图文 111 我们的系统与RocketMQ Broker之间是如何进行网络通信的?

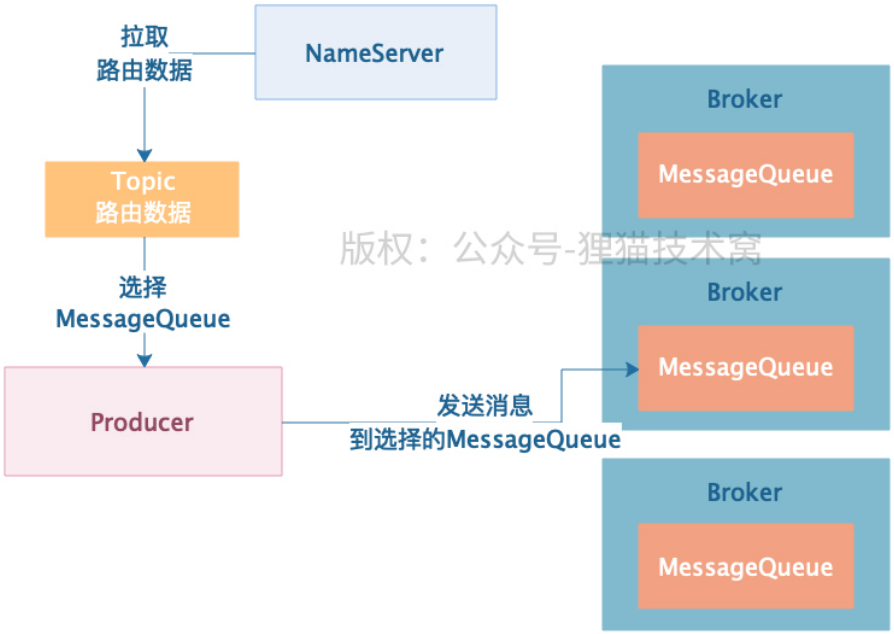
我们的系统与RocketMQBroker之间是如何进行网络通信的?

继《从零开始带你成为JVM实战高手》后，阿里资深技术专家携新作再度出山，重磅推荐：

（点击下方蓝字试听）

《从零开始带你成为MySQL实战优化高手》

上次给大家讲到了我们的Producer是如何从Topic路由数据中选择一个MessageQueue出来的，在选择一个MessageQueue出来之后，接着其实就应该要把消息投递到那个MessageQueue所在的Broker上去了，如下图。



所以今天我们就来看看，Producer是如何把消息发送给Broker的呢?

其实这块代码就在DefaultMQProducerImpl.sendDefaultImpl()方法中，在这个方法里，先是获取到了MessageQueue所在的broker名称，如下源码片段：

```
brokersSent[times] = mq.getBrokerName();
```

获取到了这个brokerName之后，接着其实就可以使用如下的代码把消息投递到那个Broker上去了，看下面的代码片段：

```
sendResult = this.sendKernelImpl(msg, mq, communicationMode, sendCallback, topicPublishInfo, timeout - costTime);
```


所以今天我们重点来看看这个代码里是如何把消息投递出去，在这个方法里，先有一些较为简单的逻辑，给大家看一下，如下所示：

```
String brokerAddr = this.mQClientFactory
        .findBrokerAddressInPublish(mq.getBrokerName());

if (null == brokerAddr) {
    tryToFindTopicPublishInfo(mq.getTopic());
    brokerAddr = this.mQClientFactory
         .findBrokerAddressInPublish(mq.getBrokerName());
}
```

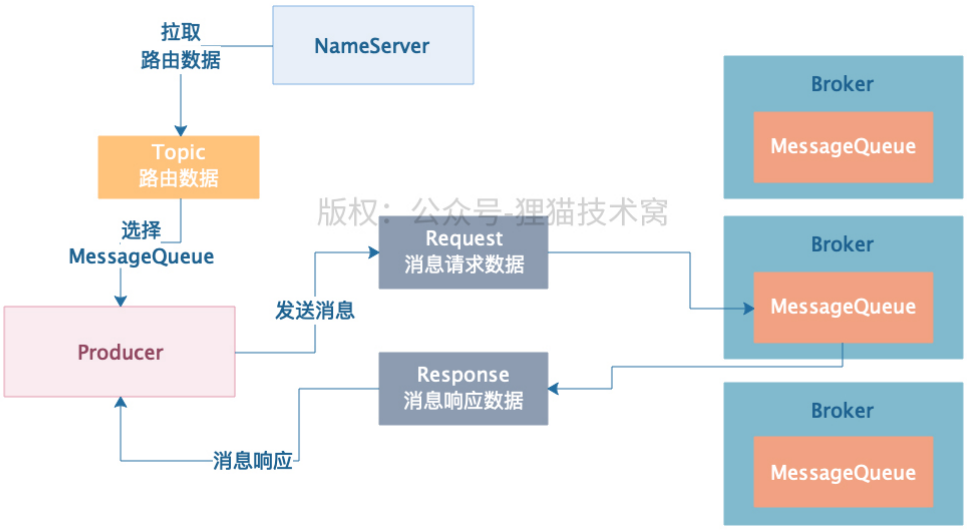
上面的代码片段其实非常简单，就是通过brokerName去本地缓存找他的实际的地址，如果找不到，就去找NameServer拉取Topic的路由数据，然后再次在本地缓存获取broker的实际地址，你有这个地址了，才能给人家进行网络通信。

接下来的源码就很繁琐细节了，其实大家不用看也行，他就是用自己的方式去封装了一个Request请求出来，这里涉及到了各种信息的封装，包括了请求头，还有一大堆所有你需要的数据，都封装在Request里了。

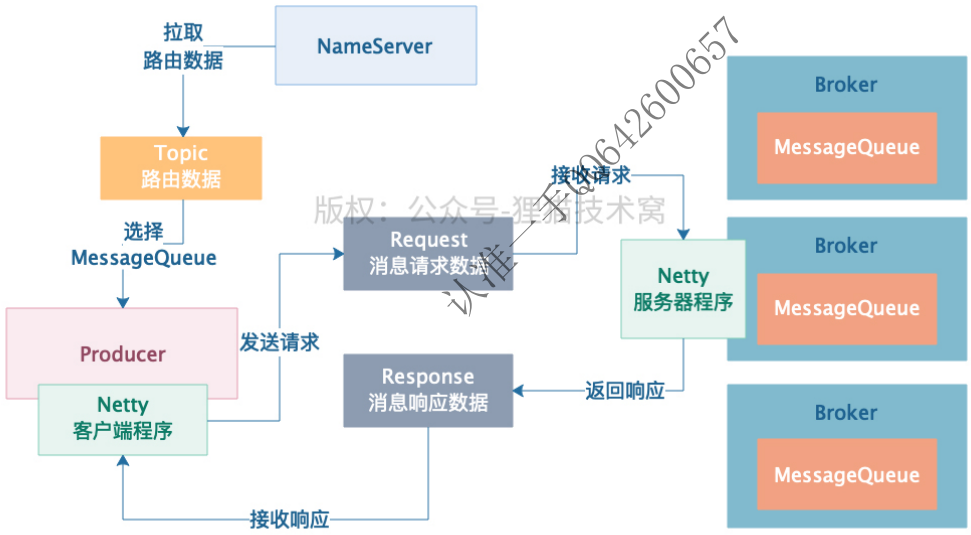
他在这里做的事情，大体上包括了给消息分配全局唯一ID、对超过4KB的消息体进行压缩，在消息Request中包含了生产者组、Topic名称、Topic的MessageQueue数量、MessageQueue的ID、消息发送时间、消息的flag、消息扩展属性、消息重试次数、是否是批量发送的消息、如果是事务消息则带上prepared标记，等等。

总之，这里就是封装了很多很多的数据就对了，这些东西都封装到一个Request里去，然后在底层还是通过Netty把这个请求发送出去，发送到指定的Broker上去就可以了

这里Producer和Broker之间都是通过Netty建立长连接，然后基于长连接进行持续的通信的，如下图所示。



其实对于我们而言，如果大家想要研究更加细致的源码细节，可以找到我说的那块代码，然后自己仔细的分析里面如何封装Request请求的细节，包括底层的基于Netty发送请求出去的细节，但是如果你不想看这些细节，那么从原理层面而言，你只要知道这个过程就可以了，另外比较重要的就是知道他底层是基于Netty发送的，如下图。



对于Producer投递消息到Broker这个过程，大家了解到这里其实就可以了，接下来明天我们就应该继续去分析Broker通过Netty服务器接收到消息之后，在底层是如何进行处理的

这个过程将会比较复杂，涉及到CommitLog、ConsumeQueue、IndexFile、Checkpoint等一系列的机制，也是RocketMQ最核心的一块机制。

End

- 《互联网Java工程师面试突击》（第3季）

重要说明：

- 如何提问：每篇文章都有评论区，大家可以尽情留言提问，我会逐一答疑
- 如何加群：购买狸猫技术窝专栏的小伙伴都可以加入狸猫技术交流群，一个非常纯粹的技术交流的地方

  具体加群方式，请参见目录菜单下的文档：《付费用户如何加群》（**购买后可见**）

返回
前进
重新加载
打印