Hadoop2 学习笔记

简介与课程大纲

此部分是基于**徐培成大数据视频**学习hadoop2. 在此之前，已经根据慕课网学习了hadoop1的基础知识，算是在hadoop门前“转了”一下。这个学习的过程非常的短暂，因为视频也非常的少。

之后根据云帆大数据公开的hadoop1视频资源【云帆大数据\_Hadoop从入门到上手企业开发】共90集视频，以hadoop1.2.1为教材，学习了HDFS文件系统以及对应的API操作HDFS文件、MapReduce框架以及对应的MapReduce编程模型，并且编写了单词统计、手机流量分析、TopK算法等MapReduce程序，同时整理除了MapReduce的编程框架。

第三次学习hadoop是也是根据【云帆大数据\_Hadoop2】共19集视频，在Hadoop1的基础之上，学习Hadoop2.这次学习以Hadoop2.6.4为教材，主要是学习了Yarn框架的知识。

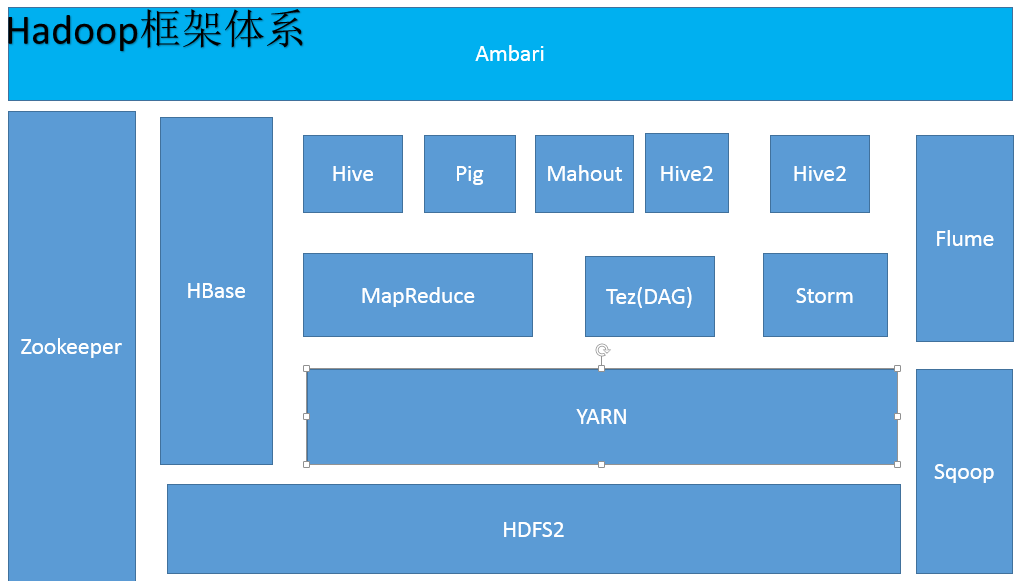
**大数据生态系统**

大数据解决的是海量数据的两部分核心问题：存储和计算，核心思想就是分布式。

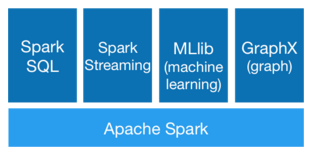
1.1操作系统：主要是linux操作系统，Ubuntu（学习）、CentOS（实际的部署）

1.2 JAVA环境：Java是大数据框架运行的环境，这里推荐使用Oracle的Java8

1.3 Hadoop生态圈：



1.4 Spark生态系统：



Spark SQL：操作结构化数据的模块，混合SQL语句插叙到Spark程序中

Spark Streaming:实时流的计算

Mlib：机器学习的框架和算法

GraphX：是Spark针对图和图并行计算的API,可以与专业的图形计算相媲美

1.5 机器学习和决策：大数据的实际市场应用

01课时 Hadoop 介绍

Hadoop现状

Yahoo：4.2W节点、10WCPU内核用于广告、用户行为分析、反垃圾邮件

FaceBook：1400台/1.12W CPU/ 15PB 数据

百度：单机群2800节点啊、上W台主机 100PB数据

阿里巴巴：3200节点/3W core、100Tmemory/60PB data，用于淘宝、天猫、支付宝等

腾讯：500/单群2000,用于游戏、QQ、财付通。

Hadoop核心问题——海量数据

存储：分割数据，分布式存储 HDFS文件系统

运算：MapReduce计算框架，map 和reduce函数模型实现分布式计算

02课时 Vmware安装Ubuntu操作系统

03课时 Ubuntu下载与虚拟机下安装

推荐下载Server版本,自己之前没有安装过Server版本的Ubuntu，使用Ubuntu16.04尝试安装Desk版本和Server版本。

Server安装不成功

Desk安装16.04 界面更加华丽了

04课时 Ubuntu常用命令

gzip

gunzip

ps –Af

mount sourcedev destdir

unmount destdir

常见的网络命令：

ifconfig

ping

hostname

查找 find 目录名称 | grep <string>

uname -a 查看操作系统的详细信息

file filename #查看文件属性

05课时 Ubuntu目录权限

Linux 文件类型： d 目录 - 文件 l link文件 b block文件

三种身份，对应不同的操作权限：u g o a(所有)

文件拥有者（user）

拥有者所在的组(group)

其他用户(other)

改变文件的权限 chomd a+r file

06课时 Ubuntu软件包桌面程序以及增强工具

apt： advanced packageing tool

软件源的更新：163

sudo apt-cache search ubuntu-desktop

sudo apt-get install ubuntu-desktop

sudo apt-get remove xxx

sudo apt-get purge xxx #彻底删除软件

sudo dpkg -l |grep XXX #查看已经安装的有哪些程序

虚拟机的Ubuntu桌面的增强工具：Vmware Tools

和宿主机之间实现文件共享：在/mnt/hgfs就是共享的文件夹

鼠标的自动捕捉和释放

高分屏

自动调整窗口大小

07课时 Ubuntu本地软件源与ISO制作

1. 制作本地源

sudo apt-get install dpkg-dev

cd /var/cahce/apt/archives

ls -al | head -10 #查看文件数量

mkdir ~/software

cp –R –v /var/cahce/apt/archives/\* ~/software

sudo dpkg-scanpackages ~/software /dev/null | gzip > ~/software/Packages.gz

sudo mkdir /mnt/hgfs/share/repo

sudo mv \* /mnt/hgfs/share/repo#移动文件到这个目录

这样新的客户机便可以使用该软件源

在/etc/apt-source.list 添加如下的内容

deb file:/home/yang/software

sudo apt-get update

sudo apt-get upgrade

sudo apt-get install xxx

1. 制作ISO文件

如果不安装增强工具，我们把软件源制作成为ISO文件，然后通过光驱加载文件。

sudo dpkg -l |grep mkisofs

genisoimage - Creates ISO-9660 CD-ROM filesystem images

xorriso - command line ISO-9660 and Rock Ridge manipulation tool

sudo apt-get install genisoimage

sudo mkisofs -r –o XXX.iso

sudo mkisofs –r[**保留原文件**] -o MyRepo.iso repo/ #生成一个iso文件

然后就可以通过光驱挂在ISO文件

在/etc/apt/source.list中修改软件的更新源为本地的文件，事先需要mount光驱到指定的目录。

08课时 Ubuntu克隆技术

虚拟机的克隆技术在VMWare提供克隆技术

克隆的技术不同的是MAC地址修改了,其他的内容相同。

网络连接模式：

1. 桥接模式：在网络中的真实主机，可以访问互联网，也可以设置IP和互相访问，若没有网络，主机之间是不互通的

2. NAT模式：网络地址转换，Client和宿主机之间形成一个网络，客户机可以访问互联网，由宿主机做DNS和NAT转换，IP是由宿主机分配的，客户机无法访问网络中的其他主机。在没有网络的环境下，宿主机和客户机是可以互通的，但是客户机不能够修改IP地址。

3. Only host：和NAT非常相似，但是不允许访问外网

09课时 Ubuntu安装JDK

使用默认的JDK安装方式，会将不同的脚本配置文件放在不同的目录，造成Hadoop不能够识别，因此我们采取手动的安装方式。这里把脚本复制到这里来：

|  |
| --- |
| sudo tar zxvf jdk\*.tar.gz -C /usr/lib/jvm  cd /usr/lib/jvm  sudo ln -s jdk\*/ java8  vim /etc/profile  add text at the end of the file  #java environment  export JAVA\_HOME=/usr/lib/jvm/java8  export JRE\_HOME=$JAVA\_HOME/jre  export CLASSPATH=$JAVA\_HOME/lib:$JRE\_HOME/lib:$CLASSPATH  export PATH=$JAVA\_HOME/bin:$JRE\_HOME/bin:$PATH  soure /etc/profile  sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/java8/bin/java 300  sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/java8/bin/javac 300  sudo update-alternatives --install /usr/bin/javah javah /usr/lib/jvm/java8/bin/javah 300  sudo update-laternatives --install /usr/bin/jar jar /usr/lib/jv,/java8/bin/jar 300 |

10课时 Hadoop安装

配置HADOP\_HOME环境变量，添加PATH

配置到/etc/profile中

11课时 Hadoop配置-Standalone

Standalone (or local) mode

There are no daemons running and everything runs in a single JVM. Standalone

mode is suitable for running MapReduce programs during development, since it is

easy to test and debug them.

In standalone mode, there is no further action to take, since the default properties are

set for standalone mode and there are no daemons to run. 摘自hadoop权威指南第四版

独立模式：不需做任何事情，解压之后就是安装完成了，默认使用的系统的文件系统。使用hadoop命令查看文件系统： hadoop dfs –ls / 显示的是本地操作系统的目录

12课时 Hadoop配置-Pseudodistributed Mode

**Pseudodistributed Mode**

In pseudodistributed mode, the configuration files should be created with the following

contents and placed in the etc/hadoop directory. Alternatively, you can copy the etc/

hadoop directory to another location, and then place the \*-site.xml configuration files

there. The advantage of this approach is that it separates configuration settings from

the installation files. If you do this, you need to set the **HADOOP\_CONF\_DIR** environment

variable to the alternative location, or make sure you start the daemons with the

--config option 摘自Hadoop权威指南第四版

将Hadoop的配置文件放在专门的目录中，这样我们启动的使用需要实现配置好HADOOP\_CONF\_DIR环境变量。默认是hadoop安装目录下面的/etc/hadoop

1. hadoop-env.sh

配置JAVA\_HOME : export JAVA\_HOME=/usr/lib/jvm/java8

1. yarn-env.sh

export JAVA\_HOME=/usr/lib/jvm/java8

1. mapred-env.sh

export JAVA\_HOME=/usr/lib/jvm/java8

1. core-site.xml

<property>

<name>fs.defaultFS</name>

<value>hdfs://hadoop-yarn:8020</value>

</property>

<property>

<name>hadoop.tmp.dir</name>

<value>/opt/hadoop/tmp</value>

</property>

1. hdfs-site.xml

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

<property>

<name>dfs.permissions</name>

<value>false</value>

</property>

1. mapred-site.xml

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

1. yarn-site.xml

<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</value>

</property>

<property>

<name>yarn.resourcemanager.hostname</name>

<value>hadoop-yarn</value>

</property>

允许多种模式共存

**ssh-sopy-id –i id\_ras.pub yang@ip**

我们在系统中配置了多种方案，所以我们需要设置hadoop的配置目录，两种方式实现：第一种设置HADOOP\_CONF\_DIR环境变量；第二种启动的时候指定参数 –config <configdir>

**hdfs namenode -format**

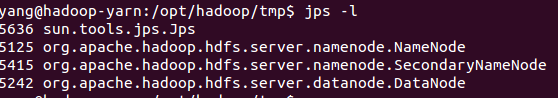
start-dfs.sh --config $HADOOP\_HOME/etc/hadoop\_pseudo

start-yarn.sh --config $HADOOP\_HOME/etc/hadoop\_pseudo

stop-yarn --config $HADOOP\_HOME/etc/hadoop\_pseudo

stop-dfs.sh --config $HADOOP\_HOME/etc/hadoop\_pseudo

使用jps查看服务进程， 详细信息使用 jps –l



我们还可以使用软连接的方式在$HADOOP\_HOME/etc创建hadoop软链接，当使用Standalone模式的时候，指向一个目录hadoop\_alone 或者执行hadoop\_pseudo or hadoop\_cluster

13课时 Hadoop完全分布式模式——修改主机名称和IP

使用Vmware克隆技术克隆完全相同的主机，然后分别修改IP和hostname

14课时 Hadoop完全分布式模式——修改主机名称

1. 修改/etc/hostname文件的内容为hostname，重新启动之后生效
2. 修改IP地址分别配置IP地址
3. 修改hosts信息在/etc/hosts文件，分别添加各个ip 与 主机名称对应的信息



然后将该hosts文件分别复制到不同的主机，使用scp命令方式

15课时 Hadoop完全分布式模式——使用软链接方式实现配置分离

在$HADOOP\_HOME/etc目录下创建hadoop的软链接，指向我们需要的hadoop配置目录。



16课时 Hadoop完全分布式模式——克隆客户机之间ssh与hosts文件拷贝

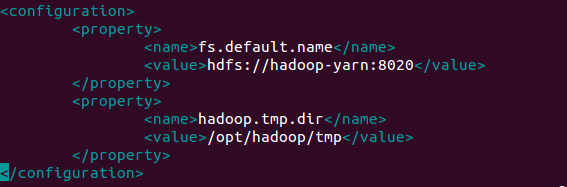
首先要将hosts文件配置好所有主机跟IP对应的关系，并且复制到每一台主机的/etc/目录下，使用scp命令。

完全分布式模式，配置一台主机到其他主机之间的免密登录。

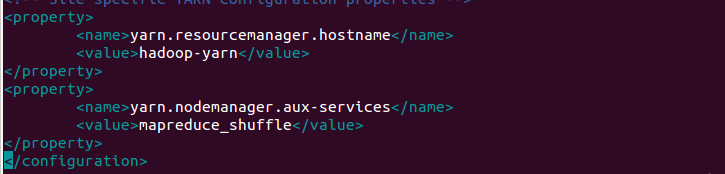
17课时Hadoop完全分布式模式——IP设置远程copy 格式化启动

在pseudodistributed配置文件的基础之上，配置完全分布式模式

1. 配置namenode中配置主机名称在core-site.xml



1. 配置yarn主机地址在yarn-site.xml



1. 配置datanode在slaves文件中

添加DataNode主机节点的名称

1. 配置SecondaryNameNode信息
2. 将hadoop的完全分布式的配置文件复制到所有的节点

scp -r $HADOOP\_HOME/etc/hadoop\_cluster <username>@hostname:/$HADOOP\_HOME/etc

1. 启动整个集群

start-all.sh

或者使用start-dfs.sh start-yarn.sh分别启动

在主节点上只有Namenode ResourceManager

数据节点上进程： DataNode NodeManager

SecondaryNamenode：只有SecondaryNamenode进程

18课时 Hadoop在Windows中的配置——软件安装 SSH 略

19课时 Hadoop在Windows中关于免Cygwin环境安装和dexpot多桌面应用 略

20课时 Hadoop架构分析——单节点体系

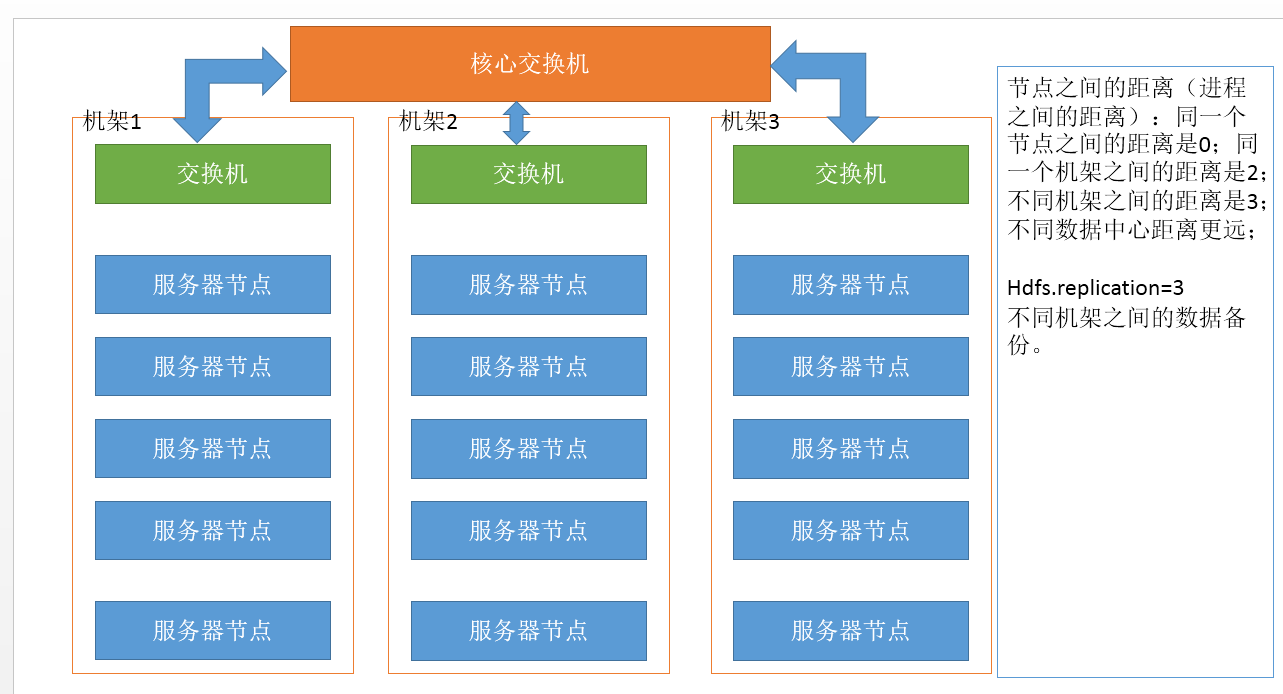
底层硬件：IBM Think HUAWEI

操作系统：linux （Ubuntu CentOS Redhat）

运行环境：Sun JAVA

运行框架：Hadoop

21课时Hadoop架构分析——集群架构分析



22课时Hadoop架构分析——HDFS架构分析

Hadoop架构分析：

HDFS文件系统

定制的MapReduce程序，并且是纯文本文件，大量数据保存

目标在意多次的文件流读取

写入成本很高

高度的数据冗余（dfs.replication=3）

每一个节点不需要RAID-独立磁盘冗余（reduntant array of independent disk）

Block size比较大：默认64M，适合存放大文件，到了Hadoop2使用的是128M

定制节点的位置感知：默认在同一个机架上

RAID -0:至少两个硬盘，条带化存储，写文件的时候将文件打散存储，放到两个盘中，在读数据的时候，提高读数据的速率。(硬RAID：主板实现，软RAID，操作系统层面实现)

RAID-1：一份实时存储，另一份是镜像；当一个硬盘读慢的时候，可以从另一个硬盘中读取。但是这种方式比较浪费硬盘资源。

RRAID-5：至少三块硬盘，第一和第二块硬盘实现条带化存储，第三块硬盘存放校验位用于恢复数据。提高读写速率，并且可以恢复，利用率为2/3

RAID-10：是四块硬盘，每两块做RAID1，然后将两个RAID1,在做RAID0

因此对于Hadoop架构来说，不同的主机使用不同的RAID

Namenode：使用RAID

DataNode：不需要实现RAID

23课时Hadoop架构分析——NameNode&DataNode源文档解读

Namenode

存储文件元数据

运行Namenode只有一个，至关重要，只有一个

只对元数据的增删做日志记录，不对block和文件流做记录

DataNode节点故障的时候，Namenode负责创建更多的block副本

The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware. It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant. HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware. HDFS provides high throughput access to application data and is suitable for applications that have large data sets. HDFS relaxes a few POSIX requirements to enable streaming access to file system data. HDFS was originally built as infrastructure for the Apache Nutch web search engine project. HDFS is part of the Apache Hadoop Core project. The project URL is <http://hadoop.apache.org/>. （摘自hadoop官方中关于HDFS Architecture的介绍）



HDFS has a **master/slave architecture**. An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. **It also determines the mapping of blocks to DataNodes.** The DataNodes are responsible for serving read and write requests from the file system’s clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

DataNode：

存储真实的块数据

可运行在多种文件系统上

通知NameNode自己有哪些block

NameNode在同一个机架上放一个副本，另一个机架放置两个副本

24课时 Hadoop启动脚本分析-start-all

关于Namenode节点启动与DataNode节点启动过程中错误：ClusterID不一致的造成的，解决办法，在格式化Namenode的时候，不会删除DataNode的数据，造成了Namenode与DataNodeClusterID不一致的错误，解决办法：1、修改数据节点的ClusterID（VERSION文件）。2删除数据保存目录下的所有文件。

start-all.cmd

设置换将变量HADOOP\_BIN\_PATH

设置HADOOP\_LIBEXEC\_DIR=$HADOOP\_HOME/libexec

Execute $HADOOP\_HOME/libexec/hadoop-config.sh

sbin/start-dfs.sh --config $HADOOP\_CONF\_DIR

sbin/start-yarn --config $HADOOP\_CONF\_DIR

hadoop-config.sh

HADOOP\_COMMON\_DIR=${HADOOP\_COMMON\_DIR:-"share/hadoop/common"}

HADOOP\_COMMON\_LIB\_JARS\_DIR=${HADOOP\_COMMON\_LIB\_JARS\_DIR:-"share/hadoop/common/lib"}

HADOOP\_COMMON\_LIB\_NATIVE\_DIR=${HADOOP\_COMMON\_LIB\_NATIVE\_DIR:-"lib/native"}

HDFS\_DIR=${HDFS\_DIR:-"share/hadoop/hdfs"}

HDFS\_LIB\_JARS\_DIR=${HDFS\_LIB\_JARS\_DIR:-"share/hadoop/hdfs/lib"}

YARN\_DIR=${YARN\_DIR:-"share/hadoop/yarn"}

YARN\_LIB\_JARS\_DIR=${YARN\_LIB\_JARS\_DIR:-"share/hadoop/yarn/lib"}

MAPRED\_DIR=${MAPRED\_DIR:-"share/hadoop/mapreduce"}

MAPRED\_LIB\_JARS\_DIR=${MAPRED\_LIB\_JARS\_DIR:-"share/hadoop/mapreduce/lib"}

Set $HADOOP\_CONF\_DIR 环境变量

Set $HADOOP\_SLAVES从节点的主机信息

**执行 ${HADOOP\_CONF\_DIR}/hadoop-env.sh脚本命令**

Set $JAVA\_HOME java 环境变量以及相关的堆栈大小信息。

JAVA\_HEAP\_MAX=-Xmx1000m

JAVA\_HEAP\_MAX="-Xmx""$HADOOP\_HEAPSIZE""m"

CLASSPATH=${CLASSPATH}:$HADOOP\_COMMON\_HOME/$HADOOP\_COMMON\_DIR

CLASSPATH=${CLASSPATH}:$HADOOP\_COMMON\_HOME/$HADOOP\_COMMON\_LIB\_JARS\_DIR'/\*

CLASSPATH=${CLASSPATH}:$HADOOP\_COMMON\_HOME/$HADOOP\_COMMON\_DIR'/\*'

HADOOP\_LOG\_DIR="$HADOOP\_PREFIX/logs"

设置CLASSPATH环境变量，添加对应的jar包

export HADOOP\_HDFS\_HOME=$HADOOP\_PREFIX

CLASSPATH=${CLASSPATH}:$HADOOP\_HDFS\_HOME/$HDFS\_DIR

CLASSPATH=${CLASSPATH}:$HADOOP\_HDFS\_HOME/$HDFS\_LIB\_JARS\_DIR'/\*'

CLASSPATH=${CLASSPATH}:$HADOOP\_HDFS\_HOME/$HDFS\_DIR'/\*'

export HADOOP\_YARN\_HOME=$HADOOP\_PREFIX

CLASSPATH=${CLASSPATH}:$HADOOP\_YARN\_HOME/$YARN\_DIR

CLASSPATH=${CLASSPATH}:$HADOOP\_YARN\_HOME/$YARN\_LIB\_JARS\_DIR'/\*'

CLASSPATH=${CLASSPATH}:$HADOOP\_YARN\_HOME/$YARN\_DIR'/\*'

export HADOOP\_MAPRED\_HOME=$HADOOP\_PREFIX

CLASSPATH=${CLASSPATH}:$HADOOP\_MAPRED\_HOME/$MAPRED\_DIR

CLASSPATH=${CLASSPATH}:$HADOOP\_MAPRED\_HOME/$MAPRED\_LIB\_JARS\_DIR'/\*'

CLASSPATH=${CLASSPATH}:$HADOOP\_MAPRED\_HOME/$MAPRED\_DIR'/\*'

hadoop-env.sh脚本分析

export JAVA\_HOME=${JAVA\_HOME}设置JAVA\_HOME环境变量

export HADOOP\_OPTS="$HADOOP\_OPTS -Djava.net.preferIPv4Stack=true"

export HADOOP\_NAMENODE\_OPTS

export HADOOP\_DATANODE\_OPTS

export HADOOP\_SECONDARYNAMENODE\_OPTS

export HADOOP\_NFS3\_OPTS="$HADOOP\_NFS3\_OPTS"

export HADOOP\_PORTMAP\_OPTS="-Xmx512m $HADOOP\_PORTMAP\_OPTS"

25课时Hadoop启动脚本分析——start-dfs hadoop-conf

start-dfs.sh 脚本分析

. $HADOOP\_LIBEXEC\_DIR/**hdfs-config.sh** 执行脚本

NAMENODES=$($HADOOP\_PREFIX/bin/hdfs getconf -namenodes)

$HADOOP\_PREFIX/sbin/**hadoop-daemons.sh"** --config "$HADOOP\_CONF\_DIR" --script "$bin/hdfs" start datanode $dataStartOpt

"$HADOOP\_PREFIX/sbin/**hadoop-daemons.sh**" --config "$HADOOP\_CONF\_DIR" --script "$bin/hdfs" start datanode $dataStartOpt

$HADOOP\_PREFIX/sbin/**hadoop-daemons.sh**" --config "$HADOOP\_CONF\_DIR" --hostnames "$SECONDARY\_NAMENODES" --script "$bin/hdfs" start secondarynamenode

hdfs-config.sh文件分析

. ${HADOOP\_LIBEXEC\_DIR}/hadoop-config.sh

调用hadoop-config.sh脚本执行

26课时 Hadoop启动脚本分析——hadoop.sh

hadoop命令文件分析

#hdfs commands

调用hdfs命令执行

#mapred commands for backwards compatibility

调用mapred命令执行

#core commands

调用java 执行对应的字节码程序

27课时Hadoop启动脚本分析——数据格式化与hdfs.cmd

hdfs命令文件

DEFAULT\_LIBEXEC\_DIR="$bin"/../libexec

HADOOP\_LIBEXEC\_DIR=${HADOOP\_LIBEXEC\_DIR:-$DEFAULT\_LIBEXEC\_DIR}

. $HADOOP\_LIBEXEC\_DIR/hdfs-config.sh

根据参数设置对应的JAVA CLASS，然后使用java命令执行对应的类实现对应的功能

**exec "$JAVA" -Dproc\_$COMMAND $JAVA\_HEAP\_MAX $HADOOP\_OPTS $CLASS "$@"**

28课时Hadoop启动脚本分析——mapred.cmd 以及虚拟机参数

mapred.sh

DEFAULT\_LIBEXEC\_DIR="$bin"/../libexec

HADOOP\_LIBEXEC\_DIR=${HADOOP\_LIBEXEC\_DIR:-$DEFAULT\_LIBEXEC\_DIR}

. ${HADOOP\_LIBEXEC\_DIR}/mapred-config.sh

根据参数设置对应的JAVA CLASS，然后使用java命令执行对应的类实现对应的功能

exec "$JAVA" -Dproc\_$COMMAND $JAVA\_HEAP\_MAX $HADOOP\_OPTS $CLASS "$@"

mapred-config.sh

DEFAULT\_LIBEXEC\_DIR="$bin"/../libexec

HADOOP\_LIBEXEC\_DIR=${HADOOP\_LIBEXEC\_DIR:-$DEFAULT\_LIBEXEC\_DIR}

. "${HADOOP\_LIBEXEC\_DIR}/hadoop-config.sh"

mapred-env.sh

export JAVA\_HOME=/home/y/libexec/jdk1.6.0/

export HADOOP\_JOB\_HISTORYSERVER\_HEAPSIZE=1000

export HADOOP\_MAPRED\_ROOT\_LOGGER=INFO,RFA

29课时Hadoop启动脚本分析——start-yarn.sh

start-yarn.sh

DEFAULT\_LIBEXEC\_DIR="$bin"/../libexec

HADOOP\_LIBEXEC\_DIR=${HADOOP\_LIBEXEC\_DIR:-$DEFAULT\_LIBEXEC\_DIR}

. $HADOOP\_LIBEXEC\_DIR/yarn-config.sh

"$bin"/**yarn-daemon.sh** --config $YARN\_CONF\_DIR start resourcemanager

"$bin"/**yarn-daemons.sh** --config $YARN\_CONF\_DIR start nodemanager

"$bin"/**yarn-daemon.sh** --config $YARN\_CONF\_DIR start proxyserver

yarn-config.sh文件分析

. ${HADOOP\_LIBEXEC\_DIR}/hadoop-config.sh

调用hadoop-config.sh脚本执行

export YARN\_CONF\_DIR="${HADOOP\_CONF\_DIR:-$HADOOP\_YARN\_HOME/conf}"

export YARN\_SLAVES="${YARN\_CONF\_DIR}/$slavesfile"

30课时 Hadoop启动脚本分析——yarn& yarn-env.sh

yarn命令脚本

DEFAULT\_LIBEXEC\_DIR="$bin"/../libexec

HADOOP\_LIBEXEC\_DIR=${HADOOP\_LIBEXEC\_DIR:-$DEFAULT\_LIBEXEC\_DIR}

. $HADOOP\_LIBEXEC\_DIR/yarn-config.sh

根据参数设置对应的JAVA CLASS，然后使用java命令执行对应的类实现对应的功能

exec "$JAVA" -Dproc\_$COMMAND $JAVA\_HEAP\_MAX $YARN\_OPTS -classpath "$CLASSPATH" $CLASS "$@"

yarn-env.sh

export HADOOP\_YARN\_USER=${HADOOP\_YARN\_USER:-yarn}

export YARN\_CONF\_DIR="${YARN\_CONF\_DIR:-$HADOOP\_YARN\_HOME/conf}"

JAVA=$JAVA\_HOME/bin/java

JAVA\_HEAP\_MAX=-Xmx1000m

JAVA\_HEAP\_MAX="-Xmx""$YARN\_HEAPSIZE""m"

Set YARN\_OPT

YARN\_OPTS="$YARN\_OPTS -Dyarn.policy.file=$YARN\_POLICYFILE"

31课时Hadoop启动脚本分析总结——hdfs部分

32课时Hadoop启动脚本分析——yarn部分

略过

2016年5月24日 关于Hadoop2.x在Ubuntu操作系统上面的安装使用，以及启动脚本的分析内容学习完毕，下一步学习关于hadoop MapReduce的内容