|  |
| --- |
| Cisco Systems |
| Unity Connection JDBC-ODBC Bridge |
| Manual |
|  |
| **Honey Bhandari, Srushti Chaukhande** |
| **3/31/2021** |

|  |
| --- |
| Unity Connection’s JDBC port is not exposed to the public and restricted due to security reasons. However, Unity Connection has an alternate service, called the DB proxy service. By enabling this service, one can access the database through ODBC. This document will help you create the setup for a remote Java application to query content from Unity Connection using a JDBC-ODBC bridge. |

Contents

[Settings on Cisco Unity Connection 2](#_Toc61545067)

[Setting up Informix ODBC driver at the Client 4](#_Toc61545068)

[Creating ODBC datasource on client (Windows) 5](#_Toc61545069)

[Creating Secure ODBC datasource on client (Windows) 8](#_Toc61545070)

[Creating ODBC datasource on client (Linux) 1](#_Toc61545071)4

[Using the datasource in the code 1](#_Toc61545072)5

# Settings on Cisco Unity Connection

1. Login to the Cisco Unity Connection Administration.
2. Search and edit the roles of the user required to be able to query the unity connection database remotely and update privileges with that of Remote Administrator.

3. Edit the Web application password settings of this user to be like this:

4. Go to System settings > Advanced > Connection administration and update “Database Proxy: Maximum Simultaneous Connections” to atleast 10. Max value is 999.

5. Go to Cisco Unity Connection Serviceability. Open Tools > Service Management.
6. Under Optional services, activate Connection Database Proxy service if not activated already.


# Setting up Informix ODBC driver at the Client

1. Download and install Informix client SDK from <http://www14.software.ibm.com/webapp/download/search.jsp?rs=ifxdl>. You need to select version 3.70 for installation. Select appropriate version depending on your client’s OS and if it is 32/64 bit.
2. Installation guide is available in the package. Use that for proper installation.
3. Restart if required.

**Note:** In subsequent sections, the installation path for Informix client SDK is considered as “C:\Program Files (x86)\Informix Client-SDK”. This path can be different depending on where Informix Client-SDK is installed.

# Creating ODBC datasource on client (Windows)

1. Go to Control Panel > Administrative Tools > Data Sources (ODBC)
2. Go to System DSN tab and click on Add

3. In Genaral tab provide the datasource name, say InformixDS

4. Go to the Connection tab and provide the following values:

Host name is that of the unity connection server.
User Id and Password are the credentials of the user which has the Remote Administrator role.
5. Go to the Environment tab and update the locale values as:

If the value is not updateable, just click ok and save the changes and open the DSN again and come back to this environment tab, the value should be updateable then.
6. Go back to the Connection tab and test the connection.

7. Save the datasource, it is ready for use.

# Creating Secure ODBC datasource on client (Windows)

1. Go to path C:\Program Files (x86)\Informix Client-SDK\etc and create a file named “conssl.cfg”, or alternatively open notepad as administrator and add following lines and save at given path:
SSL\_KEYSTORE\_FILE C:/Program Files (x86)/Informix Client-SDK/sslClient/client.kdb

SSL\_KEYSTORE\_STH C:/Program Files (x86)/Informix Client-SDK/sslClient/client.sth

1. Copy and paste server certificate HAProxy\_truststore.pem at path C:\Program Files (x86)\Informix Client-SDK\etc
2. Open command prompt as administrator and run the following commands :
cd C:\Program Files (x86)\Informix Client-SDK\etc
3. gsk8capicmd -keydb -create -db client.kdb -type cms
4. gsk8capicmd -cert -add -db client.kdb -label certificate -file HAProxy\_truststore.pem -format ascii
5. Go to Control Panel > Administrative Tools > Data Sources (ODBC)
6. Go to System DSN tab and click on Add

7. In Genaral tab provide the datasource name, say InformixDS

8. Go to the Connection tab and provide the following values:


Host name is that of the unity connection server, use IP if host name is not accepted.
User Id and Password are the credentials of the user which has the Remote Administrator role.

1. Go to the Environment tab and update the locale values as:

If the value is not updateable, just click ok and save the changes and open the DSN again and come back to this environment tab, the value should be updateable then.
2. Go back to the Connection tab and test the connection.

3. Save the datasource, it is ready for use.

**NOTE: For Mutual Authentication, perform the following additional steps –**

1. Update the conssl.cfg file at path C:\Program Files (x86)\Informix Client-SDK\etc to include this line :
SSL\_KEYSTORE\_LABEL localCA
2. Open command prompt as administrator and run :
i. gsk8capicmd -cert -create -db client.kdb -dn "CN=selfsignedcert,O=altran,OU=cisco,L=gurgaon,ST=haryana,C=IN" -size 2048 -sigalg sha256 -expire 7300 -label localCA -default\_cert yes

ii. gsk8capicmd -cert -extract -db client.kdb -label localCA -format ascii -target client.pem
3. Copy client.pem from C:\Program Files (x86)\Informix Client-SDK\etc and and paste it locally.
4. Navigate to Cisco Unity Connection’s Cisco Unified Operating System Administration page and click on Security drop down and goto Certificate Management

5. Click on upload certificate/certificate chain

6. Select tomcat-trust as certificate purpose and upload client.pem saved earlier.

7. Restart Cisco Tomcat Service using the CLI "utils service restart Cisco Tomcat" on all cluster nodes.
8. The certificate should be visible in certificate list.


**Note:** Secure ODBC connection to Cisco Unity Connection is supported on release 14 and onwards.

# Creating ODBC datasource on client (Linux)

After the installation has been done, follow the below steps:

1. export INFORMIXDIR={absolute path of install directory}
2. export INFORMIXSERVER=ciscounity
3. export INFORMIXSQLHOSTS=$INFORMIXDIR/etc/sqlhosts
4. Copy odbcinst.ini from $INFORMIXDIR/etc/ to /etc/ if not already there.
5. If the file already exists, then append else update with the below:
[ODBC]
Trace = No
TraceFile = /tmp/sql.log
ForceTrace = No
Pooling = No
[ids]
Driver={absolute path of install directory}/lib/cli/iclit09b.so
Setup={absolute path of install directory}/lib/cli/iclit09b.so
APILevel=1
ConnectFunctions=YYY
DriverODBCVer=03.81
FileUsage=0
SQLLevel=1
smProcessPerConnect=Y
Threading = 0
6. Copy odbc.ini from $INFORMIXDIR/etc/ to /etc/ if not already there.
7. If the file already exists, then append else update with the below:
[**InformixDS**]
Driver=ids
Database=UnityDirDb
Description=Unity Connection Database
logonID={user with remote administrator role}
pwd={password of user}
Servername=ciscounity
Trace=0
TraceDLL=idmrs09a.so
TraceFile=odbctrace.out
InstallDir={absolute path of install directory}
Translationdll={absolute path of install directory}/lib/esql/igo4a304.so
DB\_LOCALE=en\_us.utf8
CLIENT\_LOCALE=en\_us.utf8
SERVER\_LOCALE=en\_us.utf8
IFX\_LOCK\_MODE\_WAIT=10
8. Create a file by name sqlhosts at $INFORMIXDIR/etc/
9. Update this file with the entry:
ciscounity onsoctcp {IP address of Unity Connection Server} 20532
10. export LD\_PRELOAD=/usr/lib/libodbc.so
11. Datasource is ready for use.

# Using the datasource in the code

**Java Implementation**

The 2 important differences from a normal JDBC connection in Java from the way JDBC-ODBC connection is configured is:

**Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");**

**String sqlq = "jdbc:odbc:InformixDS";**

Note, that we use the name datasource name “InformixDS” as set during the DSN configuration. Below is a complete runnable program.

package com.cisco.db.connect;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.Statement;

import java.util.ArrayList;

import java.util.List;

/\*\*

 \* @author hbhandar

 \*

 \*/

public class JdbcOdbcConnection {

 private static String sql = "select first 10 u.alias from vw\_user u;";

 /\*\*

 \* @param args

 \*/

 public static void main(String[] args) {

 new JdbcOdbcConnection().getData(sql);

 }

 public void getData(String query) {

 Connection conn = null;

 ResultSet rs = null;

 Statement stmt = null;

 List<String> users = new ArrayList<String>();

 try {

 Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

 String sqlq = "jdbc:odbc:InformixDS";

 conn = DriverManager.getConnection(sqlq);

 stmt = conn.createStatement();

 rs = stmt.executeQuery(sql);

 while (rs.next()) {

 users.add(rs.getString("alias"));

 }

 System.out.println("Total users found: " + users.size()

 + ", Users: " + users.toString());

 } catch (Exception e) {

 e.printStackTrace();

 }

 }

}

**.NET Implementation:**

[Cisco Unity Connection .NET ODBC SDK](http://www.ciscounitytools.com/CodeSamples/Connection/ODBC/ODBC.html) is required to run program given below. Configure a new DSN for secure/non-secure access for example,“InformixDS” and substitute the name in code connection = new OdbcConnection("DSN=InformixDS");

using System;

using System.Data;

using System.Data.Odbc;

using Cisco.UnityConnection.OdbcSdk;

namespace ConsoleApp5

{

 class Program

 {

 private static UnityConnectionServerOdbcSdk \_cxn;

 static void Main()

 {

 \_cxn = new UnityConnectionServerOdbcSdk("ConsoleAppp5");

 OdbcConnection connection;

 try

 {

 Console.WriteLine("Logging in...");

 connection = new OdbcConnection("DSN=InformixDS");

 Console.WriteLine("Initially Connection state is : " + connection.State);

 connection.Open();

 Console.WriteLine("Now Connection state is : " + connection.State);

 }

 catch (Exception ex)

 {

 Console.WriteLine($"Failed logging into server: {ex}");

 Console.ReadLine();

 return;

 }

 OdbcCommand command = connection.CreateCommand();

 command.CommandText = "SELECT Alias from vw\_user";

 OdbcDataReader dataReader = command.ExecuteReader();

 int count = dataReader.FieldCount;

 Console.WriteLine("Printing DataReader");

 for(int i=0; i<count; i++)

 {

 String str = dataReader.GetName(i);

 Console.Write(str);

 }

 Console.WriteLine();

 while (dataReader.Read())

 {

 for (int i = 0; i < count; i++)

 {

 String str = dataReader.GetString(i);

 Console.Write(str + ", ");

 }

 }

 Console.WriteLine();

 dataReader.Close();

 command.Dispose();

 connection.Close();

 Console.WriteLine("Now Connection state is : " + connection.State);

 Console.WriteLine("Complete.");

 Console.ReadLine();

 }

 }

}