Installing and using the Strimzi Operator with minikube

Install `minikube`

Make certain you have **`kubectl`** installed. You can install `kubectl` according to the instructions in Install and Set Up kubectl.

If you do not already have a hypervisor installed, install one of these now:

- KVM, which also uses QEMU
- VirtualBox

Download the latest version of minikube
https://github.com/kubernetes/minikube/releases

```
$ sudo dnf install <path to>minikube-1.9.2-0.x86_64.rpm
```

or

```
$ curl -L0 minikube
https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64 \
&& chmod +x minikube

$ sudo mkdir -p /usr/local/bin/
$ sudo install minikube /usr/local/bin/

$ minikube version
minikube version: v1.9.2
commit: 93af9c1e43cab9618e301bc9fa720c63d5efa393
```

See https://kubernetes.io/docs/tasks/tools/install-minikube/ for additional info.

Start minikube

You can choose the driver, options are virtualbox, kvm, and podman (experimental).

```
$ minikube start --driver=virtualbox --kubernetes-version=1.18.0
😄 minikube v1.9.2 on Fedora 31
✨  Using the virtualbox driver based on user configuration
💿  Downloading VM boot image ...
    > minikube-v1.9.0.iso.sha256: 65 B / 65 B [--------------] 100.00% ? p/s 0s
    > minikube-v1.9.0.iso: 174.93 MiB / 174.93 MiB [-] 100.00% 9.20 MiB p/s 20s
👍  Starting control plane node m01 in cluster minikube
💾  Downloading Kubernetes v1.18.0 preload ...
    > preloaded-images-k8s-v2-v1.18.0-docker-overlay2-amd64.tar.lz4: 542.91 MiB
🔥  Creating virtualbox VM (CPUs=2, Memory=5900MB, Disk=20000MB) ...
🐳  Preparing Kubernetes v1.18.0 on Docker 19.03.8 ...
🌟  Enabling addons: default-storageclass, storage-provisioner
🏄  Done! kubectl is now configured to use "minikube"
```

```
$ minikube status
m01
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

Install the Strimzi Operator

```
$ curl -L0 https://strimzi.io/install/latest
```

```
$ curl -L0 https://strimzi.io/install/latest | sed 's/namespace: .*/namespace: default/' | kubectl apply -f -
```

```
$ kubectl get pods
```

```
NAME                                        READY   STATUS    RESTARTS   AGE
strimzi-cluster-operator-6c8d574d49-hch89   1/1     Running   0          91s
```

Create the Kafka Cluster

```
$ curl -L0 https://strimzi.io/examples/latest/kafka/kafka-persistent.yaml | vi -
```

```
apiVersion: kafka.strimzi.io/v1beta1
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    version: 2.4.0
    replicas: 3
    listeners:
      plain: {}
      tls: {}
      external:
        type: loadbalancer
        tls: false
    config:
      offsets.topic.replication.factor: 3
      transaction.state.log.replication.factor: 3
      transaction.state.log.min.isr: 2
      log.message.format.version: "2.4"
    storage:
      type: jbod
      volumes:
      - id: 0
        type: persistent-claim
        size: 100Gi
        deleteClaim: false
  zookeeper:
    replicas: 3
    storage:
      type: persistent-claim
      size: 100Gi
      deleteClaim: false
  entityOperator:
    topicOperator: {}
    userOperator: {}
:w ! kubectl apply -f -
```

```
kafka.kafka.strimzi.io/my-cluster created
```

Use 'minikube tunnel' to expose external IP

`minikube tunnel` runs as a process, creating a network route on the host to the service CIDR of the cluster using the cluster's IP address as a gateway. The tunnel command exposes the external IP directly to any program running on the host operating system.

In another terminal, run:

```
$ minikube tunnel
```

```
Status:
      machine: minikube
      pid: 112455
      route: 10.96.0.0/12 -> 192.168.99.102
      minikube: Running
      services: [my-cluster-kafka-0, my-cluster-kafka-1, my-cluster-kafka-2,
my-cluster-kafka-external-bootstrap]
    errors:
            minikube: no errors
            router: no errors
            loadbalancer emulator: no errors
```

```
$ kubectl get pods
```

```
NAME                                          READY   STATUS    RESTARTS   AGE
my-cluster-entity-operator-5dd6d7f5bd-c76b4   3/3     Running   0          88m
my-cluster-kafka-0                            2/2     Running   0          88m
my-cluster-kafka-1                            2/2     Running   0          88m
my-cluster-kafka-2                            2/2     Running   0          88m
my-cluster-zookeeper-0                        2/2     Running   0          90m
my-cluster-zookeeper-1                        2/2     Running   0          90m
my-cluster-zookeeper-2                        2/2     Running   0          90m
strimzi-cluster-operator-6c8d574d49-hch89     1/1     Running   0          107m
```

$ kubectl get kafka -o yaml

```
apiVersion: v1
items:
- apiVersion: kafka.strimzi.io/v1beta1
  kind: Kafka
  metadata:
    annotations:
      kubectl.kubernetes.io/last-applied-configuration: |

{"apiVersion":"kafka.strimzi.io/v1beta1","kind":"Kafka","metadata":{"annotations"
:{},"name":"my-cluster","namespace":"default"},"spec":{"entityOperator":{"topicOp
erator":{},"userOperator":{}},"kafka":{"config":{"log.message.format.version":"2.
4","offsets.topic.replication.factor":3,"transaction.state.log.min.isr":2,"transa
ction.state.log.replication.factor":3},"listeners":{"external":{"tls":false,"type
":"loadbalancer"},"plain":{},"tls":{}},"replicas":3,"storage":{"type":"jbod","vol
umes":[{"deleteClaim":false,"id":0,"size":"100Gi","type":"persistent-claim"}]},"v
ersion":"2.4.0"},"zookeeper":{"replicas":3,"storage":{"deleteClaim":false,"size":
"100Gi","type":"persistent-claim"}}}}
    creationTimestamp: "2020-04-15T14:40:27Z"
    generation: 1
    managedFields:
    - apiVersion: kafka.strimzi.io/v1beta1
      fieldsType: FieldsV1
      fieldsV1:
        f:spec:
          f:entityOperator:
            f:topicOperator:
              f:reconciliationIntervalSeconds: {}
              f:topicMetadataMaxAttempts: {}
              f:zookeeperSessionTimeoutSeconds: {}
            f:userOperator:
              f:reconciliationIntervalSeconds: {}
              f:zookeeperSessionTimeoutSeconds: {}
        f:status:
          f:conditions: {}
          f:listeners: {}
          f:observedGeneration: {}
      manager: okhttp
      operation: Update
      time: "2020-04-15T14:43:17Z"
    name: my-cluster
```

```
  namespace: default
  resourceVersion: "4802"
  selfLink: /apis/kafka.strimzi.io/v1beta1/namespaces/default/kafkas/my-cluster
  uid: 55993681-f909-40ad-ba4c-b94968aba11f
spec:
  entityOperator:
    topicOperator: {}
    userOperator: {}
  kafka:
    config:
      log.message.format.version: "2.4"
      offsets.topic.replication.factor: 3
      transaction.state.log.min.isr: 2
      transaction.state.log.replication.factor: 3
    listeners:
      external:
        tls: false
        type: loadbalancer
      plain: {}
      tls: {}
    replicas: 3
    storage:
      type: jbod
      volumes:
      - deleteClaim: false
        id: 0
        size: 100Gi
        type: persistent-claim
    version: 2.4.0
  zookeeper:
    replicas: 3
    storage:
      deleteClaim: false
      size: 100Gi
      type: persistent-claim
status:
  conditions:
  - lastTransitionTime: 2020-04-15T14:43:17+0000
    status: "True"
    type: Ready
  listeners:
  - addresses:
```

```
      - host: my-cluster-kafka-bootstrap.default.svc
        port: 9092
      type: plain
    - addresses:
      - host: my-cluster-kafka-bootstrap.default.svc
        port: 9093
      certificates:
      - |
        -----BEGIN CERTIFICATE-----
        MIIDLTCCAhWgAwIBAgIJAPdagdf45X7zMA0GCSqGSIb3DQEBCwUAMC0xEzARBgNV
        BAoMCmlvLnN0cmltemkxFjAUBgNVBAMMDWNsdXN0ZXItY2EgdjAwHhcNMjAwNDE1
        MTQ0MDI4WhcNMjEwNDE1MTQ0MDI4WjAtMRMwEQYDVQQKDAppby5zdHJpbXppMRYw
        FAYDVQQDDA1jbHVzdGVyLWNhIHYwMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
        CgKCAQEAxs76beb0fNze2Qo5LP83us2VNX49PPnEfOizSpl2SgaXzp3MP5WpFGZL
        fPNZzJjie1MYFQy2ivBYAIB/D486j8YZ6J/l3BCZ2HHXC+xcBzL8AvUBoUN+yf1G
        w9ZNMBXGcjHwXT5WCRSnUQHYlzIWbgUAm/Bk6DxNm31yf4udrkG9cYNvgAYPA5eY
        UXOA4Pfffuo/n2ITJtuZda5FLzmZBvYuVE6ctDAE6InPrgUmPWWgbs/L7Pk0HAVx
        bTIupJ175hCpOTaYEJFAjW2pBYiETwrhENymZtmttsCrhF1CZKCzeknNrJ5x2ZNx
        Z0byPTuFHnnJH8cE3Xz4BzSgDIg/CQIDAQABo1AwTjAdBgNVHQ4EFgQU1qzZhNtW
        ZObgH+eDvnKowaHZNb8wHwYDVR0jBBgwFoAU1qzZhNtWZObgH+eDvnKowaHZNb8w
        DAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQsFAAOCAQEAA6vgdjg/L4Pt5UHu0G5S
        N77HHC8QNLIqutzb9BX89nuMappJW4QKztJMgyyy+XTxxEbm/I3ZZ6fzowJ+Nyh0
        4yYfME/eKtnToUHpRFRrxfHGX4txMBhtgL4Zx4snWUPPSur73xbPDak+BEbqy2Ij
        3+JuXDXqXxoXtbRcOp8A5r5XnsdYev015aN/XVwlbBt4NlNZL2rp6J5jg72ZVJLW
        EUsWRRBcyjF7TfWeIaP4WENH/NMzum0OMKUTr2DJXE6CQ+8aNH0mYkMMSFItvQvL
        YEFj9fiAyb4Vlx/QiTMlVx1h7IL4GqqPnj26tTGfQNH+uILgSCP0g/GFNDzdUS0E
        VA==
        -----END CERTIFICATE-----
      type: tls
    - addresses:
      - host: 10.102.68.38    //Note the IP address
        port: 9094            //Note the port
      type: external
    observedGeneration: 1
kind: List
metadata:
  resourceVersion: ""
  selfLink: ""
```

Create a Kafka topic

Change the partitions and replicas to **3** for a more compelling demo.

```
$ curl -L0 https://strimzi.io/examples/latest/topic/kafka-topic.yaml | vi -
```

```
apiVersion: kafka.strimzi.io/v1beta1
kind: KafkaTopic
metadata:
  name: my-topic
  labels:
    strimzi.io/cluster: my-cluster
spec:
  partitions: 3
  replicas: 3
  config:
    retention.ms: 7200000
    segment.bytes: 1073741824
:w ! kubectl apply -f -
```

Edit and run the Producer/Consumer *(see files below)*

$ **cd** ***<path to source>***

$ **code .**

Add the server IP and port from the output of the '**kubectl get kafka -o yaml**' command to both Producer.java (line 31) and Consumer.java (line 32):

```
        -----END CERTIFICATE-----
      type: tls
    - addresses:
      - host: 10.102.68.38
        port: 9094
      type: external
    observedGeneration: 1
kind: List
metadata:
  resourceVersion: ""
  selfLink: ""

/*
 * Producer configuration
 */
    Map<String, Object> props = new HashMap();
    props.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG, "10.102.68.38:9094");


/*
 * Consumer configuration
 */
    Map<String, Object> props = new HashMap();
    props.put(ConsumerConfig.BOOTSTRAP_SERVERS_CONFIG, "10.102.68.38:9094");
```

Run Producer.java, notice the message was sent to partition 1, offset 1:

```
[main] INFO io.strimzi.demo.Producer - Message sent (topic my-topic, partition 1, offset 1)
```

Run Consumer.java, notice the 'Hello World' message was received:

```
[main] INFO io.strimzi.demo.Consumer - Received message: null / Hello World (from topic my-topic, partition 1, offset 1)
```

Stop minikube

```
$ minikube stop
✋  Stopping "minikube" in virtualbox ...
🛑  Node "m01" stopped.
```

Client Examples

Producer.java

```
package io.strimzi.demo;

import org.apache.kafka.clients.CommonClientConfigs;
import org.apache.kafka.clients.producer.KafkaProducer;
import org.apache.kafka.clients.producer.ProducerConfig;
import org.apache.kafka.clients.producer.ProducerRecord;
import org.apache.kafka.clients.producer.RecordMetadata;
import org.apache.kafka.common.config.SaslConfigs;
import org.apache.kafka.common.config.SslConfigs;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import java.util.HashMap;
import java.util.Map;
import java.util.concurrent.ExecutionException;

public class Producer {
    private static Logger LOG = LoggerFactory.getLogger(Producer.class);

    public static void main(String[] args) throws ExecutionException,
InterruptedException {
        /*
         * Configure the logger
         */
        System.setProperty("org.slf4j.simpleLogger.defaultLogLevel", "info");
        System.setProperty("org.slf4j.simpleLogger.showThreadName", "false");

        /*
         * Producer configuration
         */
        Map<String, Object> props = new HashMap();
        props.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG, "10.102.68.38:9094");
        props.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringSerializer");
        props.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringSerializer");
        props.put(CommonClientConfigs.SECURITY_PROTOCOL_CONFIG, "PLAINTEXT");
```

```
        //props.put(SslConfigs.SSL_TRUSTSTORE_PASSWORD_CONFIG, "EEvAqISFdgEw");
        //props.put(SslConfigs.SSL_TRUSTSTORE_LOCATION_CONFIG, "./ca.p12");
        //props.put(SaslConfigs.SASL_MECHANISM, "SCRAM-SHA-512");
        //props.put(SaslConfigs.SASL_JAAS_CONFIG,
"org.apache.kafka.common.security.scram.ScramLoginModule required
username=\"my-user\" password=\"RcusRrjvimu2\";");

        KafkaProducer<String, String> producer = new KafkaProducer<String,
String>(props);

        /*
         * Produce messages
         */
        ProducerRecord record = new ProducerRecord<String, String>("my-topic",
"Hello World");
        RecordMetadata result = (RecordMetadata) producer.send(record).get();
        LOG.info("Message sent (topic {}, partition {}, offset {})",
                result.topic(),
                result.partition(),
                result.offset());

        /*
         * Close the producer
         */
        producer.close();
    }

}
```

Red Hat

Consumer.java

```
package io.strimzi.demo;

import org.apache.kafka.clients.CommonClientConfigs;
import org.apache.kafka.clients.consumer.ConsumerConfig;
import org.apache.kafka.clients.consumer.ConsumerRecord;
import org.apache.kafka.clients.consumer.ConsumerRecords;
import org.apache.kafka.clients.consumer.KafkaConsumer;
import org.apache.kafka.common.config.SaslConfigs;
import org.apache.kafka.common.config.SslConfigs;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import java.time.Duration;
import java.util.Collections;
import java.util.HashMap;
import java.util.Map;

public class Consumer {
    private static Logger LOG = LoggerFactory.getLogger(Consumer.class);

    public static void main(String[] args) {
        /*
         * Configure the logger
         */
        System.setProperty("org.slf4j.simpleLogger.defaultLogLevel", "info");
        System.setProperty("org.slf4j.simpleLogger.showThreadName", "false");

        /*
         * Consumer configuration
         */
        Map<String, Object> props = new HashMap();
        props.put(ConsumerConfig.BOOTSTRAP_SERVERS_CONFIG, "10.102.68.38:9094");
        props.put(ConsumerConfig.GROUP_ID_CONFIG, "my-group");
        props.put(ConsumerConfig.AUTO_COMMIT_INTERVAL_MS_CONFIG, "1000");
        props.put(ConsumerConfig.SESSION_TIMEOUT_MS_CONFIG, "30000");
        props.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringDeserializer");
        props.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringDeserializer");
        props.put(ConsumerConfig.AUTO_OFFSET_RESET_CONFIG, "earliest");
```

```
        props.put(CommonClientConfigs.SECURITY_PROTOCOL_CONFIG, "PLAINTEXT");
        //props.put(SslConfigs.SSL_TRUSTSTORE_PASSWORD_CONFIG, "EEvAqISFdgEw");
        //props.put(SslConfigs.SSL_TRUSTSTORE_LOCATION_CONFIG, "./ca.p12");
        //props.put(SaslConfigs.SASL_MECHANISM, "SCRAM-SHA-512");
        //props.put(SaslConfigs.SASL_JAAS_CONFIG,
"org.apache.kafka.common.security.scram.ScramLoginModule required
username=\"my-user\" password=\"RcusRrjvimu2\";");


        KafkaConsumer<String, String> consumer = new KafkaConsumer<String,
String>(props);

        /*
         * Consume messages
         */
        consumer.subscribe(Collections.singletonList("my-topic"));
        ConsumerRecords<String, String> records =
consumer.poll(Duration.ofSeconds(30));

        if(records.isEmpty()) {
            LOG.info("No message received :-(");
        } else {
            for (ConsumerRecord<String, String> record : records)
            {
                LOG.info("Received message: {} / {} (from topic {}, partition {},
offset {})",
                        record.key(),
                        record.value(),
                        record.topic(),
                        record.partition(),
                        record.offset());
            }

            consumer.commitSync();
        }

        /*
         * Close the consumer
         */
        consumer.close();
    }
}
```

Pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
   <modelVersion>4.0.0</modelVersion>

   <groupId>io.strimzi.demo</groupId>
   <artifactId>develop-kafka-apps-with-strimzi</artifactId>
   <version>1.0-SNAPSHOT</version>

   <dependencies>
       <dependency>
           <groupId>org.apache.kafka</groupId>
           <artifactId>kafka-clients</artifactId>
           <version>2.4.1</version>
       </dependency>
       <dependency>
           <groupId>org.slf4j</groupId>
           <artifactId>slf4j-simple</artifactId>
           <version>1.7.30</version>
       </dependency>
   </dependencies>

   <build>
       <plugins>
           <plugin>
               <groupId>org.apache.maven.plugins</groupId>
               <artifactId>maven-compiler-plugin</artifactId>
               <configuration>
                   <source>1.8</source>
                   <target>1.8</target>
               </configuration>
           </plugin>
       </plugins>
   </build>

</project>
```

View the minikube dashboard

```
$ minikube dashboard

🤔  Verifying dashboard health ...
🚀  Launching proxy ...
🤔  Verifying proxy health ...
🎉  Opening
http://127.0.0.1:36805/api/v1/namespaces/kubernetes-dashboard/services/http:kuber
netes-dashboard:/proxy/ in your default browser...
[205204:205204:0416/155240.945361:ERROR:edid_parser.cc(102)] Too short EDID data:
manufacturer id
[205204:205204:0416/155240.945618:ERROR:edid_parser.cc(102)] Too short EDID data:
manufacturer id
Opening in existing browser session.
```