# HDFS配置Kerberos认证

总体步骤：

1. 生成kerberos认证文件
2. 修改配置文件
3. 重启服务
4. 用户认证
5. 正常提交任务

## 1. 环境说明

系统环境：

操作系统：CentOs 6.9

Hadoop版本：hadoop-2.7.3

JDK版本：1.8.0\_111

运行用户：root

集群各节点角色规划为：

192.168.56.121 cdh1 NameNode、ResourceManager、HBase、Hive metastore、Impala Catalog、Impala statestore、Sentry 192.168.56.122 cdh2 DataNode、SecondaryNameNode、NodeManager、HBase、Hive Server2、Impala Server 192.168.56.123 cdh3 DataNode、HBase、NodeManager、Hive Server2、Impala Server

cdh1作为master节点，其他节点作为slave节点，我们在cdh1节点安装kerberos Server，在其他节点安装kerberos client。

## 2. 准备工作

确认添加主机名解析到 /etc/hosts 文件中。

$ cat /etc/hosts 127.0.0.1 localhost

172.16.8.107 kudu1

172.16.8.108 kudu2

172.16.8.109 kudu3

注意：hostname 请使用小写，要不然在集成 kerberos 时会出现一些错误。

## 3. 安装 Kerberos

在 cdh1 上安装包 krb5、krb5-server 和 krb5-client。

$ yum install krb5-server -y

在其他节点（cdh1、cdh2、cdh3）安装 krb5-devel、krb5-workstation ：

#使用无密码登陆 $ ssh cdh1 "yum install krb5-devel krb5-workstation -y" $ ssh cdh2 "yum install krb5-devel krb5-workstation -y" $ ssh cdh3 "yum install krb5-devel krb5-workstation -y"

## 4. 修改配置文件

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

配置 Kerberos 的一种方法是编辑配置文件 /etc/krb5.conf。默认安装的文件中包含多个示例项。

[logging] default = FILE:/var/log/krb5libs.log kdc = FILE:/var/log/krb5kdc.log admin\_server = FILE:/var/log/kadmind.log [libdefaults] default\_realm = DTSTACK.COM dns\_lookup\_realm = false dns\_lookup\_kdc = false ticket\_lifetime = 24h renew\_lifetime = 7d forwardable = true default\_tgs\_enctypes = aes256-cts-hmac-sha1-96 default\_tkt\_enctypes = aes256-cts-hmac-sha1-96 permitted\_enctypes = aes256-cts-hmac-sha1-96 clockskew = 120 udp\_preference\_limit = 1 [realms] DTSTACK.COM = { kdc = kudu1 admin\_server = kudu1 } [domain\_realm] .dtstack.com = DTSTACK.COM dtstack.com = DTSTACK.COM

说明：

[logging]：表示 server 端的日志的打印位置

[libdefaults]：每种连接的默认配置，需要注意以下几个关键的小配置

default\_realm = DTSTACK.COM：设置 Kerberos 应用程序的默认领域。如果您有多个领域，只需向 [realms] 节添加其他的语句。

ticket\_lifetime： 表明凭证生效的时限，一般为24小时。

renew\_lifetime： 表明凭证最长可以被延期的时限，一般为一个礼拜。当凭证过期之后，对安全认证的服务的后续访问则会失败。

clockskew：时钟偏差是不完全符合主机系统时钟的票据时戳的容差，超过此容差将不接受此票据。通常，将时钟扭斜设置为 300 秒（5 分钟）。这意味着从服务器的角度看，票证的时间戳与它的偏差可以是在前后 5 分钟内。

udp\_preference\_limit= 1：禁止使用 udp 可以防止一个 Hadoop 中的错误

[realms]：列举使用的 realm。

kdc：代表要 kdc 的位置。格式是 机器:端口

admin\_server：代表 admin 的位置。格式是 机器:端口

default\_domain：代表默认的域名

[appdefaults]：可以设定一些针对特定应用的配置，覆盖默认配置。

修改 /var/kerberos/krb5kdc/kdc.conf ，该文件包含 Kerberos 的配置信息。例如，KDC 的位置，Kerbero 的 admin 的realms 等。需要所有使用的 Kerberos 的机器上的配置文件都同步。这里仅列举需要的基本配置。详细介绍参考：[krb5conf](http://web.mit.edu/~kerberos/krb5-devel/doc/admin/conf_files/krb5_conf.html)

[kdcdefaults] kdc\_ports = 88 kdc\_tcp\_ports = 88 [realms] DTSTACK.COM = { #master\_key\_type = aes256-cts acl\_file = /var/kerberos/krb5kdc/kadm5.acl dict\_file = /usr/share/dict/words max\_renewable\_life = 7d max\_life = 1d admin\_keytab = /var/kerberos/krb5kdc/kadm5.keytab supported\_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-sha1:normal arcfour-hmac:normal des-hmac-sha1:normal des-cbc-md5:normal des-cbc-crc:normal default\_principal\_flags = +renewable, +forwardable }

说明：

DTSTACK.COM： 是设定的 realms。名字随意。Kerberos 可以支持多个 realms，会增加复杂度。大小写敏感，一般为了识别使用全部大写。这个 realms 跟机器的 host 没有大关系。

master\_key\_type：和 supported\_enctypes 默认使用 aes256-cts。JAVA 使用 aes256-cts 验证方式需要安装 JCE 包，见下面的说明。为了简便，你可以不使用 aes256-cts 算法，这样就不需要安装 JCE 。

acl\_file：标注了 admin 的用户权限，需要用户自己创建。文件格式是：Kerberos\_principal permissions [target\_principal] [restrictions]

supported\_enctypes：支持的校验方式。

admin\_keytab：KDC 进行校验的 keytab。

关于AES-256加密：

对于使用 centos5. 9 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上安装 JCE，JDK1.8到

<https://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html>下载。

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：$JAVA\_HOME/jre/lib/security

为了能够不直接访问 KDC 控制台而从 Kerberos 数据库添加和删除主体，请对 Kerberos 管理服务器指示允许哪些主体执行哪些操作。通过编辑文件 /var/lib/kerberos/krb5kdc/kadm5.acl 完成此操作。ACL（访问控制列表）允许您精确指定特权。

$ cat /var/kerberos/krb5kdc/kadm5.acl \*/admin@DTSTACK.COM \*

## 5. 同步配置文件

将 kdc 中的 /etc/krb5.conf 拷贝到集群中其他服务器即可。

$ scp /etc/krb5.conf kudu2:/etc/krb5.conf $ scp /etc/krb5.conf kudu3:/etc/krb5.conf

请确认集群如果关闭了 selinux。

## 6. 创建数据库

在 cdh1 上运行初始化数据库命令。其中 -r 指定对应 realm。

$ kdb5\_util create -r DTSTACK.COM -s

出现 Loading random data 的时候另开个终端执行点消耗CPU的命令如 cat /dev/sda > /dev/urandom 可以加快随机数采集。该命令会在 /var/kerberos/krb5kdc/目录下创建 principal 数据库。

如果遇到数据库已经存在的提示，可以把 /var/kerberos/krb5kdc/ 目录下的 principal 的相关文件都删除掉。默认的数据库名字都是 principal。可以使用 -d 指定数据库名字。

## 7. 启动服务

在 cdh1 节点上运行：

$ chkconfig --level 35 krb5kdc on $ chkconfig --level 35 kadmin on $ service krb5kdc start $ service kadmin start

## 8. 创建 kerberos 管理员

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：

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

在 cdh1 上创建远程管理的管理员：

#手动输入两次密码，这里密码为 root $ kadmin.local -q "addprinc root/admin" # 也可以不用手动输入密码 $ echo -e "root\nroot" | kadmin.local -q "addprinc root/admin" # 或者运行下面命令 $ kadmin.local <<eoj addprinc -pw root root/admin eoj

系统会提示输入密码，密码不能为空，且需妥善保存。

## 9. 测试 kerberos

查看当前的认证用户：

# 查看principals $ kadmin: list\_principals # 添加一个新的 principal kadmin: addprinc user1 WARNING: no policy specified for user1@DTSTACK.COM; defaulting to no policy Enter password for principal "user1@DTSTACK.COM": Re-enter password for principal "user1@DTSTACK.COM": Principal "user1@DTSTACK.COM" created. # 删除 principal kadmin: delprinc user1 Are you sure you want to delete the principal "user1@DTSTACK.COM"? (yes/no): yes Principal "user1@DTSTACK.COM" deleted. Make sure that you have removed this principal from all ACLs before reusing. kadmin: exit

也可以直接通过下面的命令来执行：

# 提示需要输入密码 $ kadmin -p root/admin -q "list\_principals" $ kadmin -p root/admin -q "addprinc user2" $ kadmin -p root/admin -q "delprinc user2" # 不用输入密码 $ kadmin.local -q "list\_principals" $ kadmin.local -q "addprinc user2" $ kadmin.local -q "delprinc user2"

创建一个测试用户 test，密码设置为 test：

$ echo -e "test\ntest" | kadmin.local -q "addprinc test"

获取 test 用户的 ticket：

# 通过用户名和密码进行登录 $ kinit test Password for test@DTSTACK.COM: $ klist -e Ticket cache: FILE:/tmp/krb5cc\_0 Default principal: test@DTSTACK.COM Valid starting Expires Service principal 11/07/14 15:29:02 11/08/14 15:29:02 krbtgt/DTSTACK.COM@DTSTACK.COM renew until 11/17/14 15:29:02, Etype (skey, tkt): aes256-cts-hmac-sha1-96, aes256-cts-hmac-sha1-96 Kerberos 4 ticket cache: /tmp/tkt0 klist: You have no tickets cached

销毁该 test 用户的 ticket：

$ kdestroy $ klist klist: No credentials cache found (ticket cache FILE:/tmp/krb5cc\_0) Kerberos 4 ticket cache: /tmp/tkt0 klist: You have no tickets cached

更新 ticket：

$ kinit root/admin Password for root/admin@DTSTACK.COM: $ klist Ticket cache: FILE:/tmp/krb5cc\_0 Default principal: root/admin@DTSTACK.COM Valid starting Expires Service principal 11/07/14 15:33:57 11/08/14 15:33:57 krbtgt/DTSTACK.COM@DTSTACK.COM renew until 11/17/14 15:33:57 Kerberos 4 ticket cache: /tmp/tkt0 klist: You have no tickets cached $ kinit -R $ klist Ticket cache: FILE:/tmp/krb5cc\_0 Default principal: root/admin@DTSTACK.COM Valid starting Expires Service principal 11/07/14 15:34:05 11/08/14 15:34:05 krbtgt/DTSTACK.COM@DTSTACK.COM renew until 11/17/14 15:33:57 Kerberos 4 ticket cache: /tmp/tkt0 klist: You have no tickets cached

抽取密钥并将其储存在本地 keytab 文件 /etc/krb5.keytab 中。这个文件由超级用户拥有，所以您必须是 root 用户才能在 kadmin shell 中执行以下命令：

$ kadmin.local -q "ktadd kadmin/admin" $ klist -k /etc/krb5.keytab Keytab name: FILE:/etc/krb5.keytab KVNO Principal ---- -------------------------------------------------------------------------- 3 kadmin/admin@DTSTACK.COM 3 kadmin/admin@DTSTACK.COM 3 kadmin/admin@DTSTACK.COM 3 kadmin/admin@DTSTACK.COM 3 kadmin/admin@DTSTACK.COM

## 10. HDFS 上配置 kerberos

### 10.1 创建认证规则

在 Kerberos 安全机制里，一个 principal 就是 realm 里的一个对象，一个 principal 总是和一个密钥（secret key）成对出现的。

这个 principal 的对应物可以是 service，可以是 host，也可以是 user，对于 Kerberos 来说，都没有区别。

Kdc(Key distribute center) 知道所有 principal 的 secret key，但每个 principal 对应的对象只知道自己的那个 secret key 。这也是“共享密钥“的由来。

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

通过 yum 源安装的 cdh 集群中，NameNode 和 DataNode 是通过 hdfs 启动的，故为集群中每个服务器节点添加两个principals：hdfs、HTTP。

在 KCD server 上（这里是 cdh1）创建 hdfs principal：

kadmin.local -q "addprinc -randkey hdfs/kudu1@DTSTACK.COM" kadmin.local -q "addprinc -randkey hdfs/kudu2@DTSTACK.COM" kadmin.local -q "addprinc -randkey hdfs/kudu3@DTSTACK.COM"

kadmin.local -q "addprinc -randkey hdfs/nn1@DTSTACK.COM"

kadmin.local -q "addprinc -randkey hdfs/nn2@DTSTACK.COM"

-randkey 标志没有为新 principal 设置密码，而是指示 kadmin 生成一个随机密钥。之所以在这里使用这个标志，是因为此 principal 不需要用户交互。它是计算机的一个服务器帐户。

创建 HTTP principal：

kadmin.local -q "addprinc -randkey HTTP/kudu1@DTSTACK.COM" kadmin.local -q "addprinc -randkey HTTP/kudu2@DTSTACK.COM" kadmin.local -q "addprinc -randkey HTTP/kudu3@DTSTACK.COM"

kadmin.local -q "addprinc -randkey HTTP/nn1@DTSTACK.COM"

kadmin.local -q "addprinc -randkey HTTP/nn2@DTSTACK.COM"

创建完成后，查看：

$ kadmin.local -q "listprincs"

### 10.2 创建keytab文件

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。

创建包含 hdfs principal 和 host principal 的 hdfs keytab：

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：

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

注意： 上面的方法使用了xst的norandkey参数，有些kerberos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。

在 cdh1 节点，即 KDC server 节点上执行下面命令：

$ cd /var/kerberos/krb5kdc/ kadmin.local -q "xst -k hdfs-unmerged.keytab hdfs/kudu1@DTSTACK.COM" kadmin.local -q "xst -k hdfs-unmerged.keytab hdfs/kudu2@DTSTACK.COM" kadmin.local -q "xst -k hdfs-unmerged.keytab hdfs/kudu3@DTSTACK.COM" kadmin.local -q "xst -k HTTP.keytab HTTP/kudu1@DTSTACK.COM" kadmin.local -q "xst -k HTTP.keytab HTTP/kudu2@DTSTACK.COM" kadmin.local -q "xst -k HTTP.keytab HTTP/kudu3@DTSTACK.COM"

这样，就会在 /var/kerberos/krb5kdc/ 目录下生成 hdfs-unmerged.keytab 和 HTTP.keytab 两个文件，接下来使用 ktutil 合并者两个文件为 hdfs.keytab。

$ cd /var/kerberos/krb5kdc/ $ ktutil ktutil: rkt admin-unmerged.keytab ktutil: rkt HTTP.keytab ktutil: wkt admin.keytab ktutil: exit

使用 klist 显示 hdfs.keytab 文件列表：

$ klist -ket hdfs.keytab Keytab name: FILE:hdfs.keytab KVNO Timestamp Principal ---- ----------------- -------------------------------------------------------- 2 11/13/14 10:40:18 hdfs/cdh1@DTSTACK.COM (aes256-cts-hmac-sha1-96) 2 11/13/14 10:40:18 hdfs/cdh1@DTSTACK.COM (aes128-cts-hmac-sha1-96) 2 11/13/14 10:40:18 hdfs/cdh1@DTSTACK.COM (des3-cbc-sha1) 2 11/13/14 10:40:18 hdfs/cdh1@DTSTACK.COM (arcfour-hmac) 2 11/13/14 10:40:18 hdfs/cdh1@DTSTACK.COM (des-hmac-sha1) 2 11/13/14 10:40:18 hdfs/cdh1@DTSTACK.COM (des-cbc-md5) 2 11/13/14 10:40:18 hdfs/cdh2@DTSTACK.COM (aes256-cts-hmac-sha1-96) 2 11/13/14 10:40:18 hdfs/cdh2@DTSTACK.COM (aes128-cts-hmac-sha1-96) 2 11/13/14 10:40:18 hdfs/cdh2@DTSTACK.COM (des3-cbc-sha1) 2 11/13/14 10:40:18 hdfs/cdh2@DTSTACK.COM (arcfour-hmac) 2 11/13/14 10:40:18 hdfs/cdh2@DTSTACK.COM (des-hmac-sha1) 2 11/13/14 10:40:18 hdfs/cdh2@DTSTACK.COM (des-cbc-md5) 2 11/13/14 10:40:18 hdfs/cdh3@DTSTACK.COM (aes256-cts-hmac-sha1-96) 2 11/13/14 10:40:18 hdfs/cdh3@DTSTACK.COM (aes128-cts-hmac-sha1-96) 2 11/13/14 10:40:18 hdfs/cdh3@DTSTACK.COM (des3-cbc-sha1) 2 11/13/14 10:40:18 hdfs/cdh3@DTSTACK.COM (arcfour-hmac) 2 11/13/14 10:40:18 hdfs/cdh3@DTSTACK.COM (des-hmac-sha1) 2 11/13/14 10:40:18 hdfs/cdh3@DTSTACK.COM (des-cbc-md5) 2 11/13/14 10:40:18 HTTP/cdh1@DTSTACK.COM (aes256-cts-hmac-sha1-96) 2 11/13/14 10:40:18 HTTP/cdh1@DTSTACK.COM (aes128-cts-hmac-sha1-96) 2 11/13/14 10:40:18 HTTP/cdh1@DTSTACK.COM (des3-cbc-sha1) 2 11/13/14 10:40:18 HTTP/cdh1@DTSTACK.COM (arcfour-hmac) 2 11/13/14 10:40:18 HTTP/cdh1@DTSTACK.COM (des-hmac-sha1) 2 11/13/14 10:40:18 HTTP/cdh1@DTSTACK.COM (des-cbc-md5) 2 11/13/14 10:40:18 HTTP/cdh2@DTSTACK.COM (aes256-cts-hmac-sha1-96) 2 11/13/14 10:40:18 HTTP/cdh2@DTSTACK.COM (aes128-cts-hmac-sha1-96) 2 11/13/14 10:40:18 HTTP/cdh2@DTSTACK.COM (des3-cbc-sha1) 2 11/13/14 10:40:18 HTTP/cdh2@DTSTACK.COM (arcfour-hmac) 2 11/13/14 10:40:18 HTTP/cdh2@DTSTACK.COM (des-hmac-sha1) 2 11/13/14 10:40:18 HTTP/cdh2@DTSTACK.COM (des-cbc-md5) 2 11/13/14 10:40:18 HTTP/cdh3@DTSTACK.COM (aes256-cts-hmac-sha1-96) 2 11/13/14 10:40:18 HTTP/cdh3@DTSTACK.COM (aes128-cts-hmac-sha1-96) 2 11/13/14 10:40:18 HTTP/cdh3@DTSTACK.COM (des3-cbc-sha1) 2 11/13/14 10:40:18 HTTP/cdh3@DTSTACK.COM (arcfour-hmac) 2 11/13/14 10:40:18 HTTP/cdh3@DTSTACK.COM (des-hmac-sha1) 2 11/13/14 10:40:18 HTTP/cdh3@DTSTACK.COM (des-cbc-md5)

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。

$ kinit -k -t hdfs.keytab hdfs/kudu1@DTSTACK.COM $ kinit -k -t hdfs.keytab HTTP/kudu1@DTSTACK.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。

### 10.3 部署kerberos keytab文件

拷贝 hdfs.keytab 文件到其他节点的 /etc/hadoop/conf 目录

$ cd /var/kerberos/krb5kdc/ $ scp hdfs.keytab cdh1:/etc/hadoop/conf $ scp hdfs.keytab cdh2:/etc/hadoop/conf $ scp hdfs.keytab cdh3:/etc/hadoop/conf

并设置权限，分别在 cdh1、cdh2、cdh3 上执行：

$ ssh cdh1 "chown hdfs:hadoop /etc/hadoop/conf/hdfs.keytab ;chmod 400 /etc/hadoop/conf/hdfs.keytab" $ ssh cdh2 "chown hdfs:hadoop /etc/hadoop/conf/hdfs.keytab ;chmod 400 /etc/hadoop/conf/hdfs.keytab" $ ssh cdh3 "chown hdfs:hadoop /etc/hadoop/conf/hdfs.keytab ;chmod 400 /etc/hadoop/conf/hdfs.keytab"

由于 keytab 相当于有了永久凭证，不需要提供密码(如果修改kdc中的principal的密码，则该keytab就会失效)，所以其他用户如果对该文件有读权限，就可以冒充 keytab 中指定的用户身份访问 hadoop，所以 keytab 文件需要确保只对 owner 有读权限(0400)

### 10.4 修改 hdfs 配置文件

先停止集群：

$ for x in `cd /etc/init.d ; ls hive-\*` ; do sudo service $x stop ; done $ for x in `cd /etc/init.d ; ls impala-\*` ; do sudo service $x stop ; done $ for x in `cd /etc/init.d ; ls hadoop-\*` ; do sudo service $x stop ; done $ for x in `cd /etc/init.d ; ls zookeeper-\*` ; do sudo service $x stop ; done

在集群中所有节点的 core-site.xml 文件中添加下面的配置:

<property> <name>hadoop.security.authentication</name> <value>kerberos</value> </property> <property> <name>hadoop.security.authorization</name> <value>true</value> </property>

在集群中所有节点的 hdfs-site.xml 文件中添加下面的配置：

<property> <name>dfs.block.access.token.enable</name> <value>true</value> </property> <property> <name>dfs.datanode.data.dir.perm</name> <value>700</value> </property> <property> <name>dfs.namenode.keytab.file</name> <value>/etc/hadoop/conf/hdfs.keytab</value> </property> <property> <name>dfs.namenode.kerberos.principal</name> <value>hdfs/\_HOST@DTSTACK.COM</value> </property> <property> <name>dfs.namenode.kerberos.https.principal</name> <value>HTTP/\_HOST@DTSTACK.COM</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>/etc/hadoop/conf/hdfs.keytab</value> </property> <property> <name>dfs.datanode.kerberos.principal</name> <value>hdfs/\_HOST@DTSTACK.COM</value> </property> <property> <name>dfs.datanode.kerberos.https.principal</name> <value>HTTP/\_HOST@DTSTACK.COM</value> </property>

如果想开启 SSL，请添加（本文不对这部分做说明）：

<property> <name>dfs.http.policy</name> <value>HTTPS\_ONLY</value> </property>

如果 HDFS 配置了 QJM HA，则需要添加（另外，你还要在 zookeeper 上配置 kerberos）：

<property> <name>dfs.journalnode.keytab.file</name> <value>/etc/hadoop/conf/hdfs.keytab</value> </property> <property> <name>dfs.journalnode.kerberos.principal</name> <value>hdfs/\_HOST@DTSTACK.COM</value> </property> <property> <name>dfs.journalnode.kerberos.internal.spnego.principal</name> <value>HTTP/\_HOST@DTSTACK.COM</value> </property>

如果配置了 WebHDFS，则添加：

<property> <name>dfs.webhdfs.enabled</name> <value>true</value> </property> <property> <name>dfs.web.authentication.kerberos.principal</name> <value>HTTP/\_HOST@DTSTACK.COM</value> </property> <property> <name>dfs.web.authentication.kerberos.keytab</name> <value>/etc/hadoop/conf/hdfs.keytab</value> </property>

配置中有几点要注意的：

dfs.datanode.address表示 data transceiver RPC server 所绑定的 hostname 或 IP 地址，如果开启 security，端口号必须小于 1024(privileged port)，否则的话启动 datanode 时候会报 Cannot start secure cluster without privileged resources 错误

principal 中的 instance 部分可以使用 \_HOST 标记，系统会自动替换它为全称域名

如果开启了 security, hadoop 会对 hdfs block data(由 dfs.data.dir 指定)做 permission check，方式用户的代码不是调用hdfs api而是直接本地读block data，这样就绕过了kerberos和文件权限验证，管理员可以通过设置 dfs.datanode.data.dir.perm 来修改 datanode 文件权限，这里我们设置为700

### 10.5 检查集群上的 HDFS 和本地文件的权限

请参考 [Verify User Accounts and Groups in CDH 5 Due to Security](http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cdh_sg_users_groups_verify.html) 或者 [Hadoop in Secure Mode](http://hadoop.apache.org/docs/r2.5.0/hadoop-project-dist/hadoop-common/SecureMode.html)。

### 10.6 启动 NameNode

启动之前，请确认 JCE jar 已经替换，请参考前面的说明。

在每个节点上获取 root 用户的 ticket，这里 root 为之前创建的 root/admin 的密码。

$ ssh kudu1 "echo abc123|kinit root/admin" $ ssh kudu2 "echo abc123|kinit root/admin" $ ssh kudu3 "echo abc123|kinit root/admin"

获取 cdh1的 ticket：

$ kinit -k -t /etc/hadoop/conf/admin.keytab admin/kudu1@DTSTACK.COM

如果出现下面异常 kinit: Password incorrect while getting initial credentials，则重新导出 keytab 再试试。

然后启动服务，观察日志：

$ /etc/init.d/hadoop-hdfs-namenode start

验证 NameNode 是否启动，一是打开 web 界面查看启动状态，一是运行下面命令查看 hdfs：

$ hadoop fs -ls / Found 4 items drwxrwxrwx - yarn hadoop 0 2014-06-26 15:24 /logroot drwxrwxrwt - hdfs hadoop 0 2014-11-04 10:44 /tmp drwxr-xr-x - hdfs hadoop 0 2014-08-10 10:53 /user drwxr-xr-x - hdfs hadoop 0 2013-05-20 22:52 /var

如果在你的凭据缓存中没有有效的 kerberos ticket，执行上面命令将会失败，将会出现下面的错误：

14/11/04 12:08:12 WARN ipc.Client: Exception encountered while connecting to the server : javax.security.sasl.SaslException: GSS initiate failed [Caused by GS\*\*\*ception: No valid credentials provided (Mechanism level: Failed to find any Kerberos tgt)] Bad connection to FS. command aborted. exception: Call to cdh1/192.168.56.121:8020 failed on local exception: java.io.IOException: javax.security.sasl.SaslException: GSS initiate failed [Caused by GS\*\*\*ception: No valid credentials provided (Mechanism level: Failed to find any Kerberos tgt)]

### 10.7 启动DataNode

DataNode 需要通过 JSVC 启动。首先检查是否安装了 JSVC 命令，然后配置环境变量。

在 cdh1 节点查看是否安装了 JSVC：

$ ls /usr/lib/bigtop-utils/ bigtop-detect-classpath bigtop-detect-javahome bigtop-detect-javalibs jsvc

然后编辑 /etc/default/hadoop-hdfs-datanode，取消对下面的注释并添加一行设置 JSVC\_HOME，修改如下：

export HADOOP\_SECURE\_DN\_USER=hdfs export HADOOP\_SECURE\_DN\_PID\_DIR=/var/run/hadoop-hdfs export HADOOP\_SECURE\_DN\_LOG\_DIR=/var/log/hadoop-hdfs export JSVC\_HOME=/usr/lib/bigtop-utils

将该文件同步到其他节点：

$ scp /etc/default/hadoop-hdfs-datanode cdh2:/etc/default/hadoop-hdfs-datanode $ scp /etc/default/hadoop-hdfs-datanode cdh3:/etc/default/hadoop-hdfs-datanode

分别在 cdh2、cdh3 获取 ticket 然后启动服务：

#root 为 root/admin 的密码 $ ssh cdh1 "kinit -k -t /etc/hadoop/conf/hdfs.keytab hdfs/cdh1@DTSTACK.COM; service hadoop-hdfs-datanode start" $ ssh cdh2 "kinit -k -t /etc/hadoop/conf/hdfs.keytab hdfs/cdh2@DTSTACK.COM; service hadoop-hdfs-datanode start" $ ssh cdh3 "kinit -k -t /etc/hadoop/conf/hdfs.keytab hdfs/cdh3@DTSTACK.COM; service hadoop-hdfs-datanode start"

观看 cdh1 上 NameNode 日志，出现下面日志表示 DataNode 启动成功：

14/11/04 17:21:41 INFO security.UserGroupInformation: Login successful for user hdfs/cdh2@DTSTACK.COM using keytab file /etc/hadoop/conf/hdfs.keytab

## 11. 总结

本文介绍了 CDH Hadoop 集成 kerberos 认证的过程，其中主要需要注意以下几点：

配置 hosts，hostname 请使用小写

确保 kerberos 客户端和服务端连通

替换 JRE 自带的 JCE jar 包

为 DataNode 设置运行用户并配置 JSVC\_HOME

启动服务前，先获取 ticket 再运行相关命令

# YARN配置Kerberos认证

## 1. 生成 keytab

在 cdh1 节点，即 KDC server 节点上执行下面命令：

cd /var/kerberos/krb5kdc/ kadmin.local -q "addprinc -randkey yarn/cdh1@DTSTACK.COM " kadmin.local -q "addprinc -randkey yarn/cdh2@DTSTACK.COM " kadmin.local -q "addprinc -randkey yarn/cdh3@DTSTACK.COM " kadmin.local -q "addprinc -randkey mapred/cdh1@DTSTACK.COM " kadmin.local -q "addprinc -randkey mapred/cdh2@DTSTACK.COM " kadmin.local -q "addprinc -randkey mapred/cdh3@DTSTACK.COM " kadmin.local -q "xst -k yarn.keytab yarn/cdh1@DTSTACK.COM " kadmin.local -q "xst -k yarn.keytab yarn/cdh2@DTSTACK.COM " kadmin.local -q "xst -k yarn.keytab yarn/cdh3@DTSTACK.COM " kadmin.local -q "xst -k mapred.keytab mapred/cdh1@DTSTACK.COM " kadmin.local -q "xst -k mapred.keytab mapred/cdh2@DTSTACK.COM " kadmin.local -q "xst -k mapred.keytab mapred/cdh3@DTSTACK.COM "

拷贝 yarn.keytab 和 mapred.keytab 文件到其他节点的 /etc/hadoop/conf 目录

$ scp yarn.keytab mapred.keytab cdh1:/etc/hadoop/conf $ scp yarn.keytab mapred.keytab cdh2:/etc/hadoop/conf $ scp yarn.keytab mapred.keytab cdh3:/etc/hadoop/conf

并设置权限，分别在 cdh1、cdh2、cdh3 上执行：

$ ssh cdh1 "cd /etc/hadoop/conf/;chown yarn:hadoop yarn.keytab;chown mapred:hadoop mapred.keytab ;chmod 400 \*.keytab" $ ssh cdh2 "cd /etc/hadoop/conf/;chown yarn:hadoop yarn.keytab;chown mapred:hadoop mapred.keytab ;chmod 400 \*.keytab" $ ssh cdh3 "cd /etc/hadoop/conf/;chown yarn:hadoop yarn.keytab;chown mapred:hadoop mapred.keytab ;chmod 400 \*.keytab"

由于 keytab 相当于有了永久凭证，不需要提供密码(如果修改 kdc 中的 principal 的密码，则该 keytab 就会失效)，所以其他用户如果对该文件有读权限，就可以冒充 keytab 中指定的用户身份访问 hadoop，所以 keytab 文件需要确保只对 owner 有读权限(0400)

# 2. 修改 YARN 配置文件

修改 yarn-site.xml，添加下面配置：

<property> <name>yarn.resourcemanager.keytab</name> <value>/etc/hadoop/conf/yarn.keytab</value> </property> <property> <name>yarn.resourcemanager.principal</name> <value>yarn/\_HOST@DTSTACK.COM</value> </property> <property> <name>yarn.nodemanager.keytab</name> <value>/etc/hadoop/conf/yarn.keytab</value> </property> <property> <name>yarn.nodemanager.principal</name> <value>yarn/\_HOST@DTSTACK.COM</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>

如果想要 YARN 开启 SSL，则添加：

<property> <name>yarn.http.policy</name> <value>HTTPS\_ONLY</value> </property>

修改 mapred-site.xml，添加如下配置：

<property> <name>mapreduce.jobhistory.keytab</name> <value>/etc/hadoop/conf/mapred.keytab</value> </property> <property> <name>mapreduce.jobhistory.principal</name> <value>mapred/\_HOST@DTSTACK.COM</value> </property>

如果想要 mapreduce jobhistory 开启 SSL，则添加：

<property> <name>mapreduce.jobhistory.http.policy</name> <value>HTTPS\_ONLY</value> </property>

在 /etc/hadoop/conf 目录下创建 container-executor.cfg 文件，内容如下：

#configured value of yarn.nodemanager.linux-container-executor.group yarn.nodemanager.linux-container-executor.group=yarn #comma separated list of users who can not run applications banned.users=bin #Prevent other super-users min.user.id=0 #comma separated list of system users who CAN run applications allowed.system.users=root,nobody,impala,hive,hdfs,yarn

设置该文件权限：

$ chown root:yarn container-executor.cfg $ chmod 400 container-executor.cfg $ ll container-executor.cfg -r-------- 1 root yarn 354 11-05 14:14 container-executor.cfg

注意：

container-executor.cfg 文件读写权限需设置为 400，所有者为 root:yarn。

yarn.nodemanager.linux-container-executor.group 要同时配置在 yarn-site.xml 和 container-executor.cfg，且其值需要为运行 NodeManager 的用户所在的组，这里为 yarn。

banned.users 不能为空，默认值为 hfds,yarn,mapred,bin

min.user.id 默认值为 1000，在有些 centos 系统中，用户最小 id 为500，则需要修改该值

确保 yarn.nodemanager.local-dirs 和 yarn.nodemanager.log-dirs 对应的目录权限为 755 。

设置 /usr/lib/hadoop-yarn/bin/container-executor 读写权限为 6050 如下：

$ chown root:yarn /usr/lib/hadoop-yarn/bin/container-executor $ chmod 6050 /usr/lib/hadoop-yarn/bin/container-executor $ ll /usr/lib/hadoop-yarn/bin/container-executor ---Sr-s--- 1 root yarn 333 11-04 19:11 container-executor

测试是否配置正确：

$ /usr/lib/hadoop-yarn/bin/container-executor --checksetup

如果提示错误，则查看 NodeManger 的日志，然后对照 [YARN ONLY: Container-executor Error Codes](http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cdh_sg_other_hadoop_security.html?scroll=topic_18_unique_2) 查看错误对应的问题说明。

关于 LinuxContainerExecutor 的详细说明，可以参考 [http://hadoop.apache.org/docs/r2.5.0/hadoop-project-dist/hadoop-common/SecureMode.html#LinuxContainerExecutor](http://hadoop.apache.org/docs/r2.5.0/hadoop-project-dist/hadoop-common/SecureMode.html" \l "LinuxContainerExecutor)。

记住将修改的上面文件同步到其他节点：cdh2、cdh3，并再次一一检查权限是否正确。

$ cd /etc/hadoop/conf/ $ scp yarn-site.xml mapred-site.xml container-executor.cfg cdh2:/etc/hadoop/conf/ $ scp yarn-site.xml mapred-site.xml container-executor.cfg cdh3:/etc/hadoop/conf/ $ ssh cdh2 "cd /etc/hadoop/conf/; chown root:yarn container-executor.cfg ; chmod 400 container-executor.cfg" $ ssh cdh3 "cd /etc/hadoop/conf/; chown root:yarn container-executor.cfg ; chmod 400 container-executor.cfg"

# 3. 启动服务

启动 ResourceManager

resourcemanager 是通过 yarn 用户启动的，故在 cdh1 上先获取 yarn 用户的 ticket 再启动服务：

$ kinit -k -t /etc/hadoop/conf/yarn.keytab yarn/cdh1@DTSTACK.COM $ service hadoop-yarn-resourcemanager start

然后查看日志，确认是否启动成功。

启动 NodeManager

resourcemanager 是通过 yarn 用户启动的，故在 cdh2 和 cdh3 上先获取 yarn 用户的 ticket 再启动服务：

$ ssh cdh2 "kinit -k -t /etc/hadoop/conf/yarn.keytab yarn/cdh2@DTSTACK.COM ;service hadoop-yarn-nodemanager start" $ ssh cdh3 "kinit -k -t /etc/hadoop/conf/yarn.keytab yarn/cdh3@DTSTACK.COM ;service hadoop-yarn-nodemanager start"

启动 MapReduce Job History Server

resourcemanager 是通过 mapred 用户启动的，故在 cdh1 上先获取 mapred 用户的 ticket 再启动服务：

$ kinit -k -t /etc/hadoop/conf/mapred.keytab mapred/cdh1@DTSTACK.COM $ service hadoop-mapreduce-historyserver start

# 3.Hive配置Kerberos认证

## 1. 生成 keytab

在 cdh1 节点，即 KDC server 节点上执行下面命令：

$ cd /var/kerberos/krb5kdc/ kadmin.local -q "addprinc -randkey hive/cdh1@DTSTACK.COM " kadmin.local -q "addprinc -randkey hive/cdh2@DTSTACK.COM " kadmin.local -q "addprinc -randkey hive/cdh3@DTSTACK.COM " kadmin.local -q "xst -k hive.keytab hive/cdh1@DTSTACK.COM " kadmin.local -q "xst -k hive.keytab hive/cdh2@DTSTACK.COM " kadmin.local -q "xst -k hive.keytab hive/cdh3@DTSTACK.COM "

拷贝 hive.keytab 文件到其他节点的 /etc/hive/conf 目录

$ scp hive.keytab cdh1:/etc/hive/conf $ scp hive.keytab cdh2:/etc/hive/conf $ scp hive.keytab cdh3:/etc/hive/conf

并设置权限，分别在 cdh1、cdh2、cdh3 上执行：

$ ssh cdh1 "cd /etc/hive/conf/;chown hive:hadoop hive.keytab ;chmod 400 \*.keytab" $ ssh cdh2 "cd /etc/hive/conf/;chown hive:hadoop hive.keytab ;chmod 400 \*.keytab" $ ssh cdh3 "cd /etc/hive/conf/;chown hive:hadoop hive.keytab ;chmod 400 \*.keytab"

由于 keytab 相当于有了永久凭证，不需要提供密码(如果修改 kdc 中的 principal 的密码，则该 keytab 就会失效)，所以其他用户如果对该文件有读权限，就可以冒充 keytab 中指定的用户身份访问 hadoop，所以 keytab 文件需要确保只对 owner 有读权限(0400)

## 2. 修改 hive 配置文件

修改 hive-site.xml，添加下面配置：

<property> <name>hive.server2.authentication</name> <value>KERBEROS</value> </property> <property> <name>hive.server2.authentication.kerberos.principal</name> <value>hive/\_HOST@DTSTACK.COM</value> </property> <property> <name>hive.server2.authentication.kerberos.keytab</name> <value>/etc/hive/conf/hive.keytab</value> </property> <property> <name>hive.metastore.sasl.enabled</name> <value>true</value> </property> <property> <name>hive.metastore.kerberos.keytab.file</name> <value>/etc/hive/conf/hive.keytab</value> </property> <property> <name>hive.metastore.kerberos.principal</name> <value>hive/\_HOST@DTSTACK.COM</value> </property>

在 core-site.xml 中添加：

<property> <name>hadoop.proxyuser.hive.hosts</name> <value>\*</value> </property> <property> <name>hadoop.proxyuser.hive.groups</name> <value>\*</value> </property> <property> <name>hadoop.proxyuser.hdfs.hosts</name> <value>\*</value> </property> <property> <name>hadoop.proxyuser.hdfs.groups</name> <value>\*</value> </property> <property> <name>hadoop.proxyuser.HTTP.hosts</name> <value>\*</value> </property> <property> <name>hadoop.proxyuser.HTTP.groups</name> <value>\*</value> </property>

记住将修改的上面文件同步到其他节点：cdh2、cdh3，并再次一一检查权限是否正确。

$ scp /etc/hive/conf/hive-site.xml cdh2:/etc/hive/conf/ $ scp /etc/hive/conf/hive-site.xml cdh3:/etc/hive/conf/

## 3. 启动服务

启动 Hive MetaStore

hive-metastore 是通过 hive 用户启动的，故在 cdh1 上先获取 hive 用户的 ticket 再启动服务：

$ kinit -k -t /etc/hive/conf/hive.keytab hive/cdh1@DTSTACK.COM $ service hive-metastore start

然后查看日志，确认是否启动成功。

启动 Hive Server2

hive-server2 是通过 hive 用户启动的，故在 cdh2 和 cdh3 上先获取 hive 用户的 ticket 再启动服务：

$ kinit -k -t /etc/hive/conf/hive.keytab hive/cdh1@DTSTACK.COM $ service hive-server2 start

然后查看日志，确认是否启动成功。