PKI（公匙基础设施）：采用证书管理公匙，通过第三方可信任的机构（认证中心CA），把用户的公匙和标识信息捆绑在一起并在网络上验证用户的身份，PKI作用提供公匙加密数字签名，管理密匙和证书

PKI系统的组成：X.509格式的证书和证书撤销列表；CA系统；RA（注册机构）

证书：是用来证明实体身份和公匙绑定关系的电子文档（将公匙与对应密匙的持有者的身份信息如姓名电子邮箱相绑定）

数字化证书：由CA机构颁发的证书遵循X.509 V3标准

完整的PKI系统包括：证书颁发，证书的更新，证书废除，证书认证，证书在线查询，证书撤销列表（CRL）的公布

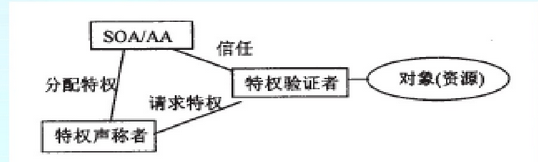
PMI（授权管理基础设施）：提供了分布式计算环境的访问控制功能，将访问控制机制从应用系统中分离出来

PKI与PMI体系当中的两种证书

公匙证书PKC：（由CA签名的证书）

属性证书AC：由PMI机构（AA权威属性机构）颁发的实体与其享有的权利捆绑在一起的数据结构

PMI（基本特权管理模型）的组成：对象特权声称者，特权验证者，SOA/AA



SOA：授权管理体系的中心业务节点相当于PKI中的跟CA

AA：授权管理中心

对象：被保护的资源

非对称加密算法：非对称加密算法需要两个密钥：[公开密钥](https://baike.baidu.com/item/%E5%85%AC%E5%BC%80%E5%AF%86%E9%92%A5)（publickey）和私有密钥（privatekey）。公开密钥与私有密钥是一对，如果用公开密钥对数据进行加密，只有用对应的私有密钥才能解密；如果用私有密钥对数据进行加密，那么只有用对应的公开密钥才能解密

对称加密算法：在对称加密算法中，数据发信方将明文（[原始数据](https://baike.baidu.com/item/%E5%8E%9F%E5%A7%8B%E6%95%B0%E6%8D%AE)）和加密[密钥](https://baike.baidu.com/item/%E5%AF%86%E9%92%A5)（mi yao）一起经过特殊加密算法处理后，使其变成复杂的加密密文发送出去。收信方收到密文后，若想解读原文，则需要使用加密用过的密钥及相同算法的逆算法对密文进行解密，才能使其恢复成可读明文。在对称加密算法中，使用的密钥只有一个，发收信双方都使用这个密钥对数据进行加密和解密，这就要求解密方事先必须知道加密密钥

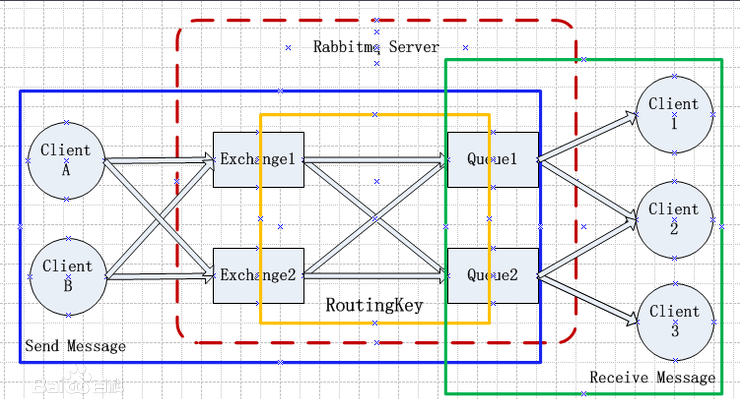
RabbitMQ

MQ全称为Message Queue, [消息队列](https://baike.baidu.com/item/%E6%B6%88%E6%81%AF%E9%98%9F%E5%88%97)（MQ）是一种应用程序对应用程序的通信方法。应用程序通过读写出入队列的消息（针对应用程序的数据）来通信，而无需专用连接来链接它们。排队指的是应用程序通过队列来通信。队列的使用除去了接收和发送应用程序同时执行的要求，异步消息交互。其中较为成熟的MQ产品有IBM WEBSPHERE MQ等等

MQ是消费-生产者模型的一个典型的代表，一端往[消息队列](https://baike.baidu.com/item/%E6%B6%88%E6%81%AF%E9%98%9F%E5%88%97" \t "_blank)中不断写入消息，而另一端则可以读取或者订阅队列中的消息。MQ和[JMS](https://baike.baidu.com/item/JMS" \t "_blank)类似，但不同的是JMS是SUN JAVA[消息中间件](https://baike.baidu.com/item/%E6%B6%88%E6%81%AF%E4%B8%AD%E9%97%B4%E4%BB%B6)服务的一个标准和API定义，而MQ则是遵循了AMQP协议的具体实现和产品。

RabbitMQ是一个在[AMQP](https://baike.baidu.com/item/AMQP" \t "_blank)基础上完成的，可复用开源的消息系统，用erlang语言开发。

几个重要的概念



Broker：简单来说就是消息队列服务器实体；

vhost：虚拟主机，一个broker里可以开设多个vhost，用作不同用户的权限分离。

producer：消息生产者，就是投递消息的程序

consumer：消息消费者，就是接受消息的程序

channel：消息通道，在客户端的每个连接里，可建立多个channel，每个channel代表一个会话任务。

Exchange:交换机，决定了消息路由规则，它指定消息按什么规则，路由到哪个队列；

Queue:消息队列消息的载体，每个消息被投入到一个或多个队列；

Binding：绑定，它的作用就是把exchange和queue按照路由规则绑定起来。

Routing Key：路由关键字，exchange根据这个关键字进行消息投递

Channel:进行消息读写的通道；

Bind:绑定了Queue和Exchange，意即为符合什么样路由规则的消息，将会放置入哪一个[消息队列](https://baike.baidu.com/item/%E6%B6%88%E6%81%AF%E9%98%9F%E5%88%97" \t "_blank)；

消息队列的使用过程如下：

客户端连接到消息队列服务器打开一个channel🡪客户端声明一个exchange设置相关属性🡪客户端声明一个queue设置相关属性🡪客户端使用routing key，在exchange和queue之间建立好绑定关系🡪客户端投递消息到exchange。

exchange接收到消息后，就根据消息的key和已经设置的binding，进行消息路由，将消息投递到一个或多个队列里。

exchange也有几个类型

完全根据key进行投递的叫做Direct交换机，例如，绑定时设置了routing key为”abc”，那么客户端提交的消息，只有设置了key为”abc”的才会投递到队列。

对key进行模式匹配后进行投递的叫做Topic交换机，

符号”#”匹配一个或多个词，

符号”\*”匹配正好一个词。

例如”abc.#”匹配”abc.def.ghi”，”abc.\*”只匹配”abc.def”。还有一种不需要key的，叫做Fanout交换机，它采取广播模式，一个消息进来时，投递到与该交换机绑定的所有队列。

RabbitMQ支持消息的持久化，也就是数据写在磁盘上，消息队列持久化包括3个部分：  
　　（1）exchange持久化，在声明时指定durable => 1  
　　（2）queue持久化，在声明时指定durable => 1  
　　（3）消息持久化，在投递时指定delivery\_mode => 2（1是非持久化）

如果exchange和queue都是持久化的，那么它们之间的binding也是持久化的。如果exchange和queue两者之间有一个持久化，一个非持久化，就不允许建立绑定。