How To Access Impala from Java via JDBC

Historically, the Hive JDBC driver was used for Impala:

<http://www.cloudera.com/documentation/archive/impala/1-x/1-0-1/Installing-and-Using-Impala/ciiu_impala_jdbc.html>

Nowadays, an Impala Connector is provided. See this page for downloading the connector:  
<http://www.cloudera.com/downloads/connectors/impala/odbc/2-5-22.html>

An example project (including dependency management is available on Github:  
<https://github.com/onefoursix/Cloudera-Impala-JDBC-Example>

All available artifacts provided for CDH5.8 are listed in this webpage:

<https://www.cloudera.com/documentation/enterprise/release-notes/topics/cdh_vd_cdh5_maven_repo_58x.html#concept_s1z_m5f_x5>

ODBC driver and JDBC driver for Impala are not listed here.

Downloads are available here:

<http://www.cloudera.com/downloads/connectors/impala/odbc/2-5-34.html>

<http://www.cloudera.com/downloads/connectors/impala/jdbc/2-5-32.html>

After extraction the example archive we have to install the JDBC driver using this command:

mvn install:install-file -Dfile=lib/cloudera-impala-jdbc-4.1\_2.5.32/ImpalaJDBC41.jar -DgroupId=impala-jdbc -DartifactId=jdbc-driver -Dversion=4.1.2 -Dpackaging=jar -DgeneratePom=true

Execution in the project base folder leads to this result:

My-Great-MBP:access-impala kamir$ mvn install:install-file -Dfile=lib/cloudera-impala-jdbc-4.1\_2.5.32/ImpalaJDBC41.jar -DgroupId=impala-jdbc -DartifactId=jdbc-driver -Dversion=4.1.2 -Dpackaging=jar -DgeneratePom=true

[INFO] Scanning for projects...

[INFO]

[INFO] ------------------------------------------------------------------------

[INFO] Building access-impala 0.0.1

[INFO] ------------------------------------------------------------------------

[INFO]

[INFO] --- maven-install-plugin:2.4:install-file (default-cli) @ access-impala ---

[INFO] Installing /GITHUB.cloudera.internal/access-impala/lib/cloudera-impala-jdbc-4.1\_2.5.32/ImpalaJDBC41.jar to /Users/kamir/.m2/repository/impala-jdbc/jdbc-driver/4.1.2/jdbc-driver-4.1.2.jar

[INFO] Installing /var/folders/3k/nyrd1dwj4d953l71ys0hnl680000gn/T/mvninstall4274411957146979091.pom to /Users/kamir/.m2/repository/impala-jdbc/jdbc-driver/4.1.2/jdbc-driver-4.1.2.pom

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

[INFO] Total time: 0.842 s

[INFO] Finished at: 2016-09-14T13:03:40+02:00

[INFO] Final Memory: 9M/245M

[INFO] ------------------------------------------------------------------------

The example project access-impala shows a Java program, which queries the example tables installed by HUE. In particular the tables sample\_07 and sample\_08 are used.

mvn install:install-file -Dfile=lib/cloudera-impala-jdbc-4.1\_2.5.32/TCLIServiceClient.jar -DgroupId=impala-jdbc -DartifactId=jdbc-driver-tcli-service-client -Dversion=4.1.2 -Dpackaging=jar -DgeneratePom=true

mvn install:install-file -Dfile=lib/cloudera-impala-jdbc-4.1\_2.5.32/ql.jar -DgroupId=impala-jdbc -DartifactId=jdbc-driver-ql -Dversion=4.1.2 -Dpackaging=jar -DgeneratePom=true

mvn install:install-file -Dfile=lib/cloudera-impala-jdbc-4.1\_2.5.32/hive\_metastore.jar -DgroupId=impala-jdbc -DartifactId=hive\_metastore -Dversion=4.1.2 -Dpackaging=jar -DgeneratePom=true

mvn install:install-file -Dfile=lib/cloudera-impala-jdbc-4.1\_2.5.32/hive\_service.jar -DgroupId=impala-jdbc -DartifactId=hive\_service -Dversion=4.1.2 -Dpackaging=jar -DgeneratePom=true

Finally, five libraries from Impala connector were installed in a local maven repository. All others which are available in a public repository are used via a dependency entry in the pom.xml file.



The class ClouderaImpalaJDBCExample uses no authentication and no encryption. In an environment with Kerberos authentication this will not work but with a standard Cloudera QSVM this should work fine.

The class SimpleApp uses additional security features. The following section is taken from the Impala connector documentation:



# Cloudera Cluster from Nihed

# **impala-shell --ssl -i ag2r-nihed-2.vpc.cloudera.com**

Starting Impala Shell without Kerberos authentication

SSL is enabled. Impala server certificates will NOT be verified (set --ca\_cert to change)

Error connecting: TTransportException, TSocket read 0 bytes

Kerberos ticket found in the credentials cache, retrying the connection with a secure transport.

Connected to ag2r-nihed-2.vpc.cloudera.com:21000

Server version: impalad version 2.6.0-cdh5.8.0 RELEASE (build 8d8652f69461f0dd8d5f474573fb5de7ceb0ee6b)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Impala shell. Copyright (c) 2015 Cloudera, Inc. All rights reserved.

(Impala Shell v2.6.0-cdh5.8.0 (8d8652f) built on Tue Jul 12 15:43:17 PDT 2016)

The SET command shows the current value of all shell and query options.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[ag2r-nihed-2.vpc.cloudera.com:21000] >

In order to be able to login via kerberos, we need a TGT. If such a key granting ticket is available can be checked by the klist command:

# **klist**

Ticket cache: FILE:/tmp/krb5cc\_0

Default principal: nihed@CLOUDERA.COM

Valid starting Expires Service principal

14.09.2016 05:42:40 15.09.2016 05:42:40 krbtgt/CLOUDERA.COM@CLOUDERA.COM

renew until 21.09.2016 05:42:40

14.09.2016 08:10:11 15.09.2016 05:42:40 impala/ag2r-nihed-2.vpc.cloudera.com@

renew until 19.09.2016 08:10:11

14.09.2016 08:10:11 15.09.2016 05:42:40 impala/ag2r-nihed-2.vpc.cloudera.com@CLOUDERA.COM

renew until 19.09.2016 08:10:11

For the next step we need a keytab for a user principal which is allowed to interact with the cluster. For simplicity we used the impala service principal. We now export a keytab.

Login to a cluster server:

# **ssh root@ag2r-nihed-1.vpc.cloudera.com**

# kinit nihed (pass: redhat)

# kadmin.local

xst -norandkey -k nihed.keytab nihed

# exit

On the developer machine change directory to the access-impala project and there into sec. Now, using scp we can grab the exported keytab file. An alternative would be to copy the impala service keytab.

# **scp root@ag2r-nihed-1.vpc.cloudera.com:nihed.keytab .**

We also need the file /etc/krb5.conf from one cluster node.

With all this details we can go back to the Java code. Please change all yellwo marked details in the 3 configuration files.

config.props

connection.url = jdbc:impala://ag2r-nihed-2.vpc.cloudera.com:21050

jdbc.driver.class.name = com.cloudera.impala.jdbc41.Driver

jdbc.query = Select count(\*) from sample\_07

keytab.file = ./sec/impala.keytab

jaas-impala.conf

Impala-client {

com.sun.security.auth.module.Krb5LoginModule required

useKeyTab=true

keyTab="/GITHUB.cloudera.internal/access-impala/sec/impala.keytab"

storeKey=true

useTicketCache=true

principal="impala/ag2r-nihed-4.vpc.cloudera.com@CLOUDERA.COM"

doNotPrompt=true

debug=true;

};

Client {

com.sun.security.auth.module.Krb5LoginModule required

useKeyTab=true

keyTab="/GITHUB.cloudera.internal/access-impala/sec/impala.keytab"

storeKey=true

useTicketCache=true

principal="impala/ag2r-nihed-4.vpc.cloudera.com@CLOUDERA.COM"

doNotPrompt=true

debug=true;

};

krb5.conf

[libdefaults]

default\_realm = CLOUDERA.COM

dns\_lookup\_kdc = false

dns\_lookup\_realm = false

ticket\_lifetime = 86400

renew\_lifetime = 604800

forwardable = true

default\_tgs\_enctypes = rc4-hmac

default\_tkt\_enctypes = rc4-hmac

permitted\_enctypes = rc4-hmac

udp\_preference\_limit = 1

kdc\_timeout = 3000

[realms]

CLOUDERA.COM = {

kdc = ag2r-nihed-1.vpc.cloudera.com

admin\_server = ag2r-nihed-1.vpc.cloudera.com

}

Using maven we compile and package the program:

# **mvn clean compile package**

The result is a so called *„fat-jar“* with name access-impala-0.0.1-job.jar in the target folder. We can start the program with this command:

$ **java -jar target/access-impala-0.0.1-job.jar**

JAAS-Config: true

KRB5-Config: true

Debug is true storeKey true useTicketCache true useKeyTab true doNotPrompt true ticketCache is null isInitiator true KeyTab is /GITHUB.cloudera.internal/access-impala/sec/impala.keytab refreshKrb5Config is false principal is impala/ag2r-nihed-4.vpc.cloudera.com@CLOUDERA.COM tryFirstPass is false useFirstPass is false storePass is false clearPass is false

Acquire TGT from Cache

Principal is impala/ag2r-nihed-4.vpc.cloudera.com@CLOUDERA.COM

null credentials from Ticket Cache

principal is impala/ag2r-nihed-4.vpc.cloudera.com@CLOUDERA.COM

Will use keytab

Commit Succeeded

Authentication succeeded!

=============================================

Cloudera Impala JDBC Example

Using Connection URL: jdbc:impala://ag2r-nihed-2.vpc.cloudera.com:21050

Running Query: Select count(\*) from sample\_07

jdbc:impala://ag2r-nihed-2.vpc.cloudera.com:21050/default;ssl=1;AuthMech=1;AllowSelfSignedCerts=1;KrbRealm=CLOUDERA.COM;KrbHostFQDN=ag2r-nihed-2.vpc.cloudera.com;KrbServiceName=impala;SSLKeyStore=./sec/ag2r-nihed-1.keystore;SSLKeyStorePwd=cloudera;

Debug is true storeKey true useTicketCache true useKeyTab true doNotPrompt true ticketCache is null isInitiator true KeyTab is /GITHUB.cloudera.internal/access-impala/sec/impala.keytab refreshKrb5Config is false principal is impala/ag2r-nihed-4.vpc.cloudera.com@CLOUDERA.COM tryFirstPass is false useFirstPass is false storePass is false clearPass is false

Acquire TGT from Cache

Principal is impala/ag2r-nihed-4.vpc.cloudera.com@CLOUDERA.COM

null credentials from Ticket Cache

principal is impala/ag2r-nihed-4.vpc.cloudera.com@CLOUDERA.COM

Will use keytab

Commit Succeeded

log4j:WARN No appenders could be found for logger (org.apache.thrift.transport.TSaslTransport).

log4j:WARN Please initialize the log4j system properly.

== Begin Query Results ======================

823

== End Query Results =======================

In this way we configure the JDBC driver to connect to a particular Impala deamon. Usually, a load balancer is used to decouple the clients from the cluster and to spread the workload among multiple Impala deamons automatically. The following link provides details for setting up the HAProxy with Impala:

<http://www.cloudera.com/documentation/enterprise/5-4-x/topics/impala_proxy.html#proxy>