HDFS配置Kerberos认证

2014.11.04 | Comments

本文主要记录 CDH Hadoop 集群上配置 HDFS 集成 Kerberos 的过程，包括 Kerberos 的安装和 Hadoop 相关配置修改说明。

1．环境说明

系统环境：

- 操作系统：CentOs 6.6
- Hadoop版本： CDH5.4
- JDK版本： 1.7.0_71
- 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

192.168.56.121 cdh1
192.168.56.122 cdh2
192.168.56.123 cdh3
```

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 = JAVACHEN.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]
 JAVACHEN.COM = {
  kdc = cdh1
  admin_server = cdh1
 }

[domain_realm]
 .javachen.com = JAVACHEN.COM
 javachen.com = JAVACHEN.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 = JAVACHEN.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]
 JAVACHEN.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：

- `JAVACHEN.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. 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上安装 JCE，如果你使用的是 JDK1.6 ，则到 Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files for JDK/JRE 6 (http://www.oracle.com/technetwork/java/javase/downloads/jce-6-download-429243.html) 页面下载，如果是 JDK1.7，则到 Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files for JDK/JRE 7 (http://www.oracle.com/technetwork/java/embedded/embedded-se/downloads/jce-7-download-432124.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@JAVACHEN.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 cdh2:/etc/krb5.conf
$ scp /etc/krb5.conf cdh3:/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 JAVACHEN.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@JAVACHEN.COM; defaulting to no policy
    Enter password for principal "user1@JAVACHEN.COM":
    Re-enter password for principal "user1@JAVACHEN.COM":
    Principal "user1@JAVACHEN.COM" created.

  # 删除 principal
  kadmin:  delprinc user1
    Are you sure you want to delete the principal "user1@JAVACHEN.COM"? (yes/no): yes
    Principal "user1@JAVACHEN.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@JAVACHEN.COM:

$ klist  -e
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: test@JAVACHEN.COM

Valid starting     Expires            Service principal
11/07/14 15:29:02  11/08/14 15:29:02  krbtgt/JAVACHEN.COM@JAVACHEN.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@JAVACHEN.COM:

$  klist
  Ticket cache: FILE:/tmp/krb5cc_0
  Default principal: root/admin@JAVACHEN.COM

  Valid starting     Expires            Service principal
  11/07/14 15:33:57  11/08/14 15:33:57  krbtgt/JAVACHEN.COM@JAVACHEN.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@JAVACHEN.COM

  Valid starting     Expires            Service principal
  11/07/14 15:34:05  11/08/14 15:34:05  krbtgt/JAVACHEN.COM@JAVACHEN.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@LASHOU-INC.COM
     3 kadmin/admin@LASHOU-INC.COM
     3 kadmin/admin@LASHOU-INC.COM
     3 kadmin/admin@LASHOU-INC.COM
     3 kadmin/admin@LASHOU-INC.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/cdh1@JAVACHEN.COM"
kadmin.local -q "addprinc -randkey hdfs/cdh2@JAVACHEN.COM"
kadmin.local -q "addprinc -randkey hdfs/cdh3@JAVACHEN.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/cdh1@JAVACHEN.COM"
kadmin.local -q "addprinc -randkey HTTP/cdh2@JAVACHEN.COM"
kadmin.local -q "addprinc -randkey HTTP/cdh3@JAVACHEN.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/cdh1@JAVACHEN.COM"
kadmin.local -q "xst  -k hdfs-unmerged.keytab  hdfs/cdh2@JAVACHEN.COM"
kadmin.local -q "xst  -k hdfs-unmerged.keytab  hdfs/cdh3@JAVACHEN.COM"

kadmin.local -q "xst  -k HTTP.keytab  HTTP/cdh1@JAVACHEN.COM"
kadmin.local -q "xst  -k HTTP.keytab  HTTP/cdh2@JAVACHEN.COM"
kadmin.local -q "xst  -k HTTP.keytab  HTTP/cdh3@JAVACHEN.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 hdfs-unmerged.keytab
ktutil: rkt HTTP.keytab
ktutil: wkt hdfs.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@JAVACHEN.COM (aes256-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 hdfs/cdh1@JAVACHEN.COM (aes128-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 hdfs/cdh1@JAVACHEN.COM (des3-cbc-sha1)
   2 11/13/14 10:40:18 hdfs/cdh1@JAVACHEN.COM (arcfour-hmac)
   2 11/13/14 10:40:18 hdfs/cdh1@JAVACHEN.COM (des-hmac-sha1)
   2 11/13/14 10:40:18 hdfs/cdh1@JAVACHEN.COM (des-cbc-md5)
   2 11/13/14 10:40:18 hdfs/cdh2@JAVACHEN.COM (aes256-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 hdfs/cdh2@JAVACHEN.COM (aes128-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 hdfs/cdh2@JAVACHEN.COM (des3-cbc-sha1)
   2 11/13/14 10:40:18 hdfs/cdh2@JAVACHEN.COM (arcfour-hmac)
   2 11/13/14 10:40:18 hdfs/cdh2@JAVACHEN.COM (des-hmac-sha1)
   2 11/13/14 10:40:18 hdfs/cdh2@JAVACHEN.COM (des-cbc-md5)
   2 11/13/14 10:40:18 hdfs/cdh3@JAVACHEN.COM (aes256-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 hdfs/cdh3@JAVACHEN.COM (aes128-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 hdfs/cdh3@JAVACHEN.COM (des3-cbc-sha1)
   2 11/13/14 10:40:18 hdfs/cdh3@JAVACHEN.COM (arcfour-hmac)
   2 11/13/14 10:40:18 hdfs/cdh3@JAVACHEN.COM (des-hmac-sha1)
   2 11/13/14 10:40:18 hdfs/cdh3@JAVACHEN.COM (des-cbc-md5)
   2 11/13/14 10:40:18 HTTP/cdh1@JAVACHEN.COM (aes256-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 HTTP/cdh1@JAVACHEN.COM (aes128-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 HTTP/cdh1@JAVACHEN.COM (des3-cbc-sha1)
   2 11/13/14 10:40:18 HTTP/cdh1@JAVACHEN.COM (arcfour-hmac)
   2 11/13/14 10:40:18 HTTP/cdh1@JAVACHEN.COM (des-hmac-sha1)
   2 11/13/14 10:40:18 HTTP/cdh1@JAVACHEN.COM (des-cbc-md5)
   2 11/13/14 10:40:18 HTTP/cdh2@JAVACHEN.COM (aes256-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 HTTP/cdh2@JAVACHEN.COM (aes128-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 HTTP/cdh2@JAVACHEN.COM (des3-cbc-sha1)
   2 11/13/14 10:40:18 HTTP/cdh2@JAVACHEN.COM (arcfour-hmac)
   2 11/13/14 10:40:18 HTTP/cdh2@JAVACHEN.COM (des-hmac-sha1)
   2 11/13/14 10:40:18 HTTP/cdh2@JAVACHEN.COM (des-cbc-md5)
   2 11/13/14 10:40:18 HTTP/cdh3@JAVACHEN.COM (aes256-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 HTTP/cdh3@JAVACHEN.COM (aes128-cts-hmac-sha1-96)
   2 11/13/14 10:40:18 HTTP/cdh3@JAVACHEN.COM (des3-cbc-sha1)
   2 11/13/14 10:40:18 HTTP/cdh3@JAVACHEN.COM (arcfour-hmac)
   2 11/13/14 10:40:18 HTTP/cdh3@JAVACHEN.COM (des-hmac-sha1)
   2 11/13/14 10:40:18 HTTP/cdh3@JAVACHEN.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/cdh1@JAVACHEN.COM
$ kinit -k -t hdfs.keytab HTTP/cdh1@JAVACHEN.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@JAVACHEN.COM</value>
</property>
<property>
  <name>dfs.namenode.kerberos.https.principal</name>
  <value>HTTP/_HOST@JAVACHEN.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@JAVACHEN.COM</value>
</property>
<property>
  <name>dfs.datanode.kerberos.https.principal</name>
  <value>HTTP/_HOST@JAVACHEN.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@JAVACHEN.COM</value>
</property>
<property>
  <name>dfs.journalnode.kerberos.internal.spnego.principal</name>
  <value>HTTP/_HOST@JAVACHEN.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@JAVACHEN.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：

- 1. 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
- 1. 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
- 1. 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 cdh1 "echo root|kinit root/admin"
$ ssh cdh1 "echo root|kinit root/admin"
$ ssh cdh1 "echo root|kinit root/admin"
```

获取 cdh1的 ticket：

```
$ kinit -k -t /etc/hadoop/conf/hdfs.keytab hdfs/cdh1@JAVACHEN.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@JAVACHEN.COM; service hadoop-hdfs-datanode start"
$ ssh cdh2 "kinit -k -t /etc/hadoop/conf/hdfs.keytab hdfs/cdh2@JAVACHEN.COM; service hadoop-hdfs-datanode start"
$ ssh cdh3 "kinit -k -t /etc/hadoop/conf/hdfs.keytab hdfs/cdh3@JAVACHEN.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@JAVACHEN.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：

- 1. 配置 hosts， hostname 请使用小写
- 1. 确保 kerberos 客户端和服务端连通
- 1. 替换 JRE 自带的 JCE jar 包
- 1. 为 DataNode 设置运行用户并配置 JSVC_HOME
- 1. 启动服务前，先获取 ticket 再运行相关命令

上面的过程比较繁琐，我总结了上面的过程并写了一些自动化的脚本方便快速安装、配置以及管理 kerberos，请参考Hadoop集群部署权限总结(/2014/11/25/quikstart-for-config-kerberos-ldap-and-sentry-in-hadoop.html)。

12．参考文章

- How-to: Quickly Configure Kerberos for Your Apache Hadoop Cluster (http://blog.cloudera.com/blog/2015/03/how-to-quickly-configure-kerberos-for-your-apache-hadoop-cluster/)
- Hadoop的kerberos的实践部署 (https://github.com/zouhc/MyHadoop/blob/master/doc/Hadoop%E7%9A%84kerberos%E7%9A%84%E5%AE%9E%E8%B7%B5%E9%83%A8%E7%BD%B2.md)
- hadoop 添加kerberos认证 (http://blog.chinaunix.net/uid-1838361-id-3243243.html)
- YARN & HDFS2 安装和配置Kerberos (http://blog.csdn.net/lalaguozhe/article/details/11570009)
- Kerberos basics and installing a KDC (http://blog.godatadriven.com/kerberos_kdc_install.html)
- Hadoop, Hbase, Zookeeper安全实践 (http://www.wuzesheng.com/?p=2345)