1. 作业5说明

211275009 陈铭浩

1.1. maven创建项目后，更改pom.xml相关配置

```
#使用maven-archetype-quickstart(version=1.4)
1. 修改编译使用的jdk版本（本机为JDK1.8）
  <properties>
    <project.build.sourceEncoding>UTF-
8</project.build.sourceEncoding>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
2. 添加项目用到的Java依赖包
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>org.apache.hadoop</groupId>
      <artifactId>hadoop-client</artifactId>
      <version>3.3.6</version>
    </dependency>
    <dependency>
      <groupId>org.apache.hadoop</groupId>
      <artifactId>hadoop-common</artifactId>
      <version>3.3.6</version>
    </dependency>
    <dependency>
      <groupId>org.apache.hadoop</groupId>
```

```
      <artifactId>hadoop-hdfs</artifactId>
      <version>3.3.6</version>
    </dependency>
    <dependency>
      <groupId>org.apache.hadoop</groupId>
      <artifactId>hadoop-mapreduce-client-core</artifactId>
      <version>3.3.6</version>
    </dependency>
    <dependency>
      <groupId>org.apache.hadoop</groupId>
      <artifactId>hadoop-mapreduce-client-common</artifactId>
      <version>3.3.6</version>
    </dependency>
  </dependencies>
```

1.2. 程序设计思路

1. 实验要求
   对于给定的两个输入文件A和B（如附件所示），文件内包含了纳斯达克100指数、标普500指
   数、道琼斯工业指数的部分成分股数据，第一列为三大指数编号（简化为101、102、103），第
   二列为成分股代码。请编写Mapreduce程序，对两个文件内容进行合并，并对重复内容进行剔
   除，最后合并为一个包含三大指数成分股的输出文件。输出文件格式与文件A和B格式相同。

2. 设计思路
   首先将xlsx文件转为csv文件便于后续处理，而后在UnionCalc主类中包括UnionCalcMapper 类，UnionCalcReducer 类和main方法

- UnionCalcMapper 类：
  这是一个Mapper类，用于处理输入数据。
  在 map 方法中，将每一行文本数据（CSV记录）转换为字符串，然后使用逗号分隔符拆分记录，提取索引和股票代码。
  为了去除不必要的空格，使用 .trim() 方法来修整索引和股票代码。
  最后，将索引作为键，股票代码作为值，发送到Reducer以分组相同索引的记录。

- UnionCalcReducer 类：
  这是一个Reducer类，用于处理来自Mapper的键值对数据，去除重复记录。
  在 reduce 方法中，创建了一个 Set 集合（stockCodeSet）来存储唯一的股票代码。
  然后，遍历相同索引下的所有股票代码，将它们添加到 stockCodeSet 中，从而去除重复。
  最后，将去除重复后的记录输出。

- main 方法：
  main 方法是程序的入口点，用于配置Hadoop作业。
  通过 Job 类，指定了Map和Reduce的类，以及输入和输出数据的格式。
  设置了输入文件路径和输出文件路径。
  最后，启动Hadoop作业，等待作业完成，然后退出。

```java
public class UnionCalc{
    public static class UnionCalcMapper extends Mapper<Object, Text, Text, Text>{
    protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {
        String line = value.toString();

        String[] fields = line.split(",");

        if (fields.length >= 2) {
            String index = fields[0].trim();
            String stockCode = fields[1].trim();
            context.write(new Text(index), new Text(stockCode));
        }
    }
}

public static class UnionCalcReducer extends Reducer<Text, Text, Text, Text>{
    protected void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
        Set<String> stockCodeSet = new HashSet<>();

        // 遍历某个index下所有相同成分股代码的记录，将其添加到集合中
        for (Text value : values) {
            stockCodeSet.add(value.toString());
        }

        // 输出去除重复后的记录
        for (String stockCode : stockCodeSet) {
            context.write(key, new Text(stockCode));
        }
    }
}
```

```java
    public static void main(String[] args) throws Exception{
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, jobName:"UnionCalc");

        job.setJarByClass(cls:UnionCalc.class);
        job.setMapperClass(cls:UnionCalcMapper.class);
        job.setReducerClass(cls:UnionCalcReducer.class);

        job.setOutputKeyClass(theClass:Text.class);
        job.setOutputValueClass(theClass:Text.class);

        job.setInputFormatClass(cls:org.apache.hadoop.mapreduce.lib.input.TextInputFormat.class);
        job.setOutputFormatClass(cls:org.apache.hadoop.mapreduce.lib.output.TextOutputFormat.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));  // 输入文件A和B的路径input

        FileOutputFormat.setOutputPath(job, new Path(args[1]));  // 输出文件路径output

        System.exit(job.waitForCompletion(verbose:true) ? 0 : 1);
    }
}
```

1.3. 打包maven项目

```
1. 运行 Maven 编译和打包命令：
#在终端中，使用以下 Maven 命令来编译和打包你的项目：
mvn clean package
#这个命令会执行 clean 生命周期（清理旧构建），然后执行 package 生命周期，将项目打包为 JAR 文件。构建好的 JAR 文件将位于项目目录的 target 子目录下。
2. 查找 JAR 文件：
#构建成功后，你可以在项目目录的 target 文件夹中找到生成的 JAR 文件，通常以项目名称和版本号命名，如 my-java-project-1.0-SNAPSHOT.jar。
```

1.4. 在hadoop中运行程序

1. 准备伪分布式hadoop

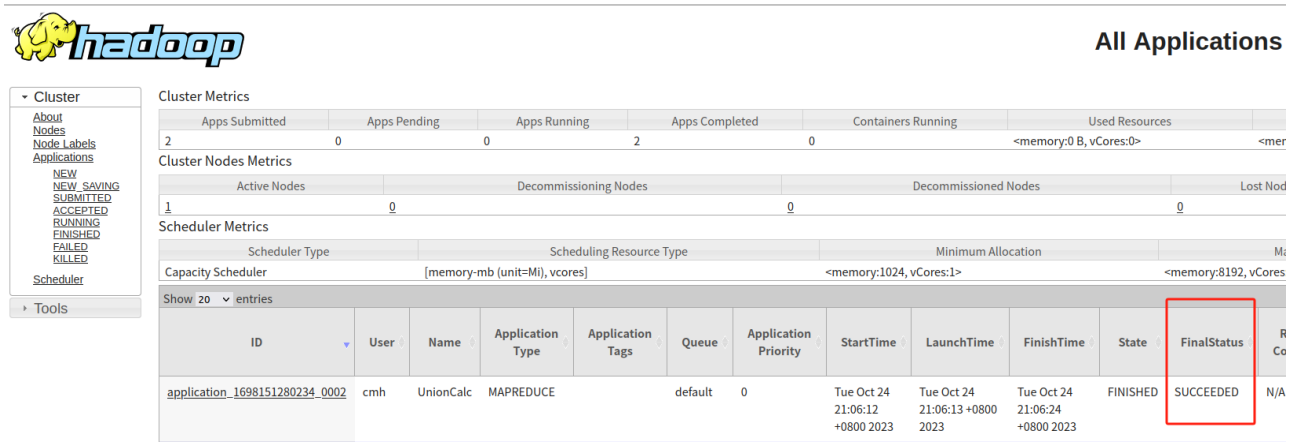
```
start-all.sh
```

2. 将jar文件和输入文件上传到Hadoop中
3. 使用 hadoop jar 命令来运行 JAR 文件

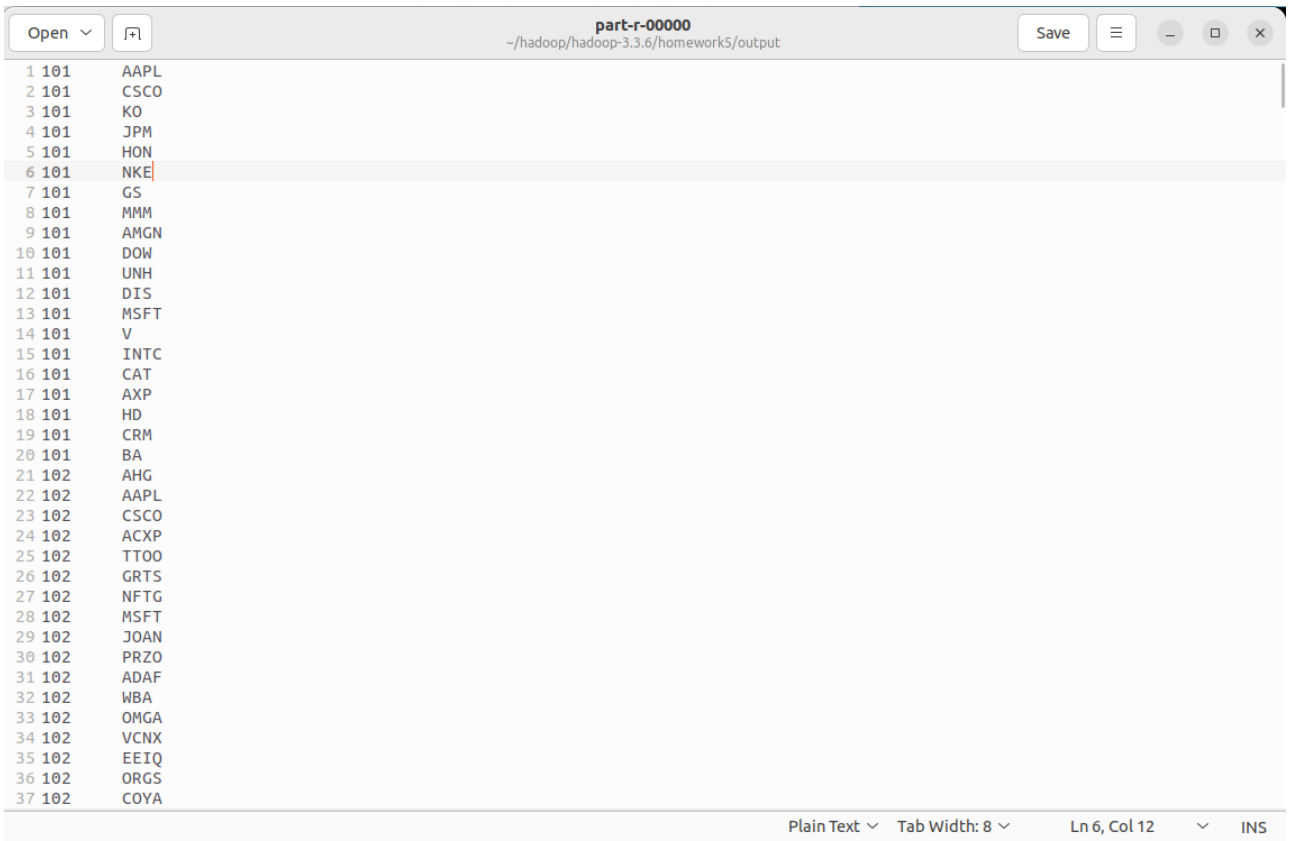
```
hadoop jar your-project.jar main-class input-path output-path
#your-project.jar 是你上传到 Hadoop 的 JAR 文件
#main-class 是包含 main 方法的 Java 类的全名，用于启动你的作业
#input-path 是输入数据的 HDFS 路径
#output-path 是输出结果的 HDFS 路径
```

1.5. 运行结果

```
2023-10-24 21:06:17,622 INFO mapreduce.Job:  map 0% reduce 0%
2023-10-24 21:06:21,664 INFO mapreduce.Job:  map 100% reduce 0%
2023-10-24 21:06:25,683 INFO mapreduce.Job:  map 100% reduce 100%
2023-10-24 21:06:25,690 INFO mapreduce.Job: Job job_1698151280234_0002 completed successfully
2023-10-24 21:06:25,749 INFO mapreduce.Job: Counters: 54
```

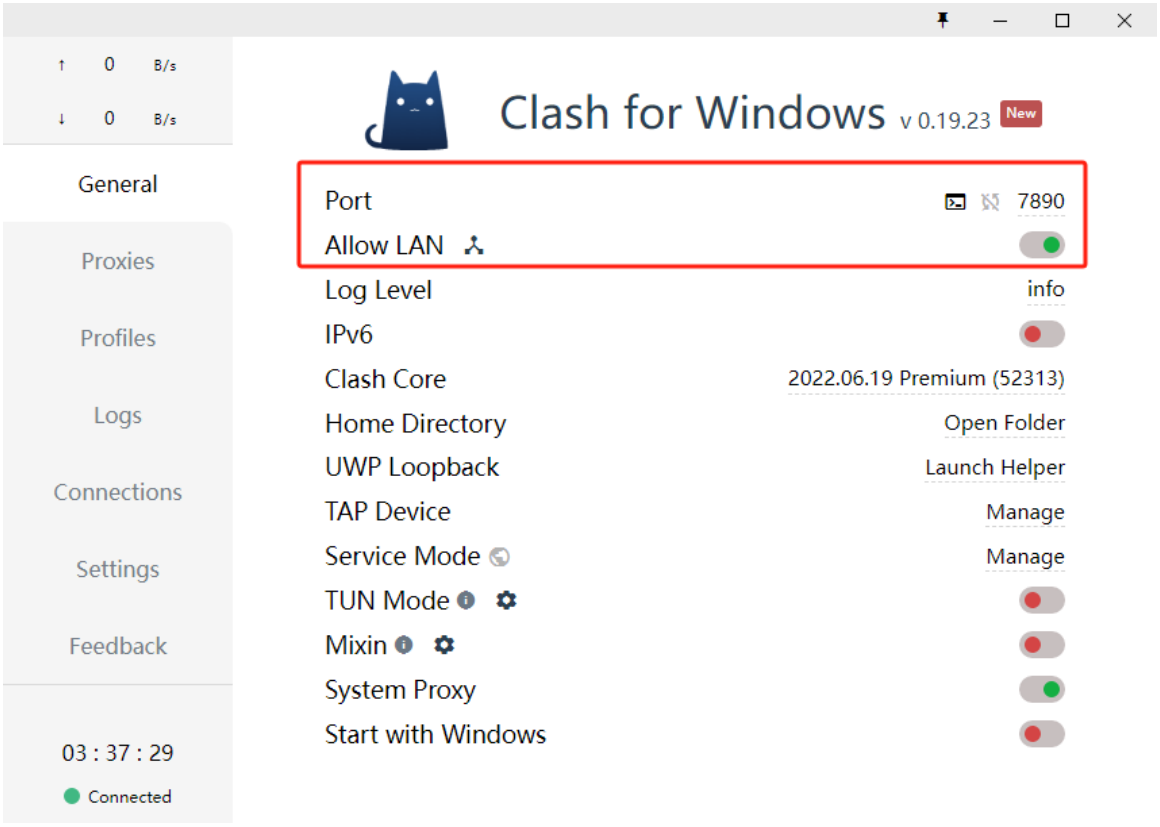


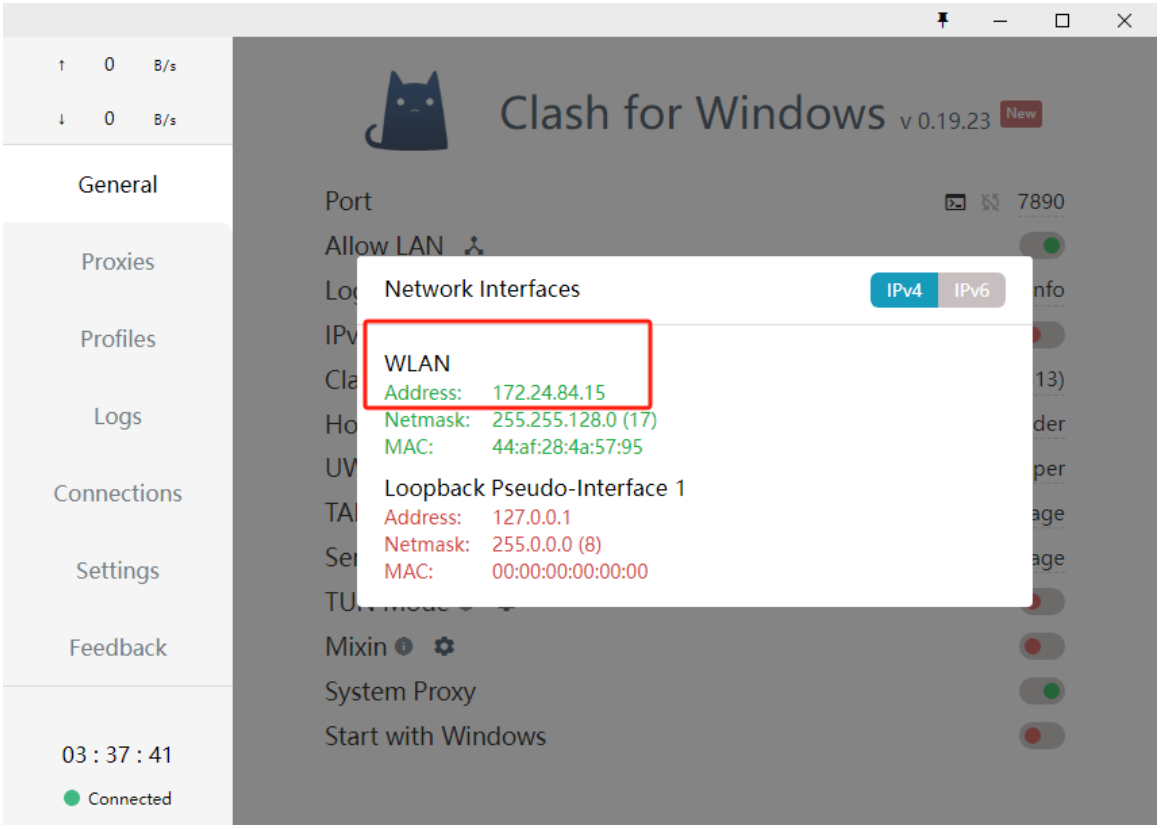
输出文件（只展示了前37行）



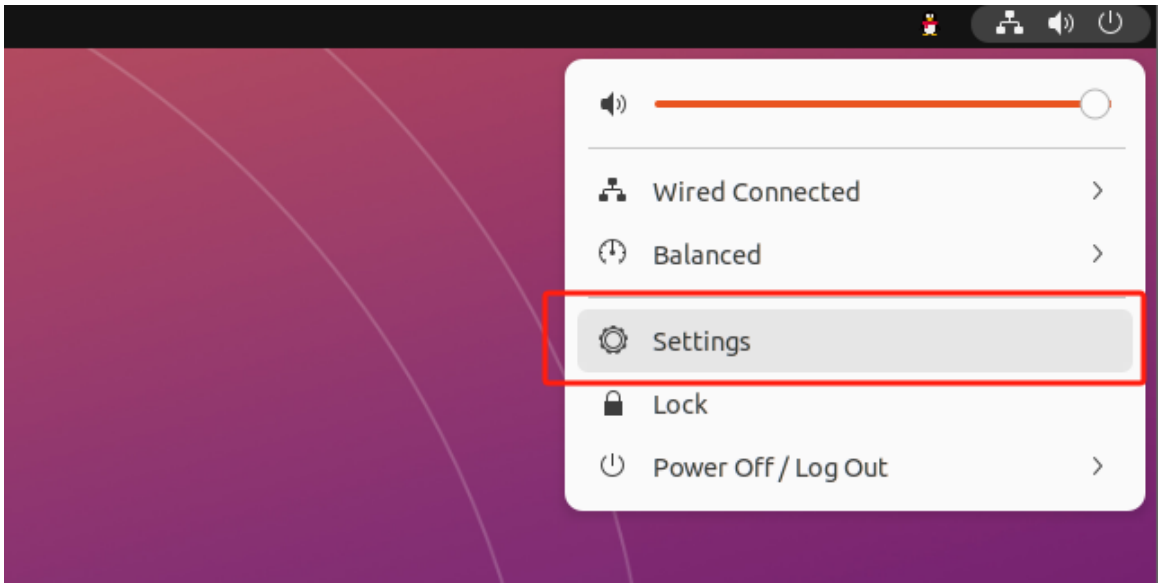
1.6. 附：由于在本作业中涉及到了虚拟机中**github**的使用，因此将在虚拟机中科学上网的方法记录如下

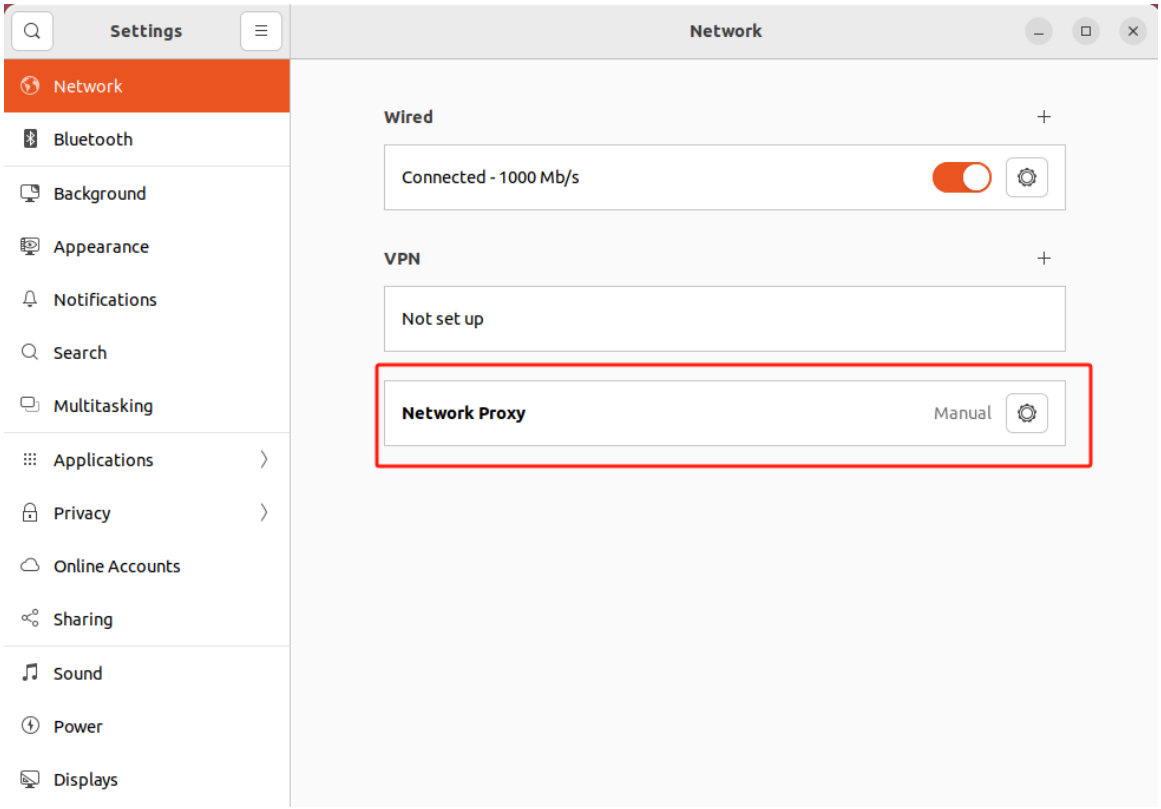
1. 打开clash中的Allow LAN按钮，点击三角符号，记录WLAN的IP地址及端口号

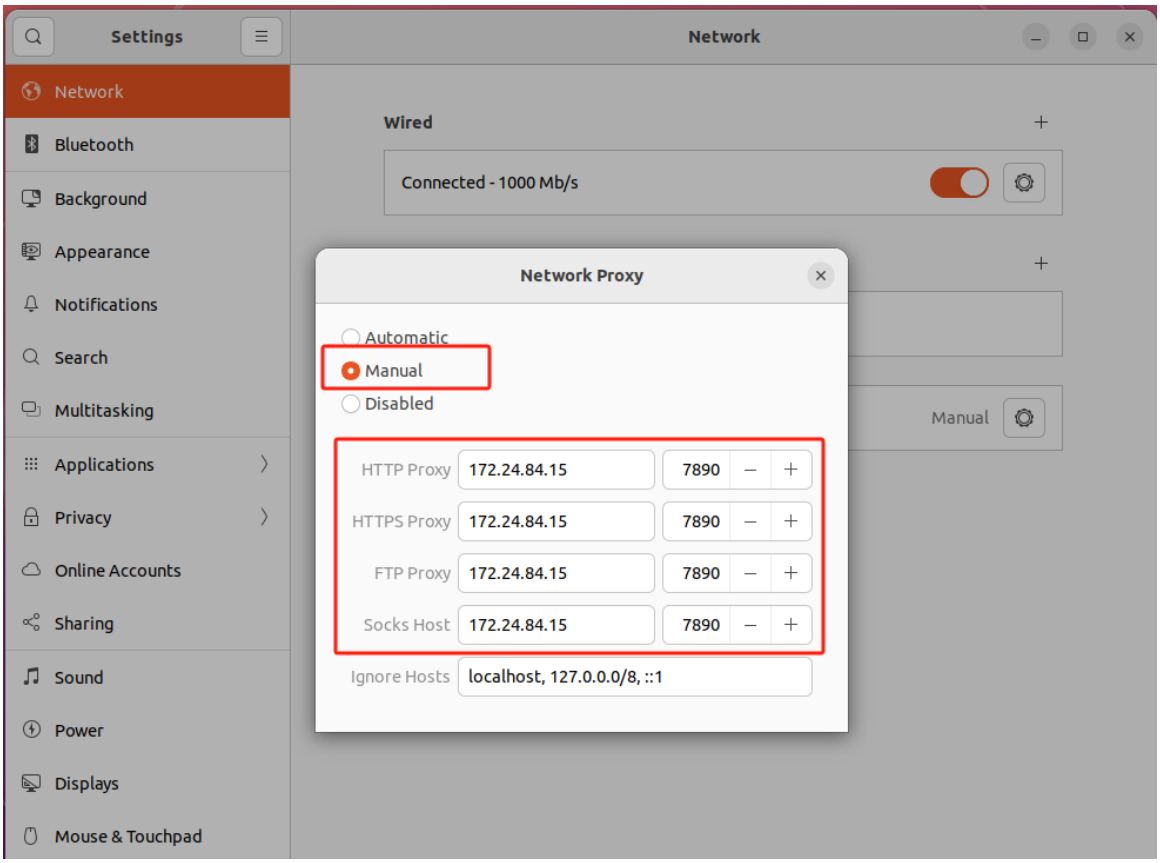




2. 打开虚拟机右上角的settings-Network，将Network Proxy修改为manual，并在四行中填入之前记录的IP及端口号


Wired Connected
Balanced
Settings
Lock
Power Off / Log Out


Settings
Network
Network
Bluetooth
Background
Appearance
Notifications
Search
Multitasking
Applications
Privacy
Online Accounts
Sharing
Sound
Power
Displays
Wired
Connected - 1000 Mb/s
VPN
Not set up
Network Proxy
Manual


Settings
Network
Network
Bluetooth
Background
Appearance
Notifications
Search
Multitasking
Applications
Privacy
Online Accounts
Sharing
Sound
Power
Displays
Mouse & Touchpad
Wired
Connected - 1000 Mb/s
Manual
Network Proxy
Automatic
Manual
Disabled
HTTP Proxy
172.24.84.15
7890
HTTPS Proxy
172.24.84.15
7890
FTP Proxy
172.24.84.15
7890
Socks Host
172.24.84.15
7890
Ignore Hosts
localhost, 127.0.0.0/8, ::1

3. 设置完成😄