Hadoop

核心组件

HDFS

- 用来调动磁盘，文件存储的时候会用到

MapReduce

- 就是一段程序，用来执行计算的逻辑

YARN

- 解决资源管理问题，调动各个服务器的资源，比如cores和memory

集群

- HDFS集群

  HDFS集群
  主要针对的是服务器之间的磁盘管理，相互沟通，也就是分布式存储。他主要分为5个节点，当然，节点的个数是动态的。人多了自然要出现管理者，选出老大来管理所有的服务器，以5台为例，3台datanode用来真正的干活，这个时候就会出现2个经理一正一副，正经理有事儿没来，副经理来顶上，也就是namenode

- YARN集群

  YARN集群
  类似于HDFS集群，不过不再是数据的存储，而是资源的管理，cores和memory那些。分别是nodeManager干活和resourceManager来管理

部署规划

跳板机

- 大型企业中一般会有多种大数据平台，不同的OP机控制不同的平台，用户则需要从跳板机跳到OP机器访问才行

OP机

- 访问主节点那些

主节点

- nn1

从节点

- nn2

工作节点

- s1, s2, s3

初始化环境

配置阿里云源

Centos是默认在国外的官网上进行下载的，故要换源

- ❶ 下载repo文件
- ❷ 使用rz命令，将下载的文件上传到Linux中
- ❸ 备份，然后替换CentOS-Bash.repo
- 具体代码

```
cp Centos-7.repo /etc/yum.repos.d/
cd /etc/yum.repos.d/
mv CentOS-Base.repo CentOS-Base.repo.bak
mv Centos-t.repo CentOS-Base.repo
```

执行yum源更新

- 具体代码

```
yum clean all

yum makecache

yum update -y
```

安装常用的软件

- 具体代码

```
yum install -y openssh-server vim gcc gcc-c++ glibc-headers bzip2-devel lzo-devel curl wget openssh-clients zlib-devel autoconf automake cmake libtool openssl-devel fuse-devel snappy-devel telnet unzip zip net-tools.x86_64 firewalld systemd ntp unrar bzip2
```

安装JDK并且配置环境

- jdk放在了public中，直接下载使用即可

```
rpm -ivf jdk-8u144-linux-x64.rpm
```

jdk配置环境变量

- 大数据组件基本都需要jdk，这个时候就要告诉他们在这个linux中去哪儿找jdk
- 具体代码

```
echo 'export JAVA_HOME=/usr/java/jdk1.8.0_144' >> /etc/profile
echo 'export PATH=$PATH:$JAVA_HOME/bin' >> /etc/profile
source /etc/profile
java -version
```

修改主机名称

- 主要是为了方便理解，用nn1和s1那些来替代
- 具体代码

```
vim /etc/hostname


# nn1执行
vim /etc/hosts

scp /etc/hosts root@主机名:/etc/hosts
```

hadoop用户与权限设置

- 具体思想
  - 设置一个群，只有在该群的用户才能够切换到root
- 操作步骤
  - ❶ 修改/etc/pam.d/su配置

```
sed -i
's/#auth\t\trequired\tpam_wheel.so/auth\t\trequired\tpam_wheel.so/g'
'/etc/pam.d/su'

sed -i
's/#auth\t\tsufficient\tpam_wheel.so/auth\t\tsufficient\tpam_wheel.so/g'
'/etc/pam.d/su'
```

  - ❷ 修改/etc/login.defs文件

```
只用wheel组的才能够su到root

备份
cp /etc/login.defs /etc/login.defs_back

# 把“SU_WHEEL_ONLY yes”字符串追加到/etc/login.defs文件底部
echo "SU_WHEEL_ONLY yes" >> /etc/login.defs
```

  - ❸ 将hadoop用户加入到wheel组中

```
#把hadoop用户加到wheel组里
gpasswd -a hadoop wheel
#查看wheel组里是否有hadoop用户
cat /etc/group | grep wheel
```

给hadoop用户配置SSH密钥

- 目的
  - 让多台机器之间可以进行免密登陆
- 原理

- ❶ 用ssh-keygen在nn1上生成private和public密钥
- ❷ 将生成的public密钥拷贝到远程机器s1上，这样就可以SSH无需密码就可以访问s1
- ❸ 把公钥和密钥都拷贝到s1上则可以进行互相登陆

- 具体代码

```
#切换到hadoop用户
su – hadoop

# 以下操作均在nn1完成
#创建.ssh目录
mkdir ~/.ssh

#生成ssh公私钥
ssh-keygen -t rsa -f ~/.ssh/id_rsa -P ''

#输出公钥文件内容并且重新输入到~/.ssh/authorized_keys文件中
cat ~/.ssh/id_rsa.pub > ~/.ssh/authorized_keys

#给~/.ssh文件加上700权限
chmod 700 ~/.ssh

#给~/.ssh/authorized_keys加上600权限
chmod 600 ~/.ssh/authorized_keys

然后将配置的信息发送到每一个机器上面
scp -r /home/hadoop/.ssh hadoop@s1:/home/hadoop
```

- 相当于所有机器用的是同一套nn1的🔒和🔑

✪ 批量执行脚本

- 前置命令
  - dirname $0
    - 只能放在脚本文件中执行，返回的是当前执行脚本的位置
  - scp命令
    - linux的远程拷贝命令
    - scp 文件名 登录用户名@目标机器IP或者主机名:目标目录
    - scp /home/hadoop/f1 hadoop@s1:/home/hadoop
  - ssh命令
    - ssh 登录用户名@目标ip或者主机名

- eval命令
  - 它有一个返回值，可以知道是否执行成功。功能是执行字符串格式的命令
- 批量脚本
  - ips
    - 主要用来存放要操作的主机列表，用回车或者空格隔开
    - 具体代码

```
nn1
nn2
s1
s2
s3
```

  - ssh_all.sh
    - 用hadoop用户可登陆其他的操作机执行相应操作（多机操作脚本）
    - 具体代码

```
#! /bin/bash
path=`dirname $0`
cd $path
ip_arr=(`cat $path/ips`)

for ip in ${ip_arr[*]}
do
        # 执行ssh hadoop会默认进入默认的home目录
        _cmd="ssh hadoop@$ip \"source /etc/profile;$*\""
        if eval $_cmd;then
                echo 'success'
        else
                echo 'fail'
        fi
done
```

      - ssh_all.sh 要执行的命令
  - scp_all.sh
    - 用hadoop用户当前机器的文件拷贝发送到其他操作机（多机分发脚本）

- 具体代码

```
#! /bin/bash

# 进入当前脚本所在的目录
path=`dirname $0`
cd $path

# 读取ips获得其他机器ip存放进数组中
ip_arr=('cat $path/ips')

# 便利数组中的每个主机名
for ip in ${ip_arr[*]}
do
	# 拼接命令
	_cmd="scp $1 hadoop@$ip:$2"

	# 打印执行的命令
	echo "====$_cmd===="

	# 判断是否执行成功
	if eval $_cmd;then
		echo 'sucess'
	else
		ech0 'fail'
	fi
done
```

  - scp_all.sh 要分发的文件 目标文件夹

- exe.sh
  - 执行su命令，与ssh_root.sh搭配使用
  - 具体代码

```
#! /bin/bash
su - << EOF
$*
EOF
```

- ssh_root.sh
  - 用hadoop用户登陆到其他的操作机，然后su转换到root用户，以root用户进行相应操作

- 具体代码

```
#! /bin/bash
path=`dirname $0`
cd $path
ip_arr=(`cat $path/ips`)
for ip in ${ip_arr[*]}
do
        _cmd="ssh hadoop@$ip ~/bin/exe.sh \"$*\""
        echo "====$_cmd===="
        if eval $_cmd;then
                echo 'success'
        else
                echo 'fail'
        fi
done
```

  - ssh_root.sh 要执行的命令

- **思考题**

  假如nn1 上的/root 目录下有个f1文件，如何将f1文件分发到5台机器的/root/op/ 目录下?

  要求：不允许单机操作直接拷贝，利用多机操作、多机分发脚本来完成。