**Monitoring Requests on the Broker**

**Monitoring Requests on the Broker** shows how you can **monitor Kafka request performance** using **JMX metrics** on the Kafka broker.

It provides a visual breakdown of the key stages a request (like produce or fetch) goes through and the **corresponding metrics** available for each stage.

**Purpose of the Diagram**

To help you understand **where time is spent** during a Kafka request and how to monitor these timings using **Kafka JMX metrics** like:

* RequestQueueTimeMs
* LocalTimeMs
* RemoteTimeMs
* ResponseQueueTimeMs
* ResponseSendTimeMs

**Flow & Monitoring Points**

**1. Kafka Producer Sends a Request**

* The **Kafka Producer** sends a request (e.g., a produce request) to the **Kafka Broker** over the **network**.
* **Metric:** ResponseSendTimeMs
  + Time taken to **send the response back** to the producer after the request is processed.

**2. Network Threads Handle the Request**

* The broker’s **Network Threads** receive the request and place it in the **Request Queue**.

**3. Request Queue**

* The request waits in the **Request Queue** before being picked up by a worker thread.
* **Metric:** RequestQueueTimeMs
  + Measures how long the request waited in the queue.

**4. Worker Threads / IO Threads**

* These threads pick up the request and interact with the **Page Cache** to write (produce) or read (fetch) data.

**5. Page Cache**

* Kafka uses the **OS-level page cache** for efficient disk I/O.
* **Metric:** LocalTimeMs
  + Time spent performing local disk I/O (e.g., writing or reading data).

**6. Purgatory Map**

* Kafka uses a **Purgatory system** when requests must wait on additional conditions:
  + **Produce:** Wait for replica acknowledgments.
  + **Fetch:** Wait for enough data to accumulate (fetch.min.bytes).
* If these conditions are met, the request continues.

**7. Kafka Brokers / Replica Coordination**

* If remote replicas are involved (e.g., acknowledgments for a produce request), Kafka waits for them to respond.
* **Metric:** RemoteTimeMs
  + Time waiting for other brokers/replicas to respond.

**8. Response Queue**

* Once complete, the response is placed into the **Response Queue**.
* **➡️ Metric:** ResponseQueueTimeMs
  + Time the response waits before being sent back by the network thread.

**9. Response Sent Back**

* Finally, the network thread sends the response to the producer.
* This completes the lifecycle of the request.

**Kafka Request JMX Metrics Summary**

| **Metric** | **Description** |
| --- | --- |
| **RequestQueueTimeMs** | Time spent in the request queue |
| **LocalTimeMs** | Time spent on local disk I/O (e.g., Page Cache) |
| **RemoteTimeMs** | Time spent waiting for remote replicas or consumers |
| **ResponseQueueTimeMs** | Time the response waits before being sent |
| **ResponseSendTimeMs** | Time taken to send the response back to the client |

**Why This Matters**

These metrics help you **troubleshoot performance** issues on brokers. For example:

* High RequestQueueTimeMs? You might need more IO threads.
* High RemoteTimeMs? Possible replica lag or network delay.
* High LocalTimeMs? Disk I/O bottleneck.