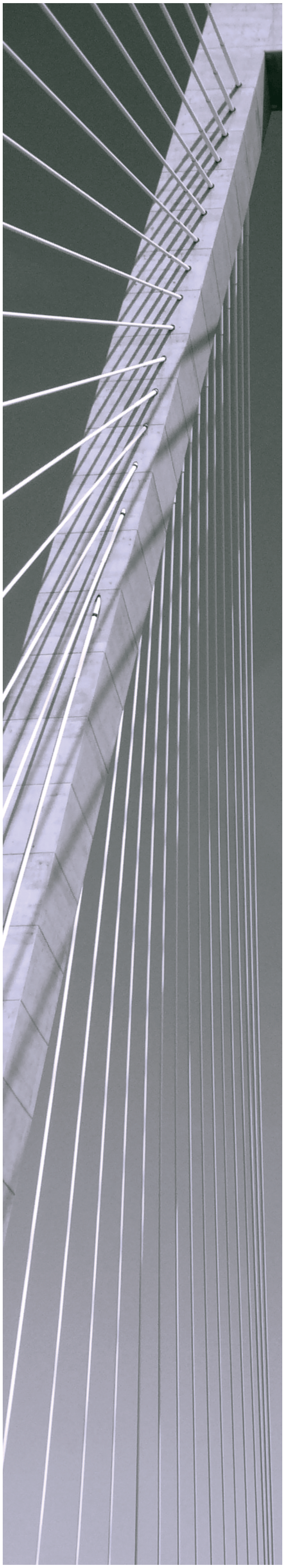
Simba Apache Spark JDBC Driver with SQL Connector

Installation and Configuration Guide

Simba Technologies Inc.

Version 1.0.1

October 29, 2015



Trademarks

Simba, the Simba logo, SimbaEngine, SimbaEngine C/S, SimbaExpress and SimbaLib are registered trademarks of Simba Technologies Inc. All other trademarks and/or servicemarks are the property of their respective owners.

Contact Us

Simba Technologies Inc.
938 West 8th Avenue
Vancouver, BC Canada
V5Z 1E5

Tel: +1 (604) 633-0008

Fax: +1 (604) 633-0004

www.simba.com

About This Guide

Purpose

The *Simba Apache Spark JDBC Driver with SQL Connector Installation and Configuration Guide* explains how to install and configure the Simba Apache Spark JDBC Driver with SQL Connector on all supported platforms. The guide also provides details related to features of the driver.

Audience

The guide is intended for end users of the Simba Apache Spark JDBC Driver with SQL Connector.

Knowledge Prerequisites

To use the Simba Apache Spark JDBC Driver with SQL Connector, the following knowledge is helpful:

- Familiarity with the platform on which you are using the Simba Apache Spark JDBC Driver with SQL Connector
- Ability to use the data source to which the Simba Apache Spark JDBC Driver with SQL Connector is connecting
- An understanding of the role of JDBC technologies in connecting to a data source
- Experience creating and configuring JDBC connections
- Exposure to SQL

Document Conventions

Italics are used when referring to book and document titles.

Bold is used in procedures for graphical user interface elements that a user clicks and text that a user types.

`Monospace font` indicates commands, source code or contents of text files.

Underline is not used.

✎ The pencil icon indicates a short note appended to a paragraph.

★ The star icon indicates an important comment related to the preceding paragraph.

👍 The thumbs up icon indicates a practical tip or suggestion.

Table of Contents

Introduction

The Simba Apache Spark JDBC Driver with SQL Connector is used for direct SQL and HiveQL access to Apache Hadoop / Spark, enabling Business Intelligence (BI), analytics, and reporting on Hadoop / Spark-based data. The driver efficiently transforms an application's SQL query into the equivalent form in HiveQL, which is a subset of SQL-92. If an application is Spark-aware, then the driver is configurable to pass the query through to the database for processing. The driver interrogates Spark to obtain schema information to present to a SQL-based application. Queries, including joins, are translated from SQL to HiveQL. For more information about the differences between HiveQL and SQL, see Features on page 20.

The Simba Apache Spark JDBC Driver with SQL Connector complies with the JDBC 4.0 and 4.1 data standards. JDBC is one of the most established and widely supported APIs for connecting to and working with databases. At the heart of the technology is the JDBC driver, which connects an application to the database. For more information about JDBC, see http://www.simba.com/resources/data-access-standards-library.

This guide is suitable for users who want to access data residing within Spark from their desktop environment. Application developers may also find the information helpful. Refer to your application for details on connecting via JDBC.

System Requirements

Each computer where you use the Simba Apache Spark JDBC Driver with SQL Connector must have Java Runtime Environment (JRE) installed. The version of JRE that must be installed depends on the version of the JDBC API you are using with the driver. Table 1 lists the required version of JRE for each version of the JDBC API.

Table 1. Driver System Requirements

JDBC API Version	JRE Version
4.0	6.0 or later
4.1	7.0 or later

The driver is supports Apache Spark versions 1.1 through 1.5.

> ★ The driver only supports connections to Shark Server 2 and Spark Thrift Server instances. It does not support connections to Shark Server instances.

Simba Apache Spark JDBC Driver with SQL Connector Files

The Simba Apache Spark JDBC Driver with SQL Connector is delivered in the following ZIP archives, where *version* is the version number of the driver:

- Simba_SparkJDBC4_*version*.zip
- Simba_SparkJDBC41_*version*.zip

Each archive contains the driver supporting the JDBC API version indicated in the archive name.

The archives contain the following file and folder structure, where *LibVersion* is the version number of the library and *APIVersion* is the JDBC API version that the driver supports:

- SparkJDBC*APIVersion*
  - hive_metastore.jar
  - hive_service.jar
  - SparkJDBC*APIVersion*.jar
  - libfb303-*LibVersion*.jar
  - libthrift-*LibVersion*.jar
  - log4j-*LibVersion*.jar
  - ql.jar
  - Simba JDBC Driver for Spark Install Guide.pdf
  - slf4j-api-*LibVersion*.jar
  - slf4j-log4j12-*LibVersion*.jar
  - TCLIServiceClient.jar

Simba License File

Before you can use the Simba Apache Spark JDBC Driver with SQL Connector, you must place the SimbaApacheSparkJDBCDriver.lic file in the same directory as the SparkJDBC4.jar or SparkJDBC41.jar file.

Using the Simba Apache Spark JDBC Driver with SQL Connector

To access a Spark data warehouse using the Simba Apache Spark JDBC Driver with SQL Connector, you need to configure the following:

- Class path
- Driver or DataSource class
- Connection URL

For sample code that demonstrates how to use the driver, see Java Sample Code on page 12.

> ★ The Simba Apache Spark JDBC Driver with SQL Connector provides read-only access to Shark Server 2 and Spark Thrift Server instances. It does not support connections to Shark Server instances.

Setting the Class Path

To use the Simba Apache Spark JDBC Driver with SQL Connector, you must set the class path to include all the JAR files from the ZIP archive containing the driver that you are using.

The class path is the path that the Java Runtime Environment searches for classes and other resource files. For more information, see the topic *Setting the Class Path* in the Java SE Documentation at http://docs.oracle.com/javase/7/docs/technotes/tools/windows/classpath.html.

Initializing the Driver Class

Before connecting to the data store, you must initialize the appropriate class for the Spark server and your application.

The following is a list of the classes used to connect the Simba Apache Spark JDBC Driver with SQL Connector to Shark Server 2 or Spark Thrift Server instances. The Driver classes extend java.sql.Driver, and the DataSource classes extend javax.sql.DataSource and javax.sql.ConnectionPoolDataSource.

To support JDBC 4.0, classes with the following FQCNs are available:

- com.simba.spark.jdbc4.Driver
- com.simba.spark.jdbc4.DataSource

To support JDBC 4.1, classes with the following FQCNs are available:

- com.simba.spark.jdbc41.Driver
- com.simba.spark.jdbc41.DataSource

The following sample code shows how to use the DriverManager to establish a connection:

```
private static Connection connectViaDM() throws Exception
{
    Connection connection = null;
    Class.forName(DRIVER_CLASS);
    connection = DriverManager.getConnection(CONNECTION_
    URL);
    return connection;
}
```

The following sample code shows how to use the DataSource class to establish a connection:

```
private static Connection connectViaDS() throws Exception
{
    Connection connection = null;
    Class.forName(DRIVER_CLASS);
    DataSource ds = new com.simba.spark.jdbc4.DataSource();
    ds.setURL(CONNECTION_URL);
    connection = ds.getConnection();
    return connection;
}
```

Building the Connection URL

Use the connection URL to supply connection information to the data source that you are accessing. The following is the format of the connection URL for the Simba Apache Spark JDBC Driver with SQL Connector, where *Host* is the DNS or IP address of the Spark server:

`jdbc:spark://Host`

By default, the driver connects to port 10000, uses the schema named **default**, and authenticates the connection using the user name **spark**.

You can specify optional settings such as the number of the TCP port to connect to, the schema to use, or any of the connection properties supported by the driver. For a list of the properties available in the driver, see Driver Configuration Options on page 30.

The following is the format of a connection URL that specifies some optional settings:

```
jdbc:spark://Host:Port[/Schema];Property1=Value;
Property2=Value;...
```

For example, to connect to port 11000 on an Spark server installed on the local machine, use a schema named **default2**, and authenticate the connection using a user name and password, you would use the following connection URL:

```
jdbc:spark://localhost:11000[/default2];AuthMech=3;
UID=simba;PWD=simba
```

★ Be aware of the following:

- Properties are case-sensitive.
- Do not duplicate properties in the connection URL.

✎ Note the following:

- If you specify a schema in the connection URL, you can still issue queries on other schemas by explicitly specifying the schema in the query. To inspect your databases and determine the appropriate schema to use, type the **show databases** command at the Spark command prompt.
- If you specify a property that is not supported by the driver, then the driver attempts to apply the property as a Spark server-side property for the client session.

Java Sample Code

The following Java code provides an example demonstrating how to use the JDBC API to do the following:

- Register the Simba Apache Spark JDBC Driver with SQL Connector
- Establish a connection to a Spark database
- Query the database
- Parse a result set
- Handle exceptions
- Clean up to avoid memory leakage

★ To use the Simba Apache Spark JDBC Driver with SQL Connector in an application, you must include all the JAR files from the ZIP archive in the class path for your Java project.

```
// java.sql packages are required
import java.sql.*;
class SimbaJDBCSparkExample {

    // Define a string as the fully qualified class name
    // (FQCN) of the desired JDBC driver
    private static final String JDBC_DRIVER =
    "com.simba.spark.jdbc4.Driver";
    // Define a string as the connection URL
    private static final String CONNECTION_URL =
    "jdbc:spark://192.168.1.1:10000";

    public static void main(String[] args) {

        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

        // Define a plain query
        String query = "SELECT first_name, last_name, emp_id
        FROM default.emp";
        // Define a parametrized query
```

```
String prepQuery = "SELECT first_name, last_name,
emp_id FROM default.emp where store_id = ?";

try {

    // Register the driver using the class name
    Class.forName(JDBC_DRIVER);

    // Establish a connection using the connection
    // URL
    con = DriverManager.getConnection(CONNECTION_
    URL);

    // Create a Statement object for sending SQL
    // statements to the database
    stmt = con.createStatement();

    // Execute the SQL statement
    rs = stmt.executeQuery(query);

    // Display a header line for output appearing in
    // the Console View
    System.out.printf("%20s%20s%20s\r\n", "FIRST
    NAME", "LAST NAME" , "EMPLOYEE ID");

    // Step through each row in the result set
    // returned from the database
    while(rs.next()) {
        // Retrieve values from the row where the
        // cursor is currently positioned using
        // column names
        String FirstName = rs.getString("first_
        name");
        String LastName = rs.getString("last_name");
        String EmployeeID = rs.getString("emp_id");
```

```
            // Display values in columns 20 characters
            // wide in the Console View using the
            // Formatter
            System.out.printf("%20s%20s%20s\r\n",
            FirstName, LastName, EmployeeID);
        }
        // Create a prepared statement
        PreparedStatement prep = con.prepareStatement
        (prepQuery);

        // Bind the query parameter with a value
        prep.setInt(1, 204);
        // Execute the query
        prep.execute();
        rs = prep.getResultSet();
        // Step through each row in the result set
        // returned from the database
        while(rs.next()) {
            // Retrieve values from the row where the
            // cursor is currently positioned using
            // column names
            String FirstName = rs.getString("first_
            name");
            String LastName = rs.getString("last_name");
            String EmployeeID = rs.getString("emp_id");

            // Display values in columns 20 characters
            // wide in the Console View using the
            // Formatter
            System.out.printf("%20s%20s%20s\r\n",
            FirstName, LastName, EmployeeID);
        }

    } catch (SQLException se) {
        // Handle errors encountered during interaction
        // with the data source
    } catch (Exception e) {
        // Handle other errors
    } finally {
        // Perform clean up
        try {
            if (rs != null) {
                rs.close();
            }
        } catch (SQLException se1) {
            // Log this
        }

        try {
            if (stmt != null) {
                stmt.close();
            }
        } catch (SQLException se2) {
            // Log this
        }
        try {
            if (prep != null) {
                prep.close();
            }
        } catch (SQLException se3) {
            // Log this
        }

        try {
            if (con != null) {
                con.close();
            }
        } catch (SQLException se4) {
            // Log this
        } // End try
    } // End try
```

```
    } // End main
} // End SimbaJDBCSparkExample
```

Configuring Authentication

The Simba Apache Spark JDBC Driver with SQL Connector supports the following authentication mechanisms:

- No Authentication
- Kerberos
- User Name
- User Name and Password

You configure the authentication mechanism that the driver uses to connect to Spark by specifying the relevant properties in the connection URL.

For information about selecting an appropriate authentication mechanism when using the Simba Apache Spark JDBC Driver with SQL Connector, see Authentication Options on page 23.

For information about the properties you can use in the connection URL, see Driver Configuration Options on page 30.

> In addition to authentication, you can configure the driver to connect over SSL. For more information, see Configuring SSL on page 19.

Using No Authentication

To configure a connection without authentication:

➢ Set the AuthMech property to 0.

For example:

```
jdbc:spark://localhost:10000;AuthMech=0
```

Using Kerberos

Kerberos must be installed and configured before you can use this authentication mechanism. For information about configuring and operating Kerberos on Windows, see Configuring Kerberos Authentication for Windows on page 25. For other operating systems, refer to the MIT Kerberos documentation.

To configure Kerberos authentication:

1. Set the AuthMech property to 1.
2. To use the default realm defined in your Kerberos setup, do not set the KrbRealm property.

   If your Kerberos setup does not define a default realm or if the realm of your Spark server is not the default, then set the KrbRealm property to the realm of the Spark server.

3. Set the KrbHostFQDN property to the fully qualified domain name of the Spark server host.
4. Set the KrbServiceName property to the service name of the Spark server.

For example:

```
jdbc:spark://localhost:
10000;AuthMech=1;KrbRealm=EXAMPLE.COM;
KrbHostFQDN=spark.example.com;KrbServiceName=spark
```

Using User Name

This authentication mechanism requires a user name but does not require a password. The user name labels the session, facilitating database tracking.

To configure User Name authentication:

1. Set the AuthMech property to 2.
2. Set the UID property to an appropriate user name for accessing the Spark server.

For example:

```
jdbc:spark://localhost:10000;AuthMech=2;UID=spark
```

Using User Name and Password

This authentication mechanism requires a user name and a password.

To configure User Name and Password authentication:

1. Set the AuthMech property to 3.
2. Set the UID property to an appropriate user name for accessing the Spark server.
3. Set the PWD property to the password corresponding to the user name you provided in step 2.

For example:

```
jdbc:spark://localhost:10000;AuthMech=3;UID=spark;PWD=*****
```

Configuring SSL

If you are connecting to a Spark server that has Secure Sockets Layer (SSL) enabled, then you can configure the driver to connect to an SSL-enabled socket.

SSL connections require a KeyStore and a TrustStore. You can create a TrustStore and configure the driver to use it, or allow the driver to use one of the default TrustStores. If you do not configure the driver to use a specific TrustStore, then the driver uses the Java TrustStore jssecacerts. If jssecacerts is not available, then the driver uses cacerts instead.

To configure SSL:

1. Create a KeyStore and configure the driver to use it:
    a) Create a KeyStore containing your signed, trusted SSL certificate.
    b) Set the SSLKeyStore property to the full path of the KeyStore, including the file name.
    c) Set the SSLKeyStorePwd property to the password for the KeyStore.
2. Optionally, create a TrustStore and configure the driver to use it:
    a) Create a TrustStore containing your signed, trusted SSL certificate.
    b) Set the SSLTrustStore property to the full path of the TrustStore, including the file name.
    c) Set the SSLTrustStorePwd property to the password for the TrustStore.
3. Set the SSL property to 1.
4. Optionally, to allow the SSL certificate used by the server to be self-signed, set the AllowSelfSignedCerts property to 1
5. Optionally, to allow the common name of a CA-issued certificate to not match the host name of the Spark server, set the CAIssuedCertNamesMismatch property to 1

    > For self-signed certificates, the driver always allows the common name of the certificate to not match the host name.

For example:

```
jdbc:spark://localhost:10000;AuthMech=3;SSL=1;
SSLKeyStore=C:\\Users\\bsmith\\Desktop\\keystore.jks;
SSLKeyStorePwd=*****;UID=spark;PWD=*****
```

> For more information about the connection properties used in SSL connections, see Driver Configuration Options on page 30

Features

More information is provided on the following features of the Simba Apache Spark JDBC Driver with SQL Connector:

- SQL Query versus HiveQL Query on page 20
- Data Types on page 20
- Catalog and Schema Support on page 21

SQL Query versus HiveQL Query

The native query language supported by Spark is HiveQL. HiveQL is a subset of SQL-92. However, the syntax is different enough that most applications do not work with native HiveQL.

Data Types

The Simba Apache Spark JDBC Driver with SQL Connector supports many common data formats, converting between Spark, SQL, and Java data types.

Table 2 lists the supported data type mappings.

Table 2. Supported Data Types

Spark Type	SQL Type	Java Type
BIGINT	BIGINT	java.math.BigInteger
BINARY	VARBINARY	byte[]
BOOLEAN	BOOLEAN	Boolean
DATE	DATE	java.sql.Date
DECIMAL	DECIMAL	java.math.BigDecimal
DOUBLE	DOUBLE	Double
FLOAT	REAL	Float
INT	INTEGER	Long
SMALLINT	SMALLINT	Integer

Spark Type	SQL Type	Java Type
TIMESTAMP	TIMESTAMP	java.sql.Timestamp
TINYINT	TINYINT	Short
VARCHAR	VARCHAR	String

Catalog and Schema Support

The Simba Apache Spark JDBC Driver with SQL Connector supports both catalogs and schemas in order to make it easy for the driver to work with various JDBC applications. Since Spark only organizes tables into schemas/databases, the driver provides a synthetic catalog called “SPARK” under which all of the schemas/databases are organized. The driver also maps the JDBC schema to the Spark schema/database.

> Setting the CatalogSchemaSwitch connection property to 1 will cause Spark catalogs to be treated as schemas in the driver as a restriction for filtering.

Contact Us

If you have difficulty using the driver, please contact our Technical Support staff. We welcome your questions, comments, and feature requests.

Technical Support is available Monday to Friday from 8 a.m. to 6 p.m. Pacific Time.

> ★ To help us assist you, prior to contacting Technical Support please prepare a detailed summary of the client and server environment including operating system version, patch level, and configuration.

You can contact Technical Support via:

- **E-mail**—support@simba.com
- **Web site**—www.simba.com
- **Telephone**—(604) 633-0008 Extension 3
- **Fax**—(604) 633-0004

You can also follow us on Twitter @SimbaTech

Appendix A Authentication Options

Shark Server 2 and Spark Thrift Server support the following authentication mechanisms:

- No Authentication
- Kerberos
- User Name
- User Name and Password

Most default configurations of Shark Server 2 or Spark Thrift Server require User Name authentication. If you are unable to connect to your Spark server using User Name authentication, then verify the authentication mechanism configured for your Spark server by examining the hive-site.xml file. Examine the following properties to determine which authentication mechanism your server is set to use:

- **hive.server2.authentication**—This property sets the authentication mode for Spark Server 2. The following values are available:
    - **NOSASL** disables the Simple Authentication and Security Layer (SASL).
    - **KERBEROS** enables Kerberos authentication.
    - **NONE** enables plain SASL transport. NONE is the default value.
    - **PLANSASL** enables user name and password authentication using a cleartext password mechanism.
- **hive.server2.enable.doAs**—If this property is set to the default value of **TRUE**, then Spark processes queries as the user who submitted the query. If this property is set to **FALSE**, then queries are run as the user that runs the hiveserver2 process.

Table 3 lists authentication mechanisms to configure for the driver based on the settings in the hive-site.xml file.

Table 3. Spark Authentication Mechanism Configurations

hive.server2.authentication	hive.server2.enable.doAs	Driver Authentication Mechanism
NOSASL	FALSE	No Authentication
KERBEROS	TRUE or FALSE	Kerberos
NONE	TRUE or FALSE	User Name
LDAP	TRUE or FALSE	User Name and Password

> ✎ It is an error to set hive.server2.authentication to NOSASL and hive.server2.enable.doAs to true. This configuration will not prevent the service from starting up, but results in an unusable service.

For more information about authentication mechanisms, refer to the documentation for your Hadoop / Spark distribution. See also *Running Hadoop in Secure Mode* at http://hadoop.apache.org/docs/r0.23.7/hadoop-project-dist/hadoop-common/ClusterSetup.html#Running_Hadoop_in_Secure_Mode.

Using No Authentication

When hive.server2.authentication is set to NOSASL, you must configure your connection to use No Authentication.

Using Kerberos

When connecting to a Spark server of type Shark Server 2 or Spark Thrift Server and hive.server2.authentication is set to KERBEROS, you must configure your connection to use Kerberos authentication.

Using User Name

When connecting to a Spark server of type Shark Server 2 or Spark Thrift Server and hive.server2.authentication is set to NONE, you must configure your connection to use User Name authentication. Validation of the credentials that you include depends on hive.server2.enable.doAs:

- If hive.server2.enable.doAs is set to TRUE, then the user name in the driver configuration must be an existing operating system user on the host that is running Shark Server 2 or Spark Thrift Server.
- If hive.server2.enable.doAs is set to FALSE, then the user name in the driver configuration is ignored.

If no user name is specified in the driver configuration, then the driver defaults to using "spark" as the user name.

Using User Name and Password

When connecting to a Spark server of type Shark Server 2 or Spark Thrift Server and the server is configured to use the SASL-PLAIN authentication mechanism with a user name and a password, you must configure your connection to use User Name and Password authentication.

Appendix B Configuring Kerberos Authentication for Windows

You can configure your Kerberos setup so that you use the MIT Kerberos Ticket Manager to get the Ticket Granting Ticket (TGT), or configure the setup so that you can use the driver to get the ticket directly from the Key Distribution Center (KDC). Also, if a client application obtains a Subject with a TGT, it is possible to use that Subject to authenticate the connection.

Downloading and Installing MIT Kerberos for Windows

To download and install MIT Kerberos for Windows 4.0.1:

1. Download the appropriate Kerberos installer:
    - For a 64-bit computer, use the following download link from the MIT Kerberos website: http://web.mit.edu/kerberos/dist/kfw/4.0/kfw-4.0.1-amd64.msi.
    - For a 32-bit computer, use the following download link from the MIT Kerberos website: http://web.mit.edu/kerberos/dist/kfw/4.0/kfw-4.0.1-i386.msi.

    The 64-bit installer includes both 32-bit and 64-bit libraries. The 32-bit installer includes 32-bit libraries only.
2. To run the installer, double-click the .msi file that you downloaded in step 1.
3. Follow the instructions in the installer to complete the installation process.
4. When the installation completes, click **Finish**.

Using the MIT Kerberos Ticket Manager to Get Tickets

Setting the KRB5CCNAME Environment Variable

You must set the KRB5CCNAME environment variable to your credential cache file.

To set the KRB5CCNAME environment variable:

1. Click the **Start** button, then right-click **Computer**, and then click **Properties**.
2. Click **Advanced System Settings**.
3. In the System Properties dialog box, on the **Advanced** tab, click **Environment Variables**.
4. In the Environment Variables dialog box, under the System variables list, click **New**.
5. In the **New System Variable** dialog box, in the Variable name field, type **KRB5CCNAME**.
6. In the **Variable Value** field, type the path for your credential cache file. For example, type **C:\KerberosTickets.txt**.
7. Click **OK** to save the new variable.

8. Make sure that the variable appears in the System Variables list.
9. Click **OK** to close the Environment Variables dialog box, and then click **OK** to close the System Properties dialog box.
10. Restart your computer.

Getting a Kerberos Ticket

To get a Kerberos ticket:

1. Click the **Start** button, then click **All Programs**, and then click the **Kerberos for Windows (64-bit)** or **Kerberos for Windows (32-bit)** program group.
2. Click **MIT Kerberos Ticket Manager**.
3. In the MIT Kerberos Ticket Manager, click **Get Ticket**.
4. In the Get Ticket dialog box, type your principal name and password, and then click **OK**.

If the authentication succeeds, then your ticket information appears in the MIT Kerberos Ticket Manager.

Authenticating to the Spark Server

To authenticate to the Spark server:

- Use a connection string that has the following properties defined:
    - AuthMech
    - KrbHostFQDN
    - KrbRealm
    - KrbServiceName

For detailed information about these properties, see Driver Configuration Options on page 30.

Using the Driver to Get Tickets

Deleting the KRB5CCNAME Environment Variable

To enable the driver to get Ticket Granting Tickets (TGTs) directly, you must ensure that the KRB5CCNAME environment variable has not been set.

To delete the KRB5CCNAME environment variable:

1. Click the **Start** button, then right-click **Computer**, and then click **Properties**.
2. Click **Advanced System Settings**.
3. In the System Properties dialog box, click the **Advanced** tab and then click **Environment Variables**.

4. In the Environment Variables dialog box, check if the KRB5CCNAME variable appears in the System variables list. If the variable appears in the list, then select the variable and click **Delete**.
5. Click **OK** to close the Environment Variables dialog box, and then click **OK** to close the System Properties dialog box.

Setting Up the Kerberos Configuration File

To set up the Kerberos configuration file:

1. Create a standard krb5.ini file and place it in the C:\Windows directory.
2. Ensure that the KDC and Admin server specified in the krb5.ini file can be resolved from your terminal. If necessary, modify "C:\Windows\System32\drivers\etc\hosts".

Setting Up the JAAS Login Configuration File

To set up the JAAS login configuration file:

1. Create a JAAS login configuration file that specifies a keytab file and "doNotPrompt=true"

   For example:

   ```
   Client {
       com.sun.security.auth.module.Krb5LoginModule
       required
           useKeyTab=true
           keyTab="PathToTheKeyTab"
           principal="simba@SIMBA"
           doNotPrompt=true;
   };
   ```

2. Set the java.security.auth.login.config environment variable to the location of the JAAS file.

   For example: C:\KerberosLoginConfig.ini

Authenticating to the Spark Server

To authenticate to the Spark server:

➢ Use a connection string that has the following properties defined:

- AuthMech
- KrbHostFQDN
- KrbRealm
- KrbServiceName

For detailed information about these properties, see Driver Configuration Options on page 30.

Using an Existing Subject to Authenticate the Connection

If the client application obtains a Subject with a TGT, then that Subject can be used to authenticate the connection to the server.

To use an existing Subject to authenticate the connection:

1. Create a PrivilegedAction for establishing the connection to the database.

   For example:

   ```
   // Contains logic to be executed as a privileged action
   public class AuthenticateDriverAction
   implements PrivilegedAction<Void>
   {
       // The connection, which is established as a
       // PrivilegedAction
       Connection con;

       // Define a string as the connection URL
       static String ConnectionURL =
       "jdbc:spark://192.168.1.1:10000";

       /**
       * Logic executed in this method will have access to
       the
       * Subject that is used to "doAs". The driver will
       get
       * the Subject and use it for establishing a
       connection
       * with the server.
       */
       @Override
       public Void run()
       {
           try
           {
               // Establish a connection using the
               connection URL
   ```

```
            con = DriverManager.getConnection
            (ConnectionURL);
        }
        catch (SQLException e)
        {
            // Handle errors that are encountered during
            // interaction with the data source
            e.printStackTrace();
        }
        catch (Exception e)
        {
            // Handle other errors
            e.printStackTrace();
        }
        return null;
    }
}
```

2. Run the PrivilegedAction using the existing Subject, and then use the connection.

   For example:

```
// Create the action
AuthenticateDriverAction authenticateAction = new
AuthenticateDriverAction();
// Establish the connection using the Subject for
// authentication.
Subject.doAs(loginConfig.getSubject(),
authenticateAction);
// Use the established connection.
authenticateAction.con;
```

Appendix C Driver Configuration Options

Appendix C lists and describes the properties that you can use to configure the behavior of the Simba Apache Spark JDBC Driver with SQL Connector.

You can set configuration properties using the connection URL. For more information, see Building the Connection URL on page 10.

AllowSelfSignedCerts

Default Value	Required
0	No

Description

When this property is set to 0, the SSL certificate used by the server cannot be self-signed.

When this property is set to 1, the SSL certificate used by the server can be self-signed.

> This property is applicable only when SSL connections are enabled.

AuthMech

Default Value	Required
0	No

Description

The authentication mechanism to use. Set the value to one of the following numbers:

- **0** for No Authentication
- **1** for Kerberos
- **2** for User Name
- **3** for User Name and Password

CAIssuedCertNamesMismatch

Default Value	Required
0	No

Description

When this property is set to 0, the name of the CA-issued SSL certificate must match the host name of the Spark server.

When this property is set to 1, the names of the certificate and the host name of the server are allowed to mismatch.

This property is applicable only when SSL connections are enabled.

CatalogSchemaSwitch

Default Value	Required
0	No

Description

When this property is set to 1, the driver treats Spark catalogs as schemas as a restriction for filtering.

When this property is set to 0, Spark catalogs are treated as catalogs, and Spark schemas are treated as schemas.

DecimalColumnScale

Default Value	Required
10	No

Description

The maximum number of digits to the right of the decimal point for numeric data types.

DefaultStringColumnLength

Default Value	Required
255	No

Description

The maximum data length for STRING columns. The range of DefaultStringColumnLength is 0 to 32,767.

By default, the columns metadata for Spark does not specify a maximum data length for STRING columns.

DelegationUID

Default Value	Required
None	No

Description

Use this option to delegate all operations against Spark to a user that is different than the authenticated user for the connection.

> This option is applicable only when connecting to a Shark Server 2 or Spark Thrift Server instance that supports this feature.

KrbHostFQDN

Default Value	Required
None	Yes, if AuthMech=1 (Kerberos)

Description

The fully qualified domain name of the Shark Server 2 or Spark Thrift Server host.

KrbRealm

Default Value	Required
Depends on Kerberos configuration.	No

Description

The realm of the Shark Server 2 or Spark Thrift Server host.

If your Kerberos configuration already defines the realm of the Shark Server 2 or Spark Thrift Server host as the default realm, then you do not need to configure this option.

KrbServiceName

Default Value	Required
None	Yes, if AuthMech=1 (Kerberos)

Description

The Kerberos service principal name of the Spark server.

PreparedMetaLimitZero

Default Value	Required
0	No

Description

When this property is set to 1, the PreparedStatement.getMetadata() call will request metadata from the server with “LIMIT 0”.

PWD

Default Value	Required
None	Yes, if AuthMech=3 (User Name and Password)

Description

The password corresponding to the user name that you provided using the property UID on page 36.

RowsFetchedPerBlock

Default Value	Required
10000	No

Description

The maximum number of rows that a query returns at a time.

Any positive 32-bit integer is a valid value, but testing has shown that performance gains are marginal beyond the default value of 10000 rows.

SocketTimeout

Default Value	Required
0	No

Description

The number of seconds after which Spark closes the connection with the client application if the connection is idle. The default value of 0 indicates that an idle connection is not closed.

SSL

Default Value	Required
0	No

Description

When this property is set to 1, the driver communicates with the Spark server through an SSL-enabled socket.

When this property is set to 0, the driver does not connect to SSL-enabled sockets.

> ✎ SSL is configured independently of authentication. When authentication and SSL are both enabled, the driver performs the specified authentication method over an SSL connection.

SSLKeyStore

Default Value	Required
None	Yes, if SSL=1

Description

The full path and file name of the Java KeyStore containing an SSL certificate to use during authentication.

See also the property SSLKeyStorePwd on page 35.

SSLKeyStorePwd

Default Value	Required
None	Yes, if SSL=1

Description

The password for accessing the Java KeyStore that you specified using the property SSLKeyStore on page 34.

SSLTrustStore

Default Value	Required
jssecacerts, if it exists. If jssecacerts does not exist, then cacerts is used. The default location of cacerts is jre\lib\security\	No

Description

The full path and file name of the Java TrustStore containing an SSL certificate to use during authentication.

See also the property SSLTrustStorePwd on page 35.

SSLTrustStorePwd

Default Value	Required
None	Yes, if using a TrustStore.

Description

The password for accessing the Java TrustStore that you specified using the property SSLTrustStore on page 35.

UID

Default Value	Required
spark	Yes, if AuthMech=3 (User Name and Password) No, if AuthMech=2 (User Name)

Description

The user name that you use to access the Spark server.

UseNativeQuery

Default Value	Required
0	No

Description

When this option is enabled (1), the driver does not transform the queries emitted by an application, so the native query is used.

When this option is disabled (0), the driver transforms the queries emitted by an application and converts them into an equivalent form in HiveQL.

> ✏ If the application is Spark-aware and already emits HiveQL, then enable this option to avoid the extra overhead of query transformation.

Appendix D Third Party Trademarks and Licenses

Third Party Trademarks

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Apache Spark, Apache, and Spark are trademarks or registered trademarks of The Apache Software Foundation or its subsidiaries in Canada, United States and/or other countries.

All other trademarks are trademarks of their respective owners.

Third Party Licenses

Simple Logging Façade for Java (SLF4J)

Copyright © 2004-2015 QOS.ch

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Apache License, Verison 2.0

The following notice is included in compliance with the Apache License, Version 2.0 and is applicable to all software licensed under the Apache License, Version 2.0.

Apache License

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

   "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

   "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

   "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

   "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

   "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

   "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

   "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

   "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

   "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

   You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or

This product includes software that is licensed under the Apache License, Version 2.0 (listed below):

Apache Commons
Copyright © 2001-2015 The Apache Software Foundation

Apache Hadoop Common
Copyright © 2014 The Apache Software Foundation

Apache Hive
Copyright © 2008-2015 The Apache Software Foundation

Apache Logging Services
Copyright © 1999-2012 The Apache Software Foundation

Apache Spark
Copyright © 2014 The Apache Software Foundation

Apache Thrift
Copyright © 2006-2010 The Apache Software Foundation

Apache ZooKeeper
Copyright © 2010 The Apache Software Foundation