[hadoop 添加kerberos认证](http://blog.chinaunix.net/uid-1838361-id-3243243.html)

**一、部署无kerberos认证的Hadoop环境**

参考另一篇笔记：[hadoop集群部署](http://blog.chinaunix.net/uid-1838361-id-287231.html)

或者按照Cloudera的官方文档：[CDH3 Installation Guide](https://ccp.cloudera.com/display/CDHDOC/CDH3+Installation+Guide).

**二、环境说明**

1、主机名

之前部署hadoop集群时，没有使用节点的hostname，而是在hosts文件里添加了ip要域名的解析，部署后的hadoop没有问题，但是在为集群添加kerberos认证时因为这一点，遇到很多的问题。所以，建议还是使用节点的hostname来做解析。

集群中包含一个NameNode/JobTracker，两个DataNode/TaskTracker。

hosts文件

1. 172.18.6.152 nn.hadoop.local
2. 172.18.6.143 dn143.hadoop.local
3. 172.18.6.145 dn145.hadoop.local

注意：hosts文件中不要包含127.0.0.1的解析。

2、hadoop安装部署相关

hadoop 和kerberos的部署需要hadoop-sbin和hadoop-native。

如果使用的是rpm部署的hadoop，需要安装上面的两个rpm包。

我的集群使用的是tar包部署的，所以默认是包含这两部分文件的，可以检查一下：

hadoop-sbin对应的文件是：

/usr/local/hadoop/sbin/Linux-amd64-64

文件夹中包含两个文件：jsvc、task-controller

hadoop-native对应的目录是：

/usr/local/hadoop/lib/native

3、AES-256加密

我的系统使用的是centos6.2和centos5.7，对于使用centos5.6及以上的系统，默认使用AES-256来加密的。这就需要集群中的所有节点和hadoop user machine上安装 [Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy File](http://www.oracle.com/technetwork/java/javase/downloads/index.html)

打开上面的链接，在页面的下方，下载jdk对应的文件，jdk1.6.0\_22下载下面的文件：

注：如果后面出现login failed的错误，应先检查是否是从官方网站下载的JCE。

[](http://blog.chinaunix.net/attachment/201206/14/1838361_1339657797576w.jpg)

下载的文件是一个zip包，解开后，将里面的两个文件放到下面的目录中：

/usr/java/jdk1.6.0\_22/jre/lib/security

注：也可以不使用AED-256加密，方法见官方文档对应的部分。

**三、部署KDC**

1、安装kdc server

只需要在kdc中安装

yum install krb5-server.x86\_64  krb5-devel.x86\_64

2、配置文件

kdc服务器涉及到三个配置文件：

/etc/krb5.conf、

/var/kerberos/krb5kdc/kdc.conf、

/var/kerberos/krb5kdc/kadm5.acl

hadoop集群中其他服务器涉及到的kerberos配置文件：/etc/krb5.conf。

将kdc中的/etc/krb5.conf拷贝到集群中其他服务器即可。

集群如果开启selinux了，拷贝后可能需要执行restorecon -R -v /etc/krb5.conf

/etc/krb5.conf

1. [logging]
2. default = FILE:/var/log/krb5libs.log
3. kdc = FILE:/var/log/krb5kdc.log
4. admin\_server = FILE:/var/log/kadmind.log
5. [libdefaults]
6. default\_realm = for\_hadoop
7. dns\_lookup\_realm = false
8. dns\_lookup\_kdc = false
9. ticket\_lifetime = 24h
10. renew\_lifetime = 2d
11. forwardable = true
12. renewable = true
13. [realms]
14. for\_hadoop = {
15. kdc = 172.18.6.152:88
16. admin\_server = 172.18.6.152:749
17. }
18. [domain\_realm]
19. [kdc]
20. profile=/var/kerberos/krb5kdc/kdc.conf

/var/kerberos/krb5kdc/kdc.conf

1. [kdcdefaults]
2. kdc\_ports = 88
3. kdc\_tcp\_ports = 88
4. [realms]
5. for\_hadoop = {
6. master\_key\_type = aes256-cts
7. max\_life = 25h
8. **max\_renewable\_life = 4w**
9. acl\_file = /var/kerberos/krb5kdc/kadm5.acl
10. dict\_file = /usr/share/dict/words
11. admin\_keytab = /var/kerberos/krb5kdc/kadm5.keytab
12. supported\_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-sha1:normal arcfour-hmac:normal des-hmac-sha1:normal des-cbc-md
13. 5:normal des-cbc-crc:normal
14. }

/var/kerberos/krb5kdc/kadm5.acl

1. \*/admin@for\_hadoop \*

3、创建数据库

1. #kdb5\_util create -r for\_hadoop -s

该命令会在/var/kerberos/krb5kdc/目录下创建principal数据库。

4、关于kerberos的管理

可以使用kadmin.local或kadmin,至于使用哪个，取决于账户和访问权限：

kadmin.local（on the KDC machine）or kadmin （from any machine）

如果有访问kdc服务器的root权限，但是没有kerberos admin账户，使用kadmin.local

如果没有访问kdc服务器的root权限，但是用kerberos admin账户，使用kadmin

5、创建远程管理的管理员

1. #kadmin.local
2. addprinc root/admin@for\_hadoop

密码不能为空，且需妥善保存。

6、创建测试用户

1. #kadmin.local
2. addprinc test

7、常用kerberos管理命令

1. #kadmin.local
2. 列出所有用户 listprincs
3. 查看某个用户属性，如 getprinc hdfs/nn.hadoop.local@for\_hadoop
4. 注意，是getprinc，没有's'
5. 添加用户 addprinc
6. 更多，查看帮助

8、添加kerberos自启动及重启服务

1. chkconfig --level 35 krb5kdc on
2. chkconfig --level 35 kadmin on
3. service krb5kdc restart
4. service kadmin restart

9、测试

使用之前创建的test用户

1. # kinit test
2. Password for test@for\_hadoop:
3. #

输入密码后，没有报错即可。

1. # klist -e
2. Ticket cache: FILE:/tmp/krb5cc\_0
3. Default principal: test@for\_hadoop
4. Valid starting Expires Service principal
5. 06/14/12 15:42:33 06/15/12 15:42:33 krbtgt/for\_hadoop@for\_hadoop
6. renew until 06/21/12 15:42:33, Etype (skey, tkt): AES-256 CTS mode with 96-bit SHA-1 HMAC, AES-256 CTS mode with 96-bit SHA-1 HMAC
7. Kerberos 4 ticket cache: /tmp/tkt0
8. klist: You have no tickets cached

可以看到，已经以test@for\_hadoop登陆成功。

**四、为hadoop创建认证规则（Principals）和keytab**

1、一些概念

Kerberos principal用于在kerberos加密系统中标记一个唯一的身份。

kerberos为kerberos principal分配tickets使其可以访问由kerberos加密的hadoop服务。

对于hadoop，principals的格式为username/fully.qualified.domain.name@YOUR-REALM.COM.

keytab是包含principals和加密principal key的文件。

keytab文件对于每个host是唯一的，因为key中包含hostname。keytab文件用于不需要人工交互和保存纯文本密码，实现到kerberos上验证一个主机上的principal。

因为服务器上可以访问keytab文件即可以以principal的身份通过kerberos的认证，所以，keytab文件应该被妥善保存，应该只有少数的用户可以访问。

按照Cloudrea的文档，我们也使用两个用户hdfs和mapred，之前已经在linux上创建了相应的用户。

2、为集群中每个服务器节点添加三个principals，分别是hdfs、mapred和host。

1. 创建hdfs principal
2. kadmin: addprinc -randkey hdfs/nn.hadoop.local@for\_hadoop
3. kadmin: addprinc -randkey hdfs/dn143.hadoop.local@for\_hadoop
4. kadmin: addprinc -randkey hdfs/dn145.hadoop.local@for\_hadoop
5. 创建mapred principal
6. kadmin: addprinc -randkey mapred/nn.hadoop.local@for\_hadoop
7. kadmin: addprinc -randkey mapred/dn143.hadoop.local@for\_hadoop
8. kadmin: addprinc -randkey mapred/dn145.hadoop.local@for\_hadoop
9. 创建host principal
10. kadmin: addprinc -randkey host/nn.hadoop.local@for\_hadoop
11. kadmin: addprinc -randkey host/dn143.hadoop.local@for\_hadoop
12. kadmin: addprinc -randkey host/dn145.hadoop.local@for\_hadoop
13. 创建完成后，查看：
14. kadmin: listprincs

3、创建keytab文件

创建包含hdfs principal和host principal的hdfs keytab

1. kadmin: xst -norandkey -k hdfs.keytab hdfs/fully.qualified.domain.name host/fully.qualified.domain.name

创建包含mapred principal和host principal的mapred keytab

1. kadmin: xst -norandkey -k mapred.keytab mapred/fully.qualified.domain.name host/fully.qualified.domain.name

注：上面的方法使用了xst的norandkey参数，有些kerberos不支持该参数，我在Centos6.2上即不支持该参数。

当不支持该参数时有这样的提示：Principal -norandkey does not exist.

需要使用下面的方法来生成keytab文件。

生成独立key

1. # cd /var/kerberos/krb5kdc
2. #kadmin
3. kadmin: xst -k hdfs-unmerged.keytab hdfs/nn.hadoop.local@for\_hadoop
4. kadmin: xst -k hdfs-unmerged.keytab hdfs/dn143.hadoop.local@for\_hadoop
5. kadmin: xst -k hdfs-unmerged.keytab hdfs/dn145.hadoop.local@for\_hadoop
7. kadmin: xst -k mapred-unmerged.keytab mapred/nn.hadoop.local@for\_hadoop
8. kadmin: xst -k mapred-unmerged.keytab mapred/dn143.hadoop.local@for\_hadoop
9. kadmin: xst -k mapred-unmerged.keytab mapred/dn145.hadoop.local@for\_hadoop
10. kadmin: xst -k host.keytab host/nn.hadoop.local@for\_hadoop
11. kadmin: xst -k host.keytab host/dn143.hadoop.local@for\_hadoop
12. kadmin: xst -k host.keytab host/dn145.hadoop.local@for\_hadoop

合并key

使用ktutil 合并前面创建的keytab

1. # cd /var/kerberos/krb5kdc
2. #ktutil
3. ktutil: rkt hdfs-unmerged.keytab
4. ktutil: rkt host.keytab
5. ktutil: wkt hdfs.keytab
6. ktutil: clear
7. ktutil: rkt mapred-unmerged.keytab
8. ktutil: rkt host.keytab
9. ktutil: wkt mapred.keytab

这个过程创建了两个文件，hdfs.keytab和mapred.keytab，分别包含hdfs和host的principals，mapred和host的principals。

使用klist显示keytab文件列表，一个正确的hdfs keytab文件看起来类似于：

1. #cd /var/kerberos/krb5kdc
2. #klist -e -k -t hdfs.keytab
3. Keytab name: WRFILE:hdfs.keytab
4. slot KVNO Principal
5. ---- ---- ---------------------------------------------------------------------
6. 1 7 host/fully.qualified.domain.name@YOUR-REALM.COM (DES cbc mode with CRC-32)
7. 2 7 host/fully.qualified.domain.name@YOUR-REALM.COM (Triple DES cbc mode with HMAC/sha1)
8. 3 7 hdfs/fully.qualified.domain.name@YOUR-REALM.COM (DES cbc mode with CRC-32)
9. 4 7 hdfs/fully.qualified.domain.name@YOUR-REALM.COM (Triple DES cbc mode with HMAC/sha1)

验证是否正确合并了key，使用合并后的keytab，分别使用hdfs和host principals来获取证书。

1. # kinit -k -t hdfs.keytab hdfs/fully.qualified.domain.name@YOUR-REALM.COM
2. # kinit -k -t hdfs.keytab host/fully.qualified.domain.name@YOUR-REALM.COM

如果出现错误：

 "kinit: Key table entry not found while getting initial credentials",

则上面的合并有问题，重新执行前面的操作。

4、部署kerberos keytab文件

在集群中所有节点，执行下面的操作来部署hdfs.keytab和mapred.keytab文件

拷贝hdfs.keytab和mapred.keytab文件到hadoop可以访问的目录。

1. scp hdfs.keytab mapred.keytab host:/usr/local/hadoop/conf

**确保hdfs.keytab对hdfs用户可读**

**确报mapred.keytab对mapred用户可读**

后面经常会遇到使用keytab login失败的问题，首先需要检查的就是文件的权限。

**五、停止hadoop集群**

**六、Enable Hadoop Security**

在集群中所有节点的core-site.xml文件中添加下面的配置

1. <property>
2. <name>hadoop.security.authentication</name>
3. <value>kerberos</value> <!-- A value of "simple" would disable security. -->
4. </property>
6. <property>
7. <name>hadoop.security.authorization</name>
8. <value>true</value>
9. </property>

**七、Configure Secure HDFS**

1、在集群中所有节点的hdfs-site.xml文件中添加下面的配置

1. <!-- General HDFS security config -->
2. <property>
3. <name>dfs.block.access.token.enable</name>
4. <value>true</value>
5. </property>
7. <!-- NameNode security config -->
8. <property>
9. <name>dfs.https.address</name>
10. <value><fully qualified domain name of NN>:50470</value>
11. </property>
12. <property>
13. <name>dfs.https.port</name>
14. <value>50470</value>
15. </property>
16. <property>
17. <name>dfs.namenode.keytab.file</name>
18. <value>/usr/local/hadoop/conf/hdfs.keytab</value> <!-- path to the HDFS keytab -->
19. </property>
20. <property>
21. <name>dfs.namenode.kerberos.principal</name>
22. <value>hdfs/\_HOST@YOUR-REALM.COM</value>
23. </property>
24. <property>
25. <name>dfs.namenode.kerberos.https.principal</name>
26. <value>host/\_HOST@YOUR-REALM.COM</value>
27. </property>
29. <!-- Secondary NameNode security config -->
30. <property>
31. <name>dfs.secondary.https.address</name>
32. <value><fully qualified domain name of 2NN>:50495</value>
33. </property>
34. <property>
35. <name>dfs.secondary.https.port</name>
36. <value>50495</value>
37. </property>
38. <property>
39. <name>dfs.secondary.namenode.keytab.file</name>
40. <value>/usr/local/hadoop/conf/hdfs.keytab</value> <!-- path to the HDFS keytab -->
41. </property>
42. <property>
43. <name>dfs.secondary.namenode.kerberos.principal</name>
44. <value>hdfs/\_HOST@YOUR-REALM.COM</value>
45. </property>
46. <property>
47. <name>dfs.secondary.namenode.kerberos.https.principal</name>
48. <value>host/\_HOST@YOUR-REALM.COM</value>
49. </property>
51. <!-- DataNode security config -->
52. <property>
53. <name>dfs.datanode.data.dir.perm</name>
54. <value>700</value>
55. </property>
56. <property>
57. <name>dfs.datanode.address</name>
58. <value>0.0.0.0:1004</value>
59. </property>
60. <property>
61. <name>dfs.datanode.http.address</name>
62. <value>0.0.0.0:1006</value>
63. </property>
64. <property>
65. <name>dfs.datanode.keytab.file</name>
66. <value>/usr/local/hadoop/conf/hdfs.keytab</value> <!-- path to the HDFS keytab -->
67. </property>
68. <property>
69. <name>dfs.datanode.kerberos.principal</name>
70. <value>hdfs/\_HOST@YOUR-REALM.COM</value>
71. </property>
72. <property>
73. <name>dfs.datanode.kerberos.https.principal</name>
74. <value>host/\_HOST@YOUR-REALM.COM</value>
75. </property>

2、启动namenode

1. # sudo -u hdfs /usr/local/hadoop/bin/hadoop namenode

启动后可以看到下面的信息

1. 10/10/25 17:01:46 INFO security.UserGroupInformation:
2. Login successful for user hdfs/fully.qualified.domain.name@YOUR-REALM.COM using keytab file /etc/hadoop/hdfs.keytab
3. 10/10/25 17:01:52 INFO security.UserGroupInformation: Login successful for user host/fully.qualified.domain.name@YOUR-REALM.COM using keytab file /etc/hadoop/hdfs.keytab
4. 10/10/25 17:01:52 INFO http.HttpServer: Added global filtersafety (class=org.apache.hadoop.http.HttpServer$QuotingInputFilter)
5. 10/10/25 17:01:57 INFO http.HttpServer: Adding Kerberos filter to getDelegationToken
6. 10/10/25 17:01:57 INFO http.HttpServer: Adding Kerberos filter to renewDelegationToken
7. 10/10/25 17:01:57 INFO http.HttpServer: Adding Kerberos filter to cancelDelegationToken
8. 10/10/25 17:01:57 INFO http.HttpServer: Adding Kerberos filter to fsck
9. 10/10/25 17:01:57 INFO http.HttpServer: Adding Kerberos filter to getimage

关于错误：

1. 12/06/13 13:24:43 WARN ipc.Server: Auth failed for 127.0.0.1:63202:null
2. 12/06/13 13:24:43 WARN ipc.Server: Auth failed for 127.0.0.1:63202:null
3. 12/06/13 13:24:43 INFO ipc.Server: IPC Server listener on 9000: readAndProcess threw exception javax.security.sasl.SaslException: GSS initiate failed [Caused by GS\*\*\*ception: Failure unspecified at GSS-API level (Mechanism level: Encryption type AES256 CTS mode with HMAC SHA1-96 is not supported/enabled)] from client 127.0.0.1. Count of bytes read: 0
4. javax.security.sasl.SaslException: GSS initiate failed [Caused by GS\*\*\*ception: Failure unspecified at GSS-API level (Mechanism level: Encryption type AES256 CTS mode with HMAC SHA1-96 is not supported/enabled)]
5. 12/06/13 13:23:21 WARN security.UserGroupInformation: Not attempting to re-login since the last re-login was attempted less than 600 seconds before.

这两个错误与前面提到的JCE jar包有关，确保已经下载并替换了相应的jar包。

如果出现Login failed，应首先使用kinit的方式登陆，如果可以登陆，检查是否使用了正确的JCE jar包。然后就是检查keytab的路径及权限。

另外，第二个错误，也有可能与SELINUX有关，在所有配置不变的情况下，关闭selinux可以解决问题。但是在/var/log/audit/audit.log里没有看到相关的错误。之后不知何故，开启selinux也不会造成上面的那个问题了。

3、验证namenode是否正确启动

两种方法：

（1）访问http://machine:50070

（2）

1. #hadoop fs -ls

注：如果在你的凭据缓存中没有有效的kerberos ticket，执行hadoop fs -ls将会失败。

可以使用klist来查看是否有有有效的ticket。

可以通过kinit来获取ticket.

kinit -k -t /usr/local/hadoop/conf/hdfs.ketab hdfs/nn.hadoop.local@for\_hadoop

如果没有有效的ticket，将会出现下面的错误：

1. 11/01/04 12:08:12 WARN ipc.Client: Exception encountered while connecting to the server : javax.security.sasl.SaslException:
2. GSS initiate failed [Caused by GS\*\*\*ception: No valid credentials provided (Mechanism level: Failed to find any Kerberos tgt)]
3. Bad connection to FS. command aborted. exception: Call to nn-host/10.0.0.2:8020 failed on local exception: java.io.IOException:
4. javax.security.sasl.SaslException: GSS initiate failed [Caused by GS\*\*\*ception: No valid credentials provided (Mechanism level: Failed to find any Kerberos tgt)]

注：如果使用的MIT kerberos 1.8.1或更高版本，ORACLE JDK6 UPDATE 26和更早的版本存在一个bug：

即使成功的使用kinit获取了ticket，java仍然无法读取kerberos 票据缓存。

解决的办法是在使用kinit获取ticket之后使用kinit -R 来renew ticket。这样，将重写票据缓存中的ticket为java可读的格式。

但是，在使用kinit -R 时遇到一个问题，就是无法renew ticket

1. kinit: Ticket expired while renewing credentials

在官方文档中也有描述：[Java is unable to read the Kerberos credentials cache created by versions of MIT Kerberos 1.8.1 or higher.](https://ccp.cloudera.com/display/CDHDOC/Appendix+A+-+Troubleshooting#AppendixA-Troubleshooting-Problem2%3AJavaisunabletoreadtheKerberoscredentialscachecreatedbyversionsofMITKerberos1.8.1orhigher.)

关于是否以获取renew的ticket，取决于KDC的设置。

是否是可以获取renew的ticket，可以通过klist来查看：

**如果不可以获取renw的ticket，”valid starting" and "renew until"的值是相同的时间。**

我为了获取renw的ticket，做了以下的尝试：

<1>、在kdc.conf中添加默认flag

default\_principal\_flags = +forwardable,+renewable

但是实际没有起作用，因为查看资料，默认的principal\_flags就包含了renewable，所以问题不是出在这里。

另外需要说明一点，default\_principal\_flags 只对这个flags生效以后创建的principal生效，之前创建的不生效，需要使用modprinc来使之前的principal生效。

<2>、在kdc.conf中添加：

1. **max\_renewable\_life = 10d**

重启kdc， 重新kinit -k -t .....，重新执行kinit -R可以正常renw了。

再次验证，修改为：

1. max\_renewable\_life = 0s

重启kdc，重新kinit -k -t ......，重新执行 kinit -R在此不能renew ticket了。

所以，是否可以获取renew的ticket是这样设置的：

**默认是可以获取renew的ticket的，但是，可以renw的最长时间是0s，所以造成无法renew，解决的办法是在kdc.conf中增大该参数。**

另外关于krb5.conf中的renew\_lifetime = 7d参数，该参数设置该服务器上的使用kinit -R时renew的时间。

另外，也可以通过modprinc来修改max\_renewable\_life的值，使用modprinc修改的值比kdc.conf中的配置有更高的优先级，例如，使用modprinc设置了为7天，kdc.conf中设置了为10天，使用getprinc可以看出，实际生效的是7天。需要注意的是，即要修改**krbtgt/for\_hadoop**@for\_hadoop，也要修改类似于hdfs/dn145.hadoop.local@for\_hadoop这样的prinicials，通过klist可以看出来：

1. # klist
2. Ticket cache: FILE:/tmp/krb5cc\_0
3. Default principal: hdfs/dn145.hadoop.local@for\_hadoop
4. Valid starting Expires Service principal
5. 06/14/12 17:15:05 06/15/12 17:15:05 krbtgt/for\_hadoop@for\_hadoop
6. renew until 06/21/12 17:15:04
7. Kerberos 4 ticket cache: /tmp/tkt0
8. klist: You have no tickets cached

如何使用modprinc来修改max\_renewable\_life

1. #kadmin.local
2. modprinc -maxrenewlife 7days krbtgt/for\_hadoop@for\_hadoop
3. getprinc krbtgt/for\_hadoop@for\_hadoop
4. Principal: krbtgt/for\_hadoop@for\_hadoop
5. Expiration date: [never]
6. Last password change: [never]
7. Password expiration date: [none]
8. Maximum ticket life: 1 day 00:00:00
9. Maximum renewable life: 7 days 00:00:00
10. Last modified: Thu Jun 14 11:25:15 CST 2012 (hdfs/admin@for\_hadoop)
11. Last successful authentication: [never]
12. Last failed authentication: [never]
13. Failed password attempts: 0
14. Number of keys: 7
15. Key: vno 1, aes256-cts-hmac-sha1-96, no salt
16. Key: vno 1, aes128-cts-hmac-sha1-96, no salt
17. Key: vno 1, des3-cbc-sha1, no salt
18. Key: vno 1, arcfour-hmac, no salt
19. Key: vno 1, des-hmac-sha1, no salt
20. Key: vno 1, des-cbc-md5, no salt
21. Key: vno 1, des-cbc-crc, no salt

到这里，kinit -R的问题解决，可以成功的执行hadoop fs -ls了。

4、启动datanode

正确的启动方法应该是使用**root**账号

1. HADOOP\_DATANODE\_USER=hdfs sudo -E /usr/local/hadoop/bin/hadoop datanode

如果使用其他用户，直接执行hadoop datanode，则会报错：

1. 11/03/21 12:46:57 ERROR datanode.DataNode: java.lang.RuntimeException: Cannot start secure cluster without privileged resources. In a secure cluster, the DataNode must
2. be started from within jsvc. If using Cloudera packages, please install the hadoop-0.20-sbin package.
3. For development purposes ONLY you may override this check by setting dfs.datanode.require.secure.ports to false. \*\*\* THIS WILL OPEN A SECURITY HOLE AND MUST NOT BE
4. USED FOR A REAL CLUSTER \*\*\*.
5. at org.apache.hadoop.hdfs.server.datanode.DataNode.startDataNode(DataNode.java:306)
6. at org.apache.hadoop.hdfs.server.datanode.DataNode.<init>(DataNode.java:280)
7. at org.apache.hadoop.hdfs.server.datanode.DataNode.makeInstance(DataNode.java:1533)
8. at org.apache.hadoop.hdfs.server.datanode.DataNode.instantiateDataNode(DataNode.java:1473)
9. at org.apache.hadoop.hdfs.server.datanode.DataNode.createDataNode(DataNode.java:1491)
10. at org.apache.hadoop.hdfs.server.datanode.DataNode.secureMain(DataNode.java:1616)
11. at org.apache.hadoop.hdfs.server.datanode.DataNode.main(DataNode.java:1626)

官方文档中提到了这个问题：

[Cannot start secure cluster without privileged resources.](https://ccp.cloudera.com/display/CDHDOC/Appendix+A+-+Troubleshooting#AppendixA-Troubleshooting-Problem5%3ACannotstartsecureclusterwithoutprivilegedresources.)

官方的解释是和jsvc有关，确实，与jsvc有关.

（1）、有可能没有安装hadoop-sbin。

 （2）、**确保jsv对于HADOOP\_DATANODE\_USER=hdfs有可执行的权限。**

（3）、通过查看hadoop这个启动脚本，可以看到这样的代码：

1. if [ "$EUID" = "0" ] ; then
2. if [ "$COMMAND" == "datanode" ] && [ -x "$\_JSVC\_PATH" ]; then
3. \_HADOOP\_RUN\_MODE="jsvc"
4. elif [ -x /bin/su ]; then
5. \_HADOOP\_RUN\_MODE="su"
6. else

检查执行hadoop命令的用户的EUID是否为0，即root，只有root用户才去执行jsvc相关的命令。

关于EUID：linux系统中每个进程都有2个ID，分别为用户ID（uid）和有效用户ID（euid），UID一般表示进程的创建者（属于哪个用户创建），而EUID表示进程对于文件和资源的访问权限（具备等同于哪个用户的权限）。一般情况下2个ID是相同的。

5、 Set the Sticky Bit on HDFS Directories.

可以针对hdfs上的目录设置sticky bit，用于防止除superuser，owner以外的用户删除文件夹中的文件。对一个文件设置sticky bit是无效的。

**八、Start up the Secondary NameNode**

跳过

**九、Configure Secure MapReduce**

在mapred-site.xml中添加

1. <!-- JobTracker security configs -->
2. <property>
3. <name>mapreduce.jobtracker.kerberos.principal</name>
4. <value>mapred/\_HOST@YOUR-REALM.COM</value>
5. </property>
6. <property>
7. <name>mapreduce.jobtracker.kerberos.https.principal</name>
8. <value>host/\_HOST@YOUR-REALM.COM</value>
9. </property>
10. <property>
11. <name>mapreduce.jobtracker.keytab.file</name>
12. <value>/usr/local/hadoop/conf/mapred.keytab</value> <!-- path to the MapReduce keytab -->
13. </property>
15. <!-- TaskTracker security configs -->
16. <property>
17. <name>mapreduce.tasktracker.kerberos.principal</name>
18. <value>mapred/\_HOST@YOUR-REALM.COM</value>
19. </property>
20. <property>
21. <name>mapreduce.tasktracker.kerberos.https.principal</name>
22. <value>host/\_HOST@YOUR-REALM.COM</value>
23. </property>
24. <property>
25. <name>mapreduce.tasktracker.keytab.file</name>
26. <value>/usr/local/hadoop/conf/mapred.keytab</value> <!-- path to the MapReduce keytab -->
27. </property>
29. <!-- TaskController settings -->
30. <property>
31. <name>mapred.task.tracker.task-controller</name>
32. <value>org.apache.hadoop.mapred.LinuxTaskController</value>
33. </property>
34. <property>
35. <name>mapreduce.tasktracker.group</name>
36. <value>mapred</value>
37. </property>

创建一个taskcontroller.cfg文件，路径为

<path of task-controller binary>/../../conf/taskcontroller.cfg

即/usr/local/hadoop/sbin/Linux-amd64-64/../../conf/taskcontroller.cfg

即conf目录，和site文件相同的目录

1. mapred.local.dir=/hadoop\_data/tmp/mapred/local
2. hadoop.log.dir=/usr/local/hadoop/logs
3. mapreduce.tasktracker.group=hadoop
4. banned.users=hadoop,hdfs,bin
5. min.user.id=500

其中：

mapred.local.dir需要和mapred-site.xml中指定的相同，否则[this error message](https://ccp.cloudera.com/display/CDHDOC/Appendix+A+-+Troubleshooting#AppendixA-Troubleshooting-mapreddirectoryerror)

hadoop.log.dir要和hadoop所使用的目录相同，可以在core-site.xml中指定，不同的话会报错：[this error message](https://ccp.cloudera.com/display/CDHDOC/Appendix+A+-+Troubleshooting#AppendixA-Troubleshooting-hadooplogsdirectoryerror)

另外mapred.local.dir的属主为mapred用户:

1. chown -R mapred.mapred  /hadoop\_data/tmp/mapred/local

**Note**  
In the taskcontroller.cfg file, the default setting for the banned.users property is mapred, hdfs, and bin to prevent jobs from being submitted via those user accounts. The default setting for themin.user.id property is 1000 to prevent jobs from being submitted with a user ID less than 1000, which are conventionally Unix super users. Note that some operating systems such as CentOS 5 use a default value of 500 and above for user IDs, not 1000. If this is the case on your system, change the default setting for the min.user.id property to 500. If there are user accounts on your cluster that have a user ID less than the value specified for the min.user.id property, the TaskTracker returns an error code of 255.

修改task-controller文件的权限：

[More Information about the hadoop-0.20-sbin Binary Programs](https://ccp.cloudera.com/display/CDHDOC/Appendix+B+-+More+Information+about+the+hadoop-0.20-sbin+Binary+Programs).

1. chown root:mapred /usr/local/hadoop/sbin/Linux-amd64-64/task-controller
2. chmod 4754 /usr/local/hadoop/sbin/Linux-amd64-64/task-controller

**启动JOBTRACKER**

1. sudo -u mapred /usr/local/hadoop/bin/hadoop jobtracker

错误：

1. FATAL mapred.JobTracker: org.apache.hadoop.security.AccessControlException: The systemdir hdfs://nn.hadoop.local:9000/hadoop\_data/tmp/mapred/system is not owned by mapred

修改hdfs上对应目录的属性

1. hadoop fs -chown -R mapred /hadoop\_data/tmp/mapred

注意，是mapred而不是mapred.mapred，否则会变成 mapred.mapred supergroup          0 2012-06-08 11:41 /hadoop\_data/tmp/mapred/system

重新启动JobTracker。

到这里JobTracker启动完成，最后一步，启动TaskTracker

修改taskcontroller.cfg文件属性，启动tasktracker时会检查（jobtracker不需要？待验证）

1. chown root.mapred taskcontroller.cfg
2. chmod 600 taskcontroller.cfg

同样的，也需要修改task-controler的属性

1. chown root:mapred  /usr/local/hadoop/sbin/Linux-amd64-64/task-controller
2. chmod 4754 /usr/local/hadoop/sbin/Linux-amd64-64/task-controller

启动

1. sudo -u mapred /usr/local/hadoop/bin/hadoop tasktracker

错误：

1. ERROR mapred.TaskTracker: Can not start task tracker because java.io.IOException: Login failure for mapred/srv143.madeforchina.co@for\_hadoop from keytab /usr/local/hadoop/mapred.keytab

使用kinit可以登陆？确保key对于mapred用户可读。

另外，可以还需要修改log目录的权限

1. chown -R mapred.hadoop /usr/local/hadoop/logs/

到这里，hadoop + kerberos基本完后。

后面需要做的工作包括修改启动hadoop的脚本，部署kerberos slave服务器。