## 实验一、

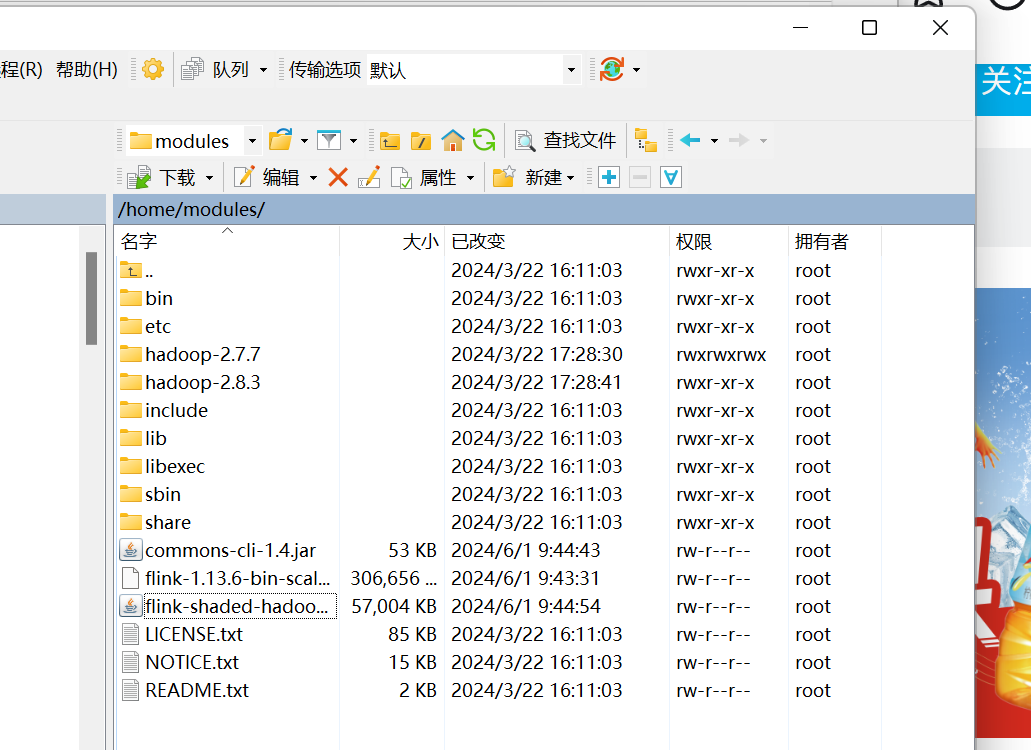
1. 打开8088、8081、8032端口：

安全组——入方向规则——添加规则

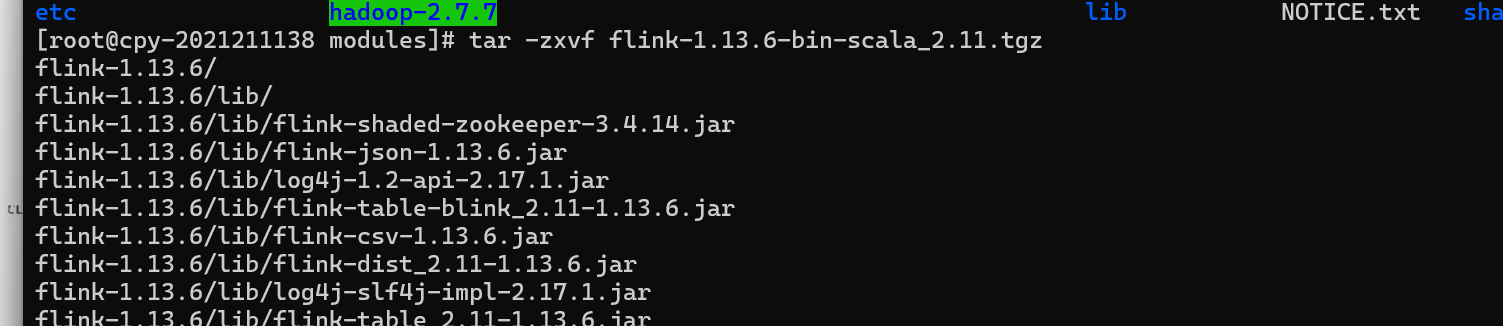


2. 下载flink和依赖包

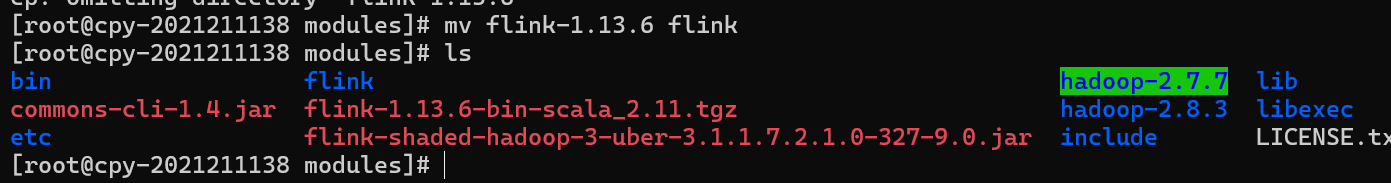
下载好后，上传到/home/modules



解压缩：



重命名



3. 修改环境变量（四台服务器都要改）：

/etc/profile

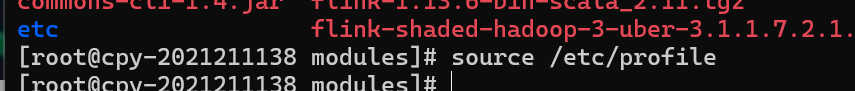
添加：

# Flink

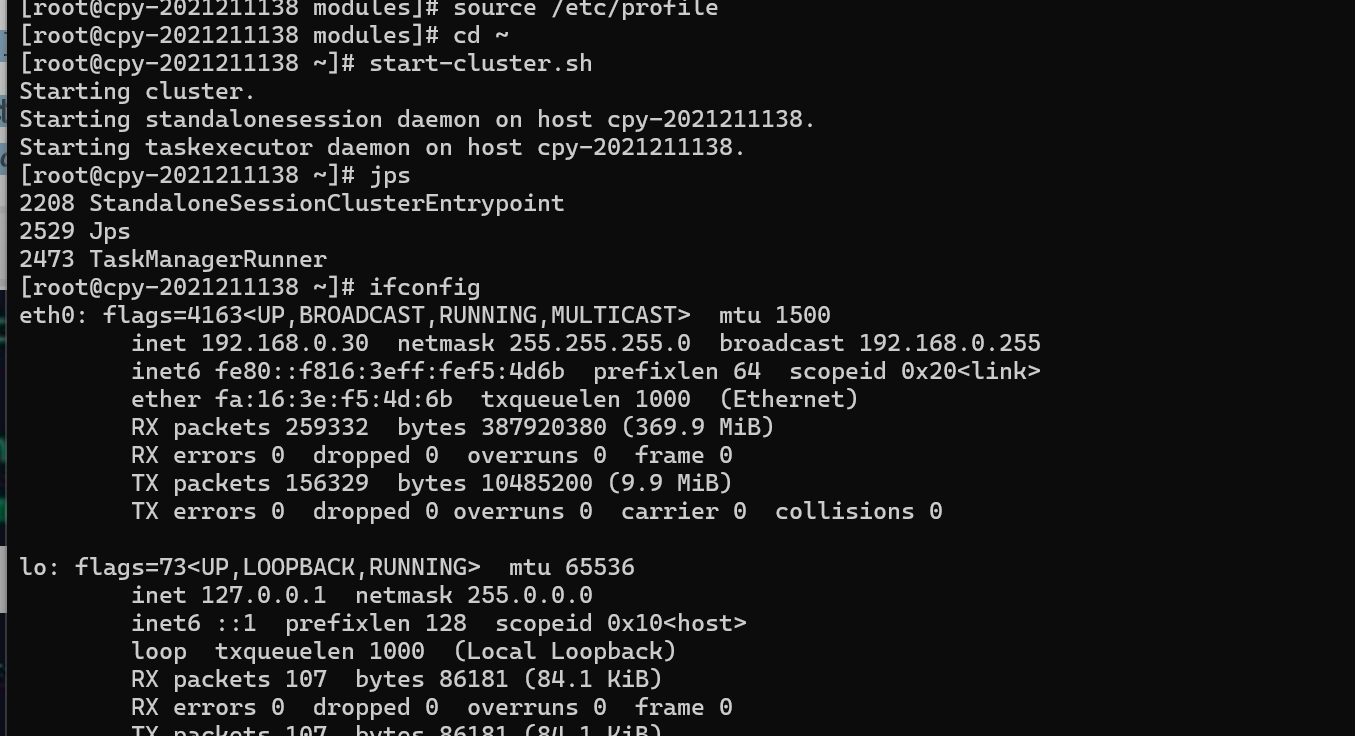
export FLINK\_HOME=/home/modules/flink

export PATH=$FLINK\_HOME/bin:$PATH

再刷新一下环境变量：



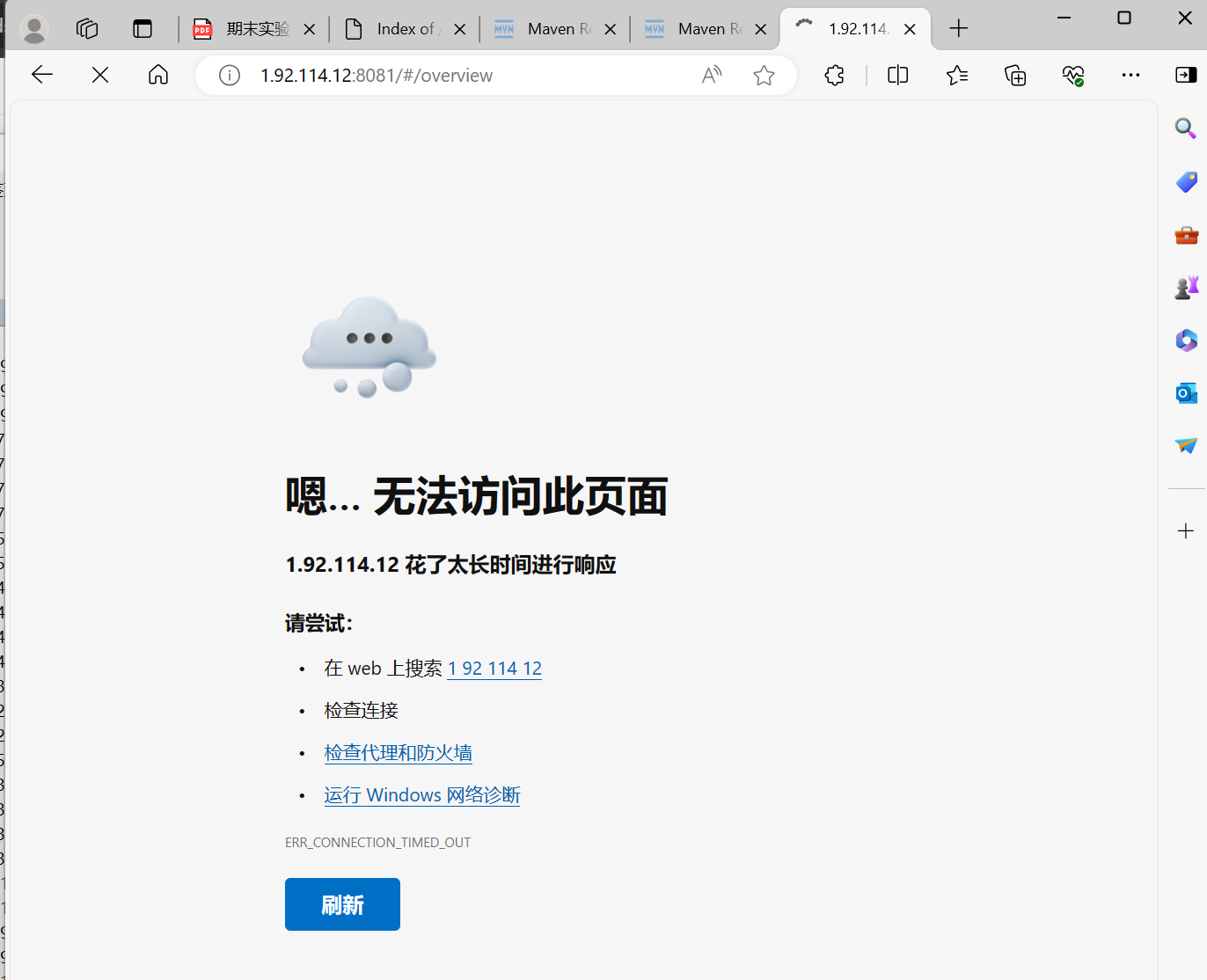
4. 脚本启动Flink



5. Web界面访问

<http://1.92.114.12:8081/#/overview>

问题：



访问不了。

原因如下：

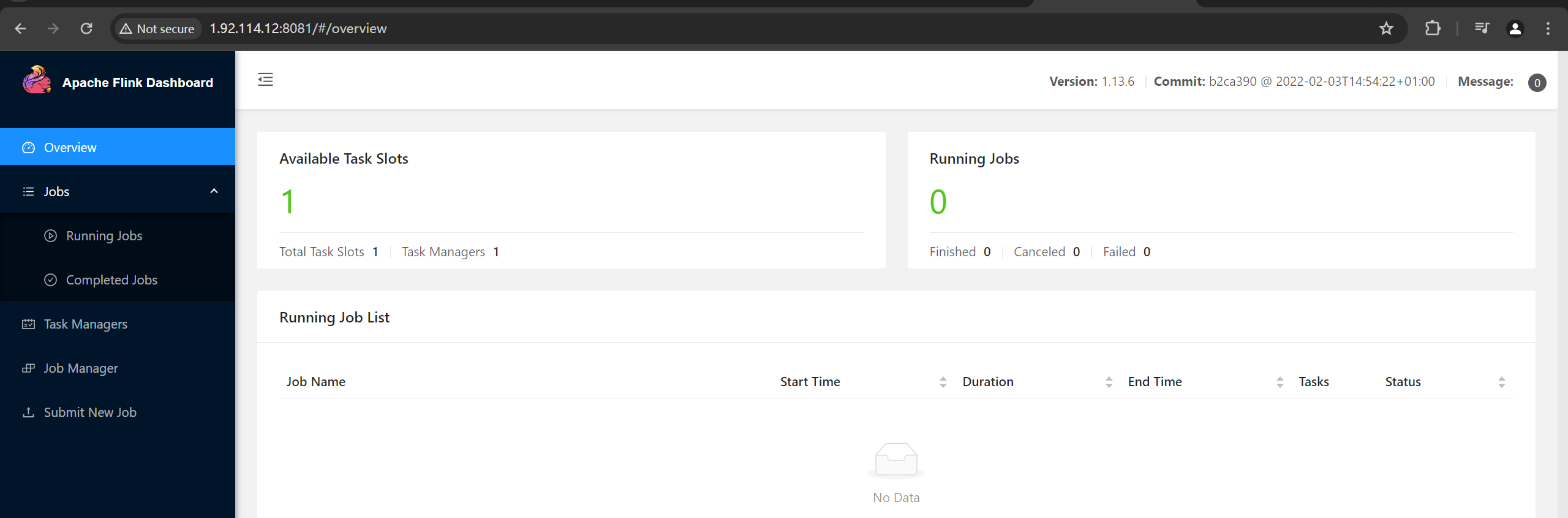


我的允许访问的ip地址网关设置为了/32，但如果是要让所有ip都能访问到这台服务器的8081端口的话，应该让IP地址和网关设置为0.0.0.0/0，才可以。

修改如下：



之后就能正常访问了

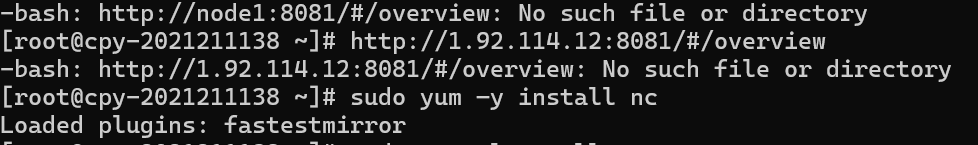


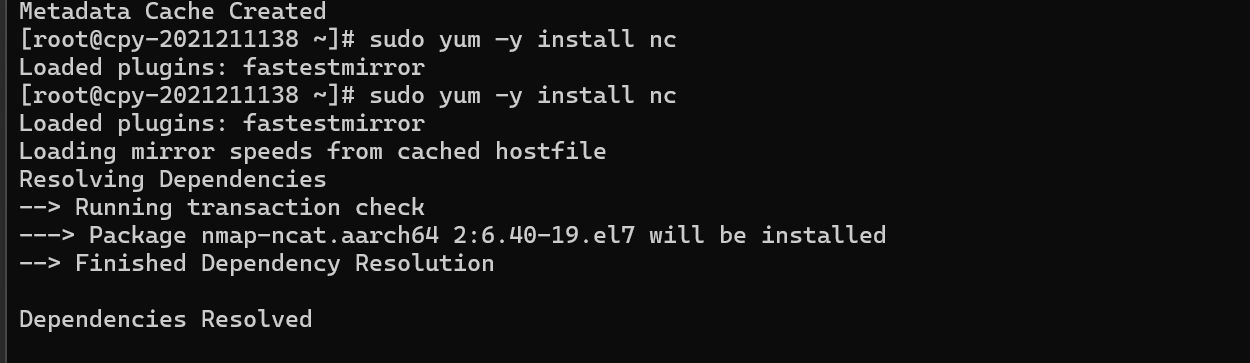
运行Flink自带的测试用例：

安装nc工具：

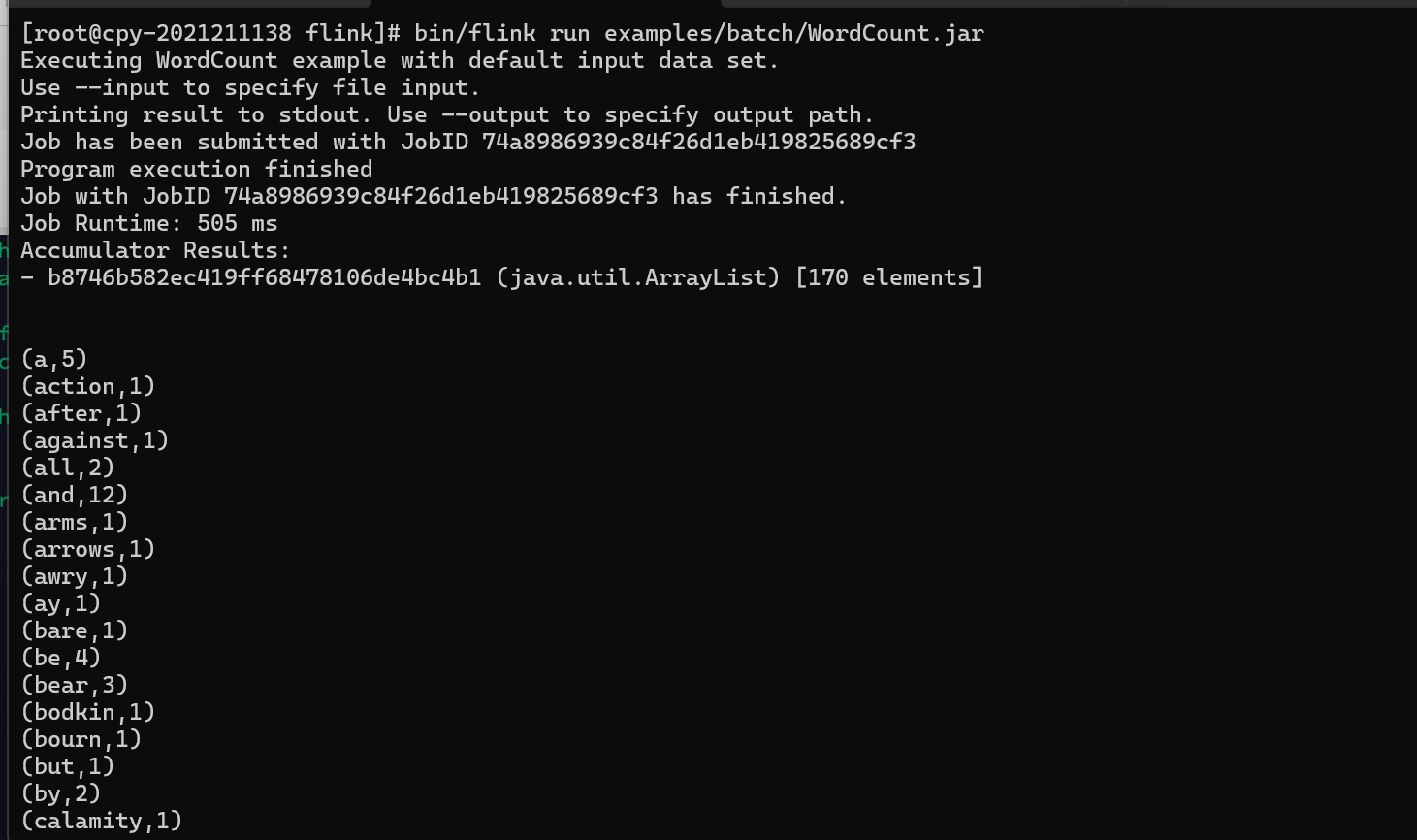
sudo yum –y install nc

遇到问题：

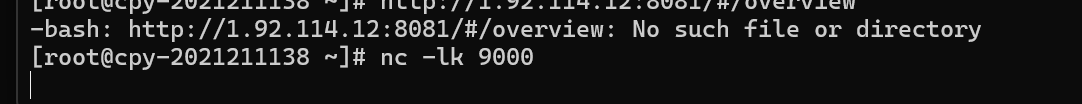




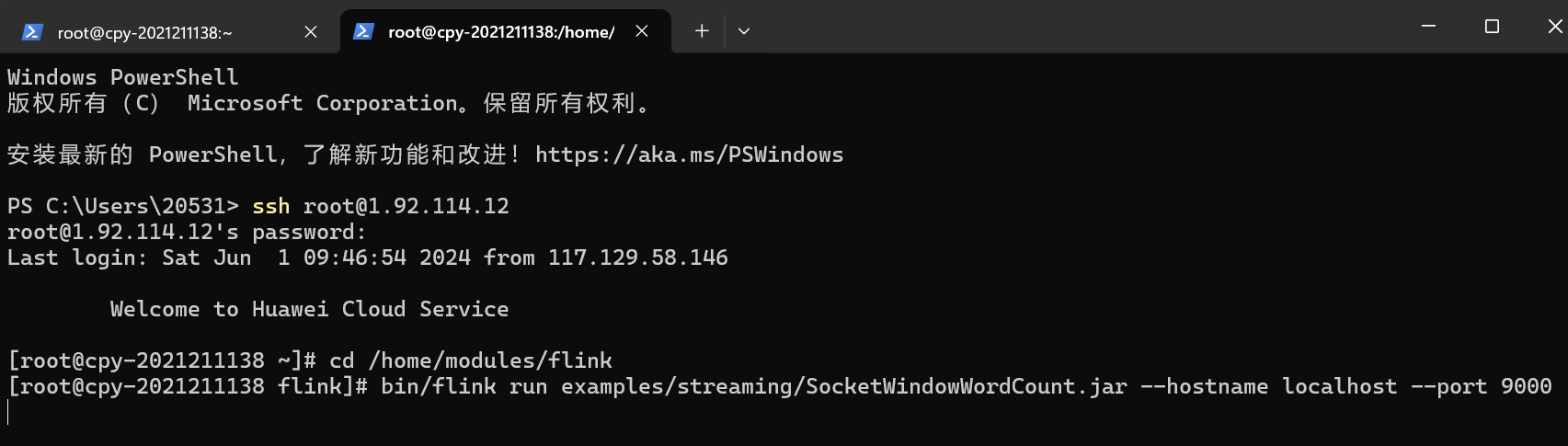
原因是，从word复制的命令 -y前面这个‘-’是破折号，而命令行里需要用到的是连字符，所以要自己敲命令。



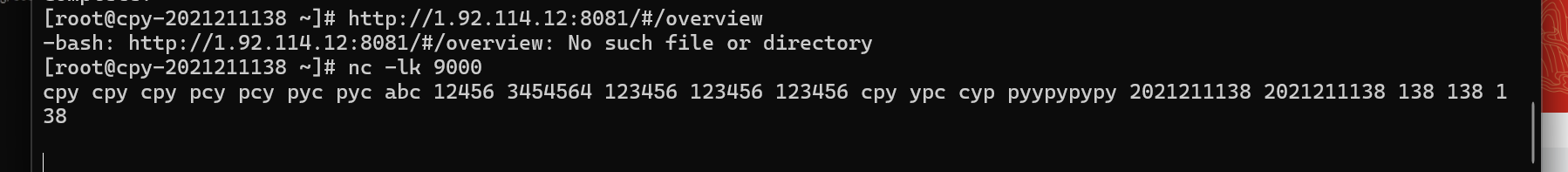
打开9000端口



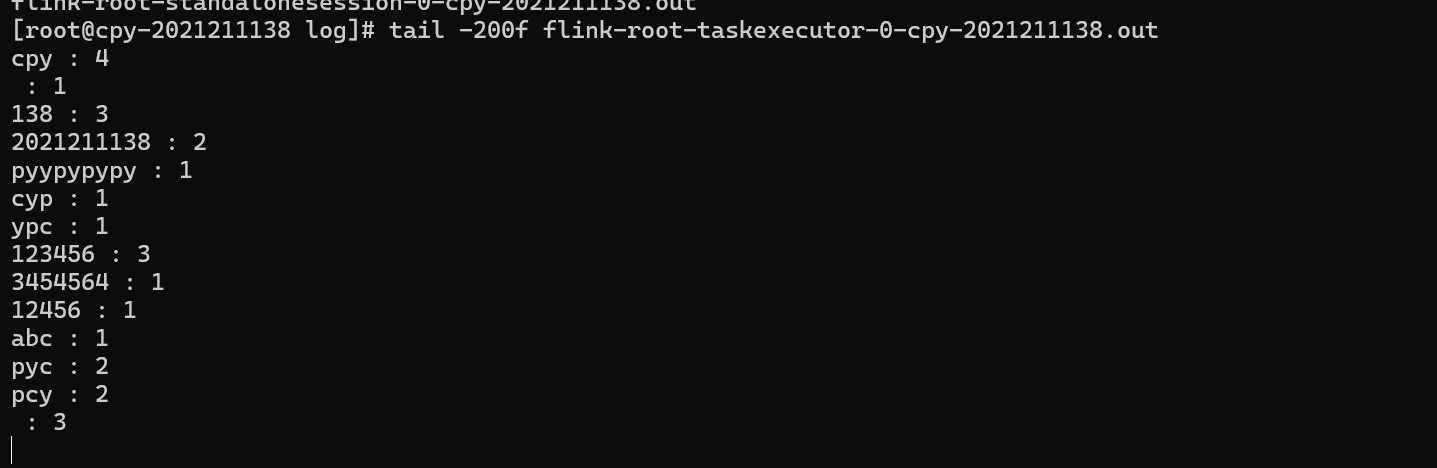
在另一个进程里，运行自带的程序

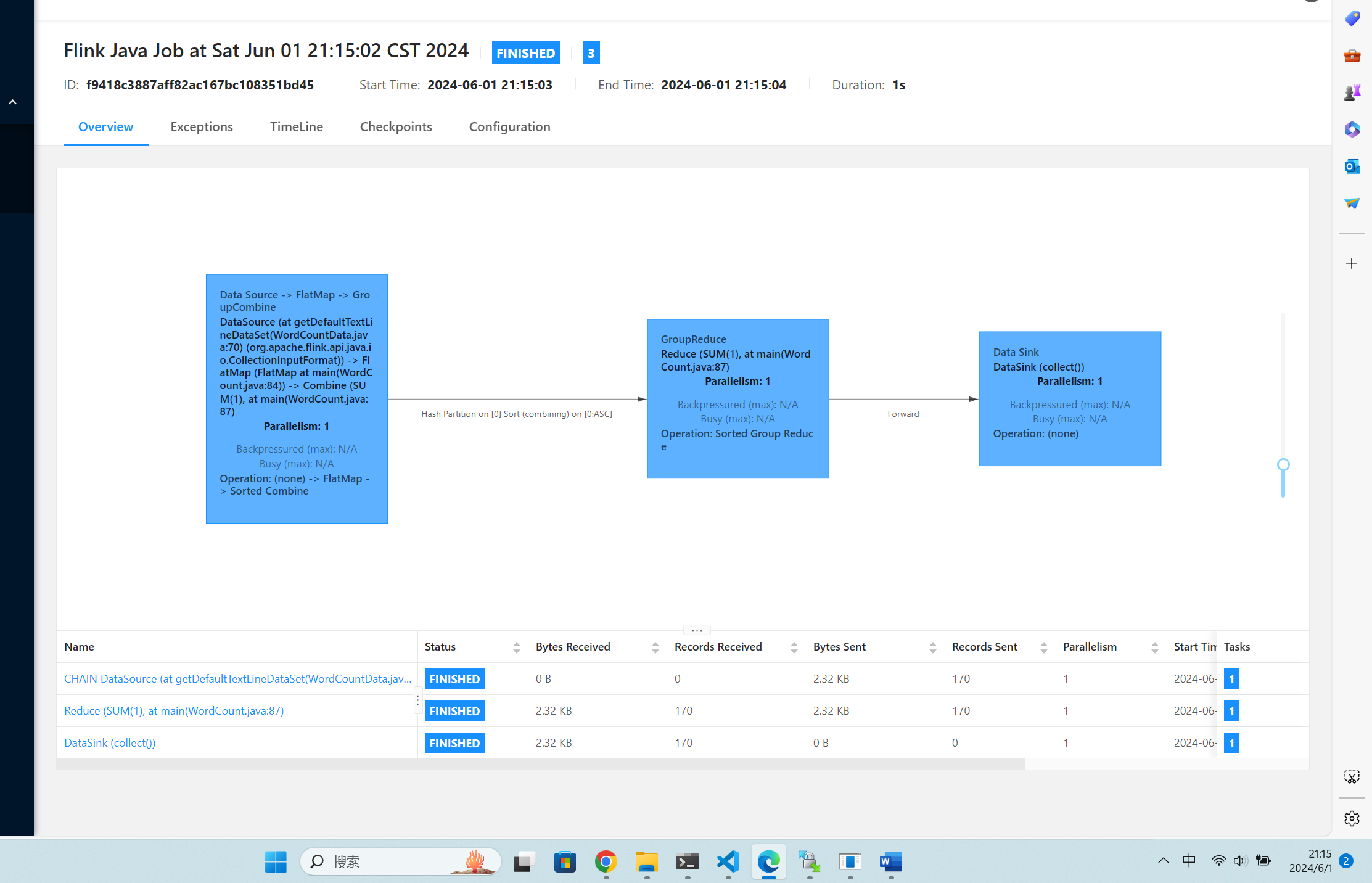


输入单次



再打开一个终端，查看结果：



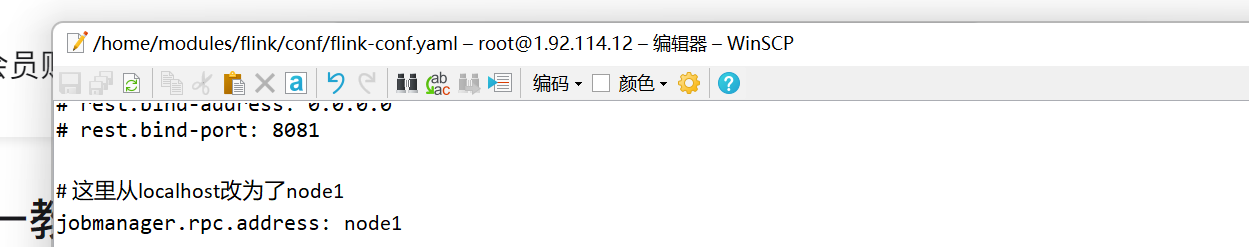


## 实验二

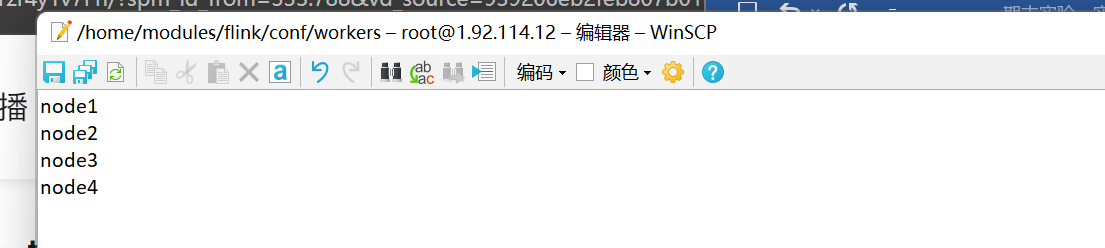
注释掉 host文件的127.0.0.1

只需要注释掉主服务器的host

修改flink-conf.yaml文件，如下：



修改workers文件：



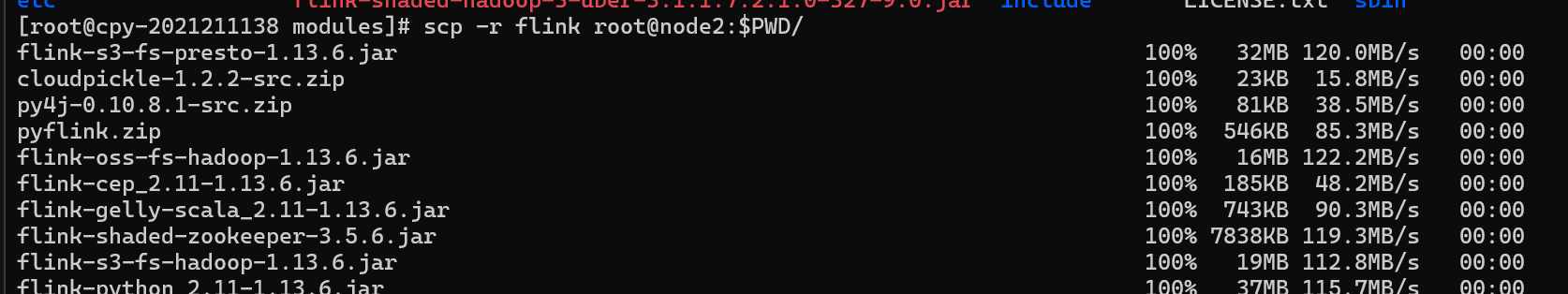
再将配置好的flink传给其他三个节点

cd /home/modules/

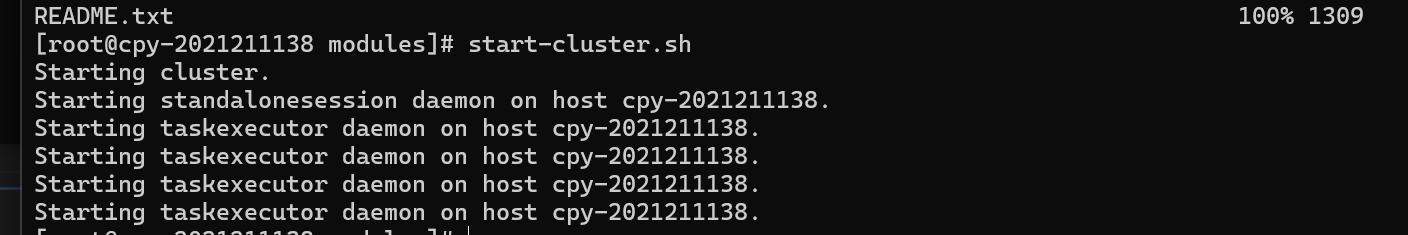
scp -r flink root@node2:$PWD/

scp -r flink root@node3:$PWD/

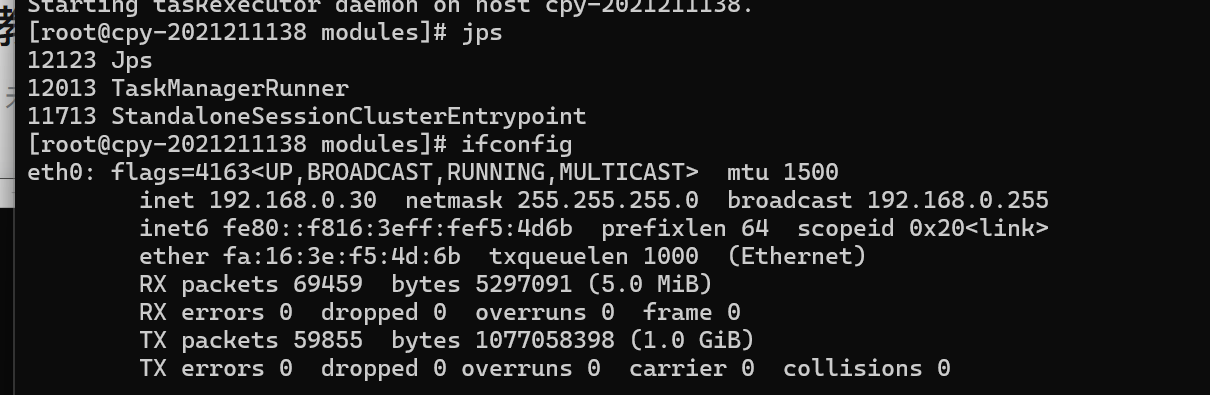
scp -r flink root@node4:$PWD/



启动：

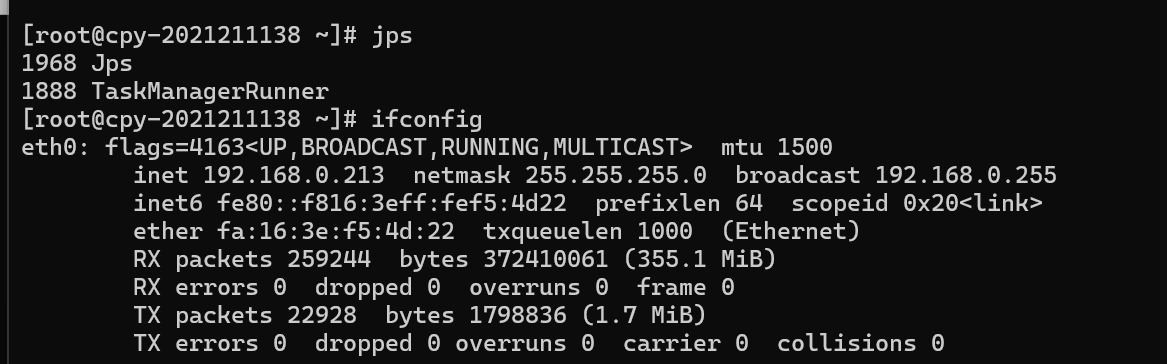


主节点的jps：

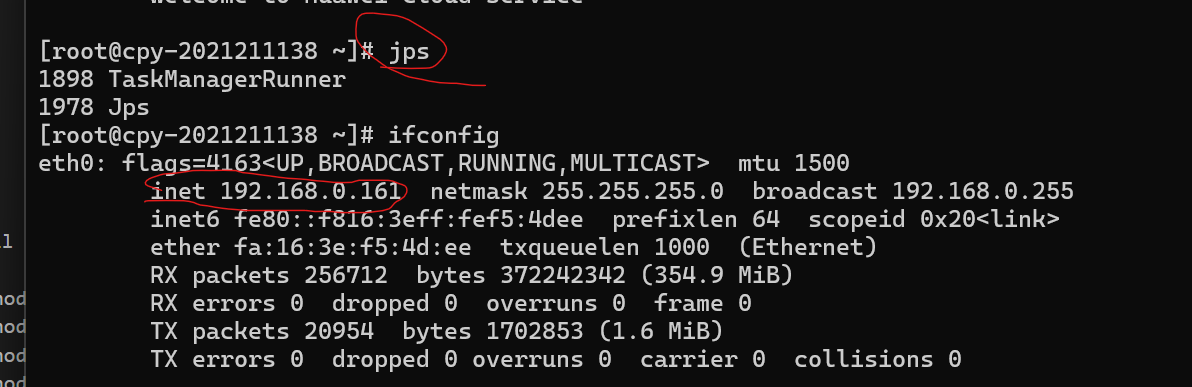


从节点的jps：

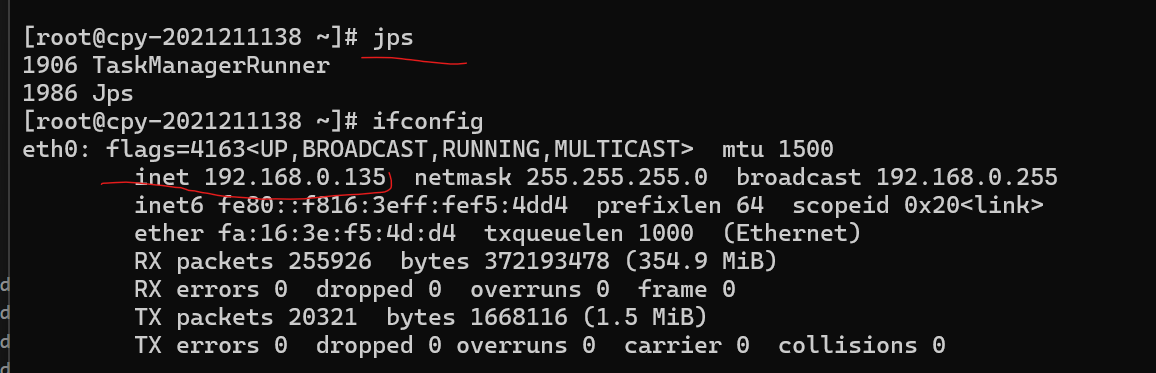
node2：



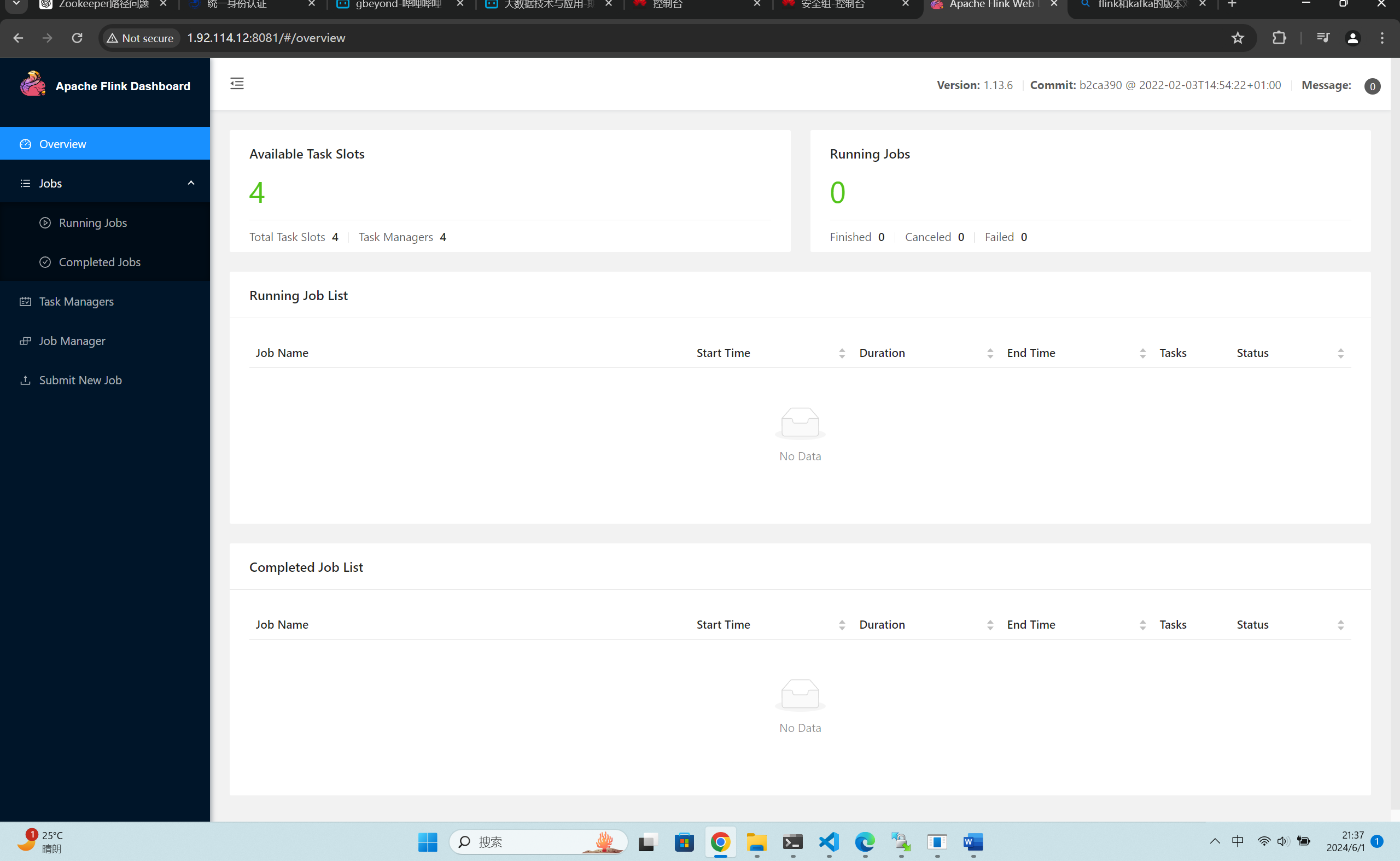
node3：



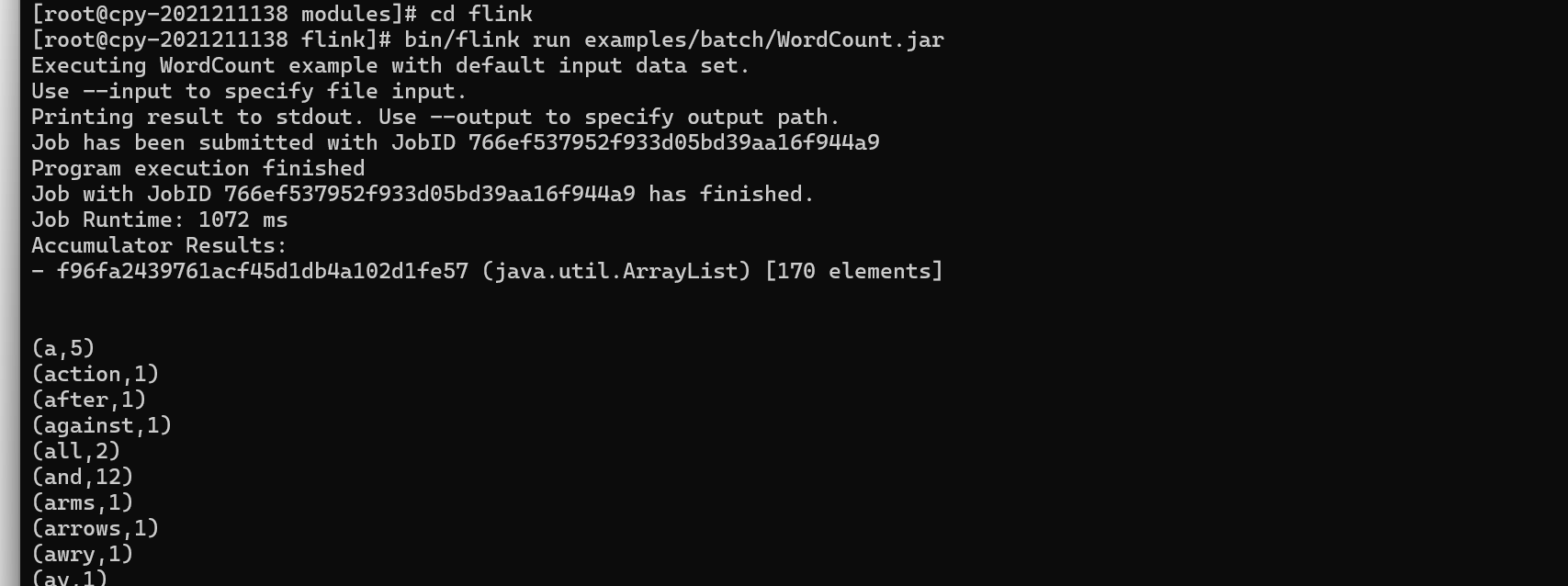
node4：



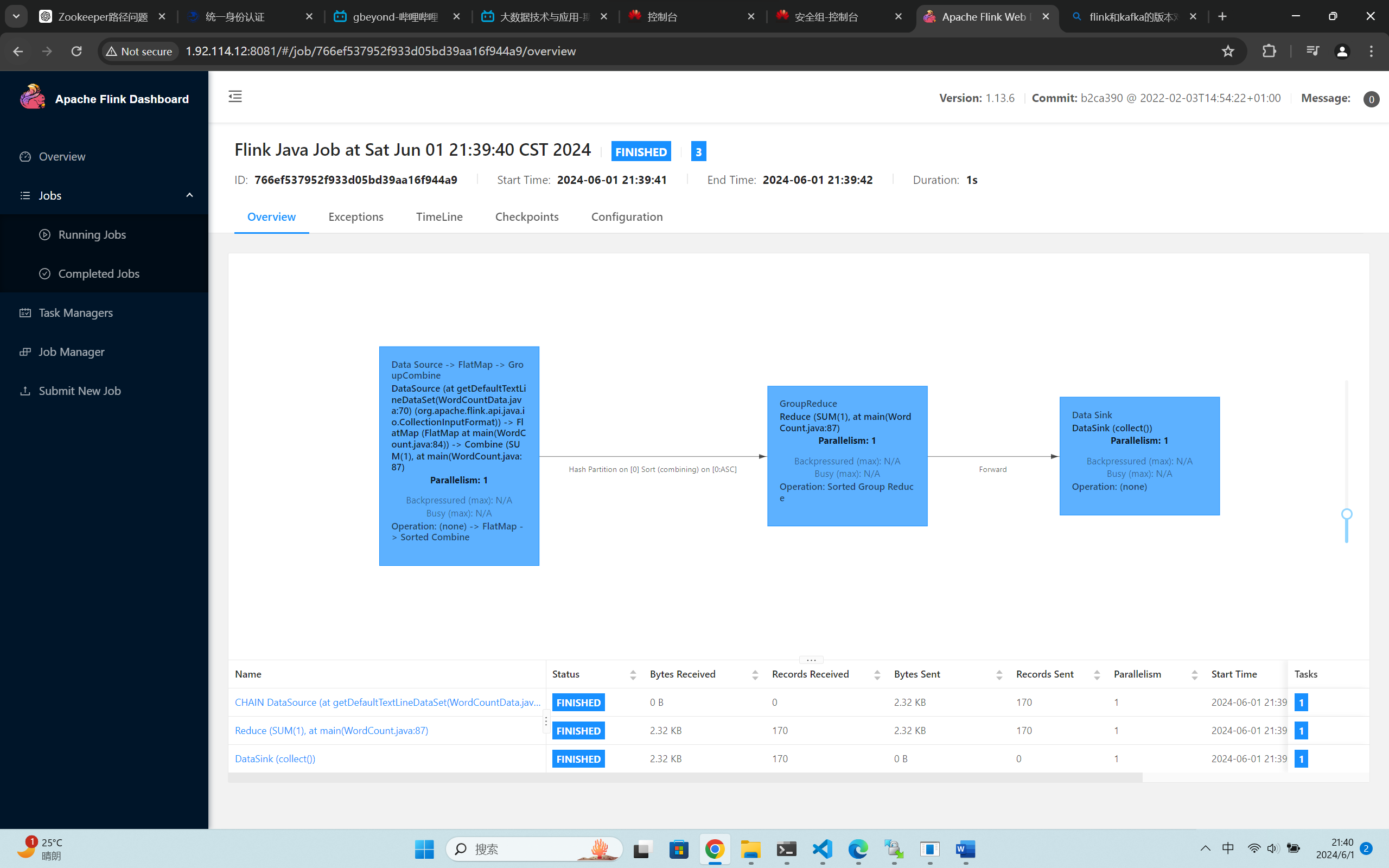
web管理页面：



运行测试用例：

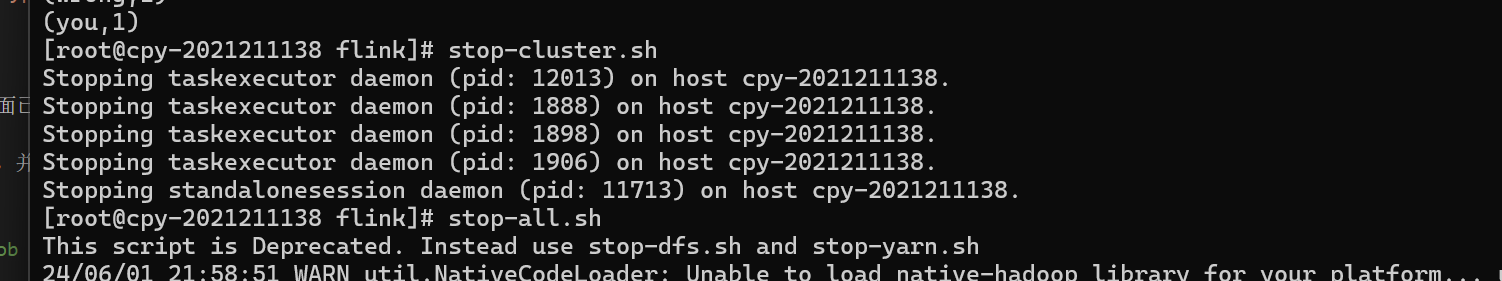


查看web管理页面的Completed job：



## 实验三、Flink on Yarn

首先停掉所有服务



修改hadoop路径下的yarn-site.xml

<property>

  <name>yarn.resourcemanager.am.max-attempts</name>

  <value>4</value>

  <description>The maximum number of application master execution attempts.</description>

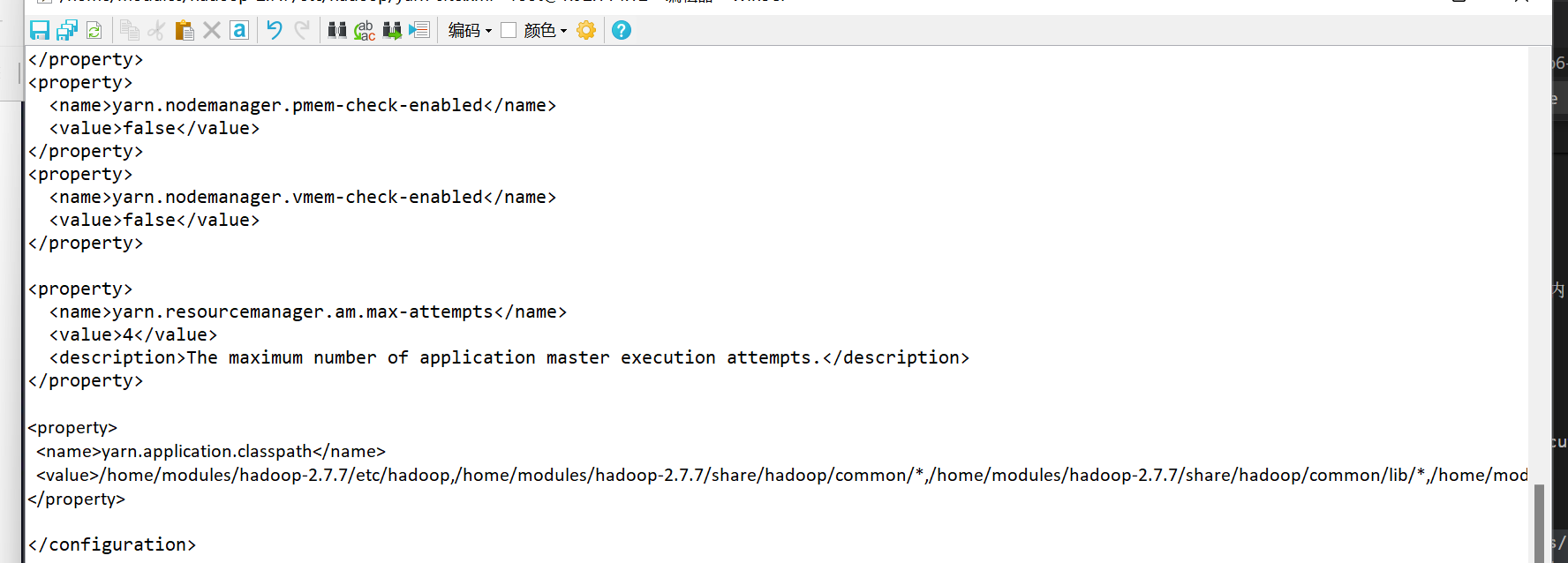
</property>

<property>

  <name>yarn.application.classpath</name>

  <value>/home/modules/hadoop-2.7.7/etc/hadoop,/home/modules/hadoop-2.7.7/share/hadoop/common/\*,/home/modules/hadoop-2.7.7/share/hadoop/common/lib/\*,/home/modules/hadoop-2.7.7/hdfs/\*,/home/modules/hadoop-2.7.7/share/hadoop/hdfs/lib/\*,/home/modules/hadoop-2.7.7/share/hadoop/yarn/\*,/home/modules/hadoop-2.7.7/share/hadoop/yarn/lib/\*,/home/modules/hadoop-2.7.7/mapreduce/\*,/home/modules/hadoop-2.7.7/share/hadoop/mapreduce/lib/\*</value>

</property>



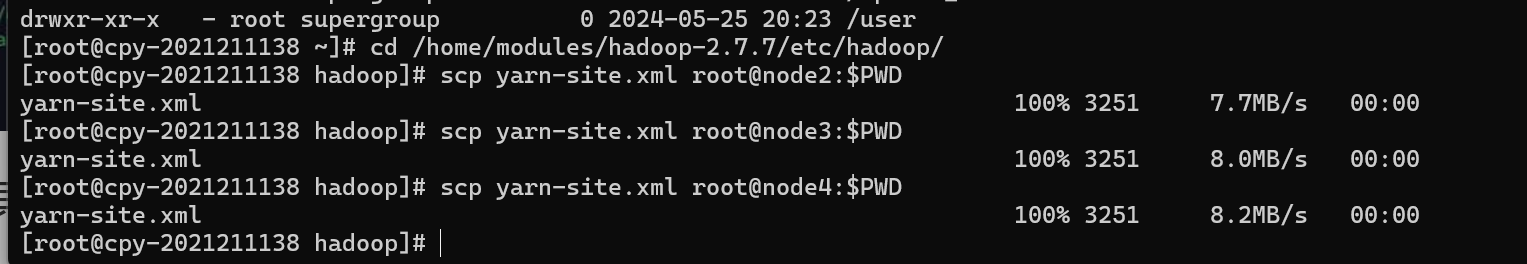
将修改好后的yarn-site.xml送给其他节点：

cd /home/modules/hadoop-2.7.7/etc/hadoop/

scp yarn-site.xml root@node2:$PWD

scp yarn-site.xml root@node3:$PWD

scp yarn-site.xml root@node4:$PWD

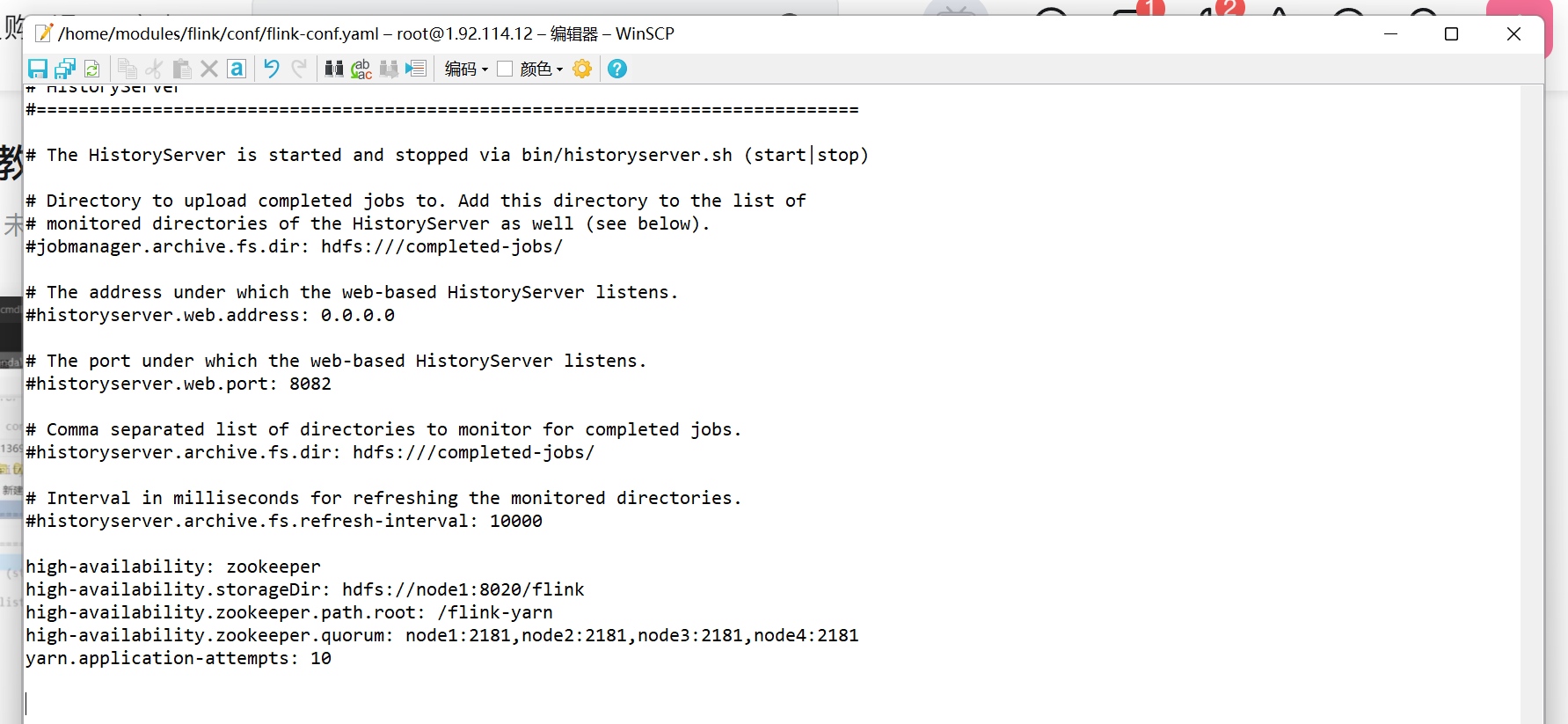


再打开hadoop 和zookeeper服务



修改node1的flink-conf.yaml文件

添加四行：



# 分发配置文件

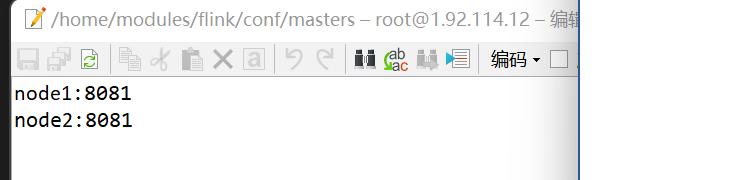
cd /home/modules/flink/conf/

scp -r flink-conf.yaml masters root@node2:$PWD

scp -r flink-conf.yaml masters root@node3:$PWD

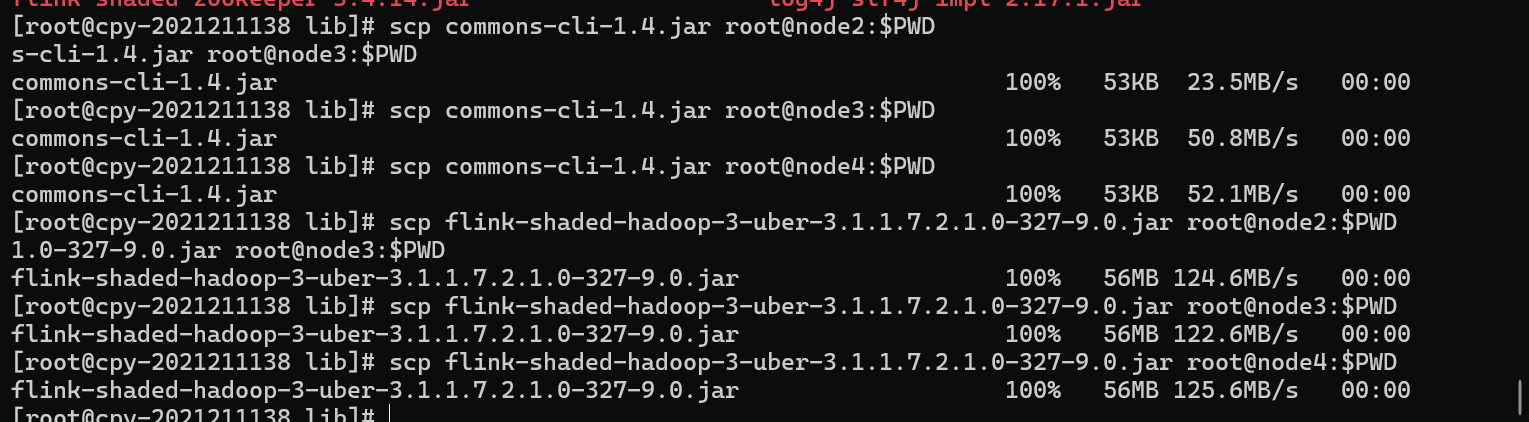
scp -r flink-conf.yaml masters root@node4:$PWD

修改masters文件：

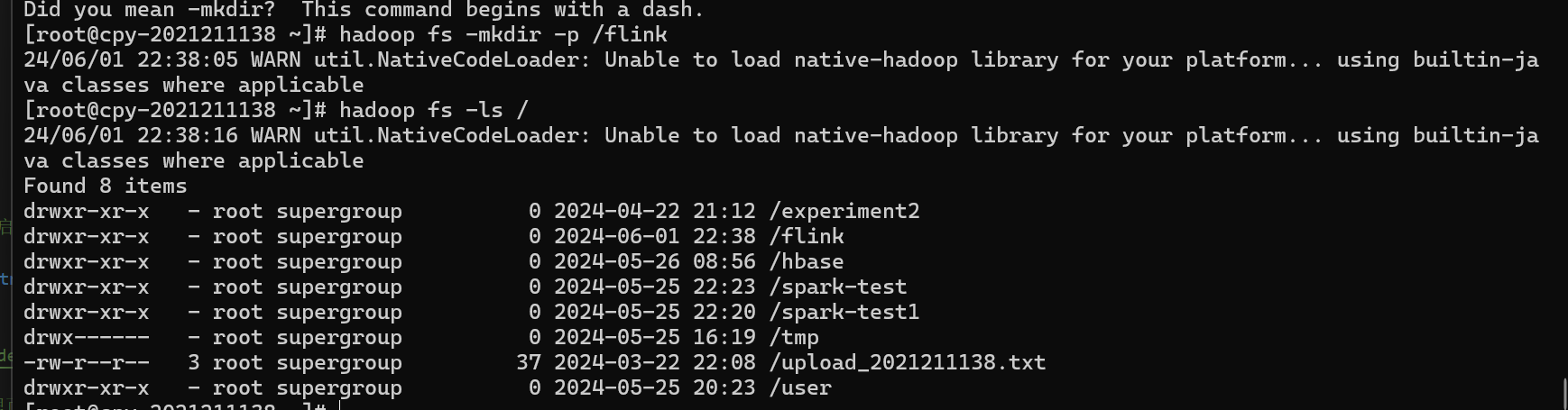


其他四个节点也是相同修改

将下面两个jar包放入lib中，并分发给其他四个节点：



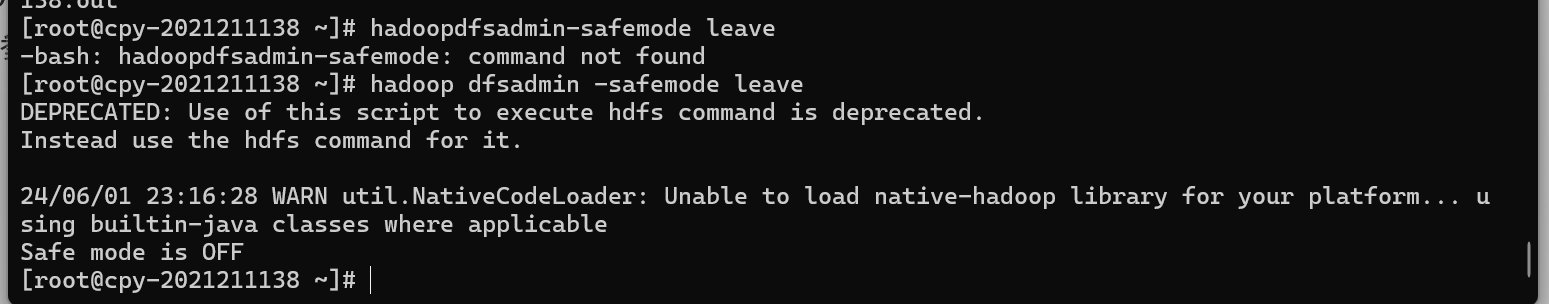
在hdfs文件系统中添加路径：



各个节点都启动zookeeper：

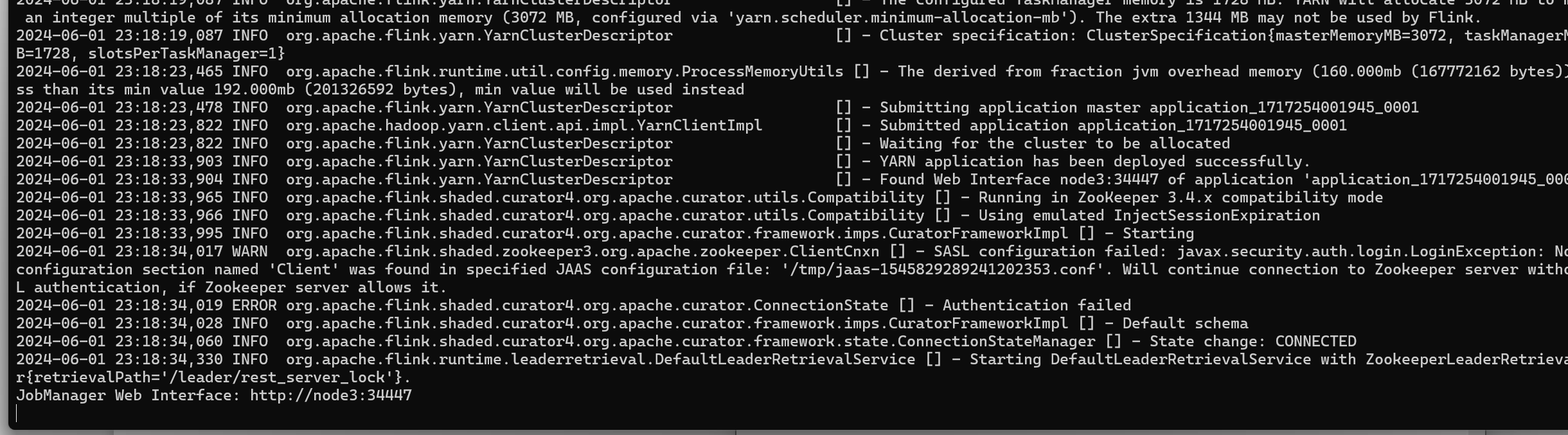
解除hadoop的安全模式：

hadoop dfsadmin -safemode leave



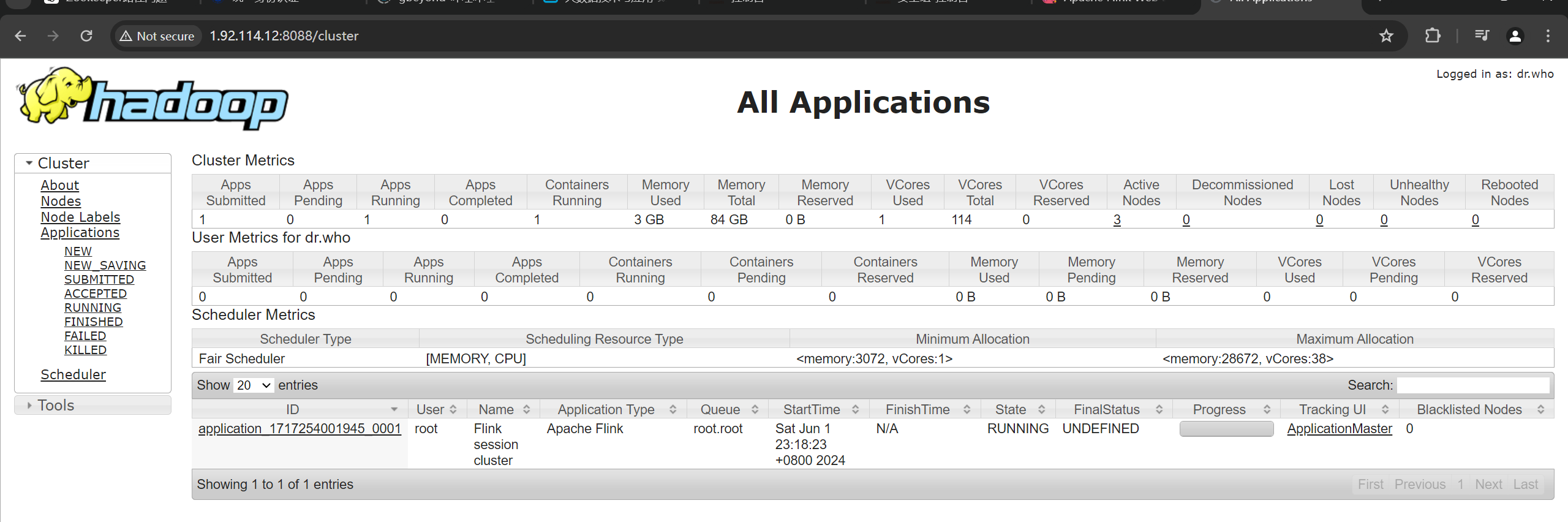
输入以下命令：

yarn-session.sh -n 2 -jm 1024 -tm 1024 -d



之后卡在上面这个地方,并且不要关闭。

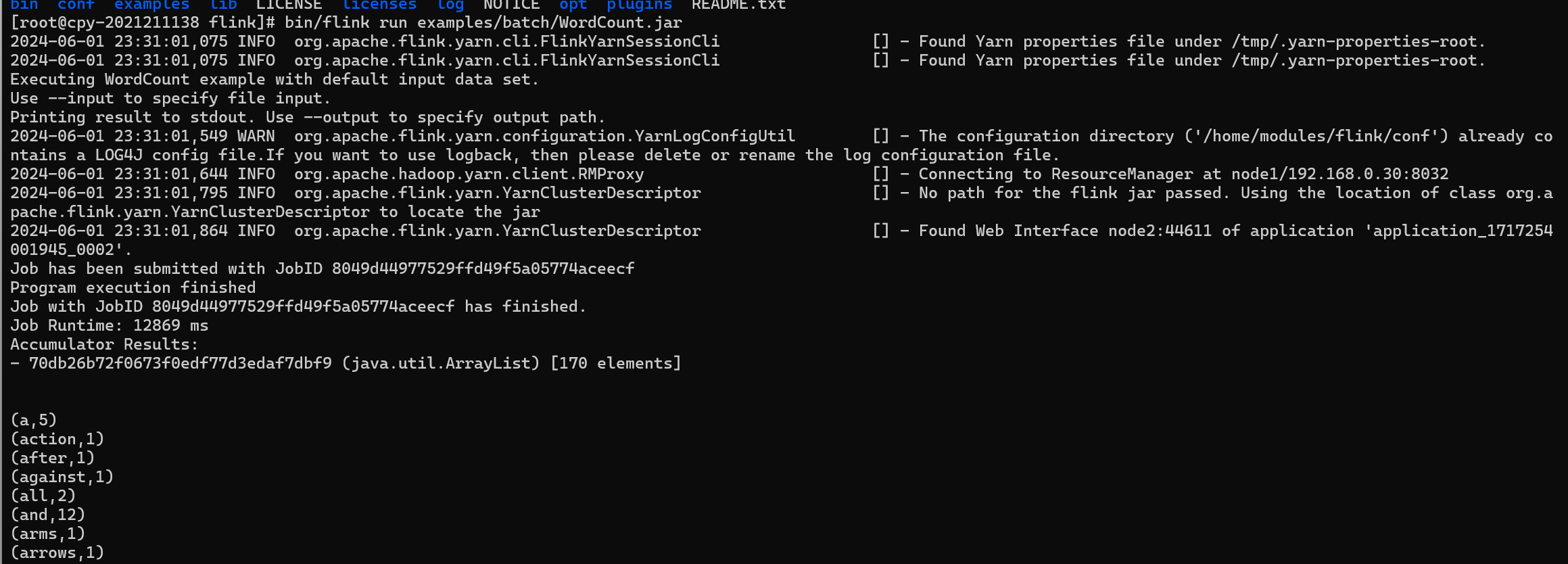
接下来访问yarn的8088管理界面http://1.92.114.12:8088/cluster



再开一个终端连接主节点，进入flink安装目录，输入以下命令

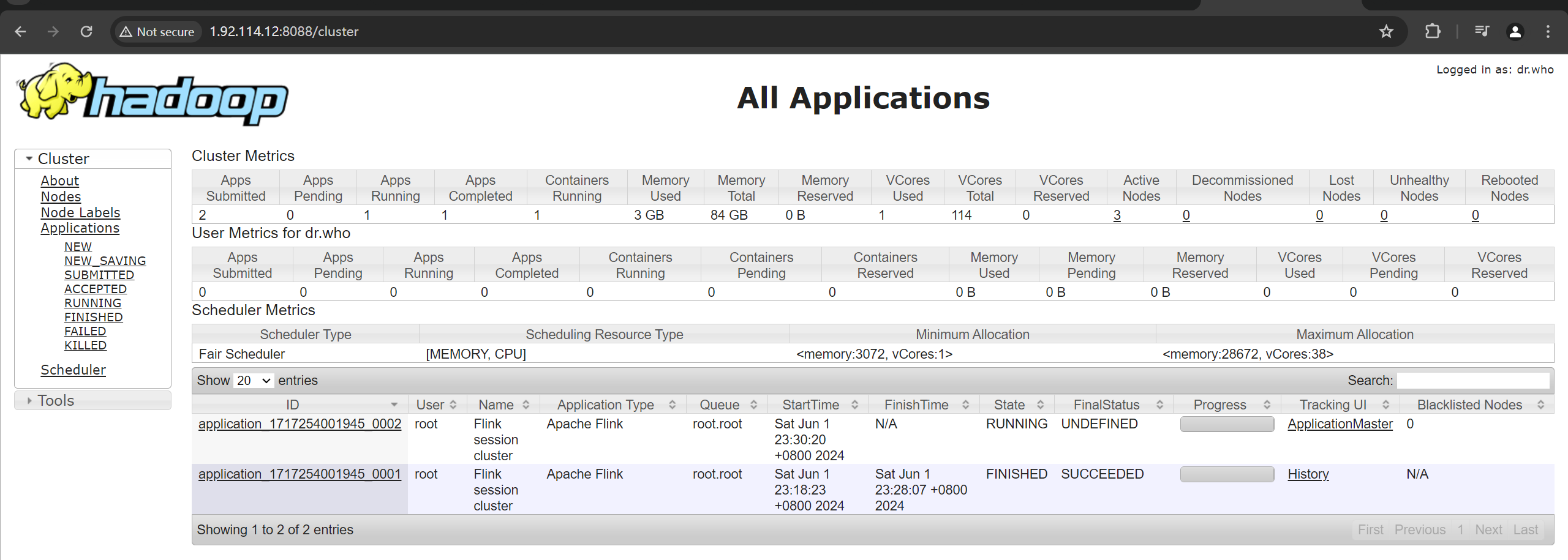
bin/flink run examples/batch/WordCount.jar

结果：

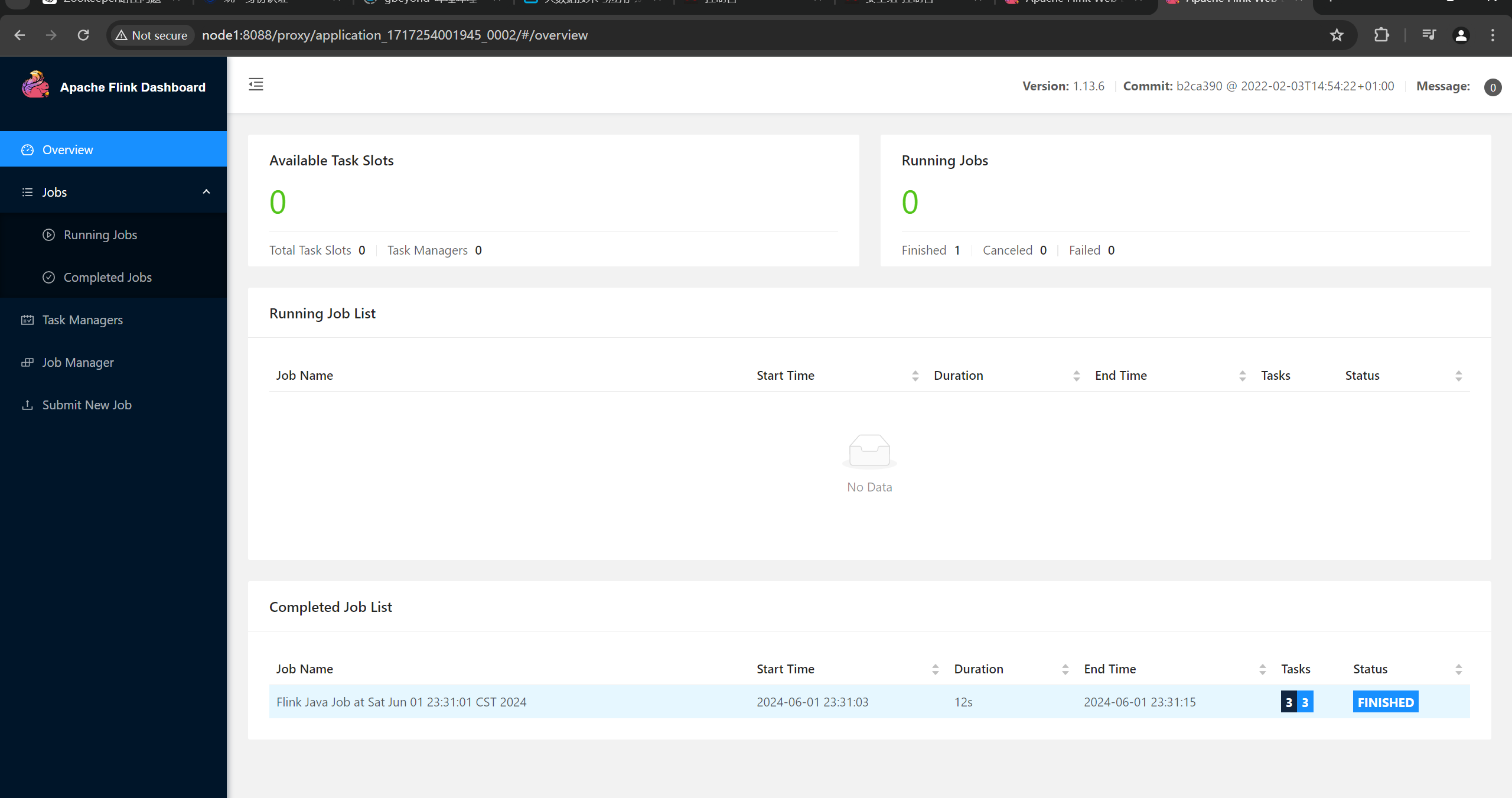




在yarn中有对应的任务：



点击ApplicationMaster，可以看到Job Name



## 实验四、

下载kafka\_1.12\_1.0.2.tgz到/home/modules目录下，并解压缩

换名成kafka

然后将kafka分发给其他三个从节点

scp -r kafka root@node2:$PWD

scp -r kafka root@node3:$PWD

scp -r kafka root@node4:$PWD

更改四个节点的/etc/profile环境路径：

# Kafka

export KAFKA\_HOME=/home/modules/kafka

export PATH=$KAFKA\_HOME/bin:$PATH

再重新刷新一下

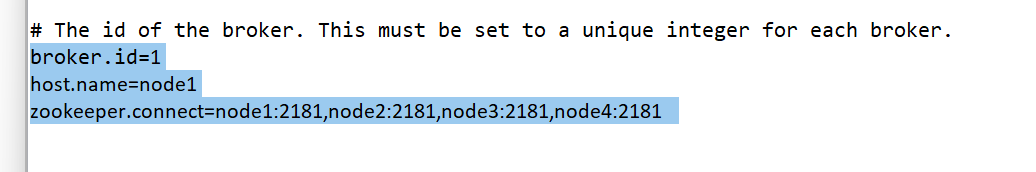
source /etc/profile

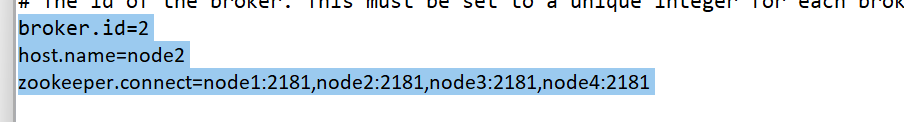
进入kafka安装包的config目录，在server.properties添加内容：

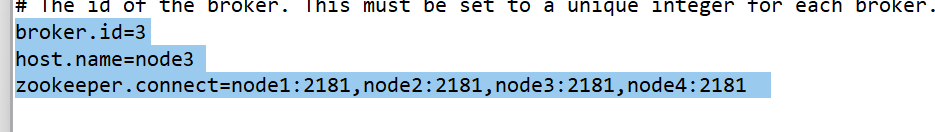
broker.id=1

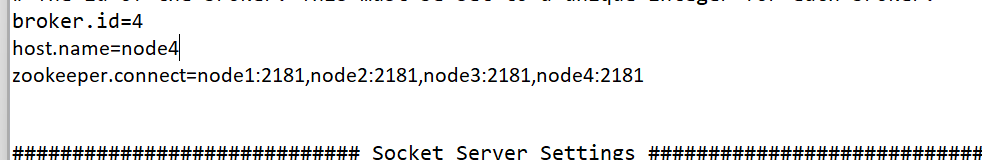
host.name=node1

zookeeper.connect=node1:2181,node2:2181,node3:2181,node4:2181









配置完成，验证是否安装成功，前提要先启动zookeeper

然后在各个节点分别启动kafka

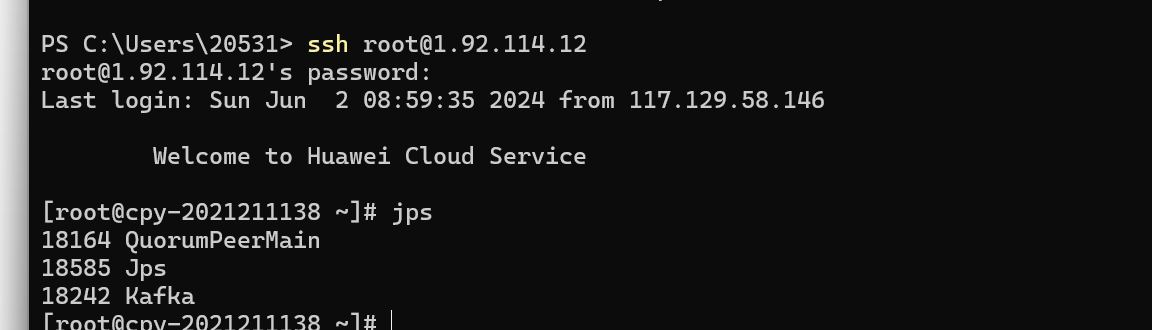
先进入kafka目录

然后输入以下命令：

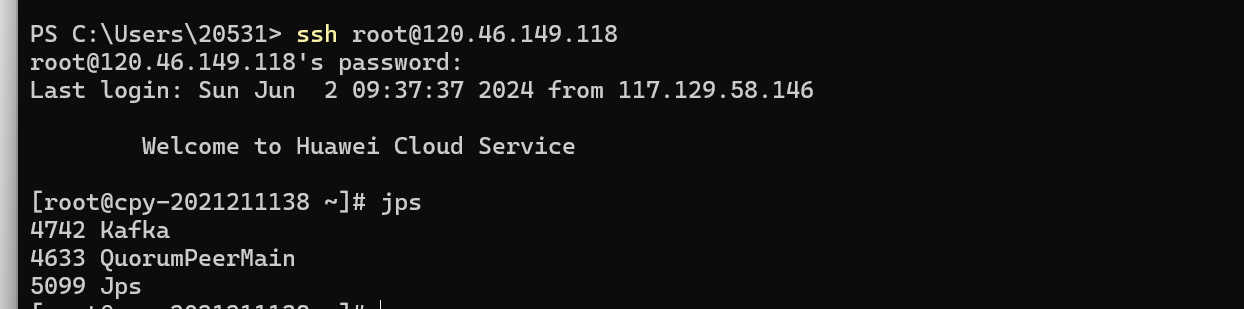
./bin/kafka-server-start.sh ./config/server.properties

再启动一个终端，查看jps

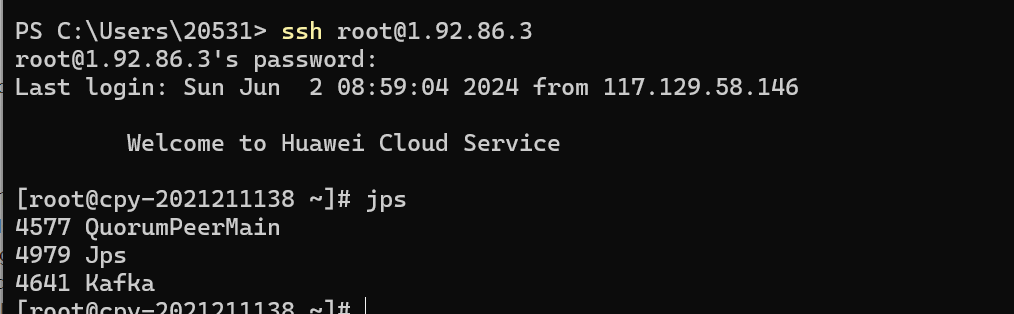
这是主节点的jps：



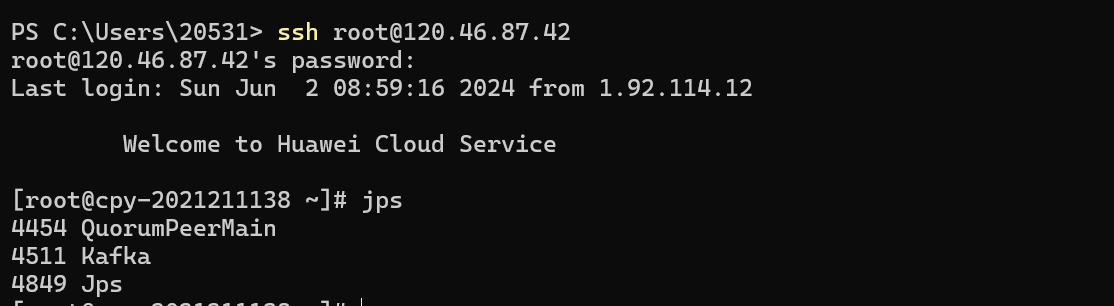
node2的jps：



node3的jps：



node4的jps：



检查完毕后，关闭各个节点的kafka进程，

编写代码：pom.xml文件：

<?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>org.example</groupId>  
 <artifactId>MyWordCount</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <properties>  
 <maven.compiler.source>8</maven.compiler.source>  
 <maven.compiler.target>8</maven.compiler.target>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <hadoop.version>2.7.7</hadoop.version>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>org.javatuples</groupId>  
 <artifactId>javatuples</artifactId>  
 <version>1.2</version>  
 </dependency>  
  
 <dependency>  
 <groupId>log4j</groupId>  
 <artifactId>log4j</artifactId>  
 <version>1.2.17</version>  
 <scope>provided</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.slf4j</groupId>  
 <artifactId>slf4j-api</artifactId>  
 <version>1.7.30</version>  
 <scope>runtime</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.flink</groupId>  
 <artifactId>flink-java</artifactId>  
 <version>1.13.6</version>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.kafka</groupId>  
 <artifactId>kafka\_2.12</artifactId>  
 <version>1.0.2</version>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.flink</groupId>  
 <artifactId>flink-clients\_2.11</artifactId>  
 <version>1.13.6</version>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.kafka</groupId>  
 <artifactId>kafka-clients</artifactId>  
 <version>1.0.2</version>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.flink</groupId>  
 <artifactId>flink-streaming-scala\_2.12</artifactId>  
 <version>1.13.6</version>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.flink</groupId>  
 <artifactId>flink-connector-kafka\_2.11</artifactId>  
 <version>1.13.6</version>  
 <scope>provided</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.flink</groupId>  
 <artifactId>flink-connector-base</artifactId>  
 <version>1.13.6</version>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.hadoop</groupId>  
 <artifactId>hadoop-client</artifactId>  
 <version>${hadoop.version}</version>  
 <scope>provided</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.hadoop</groupId>  
 <artifactId>hadoop-common</artifactId>  
 <version>${hadoop.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.hadoop</groupId>  
 <artifactId>hadoop-hdfs</artifactId>  
 <version>${hadoop.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>RELEASE</version>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
</project>

Flink\_Kafka.java文件：

import org.apache.flink.api.common.functions.FlatMapFunction;  
import org.apache.flink.api.common.serialization.SimpleStringSchema;  
import org.apache.flink.streaming.api.datastream.DataStream;  
import org.apache.flink.streaming.api.environment.StreamExecutionEnvironment;  
import org.apache.flink.streaming.connectors.kafka.FlinkKafkaConsumer;  
import org.apache.flink.util.Collector;  
import org.apache.flink.api.java.tuple.Tuple2;  
  
import java.util.Properties;  
  
*/\*\*  
 \* Flink\_Kafka - A simple Flink application that reads messages from a Kafka topic,  
 \* splits each message into words, and counts the occurrences of each word.  
 \*  
 \** ***@author*** *陈朴炎  
 \** ***@version*** *1.0  
 \*/*public class Flink\_Kafka {  
 public static void main(String[] args) throws Exception {  
  
 // 获取Flink运行环境  
 StreamExecutionEnvironment env = StreamExecutionEnvironment.*getExecutionEnvironment*();  
  
 // 配置Kafka连接属性  
 Properties properties = new Properties();  
 properties.setProperty("bootstrap.servers", "node1:9092"); // Kafka服务器地址  
 properties.setProperty("zookeeper.connect", "node1:2181"); // Zookeeper地址（可选，通常用于旧版Kafka）  
 properties.setProperty("group.id", "1"); // 消费者组ID  
  
 // 创建FlinkKafkaConsumer来读取Kafka中的数据  
 FlinkKafkaConsumer<String> myConsumer = new FlinkKafkaConsumer<>("test", new SimpleStringSchema(), properties);  
 myConsumer.setStartFromGroupOffsets(); // 从消费者组的偏移量开始消费  
 DataStream<String> dataStream = env.addSource(myConsumer); // 将Kafka消费者添加为数据源  
  
 // 处理数据流：将每行文本拆分成单词，并计算每个单词的出现次数  
 DataStream<Tuple2<String, Integer>> result = dataStream  
 .flatMap(new MyFlatMapper()) // 将每行文本拆分成单词  
 .keyBy(0) // 按照单词分组  
 .sum(1); // 计算每个单词的出现次数  
  
 // 打印结果到控制台，设置并行度为1  
 result.print().setParallelism(1);  
  
 // 执行Flink作业  
 env.execute();  
 }  
  
 */\*\*  
 \* MyFlatMapper - A FlatMapFunction that splits lines of text into words.  
 \* 每行文本拆分成单词，并将每个单词作为Tuple2<单词, 1>输出  
 \*/* public static class MyFlatMapper implements FlatMapFunction<String, Tuple2<String, Integer>> {  
 @Override  
 public void flatMap(String s, Collector<Tuple2<String, Integer>> collector) throws Exception {  
 // 将行文本拆分成单词  
 String[] words = s.split(" ");  
 // 输出每个单词，附带初始计数1  
 for (String word : words) {  
 collector.collect(new Tuple2<>(word, 1));  
 }  
 }  
 }  
}

打包、上传jar包

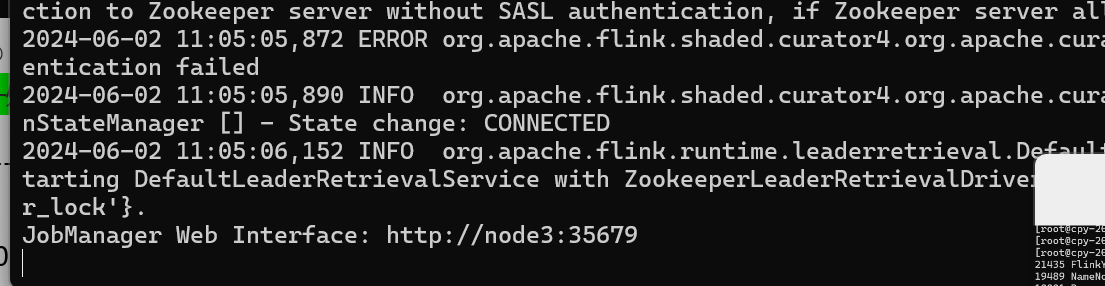
停掉zookeeper

开启flink on yarn 集群

打开hadoop、关闭hadoop的安全模式：

cd /home/modules/flink，输入：

bin/yarn-session.sh -n 2 -jm 1024 -tm 1024 -d



看到卡住，说明成功

再开一个终端，进入kafka的config目录下启动kafka：



停止node1的zookeeper，然后启动kafka自带的zookeeper

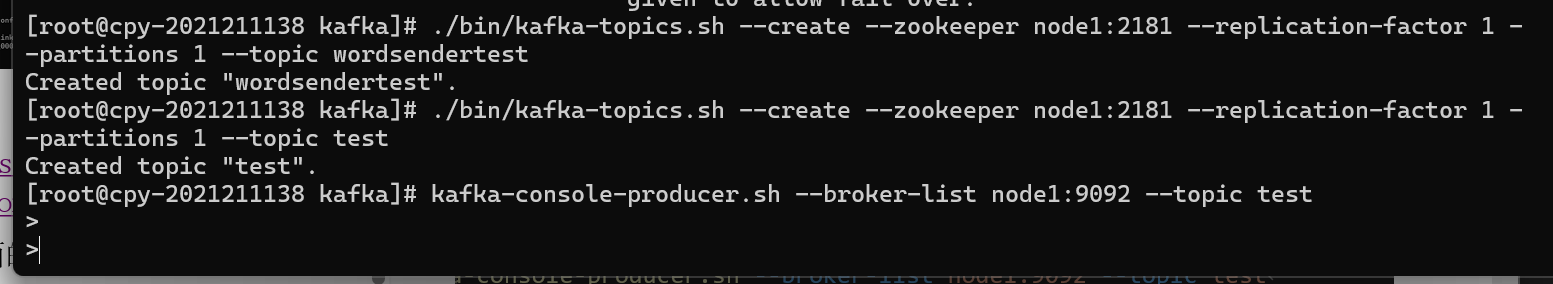
./bin/zookeeper-server-start.sh config/zookeeper.properties

进入到kafka的安装目录，输入下列指令：

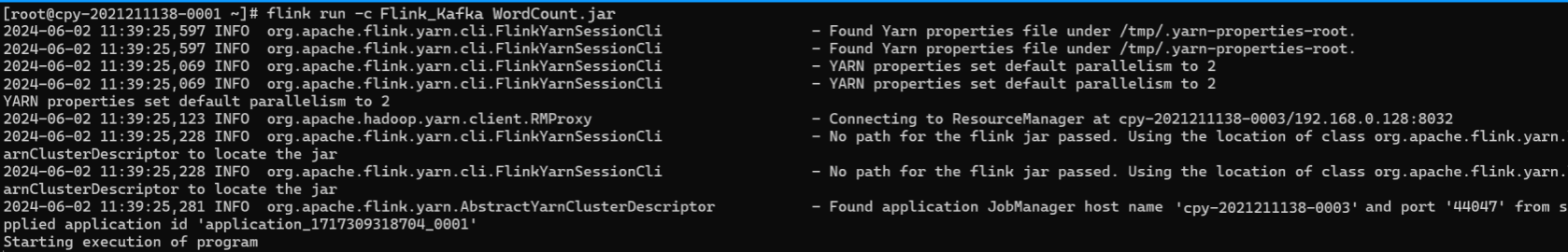
./bin/kafka-topics.sh --create --zookeeper node1:2181 --replication-factor 1 --partitions 1 --topic test

kafka-console-producer.sh --broker-list node1:9092 --topic test

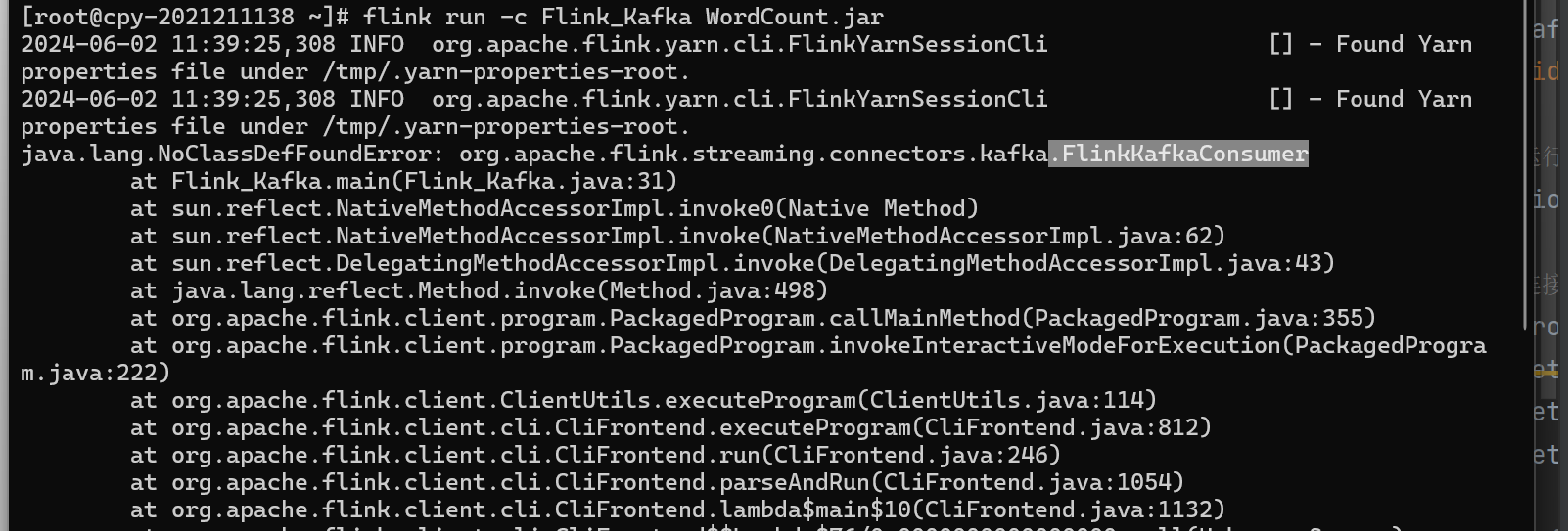
由于我输入的是 --zookeeper 选项，所以没有指导书中的那些提示信息



再启动一个终端，输入 flink run -c Flink\_Kafka WordCount.jar

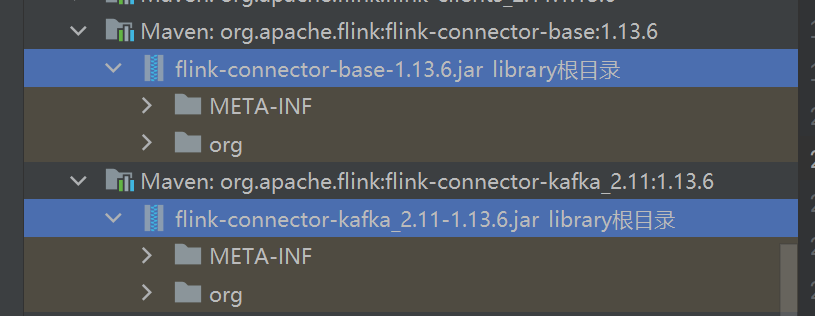


出现报错：

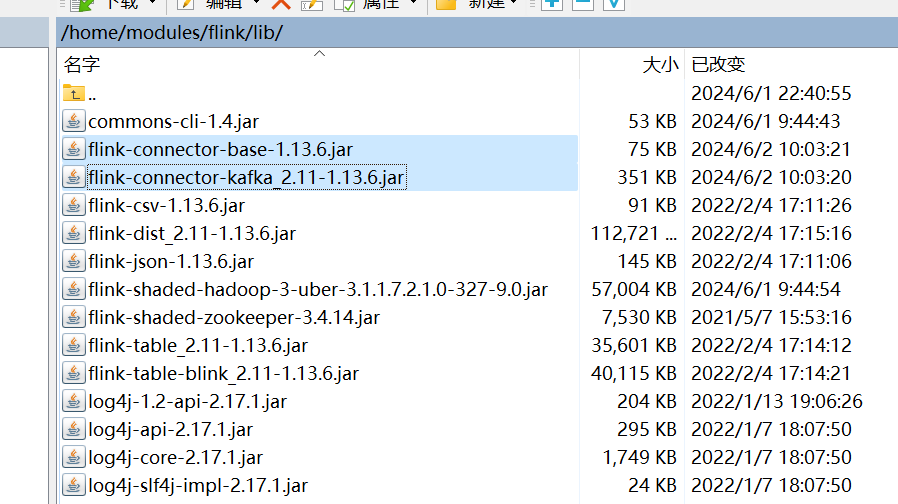


解决：

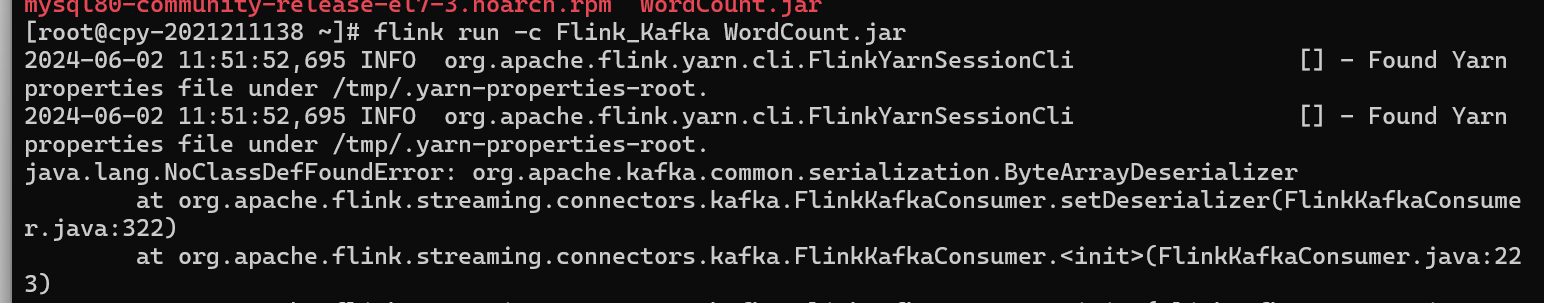
在maven工程中的外部库中找到FlinkKafkaConsumer类和FlinkKafkaConsumerBase类，并将这两个jar包上传到服务器的flink下的lib中



上传：



重新运行



出现上述错误，

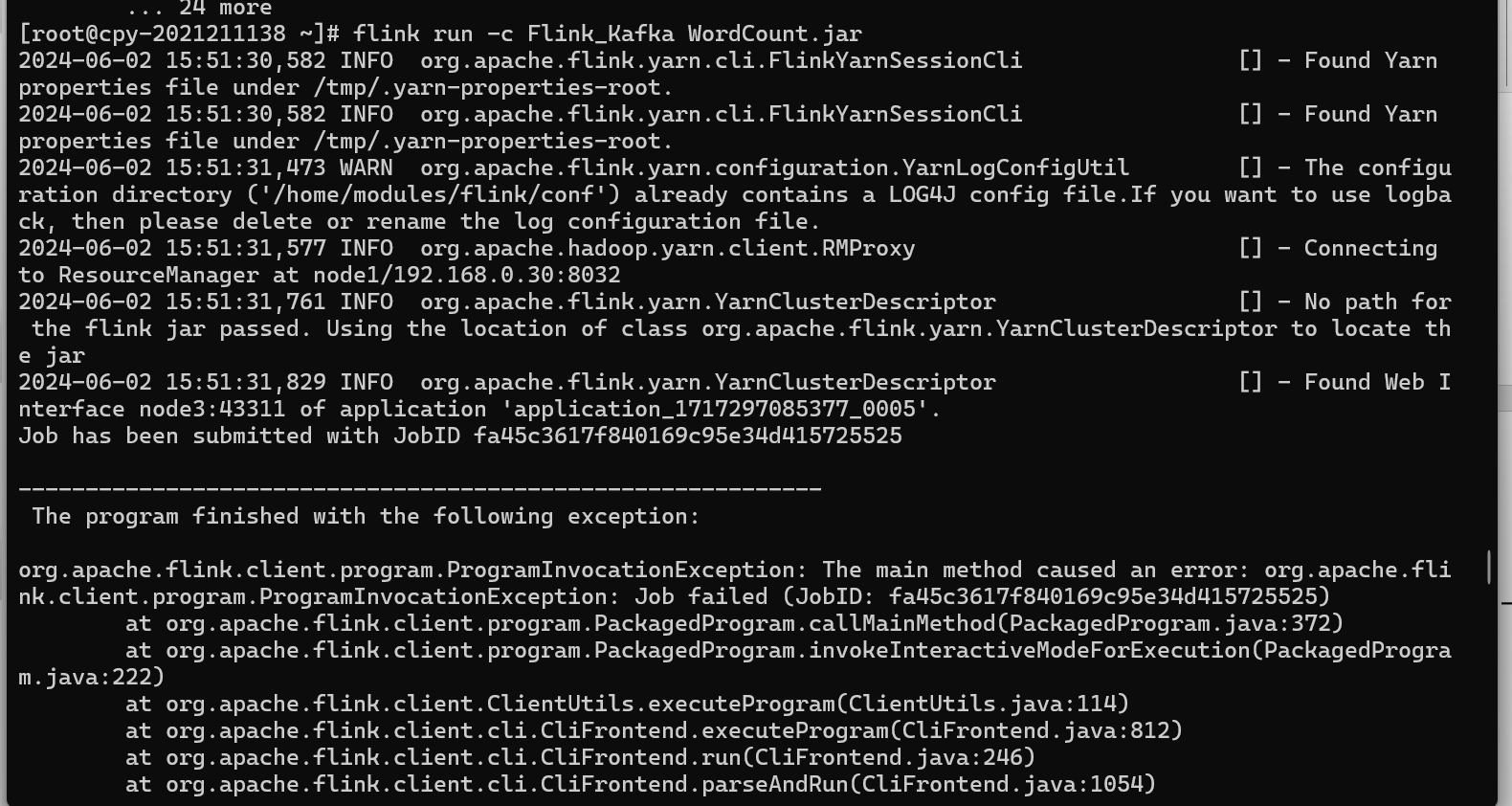
再次找到相关的jar包，传给kafka的lib下：

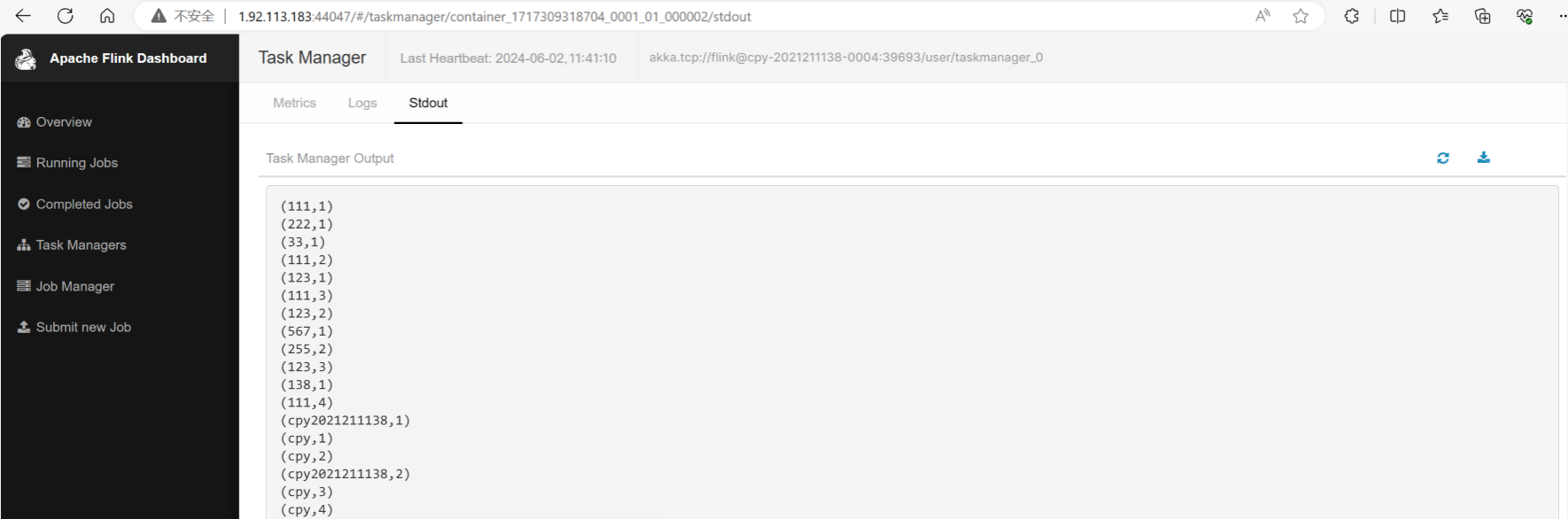
但是kafka的lib下已经有这个包了

但是flink下面没有这个包，于是把这个包放到flink的lib下面。

重新运行，没有这个报错了。

但是有了新的报错：





[root@cpy-2021211138 flink]# bin/flink run examplex/batch/WordCount.jar -inputhdfs://node1:8020/flink\_input -output hdfs://node1:8020/flink\_coutput/wordcount-result.txt