RabbitMQ,遵循AMQP协议，由内在高并发的erlanng语言开发，用在实时的对可靠性要求比较高的消息传递上。

kafka是Linkedin于2010年12月份开源的消息发布订阅系统,它主要用于处理活跃的流式数据,大数据量的数据处理上。

翻阅了一些资料，kafka一般是不会丢失消息数据的，只有在以下特殊情况下才会出现数据丢失的情况：

1,网络异常

acks设置为0时，不和Kafka集群进行消息接受确认，当网络发生异常等情况时，存在消息丢失的可能；

2,客户端异常

异步发送时，消息并没有直接发送至Kafka集群，而是在Client端按一定规则缓存并批量发送。在这期间，如果客户端发生死机等情况，都会导致消息的丢失；

3,缓冲区满了

异步发送时，Client端缓存的消息超出了缓冲池的大小，也存在消息丢失的可能；

4,Leader副本异常

acks设置为1时，Leader副本接收成功，Kafka集群就返回成功确认信息，而Follower副本可能还在同步。这时Leader副本突然出现异常，新Leader副本(原Follower副本)未能和其保持一致，就会出现消息丢失的情况；

以上就是消息丢失的几种情况，在日常应用中，我们需要结合自身的应用场景来选择不同的配置。

想要更高的吞吐量就设置：异步、ack=0；

想要不丢失消息数据就选：同步、ack=-1策略

Kafka消息丢失没有找到明确的数据信息来支撑（一般不会这么丢失啦，可以通过一些降低处理性能的配置手段来提供消息的更高可靠新）。

RabbitMQ一般情况很少丢失，但是不能排除意外，在某些特定场景下，也会发生数据丢失，如：

消息发送丢失：producer -> mq

| 序号 | Producer | MQ | 数据丢失说明 |
| --- | --- | --- | --- |
| 1 | 发送数据失败 | MQ服务宕机 | Producer未做数据本地化导致数据丢失 |
| 2 | 发送数据成功 | MQ服务良好 | 由于某种不良因素，MQ未收到消息，但Producer仍以为本身发送数据成功，导致数据丢失 |

消息持久化丢失：

在MQ的消息持久化的过程中，即写盘（数据从内存到磁盘）的过程中，MQ宕机导致的数据丢失

消息消费丢失：

自动ack导致的数据丢失

业务处理异常导致的数据丢失

为了保证我们自己系统高可用，我们必须作出更好完善措施，保证系统的稳定性。

下面来介绍下，如何保证消息的绝对不丢失的问题，下面分享的绝对干货，都是在知名互联网产品的产线中使用。

1.消息持久化

2.ACK确认机制

3.设置集群镜像模式

4.消息补偿机制