Hadoop security学习笔记

# [Hadoop的安全模式——hadoop2.2.0的Kerberos的配置](http://blog.csdn.net/may_violin/article/details/41212807)

标签： [kerberos](http://www.csdn.net/tag/kerberos" \t "_blank)[集群](http://www.csdn.net/tag/%e9%9b%86%e7%be%a4)[安全](http://www.csdn.net/tag/%e5%ae%89%e5%85%a8)[hadoop](http://www.csdn.net/tag/hadoop)

2014-11-17 17:38 492人阅读 [评论](http://blog.csdn.net/may_violin/article/details/41212807#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/may_violin/article/details/41212807#report)

http://static.blog.csdn.net/images/category_icon.jpg 分类：

hadoop相关（3） http://static.blog.csdn.net/images/arrow_triangle%20_down.jpg

版权声明：本文为博主原创文章，未经博主允许不得转载。

我的环境：

Red Hat Enterprise Linux Server release 5.5 （查看系统版本的命令：lsb\_release -a）  64位

Java 环境：jdk1.7.0\_51

Hadoop环境：最先开始使用的是未编译的hadoop2.3.0，最后使用的是编译过的hadoop2.2.0。二者配置过程差不多，但是64位的系统一定要先编译源码！！！

1.hadoop的安全特性主要包括：鉴权（authentication），服务级授权（service level authorization），用于Web控制台的身份验证（authentication for Web consoles），数据保密性（confidentiality）。

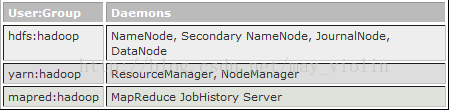
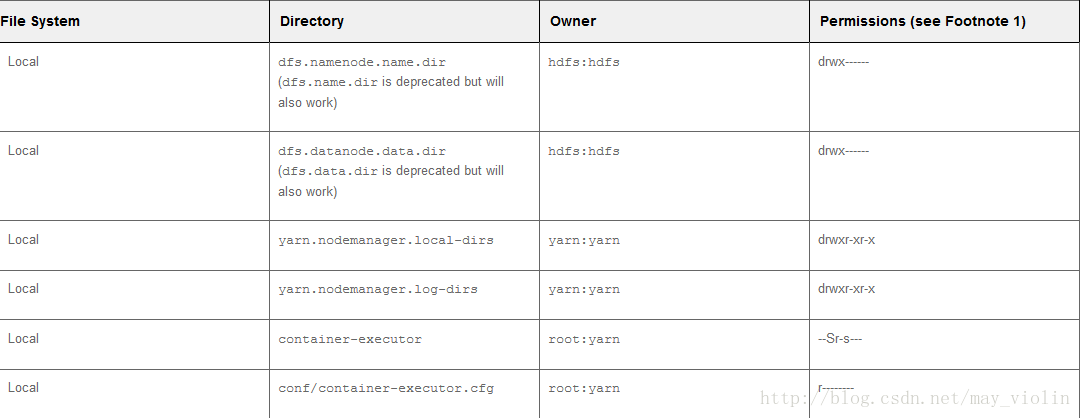
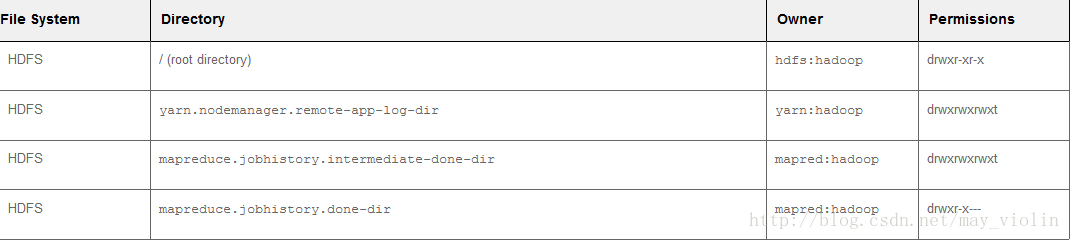
2.Authentication

1）最简单的方式：用kinit的命令。

2）要求不同的进程运行在不同的用户之下(每个用户都要配置SSH)：

HDFS-------hdfs

YARN-------yarn

JOBHISTORY-------mapred  
  
3)因为不同的进程要运行在不同的用户下，那么本地的目录的权限是必须要设置的，如下表所示：  
  
同时，在HDFS上，将目录权限设置为下表所示：  


4）如果使用的是CentOS，Red Hat Enterprise Linux 5.6及以上版本，或者Ubuntu，这些操作系统是以AES-256作为默认的票据加密方式的，需要安装jce\_policy-X.zip,连接为：

<http://www.oracle.com/technetwork/java/javase/downloads/jce-6-download-429243.html>。

把这个文件解压后，得到两个jar包，放到$JAVA\_HOME/jre/lib/security目录下面，删掉原来的Jar包。如果不这样做，之后的票据验证可能会出现GSS Init失败的错误等。

但是我们使用的系统是Red Hat 4.1.2-48，所以不存在这个问题。保持默认的加密方式就好。

PS：测试集群加密方式的方法：

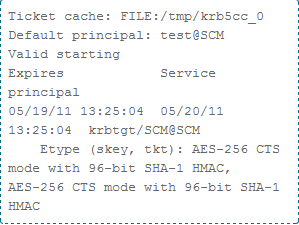
执行命令：

$ kadmin -q "addprinc test"

$ kinit test

$ klist -e

如果使用的是AES-256加密，则显示如下：



5）创建和部署Kerberos 的principal和keytab文件

    首先，在Kerberos安全的集群中，每个Kerberos principal都代表一个唯一的身份。Kerberos为这些principal分发票据使得他们可以使用hadoop的服务。而在hadoop机群中，每个principal的命名应该是：

username/fully.qualified.domain.name@YOUR-REALM.COM

这个应该对应着linux系统中的一个账号，例如hdfs和mapred，yarn等。

    其次，keytab文件包含着Kerberos principal和加密的principal key。对于每个主机来说，keytab文件时唯一的，因为在keytab文件中包含主机名。有了这个文件，principal可以不用输入密码来获得票据。因此必须严格保密，只有相应的principal有读的权限。

注意：每台机器上，必须为hdfs和mapred，yarn都设置一个keytab，而且这些keytabfile必须包含相应的principal和名为HTTP的principal，而对于每台主机来说，只有一个HTTP的principal。也就是说，hdfs和mpared，yarn的keytab中包含的HTTP的principal是同一个。

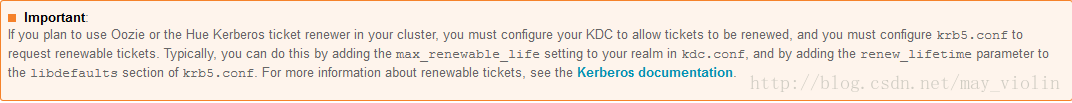
   最后，创建principal和keytab的过程如下：

可以使用kadmin或者kadmin.local两种命令进入kadmin的shell。如果拥有KDC那台机器的root账号，但是没有kerberos的admin账号，可以用kadmin.local命令；如果没有KDC那台机器的root账号，但是有admin账号，可以使用kadmin命令；如果二者都有，两个命令都可以使用。如下所示：

$ kadmin.local （运行这个命令的时候，有可能提示你输入密码，此时应该输入Unix密码）

或者

$ kadmin （运行这个命令的时候，提示输入密码的时候，输入admin的密码）

a)创建Kerberos Principals  


  命令如下：

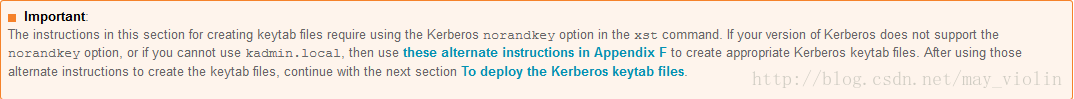
kadmin: addprinc -randkey hdfs/hostname1@YOUR\_REALM

kadmin: addprinc -randkey yarn/hostname2@YOUR\_REALM

kadmin: addprinc -randkey mapred/hostname1@YOUR\_REALM

kadmin: addprinc -randkey HTTP/hostname1@YOUR\_REALM

b)创建kerberos keytab 文件



因为是在vbaby1-3上面搭建的环境，而且我们集群上的kerberos不支持-norandkey这个参数，使用yarn作为计算框架。所以命令如下：

kadmin: xst -k hdfs.keytab hdfs/hostname1@YOUR\_REALM HTTP/hostname1@YOUR\_REALM

kadmin: xst -k yarn.keytab yarn/hostname1@YOUR\_REALM HTTP/hostname1@YOUR\_REALM

在Namenode的机器上，执行以下命令（为了配置安全的WebHDFS）：

kadmin: xst -k HTTP.keytab HTTP/hostname1@YOUR\_REALM

在有Jobhistory的机器上，执行以下命令：

kadmin: xst -k mapred.keytab mapred/hostname1@YOUR\_REALM HTTP/hostname1@YOUR\_REALM

用klist命令来查看keytab文件，各个文件应该像这样：

Keytab name: FILE: /path/to/keytabs/hdfs.keytab

KVNO Timestamp Principal

---- ----------------- --------------------------------------------------------

4 03/10/14 11:40:38 hdfs/hostname1@YOUR\_REALM (Triple DES cbc mode with HMAC/sha1)

4 03/10/14 11:40:38 hdfs/hostname1@YOUR\_REALM (DES cbc mode with CRC-32)

9 03/10/14 11:40:38 HTTP/hostname1@YOUR\_REALM (Triple DES cbc mode with HMAC/sha1)

9 03/10/14 11:40:38 HTTP/hostname1@YOUR\_REALM (DES cbc mode with CRC-32)

注意：HTTP 这个principal必须以HTTP/fully.qualified.domain.name@YOUR-REALM.COMz这种形式命名，尤其是第一个部分必须是“HTTP”，这个形式在SPNEGO中是标准的，而且在hadoop中被硬编码的，必须这样写。创建成功后看起来应该是这样：

Keytab name: FILE:/etc/security/keytabs/HTTP.keytab

KVNO Timestamp Principal

---- ----------------- --------------------------------------------------------

13 03/10/14 11:39:36 HTTP/vbaby2.cloud.eb@CLOUD.EB (Triple DES cbc mode with HMAC/sha1)

13 03/10/14 11:39:36 HTTP/vbaby2.cloud.eb@CLOUD.EB (DES cbc mode with CRC-32)

c)部署kerberos的keytab文件

将创建好的keytab文件放到kerberos定义的文件夹中，默认是/etc/security/keytabs，即执行以下命令：

$ cp hdfs.keytab /path/to/keytabs

$ cp yarn.keytab /path/to/keytabs

$ cp mapred.keytab /path/to/keytabs

$ cp HTTP.keytab /path/to/keytabs

然后设置权限：

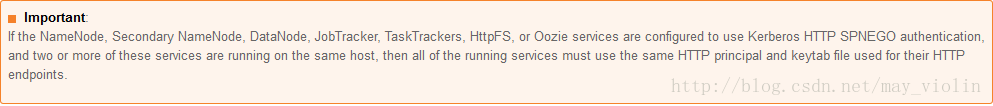
$chown hdfs:hadoop /path/to/keytabs/hdfs.keytab

$chown hdfs:hadoop /path/to/keytabs/HTTP.keytab

$chown yarn:hadoop /path/to/keytabs/yarn.keytab

$chown mapred:hadoop /path/to/keytabs/mapred.keytab

$chmod 400 /path/to/keytabs/\*.keytab



6)停止集群之后，需要修改配置文件，来使得hadoop的安全机制发挥作用

注意：每个机器上的配置文件要是一样的才可以。

a)core-site.xml文件，添加如下内容：

<property>

<name>hadoop.security.authentication</name>

<value>kerberos</value>

</property>

<property>

<name>hadoop.security.authorization</name>

<value>true</value>

</property>

b)hdfs-site.xml文件，

<!-- General HDFS security config -->

  <property>

    <name>dfs.block.access.token.enable</name>

    <value>true</value>

  </property>

<!-- NameNode security config -->

  <property>

    <name>dfs.namenode.keytab.file</name>

    <value>/path/to/keytabs/hdfs.keytab</value>

  </property>

  <property>

    <name>dfs.namenode.kerberos.principal</name>

    <value>hdfs/\_HOST@YOUR\_REALM</value>

  </property>

  <property>

    <name>dfs.namenode.kerberos.internal.spnego.principal</name>

    <value>HTTP/\_HOST@YOUR\_REALM</value>

  </property>

<!-- Secondary NameNode security config -->

  <property>

    <name>dfs.secondary.namenode.keytab.file</name>

    <value>/path/to/keytabs/hdfs.keytab</value>

  </property>

  <property>

    <name>dfs.secondary.namenode.kerberos.principal</name>

    <value>hdfs/\_HOST@YOUR\_REALM</value>

  </property>

  <property>

    <name>dfs.namenode.secondary.kerberos.internal.spnego.principal</name>

    <value>HTTP/\_HOST@YOUR\_REALM</value>

  </property>

<!-- DataNode security config -->

  <property>

    <name>dfs.datanode.data.dir.perm</name>

    <value>700</value>

  </property>

  <property>

    <name>dfs.datanode.address</name>

    <value>0.0.0.0:1004</value>

  </property>

  <property>

    <name>dfs.datanode.http.address</name>

    <value>0.0.0.0:1006</value>

  </property>

  <property>

    <name>dfs.datanode.keytab.file</name>

    <value>/path/to/keytabs/hdfs.keytab</value>

  </property>

  <property>

    <name>dfs.datanode.kerberos.principal</name>

    <value>hdfs/\_HOST@YOUR\_REALM</value>

  </property>

  <property>

    <name>dfs.encrypt.data.transfer</name>

    <value>false</value>

  </property>

<!-- WebHDFS security config -->

  <property>

    <name>dfs.webhdfs.enabled</name>

    <value>true</value>

  </property>

  <property>

    <name>dfs.web.authentication.kerberos.principal</name>

    <value>HTTP/\_HOST@YOUR\_REALM</value>

  </property>

  <property>

    <name>dfs.web.authentication.kerberos.keytab</name>

    <value>/path/to/keytabs/HTTP.keytab</value>

  </property>

  <property>

    <name>dfs.webhdfs.user.provider.user.pattern</name>

    <value>hadoop</value>

  </property>

配置hdfs-site.xml这个文件时，要注意以下几点:

* 每个进程 (NameNode, Secondary NameNode, and DataNode)的property配置的时候必须明确HDFS和HTTP这两个principal和HDFS的keytab文件路径。
* 集群中的每个节点都必须有一致的hdfs-site.xml文件。因为不仅namenode要知道Datanode，datanode也必须知道Namenode的principal名，来传递心跳信号。而此时kerberos的鉴权时双向的。
* “\_HOST”这个特别的字符串在运行时会自动替换为当时进程所在机器的完整域名。但是只有hdfs/\_HOST@YOUR-REALM.COM是合法的用法。hdfs.\_HOST@YOUR-REALM.COM 和hdfs/\_HOST.example.com@YOUR-REALM.COM都是不合法的用法。
* 在执行\_HOST替换的时候，Namenode是根据fs.default.name来确定主机名，Datanode是用反向DNS解析的结果来确定主机名；同样，JobTracker是用mapred.job.tarcker来确定主机名，Tasktracker是用反向DNS解析的结果来确定主机名。虽然在我们的集群中没有Jobtracker和TaskTracker。
* Datanode的dfs.datanode.address 和 dfs.datanode.http.address  这两个配置项的端口都不能大于1024，因为这样才能根据安全机制来防止用户冒充Datanode来执行map 任务。

c)为Secure Datanode设置环境变量，修改hadoop-env.sh文件如下：

export HADOOP\_SECURE\_DN\_LOG\_DIR=/home/hadoop/sdnlogs

export HADOOP\_SECURE\_DN\_PID\_DIR=/home/hadoop/sdnpid

export HADOOP\_SECURE\_DN\_USER=hdfs

export JSVC\_HOME=/home/hadoop/hadoop-2.3.0/libexec

这几条中，最后一条尤为重要，因为datanode需要以root用户用jsvc来启动，而hadoop2.x自身带的是jsvc的32位版本，需要续jsvc官网上重新下载编译。这一部分见***附录A jsvc的编译***。

7）之后，就可以启动namenode了，我使用的是hdfs这个用户。执行：

$HADOOP\_PREFIX/sbin/start-dfs.sh

之后，会启动namenode和secondnamenode，但是不会启动datanode。

启动datanode需要以root用户启动，执行：

$HADOOP\_PREFIX/sbin/start-secure-dns.sh

8)为HDFS的目录设置sticky bit。方法：[http://www.cloudera.com/content/cloudera-content/cloudera-docs/CDH4/4.2.2/CDH4-Security-Guide/cdh4sg\_topic\_3\_13.html](http://www.cloudera.com/content/cloudera-content/cloudera-docs/CDH4/4.2.2/CDH4-Security-Guide/cdh4sg_topic_3_13.html" \t "_blank)

9）配置安全的Yarn

a)修改yarn-site.xml文件，增加如下内容：

<!-- ResourceManager security configs -->

  <property>

    <name>yarn.resourcemanager.keytab</name>

    <value>/path/to/keytabs/yarn.keytab</value>

  </property>

  <property>

    <name>yarn.resourcemanager.principal</name>

    <value>yarn/\_HOST@YOUR\_REALM</value>

  </property>

<!-- NodeManager security configs -->

  <property>

    <name>yarn.nodemanager.keytab</name>

    <value>/path/to/keytabs/yarn.keytab</value>

  </property>

  <property>

    <name>yarn.nodemanager.principal</name>

    <value>yarn/\_HOST@YOUR\_REALM</value>

  </property>

  <property>

    <name>yarn.nodemanager.container-executor.class</name>

    <value>org.apache.hadoop.yarn.server.nodemanager.LinuxContainerExecutor</value>

  </property>

  <property>

    <name>yarn.nodemanager.linux-container-executor.group</name>

    <value>yarn</value>

  </property>

b）修改mapred-site.xml，添加如下内容：

<!-- MapReduce Job History Server security configs -->

  <property>

    <name>mapreduce.jobhistory.keytab</name>

    <value>/path/to/keytabs/mapred.keytab</value>

  </property>

  <property>

    <name>mapreduce.jobhistory.principal</name>

    <value>mapred/\_HOST@YOUR\_REALM</value>

  </property>

c）修改container-executor.cfg：

yarn.nodemanager.linux-container-executor.group=hadoop

min.user.id=499

注意：

* hadoop自带的container-executor是 32位的，需要重新编译hadoop的源码来得到64为的container-executor。编译方法见 ***附录B——编译hadoop源码***
* 需要将container-executor.cfg文件拷贝到/etc之下，因为启动nodemanager的时候要求改文件及其所有父目录的所有者均为root，所以干脆就把这个文件拷贝到/etc之下了。
* 将编译好的container-executor拷贝到$HADOOP\_PREFIX/bin目录下，然后执行以下命令：

$chown root:hadoop container-executor /path/to/container-executor.cfg

$chmod 6050 container-executor

$chmod 400 /etc/container-executor.cfg

10）启动ResourceManager和NodeManager：

用yarn这个用户，执行：

$HADOOP\_PREFIX/sbin/start-yarn.sh就可以启动整个yarn

11）跑一个mapreduce程序

用yarn跑了一个程序，验证是否成功。

**附录A jsvc的编译**

1.执行如下命令，下载commons-daemon的tar包

$wget http://mirror.esocc.com/apache//commons/daemon/binaries/commons-daemon-1.0.15-src.tar.gz

2.解压并编译

$tar zxf commons-daemon-1.0.15-src.tar.gz

$cd commons-daemon-1.0.15-src/src/native/unix

$./configure

$make

3.编译完成后会在该目录下看到jsvc文件，执行：

$file jsvc

可以看到如下信息，成功：

jsvc: ELF 64-bit LSB executable, AMD x86-64, version 1 (SYSV), for GNU/Linux 2.6.9, dynamically linked (uses shared libs), for GNU/Linux 2.6.9, not stripped

4.将生成的jsvc文件拷贝到$HADOOP\_PREFIX/libexec中，然后在hadoop-env.sh中指定$JSVC\_HOME到此路径。

5.在commons-daemon-1.0.15-src目录下执行以下命令：

$mvn -e -Dmaven.test.skip.exec=true -Dhadoop.profile=2.0 package

在target目录下生成commons-daemon-1.0.15.jar包，拷贝到$HADOOP\_PREFIX/share/hadoop/hdfs/lib下，同时删除自带版本的jar包。

**附录B hadoop源码编译**

1. 安装maven

1）从官网下载tar包，解压，得到 apache-maven-3.2.1

2）修改环境变量：在/etc/profile中修改如下：

l

3）执行source /etc/profile的命令

4）执行mvn -version测试是否成功，成功则：

Apache Maven 3.2.1 (ea8b2b07643dbb1b84b6d16e1f08391b666bc1e9; 2014-02-15T01:37:52+08:00)

Maven home: /usr/local/apache-maven-3.2.1

Java version: 1.7.0\_51, vendor: Oracle Corporation

Java home: /usr/java/jdk1.7.0\_51/jre

Default locale: en\_US, platform encoding: UTF-8

OS name: "linux", version: "2.6.18-194.el5", arch: "amd64", family: "unix"

2.安装protoc2.5.0。

（注意，一定要是2.5.0版，因为编译需要这个的支持。下载地址：[https://code.google.com/p/protobuf/downloads/list](https://code.google.com/p/protobuf/downloads/list" \t "_blank)）

1）安装依赖包：

$yum install gcc

$yum intall gcc-c++

$yum install make

2)解压protoc的tar包，然后，编译，执行如下的命令：

$tar -xvf protobuf-2.5.0.tar.bz2

$cd protobuf-2.5.0

$./configure --prefix=/opt/protoc/

$make && make install

3)配置环境变量，与之前的方法类似

export PROTOC\_HOME=/root/protobuf-2.5.0

export PATH=$PATH:${PROTOC\_HOME}/src

3.编译hadoop源码

1）安装依赖包

$yum install cmake

$yum install openssl-devel

$yum install ncurses-devel

我们的集群上没有cmake这个软件包，需要去下载安装

2）安装cmake

去[http://www.cmake.org/cmake/resources/software.html](http://www.cmake.org/cmake/resources/software.html" \t "_blank)这个地址下载包，解压后得到cmake-2.8.12.2目录。

执行：

$cd cmake-2.8.12.2

$./configure

$make

$make install

3)将编译好的cmake-2.8.12.2拷贝到/usr/local文件夹下，修改名字为cmake。命令如下：

$cp -r cmake-2.8.12.2 /usr/local

$cd /usr/local

$mv cmake-2.8.12.2 cmake

4）修改环境变量，修改/etc/profile这个文件，添加：

export PATH=/usr/local/cmake/bin:$PATH

5）让环境变量生效，执行：

$source /etc/profile

6)执行cmake --version命令，若出现以下内容，则成功了：

cmake version 2.8.12.2

3）将下载好的hadoop源码包解压，然后进入到hadoop-2.3.0-src目录下，执行以下命令：

$mvn package -Pdist,native -DskipTests -Dtar -Dcontainer-executor.conf.dir=/etc

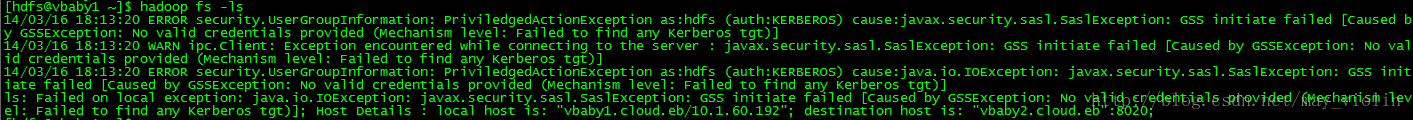
（container-executor.conf.dir必须显示注明，它表示setuid可执行文件依赖的配置文件(container-executor.cfg)路径，默认会在$HADOOP\_HOME/etc/hadoop下，不过由于该文件需要父目录和以上的目录的owner都为root，要不然会有以下报错，所以为了方便我们设置为/etc）

然后等待……

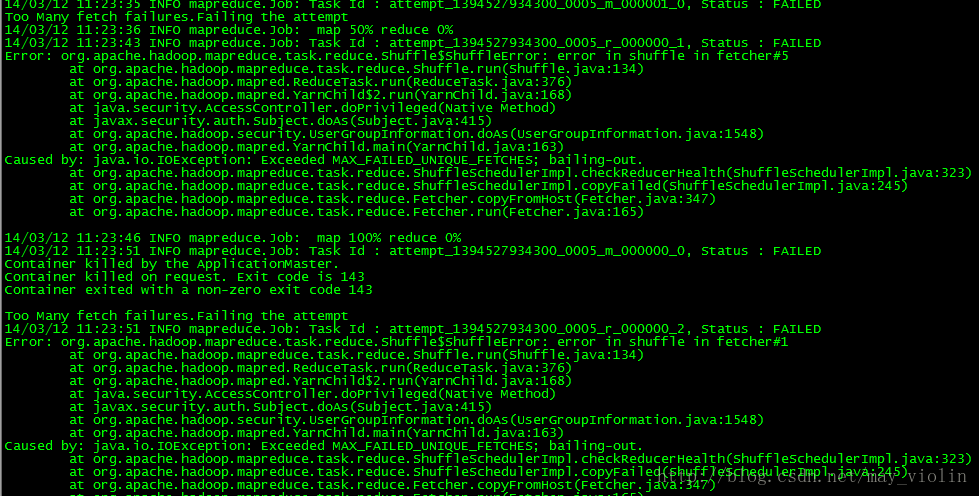
看到如下内容表示编译完成：  


**附录Ｃ　遇到的问题**

1.出现关于GSS的问题：



解决办法：此时的kerberos票据已经因为超时而失效，需要重新用kinit命令来重新申请票据。（关于是否能否自动renew票据，还需要调研）

2.在Shuffle阶段报错如下：  


解决办法：查看hadoop源码，这是由于本地库没有加载的原因，所以无法初始化SecureIOUtils这个类，所以需要加载本地库。但是，加载本地库，也没有那么简单，需要重新编译hadoop的源码，得到可以在64位机上跑的hadoop版本。编译方法详见参考链接3。加载本地库的时候，要注意在hadoop-env.sh和yarn-env.sh这两个文件中加入这样的两句话：

export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=${HADOOP\_PREFIX}/lib/native

export HADOOP\_OPTS="-Djava.library.path=$HADOOP\_PREFIX/lib/native"

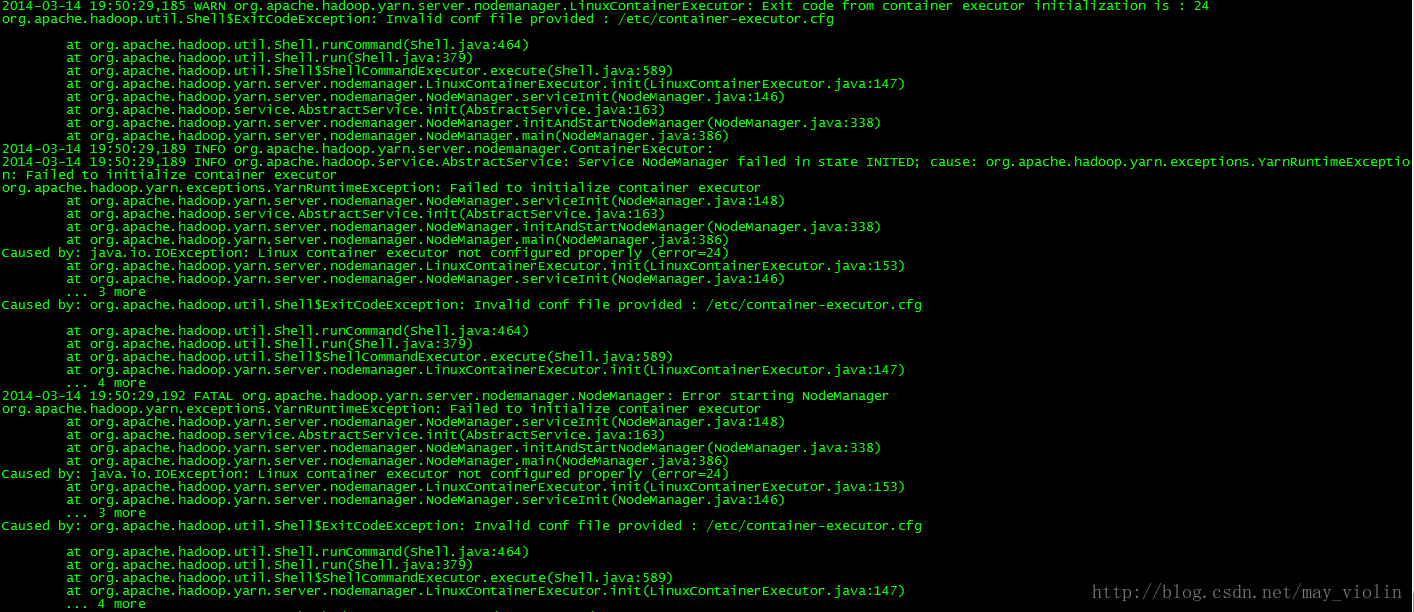
用来配置本地库的位置。

这样的话，HDFS的本地库肯定可以导入，yarn的本地库可能导入出现问题，这时候，要把yarn-env.sh中上面的第二句话改成以下的样子：

export LD\_LIBRARY\_PATH=${HADOOP\_PREFIX}/lib/native:${LD\_LIBRARY\_PATH}

这是因为如果不按照这么写的话，container的初始化可能会被nodemanager杀死。

3.找不到container-executor.cfg文档的问题，所以无法启动nodemanager：



解决办法：仔细检查container-executor的权限，必须是6050或者---Sr-s---，而且所有者一定是root：hadoop才可以。再检查一下container-executor是否加载的是要加载的那个container-executor.cfg文件。用以下命令可以得知：  
  
$ strings container-executor | grep container-executor.cfg

出现这样的结果加载的是我们编译时候使用的container-executor.cfg文件，确认一下它的权限，必须是400，也就是-r--------，然后所有者一定是root：hadoop。这些都正确的时候，检查文件的内容是否合法，如果都没问题的话，应该可以正常启动。

4.hdfs账户下跑mapreduce程序失败

解决办法：这个是hadoop-2.2.0还未解决的一个已知Bug，主要原因是如果在container-executor.cfg中没有提到banneduser的话，就默认是“hdfs，mapred，bin，0“，而且报错很模糊，具体见以下链接：<https://issues.apache.org/jira/browse/YARN-1681>

**附录Ｄ 参考链接**

1.<http://tech.ddvip.com/2013-10/1381239764203714_2.html>

2.<http://www.cloudera.com/content/cloudera-content/cloudera-docs/CDH4/4.2.2/CDH4-Security-Guide/cdh4sg_topic_3.html>

3.<http://blog.csdn.net/w13770269691/article/details/16883663>

4.<http://www.jb51.net/os/RedHat/73030.html>

5.<http://docs.hortonworks.com/HDPDocuments/HDP1/HDP-1.3.0/bk_installing_manually_book/content/rpm-chap14.html>