IBM Software Group

Troubleshooting Database and Connection Pooling Problems

Thu-Giang Pham

WebSphere® Support Technical Exchange

Agenda

- WebSphere Relational Resource Adapters (RRA)
- Common Problems with RRA
- WebSphere Connection Pooling
- Common Problems with Connection Pooling

Relational Resource Adapters

Resource Adapter (RA)

- System-level software driver that a Java application uses to connect to an enterprise information system (EIS)
- Provides connectivity between the EIS, the application server, and the enterprise application.

Relational Resource Adapters (RRA)

- Is JCA (J2EE Connector Architecture) v1.5 compliant and shipped as a built-in of WebSphere Application Server
- Implements the CCI (Common Connection Interface) contracts for an outbound Resource Adapter as defined in the JCA 1.0 or 1.5 specification
- Supports the configuration and use of data sources implemented as either JDBC data sources (JDBC 2.0 or JDBC 2.0 optional package API or the JDBC 3.0 API) or J2EE Connector Architecture connection factories

Relational Resource Adapters (RRA)

- Preinstalled to provide access to relational databases
- From Administrative Console:
  Resources > Resource Adapters > WebSphere Relational Resource Adapter

  Note: Although the default relational resource adapter settings are viewable, they cannot be changed.

JDBC Providers

- Installed applications use JDBC providers to interact with databases.
- The JDBC provider object supplies the specific JDBC driver implementation class for access to a specific vendor database. To create a pool of connections to that database, you associate a data source with the JDBC provider.
- JDBC provider and the data source objects are functionally equivalent to the J2EE Connector Architecture (JCA) connection factory, which provides connectivity with a non-relational database.

Supported Database and JDBC Drivers

- List current list of supported databases and JDBC drivers for specific version of WebSphere can be found at:
  http://www-1.ibm.com/support/docview.wss?rs=180&uid=swg27006921

JDBC Driver’s Configuration

- From Administrative Console: Resources > JDBC Providers
- Class path - list of paths or JAR file names which together form the location for the resource provider classes.
- Native Library Path - list of paths that form the location for the resource provider native libraries.
  - For type 2 JDBC drivers only (e.g. Oracle OCI, DB2 Universal JDBC Type 2, DB2 Legacy CLI-based).
- Implementation class name
  - Java class name of the JDBC driver implementation. This class is available in the driver file mentioned in the class path description above.

Unsupported JDBC Driver

- The JDBC Driver must implement two data source interfaces defined by Sun Microsystems for working with connection pooling
  - *ConnectionPoolDataSource* - a data source that supports application participation in local and global transactions, excepting two-phase commit transactions
  - *XADataSource* - a data source that supports application participation in any single-phase or two-phase transaction environment. When this data source is involved in a global transaction, the WebSphere Application Server transaction manager provides transaction recovery.

Unsupported JDBC Driver

- Use User-defined JDBC Provider
- The **Implementation class name** is the class that implements the interface *ConnectionPoolDataSource*
  - For example: The implementation class name for **i-net OPTA JDBC Driver** is com.inet.tds.XDataSource.

Data Source

- Is used to obtain connections to a relational database.
- Is analogous to the J2EE Connector Architecture (JCA) connection factory, which provides connectivity to other types of enterprise information systems (EIS).
- Is associated with a JDBC provider, which supplies the driver implementation classes that are required for JDBC connectivity with specific database.
- Multiple data sources with different settings can associate them with the same JDBC provider

Connection Manager Architectures

- In WSAS releases prior to version 5.0, the function of data access was provided by a single connection manager (CM) architecture. This connection manager architecture remains available to support J2EE 1.2 applications.
- New connection manager architecture is provided, based on the JCA architecture supporting the new J2EE 1.3 application style (also for J2EE 1.4 applications).
- These two separate architectures are represented by two types of data sources. To choose the right data source, administrators must understand the nature of their applications, EJB modules, and enterprise beans.

Choice of Data Source

- J2EE 1.2 application - all EJB 1.1 enterprise beans, JDBC applications, or Servlet 2.2 components must use the **4.0** data source.
- J2EE 1.3 (and subsequent releases) application -
  - EJB 1.1 Module - all EJB 1.x beans must use the **4.0** data source.
  - EJB 2.0 (and subsequent releases) Module - enterprise beans that include container-managed persistence (CMP) Version 1.x, 2.0, and beyond must use the **new** data source.
  - JDBC applications and Servlet 2.3+ components - must use the **new** data source.

Creating Data Source

- From Administrative Console: **Resources > JDBC Providers**
- Choose the JDBC Provider
- Under Additional Properties, click the **Data Sources** link that is appropriate for your application
- Customer properties can be defined or added to the data source

Configuration Requirement

- Check Information Center section “**Data source minimum required settings, by vendor**” at URL:

  http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/topic/com.ibm.websphere.base.doc/info/aes/ae/rdat_minreq.html

  and click on the link Detailed Requirements for creating data sources.

Invalid Custom Property

- Check SystemOut.log for DSRA message. For example:

  ```
  DSRA8020E: Warning: The property
  'SetBigStringTryClob' does not exist on the
  DataSource class
  oracle.jdbc.pool.OracleConnectionPoolDataSour
  ce
  ```

  This error indicates that custom property ‘SetBigStringTryClob’ is not valid to Oracle JDBC Driver.

Tip for Using Oracle JDBC Driver

- To specify connection properties for Oracle data source, create custom property with Name/Value pair:
  Name: connectionProperties
  Value: oracle.jdbc.V8Compatible=true

- Multiple connection properties can by specified using ";" (semicolon) as delimiter. For example:
  Value:
  oracle.jdbc.V8Compatible=true;oracle.jdbc.RetainV9LongBindBehavior=true

Using Type 2 JDBC Driver

- Must set all required environment variables required by the JDBC Driver.
- For Oracle
  - ORACLE_HOME
  - LD_LIBRARY_PATH (Native library path)
- For DB2
  - DB2INSTANCE
  - LD_LIBRARY_PATH (Native library path)

Common Configuration Problems

Common Class Loading Exceptions

- ClassCastException
- ClassNotFoundException
- NoClassDefFoundException
- UnsatisfiedLinkError
- See Information Center, section **Class loading exceptions** for explanations and troubleshooting tips.

ClassNotFoundException

- Verify that the class path is correct and the JAR file(s) for the JDBC provider exist and have access permission on the server where the application with the associated data source is running.
- If the definition of the data source includes a WebSphere variable, make sure that the variable is defined and set at a correct scope. From the Administrative console:
  Environment > WebSphere Variables

ClassCastException

- Check to make sure that there is only one copy of each JAR file that is required for the JDBC provider. Search the file system starting from the <*WAS_install_root*> directory for duplication.
- Make sure the application's EAR file does not include the JDBC driver's JAR files.

UnsatisfiedLinkError

- The JVM throws UnsatisfiedLinkError when it cannot find the native library or when the JVM has already loaded the native library.
- This exception only occurs when you are using type 2 JDBC drivers (for example, DB2 CLI Legacy, DB2 Universal Type 2, and Oracle OCI drivers).
- Note: Type 2 JDBC drivers require that some binary code is loaded on each client machine (WebSphere Application Server machine).

UnsatisfiedLinkError

- If there is no value specified for the native library path in your JDBC Provider’s configuration, you might get the UnsatisfiedLinkError:
  - On Windows platforms, set the system environment PATH to include the path where the *.dll files can be loaded.
  - On UNIX platforms, you might need to set up the user’s profile to set the environment variables before starting the WebSphere Application Server processes.
- 32-bit versus 64-bit native libraries
  - 32-bit WebSphere Application Server, must use 32-bit JDBC Driver
  - 64-bit WebSphere Application Server, must use 64-bit JDBC Driver
- Examine SystemOut.log for the UnsatisfiedLink error, then review the message text for native library file name (see speaker note for example)

Limitation and Known Problem

- Currently, only one version of JDBC Driver can be used within an application server.
- Do not bundle the JDBC driver's jar files within applications
- Oracle 10g is supported officially in WebSphere Application Server V6.
- If using the Oracle 10g JDBC driver, data source must use the data store helper class `com.ibm.websphere.rsadapter.Oracle10gDataStoreHelper`

Checking JDBC Driver Version

- Some JDBC Driver provides a main class which can be executed from java command to get the version. For example
  - DB2 Universal JDBC Driver
    ```
    java com.ibm.db2.jcc.DB2Jcc –version
    ```
  - WebSphere embed Connect JDBC
    ```
    java com.ibm.websphere.jdbc.sqlserver.SQLServerDriver
    ```
- Can use the JDBC DatabaseMetaData object to get the version information. For example:

```
DriverManager.registerDriver(new
    oracle.jdbc.driver.OracleDriver());
Connection con =
    DriverManager.getConnection(url, user, password);
DatabaseMetaData dmd = con.getMetaData();
System.out.println("JDBC Driver version: " +
    dmd.getDriverVersion());
```

Tool to Use for Class Loader Problem

- Class Loader Viewer
  - From Administrative Console
    Troubleshooting > Class Loader Viewer
  - Use the “Search” function to search class loaders for the following:
    - Specific strings
    - Specific .jar files
    - The names of files in a specific directory
    - The names of files loaded by a specific class loader

Connection Pooling

J2C Component

- Is the WSAS implementation of JCA 1.5.
- JCA 1.5
  - provides a standardized way (Outbound Contracts) for a J2EE container to provide Connection Management services, such as Connection pooling, Transaction Management, and Security Management. Thus any JCA 1.5 compliant Resource Adapter should be able to interoperate with WSAS.
  - defines system level contracts (Inbound Contracts) for a message provider to connect to endpoints (i.e. message-driven bean applications) in WSAS via a resource adapter.
  - defines a contract for the application server to provide thread management services for resource adapters.

Interaction with other WSAS Components

- J2C calls to:
  - Transaction Manager
  - Security
  - Resource Adapters
- J2C is called by:
  - Resource Adapters
  - System Management (startRA, etc…)
  - JNDI (resource lookup)
- Dependencies on security component
  - Main Security dependency is on getSubject method which is called during allocateConnection to obtain credentials for Container-managed authentication case.
  - J2C also calls utility methods to extract credential information from a Subject for Connection matching.

Connection Management (CM)

- The CM architecture for both relational and procedural access to enterprise information systems (EIS)
- Based on the J2EE Connector Architecture (JCA) specification.
- Pools and manages connections within an application server
- Manages connections obtained through both resource adapters (RAs) defined by the JCA specification, and data sources defined by the Java Database Connectivity (JDBC) 2.0 (and later) Extensions specification.
- To make data source connections manageable by the CM, the WSAS provides a relational resource adapter (WebSphere RRA) that enables JDBC data sources to be managed by the same CM that manages JCA connections.

Common Problems

ConnectionWaitTimeoutException

- Can occur due to the pool over use
- Review connection pool:
  - The maximum number of connections for a given pool is set too low.
  - Connection timeout is set too low.
  - Application does not close some connections (connection leak), or it returns connections back to the pool at a very slow rate.

ConnectionWaitTimeoutException

- **Can occur when servlets are migrated from v2.2 to 2.3**
  - **See technote “Servlet connection to database fails after migrating from WebSphere Application Server V4 to V5 and later versions” for explanation: http://www-1.ibm.com/support/docview.wss?&uid=swg21192666**

Connection Leak

- A connection leak occurs when the application uses a connection but it never explicitly calls the close() method on the connection object to return the connection back to the connection pool for reuse.
- You might have a connection leak if your application receives exceptions such as
  - com.ibm.websphere.ce.cm.ConnectionWaitTimeoutException or
  - com.ibm.websphere.ce.j2c.ConnectionWaitTimeoutException when attempting to access a data source or JCA-compliant resource adapter, respectively.

Troubleshoot Connection Leak

- Trace Specification
  - ConnLeakLogic=all
- Traces details such as the getConnection call stack for connections that have been in use for a long time (10 seconds).
- Included in 'WAS.j2c' trace.
- ConnLeakLogic may be used alone, but will only generate output during FFDC processing on a ConnectionWaitTimeout scenario

Many Connections to Backend Resource

- A separate connection pool exists for each configured data source or connection factory.
- Application Server creates a separate instance of the connection pool in each application server that uses the data source or connection factory.
  - For example: In three server cluster in which all of the servers use *myDS*, and *myDS* has a max connections setting of 10, then you would generate up to 30 connections to the backend resource.
- Each entity bean transaction requires an additional database connection, dedicated to handling the transaction.

Deadlock

- Application requires more than one concurrent connection per thread, and the database connection pool is not large enough for the number of threads.
- Troubleshooting
  - Collect Javacores/Javadumps (see MustGather document for the corresponding version and operating system)
  - Use **IBM Thread and Monitor Dump Analyzer for Java Java Technology**
- Solution
  - Increase the **Maximum Connections** value for the database connection pool by at least one.

High Number of Connection Handles

- **Unshareable** means that there is a 1:1 relationship between the connection handle and physical connection
- **Shareable** allows for greater scalability. There is a n:1 relationship between connection handle and physical connection (managed connection)
  - The maximum number of connection handles can’t be defined. That means it is unlimited depending on system resource. The JVM heapsize may run out

Authentication data configuration

- Missing J2C Authentication Alias on the data source
- Application versus Container *res-auth*
- APAR PK03287 "Dynamically update J2C authentication data at run time”
  - WebSphere security mBean provide a method for user to dynamically push new authentication data to run time. So new authentication data takes effect without restarting servers.
  - The wsadmin command:

    ```
    $AdminControl invoke $SecurityAdmin
    updateAuthDataCfg null
    ```

Programming model

- Improper caching of connection handles
- Direct JNDI lookup of connection factory or data source
  - J2CA0122I: Resource reference {0} could not be located, so the following default values are used: {1}
- Spinning threads from a resource adapter
- LTC scope issues
  - Multiple connections
  - Connections reserved for long periods
  - See technote 1217062 and developerWorks article Default behavior of managed connections in WebSphere Application Server (See link on References slide)

StaleConnectionExceptions

- Indicates that the connection currently held is no longer valid.
- See Stale connections section in the WebSphere Information Center for information about how to detect stale connections and how to recover from stale connections
- Check timeout setting on the connection pool
- Determine if a specific query is getting the exception
- Check for potential network problem

Statement Cache

- Application calls prepareStatement on its Connection handle. RRA in this example finds the statement in the statement cache. A wrapper is created for the PreparedStatement object, attached to the Connection wrapper, and the new wrapper is returned to the app
- If same prepared statement works the first time and fails on subsequent time, there is a chance that the statement cache is corrupt – try disabling the cache by setting the statement cache size to zero

Debug Procedure for CM

- Start by searching the SystemOut.log files for J2CAxxxx messages.
- If traces are needed, enable them on the affected server and recreate the problem.
- If RA initialization is important to the problem
  - Make sure the trace includes the startRA timeframe.
- If ConnectionFactory initialization is important to the problem
  - Make sure the trace includes the getObjectInstance timeframe.

Debug Procedure for CM

- Use collector tool to gather important files
- Configuration files of interest:
  - resources.xml – for ConnectionFactory/DataSource configuration. Make sure to get the one for the corresponding scope
  - ra.xml – for RA configuration
  - ibm-ejb-bnd.xmi or ibm-web-bnd.xmi – for resource reference data
  - ejb-jar.xml or web.xml – for res-auth value

Trace Specifications

- J2C/Connection Pooling problem
  - WAS.j2c=all
- RRA problem
  - RRA=all
- JDBC Driver
  - WAS.database=all
    Note: Need the debug jar file (e.g. ojdbc14_g.jar for Oracle JDBC Driver)

Trace Analysis

- Find primary error (J2CAxxxx or RA-specific message).
- Identify J2C, RA, and/or Transaction objects and the WSAS thread involved.
- Follow progress of thread leading up to the problem.
- Follow history of significant J2C, RA, and/or Transaction objects leading up to the problem.
- Identify abnormal events or configuration data that would lead to the problem.
- Decide whether the cause lies with configuration, application, WSAS, or RA.

Summary

- Common problems and troubleshooting configurations of JDBC Provider and data sources
- Common problems and troubleshooting connection pooling

References

- Sharing connections in WebSphere Application Server V5
- Database authentication in WebSphere Application Server V5
- Default behavior of managed connections in WebSphere Application Server
- WebSphere Application Server V6: JCA Connection Problem Determination

Additional WebSphere Product Resources

- Discover the latest trends in WebSphere Technology and implementation, participate in technically-focused briefings, webcasts and podcasts at: www.ibm.com/developerworks/websphere/community/
- Learn about other upcoming webcasts, conferences and events: www.ibm.com/software/websphere/events_1.html
- Join the Global WebSphere User Group Community: www.websphere.org
- Access key product show-me demos and tutorials by visiting IBM Education Assistant: www.ibm.com/software/info/education/assistant
- Learn about the Electronic Service Request (ESR) tool for submitting problems electronically: www.ibm.com/software/support/viewlet/ESR_Overview_viewlet_swf.html
- Sign up to receive weekly technical My support emails: www.ibm.com/software/support/einfo.html

Questions and Answers