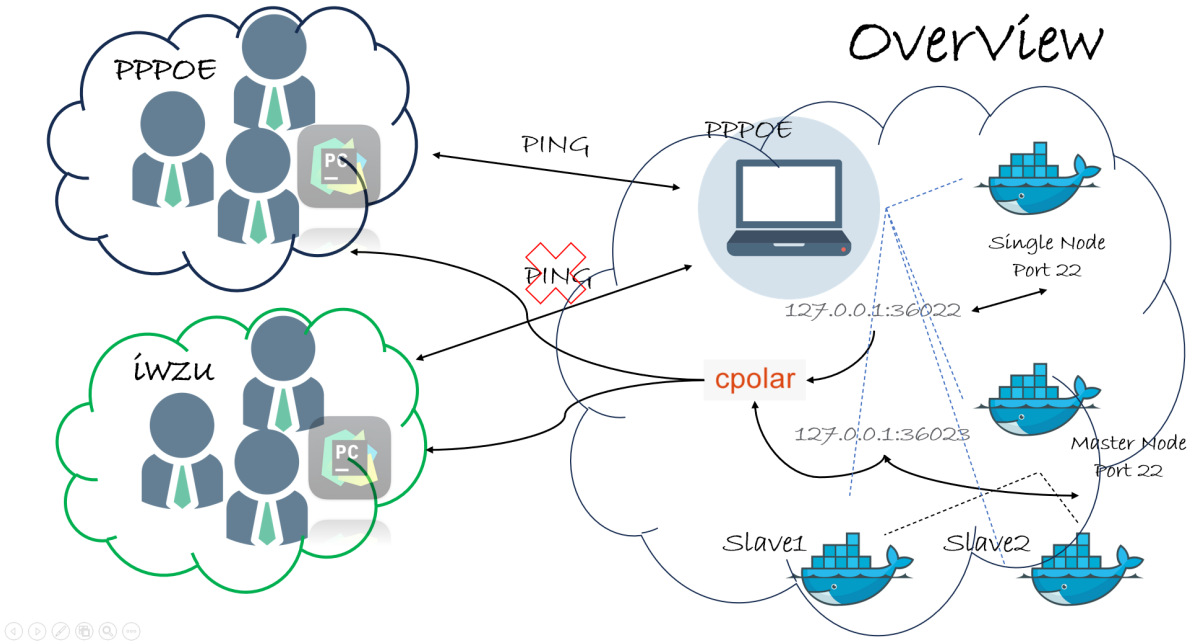
小组工作流程及项目介绍

1. 工作环境

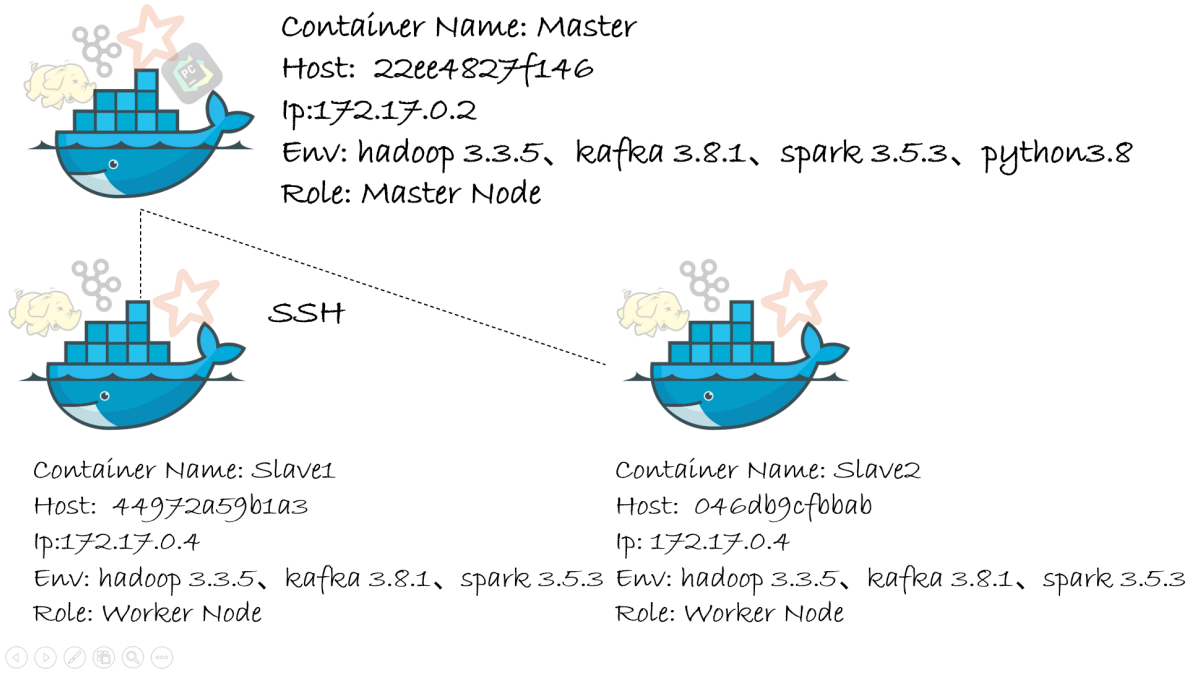
由于该项目分为`数据采集`、`数据读取及消息队列`、`spark数据处理`、`实时数据展示`等部分，且每个小组成员负责每一部分，各个部分又存在依赖，所有选择一台主机作为宿主机是最好的选择，保证代码同步和互用。



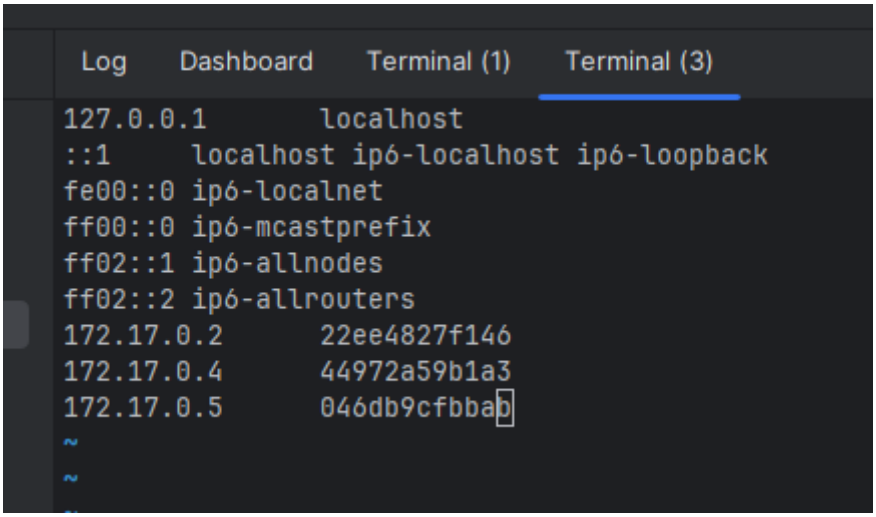
其中,`Single Node`节点用于在等待分布式环境搭建时进行单机测试；`Master Node`、`Slave1`、`Slave2`为分布式节点.

2. 项目环境

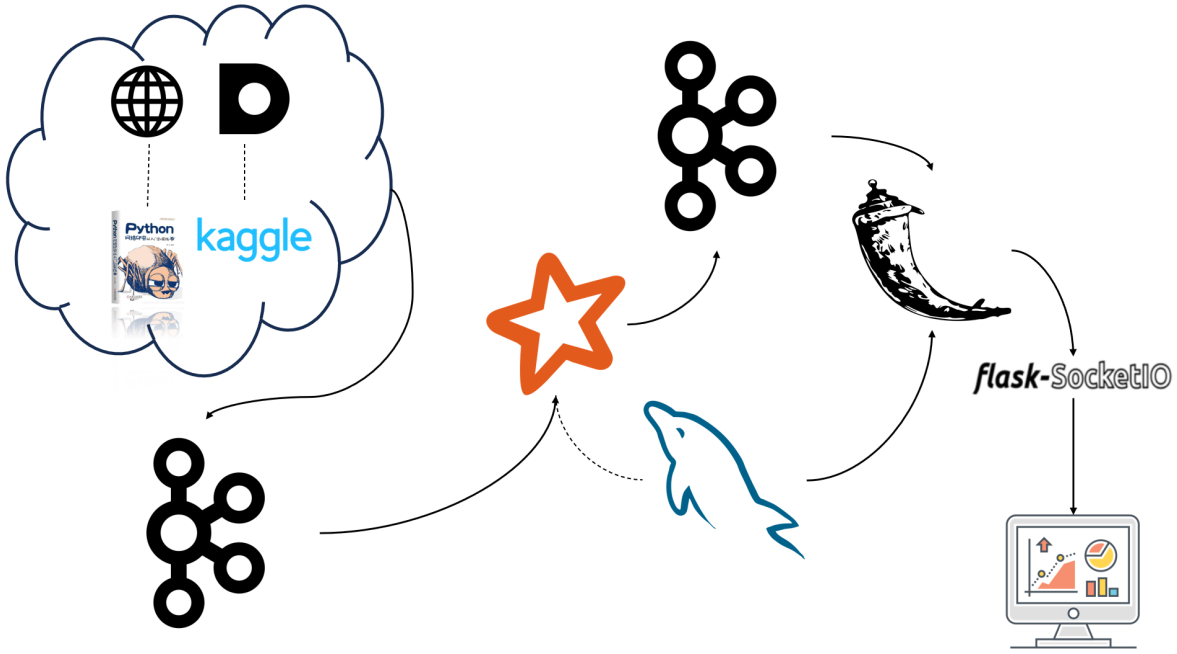
部署分布式环境需要几台处于同一局域网下的机子，其内部的通讯是对外透明的。本次实验的宿主机选择Windows，使用docker desktop创建容器集群，模拟同一局域网，当然也可以docker net创建专用网络实现项目间的隔离。



集群间容器可以相互PING通，且根据hostname进行通讯，配置hosts文件：



3.部署方法



集群部署

集群的三个节点：

hostname	ip	role
22ee4827f146	172.17.0.2	master
44972a59b1a3	172.17.0.4	slave1
046db9cfbbab	172.17.0.5	slave2

kafka集群搭建：修改每个容器的配置文件`server.properties`、`zoo.cfg`：

```
############################# Server Basics #############################

# The id of the broker. This must be set to a unique integer for each broker.
broker.id=0
```

每个节点的唯一标识

```
############################# Socket Server Settings #############################

# The address the socket server listens on. If not configured, the host name will be equal to the value
# java.net.InetAddress.getCanonicalHostName(), with PLAINTEXT listener name, and port 9092.
#   FORMAT:
#     listeners = listener_name://host_name:port
#   EXAMPLE:
#     listeners = PLAINTEXT://your.host.name:9092
#listeners=PLAINTEXT://:9092
listeners=PLAINTEXT://22ee4827f146:9092
# Listener name, hostname and port the broker will advertise to clients.
# If not set, it uses the value for "listeners".
#advertised.listeners=PLAINTEXT://your.host.name:9092
```

集群中的监听端口要设置成节点的

相关命令：

- kafka集群启动：

```
#分别进入 Master节点和slave1和slave2节点
./bin/zkServer.sh start zoo.cfg #zookeeper启动

#jps 可以看到每个节点都有 3069 QuorumPeerMain 信息
./bin/kafka-server-start.sh -daemon config/server.properties #后台启动kafka

#此时每个节点的jps信息：
2132 Jps
2056 Kafka
1448 QuorumPeerMain


#进入Master容器
#创建topic
 ./bin/kafka-topics.sh --bootstrap-server
22ee4827f146:9092,44972a59b1a3:9092,046db9cfbbab:9092 --create --topic test
--partitions 3 --replication-factor 3
#注意副本因子数不能大于broker数 否则报错

#查看创建的topic的具体描述
./bin/kafka-topics.sh  --bootstrap-server
22ee4827f146:9092,44972a59b1a3:9092,046db9cfbbab:9092 --describe --topic
test


```

```
Topic: test     TopicId: BfEAiDF9QuaarsD589XVhw PartitionCount: 3
ReplicationFactor: 3    Configs:
        Topic: test     Partition: 0    Leader: 1       Replicas: 1,0,2 Isr:
1,0,2       Elr: N/A        LastKnownElr: N/A
        Topic: test     Partition: 1    Leader: 0       Replicas: 0,2,1 Isr:
0,2,1       Elr: N/A        LastKnownElr: N/A
        Topic: test     Partition: 2    Leader: 2       Replicas: 2,1,0 Isr:
2,1,0       Elr: N/A        LastKnownElr: N/A
```

- 相关结果

  对每个节点启动zookeeper、kafka

```
root@22ee4827f146:/usr/local/kafka# ./bin/kafka-server-start.sh -daemon config/server.properties
root@22ee4827f146:/usr/local/kafka# jps
81 QuorumPeerMain
2339 -- process information unavailable
2826 Jps
2714 Kafka
```

创建topic：

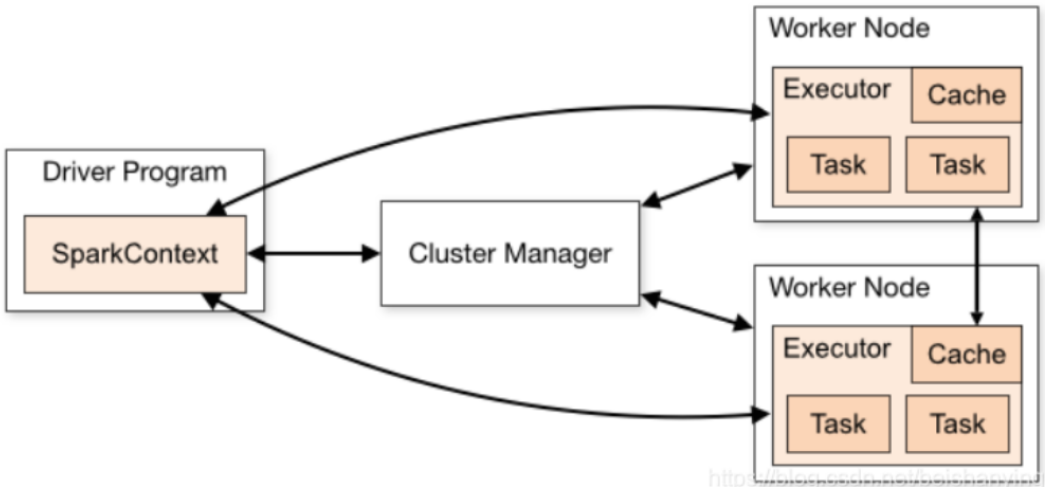
```
root@22ee4827f146:/usr/local/kafka# ./bin/kafka-topics.sh --bootstrap-server 22ee4827f146:9092,44972a59b1a3:9092,046db9cfbbab:9092 --create --topic test --partitions 3 --replication-fa
ctor 3
Created topic test.
```

查看topic信息：

```
root@22ee4827f146:/usr/local/kafka# ./bin/kafka-topics.sh  --bootstrap-server 22ee4827f146:9092,44972a59b1a3:9092,046db9cfbbab:9092 --describe --topic test
[2024-12-14 14:41:18,195] WARN [AdminClient clientId=adminclient-1] The DescribeTopicPartitions API is not supported, using Metadata API to describe topics. (org.apache.kafka.clients.a
dmin.KafkaAdminClient)
Topic: test     TopicId: BfEAiDF9QuaarsD589XVhw PartitionCount: 3       ReplicationFactor: 3    Configs:
        Topic: test     Partition: 0    Leader: 1       Replicas: 1,0,2 Isr: 1,0,2      Elr: N/A        LastKnownElr: N/A
        Topic: test     Partition: 1    Leader: 0       Replicas: 0,2,1 Isr: 0,2,1      Elr: N/A        LastKnownElr: N/A
        Topic: test     Partition: 2    Leader: 2       Replicas: 2,1,0 Isr: 2,1,0      Elr: N/A        LastKnownElr: N/A
```

> Leader："Leader: 1"表示分区0的领导者是节点 1。在 Kafka 的分区副本机制中，领导者副本负责处理该分区所有的读写请求，是数据交互的核心节点。比如生产者发送消息到这个分区、消费者从这个分区拉取消息，都是和分区的领导者进行直接交互的。
>
> Replicas:"Replicas: 1.0,2"说明分区0的副本分布在节点 1、节点0和节点2 这三个节点上这些副本会不断地从领导者副本那里同步数据，以保持数据的一致性，这样即使领导者副本出现故障其他副本也能及时接替它的工作。
>
> Isr:"sr: 1,0,2"显示当前处于同步状态的副本就是节点 1、节点0和节点 2。同步副本指的是那些已经成功从领导者那里复制了全部数据，并且能够持续跟进领导者的数据更新，保持数据最新状态的副本。Isr 集合中的副本可以在领导者副本故障时参与选举成为新的领导者。

- spark集群搭建：



重点是修改Master的配置文件`spark-env.sh`，以及设置work的host

```
# Options for beeline
# - SPARK_BEELINE_OPTS, to set config properties only for the beeline cli (e.g. "-Dx=y")
# - SPARK_BEELINE_MEMORY, Memory for beeline (e.g. 1000M, 2G) (Default: 1G)
export SPARK_DIST_CLASSPATH=$(/usr/local/hadoop/bin/hadoop classpath):/usr/local/spark/jars/kafka/*:/usr/local/kafka/libs/*
export JAVA_HOME=/usr/local/jdk1.8.0_202
export SPARK_MASTER_HOST=22ee4827f146
export SPARK_MASTER_PORT=7077
```

spark-env.sh

```
# A Spark Worker will be started on each of the machines listed below.
44972a59b1a3
046db9cfbbab
~
~
~
~
```

worker配置从节点

- spark集群启动：

```
#进入Master节点
#一键启动主从节点
sbin/start-all.sh
#返回
starting org.apache.spark.deploy.master.Master, logging to
/usr/local/spark/logs/spark-root-org.apache.spark.deploy.master.Master-1-
22ee4827f146.out
046db9cfbbab: starting org.apache.spark.deploy.worker.Worker, logging to
/usr/local/spark/logs/spark-root-org.apache.spark.deploy.worker.Worker-1-
046db9cfbbab.out
44972a59b1a3: starting org.apache.spark.deploy.worker.Worker, logging to
/usr/local/spark/logs/spark-root-org.apache.spark.deploy.worker.Worker-1-
44972a59b1a3.out
#Master&slaves都会返回info
#Master jps
6569 Master
#slave1 jps
2673 Worker
#slave2 jps
2657 Worker

#Master关闭spark集群
sbin/stop-all.sh
```

- 相关结果：Master节点一键启动主从节点

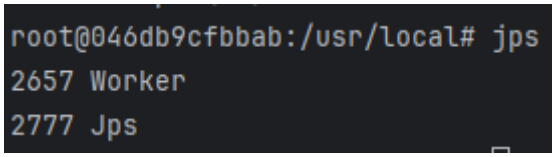
```
root@22ee4827f146:/usr/local/spark# sbin/start-all.sh
starting org.apache.spark.deploy.master.Master, logging to /usr/local/spark/logs/spark-root-org.apache.spark.deploy.master.Master-1-22ee4827f146.out
046db9cfbbab: starting org.apache.spark.deploy.worker.Worker, logging to /usr/local/spark/logs/spark-root-org.apache.spark.deploy.worker.Worker-1-046db9cfbbab.out
44972a59b1a3: starting org.apache.spark.deploy.worker.Worker, logging to /usr/local/spark/logs/spark-root-org.apache.spark.deploy.worker.Worker-1-44972a59b1a3.out
```

观察主从节点的进程：

Master:

```
root@22ee4827f146:/usr/local/spark# jps
2339 -- process information unavailable
6569 Master
6717 Jps
```

Works:



- zookeeper在spark集群中的Role：
  - 帮助 Spark Standalone 高可用
  - 实现 Masters 的主备切换
  - ...

Stay updated ... ...