

Introduction to Kafka

- Kafka Installation
- Kafka Practice Commands
- Kafka Cluster Practice Commands

Kafka Installation

1. Create ``/home/orienit/work`` folder using below command

◆ `mkdir /home/orienit/work`



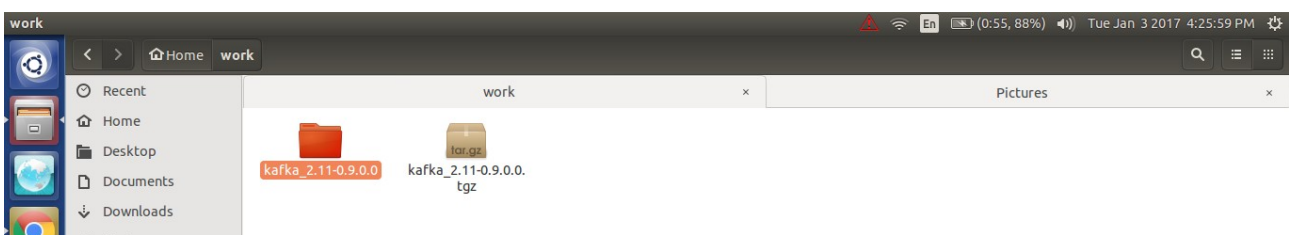
```
orienit@kalyan: ~  
orienit@kalyan:~$ mkdir /home/orienit/work  
orienit@kalyan:~$
```

2. Download ``kafka_2.11-0.9.0.0.tgz`` file from this [link](https://archive.apache.org/dist/kafka/0.9.0.0/kafka_2.11-0.9.0.0.tgz).
(https://archive.apache.org/dist/kafka/0.9.0.0/kafka_2.11-0.9.0.0.tgz)

3. Copy ``kafka_2.11-0.9.0.0.tgz`` file into ``/home/orienit/work`` folder



4. Extract ``kafka_2.11-0.9.0.0.tgz`` file into ``/home/orienit/work`` folder



5. Execute below command to open `~/.bashrc` file

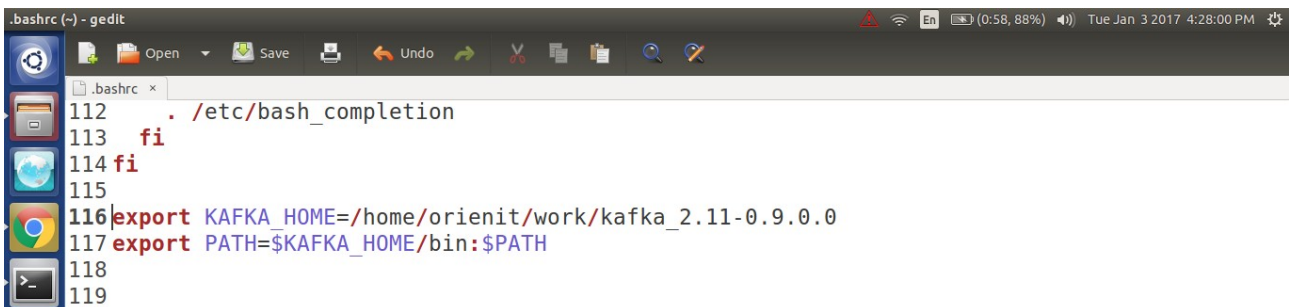
- ◆ **Command:** gedit ~/.bashrc



```
orientit@kalyan: ~  
orientit@kalyan:~$ gedit ~/.bashrc
```

6. Update `~/.bashrc` file with below changes

- ◆ export KAFKA_HOME=/home/orienit/work/kafka_2.11-0.9.0.0
- ◆ export PATH=\$KAFKA_HOME/bin:\$PATH



```
.bashrc (~) - gedit  
112 . /etc/bash_completion  
113 fi  
114 fi  
115  
116 export KAFKA_HOME=/home/orienit/work/kafka_2.11-0.9.0.0  
117 export PATH=$KAFKA_HOME/bin:$PATH  
118  
119
```

7. Re-open the terminal

8. Verify with below command

- ◆ **Command:** echo \$KAFKA_HOME

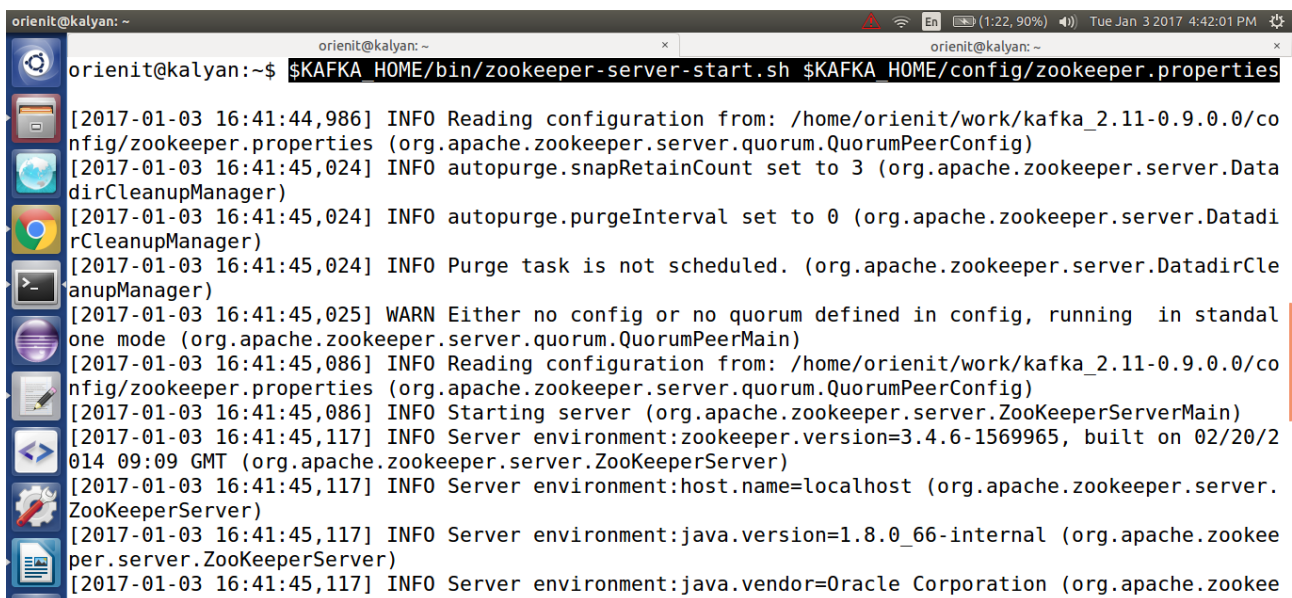


```
orientit@kalyan: ~  
orientit@kalyan:~$ echo $KAFKA_HOME  
/home/orienit/work/kafka_2.11-0.9.0.0  
orientit@kalyan:~$
```

9. Let's play with kafka

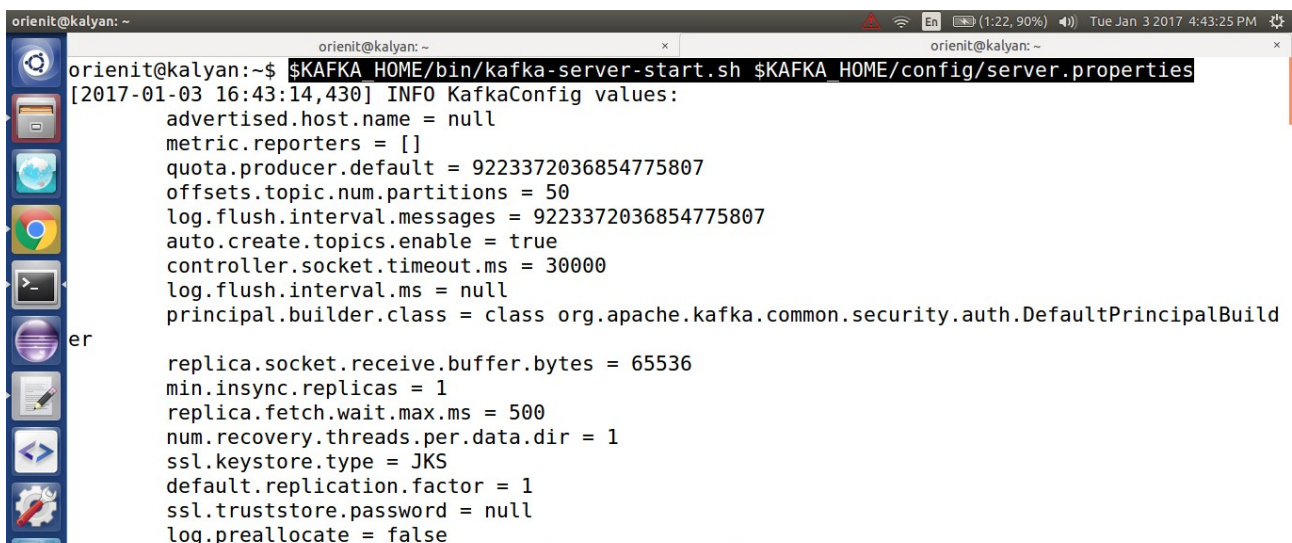
Kafka Practice Commands

1. Follow above procedure to install the kafka
2. Start the `**zookeeper**` using below command (New Terminal)
`$KAFKA_HOME/bin/zookeeper-server-start.sh $KAFKA_HOME/config/zookeeper.properties`



```
orientit@kalyan: ~$ $KAFKA_HOME/bin/zookeeper-server-start.sh $KAFKA_HOME/config/zookeeper.properties
[2017-01-03 16:41:44,986] INFO Reading configuration from: /home/orienit/work/kafka_2.11-0.9.0.0/conf/zookeeper.properties (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2017-01-03 16:41:45,024] INFO autopurge.snapRetainCount set to 3 (org.apache.zookeeper.server.DataDirCleanupManager)
[2017-01-03 16:41:45,024] INFO autopurge.purgeInterval set to 0 (org.apache.zookeeper.server.DataDirCleanupManager)
[2017-01-03 16:41:45,024] INFO Purge task is not scheduled. (org.apache.zookeeper.server.DataDirCleanupManager)
[2017-01-03 16:41:45,025] WARN Either no config or no quorum defined in config, running in standalone mode (org.apache.zookeeper.server.quorum.QuorumPeerMain)
[2017-01-03 16:41:45,086] INFO Reading configuration from: /home/orienit/work/kafka_2.11-0.9.0.0/conf/zookeeper.properties (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2017-01-03 16:41:45,086] INFO Starting server (org.apache.zookeeper.server.ZooKeeperServerMain)
[2017-01-03 16:41:45,117] INFO Server environment:zookeeper.version=3.4.6-1569965, built on 02/20/2014 09:09 GMT (org.apache.zookeeper.server.ZooKeeperServer)
[2017-01-03 16:41:45,117] INFO Server environment:host.name=localhost (org.apache.zookeeper.server.ZooKeeperServer)
[2017-01-03 16:41:45,117] INFO Server environment:java.version=1.8.0_66-internal (org.apache.zookeeper.server.ZooKeeperServer)
[2017-01-03 16:41:45,117] INFO Server environment:java.vendor=Oracle Corporation (org.apache.zookeeper.server.ZooKeeperServer)
```

3. Start the `**kafka server**` using below command (New Terminal)
`$KAFKA_HOME/bin/kafka-server-start.sh $KAFKA_HOME/config/server.properties`



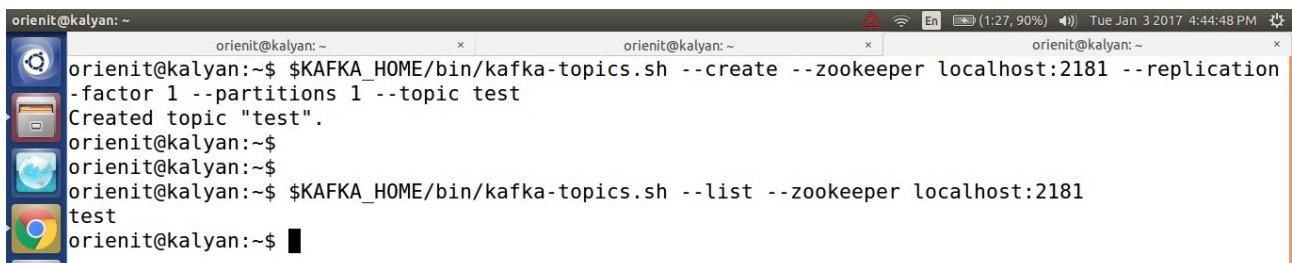
```
orientit@kalyan: ~$ $KAFKA_HOME/bin/kafka-server-start.sh $KAFKA_HOME/config/server.properties
[2017-01-03 16:43:14,430] INFO KafkaConfig values:
  advertised.host.name = null
  metric.reporters = []
  quota.producer.default = 9223372036854775807
  offsets.topic.num.partitions = 50
  log.flush.interval.messages = 9223372036854775807
  auto.create.topics.enable = true
  controller.socket.timeout.ms = 30000
  log.flush.interval.ms = null
  principal.builder.class = class org.apache.kafka.common.security.auth.DefaultPrincipalBuilder
  replica.socket.receive.buffer.bytes = 65536
  min.insync.replicas = 1
  replica.fetch.wait.max.ms = 500
  num.recovery.threads.per.data.dir = 1
  ssl.keystore.type = JKS
  default.replication.factor = 1
  ssl.truststore.password = null
  log.preallocate = false
```

4. Create a **test** topic using below command (New Terminal)

```
$KAFKA_HOME/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test
```

5. List out all the topics

```
$KAFKA_HOME/bin/kafka-topics.sh --list --zookeeper localhost:2181
```

A terminal window titled 'orientit@kalyan: ~' showing the execution of Kafka commands. The first command is '\$KAFKA_HOME/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test', which outputs 'Created topic "test".'. The second command is '\$KAFKA_HOME/bin/kafka-topics.sh --list --zookeeper localhost:2181', which outputs 'test'.

```
orientit@kalyan: ~  
orientit@kalyan:~$ $KAFKA_HOME/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test  
Created topic "test".  
orientit@kalyan:~$  
orientit@kalyan:~$  
orientit@kalyan:~$ $KAFKA_HOME/bin/kafka-topics.sh --list --zookeeper localhost:2181  
test  
orientit@kalyan:~$
```

6. Start the **kafka producer** using below command (New Terminal)

```
$KAFKA_HOME/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic test
```

7. Start the **kafka consumer** using below command (New Terminal)

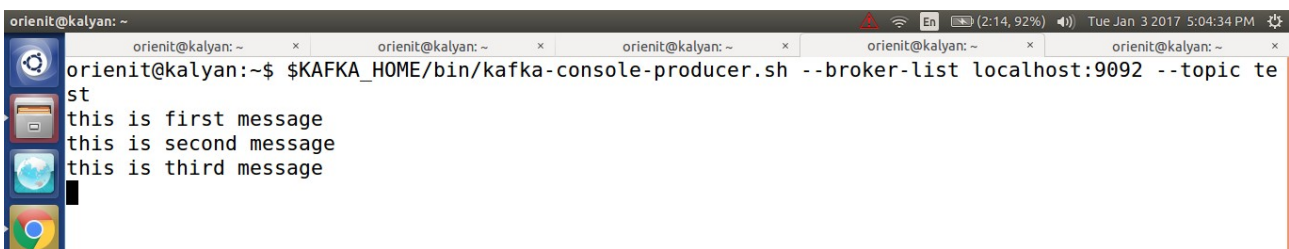
- **Command in Kafka-0.9.x**

```
$KAFKA_HOME/bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic test --from-beginning
```

- **Command in Kafka-0.10.x**

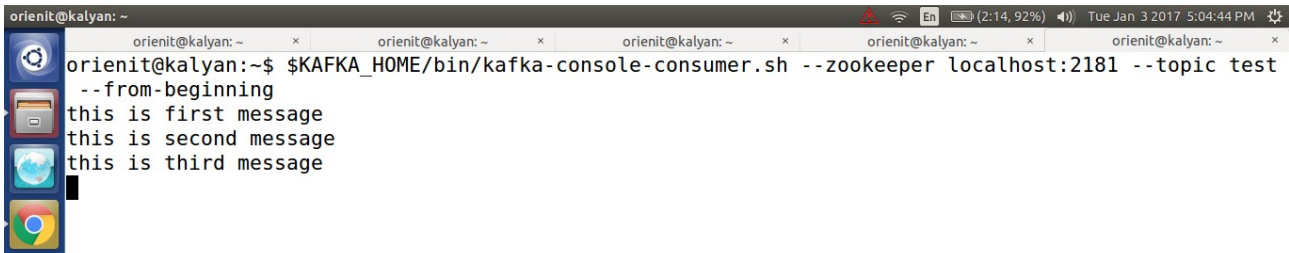
```
$KAFKA_HOME/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic test --from-beginning
```

8. After starting the **kafka producer** then Send some messages to **kafka producer** like below

A terminal window titled 'orientit@kalyan: ~' showing the execution of the Kafka console producer command. The output shows three messages being sent: 'this is first message', 'this is second message', and 'this is third message'.

```
orientit@kalyan: ~  
orientit@kalyan:~$ $KAFKA_HOME/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic test  
st  
this is first message  
this is second message  
this is third message  
█
```

9. After starting the **kafka consumer** then Recieve some messages from **kafka producer** like below

A terminal window titled 'orientit@kalyan: ~' showing the execution of the Kafka console consumer command. The command is '\$KAFKA_HOME/bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic test --from-beginning'. The output shows three messages: 'this is first message', 'this is second message', and 'this is third message'. The terminal has a dark background with a sidebar on the left showing application icons like a gear, folder, and browser.

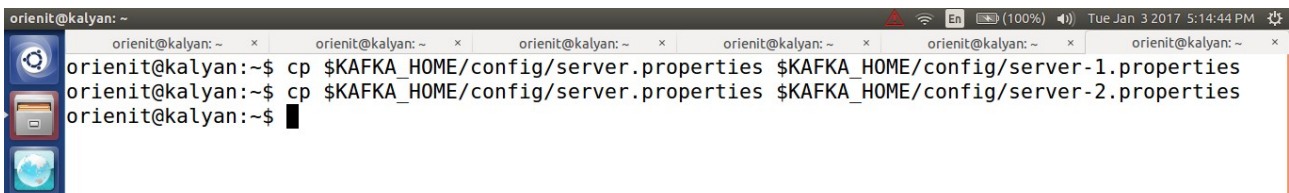
```
orientit@kalyan: ~  
orientit@kalyan:~$ $KAFKA_HOME/bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic test --from-beginning  
this is first message  
this is second message  
this is third message  
█
```

Kafka Cluster Practice Commands

10. Create new configuration files in ``$KAFKA_HOME/config`` folder for kafka brokers

```
cp $KAFKA_HOME/config/server.properties $KAFKA_HOME/config/server-1.properties
```

```
cp $KAFKA_HOME/config/server.properties $KAFKA_HOME/config/server-2.properties
```

A terminal window titled 'orientit@kalyan: ~' showing two commands being executed to copy Kafka configuration files. The first command is 'cp \$KAFKA_HOME/config/server.properties \$KAFKA_HOME/config/server-1.properties' and the second is 'cp \$KAFKA_HOME/config/server.properties \$KAFKA_HOME/config/server-2.properties'. The terminal has a dark background with a sidebar on the left showing application icons.

```
orientit@kalyan: ~  
orientit@kalyan:~$ cp $KAFKA_HOME/config/server.properties $KAFKA_HOME/config/server-1.properties  
orientit@kalyan:~$ cp $KAFKA_HOME/config/server.properties $KAFKA_HOME/config/server-2.properties  
orientit@kalyan:~$ █
```

11. Now edit these new files and set the following properties:

`$KAFKA_HOME/config/server-1.properties`:

```
broker.id=1
```

```
listeners=PLAINTEXT://:9093
```

```
log.dir=/tmp/kafka-logs-1
```

`$KAFKA_HOME/config/server-2.properties`:

```
broker.id=2
```

```
listeners=PLAINTEXT://:9094
```

```
log.dir=/tmp/kafka-logs-2
```

12. Start the `**kafka server**` using below command (New Terminal)

```
$KAFKA_HOME/bin/kafka-server-start.sh $KAFKA_HOME/config/server-1.properties
```

13. Start the `**kafka server**` using below command (New Terminal)

```
$KAFKA_HOME/bin/kafka-server-start.sh $KAFKA_HOME/config/server-2.properties
```

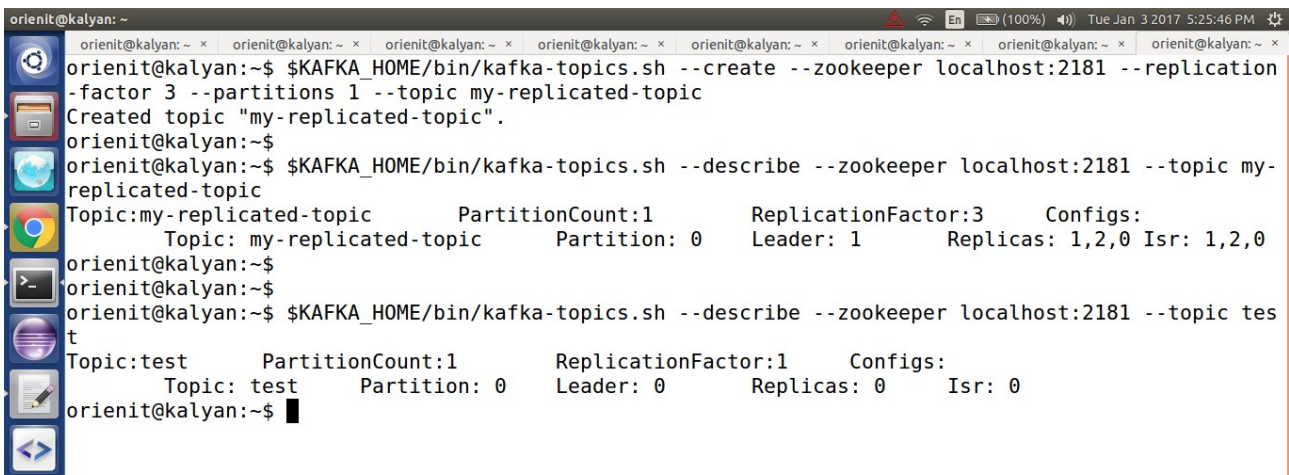

14. Now create a new topic with a replication factor of three:

```
$KAFKA_HOME/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 3 --partitions 1 --topic my-replicated-topic
```

15. Describe the topics run below command

```
$KAFKA_HOME/bin/kafka-topics.sh --describe --zookeeper localhost:2181 --topic my-replicated-topic
```

```
$KAFKA_HOME/bin/kafka-topics.sh --describe --zookeeper localhost:2181 --topic test
```



```
orienit@kalyan: ~
orienit@kalyan: ~ $KAFKA_HOME/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 3 --partitions 1 --topic my-replicated-topic
Created topic "my-replicated-topic".
orienit@kalyan: ~$
orienit@kalyan: ~$ $KAFKA_HOME/bin/kafka-topics.sh --describe --zookeeper localhost:2181 --topic my-replicated-topic
Topic:my-replicated-topic      PartitionCount:1      ReplicationFactor:3      Configs:
Topic: my-replicated-topic      Partition: 0          Leader: 1                 Replicas: 1,2,0 Isr: 1,2,0
orienit@kalyan: ~$
orienit@kalyan: ~$
orienit@kalyan: ~$ $KAFKA_HOME/bin/kafka-topics.sh --describe --zookeeper localhost:2181 --topic test
Topic:test      PartitionCount:1      ReplicationFactor:1      Configs:
Topic: test      Partition: 0          Leader: 0                 Replicas: 0          Isr: 0
orienit@kalyan: ~$
```

16. Start the **`kafka producer`** using below command (New Terminal)

```
$KAFKA_HOME/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic my-replicated-topic
```

17. Start the **`kafka consumer`** using below command (New Terminal)

- **Command in Kafka-0.9.x**

```
$KAFKA_HOME/bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic my-replicated-topic --from-beginning
```

- **Command in Kafka-0.10.x**

```
$KAFKA_HOME/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic my-replicated-topic --from-beginning
```

18. After starting the **`kafka producer`** then Send some messages to **`kafka producer`** like below

```
orientit@kalyan: ~  
orientit@kalyan:~$ $KAFKA_HOME/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic my-replicated-topic  
replica message 1  
replica message 2  
replica message 3
```

19. After starting the **`kafka consumer`** then Recieve some messages from **`kafka producer`** like below

```
orientit@kalyan: ~  
orientit@kalyan:~$ $KAFKA_HOME/bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic my-replicated-topic --from-beginning  
replica message 1  
replica message 2  
replica message 3
```

20. Descibe the topics run below command

`$KAFKA_HOME/bin/kafka-topics.sh --describe --zookeeper localhost:2181 --topic my-replicated-topic`

```
orientit@kalyan: ~  
orientit@kalyan:~$ $KAFKA_HOME/bin/kafka-topics.sh --describe --zookeeper localhost:2181 --topic my-replicated-topic  
Topic:my-replicated-topic      PartitionCount:1      ReplicationFactor:3      Configs:  
Topic: my-replicated-topic      Partition: 0      Leader: 1      Replicas: 1,2,0 Isr: 1,2,0  
orientit@kalyan:~$
```

21. Now let's test out fault-tolerance. Broker 1 was acting as the leader (as per above screenshot)

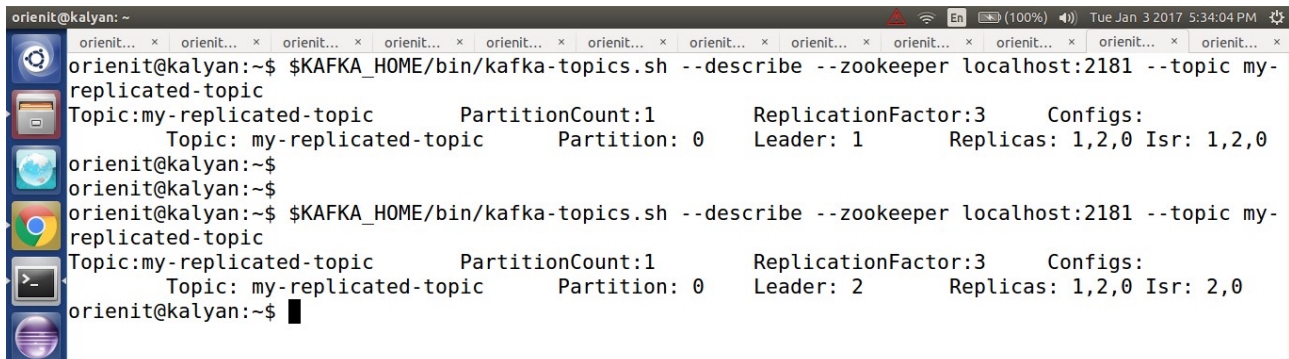
22. Get the process id of Broker 1 then kill it using below command

`ps aux | grep server-1.properties`
`kill -9 26860`

```
orientit@kalyan: ~  
orientit@kalyan:~$ ps aux | grep server-1.properties  
orientit 26860 2.5 1.2 4629164 208580 pts/20 Sl+ 17:21 0:16 /usr/lib/jvm/java-1.8.0-openjdk/bin/java -Xmx1G -Xms1G -server -XX:+UseG1GC -XX:MaxGCPauseMillis=20 -XX:InitiatingHeapOccupancyPercent=35 -XX:+DisableExplicitGC -Djava.awt.headless=true -Xloggc:/home/orientit/work/kafka_2.11-0.9.0.0/bin/../logs/kafkaServer-gc.log -verbose:gc -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:+PrintGCTimeStamps -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false -Dkafka.logs.dir=/home/orientit/work/kafka_2.11-0.9.0.0/bin/../logs -Dlog4j.configuration=file:/home/orientit/work/kafka_2.11-0.9.0.0/bin/../config/log4j.properties -cp /home/orientit/work/kafka_2.11-0.9.0.0/bin/../libs/* kafka.Kafka /home/orientit/work/kafka_2.11-0.9.0.0/config/server-1.properties  
orientit 28584 0.0 0.0 13700 2128 pts/24 S+ 17:32 0:00 grep --color=auto server-1.properties  
orientit@kalyan:~$  
orientit@kalyan:~$ kill -9 26860
```

23. Describe the topics run below command

```
$KAFKA_HOME/bin/kafka-topics.sh --describe --zookeeper localhost:2181 --topic my-replicated-topic
```



The screenshot shows a terminal window with the command `$KAFKA_HOME/bin/kafka-topics.sh --describe --zookeeper localhost:2181 --topic my-replicated-topic` being executed twice. The output for both runs is:

```
Topic:my-replicated-topic      PartitionCount:1      ReplicationFactor:3      Configs:
Topic: my-replicated-topic      Partition: 0          Leader: 1                Replicas: 1,2,0 Isr: 1,2,0
```

The first run shows Broker 1 as the leader, and the second run shows Broker 2 as the leader.

24. Broker 2 was acting as the leader (as per above screenshot)

25. But the messages are still available for consumption even though the leader that took the writes originally is down:

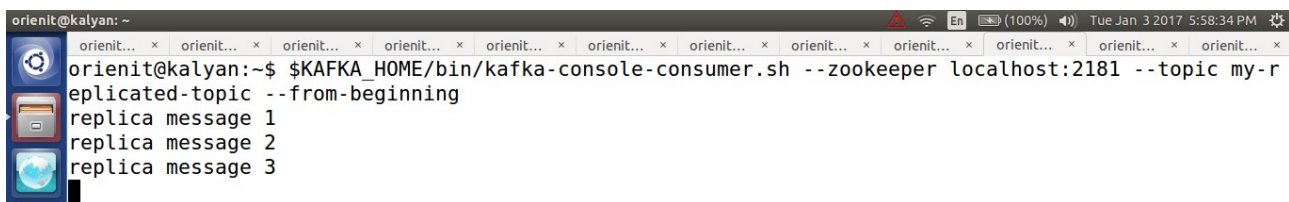
26. Start the **`kafka consumer`** using below command (New Terminal)

- **Command in Kafka-0.9.x**

```
$KAFKA_HOME/bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic my-replicated-topic --from-beginning
```

- **Command in Kafka-0.10.x**

```
$KAFKA_HOME/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic my-replicated-topic --from-beginning
```



The screenshot shows a terminal window with the command `$KAFKA_HOME/bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic my-replicated-topic --from-beginning` being executed. The output is:

```
replica message 1
replica message 2
replica message 3
```

27. As per above information we can understand **`kafka provides fault-tolerant`**