**说明：本次测试，使用kafaka，activityMQ，rabbitMQ消息中间件进行对比，均采用一个消息队列，测试中间件在收发消息时时延。**

**测试前置条件：**

1. 消费者端只配置一个消费者来消费数据。
2. 多线程才用jmeter通过http请求来进行测试。
3. actitvityMQ在1000线程100次时，时延太大，受制于电脑性能未测试。

**测试结果：**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 单位（ms） | | 单线程调用1000 | 单线程调用5000 | 单线程调用10000 | 100线程100次 | 500线程100次 | 1000线程100次 |
| 消息发送时延 | activityMQ | 94997 | 342142 | 869012 | 869841.38 | 1328734.291 | -- |
| kafaka | 6 | 40 | 147 | 1.48 | 1.916 | 1.991 |
| rabbitMQ | 62 | 358 | 1121 | 377.08 | 4765.66 | 6492.54 |
| 消息处理时延平均值 | activityMQ | 68.007 | 38.0308 | 86.671 | 14571.9765 | 29567.1105 | -- |
| kafaka | 141.459 | 187.4012 | 183.4737 | 48.1963 | 1614.25334 | 2257.62926 |
| rabbitMQ | 203.908 | 844.3368 | 1489.2252 | 1453.5839 | 7825.11326 | 18138.44642 |
| 消息处理时延最大值 | activityMQ | 1153 | 575 | 7774 | 29098 | 43711 | -- |
| kafaka | 168 | 227 | 229 | 153 | 2398 | 3128 |
| rabbitMQ | 328 | 1476 | 2670 | 2868 | 19267 | 41169 |
| 消息处理时延最小值 | activityMQ | 10 | 8 | 9 | 847 | 219 | -- |
| kafaka | 76 | 61 | 25 | 5 | 67 | 117 |
| rabbitMQ | 8 | 30 | 62 | 42 | 59 | 137 |

消息处理时延

消息发送时延

**结论：**从结果上来看，在消息发送方面，kafak的效率较高，rabbitMQ次之，activityMQ最慢；在消息处理时延方面单线程情况下activityMQ的效率较高，kafak次之，rabbitMQ最慢，但考虑到activityMQ发送较慢，队列中堆积消息少，故处理时延较小，参考性不强；在多线程方面kafak处理较快，rabbitMQ次之，activityMQ最慢。在消息可靠性方面在测试10万条消息时均未发现消息丢失问题，但阅读资料kafak会有少量丢消息的情况存在。结合业务方面考虑，在需要消息可靠保证的需求可以采用rabbitMQ，例如：计划下达生产，控投等业务。对于吞吐量高，但对于可靠性要求不是太高的可以采用kafak，例如记录日志的消息。