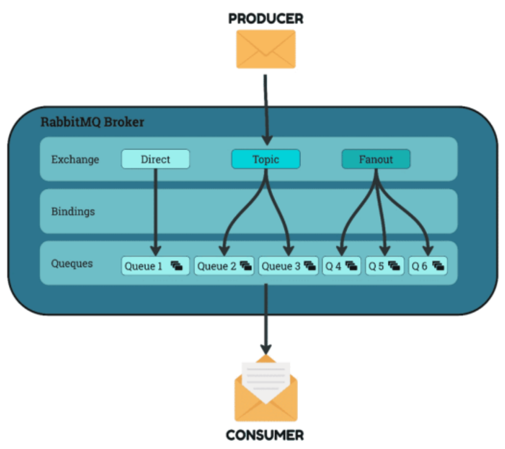
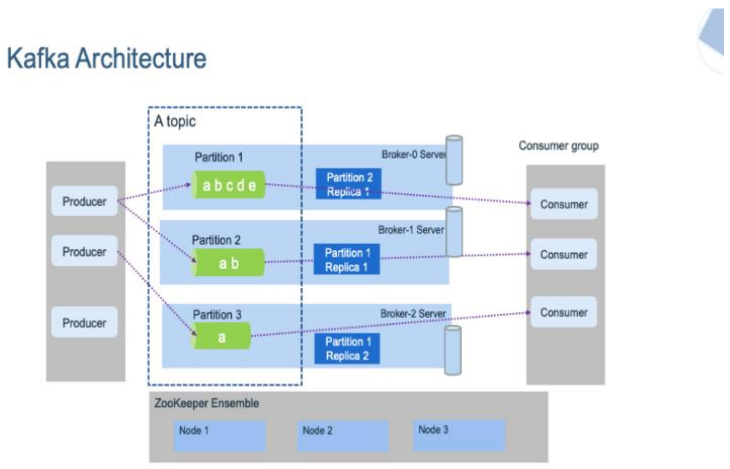
## 2) **Differences between RabbitMQ and Kafka**





| | |
|---|---|
| RabbitMQ is a in memory message broker like ActiveMQ or AmazonSQS designed for complex message routing.<br><br>RabbitMQ is based on AMQP(Advanced Message Queueing message model.<br><br>RabbitMQ supports a wide variety of languages and legacy protocols. For example JavaScript, Go, C, Swift, Spring, Elixir, PHP and .NET.<br><br>Supports message priority.<br><br>In RabbitMQ, a broker guarantees consumers to receive messages. The consumer application takes on a passive role and waits for the RabbitMQ broker to send the message to the queue.<br><br>The producer forwards the message to the exchange first, rather than sending it directly to the queue.<br><br>Queues are connected to the consumer by a binding key. | Kafka is a log-based message broker like Amazon Kinesis designed for real-time, high-throughput stream processing.<br><br>Kafka supports Python and Node.js additionally.<br><br>Kafka has limited programming language options. It uses a binary protocol over TCP for data transmission.<br><br>There is no message priority.<br><br>Kafka consumers are more proactive in reading and tracking information. As messages are appended to physical log files, Kafka consumers track the last message they have read and update their offset monitors accordingly. The offset tracker is an increasing counter after reading a message. In Kafka, producers are unaware of messages consumed by consumers.<br><br>Kafka uses topics and partitions to sequence messages. When a producer publishes a message, it goes to a specific topic and partition. Since Kafka does not support direct producer-consumer |

| | |
|---|---|
| If there is no message with a higher priority in the queue, RabbitMQ sends and queues the messages in a specific order. This specific order means that consumers receive the messages in the order they were sent.<br><br>The RabbitMQ broker routes the message to the target queue. After being read, the consumer sends an acknowledgment (ACK) response to the broker, which then deletes the message from the queue.<br><br>It has low latency. It sends thousands of messages per second.<br><br>Consumer has the control, not the message broker, they define the message metadata.<br><br>It has good security supports FASL, LDAP and TLS.<br><br>RabbitMQ supports a wide variety of languages and legacy protocols. | exchanges, consumers receive messages from partitions in a different order.<br><br>Kafka appends the message to a log file that will be retained until its retention period expires. This way, consumers can reprocess streaming data at any time within the specified retention period.<br><br>Kafka has a better message passing capacity and provides a real-time transmission of up to millions of messages per second.<br><br>Apache Kafka architecture provides secure event streams with TLS and Java Authentication and Authorization Service (JAAS). |