

## WEEK-5 (Stress testing)

JMX exporter setup:

Step 1: create jmx directory

```
sudo mkdir -p /opt/jmx-exporter
sudo chown $USER:$USER /opt/jmx-exporter
cd /opt/jmx-exporter
```

Step 2: download jmx

```
wget
https://repo1.maven.org/maven2/io/prometheus/jmx/jmx_prometheus_javaagent/1.0.1/jmx_prometheus_javaagent-1.0.1.jar
```

Step 3: update kafka start command:

```
KAFKA_HEAP_OPTS="-Xmx512M -Xms512M" export KAFKA_OPTS="$KAFKA_OPTS -
javaagent:/opt/jmx-exporter/jmx_prometheus_javaagent-
1.0.1.jar=7071:/opt/jmx-exporter/kafka-jmx-config.yaml" bin/kafka-
server-start.sh config/server.properties
```

### Core Kafka JMX Metrics (via JMX Exporter)

#### 1. Log End Offset

Tracks the latest offset in a partition (i.e., how large each partition is getting):

```
kafka.log:type=Log,name=LogEndOffset
```

#### Why it's important:

More log segments = more disk usage. You can use this to estimate growth rate.

#### 2. Messages In Per Second

Rate of incoming messages:

```
kafka.server:type=BrokerTopicMetrics,name=MessagesInPerSec
```

### **Why it's important:**

High message throughput = faster disk consumption.

### **3. Bytes In/Out Per Second**

Useful for understanding I/O and topic traffic:

```
kafka.server:type=BrokerTopicMetrics,name=BytesInPerSec
```

```
kafka.server:type=BrokerTopicMetrics,name=BytesOutPerSec
```

### **4. Under-Replicated Partitions**

If disk fills up, Kafka might fail to replicate data:

```
kafka.server:type=ReplicaManager,name=UnderReplicatedPartitions
```

### **5. Log Flush Latency**

Log flushing delays can mean I/O issues due to disk pressure:

```
kafka.log:type=LogFlushStats,name=LogFlushRateAndTimeMs
```

### **6. Controller State / Offline Partitions**

To detect if Kafka is starting to fail due to low disk:

Step 4: To export these metrics out of kafka , create kafka-jmx-config.yaml file:

```
startDelaySeconds: 0
```

```
rules:
```

- pattern: 'kafka.server<type=(.+), name=(.+)><>Count' name: kafka\_server\_\$1\_\$2\_total type: COUNTER
- pattern: 'kafka.server<type=(.+), name=(.+)><>OneMinuteRate' name: kafka\_server\_\$1\_\$2\_1m\_rate type: GAUGE
- pattern: 'kafka.server<type=(.+), name=(.+)><>MeanRate' name: kafka\_server\_\$1\_\$2\_mean\_rate type: GAUGE
- pattern: 'kafka.server<type=(.+), name=(.+)><>Value' name: kafka\_server\_\$1\_\$2 type: GAUGE labels: kafka\_server: "\$1"
- pattern: 'kafka.log<type=Log, name=(.+), topic=(.+), partition=(.)><>Value' name: kafka\_log\_\$1 type: GAUGE labels: topic: "\$2" partition: "\$3"

Step 5: Start the kafka broker now.

The jmx exporter should ship the metrics in port 7071.

```
kafka_server_BrokerTopicMetrics_InvalidOffsetOrSequenceRecordsPerSec_mean_rate 0.0
# HELP kafka_server_BrokerTopicMetrics_MessagesInPerSec_total Kafka server metric count for BrokerTopicMetrics MessagesInPerSec
# TYPE kafka_server_BrokerTopicMetrics_MessagesInPerSec_total counter
kafka_server_BrokerTopicMetrics_MessagesInPerSec_total 64.0
# HELP kafka_server_BrokerTopicMetrics_MessagesInPerSec_1m_rate Kafka server 1-minute rate for BrokerTopicMetrics MessagesInPerSec
# TYPE kafka_server_BrokerTopicMetrics_MessagesInPerSec_1m_rate gauge
kafka_server_BrokerTopicMetrics_MessagesInPerSec_1m_rate 1.023431492744662
# HELP kafka_server_BrokerTopicMetrics_MessagesInPerSec_mean_rate Kafka server mean rate for BrokerTopicMetrics MessagesInPerSec
# TYPE kafka_server_BrokerTopicMetrics_MessagesInPerSec_mean_rate gauge
kafka_server_BrokerTopicMetrics_MessagesInPerSec_mean_rate 0.21443547911713068
# HELP kafka_server_BrokerTopicMetrics_MessagesInPerSec_topic_node_metrics_total Kafka server metric count for BrokerTopicMetrics Message
# TYPE kafka_server_BrokerTopicMetrics_MessagesInPerSec_topic_node_metrics_total counter
kafka_server_BrokerTopicMetrics_MessagesInPerSec_topic_node_metrics_total 64.0
# HELP kafka_server_BrokerTopicMetrics_MessagesInPerSec_topic_node_metrics_1m_rate Kafka server 1-minute rate for BrokerTopicMetrics Message
```

1) You can configure Prometheus to scrape this metrics from port 7071:

In prometheus.yml:

```
scrape_configs:
- job_name: 'kafka'
  static_configs:
    - targets: ['<kafka-host>:7071']
```

2) Reload Prometheus after editing.

Test:

Visit <http://<prometheus-host>:9090> and check for metric like kafka\_log\_remaining\_bytes.

3) Make sure kafka broker and zookeeper is running and a producer is producing messages.

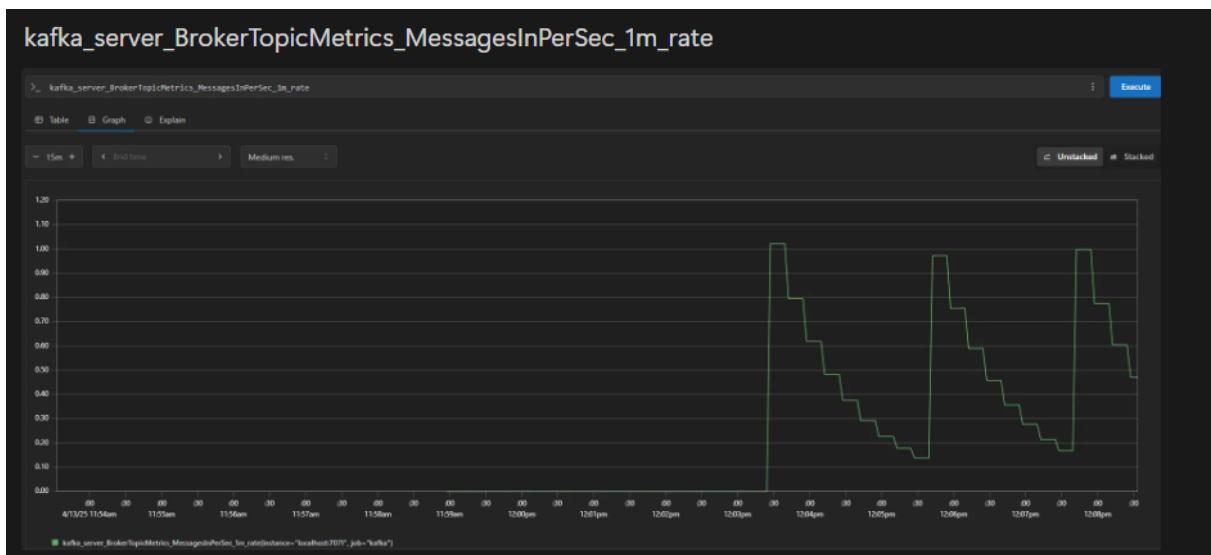
### Stress-testing (High CPU Load):

1) We will be using the stress-ng package.

Download stress-ng package by:

```
Sudo apt-install stress-ng
```

2) First analyse metrics over Prometheus before running the stress test:



### 3) Run stress testing (let's flood the cpu load upto 70%-80%)

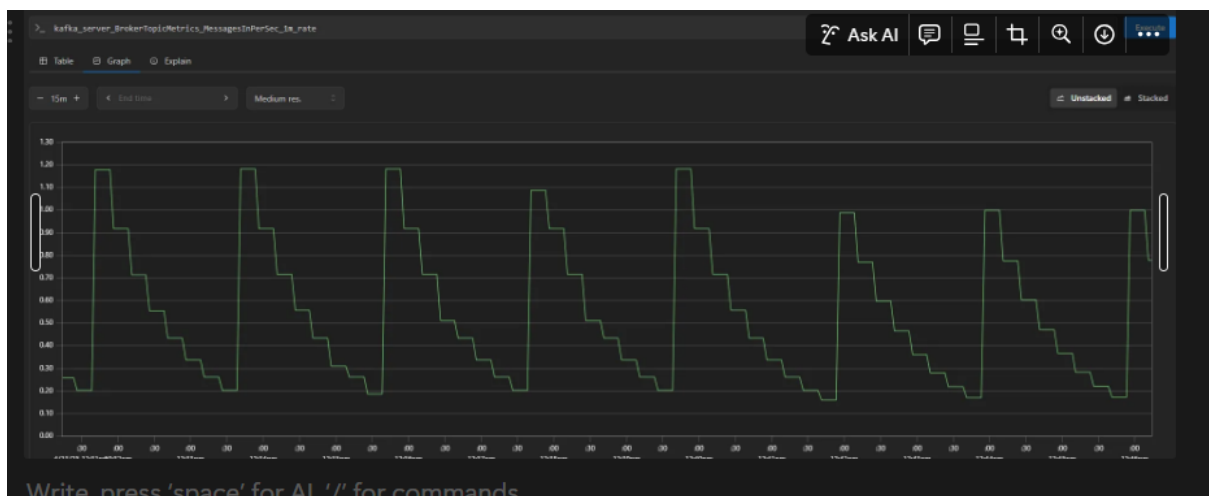
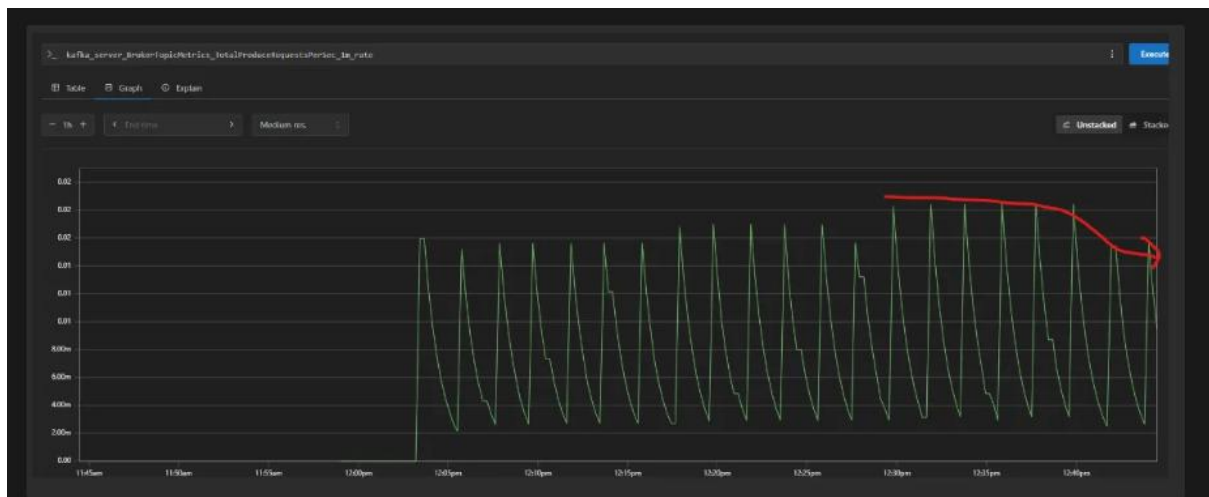
**Command:** `stress --cpu 4 --timeout 600`

This spawns **4 CPU workers** on a **2-core system**, creating **fierce contention** and pushing Kafka's JVM threads to the edge.

JVM threads will fight for CPU time, causing request handler timeouts, GC delays, lag in producers/consumers, or even ZooKeeper timeouts.

Results:

Decreased rate of message requests produced per sec.



#### 4)Extreme Stress testing

##### Using stress-ng

```
stress-ng --cpu 6 --vm 2 --vm-bytes 90% --timeout 600s
stress-ng \
--cpu 6 \
--cpu-method matrixprod \
--vm 2 \
--vm-bytes 90% \
--hdd 2 \
--io 4 \
--timeout 600s \
--metrics-brief
```

Check memory usage using command : top

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3060	ubuntu	20	0	283676	11108	1280	R	24.9	1.2	0:05.81	stress-+
3061	ubuntu	20	0	283676	11108	1280	R	24.9	1.2	0:05.72	stress-+
3062	ubuntu	20	0	283676	11108	1280	R	24.9	1.2	0:05.73	stress-+
3063	ubuntu	20	0	283676	10980	1280	R	24.9	1.2	0:05.79	stress-+
3064	ubuntu	20	0	283676	10980	1280	R	24.9	1.2	0:05.75	stress-+
3065	ubuntu	20	0	283676	11108	1280	R	24.9	1.2	0:05.75	stress-+
3066	ubuntu	20	0	283676	11108	1280	R	24.9	1.2	0:05.78	stress-+
3067	ubuntu	20	0	283676	11108	1280	R	24.9	1.2	0:05.73	stress-+
1532	ubuntu	20	0	2929892	395880	15176	S	0.3	42.3	2:02.90	java
1	root	20	0	22188	8132	4420	S	0.0	0.9	0:01.62	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pool_wor
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker+
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker+

Tasks: 127 total, 10 running, 117 sleeping, 0 stopped, 0 zombie  
%Cpu(s): 71.6 us, 13.6 sy, 0.0 ni, 0.0 id, 2.3 wa, 0.0 hi, 0.0 si, 12.4 st  
MiB Mem : 914.1 total, 63.0 free, 910.4 used, 76.8 buff/cache  
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 3.7 avail Mem

As you can see, we are almost at 98-99% of memory usage, 910 MiB used out of 914 Mib

## Results:



Prometheus starts breaking, unable to capture metrics of kafka.

Behaviour on kafka broker:

```
[2025-04-13 13:31:29,884] INFO [ThrottledChannelReaper-Request]: Stopped (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2025-04-13 13:31:29,884] INFO [ThrottledChannelReaper-Request]: Shutdown completed (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2025-04-13 13:31:29,884] INFO [ThrottledChannelReaper-ControllerMutation]: Shutting down (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2025-04-13 13:31:29,890] INFO [ThrottledChannelReaper-ControllerMutation]: Shutdown completed (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2025-04-13 13:31:29,893] INFO [SocketServer listenerType=ZK_BROKER, nodeId=0] Shutting down socket server (kafka.network.SocketServer)
[2025-04-13 13:31:29,893] INFO [ThrottledChannelReaper-ControllerMutation]: Stopped (kafka.server.ClientQuotaManager$ThrottledChannelReaper)
[2025-04-13 13:31:30,245] INFO [SocketServer listenerType=ZK_BROKER, nodeId=0] Shutdown completed (kafka.network.SocketServer)
[2025-04-13 13:31:30,267] INFO Metrics scheduler closed (org.apache.kafka.common.metrics.Metrics)
[2025-04-13 13:31:30,293] INFO Closing reporter org.apache.kafka.common.metrics.JmxReporter (org.apache.kafka.common.metrics.Metrics)
[2025-04-13 13:31:30,293] INFO Metrics reporters closed (org.apache.kafka.common.metrics.Metrics)
[2025-04-13 13:31:30,311] INFO Broker and topic stats closed (kafka.server.BrokerTopicStats)
[2025-04-13 13:31:30,312] INFO App info kafka.server for 0 unregistered (org.apache.kafka.common.utils.AppInfoParser)
[2025-04-13 13:31:30,323] INFO [KafkaServer id=0] shut down completed (kafka.server.KafkaServer)
```

Observation: Controlled shutdown initiated by kafka broker after few seconds.