Creating uvuyo cluster

Installation · https://2yetis.net/docs/creating-uvuyo-cluster/

If you want to run in uvuyo in a high available (HA) environment you need to install uvuyo in a cluster. An uvuyo cluster consists of the uvuyo node cluster and the kafka cluster. Both - the uvuyo node cluster and the kafka cluster - build a Raft cluster and therefor should have at least three nodes or an uneven number of nodes to ensure that a leader of the cluster can be elected. Technically we have two clusters: the uvuyo node cluster and the kafka cluster.

Configuring the kafka cluster

If you already had kafka running on one node and now you want to create a cluster you would need to setup the kafka cluster again. Therefor make sure you shutdown uvuyo (and the kafka cluster) and make a backup of the $UVUYO_HOME directory.

Note

The cluster uses the ports 9092 and 9093 to communicate. When you run the nodes on different machines you need to make sure that the ports are opened in both directions is you have a firewall between the nodes.

You can change the ports used. For simplicity we will stick to the default ports 9092 and 9093 in this documentation.

Delete the kafka logs

After creating the backup you need to delete the kafka data. The kafka data is usually stored in the $UVUYO_HOME/data/kafka directory. To verify where the kafka data is stored open the $UVUYO_HOME/etc/kafka.properties file and search for the `log.dirs` property. This points to the directory where the kafka logs are stored.

After backing up the directory, delete the directory. Be sure that the kafka server is not running.

```
rm -rf $UVUYO_HOME/data/kafka
```

Edit the kafka configuration

Now edit the $UVUYO_HOME/etc/kafka.properties file and configure the kafka node. This step needs to be repeated on each node.

Create a unique Node id `(node.id)`

Create a node.id for every cluster node. The node id must be unique for every cluster node.

Configure the role of the kafka node `(process.roles)`

Every node in a kafka cluster can have the role broker or controller or both (broker and controller). A broker node can host kafka topics while a controller node is used to control the cluster. In large kafka clusters you might consider having dedicated controller and broker nodes. For now we will configure our nodes are having both roles broker and controller.

Therefor set the process.roles=broker, controller in your $UVUYO_HOME/etc/kafka.properties file.

Configure the known kafka controllers `(controller.quorum.voters)`

Each controller node in a kafka cluster needs to know the other controller nodes in the kafka cluster. We configure them by adding the `controller.quorum.voters`.

All of the servers (controllers and brokers) in a Kafka cluster discover the quorum voters using this property, and you must identify all of the controllers by including them in the list you provide for the property.

Each controller is identified with their ID, host and port information in the format of `{id}@{host}:{port}`. Multiple entries are separated by commas.

Configure Replication

To ensure HA the kafka topics need to be replicated in the cluster. Therefor we need to set default replication factor for topic and how many replicas must be insync. The `default.replication.factor` must be set to 3 and the `min.insync.replicas` must be set to 2. This means that we have three

Example $UVUYO_HOME/etc/kafka.properties

This is an example of the properties file. Be aware that the that the configuration file will also include additional properties.

```
# Unique ID for this node (must be different per broker)
node.id=1

# Roles this node plays
process.roles=broker,controller

# List of all controllers in the cluster (use hostnames/IPs) replace the
# hostnames with the real hostnames in your cluster
# controller.quorum.voters=1@broker1:9093,2@broker2:9093,3@broker3:9093

# KRaft listeners
listeners=PLAINTEXT://broker1:9092,CONTROLLER://broker1:9093
advertised.listeners=PLAINTEXT://broker1:9092

# Internal topic replication
default.replication.factor=3
min.insync.replicas=2
```

Configure the cluster ID

Each kafka node needs to have the same cluster ID. We create a unique cluster Id with the following command.

```
CLUSTER_ID=$(bin/kafka-storage.sh random-uuid)
```

On each server you need to format the storage using the cluster Id created above.

```
bin/kafka-storage.sh format -t $CLUSTER_ID -c config/kraft-server.properties
```

After this you can start the uvuyo server again.