Distributed Message Service for Kafka

Performance White Paper

Issue	01
Date	2023-03-28

HUAWEI TECHNOLOGIES CO., LTD.

Contents

1 Test Scenarios

This document describes performance tests on Distributed Message Service (DMS) for Kafka. The performance is measured by the message production rate on the client side and CPU usage on the server side. The tests cover the following scenarios:

- Test scenario 1 (batch size): same Kafka instance, same topics, different **batch.size** settings
- Test scenario 2 (cross-AZ or intra-AZ production): same Kafka instance, same topics, different AZ settings for the client and server
- Test scenario 3 (number of replicas): same Kafka instance, different numbers of replicas
- Test scenario 4 (synchronous or asynchronous replication): same Kafka instance, topics with different replication settings

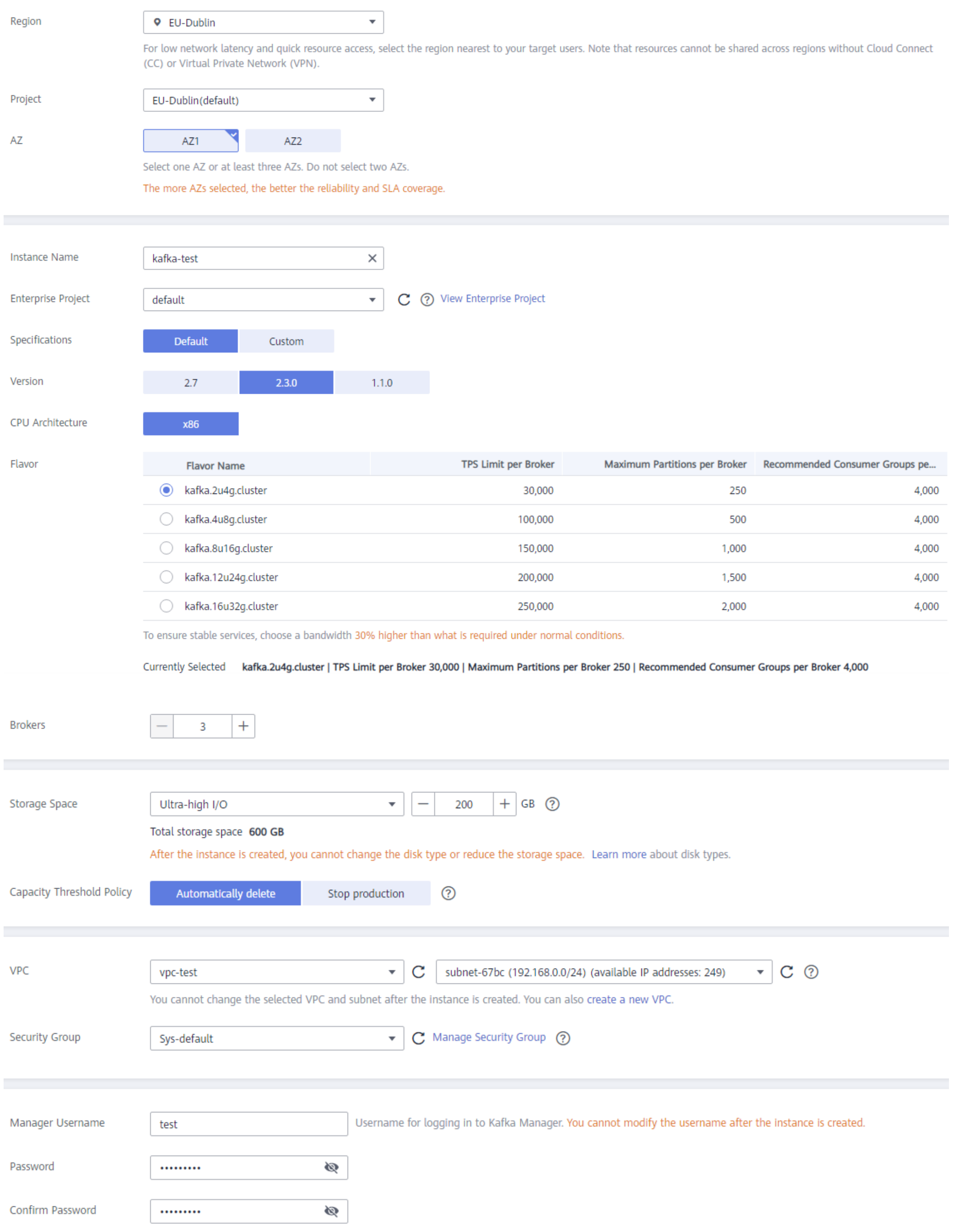
2 Test Environment

Perform the following steps to set up the test environment.

Step 1: Buy a Kafka Instance

Buy a Kafka instance with the following configurations. For details about how to buy a Kafka instance, see **Buying an Instance**.

- Region: EU-Dublin
- Project: EU-Dublin
- AZ: AZ1
- Instance name: kafka-test
- Enterprise project: default
- Version: 2.3.0
- Broker flavor: kafka.2u4g.cluster
- Brokers: 3
- Flavor: ultra-high I/O, 200 GB
- Capacity threshold policy: automatically delete
- VPC: If there is no available VPC, create one by referring to **Preparing Required Resources**.
- Security group: Select a security group that meets the requirements specified in **Preparing Required Resources**.
- Manager username: username used to log in to Kafka Manager
- Password: password used to log in to Kafka Manager
- Other settings: Do not enable public access, Kafka SASL_SSL, and automatic topic creation.



Region	EU-Dublin
	For low network latency and quick resource access, select the region nearest to your target users. Note that resources cannot be shared across regions without Cloud Connect (CC) or Virtual Private Network (VPN).
Project	EU-Dublin(default)
AZ	[✓ AZ1] AZ2
	Select one AZ or at least three AZs. Do not select two AZs.
	The more AZs selected, the better the reliability and SLA coverage.
Instance Name	kafka-test
Enterprise Project	default
Specifications	[✓ Default] Custom
Version	2.7 [✓ 2.3.0] 1.1.0
CPU Architecture	[✓ x86]

Flavor

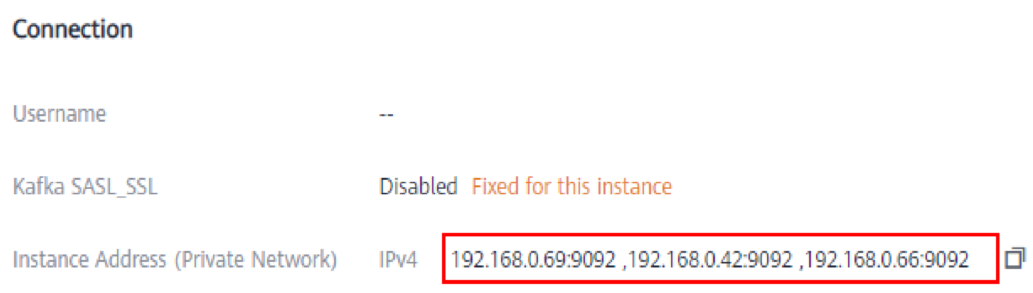
	Flavor Name	TPS Limit per Broker	Maximum Partitions per Broker	Recommended Consumer Groups pe...
●	kafka.2u4g.cluster	30,000	250	4,000
○	kafka.4u8g.cluster	100,000	500	4,000
○	kafka.8u16g.cluster	150,000	1,000	4,000
○	kafka.12u24g.cluster	200,000	1,500	4,000
○	kafka.16u32g.cluster	250,000	2,000	4,000

To ensure stable services, choose a bandwidth 30% higher than what is required under normal conditions.

Currently Selected **kafka.2u4g.cluster | TPS Limit per Broker 30,000 | Maximum Partitions per Broker 250 | Recommended Consumer Groups per Broker 4,000**

Brokers	3
Storage Space	Ultra-high I/O, 200 GB
	Total storage space **600 GB**
	After the instance is created, you cannot change the disk type or reduce the storage space. Learn more about disk types.
Capacity Threshold Policy	[✓ Automatically delete] Stop production
VPC	vpc-test, subnet-67bc (192.168.0.0/24) (available IP addresses: 249)
	You cannot change the selected VPC and subnet after the instance is created. You can also create a new VPC.
Security Group	Sys-default — Manage Security Group
Manager Username	test
	Username for logging in to Kafka Manager. You cannot modify the username after the instance is created.
Password	•••••••••
Confirm Password	•••••••••

After the purchase is complete, obtain the address of the Kafka instance on the instance details page.



Connection

Username	--
Kafka SASL_SSL	Disabled Fixed for this instance
Instance Address (Private Network)	IPv4 192.168.0.69:9092 ,192.168.0.42:9092 ,192.168.0.66:9092

Step 2: Create Topics

Create the following three topics for the **Kafka instance**. For details about how to create a topic, see **Creating a Topic**.

- Topic-01: 3 partitions, 1 replica, asynchronous replication
- Topic-02: 3 partition, 3 replicas, asynchronous replication
- Topic-03: 3 partition, 3 replicas, synchronous replication

Step 3: Obtain the Testing Tool

Download **Kafka CLI v2.3.0**.

Step 4: Buy ECSs

Buy two ECSs with the following configurations. For details about how to purchase an ECS, see **Purchasing an ECS**.

- One ECS is 4 vCPUs | 8 GB, runs Linux, and is configured with the same region, AZ, VPC, subnet, and security group as the Kafka instance purchased in **Step 1: Buy a Kafka Instance**.
- The other ECS is 4 vCPUs | 8 GB, runs Linux, and is configured with the same region, VPC, subnet, and security group but a different AZ as the Kafka instance purchased in **Step 1: Buy a Kafka Instance**.

Perform the following operations on the ECSs:

- Install **Java JDK** and configure the environment variables **JAVA_HOME** and **PATH**.
  ```
  export JAVA_HOME=/root/jdk1.8.0_231
  export PATH=$JAVA_HOME/bin:$PATH
  ```
- Download **Kafka CLI v2.3.0** and decompress it.
  ```
  tar -zxf kafka_2.11-2.3.0.tgz
  ```

3 Test Procedure

Test the message production rate on the client side and the CPU usage on the server side in the following scenarios:

- Test scenario 1 (batch size): same Kafka instance, same topics, different **batch.size** settings
- Test scenario 2 (cross-AZ or intra-AZ production): same Kafka instance, same topics, different AZ settings for the client and server
- Test scenario 3 (number of replicas): same Kafka instance, different numbers of replicas
- Test scenario 4 (synchronous or asynchronous replication): same Kafka instance, topics with different replication settings

Table 3-1 Test parameters

Partitions	Replicas	Synchronous Replication Enabled	batch.size	Cross-AZ Production
3	1	No	1 KB	No
3	1	No	16 KB	No
3	1	No	1 KB	Yes
3	3	Yes	1 KB	No
3	3	No	1 KB	No

Test script:

```
./kafka-producer-perf-test.sh --producer-props bootstrap.servers=${connection address} acks=1 batch.size=${batch.size} linger.ms=0 --topic ${topic name} --num-records ${num-records} --record-size 1024 --throughput -102400
```

- **bootstrap.servers**: address of the Kafka instance obtained in **Step 1: Buy a Kafka Instance**.
- **acks**: the message synchronization policy. acks=1 indicates asynchronous replication, and acks=-1 indicates synchronous replication.

- **batch.size**: size of messages sent in each batch, in bytes.
- **linger.ms**: interval between two batches.
- **topic**: topic name set in **Step 2: Create Topics**.
- **num-records**: total number of messages to be sent.
- **record-size**: size of each message.
- **throughput**: number of messages sent per second.

Test Scenario 1: Varied Batch Sizes

Step 1 Log in to the client server, go to the **kafka_2.11-2.3.0/bin** directory, and run the following scripts.

Set **batch.size** to 1 KB, and run the following script:

```
./kafka-producer-perf-test.sh --producer-props
bootstrap.servers=192.168.0.69:9092,192.168.0.42:9092,192.168.0.66:9092 acks=1 batch.size=1024
linger.ms=0 --topic Topic-01 --num-records 8000000 --record-size 1024 --throughput 102400
```

The result is as follows:

```
8000000 records sent, 27417.353814 records/sec (26.77 MB/sec), 1095.79 ms avg latency, 5989.00 ms max
latency, 679 ms 50th, 2957 ms 95th, 3505 ms 99th, 5951 ms 99.9th.
```

Message production rate: 27,417 records/second

Set **batch.size** to 16 KB, and run the following script:

```
./kafka-producer-perf-test.sh --producer-props
bootstrap.servers=192.168.0.69:9092,192.168.0.42:9092,192.168.0.66:9092 acks=1 batch.size=16384
linger.ms=0 --topic Topic-01 --num-records 100000000 --record-size 1024 --throughput 102400
```

The result is as follows:

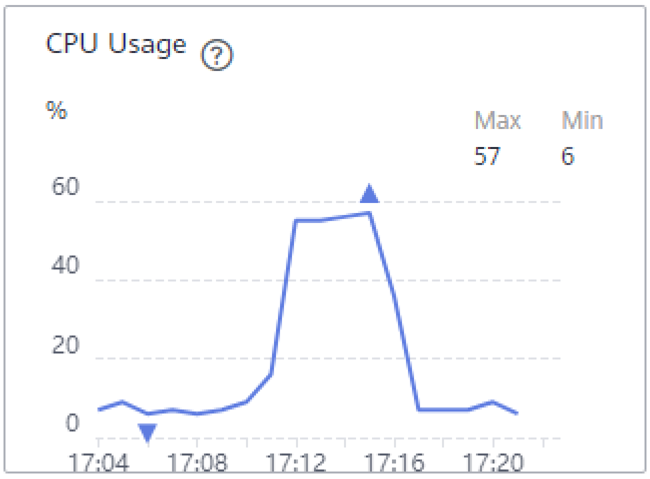
```
100000000 records sent, 102399.213574 records/sec (100.00 MB/sec), 11.27 ms avg latency, 876 ms max
latency, 1 ms 50th, 15 ms 95th, 384 ms 99th, 724 ms 99.9th.
```

Message production rate: 102,399 records/second

Step 2 Log in to the Kafka console, locate the row that contains the test instance, and click to go to the Cloud Eye console.

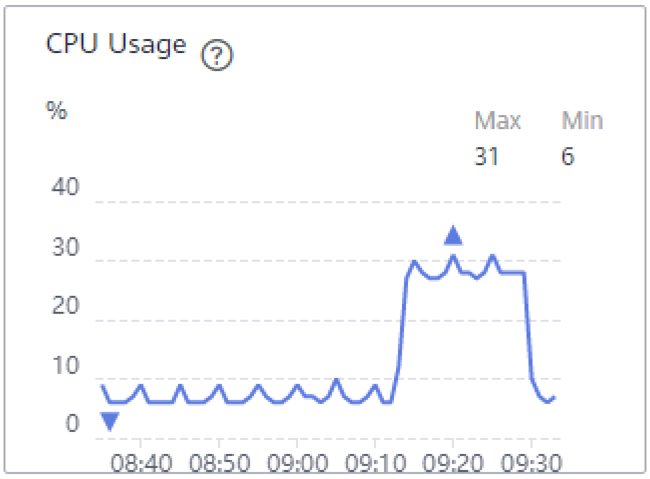
Step 3 On the **Brokers** tab page, view the CPU usage of the server nodes.

Figure 3-1 broker-0 CPU usage (batch.size = 1 KB)



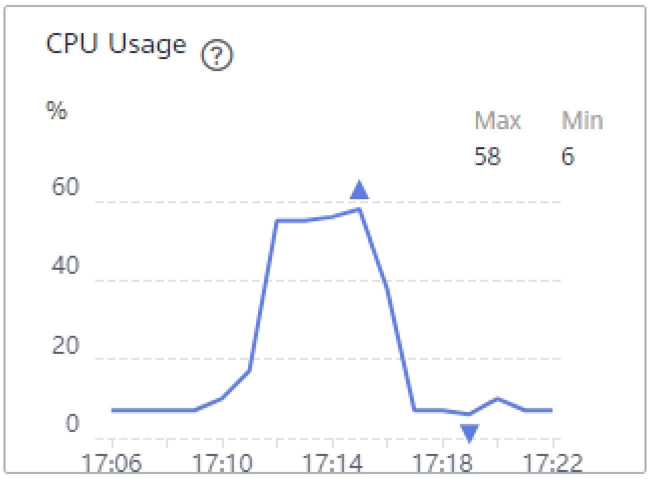
CPU usage: 57%

Figure 3-2 broker-0 CPU usage (batch.size = 16 KB)



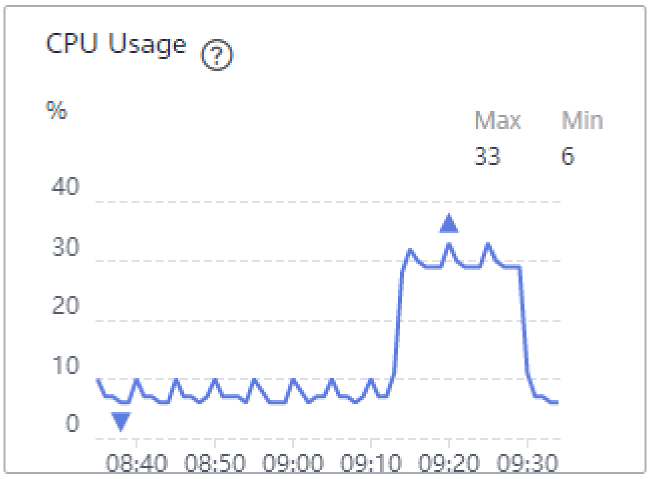
CPU usage: 31%

Figure 3-3 broker-1 CPU usage (batch.size = 1 KB)



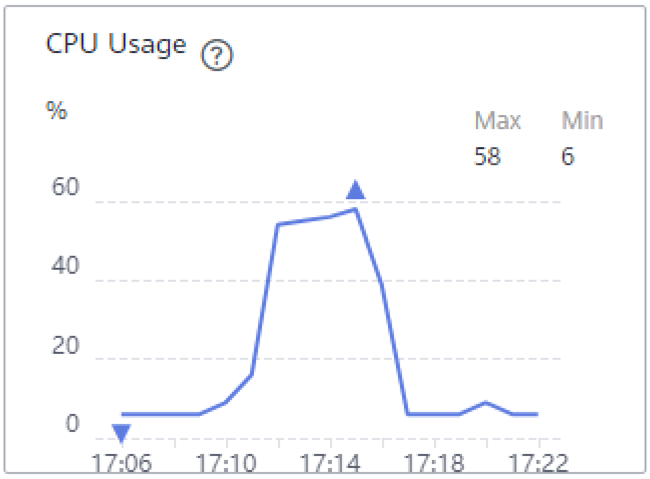
CPU usage: 58%

Figure 3-4 broker-1 CPU usage (batch.size = 16 KB)



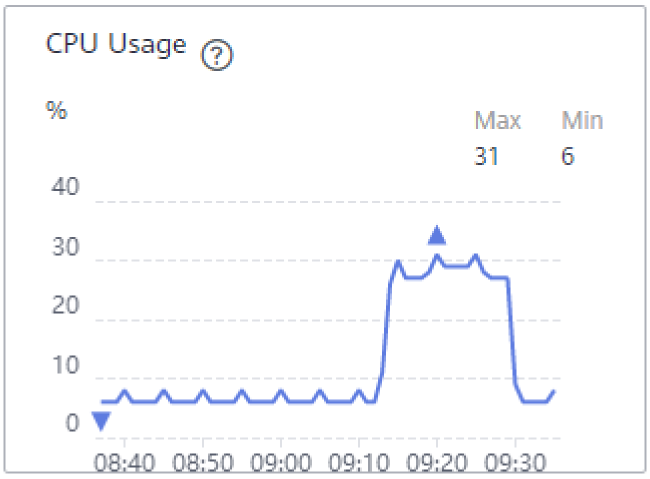
CPU usage: 33%

Figure 3-5 broker-2 CPU usage (batch.size = 1 KB)



CPU usage: 58%

Figure 3-6 broker-2 CPU usage (batch.size = 16 KB)



CPU usage: 31%

----End

Test Scenario 2: Cross-AZ or Intra-AZ Production

Step 1 Log in to the client server, go to the **kafka_2.11-2.3.0/bin** directory, and run the following scripts.

Configure the same AZ for the client and the instance, and run the following script:

```
./kafka-producer-perf-test.sh --producer-props
bootstrap.servers=192.168.0.69:9092,192.168.0.42:9092,192.168.0.66:9092 acks=1 batch.size=1024
linger.ms=0 --topic Topic-01 --num-records 8000000 --record-size 1024 --throughput 102400
```

The result is as follows:

```
8000000 records sent, 27417.353814 records/sec (26.77 MB/sec), 1095.79 ms avg latency, 5989.00 ms max
latency, 679 ms 50th, 2957 ms 95th, 3505 ms 99th, 5951 ms 99.9th.
```

Message production rate: 27,417 records/second

Configure different AZs for the client and the instance, and run the following script:

```
./kafka-producer-perf-test.sh --producer-props
bootstrap.servers=192.168.0.69:9092,192.168.0.42:9092,192.168.0.66:9092 acks=1 batch.size=1024
linger.ms=0 --topic Topic-01 --num-records 4000000 --record-size 1024 --throughput 102400
```

The result is as follows:

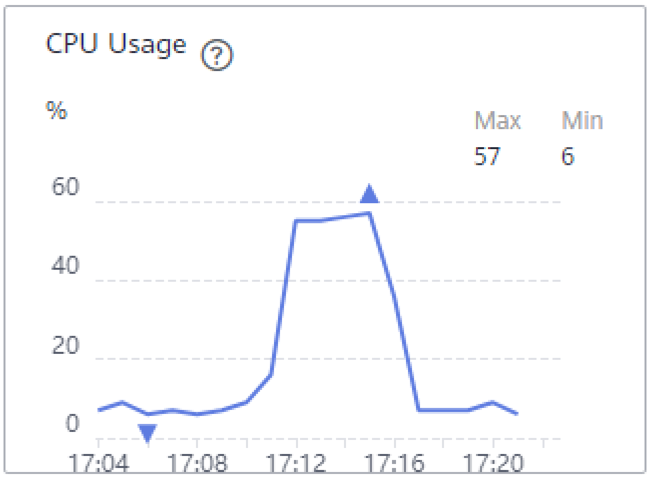
```
4000000 records sent, 15351.079181 records/sec (14.99 MB/sec), 1927.26 ms avg latency, 8974.00 ms max
latency, 15 ms 50th, 5925 ms 95th, 6838 ms 99th, 8925 ms 99.9th.
```

Message production rate: 15,351 records/second

Step 2 Log in to the Kafka console, locate the row that contains the test instance, and click to go to the Cloud Eye console.

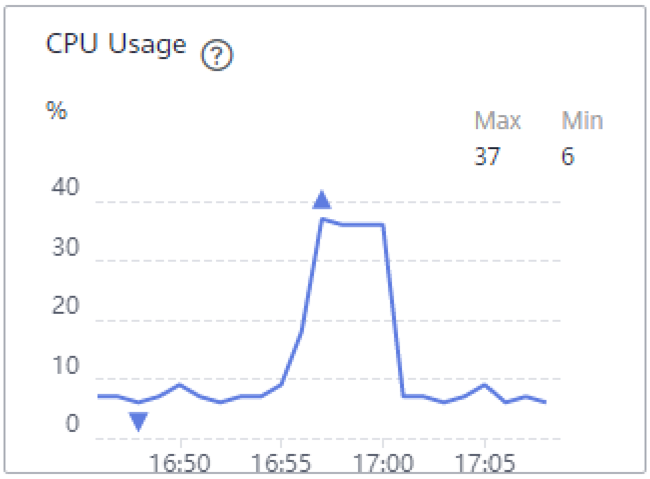
Step 3 On the **Brokers** tab page, view the CPU usage of the server nodes.

Figure 3-7 broker-0 CPU usage (same AZ)



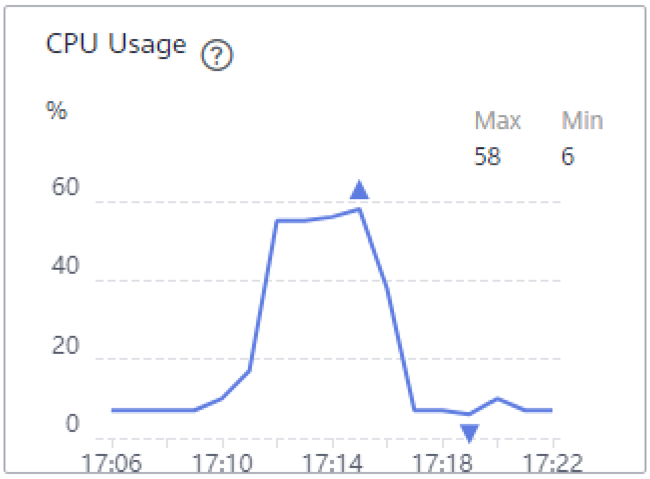
CPU usage: 57%

Figure 3-8 broker-0 CPU usage (different AZs)



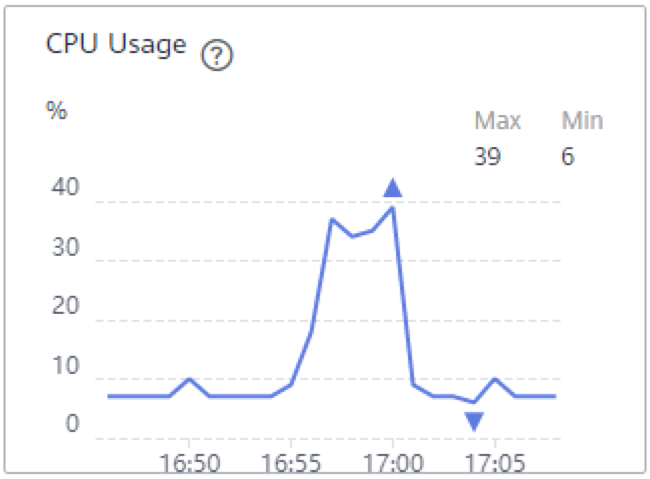
CPU usage: 37%

Figure 3-9 broker-1 CPU usage (same AZ)



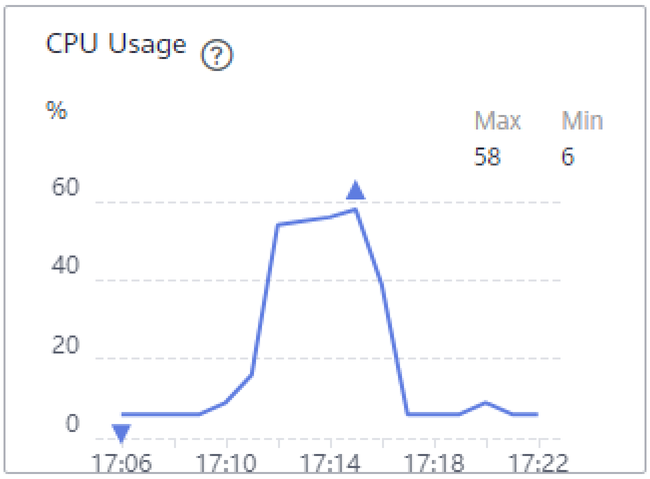
CPU usage: 58%

Figure 3-10 broker-1 CPU usage (different AZs)



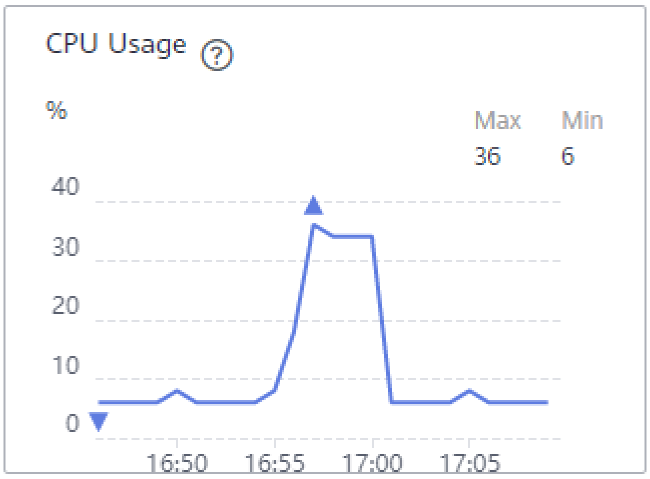
CPU usage: 39%

Figure 3-11 broker-2 CPU usage (same AZ)



CPU usage: 58%

Figure 3-12 broker-2 CPU usage (different AZs)



CPU usage: 36%

----End

Test Scenario 3: Varied Numbers of Replicas

Step 1 Log in to the client server, go to the **kafka_2.11-2.3.0/bin** directory, and run the following scripts.

For the **one-replica** topic, run the following script:

```
./kafka-producer-perf-test.sh --producer-props
bootstrap.servers=192.168.0.69:9092,192.168.0.42:9092,192.168.0.66:9092 acks=1 batch.size=1024
linger.ms=0 --topic Topic-01 --num-records 8000000 --record-size 1024 --throughput 102400
```

The result is as follows:

```
8000000 records sent, 27417.353814 records/sec (26.77 MB/sec), 1095.79 ms avg latency, 5989.00 ms max
latency, 679 ms 50th, 2957 ms 95th, 3505 ms 99th, 5951 ms 99.9th.
```

Message production rate: 27,417 records/second

For the **three-replica** topic, run the following script:

```
./kafka-producer-perf-test.sh --producer-props
bootstrap.servers=192.168.0.69:9092,192.168.0.42:9092,192.168.0.66:9092 acks=1 batch.size=1024
linger.ms=0 --topic Topic-02 --num-records 4000000 --record-size 1024 --throughput 102400
```

The result is as follows:

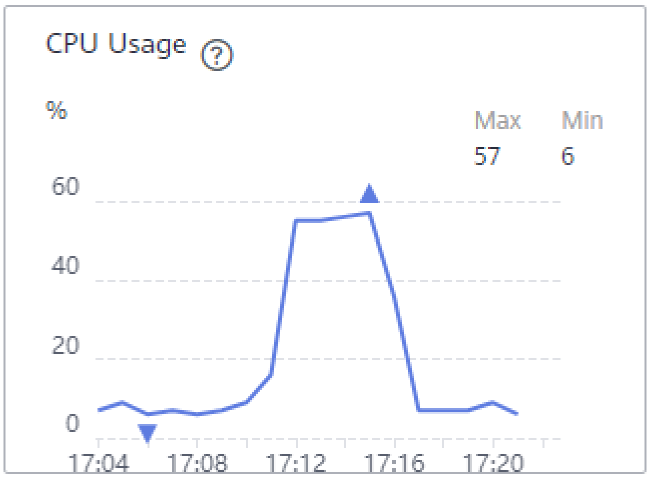
```
4000000 records sent, 11557.954473 records/sec (11.29 MB/sec), 2591.66 ms avg latency, 8071.00 ms max
latency, 2566 ms 50th, 5396 ms 95th, 6276 ms 99th, 8003 ms 99.9th.
```

Message production rate: 11,558 records/second

Step 2 Log in to the Kafka console, locate the row that contains the test instance, and click to go to the Cloud Eye console.

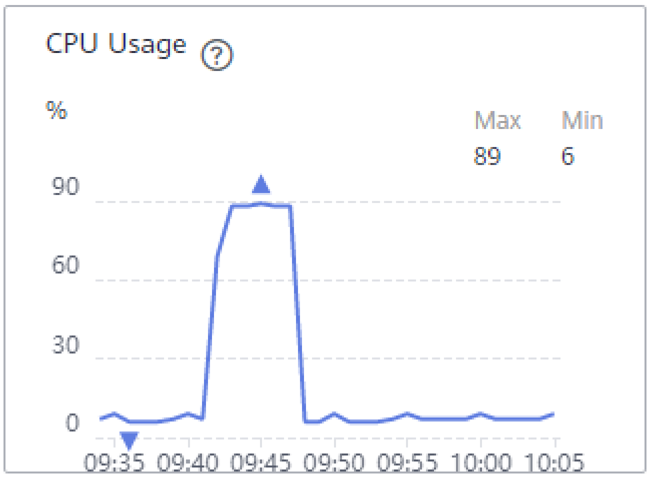
Step 3 On the **Brokers** tab page, view the CPU usage of the server nodes.

Figure 3-13 broker-0 CPU usage (one replica)



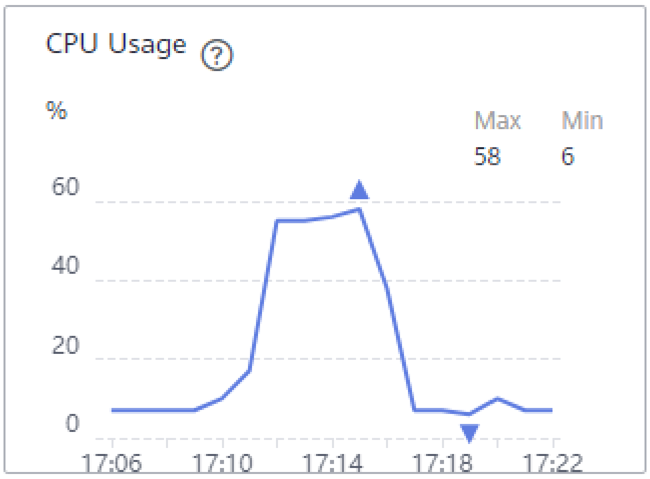
CPU usage: 57%

Figure 3-14 broker-0 CPU usage (three replicas)



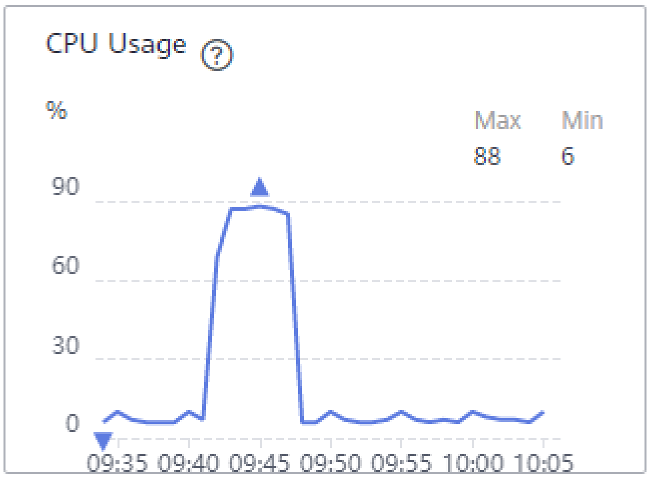
CPU usage: 89%

Figure 3-15 broker-1 CPU usage (one replica)



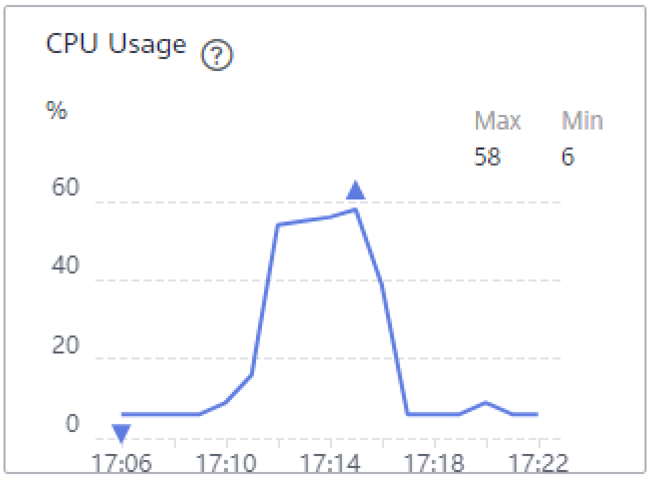
CPU usage: 58%

Figure 3-16 broker-1 CPU usage (three replicas)



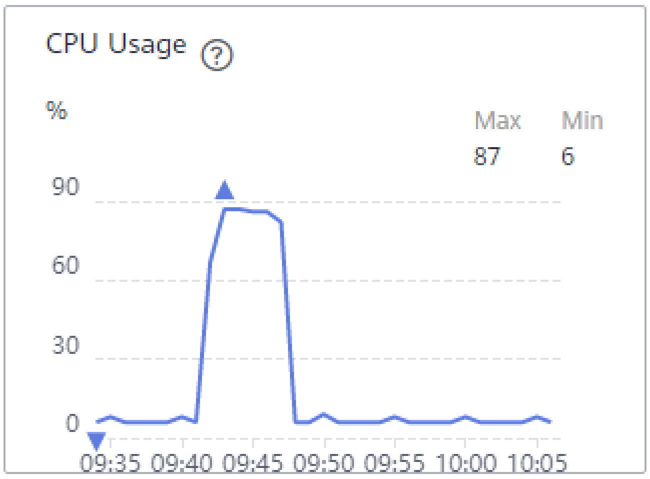
CPU usage: 88%

Figure 3-17 broker-2 CPU usage (one replica)



CPU usage: 58%

Figure 3-18 broker-2 CPU usage (three replicas)



CPU usage: 87%

----End

Test Scenario 4: Synchronous/Asynchronous Replication

Step 1 Log in to the client server, go to the **kafka_2.11-2.3.0/bin** directory, and run the following scripts.

For **asynchronous replication**, run the following script:

```
./kafka-producer-perf-test.sh --producer-props
bootstrap.servers=192.168.0.69:9092,192.168.0.42:9092,192.168.0.66:9092 acks=1 batch.size=1024
linger.ms=0 --topic Topic-02 --num-records 4000000 --record-size 1024 --throughput 102400
```

The result is as follows:

```
4000000 records sent, 11557.954473 records/sec (11.29 MB/sec), 2591.66 ms avg latency, 8071.00 ms max
latency, 2566 ms 50th, 5396 ms 95th, 6276 ms 99th, 8003 ms 99.9th.
```

Message production rate: 11,558 records/second

For **synchronous replication**, run the following script:

```
./kafka-producer-perf-test.sh --producer-props
bootstrap.servers=192.168.0.69:9092,192.168.0.42:9092,192.168.0.66:9092 acks=-1 batch.size=1024
linger.ms=0 --topic Topic-03 --num-records 1000000 --record-size 1024 --throughput 102400
```

The result is as follows:

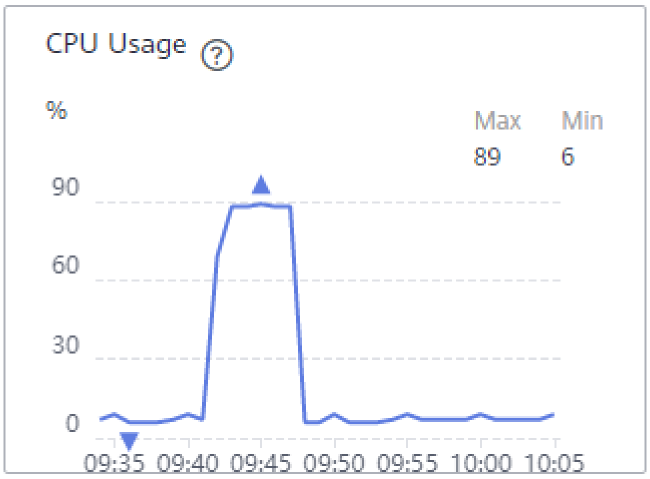
```
1000000 records sent, 3859.960628 records/sec (3.77 MB/sec), 7675.00 ms avg latency, 13852.00 ms max
latency, 7695 ms 50th, 11056 ms 95th, 12907 ms 99th, 13809 ms 99.9th.
```

Message production rate: 3859 records/second

Step 2 Log in to the Kafka console, locate the row that contains the test instance, and click to go to the Cloud Eye console.

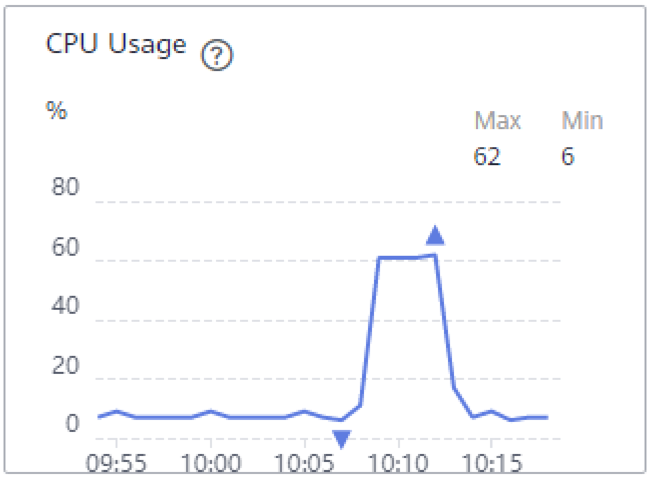
Step 3 On the **Brokers** tab page, view the CPU usage of the server nodes.

Figure 3-19 broker-0 CPU usage (asynchronous replication)



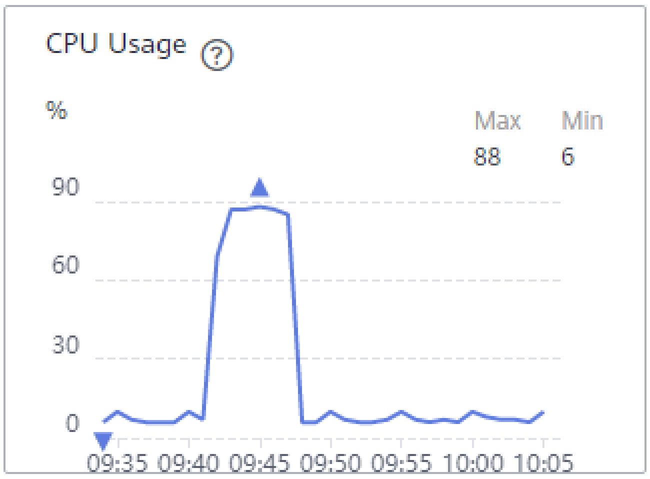
CPU usage: 89%

Figure 3-20 broker-0 CPU usage (synchronous replication)



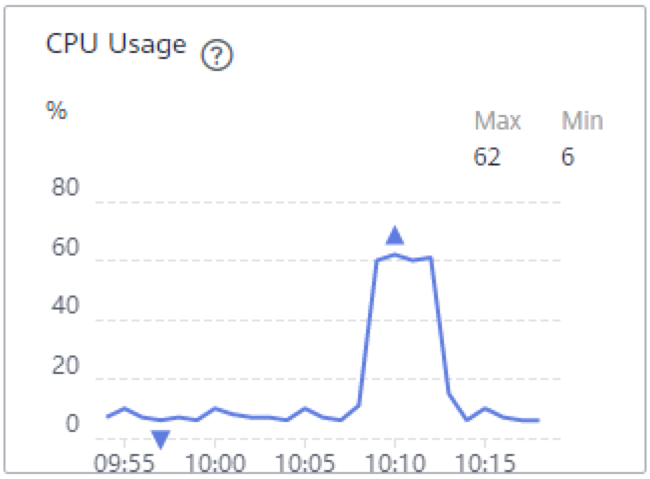
CPU usage: 62%

Figure 3-21 broker-1 CPU usage (asynchronous replication)



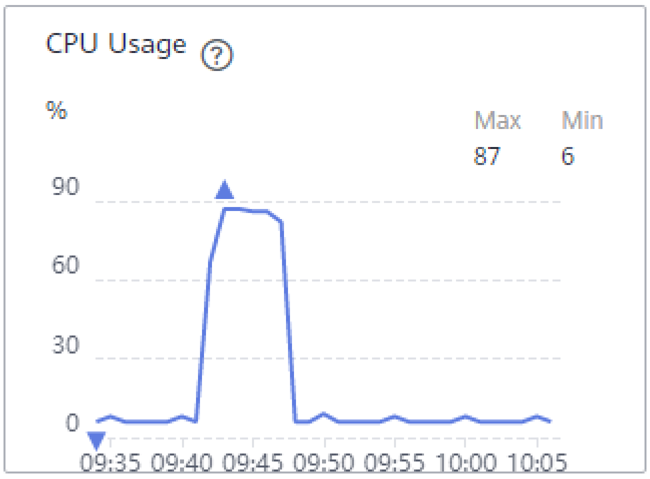
CPU usage: 88%

Figure 3-22 broker-1 CPU usage (synchronous replication)



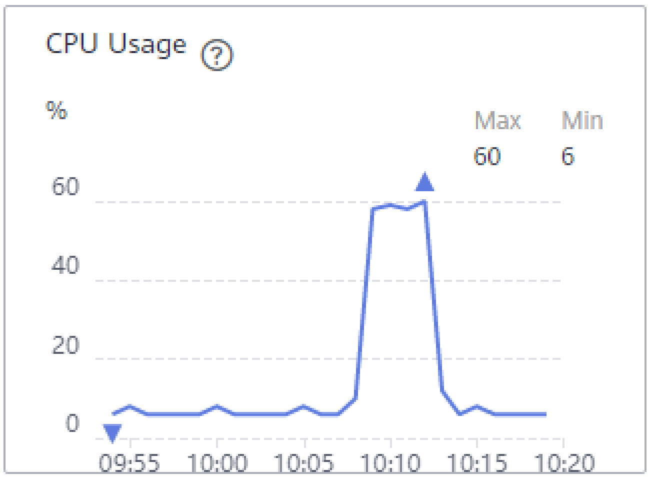
CPU usage: 62%

Figure 3-23 broker-2 CPU usage (asynchronous replication)



CPU usage: 87%

Figure 3-24 broker-2 CPU usage (synchronous replication)



CPU usage: 60%

----End

4 Test Results

Table 4-1 Test Results

Partitions	Replicas	Synchronous Replication Enabled	batch.size	Cross-AZ Production	Message Production Rate on the Client Side (Records/Second)	CPU Usage on the Server Side (broker-0)	CPU Usage on the Server Side (broker-1)	CPU Usage on the Server Side (broker-2)
3	1	No	1 KB	No	27,417	57%	58%	58%
3	1	No	16 KB	No	102,399	31%	33%	31%
3	1	No	1 KB	Yes	15,351	37%	39%	36%
3	3	Yes	1 KB	No	3859	62%	62%	60%
3	3	No	1 KB	No	11,558	89%	88%	87%

Based on the test results, the following conclusions are drawn (for reference only):

- When the **batch.size** of production requests is 16 times larger, the message production rate increases, and the CPU usage decreases.
- Compared with cross-AZ production, intra-AZ production significantly increases message production rate and CPU usage.
- When the number of replicas changes from 1 to 3, the message production rate decreases significantly, and the CPU usage increases.
- Compared with synchronous replication, asynchronous replication increases the message production rate and the CPU usage.