Hadoop+Kerberos+Sentry

0、参考资料

- **Kerberos：**

  https://web.mit.edu/kerberos/krb5-1.12/doc/index.html

- **Hadoop：**

  http://hadoop.apache.org/docs/r2.6.0/hadoop-project-dist/hadoop-common/SecureMode.html

  http://hadoop.apache.org/docs/r2.6.0/hadoop-project-dist/hadoop-common/core-default.xml http://hadoop.apache.org/docs/r2.6.0/hadoop-project-dist/hadoop-hdfs/hdfs-default.xml http://hadoop.apache.org/docs/r2.6.0/hadoop-mapreduce-client/hadoop-mapreduce-client-core/mapred-default.xml http://hadoop.apache.org/docs/r2.6.0/hadoop-yarn/hadoop-yarn-common/yarn-default.xml

- **集群部署说明**

主机	IP	组件
bd001	192.168.10.101	Kerberos客户端、NameNode、DataNode、SecondaryNameNode、NodeManager、HiveMetastore、HiveServer2、SentryServer
bd002	192.168.10.102	Kerberos客户端、ResourceManager、DataNode、NodeManager、JobHistoryServer、TimeLine
bd003	192.168.10.103	Kerberos客户端、DataNode、NodeManager
bd004	192.168.10.104	Kerberos KDC

- **软件清单**

组件	版本	下载地址
maven	3.3.9	http://archive.apache.org/dist/maven/maven-3/3.3.9/binaries/apache-maven-3.3.9-bin.tar.gz
jdk	1.8.0_202	https://download.oracle.com/otn/java/jdk/8u202-b08/1961070e4c9b4e26a04e7f5a083f551e/jdk-8u202-linux-x64.tar.gz
jce	8	http://download.oracle.com/otn-pub/java/jce/8/jce_policy-8.zip
hadoop	2.6.0	http://archive.cloudera.com/cdh5/cdh/5/hadoop-2.6.0-cdh5.14.4.tar.gz
protobuf	2.5.0	https://github.com/protocolbuffers/protobuf/tree/v2.5.0
hive	1.1.0	http://archive.cloudera.com/cdh5/cdh/5/hive-1.1.0-cdh5.14.4.tar.gz
kerberos	1.15.1	bd004（KDC）：yum install krb5-libs krb5-server krb5-workstation bd001~bd003：yum -y install krb5-libs krb5-workstation
sentry	1.5.1	http://archive.cloudera.com/cdh5/cdh/5/sentry-1.5.1-cdh5.14.4.tar.gz

1、安装MIT Kerberos的KDC（bd004）

1.1、使用yum安装KDC

```
yum install krb5-libs krb5-server krb5-workstation
```

1.2、配置/etc/krb5.conf

```
vim /etc/krb5.conf
```

```
configuration snippets may be placed in this directory as well

includedir /etc/krb5.conf.d/
```

```
[logging]
  default = FILE:/var/log/krb5libs.log
  kdc = FILE:/var/log/krb5kdc.log
  admin_server = FILE:/var/log/kadmind.log

[libdefaults]
  default_realm = BD004.COM
  dns_lookup_realm = false
  dns_lookup_kdc = false
  ticket_lifetime = 24h
  forwardable = true
  udp_preference_limit = 1000000
  default_tkt_enctypes = des-cbc-md5 des-cbc-crc des3-cbc-sha1
  default_tgs_enctypes = des-cbc-md5 des-cbc-crc des3-cbc-sha1
  permitted_enctypes = des-cbc-md5 des-cbc-crc des3-cbc-sha1

[realms]
  BD004.COM = {
    kdc = bd004:88
    admin_server = bd004:749
   default_domain = bd004
  }

[domain_realm]
  .bd004 = BD004.COM
  bd004 = BD004.COM
```

1.3、配置/var/kerberos/krb5kdc/kdc.conf

```
vim /var/kerberos/krb5kdc/kdc.conf
```

```
default_realm = BD004.COM

[kdcdefaults]
  kdc_ports = 0
  v4_mode = nopreauth

[realms]
  BD004.COM = {
    kdc_ports = 88
    admin_keytab = /etc/kadm5.keytab
    database_name = /var/kerberos/krb5kdc/principal
    acl_file = /var/kerberos/krb5kdc/kadm5.acl
    key_stash_file = /var/kerberos/krb5kdc/stash
    max_life = 10h 0m 0s
    max_renewable_life = 7d 0h 0m 0s
    master_key_type = des3-hmac-sha1
    supported_enctypes = arcfour-hmac:normal des3-hmac-sha1:normal des-cbc-crc:normal
des:normal des:v4 des:norealm des:onlyrealm des:afs3
    default_principal_flags = +preauth
  }
```

1.4、配置/var/kerberos/krb5kdc/kadm5.acl

```
vim /var/kerberos/krb5kdc/kadm5.acl
```

```
*/admin@BD004.COM       *
```

1.5、创建KDC数据库

```
kdb5_util create  -s -r BD004.COM
    输入密码：Enter KDC database master key:krb5kdc
    确认密码：Re-enter KDC database master key to verify:krb5kdc
    检查：ll /var/kerberos/krb5kdc/（必须包含kadm5.acl、kdc.conf、principal、principal.kadm5、
principal.kadm5.lock、principal.ok）
```

1.6、创建管理员

```
kadmin.local
    新增管理员用户：addprinc root/admin@BD004.COM（提示输入密码和确认密码，都是krb5kdc）
    新增测试用户：addprinc krbtest/admin@BD004.COM（提示输入密码和确认密码，都是krb5kdc）
    listprincs
```

1.7、重启Kerberos并设为开机启动

```
systemctl restart krb5kdc kadmin
systemctl enable krb5kdc kadmin
```

2、安装MIT Kerberos的客户端（bd001、bd002、bd003）

2.1、使用yum安装kerberos客户端

```
在bd001、bd002、bd003节点上都执行如下命令：
安装客户端：yum -y install krb5-libs krb5-workstation
同步KDC配置：scp root@bd004:/etc/krb5.conf /etc/krb5.conf
```

2.2、验证客户端能否连接到KDC服务

2.2.1、验证客户端使用测试用户能否连接到KDC

```
kinit krbtest/admin@BD004.COM（提示输入密码，如果校验成功，kerberos不会有任何提示）
```

2.2.2、再验证客户端登录kadmin

登录KDC命令：kadmin（提示输入密码，登录成功使用listprincs查看是否与KDC节点kadmin.local查看到的内容一致，一致则成功）

3、创建hadoop相关组件所需的用户（bd001、bd002、bd003）

3.1、创建hadoop组件用户

```
新增hadoop用户组：groupadd hadoop
新增hdfs用户：adduser hdfs -g hadoop -p hdfs
新增hdfs用户：adduser yarn -g hadoop -p yarn
新增hdfs用户：adduser mapred -g hadoop -p mapred
检查用户所属组：groups hdfs yarn mapred
```

3.2、设置hadoop组件使用路径的权限

3.2.1、建议设置的hdfs路径权限

文件系统	配置属性指定的路径	User:Group	Permissions
local	dfs.namenode.name.dir	hdfs:hadoop	drwx------
local	dfs.datanode.data.dir	hdfs:hadoop	drwx------
local	$HADOOP_LOG_DIR	hdfs:hadoop	drwxrwxr-x
local	$YARN_LOG_DIR	yarn:hadoop	drwxrwxr-x
local	yarn.nodemanager.local-dirs	yarn:hadoop	drwxr-xr-x
local	yarn.nodemanager.log-dirs	yarn:hadoop	drwxr-xr-x
local	container-executor	root:hadoop	---Sr-s---
local	container-executor.cfg	root:hadoop	-r--------
hdfs	/	hdfs:hadoop	drwxr-xr-x
hdfs	/tmp	hdfs:hadoop	drwxrwxrwxt
hdfs	/user	hdfs:hadoop	drwxr-xr-x
hdfs	yarn.nodemanager.remote-app-log-dir	yarn:hadoop	drwxrwxrwxt
hdfs	mapreduce.jobhistory.intermediate-done-dir	mapred:hadoop	drwxrwxrwxt
hdfs	mapreduce.jobhistory.done-dir	mapred:hadoop	drwxr-x---

3.2.2、设置hdfs和yarn组件所需路径的权限（bd001、bd002、bd003）

脚本文件所需权限：

```
    chgrp hadoop -R /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/distribute-
exclude.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/hadoop-daemon.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/hadoop-daemons.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/hdfs-config.cmd
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/hdfs-config.sh
    chown yarn:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/mr-jobhistory-
daemon.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/refresh-namenodes.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/slaves.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/start-all.cmd
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/start-all.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/start-balancer.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/start-dfs.cmd
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/start-dfs.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/start-secure-dns.sh
    chown yarn:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/start-yarn.cmd
    chown yarn:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/start-yarn.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/stop-all.cmd
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/stop-all.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/stop-balancer.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/stop-dfs.cmd
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/stop-dfs.sh
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/stop-secure-dns.sh
    chown yarn:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/stop-yarn.cmd
    chown yarn:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/stop-yarn.sh
    chown yarn:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/yarn-daemon.sh
    chown yarn:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin/yarn-daemons.sh
    chown mapred:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/bin/mapred*
    chown yarn:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/bin/yarn*
    chown hdfs:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/bin/hdfs*
```

脚本文件所需路径权限：

```
    HADOOP_CONF_DIR=/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/etc/hadoop ( chown
hdfs:hadoop , chmod 755 )
    HADOOP_LOG_DIR=/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/logs/hdfs ( chown
hdfs:hadoop , chmod 775 -R  )
    YARN_LOG_DIR=/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/logs/yarn ( chown  yarn:hadoop ,
chmod 775 -R  )
```

设置hdfs本地文件系统路径权限

```
    dfs.namenode.name.dir=/usr/local/cdh5.14.4/hadoop-2.6.0-
cdh5.14.4/metadata/hdfs/name ( chown  hdfs:hadoop , chmod 700 -R  )
    dfs.namenode.data.dir=/usr/local/cdh5.14.4/hadoop-2.6.0-
cdh5.14.4/metadata/hdfs/data ( chown  hdfs:hadoop , chmod 700 -R  )
```

设置yarn

```
    yarn.nodemanager.local-dirs=/usr/local/cdh5.14.4/hadoop-2.6.0-
cdh5.14.4/metadata/yarn/nodemanager ( chown  yarn:hadoop , chmod 755 -R )
    yarn.nodemanager.log-dirs=/usr/local/cdh5.14.4/hadoop-2.6.0-
cdh5.14.4/logs/yarn/nodemanager ( chown  yarn:hadoop , chmod 755 -R )
    yarn.nodemanager.remote-app-log-dir=/logs/remote_app_log ( chown  yarn:hadoop )
    yarn.nodemanager.linux-container-executor.path=/usr/local/cdh5.14.4/hadoop-2.6.0-
cdh5.14.4/metadata/yarn/nodemanager/container
```

设置hdfs文件系统路径权限：

```
    /（hdfs dfs -chown hdfs:hadoop）
    /user（hdfs dfs -chown -R hdfs:hadoop）
    /tmp（hdfs dfs -chown -R hdfs:hadoop）
    yarn.nodemanager.remote-app-log-dir=/logs/remote_app_log（hdfs dfs -chown -R
yarn:hadoop）
    mapreduce.jobhistory.intermediate-done-dir=/mr-history/tmp（hdfs dfs -chown -R
mapred:hadoop）
    mapreduce.jobhistory.done-dir=/mr-history/done（hdfs dfs -chown -R mapred:hadoop）
设置keytabs文件权限
    在bd001上：
        ll /etc/security/keytabs/
        chmod 400 /etc/security/keytabs/*
            dn.service.keytab（chown hdfs:hadoop）
            nn.service.keytab（chown hdfs:hadoop）
            sn.service.keytab（chown hdfs:hadoop）
            spnego.service.keytab（chown hdfs:hadoop）
            nm.service.keytab（chown yarn:hadoop）
    在bd002上：
        ll /etc/security/keytabs/
        chmod 400 /etc/security/keytabs/*
            dn.service.keytab（chown hdfs:hadoop）
            jhs.service.keytab（chown mapred:hadoop）
            nm.service.keytab（chown yarn:hadoop）
            rm.service.keytab（chown yarn:hadoop）
            spnego.service.keytab（chown yarn:hadoop）
            tl.service.keytab（chown yarn:hadoop）
    在bd003上：
        ll /etc/security/keytabs/
        chmod 400 /etc/security/keytabs/*
            dn.service.keytab（chown hdfs:hadoop）
            nm.service.keytab（chown yarn:hadoop）
```

3.2.3、建议设置hadoop各守护进程的principals和keytabs

服务	组件	Principal名称	keytab名称
HDFS	NameNode	nn/bd001@BD004.COM	nn.service.keytab
HDFS	NameNode HTTP	HTTP/bd001@BD004.COM	spnego.service.keytab
HDFS	SecondaryNameNode	sn/bd001@BD004.COM	sn.service.keytab
HDFS	SecondaryNameNode HTTP	HTTP/bd001@BD004.COM	spnego.service.keytab
HDFS	DataNode	dn/bd001@BD004.COM	dn.service.keytab
HDFS	DataNode	dn/bd002@BD004.COM	dn.service.keytab
HDFS	DataNode	dn/bd003@BD004.COM	dn.service.keytab
MR2	HistoryServer	jhs/bd002@BD004.COM	jhs.service.keytab
MR2	HistoryServer HTTP	HTTP/bd002@BD004.COM	spnego.service.keytab
YARN	ResourceManager	rm/bd002@BD004.COM	rm.service.keytab
YARN	NodeManager	nm/bd001@BD004.COM	nm.service.keytab
YARN	NodeManager	nm/bd002@BD004.COM	nm.service.keytab
YARN	NodeManager	nm/bd003@BD004.COM	nm.service.keytab
YARN	TimelineServer	tl/bd002@BD004.COM	tl.service.keytab

3.2.3、创建hdfs各个守护进程对应Kerberos认证的principals和keytabs文件

```
在每个节点中执行：mkdir /etc/security/keytabs
在bd001中创建：
    addprinc -randkey nn/bd001@BD004.COM
    addprinc -randkey HTTP/bd001@BD004.COM
    addprinc -randkey sn/bd001@BD004.COM
    addprinc -randkey dn/bd001@BD004.COM
    addprinc -randkey nm/bd001@BD004.COM
    ktadd -k /etc/security/keytabs/nn.service.keytab nn/bd001@BD004.COM
    ktadd -k /etc/security/keytabs/spnego.service.keytab HTTP/bd001@BD004.COM
    ktadd -k /etc/security/keytabs/sn.service.keytab sn/bd001@BD004.COM
    ktadd -k /etc/security/keytabs/dn.service.keytab dn/bd001@BD004.COM
    ktadd -k /etc/security/keytabs/nm.service.keytab nm/bd001@BD004.COM
在bd002中创建：
    addprinc -randkey dn/bd002@BD004.COM
    addprinc -randkey jhs/bd002@BD004.COM
    addprinc -randkey HTTP/bd002@BD004.COM
    addprinc -randkey rm/bd002@BD004.COM
    addprinc -randkey nm/bd002@BD004.COM
    addprinc -randkey tl/bd002@BD004.COM
    ktadd -k /etc/security/keytabs/dn.service.keytab dn/bd002@BD004.COM
    ktadd -k /etc/security/keytabs/jhs.service.keytab jhs/bd002@BD004.COM
    ktadd -k /etc/security/keytabs/spnego.service.keytab HTTP/bd002@BD004.COM
```

```
    ktadd -k /etc/security/keytabs/rm.service.keytab rm/bd002@BD004.COM
    ktadd -k /etc/security/keytabs/nm.service.keytab nm/bd002@BD004.COM
    ktadd -k /etc/security/keytabs/tl.service.keytab tl/bd002@BD004.COM
在bd003中创建：
    addprinc -randkey dn/bd003@BD004.COM
    addprinc -randkey nm/bd003@BD004.COM
    ktadd -k /etc/security/keytabs/dn.service.keytab dn/bd003@BD004.COM
    ktadd -k /etc/security/keytabs/nm.service.keytab nm/bd003@BD004.COM
```

3.3.4、创建https证书

```
mkdir /etc/security/cdh5.14.4.https
cd /etc/security/cdh5.14.4.https
在bd001节点中创建CA证书：
    openssl req -new -x509 -keyout bd_ca_key -out bd_ca_cert -days 9999 -subj
'/C=CN/ST=beijing/L=beijing/O=test/OU=test/CN=test'（输入密码和确认密码是123456，此命令成功后输出
bd_ca_key和bd_ca_cert两个文件）
    同步到bd002、bd003上
        scp -r /etc/security/cdh5.14.4.https bd002:/etc/security/
        scp -r /etc/security/cdh5.14.4.https bd003:/etc/security/
        keytool -keystore keystore -alias localhost -validity 9999 -genkey -keyalg RSA -
keysize 2048 -dname "CN=test, OU=test, O=test, L=beijing, ST=beijing, C=CN"（输入密码和确认密
码：123456，此命令成功后输出keystore文件）
        keytool -keystore truststore -alias CARoot -import -file bd_ca_cert（输入密码和确认密
码：123456，提示是否信任证书：输入yes，此命令成功后输出truststore文件）
        keytool -certreq -alias localhost -keystore keystore -file cert（输入密码和确认密码：
123456，此命令成功后输出cert文件）
        openssl x509 -req -CA bd_ca_cert -CAkey bd_ca_key -in cert -out cert_signed -days
9999 -CAcreateserial -passin pass:123456（此命令成功后输出cert_signed文件）
        keytool -keystore keystore -alias CARoot -import -file bd_ca_cert（输入密码和确认密
码：123456，是否信任证书，输入yes，此命令成功后更新keystore文件）
        keytool -keystore keystore -alias localhost -import -file cert_signed（输入密码和确认
密码：123456）
在bd002节点中：
    keytool -keystore keystore -alias localhost -validity 9999 -genkey -keyalg RSA -keysize
2048 -dname "CN=test, OU=test, O=test, L=beijing, ST=beijing, C=CN"（输入密码和确认密码：
123456，此命令成功后输出keystore文件）
    keytool -keystore truststore -alias CARoot -import -file bd_ca_cert（输入密码和确认密码：
123456，提示是否信任证书：输入yes，此命令成功后输出truststore文件）
    keytool -certreq -alias localhost -keystore keystore -file cert（输入密码和确认密码：
123456，此命令成功后输出cert文件）
    openssl x509 -req -CA bd_ca_cert -CAkey bd_ca_key -in cert -out cert_signed -days 9999
-CAcreateserial -passin pass:123456（此命令成功后输出cert_signed文件）
    keytool -keystore keystore -alias CARoot -import -file bd_ca_cert（输入密码和确认密码：
123456，是否信任证书，输入yes，此命令成功后更新keystore文件）
    keytool -keystore keystore -alias localhost -import -file cert_signed（输入密码和确认密
码：123456）
在bd003节点中：
    keytool -keystore keystore -alias localhost -validity 9999 -genkey -keyalg RSA -keysize
2048 -dname "CN=test, OU=test, O=test, L=beijing, ST=beijing, C=CN"（输入密码和确认密码：
123456，此命令成功后输出keystore文件）
    keytool -keystore truststore -alias CARoot -import -file bd_ca_cert（输入密码和确认密码：
123456，提示是否信任证书：输入yes，此命令成功后输出truststore文件）
```

```
    keytool -certreq -alias localhost -keystore keystore -file cert（输入密码和确认密码：123456，此命令成功后输出cert文件）
    openssl x509 -req -CA bd_ca_cert -CAkey bd_ca_key -in cert -out cert_signed -days 9999 -CAcreateserial -passin pass:123456（此命令成功后输出cert_signed文件）
    keytool -keystore keystore -alias CARoot -import -file bd_ca_cert（输入密码和确认密码：123456，是否信任证书，输入yes，此命令成功后更新keystore文件）
    keytool -keystore keystore -alias localhost -import -file cert_signed（输入密码和确认密码：123456）
```

4、配置hadoop

4.1、在bd001节点上

```
cd /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4
```

4.1.1、配置hadoop-env.sh

```
vim etc/hadoop/hadoop-env.sh
    export JAVA_HOME=/usr/jdk
    export HADOOP_CONF_DIR=/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/etc/hadoop
    export HADOOP_LOG_DIR=/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/logs/hdfs
```

4.1.2、配置yarn-env.sh

```
vim etc/hadoop/yarn-env.sh
    export JAVA_HOME=/usr/jdk
    export YARN_CONF_DIR=/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/etc/hadoop
    export YARN_LOG_DIR=/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/logs/yarn
```

4.1.3、配置mapred-env.sh

```
vim etc/hadoop/mapred-env.sh
    export JAVA_HOME=/usr/jdk
```

4.1.4、配置core-site.xml

```
vim etc/hadoop/core-site.xml
```

```
<configuration>
    <!-- config defaultFS -->
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://bd001:9000</value>
    </property>
    <!-- Enable Hadoop Security -->
    <property>
```

```
        <name>hadoop.security.authorization</name>
        <value>true</value>
    </property>
    <property>
        <name>hadoop.security.authentication</name>
        <value>kerberos</value>
    </property>
    <!-- Mapping from Kerberos principals to OS user accounts -->
    <property>
        <name>hadoop.security.auth_to_local</name>
        <value>
                RULE:[2:$1@$0](nn/.*@.*BD004.COM)s/.*/hdfs/
                RULE:[2:$1@$0](sn/.*@.*BD004.COM)s/.*/hdfs/
                RULE:[2:$1@$0](dn/.*@.*BD004.COM)s/.*/hdfs/
                RULE:[2:$1@$0](nm/.*@.*BD004.COM)s/.*/yarn/
                RULE:[2:$1@$0](rm/.*@.*BD004.COM)s/.*/yarn/
                RULE:[2:$1@$0](tl/.*@.*BD004.COM)s/.*/yarn/
                RULE:[2:$1@$0](jhs/.*@.*BD004.COM)s/.*/mapred/
                RULE:[2:$1@$0](HTTP/.*@.*BD004.COM)s/.*/hdfs/
                DEFAULT
        </value>
    </property>
</configuration>
```

4.1.5、配置hdfs-site.xml

```
vim etc/hadoop/hdfs-site.xml
```

```
<configuration>
    <!-- General HDFS security config-->
    <property>
        <name>dfs.block.access.token.enable</name>
        <value>true</value>
    </property>
    <property>
        <name>dfs.namenode.name.dir</name>
        <value>/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/metadata/hdfs/name</value>
    </property>
    <property>
        <name>dfs.datanode.data.dir</name>
        <value>/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/metadata/hdfs/data</value>
    </property>
    <!-- NameNode security config -->
    <property>
        <name>dfs.namenode.kerberos.principal</name>
        <value>nn/_HOST@BD004.COM</value>
    </property>
    <property>
        <name>dfs.namenode.keytab.file</name>
        <!-- path to the HDFS keytab -->
        <value>/etc/security/keytabs/nn.service.keytab</value>
    </property>
```

```
    <property>
        <name>dfs.namenode.kerberos.internal.spnego.principal</name>
        <value>HTTP/_HOST@BD004.COM</value>
    </property>
    <!--Secondary NameNode security config -->
    <property>
        <name>dfs.secondary.namenode.kerberos.principal</name>
        <value>sn/_HOST@BD004.COM</value>
    </property>
    <property>
        <name>dfs.secondary.namenode.keytab.file</name>
        <!-- path to the HDFS keytab -->
        <value>/etc/security/keytabs/sn.service.keytab</value>
    </property>
    <property>
        <name>dfs.secondary.namenode.kerberos.internal.spnego.principal</name>
        <value>HTTP/_HOST@BD004.COM</value>
    </property>
    <!-- DataNode security config -->
    <property>
        <name>dfs.datanode.kerberos.principal</name>
        <value>dn/_HOST@BD004.COM</value>
    </property>
    <property>
        <name>dfs.datanode.keytab.file</name>
        <!-- path to the HDFS keytab -->
        <value>/etc/security/keytabs/dn.service.keytab</value>
    </property>
    <property>
        <name>dfs.datanode.data.dir.perm</name>
        <value>700</value>
    </property>
    <property>
        <name>dfs.datanode.address</name>
        <value>0.0.0.0:61004</value>
    </property>
    <property>
        <name>dfs.datanode.http.address</name>
        <value>0.0.0.0:61006</value>
    </property>
    <!--configure secure WebHDFS -->
    <property>
        <name>dfs.http.policy</name>
        <value>HTTPS_ONLY</value>
    </property>
    <property>
        <name>dfs.data.transfer.protection</name>
        <value>integrity</value>
    </property>
    <property>
        <name>dfs.https.port</name>
        <value>50470</value>
    </property>
```

```
    <property>
        <name>dfs.https.address</name>
        <value>bd001:50470</value>
    </property>
    <property>
        <name>dfs.webhdfs.enabled</name>
        <value>true</value>
    </property>
    <property>
        <name>dfs.web.authentication.kerberos.principal</name>
        <value>HTTP/_HOST@BD004.COM</value>
    </property>
    <property>
        <name>dfs.web.authentication.kerberos.keytab</name>
        <value>/etc/security/keytabs/spnego.service.keytab</value>
    </property>
    <property>
        <name>dfs.permissions.supergroup</name>
        <value>hdfs</value>
    </property>
</configuration>
```

4.1.6、配置ssl-server.xml和ssl-client.xml

```
cp etc/hadoop/ssl-server.xml.example etc/hadoop/ssl-server.xml
vim etc/hadoop/ssl-server.xml
```

```
<configuration>
    <property>
        <name>ssl.server.truststore.location</name>
        <value>/etc/security/cdh5.14.4.https/truststore</value>
    </property>
    <property>
        <name>ssl.server.truststore.password</name>
        <value>123456</value>
    </property>
    <property>
        <name>ssl.server.truststore.type</name>
        <value>jks</value>
    </property>
    <property>
        <name>ssl.server.truststore.reload.interval</name>
        <value>10000</value>
    </property>
    <property>
        <name>ssl.server.keystore.location</name>
        <value>/etc/security/cdh5.14.4.https/keystore</value>
    </property>
    <property>
        <name>ssl.server.keystore.password</name>
        <value>123456</value>
    </property>
```

```
    <property>
        <name>ssl.server.keystore.keypassword</name>
        <value>123456</value>
    </property>
    <property>
        <name>ssl.server.keystore.type</name>
        <value>jks</value>
    </property>
    <property>
        <name>ssl.server.exclude.cipher.list</name>
        <value>TLS_ECDHE_RSA_WITH_RC4_128_SHA,SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA,
SSL_RSA_WITH_DES_CBC_SHA,SSL_DHE_RSA_WITH_DES_CBC_SHA,
SSL_RSA_EXPORT_WITH_RC4_40_MD5,SSL_RSA_EXPORT_WITH_DES40_CBC_SHA,
SSL_RSA_WITH_RC4_128_MD5</value>
    </property>
</configuration>
```

```
cp etc/hadoop/ssl-client.xml.example etc/hadoop/ssl-client.xml
vim etc/hadoop/ssl-client.xml
```

```
<configuration>
    <property>
        <name>ssl.client.truststore.location</name>
        <value>/etc/security/cdh5.14.4.https/truststore</value>
    </property>
    <property>
        <name>ssl.client.truststore.password</name>
        <value>123456</value>
    </property>
    <property>
        <name>ssl.client.truststore.type</name>
        <value>jks</value>
    </property>
    <property>
        <name>ssl.client.truststore.reload.interval</name>
        <value>10000</value>
    </property>
    <property>
        <name>ssl.client.keystore.location</name>
        <value>/etc/security/cdh5.14.4.https/keystore</value>
    </property>
    <property>
        <name>ssl.client.keystore.password</name>
        <value>123456</value>
    </property>
    <property>
        <name>ssl.client.keystore.keypassword</name>
        <value></value>
    </property>
    <property>
        <name>ssl.client.keystore.type</name>
        <value>jks</value>
```

```
    </property>
</configuration>
```

4.1.7、配置mapred-site.xml

```
vim etc/hadoop/mapred-site.xml
```

```
<configuration>
    <!-- General MapReduce configs -->
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.intermediate-done-dir</name>
        <value>/mr-history/tmp</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.done-dir</name>
        <value>/mr-history/done</value>
    </property>
    <!-- MapReduce Job History Server security configs -->
    <property>
        <name>mapreduce.jobhistory.address</name>
        <value>bd002:10020</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.webapp.address</name>
        <value>bd002:19888</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.keytab</name>
        <value>/etc/security/keytabs/jhs.service.keytab</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.principal</name>
        <value>jhs/_HOST@BD004.COM</value>
    </property>
    <!-- MapReduce Job History Webapp security configs -->
    <property>
        <name>mapreduce.jobhistory.webapp.address</name>
        <value>bd002:19888</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.webapp.spnego-principal</name>
        <value>HTTP/_HOST@BD004.COM</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.webapp.spnego-keytab-file</name>
        <value>/etc/security/keytabs/spnego.service.keytab</value>
    </property>
    <!-- To enable SSL -->
```

```
    <property>
        <name>mapreduce.jobhistory.http.policy</name>
        <value>HTTPS_ONLY</value>
    </property>
</configuration>
```

4.1.8、配置yarn-site.xml

```
vim etc/hadoop/yarn-site.xml
```

```
<configuration>
    <!-- General Yarn configs -->
    <property>
        <name>yarn.nodemanager.local-dirs</name>
        <value>/usr/local/cdh5.14.4/hadoop-2.6.0-
cdh5.14.4/metadata/yarn/nodemanager</value>
    </property>
    <property>
        <name>yarn.nodemanager.log-dirs</name>
        <value>/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/logs/yarn/nodemanager</value>
    </property>
    <property>
        <name>yarn.nodemanager.remote-app-log-dir</name>
        <value>/logs/remote_app_log</value>
    </property>
    <!-- ResourceManager security configs -->
    <property>
        <name>yarn.resourcemanager.principal</name>
        <value>rm/_HOST@BD004.COM</value>
    </property>
    <property>
        <name>yarn.resourcemanager.keytab</name>
        <value>/etc/security/keytabs/rm.service.keytab</value>
    </property>
    <property>
        <name>yarn.resourcemanager.webapp.delegation-token-auth-filter.enabled</name>
        <value>true</value>
    </property>
    <!-- NodeManager security configs -->
    <property>
        <name>yarn.nodemanager.principal</name>
        <value>nm/_HOST@BD004.COM</value>
    </property>
    <property>
        <name>yarn.nodemanager.keytab</name>
        <value>/etc/security/keytabs/nm.service.keytab</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.path</name>
        <value>/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/bin/container-executor</value>
    </property>
    <property>
        <name>yarn.nodemanager.linux-container-executor.group</name>
        <value>hadoop</value>
    </property>
    <!-- TimeLine security configs -->
    <property>
        <name>yarn.timeline-service.principal</name>
        <value>tl/_HOST@BD004.COM</value>
    </property>
    <property>
        <name>yarn.timeline-service.keytab</name>
        <value>/etc/security/keytabs/tl.service.keytab</value>
    </property>
    <property>
        <name>yarn.timeline-service.http-authentication.type</name>
        <value>kerberos</value>
    </property>
    <property>
        <name>yarn.timeline-service.http-authentication.kerberos.principal</name>
        <value>HTTP/_HOST@BD004.COM</value>
    </property>
    <property>
        <name>yarn.timeline-service.http-authentication.kerberos.keytab</name>
        <value>/etc/security/keytabs/spnego.service.keytab</value>
    </property>
    <!-- To enable SSL -->
    <property>
        <name>yarn.http.policy</name>
        <value>HTTPS_ONLY</value>
    </property>
</configuration>
```

4.1.9、配置container-executor.cfg (chown root:hadoop，chmod 400)

```
vim etc/hadoop/container-executor.cfg
```

```
yarn.nodemanager.local-dirs=/usr/local/cdh5.14.4/hadoop-2.6.0-
cdh5.14.4/metadata/yarn/nodemanager
yarn.nodemanager.log-dirs=/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/logs/yarn/nodemanager
yarn.nodemanager.linux-container-executor.group=hadoop
banned.users=hdfs,yarn,mapred,bin
min.user.id=1000
```

4.1.10、配置slaves

```
vim etc/hadoop/slaves
    bd001
    bd002
    bd003
```

4.1.11、编译LinuxContainerExecutor

```
先安装apache-maven-3.3.9配置好环境变量
    cd /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/src/hadoop-yarn-project/hadoop-yarn/hadoop-yarn-server/hadoop-yarn-server-nodemanager（如果是apache hadoop则要下载bin对应的src包）
    修改pom.xml（在<build>标签中添加<defaultGoal>compile</defaultGoal>）
    mvn package -Pdist,native -DskipTests -Dtar -Dcontainer-executor.conf.dir=/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/etc/hadoop/
        报错1： hadoop-yarn-server-nodemanager: org.apache.maven.plugin.MojoExecutionException: 'protoc --version' did not return a version
            yum -y groupinstall "Development Tools"
            yum install ant
            yum install cmake
            安装protobuf-2.5.0.tar.gz解压，进入protobuf-2.5.0，执行./configure && make && make check && make install（查看pom.xml或lib下的protobuf-version.jar确定protobuf包版本。使用命令protoc --version验证，输出libprotoc 2.5.0为安装成功）
        报错2：(use of '_' as an identifier might not be supported in releases after Java SE 8)
            使用JDK7编译hadoop-2.6.0的源码，在apache-maven-3.3.9/bin/mvn脚本中添加JAVA_HOME=/usr/local/softs/jdk1.7.0_65
            编译完成后，在/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/src/hadoop-yarn-project/hadoop-yarn/hadoop-yarn-server/hadoop-yarn-server-nodemanager/target/native/target/usr/local/bin路径下找有一个编译好的container-executor，复制到$HADOOP_YARN_HOME/bin（hadoop-2.6.0/bin）配置路径即可
            查看可执行文件是否存在：ll /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/src/hadoop-yarn-project/hadoop-yarn/hadoop-yarn-server/hadoop-yarn-server-nodemanager/target/native/target/usr/local/bin/container-executor
            复制到hadoop-2.6.0/bin下：cp container-executor /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/bin/
            chown root:hadoop /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/bin/container-executor
            chmod 6050 /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/bin/container-executor（权限为---Sr-s---，container-extractor可执行文件只能由root用户和hadoop组的成员执行，并且可执行文件使用有效的uid是root，有效的gid是hadoop）
            测试container-executor配置是否正确：
            /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/bin/container-executor -checksetup（需把/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/etc/hadoop所有文件夹所有者必须为root，出现Usage: container-executor --checksetup表示成功）
```

4.1.12、格式化HDFS（在bd001上使用root用户格式化hdfs）

```
[root@bd001 ~]# cd /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4
[root@bd001 hadoop-2.6.0-cdh5.14.4]# ./bin/hdfs namenode -format
```

4.1.13、将bd001上配置好的hadoop同步到bd002和bd003上

```
scp -r /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4 root@bd002:/usr/local/cdh5.14.4/
scp -r /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4 root@bd003:/usr/local/cdh5.14.4/
```

4.2、启动hdfs（在bd001上）

切换到hdfs用户：

```
su - hdfs
cd /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/
```

配置hdfs需要以kerberos principal用户登陆到hdfs，使用kinit命令初始化登陆用户，即用户nn/bd001@BD004.COM登录。

```
kinit -kt /etc/security/keytabs/nn.service.keytab nn/bd001@BD004.COM
```

启动hdfs

```
sudo ./sbin/start-dfs.sh
```

验证hdfs WebUI

```
https://bd001:50470
https://bd001:50091
```

```
[hdfs@bd001 ~]$ hdfs dfs -ls -R /
19/05/05 06:49:01 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
-rw-r--r--   3 nn     hadoop      14978 2019-04-29 04:00 /NOTICE.txt
-rw-r--r--   3 nn     hadoop       1366 2019-04-29 03:45 /README.txt
drwxr-xr-x   - yarn   hadoop          0 2019-04-29 03:34 /logs
drwxr-xr-x   - yarn   hadoop          0 2019-04-29 03:34 /logs/remote_app_log
drwxr-xr-x   - mapred hadoop          0 2019-04-29 03:35 /mr-history
drwxr-xr-x   - mapred hadoop          0 2019-04-29 03:35 /mr-history/done
drwxr-xr-x   - mapred hadoop          0 2019-04-29 03:35 /mr-history/tmp
drwxrwxrwx   - hdfs   hadoop          0 2019-05-05 02:39 /tmp
drwx------   - hive   hadoop          0 2019-05-05 02:39 /tmp/hadoop-yarn
drwx------   - hive   hadoop          0 2019-05-05 02:39 /tmp/hadoop-yarn/staging
drwx------   - hive   hadoop          0 2019-05-05 02:39 /tmp/hadoop-yarn/staging/hive
drwx------   - hive   hadoop          0 2019-05-05 02:39 /tmp/hadoop-yarn/staging/hive/.staging
drwx-wx-wx   - hive   hadoop          0 2019-04-30 04:35 /tmp/hive
drwx------   - hive   hadoop          0 2019-05-05 06:48 /tmp/hive/hive
drwx------   - hive   hadoop          0 2019-05-05 06:21 /tmp/hive/hive/300a7e00-44a0-4734-943d-1550b64dadd4
drwx------   - hive   hadoop          0 2019-05-05 06:21 /tmp/hive/hive/300a7e00-44a0-4734-943d-1550b64dadd4/_tmp_space.db
drwx------   - hive   hadoop          0 2019-05-05 06:48 /tmp/hive/hive/889adfe4-3d12-47e1-ad0a-40b85d96dfb6
drwx------   - hive   hadoop          0 2019-05-05 06:48 /tmp/hive/hive/889adfe4-3d12-47e1-ad0a-40b85d96dfb6/_tmp_space.db
drwx------   - hive   hadoop          0 2019-05-05 03:04 /tmp/hive/hive/c4ab4afc-0e54-4ba6-87cb-5a5b88875a69
drwx------   - hive   hadoop          0 2019-05-05 03:04 /tmp/hive/hive/c4ab4afc-0e54-4ba6-87cb-5a5b88875a69/_tmp_space.db
drwxrwxr--   - hdfs   hadoop          0 2019-04-30 04:38 /user
drwxrwxr--   - hive   hadoop          0 2019-04-30 04:38 /user/hive
drwxrwx---   - hive   hive            0 2019-05-05 02:38 /user/hive/warehouse
drwxrwx---   - hive   hive            0 2019-05-05 03:21 /user/hive/warehouse/tbl_test_1
[hdfs@bd001 ~]$
```

4.3、启动yarn（在bd002上）

切换到yarn用户：

```
su - yarn
cd /usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/
```

配置yarn需要以kerberos principal用户登陆到yarn，使用kinit命令初始化登陆用户，即用户rm/bd002@BD004.COM登录。

```
kinit -kt /etc/security/keytabs/rm.service.keytab rm/bd002@BD004.COM
```

启动yarn

```
sudo ./sbin/start-yarn.sh
```

验证yarn WebUI

```
https://bd002:8090
```

启动HistoryServer

```
mr-jobhistory-daemon.sh start historyserver
```

验证historyserver WebUI

```
https://bd002:19890/
```

```
[yarn@bd002 ~]$ yarn application -list
19/05/05 06:51:50 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
19/05/05 06:51:50 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Total number of applications (application-types: [] and states: [SUBMITTED, ACCEPTED, RUNNING]):0
                Application-Id      Application-Name        Application-Type          User           Queue                   State             Final-State             Progress                     Tracking-URL
[yarn@bd002 ~]$
```

5、配置hive

5.1、创建hive用户

```
adduser hive -g hadoop
passwd hive ( 密码hive )
```

5.2、在bd001节点中配置Kerberos用户

```
addprinc -randkey hive/bd001@BD004.COM
ktadd -k /etc/security/keytabs/hive.keytab hive/bd001@BD004.COM
chown hive:hadoop /etc/security/keytabs/hive.keytab
chmod 400 /etc/security/keytabs/hive.keytab
```

5.3、解压hive并添加mysql-connector-java-5.1.34.jar

```
su - hive
cd /usr/local/cdh5.14.4/
tar -zxf hive-1.1.0-cdh5.14.4.tar.gz
cd hive-1.1.0-cdh5.14.4
上传mysql-connector-java-5.1.34.jar到lib/下
```

5.4、配置hive-env.sh

```
export JAVA_HOME=/usr/jdk
HADOOP_HOME=/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4
export HIVE_CONF_DIR=/usr/local/cdh5.14.4/hive-1.1.0-cdh5.14.4/conf
```

5.5、配置hive-site.xml

```
vim hive-site.xml
```

```
<property>
    <name>javax.jdo.option.ConnectionURL</name>
    <value>jdbc:mysql://bd001:3306/hive?createDatabaseIfNotExist=true</value>
</property>
    <name>javax.jdo.option.ConnectionDriverName</name>
    <value>com.mysql.jdbc.Driver</value>
</property>
<property>
    <name>javax.jdo.option.ConnectionUserName</name>
    <value>root</value>
```

```
</property>
<property>
    <name>javax.jdo.option.ConnectionPassword</name>
    <value>Abcd1234@</value>
</property>
<property>
    <name>hive.server2.authentication</name>
    <value>KERBEROS</value>
</property>
<property>
    <name>hive.server2.authentication.kerberos.principal</name>
    <value>hive/_HOST@BD004.COM</value>
</property>
<property>
    <name>hive.server2.authentication.kerberos.keytab</name>
    <value>/etc/security/keytabs/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/security/keytabs/hive.keytab</value>
</property>
<property>
    <name>hive.metastore.kerberos.principal</name>
    <value>hive/_HOST@BD004.COM</value>
</property>
```

5.6、配置core-site.xml

```
vim core-site.xml
```

```
<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>
```

5.7、初始化hive元存储（在bd001）

```
./bin/schematool -dbType mysql -initSchema -verbose
```

5.8、启动hive服务（在bd001）

```
kinit -kt /etc/security/keytabs/hive.keytab hive/bd001@BD001.COM
```

5.8.1、启动HiveMetastore

```
nohup ./bin/hive --service metastore &
```

5.8.2、启动HiveServer2

```
nohup ./bin/hive --service hiveserver2 &
```

5.9、验证Hive的CLI和Beeline

5.9.1、验证CLI

```
./bin/hive --service cli
```

5.9.2、验证Beeline

```
./bin/beeline
0: jdbc:hive2://bd001:10000/default> !connect
jdbc:hive2://bd001:10000/default;principal=hive/bd001@BD001.COM
```

```
[hive@bd001 ~]$ /usr/local/cdh5.14.4/hive-1.1.0-cdh5.14.4/bin/beeline
which: no hbase in (.:/usr/local/cdh5.14.4/hive-1.1.0-cdh5.14.4/bin:/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/bin:/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/sbin
in:/home/hive/.local/bin:/home/hive/bin)
Beeline version 1.1.0-cdh5.14.4 by Apache Hive
beeline> !connect jdbc:hive2://bd001:10000/default;principal=hive/bd001@BD001.COM
scan complete in 3ms
Connecting to jdbc:hive2://bd001:10000/default;principal=hive/bd001@BD001.COM
19/05/05 03:08:53 [main]: WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Connected to: Apache Hive (version 1.1.0-cdh5.14.4)
Driver: Hive JDBC (version 1.1.0-cdh5.14.4)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://bd001:10000/default> use default;
INFO  : Compiling command(queryId=hive_20190505030909_9f685032-e431-4d9a-89dc-be3e53977fe0): use default
INFO  : Semantic Analysis Completed
INFO  : Returning Hive schema: Schema(fieldSchemas:null, properties:null)
INFO  : Completed compiling command(queryId=hive_20190505030909_9f685032-e431-4d9a-89dc-be3e53977fe0); Time taken: 3.395 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20190505030909_9f685032-e431-4d9a-89dc-be3e53977fe0): use default
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hive_20190505030909_9f685032-e431-4d9a-89dc-be3e53977fe0); Time taken: 0.283 seconds
INFO  : OK
No rows affected (4.987 seconds)
0: jdbc:hive2://bd001:10000/default> show tables;
INFO  : Compiling command(queryId=hive_20190505030909_3026f843-e816-4db3-8a9d-0eeb4ae4f8ea): show tables
INFO  : Semantic Analysis Completed
INFO  : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:tab_name, type:string, comment:from deserializer)], properties:null)
INFO  : Completed compiling command(queryId=hive_20190505030909_3026f843-e816-4db3-8a9d-0eeb4ae4f8ea); Time taken: 1.188 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20190505030909_3026f843-e816-4db3-8a9d-0eeb4ae4f8ea): show tables
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hive_20190505030909_3026f843-e816-4db3-8a9d-0eeb4ae4f8ea); Time taken: 0.5 seconds
INFO  : OK
+------------+--+
|  tab_name  |
+------------+--+
| tbl_test_1 |
+------------+--+
1 row selected (2.654 seconds)
0: jdbc:hive2://bd001:10000/default>
```

6、安装Sentry

6.1、创建sentry用户和kerberos的凭证、秘钥

```
创建sentry用户
    useradd sentry -g hadoop -p sentry
创建kerberos所需的凭证和秘钥
    chown sentry:hadoop /etc/security/keytabs/sentry.keytab
    chmod 400 /etc/security/keytabs/sentry.keytab
    addprinc -randkey sentry/bd001@BD004.COM
    ktadd -k /etc/security/keytabs/sentry.keytab sentry/bd001@BD004.COM
配置Hive Warehouse路径的权限（chmod 770，chown hive:hive）
    groupadd hive
    hdfs dfs -chmod -R 770 /user/hive/warehouse
    hdfs dfs -chown -R hive:hive /user/hive/warehouse
su - sentry
cd /usr/local/cdh5.14.4/apache-sentry-1.5.1-cdh5.14.4-bin
```

6.2、创建mysql的sentry用户和元数据库

```
CREATE DATABASE sentry;
CREATE USER sentry IDENTIFIED BY 'Abcd1234@';
GRANT all ON sentry.* TO sentry@'%' IDENTIFIED BY'Abcd1234@';
FLUSH PRIVILEGES;
```

6.3、配置sentry-site.xml

```
cp /usr/local/cdh5.14.4/apache-sentry-1.5.1-cdh5.14.4-bin/conf/sentry-
site.xml.service.template /usr/local/cdh5.14.4/apache-sentry-1.5.1-cdh5.14.4-
bin/conf/sentry-site.xml
vim /usr/local/cdh5.14.4/apache-sentry-1.5.1-cdh5.14.4-bin/conf/sentry-site.xml
```

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
    <property>
        <name>sentry.service.admin.group</name>
        <value>hive</value>
    </property>
    <property>
        <name>sentry.service.allow.connect</name>
        <value>hive</value>
    </property>
    <property>
        <name>sentry.hive.server</name>
        <value>server1</value>
    </property>
    <!-- 配置webserver -->
    <property>
        <name>sentry.service.web.enable</name>
        <value>true</value>
    </property>
    <property>
        <name>sentry.service.web.port</name>
        <value>51000</value>
    </property>
    <!--  开启kerberos认证 -->
    <property>
        <name>sentry.service.security.mode</name>
        <value>kerberos</value>
    </property>
    <property>
        <name>sentry.service.server.principal</name>
        <value>sentry/bd001@BD004.COM</value>
    </property>
    <property>
        <name>sentry.service.server.keytab</name>
        <value>/etc/security/keytabs/sentry.keytab</value>
    </property>
    <!--  配置jdbc -->
    <property>
        <name>sentry.verify.schema.version</name>
        <value>true</value>
    </property>
    <property>
        <name>sentry.store.jdbc.driver</name>
        <value>com.mysql.jdbc.Driver</value>
    </property>
    <property>
```

```
        <name>sentry.store.jdbc.url</name>
        <value>jdbc:mysql://bd001:3306/sentry</value>
    </property>
    <property>
        <name>sentry.store.jdbc.user</name>
        <value>sentry</value>
    </property>
    <property>
        <name>sentry.store.jdbc.password</name>
        <value>Abcd1234@</value>
    </property>
</configuration>
```

6.4、添加mysql包到sentry库中

```
cp /usr/local/cdh5.14.4/hive-1.1.0-cdh5.14.4/lib/mysql-connector-java-5.1.34.jar
/usr/local/cdh5.14.4/apache-sentry-1.5.1-cdh5.14.4-bin/lib/
```

6.5、初始化sentry元数据库

```
sentry --command schema-tool --conffile /usr/local/cdh5.14.4/apache-sentry-1.5.1-cdh5.14.4-
bin/conf/sentry-site.xml --dbType mysql --initSchema
```

6.6、获取sentry的ticket并启动sentry服务

```
kinit -k -t /etc/security/keytabs/sentry.keytab sentry/bd001@BD004.COM
sentry --command service --conffile /usr/local/cdh5.14.4/apache-sentry-1.5.1-cdh5.14.4-
bin/conf/sentry-site.xml
```

```
[root@bd001 ~]# /usr/local/cdh5.14.4/apache-sentry-1.5.1-cdh5.14.4-bin/bin/sentry  --command service --conffile /usr/local/cdh5.14.4/apache-sentry-1.5.1-cdh5.14.4-bin/conf/sentry-site.xml
19/05/05 06:18:54 INFO thrift.SentryService: Configured on address /0.0.0.0:8038
19/05/05 06:18:54 INFO thrift.SentryService: Using kerberos principal: sentry/bd001@BD004.COM
19/05/05 06:18:55 INFO DataNucleus.Persistence: Property datanucleus.cache.level2 unknown - will be ignored
19/05/05 06:18:57 WARN bonecp.BoneCPConfig: Max Connections < 1. Setting to 20
19/05/05 06:19:03 WARN DataNucleus.MetaData: Metadata has jdbc-type of CHAR(40) yet this is not valid. Ignored
19/05/05 06:19:05 WARN bonecp.BoneCPConfig: Max Connections < 1. Setting to 20
19/05/05 06:19:06 INFO thrift.LeaderStatusMonitor: Leader election protocol disabled, assuming single active server
19/05/05 06:19:06 INFO thrift.SentryService: Attempting to start...
19/05/05 06:19:06 INFO thrift.SentryKerberosContext: Logging in with new Context
Debug is  true storeKey true useTicketCache true useKeyTab true doNotPrompt true ticketCache is null isInitiator false KeyTab is /etc/security/keytabs/sentry.keytab refreshKrb5Config is tr
 storePass is false clearPass is false
Refreshing Kerberos configuration
Acquire TGT from Cache
Principal is sentry/bd001@BD004.COM
null credentials from Ticket Cache
principal is sentry/bd001@BD004.COM
Will use keytab
Commit Succeeded

19/05/05 06:19:06 INFO thrift.SentryService: sentry store cleaner is scheduled with interval 43200 seconds
19/05/05 06:19:06 INFO persistent.SentryStore: Purging MSentryPathUpdate and MSentyPermUpdate tables, leaving 200 entries
19/05/05 06:19:06 INFO persistent.SentryStore: MSentryPermChange table has been purged.
19/05/05 06:19:07 INFO persistent.SentryStore: MSentryPathUpdate table has been purged.
19/05/05 06:19:08 INFO thrift.SentryService: Metastore uri is not configured. Do not start HMSFollower
19/05/05 06:19:08 INFO thrift.SentryService: ProcessorFactory being used: org.apache.sentry.provider.db.service.thrift.SentryPolicyStoreProcessorFactory
19/05/05 06:19:08 INFO thrift.SentryService: ProcessorFactory being used: org.apache.sentry.provider.db.generic.service.thrift.SentryGenericPolicyProcessorFactory
19/05/05 06:19:09 INFO DataNucleus.Persistence: Property datanucleus.cache.level2 unknown - will be ignored
19/05/05 06:19:10 WARN DataNucleus.MetaData: Metadata has jdbc-type of CHAR(40) yet this is not valid. Ignored
19/05/05 06:19:11 WARN bonecp.BoneCPConfig: Max Connections < 1. Setting to 20
19/05/05 06:19:11 WARN bonecp.BoneCPConfig: Max Connections < 1. Setting to 20
19/05/05 06:19:11 INFO thrift.SentryService: Serving on /0.0.0.0:8038
19/05/05 06:19:11 INFO thrift.SentryService: Sentry service is ready to serve client requests
Sentry service is ready to serve client requests
```

6.7、配置sentry-hive.xml

```
cp /usr/local/cdh5.14.4/apache-sentry-1.5.1-cdh5.14.4-bin/conf/sentry-site.xml.hive-
client.example /usr/local/cdh5.14.4/hive-1.1.0-cdh5.14.4/conf/sentry-hive.xml
```

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
    <property>
        <name>sentry.hive.provider</name>
<value>org.apache.sentry.provider.file.HadoopGroupResourceAuthorizationProvider</value>
    </property>
    <property>
        <name>sentry.hive.server</name>
        <value>server1</value>
    </property>
    <property>
        <name>sentry.hive.testing.mode</name>
        <value>false</value>
    </property>
    <property>
        <name>sentry.service.client.server.rpc-port</name>
        <value>8038</value>
    </property>
    <property>
        <name>sentry.service.client.server.rpc-addresses</name>
        <value>bd001</value>
    </property>
    <property>
        <name>sentry.service.client.server.rpc-connection-timeout</name>
        <value>200000</value>
    </property>
    <property>
        <name>sentry.hive.provider.backend</name>
        <value>org.apache.sentry.provider.db.SimpleDBProviderBackend</value>
    </property>
    <property>
        <name>sentry.service.security.mode</name>
        <value>kerberos</value>
    </property>
    <property>
        <name>sentry.service.server.principal</name>
        <value>sentry/bd001@BD004.COM</value>
    </property>
    <property>
        <name>sentry.metastore.service.users</name>
        <value>hive</value>
    </property>
</configuration>
```

6.8、添加hive的sentry支持

```
cp /usr/local/cdh5.14.4/apache-sentry-1.5.1-cdh5.14.4-bin/lib/sentry*.jar
/usr/local/cdh5.14.4/hive-1.1.0-cdh5.14.4/lib/
```

6.9、重启hive服务

```
su - hive
[hive@ ~] /usr/local/cdh5.14.4/hive-1.1.0-cdh5.14.4/bin/hive --service metastore
[hive@ ~] /usr/local/cdh5.14.4/hive-1.1.0-cdh5.14.4/bin/hive --service hiveserver2
```

7、集成测试

maven依赖如图：

```
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <junit.version>4.12</junit.version>
        <cdh.hadoop.version>2.6.0-cdh5.14.4</cdh.hadoop.version>
        <cdh.hive.version>1.1.0-cdh5.14.4</cdh.hive.version>
        <cdh.sentry.version>1.5.1-cdh5.14.4</cdh.sentry.version>
        <wagon-ssh.version>3.1.0</wagon-ssh.version>
        <maven-compiler-plugin.version>3.6.0</maven-compiler-plugin.version>
        <maven-shade-plugin.version>3.2.1</maven-shade-plugin.version>
        <wagon-maven-plugin.version>2.0.0</wagon-maven-plugin.version>
    </properties>

    <dependencies>
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>${junit.version}</version>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>org.apache.hadoop</groupId>
            <artifactId>hadoop-common</artifactId>
            <version>${cdh.hadoop.version}</version>
        </dependency>
        <dependency>
            <groupId>org.apache.hadoop</groupId>
            <artifactId>hadoop-hdfs</artifactId>
            <version>${cdh.hadoop.version}</version>
        </dependency>
        <dependency>
            <groupId>org.apache.hive</groupId>
            <artifactId>hive-exec</artifactId>
            <version>${cdh.hive.version}</version>
        </dependency>
        <dependency>
            <groupId>org.apache.sentry</groupId>
            <artifactId>sentry-binding-hive</artifactId>
            <version>${cdh.sentry.version}</version>
        </dependency>
    </dependencies>
```

7.1、HDFS测试

7.1.1、切换到hdfs用户

```
上传security.jar到/usr/local/cdh5.14.4
su - hdfs
cd /usr/local/cdh5.14.4
kinit -k -t /etc/security/keytabs/nn.service.keytab nn/bd001@BD004.COM
java -cp security.jar com.itcast.security.hadoop.FSTools
```

7.1.2、测试代码：

```java
package com.itcast.security.hadoop;

import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Arrays;
import java.util.Date;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IOUtils;
import org.apache.hadoop.security.UserGroupInformation;

/**
 *
 * @author mengyao
 *
 */
public class FSTools {

    private Configuration conf;
    private static FileSystem fs;


    FSTools(){
        this("nn/bd001@BD004.COM", "/etc/security/keytabs/nn.service.keytab");
    }

    FSTools(String principal, String keytab){
        initiali(principal, keytab);
    }

    /**
     *
     * @param principal
     * @param keytab
     */
    private void initiali(String principal, String keytab) {
        try {
            conf = new Configuration();
            conf.set("fs.hdfs.impl", "org.apache.hadoop.hdfs.DistributedFileSystem");
            fs = FileSystem.newInstance(conf);
            UserGroupInformation.setConfiguration(conf);
            UserGroupInformation.loginUserFromKeytab(principal, keytab);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    /**
     *
     * @param path
     */
    public void list(String path) {
        try {
            Arrays.asList(fs.listStatus(new Path(path))).forEach(f -> {
                    System.out.println(
                            f.getPermission().toString()+"\t"+
                            f.getReplication()+"\t"+
                            f.getLen()+"\t"+
                            new SimpleDateFormat("yyyy-MM-dd hh:MM:ss").format(new Date(f.getAccessTime()))+"\t"+
                            f.getPath().toString());
            });
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IllegalArgumentException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    /**
     *
     * @param src
     * @param dist
     */
    public void upload(String src, String dist) {
        try {
            IOUtils.copyBytes(
                    new BufferedInputStream(new FileInputStream(src)),
                    fs.create(new Path(dist), true, 65536),
                    fs.getConf(),
                    true);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IllegalArgumentException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    /**
     *
     * @param dfsPath
     * @param localPath
     */
    public void download(String dfsPath, String localPath) {
        try {
            IOUtils.copyBytes(
                    fs.open(new Path(dfsPath)),
```

```
                fs.open(new Path(dfsPath)),
                new BufferedOutputStream(new FileOutputStream(localPath)),
                fs.getConf(),
                true);
        } catch (IllegalArgumentException e) {
            e.printStackTrace();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    /**
     *
     * @param file
     */
    public void delete(String file) {
        try {
            Path path = new Path(file);
            if (fs.exists(path)) {
                System.out.println(fs.delete(path, true));
            }
        } catch (IllegalArgumentException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    /**
     *
     */
    public void cleaner() {
        try {
            fs.close();
            conf.clear();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        System.setProperty("java.security.krb5.conf", "/etc/krb5.conf");
        args = new String[] {"/", "/usr/local/cdh5.14.4/hadoop-2.6.0-cdh5.14.4/logs/hdfs/hadoop-root-namenode-bd001.log", "/nn.log", "/home/hdfs/nn.log"};
        if (args.length < 4) {
            System.out.println("Usage: input params require is 4!");
            System.exit(1);
        }
        FSTools fsTools = new FSTools();
        System.out.println("==== 查询 ====");
        fsTools.list(args[0]);
        System.out.println("==== 上传 ====");
        fsTools.upload(args[1], args[2]);
        System.out.println("==== 查询 ====");
        fsTools.list(args[0]);
        System.out.println("==== 下载 ====");
        fsTools.download(args[2], args[3]);
        System.out.println("==== 删除 ====");
        fsTools.delete(args[2]);
        System.out.println("==== 查询 ====");
        fsTools.list(args[0]);
    }

}
```

```
[hdfs@bd001 cdh5.14.4]$ kinit -k -t /etc/security/keytabs/nn.service.keytab nn/bd001@BD004.COM
[hdfs@bd001 cdh5.14.4]$ java -cp security.jar com.itcast.security.hadoop.FSTools
log4j:WARN No appenders could be found for logger (org.apache.hadoop.metrics2.lib.MutableMetricsFactory).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
==== 查询 ====
rw-r--r--       3       14978   2019-04-29 04:04:50     hdfs://bd001:9000/NOTICE.txt
rw-r--r--       3       1366    2019-04-29 03:04:35     hdfs://bd001:9000/README.txt
rwxr-xr-x       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/logs
rwxr-xr-x       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/mr-history
rwxrwxrwx       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/tmp
rwxrwxr--       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/user
==== 上传 ====
==== 查询 ====
rw-r--r--       3       14978   2019-04-29 04:04:50     hdfs://bd001:9000/NOTICE.txt
rw-r--r--       3       1366    2019-04-29 03:04:35     hdfs://bd001:9000/README.txt
rwxr-xr-x       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/logs
rwxr-xr-x       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/mr-history
rw-r--r--       3       14392235        2019-05-06 05:05:43     hdfs://bd001:9000/nn.log
rwxrwxrwx       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/tmp
rwxrwxr--       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/user
==== 下载 ====
==== 删除 ====
true
==== 查询 ====
rw-r--r--       3       14978   2019-04-29 04:04:50     hdfs://bd001:9000/NOTICE.txt
rw-r--r--       3       1366    2019-04-29 03:04:35     hdfs://bd001:9000/README.txt
rwxr-xr-x       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/logs
rwxr-xr-x       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/mr-history
rwxrwxrwx       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/tmp
rwxrwxr--       0       0       1969-12-31 07:12:00     hdfs://bd001:9000/user
[hdfs@bd001 cdh5.14.4]$ ll ~
total 16320
-rw-r--r--. 1 hdfs hadoop 14392235 May  6 05:28 nn.log
[hdfs@bd001 cdh5.14.4]$
```

7.2、YARN作业测试

7.2.1、切换到yarn用户

```
su - yarn
cd /usr/local/cdh5.14.4
kinit -k -t /etc/security/keytabs/rm.service.keytab rm/bd002@BD004.COM
hadoop jar security.jar com.itcast.security.hadoop.WordCountApp
hdfs://bd001:9000/README.txt hdfs://bd001:9000/apps/wc/out rm/bd002@BD004.COM
/etc/security/keytabs/rm.service.keytab
```

7.2.1、测试代码：

```java
package com.itcast.security.hadoop;

import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.security.UserGroupInformation;
import org.apache.hadoop.util.GenericOptionsParser;

public class WordCountApp {
    public static class TokenizerMapper extends Mapper<Object, Text, Text, IntWritable> {
        private static final IntWritable one = new IntWritable(1);
        private Text word = new Text();

        public void map(Object key, Text value, Mapper<Object, Text, Text, IntWritable>.Context context)
                throws IOException, InterruptedException {
            StringTokenizer itr = new StringTokenizer(value.toString());
            while (itr.hasMoreTokens()) {
                this.word.set(itr.nextToken());
                context.write(this.word, one);
            }
        }
    }

    public static class IntSumReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
        private IntWritable result = new IntWritable();
        public void reduce(Text key, Iterable<IntWritable> values,
                Reducer<Text, IntWritable, Text, IntWritable>.Context context)
                throws IOException, InterruptedException {
            int sum = 0;
            for (IntWritable val : values) {
                sum += val.get();
            }
            this.result.set(sum);
            context.write(key, this.result);
        }
    }

    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
        if (otherArgs.length < 4) {
            System.err.println("Usage: wordcount <in> <out> <principal> <keytab>");
            System.exit(1);
        }
        UserGroupInformation.setConfiguration(conf);
        UserGroupInformation.loginUserFromKeytab(otherArgs[2], otherArgs[3]);
        Job job = Job.getInstance(conf, WordCountApp.class.getSimpleName());
        job.setJarByClass(WordCountApp.class);
        job.setMapperClass(TokenizerMapper.class);
        job.setCombinerClass(IntSumReducer.class);
        job.setReducerClass(IntSumReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));

        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```

7.3、Hive测试

7.3.1、切换到hive用户

```
su - hive
cd /usr/local/cdh5.14.4/
kinit -k -t /etc/security/keytabs/hive.keytab hive/bd001@BD004.COM
java -cp security.jar com.itcast.security.hadoop.HS2Tools
```

7.3.2、测试代码

```java
package com.itcast.security.hive;

import org.apache.hadoop.security.UserGroupInformation;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import org.apache.hadoop.conf.Configuration;

/**
 *
 * @author mengyao
 *
 */
public class HS2Tools {

    private static String driverName = "org.apache.hive.jdbc.HiveDriver";
    private static String url = "jdbc:hive2://bd001:10000/default;principal=hive/bd001@BD001.COM";
    private static ResultSet res;

    public static Connection getConnection()  {
        Configuration conf = new Configuration();
        try {
            UserGroupInformation.setConfiguration(conf);
            UserGroupInformation.loginUserFromKeytab("hive/bd001@BD004.COM", "/etc/security/keytabs/hive.keytab");
        } catch (IOException e) {
            e.printStackTrace();
        }
        try {
            Class.forName(driverName);
            return DriverManager.getConnection(url);
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return null;
    }

    /**
     *
     * @param statement
     * @return
     */
    public void showTables(Statement statement) {
        try {
            ResultSet res = statement.executeQuery("SHOW TABLES");
            while (res.next()) {
                System.out.println(res.getString(1));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    /**
     *
     * @param statement
     * @param tableName
     * @return
     */
    public void descTable(Statement statement, String tableName) {
        try {
            res = statement.executeQuery("DESCRIBE " + tableName);
            while (res.next()) {
                System.out.println(res.getString(1) + "\t" + res.getString(2));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    /**
     *
     * @param statement
     * @param tableName
     * @return
```

```
 */
public void dropTable(Statement statement, String tableName) {
    try {
        statement.execute("DROP TABLE IF EXISTS " + tableName);
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

/**
 *
 * @param statement
 * @return
 */
public void queryData(Statement statement, String tableName) {
    try {
        res = statement.executeQuery("SELECT * FROM " + tableName + " LIMIT 20");
        while (res.next()) {
            System.out.println(res.getString(1) + "," + res.getString(2) + "," + res.getString(3));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

/**
 *
 * @param statement
 * @return
 */
public void createTable(Statement statement, String tableName) {
    try {
        statement.execute("CREATE TABLE test_1m_test2 AS SELECT * FROM test_1m_test");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

public static void main(String[] args) throws Exception {
    System.setProperty("java.security.krb5.conf", "/etc/krb5.conf");
    args = new String[] {"test_100m"};
    if (args.length < 2) {
        System.out.println("Usage: input params require is 2!");
        System.exit(1);
    }
    HS2Tools tools = new HS2Tools();
    Connection connection = getConnection();
    Statement stmt = connection.createStatement();
    System.out.println("==== 显示表 ====");
    tools.showTables(stmt);
    System.out.println("==== 建表详情 ====");
    tools.descTable(stmt, args[1]);
    System.out.println("==== 删表 ====");
    tools.dropTable(stmt, args[1]);
    System.out.println("==== 显示表 ====");
```

8、问题

1、javax.security.sasl.SaslException: GSS initiate failed [Caused by GSSException: No valid credentials provided (Mechanism level: Failed to find any Kerberos tgt)]; Host Details : local host is: "bd001/192.168.10.101"; destination host is: "bd001":9000;

访问特定组件时，需在组件所在节点初始化principal凭证，如：访问NameNode时，需切换到NameNode对应的Uinx用户（hdfs），再初始化hdfs用户的NameNode组件对应的principal凭证。

```
su - hdfs
kinit -k -t /etc/security/keytabs/nn.service.keytab nn/bd001@BD004.COM
hdfs dfs -chmod -R 777 /tmp
```