的集群搭建

RocketMQ 中的高可用机制



RocketMQ 分布式集群是通过 Master 和 Slave 的配合达到高可用性的。

Master 和 Slave 的区别：在 Broker 的配置文件中，参数 brokerId 的值为 0 表明这个 Broker 是 Master，大于 0 表明这个 Broker 是 Slave，同时 brokerRole 参数也会说明这个 Broker 是 Master 还是 Slave。

Master 角色的 Broker 支持读和写，Slave 角色的 Broker 仅支持读，也就是 Producer 只能和 Master 角色的 Broker 连接写入消息；Consumer 可以连接 Master 角色的 Broker，也可以连接 Slave 角色的 Broker 来读取消息。

集群部署模式

1）单 master 模式

也就是只有一个 master 节点，称不上是集群，一旦这个 master 节点宕机，那么整个服务就不可用。

2）多 master 模式

多个 master 节点组成集群，单个 master 节点宕机或者重启对应用没有影响。

优点：所有模式中性能最高（一个 Topic 的可以分布在不同的 master，进行横向拓展）

在多主多从的架构体系下，无论使用客户端还是管理界面创建主题，一个主题都会创建多份队列在多主中（默认是 4 个的话，双主就会有 8 个队列，每台主 4 个队列，所以双主可以提高性能，一个 Topic 的分布在不同的 master，方便进行横向拓展。

缺点：单个 master 节点宕机期间，未被消费的消息在节点恢复之前不可用，消息的实时性就受到影响。

3）多 master 多 slave 异步复制模式

而从节点(Slave)就是复制主节点的数据，对于生产者完全感知不到，对于消费者正常情况下也感知不到。(只有当 Master 不可用或者繁忙的时候，Consumer 会被自动切换到从 Slave 读。)

在多 master 模式的基础上，每个 master 节点都有至少一个对应的 slave。master 节点可读可写，但是 slave 只能读不能写，类似于 mysql 的主备模式。

优点： 一般情况下都是 master 消费，在 master 宕机或超过负载时，消费者可以从 slave 读取消息，消息的实时性不会受影响，性能几乎和多 master 一样。

缺点：使用异步复制的同步方式有可能会有消息丢失的问题。（Master 宕机后，生产者发送的消息没有消费完，同时到 Slave 节点的数据也没有同步完）

4）多 master 多 slave 主从同步复制+异步刷盘（最优推荐）

优点：主从同步复制模式能保证数据不丢失。

缺点：发送单个消息响应时间会略长，性能相比异步复制低 10%左右。

对数据要求较高的场景，主从同步复制方式，保存数据热备份，通过异步刷盘方式，保证 rocketMQ 高吞吐量。

~~5）Dlegder（不推荐）~~

~~在 RocketMQ4.5 版本之后推出了 Dlegder 模式，但是这种模式一直存在严重的 BUG，同时性能有可能有问题，包括升级到了 4.8 的版本后也一样，所以目前不讲这种模式。（类似于 Zookeeper 的集群选举模式）~~

刷盘与主从同步

生产时首先将消息写入到 MappedFile,内存映射文件，然后根据刷盘策略刷写到磁盘。

大致的步骤可以理解成使用 MMAP 中的 MappedByteBuffer 中实际用 flip().

```java
//映射的文件
File file1 = new File(path, child: "1");
//映射文件的fileChannel对象（操作文件，）
FileChannel fileChannel = new RandomAccessFile(file1, mode: "rw").getChannel();
//fileChanne 定义了map方法，MMAP的映射  1k
MappedByteBuffer mmap = fileChannel.map(FileChannel.MapMode.READ_WRITE, position: 0, siz
// 向mmap 写入数据
mmap.put("king".getBytes());
//刷新写入磁盘（刷盘）
mmap.flip();
byte[] bb = new byte[4];
//从mmap中读取文件
mmap.get(bb, offset: 0, length: 4);
System.out.println(new String(bb));
//解除MMAP
unmap(mmap);
```

RocketMQ 的刷盘是把消息存储到磁盘上的，这样既能保证断电后恢复， 又可以让存储的消息量超出内存的限制。RocketMQ 为了提高性能，会尽可能地保证磁盘的顺序写。消息在通过 Producer 写入 RocketMQ 的时候，有两种写磁盘方式，同步刷盘和异步刷盘。

同步刷盘

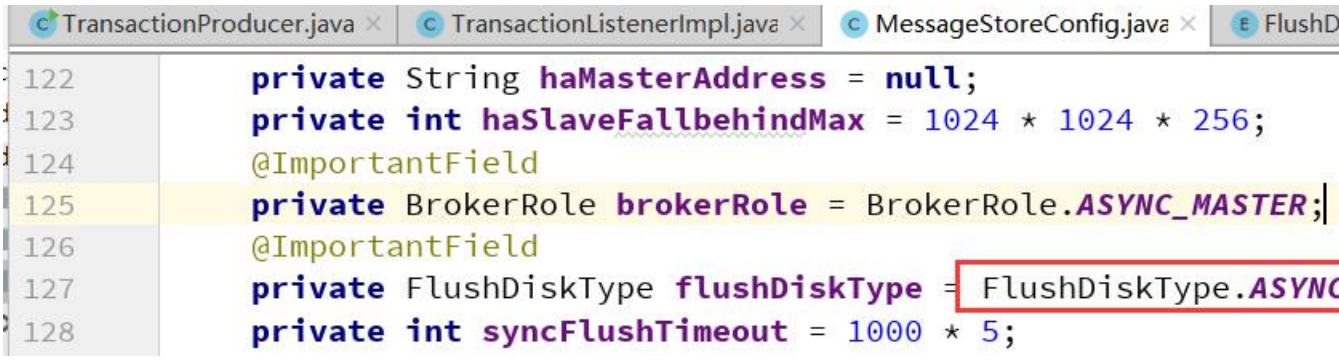
SYNC_FLUSH（同步刷盘）：生产者发送的每一条消息都在保存到磁盘成功后才返回告诉生产者成功。这种方式不会存在消息丢失的问
题，但是有很大的磁盘 IO 开销，性能有一定影响。

异步刷盘

ASYNC_FLUSH（异步刷盘）：生产者发送的每一条消息并不是立即保存到磁盘，而是暂时缓存起来，然后就返回生产者成功。随后再异步的将缓存数据保存到磁盘，有两种情况：1 是定期将缓存中更新的数据进行刷盘，2 是当缓存中更新的数据条数达到某一设定值后进行刷盘。这种异步的方式会存在消息丢失（在还未来得及同步到磁盘的时候宕机），但是性能很好。默认是这种模式。

```
17 brokerId=0
18 deleteWhen=04
19 fileReservedTime=48
20 brokerRole=SYNC_MASTER
21 flushDiskType=ASYNC_FLUSH
22
```

4.8.0 版本中默认值下是异步刷盘，如下图：



主从同步复制

集群环境下需要部署多个 Broker，Broker 分为两种角色：一种是 master，即可以写也可以读，其 brokerId=0，只能有一个；另外一种是 slave，只允许读，其 brokerId 为非 0。一个 master 与多个 slave 通过指定相同的 brokerClusterName 被归为一个 broker set（broker 集）。通常生产环境中，我们至少需要 2 个 broker set。Slave 是复制 master 的数据。一个 Broker 组有 Master 和 Slave，消息需要从 Master 复制到 Slave 上，有同步和异步两种复制方式。

主从同步复制方式（**Sync Broker**）：生产者发送的每一条消息都至少同步复制到一个 slave 后才返回告诉生产者成功，即“同步双写”

在同步复制方式下，如果 Master 出故障， Slave 上有全部的备份数据，容易恢复，但是同步复制会增大数据写入延迟，降低系统吞吐量。

主从异步复制

主从异步复制方式（**Async Broker**）：生产者发送的每一条消息只要写入 master 就返回告诉生产者成功。然后再“异步复制”到 slave。

在异步复制方式下，系统拥有较低的延迟和较高的吞吐量，但是如果 Master 出了故障，有些数据因为没有被写 入 Slave，有可能会丢失；

同步复制和异步复制是通过 Broker 配置文件里的 brokerRole 参数进行设置的，这个参数可以被设置成 ASYNC_MASTER、 SYNC_MASTER、SLAVE 三个值中的一个。

配置参数及意义

brokerId=0 代表主

brokerId=1 代表从（大于 0 都代表从）

brokerRole=SYNC_MASTER 同步复制（主从）

brokerRole=ASYNC_MASTER 异步复制（主从）

flushDiskType=SYNC_FLUSH 同步刷盘

flushDiskType=ASYNC_FLUSH　异步刷盘

搭建双主双从同步复制+异步刷盘

NameServer 集群

106.55.246.66

94.191.83.120

Broker 服务器

106.55.246.66　　------MasterA

94.191.83.120　　------MasterB

106.53.195.121　　------SlaveA

106.55.248.74　　------SlaveB

配置文件

注意，因为 RocketMQ 使用外网地址，所以配置文件(MQ 文件夹/conf/2m-2s-sync/)需要修改(同时修改 nameserver 地址为集群地址)：

注意如果机器内存不够，建议把启动时的堆内存改小，具体见《RocketMQ 的安装.docx》中 --- 3、RocketMQ 在 Linux 下的安装/注意事项

106.55.246.66　　------主 A

broker-a.properties　　增加：　brokerIP1=106.55.246.66

namesrvAddr=106.55.246.66:9876;94.191.83.120:9876

```
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
brokerClusterName=DefaultCluster
brokerName=broker-a
brokerId=0
deleteWhen=04
fileReservedTime=48
brokerRole=SYNC_MASTER
flushDiskType=ASYNC_FLUSH
brokerIP1=106.55.246.66
namesrvAddr=106.55.246.66:9876;94.191.83.120:9876
~
~
```

94.191.83.120　　------主 B

broker-b.properties　　增加：　brokerIP1=94.191.83.120

namesrvAddr=106.55.246.66:9876;94.191.83.120:9876

```
# See the License for the specific language governing permissions and
# limitations under the License.
brokerClusterName=DefaultCluster
brokerName=broker-b
brokerId=0
deleteWhen=04
fileReservedTime=48
brokerRole=SYNC_MASTER
flushDiskType=ASYNC_FLUSH
brokerIP1=94.191.83.120
namesrvAddr=106.55.246.66:9876;94.191.83.120:9876
~
```

106.53.195.121　　------从 A

broker-a-s.properties　　增加：brokerIP1=106.53.195.121

namesrvAddr=106.55.246.66:9876;94.191.83.120:9876

```
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or imp
# See the License for the specific language governing permissions and
# limitations under the License.
brokerClusterName=DefaultCluster
brokerName=broker-a
brokerId=1
deleteWhen=04
fileReservedTime=48
brokerRole=SLAVE
flushDiskType=ASYNC_FLUSH
brokerIP1=106.53.195.121
namesrvAddr=106.55.246.66:9876;94.191.83.120:9876
~
```

106.55.248.74　　------从 B

broker-b-s.properties　　增加：brokerIP1=106.55.248.74

namesrvAddr=106.55.246.66:9876;94.191.83.120:9876

```
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
brokerClusterName=DefaultCluster
brokerName=broker-b
brokerId=1
deleteWhen=04
fileReservedTime=48
brokerRole=SLAVE
flushDiskType=ASYNC_FLUSH
brokerIP1=106.55.248.74
namesrvAddr=106.55.246.66:9876;94.191.83.120:9876
```

不管是主还是从，如果属于一个集群，使用相同的 brokerClusterName 名称

```
# limitations under the License.
brokerClusterName=DefaultCluster
```

启动步骤

启动 NameServer

(记得关闭防火墙或者要开通 9876 端口)

1.启动 NameServer 集群，这里使用 106.55.246.66 和 94.191.83.120 两台作为集群即可。

1） 在机器 A，启动第 1 台 NameServer: 102 服务器进入至‘MQ 文件夹/bin’下：然后执行‘nohup sh mqnamesrv &’

查看日志的命令：tail -f ~/logs/rocketmqlogs/namesrv.log

```
[root@VM-0-16-centos bin]# nohup sh mqnamesrv &
[7] 13224
[root@VM-0-16-centos bin]# nohup: ignoring input and appending output to 'nohup.out'

[root@VM-0-16-centos bin]# tail -f ~/logs/rocketmqlogs/namesrv.log
2021-05-24 15:43:34 INFO main - tls.client.keyPassword = null
2021-05-24 15:43:34 INFO main - tls.client.certPath = null
2021-05-24 15:43:34 INFO main - tls.client.authServer = false
2021-05-24 15:43:34 INFO main - tls.client.trustCertPath = null
2021-05-24 15:43:34 INFO main - Using OpenSSL provider
2021-05-24 15:43:34 INFO main - SSLContext created for server
2021-05-24 15:43:34 INFO main - Try to start service thread:FileWatchService started:false last
l
2021-05-24 15:43:34 INFO NettyEventExecutor - NettyEventExecutor service started
2021-05-24 15:43:34 INFO main - The Name Server boot success. serializeType=JSON
2021-05-24 15:43:34 INFO FileWatchService - FileWatchService service started
```

2） 在机器 B，启动第 2 台 NameServer: 103 服务器进入至‘MQ 文件夹/bin’下：然后执行‘nohup sh mqnamesrv &’

查看日志的命令：tail -f ~/logs/rocketmqlogs/namesrv.log

```
[root@VM-0-6-centos bin]# nohup sh mqnamesrv &
[2] 17384
[root@VM-0-6-centos bin]# nohup: ignoring input and appending output to 'nohup.out'

[root@VM-0-6-centos bin]# tail -f ~/logs/rocketmqlogs/namesrv.log
2021-05-24 15:42:10 INFO main - tls.client.keyPassword = null
2021-05-24 15:42:10 INFO main - tls.client.certPath = null
2021-05-24 15:42:10 INFO main - tls.client.authServer = false
2021-05-24 15:42:10 INFO main - tls.client.trustCertPath = null
2021-05-24 15:42:11 INFO main - Using OpenSSL provider
2021-05-24 15:42:11 INFO main - SSLContext created for server
2021-05-24 15:42:11 INFO main - Try to start service thread:FileWatchService started:false lastT
l
2021-05-24 15:42:11 INFO NettyEventExecutor - NettyEventExecutor service started
2021-05-24 15:42:11 INFO FileWatchService - FileWatchService service started
2021-05-24 15:42:11 INFO main - The Name Server boot success. serializeType=JSON
2021-05-24 15:43:11 INFO NSScheduledThread1 -
```

启动 Broker

2.启动双主双从同步集群，顺序是先启动主，然后启动从。

3）启动主 A:　　102 服务器进入至‘MQ 文件夹/bin’下：执行以下命令（autoCreateTopicEnable=true 测试环境开启，生产环境建议关闭）：

nohup sh mqbroker -c ../conf/2m-2s-sync/broker-a.properties autoCreateTopicEnable=true &

查看日志的命令：tail -f ~/logs/rocketmqlogs/broker.log

```
[root@VM-0-16-centos bin]# tail -f ~/logs/rocketmqlogs/broker.log
2021-05-24 15:45:28 INFO brokerOutApi_thread_3 - register broker[0]to name server 106.55.246.66:9876 O
2021-05-24 15:45:31 WARN brokerOutApi_thread_4 - registerBroker Exception, 94.191.83.120:9876
org.apache.rocketmq.remoting.exception.RemotingConnectException: connect to 94.191.83.120:9876 failed
        at org.apache.rocketmq.remoting.netty.NettyRemotingClient.invokeSync(NettyRemotingClient.java:
) ~[rocketmq-remoting-4.8.0.jar:4.8.0]
        at org.apache.rocketmq.broker.out.BrokerOuterAPI.registerBroker(BrokerOuterAPI.java:193) ~[roc
```

4）启动主 B:　　103 服务器进入至‘MQ 文件夹\bin’下：执行以下命令：

nohup sh mqbroker -c ../conf/2m-2s-sync/broker-b.properties autoCreateTopicEnable=true &

查看日志的命令：tail -f ~/logs/rocketmqlogs/broker.log

5）启动从 A:　　104 服务器进入至‘MQ 文件夹\bin’下：执行以下命令：

nohup sh mqbroker -c ../conf/2m-2s-sync/broker-a-s.properties autoCreateTopicEnable=true &

查看日志的命令：tail -f ~/logs/rocketmqlogs/broker.log

6）启动从 B:　　105 服务器进入至‘MQ 文件夹\bin’下：执行以下命令：

nohup sh mqbroker -c ../conf/2m-2s-sync/broker-b-s.properties autoCreateTopicEnable=true &

查看日志的命令：tail -f ~/logs/rocketmqlogs/broker.log

每台服务器查看日志：tail -f ~/logs/rocketmqlogs/broker.log

如果是要启动控制台，则需要重新打包：

进入‘\rocketmq-console\src\main\resources’文件夹，打开‘application.properties’进行配置。（多个 NameServer 使用;分隔）

rocketmq.config.namesrvAddr=106.55.246.66:9876;94.191.83.120:9876

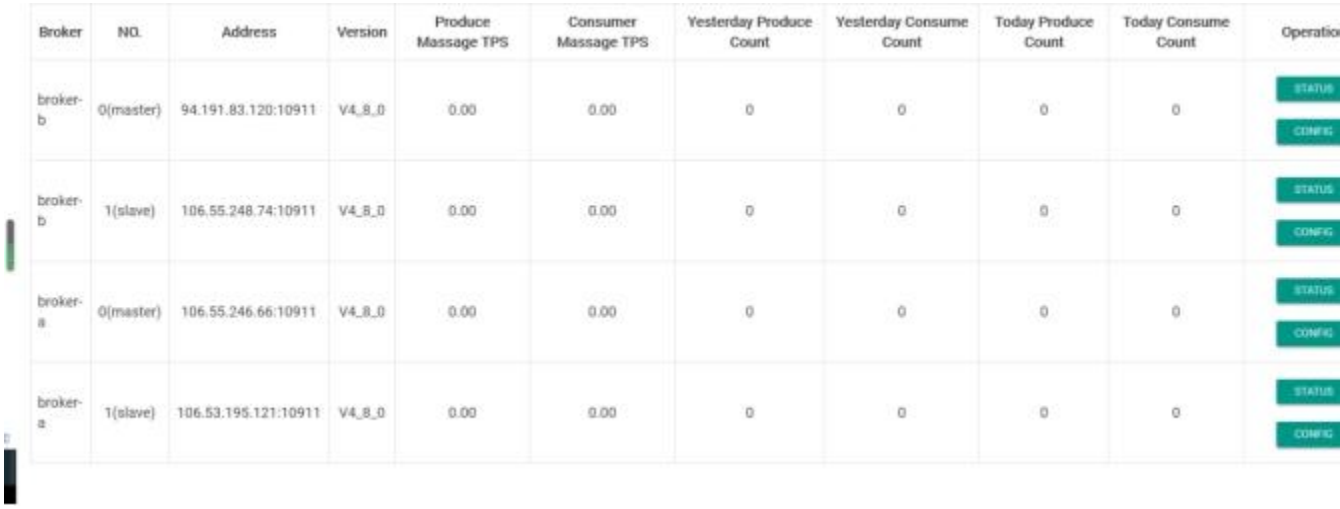
进入‘\rocketmq-externals\rocketmq-console’文件夹，执行‘mvn clean package -Dmaven.test.skip=true’，编译生成。

在把编译后的 jar 包丢上服务器：

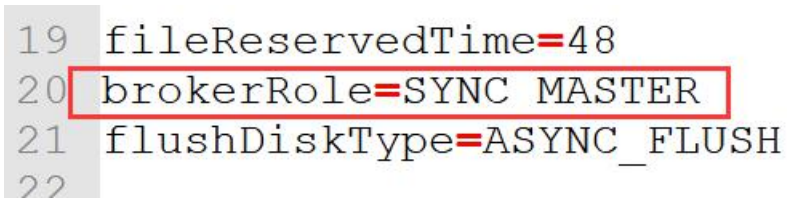
nohup java -jar rocketmq-console-ng-2.0.0.jar &

进入控制台 http://106.55.246.66:8089/#/cluster

集群搭建成功。



Broker	NO.	Address	Version	Produce Massage TPS	Consumer Massage TPS	Yesterday Produce Count	Yesterday Consume Count	Today Produce Count	Today Consume Count	Operatio
broker-b	0(master)	94.191.83.120:10911	V4_8_0	0.00	0.00	0	0	0	0	STATUS CONFIG
broker-b	1(slave)	106.55.248.74:10911	V4_8_0	0.00	0.00	0	0	0	0	STATUS CONFIG
broker-a	0(master)	106.55.246.66:10911	V4_8_0	0.00	0.00	0	0	0	0	STATUS CONFIG
broker-a	1(slave)	106.53.195.121:10911	V4_8_0	0.00	0.00	0	0	0	0	STATUS CONFIG



```
19 fileReservedTime=48
20 brokerRole=SYNC_MASTER
21 flushDiskType=ASYNC_FLUSH
22
```