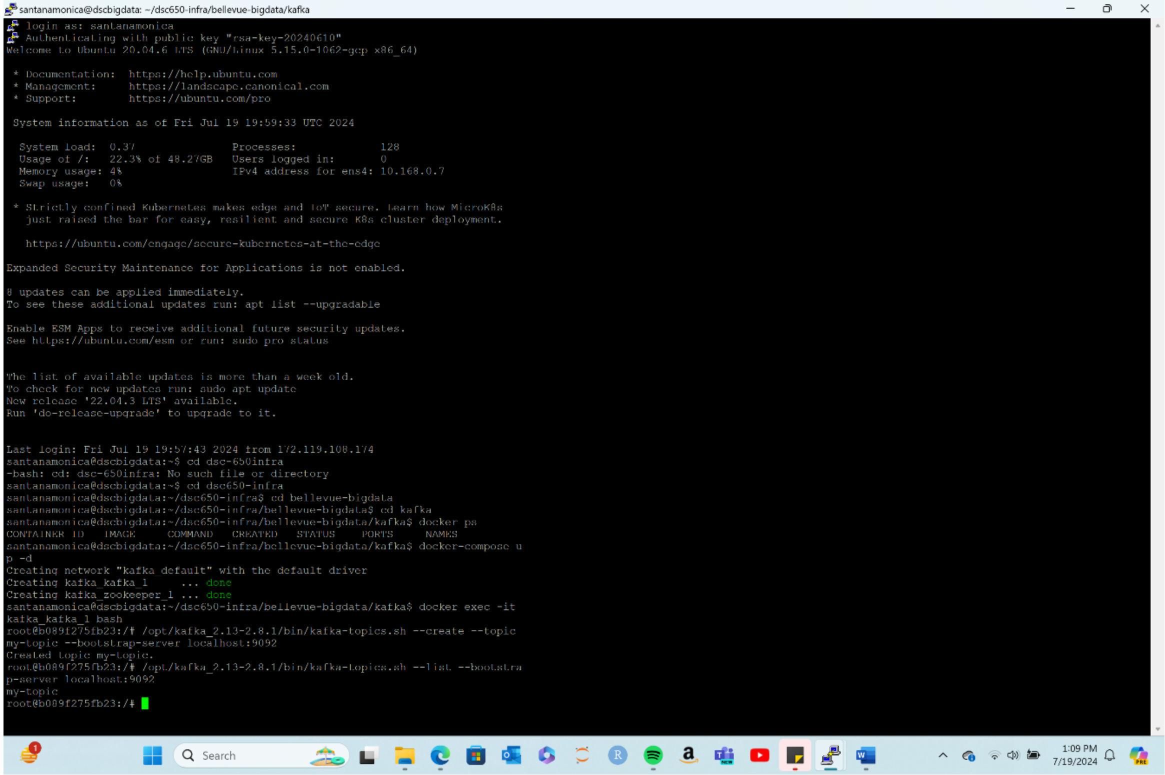
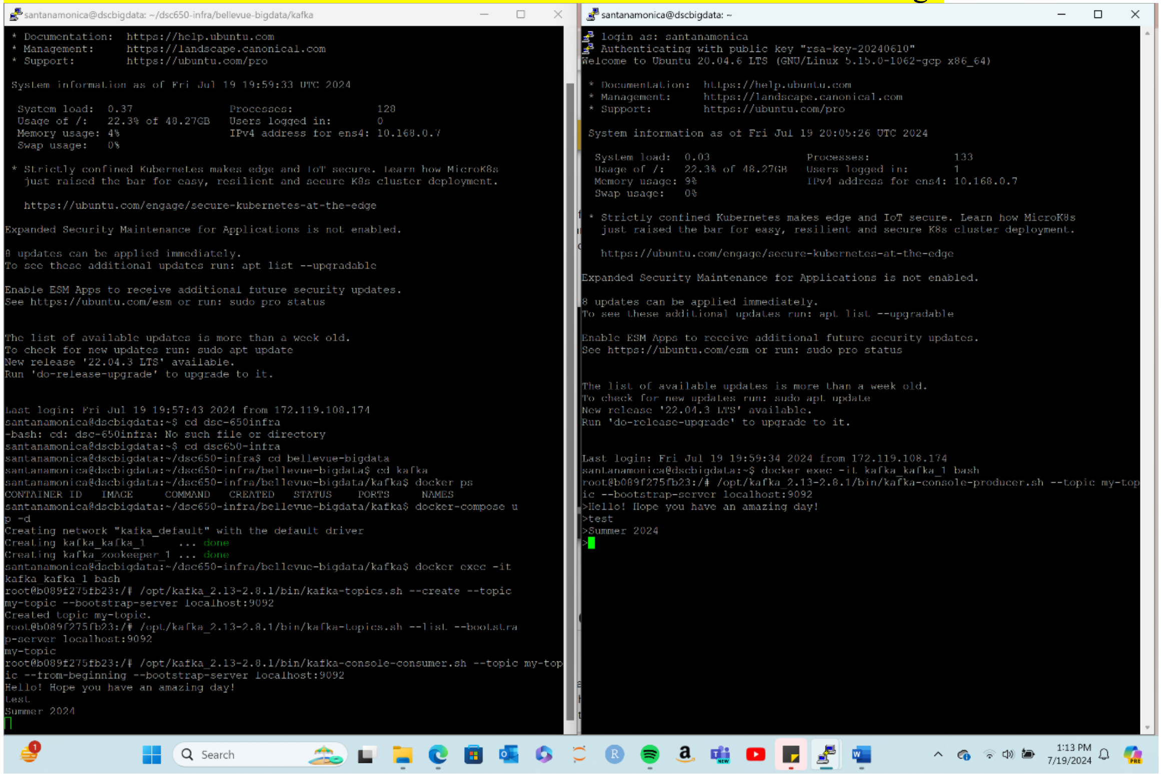
Monica Santana – Exercise 7

My-topic listed amongst the topics


santanamonica@dscbigdata: ~/dsc650-infra/bellevue-bigdata/kafka
Search
1:09 PM
7/19/2024

```
login as: santanamonica
Authenticating with public key "rsa-key-20240610"
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-1062-gcp x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Fri Jul 19 19:59:33 UTC 2024

  System load:  0.37               Processes:             128
  Usage of /:   22.3% of 48.27GB   Users logged in:       0
  Memory usage: 4%                 IPv4 address for ens4: 10.168.0.7
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

8 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
New release '22.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Fri Jul 19 19:57:43 2024 from 172.119.108.174
santanamonica@dscbigdata:~$ cd dsc-650infra
-bash: cd: dsc-650infra: No such file or directory
santanamonica@dscbigdata:~$ cd dsc650-infra
santanamonica@dscbigdata:~/dsc650-infra$ cd bellevue-bigdata
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata$ cd kafka
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata/kafka$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata/kafka$ docker-compose u
p -d
Creating network "kafka_default" with the default driver
Creating kafka_kafka_1     ... done
Creating kafka_zookeeper_1 ... done
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata/kafka$ docker exec -it
kafka_kafka_1 bash
root@b089f275fb23:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic
my-topic --bootstrap-server localhost:9092
Created topic my-topic.
root@b089f275fb23:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --list --bootstra
p-server localhost:9092
my-topic
root@b089f275fb23:/#
```

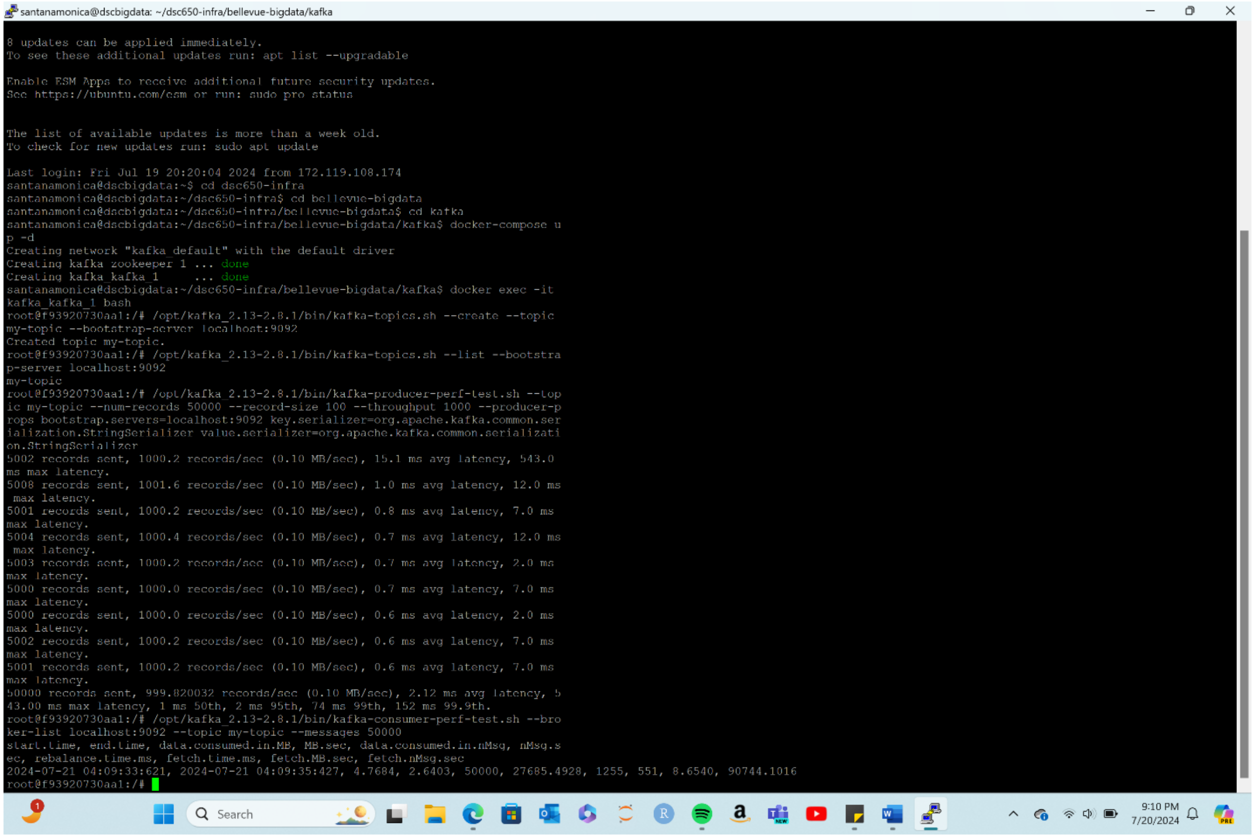
Producer and consumer terminals with entered text and received message


santanamonica@dscbigdata: ~/dsc650-infra/bellevue-bigdata/kafka
santanamonica@dscbigdata: ~
Search
1:13 PM
7/19/2024

```
 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Fri Jul 19 19:59:33 UTC 2024

  System load:  0.37               Processes:             128
  Usage of /:   22.3% of 48.27GB   Users logged in:       0
  Memory usage: 4%                 IPv4 address for ens4: 10.168.0.7
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

8 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
New release '22.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Fri Jul 19 19:57:43 2024 from 172.119.108.174
santanamonica@dscbigdata:~$ cd dsc-650infra
-bash: cd: dsc-650infra: No such file or directory
santanamonica@dscbigdata:~$ cd dsc650-infra
santanamonica@dscbigdata:~/dsc650-infra$ cd bellevue-bigdata
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata$ cd kafka
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata/kafka$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata/kafka$ docker-compose u
p -d
Creating network "kafka_default" with the default driver
Creating kafka_kafka_1     ... done
Creating kafka_zookeeper_1 ... done
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata/kafka$ docker exec -it
kafka_kafka_1 bash
root@b089f275fb23:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic
my-topic --bootstrap-server localhost:9092
Created topic my-topic.
root@b089f275fb23:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --list --bootstra
p-server localhost:9092
my-topic
root@b089f275fb23:/# /opt/kafka_2.13-2.8.1/bin/kafka-console-consumer.sh --topic my-top
ic --from-beginning --bootstrap-server localhost:9092
Hello! Hope you have an amazing day!
test
Summer 2024
```

```
login as: santanamonica
Authenticating with public key "rsa-key-20240610"
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-1062-gcp x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Fri Jul 19 20:05:26 UTC 2024

  System load:  0.03               Processes:             133
  Usage of /:   22.3% of 48.27GB   Users logged in:       1
  Memory usage: 9%                 IPv4 address for ens4: 10.168.0.7
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

8 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
New release '22.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Fri Jul 19 19:59:34 2024 from 172.119.108.174
santanamonica@dscbigdata:~$ docker exec -it kafka_kafka_1 bash
root@b089f275fb23:/# /opt/kafka_2.13-2.8.1/bin/kafka-console-producer.sh --topic my-top
ic --bootstrap-server localhost:9092
>Hello! Hope you have an amazing day!
>test
>Summer 2024
>
```

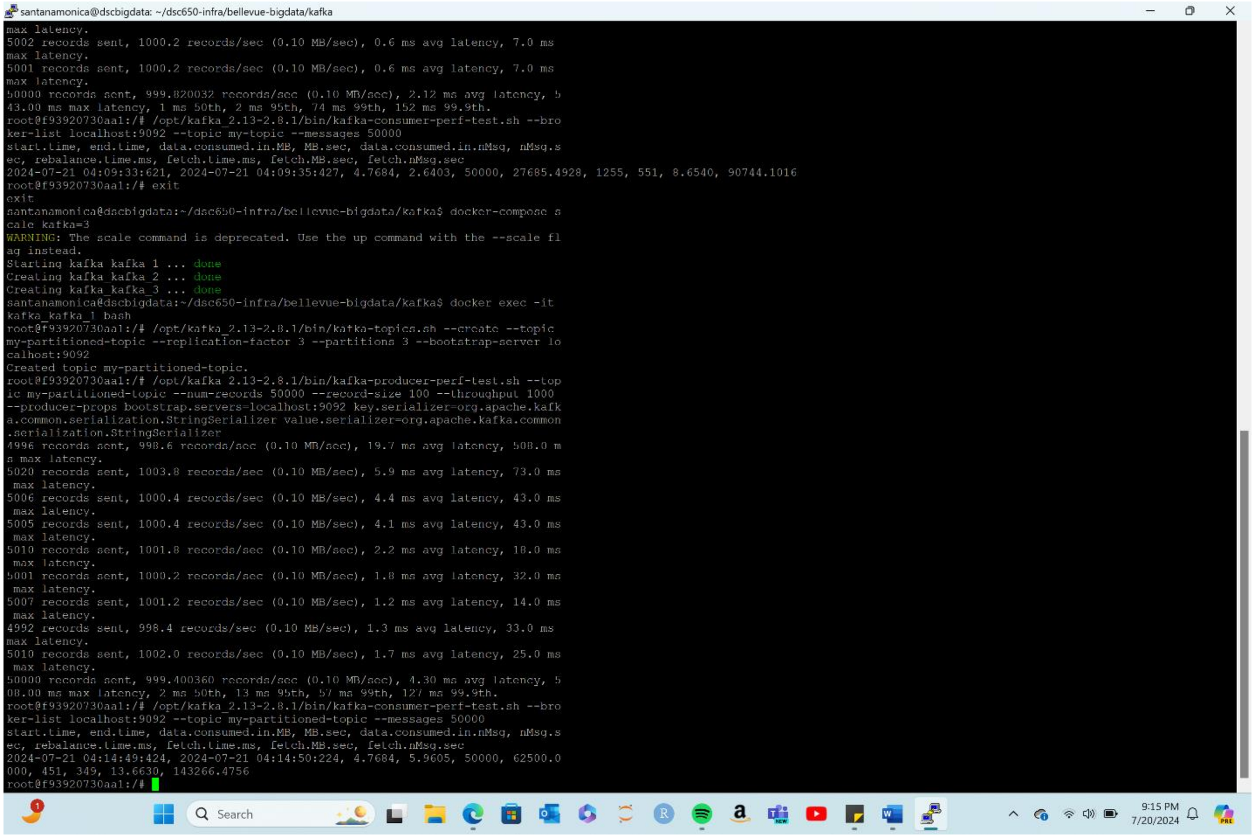
Producer and consumer performance test results


santanamonica@dscbigdata: ~/dsc650-infra/bellevue-bigdata/kafka
Search
9:10 PM
7/20/2024

```
8 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Fri Jul 19 20:20:04 2024 from 172.119.108.174
santanamonica@dscbigdata:~$ cd dsc650-infra
santanamonica@dscbigdata:~/dsc650-infra$ cd bellevue-bigdata
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata$ cd kafka
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata/kafka$ docker-compose u
p -d
Creating network "kafka_default" with the default driver
Creating kafka_zookeeper_1 ... done
Creating kafka_kafka_1     ... done
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata/kafka$ docker exec -it
kafka_kafka_1 bash
root@f93920730aa1:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic
my-topic --bootstrap-server localhost:9092
Created topic my-topic.
root@f93920730aa1:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --list --bootstra
p-server localhost:9092
my-topic
root@f93920730aa1:/# /opt/kafka_2.13-2.8.1/bin/kafka-producer-perf-test.sh --top
ic my-topic --num-records 50000 --record-size 100 --throughput 1000 --producer-p
rops bootstrap.servers=localhost:9092 key.serializer=org.apache.kafka.common.ser
ialization.StringSerializer value.serializer=org.apache.kafka.common.serializati
on.StringSerializer
5002 records sent, 1000.2 records/sec (0.10 MB/sec), 15.1 ms avg latency, 543.0
ms max latency.
5008 records sent, 1001.6 records/sec (0.10 MB/sec), 1.0 ms avg latency, 12.0 ms
 max latency.
5001 records sent, 1000.2 records/sec (0.10 MB/sec), 0.8 ms avg latency, 7.0 ms
max latency.
5004 records sent, 1000.4 records/sec (0.10 MB/sec), 0.7 ms avg latency, 12.0 ms
 max latency.
5003 records sent, 1000.2 records/sec (0.10 MB/sec), 0.7 ms avg latency, 2.0 ms
max latency.
5000 records sent, 1000.0 records/sec (0.10 MB/sec), 0.7 ms avg latency, 7.0 ms
max latency.
5000 records sent, 1000.0 records/sec (0.10 MB/sec), 0.6 ms avg latency, 2.0 ms
max latency.
5002 records sent, 1000.2 records/sec (0.10 MB/sec), 0.6 ms avg latency, 7.0 ms
max latency.
5001 records sent, 1000.2 records/sec (0.10 MB/sec), 0.6 ms avg latency, 7.0 ms
max latency.
50000 records sent, 999.820032 records/sec (0.10 MB/sec), 2.12 ms avg latency, 5
43.00 ms max latency, 1 ms 50th, 2 ms 95th, 74 ms 99th, 152 ms 99.9th.
root@f93920730aa1:/# /opt/kafka_2.13-2.8.1/bin/kafka-consumer-perf-test.sh --bro
ker-list localhost:9092 --topic my-topic --messages 50000
start.time, end.time, data.consumed.in.MB, MB.sec, data.consumed.in.nMsg, nMsg.s
ec, rebalance.time.ms, fetch.time.ms, fetch.MB.sec, fetch.nMsg.sec
2024-07-21 04:09:33:621, 2024-07-21 04:09:35:427, 4.7684, 2.6403, 50000, 27685.4928, 1255, 551, 8.6540, 90744.1016
root@f93920730aa1:/#
```

These results show that the producer test had a performance of 50,000 records sent out with 999.82 records being sent out per second. There was 2.12 ms average latency with a max of 543 ms latency here and there. The consumer performance test results showed that 50,000 messages were consumed, and 27,685 of those messages were consumed per second. Around 4.77 megabytes of messages were consumed at 2.64 megabytes per second. Messages were fetched at a rate of 8.65 megabytes per second. The consumer successfully consumed all the messages with low latency.

Producer and consumer performance test results with a partitioned topic


santanamonica@dscbigdata: ~/dsc650-infra/bellevue-bigdata/kafka
Search
9:15 PM
7/20/2024

```
max latency.
5002 records sent, 1000.2 records/sec (0.10 MB/sec), 0.6 ms avg latency, 7.0 ms
max latency.
5001 records sent, 1000.2 records/sec (0.10 MB/sec), 0.6 ms avg latency, 7.0 ms
max latency.
50000 records sent, 999.820032 records/sec (0.10 MB/sec), 2.12 ms avg latency, 5
43.00 ms max latency, 1 ms 50th, 2 ms 95th, 74 ms 99th, 152 ms 99.9th.
root@f93920730aa1:/# /opt/kafka_2.13-2.8.1/bin/kafka-consumer-perf-test.sh --bro
ker-list localhost:9092 --topic my-topic --messages 50000
start.time, end.time, data.consumed.in.MB, MB.sec, data.consumed.in.nMsg, nMsg.s
ec, rebalance.time.ms, fetch.time.ms, fetch.MB.sec, fetch.nMsg.sec
2024-07-21 04:09:33:621, 2024-07-21 04:09:35:427, 4.7684, 2.6403, 50000, 27685.4928, 1255, 551, 8.6540, 90744.1016
root@f93920730aa1:/# exit
exit
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata/kafka$ docker-compose s
cale kafka=3
WARNING: The scale command is deprecated. Use the up command with the --scale fl
ag instead.
Starting kafka kafka 1 ... done
Creating kafka_kafka_2 ... done
Creating kafka_kafka_3 ... done
santanamonica@dscbigdata:~/dsc650-infra/bellevue-bigdata/kafka$ docker exec -it
kafka_kafka_1 bash
root@f93920730aa1:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic
my-partitioned-topic --replication-factor 3 --partitions 3 --bootstrap-server lo
calhost:9092
Created topic my-partitioned-topic.
root@f93920730aa1:/# /opt/kafka_2.13-2.8.1/bin/kafka-producer-perf-test.sh --top
ic my-partitioned-topic --num-records 50000 --record-size 100 --throughput 1000
--producer-props bootstrap.servers=localhost:9092 key.serializer=org.apache.kafk
a.common.serialization.StringSerializer value.serializer=org.apache.kafka.common
.serialization.StringSerializer
4996 records sent, 998.6 records/sec (0.10 MB/sec), 19.7 ms avg latency, 508.0 m
s max latency.
5020 records sent, 1003.8 records/sec (0.10 MB/sec), 5.9 ms avg latency, 73.0 ms
 max latency.
5006 records sent, 1000.4 records/sec (0.10 MB/sec), 4.4 ms avg latency, 43.0 ms
 max latency.
5005 records sent, 1000.4 records/sec (0.10 MB/sec), 4.1 ms avg latency, 43.0 ms
 max latency.
5010 records sent, 1001.8 records/sec (0.10 MB/sec), 2.2 ms avg latency, 18.0 ms
 max latency.
5001 records sent, 1000.2 records/sec (0.10 MB/sec), 1.8 ms avg latency, 32.0 ms
 max latency.
5007 records sent, 1001.2 records/sec (0.10 MB/sec), 1.2 ms avg latency, 14.0 ms
 max latency.
4992 records sent, 998.4 records/sec (0.10 MB/sec), 1.3 ms avg latency, 33.0 ms
max latency.
5010 records sent, 1002.0 records/sec (0.10 MB/sec), 1.7 ms avg latency, 25.0 ms
 max latency.
50000 records sent, 999.400360 records/sec (0.10 MB/sec), 4.30 ms avg latency, 5
08.00 ms max latency, 2 ms 50th, 13 ms 95th, 57 ms 99th, 127 ms 99.9th.
root@f93920730aa1:/# /opt/kafka_2.13-2.8.1/bin/kafka-consumer-perf-test.sh --bro
ker-list localhost:9092 --topic my-partitioned-topic --messages 50000
start.time, end.time, data.consumed.in.MB, MB.sec, data.consumed.in.nMsg, nMsg.s
ec, rebalance.time.ms, fetch.time.ms, fetch.MB.sec, fetch.nMsg.sec
2024-07-21 04:14:49:424, 2024-07-21 04:14:50:224, 4.7684, 5.9605, 50000, 62500.0
000, 451, 349, 13.6630, 143266.4756
root@f93920730aa1:/# 
```

These results show that the producer test had a performance of 50,000 records sent out with 999.40 records being sent out per second. There was 4.30 ms average latency with a max of 508 ms latency here and there. The consumer performance test results showed that 50,000 messages were consumed, and 62,500 of those messages were consumed per second. Around 4.77 megabytes of messages were consumed at 5.96 megabytes per second. Messages were fetched at a rate of 13.66 megabytes per second. The consumer successfully consumed all the messages with a relatively low latency.

The partitioned topic with 3 different partitions vs just 1 in the single topic test showed the scaled topic to perform worse. The average latency was higher for the scaled topic 4.30 ms vs 2.12 ms. The reason for this is due to the 3 partitions and the messages being divided by the 3 vs the 1 partition creating higher latency. To minimize time, the less partitions, the quicker Kafka will process the data.