1. hadoop集群环境搭建
2. 安装VM12
3. 安装CENTOS7
4. 安装时主机叫MASTER
5. 克隆主机 选择创建完整克隆 取名slave0 slave1
6. 启动三个虚拟机
7. 先从master虚拟机走起
8. 切换到root用户
9. 修改主机名 运行该命令 gedit /etc/sysconfig/network

I 输入 NETWORKING=yes

HOSTNAME=master

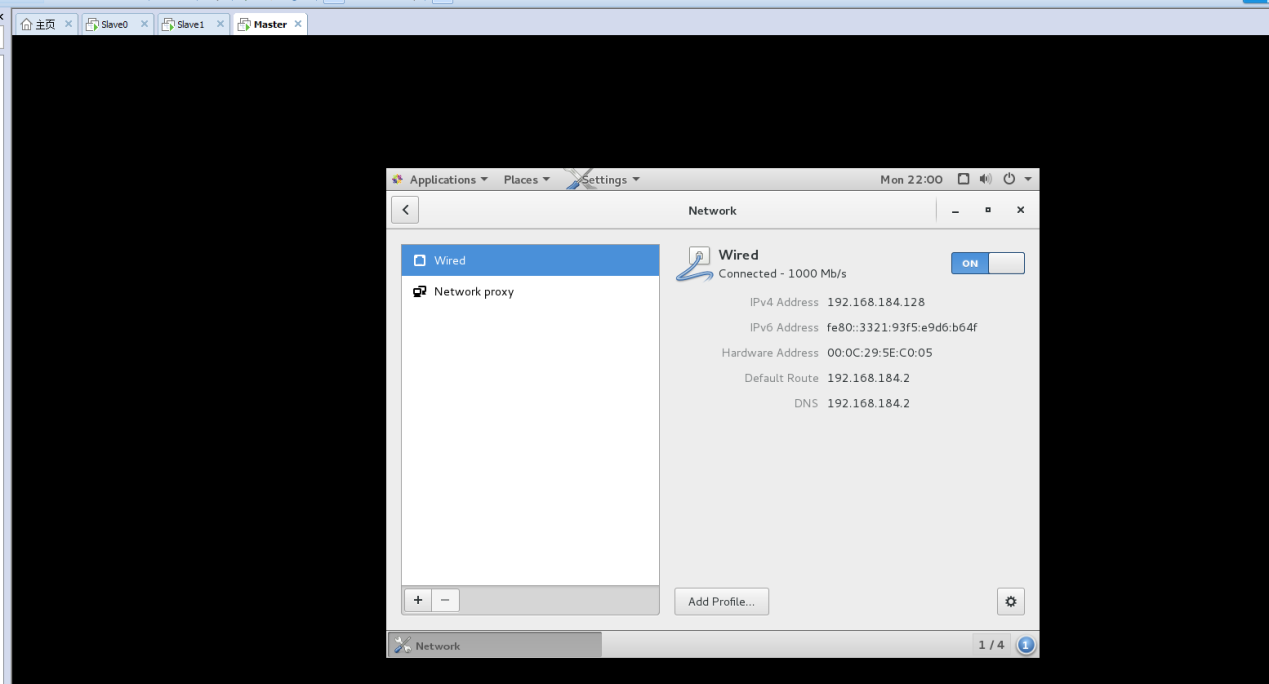
运行命令gedit /etc/hostname 将内容换成master

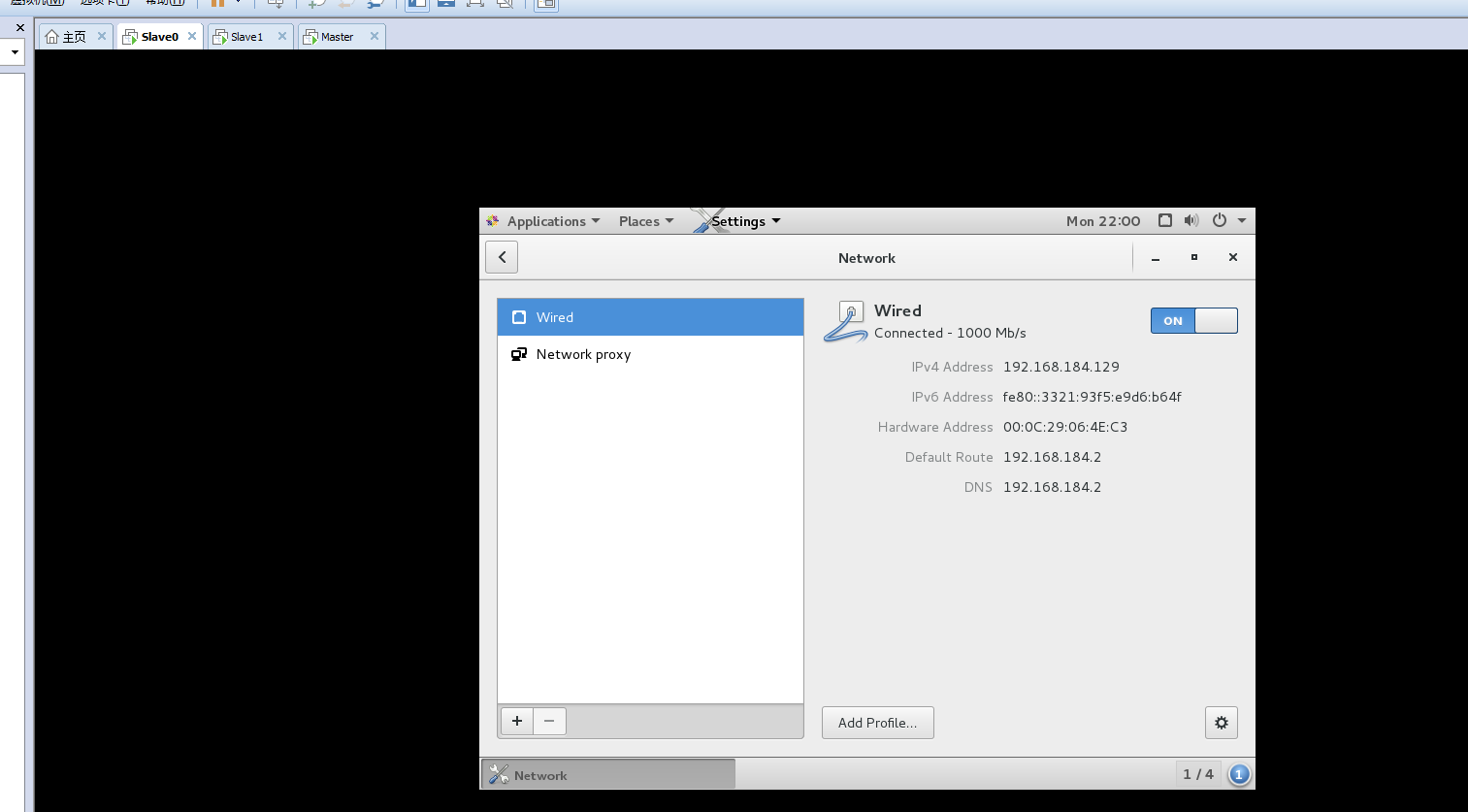
输入hostname master确认修改生效，然后必须先关闭终端再重新打开一个终端

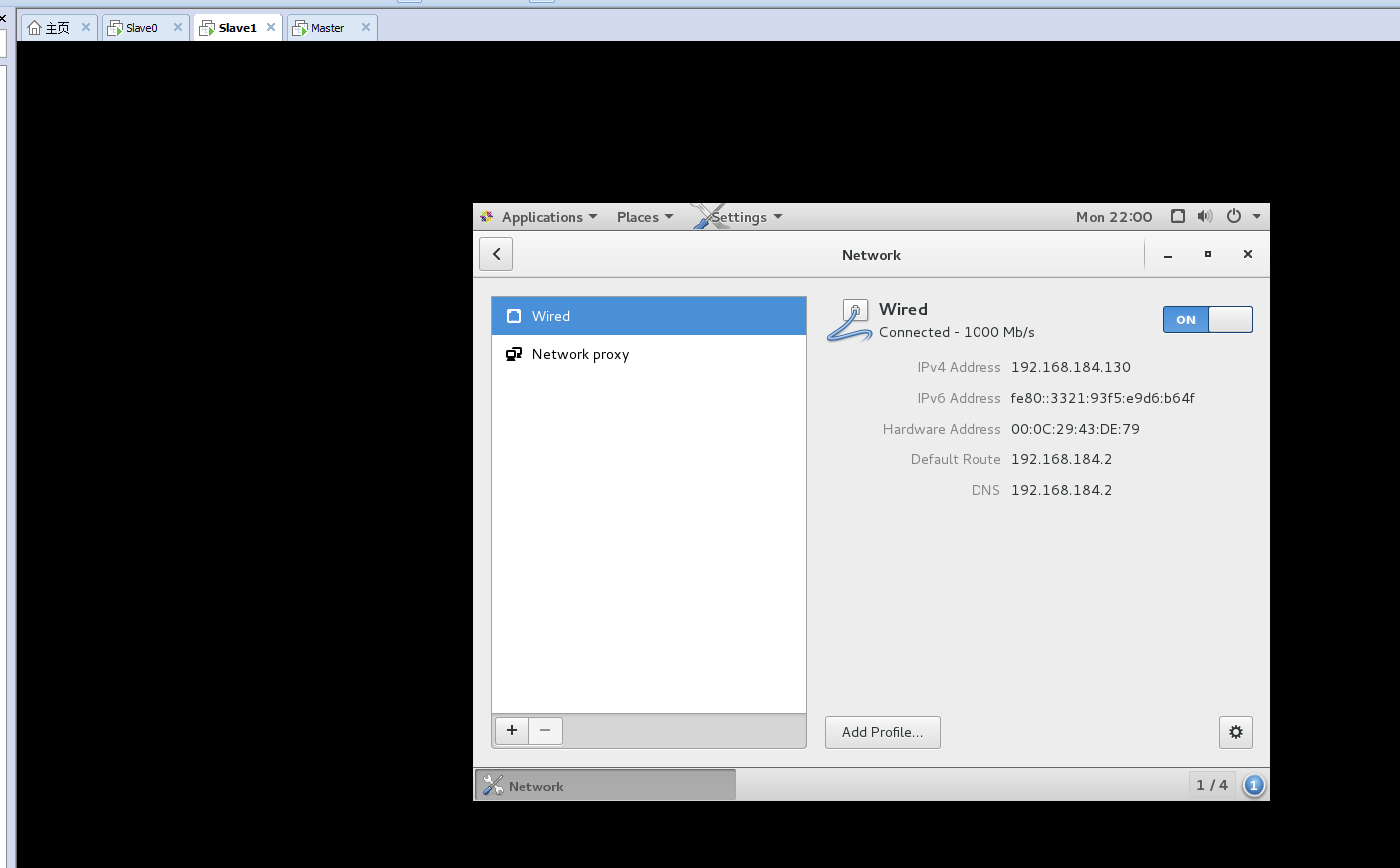
输入hostname查看主机名

接下来同上修改slave0和slave1的主机名

接下来打开三台虚拟机的网络设置 保证在同一个网段内

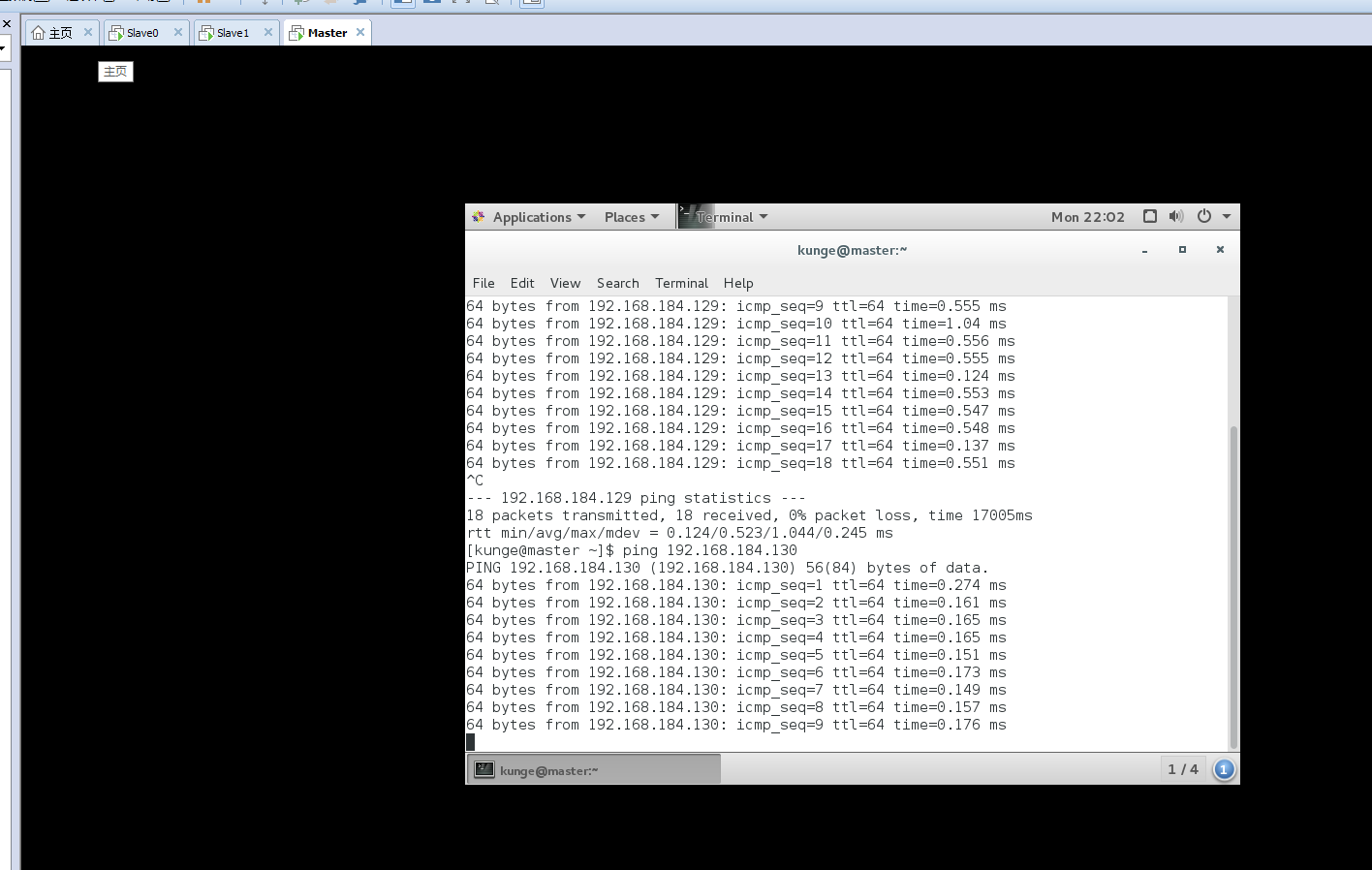






可见三个虚拟机都是192.168.184.128 129 130

接下来要从master虚拟机上ping slave0的连接 看看能不能测试通过



如上图所示表示 ping通

设置地址对照表

Gedit /etc/hosts

输入

192.168.184.128 master

192.168.184.129 slave0

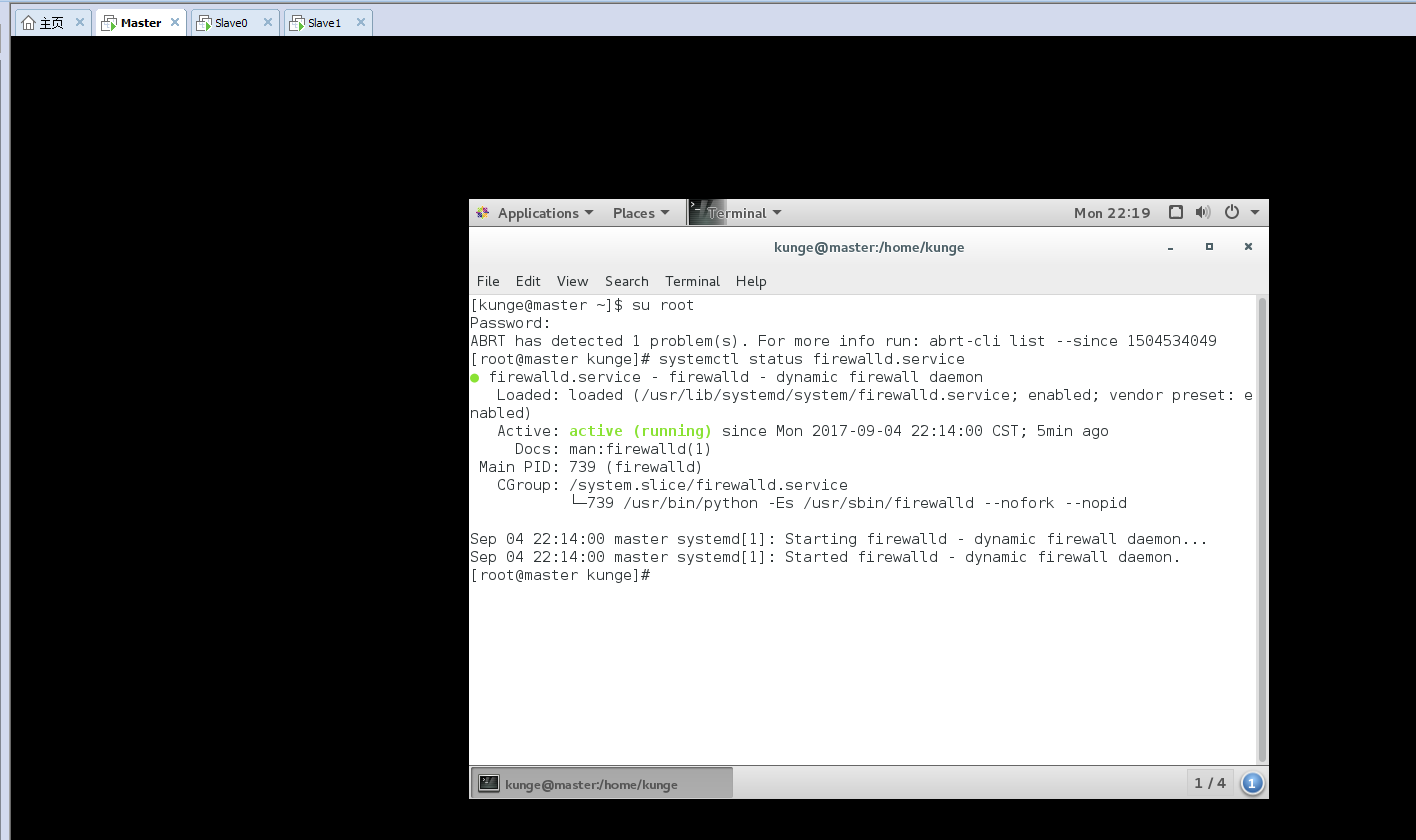
192.168.184.130 slave1

以后不需要记忆ip地址 只需要ping slave0(slave1)就可以了

（！每台机器都要配置）

集群目前配置完成

接下来需要关闭防火墙



上图输入命令： systemctl status firewalld.service

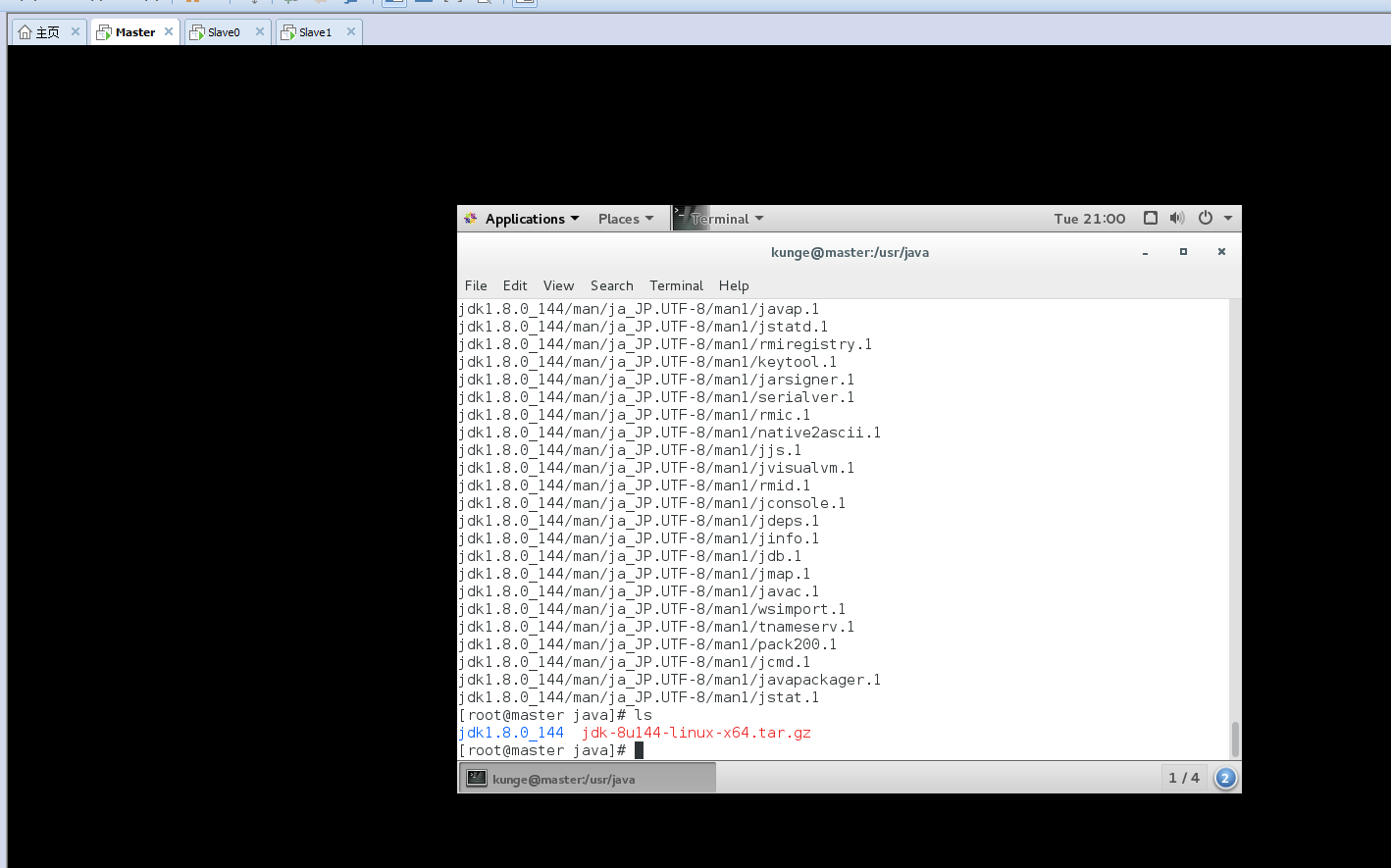
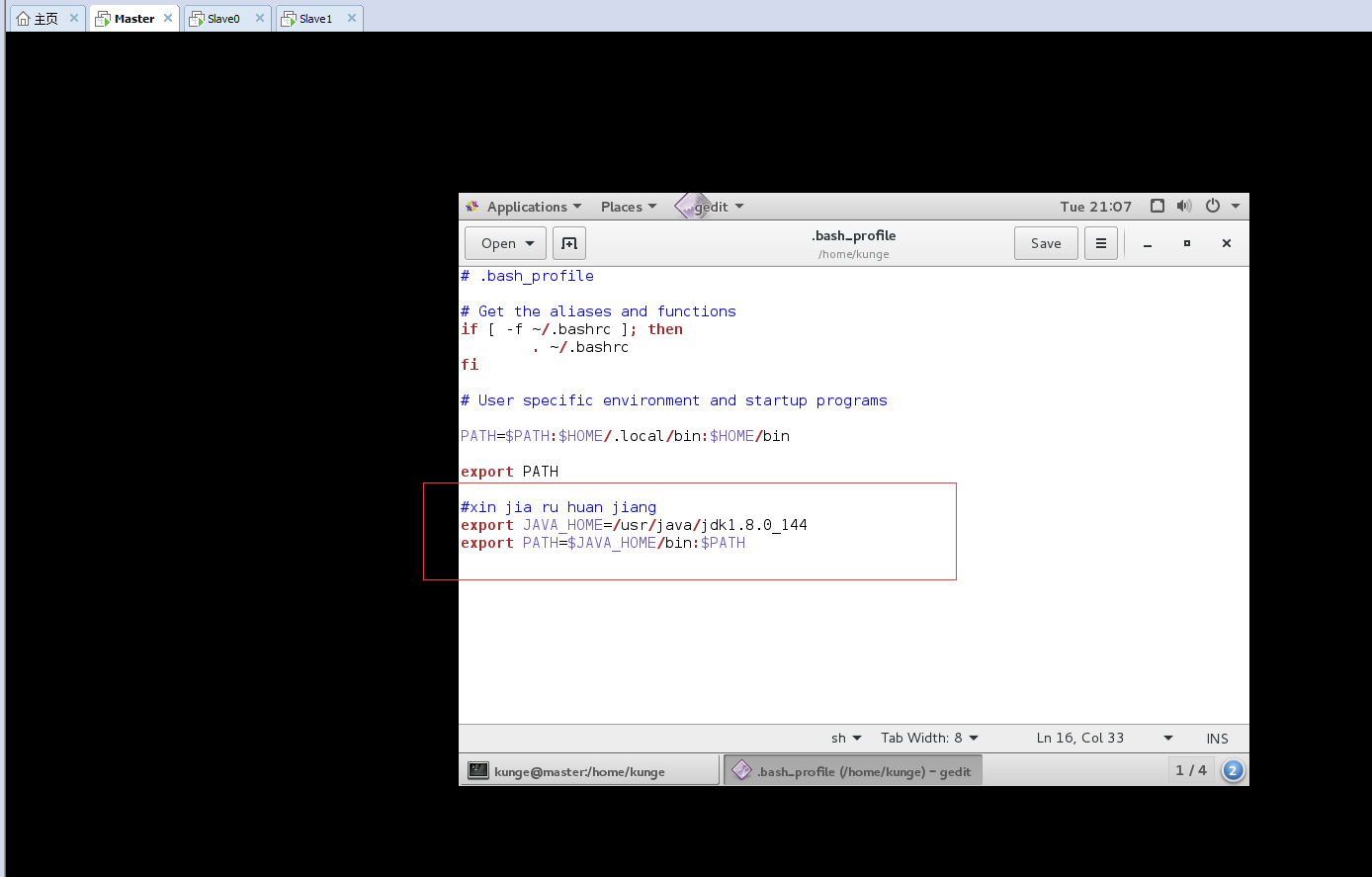
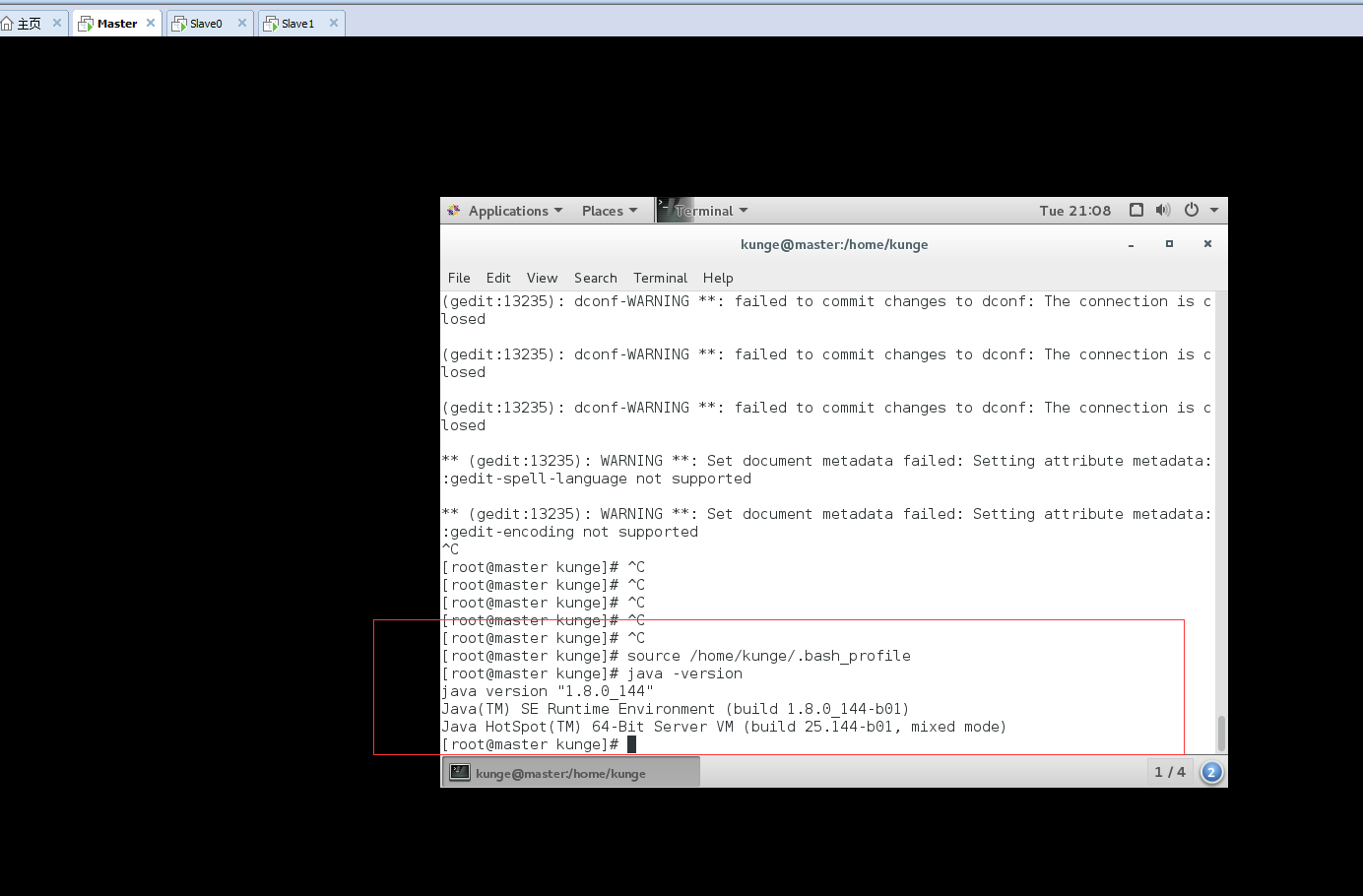
看到active表示开启 现需要关闭

命令： systemctl stop firewalld.service 停止防火墙服务

Systemctl disable firewalld.service 下次开机不启动防火墙

（！每台机器都要配置）

安装JDK

1. 下载jdk1.8解压版
2. 将其传输到usr/java目录下 没有该目录就创建一个
3. 解压 tar –zxvf
4. 
5. 看到上图不断闪动表示解压ok
6. 配置环境变量
7. Gedit /home/kunge/.bash\_profile 其中kunge是刚创建用户的名字
8. 输入
9. 大小写敏感 要注意
10. 保存关闭该文件
11. 执行source /home/kunge/.bash\_profile 使修改生效
12. 
13. 最后输入java –version如上图显示则表示jdk配置安装成功
14. 每一台虚拟机都按照该方法进行

因为未来集群之间需要频繁通信，但linux系统相互通信需要用户身份认证，也就是输入登录密码，在集群规模不大情况下，可以少量输入，但是集群一旦是几十台上百台甚至更多

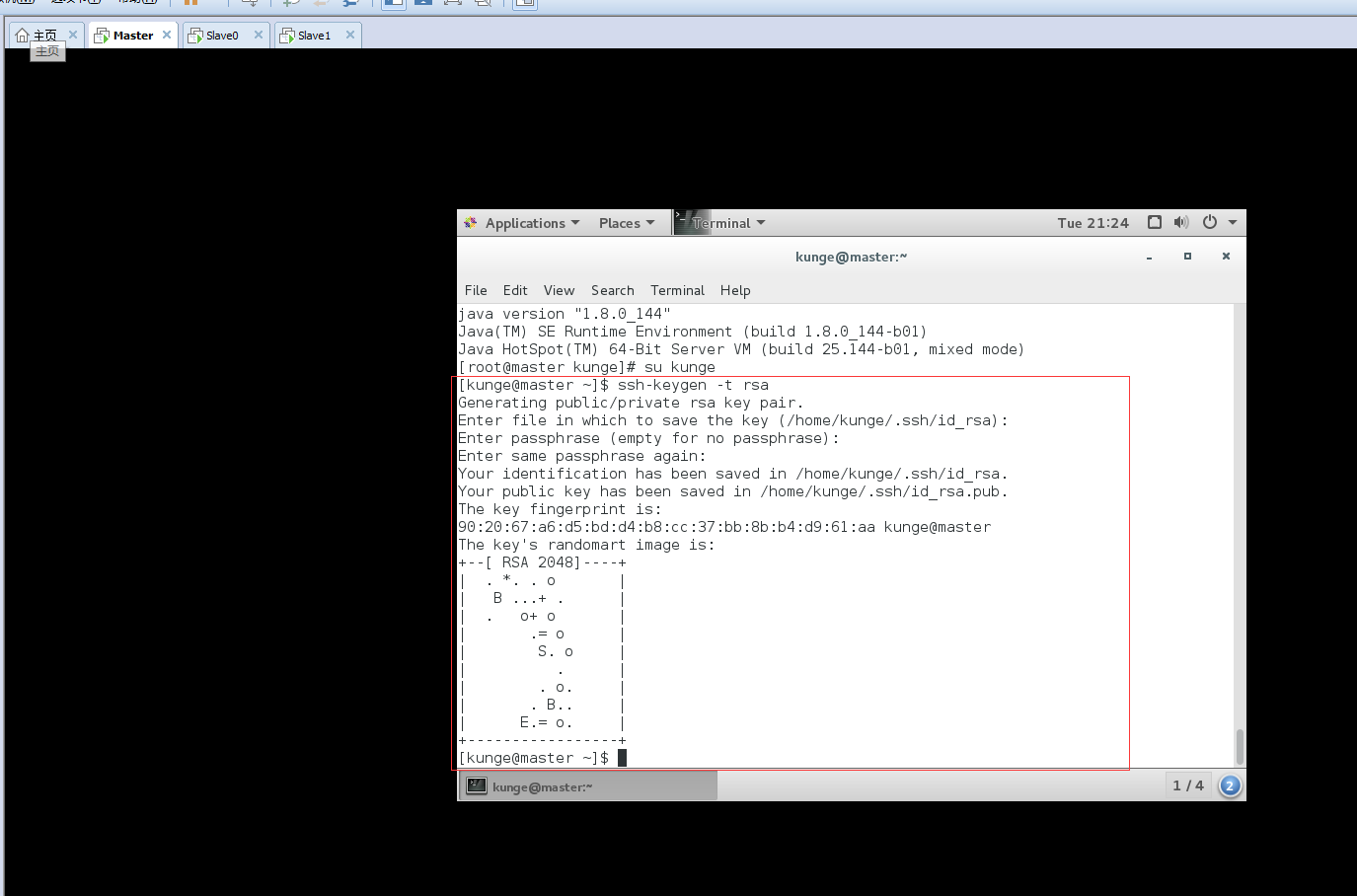
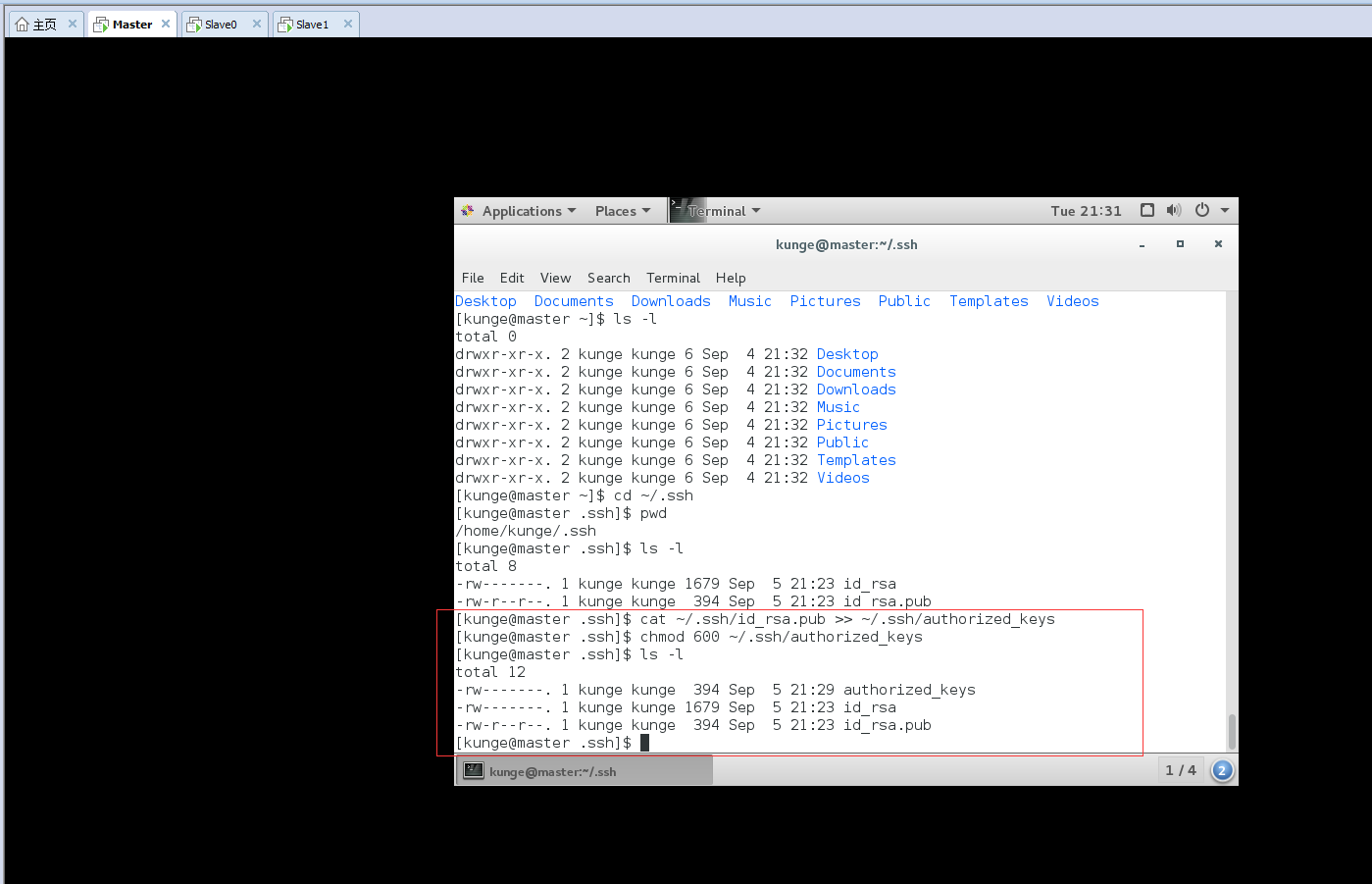
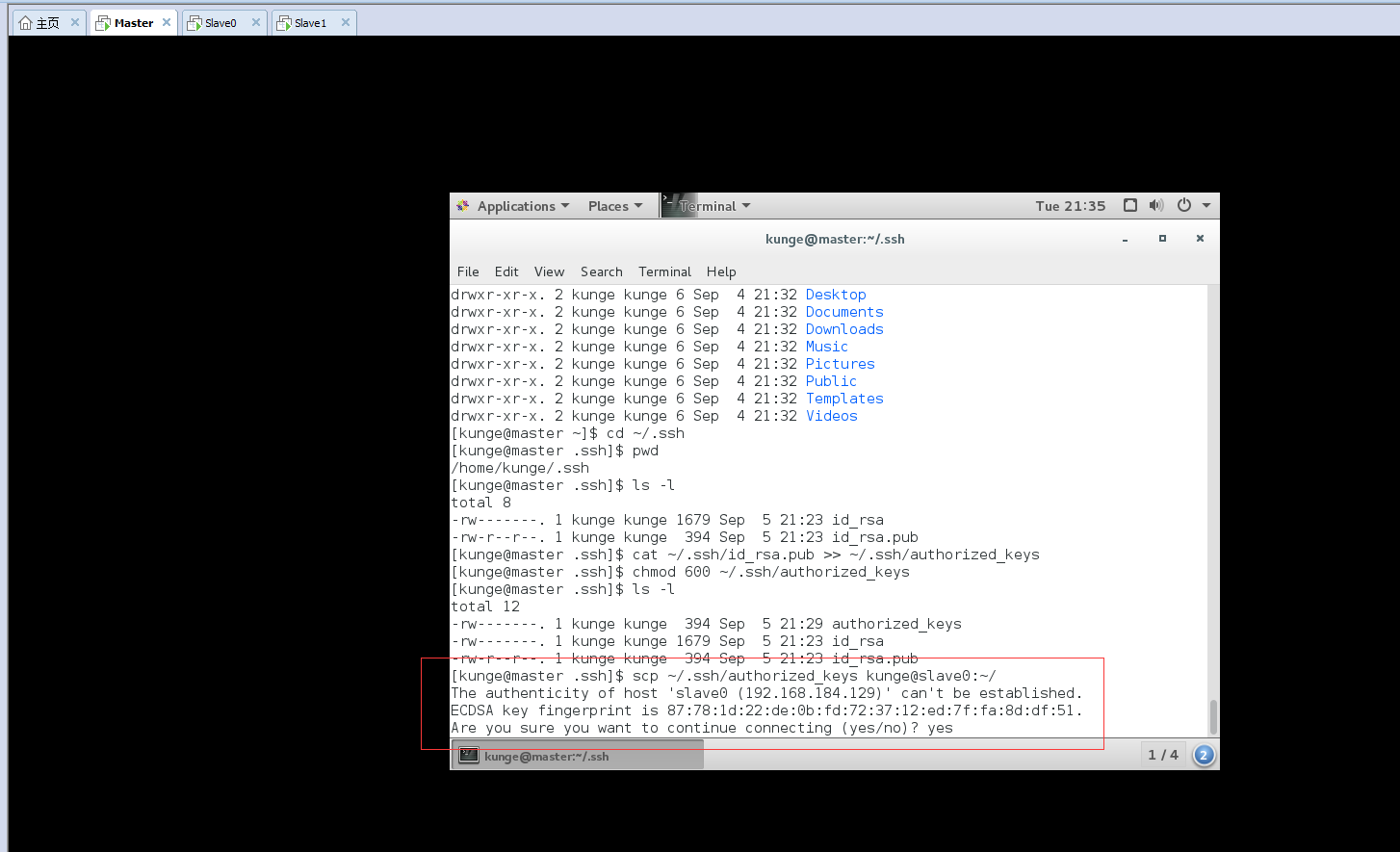
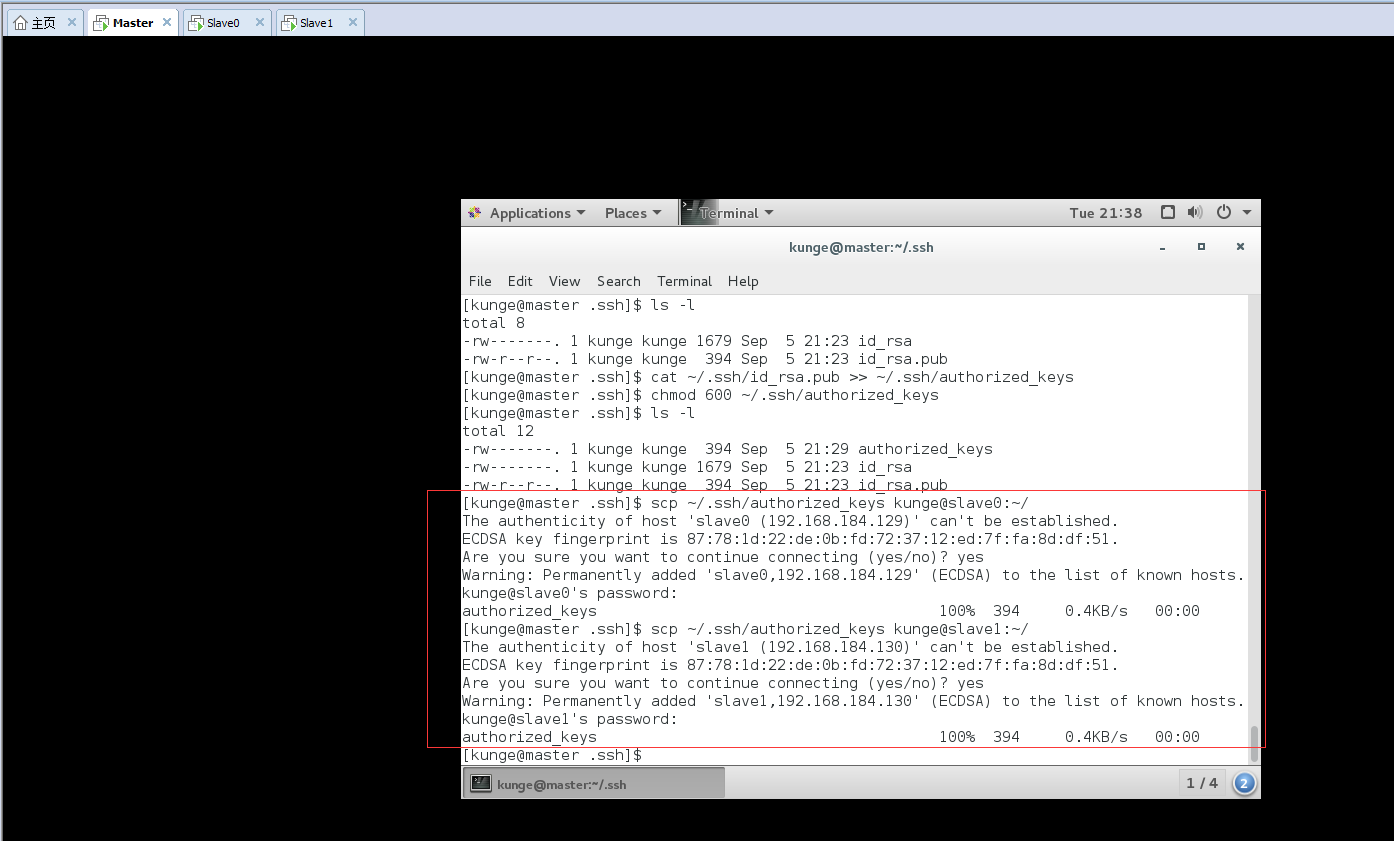
频繁认证就很头疼了，现需要配置免密钥登录

免密钥登录指两台linux机器之间使用SSH连接时不需要用户名和密码（Secure Shell Protocol）安全外壳协议

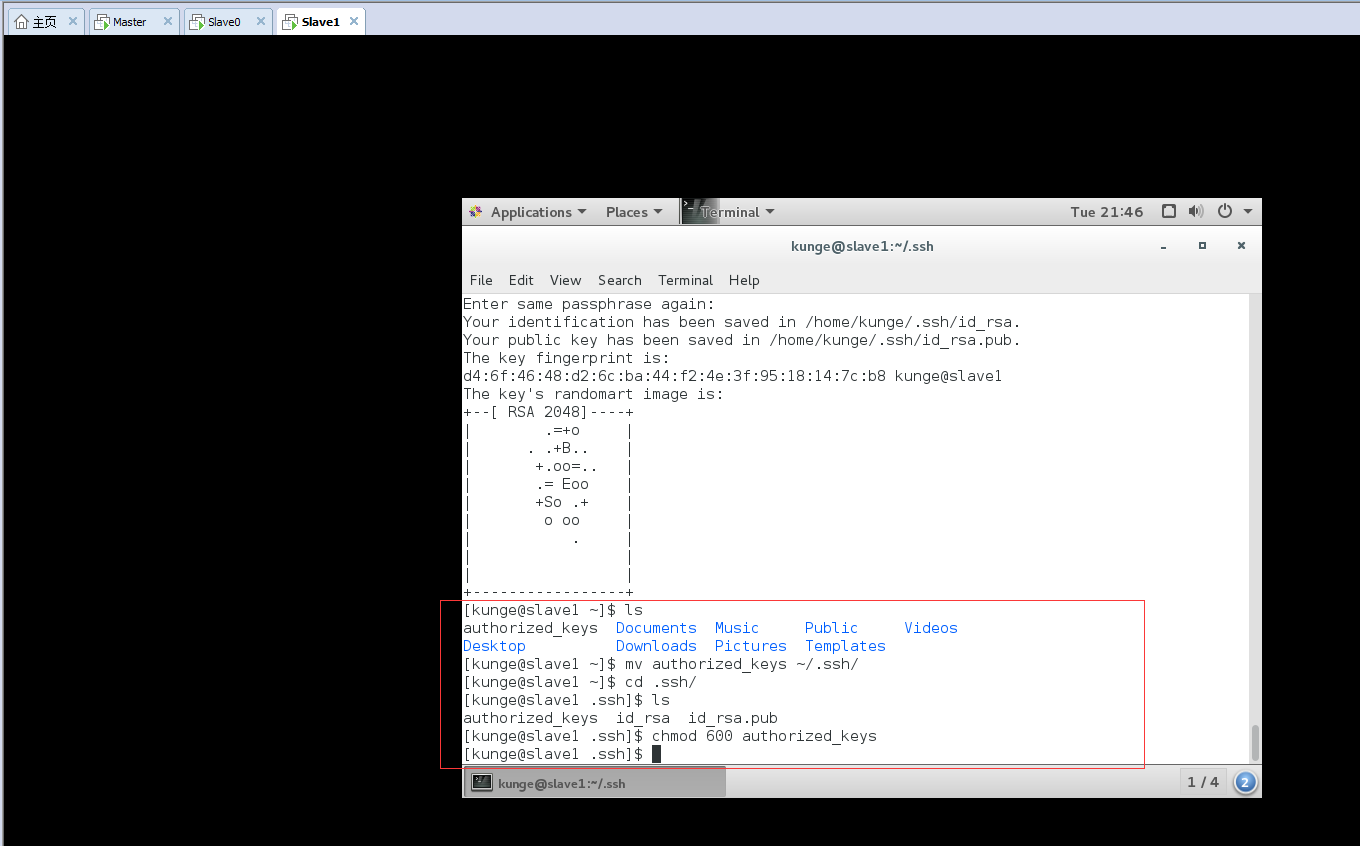
先配置master节点（机器） 注意 很重要 这里要用普通用户操作 如果处于root用户需要切换回普通用户

Master节点配置开始

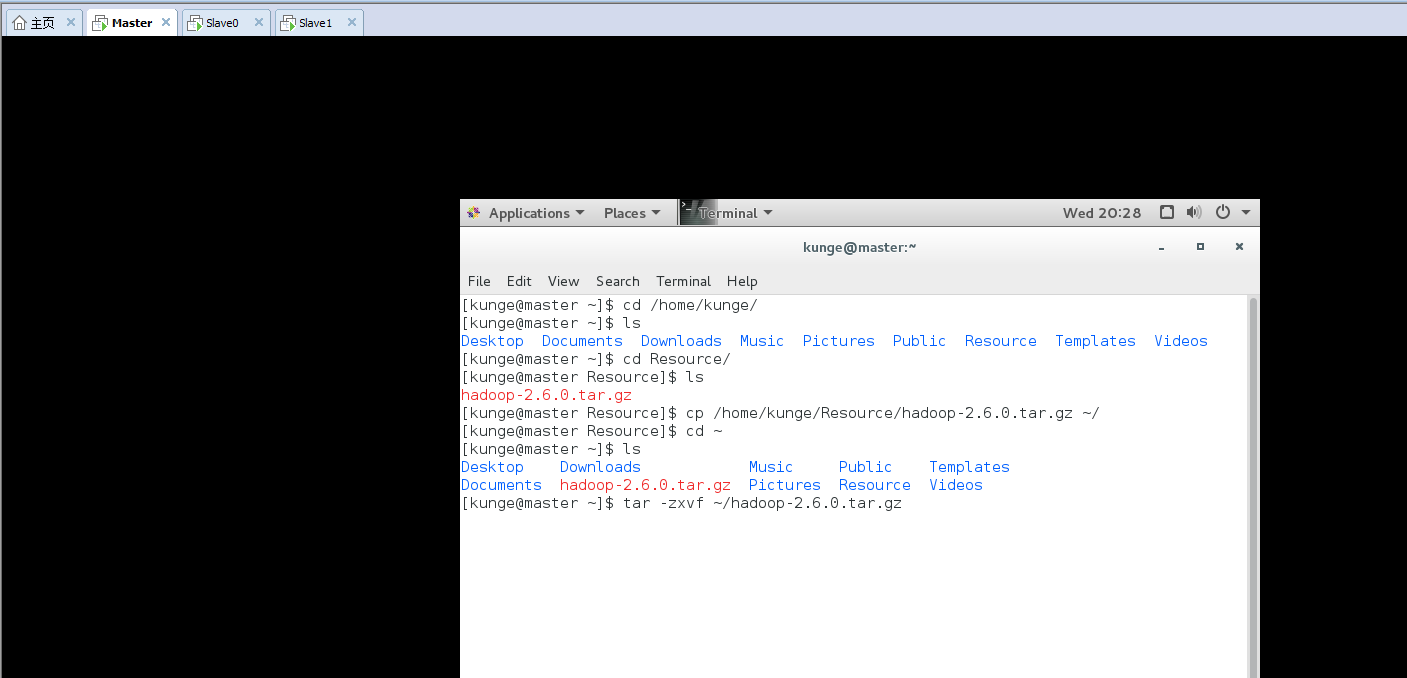
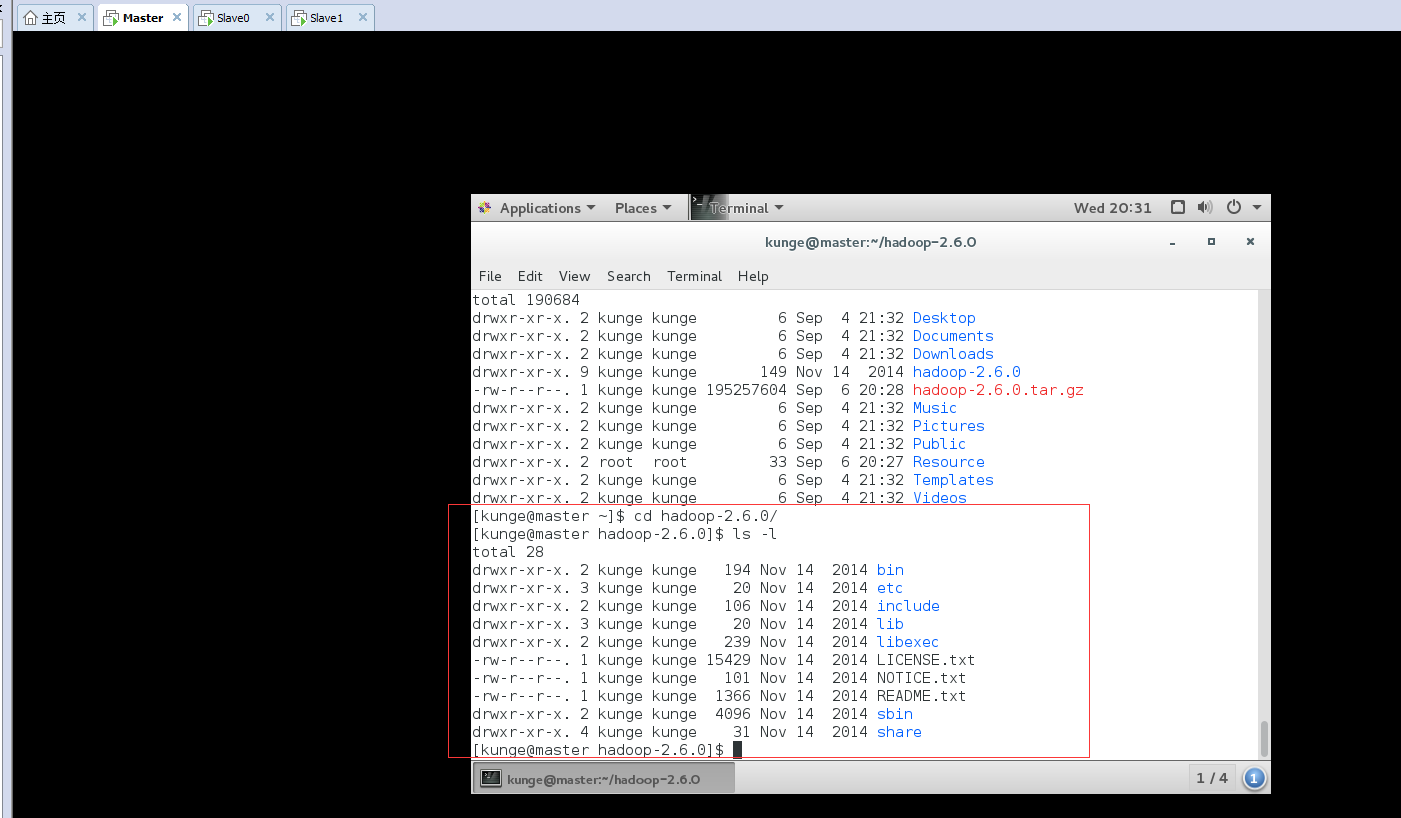
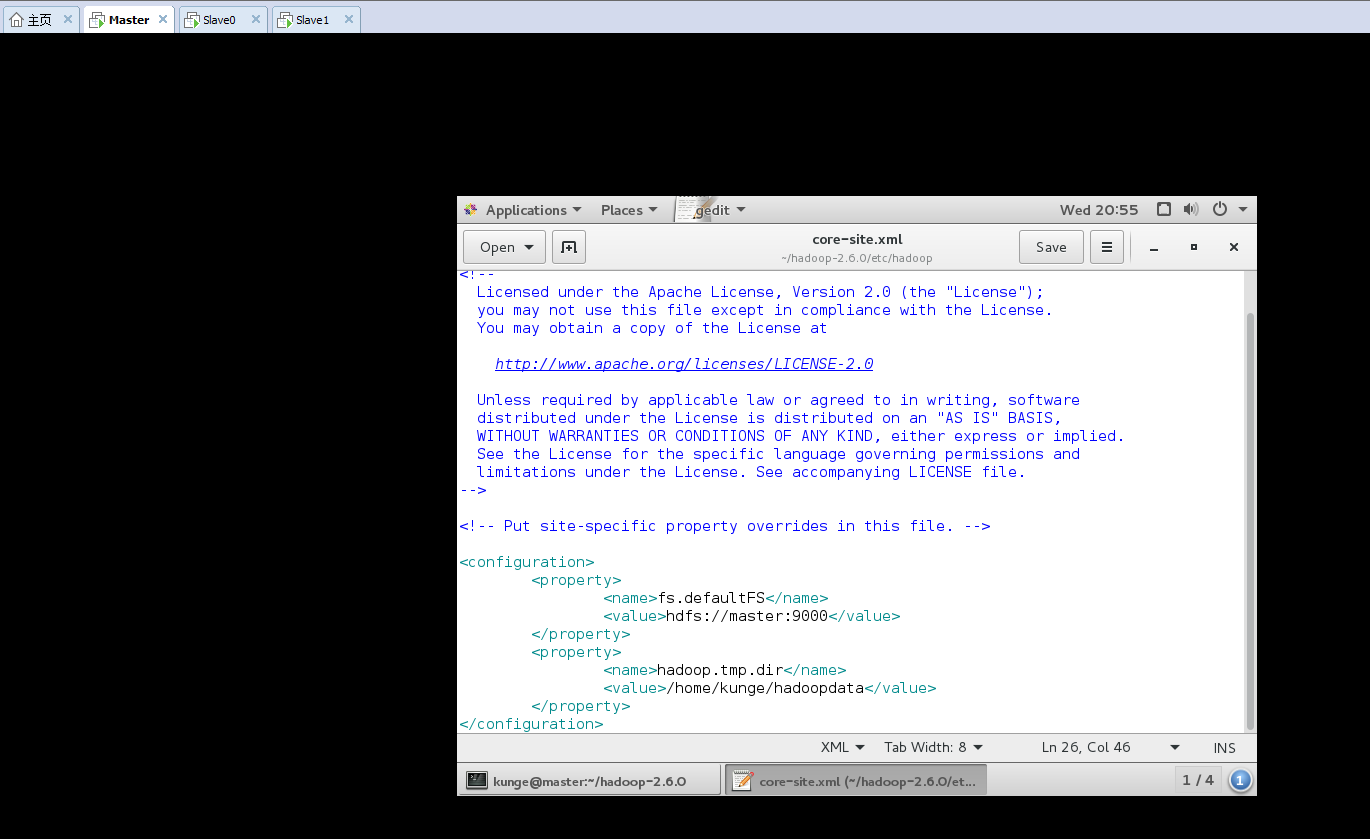
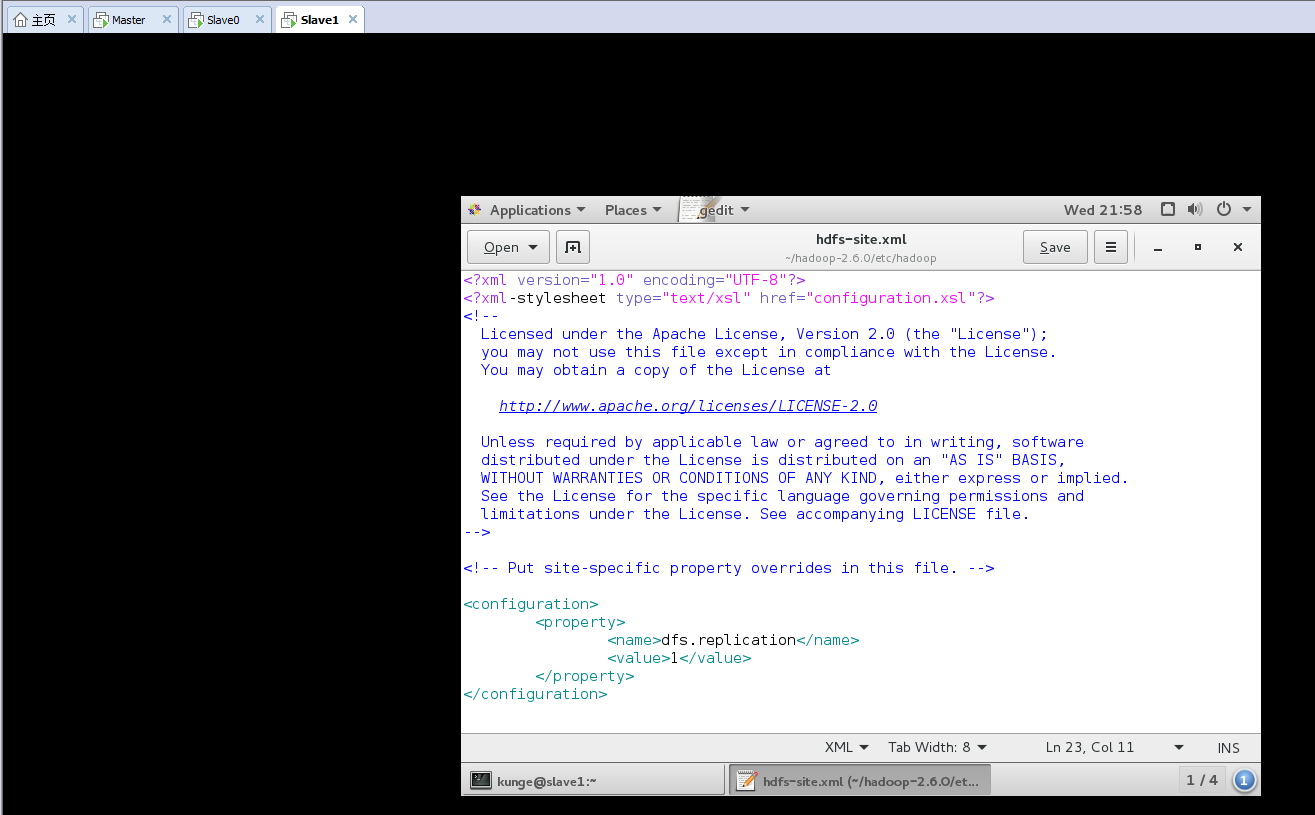
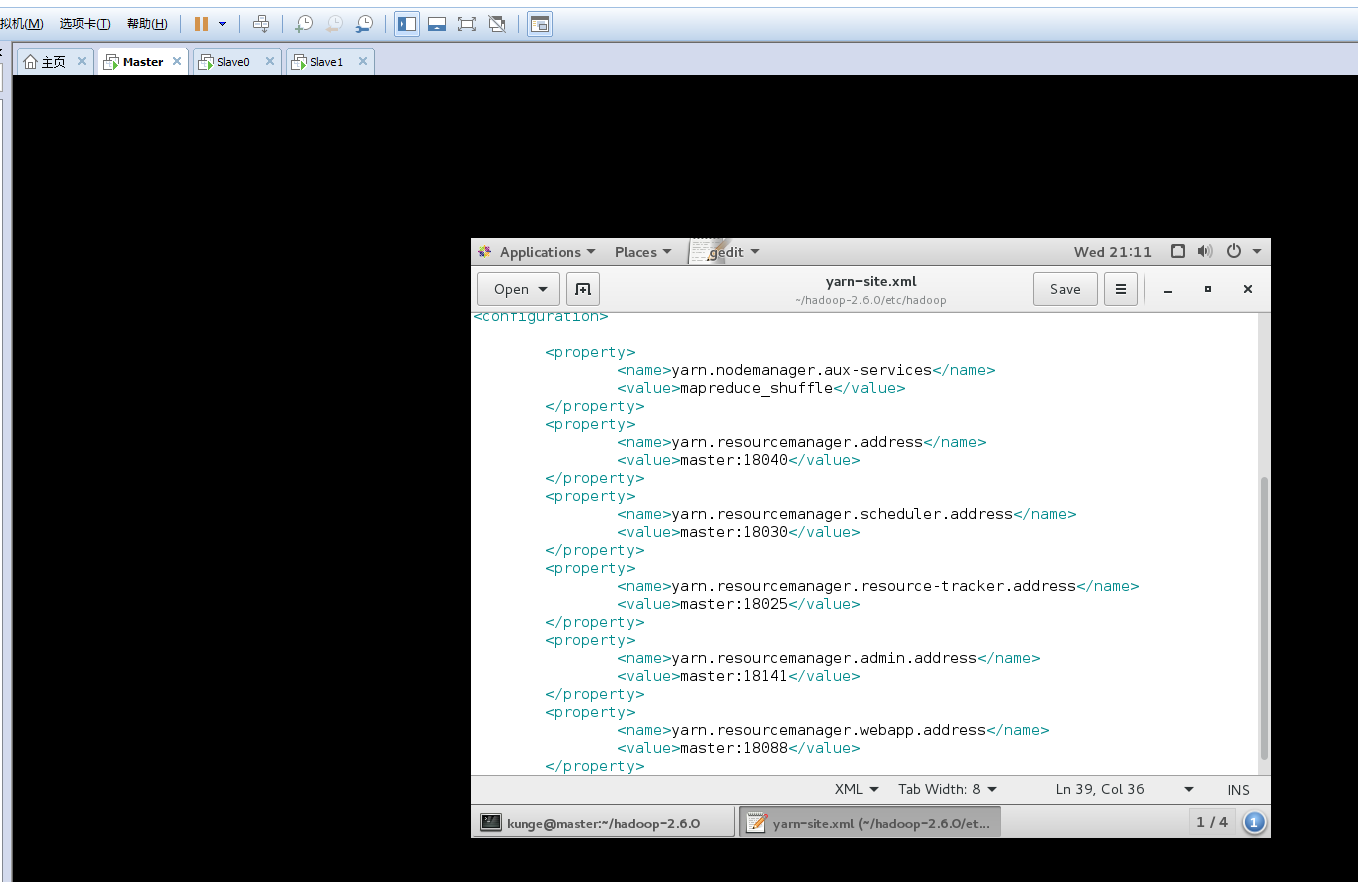
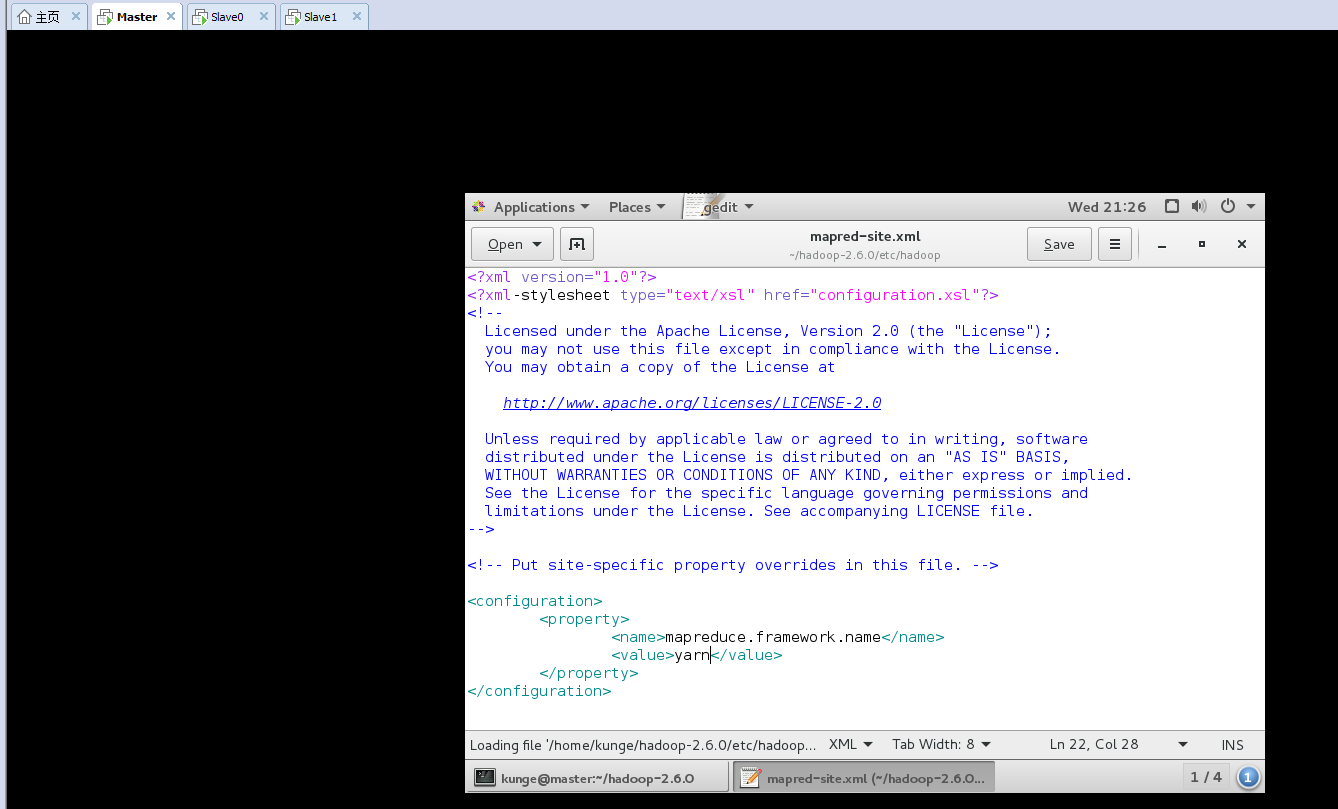
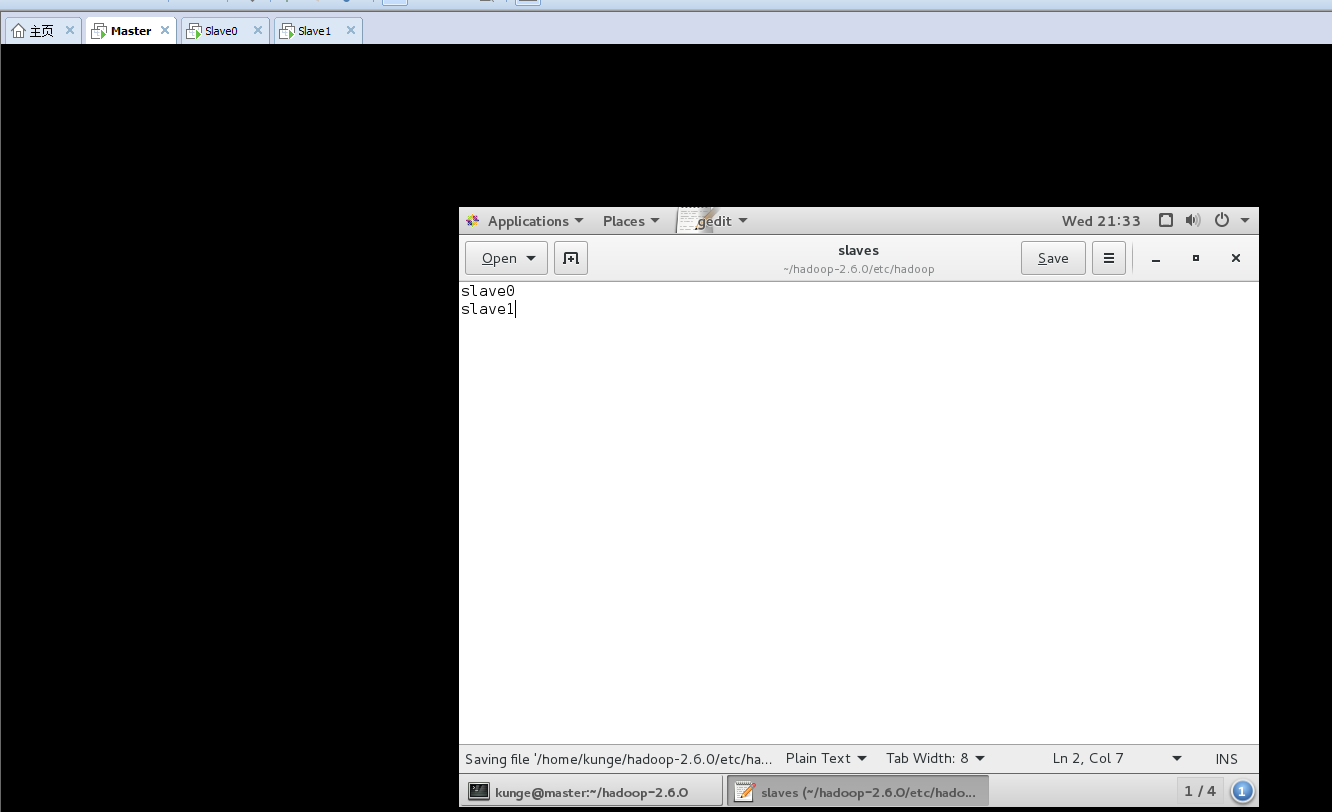
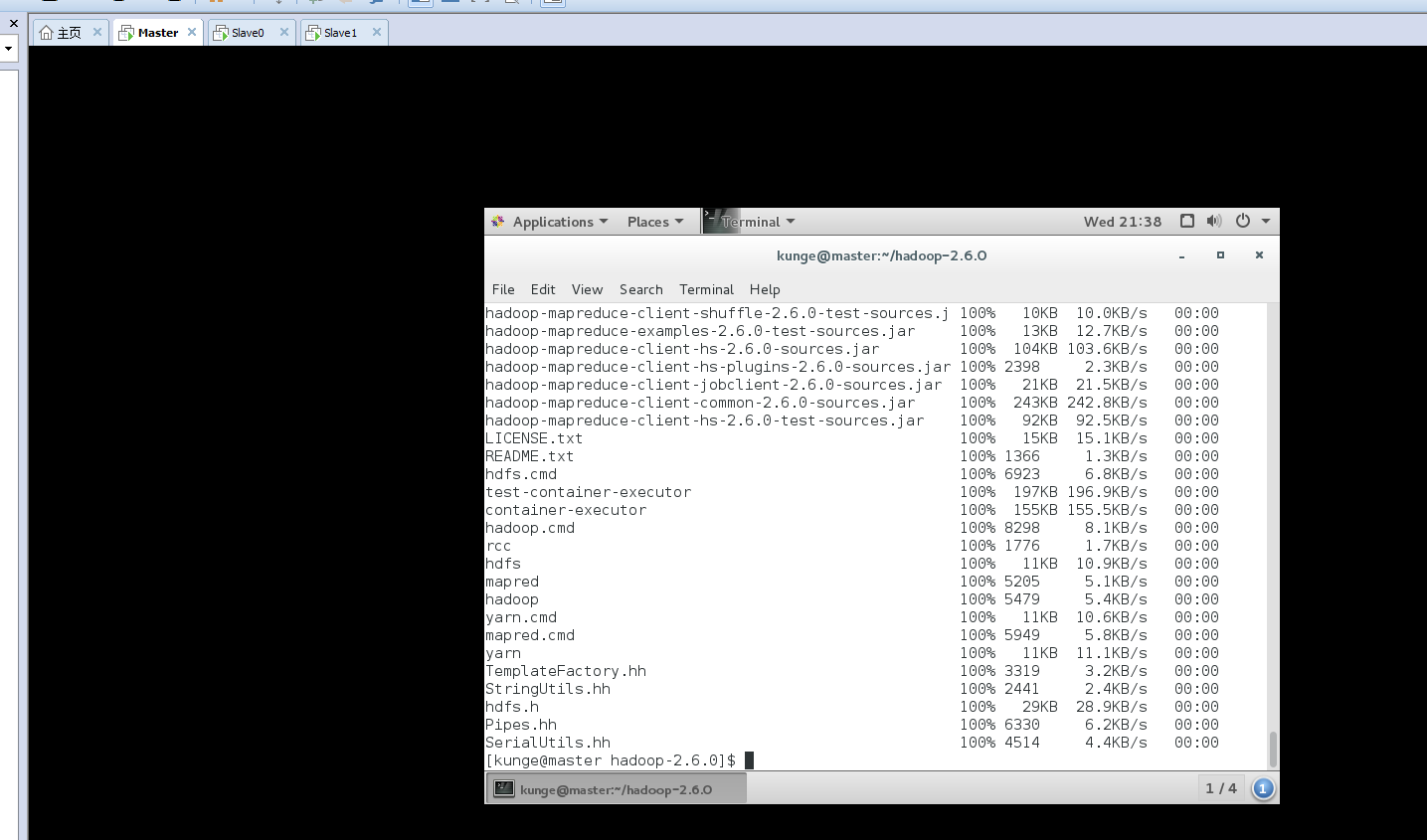
开始

1. 先输入 su kunge 切换回kunge用户
2. 生成密钥 ssh-keygen –t rsa
3. 会出现一系列的提示 直接回车键
4. 
5. 生成的密钥在.ssh目录下 可以通过cd ~/.ssh进入该目录 用ls –l命令查看
6. 接下来将公钥文件复制到.ssh目录下 cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys
7. 复制过来是为了便于修改该文件的权限 请用chmod 600 ~/.ssh/authorized\_keys进行修改
8. 
9. 修改后会变成这个样子
10. 最后将authorized\_keys文件复制到所有的slave节点 这里需要分别复制到slave0和slave1 两个节点上 命令 scp ~/.ssh/authorized\_keys kunge@slave0:~/ 和 scp ~/.ssh/authorized\_keys kunge@slave1:~/
11. 
12. 输入yes即可
13. 
14. 将公钥传出去

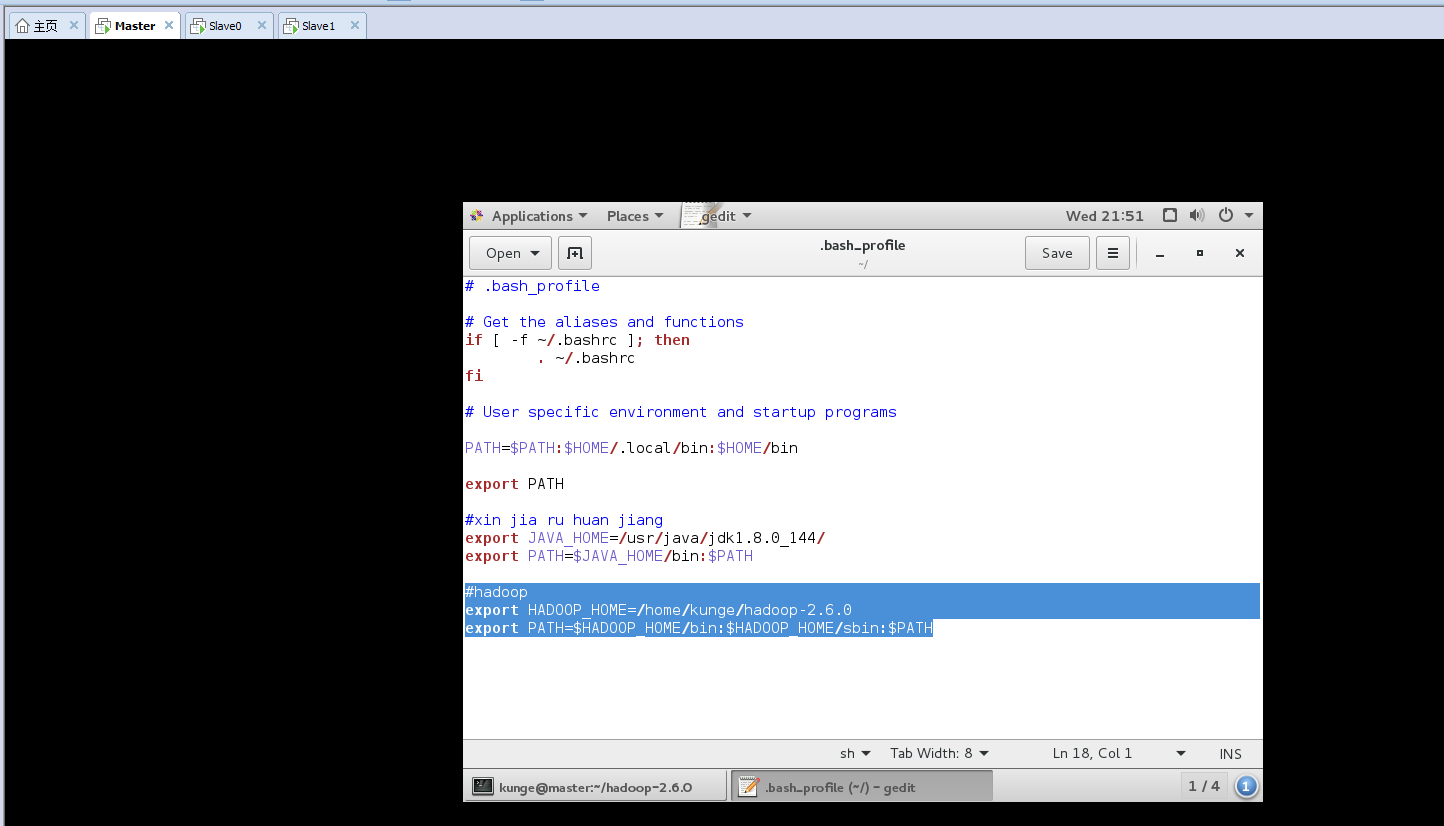
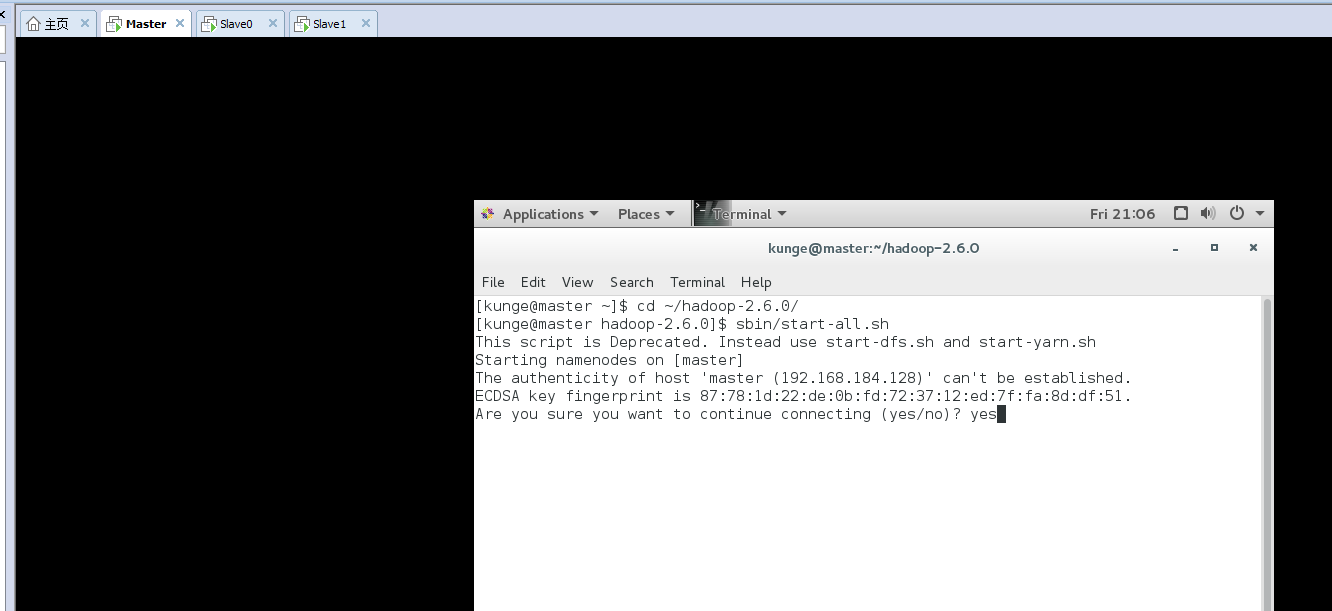
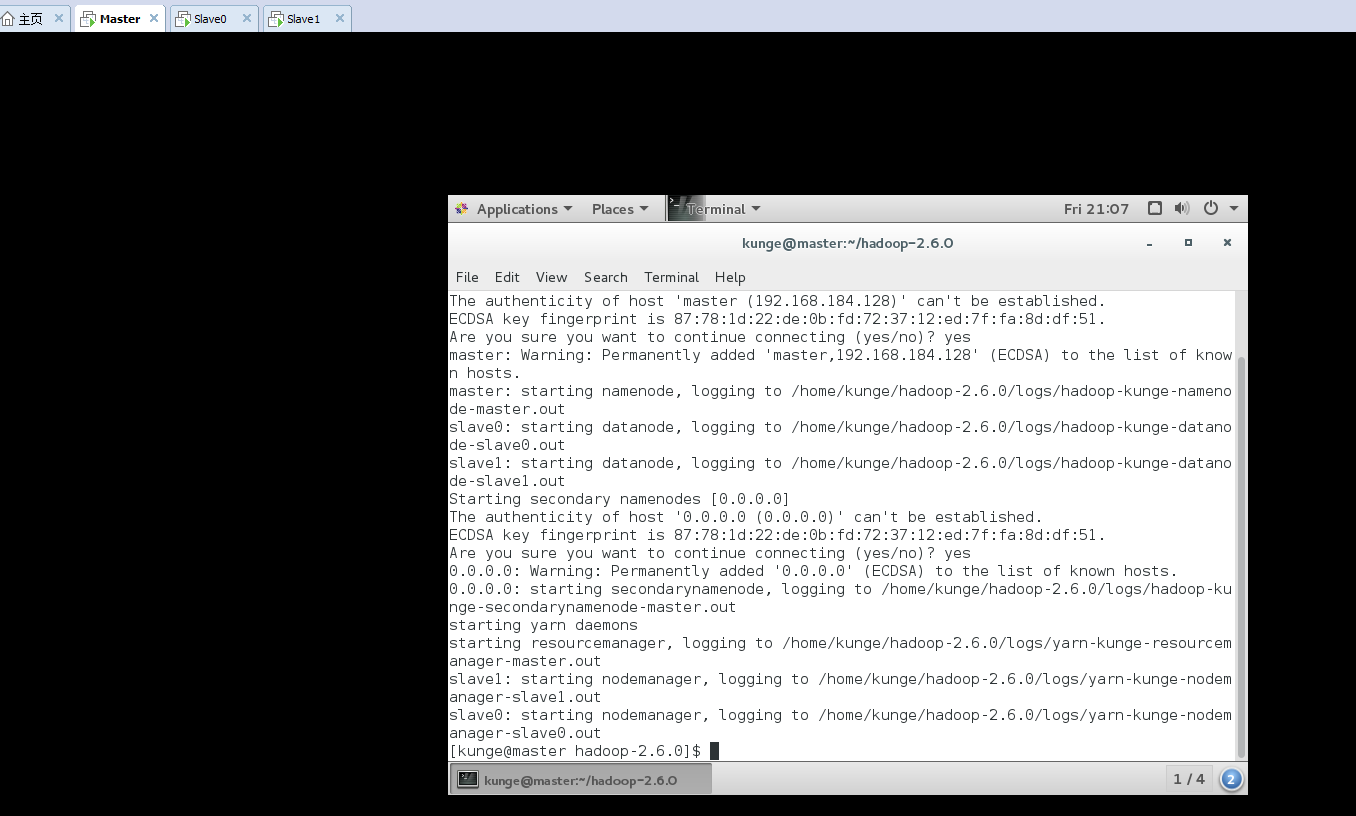
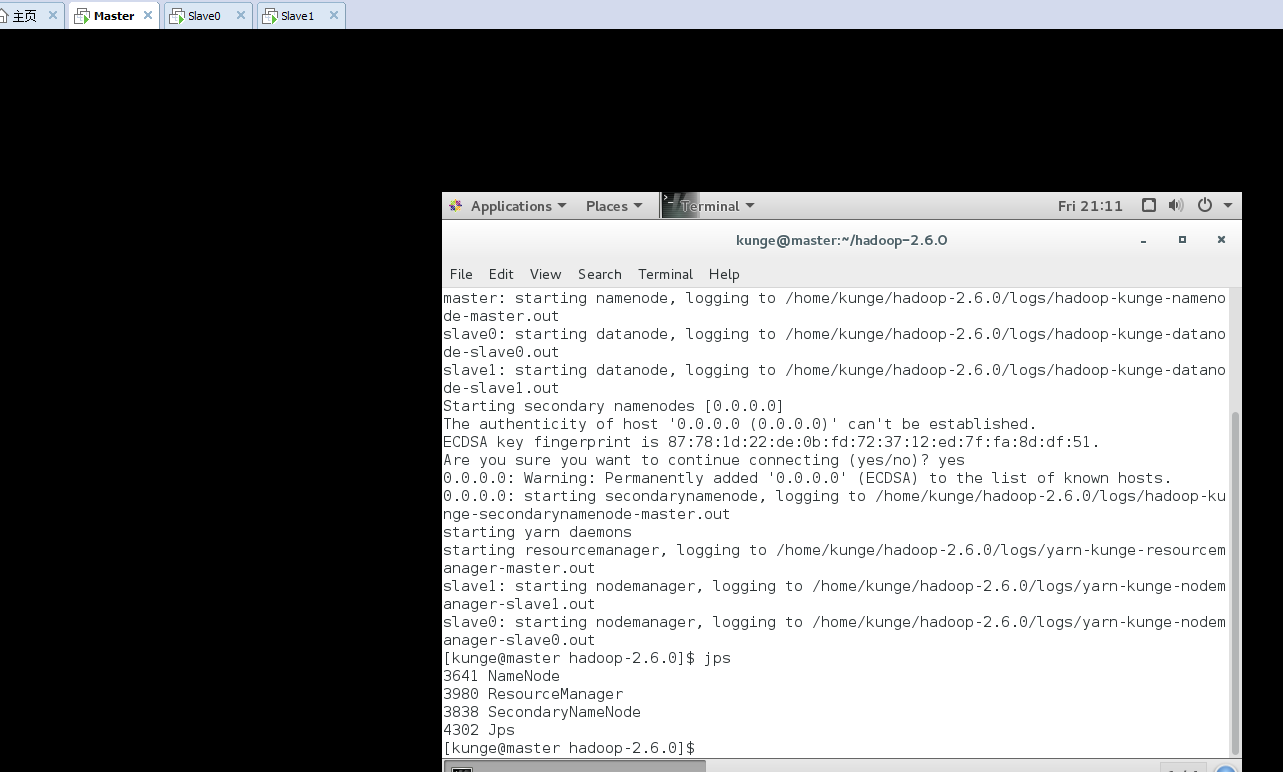
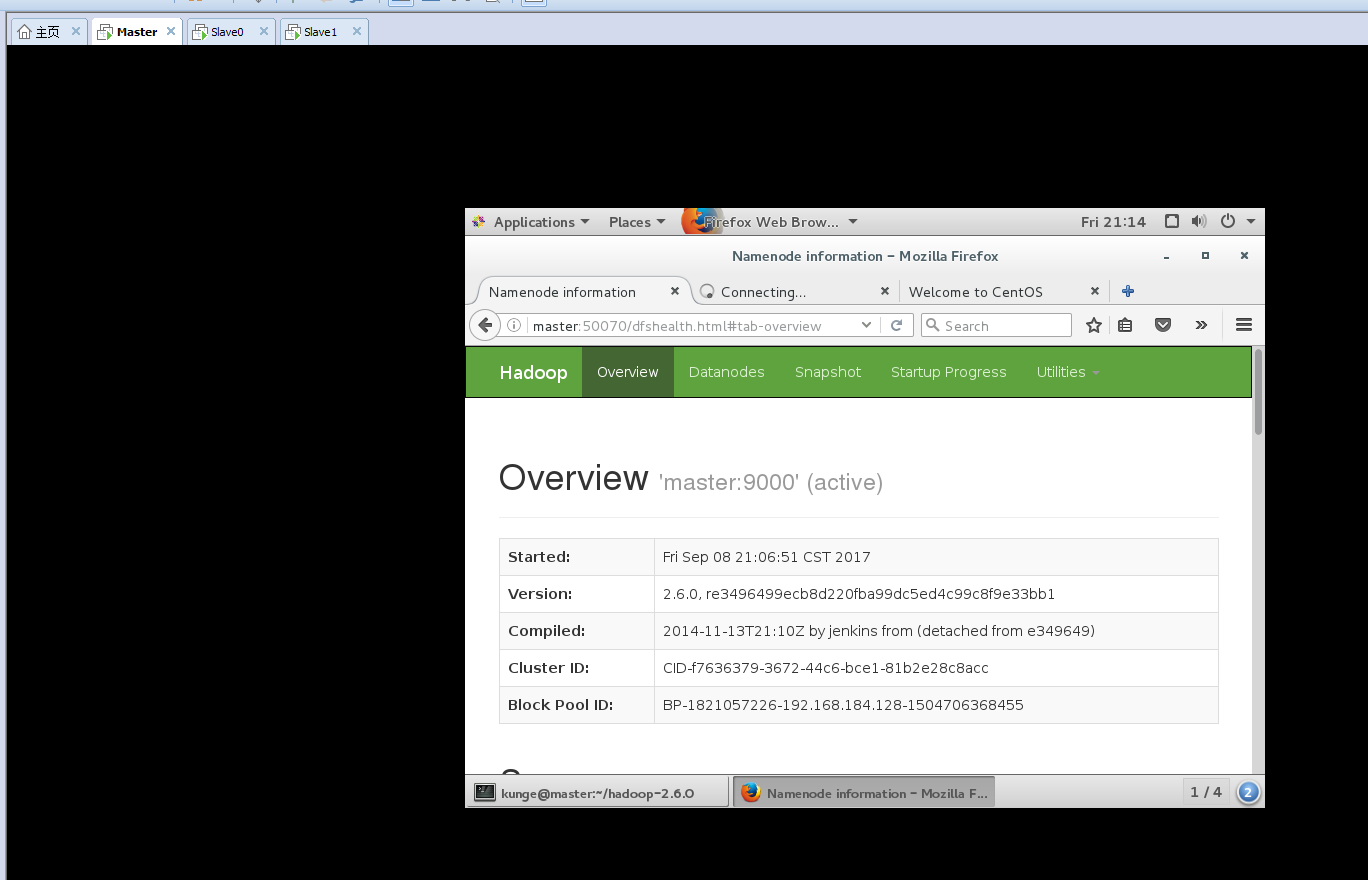
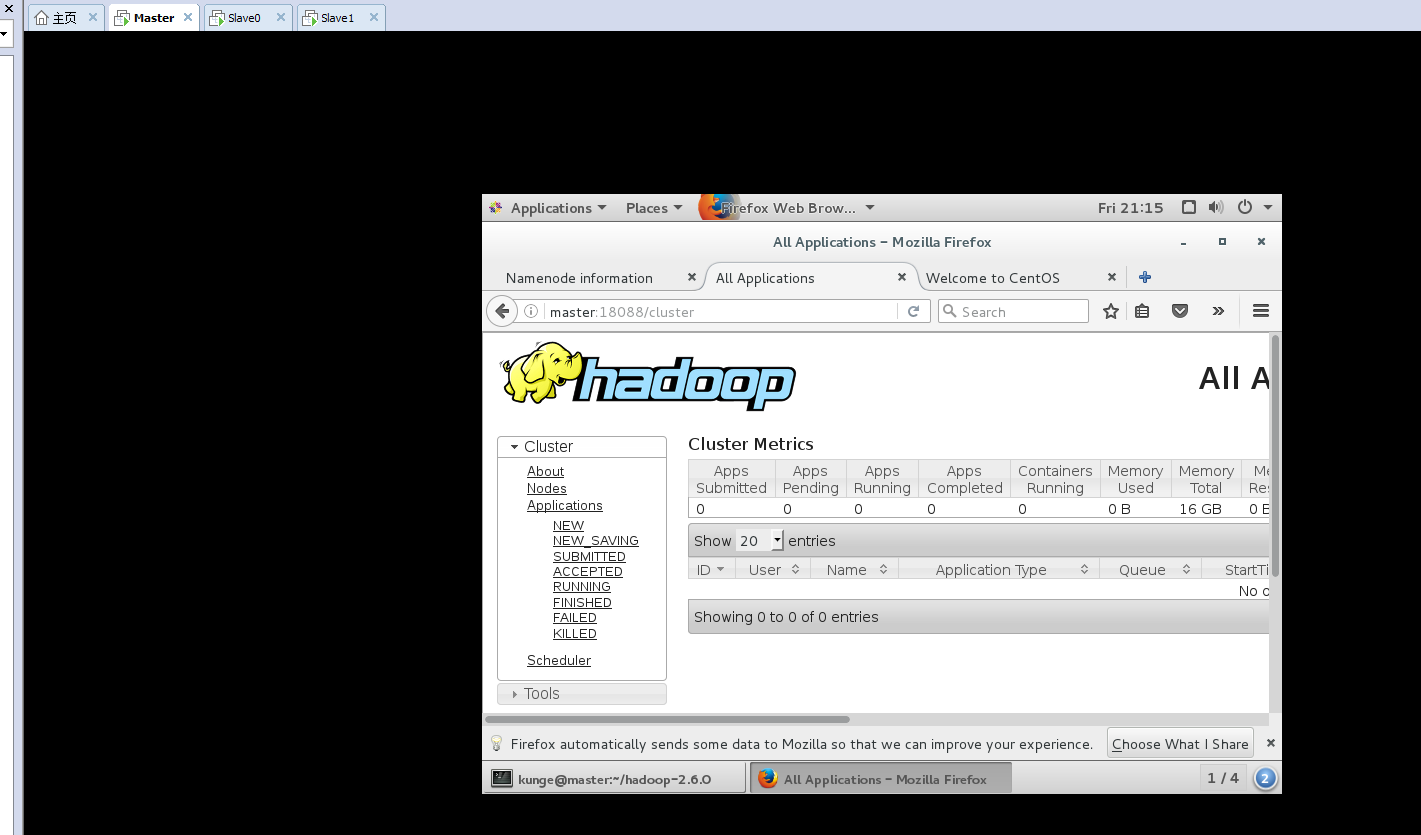
Slave节点配置

1. 进入slave0（节点）机器
2. 首先还是用ssh-keygen –t rsa生成密钥
3. 接着将authorized\_keys(这个是从master复制过来的) 移动到.ssh目录下 命令为 mv authorized\_keys ~/.ssh
4. 修改authorized\_keys文件的权限 用cd命令进入.ssh目录，在输入chmod 600 authorized\_keys修改权限
5. Over slave1(节点)机器按上述步骤再来一遍
6. 
7. 接下来用master机器执行ssh slave0 或者ssh slave1直接登陆并且免输入密码
8. 
9. 如果退出远程 输入exit即可

HADOOP HDFS安装配置

1. 每个节点（虚拟机）的安装和配置都一样，实际工作中，通常在master节点安装配置，然后复制安装目录到其他节点即可，注意，这里的操作是一般用户权限
2. 复制hadoop安装压缩包，到/home/kunge的Resources子目录下 利用xftp直接拖拽
3. 复制压缩包到~/目录下
4. 解压
5. 进入hadoop目录 查看列表
6. 
7. 配置hadoop环境变量 通过gedit /home/kunge/hadoop-2.6.0/etc/hadoop/hadoop-env.sh修改该文件
8. 找到export JAVA\_HOME=${JAVA\_HOME}修改成实际安装路径export JAVA\_HOME=/usr/java/jdk1.8.0\_144/
9. 配置Yarn环境变量 通过 gedit /home/kunge/hadoop-2.6.0/etc/hadoop/yarn-env.sh修改该文件
10. 找到#export JAVA\_HOME=${JAVA\_HOME}修改成实际安装路径export JAVA\_HOME=/usr/java/jdk1.8.0\_144/
11. 配置核心组件文件
12. 核心组件文件是core-site.xml 也位于~/hadoop-2.6.0/etc/hadoop 开始编辑
13. 
14. 在configuration目录下配置2个内容 第一个是默认文件系统的访问地址 第二个是hadoop临时数据目录
15. 配置文件系统
16. Hadoop文件系统的配置文件是hdfs-site.xml也位于~/hadoop-2.6.0/etc/hadoop 开始编辑
17. 
18. 在里面配置1个内容 意思为副本数，缺省值为3，改成1那么副本就是1，注意最大就是3，超过3没有意义
19. 配置yarn-site.xml (站点配置文件) 也位于~/hadoop-2.6.0/etc/hadoop 开始编辑
20. 
21. 5个配置分别是节点管理服务，资源管理地址，资源执行地址，资源追踪地址，资源管理员地址，web应用地址
22. 配置mapreduce计算框架文件 位于~/hadoop-2.6.0/etc/hadoop目录下，系统有一个mapred-site.xml.template文件，需要改个名，位置不变。
23. 命令cp /home/kunge/hadoop-2.6.0/etc/hadoop/mapred-site.xml.template /home/kunge/hadoop-2.6.0/etc/hadoop/mapred-site.xml
24. 
25. 配置一个 表示计算使用yarn
26. 配置master的slaves文件 该文件给出了Hadoop集群的slave节点列表，十分重要。因为启动hadoop，系统总是根据当前slaves文件的slave节点名称启动集群，不在列表中的slave节点不会视为计算节点 文件位于 ~/hadoop-2.6.0/etc/hadoop
27. 
28. 复制master上的hadoop到slavee节点
29. 命令scp –r /home/kunge/hadoop-2.6.0 kunge@slave0:~/
30. scp –r /home/kunge/hadoop-2.6.0 kunge@slave1:~/
31. 
32. 如上图所示
33. 至此完成hadoop集群安装和配置

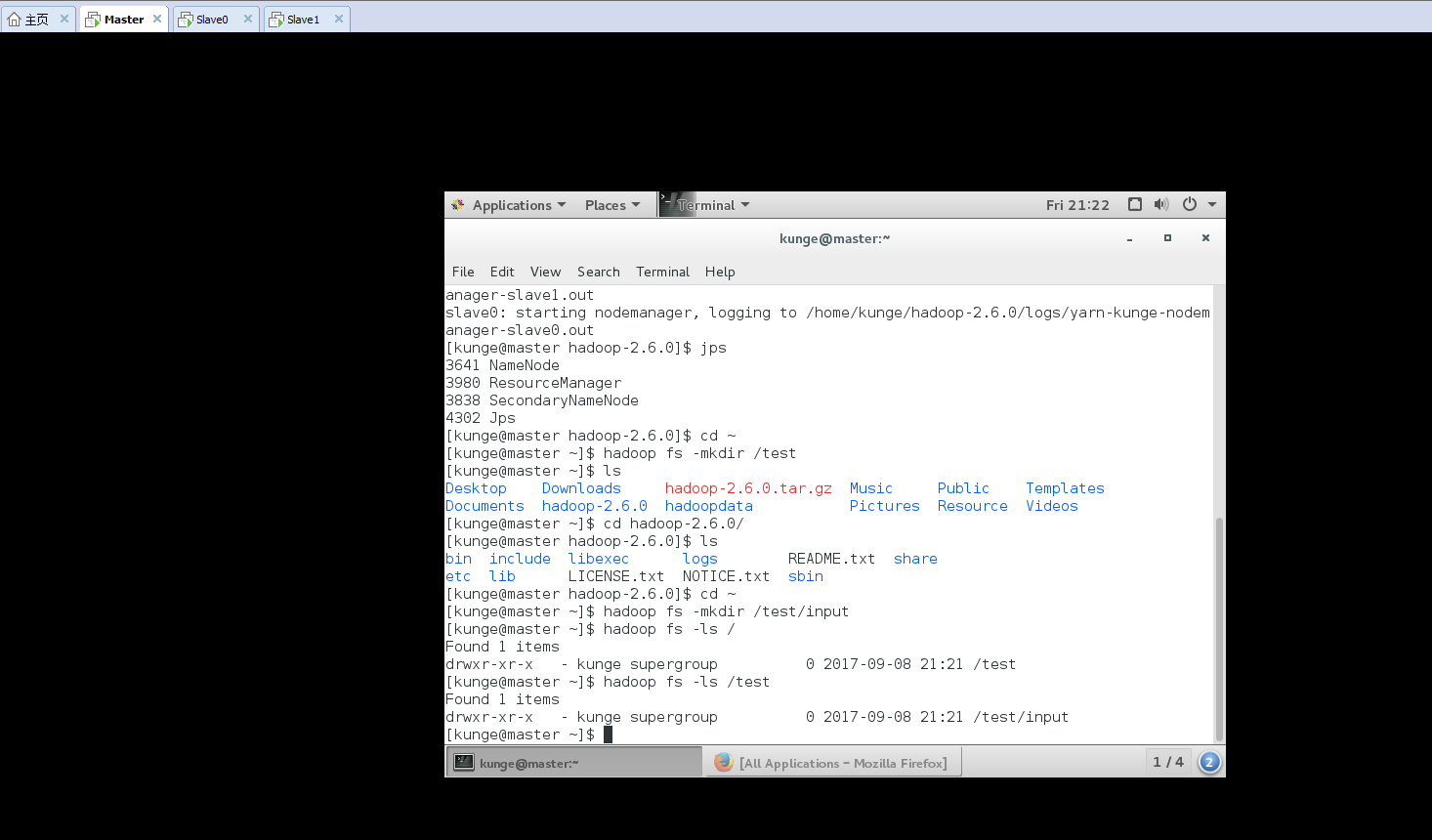
Hadoop集群的启动

1. 配置操作系统环境变量，类似jdk，注意master和2个slave节点都要配置
2. 
3. 创建hadoop数据目录
4. 和前面的core-site.xml配置是一致的 (所有节点都要配置) mkdir /home/kunge/hadoopdata
5. 格式化文件系统 只需要在master节点上进行 命令：hdfs namenode –format
6. 
7. 注意仔细看报错，尤其是配置文件，因为纯手打，很容易敲错。如果不是xml文件问题可以先删除dfs.name.dir指定目录，确保目录不存在，这样做防止错误将已经存在的集群格式化了
8. 启动hadoop 终于~
9. 进入hadoop目录 cd ~/hadoop-2.6.0
10. 启动 sbin/start-all.sh
11. 
12. 输入yes
13. 
14. 继续yes 但是有必要指出 下次启动hadoop 无需namenode初始化，只需要使用start-dfs.sh 然后接着使用start-yarn.sh启动yarn 实际上hadoop系统建议放弃使用start-all.sh(开启)和stop-all.sh(关闭) 而改用start-dfs.sh和start-yarn.sh
15. 验证hadoop是否启动成功
16. 终端执行jps命令查看hadoop启动成功 在master节点上，执行jps后如果显示结果是四个进程：secondaryNameNode,ResourceManager,JPS和namenode，表示主节点（master）启动成功
17. 
18. 如果jps命令不管用显示（commond not found）,那么就是jdk安装或者系统环境变量配错
19. 启动火狐浏览器输入<http://master:50070>看到hadoop监视页面
20. 
21. 输入<http://master:18088>能看到yarn运行情况
22. 

Hadoop集群的基本应用

1. hdfs基本命令

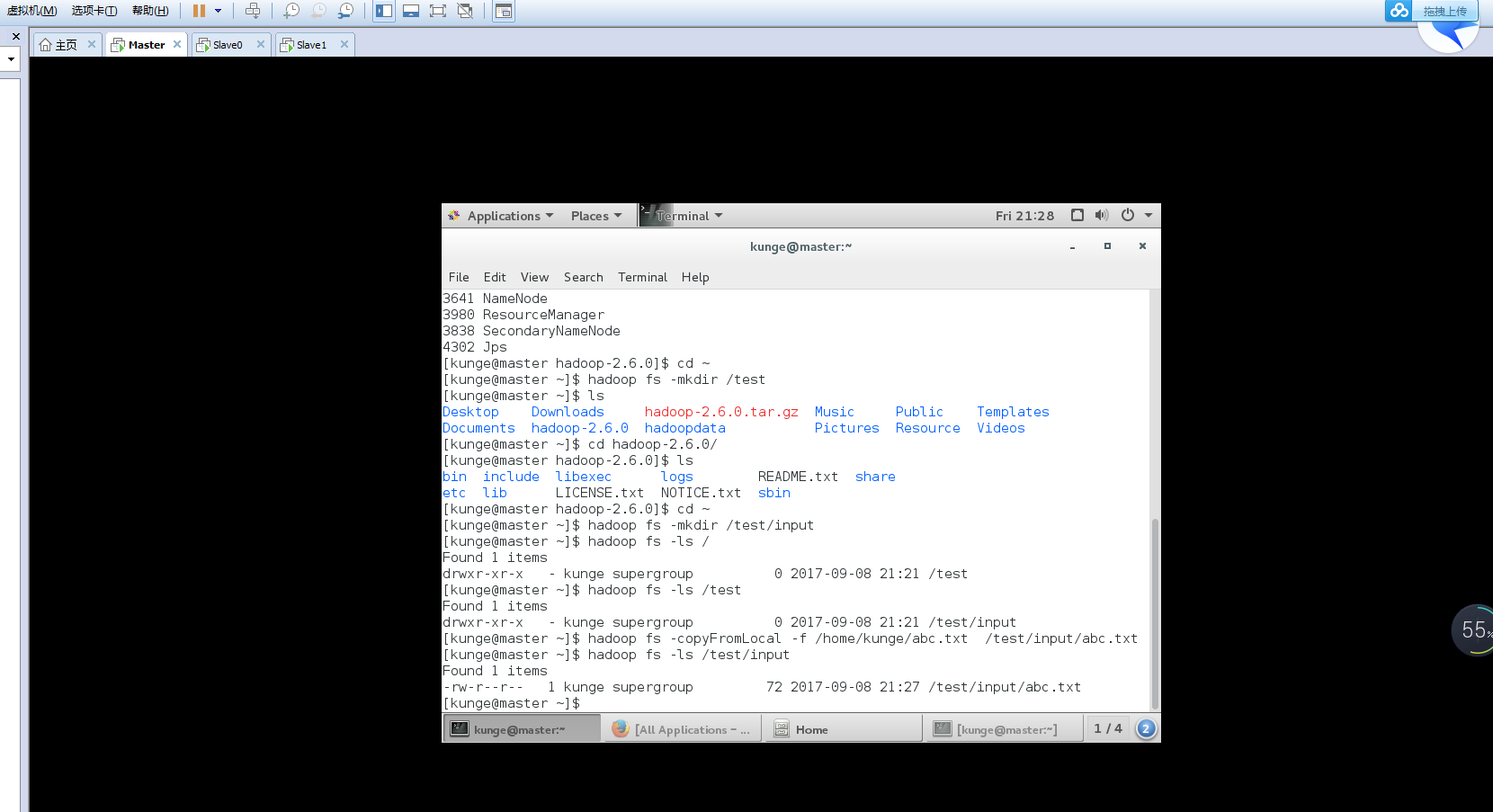
a.创建目录 –mkdir hadoop fs –mkdir /test



实现创建根目录再创建子目录

b.上传文件到HDFS

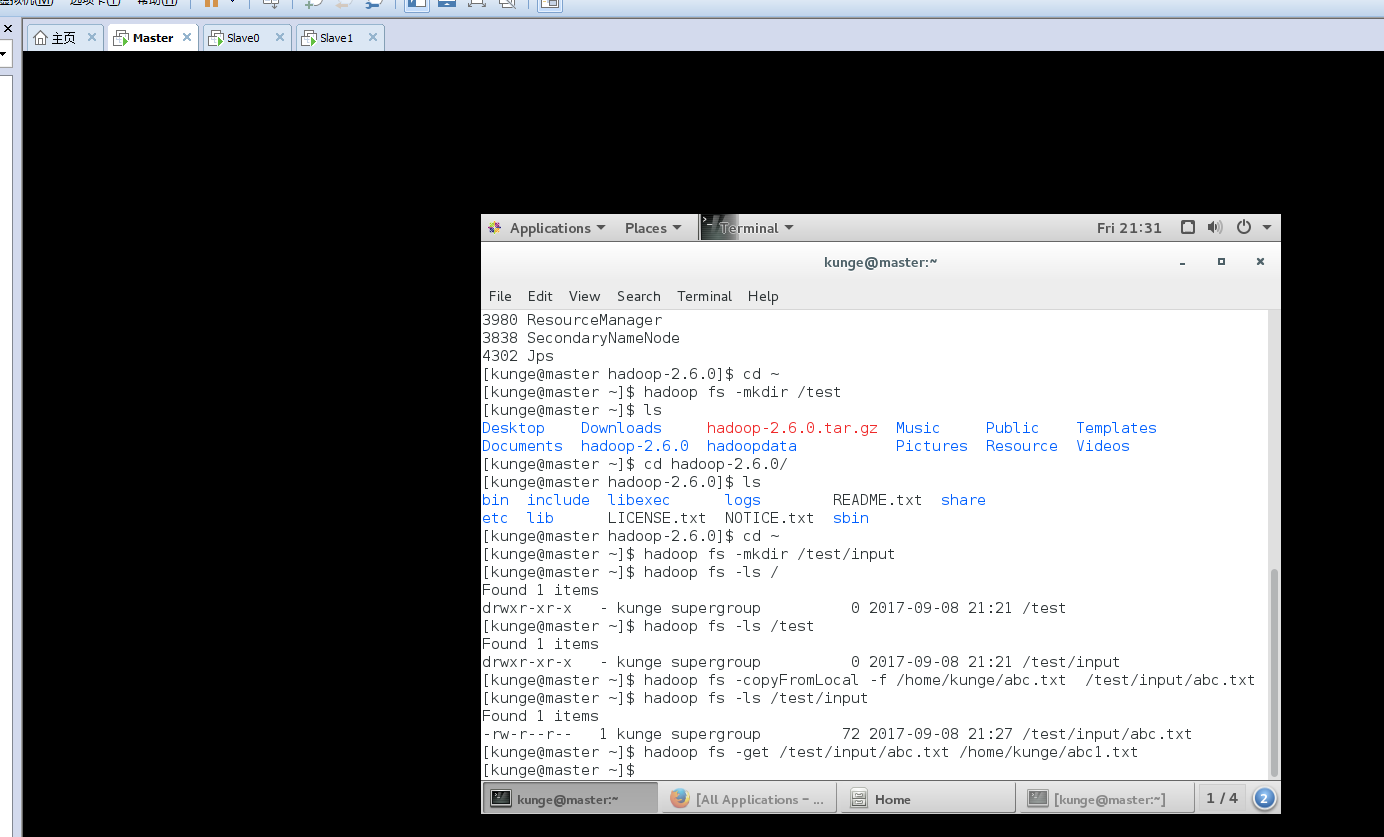
两个命令可用1. Hadoop fs –put和hadoop fs –copyFromLocal



文件从本地拷入hadoop环境

c.下载文件到本地

hadoop fs –get /test/input/abc.txt /home/kunge/abc1.txt



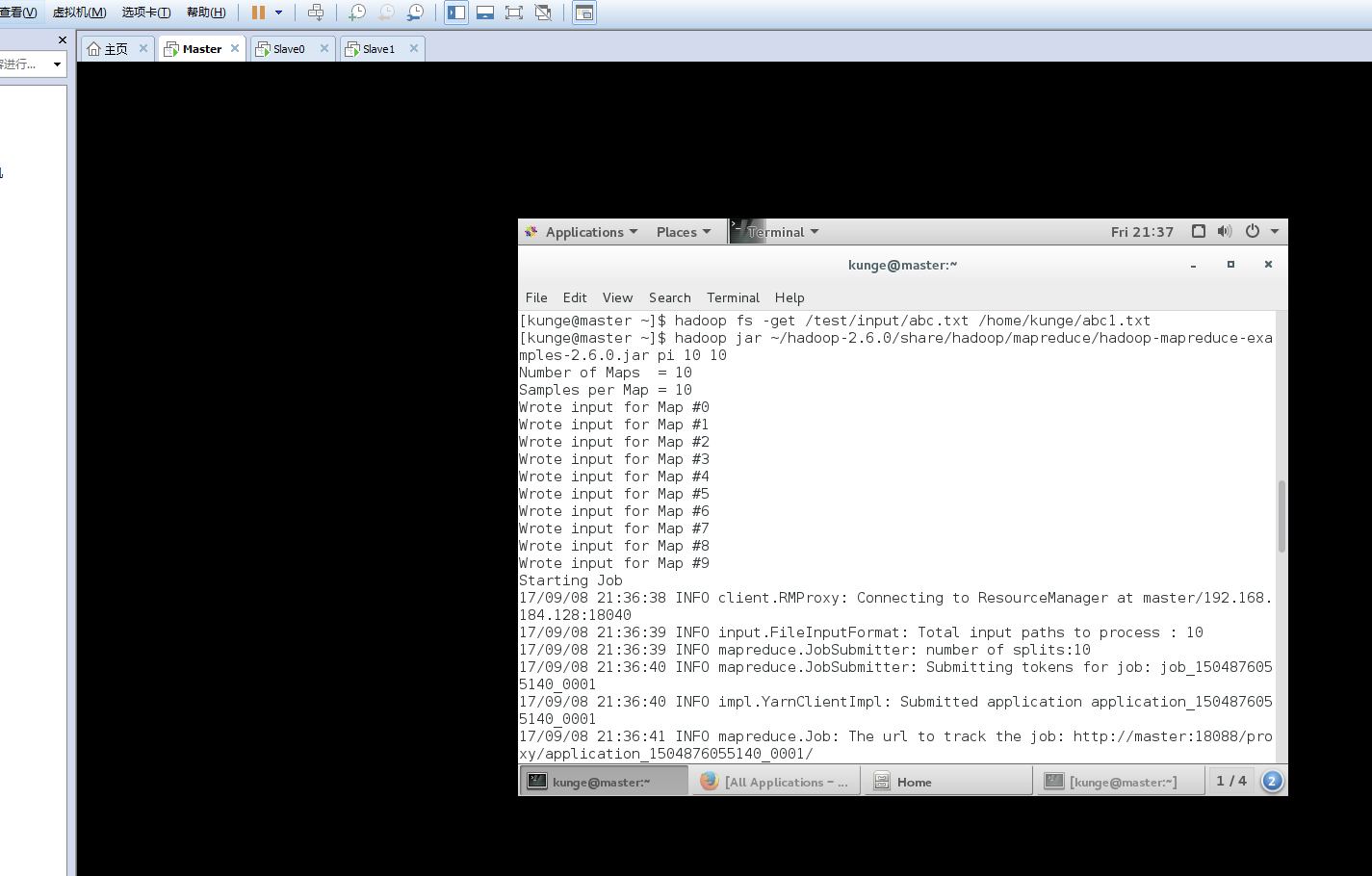
2.在hadoop集群中运行程序

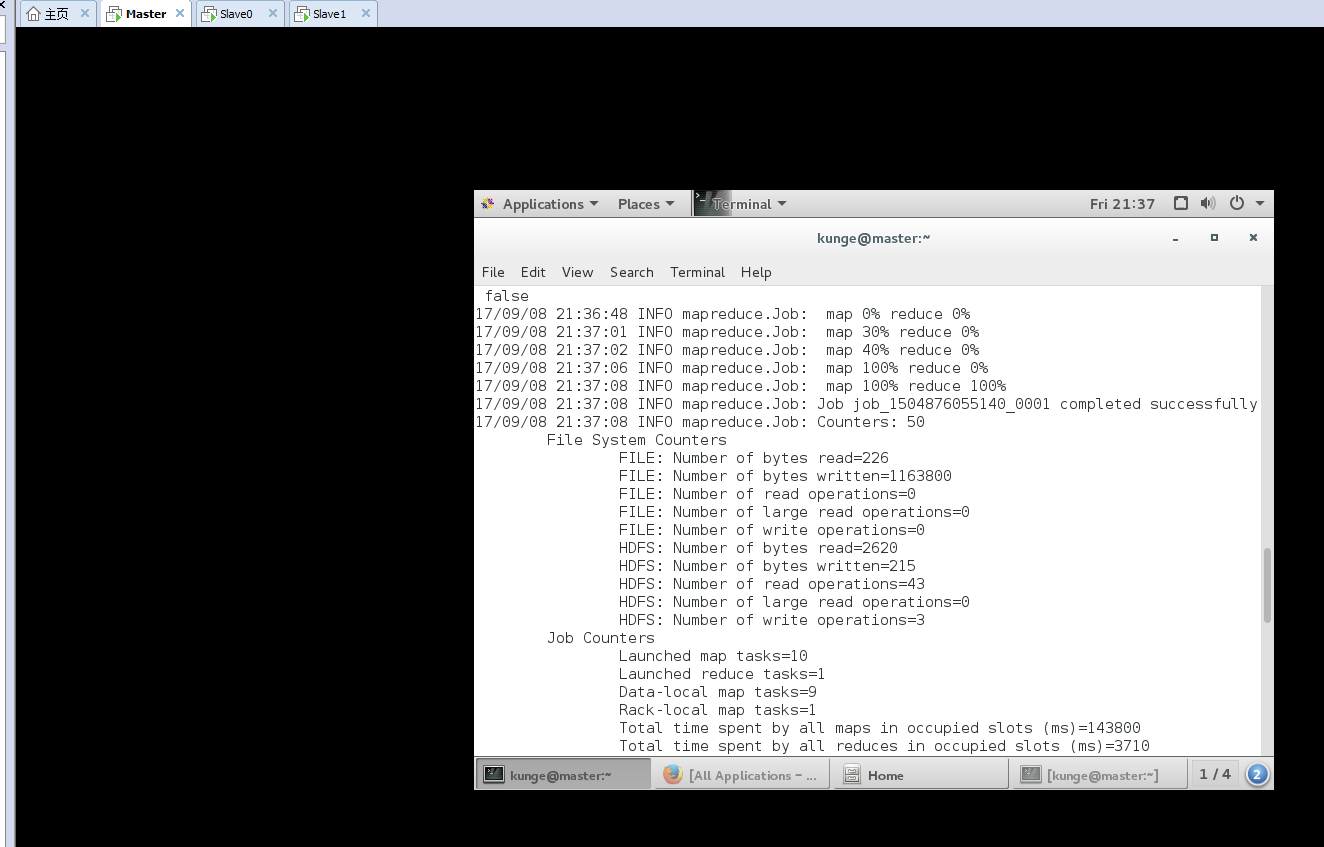
系统已经提供一些MapReduce的示例程序，一个典型用户计算圆周率的pi的java程序包，直接玩

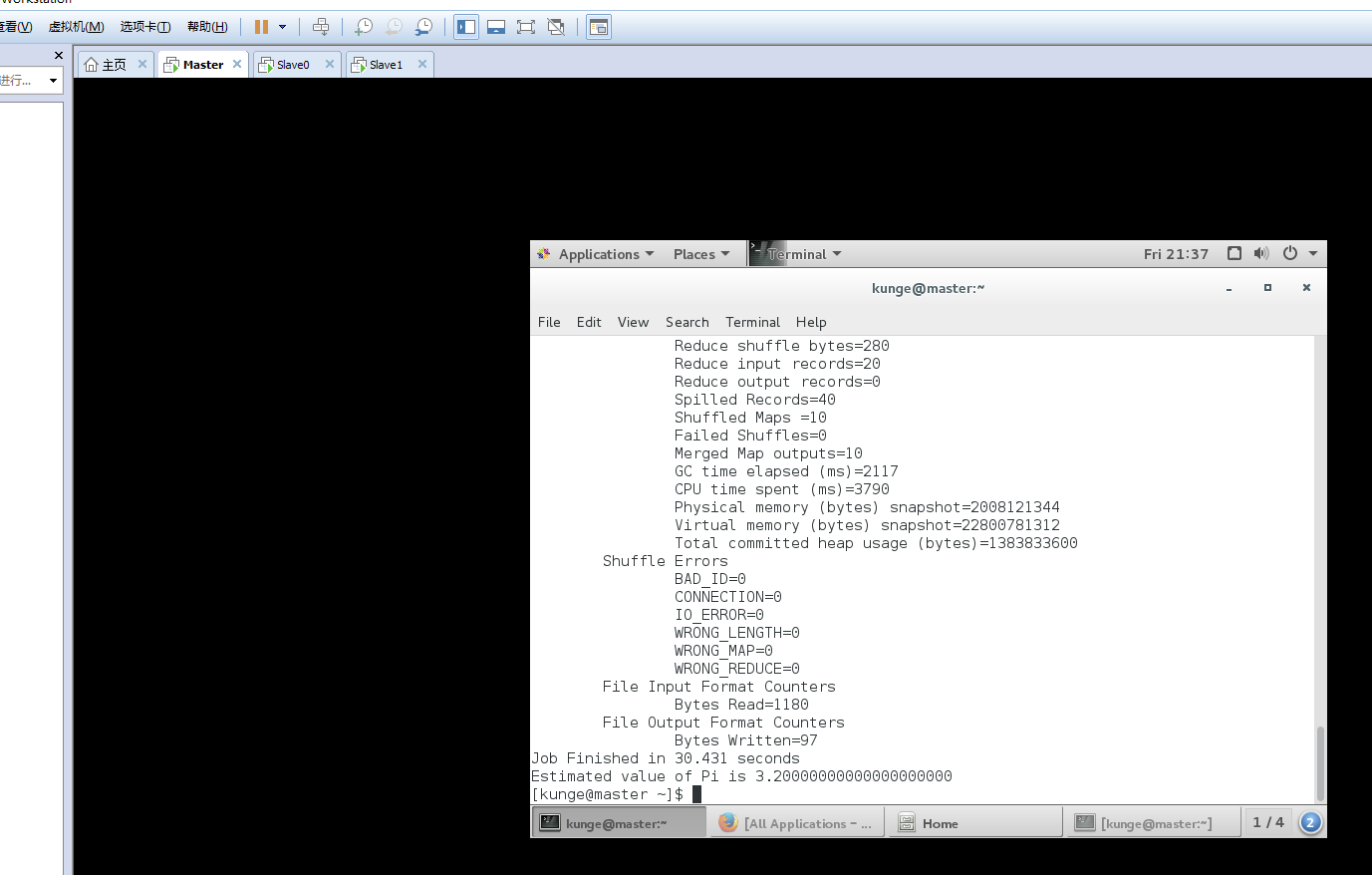
包在~/hadoop-2.6.0/share/hadoop mapreduce/hadoop-mapreduce-examples-2.6.0.jar

运行命令：hadoop jar ~/hadoop-2.6.0/share/hadoop mapreduce/hadoop-mapreduce-examples-2.6.0.jar pi 10 10

pi是类名 10表示map次数 第二个10表示随机生成点的次数（与计算原理有关）







最后结果近似3.2 这个只是一个测试程序 验证hadoop工作

MapReduce概念

以WordCount举例

A文件有Hello World Bye World 4个字符串

B文件有 Hello World Bye World 4个字符串

1.首先先拆分数据 形成一组key/value数据

A文件分割成(0,Hello World)

(12,Bye World) 0和12表示偏移量就是包括回车符的字符数

B文件同上

2.接下来执行Map方法将上面的key/value 再生成全新的key/value

A文件

(0,Hello World) （Hello,1）

(12,Bye World) 变成 (World,1)

(Bye,1)

(World,1)

B文件同上

3.接下来排序与合并处理

（Hello,1） （Bye,1） （Bye,1）

(World,1) map端排序 （Hello,1） Combine 合并 （Hello,1）

(Bye,1) (World,1) (World,2)

(World,1) (World,1)

4.Reduce阶段的排序和合并

（Bye,1） （Bye,list(1,1)）

（Hello,1） （Bye,2）

(World,2) (Hello,list(2))

Reduce端排序 Reduce() (Hello,2)

（Bye,1） (World,list(1,1))

（Hello,1） (World,4)

(World,2) (World,list(2))

上图list(1,1)表示两个未合并的相同元素 list(2)表示已经合并的相同元素

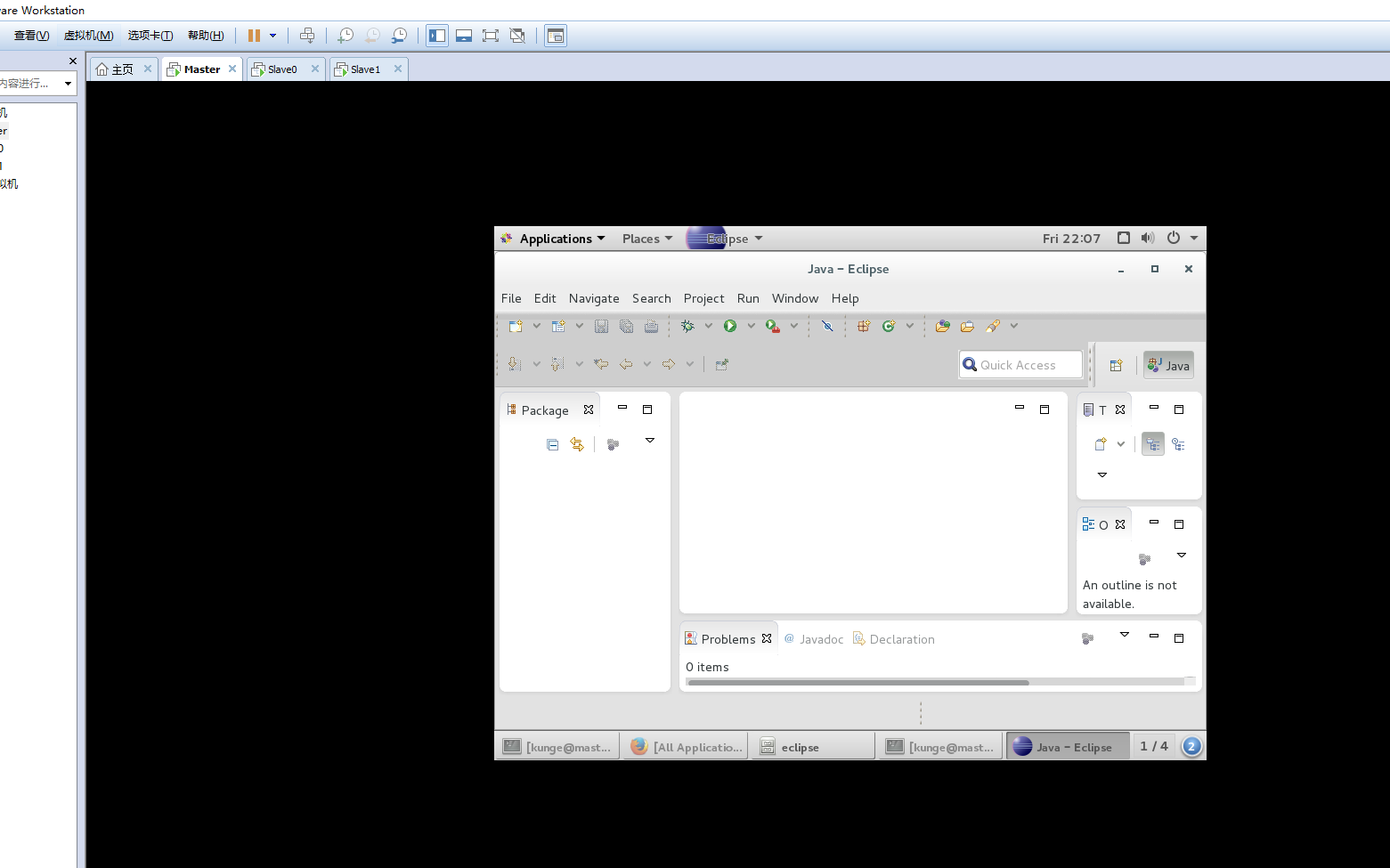
原因在于MapReduce内部管理机制但是最有Reduce()能进行最后的合并

Yarn工作流程

1. 客户端提交MapReduce作业
2. Yarn资源管理器负责协调集群上计算资源的分配
3. Yarn节点管理器负责启动和监视集群中机器上的计算容器
4. 应用程序Master负责协调运行MapReduce作业和任务，它和MapReduce任务在容器中运行，这些容器由资源管理器分配对节点管理器进行管理
5. 分布式文件系统（HDFS）用来与其他尸体见共享作业文件

linux安装eclipse

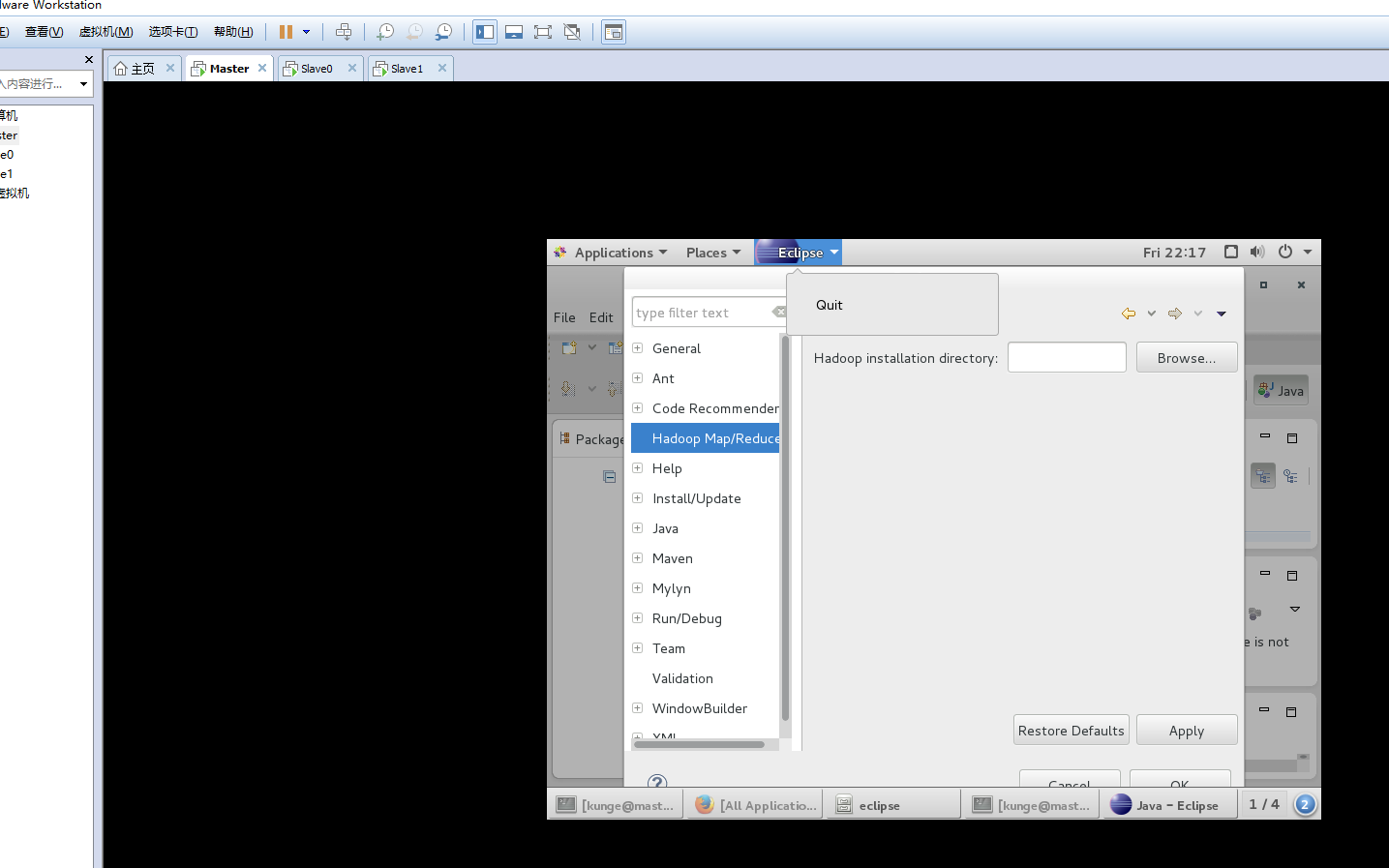
将eclipse上传到/home/kunge/eclipse文件夹中 解压，打开



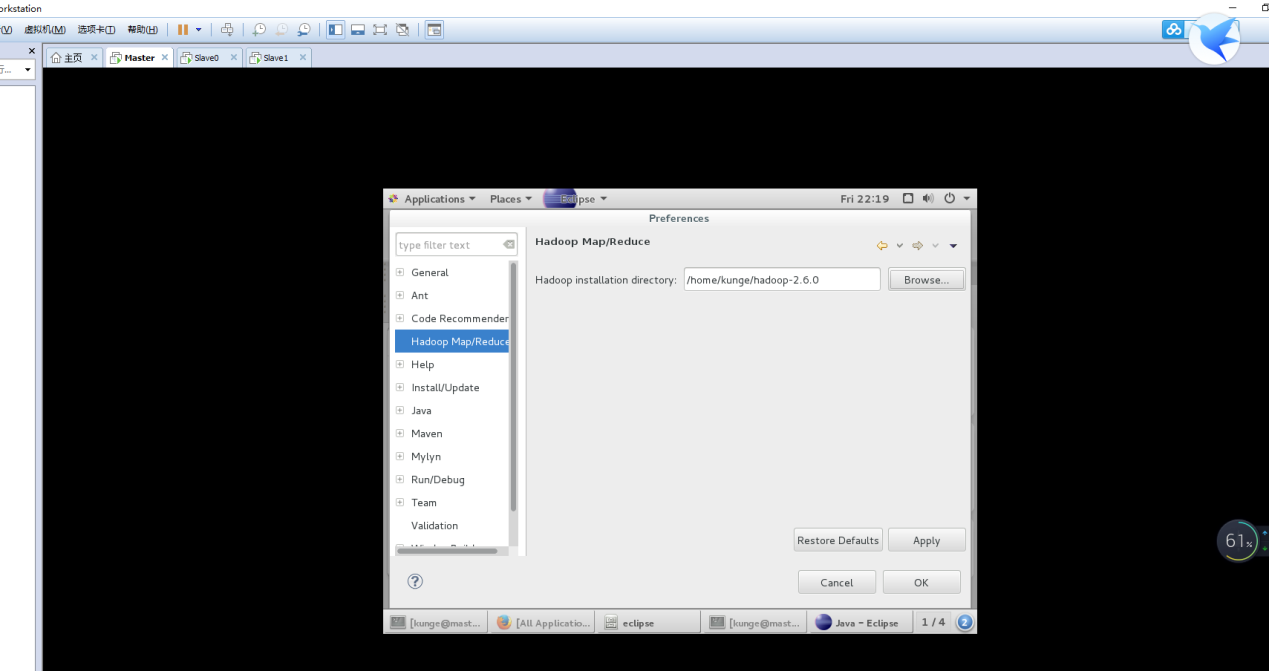
安装hadoop插件

将hadoop-eclipse-2.6.0.jar复制到eclipse/plugin目录下

打开eclipse, window->preferences找到下图

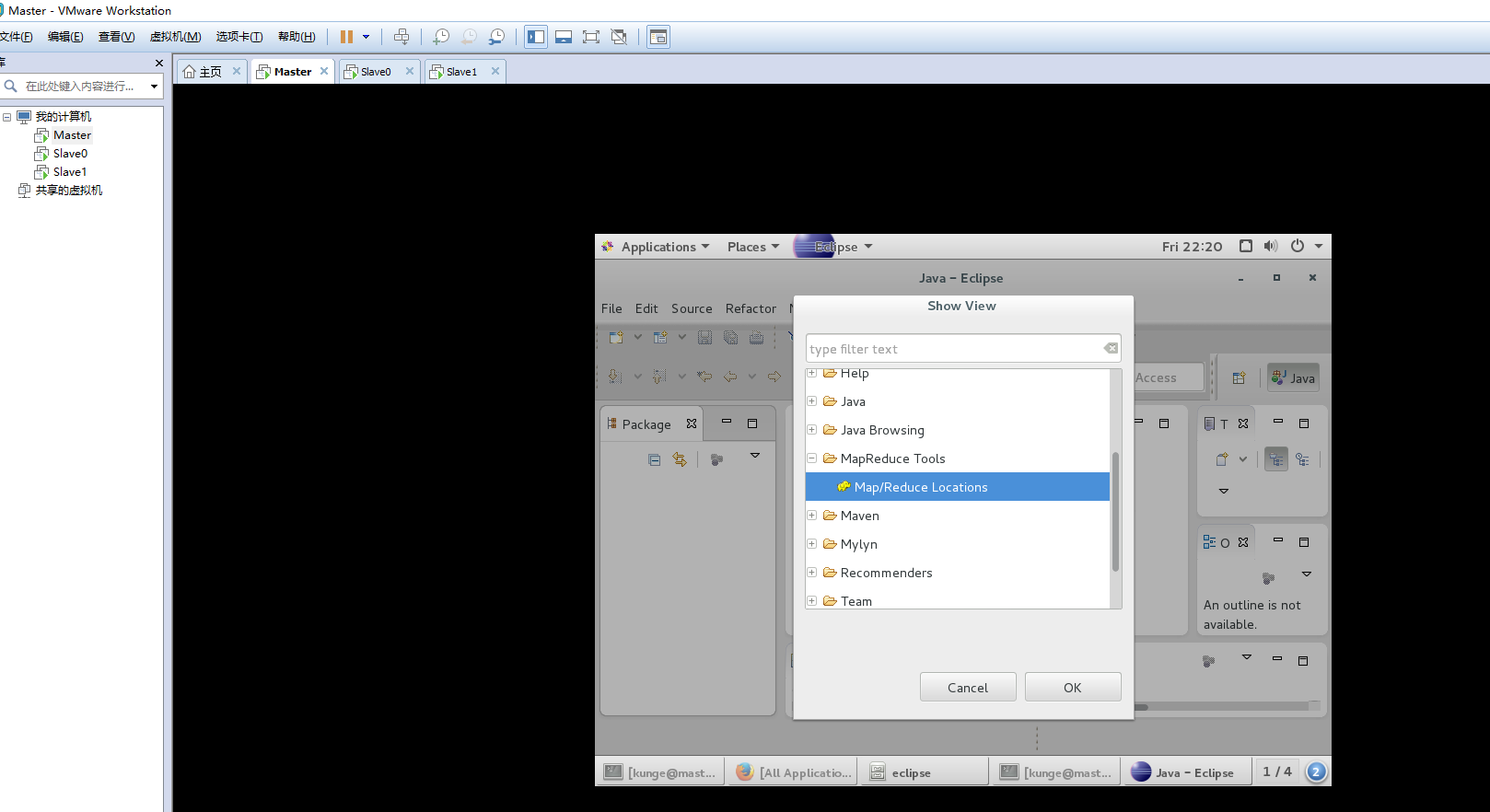


目录就是hadoop的根目录

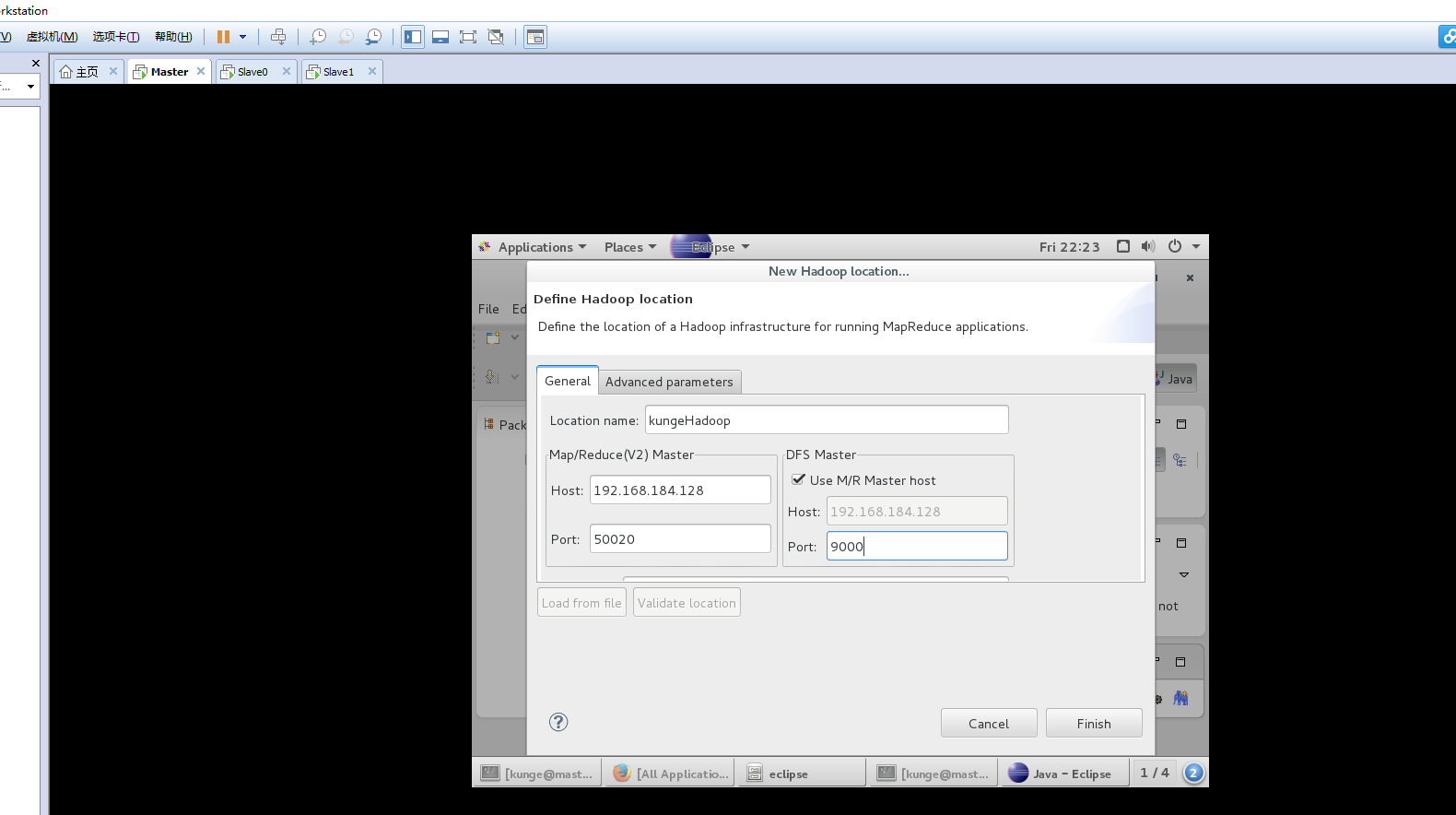


创建配置Map/Reduce Location

Window->showview->other找到下图所示

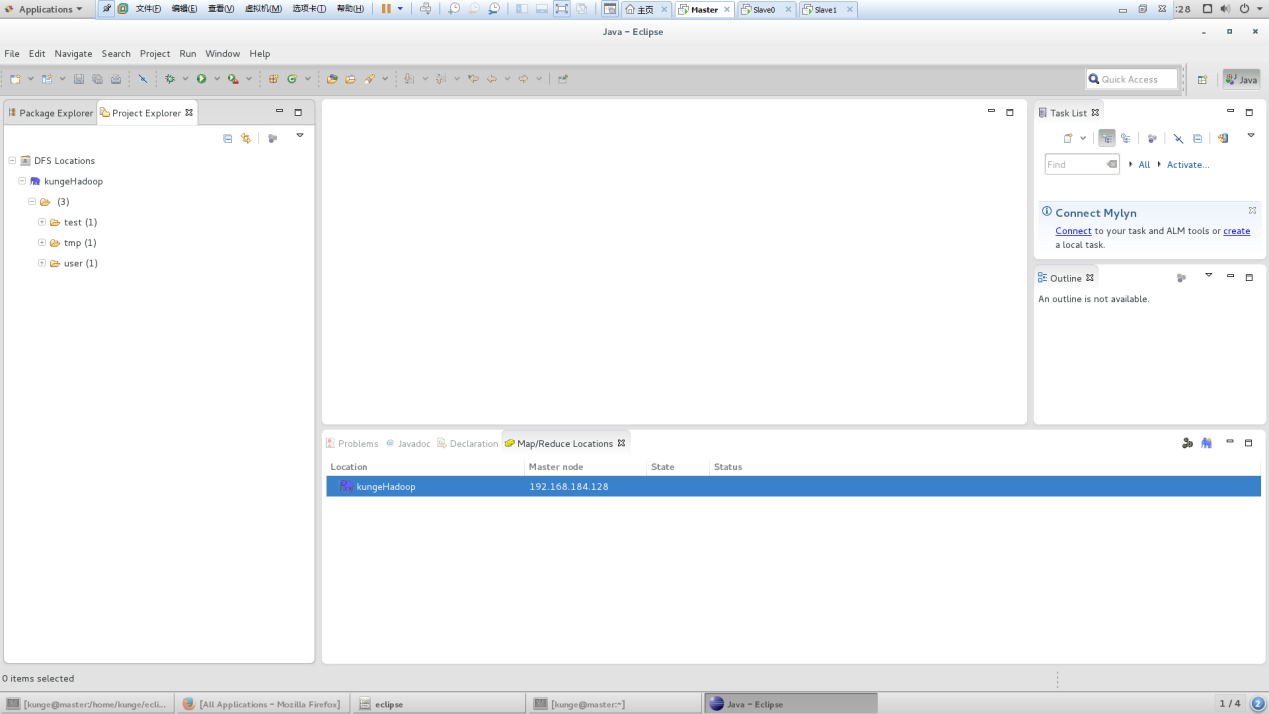


在下方的Map/Reduce Location窗口内 新建一个配置



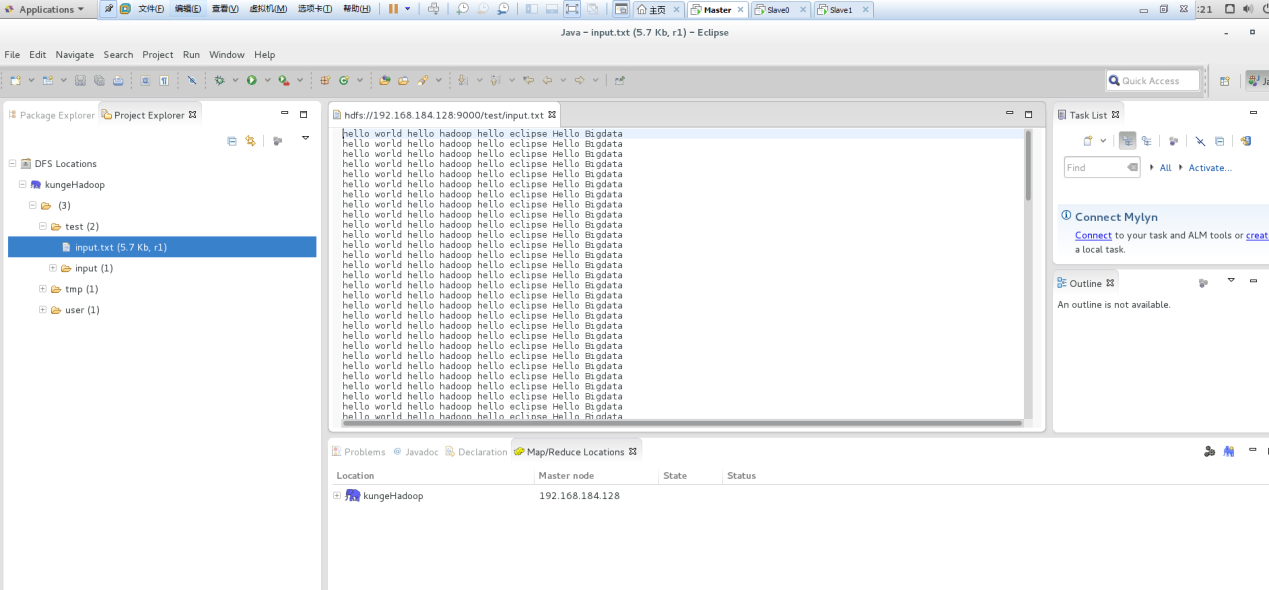
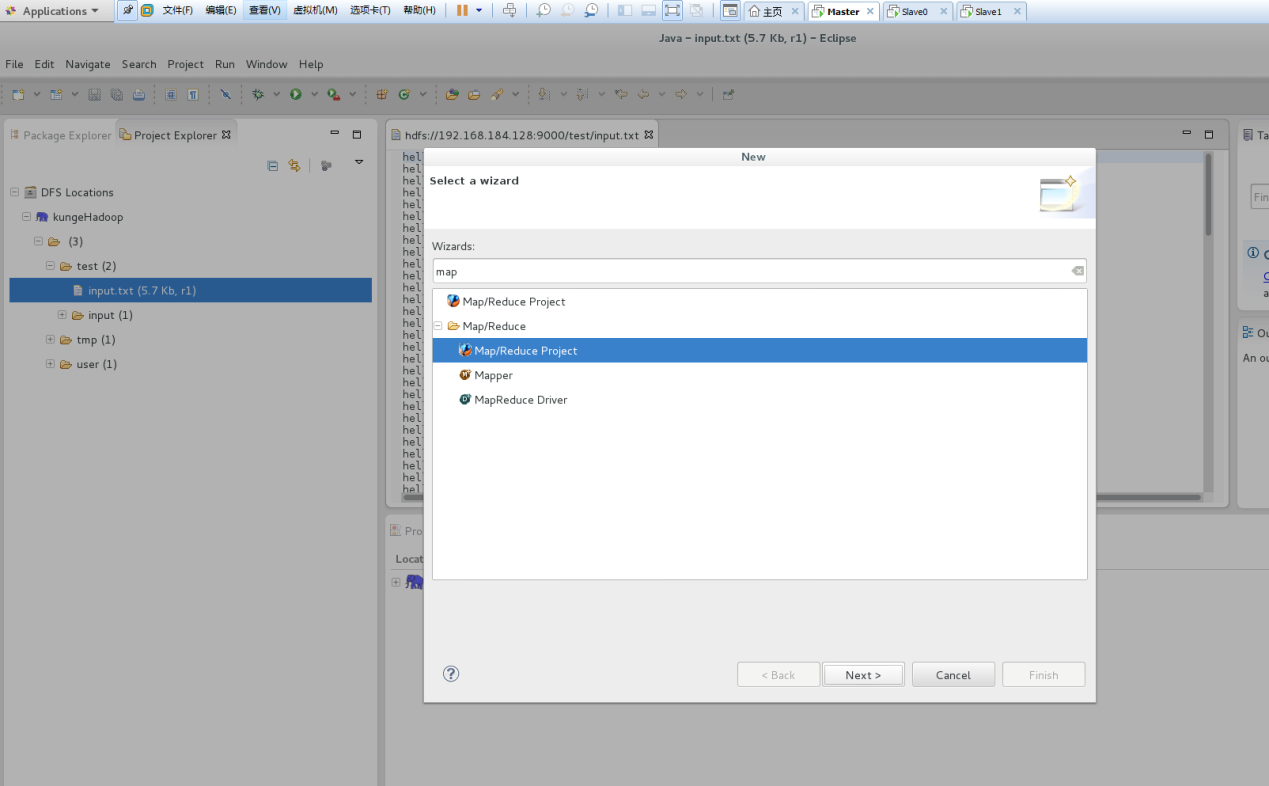
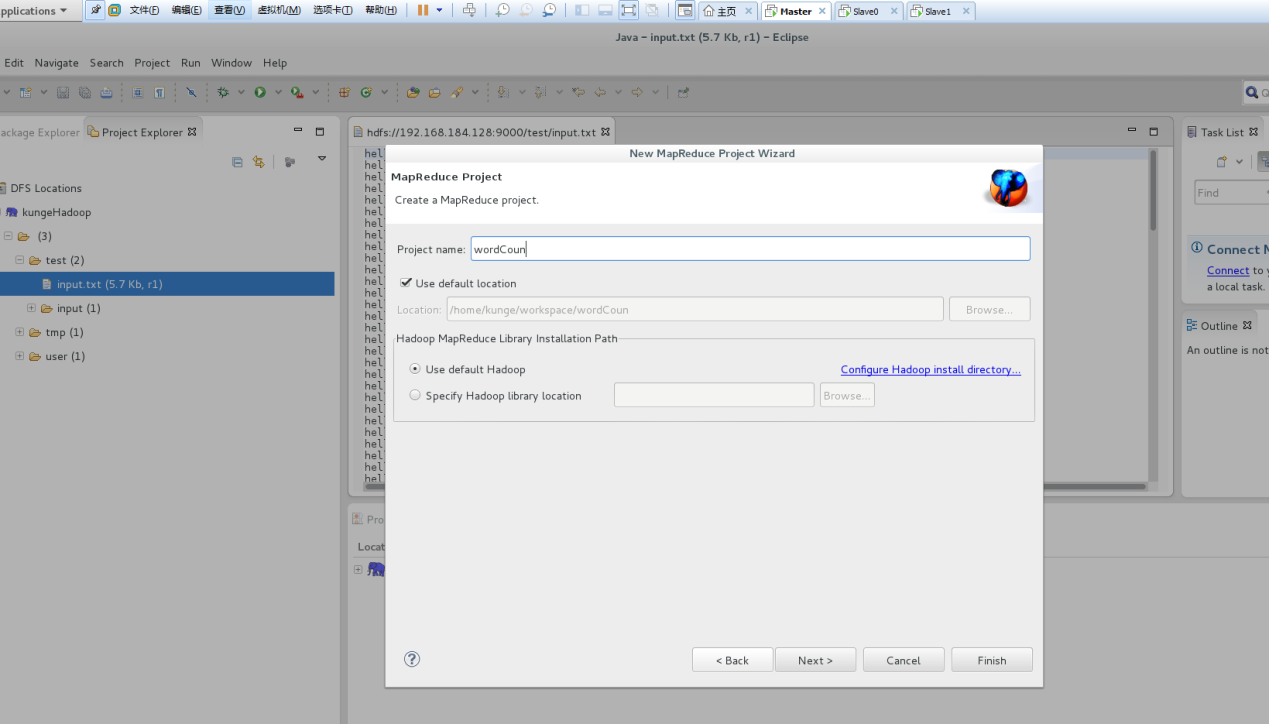
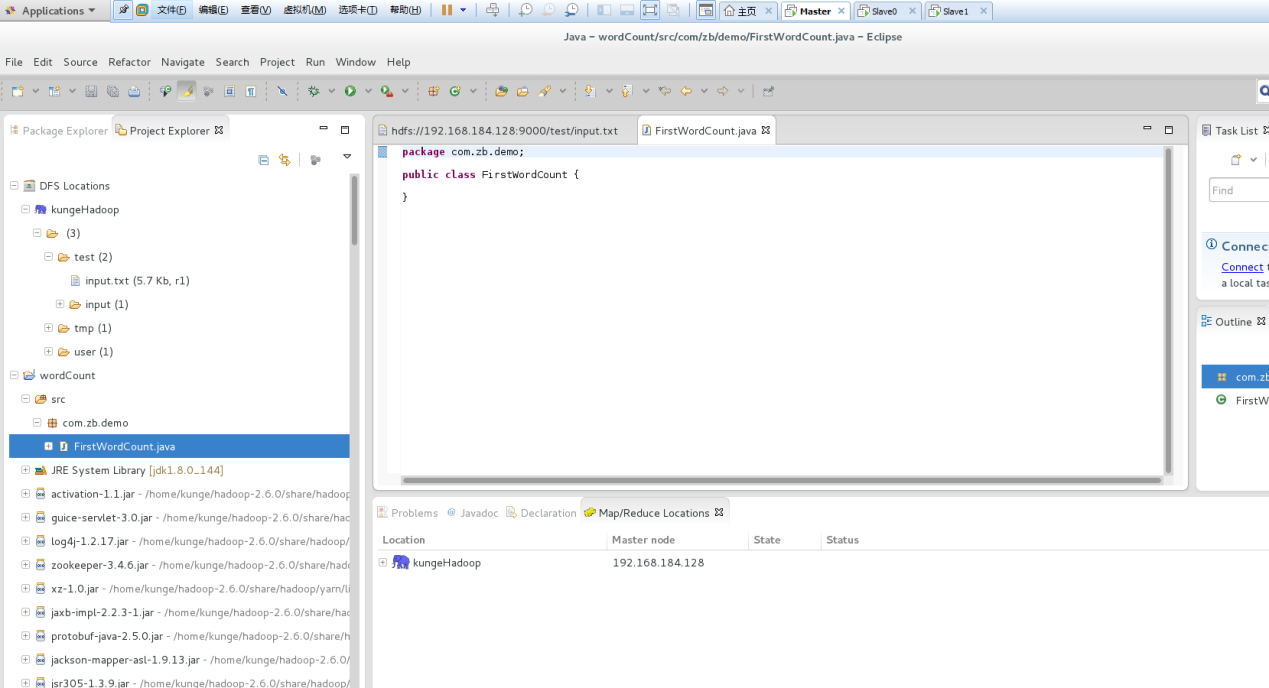
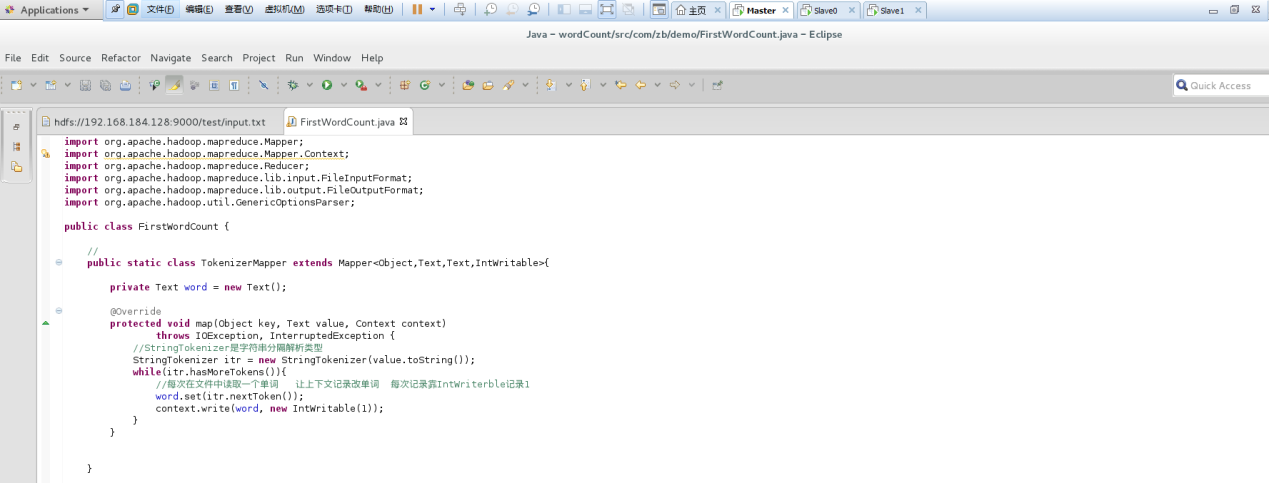
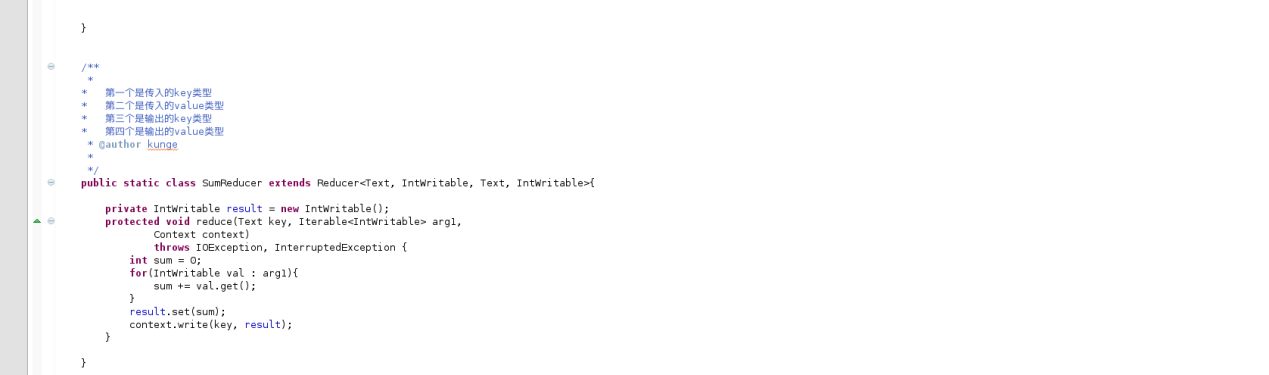
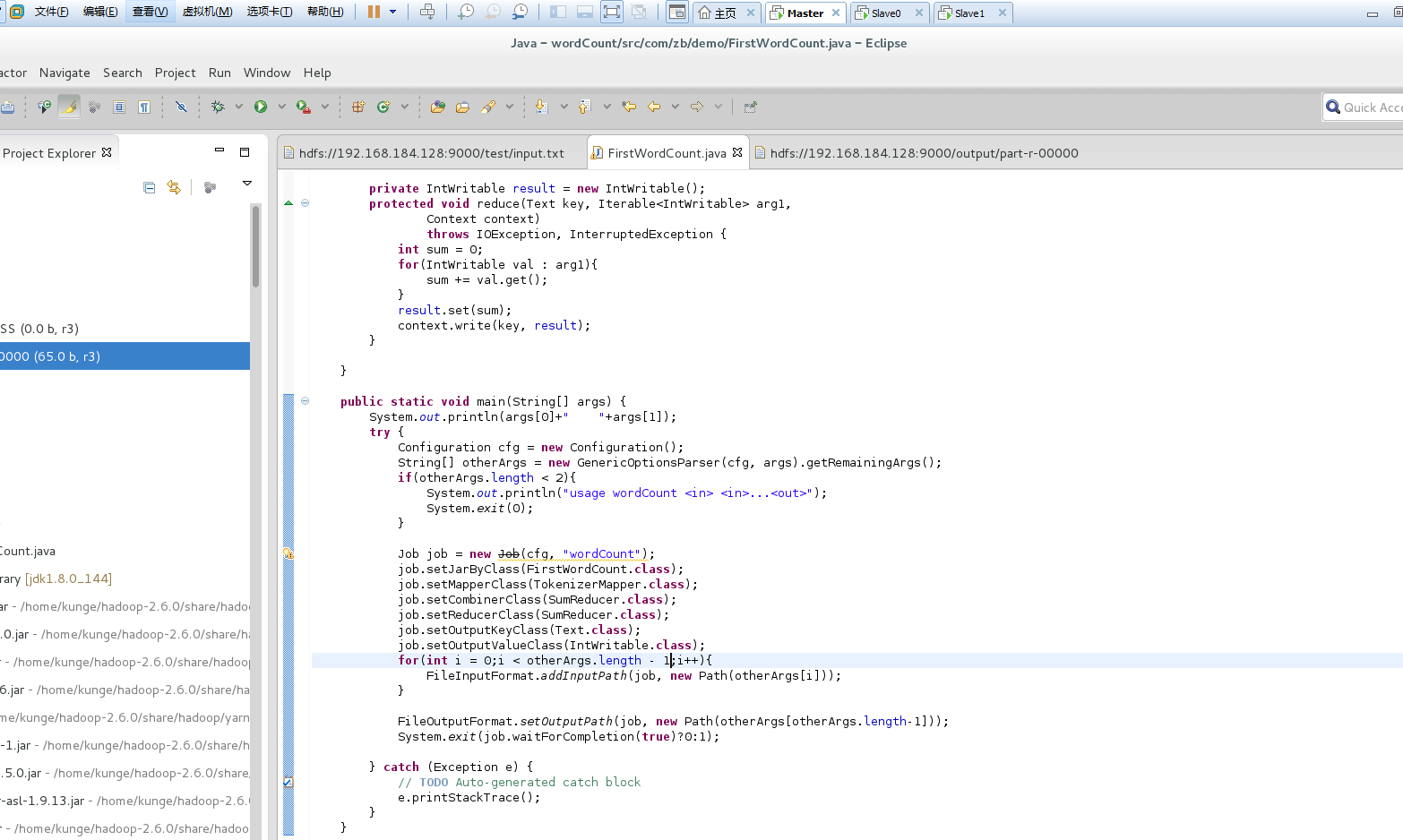
最好把地址换成ip，端口号默认50020

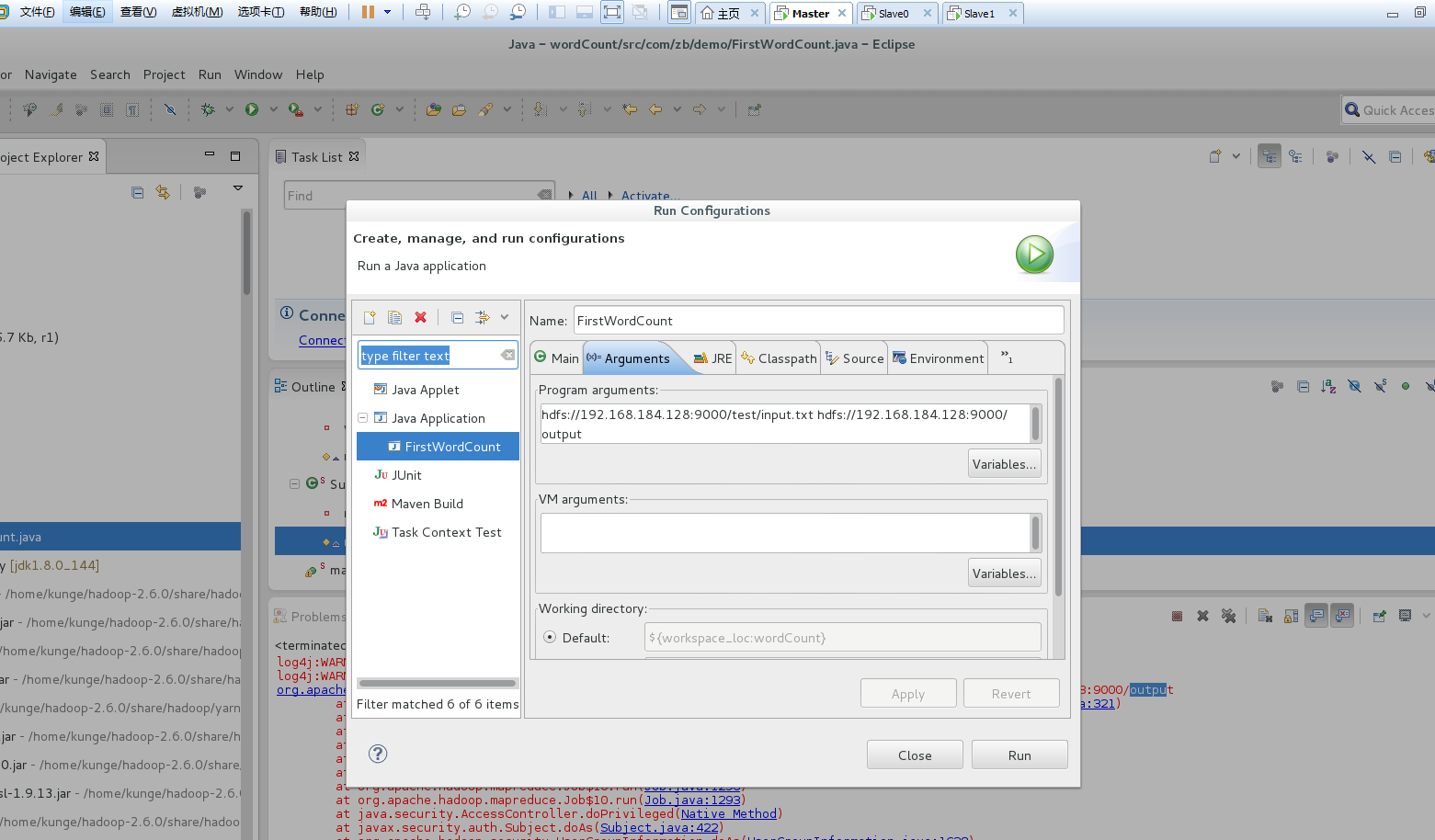
DFS Master的host和port在核心文件core-site.xml中提取，主机名自动锁定，端口改成9000



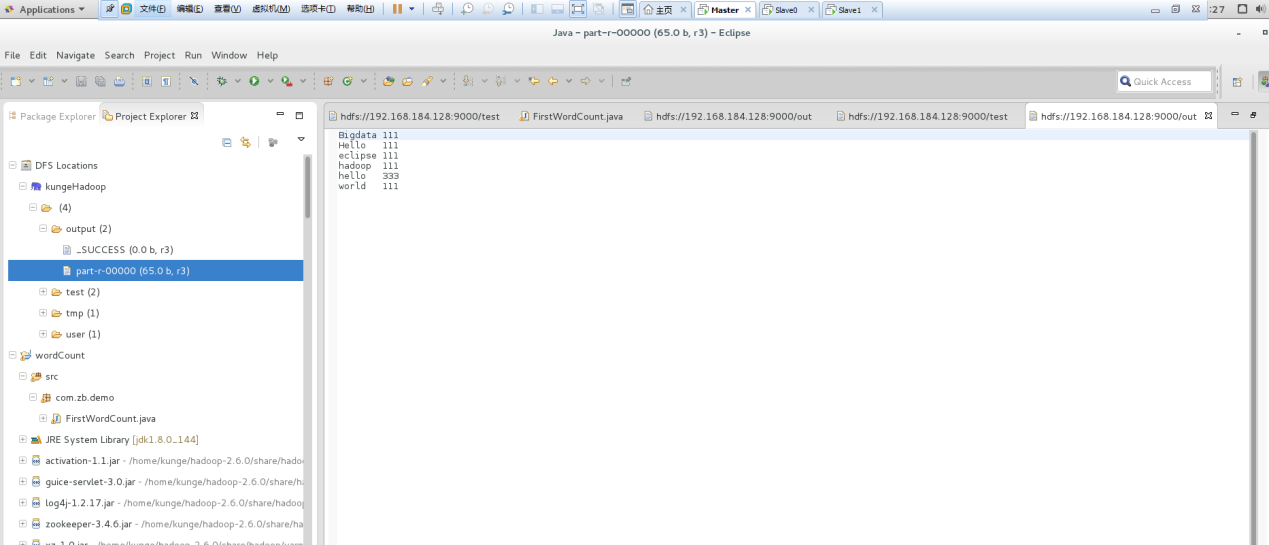
下面的大象终于出来了 还有左边的浏览器

第一个MapReduce程序

1. 准备一个txt文件 里面随便写点东西 要多一点
2. 通过haddop fs –put /input.txt /test/ 命令将上述txt传入HDFS目录下
3. 
4. 新建一个Map/Reduce project
5. 
6. 
7. 下一步下一步就好
8. 
9. 建包建类
10. 
11. 分析：在父类的mapper对象中，每次先记录一个横行数据，使用StringTokenizer类先进行每行分割，分割完成<偏移量（Long类型）,字符串>的map设置，然后进入该类map，完成上面的map分割，将value值继续分割，按照空格统计<单词,1>结果，进行收集到context对象中
12. 
13. Reduce方法分析，reduce会将上面map的每一个进行合并，因为reduce的排序和并会出现<单词,list(1,1,1,1,1)>又或者为<单词,list(10)>,上面的形式以来迭代器累加算出总数量，下面方式就一个直接就是结果，迭代就一次
14. 
15. Main函数分析
16. Job job = **new** ~~Job~~(cfg, "wordCount");进行mapreduce作业的初始化
17. job.setJarByClass(FirstWordCount.class); 设置本类作为jar处理对象
18. job.setMapperClass(TokenizerMapper.class);拆分类设置
19. job.setCombinerClass(SumReducer.class);中间结果合并设置
20. job.setReducerClass(SumReducer.class);合并设置
21. job.setOutputKeyClass(Text.class);
22. job.setOutputValueClass(IntWritable.class);
23. 因为结果是<单词,个数> 所以输出的key类型为Text(相当于java的String) value类型为IntWritable(相当于java的int)
24. for(int i = 0;i < otherArgs.length - 1;i++){
25. FileInputFormat.addInputPath(job, new Path(otherArgs[i]));
26. }
27. 未来调用该函数，会从args传入几个文件路径，我设置前面的为读取文件，最后一个文件为输出文件
29. FileOutputFormat.setOutputPath(job, new Path(otherArgs[otherArgs.length-1]));
30. System.exit(job.waitForCompletion(true)?0:1);
31. Run As->Run configuration



如图配置参数 一个是输入文件地址 一个是输出地址

1. 右击程序run on hadoop 运行完毕会在下图形成一个输出文件夹，可以看到2个文件一个是success，一个是part-r-0000,打开后者就能看到计数结果
2. 

HIVE和HBase安装