首届“智警杯”全国公安院校大学生大数据技能竞赛培训手册

北京红亚华宇科技有限公司

2019 年度

目录

一、 竞赛简介

1. 竞赛背景

2018 年 1 月 24 日，全国公安厅局长会议在北京召开，会议强调要牢牢把握大数据发展大势，牢固树立科技是核心战斗力的思想，按照推进政法工作智能化建设的要求，把大数据作为公安工作创新发展的大引擎、培育战斗力生成新的增长点，大力实施公安大数据战略，着力打造数据警务、建设智慧公安，全面推动公安工作质量变革、效率变革、动力变革，努力实现公安机关战斗力的跨越式发展。

目前，在我国公安院校设置数据挖掘与大数据分析相关课程的专业包括：计算机科学与技术、网络安全与执法、交通管理工程、侦查学、警务指挥与战术、犯罪学、公安管理学、刑事科学技术等。学习大数据需要从原理、技术与应用等不同角度掌握大数据分析的理论与方法。学生要很好地掌握这些课程，除了课堂学习，通过大数据竞赛活动加深理解和提高实际应用操作能力是非常重要的途径。目前，在我国高校还没有一个完整体系的大数据竞赛。因此，举办一个全国性的公安院校大学生大数据技能竞赛是非常有必要的。

2. 竞赛相关单位

赛项名称：首届“智警杯”全国公安院校大学生大数据技能竞赛

主办单位：基于大数据架构的公安信息化应用公安部重点实验室

中国大数据技术与应用联盟

承办单位：南京森林警察学院

北京警察学院

浙江警察学院

协办单位：江苏警官学院

广东警官学院

河南警察学院

贵州警察学院

江西警察学院

竞赛平台：北京红亚华宇科技有限公司

3. 竞赛目的

赛项的举办，旨在有效促进公安院校计算机科学与技术、网络安全与执法、交通管理工程、侦查学、警务指挥与战术、犯罪学、公安管理学、刑事科学技术等专业方向教学模式的探索性改良，推进相关专业课程体系、教学内容和教学方法等教学资源的提质升级，进而推动我国公安院校大数据方向教育上层建筑体系质的飞跃。

通过大数据技能竞赛，能够激发学生的自主学习热情，树立正确积极的职业价值观和人生观。通过大赛，可以提高实践教学课时量，学生可在“大数据竞赛平台”中以实际警务大数据项目案例开展平台搭建、数据采集、数据分析与挖掘等工作，提高学生的专业技能，并逐步实践“理实一体化”、“做学教一体化”的教学模式。

以大数据技能竞赛为支撑，提升公安院校大数据方向以及其他专业学生的技能及职业素养，满足我国公安部门的用人需求，实现行业资源、公安机关与教学资源的有机融合，使公安院校在专业建设、课程建设、人才培养方案和人才培养模式等方面紧跟机关及社会发展的需求，缩小学生能力与公安机关需求之间的差距，促进教学建设和教学改革。

4. 竞赛内容简介

基于大数据架构的公安信息化应用公安部重点实验室、中国大数据技术与应用联盟联合主办的首届“智警杯”全国公安院校大学生大数据技能竞赛，旨在提升公安院校大数据人才技能水平，增强领域知识覆盖范围，完善大数据人才培养机制，为我国公安大数据战略的稳步实施提供坚实可靠的人才基础。

赛项设计，以 2018 年 1 月 24 日召开的全国公安厅局长会议的指导意见为核心思想，着力提高预测预警能力，应用大数据、机器学习、人工智能等新技术，实现对各类风险隐患的敏锐感知、精确预警。

在充分调研公安单位的大数据岗位用人需求的基础上，梳理警务大数据人才所需知识技能，并据此设计出贴近实战的赛项内容。此外，赛项设置大数据环境搭建、数据预处理、数据初步分析、数据分析展示等环节。通过赛项的设置，主要考核锻炼选手以下能力：

1.大数据环境搭建与运维水平；

2.警务数据应用与预处理能力；

3.警务数据分析软件使用水平；

4.警务数据关联分析、数据挖掘能力。

5. 竞赛方式

（一） 竞赛采取团队比赛形式。

（二） 每个参赛队由 3 名选手和 1-2 名指导教师组成，参赛选手须为全日制在校学生、研究生，指导教师须为本校专职教师。

（三） 不得跨校组队，同一学校相同项目报名参赛选手不超过 2 队。

（四） 参赛选手在竞赛现场按照竞赛要求，完成比赛任务。

（五）竞赛分培训、选拔比赛、总决赛三个阶段。其中，培训在协办单位学校举行；比赛由大数据技能竞赛组委会统一组织，选拔赛总选为 3 个赛区，每个赛区前 10 名入围总决赛，另邀请 2 队参赛。

（六） 本赛项暂不邀请境外代表队参赛。

6. 竞赛方案的特点与创新点

竞赛设计重点突出以下几个方面的特征：

（一）竞赛内容覆盖行业主流大数据技术

竞赛内容选用大数据岗位的真实工作过程，从需求到具体实施都体现单位实际业务及大数据应用场景。

大数据集群搭建与运维赛项涵盖大数据平台安装部署、数据处理、数据分析以及平台的调优和维护等环节，综合考察大数据集群搭建与运维的相关知识技能；竞赛过程中所使用的工具和方法符合当前公安大数据相关岗位技能要求。

通过赛前准备及竞赛，让参赛选手及教师在 Linux 操作系统、Hadoop、Spark、MySQL 等大数据领域主流框架及工具的使用方式和配置方法、大数据处理技术、

大数据分析技术、大数据相关算法的应用、大数据平台运维等知识技能方面得到的充分锻炼和认知。

（二）竞赛过程是大数据技术应用的实战过程

赛项方案设计采用项目实战的模式，从项目背景、项目需求、项目任务、项目目标等方面进行设计和实施。

参赛选手的竞赛过程就是一次完整的大数据应用项目实战过程。竞赛任务按照平台搭建、数据处理、数据分析、数据分析及平台运维等标准项目流程逐步进行。参赛选手基于项目管理相关要求进行分工合作，按照竞赛任务规范操作，完成每个阶段任务，从而完成竞赛。让学生体会和提高大数据处理方面的职业技能。

（三）竞赛过程公开透明

大数据技术与应用赛项的竞赛方式、考察的知识技能范围、样题、赛项规程、赛项平台环境等内容按照组织规划通过高等院校能大赛官方网站进行公开、公示，按时召开赛项说明会，让参赛选手、教师对竞赛组织过程有充分的了解，安排赛场参观及竞赛现场观摩等组织过程，对赛场现场参观环节、赛场实况进行实时转播、网络直播或其它媒体等多渠道宣传报道，充分体现了竞赛的公平、公正、公开的原则。

（四）竞赛评分公平、公开、公正

在赛题设计方面，按照客观赛题评判唯一性的设计原则，进行赛题和评分标准的设计，考察参赛选手对关键知识技能掌握程度，保证了竞赛的公平、公正性。

本次竞赛任务采用自动化评分方式，避免了人为因素对竞赛结果的影响，同时设定详细的考核步骤与评分环节，使最终的竞赛结果更具公平与权威性。

（五）竞赛资源转化

本次赛项为专业建设服务、为教学服务的原则，积极贯彻“以赛促教、以赛促改、以赛促学”的精神，努力探索竞赛内容向教学资源的转化。结合本赛项的教学资源转化工作，协调相关专业院校及其工作单位，建立公安院校大数据技术方向核心技能标准、教学资源及其相关资源库，将竞赛内容转化为综合人才培养解决方案，实现对各公安院校相关专业现有基础资源的提升，达到赛项资源转化目标。

赛项竞赛任务均可转化为实际教学当中的课程建设资源或项目教学实验实

训案例，通过生成数字资源，强化教学中的实战化演练，贴近实战需求，提高学生综合技术能力。

通过对技能训练指导书和操作规范学习，计算机科学与技术、网络安全与执法、交通管理工程、侦查学、警务指挥与战术、犯罪学、公安管理学、刑事科学技术等相关专业的学生可学到大数据相关知识技能，能胜任公安部门的大数据应用相关岗位需要。

7. 竞赛时间安排与流程

1、竞赛报名时间

2019 年 3 月 1 日-2019 年 4 月 30 日

2、赛前培训时间

第一场：2019 年 4 月，广东警官学院

第二场：2019 年 4 月，南京森林警察学院

第三场：2019 年 4 月，吉林警察学院

3、选拔赛时间

场次	举办地	举办时间	参赛省份
第一场	浙江警察学院	2019 年 5 月	上海、浙江、江西、福建、江苏、安徽、湖南、湖北、山东、广东
第二场	北京警察学院	2019 年 5 月	黑龙江、吉林、辽宁、内蒙、北京、天津、河北、山西、新疆、河南
第三场	贵州警察学院	2019 年 6 月	贵州、四川、重庆、云南、陕西、甘肃、宁夏、青海、广西、海南、西藏、香港、澳门、台湾。

4、选拔赛日程安排：

日期	时间	内容
第一天	全天报道	各参赛队报到

	10:00-22:00	领队会（赛场纪律和赛场要求）
	17:00-19:00	场地参观，领队参观场地
第二天	8:00-8:30	参赛队赛场检录
	8:45-9:00	开放竞赛系统
	9:00-11:30	参赛队竞赛
	12:15-12:30	评分核分排名

5、总决赛时间地点安排

时间：2019 年 11 月

地点：北京警察学院

6、总决赛日程安排：

日期	时间	内容
第一天	全天报道 10:00-22:00	各参赛队报到
		领队会（赛场纪律和赛场要求）
	17:00-19:00	场地参观，领队参观场地
第二天	8:00-8:30	参赛队赛场检录
	8:45-9:00	开放竞赛系统
	9:00-12:00	参赛队竞赛
	12:15-12:30	评分核分
	13:30-15:30	赛项闭幕式及颁奖

8. 奖项设置

按照技能竞赛的有关规定，各赛项设参赛选手团体特等、一、二、三等奖。以赛项实际参赛队总数为基数，特、一、二、三等奖，获奖比例分别为 3 名、5 名、10 名、18 名（小数点后四舍五入），竞赛特等奖为 3000 元人民币、一等奖为 1500 元人民币、二等奖为 900 元人民币，三等奖为竞赛纪念品以此来鼓励成绩优异的参赛队伍。

同时，团队获得特、一、二等奖的参赛队伍的指导教师获“优秀指导教师奖”。

9. 竞赛会务组

9.1 竞赛会务负责人

张 京 晶

联系方式:18810696426

邮　　箱：zhangjingjing@hongyaa.com.cn

9.2 竞赛培训负责人

张福华　联系方式：18310396898

罗树国　联系方式：18513688920

赵利平　联系方式：18812615905

张　程　联系方式：17611347725

技术指导　QQ 群：696915346

二、 中国大数据技术与应用联盟简介

中国大数据技术与应用联盟（China Big Data Technology and Application Alliance，缩写：BDTAA，以下简称：联盟）是在工业和信息化部的指导下， 由中国通信企业协会通信网络运营专业委员会、北京邮电大学、中国管理科学研究院学术委员会共同发起，联合中国信息通信研究院、中国邮政集团公司、中国电信集团公司、中国移动通信集团有限公司、中国联合网络通信集团有限公司、中国铁塔股份有限公司、华为技术有限公司、江苏亨通产业集团、北京梅泰诺通信技术股份有限公司、北京大数据研究院、重庆大数据研究院有限公司等多家与大数据密切相关的企业、高校、科研机构和投资机构成立的非盈利性组织，是集大数据标准研究制定、技术应用推进、产业链合作、人才培养和投融资于一体的合作服务平台。

三、 北京红亚华宇科技有限公司简介

北京红亚华宇科技有限公司（简称：红亚科技）是一家聚焦信息技术发展，为教育从业者提供优质教育服务的创新型科技公司。面向国内本科及职业院校服务项目有大数据、人工智能、信息安全、网络工程、及软件工程等专业建设服务、

师资建设服务、实训基地建设服务及校企共建服务。现在全国已服务 600 多所高校。红亚人始终坚持用技术改变教育方式，让教育变的更加智慧的发展理念，致力于成为国内信息技术教育服务的龙头企业。

红亚科技的大数据实训平台是基于云模式的智慧教育大数据实验室的设计全面落实“产、学、用、监、评”一体化的思想和模式，从教学、实践、使用、监控、评估等多方面注重专业人才和特色人才的培养；平台内有大数据专项练习课程近 670 个，涵盖基础学习、大数据算法分析、算法应用、大数据生命周期大型案例实训等课程，将理论学习、实践教学和大数据搭建、挖掘、存储、分析实战融为一体，从易到难、循序渐进，逐步提升学生的学习技能和实践水平；定制专业化技能评估与教学监控功能，秉承“精准、先进、创新”思想，可实时监控学生操作，分析学习情况，评估学生知识水平；平台辅以配套的 6 本大数据实践系列教材及 PPT 讲义，深度的解决教师授课的压力。

红亚科技创始团队全部出身教育行业，公司现有员工 150 多名，本科学历以上人员占公司总人数的 90%。公司先后获得了国家高新技术企业、双软企业认证，并通过了 ISO9001 质量体系认证，AAA 级信用企业，A 级纳税企业、第五届北京最具文化影响企业 30 强认证企业、公安系统网络攻防竞赛优秀支持单位、工兵团网络攻防竞赛优秀支持单位。红亚科技自主研发的软件产品近 30 个，获得国家著作权的近 20 项，其中 3 个产品被评为北京市技术创新产品。

四、 红亚大数据赛前实训平台使用说明

1. 平台登陆

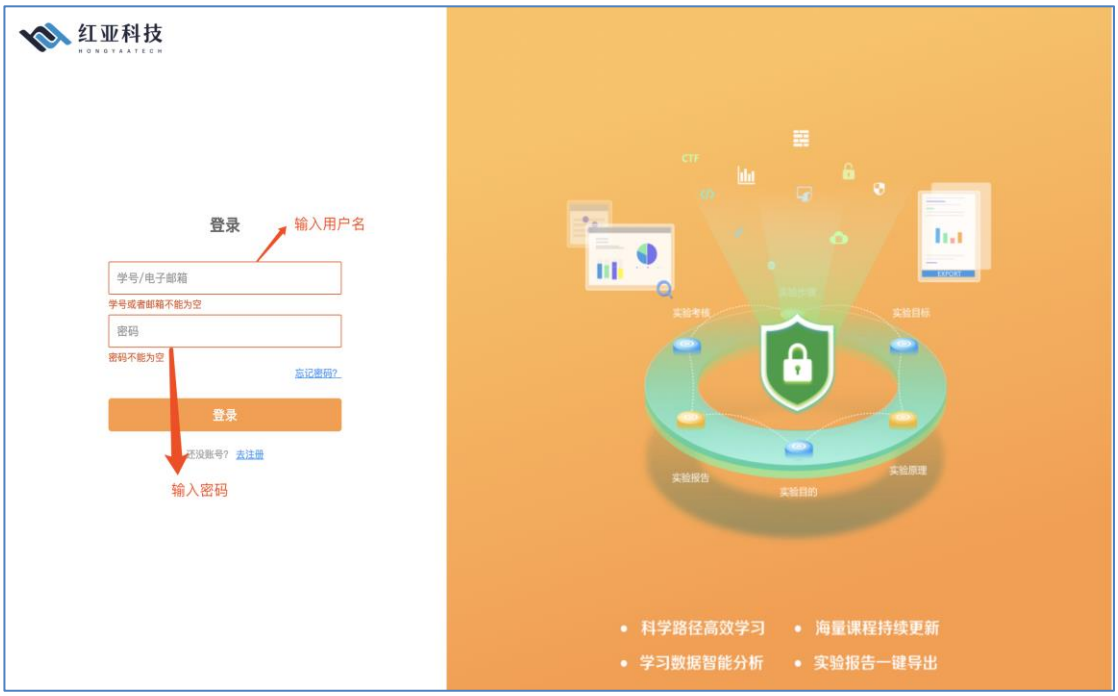
1.1 训练平台登录地址

红亚科技提供的大数据竞赛训练平台登录地址、账号、密码由培训人员在培训完成后单独发邮件给学员。

1.2 训练平台登录使用

红亚科技的大数据实训平台，采用的是私有云技术，平台采用 B/S 访问模式，可以在学校的实验室、图书馆、宿舍等通过网络访问。

请使用 Chrome68.0 或更高版登录，详细登录界面如下图所示：



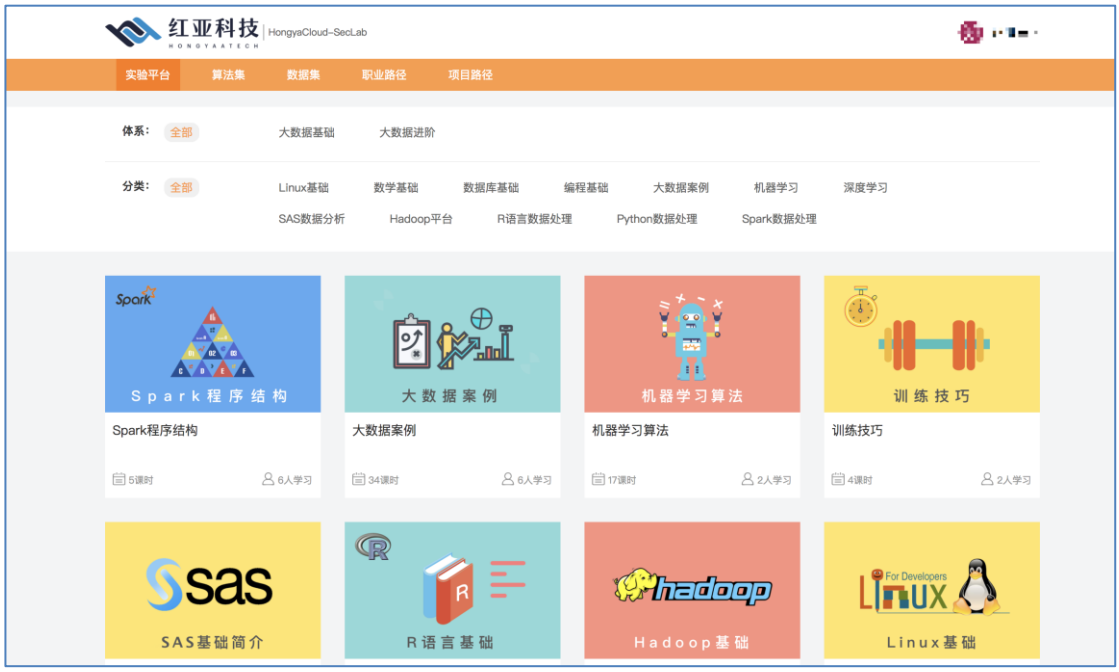
图：大数据实训系统登录界面

2. 平台功能简介

红亚科技大数据实训平台包括的主要功能有课程实验、算法集、数据集、职业路径、项目路径、个人中心、过程监控、教学中心等功能，可满足学生的实训学习，也可满足教师的智能管理。

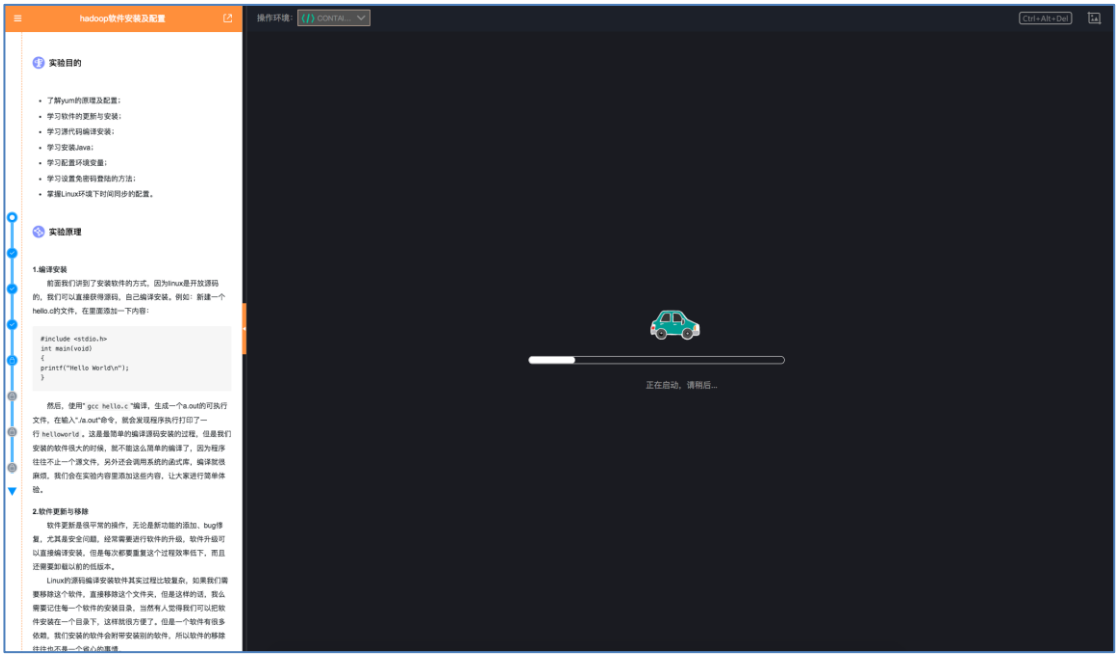
2.1 实验平台

实验平台为学生提供的是各种不同类型、不同层次实验的学习内容。学生可以在此进行大数据相关知识实验的学习和训练，其中内容包含：大数据、云计算、人智能等方向，可开展的实验共计 9 大体系 86 个模块，近千个实验，功能如下图：



图：系统主界面

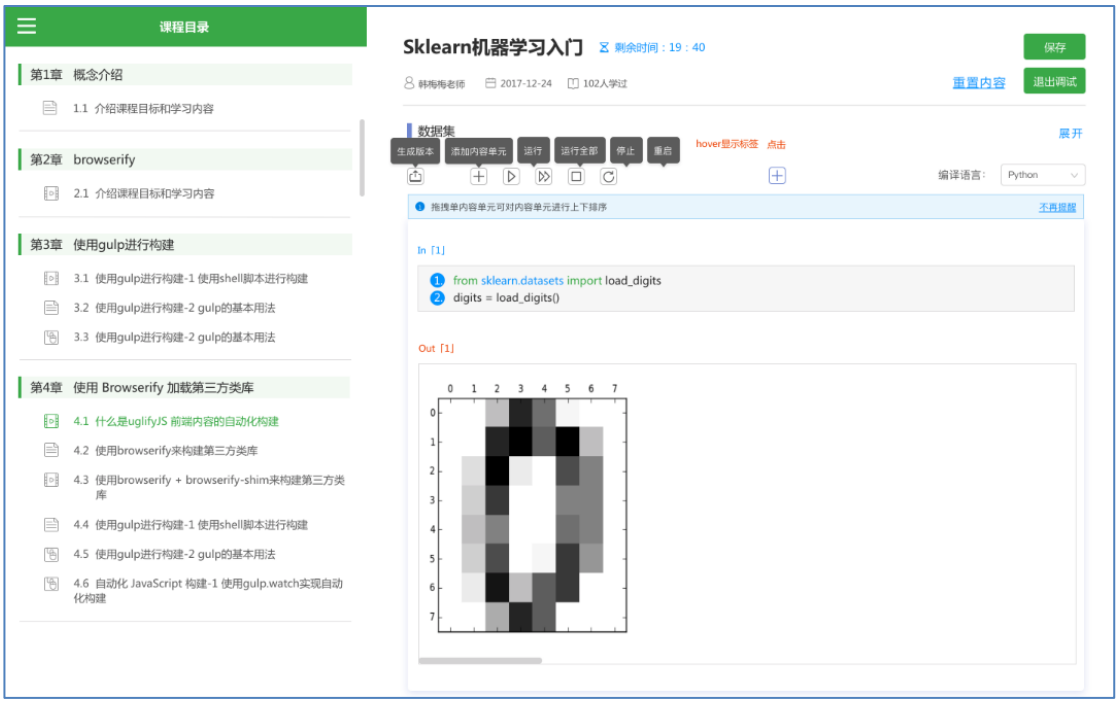
系统具有学习任务功能，为教师端课程进度分析功能提供数据基础，可记录学生已经学习课程及未学习课程情况，进入某一实验后会看到该实验的任务卡，直观展示当前实验的考核知识点以及综合测验和实验报告，按步骤考核点进行操作，实验内容包括目的和原理、实验步骤、虚拟机列表、配套的实验虚拟机；完成实验后，可对实验报告内的实操总用时、本班完成名次、步骤考核提交正确率、综合测验正确率、实验总结字数等信息进行查看；完成对实验报告的批阅。



图：实验操作台

2.2 算法集

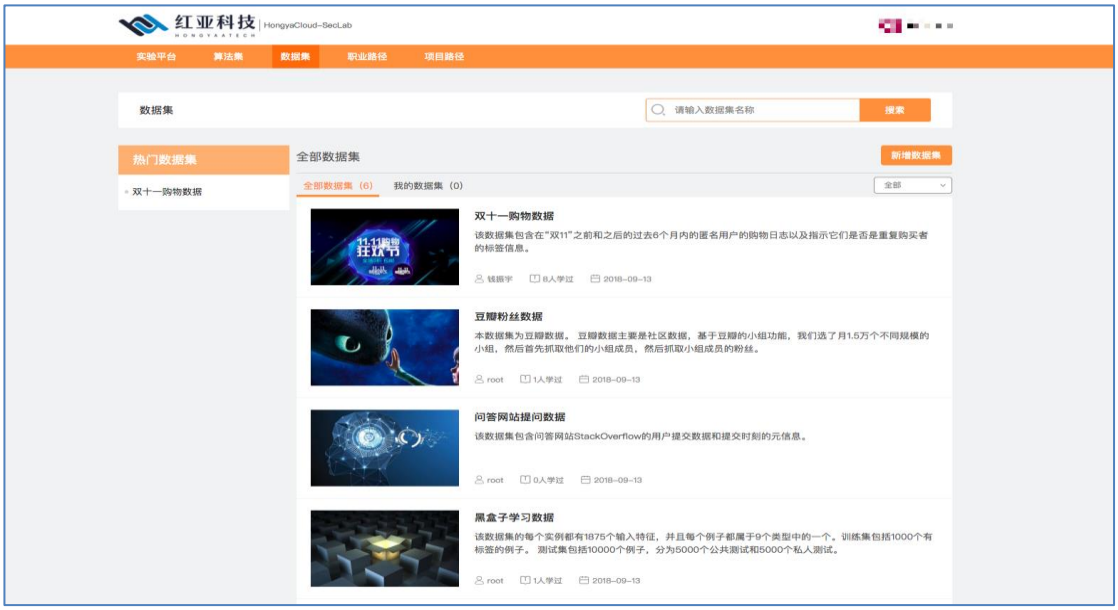
算法集是基于底层 Docker 技术进行开发以 Jupyter 在线编程为表现形式的一款线上算法编程功能，其中涉及到的算法包括 python、java、C++等。该功能可帮助学生在线上完成一些简单的算法学习，可调用平台上的数据集进行算法的测试和验证；同时也可帮助老师完成课上的针对某一算法与学生的互动。详细功能如下图所示：

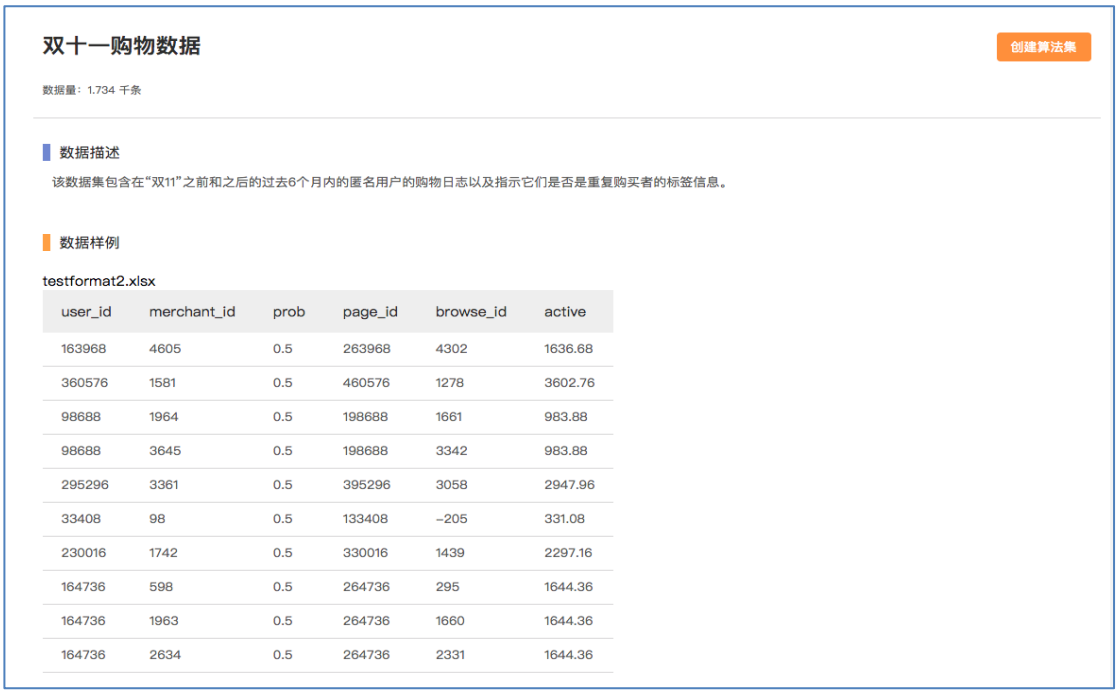


图：算法集

2.3 数据集

数据集顾名思义即数据的集合，平台自带一部分数据供学生学习、学生亦可通过算法集内的算法调用数据集中的数据完成大数据相关知识的训练；如 Hadoop 平台架构的搭建、运用 python 算法对数据的处理等。后台的管理功能是为教师提供对数据的名称、数据量大小、种类编辑和添加，实现平台的自定义功能，平台共集成了数据集达电子商务、农业、交通等 29 套数据集，详细功能如下图所示：





user_id	merchant_id	prob	page_id	browse_id	active
163968	4605	0.5	263968	4302	1636.68
360576	1581	0.5	460576	1278	3602.76
98688	1964	0.5	198688	1661	983.88
98688	3645	0.5	198688	3342	983.88
295296	3361	0.5	395296	3058	2947.96
33408	98	0.5	133408	-205	331.08
230016	1742	0.5	330016	1439	2297.16
164736	598	0.5	264736	295	1644.36
164736	1963	0.5	264736	1660	1644.36
164736	2634	0.5	264736	2331	1644.36

图 数据集功能

2.4 职业路径

职业路径是根据大数据专业不同的就业方向，如：大数据开发工程师、大数据架构师、数据分析师等就业岗位；根据这些岗位的不同特点制定不同学习课程，每一岗位对应的课程就是一套职业路径课程。

图：职业路径课程体系

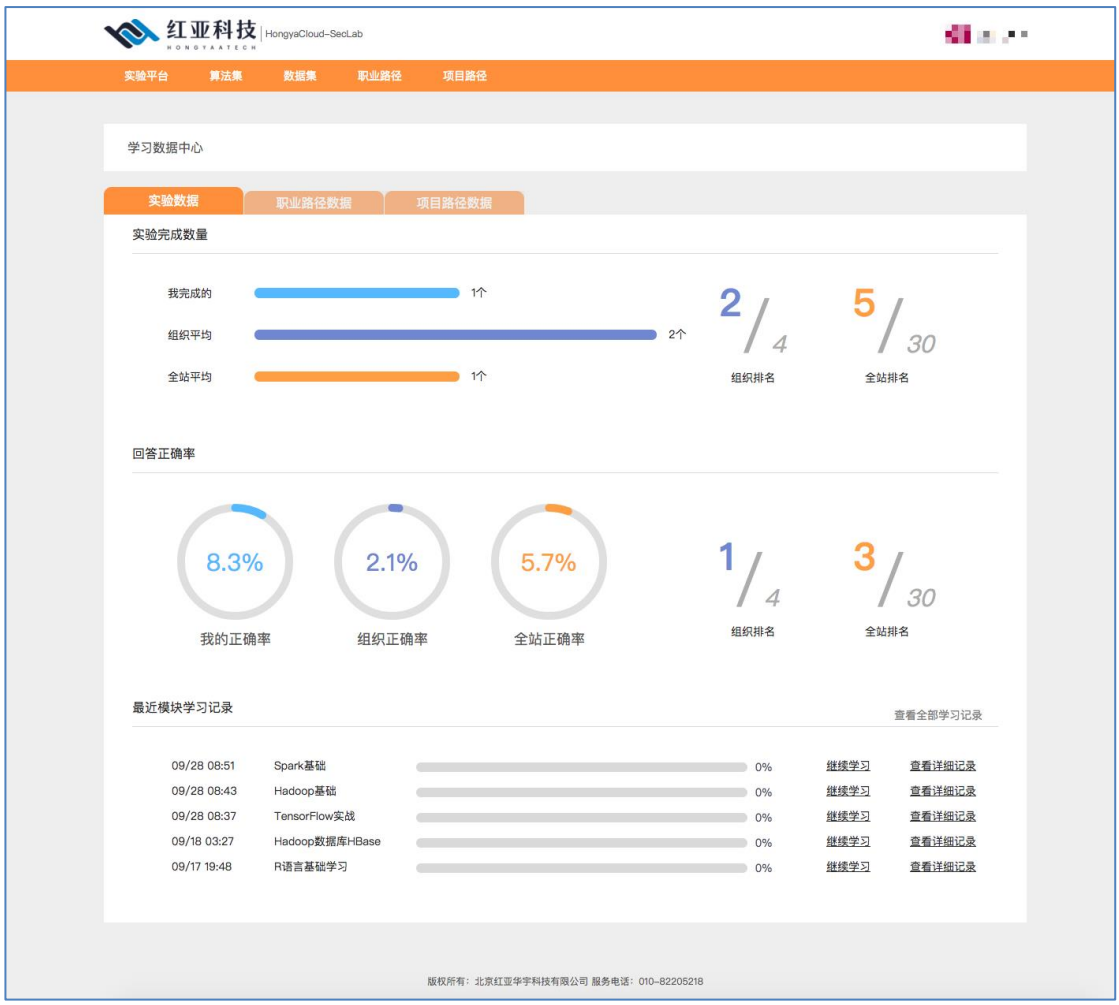
2.5 项目路径

项目路径是根据大数据项目从开始到结束的进程进行设计的，如：数据的采集、清洗、分析、可视化等环节；根据每个项目中的不同环节制定一套对应的学习课程，每一个项目对应的课程就是一套项目路径课程。

图：项目路径

2.6 个人中心

为了便于学生实时了解个人详细的学习情况，在原来实验记录的基础上，单独设置了学习数据中心并增加了学生实验完成数量、正确率、详细的实验学习情况，详细功能如下：

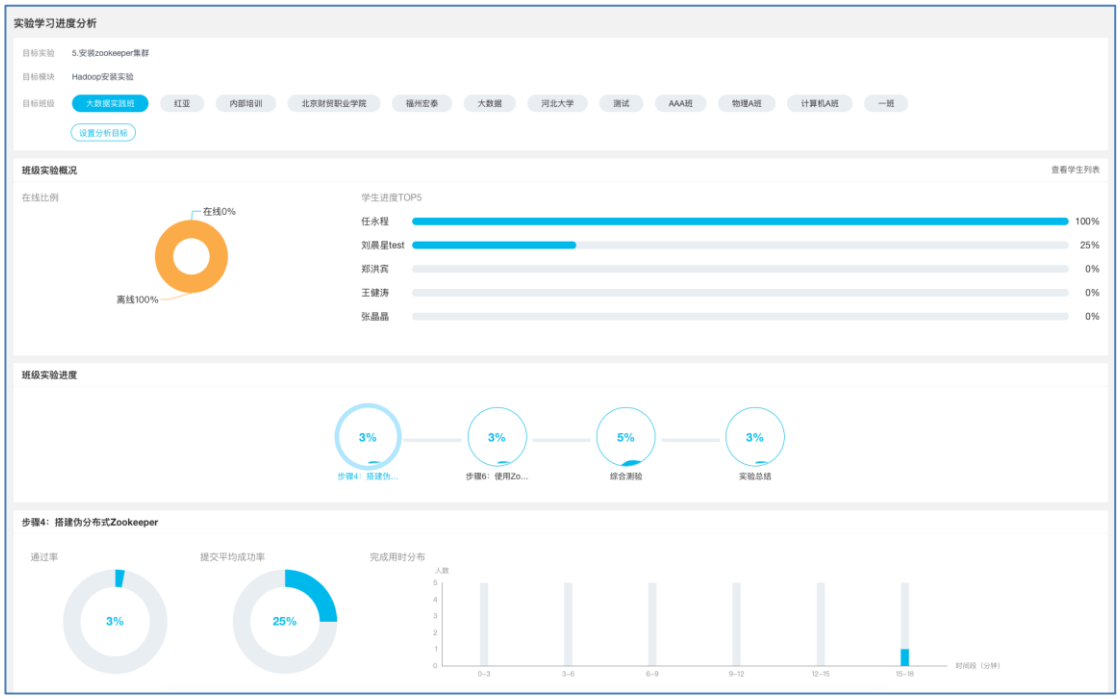


图：个人学习中心

2.7 学习进度管理

学习进度管理功能可自动检测到是否有学生正在进行实验，并可检测到哪些学生在上哪些课程，可设定当前正在进行实验的班级和实验进行实验状态分析；或者自行设定分析目标，可对目标实验、目标模块、目标班级等信息进行进度分析和对应情况的查看；其中当前班级学生在线与离线比例情况、学生学习进度 top5 可用图表形式进行展示。

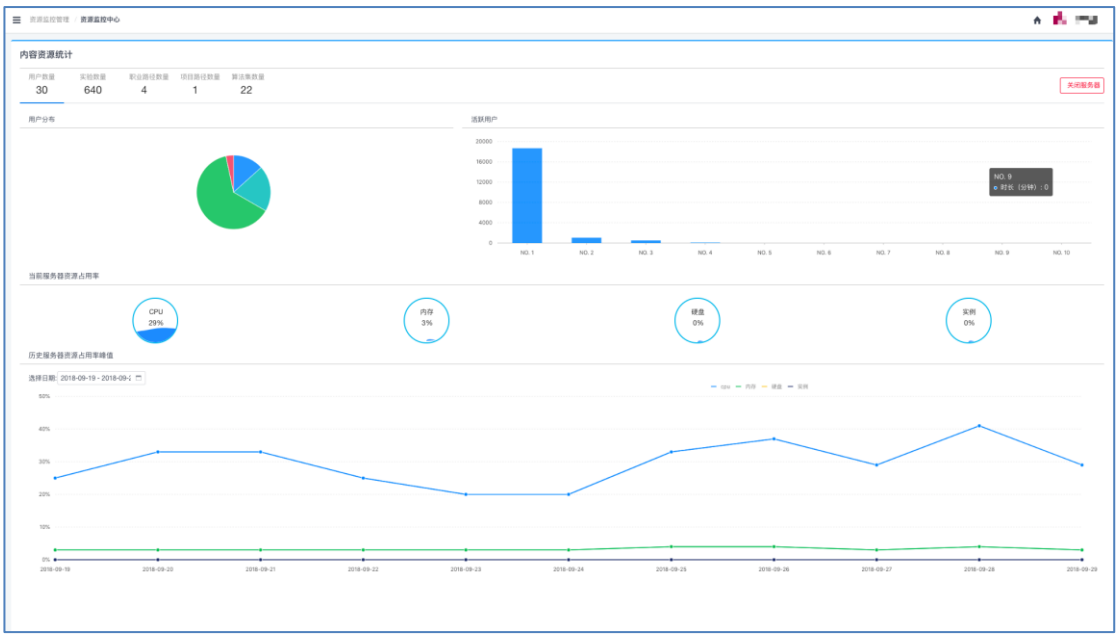
通过平台可查看实验步骤综合检测信息，包括以环状图展示通过率，可查看哪些学生完成了该实验哪些学生未完成的实验，以柱状图展示题目错误率分步，以柱状图展示所有人员完成检测的用时分布。



图：学习进度管理

2.8 教学管理中心

教学管理中心可实时的展示出整个实训平台的情况，并支持对学生直接管理，辅导等功能。



图：教学管理中心

3. 课程内容

平台提供的大数据课程资源库，主要以大数据、云计算、人工智能等方向进行设计和编写。其中包括 9 大体系，86 个模块，近千个实验。体系分别为：大数据基础、Hadoop 平台、Python 数据处理、SAS 数据分析、Spark 数据处理、R 语言数据处理、深度学习、机器学习、大数据案例。具体的模块分类如下表：

体系	模块	课程
大数据基础	编程基础	Python 基础
		R 语言基础
		Scala 基础
		Java 基础
	数学基础	信息论
		线性代数
		概率论
		数值计算
	Linux 基础	Linux 系统概述
		字符操作环境
		li 进程管理
		常用命令介绍
		系统监控与备份
		软件包管理
	数据库基础	Excel
		MySQL
		Oracle
		MongoDB
		Redis
Hadoop 平台	Hadoop 初始简介	Hadoop 介绍
		HDFS
		MapReduce
		Hive
		Hbase
		pig 语言
	Hadoop 组件安装与使用	Zookeeper
		Kafka
		Flume
		Mahout
		Storm
		ELK
		Impala
Python 数据处理	python 基础知识	Python 初始

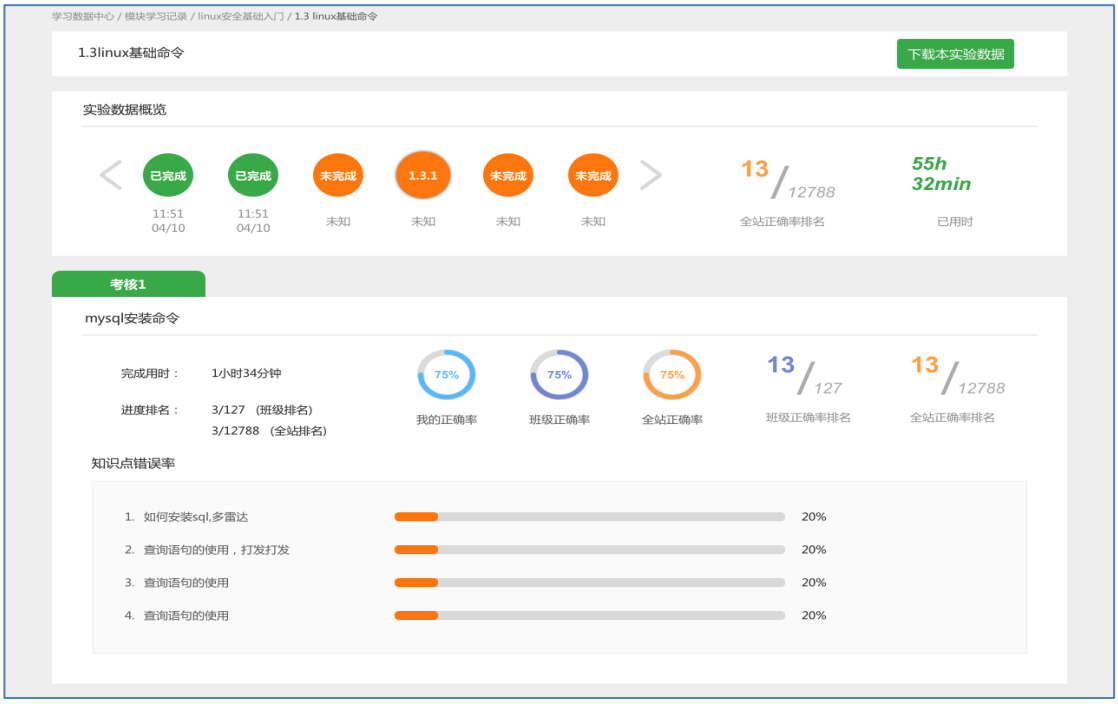
		列表与元组
		字符串与正则表达式
		字典与集合
		文件操作
	python 爬虫	爬虫应用场景
		HTTP 请求格式
		响应状态码
	Python 算法	常用聚类分析算法
		关联规则算法
		协同过滤算法
		离群点检测方法
SAS 数据分析	SAS 基础简介	SAS 概述
		SAS 软件基本介绍
		SAS 特点及模块组成
	SAS 数据可视化	图形绘制
		散点图
		箱型图
		图形编辑
	SAS 统计与建模	线性回归模型
		广义线性回归模型
		方差分析
	SAS 数据挖掘	因子分析
		聚类分析
		判断分析
		相关分析
		生存分析
Spark 数据处理	Spark 程序结构	Spark 结构设计
		Spark 算子分类
		Spark 核心组件
	Spark 流式计算	SparkStreaming 介绍
		SparkStreaming 结构与部署
	SparkSQL	SparkSQL 介绍
		SparkSQL 架构
		SparkSQL 的 shell
	Spark 与机器学习	线性回归算法原理
		SparkMlib 的数据类型
		K-Means 算法原理与使用
R 语言数据分析	R 语言数据处理	R 语言数据导入
		R 语言数据导出
		R 语言重复值处理
		R 语言缺失值处理
		R 语言空格值处理和字段抽取

		R 语言记录抽取和随机抽样
		R 语言记录合并
		R 语言字段匹配
		R 语言简单计算和数据标准化
		R 语言数据分组
		R 语言日期格式处理与日期抽取
		R 语言虚拟变量
	R 语言统计与建模	R 语言常用概率分布和渐进性
		R 语言置信区间和假设实验
		R 语言单元线性回归模型
		R 语言多元线性回归模型
		R 语言广义线性回归模型
	R 语言数据分析	R 语言基本统计
		R 语言对比分析
		R 语言分组分析
		R 语言分布分析
		R 语言交叉分析
		R 语言结构分析
		R 语言相关分析
		R 语言简单线性回归分析
		R 语言多重线性回归分析
		R 语言 RFM 分析
		R 语言矩阵分析
	R 语言数据可视化	饼图
		散点图
		折线图
		柱形图
		直方图
		箱线图
		树形图
		热力地图+地图
	R 语言数据分析综合应用	建立销售响应模型
		预测销售额
		水质评估
		财政收入分析预测模型
		骑车数据可视化分析
		房价指数的分析与预测
		电商评论情感分析
		航空公司价值分析
		游戏玩家付费行为预测
		用户留存分析实战
深度学习	神经网络	优化算法

		参数初始化
		超参数设计
	卷积神经网络	卷积基本概念
		卷积核
		池化层
		CNN 网络
	循环神经网络	RNN 和 BPTT 算法
		LSTM 算法
		应用场景
机器学习	机器学习基础	机器学习介绍
		逻辑回归模型
		损失函数
		梯度下降
	聚类算法	层次聚类
		密度聚类
		聚类评估
	分类方法	朴素贝叶斯
		决策树归纳
		随机森林
		SVM
		遗传算法
	用户画像	用户画像的维度
		用户标签
		用户画像使用方法
大数据案例	数据采集与清洗	数据采集原理
		数据清洗原理
	可视化工具	Python
		Echart
		NodeBox
		OpenLayers
		Leaflet
	数据案例分析	模型评估与优化
		出租车数据分析
		音乐分析
		电影评论情感分析
		金融数据分析
		大型商场销售额预测

4. 实验成绩及报告

通过该功能可以自动将学生的实验情况，生成实验报告，并由老师结合系统给出的数据进行打分，并存档，实验报告包括的内容有：教师批注、目的和原理、实验步骤及考核、综合测验、实验总结、实验详细数据分析。



图：自动生成实验报告

五、 评分标准制定原则

1. 竞赛规则

赛项评分采用结果评分方法，始终贯彻落实大赛一贯坚持的公开、公平和公正原则。结果评分：依据赛项评价标准，对参赛选手提交的竞赛成果由系统自动进行评分。赛项最终按总评分得分高低，确定奖项归属。

参与大赛赛项成绩管理的组织机构包括：裁判组、监督组和仲裁组，受赛项执委会统一领导。

1.1 裁判组

裁判组设裁判 3 名，全面负责赛项的裁判与管理工作。

裁判员工作分为三项工作：检录裁判、加密裁判、现场裁判。

检录裁判负责对参赛队伍（选手）进行姓名登记、身份核对等工作；现场裁判按规定做好赛场记录，维护赛场纪律。

1.2 监督组

监督组负责对裁判组整个过程的工作进行全程监督，并对竞赛成绩抽检复核。

1.3 仲裁组

仲裁组负责接受由参赛队领队提出的对裁判结果的申诉，组织复议并及时反馈复议结果。

2. 比赛秩序

- 参赛人员需提前准备笔记本电脑、电源(以比赛通知为准)。资料(包括书籍、电子资料)及移动存储工具；
- 比赛不得携带手机，不得使用 QQ、微信、钉钉等通讯工具；
- 在比赛开始前签到，并提供有效证件（学生证、身份证）以核实身份信息。
- 比赛正式开始后迟到 15 分钟者不得进入比赛场地；
- 竞赛工位通过事先分派决定，竞赛期间参赛选手不得随意离开竞赛工位，如需离开，请举手示意；
- 比赛过程中或比赛后发现问题（包括反映比赛或其它问题），应当示意裁判组，由裁判组进行解答；
- 禁止任何对比赛相关平台的网络攻击，违规者一律取消参赛资格；
- 比赛过程中不能相互交流，禁止参赛队伍之间分享任何解题思路，违规者一律取消参

赛资格；

- 竞赛结束（或提前完成）后，参赛选手要确认已成功提交的所有竞赛文档，在确认后不得进行任何操作。

3. 评分标准

3.1 竞赛成绩

所有题目分值随排名递减 1%，减到总分值 40%时不在递减分值。例如：得分点 A，分值：100 分；第 1 名提交本题答案者得 100 分、第 2 名提交本题答案者 99 分、第 3 名提交本题答案者 98 分、第 60 名提交本题答案者 41 分、第 61 名提交本题答案者 40 分、第 62 名提交本题答案者 40 分…以此类推。

竞赛总成绩是所有题目得分的总和。

3.2 成绩排名

排名从高到低，以总分成绩榜的顺序为准；

出现并列分数排名原则：若队伍间出现总分分数并列情况时，则按照第一个得分点提交答案时间先后进行排列；则同分情况下，首题答案提交时间靠前，排名靠前；

3.3 考核内容

评分标准以技能考核为主，突出创新能力考核，兼顾团队协作精神和职业道德素养综合评定。本表仅供参考（总分 100 分）。

序号	名称	分值占比	考核内容	考核技能
选拔赛：大数据集群搭建				
1	基础环境	30%	完成任务书要求的 Linux 基本环境配置。主要考试选手对于 Linux 操作系统的使用及配置方法。	配置 hostname 安装 net-tools 配置 firewall 配置 ntp 服务 配置 java 环境 配置 ssh 登陆
2	Hadoop 集群环境搭建	50%	按照任务书要求完成集群搭建，按照正常的大数据集群搭建顺序	1. 安装 zookeeper 2. 安装 Hadoop

			构建环境，涉及到 Hadoop 的配置情况、配置文件的存在意义以及集群开启方式；	
3	Spark 环境搭建	20%	完成任书要求的 Spark 环境安装配置。主要考察选手对于 Spark 组件工具的使用。	1. 配置 Spark 组件 2. 启动 Spark 环境
决赛：大数据分析挖掘				
1	构建数据仓库	20%	完成任书要求完成数据仓库的构架及原始数据的导入。主要考核数据库的构建能力、数据导入方式。	1. 构建数据仓库 2. 数据导入
2	数据清洗	20%	完成任务书要求的对于原始数据清洗的要求。主要考核选手对于数据清洗过程里掌握。	1. 数据观察 2. 数据抽取 3. 数据清洗 4. 数据转换
3	数据分析	30%	完成任务要求的编程，查看算法合理性以及代码规范性。主要考核学生数据分析算法设计与逻辑思维。	1.统计违法行为时段 2.统计事故地点 3.统计违法行为类型 4.实现统计事故车辆类型 5.道路交通安全预警
4	分析结果展示	30%	与实际分析结果进行比对。考核学生综合案例操作水平，越接近真实数据的，成绩越高。检验选手构建模型的有效性。	1.统计违法行为时段 2.统计事故地点 3.统计违法行为类型 4.统计事故车辆类型 5.严重交通事故数据

竞赛设计原则如下：

（一）公开、公平、公正。

（二）赛项关联专业人才需求量大或专业人才紧缺，服务国家重点战略。

（三）竞赛内容对应相关职业岗位或岗位群、体现专业核心能力与核心知识、涵盖丰富的专业知识与专业技能点。

（四）竞赛平台成熟。根据行业特点，赛项选择相对先进、通用性强、社会保有量高的设备与软件。

六、 考核点培训教案

1. 环境说明

1.1 采用红亚科技大数据平台

红亚科技大数据平台操作方式详细见本文件第四章，由于公网带宽有限，训练环境将为时分段为学员提供训练服务，因此使用红亚科技平台进行练习，需要严格按要求在规定的时间段里进行训练。

1.2 自行搭建集群

我们建议学员使用 VMware 自行搭建环境，搭建环境请参考本章第 2 节环境配置说明，结合配置环境情况进行搭建环境，相关软件可在百度网盘下载(链接：https://pan.baidu.com/s/1Fj822u0yIIXl8zsrV0cCbg 提取码：0h40)。搭建好环境后，请参考以下操作方法进行训练。

2. 竞赛环境配置说明表

所用软件	版本号
hadoop	hadoop-2.7.3.tar.gz
zookeeper	zookeeper-3.4.10.tar.gz
jdk	jdk-8u171-linux-x64.tar.gz
scala	scala-2.11.12.tgz
spark	spark-2.4.0-bin-hadoop2.7.tgz
操作系统及CPU等配置信息	详细信息
Centos7	CentOS Linux release 7.3.1611 (Core)

CPU	2CPU
内存	4G
硬盘	100G
安装包目录	/opt/soft
工作目录	/usr/

3. 大数据集群搭建

3.1 基本环境与 zookeeper 安装

本次集群搭建共有三个节点，包括一个主节点 master，和两个从节点 slave1 和 slave2。具体操作如下：

3.1.1 修改主机名（三台机器均执行）

（1）以主机点 master 为例，首次切换到 root 用户：su

（2）修改主机名为 master：

hostnamectl set-hostname master

```
[root@host-192-168-15-104 ~]# su
[root@master ~]# hostnamectl set-hostname master
```

（3）永久修改主机名，编辑/etc/sysconfig/network 文件，内容如下：

NETWORKING=yes

HOSTNAME=master

```
[root@master ~]# vi /etc/sysconfig/network

# Created by anaconda
NETWORKING=yes
HOSTNAME=master
```

注意保存退出。

（4）下载相关工具

yum install -y net-tools

```
[root@master ~]# yum install -y net-tools
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Running transaction check
---> Package net-tools.x86_64 0:2.0-0.22.20131004git.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package             Arch             Version                       Repository       Size
================================================================================
Installing:
 net-tools           x86_64           2.0-0.22.20131004git.el7      base            305 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 305 k
Installed size: 917 k
Downloading packages:
net-tools-2.0-0.22.20131004git.el7.x86_64.rpm                | 305 kB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : net-tools-2.0-0.22.20131004git.el7.x86_64                   1/1
  Verifying  : net-tools-2.0-0.22.20131004git.el7.x86_64                   1/1

Installed:
  net-tools.x86_64 0:2.0-0.22.20131004git.el7

Complete!
[root@master ~]#
```

（5）保存该文件，重启计算机：reboot

（6）查看是否生效：hostname

```
[root@master ~]# reboot

Connection closed by foreign host.

Disconnected from remote host(zook_master) at 19:41:07.

Type `help' to learn how to use Xshell prompt.
[d:\~]$

Connecting to 192.168.15.104:22...
Connection established.
To escape to local shell, press 'Ctrl+Alt+]'.

Last login: Fri Sep 28 19:29:20 2018
[root@master ~]#
```

重启机器

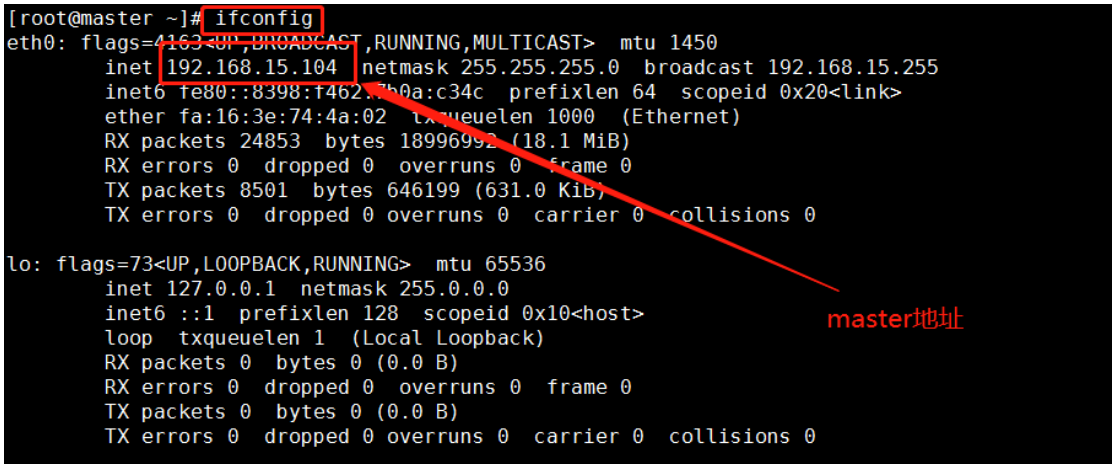
主机名已修改为master

3.1.2 配置 host 文件（三台机器）

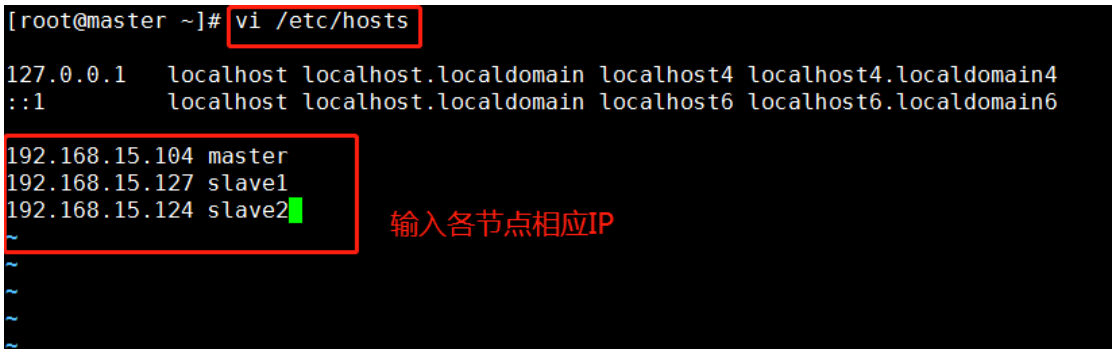
使各个节点能使用对应的节点主机名连接对应的地址。

hosts 文件主要用于确定每个结点的 IP 地址，方便后续各结点能快速查到并访问。在上述 3 个虚机结点上均需要配置此文件。由于需要确定每个结点的 IP 地址,所以在配置 hosts 文件之前需要先查看当前虚机结点的 IP 地址是多少.

（3） 可以通过 ifconfig 命令进行查看。



（2）查看节点地址之后将三个节点的 ip 地址以及其对应的名称写进 hosts 文件。这里我们设置为 master、slave1、slave2。注意保存退出。

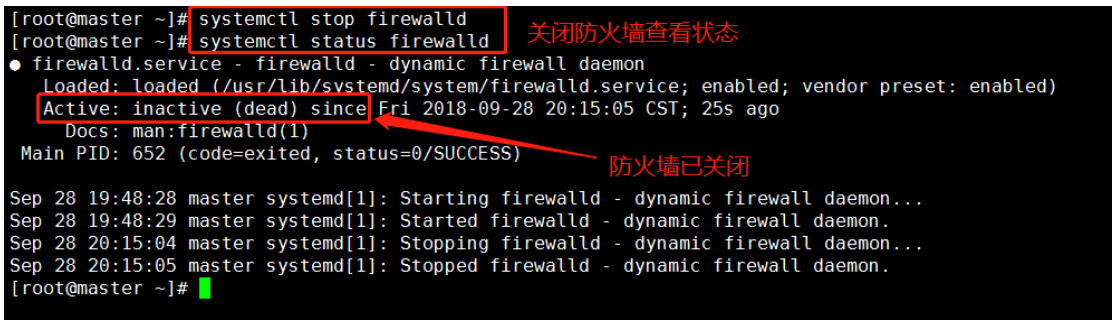


3.1.3 关闭防火墙（三台机器）

centos7 中防火墙命令用 firewalld 取代了 iptables，当其状态是 dead 时，即防火墙关闭。

关闭防火墙：systemctl stop firewalld

查看状态：systemctl status firewalld



3.1.4 时间同步

（1）时区一致。要保证设置主机时间准确，每台机器时区必须一致。实验中我们需要同步网络时间，因此要首先选择一样的时区。先确保时区一样，否则同步以后时间也是有时区差。

可以使用 date 命令查看自己的机器时间。

```
[root@master ~]# date
Fri Sep 28 20:29:39 CST 2018
```

（2）选择时区：tzselect

```
[root@master ~]# tzselect
Please identify a location so that time zone rules can be set correctly.
Please select a continent or ocean.
 1) Africa
 2) Americas
 3) Antarctica
 4) Arctic Ocean
 5) Asia
 6) Atlantic Ocean
 7) Australia
 8) Europe
 9) Indian Ocean
10) Pacific Ocean
11) none - I want to specify the time zone using the Posix TZ format.
#? 5
Please select a country.
 1) Afghanistan           18) Israel                35) Palestine
 2) Armenia               19) Japan                 36) Philippines
 3) Azerbaijan            20) Jordan                37) Qatar
 4) Bahrain               21) Kazakhstan            38) Russia
 5) Bangladesh            22) Korea (North)         39) Saudi Arabia
 6) Bhutan                23) Korea (South)         40) Singapore
 7) Brunei                24) Kuwait                41) Sri Lanka
 8) Cambodia              25) Kyrgyzstan            42) Syria
 9) China                 26) Laos                  43) Taiwan
10) Cyprus                27) Lebanon               44) Tajikistan
11) East Timor            28) Macau                 45) Thailand
12) Georgia               29) Malaysia              46) Turkmenistan
13) Hong Kong             30) Mongolia              47) United Arab Emirates
14) India                 31) Myanmar (Burma)       48) Uzbekistan
15) Indonesia             32) Nepal                 49) Vietnam
16) Iran                  33) Oman                  50) Yemen
17) Iraq                  34) Pakistan
#? 9
Please select one of the following time zone regions.
1) Beijing Time
2) Xinjiang Time
#? 1

The following information has been given:

        China
        Beijing Time

Therefore TZ='Asia/Shanghai' will be used.
Local time is now:      Fri Sep 28 20:33:01 CST 2018.
Universal Time is now:  Fri Sep 28 12:33:01 UTC 2018.
Is the above information OK?
1) Yes
2) No
#? 1
```

选择时区
选择亚洲
中国
北京时间
覆盖时间

由于 hadoop 集群对时间要求很高，所以集群内主机要经常同步。我们使用 ntp 网络时间协议进行时间同步，master 作为 ntp 服务器，其余的当做 ntp 客户端。

（3）下载 ntp（三台机器）

yum install -y ntp

```
[root@master ~]# yum install -y ntp
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Running transaction check
---> Package ntp.x86_64 0:4.2.6p5-28.el7.centos will be installed
--> Processing Dependency: ntpdate = 4.2.6p5-28.el7.centos for package: ntp-4.2.6p5-28.el7.centos.x86
_64
--> Processing Dependency: libcrypto.so.10(OPENSSL_1.0.2)(64bit) for package: ntp-4.2.6p5-28.el7.cent
os.x86_64
--> Processing Dependency: libopts.so.25()(64bit) for package: ntp-4.2.6p5-28.el7.centos.x86_64
--> Running transaction check
---> Package autogen-libopts.x86_64 0:5.18-5.el7 will be installed
---> Package ntpdate.x86_64 0:4.2.6p5-28.el7.centos will be installed
---> Package openssl-libs.x86_64 1:1.0.1e-60.el7 will be updated
--> Processing Dependency: openssl-libs(x86-64) = 1:1.0.1e-60.el7 for package: 1:openssl-1.0.1e-60.el
7.x86_64
---> Package openssl-libs.x86_64 1:1.0.2k-12.el7 will be an update
```

（4）master 作为 ntp 服务器，修改 ntp 配置文件。（master 上执行）

vi /etc/ntp.conf

server 127.127.1.0 # local clock

fudge 127.127.1.0 stratum 10 #stratum 设置为其它值也是可以的，其范围为 0~15

```
disable monitor

server  127.127.1.0
fudge   127.127.1.0     stratum 10
```

重启 ntp 服务。

/bin/systemctl restart ntpd.service

（5） 其他机器同步（slave1，slave2）

等待大概五分钟，再到其他机上同步该 master 服务器时间。

ntpdate master

```
[root@slave2 ~]# ntpdate  master
28 Sep 20:51:27 ntpdate[2338]: adjust time server 192.168.15.104 offset 0.201392 sec
[root@slave2 ~]#
```

同步master时间

如果配置平台式没有外网连接可以将三台机器设为统一时间，输入命令：

date -s 10:00（时间）

3.1.5 配置 ssh 免密

SSH 主要通过 RSA 算法来产生公钥与私钥，在数据传输过程中对数据进行加密来保障数

据的安全性和可靠性，公钥部分是公共部分，网络上任一结点均可以访问，私钥主要用于对数据进行加密，以防他人盗取数据。总而言之，这是一种非对称算法，想要破解还是非常有难度的。Hadoop 集群的各个结点之间需要进行数据的访问，被访问的结点对于访问用户结点的可靠性必须进行验证，hadoop 采用的是 ssh 的方法通过密钥验证及数据加解密的方式进行远程安全登录操作，当然，如果 hadoop 对每个结点的访问均需要进行验证，其效率将会大大降低，所以才需要配置 SSH 免密码的方法直接远程连入被访问结点，这样将大大提高访问效率。

（1） 每个结点分别产生公私密钥：

ssh-keygen -t dsa -P ‘’ -f ~/.ssh/id_dsa（三台机器）

秘钥产生目录在用户主目录下的.ssh 目录中，进入相应目录查看：

cd .ssh/

```
[root@master ~]# ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
Generating public/private dsa key pair.
Your identification has been saved in /root/.ssh/id_dsa.
Your public key has been saved in /root/.ssh/id_dsa.pub.
The key fingerprint is:
1d:ee:82:6a:09:1d:3f:a5:9b:13:6e:b4:9b:87:a8:ab root@master
The key's randomart image is:
+--[ DSA 1024]----+
|                 |
|                 |
|          .      |
|     .   .o .    |
|    . o oS o     |
|   . . *. .      |
|    . =.B. .     |
|     +.O...      |
|E.o+..o+         |
+-----------------+
[root@master ~]# cd .ssh/
[root@master .ssh]# ls
id_dsa  id_dsa.pub
```

id_dsa为私钥，id_dsa.pub为公钥

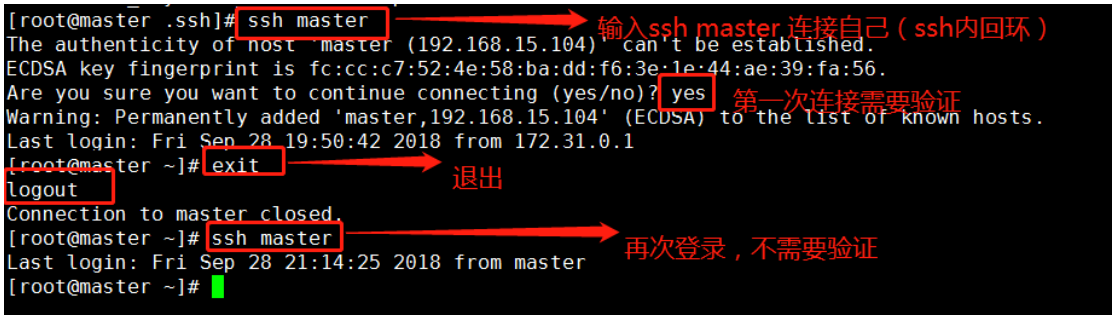
（2） Id_dsa.pub 为公钥，id_dsa 为私钥，紧接着将公钥文件复制成 authorized_keys 文件：（仅 master）

cat id_dsa.pub >> authorized_keys（注意在.ssh/路径下操作）

```
[root@master .ssh]# cat id_dsa.pub >> authorized_keys
[root@master .ssh]# ls
authorized_keys  id_dsa  id_dsa.pub
[root@master .ssh]#
```

在主机上连接自己，也叫做 ssh 内回环。

ssh master



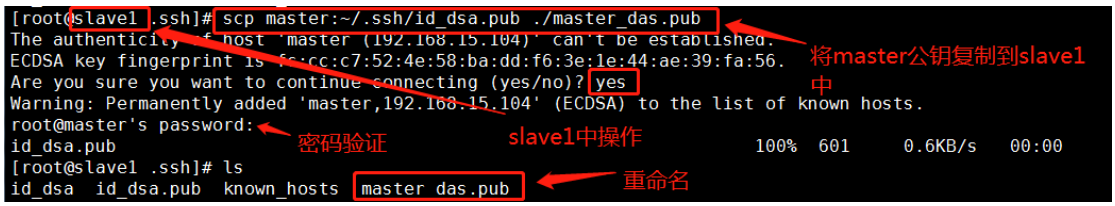
（3） 让主结点 master 能通过 SSH 免密码登录两个子结点 slave。（slave 中操作）

为了实现这个功能，两个 slave 结点的公钥文件中必须要包含主结点的公钥信息，这样

当 master 就可以顺利安全地访问这两个 slave 结点了。

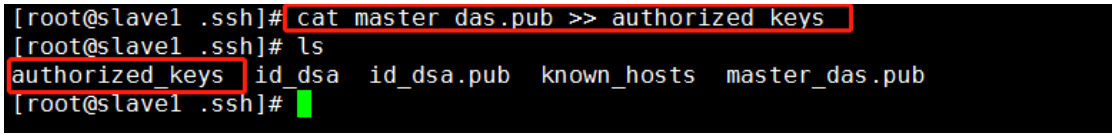
slave1 结点通过 scp 命令远程登录 master 结点，并复制 master 的公钥文件到当前的目录下，且重命名为 master_das.pub，这一过程需要密码验证。

scp master:~/.ssh/id_dsa.pub ./master_das.pub

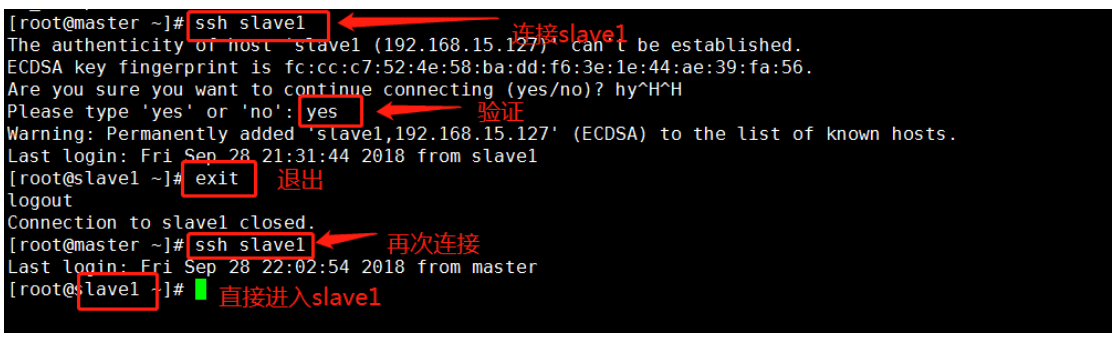


将 master 结点的公钥文件追加至 authorized_keys 文件：

cat master_dsa.pub >> authorized_keys



这时，master 就可以连接 slave1 了。



slave1 结点首次连接时需要，“yes”确认连接，这意味着 master 结点连接 slave1 结点时需要人工询问，无法自动连接，输入 yes 后成功接入，紧接着注销退出至 master 结点。

同理 slave2 中也是这么操作。

注意：两个结点的 ssh 免密码登录已经配置成功，还需要对主结点 master 也要进行上面的同样工作，因为 jobtracker 有可能会分布在其它结点上，jobtracker 有不存在 master 结点上的可能性。在上一步骤中，我们已经进行过此操作，这里仅做提醒。

3.1.6 安装 JDK（三台机器）

（1）首先建立工作路径/usr/java。

```
mkdir -p /usr/java
tar -zxvf /opt/soft/jdk-8u171-linux-x64.tar.gz -C /usr/java/
```

```
[root@master ~]# mkdir -p /usr/java
[root@master ~]# tar -zxvf /opt/soft/jdk-8u171-linux-x64.tar.gz -C /usr/java/
```

2.修改环境变量

```
[root@master ~]# cd /usr/java/
[root@master java]# ls
jdk1.8.0_171
[root@master java]# cd jdk1.8.0_171/
[root@master jdk1.8.0_171]# pwd
/usr/java/jdk1.8.0_171
[root@master jdk1.8.0_171]# vi /etc/profile
```

进入JDK目录

查看路径

修改环境变量：vi /etc/profile

添加内容如下：

```
export JAVA_HOME=/usr/java/jdk1.8.0_171
export CLASSPATH=$JAVA_HOME/lib/
export PATH=$PATH:$JAVA_HOME/bin
export PATH JAVA_HOME CLASSPATH
```

```
export PATH USER LOGNAME MAIL HOSTNAME HISTSIZE HISTCONTROL

export JAVA_HOME=/usr/java/jdk1.8.0_171
export CLASSPATH=$JAVA_HOME/lib/
export PATH=$PATH:$JAVA_HOME/bin
export PATH JAVA_HOME CLASSPATH
```

添加环境变量

生效环境变量：source /etc/profile

查看 java 版本：java -version

```
[root@master jdk1.8.0_171]# source /etc/profile
[root@master jdk1.8.0_171]# java -version
java version "1.8.0_171"
Java(TM) SE Runtime Environment (build 1.8.0_171-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.171-b11, mixed mode)
[root@master jdk1.8.0_171]#
```

jdk安装成功

同理 slave 节点，相同安装步骤。

注意：如果在 slave 节点中安装较慢，可以使用 scp 命令，将相同的文件从 master 中复制过来。

在 master 中将 JDK 复制到 slave2 中（要保证 slave2 中已有相应目录）。

```
[root@master jdk1.8.0_171]# scp -r /usr/java/jdk1.8.0_171 slave2:/usr/java/
```

3.1.7 安装 zookeeper

（1）修改主机名称到 IP 地址映射配置。

```
vi /etc/hosts
192.168.15.104 master master.root
192.168.15.127 slave1 slave1.root
192.168.15.124 slave2 slave2.root
```

```
[root@slave2 ~]# vi /etc/hosts

127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6


192.168.15.104 master master.root
192.168.15.127 slave1 slave1.root
192.168.15.124 slave2 slave2.root
```

（2）修改 ZooKeeper 配置文件。在其中 master 机器上，用 tar -zxvf 命令解压缩 zookeeper-3.4.10.tar.gz。

创建工作目录：mkdir -p /usr/zookeeper

解压：tar -zxvf /opt/soft/zookeeper-3.4.10.tar.gz -C /usr/zookeeper/

```
[root@master ~]# mkdir -p /usr/zookeeper
[root@master ~]# tar -zxvf /opt/soft/zookeeper-3.4.10.tar.gz -C /usr/zookeeper/
```

（3）配置文件 conf/zoo.cfg

用 cd 命令进入 zookeeper-3.4.10/conf 目录下，将 zoo_sample.cfg 文件拷

贝一份，命名为为“zoo.cfg”。

scp zoo_sample.cfg zoo.cfg

Zoo.cfg 文件配置

tickTime=2000

initLimit=10

syncLimit=5

dataDir=/usr/zookeeper/zookeeper-3.4.10/zkdata

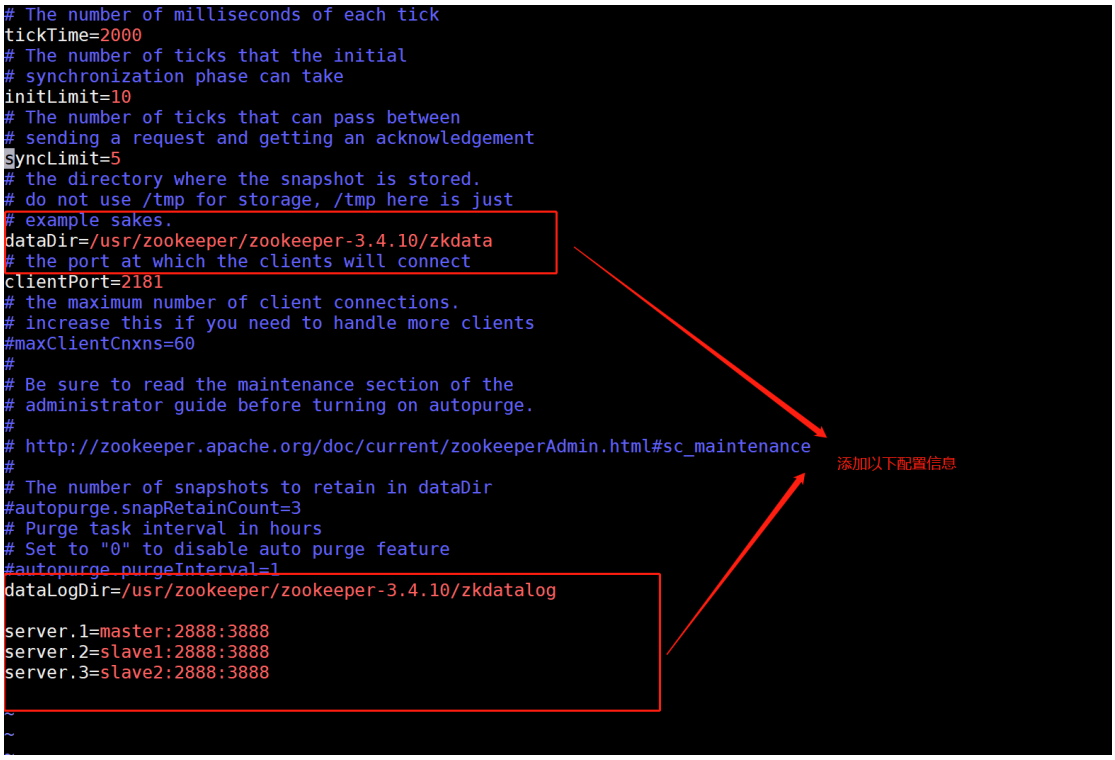
clientPort=2181

dataLogDir=/usr/zookeeper/zookeeper-3.4.10/zkdatalog

server.1=master:2888:3888

server.2=slave1:2888:3888

server.3=slave2:2888:3888



（4）在 zookeeper 的目录中，创建 zkdata 和 zkdatalog 两个文件夹。zkdatalog 文件夹，是为了指定 zookeeper 产生日志指定相应的路径。

mkdir zkdata

mkdir zkdatalog

```
[root@master zookeeper-3.4.10]# mkdir zkdata
[root@master zookeeper-3.4.10]# mkdir zkdatalog
[root@master zookeeper-3.4.10]# ls
bin         contrib     ivysettings.xml  LICENSE.txt           README.txt  zkdata                zookeeper-3.4.10.jar.asc
build.xml   dist-maven  ivy.xml          NOTICE.txt            recipes     zkdatalog             zookeeper-3.4.10.jar.md5
conf        docs        lib              README_packaging.txt  src         zookeeper-3.4.10.jar  zookeeper-3.4.10.jar.sha1
[root@master zookeeper-3.4.10]# pwd
/usr/zookeeper/zookeeper-3.4.10
[root@master zookeeper-3.4.10]#
```

在zookeeper安装目录下创建 zkdata和zkdatalog

复制此路径写入zoo.cfg文件中

（5）进入 zkdata 文件夹，创建文件 myid。

```
[root@master zookeeper-3.4.10]# cd zkdata
[root@master zkdata]# vi myid

1

~
```

（6）远程复制分发安装文件

上面已经在一台机器 master 上配置完成 ZooKeeper，现在可以将该配置好的安装文件远程拷贝到集群中的各个结点对应的目录下：

```
scp -r /usr/zookeeper root@slave1:/usr/
scp -r /usr/zookeeper root@slave2:/usr/
```

```
[root@master usr]# scp -r /usr/zookeeper root@slave1:/usr/
```

（7）设置 myid。在我们配置的 dataDir 指定的目录下面，创建一个 myid 文件，里面内容为一个数字，用来标识当前主机，conf/zoo.cfg 文件中配置的 server.X 中 X 为什么数字，则 myid 文件中就输入这个数字。

slave1 中为 2；slave2 中为 3。

```
cd /usr/zookeeper/zookeeper-3.4.10/zkdata
```

```
[root@slave1 ~]# cd /usr/zookeeper/zookeeper-3.4.10/zkdata
[root@slave1 zkdata]# ls
myid
[root@slave1 zkdata]# vi myid

2

~
~
```

slave1中为2

```
[root@slave2 ~]# cd /usr/zookeeper/zookeeper-3.4.10/zkdata
[root@slave2 zkdata]# vi myid

3
~
```

（8）配置环境变量并启动 ZooKeeper。在每台机器上操作如下：

```
vi /etc/profile
```

```
#set zookeeper environment
export  ZOOKEEPER_HOME=/usr/zookeeper/zookeeper-3.4.10
PATH=$PATH:$ZOOKEEPER_HOME/bin
```

```
# Functions and aliases go in /etc/bashrc

# It's NOT a good idea to change this file unless you know what you
# are doing. It's much better to create a custom.sh shell script in
# /etc/profile.d/ to make custom changes to your environment, as this
# will prevent the need for merging in future updates.
export JAVA_HOME=/usr/java/jdk1.8.0_171
export CLASSPATH=$JAVA_HOME/lib/
export PATH=$PATH:$JAVA_HOME/bin
export PATH JAVA_HOME CLASSPATH

export ZOOKEEPER_HOME=/usr/zookeeper/zookeeper-3.4.10
PATH=$PATH:$ZOOKEEPER_HOME/bin

pathmunge () {
    case ":${PATH}:" in
        *:"$1":*)
            ;;
        *)
            if [ "$2" = "after" ] ; then
                PATH=$PATH:$1
            else
                PATH=$1:$PATH
            fi
    esac
}
```

ZK环境变量配置

生效：source /etc/profile

（9）启动 ZooKeeper 集群

在 ZooKeeper 集群的每个结点上，执行启动 ZooKeeper 服务的脚本，如下所示：

```
bin/zkServer.sh start
bin/zkServer.sh status
```

```
[root@master zookeeper-3.4.10]# bin/zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/zookeeper-3.4.10/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[root@master zookeeper-3.4.10]# bin/zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/zookeeper-3.4.10/bin/../conf/zoo.cfg
Mode: follower
[root@master zookeeper-3.4.10]#
```

启动ZK

查看启动状态

启动状态为跟随者

```
[root@slave1 zookeeper-3.4.10]# bin/zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/zookeeper-3.4.10/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[root@slave1 zookeeper-3.4.10]# bin/zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/zookeeper-3.4.10/bin/../conf/zoo.cfg
Mode: leader
[root@slave1 zookeeper-3.4.10]#
```

启动zk

查看状态

启动状态为领导者

```
[root@slave2 zookeeper-3.4.10]# bin/zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/zookeeper-3.4.10/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[root@slave2 zookeeper-3.4.10]# bin/zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/zookeeper-3.4.10/bin/../conf/zoo.cfg
Mode: follower
[root@slave2 zookeeper-3.4.10]#
```

启动zk

查看状态

同样为跟随者

通过上面状态查询结果可见，一个节点是 Leader，其余的结点是 Follower。

3.2 安装 hadoop

（1）创建对应工作目录/usr/hadoop：

```
[root@master soft]# cd /usr/
[root@master usr]# ls
bin  etc  games  include  java  lib  lib64  libexec  local  sbin  share  src  tmp
[root@master usr]# mkdir hadoop
[root@master usr]# ls
bin  etc  games  hadoop  include  java  lib  lib64  libexec  local  sbin  share  src  tmp
[root@master usr]#
```

在 /usr/文件夹下创建 hadoop 目录为下一步将hadoop解压到此文件夹做准备

我们创建的hadoop文件夹

解压 hadoop 到相应目录：

```
[root@master usr]# cd /opt/soft/
[root@master soft]# ls
hadoop-2.7.3.tar.gz  hbase-1.2.4-bin.tar.gz  jdk-8u171-linux-x64.tar.gz  zookeeper-3.4.10.tar.gz
[root@master soft]# cp hadoop-2.7.3.tar.gz /usr/hadoop/
[root@master soft]# ls /usr/hadoop/
hadoop-2.7.3.tar.gz
[root@master soft]#
```

打开此文件夹

将hadoop压缩包通过cp命令复制到 /usr/hadoop文件夹下

解压后：

```
[root@master hadoop]# ls
hadoop-2.7.3  hadoop-2.7.3.tar.gz
[root@master hadoop]#
```

解压后的hadoop文件

配置环境变量：

```
vim   /etc/profile

export HADOOP_HOME=/usr/41adoop/41adoop-2.7.3

export CLASSPATH=$CLASSPATH:$HADOOP_HOME/lib

export PATH=$PATH:$HADOOP_HOME/bin
```

```
# # # JAVA
export JAVA_HOME=/usr/java/jdk1.8.0_171
export CLASSPATH=$JAVA_HOME/lib/
export PATH=$PATH:$JAVA_HOME/bin
export PATH JAVA_HOME CLASSPATH
# # # HADOOP
export HADOOP_HOME=/usr/hadoop/hadoop-2.7.3
export CLASSPATH=$CLASSPATH:$HADOOP_HOME/lib
export PATH=$PATH:$HADOOP_HOME/bin

pathmunge () {
    case ":${PATH}:" in
        *:"$1":*)
            ;;
        *)
            if [ "$2" = "after" ] ; then
                PATH=$PATH:$1
            else
                PATH=$1:$PATH
            fi
    esac
}


if [ -x /usr/bin/id ]; then
    if [ -z "$EUID" ]; then
        # ksh workaround
        EUID=`/usr/bin/id -u`
        UID=`/usr/bin/id -ru`
    fi
    USER="`/usr/bin/id -un`"
    LOGNAME=$USER
    MAIL="/var/spool/mail/$USER"
fi
:wq
```

配置hadoop环境变量

使用以下命令使 profile 生效：

source /etc/profile

（2）编辑 hadoop 环境配置文件 hadoop-env.sh

```
[root@master hadoop-2.7.3]# cd etc
[root@master etc]# ls
hadoop
[root@master etc]# cd hadoop/
[root@master hadoop]# ls
capacity-scheduler.xml  hadoop-env.sh               httpfs-env.sh            kms-env.sh            mapred-env.sh              ssl-server.xml.example
configuration.xsl       hadoop-metrics2.properties  httpfs-log4j.properties  kms-log4j.properties  mapred-queues.xml.template  yarn-env.cmd
container-executor.cfg  hadoop-metrics.properties   httpfs-signature.secret  kms-site.xml          mapred-site.xml.template    yarn-env.sh
core-site.xml           hadoop-policy.xml           httpfs-site.xml          log4j.properties      slaves                      yarn-site.xml
hadoop-env.cmd          hdfs-site.xml               kms-acls.xml             mapred-env.cmd        ssl-client.xml.example
[root@master hadoop]#
```

进入 hadoop 2.7.3 --> etc----> hadoop

使用vim命令编辑 hadoop-env.sh文件

输入内容：export JAVA_HOME=/usr/java/jdk1.8.0_171

```
# The java implementation to use.
export JAVA_HOME=${JAVA_HOME}

#The jsvc implementation to use. Jsvc is required to run secure datanodes
export JAVA_HOME=/usr/java/jdk1.8.0_171
# that bind to privileged ports to provide authentication of data transfer
# protocol.  Jsvc is not required if SASL is configured for authentication of
# data transfer protocol using non-privileged ports.
#export JSVC_HOME=${JSVC_HOME}

export HADOOP_CONF_DIR=${HADOOP_CONF_DIR:-"/etc/hadoop"}

# Extra Java CLASSPATH elements.  Automatically insert capacity-scheduler.
for f in $HADOOP_HOME/contrib/capacity-scheduler/*.jar; do
  if [ "$HADOOP_CLASSPATH" ]; then
    export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$f
  else
    export HADOOP_CLASSPATH=$f
  fi
done
:wq
```

添加java环境变量，然后保存退出

（3） core-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
  <name>fs.default.name</name>
   <value>hdfs://master:9000</value>
</property>
<property>
  <name>hadoop.tmp.dir</name>
   <value>/usr/hadoop/hadoop-2.7.3/hdfs/tmp</value>
<description>A base for other temporary directories.</description>
</property>
<property>
  <name>io.file.buffer.size</name>
   <value>131072</value>
</property>
<property>
  <name>fs.checkpoint.period</name>
   <value>60</value>
</property>
<property>
  <name>fs.checkpoint.size</name>
   <value>67108864</value>
</property>
</configuration>
~
~
~
```

（3） yarn-site.xml

```
<configuration>

 <property>

 <name>yarn.resourcemanager.address</name>
```

```
    <value>master:18040</value>
  </property>
  <property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>master:18030</value>
  </property>
  <property>
    <name>yarn.resourcemanager.webapp.address</name>
    <value>master:18088</value>
  </property>
  <property>
    <name>yarn.resourcemanager.resource-tracker.address</name>
    <value>master:18025</value>
  </property>
  <property>
   <name>yarn.resourcemanager.admin.address</name>
   <value>master:18141</value>
  </property>
  <property>
   <name>yarn.nodemanager.aux-services</name>
   <value>mapreduce_shuffle</value>
  </property>
  <property>
 <name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>

 <!-Site specific YARN configuration properties →
```

```
</configuration>
```

```
<?xml version="1.0"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->
<configuration>
<property>
 <name>yarn.resourcemanager.address</name>
   <value>master:18040</value>
 </property>
 <property>
   <name>yarn.resourcemanager.scheduler.address</name>
   <value>master:18030</value>
 </property>
 <property>
   <name>yarn.resourcemanager.webapp.address</name>
   <value>master:18088</value>
 </property>
 <property>
   <name>yarn.resourcemanager.resource-tracker.address</name>
   <value>master:18025</value>
 </property>
 <property>
  <name>yarn.resourcemanager.admin.address</name>
  <value>master:18141</value>
 </property>
 <property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
 </property>
 <property>
  <name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
 </property>

<!-- Site specific YARN configuration properties -->

</configuration>
~
~
```

（3） 编写 slavs 文件

```
slave1
slave2
~
```

（6）master 文件

（7）hdfs-site.xml

```
<configuration>
 <property>
<name>dfs.replication</name>
  <value>2</value>
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>file:/usr/47adoop/47adoop-2.7.3/hdfs/name</value>
  <final>true</final>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file:/usr/47adoop/47adoop-2.7.3/hdfs/data</value>
  <final>true</final>
</property>
<property>
 <name>dfs.namenode.secondary.http-address</name>
  <value>master:9001</value>
</property>
<property>
  <name>dfs.webhdfs.enabled</name>
  <value>true</value>
</property>
<property>
```

```
    <name>dfs.permissions</name>
    <value>false</value>
  </property>
</configuration>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
 <name>dfs.replication</name>
   <value>2</value>
 </property>
 <property>
   <name>dfs.namenode.name.dir</name>
   <value>file:/usr/hadoop/hadoop-2.7.3/hdfs/name</value>
   <final>true</final>
</property>
 <property>
   <name>dfs.datanode.data.dir</name>
   <value>file:/usr/hadoop/hadoop-2.7.3/hdfs/data</value>
   <final>true</final>
 </property>
 <property>
  <name>dfs.namenode.secondary.http-address</name>
   <value>master:9001</value>
 </property>
 <property>
   <name>dfs.webhdfs.enabled</name>
   <value>true</value>
 </property>
 <property>
   <name>dfs.permissions</name>
   <value>false</value>
 </property>

</configuration>
~
```

（8） mapred-site.xml

首先将模板文件复制为 xml 文件，对其进行编辑：

```
[root@master hadoop]# cp mapred-site.xml.template mapred-site.xml
[root@master hadoop]# vim mapred-site.xml
[root@master hadoop]#
```

```
<property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
 </property>
```

（9）分发 hadoop：

scp -r /usr/49adoop root@slave1:/usr/

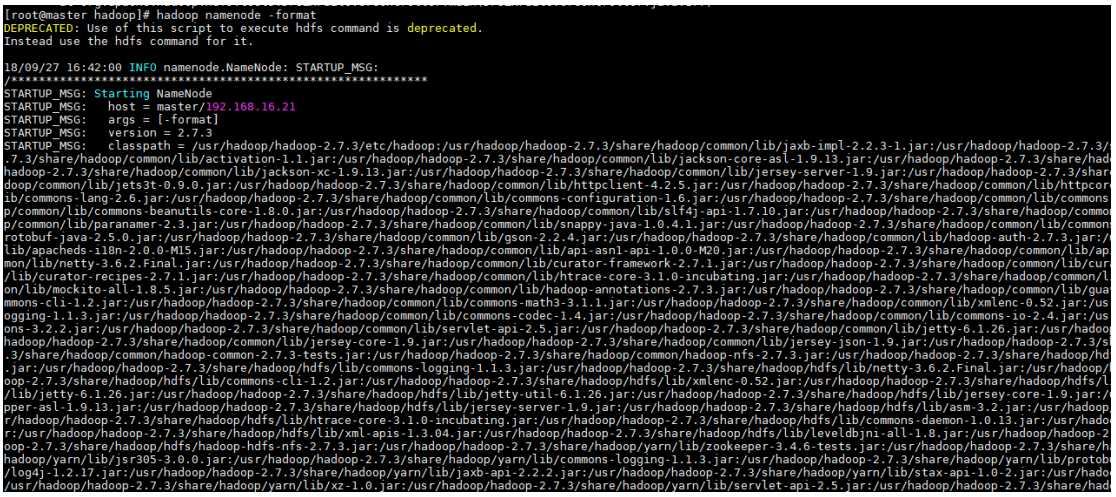
scp -r /usr/49adoop root@slave2:/usr/

```
[root@master usr]# scp -r /usr/hadoop root@slave1:/usr/
```

注意：slave 节点上还需要配置环境变量，参考 hadoop 中第一个步骤。

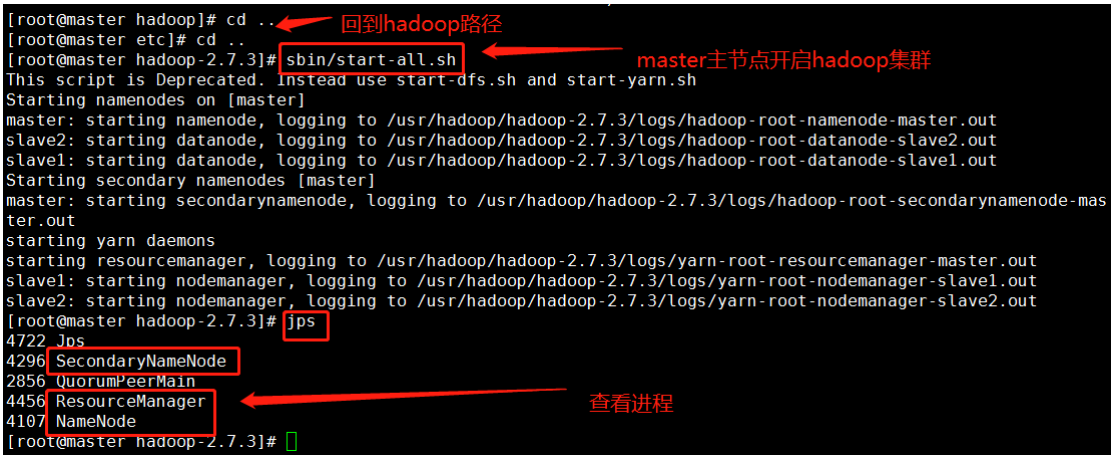
（10）master 中格式化 hadoop

hadoo namenode -format



（11）各节点进行如下：

master：



slave1：

```
[root@slave1 hadoop]# jps
3570 DataNode
3782 Jps
2519 QuorumPeerMain
3671 NodeManager
[root@slave1 hadoop]#
```

子节点中进程

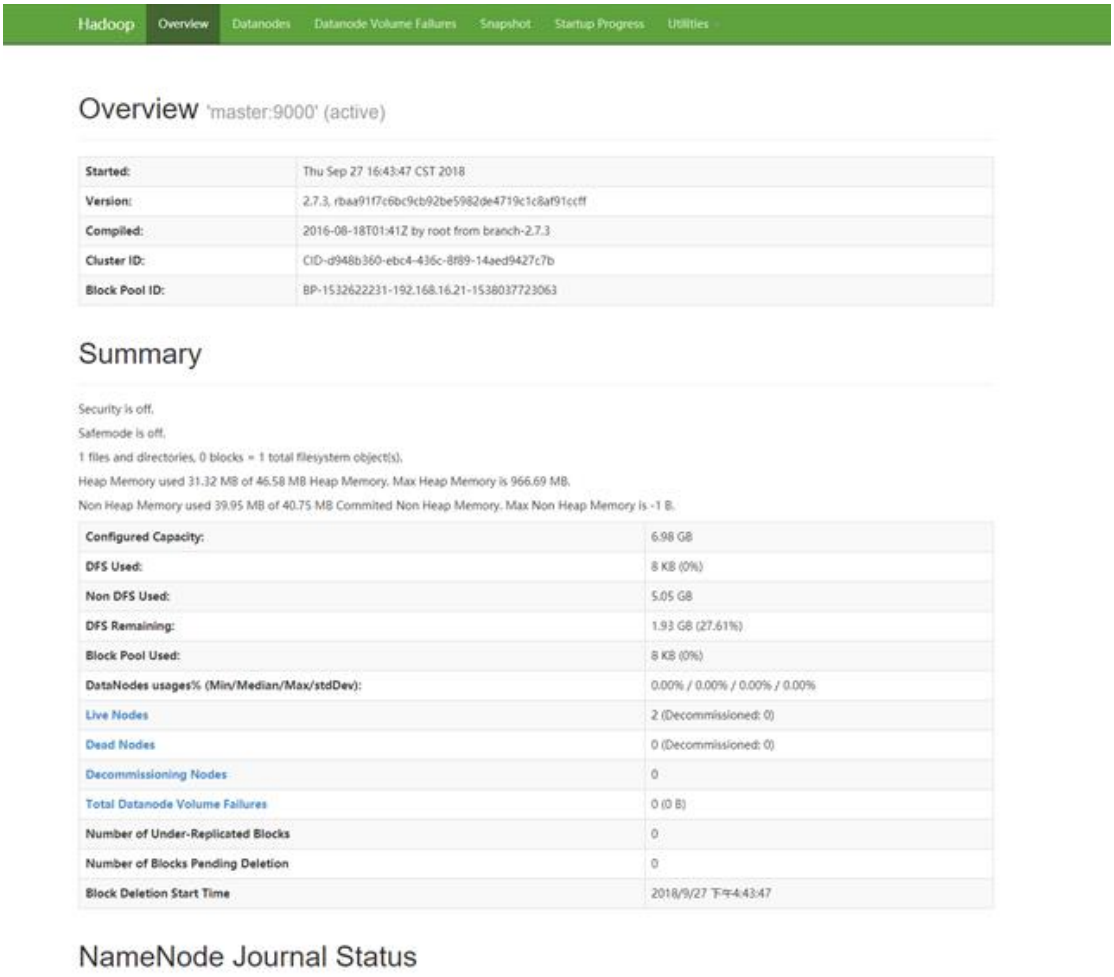
slave2：

```
[root@slave2 hadoop]# jps
2547 QuorumPeerMain
3603 DataNode
3815 Jps
3704 NodeManager
[root@slave2 hadoop]#
```

子节点slave2

访问主节点 master：50070（50070 是 hdfs 的 web 管理页面）

注意，如果发现集群已启动，但是访问不了，可能是防火墙没有关闭。



（12）查看 hdfs

Hadoop fs -ls /　　（最开始创建的是一个空的文件系统所以什么也没有）

Hadoop fs -mkdir /a （在 hdfs 上传到 a 文件夹）

```
[root@slave1 ~]# hadoop fs -ls /
[root@slave1 ~]# hadoop fs -mkdir /a
[root@slave1 ~]# hadoop fs -ls /
Found 1 items
drwxr-xr-x   - root supergroup          0 2018-09-27 17:06 /a
[root@slave1 ~]#
```

3.3 安装 spark

（1）安装 scala 环境

我们需要在拥有 hadoop 集群的所有节点中安装 scala 语言环境，因为 spark 的源代码为 scala 语言所编写，所以接下来我们进行安装 scala。

1.解压 scala 的 tar 包：

首先我们进入到本系统的`/opt/soft`路径下可以看到我们所提供的 scala 安装包，接下来我们在`/usr/`下创建 scala 文件夹，然后解压 scala 到我们所创建的 scala 工作路径中，具体 操

```
[root@master ~]# cd /opt/soft/
[root@master soft]# ls
apache-hive-2.1.1-bin.tar.gz  hbase-1.2.4-bin.tar.gz      scala-2.11.12.tgz            zookeeper-3.4.10.tar.gz
hadoop-2.7.3.tar.gz           jdk-8u171-linux-x64.tar.gz  spark-2.4.0-bin-hadoop2.7.tgz
[root@master soft]# mkdir -p /usr/scala
[root@master soft]# tar -zxvf scala-2.11.12.tgz -C /usr/scala/
scala-2.11.12/
scala-2.11.12/lib/
scala-2.11.12/lib/akka-actor_2.11-2.3.16.jar
scala-2.11.12/lib/scala-reflect.jar
scala-2.11.12/lib/config-1.2.1.jar
scala-2.11.12/lib/scala-continuations-plugin_2.11.12-1.0.2.jar
scala-2.11.12/lib/scala-parser-combinators_2.11-1.0.4.jar
scala-2.11.12/lib/scala-swing_2.11-1.0.2.jar
scala-2.11.12/lib/scala-compiler.jar
scala-2.11.12/lib/scala-actors-migration_2.11-1.1.0.jar
scala-2.11.12/lib/scalap-2.11.12.jar
scala-2.11.12/lib/scala-library.jar
scala-2.11.12/lib/jline-2.14.3.jar
scala-2.11.12/lib/scala-xml_2.11-1.0.5.jar
scala-2.11.12/lib/scala-continuations-library_2.11-1.0.2.jar
scala-2.11.12/lib/scala-actors-2.11.0.jar
scala-2.11.12/bin/
scala-2.11.12/bin/scala
scala-2.11.12/bin/scalac.bat
scala-2.11.12/bin/scala.bat
scala-2.11.12/bin/scalap
scala-2.11.12/bin/scalap.bat
scala-2.11.12/bin/scaladoc.bat
scala-2.11.12/bin/fsc
scala-2.11.12/bin/fsc.bat
scala-2.11.12/bin/scalac
scala-2.11.12/bin/scaladoc
scala-2.11.12/man/
scala-2.11.12/man/man1/
```

打开/opt/soft

创建一个scala的工作路径

解压scala到我们所创建的scala工作路径中

2.配置 scala 的环境变量：

当我们解压好 scala 安装包之后，我们需要对 scala 进行配置环境变量，我们需要将环境变量配置到`/etc/profile`文件中，首先我们进入 scala 的工作路径，然后使用`pwd`命令进行查看 scala 的安装路径，接下来就可以复制此路径到我们的 profile 文件中了，具体操作如下图所示：

```
[root@master scala]# cd /usr/scala/scala-2.11.12/
[root@master scala-2.11.12]# ls
bin  doc  lib  man
[root@master scala-2.11.12]# pwd
/usr/scala/scala-2.11.12
[root@master scala-2.11.12]# vim /etc/profile
profile    profile.d/
[root@master scala-2.11.12]# vim /etc/profile
```

进入到scala工作路径

查看当前路径

使用Vim编辑环境变量文件

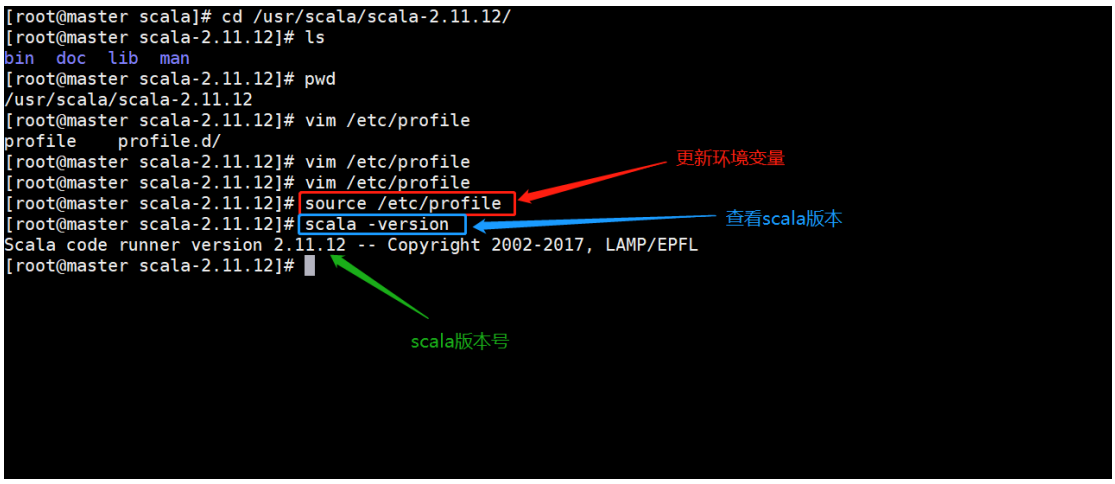
接下来我们就可以配置环境变量了，使用`vim /etc/profile`命令去配置 scala 的环境变量，具体操作如下图所示：

```
# /etc/profile

# System wide environment and startup programs, for login setup
# Functions and aliases go in /etc/bashrc

# It's NOT a good idea to change this file unless you know what you
# are doing. It's much better to create a custom.sh shell script in
# /etc/profile.d/ to make custom changes to your environment, as this
# will prevent the need for merging in future updates.
export JAVA_HOME=/usr/java/jdk1.8.0_171
export CLASSPATH=$JAVA_HOME/lib/
export PATH=$PATH:$JAVA_HOME/bin
export PATH JAVA_HOME CLASSPATH
#set zookeeper environment
export ZOOKEEPER_HOME=/usr/zookeeper/zookeeper-3.4.10
PATH=$PATH:$ZOOKEEPER_HOME/bin
# # # HADOOP
export HADOOP_HOME=/usr/hadoop/hadoop-2.7.3
export CLASSPATH=$CLASSPATH:$HADOOP_HOME/lib
export PATH=$PATH:$HADOOP_HOME/bin


# # # HADOOP
export HADOOP_HOME=/usr/hadoop/hadoop-2.7.3
export CLASSPATH=$CLASSPATH:$HADOOP_HOME/lib
export PATH=$PATH:$HADOOP_HOME/bin
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop

# set hbase environment
export HBASE_HOME=/usr/hbase/hbase-1.2.4
export PATH=$PATH:$HBASE_HOME/bin
PATH=$PATH:$ZOOKEEPER_HOME/bin

#set HIVE
export HIVE_HOME=/usr/hive/apache-hive-2.1.1-bin
export PATH=$PATH:$HIVE_HOME/bin

#Scala Home
export SCALA_HOME=/usr/scala/scala-2.11.12
export PATH=$SCALA_HOME/bin:$PATH

pathmunge () {
    case ":${PATH}:" in
        *:"$1":*)
            ;;
        *)
            if [ "$2" = "after" ] ; then
                PATH=$PATH:$1
            else
                PATH=$1:$PATH
            fi
-- INSERT --
```

SCALA_HOME指向我们的scala安装目录

将scala/bin配置到path中

当我们配置好环境变量之后我们需要使用 source 命令去更新我们的环境变量文件，最后我们使用`scala -version`查看我们的 scala 是否安装成功，具体操作如下图所示：



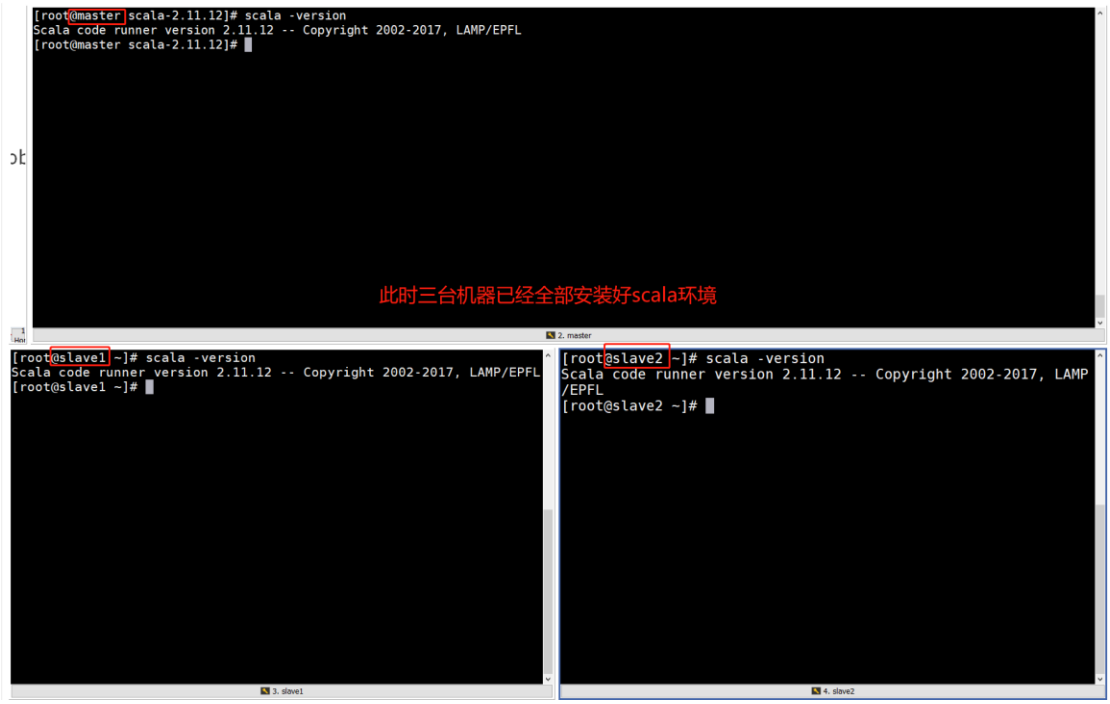
3.复制 scala 到子节点：

因为我们是集群环境，所以接下来我们需要将我们的 scala 环境发送到我们的其他子节点上，具体操作如下图所示：

命令：`scp -r /usr/scala root@slave1:/usr/`

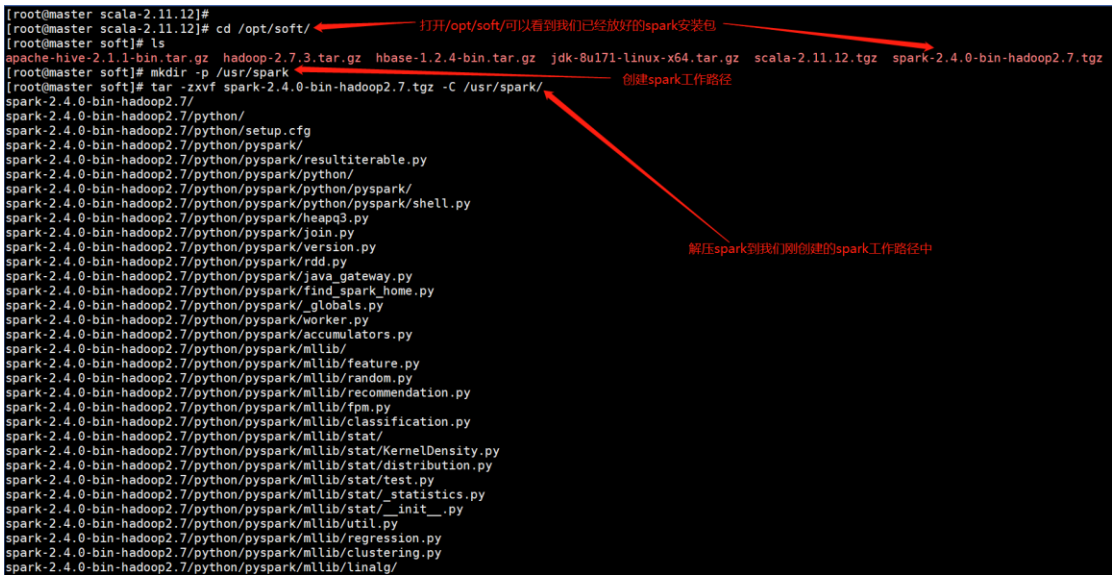
注：图中只展示了复制到 slave1 中的操作，slave2 的操作同理，请同学们自行操作。

切换 slave1 和 slave2 节点去编写环境变量将 scala 环境变量填进去，然后更新环境变量，操作和 master 节点操作一样，这里就不赘述了。当环境变量配置成功后我们需要检测每个节点的 scala 环境是否安装成功，具体操作如下图所示：



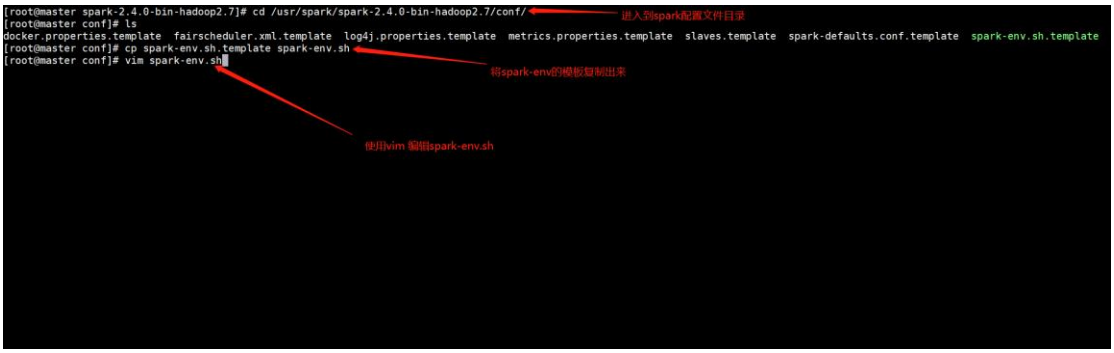
（2）解压 spark 的 tar 包

首先我们进入到本系统的`/opt/soft`路径下可以看到我们所提供的 spark 安装包，接下来我们在`/usr/`下创建 spark 文件夹，然后解压 spark 到我们所创建的 spark 工作路径中，具体 操作如下图所示：



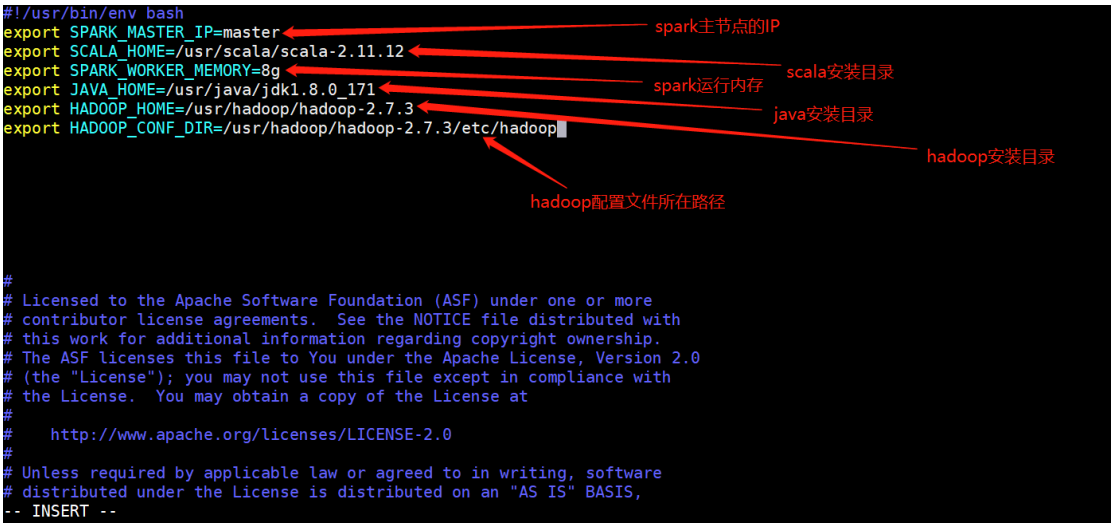
1.复制 spark-env.sh 模板

我们需要将 spark-env.sh.template 复制为 spark-env.sh，命令为：`cp spark-env.sh.template spark-env.sh`.当复制出 spark-env.sh 文件后我们可以使用 vim 进行编译，具体操作如下图所示：



2.配置 spark-env.sh 文件，并添加一下内容，具体操作如下图所示：

```
export SPARK_MASTER_IP=master
export SCALA_HOME=/usr/scala/scala-2.11.12
export SPARK_WORKER_MEMORY=8g
export JAVA_HOME=/usr/java/jdk1.8.0_171
export HADOOP_HOME=/usr/hadoop/hadoop-2.7.3
export HADOOP_CONF_DIR=/usr/hadoop/hadoop-2.7.3/etc/Hadoop
```



3.配置 spark 从节点，修改 slaves 文件

命令：cp slaves.template.template slaves

使用 vim 命令编辑 slaves，其内容如下图所示：

```
#
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#

# A Spark Worker will be started on each of the machines listed below.
slave1
slave2
~                                   添加spark的工作节点
~
~
~
~
~
-- INSERT --
```

4.配置 spark 环境变量

命令：vim /etc/profile

在其中添加如下内容：

```
export SPARK_HOME=/usr/spark/spark-2.4.0-bin-hadoop2.7
export PATH=$SPARK_HOME/bin:$PATH
```

```
#set HIVE
export HIVE_HOME=/usr/hive/apache-hive-2.1.1-bin
export PATH=$PATH:$HIVE_HOME/bin

#Scala Home
export SCALA_HOME=/usr/scala/scala-2.11.12
export PATH=$SCALA_HOME/bin:$PATH

#Spark Home

export SPARK_HOME=/usr/spark/spark-2.4.0-bin-hadoop2.7
export PATH=$SPARK_HOME/bin:$PATH


pathmunge () {
    case ":${PATH}:" in
        *:"$1":*)
            ;;
        *)
            if [ "$2" = "after" ] ; then
                PATH=$PATH:$1
            else
                PATH=$1:$PATH
            fi
-- INSERT --
```

添加spark环境变量

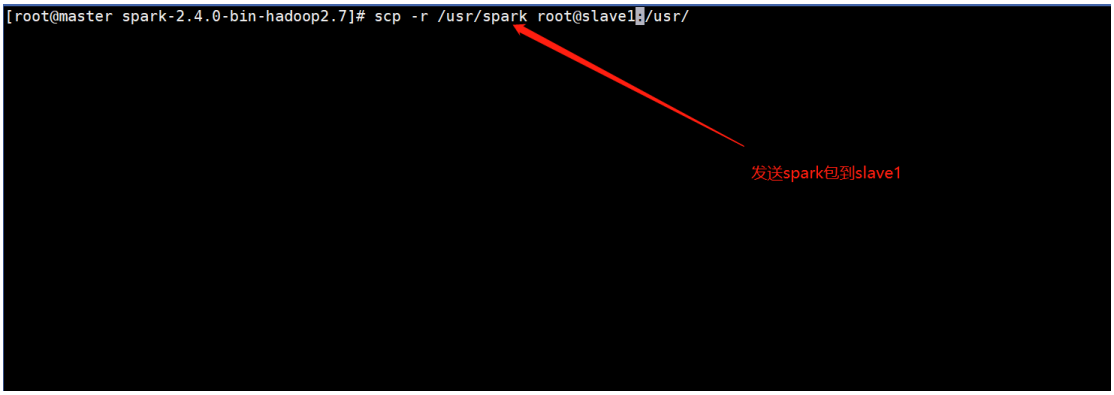
使环境变量生效：`source /etc/profile`

（3）接下来向所有子节点发送 spark 配置好的安装包，具体操作如下图所示：

注：slave2 同理，请同学们自己进行操作。

命令：scp -r /usr/spark root@slave1:/usr/

命令：scp -r /usr/spark root@slave2:/usr/

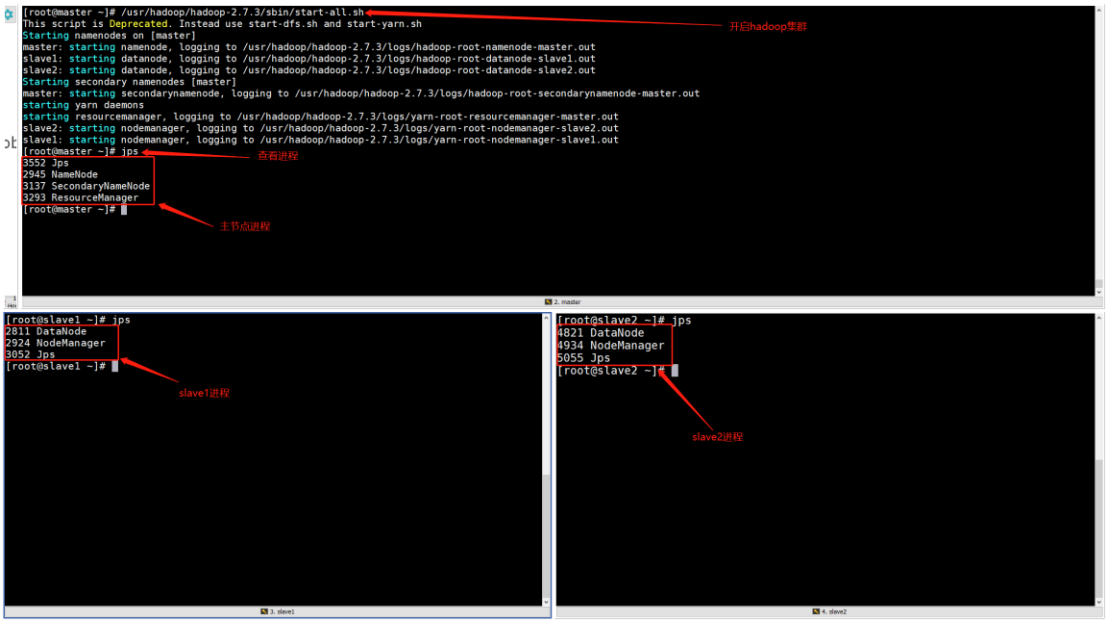


修改 slave1 和 slave2 的环境变量，此步骤和修改 master 中 spark 环境变量相同，这里就不多介绍了，最后记得是环境变量生效。这时我们的 spark 环境就安装成功了。

（4）测试 spark 环境

因为我们安装的是 spark on yarn 模式，所有接下来我们需要开启 hadoop 环境，我们只需要在 master 节点上执行如下命令即可开启 hadoop 集群。

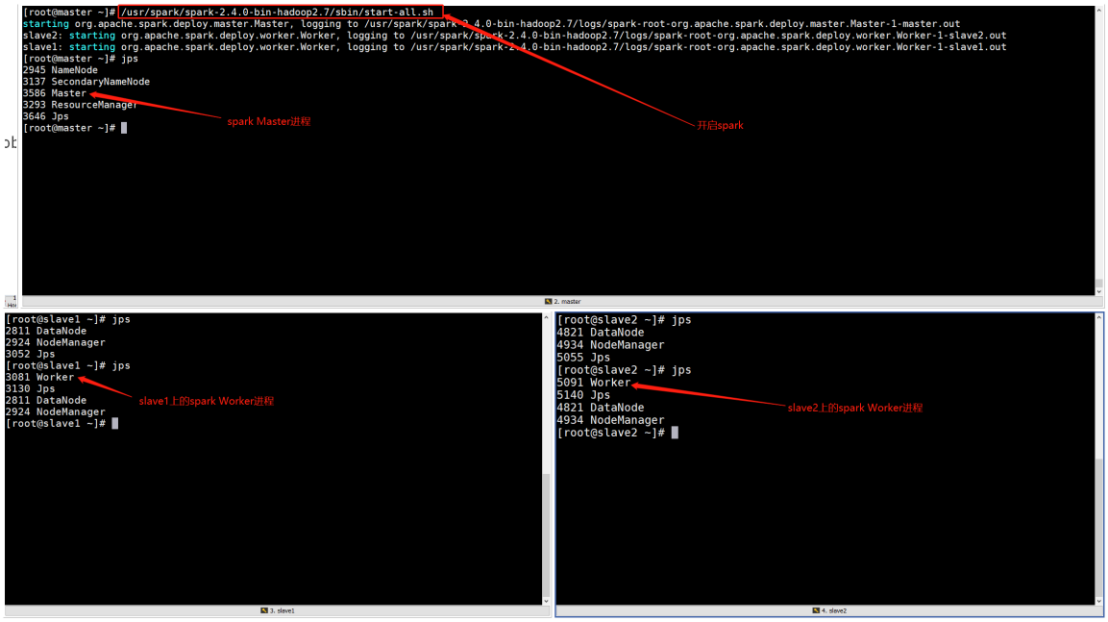
命令：/usr/hadoop/hadoop-2.7.3/sbin/start-all.sh



（5）开启 spark 集群

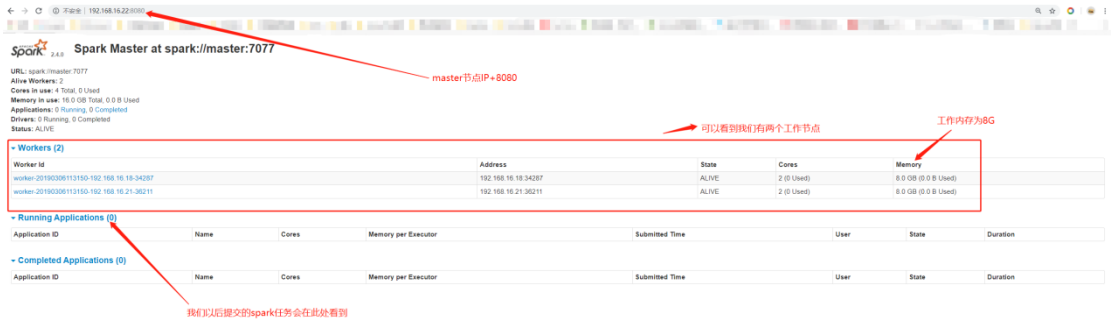
我们只需要在 master 节点上执行如下命令即可。

命令：/usr/spark/spark-2.4.0-bin-hadoop2.7/sbin/start-all.sh



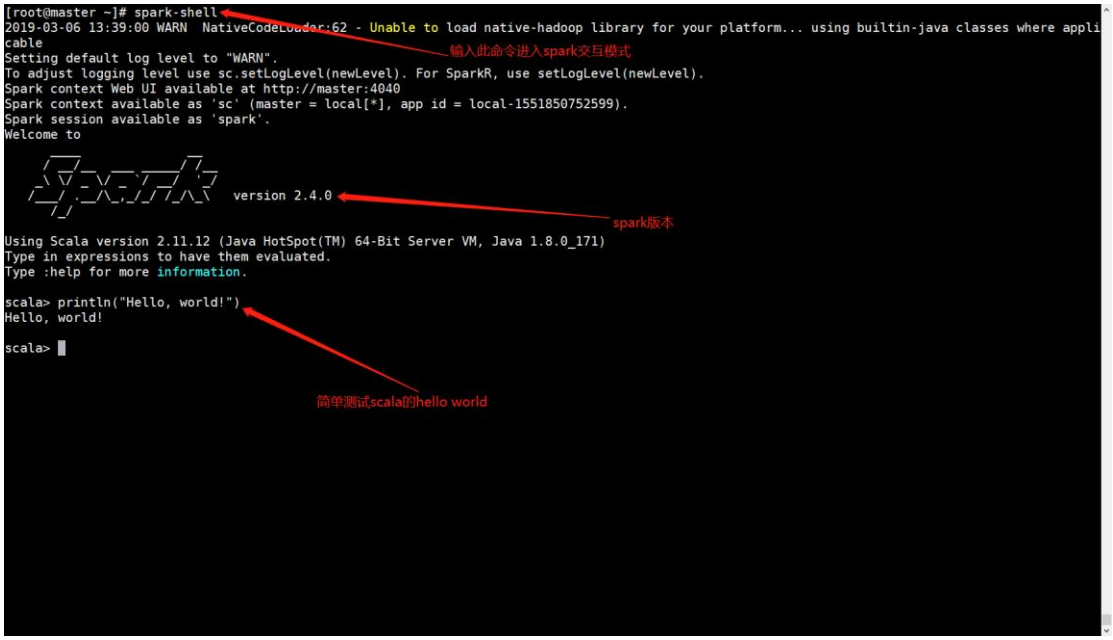
（6）访问 SparkWeb 界面

我们可以在浏览器中输入我们 master 节点的 IP 地址，端口号为 8080 具体操作如下图所示：



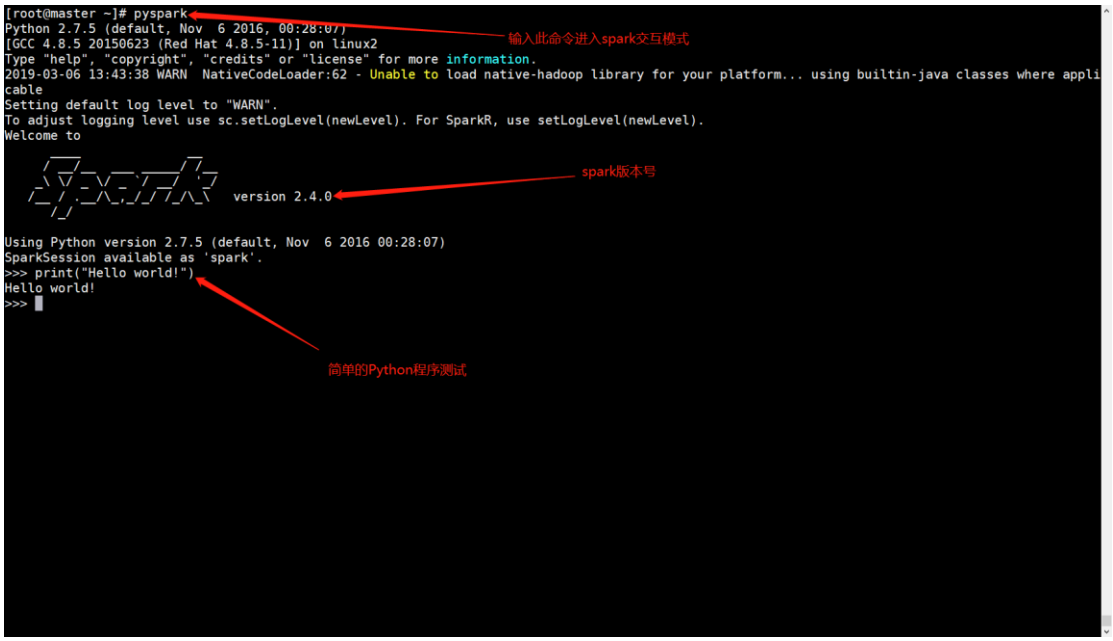
接下来我们开启我们的 spark-shell 以及 pyspark 进入到 spark 的交互模式：

首先 spark-shell 此时进入的是 scala 环境的 spark 交互模式，具体操作如下图所示：



接下来我们输入命令进入 python 环境下的 spark 交互模式，具体操作如下图所示：

命令：pyspatk



到此我们的 spark on yarn 环境就已经搭建成功了！