Red Hat JBoss Data Virtualization 6.4

Administration and Configuration Guide

This guide is for administrators.

Red Hat JBoss Data Virtualization 6.4 Administration and Configuration Guide

This guide is for administrators.

Red Hat Customer Content Services

Legal Notice

Abstract

Read this guide to learn how to administer Red Hat JBoss Data Virtualization.

Table of Contents

PART I. INTRODUCTION TO ADMINISTERING JBOSS DATA VIRTUALIZATION

CHAPTER 1. ADMINISTRATION TOOLS

1.1. ADMINISTRATION TOOLS OVERVIEW

The following tools are available for you to use to administer Red Hat JBoss Data Virtualization:

Management Console

Management Command Line Interface (CLI)

AdminShell

Admin API

JBoss Operations Network

1.2. MANAGEMENT CONSOLE

1.2.1. Management Console and JBoss Data Virtualization

Red Hat JBoss Data Virtualization provides a plug-in for the web-based Management Console. It provides a web interface that allows you to configure and monitor Data Virtualization services deployed upon a running JBoss EAP instance.

> **NOTE**
>
> For more information about general use of the Management Console, not specific to JBoss Data Virtualization, see the *Red Hat JBoss Enterprise Application Platform Administration and Configuration Guide*.

1.2.2. Log in to the Management Console

Prerequisites

- JBoss EAP 6 must be running.

- You must have already created a user with permissions to access the Console.

1. Launch your web browser and go to this address: http://localhost:9990/console/App.html
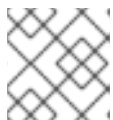
   > **NOTE**
   >
   > Port 9990 is predefined as the Management Console socket binding.

2. Enter your username and password to log in to the Management Console.
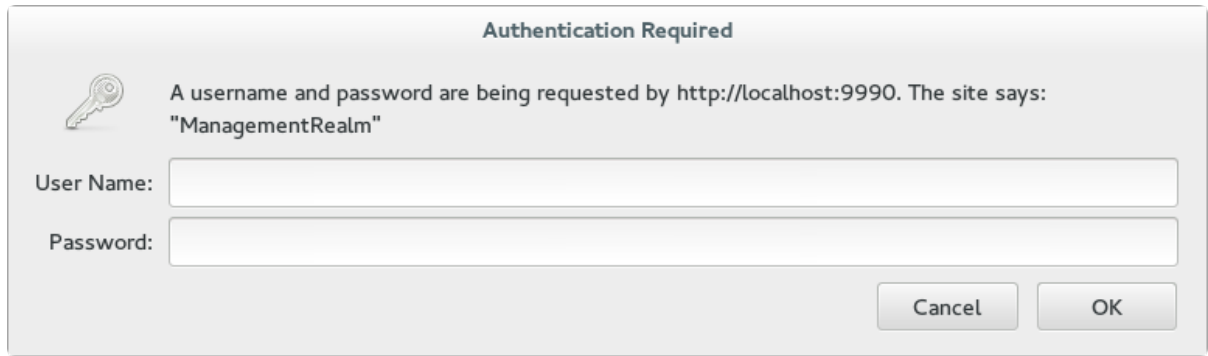
Figure 1.1. Log in screen for the Management Console

Result

Once logged in, you are redirected to the following address and the Management Console landing page appears: http://localhost:9990/console/App.html#home

1.2.3. Management Console - Configuration tab

The `Configuration` tab contains both general and Data Virtualization-specific configuration properties. Click `Teiid` in the left-hand navigational tree to view Teiid configuration. These are the sub-categories:
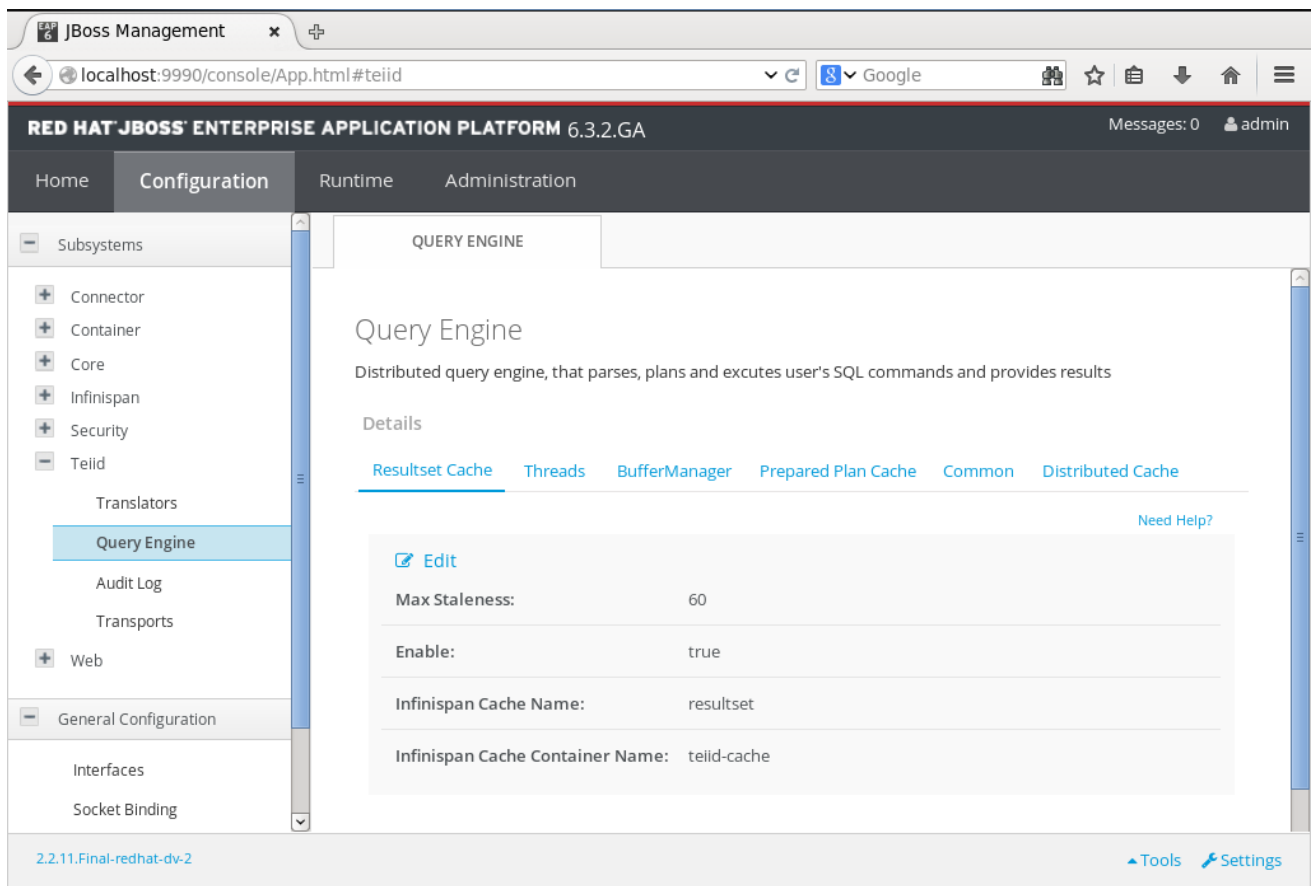
Figure 1.2. Configuration tab

- Query Engine - view and configure core Teiid engine properties.

- Translators - view and remove the Translators configured in Teiid.

- Transports - view, add and remove transports to the Teiid engine.

- Audit Logs - turn on or off audit logging for Teiid.

Use the `Configuration` tab to change any aspect of Teiid's configuration Various different configuration properties are organised on different tabs. Click the `Need Help` link on each tab to see description of each field.
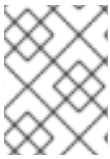
> **NOTE**
>
> Some properties require you to restart the server before changes you have made will take affect.

1.2.4. Management Console - Runtime tab

The `Runtime` tab shows runtime information about the Red Hat JBoss EAP server and the Teiid subsystem. You can view runtime information about the subsystem by clicking `Virtual Databases` in the left-hand side navigational tree.

Figure 1.3. Runtime tab

All the VDBs deployed on the server are shown in top level table. When you select and highlight a VDB, more details about that VDB are displayed in the tabs below. Each of these tabs are divided into grouping of the functionality.

- Summary: This panel shows the details about the selected VDB and also any properties that are associated with that VDB, along with any other VDBs this VDB imports. This tab is designed to give a quick overview of the VDB status.

- Models: This panel shows all the models that are defined in a given VDB, and shows each model's translator name and source connection JNDI name. It also shows the type of models and if it is multi-source or not. When a particular model is selected it shows all properties of that model that are defined and also shows any errors associated with the model. When your VDB is not deployed in the active status, you must verify these errors and fix to resolve any deployment issues.

  The **DDL** button in the Models panel shows the schema for the given model. Note that this does

not work for XML based models that are designed using Teiid Designer. The tool lets you edit the JNDI name by double clicking on them and modifying them. This useful if you would like to change the JNDI name in a given environment.

- Overrides: This panel shows the all the overridden translators and their properties.

- Teiid: This screen shows Teiid runtime metrics.

- Caching: Caching panel shows caching statistics of resultset cache as to how effectively the cache is being used. It also shows all the internal materialized views in the VDB and their load status as to when they were loaded. It also gives options to invalidate a specific view or all the views in a VDB, so that they can refresh or reload the contents from source. This panel also provides a UI to flush the entire the resultset cache contents or prepared plan cache contents for the selected VDB.

- Data Roles: Data Roles panel shows the all the policies that defined in the VDB using the Teiid Designer or hand coded in the vdb.xml file. For each selected policy, it also lists the permissions for that policy as to what kind of authorizations user has and shows the mapped enterprise role assignments to that policy. You can even add or remove an enterprise role from the policy using the this UI.

- Requests: This panel shows all the current requests against the selected VDB at the time of VDB selection. You can click refresh to get a more up to date requests. The top table in the panel shows the user submitted requests to the teiid engine, when one of those requests are selected, then the bottom table shows all the source level queries that are submitted to the physical sources by Teiid engine. Using this UI, user can also submit a cancel request to a user level query. Since cancel is an asynchronous operation, the operation is not guaranteed as query may already been finished, by the time cancel is submitted.

- Sessions: This panel shows all the active sessions that are connected to the selected VDB. It shows their connection properties and also gives an option to terminate either a selected session or all the sessions.

1.2.5. Management Console - Administration tab

You can configure roles and groups using Administration tab. Roles for a user to be included in and excluded from can be configured in the Management Console. You can also configure groups to be included or excluded from a role. Only users in the SuperUser or Administrator roles can perform this configuration.
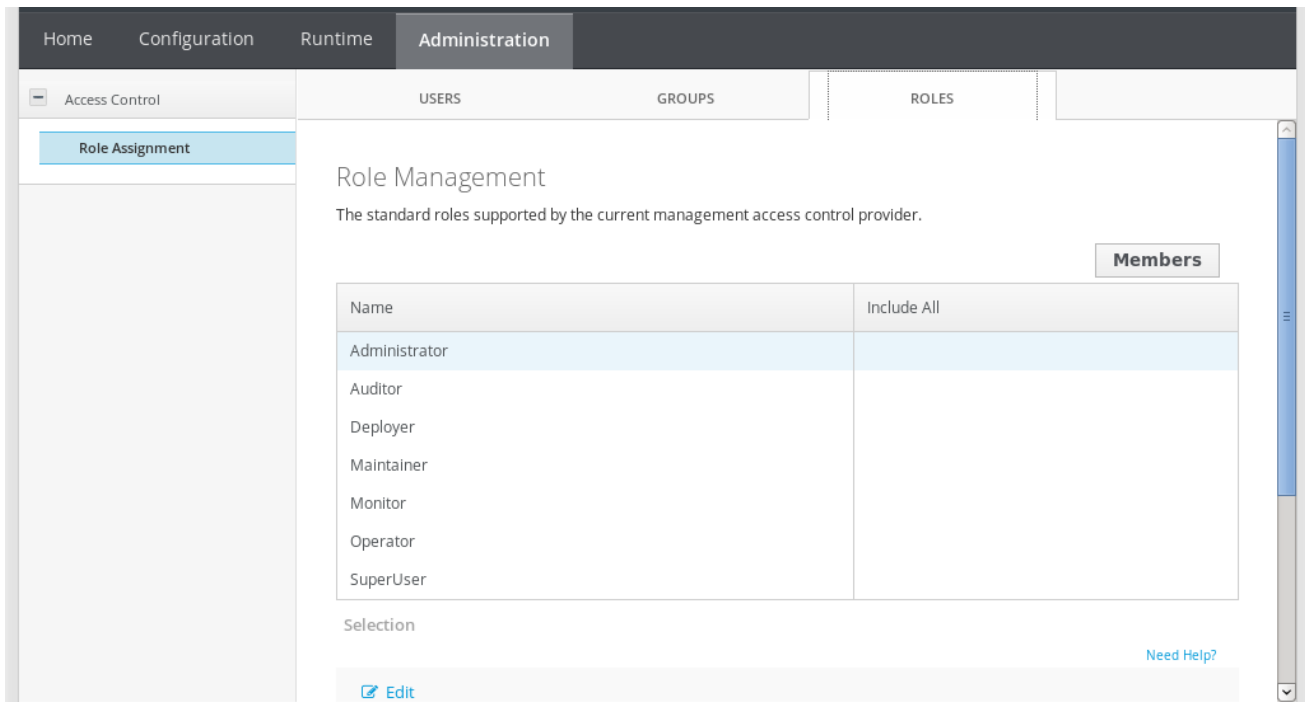
Figure 1.4. Administration tab

For more information related to User Management, see *Red JBoss EAP Administration and Configuration guide*.

1.3. MANAGEMENT CLI

1.3.1. Launch the Management CLI

Procedure 1.1. Launch CLI in Linux or Microsoft Windows Server

- - **Launch the CLI in Linux**
    Run the **EAP_HOME/bin/jboss-cli.sh** file by entering the following at a command line:

    ```
    $ EAP_HOME/bin/jboss-cli.sh
    ```

  - **Launch the CLI in Microsoft Windows Server**
    Run the **EAP_HOME\bin\jboss-cli.bat** file by double-clicking it, or by entering the following at a command line:

    ```
    C:\>EAP_HOME\bin\jboss-cli.bat
    ```

1.3.2. Quit the Management CLI

From the Management CLI, enter the **quit** command:

```
[domain@localhost:9999 /] quit
```

1.3.3. Connect to a Managed Server Instance Using the Management CLI

Prerequisites

- Section 1.3.1, "Launch the Management CLI"

Procedure 1.2. Connect to a Managed Server Instance

- **Run the `connect` command**
  From the Management CLI, enter the `connect` command:

  ```
  [disconnected /] connect
  Connected to domain controller at localhost:9999
  ```

  - Alternatively, to connect to a managed server when starting the Management CLI on a Linux system, use the `--connect` parameter:

    ```
    $ EAP_HOME/bin/jboss-cli.sh --connect
    ```

  - The `--connect` parameter can be used to specify the host and port of the server. To connect to the address **192.168.0.1** with the port value **9999** the following would apply:

    ```
    $ EAP_HOME/bin/jboss-cli.sh --connect --
    controller=192.168.0.1:9999
    ```

1.3.4. Obtain Help with the Management CLI

Summary

Sometimes you might need guidance if you need to learn a CLI command or feel unsure about what to do. The Management CLI features a help dialog with general and context-sensitive options. (Note that the help commands dependent on the operation context require an established connection to either a standalone or domain controller. These commands will not appear in the listing unless the connection has been established.)

Prerequisites

- Section 1.3.1, "Launch the Management CLI"

1. **For general help**
   From the Management CLI, enter the **help** command:

   ```
   [standalone@localhost:9999 /] help
   ```

2. **Obtain context-sensitive help**
   From the Management CLI, enter the **help -commands** extended command:

   ```
   [standalone@localhost:9999 /] help --commands
   ```

3. For a more detailed description of a specific command, enter the command, followed by *--help*.

   ```
   [standalone@localhost:9999 /] deploy --help
   ```

Result

The CLI help information is displayed.

1.4. ADMINSHELL

1.4.1. AdminShell Features

AdminShell provides the following features:

Administration

AdminShell can be used to connect to a JBoss Data Virtualization instance in order to perform various administrative tasks.

Data Access

AdminShell can be used to connect to a virtual database (VDB) and run SQL commands to query VDB data and view results.

Migration

AdminShell can be used to develop scripts that will move VDBs and associated components from one development environment to another. (Users can test and automate migration scripts before executing them in production deployments.)

Testing

The built-in JUnit Test Framework allows users to write regression tests to check system health and data integrity. The written tests validate system functionality automatically, removing the need for manual verification by QA Personnel.

1.4.2. AdminShell Scripting

Use the Groovy language to write AdminShell scripts to automate tasks. AdminShell provides a custom library of Groovy functions to help administer JBoss Data Virtualization. These custom functions can be used in isolation or alongside other Groovy code.

Here are some basic rules. Memorise these to become familiar with the syntax:

- All commands and functions are case sensitive.

- Groovy commands are not required to end with a semicolon; this is optional.

- A function will take zero or more parameters as input.

    - Function parameters are provided within parentheses.

    - String parameters must be provided within single or double quotes.

    ```
    connectAsAdmin("localhost", "9999", "user", "password", "conn1")
    ```

- Other Java classes and methods can be accessed and invoked from within scripts as long as required libraries are already in the class path. (JAR files added to the **lib** directory will be included in the class path automatically.)

```
import my.package.*;
myObject = new MyClass();
myObject.doSomething();
```

NOTE

You should disconnect from JBoss Data Virtualization before exiting AdminShell by entering the **disconnect()** command.

References

- For more information about Groovy scripting, refer to http://groovy.codehaus.org/.

- For more information about Groovy scripting and SQL, refer to http://groovy.codehaus.org/Database+features.

- For more information about testing using Groovy scripting, refer to http://groovy.codehaus.org/Unit+Testing and http://junit.org/.

1.4.3. AdminShell Help

All of the custom administrative methods available in AdminShell can be listed by using the **adminHelp()** method (note that only the custom AdminShell methods will be shown):

```
adminHelp()
```

If you require a specific method's definition and input parameters, use **adminHelp("*METHOD*")**:

```
adminHelp("deploy")

/*
*Deploy a VDB from file
*/
void deploy(
String /* file name */)
throws AdminException
throws FileNotFoundException
```

Use the **sqlHelp()** method to list all SQL extension methods:

```
sqlHelp()
```

If you want to obtain a specific method's definition and input parameters, use **sqlHelp("*METHOD*")**.

1.4.4. Install the Interactive AdminShell

The AdminShell can be installed and run from any machine that has access to the running Red Hat JDV server.

Procedure 1.3. Install the AdminShell Anywhere

1. Open a command line terminal.

2. Go to the AdminShell directory: cd **/EAP_HOME/dataVirtualization/teiid-adminshell/**.

3. Copy the zipped **teiid-VERSION-adminshell-dist.zip** file to your desired location.

4. Unzip the AdminShell: **unzip teiid-VERSION-adminshell-dist.zip**

1.4.5. Launch the Interactive AdminShell

Procedure 1.4. Launch the Interactive AdminShell

- Run the **./adminshell.sh** command.

  The log output is as follows:

  ```
  ========================================================================
  ==

    Teiid AdminShell Bootstrap Environment

    TEIID_HOME  = /EAP_HOME/dataVirtualization/teiid-
  adminshell-VERSION
    CLASSPATH   = /EAP_HOME/dataVirtualization/teiid-
  adminshell-VERSION/lib/patches/*:/EAP_HOME/dataVirtualization/teiid-
  adminshell-VERSION/lib/teiid-
  adminshell-VERSION.jar:/EAP_HOME/dataVirtualization/teiid-
  adminshell-VERSION/lib/*
    JAVA        = /usr/lib/jvm/java-1.7.0-openjdk.x86_64/bin/java


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

  ===> [import static org.teiid.adminshell.AdminShell.*; import static
  org.teiid.adminshell.GroovySqlExtensions.*; import
  org.teiid.adminapi.*;]
  Groovy Shell (1.7.2, JVM: 1.7.0_25)
  Type 'help' or '\h' for help.
  ----------------------------------------------------------------
  ---------------
  groovy:000>
  ```

Result

The interactive AdminShell is running. At this point you can execute individual commands line by line.

> **NOTE**
>
> You can exit the interactive AdminShell at any time by entering the **exit** command.

1.4.6. AdminShell Basic Commands

Table 1.1. AdminShell Basic Commands

Command	Description
`adminHelp();`	Displays all of the custom AdminShell methods available.
`adminHelp("METHOD");`	Displays the definition and input parameters for the AdminShell method supplied.
`sqlHelp();`	Displays all of the custom SQL AdminShell methods available.
`sqlHelp("METHOD");`	Displays the definition and input parameters for the SQL AdminShell method supplied.
`println "STRING";`	Print something to console.
`sql = connect();`	Get an extended Groovy SQL connection using the default connection specified in the `connection.properties` file.
`sql.execute("SQL");`	Run any SQL command.
`connectAsAdmin();`	Connects as an admin user using the default connection specified in the `connection.properties` file. This command does not require a VDB name. Note that this is an admin connection, not a JDBC connection, so you cannot issue SQL commands using this connection. Note that if SSL is being used, you would need to adjust the connection URL and the client SSL settings as necessary (typically only for 2-way SSL).
`println getConnectionName();`	Returns the active connection name.
`useConnection("cNAME");`	Switches to using the given connection.
`disconnect();`	Disconnects the active connection.
`disconnectAll();`	Disconnects all connections (both active and suspended).

1.4.7. Execute a Script in Non-interactive AdminShell

Procedure 1.5. Execute a Script in Non-interactive AdminShell

1. **Open a Command Line Terminal**

2. **Run the Script**
   Run the **/adminshell.sh load [-D*param*=*value*] *PATH/FILENAME*** command.

> **NOTE**
>
> Optional properties passed in using **-D** can be accessed from within the script file by using **System.getProperty()**.
>
> ```
> value = System.getProperty("param")
> ```

IMPORTANT

Any connections established within the script should be disconnected before the script finishes.

1.4.8. Helpful Tips for the Interactive AdminShell

Table 1.2. Interactive AdminShell Commands

Command	Description
↑ ('up arrow')	Displays previous commands executed, starting with the most recent.
`help`	Displays additional commands provided specifically for use within the interactive shell.

Shell Behavior

The interactive shell uses a special interpreter and displays behavior different from what would be expected from running a Groovy script:

- def statements do not define a variable in the context of the shell; For example, do not use `def x = 1`, use `x = 1`.

- The shell cannot parse Groovy classes that use annotations.

1.4.9. Save a Script in Interactive AdminShell

Procedure 1.6. Save a Script in Interactive AdminShell

1. **Open the Interactive AdminShell**

    a. Open a command line terminal.

    b. Navigate to `/EAP_HOME/dataVirtualization/teiid-adminshell-VERSION/`.

    c. Run the `./adminshell.sh` command.

2. **Start Recording to a File**
   Enter the `record start PATH/FILENAME` command.

3. **Enter Desired Commands to Record**
   Enter a series of commands for which you want to record the input/output.

4. **Stop Recording to the File**
   Enter the `record stop` command.

Result

All input and output between the moments you issue the `record start` and `record stop` commands are captured in *PATH/FILENAME*.

NOTE

Since both input and output are recorded, the file will need some editing before it can be executed as a script file.

1.4.10. Execute a Script in Interactive AdminShell

Procedure 1.7. Execute a Script in Interactive AdminShell

1. **Open the Interactive AdminShell**

   a. Open a command line terminal.

   b. Navigate to **/EAP_HOME/dataVirtualization/teiid-adminshell-VERSION/**.

   c. Run the **./adminshell.sh** command.

2. **Run the Script**

   o **Run the Script using the Interactive Shell load Command**
   Run the **load PATH/FILENAME** command.

   o **Run the Script using the Groovy evaluate Command**
   Run the **evaluate("PATH/FILENAME" as File)** command.

   NOTE

   You can also execute a script by entering it line by line after opening the interactive shell.

1.4.11. Launch the AdminShell GUI

Procedure 1.8. Launch the AdminShell GUI

1. **Navigate to the adminshell Directory**

   a. Open a command line terminal.

   b. Navigate to **/EAP_HOME/dataVirtualization/teiid-adminshell-VERSION/**.

2. **Launch the adminshell-console Script**
   Run the **./adminshell-console.sh** command.

Result

The AdminShell GUI is displayed.
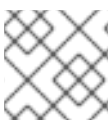
   NOTE

   You can exit the AdminShell GUI at any time by clicking on **File → Exit**.

1.4.12. Save a Script in AdminShell GUI

Procedure 1.9. Save a Script in AdminShell GUI

1. **Navigate to the `adminshell` Directory**

   a. Open a command line terminal.

   b. Navigate to **`/EAP_HOME/dataVirtualization/teiid-adminshell-VERSION/`**.

2. **Type a Script**
   Type your script in the upper *script window* of the AdminShell GUI. You may find it useful to test the script as you are preparing it by executing it via **Script → Run**.

3. **Save the Script**

   a. Select **File → Save As**.

   b. Choose a location and filename for the script and click on **Save**.

1.4.13. Execute a Script in AdminShell GUI

Procedure 1.10. Execute a Script in AdminShell GUI

1. **Open the AdminShell GUI**

   a. Open a command line terminal.

   b. Navigate to **`/EAP_HOME/dataVirtualization/teiid-adminshell-VERSION/`**.

   c. Run the **`./adminshell-console.sh`** command.

2. **Prepare the Script**

   ○ **Type a New Script**
      Type your script in the upper *script window* of the AdminShell GUI.

   ○ **Load a Saved Script**

      a. Select **File → Open**.

      b. Locate the desired script file and click on **Open**.

         The script will appear in the upper *script window* of the AdminShell GUI.

3. **Run the Script**
   Select **Script → Run**.

1.4.14. AdminShell Connection Properties

The **`EAP_HOME/dataVirtualization/teiid-adminshell-VERSION/connection.properties`** file provides the default connection properties used by AdminShell to connect to an JBoss Data Virtualization instance:

```
jdbc.user=user
jdbc.password=user
jdbc.url=jdbc:teiid:admin@mm://localhost:31000;

admin.host=localhost
```

```
admin.port=9999
admin.user=admin
admin.password=admin
```

A call to **connect()** or **connectionAsAdmin()** without any input parameters will connect to JBoss Data Virtualization using the settings defined in this properties file. **connect()** uses those properties prefixed with **jdbc** and **connectAsAdmin()** uses those properties prefixed with **admin**. Alternatively, you can include parameters in the **connect()** or **connectAsAdmin()** methods to connect using properties of your choice:

```
connect("URL", "USER", "PASSWORD")
```

> **WARNING**
>
> Do not leave passwords stored in clear text. The example above is for demonstration purposes only.
>
> If you want to store your password in the file, take the necessary measures to secure it. Otherwise, do not use this feature at all and, instead, input your password interactively (or some other secure way.)

1.4.15. Multiple Connections in AdminShell

Using AdminShell, users can manage more than one connection to one or more JBoss Data Virtualization instances. For example, a user might want to have one connection to a development server and another to an integration server, simultaneously.

Every time a new connection is made, it is assigned a unique name and it becomes the active connection. If there is already a connection, it will be suspended (it will not be closed).

The **getConnectionName()** method returns the name of the active connection. The connection name can be assigned to a variable *cName* by the following command:

```
cName = getConnectionName();
```

The name of the current connection is required in order to switch from one to another. To change the active connection, use the **useConnection()** command, supplying the name (or a variable with the name assigned) of the connection you wish to use:

```
useConnection(cName);
```

Example

The following example demonstrates how to use two connections and switch between them:

```
// Create a connection
connectAsAdmin();

//Assign the connection name to conn1
```

```
conn1 = getConnectionName();

deploy("file.vdb")

// Create a second connection (which is now the active connection)
connectAsAdmin();

//Assign the new connection name to conn2
conn2 = getConnectionName();

deploy("file.vdb")

// Switch the active connection back to conn1
useConnection(conn1);

// Close the active connection (conn1)
disconnect();
```

1.4.16. Example Scripts

Example 1.1. Deploying a VDB

```
connectAsAdmin();
deploy("/path/to/<name>.vdb");

// check to validate the deployment
VDB vdb = getVDB("<name>", 1);
if (vdb != null){
    print (vdb.getName()+"."+vdb.getVersion()+" is deployed";
}
else {
    print ("<name>.vdb failed to deploy";
}
```

Example 1.2. Create a Datasource (Oracle)

```
connectAsAdmin();

// first deploy the JDBC jar file for Oracle
deploy("ojdbc6.jar");

props = new Properties();
props.setProperty("connection-url","jdbc:oracle:thin:@<host>:1521:
<sid>");
props.setProperty("user-name", "scott");
props.setProperty("password", "tiger");

createDataSource("oracleDS", "ojdbc6.jar", props);
```

Example 1.3. Execute SQL Query against Teiid

```
sql = connect("jdbc:teiid:<vdb>@mm://<host>:31000", "user", "user");

// select
sql.eachRow("select * from sys.tables") { println "${it}" }

// update, insert, delete
sql.execute(<sql command>);
```

1.4.17. AdminShell FAQ

Q: Why won't the `adminhelp` command work in the GUI tool?

A: The AdminShell GUI environment does not understand shell commands such as **load**, **help**, and **adminhelp**, since they are not directly supported by Groovy. In the GUI you should use the equivalent functional form; for example, instead of using **adminhelp** use **adminHelp()**.

Q: Are there any pre-built scripts available?

A: Not currently.

Q: What is the difference between `connectAsAdmin()` and `connect()`?

A: The **connectAsAdmin()** method creates a contextual connection to the AdminAPI of JBoss Data Virtualization. The **connect()** method returns an extension of the Groovy SQL object to be used for SQL calls to JBoss Data Virtualization.

Q: What does `getAdmin()` do? Why do I need it?

A: The **getAdmin()** method returns the current contextual connection object that was created when you connected with **connectAsAdmin()**. This object implements the interface org.teiid.adminapi.Admin . The AdminShell commands provided are wrappers around this API. Advanced users can use this API directly if the provided wrapper commands do not meet their needs.

Q: Is IDE support available for writing the scripts?

A: The AdminShell GUI tool is a light-weight IDE. Full IDE support is available for Groovy, but requires manual manipulation of the class path and script imports.

Q: Can I use AdminShell methods in other environments?

A: The AdminShell methods (including the named contextual connections, AdminAPI wrapper, and help system) have no direct dependencies on Groovy and can be used in other scripting languages.

To use the AdminShell methods in another language, simply import the static methods and Admin classes into your script. You will also need to ensure that the *EAP_HOME*/dataVirtualization/teiid-

adminshell-*VERSION*/lib/teiid-*VERSION*.jar and
***EAP_HOME*/dataVirtualization/teiid-adminshell-*VERSION*/lib/teiid-adminshell-*VERSION*.jar** are in your class path.

The following snippet shows import statements that would work in Java, BeanShell, Groovy and others:

```
import static org.teiid.adminshell.AdminShell.*;
import org.teiid.adminapi.*;
```

Q: **Is debugging support available?**

A: The Interactive AdminShell and GUI have built-in support for inspection of the current state; however, line based debugging is beyond the scope of this document.

1.5. JBOSS OPERATIONS NETWORK

1.5.1. Installing JBoss Agent Plug-in Packs

JBoss Operations Network has additional agent plug-ins to manage other JBoss products. Although these are JBoss ON resource plug-ins, they are included in separate packages and require a separate subscription to download them.

1. Download the plug-in JAR files from the Customer Support Portal.

   a. Navigate to https://access.redhat.com/jbossnetwork/.

   b. Select JBoss ON for *PRODUCT*.

   c. Select **Download**.

2. Extract the plug-in JAR files into the ***JON_HOME*/plugins** directory.

3. Have the JBoss ON server update its plug-ins. This can be done through the JBoss ON GUI or by restarting the server.

4. To load the plug-ins through the GUI:

   a. Open the **Administration** tab.

   b. In the **Configuration** area on the left, select the **Agent Plug-ins** link.

   c. At the bottom of the list of loaded agent plug-ins, click the **SCAN FOR UPDATES** button.

5. Have the agents reload their plug-ins to load the new plug-ins. This can be done from the agent's command prompt using the plugins command:

```
> plugins update
```
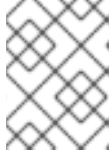
NOTE

This can also be done in the JBoss ON GUI by scheduling an update plugins operation for an agent or a group or agents. Alternatively, restart the agents.

PART II. USER MANAGEMENT

CHAPTER 2. USER ACCOUNTS

2.1. USER ACCOUNTS

The following account types are available for you to use in JBoss Data Virtualization:

JBoss EAP Management User

> You must have a JBoss EAP Management User in order to administer your JBoss Data Virtualization installation via the Management Console, Management CLI, JBoss Developer Studio, and Admin API.

JBoss Data Virtualization User

> JBoss Data Virtualization users can access the virtual databases (VDBs). You can define what permissions each of these users has to dictate what they can do with these databases.

Hierarchical Database User

> Hierarchical database users can access the provided hierarchical database. You can define what permissions each of these users has to dictate what they can do with this database.

> **NOTE**
>
> User/role names and details are independent of any other accounts such as operating system accounts. They relate only to JBoss EAP and JBoss Data Virtualization.

2.2. DATA ROLES

All authenticated users have access to a VDB. To restrict access, configure data roles. Set these in the Teiid Designer or the dynamic VDB's META-INF/vdb.xml file.

As part of the data role definition, you can map them to JAAS roles specified in **<mapped-role-name>** tags. You can set up JAAS roles using the web console.

How these JAAS roles are associated with users depends on which particular JAAS login module you use. For instance, the default **UsersRolesLoginModule** associates users with JAAS roles in plain text files.

For more information about data roles, see *Red Hat JBoss Data Virtualization Development Guide: Reference Material*.
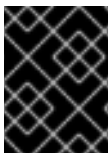
> **IMPORTANT**
>
> Do not use "admin" or "user" as JAAS role names as these are reserved specifically for Dashboard Builder permissions.

2.3. ADDING A JBOSS EAP MANAGEMENT USER

2.3.1. Add the User for the Management Interfaces

The following procedure documents how to create the initial administrative user, in the event such a user is not created by the chosen installation method. This initial administrative user can use the web-based Management Console and remote instances of the Management CLI to configure and administer JBoss

EAP 6 from remote systems.

Procedure 2.1. Create the Initial Administrative User for the Remote Management Interfaces

1. **Run the `add-user.sh` or `add-user.bat` script.**
   Change to the *EAP_HOME/bin/* directory. Invoke the appropriate script for your operating system.

   Red Hat Enterprise Linux

   ```
   [user@host bin]$ ./add-user.sh
   ```

   Microsoft Windows Server

   ```
   C:\bin>  add-user.bat
   ```

2. **Choose to add a Management user.**
   Press **ENTER** to select the default option **a** to add a Management user.

   This user is added to the `ManagementRealm` and is authorized to perform management operations using the web-based Management Console or command-line based Management CLI. The other choice, **b**, adds a user to the `ApplicationRealm`, and provides no particular permissions. That realm is provided for use with applications.

3. **Enter the desired username and password.**
   When prompted, enter the username and password. You will be prompted to confirm the password.

4. **Enter group information.**
   Add the group or groups to which the user belongs. If the user belongs to multiple groups, enter a comma-separated list. Leave it blank if you do not want the user to belong to any groups.

5. **Review the information and confirm.**
   You are prompted to confirm the information. If you are satisfied, type **yes**.

6. **Choose whether the user represents a remote JBoss EAP 6 server instance.**
   Besides administrators, the other type of user which occasionally needs to be added to JBoss EAP 6 in the `ManagementRealm` is a user representing another instance of JBoss EAP 6, which must be able to authenticate to join a cluster as a member. The next prompt allows you to designate your added user for this purpose. If you select **yes**, you will be given a hashed `secret` value, representing the user's password, which would need to be added to a different configuration file. For the purposes of this task, answer **no** to this question.

7. **Enter additional users.**
   You can enter additional users if desired, by repeating the procedure. You can also add them at any time on a running system. Instead of choosing the default security realm, you can add users to other realms to fine-tune their authorizations.

8. **Create users non-interactively.**
   You can create users non-interactively, by passing in each parameter at the command line. This approach is not recommended on shared systems, because the passwords will be visible in log and history files. The syntax for the command, using the management realm, is:

```
[user@host bin]$ ./add-user.sh username password
```

To use the application realm, use the **-a** parameter.

```
[user@host bin]$ ./add-user.sh -a username password
```

9. You can suppress the normal output of the add-user script by passing the **--silent** parameter. This applies only if the minimum parameters **username** and **password** have been specified. Error messages will still be shown.

Result

Any users you add are activated within the security realms you have specified. Users active within the **ManagementRealm** realm are able to manage JBoss EAP 6 from remote systems.

2.3.2. Default User Security Configuration

Introduction

All management interfaces in JBoss EAP 6 are secured by default. This security takes two different forms:

- Local interfaces are secured by a SASL contract between local clients and the server they connect to. This security mechanism is based on the client's ability to access the local filesystem. This is because access to the local filesystem would allow the client to add a user or otherwise change the configuration to thwart other security mechanisms. This adheres to the principle that if physical access to the filesystem is achieved, other security mechanisms are superfluous. The mechanism happens in four steps:

  NOTE

  HTTP access is considered to be remote, even if you connect to the localhost using HTTP.

  1. The client sends a message to the server which includes a request to authenticate with the local SASL mechanism.

  2. The server generates a one-time token, writes it to a unique file, and sends a message to the client with the full path of the file.

  3. The client reads the token from the file and sends it to the server, verifying that it has local access to the filesystem.

  4. The server verifies the token and then deletes the file.

- Remote clients, including local HTTP clients, use realm-based security. The default realm with the permissions to configure the JBoss EAP 6 instance remotely using the management interfaces is **ManagementRealm**. A script is provided which allows you to add users to this realm (or realms you create). For more information on adding users, see the *User Management* chapter of the *JBoss EAP 6 Administration and Configuration Guide*. For each user, the username and a hashed password are stored in a file.

  Managed domain

  ***EAP_HOME*/domain/configuration/mgmt-users.properties**

Standalone server

EAP_HOME/standalone/configuration/mgmt-users.properties

Even though the contents of the `mgmt-users.properties` are masked, the file must still be treated as a sensitive file. It is recommended that it be set to the file mode of `600`, which gives no access other than read and write access by the file owner.

2.3.3. Adding a JBoss Data Virtualization User

JBoss Data Virtualization's default configuration uses hashed files to define users and their roles. Follow these steps to add a new user for this default configuration.

1. Navigate to the bin directory.

2. Launch the ./add-user.sh script.

3. Selection option the "b) Application User (application-users.properties)" and add the username and password you desire.
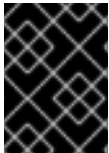
> **IMPORTANT**
>
> The "admin" and "user" role names are reserved specifically for Dashboard Builder permissions. The odata group is needed for OData access.

2.4. ADDING A MODESHAPE USER

2.4.1. Create a Modeshape Publishing User

To use Modeshape publishing, you need to ensure that at least one user has the connect and readwrite roles or the connect and admin roles.

By default, all the roles are assigned to all anonymous-roles users.

PART III. CONFIGURING THE PRODUCT

CHAPTER 3. VIRTUAL DATABASES

3.1. VIRTUAL DATABASES AND JBOSS DATA VIRTUALIZATION

JBoss Data Virtualization uses virtual databases (VDBs) to facilitate an enterprise data integration solution.

3.2. VIRTUAL DATABASE DEPLOYMENT

A virtual database (VDB) must be deployed before it can be accessed by client applications.

A VDB can be deployed by the following means:

File Deployment

File deployment is recommended for quick deployment during development, when the server is running locally on the developer's workstation.

Management Console

Deployment via the web-based Management Console is recommended for a simple way of deploying a VDB to a remote server.

Management Command Line Interface

Deployment via the EAP Management Command Line Interface (CLI) is another simple option for deployment.

AdminShell

Deployment via the AdminShell is recommended for more advanced deployments such as when you want to automate the deployment of artifacts in your environment.

Admin API

Deployment via the Admin API is recommended for more advanced deployments such as when you want to deploy a VDB from within other applications.
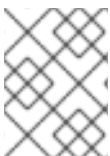
> **NOTE**
>
> A VDB can also be deployed within the Teiid Designer. See the *Red Hat JBoss Data Virtualization User Guide*.
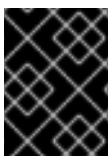
> **IMPORTANT**
>
> Removal of a VDB cleans up all of its resources automatically; however, existing sessions will not be terminated automatically.

> **WARNING**
>
> When deploying a VDB, overwriting a VDB with one of the same name will terminate all connections to the old VDB unless VDB versioning is used.
>
> Red Hat recommends the use of VDB versioning on production systems.

> **WARNING**
>
> You can locally name VDB artifacts as you wish, but the runtime names of your deployed VDB artifacts must have an extension of *.vdb for a zip file or *-vdb.xml for an xml file. Failure to name your file correctly will result in a deployment failure as the Teiid subsystem will not know how to handle it.

3.3. DEPLOY A VDB VIA FILE DEPLOYMENT

Prerequisites

- Red Hat JBoss Data Virtualization must be installed.

Procedure 3.1. Deploy a VDB via File Deployment

1. **Copy your VDB into the `deploy` directory**
   Copy your VDB file into the *`EAP_HOME`*`/standalone/deployments` directory.

2. **Create a marker file**
   Create an empty marker file of the same name with extension `.dodeploy` in the same directory. For example, if your VDB name is `enterprise.vdb`, then the marker file name must be `enterprise.vdb.dodeploy`.

> **NOTE**
>
> This only works in standalone mode. For domain mode, you must use one of the other available methods.

3.4. DEPLOY A VDB VIA MANAGEMENT CONSOLE

Prerequisites

- Red Hat JBoss Data Virtualization must be installed.

- The JBoss Enterprise Application Platform (EAP) server must be running.

- You must have a JBoss EAP Management User registered.

Procedure 3.2. Deploy a VDB via Management Console

1. **Launch the console in a Web browser**
   Open http://localhost:9990/console/ in a web browser.

2. **Authenticate to the console**
   Enter your JBoss EAP administrator username and password when prompted.

3. **Open the `Deployments` panel**
   In the `Runtime` view, select **Server → Manage Deployments**.

4. **Add the virtual database**

   a. Select the **Add** button.

   b. Select **`Choose File`** and choose the VDB file you want to deploy.

   c. Select **`Next`** to review the deployment names then select **`Save`**.

   d. Select **`En/Disable`** to enable the VDB.

3.5. DEPLOY A VDB VIA CLI

Prerequisites

- Red Hat JBoss Data Virtualization must be installed.

- The JBoss Enterprise Application Platform (EAP) server must be running.

Procedure 3.3. Deploy a VDB via CLI

1. **Open the command line interface**
   Run the *`EAP_HOME/bin/jboss-cli.sh`* command.

2. **Connect to the server**
   Run the **`connect`** command.

3. **Deploy the virtual database**
   If in standalone mode, run **`deploy PATH/DATABASE.vdb`**.

   If in domain mode, run **`deploy -all-server-groups PATH/DATABASE.vdb`**.

> **NOTE**
>
> In domain mode, you need to select a particular "server-group" or all available server groups are deployment options.

3.6. DEPLOY A VDB VIA ADMINSHELL

Prerequisites

- Red Hat JBoss Data Virtualization must be installed.

- The JBoss Enterprise Application Platform (EAP) server must be running.

Procedure 3.4. Deploy a VDB via AdminShell

1. **Open the interactive AdminShell interface**
   Run the **./adminshell.sh** command.

2. **Open a connection**
   Within the interactive AdminShell, run the **connectAsAdmin()** command.

3. **Deploy your virtual database**
   Run the **deploy("*PATH/DATABASE*.vdb")** command.

4. **Close the connection**
   Run the **disconnect()** command.

5. **Exit the interactive shell**

   a. Enter the **exit** command to leave the interactive shell.

> **NOTE**
>
> In domain mode, when deploying using AdminShell, the VDB is deployed to all available servers.
>
> A VDB can also be deployed via the AdminShell console or using a script via the non-interactive AdminShell. For more information on using these, refer to topics on the Administration Shell.

3.7. DEPLOY A VDB VIA ADMIN API

You can deploy a VDB using the **deploy** method provided by the Admin interface within the Admin API package (**org.teiid.adminapi**).

Javadocs for Red Hat JBoss Data Virtualization can be found on the Red Hat Customer Portal.

> **NOTE**
>
> In domain mode, when deploying using the Admin API, the VDB is deployed to all available servers.

3.8. VDB DEPENDENCIES

When deploying a virtual database (VDB) in JBoss Data Virtualization, you also have to provide dependency libraries and configuration settings for accessing the physical data sources used by your VDB. (You can identify all dependent physical data sources by looking in **META-INF/vdb.xml** within the *EAP_HOME/MODE***/deployments/*DATABASE*.vdb** file.)

For example, if you are trying to integrate Oracle and file sources in your VDB, then you are responsible for providing both the JDBC driver for the Oracle source, and any necessary documents and configuration files that are needed by the file translator.

Data source instances may be shared between multiple VDBs and applications. Consider sharing connections to sources that are heavy-weight and resource-constrained.

Once you have deployed the VDB and its dependencies, client applications can connect using the **JDBC** API. If there are any errors in the deployment, the connection attempt will fail and a message will be logged. Use the Management Console (or check the log files) to identify any errors and correct them so you can proceed. See *Red Hat JBoss Data Virtualization Development Guide: Server Development* for information on how to use JDBC to connect to your VDB.

> **WARNING**
>
> Some data source configuration files may contain passwords or other sensitive information. For instructions on how to avoid storing passwords in plaintext, refer to the JBoss Enterprise Application Platform *Security Guide*.

3.9. DATA SOURCE DEPLOYMENT

3.9.1. Accumulo Data Sources

Accumulo data sources use a Data Virtualization-specific JCA connector that is deployed into JBoss EAP during installation. There are many ways to create a Accumulo data source, using CLI, AdminShell, admin-console, and so forth. The example below uses the CLI tool.

Execute the following command using the CLI once you connected to the Server. Ensure you provide the correct URL and user credentials. Add any additional properties required by the connector by duplicating the "connection-definitions" command below. Edit the JNDI name to match the JNDI name you used in VDB:

```
batch
/subsystem=resource-adapters/resource-adapter=accumulo/connection-
definitions=teiid:add(jndi-name=java:/accumulo-ds, class-
name=org.teiid.resource.adapter.accumulo.AccumuloManagedConnectionFactory,
enabled=true, use-java-context=true)
/subsystem=resource-adapters/resource-adapter=accumulo/connection-
definitions=teiid/config-
properties=ZooKeeperServerList:add(value=localhost:2181)
/subsystem=resource-adapters/resource-adapter=accumulo/connection-
definitions=teiid/config-properties=Username:add(value=user)
/subsystem=resource-adapters/resource-adapter=accumulo/connection-
definitions=teiid/config-properties=Password:add(value=password)
/subsystem=resource-adapters/resource-adapter=accumulo/connection-
definitions=teiid/config-properties=InstanceName:add(value=instancename)
/subsystem=resource-adapters/resource-adapter=accumulo/connection-
definitions=teiid/config-properties=Roles:add(value=public)
/subsystem=resource-adapters/resource-adapter=accumulo:activate
runbatch
```

These are the properties defined in the RAR file:

Table 3.1. Properties

Property	Description	Required?	Default?
ZooKeeperServerList	A comma-separated list of zoo keeper server locations. Each location can contain an optional port, of the format host:port	True.	None.
Username	Connection User's Name	True.	None.
Password	Connection User's password	True.	None.
InstanceName	Accumulo instance name	True.	None.
Roles	optional visibility for user, supply multiple with comma separated	False.	None.

To find out all the properties that are supported by this Accumulo Connector execute this command in the CLI:

```
/subsystem=teiid:read-rar-description(rar-name=accumulo)
```

3.9.2. Cassandra Data Sources

Cassandra's data sources use a Data Virtualization-specific JCA connector that is deployed on EAP during installation. There are many ways to create a Cassandra data source, using CLI, AdminShell, and admin-console, and so forth. The example below uses the CLI tool.

Execute the following command using the CLI once you connected to the Server. Make sure you provide the correct URL and user credentials. Add any additional properties required by the connector by duplicating the "connection-definitions" command below. Edit the JNDI name to match the JNDI name you used in VDB:

```
batch
/subsystem=resource-adapters/resource-adapter=cassandra/connection-
definitions=cassandraDS:add(jndi-name=java:/cassandraDS, class-
name=org.teiid.resource.adapter.cassandra.CassandraManagedConnectionFactor
y, enabled=true, use-java-context=true)
/subsystem=resource-adapters/resource-adapter=cassandra/connection-
definitions=cassandraDS/config-properties=Address:add(value=127.0.0.1)
/subsystem=resource-adapters/resource-adapter=cassandra/connection-
definitions=cassandraDS/config-properties=Keyspace:add(value=my-keyspace)
/subsystem=resource-adapters/resource-adapter=cassandra:activate
runbatch
```

To find out all the properties that are supported by this Cassandra Connector execute the following command in the CLI:

```
/subsystem=teiid:read-rar-description(rar-name=cassandra)
```

3.9.3. File Data Source

File data sources use a Teiid specific JCA connector that is deployed into EAP during installation. There are many ways to create the file data source, using CLI, AdminShell, admin-console, and so forth. The example below uses the CLI tool.

Execute following command using the CLI once you connected to the Server. Make sure you provide the correct directory name and other properties below. Add any additional properties required by the connector by duplicating the "connection-definitions" command below. Edit the JNDI name to match the JNDI name you used in the VDB:

```
batch
/subsystem=resource-adapters/resource-adapter=file/connection-
definitions=fileDS:add(jndi-name=java:/fileDS, class-
name=org.teiid.resource.adapter.file.FileManagedConnectionFactory,
enabled=true, use-java-context=true)
/subsystem=resource-adapters/resource-adapter=file/connection-
definitions=fileDS/config-
properties=Parentdirectory:add(value=/home/rareddy/testing/)
/subsystem=resource-adapters/resource-adapter=file/connection-
definitions=fileDS/config-properties=AllowParentPaths:add(value=true)
/subsystem=resource-adapters/resource-adapter=file:activate
runbatch
```

To find out all the properties that are supported by this File Connector execute the following command in the CLI:

```
/subsystem=teiid:read-rar-description(rar-name=file)
```

3.9.4. Google Spreadsheet Data Sources

The Google JCA connector is named teiid-connector-google.rar. The examples include a sample google.xml file. The JCA connector has number of config-properties to drive authentication. The JCA connector connects to exactly one spreadsheet with each sheet exposed as a table. Authentication to your google account may be done using OAuth, which requires a refresh token:

Table 3.2. Configuration Properties

Property	Description
AuthMethod	Method to access Google. This property can be set to OAuth2. If you do so, it is necessary to provide RefreshToken.
SpreadsheetName	Required property with name of the Spreadsheet that is datasource for this connector.

Property	Description
BatchSize	Maximum number of rows that can be fetched at a time. Defaults to 4096.

To obtain an OAuth Refresh Token, go to Google's authentication site:

```
https://accounts.google.com/o/oauth2/auth?
scope=https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdrive+https%3A%2F%2Fspread
sheets.google.com%2Ffeeds&redirect_uri=urn:ietf:wg:oauth:2.0:oob&response_
type=code&client_id=217138521084.apps.googleusercontent.com
```

Next, copy the authorization code into the following POST request and run it on command line. (The refresh token will be in the response.)

```
curl \--data-urlencode code=<AUTH_CODE> \
--data-urlencode client_id=217138521084.apps.googleusercontent.com \
--data-urlencode client_secret=gXQ6-lOkEjE1lVcz7giB4Poy \
--data-urlencode redirect_uri=urn:ietf:wg:oauth:2.0:oob \
--data-urlencode grant_type=authorization_code
https://accounts.google.com/o/oauth2/token
```

3.9.5. Red Hat JBoss Data Grid Hot Rod Data Sources

Red Hat JBoss Data Grid's Hot Rod data sources use a Red Hat JBoss Data Virtualization-specific JCA connector that is deployed upon installation. Detailed below are the different configuration options available to you depending on the version of Red Hat JBoss Data Grid you are using.

3.9.5.1. Red Hat JBoss Data Grid 6.x Configuration

You can configure the JCA connector to support accessing a remote JDG cache using the Hot Rod client.

There are two options for how the JDG schema can be configured in the connector: protobuf annotations or protobuf (.proto) file with marshaller(s). The following are the requirements:

1. Minimum requirement: JDG 6.2 - this requires you provide a protobuf definition file and pojo marshaller for the pojo to be cached

2. Minimum requirement: JDG 6.6 - this can be used when the pojo has protobuf annotations which trigger the creation of the protobuf definition and pojo marshaller by JDG

Regardless of the option you choose, the following property is required as it provides the mapping to the JDG cache and pojo that will be accessed.

Table 3.3. Required Cache Properties

Property Name	Property Template	Description

Property Name	Property Template	Description
CacheTypeMap	cacheName:className[;pkFieldName[:cacheKeyJavaType]]	For the indicated cache, map the root Java Object class name. Optionally, but required for updates, identify which class attribute is the primary key to the cache. Identify primary key java type when different than class attribute type.

The pojo class is the object that will be used to store the data in the cache. It should be built as follows:

- If the pojo is to be used to define the schema, then you should use the protobuf annotations. See JDG Protobuf Annotations in the Custom Fields Indexing with Protobuf section of the Red Hat JBoss Data Grid 7.1 Developer Guide.

- If the protobuf definition and mashaller(s) are to be used, then these should also be packaged in the jar (or a separate jar that is included in the classpath).

- The class should be packaged into a jar so that it can be deployed as a module.

To configure the resource-adapter to use the pojo, do the following:

1. Deploy the pojo jar as a module in the Red Hat JBoss EAP server.

2. Define the "lib" property in the -vdb.xml and assign the correct module name. You can do so using this template:

```
<property name ="lib" value ="{pojo_module_name}"></property>
```

One of the following properties is required for defining how the RemoteCacheManager will be created and accessed:

Table 3.4. Optional Cache Properties

Property Name	Required?	Property Template	Description
CacheJndiName	No	NA	This is the JNDI name to find the CacheContainer.
RemoteServerList	No	host:port\[;host:port….\]	Specify the host and ports that will be clustered together to access the caches.
HotRodClientProperties File	No	NA	The HotRod properties file for configuring a connection to a remote cache.

The following additional properties are required when the protobuf definition file (.proto) and the pojo marshaller(s) are being used to configure the JDG schema:

Table 3.5. Properties when using protobuf definition file (.proto) and pojo marshaller(s)

Property Name	Required?	Property Template	Description
ProtobufDefinitionFile	Yes	NA	Path to the Google Protobuf file that is packaged in a jar (here is an example: **/quickstart/addr essbook.proto**)
MessageMarshallers	Yes	marshaller \ [,marshaller,..\]	Contains class names mapped to its respective message marshaller (class:marshaller,\ [class:marshaller,..\]) that are registered for serialization.
MessageDescriptor	Yes	NA	Message descriptor class name for the root object in cache.
Module	No	NA	JBoss EAP server module where the jar with pojos is located.

These are the additional properties that need to be configured if you are using the Remote Cache for external materialization:

Table 3.6. Remote Cache Properties for External Materialization

Property Name	Required?	Description
StagingCacheName	Yes	Cache name for the staging cache used in materialization
AliasCacheName	Yes	Cache name for the alias cache used in tracking aliases of the caches used in materialization.

There are many ways to create the data source, including using CLI, AdminShell, admin-console and so forth. The first example shows a resource-adapter's XML code. This code is used to connect to the JDG remote-query quick start:

```
<resource-adapter id="infinispanRemQS">
  <module slot="main" id="org.jboss.teiid.resource-
adapter.infinispan.hotrod"/>
  <connection-definitions>
```

```
    <connection-definition class-
name="org.teiid.resource.adapter.infinispan.hotrod.InfinispanManagedConnec
tionFactory" jndi-name="java:/infinispanRemote" enabled="true" use-java-
context="true" pool-name="infinispanDS">
        <config-property name="CacheTypeMap">

addressbook:org.jboss.as.quickstarts.datagrid.hotrod.query.domain.Person;i
d
        </config-property>
        <config-property name="Module">
         com.client.quickstart.pojos
        </config-property>
        <config-property name="RemoteServerList">
         127.0.0.1:11322
        </config-property>
      </connection-definition>
    </connection-definitions>
  </resource-adapter>
```

This example show you how to configure it for external materialization:

```
<resource-adapter id="infinispanRemQSDSL">
  <module slot="main" id="org.jboss.teiid.resource-
adapter.infinispan.hotrod"/>
  <connection-definitions>
   <connection-definition class-
name="org.teiid.resource.adapter.infinispan.hotrod.InfinispanManagedConnec
tionFactory" jndi-name="java:/infinispanRemoteDSL" enabled="true" use-
java-context="true" pool-name="infinispanRemoteDSL">
      <config-property name="CacheTypeMap">

addressbook_indexed:org.jboss.as.quickstarts.datagrid.hotrod.query.domain.
Person;id
      </config-property>
      <config-property name="Module">
       com.client.quickstart.addressbook.pojos
      </config-property>
      <config-property name="RemoteServerList">
       127.0.0.1:11322
      </config-property>
    </connection-definition>
  </connection-definitions>
</resource-adapter>
```

3.9.5.2. Red Hat JBoss Data Grid 7.x Configuration

These are the properties defined in the RAR file:

Table 3.7. Properties

Property Name	Required?	Description

Property Name	Required?	Description
RemoteServerList	Yes	A comma separated list of server locations. Each location can contain an optional port, of the format host:port.
UserName	No	If remote server is secured, this property used as username to log in.
Password	No	If remote server is secured, this property used as password to log in.
SaslMechanism	No	If remote server is secured, this property defines the type of security. Allowed values are "CRAM-MD5", "DIGEST-MD5", "PLAIN", "EXTERNAL". "EXTERNAL" is when certificate based security at use, all others use username/password.
AuthenticationRealm	No	Realm to use for authentication.
AuthenticationServerName	No	Infinispan server name where the Authentication is handled.
TrustStoreFileName	No	When "EXTERNAL" SaslMechnism used, use this property to define truststore. Alternatively JAVA system property "javax.net.ssl.trustStore" can also be defined instead.
TrustStorePassword	No	When "EXTERNAL" SaslMechnism used, use this property to define truststore password. Alternatively JAVA system property "javax.net.ssl.trustStorePassword" can also be defined instead.
KeyStoreFileName	No	When "EXTERNAL" SaslMechnism used, use this property to define keystore. Alternatively JAVA system property "javax.net.ssl.keyStore" can also be defined instead.

Property Name	Required?	Description
KeyStorePassword	No	When "EXTERNAL" SaslMechnism used, use this property to define keystore password. Alternatively JAVA system property "javax.net.ssl.keyStorePassword" can also be defined instead.

The example below shows a resource-adapter's XML code. This code is used to connect to the JDG StockMat quick start:

```
<resource-adapter id="infinispanRemQS">
  <module slot="main" id="org.jboss.teiid.resource-
adapter.infinispan.hotrod"/>
  <connection-definitions>
   <connection-definition class-
name="org.teiid.resource.adapter.infinispan.hotrod.InfinispanManagedConnec
tionFactory" jndi-name="java:/infinispanStockMat" enabled="true" use-java-
context="true" pool-name="infinispanStockMat">
    <config-property name="RemoteServerList">
     127.0.0.1:11322
    </config-property>
   </connection-definition>
  </connection-definitions>
</resource-adapter>
```

The example below shows how to use the CLI tool to create the same JDG Hot Rod data source as above. You can use this in both standalone and domain modes. Execute the following commands using the CLI once you are connected to the server. Make sure you provide the correct URL and user credentials. Add any additional properties required by the connector by duplicating the "connection-definitions" command below. Edit the JNDI name to match the JNDI name you used in VDB.

```
batch
/subsystem=resource-adapters/resource-
adapter=ispnStockMat:add(module=org.jboss.teiid.resource-
adapter.infinispan.hotrod)
/subsystem=resource-adapters/resource-adapter=ispnStockMat/connection-
definitions=infinispanStockMat:add(jndi-name="java:/infinispanStockMat",
class-
name=org.teiid.resource.adapter.infinispan.hotrod.InfinispanManagedConnect
ionFactory, enabled=true, use-java-context=true)
/subsystem=resource-adapters/resource-adapter=ispnStockMat/connection-
definitions=infinispanStockMat/config-
properties=RemoteServerList:add(value="127.0.0.1:11322")
/subsystem=resource-adapters/resource-adapter=ispnStockMat:activate
run-batch
```

To discover all the properties that are supported by this JDG Hot Rod Connector, first create a resource-adapter and then execute this command in the CLI, replacing the rar-name with the name of that resource-adapter:

```
/subsystem=teiid:read-rar-description(rar-name=jdghotrod)
```

3.9.6. JDBC Data Sources

3.9.6.1. JDBC Data Sources

Here is an example highlighting configuring an Oracle data source. The process is nearly identical regardless of the database vendor. Typically the JDBC jar and the configuration like connection URL and user credentials change.

There are configuration templates for all the data sources in the **EAP_HOME/docs/teiid/datasources** directory. A complete description how a data source can be added into JBoss EAP is also described here. Here are two different ways to create a datasource:

Firstly, deploy the required JDBC jar file. For example, if you are trying to create a Oracle data source, first you need to deploy the **ojdbc6.jar** file: **deploy /path/to/ojdbc6.jar**

> **NOTE**
>
> If JBoss EAP is running in standalone mode, you can also manually copy this **ojdbc6.jar** to the **EAP_HOME/standalone/deployments** directory, to automatically deploy without using the CLI tool.

Now create a data source using this driver. There are many ways to create the datasource using CLI, AdminShell, and so forth. The example below uses the CLI tool.

Execute the following command using the CLI once you connected to the Server. Make sure you provide the correct URL and user credentials and edit the JNDI name to match the JNDI name you used in the VDB:

```
/subsystem=datasources/data-source=oracle-ds:add(jndi-name=java:/OracleDS,
driver-name=ojdbc6.jar, connection-url=jdbc:oracle:thin:
{host}:1521:orcl,user-name={user}, password={password})
/subsystem=datasources/data-source=oracle-ds:enable
```

3.9.6.2. Configuring JDBC Data Sources

For JBoss Data Virtualization to connect to JDBC data sources, you must configure the data source in the JBoss EAP instance with the following two steps:

1. Install (or deploy) the JDBC JAR driver file.

2. Create (or configure) it as a data source in JBoss EAP.

> **NOTE**
>
> See the *Red Hat JBoss Enterprise Application Platform Administration and Configuration Guide* for more comprehensive information about configuring data sources.
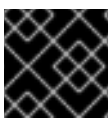
> **IMPORTANT**
>
> The jdbc driver to support JDK 1.4/1.5 has been deprecated.

3.9.6.3. Example Configuration

Example configuration, as it would appear in the server configuration file, can be found in the
***EAP_HOME*/docs/teiid/datasources/** directory.

> **NOTE**
>
> This configuration will need to be adjusted with file paths and properties appropriate to
> your installation. The JNDI name must be the same JNDI name used in the VDB.

3.9.6.4. Install a JDBC Driver with the Management CLI

Procedure 3.5. Install a JDBC Driver with the Management CLI

1. Start the Management CLI: **EAP_HOME/bin/jboss-cli.sh**

2. Enter the **connect** command.

3. Enter the **deploy *PATH/FILE*.jar** command: **deploy ojdbc6.jar**

4. Enter the **quit** command.

3.9.6.5. Install a JDBC Driver with the Management Console

Summary

Before your application can connect to a JDBC datasource, your datasource vendor's JDBC drivers need
to be installed in a location where JBoss EAP 6 can use them. JBoss EAP 6 allows you to deploy these
drivers like any other deployment. This means that you can deploy them across multiple servers in a
server group, if you use a managed domain.

Prerequisites

Before performing this task, you need to meet the following prerequisites:

- Download the JDBC driver from your database vendor.

> **NOTE**
>
> Any JDBC 4-compliant driver is automatically recognized and installed in the system by
> name and version. A JDBC JAR is identified using the Java service provider mechanism.
> Such JARs contain the META-INF/services/java.sql.Driver text, which contains the name
> of the Driver classes in that JAR.

Procedure 3.6. Modify the JDBC Driver JAR

If the JDBC driver JAR is not JDBC 4-compliant, it can be made deployable using the following method.

1. Change to, or create, an empty temporary directory.

2. Create a META-INF subdirectory.

3. Create a META-INF/services subdirectory.

4. Create a META-INF/services/java.sql.Driver file, which contains one line indicating the fully-qualified class name of the JDBC driver.

5. Use the JAR command-line tool to update the JAR like this:

```
jar \-uf jdbc-driver.jar META-INF/services/java.sql.Driver
```

6. Section 1.2.2, "Log in to the Management Console"

7. If you use a managed domain, deploy the JAR file to a server group. Otherwise, deploy it to your server. See the Deploy with the Management Console section of the Administration and Configuration Guide for JBoss EAP 6 for more information.

Result:

The JDBC driver is deployed, and is available for your applications to use.

3.9.6.6. Create a Non-XA Datasource with the Management Interfaces

Summary

This topic covers the steps required to create a non-XA datasource, using either the Management Console or the Management CLI.

Prerequisites

- The JBoss EAP 6 server must be running.

NOTE

Prior to version 10.2 of the Oracle datasource, the <no-tx-separate-pools/> parameter was required, as mixing non-transactional and transactional connections would result in an error. This parameter may no longer be required for certain applications.

NOTE

To prevent issues such as duplication of driver listing, selected driver not available in a profile, or driver not displayed if a server for the profile is not running, in JBoss EAP 6.4 onwards, only JDBC drivers that are installed as modules and correctly referenced from profiles are detectable while creating a datasource using the Management Console in domain mode.

Procedure 3.7. Create a Datasource using either the Management CLI or the Management Console

- ○ **Management CLI**

  a. Launch the CLI tool and connect to your server.

  b. Run the following Management CLI command to create a non-XA datasource, configuring the variables as appropriate:

NOTE

The value for *DRIVER_NAME* depends on the number of classes listed in the **/META-INF/services/java.sql.Driver** file located in the JDBC driver JAR. If there is only one class, the value is the name of the JAR. If there are multiple classes, the value is the name of the JAR + driverClassName + "_" + majorVersion +"_" + minorVersion. Failure to do so will result in the following error being logged:

> JBAS014775:   New missing/unsatisfied dependencies

For example, the *DRIVER_NAME* value required for the MySQL 5.1.31 driver, is **mysql-connector-java-5.1.31-bin.jarcom.mysql.jdbc.Driver_5_1**.

```
data-source add --name=DATASOURCE_NAME --jndi-name=JNDI_NAME --driver-name=DRIVER_NAME  --connection-url=CONNECTION_URL
```

c. Enable the datasource:

```
data-source enable --name=DATASOURCE_NAME
```

○ **Management Console**

a. Login to the Management Console.

b. **Navigate to the `Datasources` panel in the Management Console**

i. Select the **`Configuration`** tab from the top of the console.

ii. For Domain mode only, select a profile from the drop-down box in the top left.

iii. Expand the **Subsystems** menu on the left of the console, then expand the **Connector** menu.

iv. Select **Datasources** from the menu on the left of the console.

c. **Create a new datasource**

i. Click **Add** at the top of the **`Datasources`** panel.

ii. Enter the new datasource attributes in the **`Create Datasource`** wizard and proceed with the **`Next`** button.

iii. Enter the JDBC driver details in the **`Create Datasource`** wizard and click **`Next`** to continue.

iv. Enter the connection settings in the **`Create Datasource`** wizard.

v. Click the **`Test Connection`** button to test the connection to the datasource and verify the settings are correct.

vi. Click **Done** to finish

Result

The non-XA datasource has been added to the server. It is now visible in either the **standalone.xml** or **domain.xml** file, as well as the management interfaces.

3.9.6.7. Create an XA Datasource with the Management Interfaces

Summary

This topic covers the steps required to create an XA datasource, using either the Management Console or the Management CLI.

> **NOTE**
>
> Prior to version 10.2 of the Oracle datasource, the <no-tx-separate-pools/> parameter was required, as mixing non-transactional and transactional connections would result in an error. This parameter may no longer be required for certain applications.

Procedure 3.8. Create an XA Datasource, Using Either the Management CLI or the Management Console

- ○ **Management CLI**

    a. Section 1.3.1, "Launch the Management CLI".

    b. Run the following Management CLI command to create an XA datasource, configuring the variables as appropriate:

    > **NOTE**
    >
    > The value for *DRIVER_NAME* depends on the number of classes listed in the **/META-INF/services/java.sql.Driver** file located in the JDBC driver JAR. If there is only one class, the value is the name of the JAR. If there are multiple classes, the value is the name of the JAR + driverClassName + "_" + majorVersion +"_" + minorVersion. Failure to do so will result in the following error being logged:
    >
    > ```
    > JBAS014775:    New missing/unsatisfied dependencies
    > ```
    >
    > For example, the *DRIVER_NAME* value required for the MySQL 5.1.31 driver, is **mysql-connector-java-5.1.31-bin.jarcom.mysql.jdbc.Driver_5_1**.

    ```
    xa-data-source add --name=XA_DATASOURCE_NAME --jndi-
    name=JNDI_NAME --driver-name=DRIVER_NAME --xa-datasource-
    class=XA_DATASOURCE_CLASS
    ```

    c. **Configure the XA datasource properties**

        i. **Set the server name**
           Run the following command to configure the server name for the host:

           ```
           /subsystem=datasources/xa-data-
           source=XA_DATASOURCE_NAME/xa-datasource-
           properties=ServerName:add(value=HOSTNAME)
           ```

ii. **Set the database name**
Run the following command to configure the database name:

```
/subsystem=datasources/xa-data-
source=XA_DATASOURCE_NAME/xa-datasource-
properties=DatabaseName:add(value=DATABASE_NAME)
```

d. Enable the datasource:

```
xa-data-source enable --name=XA_DATASOURCE_NAME
```

○ **Management Console**

a. Section 1.2.2, "Log in to the Management Console".

b. **Navigate to the `Datasources` panel in the Management Console**

i. Select the **`Configuration`** tab from the top of the console.

ii. For Domain mode only, select a profile from the drop-down box at the top left.

iii. Expand the **Subsystems** menu on the left of the console, then expand the **Connector** menu.

iv. Select **Datasources**.

c. Select the **`XA Datasource`** tab.

d. **Create a new XA datasource**

i. Click **Add**.

ii. Enter the new XA datasource attributes in the **`Create XA Datasource`** wizard and click **Next**.

iii. Enter the JDBC driver details in the **`Create XA Datasource`** wizard and click **Next**.

iv. Enter the XA properties and click **Next**.

v. Enter the connection settings in the **`Create XA Datasource`** wizard.

vi. Click the **`Test Connection`** button to test the connection to the XA datasource and verify the settings are correct.

vii. Click **Done** to finish

Result

The XA datasource has been added to the server. It is now visible in either the **`standalone.xml`** or **`domain.xml`** file, as well as the management interfaces.

3.9.6.8. Datasource Parameters

Table 3.8. Datasource parameters common to non-XA and XA datasources

Parameter	Description
jndi-name	The unique JNDI name for the datasource.
pool-name	The name of the management pool for the datasource.
enabled	Whether or not the datasource is enabled.
use-java-context	Whether to bind the datasource to global JNDI.
spy	Enable **spy** functionality on the JDBC layer. This logs all JDBC traffic to the datasource. Note that the logging category `jboss.jdbc.spy` must also be set to the log level **DEBUG** in the logging subsystem.
use-ccm	Enable the cached connection manager.
new-connection-sql	A SQL statement which executes when the connection is added to the connection pool.
transaction-isolation	One of the following: • TRANSACTION_READ_UNCOMMITTED • TRANSACTION_READ_COMMITTED • TRANSACTION_REPEATABLE_READ • TRANSACTION_SERIALIZABLE • TRANSACTION_NONE
url-selector-strategy-class-name	A class that implements interface `org.jboss.jca.adapters.jdbc.URLSelectorStrategy`.
security	Contains child elements which are security settings. See Table 3.13, "Security parameters".
validation	Contains child elements which are validation settings. See Table 3.14, "Validation parameters".
timeout	Contains child elements which are timeout settings. See Table 3.15, "Timeout parameters".
statement	Contains child elements which are statement settings. See Table 3.16, "Statement parameters".

Table 3.9. Non-XA datasource parameters

Parameter	Description
jta	Enable JTA integration for non-XA datasources. Does not apply to XA datasources.
connection-url	The JDBC driver connection URL.
driver-class	The fully-qualified name of the JDBC driver class.
connection-property	Arbitrary connection properties passed to the method **Driver.connect(url,props)**. Each connection-property specifies a string name/value pair. The property name comes from the name, and the value comes from the element content.
pool	Contains child elements which are pooling settings. See Table 3.11, "Pool parameters common to non-XA and XA datasources".
url-delimiter	The delimiter for URLs in a connection-url for High Availability (HA) clustered databases.

Table 3.10. XA datasource parameters

Parameter	Description
xa-datasource-property	A property to assign to implementation class **XADataSource**. Specified by *name=value*. If a setter method exists, in the format **set*Name***, the property is set by calling a setter method in the format of **set*Name*(*value*)**.
xa-datasource-class	The fully-qualified name of the implementation class **javax.sql.XADataSource**.
driver	A unique reference to the class loader module which contains the JDBC driver. The accepted format is *driverName#majorVersion.minorVersion*.
xa-pool	Contains child elements which are pooling settings. See Table 3.11, "Pool parameters common to non-XA and XA datasources" and Table 3.12, "XA pool parameters".
recovery	Contains child elements which are recovery settings. See Table 3.17, "Recovery parameters".

Table 3.11. Pool parameters common to non-XA and XA datasources

Parameter	Description
min-pool-size	The minimum number of connections a pool holds.

Parameter	Description
max-pool-size	The maximum number of connections a pool can hold.
prefill	Whether to try to prefill the connection pool. An empty element denotes a **true** value. The default is **false**.
use-strict-min	Whether the idle connection scan should strictly stop marking for closure of any further connections, once the **min-pool-size** has been reached. The default value is **false**.
flush-strategy	Whether the pool is flushed in the case of an error. Valid values are: ● FailingConnectionOnly ● IdleConnections ● EntirePool The default is **FailingConnectionOnly**.
allow-multiple-users	Specifies if multiple users will access the datasource through the getConnection(user, password) method, and whether the internal pool type accounts for this behavior.

Table 3.12. XA pool parameters

Parameter	Description
is-same-rm-override	Whether the **javax.transaction.xa.XAResource.isSameRM(XAResource)** class returns **true** or **false**.
interleaving	Whether to enable interleaving for XA connection factories.
no-tx-separate-pools	Whether to create separate sub-pools for each context. This is required for Oracle datasources, which do not allow XA connections to be used both inside and outside of a JTA transaction. Using this option will cause your total pool size to be twice **max-pool-size**, because two actual pools will be created.
pad-xid	Whether to pad the Xid.

Parameter	Description
wrap-xa-resource	Whether to wrap the XAResource in an `org.jboss.tm.XAResourceWrapper` instance.

Table 3.13. Security parameters

Parameter	Description
user-name	The username to use to create a new connection.
password	The password to use to create a new connection.
security-domain	Contains the name of a JAAS security-manager which handles authentication. This name correlates to the application-policy/name attribute of the JAAS login configuration.
reauth-plugin	Defines a reauthentication plug-in to use to reauthenticate physical connections.

Table 3.14. Validation parameters

Parameter	Description
valid-connection-checker	An implementation of interface `org.jboss.jca.adaptors.jdbc.ValidCon nectionChecker` which provides a `SQLException.isValidConnection(Conne ction e)` method to validate a connection. An exception means the connection is destroyed. This overrides the parameter `check-valid-connection-sql` if it is present.
check-valid-connection-sql	An SQL statement to check validity of a pool connection. This may be called when a managed connection is taken from a pool for use.
validate-on-match	Indicates whether connection level validation is performed when a connection factory attempts to match a managed connection for a given set. Specifying "true" for `validate-on-match` is typically not done in conjunction with specifying "true" for `background-validation`. `Validate-on-match` is needed when a client must have a connection validated prior to use. This parameter is false by default.

Parameter	Description
background-validation	Specifies that connections are validated on a background thread. Background validation is a performance optimization when not used with **validate-on-match**. If **validate-on-match** is true, using **background-validation** could result in redundant checks. Background validation does leave open the opportunity for a bad connection to be given to the client for use (a connection goes bad between the time of the validation scan and prior to being handed to the client), so the client application must account for this possibility.
background-validation-millis	The amount of time, in milliseconds, that background validation runs.
use-fast-fail	If true, fail a connection allocation on the first attempt, if the connection is invalid. Defaults to **false**.
stale-connection-checker	An instance of **org.jboss.jca.adapters.jdbc.StaleConnectionChecker** which provides a Boolean **isStaleConnection(SQLException e)** method. If this method returns **true**, the exception is wrapped in an **org.jboss.jca.adapters.jdbc.StaleConnectionException**, which is a subclass of **SQLException**.
exception-sorter	An instance of **org.jboss.jca.adapters.jdbc.ExceptionSorter** which provides a Boolean **isExceptionFatal(SQLException e)** method. This method validates whether an exception is broadcast to all instances of **javax.resource.spi.ConnectionEventListener** as a **connectionErrorOccurred** message.

Table 3.15. Timeout parameters

Parameter	Description
use-try-lock	Uses **tryLock()** instead of **lock()**. This attempts to obtain the lock for the configured number of seconds, before timing out, rather than failing immediately if the lock is unavailable. Defaults to **60** seconds. As an example, to set a timeout of 5 minutes, set **<use-try-lock>**300**</use-try-lock>**.

Parameter	Description
blocking-timeout-millis	The maximum time, in milliseconds, to block while waiting for a connection. After this time is exceeded, an exception is thrown. This blocks only while waiting for a permit for a connection, and does not throw an exception if creating a new connection takes a long time. Defaults to 30000, which is 30 seconds.
idle-timeout-minutes	The maximum time, in minutes, before an idle connection is closed. The actual maximum time depends upon the idleRemover scan time, which is half of the smallest **idle-timeout-minutes** of any pool.
set-tx-query-timeout	Whether to set the query timeout based on the time remaining until transaction timeout. Any configured query timeout is used if no transaction exists. Defaults to **false**.
query-timeout	Timeout for queries, in seconds. The default is no timeout.
allocation-retry	The number of times to retry allocating a connection before throwing an exception. The default is **0**, so an exception is thrown upon the first failure.
allocation-retry-wait-millis	How long, in milliseconds, to wait before retrying to allocate a connection. The default is 5000, which is 5 seconds.
xa-resource-timeout	If non-zero, this value is passed to method **XAResource.setTransactionTimeout**.

Table 3.16. Statement parameters

Parameter	Description
track-statements	Whether to check for unclosed statements when a connection is returned to a pool and a statement is returned to the prepared statement cache. If false, statements are not tracked. **Valid values** • **true**: statements and result sets are tracked, and a warning is issued if they are not closed. • **false**: neither statements or result sets are tracked. • **nowarn**: statements are tracked but no warning is issued. This is the default.

Parameter	Description
prepared-statement-cache-size	The number of prepared statements per connection, in a Least Recently Used (LRU) cache.
share-prepared-statements	Whether asking for the same statement twice without closing it uses the same underlying prepared statement. The default is **false**.

Table 3.17. Recovery parameters

Parameter	Description
recover-credential	A username/password pair or security domain to use for recovery.
recover-plugin	An implementation of the `org.jboss.jca.core.spi.recoveryRecoveryPlugin` class, to be used for recovery.

3.9.6.9. JDBC Driver Download Locations

The following table gives the standard download locations for JDBC drivers of common databases used with JBoss EAP 6.

NOTE

These links point to third-party websites which are not controlled or actively monitored by Red Hat. For the most up-to-date drivers for your database, check your database vendor's documentation and website.

Table 3.18. JDBC driver download locations

Vendor	Download Location
MySQL	http://www.mysql.com/products/connector/
PostgreSQL	http://jdbc.postgresql.org/
Oracle	http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html
IBM	http://www-306.ibm.com/software/data/db2/java/
Sybase	http://www.sybase.com/products/allproductsa-z/softwaredeveloperkit/jconnect
Microsoft	http://msdn.microsoft.com/data/jdbc/

3.9.7. LDAP Data Sources

LDAP data sources use a Data Virtualization-specific JCA connector which is deployed into EAP during installation. There are many ways to create the ldap data source, using CLI, AdminShell, admin-console, and so forth. The example below uses the CLI tool.

Execute the following command using the CLI once you connected to the Server. Make sure you provide the correct URL and user credentials. Add any additional properties required by the connector by duplicating the "connection-definitions" command below. Edit the JNDI name to match the JNDI name you used in the VDB:

```
batch
/subsystem=resource-adapters/resource-adapter=ldap/connection-
definitions=ldapDS:add(jndi-name=java:/ldapDS, class-
name=org.teiid.resource.adapter.ldap.LDAPManagedConnectionFactory,
enabled=true, use-java-context=true)
/subsystem=resource-adapters/resource-adapter=ldap/connection-
definitions=ldapDS/config-
properties=LdapUrl:add(value=ldap://ldapServer:389)
/subsystem=resource-adapters/resource-adapter=ldap/connection-
definitions=ldapDS/config-properties=LdapAdminUserDN:add(value=
{cn=???,ou=???,dc=???})
/subsystem=resource-adapters/resource-adapter=ldap/connection-
definitions=ldapDS/config-properties=LdapAdminUserPassword:add(value=
{pass})
/subsystem=resource-adapters/resource-adapter=ldap/connection-
definitions=ldapDS/config-properties=LdapTxnTimeoutInMillis:add(value=-1)
/subsystem=resource-adapters/resource-adapter=ldap:activate
runbatch
```

To find out all the properties that are supported by this LDAP Connector execute the following command in the CLI:

```
/subsystem=teiid:read-rar-description(rar-name=ldap)
```

3.9.8. MongoDB Data Sources

The MongoDB data sources use a Data Virtualization-specific JCA connector that is deployed into EAP during installation. There are many ways to create a MongoDB data source, using CLI, AdminShell, admin-console, and so forth. The example below uses the CLI tool.

Execute the following command using the CLI once you connected to the Server. Make sure you provide the correct URL and user credentials. Add any additional properties required by the connector by duplicating the "connection-definitions" command below. Edit the JNDI name to match the JNDI name you used in the VDB:

```
batch
/subsystem=resource-adapters/resource-adapter=mongodb/connection-
definitions=mongodbDS:add(jndi-name="java:/mongoDS", class-
name=org.teiid.resource.adapter.mongodb.MongoDBManagedConnectionFactory,
enabled=true, use-java-context=true)
/subsystem=resource-adapters/resource-adapter=mongodb/connection-
definitions=mongodbDS/config-properties=RemoteServerList:add(value="
{host}:27017")
/subsystem=resource-adapters/resource-adapter=mongodb/connection-
```

```
definitions=mongodbDS/config-properties=Database:add(value="{db-name}")
/subsystem=resource-adapters/resource-adapter=mongodb:activate
runbatch
```

These are the properties that are defined in the RAR file:

Table 3.19. Properties

Property	Description	Required?	Default
RemoteServerList	A comma-separated list of server locations. Each location can contain an optional port, of the format host:port	False.	Not applicable.
Username	Connection User's Name	False.	None.
Password	Connection User's password	False.	None.
Database	MongoDB database name	True.	None.

To find out all the properties that are supported by this MongoDB Connector execute the following command in the CLI:

```
/subsystem=teiid:read-rar-description(rar-name=mongodb)
```

3.9.9. Apache Phoenix Data Source

The following is an example that teaches you to set up Phoenix Data Sources, which are needed before you can use the Apache HBase Translator.

There are configuration templates for Phoenix data sources in the EAP_HOME/docs/teiid/datasources directory. A complete description on how a data source can be added into Red Hat JBoss EAP is also described here.

Configuring a Phoenix data source is nearly identical to configuring JDBC Data Sources. The first step is deploying the Phoenix driver jar. Using below CLI command to deploy the Phoenix driver:

```
module add --name=org.apache.phoenix --resources=/path/to/phoenix-
[version]-client.jar --
dependencies=javax.api,sun.jdk,org.apache.log4j,javax.transaction.api
/subsystem=datasources/jdbc-driver=phoenix:add(driver-name=phoenix,driver-
module-name=org.apache.phoenix,driver-class-
name=org.apache.phoenix.jdbc.PhoenixDriver
```

The driver jar can be downloaded from Phoenix.

Next, create the Data Source base on above deployed driver, which is also like creating JDBC Data Source. Using below CLI command to create the data source:

```
/subsystem=datasources/data-source=phoenixDS:add(jndi-
```

```
name=java:/phoenixDS, driver-name=phoenix, connection-url=jdbc:phoenix:
{zookeeper quorum server}, enabled=true, use-java-context=true, user-name=
{user}, password={password})
```

Ensure the URL, Driver, and other properties are configured correctly: The JNDI name need to match the JNDI name you used in the VDB. The driver name need to match the driver you deployed in above steps

The URL need to match the HBase zookeeper quorum server. Here is an example:

```
 jdbc:phoenix [ :<zookeeper quorum> [ :<port number> ] [ :<root node> ] ],
'jdbc:phoenix:127.0.0.1:2181'
```

user-name/password - The user credentials for Phoenix Connection

Mapping a Phoenix table to an existing HBase table takes two steps. The first step is installing phoenix-[version]-server.jar to the classpath of every HBase region server. An easy way to do this is to copy it into the HBase lib. (For more details see the Phoenix documentation.)

Next, you must executing the DDL to map a Phoenix table to an existing HBase table. The DDL can either be executed via the Phoenix Command Line, or executed by JDBC.

Creating a new Phoenix table is just like mapping to an existing HBase table. Phoenix will create any metadata (table, column families) that do not exist. Similar to the above example the DDL to create the Phoenix/HBase Customer table would be:

```
CREATE TABLE IF NOT EXISTS "Customer"("ROW_ID" VARCHAR PRIMARY KEY,
"customer"."city" VARCHAR, "customer"."name" VARCHAR, "sales"."amount"
VARCHAR, "sales"."product" VARCHAR)
```

3.9.10. Salesforce Data Sources

Salesforce data sources use a Data Virtualization-specific JCA connector that is deployed into EAP during installation. There are many ways to create the salesforce data source, using CLI,AdminShell, admin-console, and so forth. The example below uses the CLI tool.

Execute the following command using the CLI once you connected to the server. Make sure you provide the correct URL and user credentials. Add any additional properties required by the connector by duplicating the "connection-definitions" command below. Edit the JNDI name to match the JNDI name you used in the VDB:

```
batch
/subsystem=resource-adapters/resource-adapter=salesforce/connection-
definitions=sfDS:add(jndi-name=java:/sfDS, class-
name=org.teiid.resource.adapter.salesforce.SalesForceManagedConnectionFact
ory, enabled=true, use-java-context=true)
/subsystem=resource-adapters/resource-adapter=salesforce/connection-
definitions=sfDS/config-
properties=URL:add(value=https://www.salesforce.com/services/Soap/u/22.0)
/subsystem=resource-adapters/resource-adapter=salesforce/connection-
definitions=sfDS/config-properties=username:add(value={user})
/subsystem=resource-adapters/resource-adapter=salesforce/connection-
definitions=sfDS/config-properties=password:add(value={password})
/subsystem=resource-adapters/resource-adapter=salesforce:activate
runbatch
```

To find out all the properties that are supported by this Salesforce Connector execute the following command in the CLI:

```
/subsystem=teiid:read-rar-description(rar-name=salesforce)
```

3.9.11. Solr Data Sources

Solr data sources use a Data Virtualization-specific JCA connector that is deployed into EAP during installation. There are many ways to create a Solr data source, using CLI, AdminShell, admin-console, and so forth. The example below uses the CLI tool.

Execute the following command using the CLI once you connected to the Server. Make sure you provide the correct URL and user credentials. Add any additional properties required by the connector by duplicating the "connection-definitions" command below. Edit the JNDI name to match the JNDI name you used in the VDB:

```
batch
/subsystem=resource-adapters/resource-adapter=solr/connection-
definitions=solrDS:add(jndi-name=java:/solrDS, class-
name=org.teiid.resource.adapter.solr.SolrManagedConnectionFactory,
enabled=true, use-java-context=true)
/subsystem=resource-adapters/resource-adapter=solr/connection-
definitions=solrDS/config-
properties=url:add(value=http://localhost:8983/solr/)
/subsystem=resource-adapters/resource-adapter=solr/connection-
definitions=solrDS/config-properties=CoreName:add(value=collection1)
/subsystem=resource-adapters/resource-adapter=solr:activate
runbatch
```

To find out all the properties that are supported by this Solr Connector execute the following command in the CLI:

```
/subsystem=teiid:read-rar-description(rar-name=solr)
```

3.9.12. Web Service Data Sources

Web service data sources use a Data Virtualization-specific JCA connector that is deployed into EAP during installation. There are many ways to create the file data source, using CLI, AdminShell, admin-console, and so forth. The example below uses the CLI tool.

Execute following command using the CLI once you connected to the Server. Make sure you provide the correct endpoint and other properties below. Add any additional properties required by the connector by duplicating the "connection-definitions" command below. Edit the JNDI name to match the JNDI name you used in the VDB:

```
batch
/subsystem=resource-adapters/resource-adapter=webservice/connection-
definitions=wsDS:add(jndi-name=java:/wsDS, class-
name=org.teiid.resource.adapter.ws.WSManagedConnectionFactory,
enabled=true, use-java-context=true)
/subsystem=resource-adapters/resource-adapter=webservice/connection-
```

```
definitions=wsDS/config-properties=EndPoint:add(value={end_point})
/subsystem=resource-adapters/resource-adapter=webservice:activate
runbatch
```

To find out all the properties that are supported by this Web Service Connector execute the following command in the CLI:

```
/subsystem=teiid:read-rar-description(rar-name=webservice)
```

The Web Service Data Source supports specifying a WSDL using the Wsdl property. If the Wsdl property is set, then the ServiceName, EndPointName, and NamespaceUri properties should also be set. The Wsdl property may be a URL or file location or the WSDL to use. You must restart the server.

Table 3.20. Registry Properties

Property	Application	Required?	Default?	Description
EndPoint	HTTP and SOAP.	True	Not applicable.	URL for HTTP ans service endpoint for SOAP.
SecurityType	HTTP and SOAP.	false	none	Type of Authentication to used with the web service. Allowed values are "None","HTTPBasic","WSSecurity" and "Kerberos".
AuthUserName	HTTP and SOAP.	false	Not applicable.	Name value for authentication, used in HTTPBasic and WsSecurity.
AuthPassword	HTTP and SOAP.	false	Not applicable.	Password value for authentication, used in HTTPBasic and WsSecurity
ConfigFile	HTTP and SOAP.	False	Not applicable.	CXF client configuration File or URL.
ConfigName	HTTP and SOAP.	False	Not applicable.	Note that this property is deprecated.
EndPointName	HTTP and SOAP.	False	Teiid	Local part of the end point QName to use with this connection, needs to match one defined in CXF file.

Property	Application	Required?	Default?	Description
ServiceName	SOAP	False	Not applicable.	Local part of the service QName to use with this connection.
NamespaceUri	SOAP.	False	http://teiid.org	Namespace URI of the service QName to use with this connection
RequestTimeout	HTTP and SOAP.	False	Not applicable.	Timeout for request.
ConnectTimeout	HTTP and SOAP.	False	Not applicable.	Timeout for connection.
Wsdl	SOAP.	False	Not applicable.	WSDL file or URL for the web service.

Each web service data source may choose a particular CXF config file and port configuration. The ConfigFile config property specifies the Spring XML configuration file for the CXF Bus and port configuration to be used by connections. If no config file is specified then the system default configuration will be used.

Only one port configuration can be used by this data source. You may explicitly set the local name of the port QName to use via the EndPointName property. The namespace URI for the QName in your config file should match your WSDL/namespace setting on the data source or use the default of http://teiid.org. See the CXF Documentation and and the CXF Configuration chapter of this book for examples of using the CXF configuration file.

3.10. MANAGING DEPLOYED VIRTUAL DATABASES USING MANAGEMENT CONSOLE

3.10.1. Managing Deployed Virtual Databases Using Management Console

Administrators can manage deployed virtual databases from within the **Virtual Databases** panel.

3.10.2. Opening the Virtual Databases Panel

1. Log in to the JBoss Management Console.

2. Select the **Runtime** tab.

3. From the navigation tree, select **Status** → **Subsystems** → **Virtual Databases**.

3.11. RESOURCE ADAPTERS

3.11.1. Resource Adapters in JBoss Data Virtualization

With the exception of JDBC data sources, JBoss Data Virtualization provides a JCA adapter for each supported data source. These are the resource adapter identifiers, as specified in the server configuration file:

- File Adapter - **file**

- Google Spreadsheet Adapter - **google**

- LDAP Adapter - **ldap**

- Salesforce Adapter - **salesforce**

- Salesforce 34 Adapter - **salesforce-34**

- Web Services Adapter - **webservice**

- MongoDB Adapter - **mongodb**

- Cassandra Adapter - **cassandra**

- Accumulo Adapter - **accumulo**

- Solr Adapter - **solr**

- Couchbase Adapter - **couchbase**

- Infinispan Hotrod Adapter (not added by default) - **infinispan-hotrod**

> **NOTE**
>
> A resource adapter for the JDBC translator is provided with Red Hat JBoss EAP by default.

3.11.2. Configuring Resource Adapters

You can use Management Console, AdminShell, or Management CLI (or, in standalone mode, directly edit the server configuration file) to configure resource adapters for data sources required by a VDB.

> **NOTE**
>
> Note that in domain mode, you must use Management CLI, Management Console or AdminShell to configure data sources. Refer to the JBoss Enterprise Application Platform *Administration and Configuration Guide* for an example of how to configure resource adapters using the Management Console.

IMPORTANT

If you configure a resource adapter using either a CLI script or the AdminShell, the name of the resource adapter must not be the same as that of an existing one. (Teiid Designer handles this by creating a unique name for the resource adapter.)

The reasons for this are as follows:

- Any time a resource adapter with a duplicate name is added, the server has to be restarted. This conforms with the way Red Hat JBoss EAP's JCA system was designed, in that multiple instances of resource adapters with the same name are only recognised when the server is launched.

- Anytime you delete a resource adapter by name, all instances with the same name will be deleted.

3.11.3. Example Configuration

Example configuration, as it would appear in the server configuration file, can be found in the **EAP_HOME/docs/teiid/datasources/** directory.

NOTE

This configuration will need to be adjusted with file paths and properties appropriate to your installation. The JNDI name must be the same JNDI name used in the VDB.

3.11.4. Resource Adapter Properties

For a full listing of configuration properties, you can run the following command from within the Management CLI:

```
/subsystem=teiid:read-rar-description(rar-name=ADAPTER_ID)
```

ADAPTER_ID refers to the identifier of the resource adapter as specified in the server configuration file.

3.11.5. Configuring Resource Adapters Using CLI Scripts

This is the preferred method for configuring resource adapters.

JBoss Data Virtualization provides an example Management CLI script for configuring each of the provided resource adapters. A script for a particular resource adapter can be found in the **EAP_HOME/docs/teiid/datasources/DATASOURCE** directory.

When configuring resource adapters using the Management CLI, you must provide a properties file. Sample properties files are provided in the same directory as the sample scripts.

You can run these scripts (once JBoss Data Virtualization is running) by executing the following command:

```
./EAP_HOME/bin/jboss-cli.sh --connect --
file=EAP_HOME/docs/teiid/datasources/DATASOURCE/SCRIPT.cli --
properties=EAP_HOME/docs/teiid/datasources/DATASOURCE/SCRIPT.properties
```

For more information about using the Management CLI, see the JBoss Enterprise Application Platform *Administration and Configuration Guide*.

NOTE

These scripts will need to be adjusted with file paths and properties appropriate to your installation. The JNDI name must be the same JNDI name used in the VDB.

CHAPTER 4. VERSIONING

4.1. VIRTUAL DATABASE VERSIONING

Virtual database (VDB) versioning allows you to deploy multiple versions of a VDB simultaneously, and to specify which version will be used in certain scenarios.

When an application connects to JBoss Data Virtualization, you can set the desired version of a VDB using a connection property. (Refer to the *Red Hat JBoss Data Virtualization User Guide* for more information.)

If a specific version is set, then you can only connect to that VDB. If no version is set, the deployed VDBs are searched until the appropriate version is found. This feature helps support more dynamic migration scenarios.

4.2. SET THE VDB VERSION

You can set the version in one of two ways: through the **vdb.xml** file, (which is useful for dynamic VDBs), or by specifying a naming convention in the deployment file (such as *VDBNAME.VERSION*.**vdb**). The deployer is responsible for choosing an appropriate version number. If there is already a VDB name and version combination that matches the current deployment, then connections to the previous VDB will be terminated and its cache entries will be flushed. Any new connections will then be made to the new VDB.

4.3. VIRTUAL DATABASE CONNECTION TYPE

Once your VDB is deployed, you can configure a property in it called connection type. By setting this property, you can determine what type of connections can be made to the VDB. You can set it to one of the following:

- *NONE*: disallow new connections.

- *BY_VERSION*: (the default setting) allow connections only if the version is specified or if this is the earliest BY_VERSION VDB and there are no VDBs marked as ANY.

- *ANY*: allow connections with or without a version specified.

If you only want to migrate a few of your applications to the new version of the VDB, then set it to BY_VERSION. This ensures that only applications that know of the new version may use it.

If only a select few applications are to remain on the current VDB version, then you will need to update their connection settings to reference the current VDB by its version. The newly deployed VDB will then have its connection type set to ANY, which allows all new connections to be made against the newer version.

If you need to undertake a rollback in this scenario, then the newly-deployed VDB will, accordingly, have its connection type set to NONE or BY_VERSION.

4.4. SET THE VDB CONNECTION TYPE VIA ADMIN API

You can change a VDB connection type using the **changeVDBConnectionType** method provided by the Admin interface within the Admin API package (**org.teiid.adminapi**).

Javadocs for Red Hat JBoss Data Virtualization can be found on the Red Hat Customer Portal.

CHAPTER 5. CXF CONFIGURATION

5.1. CXF CONFIGURATION

If your configuration uses CXF, you need to specify the data source configuration file and port configuration.

The ConfigFile property, set in the server configuration file for the resource-adapter subsystem, specifies the Spring XML configuration file. (The system's default configuration is used if the configuration file is not specified.)

The EndPointName property, also set in the server configuration file for the resource-adapter subsystem, specifies the port configuration. Set this to the local part of the desired port QName.

The configuration file property is specified via this command:

```
/subsystem=resource-adapters/resource-adapter=webservice/connection-
definitions=wsDS/config-
properties=ConfigFile:add(value=${jboss.server.home.dir}/standalone/config
uration/xxx-jbossws-cxf.xml)
```

5.2. CONFIGURE CXF FOR A WEB SERVICE DATA SOURCE

Prerequisites

- You must have configured a web service data source.

Procedure 5.1. Configure CXF for a Web Service Data Source

1. **Specify the Web Service CXF `ConfigFile`**
   Run the following command from within the Management CLI, specifying the CXF configuration file for the data source:

   ```
   /subsystem=resource-adapters/resource-adapter=webservice/connection-
   definitions=wsDS/config-properties=ConfigFile:add(value=CONFIG-
   FILE.xml)
   ```

2. **Specify the Web Service CXF `EndPointName`**
   Specify the port configuration by running the following command from within the Management CLI, using the port QName (local part only) for the value:

   ```
   /subsystem=resource-adapters/resource-adapter=webservice/connection-
   definitions=wsDS/config-properties=EndPointName:add(value=CONFIG-
   NAME)
   ```

3. **Create/Edit the CXF Configuration File**
   Open/create the *EAP_HOME/MODE/*configuration/*CONFIG-FILE*.xml configuration file.

   ```
   <http-conf:conduit name="{NAMESPACE}CONFIG-NAME.http-conduit">
   ...
   </http-conf:conduit>
   ```
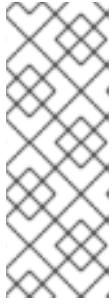
> **NOTE**
>
> *CONFIG-NAME* is the same as that specified above, with a default value of **teiid**. *NAMESPACE* is the namespace URI for the QName in your config file, which should match your WSDL/namespace setting on the data source or use the default of "http://teiid.org". The namespace may be set via the namespace datasource property. Typically that will only need done when also supplying the WSDL setting.

The following is an example of a CXF file configuring timeouts:

```
<beans xmlns="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:http-
conf="http://cxf.apache.org/transports/http/configuration"

xsi:schemaLocation="http://cxf.apache.org/transports/http/configurat
ion
  http://cxf.apache.org/schemas/configuration/http-conf.xsd
  http://www.springframework.org/schema/beans
  http://www.springframework.org/schema/beans/spring-beans.xsd">

  <http-conf:conduit name="{NAMESPACE}CONFIG-NAME.http-conduit">
    <http-conf:client ConnectionTimeout="120000"
ReceiveTimeout="240000"/>
  </http-conf:conduit>
</beans>
```

> **NOTE**
>
> For web service data sources, CXF configuration is only applicable to non-binary calls.

5.3. CONFIGURE CXF FOR A WEB SERVICE DATA SOURCE: WS-SECURITY

Prerequisites

- The web service data source must be configured and the **ConfigFile** and **EndPointName** properties must be configured for CXF.

Procedure 5.2. Configure CXF for a Web Service Data Source: WS-Security

1. **Specify the CXF SecurityType**
   Run the following command from within the Management CLI, using **WSSecurity** as the value for **SecurityType**:

   ```
   /subsystem=resource-adapters/resource-adapter=webservice/connection-
   definitions=wsDS/config-
   properties=SecurityType:add(value=WSSecurity)
   ```

2. **Modify the CXF Configuration File**
   Open the CXF configuration file for the web service data source and add your desired properties.

The following is an example of a web service data source CXF configuration file adding a timestamp to the SOAP header:

```
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:jaxws="http://cxf.apache.org/jaxws"

xsi:schemaLocation="http://www.springframework.org/schema/beans
            http://www.springframework.org/schema/beans/spring-
beans.xsd
            http://cxf.apache.org/jaxws
            http://cxf.apache.org/schemas/jaxws.xsd">

    <jaxws:client name="{http://teiid.org}.teiid"
        createdFromAPI="true">
        <jaxws:outInterceptors>
            <ref bean="Timestamp_Request"/>
        </jaxws:outInterceptors>
    </jaxws:client>

    <bean
        id="Timestamp_Request">
        <constructor-arg>
            <map>
                <entry key="action" value="Timestamp"/>
            </map>
        </constructor-arg>
    </bean>

</beans>
```

NOTE

- A WSDL is not expected to describe the service being used.

- The Spring XML configuration file must contain the relevant policy configuration.

- The client port configuration is matched to the data source instance by the **CONFIG-NAME**. The configuration may contain other port configurations with different local names.

References

- For more information about WS-Security and CXF configuration options refer to http://cxf.apache.org/docs/ws-security.html.

5.4. CONFIGURE CXF FOR A WEB SERVICE DATA SOURCE: LOGGING

CXF configuration can control the logging of requests and responses for specific or all ports. Logging, when enabled, is performed at an **INFO** level to the **org.apache.cxf.interceptor** context.

Prerequisites

- The web service data source must be configured and the **ConfigFile** and **EndPointName** properties must be configured for CXF.

Procedure 5.3. Configure CXF for a Web Service Data Source: Logging

- **Modify the CXF Configuration File**
  Open the CXF configuration file for the web service data source and add your desired logging properties.

  The following is an example of a CXF configuration file for a web service data source that enables logging:

  ```
  <beans xmlns="http://www.springframework.org/schema/beans"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns:jaxws="http://cxf.apache.org/jaxws"

  xsi:schemaLocation="http://www.springframework.org/schema/beans
            http://www.springframework.org/schema/beans/spring-
  beans.xsd
            http://cxf.apache.org/jaxws
            http://cxf.apache.org/schemas/jaxws.xsd">

      <jaxws:client name="{http://teiid.org}teiid"
          createdFromAPI="true">
          <jaxws:features>
              <bean class="org.apache.cxf.feature.LoggingFeature"/>
          </jaxws:features>
      </jaxws:client>

  </beans>
  ```

References

- For more information about CXF logging configuration options see
  http://cxf.apache.org/docs/debugging-and-logging.html.

5.5. CONFIGURE CXF FOR A WEB SERVICE DATA SOURCE: TRANSPORT SETTINGS

CXF configuration can also control low level aspects of the HTTP transport.

Prerequisites

- The web service data source must be configured and the **ConfigFile** and **EndPointName** properties must be configured for CXF.

Procedure 5.4. Configure CXF for a Web Service Data Source: Transport Settings

- Open the CXF configuration file for the web service data source and add your desired transport properties.

  The following is an example of a CXF configuration file for a web service data source that disables hostname verification:

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:jaxws="http://cxf.apache.org/jaxws"

xsi:schemaLocation="http://cxf.apache.org/transports/http/configurat
ion
           http://cxf.apache.org/schemas/configuration/http-conf.xsd
           http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-
beans.xsd">

    <http-conf:conduit name="{http://teiid.org}teiid.http-conduit">
         <http-conf:tlsClientParameters disableCNcheck="tru" />
    </http-conf:conduit>

</beans>
```

> **WARNING**
>
> **disableCNcheck=true** must NOT be used in production.

5.6. CONFIGURE CXF FOR A WEB SERVICE DATA SOURCE: SSL SUPPORT (HTTPS)

For using HTTPS, you can configure the CXF configuration file as below:

```
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:sec="http://cxf.apache.org/configuration/security"
    xmlns:http-conf="http://cxf.apache.org/transports/http/configuration"
    xmlns:jaxws="http://java.sun.com/xml/ns/jaxws"

xsi:schemaLocation="http://cxf.apache.org/transports/http/configuration
http://cxf.apache.org/schemas/configuration/http-conf.xsd
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
http://cxf.apache.org/configuration/security
http://cxf.apache.org/schemas/configuration/security.xsd">

    <http-conf:conduit name="*.http-conduit">
        <http-conf:client ConnectionTimeout="120000"
ReceiveTimeout="240000"/>
        <http-conf:tlsClientParameters secureSocketProtocol="SSL">
          <sec:trustManagers>
            <sec:keyStore type="JKS" password="changeit"
file="/path/to/truststore.jks"/>
          </sec:trustManagers>
        </http-conf:tlsClientParameters>
    </http-conf:conduit>
</beans>
```

■

For more information about http-conduit based configuration see http://cxf.apache.org/docs/client-http-transport-including-ssl-support.html . You can also configure for services such as HTTPBasic and Kerberos.

5.7. CONFIGURE CXF FOR A SALESFORCE DATA SOURCE

Prerequisites

- You must have configured a Salesforce data source.

Procedure 5.5. Configure CXF for a Salesforce Data Source

1. Run the following command from within the Management CLI, specifying the CXF configuration file for the data source:

   ```
   /subsystem=resource-adapters/resource-adapter=salesforce/connection-
   definitions=sfDS/config-properties=ConfigFile:add(value=CONFIG-
   FILE.xml)
   ```

2. Specify the port configuration by running the following command from within the Management CLI, using the port QName (local part only) for the value, in this case **Soap**:

   ```
   /subsystem=resource-adapters/resource-adapter=salesforce/connection-
   definitions=sfDS/config-properties=EndPointName:add(value=Soap)
   ```

3. Open/create the *EAP_HOME/MODE*/**configuration/***CONFIG-FILE*.**xml** configuration file. Set the namespace URI for the QName to **{urn:partner.soap.sforce.com}**, using **Soap** as the value for **EndPointName**:

   ```
   <http-conf:conduit name="{urn:partner.soap.sforce.com}Soap.http-
   conduit">
   ...
   </http-conf:conduit>
   ```

   The following is an example of a CXF file that configures timeout values:

   ```
   <beans xmlns="http://www.springframework.org/schema/beans"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns:http-
   conf="http://cxf.apache.org/transports/http/configuration"

   xsi:schemaLocation="http://cxf.apache.org/transports/http/configurat
   ion
     http://cxf.apache.org/schemas/configuration/http-conf.xsd
     http://www.springframework.org/schema/beans
     http://www.springframework.org/schema/beans/spring-beans.xsd">

     <http-conf:conduit name="{urn:partner.soap.sforce.com}Soap.http-
   conduit">
       <http-conf:client ConnectionTimeout="120000"
   ReceiveTimeout="240000"/>
     </http-conf:conduit>
   ```

```
</beans>
```
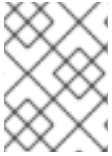
> **NOTE**
>
> CXF configuration for Salesforce data sources is only used for http bus configuration. It is not used for WS-Security. Salesforce has its own security authentication.

CHAPTER 6. LOGGING

6.1. OVERVIEW OF LOGGING

JBoss EAP 6 provides highly configurable logging facilities for both its own internal use and for use by deployed applications. The logging subsystem is based on JBoss LogManager and it supports several third party application logging frameworks in addition to JBoss Logging.

The logging subsystem is configured using a system of log categories and log handlers. Log categories define what messages to capture, and log handlers define how to deal with those messages (write to disk, send to console etc).

Logging Profiles allow uniquely named sets of logging configuration to be created and assigned to applications independent of any other logging configuration. The configuration of logging profiles is almost identical to the main logging subsystem.

6.2. DEFAULT LOG FILE LOCATIONS

These are the log files that get created for the default logging configurations. The default configuration writes the server log files using periodic log handlers.

Table 6.1. Default Log File for a standalone server

Log File	Description
EAP_HOME/standalone/log/server.log	Server Log. Contains all server log messages, including server startup messages.
EAP_HOME/standalone/log/gc.log	Garbage collection log. Contains details of all garbage collection.

Table 6.2. Default Log Files for a managed domain

Log File	Description
EAP_HOME/domain/log/host-controller.log	Host Controller boot log. Contains log messages related to the startup of the host controller.
EAP_HOME/domain/log/process-controller.log	Process controller boot log. Contains log messages related to the startup of the process controller.
EAP_HOME/domain/servers/*SERVERNAME*/log/server.log	The server log for the named server. Contains all log messages for that server, including server startup messages.

6.3. JBOSS DATA VIRTUALIZATION LOG CATEGORIES

Within EAP log files, all information specific to JBoss Data Virtualization is prefixed with the `org.teiid` log category identifier.

Logs originating from third-party code and other components not related to JBoss Data Virtualization, including integrated org.jboss components, are prefixed with their own log category identifiers, not **org.teiid**.

The table below summarizes the categories relevant to JBoss Data Virtualization. For a complete listing, see the **standalone.xml** (or **domain.xml** for domain mode) file.

Table 6.3. JBoss Data Virtualization Log Categories

Log Category	Description
com.arjuna	Third-party transaction manager. This will include information about all transactions, not just those for JBoss Data Virtualization.
org.teiid	Root category identifier for all JBoss Data Virtualization logs. Note: there are potentially more contexts used under org.teiid than those shown in this table.
org.teiid.PROCESSOR	Query processing logs. Also see org.teiid.PLANNER.
org.teiid.PLANNER	Query planning logs.
org.teiid.SECURITY	Session/Authentication events. Also see org.teiid AUDIT_LOG.
org.teiid.TRANSPORT	Events related to the socket transport.
org.teiid.RUNTIME	Events related to work management and system start/stop.
org.teiid.CONNECTOR	Connector logs.
org.teiid.BUFFER_MGR	Buffer and storage management logs.
org.teiid.TXN_LOG	Detailed log of all transaction operations.
org.teiid.COMMAND_LOG	Refer to Section 6.4, "Command Logging".
org.teiid.AUDIT_LOG	Refer to Section 6.5, "Audit Logging".
org.teiid.ADMIN_API	Admin API logs.
org.teiid.ODBC	ODBC logs.

NOTE

Refer to the JBoss Enterprise Application Platform *Administration and Configuration Guide* for more information about logging.

6.4. COMMAND LOGGING

Command logging captures user commands that have been submitted to JBoss Data Virtualization, query plan commands when query planning is performed, and data source commands that are being executed by connectors.

The user command, "START USER COMMAND", is logged when JBoss Data Virtualization starts working on the query for the first time. This does not include the time the query was waiting in the queue. A corresponding user command, "END USER COMMAND", is logged when the request is complete (that is, when the statement is closed or all the batches are retrieved). There is only one pair of these for every user query.

The query plan command, "PLAN USER COMMAND", is logged when JBoss Data Virtualization finishes the query planning process. There is no corresponding ending log entry.

Non-plan user events are logged at the INFO level.

The data source command, "START DATA SRC COMMAND", is logged when a query is sent to the data source. And a corresponding data source command, "END SRC COMMAND", is logged when the execution is closed (that is, all the rows has been read). There can be one pair for each data source query that has been executed by JBoss Data Virtualization, and there can be number of pairs depending upon your query.

With this information being captured, the overall query execution time can be calculated. Additionally, each source query execution time can be calculated. If the overall query execution time is showing a performance issue, then look at each data source execution time to see where the issue may be.

With this information being captured, the overall query execution time in Teiid can be calculated. Additionally, each source query execution time can be calculated. If the overall query execution time is showing a performance issue, then look at each data source execution time to see where the issue maybe.

To enable command logging to the default log location, simply enable the DEBUG level of logging for the org.teiid.COMMAND_LOG context. You can use the Admin Console to enable or disable it. Note that you can also turn on command logging using the web-console.

To enable command logging to an alternative file location, configure a separate file appender for the DETAIL logging of the org.teiid.COMMAND_LOG context. An example of this is shown below and can also be found in the standalone.xml file.

```
<periodic-rotating-file-handler name="COMMAND_FILE">
    <level name="DEBUG" />
    <formatter>
        <pattern-formatter pattern="%d{HH:mm:ss,SSS} %-5p [%c] (%t)
%s%E%n" />
    </formatter>
    <file relative-to="jboss.server.log.dir" path="command.log" />
    <suffix value=".yyyy-MM-dd" />
</periodic-rotating-file-handler>

<logger category="org.teiid.COMMAND_LOG">
    <level name="DEBUG" />
    <handlers>
        <handler name="COMMAND_FILE" />
    </handlers>
</logger>
```

See *Red Hat JBoss Data Virtualization Development Guide: Server Development* for information on developing a custom logging solution if file based (or any other built-in log4j) logging is not sufficient.

The following is an example of a data source command and what one would look like when printed to the command log:

```
2012-02-22 16:01:53,712 DEBUG [org.teiid.COMMAND_LOG]
(Worker1_QueryProcessorQueue11 START DATA SRC COMMAND: startTime=2012-02-
22 16:01:53.712
requestID=Ku4/dgtZPYk0.5 sourceCommandID=4 txID=null modelName=DTHCP
translatorName=jdbc-simple sessionID=Ku4/dgtZPYk0
principal=user@teiid-security
sql=HCP_ADDR_XREF.HUB_ADDR_ID, CPN_PROMO_HIST.PROMO_STAT_DT FROM
CPN_PROMO_HIST, HCP_ADDRESS, HCP_ADDR_XREF
WHERE (HCP_ADDRESS.ADDR_ID = CPN_PROMO_HIST.SENT_ADDR_ID) AND
(HCP_ADDRESS.ADDR_ID = HCP_ADDR_XREF.ADDR_ID) AND
(CPN_PROMO_HIST.PROMO_STAT_CD NOT LIKE 'EMAIL%') AND
(CPN_PROMO_HIST.PROMO_STAT_CD <> 'SENT_EM') AND
(CPN_PROMO_HIST.PROMO_STAT_DT > {ts'2010-02-22 16:01:52.928'})
```

Note the following pieces of information:

- modelName: this represents the physical model for the data source that the query is being issued.

- translatorName: shows type of translator used to communicate to the data source.

- principal: shows the user account who submitted the query

- startTime/endTime: the time of the action, which is based on the type command being executed.

- sql: is the command submitted to the translator for execution, which is NOT necessarily the final sql command submitted to the actual data source. But it does show what the query engine decided to push down.

6.5. AUDIT LOGGING

Audit logging captures important security events, including the enforcement of permissions, and authentication success/failure.

See *Red Hat JBoss Data Virtualization Development Guide: Server Development* for information on developing a custom logging solution if file based (or any other built-in log4j) logging is not sufficient.

6.6. AUDIT AND COMMAND LOGGING

Command logging is enabled by default.

> **NOTE**
>
> You can replace file-based logging with database logging through the CLI or the web management console.

CHAPTER 7. CLUSTERING

7.1. CLUSTERING IN RED HAT JBOSS DATA VIRTUALIZATION

Clustering may be used to improve the performance of the system through:

Load Balancing

See *Red Hat JBoss Data Virtualization Development Guide: Client Development* for information on load balancing between multiple nodes.

Failover

See *Red Hat JBoss Data Virtualization Development Guide: Client Development* for information on failover with multiple nodes.

Distributed Caching

In cluster mode, resultsset caches and internal materialization caches are shared across the cluster automatically. Therefore no further configuration is needed.

Event Distribution

Metadata and data modifications will be distributed to all members of a cluster automatically when clustering is configured.

7.2. ENABLE CLUSTERING IN JBOSS DATA VIRTUALIZATION

Ensure JBoss Data Virtualization is installed on each JBoss EAP node and that JBoss EAP has started using either the **standalone-ha.xml** or the **standalone-full-ha.xml** profile before starting the cluster. (These profiles provide high availability, or clustering, as their names imply.)

For more information, see the *Red Hat JBoss Data Virtualization Installation Guide* and Red Hat JBoss Enterprise Application Platform documentation.

> **NOTE**
>
> Artifacts such as VDBs cannot be deployed via file deployment when clustering is used. Use other methods of VDB deployment.

PART IV. MONITORING AND PERFORMANCE

CHAPTER 8. MONITORING

8.1. MONITORING RED HAT JBOSS DATA VIRTUALIZATION

Red Hat JBoss Data Virtualization provides information about its current operational state. This information can be useful in tuning, monitoring, and managing load and throughput. Runtime data can be accessed using administrative tools such as web-console, AdminShell or Admin API.

8.2. QUERY/SESSION DETAILS

Name	Description
Current Sessions	List current connected sessions
Current Request	List current executing requests
Current Transactions	List current executing transactions
Query Plan	Retrieves the query plan for a specific request

There are administrative options for terminating sessions, queries, and transactions.

8.3. SESSION/QUERY METRICS

Name	Property	Description	Comment
Session Count	sessionCount	Indicates the number of user connections currently active.	To ensure number of sessions are not restricted at peak times, check max-sessions-allowed (default 5000) is set accordingly and review sessions-expiration-timelimit.
Query Count	queryCount	Indicates the number of queries currently active.	
Active Query Plan Count	ENGINE_STATISTIC.active-plans-count	Number of query plans currently being processed.	To ensure maximum throughput, see the performance tuning section on QueryEngine and threading.
Waiting Query Plan Count	ENGINE_STATISTIC.waiting-plans-count	Number of query plans currently waiting.	

Name	Property	Description	Comment
Max Waiting Query Plan Watermark	ENGINE_STATISTIC.max-waitplan-watermark	The maximum number of query plans that have been waiting at one time, since the last time the server started.	
Long Running Queries	longRunningQueries	List current executing queries that have surpassed the query threshold (query-threshold-in-seconds.).	Setup alert to warn when one or more queries are consuming resources for an extended period of time. If running too long, an option is to cancel request or increase threshold.

8.4. BUFFER MANAGER METRICS

Name	Property	Description	Comment
Disk Write Count	ENGINE_STATISTIC.buffermgr-disk-write-count	Disk write count for the buffer manager.	
Disk Read Count	ENGINE_STATISTIC.buffermgr-disk-read-count	Disk read count for the buffer manager.	
Cache Write Count	ENGINE_STATISTIC.buffermgr-cache-write-count	Cache write count for the buffer manager.	
Cache Read Count	ENGINE_STATISTIC.buffermgr-cache-read-count	Cache read count for the buffer manager.	
Disk Space Used (MB)	ENGINE_STATISTIC.buffermgr-diskspace-used-mb	Indicates amount of storage space currently used by buffer files	Setup alert to warn when used buffer space is at an unacceptable level, based on the setting of max-buffer-space
Total memory in use (KB)	ENGINE_STATISTIC.total-memory-inuse-kb	Estimate of the current memory usage in kilobytes.	
Total memory in use by active plans (KB)	ENGINE_STATISTIC.total-memory-inuse-active-plans-kb	Estimate of the current memory usage by active plans in kilobytes	

8.5. CACHE METRICS

Name	Property	Description
Prepared Plan Cache Size	PREPARED_PLAN_CACHE.total-entries	Current number of entries in cache.
Prepared Plan Cache # of Requests	PREPARED_PLAN_CACHE.request-count	Total number of requests made against cache.
Prepared Plan Cache Hit Ratio %	PREPARED_PLAN_CACHE.hit-ratio	Percentage of positive cache hits
ResultSet Cache Size	QUERY_SERVICE_RESULT_SET_CACHE.total-entries	Current number of entries in cache.
ResultSet Cache # of Requests	QUERY_SERVICE_RESULT_SET_CACHE.request-count	Total number of requests made against cache.
ResultSet Cache Hit Ratio %	QUERY_SERVICE_RESULT_SET_CACHE.hit-ratio	Percentage of positive cache hits.

CHAPTER 9. PERFORMANCE TUNING

9.1. MEMORY MANAGEMENT CONSIDERATIONS

Here is some information about settings related to memory management.

Red Hat JBoss Data Virtualization uses a BufferManager to track both memory and disk usage. Configuring the BufferManager properly is one of the most important parts of ensuring high performance. In most instances though the default settings are sufficient as they will scale with the JVM and consider other properties such as the setting for max active plans. Execute the following command using the CLI to find all of the possible settings for the BufferManager:

```
/subsystem=teiid:read-resource
```

All of the properties that start with "buffer-service" can be used to configure BufferManager. Here are the CLI commands you can use to change these settings:

```
/subsystem=teiid:write-attribute(name=buffer-service-use-disk,value=true)
/subsystem=teiid:write-attribute(name=buffer-service-encrypt-
files,value=false)
/subsystem=teiid:write-attribute(name=buffer-service-processor-batch-
size,value=256)
/subsystem=teiid:write-attribute(name=buffer-service-max-open-
files,value=64)
/subsystem=teiid:write-attribute(name=buffer-service-max-file-
size,value=2048)
/subsystem=teiid:write-attribute(name=buffer-service-max-processing-
kb,value=-1)
/subsystem=teiid:write-attribute(name=buffer-service-max-reserve-
kb,value=-1)
/subsystem=teiid:write-attribute(name=buffer-service-max-buffer-
space,value=51200)
/subsystem=teiid:write-attribute(name=buffer-service-max-inline-
lobs,value=true)
/subsystem=teiid:write-attribute(name=buffer-service-memory-buffer-
space,value=-1)
/subsystem=teiid:write-attribute(name=buffer-service-max-storage-object-
size,value=8388608)
/subsystem=teiid:write-attribute(name=buffer-service-memory-buffer-off-
heap,value=false)
```

> **WARNING**
>
> Do not change these properties until you understand the properties and any potential issue that is being experienced.

Here is an explanation of the main settings:

- max-reserve-kb (default -1) - setting determines the total size in kilobytes of batches that can be

held by the BufferManager in memory. This number does not account for persistent batches held by soft (such as index pages) or weak references. The default value of -1 will auto-calculate a typical max based upon the max heap available to the VM. The auto-calculated value assumes a 64bit architecture and will limit buffer usage to 40% of the first gigabyte of memory beyond the first 300 megabytes (which are assumed for use by the AS and other Red Hat JBoss Data Virtualization purposes) and 50% of the memory beyond that. The additional caveat here is that if the size of the memory buffer space is not specified, then it will effectively be allocated out of the max reserve space. A small adjustment is also made to the max reserve to account for batch tracking overhead.

With default settings and an 8GB VM size, then max-reserve-kb will at a max use: $(((1024-300) * 0.4) + (7 * 1024 * 0.5)) = 4373.6$ MB or 4,478,566 KB

- The BufferManager automatically triggers the use of a canonical value cache if enabled when more than 25% of the reserve is in use. This can dramatically cut the memory usage in situations where similar value sets are being read through Red Hat JBoss Data Virtualization but does introduce a lookup cost. If you are processing small or highly similar datasets through Red Hat JBoss Data Virtualization and wish to conserve memory, you should consider enabling value caching.

  Memory consumption can be significantly more or less than the nominal target depending upon actual column values and whether value caching is enabled. Large non built-in type objects can exceed their default size estimate. If an out of memory errors occur, then set a lower max-reserve-kb value. Also note that source lob values are held by memory references that are not cleared when a batch is persisted. With heavy LOB usage you should ensure that buffers of other memory associated with lob references are appropriately sized.

- max-processing-kb (default -1) - setting determines the total size in kilobytes of batches that can be guaranteed for use by one active plan and may be in addition to the memory held based on max-reserve-kb. Typical minimum memory required by Red Hat JBoss Data Virtualization when all the active plans are active is #active-plans*max-processing-kb. The default value of -1 will auto-calculate a typical max based upon the max heap available to the VM and max active plans. The auto-calculated value assumes a 64bit architecture and will limit nominal processing batch usage to less than 10% of total memory.

  With default settings including 20 active-plans and an 8GB VM size, then max-processing-kb will be: $(.07 * 8 * 1024)/20^.8 = 537.4$ MB/11 = 52.2 MB or 53,453 KB per plan. This implies a nominal range between 0 and 1060 MB that may be reserved with roughly 53 MB per plan. You should be cautious in adjusting max-processing-kb on your own. Typically it will not need adjusted unless you are seeing situations where plans seem memory constrained with low performing large sorts.

- max-file-size (default 2GB) - Each intermediate result buffer, temporary LOB, and temporary table is stored in its own set of buffer files, where an individual file is limited to max-file-size megabytes. Consider increasing the storage space available to all such files by increasing max-buffer-space, if your installation makes use of internal materialization, makes heavy use of SQL/XML, or processes large row counts.

- processor-batch-size (default 256) - Specifies the target row count of a batch of the query processor. A batch is used to represent both linear data stores, such as saved results, and temporary table pages. Teiid will adjust the processor-batch-size to a working size based upon an estimate of the data width of a row relative to a nominal expectation of 2KB. The base value can be doubled or halved up to three times depending upon the data width estimation. For example a single small fixed width (such as an integer) column batch will have a working size of processor-batch-size * 8 rows. A batch with hundreds of variable width data (such as string) will

have a working size of processor-batch-size / 8 rows. Any increase in the processor batch size beyond the first doubling should be accompanied with a proportional increase in the max-storage-object-size to accommodate the larger storage size of the batches.

Additional considerations are needed if large VM sizes and/or datasets are being used. Red Hat JBoss Data Virtualization has a non-negligible amount of overhead per batch/table page on the order of 100-200 bytes. If you are dealing with datasets with billions of rows and you run into OutOfMemory issues, consider increasing the processor-batch-size to force the allocation of larger batches and table pages. A general guideline would be to double processor-batch-size for every doubling of the effective heap for Red Hat JBoss Data Virtualizationbeyond 4 GB - processor-batch-size = 512 for an 8 GB heap, processor-batch-size = 1024 for a 16 GB heap, etc.

- max-storage-object-size (default 8288608 or 8MB) - The maximum size of a buffered managed object in bytes and represents the individual batch page size. If the processor-batch-size is increased and/or you are dealing with extremely wide result sets (several hundred columns), then the default setting of 8MB for the max-storage-object-size may be too low. The inline-lobs setting also can increase the size of batches containing small lobs. The sizing for max-storage-object-size is in terms of serialized size, which will be much closer to the raw data size than the Java memory footprint estimation used for max-reserved-kb. max-storage-object-size should not be set too large relative to memory-buffer-space since it will reduce the performance of the memory buffer. The memory buffer supports only 1 concurrent writer for each max-storage-object-size of the memory-buffer-space. Note that this value does not typically need to be adjusted unless the processor-batch-size is adjusted, in which case consider adjusting it in proportion to the increase of the processor-batch-size.

  If exceptions occur related to missing batches and "TEIID30001 Max block number exceeded" is seen in the server log, then increase the max-storage-object-size to support larger storage objects. Alternatively you could make the processor-batch-size smaller. memory-buffer-space (default -1) - This controls the amount of on or off heap memory allocated as byte buffers for use by the Red Hat JBoss Data Virtualization buffer manager measured in megabytes. This setting defaults to -1, which automatically determines a setting based upon whether it is on or off heap and the value for max-reserve-kb. The memory buffer supports only 1 concurrent writer for each max-storage-object-size of the memory-buffer-space. Any additional space serves as a cache for the serialized for of batches.

  When left at the default setting the calculated memory buffer space will be approximately 40% of the max-reserve-kb size. If the memory buffer is on heap and the max-reserve-kb is automatically calculated, then the memory buffer space will be subtracted out of the effective max-reserve-kb. If the memory buffer is off heap and the max-reserve-kb is automatically calculated, then its size will be reduced slightly to allow for effectively more working memory in the vm. memory-buffer-off-heap (default false) - Take advantage of the BufferManager memory buffer to access system memory without allocating it to the heap. Setting memory-buffer-off-heap to "true" will allocate the Red Hat JBoss Data Virtualization memory buffer off heap. Depending on whether your installation is dedicated to Red Hat JBoss Data Virtualization and the amount of system memory available, this may be preferable to on-heap allocation. The primary benefit is additional memory usage for Red Hat JBoss Data Virtualization without additional garbage collection tuning. This becomes especially important in situations where more than 32GB of memory is desired for the VM. Note that when using off-heap allocation, the memory must still be available to the java process and that setting the value of memory-buffer-space too high may cause the VM to swap rather than reside in memory. With large off-heap buffer sizes (greater than several gigabytes) you may also need to adjust VM settings.

For Sun VMs the relevant VM settings are MaxDirectMemorySize and UseLargePages. Here is an example:

```
-XX:MaxDirectMemorySize=12g -XX:+UseLargePages
```

Adding this to the VM process arguments would allow for an effective allocation of approximately an 11GB Red Hat JBoss Data Virtualization memory buffer (the memory-buffer-space setting) accounting for any additional direct memory that may be needed by Red Hat JBoss EAP or applications running within EAP.

9.2. SCALABILITY CONSIDERATIONS

Although, you can find information about all JBoss Data Virtualization settings using the Management CLI (see Section 10.1, "JBoss Data Virtualization Settings"), this section provides some additional information about those settings related to scalability.

`buffer-service-processor-batch-size`

Default is 256. This property specifies the maximum row count of a batch sent internally within the query processor. Additional considerations are needed if extremely large VM sizes and or datasets are being used. JBoss Data Virtualization has a non-negligible amount of overhead per batch/table page on the order of 100-200 bytes. Depending on the data types involved, each full batch/table page will represent a variable number of rows (a power of two multiple above or below the processor batch size).

If you are working with extremely large datasets and you run into memory issues, consider increasing the **`buffer-service-processor-batch-size`** property to force the allocation of larger batches and table pages.

`buffer-service-max-storage-object-size`

Default is 8288608 or 8MB. This value is the maximum size of a buffered managed object in bytes and represents the individual batch page size.

If **`buffer-service-processor-batch-size`** is increased or you are dealing with extremely wide result sets, then the default setting of 8MB for the **`buffer-service-max-storage-object-size`** may be too low. The inline LOBS also contribute to this size if the batch contains them. The sizing for **`buffer-service-max-storage-object-size`** is in terms of serialized size, which will be much closer to the raw data size than the Java memory footprint estimation used for **`buffer-service-max-reserve-kb`**.

`buffer-service-max-storage-object-size` should not be set too large relative to **`buffer-service-memory-buffer-space`** since it will reduce the performance of the memory buffer. The memory buffer supports only 1 concurrent writer for each **`buffer-service-max-storage-object-size`** of the **`buffer-service-memory-buffer-space`**.

> **NOTE**
>
> JBoss Data Virtualization temporary tables (also used for internal materialization) can only support 2^31-1 rows per table.

`buffer-service-memory-buffer-space`

Default is -1. This controls the amount of on or off heap memory allocated as byte buffers for use by the JBoss Data Virtualization buffer manager. This setting defaults to -1, which automatically determines a setting based upon whether it is on or off heap and the value for **`buffer-service-max-reserve-kb`**.
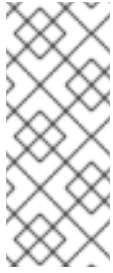
NOTE

When left at the default setting the calculated memory buffer space will be approximately one quarter of the `buffer-service-max-reserve-kb` setting. If the memory buffer is off heap and the `buffer-service-max-reserve-kb` setting is automatically calculated, the memory buffer space will be subtracted out of the effective `buffer-service-max-reserve-kb`.

`buffer-service-memory-buffer-off-heap`

Default is false. Determines whether to take advantage of the buffer manager memory buffer to access system memory without allocating it to the heap. Setting `buffer-service-memory-buffer-off-heap` to `true` will allocate the JBoss Data Virtualization memory buffer off heap. Depending on whether your installation is dedicated to JBoss Data Virtualization and the amount of system memory available, this may be preferred over on-heap allocation.

The primary benefit of off-heap memory is additional memory usage for JBoss Data Virtualization without additional garbage collection tuning. This becomes especially important in situations where more than 32GB of memory is desired for the JVM. Note that when using off-heap allocation, the memory must still be available to the java process and that setting the value of `buffer-service-memory-buffer-space` too high may cause the JVM to swap rather than reside in memory. With large off-heap buffer sizes (greater than several gigabytes) you may also need to adjust JVM settings.

For Sun JVMs the relevant JVM settings are `MaxDirectMemorySize` and `UseLargePages`. For example adding:

```
-XX:MaxDirectMemorySize=12g -XX:+UseLargePages
```

to the JVM process arguments would allow for an effective allocation of an (approximately) 11GB JBoss Data Virtualization memory buffer (the `buffer-service-memory-buffer-space` property) accounting for any additional direct memory that may be needed by JBoss EAP or applications running within it.

9.3. DISK USAGE CONSIDERATIONS

Although, you can find information about all JBoss Data Virtualization settings using the Management CLI (see Section 10.1, "JBoss Data Virtualization Settings"), this section provides some additional information about those settings related to disk usage.

`buffer-service-max-buffer-space`

Default is 51200. For table page and result batches, the buffer manager will limit the number of files that are dedicated to a particular storage size. However, creation of Large Object (LOB) values (for example through SQL/XML) will typically create one file per LOB once the LOB exceeds the allowable in memory size of 8KB. In heavy usage scenarios, consider pointing the buffer directory on a partition that is routinely defragmented. By default, JBoss Data Virtualization will use up to 50GB of disk space. This is tracked in terms of the number of bytes written by JBoss Data Virtualization. For large data sets, you may need to increase the `buffer-service-max-buffer-space` property.

9.4. THREADING CONSIDERATIONS

Although, you can find information about all JBoss Data Virtualization settings using the Management CLI (see Section 10.1, "JBoss Data Virtualization Settings"), this section provides some additional information about those settings related to threading.

max-threads

Default is 64. The query engine has several settings that determine its thread utilization. **max-threads** sets the total number of threads available in the process pool for query engine work (such as processing plans, transaction control operations, and processing source queries).

You should consider increasing the maximum threads on systems with a large number of available processors and/or when it is necessary to issue non-transactional queries involving a large number of concurrent source requests.

max-active-plans

Default is 20. This value should always be smaller than **max-threads**. By default, thread-count-for-source-concurrency is calculated by (max-threads / max_active_plans) * 2 to determine the threads available for processing concurrent source requests for each user query. Increasing the max-active-plans should be considered for workloads with a high number of long running queries and/or systems with a large number of available processors. If memory issues arise from increasing the max-threads and max-active-plans, then consider decreasing the amount of heap held by the buffer manager or decreasing the processor-batch-size to limit the base number of memory rows consumed by each plan.

Increasing **max-active-plans** should be considered for workloads with a high number of long running queries and/or systems with a large number of available processors. If memory issues arise from increasing **max-threads** and **max-active-plans**, then consider decreasing **buffer-service-processor-batch-size** to limit the base number of memory rows consumed by each plan.

thread-count-for-source-concurrency

Default is 0. This value should always be smaller than **max-threads**. This property sets the number of concurrently executing source queries per user request. 0 indicates to use the default calculated value based on 2 * (**max-threads** / **max-active-plans**). Setting this to 1 forces serial execution of all source queries by the processing thread. Any number greater than 1 limits the maximum number of concurrently executing source requests accordingly.

Using the defaults, each user request would be allowed 6 concurrently executing source queries. If the default calculated value is not applicable to your workload, for example, if you have queries that generate more concurrent long running source queries, you should adjust this value.

Also see Section 9.6, "Transport Considerations" for **max-socket-threads**.

9.5. CACHING CONSIDERATIONS

Although, you can find information about all JBoss Data Virtualization settings using the Management CLI (see Section 10.1, "JBoss Data Virtualization Settings"), this section provides some additional information about those settings related to caching.

JBoss Data Virtualization settings regarding cache tuning are divided amongst:

- Resultset Cache Tuning

- Prepared Plan Cache Tuning

Cache statistics can be obtained through the Management Console or AdminShell. The statistics can be used to help tune cache parameters and ensure a hit ratio.

Plans are currently fully held in memory and may have a significant memory footprint. When making extensive use of prepared statements and/or virtual procedures, the size of the plan cache may be increased proportionally to number of gigabytes intended for use by JBoss Data Virtualization.

While the result cache parameters control the cache result entries (such as max number, and eviction), the result batches themselves are accessed through the buffer manager. If the size of the result cache is increased, you may need to tune the buffer manager configuration to ensure there is enough buffer space.

Result set and prepared plan caches have their entries invalidated by data and metadata events. By default, these events are captured by running commands through JBoss Data Virtualization (see the *Red Hat JBoss Data Virtualization Developer Guide* for further customization). JBoss Data Virtualization stores compiled forms of update plans or trigger actions with the prepared plan so that if metadata changes, the changes may take effect immediately.

The default **resultset-cache-max-staleness** for resultset caching is 60 seconds to improve efficiency with rapidly changing sources. Consider decreasing this value to make the resultset cache more consistent with the underlying data. Even with a setting of 0, full transactional consistency is not guaranteed.

> **WARNING**
>
> Disabling or constraining these caches will lead to poor performance.

9.6. TRANSPORT CONSIDERATIONS

Although, you can find information about all JBoss Data Virtualization settings using the Management CLI (see Section 10.1, "JBoss Data Virtualization Settings"), this section provides some additional information about those settings related to transports.

JBoss Data Virtualization provides three transports by default: odbc, jdbc and embedded. Transport settings (such as those listed below) are configured for each.

max-socket-threads

Default is 0. Determines the maximum number of threads dedicated to the initial request processing. Zero indicates to use the system default of maximum available processors. Socket threads handle NIO non-blocking IO operations as well as directly servicing any operation that can run without blocking. For longer running operations, the socket threads queue works with the query engine. (The query engine has two properties that determine its thread utilization: **max-threads** and **max-active-plans**.)

All JDBC/ODBC socket operations are non-blocking, so setting the number of **max-socket-threads** higher than the maximum effective parallelism of the machine should not result in greater performance.

input-buffer-size

Default is 0 which will use the system default. Before adjusting **input-buffer-size** for any of the

transports, keep in mind that each client will create a new socket connection. Increasing this value should only be done if the number of clients is constrained.

`output-buffer-size`

Default is 0 which will use the system default. Before adjusting **`output-buffer-size`** for any of the transports, keep in mind that each client will create a new socket connection. Increasing this value should only be done if the number of clients is constrained.

JDBC clients may need to adjust low-level transport values, in addition to SSL client connection properties via a **`teiid-client-settings.properties`** file placed in the client application's classpath. (An example file can be found within the **`EAP_HOME/modules/system/layers/base/org/jboss/teiid/client/main/teiid-client-VERSION.jar`** file.)

> **NOTE**
>
> Typical installations will not need to have any of these settings adjusted.

9.7. LARGE OBJECTS (LOBS)

Large Objects (LOBs) consist of data. The three main large object runtime data types used by JBoss are:

1. Binary (BLOB)

   - Contains multimedia objects such as images and audio.

2. Character (CLOB)

   - Contains ASCII characters.

3. Extensible Markup Language (XML)

   - Contains textual data.

LOBs and JBoss

The JBoss Data Services Connector API returns a reference to the LOB if allowed by the JBoss Data Services server. The JBoss Data Services server or JDBC driver can then access the data via a stream rather than retrieving the data all at once. This is useful for several reasons:

- Reduces memory usage when returning the result set to the user.

- Improves performance by passing less data in the result set.

- Enables access to LOBs when required rather than assuming that users will always use the LOB data.

- Enables handling of arbitrarily large data values within a fixed JBoss Data Services memory usage.

These benefits are achieved if the Connector itself does not materialize an entire LOB all at once. For example, the JDBC API supports a streaming interface for BLOB and CLOB data.

Source LOB values are typically accessed by reference, rather than having the value copied to a temporary location. Care must be taken to ensure that source LOBs are returned in a memory-safe manner.

LOBs are broken into pieces when being created and streamed. The size of each piece when fetched by the client can be configured.

Cached lobs will be copied rather than relying on the reference to the source lob.

Temporary lobs created by Teiid will be cleaned up when the result set or statement is closed. To rely on implicit garbage collection based cleanup instead of statement close, the Teiid session variable clean_lobs_onclose can be set to false (by issuing the query "SELECT teiid_session_set('clean_lobs_onclose', false)" - which can be done for example via the new connection sql in the datasource definition). This can be used for local client scenarios that relied on the implicit behavior, such as Designer generated REST VDBs.

9.8. LOB CONSIDERATIONS

Although, you can find information about all JBoss Data Virtualization settings using the Management CLI (see Section 10.1, "JBoss Data Virtualization Settings"), this section provides some additional information about those settings related to large objects (LOBs).

`lob-chunk-size-in-kb`

LOBs and XML documents are streamed from the JBoss Data Virtualization server to the JDBC API. Normally, these values are not materialized in the server memory, avoiding potential out-of-memory issues. When using style sheets, or XQuery, whole XML documents must be materialized on the server. Even when using the XMLQuery or XMLTable functions and document projection is applied, memory issues may occur for large documents.

LOBs are broken into pieces when being created and streamed. The maximum size of each piece when fetched by the client can be configured with the `lob-chunk-size-in-kb` property.

The default value is 100. When dealing with extremely large LOBs, you may consider increasing `lob-chunk-size-in-kb` to decrease the amount of round-trips to stream the result. Setting the value too high may cause the server or client to have memory issues.

Source LOB values are typically accessed by reference, rather than having the value copied to a temporary location. Thus care must be taken to ensure that source LOBs are returned in a memory-safe manner. This caution is more for the source driver vendors not to consume VM memory for LOBs.

9.9. OTHER PERFORMANCE TUNING CONSIDERATIONS

Although, you can find information about all JBoss Data Services settings using the Management CLI (see Section 10.1, "JBoss Data Virtualization Settings"), this section provides some additional information about the `max-source-rows` setting.

`max-source-rows`

When using JBoss Data Services in a development environment, you may consider setting `max-source-rows` to a small value (for example, 10000) to prevent large amounts of data from being pulled from sources. Leaving the `exception-on-max-source-rows` property set to `true` will alert the developer through an exception that an attempt was made to retrieve more than the specified number of rows.

PART V. REFERENCE

CHAPTER 10. GENERAL CONFIGURATION

10.1. JBOSS DATA VIRTUALIZATION SETTINGS

The following types of JBoss Data Virtualization settings are available for viewing and modification:

- Buffer service settings

- Cache settings (including result set and prepared plan cache settings)

- Runtime engine deployer settings

- Authorization validator and policy decider settings

- Transport and SSL settings

- Translator settings

To view all of the available settings for JBoss Data Virtualization, run the following command within the Management CLI:

```
/subsystem=teiid:read-resource-description
```

> **NOTE**
>
> To look further into translator and transport (including SSL) settings see Section 10.4, "Managing Transport and SSL Settings Using Management CLI" and Section 10.5, "Managing Translator Settings Using Management CLI".

10.2. VIEWING JBOSS DATA VIRTUALIZATION SETTINGS USING MANAGEMENT CLI

To view all of the current settings for JBoss Data Virtualization, run the following command within the Management CLI:

```
/subsystem=teiid:read-resource
```

To view a particular JBoss Data Virtualization setting, run the following command within the Management CLI:

```
/subsystem=teiid:read-attribute(name=SETTING_NAME)
```

For example:

```
/subsystem=teiid:read-attribute(name=max-active-plans)
```

> **NOTE**
>
> To view current translator and transport (including SSL) settings refer to Section 10.4, "Managing Transport and SSL Settings Using Management CLI" and Section 10.5, "Managing Translator Settings Using Management CLI".

10.3. CHANGING JBOSS DATA VIRTUALIZATION SETTINGS USING MANAGEMENT CLI

To edit a particular JBoss Data Virtualization setting, run the following command within the Management CLI:

```
/subsystem=teiid:write-attribute(name=SETTING_NAME, value=VALUE)
```

NOTE

To change translator and transport (including SSL) settings refer to Section 10.4, "Managing Transport and SSL Settings Using Management CLI" and Section 10.5, "Managing Translator Settings Using Management CLI".

For example:

```
/subsystem=teiid:write-attribute(name=max-active-plans, value=50)
```

For more information about the Management CLI, refer to the Red Hat JBoss Enterprise Application Platform *Administration and Configuration Guide*.
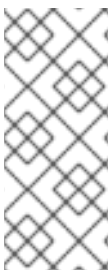
NOTE

After modifying any of these settings, you will be prompted to reload the server. Changes to these settings will not take effect until the server is restarted. You can reload the server by running the **reload** command from within the Management CLI. After the server reloads, you may have to reconnect by running the **connect** command to continue working within the Management CLI.

10.4. MANAGING TRANSPORT AND SSL SETTINGS USING MANAGEMENT CLI

To manage JBoss Data Virtualization transport settings, you can use the same commands as those used for the base JBoss Data Virtualization settings, specifying a particular transport in the command. For example:

```
/subsystem=teiid/transport=TRANSPORT_NAME:read-resource
```

Available transport names are listed under **transport** when you run the following command to output current JBoss Data Virtualization settings:

```
/subsystem=teiid:read-resource
```

10.5. MANAGING TRANSLATOR SETTINGS USING MANAGEMENT CLI

To manage JBoss Data Virtualization translator settings, you can use the same commands as those used for the base JBoss Data Virtualization settings, specifying a particular translator in the command. For example:

```
/subsystem=teiid/translator=TRANSLATOR_NAME:read-resource
```

Available translator names are listed under **translator** when you run the following command to output current JBoss Data Virtualization settings:

```
/subsystem=teiid:read-resource
```

10.6. TRANSPORT SECURITY AUTHENTICATION MODES

The following authentication modes are available:

anonymous

No certificates are exchanged. Settings are not needed for the keystore and truststore properties. The client must have **org.teiid.ssl.allowAnon** set to true (the default) to connect to an anonymous server. Communications are encrypted using the TLS_DH_anon_WITH_AES_128_CBC_SHA SSL cipher suite. This is suitable for most secure intranets.

1-way

Athenticates the server to the client. The server presents a certificate which is signed by the private key stored in the server's keystore. The server's corresponding public key must be in the client's truststore.

2-way

Mutual client and server authentication. The server presents a certificate which is signed by the private key stored in the server's keystore. The server's corresponding public key must be in the client's truststore. Additionally, the client presents a certificate signed by its private key stored in the client's keystore. The client's corresponsing public key must be in the server's truststore.

NOTE

You can use keytool to generate encryption keys; however, you should first consider your local requirements for managing public key cryptography.

10.7. MANAGING CORE CONFIGURATION USING JBOSS MANAGEMENT CONSOLE

1. Log in to the JBoss Management Console.

2. Select the **Profile** tab.

3. From the navigation tree, select **Subsystems → Teiid**.

4. Select one of the following options:

   - Query Engine - From here you can view and configure core query engine properties.

   - Translators - From here you can view, add and remove translators.

   - Transports - From here you can view, add and remove transports.

5. View and modify configuration as necessary.
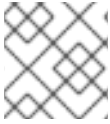
> **NOTE**
>
> If you need help at any stage, select the **Need Help?** link.

> **WARNING**
>
> Some properties require you to restart the server before they will take effect.

10.8. PORTS USED BY RED HAT JBOSS DATA VIRTUALIZATION

Red Hat JBoss Data Virtualization inherits the ports used by JBoss Enterprise Application Platform. You can find a full list here: https://access.redhat.com/documentation/en-US/JBoss_Enterprise_Application_Platform/6/html/Administration_and_Configuration_Guide/Network_Ports

In addition, to these, ports 31000 and 35432 must also be kept open.

10.9. CHANGE THE DEFAULT JDBC PORT USING MANAGEMENT CONSOLE

1. Login to the Management Console.

2. **Navigate to the `Socket Binding` panel in the Management Console**

   a. ■ **Standalone Mode**
      Select the **`Profile`** tab from the top-right of the console.

      ■ **Domain Mode**

         i. Select the **`Profiles`** tab from the top-right of the console.

         ii. Select the appropriate profile from the drop-down box in the top left.

         iii. Expand the **Subsystems** menu on the left of the console.

   b. Select **General Configuration → Socket Binding** from the menu on the left of the console.

3. **Modify the port number**

   a. Select the **`teiid-jdbc`** configuration.

   b. Select the **`Edit`** button.

   c. Set the **`Port`** to the new port number.

   d. Select **`Save`**.

10.10. SYSTEM PROPERTIES

Some behavior can be configured via system properties, rather than configuration files. A typical place to set system properties for JBoss Data Virtualization is in the *EAP_HOME*/bin/*standalone*.conf. A property setting has the format **-D*property=value***.

Table 10.1. System Properties

Setting	Description	Default Value
org.teiid.allowNanInfinity	Set to true to allow numeric functions to return NaN (Not A Number) and +-Infinity. Note that these values are not covered by the SQL specification.	Defaults to false.
org.teiid.useValueCache	Set to true to enable the canonical value cache. Value caching is used dynamically when buffer memory is running low to reuse identical values, reducing the memory consumed by JBoss Data Virtualization. However, there is a computation cost associated with the cache lookup, so enabling this setting is not appropriate for installations handling large volumes of dissimilar data.	Defaults to false.
org.teiid.ansiQuotedIdentifiers	Set to false to emulate prior behavior of treating double quoted values without leading identifier parts as string literals, which is not expected by the SQL specification.	Defaults to true.
org.teiid.subqueryUnnestDefault	Set to true to aggressively unnest subquery IN and EXISTS predicates. If possible, the predicate will be unnested to a traditional join and will be eligible for dependent join planning. If a traditional join is not possible (such as with NOT IN) a merge join version of the semijoin or antijoin will be considered based upon the costing information available.	Defaults to false.
org.teiid.ODBCPacketSize	Target size in bytes of the ODBC results buffer. This is not a hard maximum, LOBS and wide rows may use larger buffers.	Defaults to 307200.

Setting	Description	Default Value
org.teiid.decimalAsDouble	Set to true to parse exact fixed point literals (for example, 1.0) as double values rather than as decimal/BigDecimal values and to return a double value from the AVG function for integral values in the same way as previous versions.	Defaults to false.
org.teiid.comparableLobs	Set to true to allow BLOB and CLOB column values to be comparable in JBoss Data Virtualization. Source type metadata will determine if the comparison can be pushed down.	Defaults to false.
org.teiid.comparableObject	Set to true to allow object column values to be comparable in JBoss Data Virtualization. Source type metadata will determine if the comparison can be pushed down. The object instances are expected to correctly implement **java.lang.Comparable.compareTo**. If the instance object is not **Comparable**, then **ClassCastExceptions** may the thrown.	Defaults to false.
org.teiid.padSpace	Set to true to compare strings as if PAD SPACE collation is being used. This means strings will be right padded to the same length for comparison. If this property is set, it is not necessary to use the **trimStrings** translator option.	Defaults to false.
org.teiid.collationLocale	Set to a Java locale string language[_country[_varient]], where language, country, and variant are two letter codes - see java.util.Locale for more on valid codes. Note that even if **org.teiid.comparableLobs** is set, CLOB values will not be compared using the locale collator.	Not set by default, which means that Java's natural (UTF-16) string comparison will be used.

Setting	Description	Default Value
org.teiid.clientVdbLoadTimeoutMillis	The default amount of time a client (currently only local clients) will wait to make a connection to an active VDB before throwing an exception. Clients may override this setting via the **waitForLoad** connection property.	Defaults to 5 minutes.
org.teiid.enDateNames	Set to true to use English month and day names for the system function **dayName** and **monthName**, rather than returning names from the Java default locale. Prior to this release, **dayName** and **monthName** always returned English names.	Defaults to false.
org.teiid.pushdownDefaultNullOrder	Set to true to mimic prior release behavior of pushing the default null order of nulls low if the source has a different default null order and supports explicit null ordering.	Defaults to false.
org.teiid.implicitMultiSourceJoin	Set to false to disable prior release behavior of implicitly partitioning joins between multi-source tables. When set to false an explicit predicate such as **tbl1.source_name = tbl2.source_name** is required to partition the results of the join.	Defaults to true.
org.teiid.joinPrefetchBatches	Sets the number of batches that can be pre-fetched/buffered for join processing that may otherwise be streamed from the source. Consider increasing when you have slow ALREADY_SORTED sources participating in a non-dependent join that return significantly more batches. Note however that increasing this value can lead to an increase in disk utilization to perform the buffering.	Defaults to 10.

Setting	Description	Default Value
org.teiid.maxStringLength	Sets the nominal maximum length of strings in JBoss Data Virtualization. Most operations will truncate strings that are larger than this value. Setting this value can also adjust the maximum size of LOB bytes held in memory. Note that sources may not appropriately handle string values that are larger than what the source supports. ⚠ **WARNING** Strings are held entirely in memory. Do not set this value too high as you may experience out of memory errors.	Defaults to 4000.
org.teiid.calendarTimestampDiff	Set to false to use the timestampdiff behaviour from previous versions. Note that using the old behavior can result in differing results between pushed and non-pushed versions of timestampdiff for intervals greater than seconds. This is because sources use the date part and not approximate interval differences.	Defaults to true

Setting	Description	Default Value
org.teiid.widenComparisonToString	This defaults to false. Set to true to enable widening of values to string in comparisons, (which was the default behavior earlier releases). For example, with this set to false, a timestamp_col < 'a' will produce an exception whereas when it is set to true it evaluates cast(timestamp_col as string) < `a'.	Defaults to false.

The following properties are provided for PostgreSQL compatibility.

Table 10.2. System Properties for PostgreSQL Compatibility

Setting	Description	Default Value
org.teiid.iso8601Week	Set to true to use ISO 8601 rules for week calculations regardless of the locale. When set to true, the **dayOfWeek** function will begin with 1 for MONDAY rather than SUNDAY, and the week function will require that week 1 of a year contains the year's first Thursday.	Defaults to false.
org.teiid.backslashDefaultMatchEscape	Set to true to use '\' as the default escape character for LIKE and SIMILAR TO predicates when no escape is specified. Otherwise JBoss Data Virtualization assumes the SQL specification compliant behavior of treating each non-wildcard character as an exact match character.	Defaults to false.
org.teiid.honorDeclareFetchTxn	When set to false, the wrapping begin/commit of a UseDeclareFetch cursor will be ignored as Red Hat JBoss Data Virtualization does not require a transaction.	Defaults to false.

Setting	Description	Default Value
org.teiid.maxSessionBufferSizeEstimate	Set this to your desired size in bytes in order to limit the amount of buffer resources (heap and disk) consumed by a single session's tuple buffers and table structures. (This is based upon the memory footprint estimate and may not correspond exactly to heap or disk consumption.)	2^63 - 1
org.teiid.pgVersion	This is the value that is reported by the server_version function.	"PostgreSQL 8.2"

NOTE

These properties for PostgreSQL compatibility affect JBoss Data Virtualization globally, not just through the ODBC transport.

10.11. TEIID MANAGEMENT CLI

The AS CLI is a command line based administrative and monitoring tool for Teiid. AdminShell provides a binding into the Groovy scripting language and higher level methods that are often needed when interacting with Teiid. It is still useful to know the underlying CLI commands in many circumstances. The below is a series useful CLI commands for administering a Teiid Server.

VDB Operations

```
deploy adminapi-test-vdb.xml
undeploy adminapi-test-vdb.xml

/subsystem=teiid:restart-vdb(vdb-name=AdminAPITestVDB, vdb-version=1,
model-names=TestModel)

/subsystem=teiid:list-vdbs()
/subsystem=teiid:get-vdb(vdb-name=AdminAPITestVDB,vdb-version=1)
/subsystem=teiid:change-vdb-connection-type(vdb-name=AdminAPITestVDB, vdb-
version=1,connection-type=ANY)

/subsystem=teiid:add-data-role(vdb-name=AdminAPITestVDB, vdb-version=1,
data-role=TestDataRole, mapped-role=test)
/subsystem=teiid:remove-data-role(vdb-name=AdminAPITestVDB, vdb-version=1,
data-role=TestDataRole, mapped-role=test)
```

Source Operations

```
/subsystem=teiid:add-source(vdb-name=AdminAPITestVDB, vdb-version=1,
source-name=text-connector-test, translator-name=file, model-
name=TestModel, ds-name=java:/test-file)
/subsystem=teiid:remove-source(vdb-name=AdminAPITestVDB, vdb-version=1,
source-name=text-connector-test, model-name=TestModel)
```

```
/subsystem=teiid:update-source(vdb-name=AdminAPITestVDB, vdb-version=1,
source-name=text-connector-test, translator-name=file, ds-
name=java:/marketdata-file)
```

Translator Operations

```
/subsystem=teiid:list-translators()
/subsystem=teiid:get-translator(translator-name=file)
/subsystem=teiid:read-translator-properties(translator-
name=file,type=OVERRIDE)
/subsystem=teiid:read-rar-description(rar-name=file)
```

Runtime Operations

```
/subsystem=teiid:workerpool-statistics()

/subsystem=teiid:cache-types()
/subsystem=teiid:clear-cache(cache-type=PREPARED_PLAN_CACHE)
/subsystem=teiid:clear-cache(cache-type=QUERY_SERVICE_RESULT_SET_CACHE)
/subsystem=teiid:clear-cache(cache-type=PREPARED_PLAN_CACHE, vdb-
name=AdminAPITestVDB,vdb-version=1)
/subsystem=teiid:clear-cache(cache-type=QUERY_SERVICE_RESULT_SET_CACHE,
vdb-name=AdminAPITestVDB,vdb-version=1)
/subsystem=teiid:cache-statistics(cache-type=PREPARED_PLAN_CACHE)
/subsystem=teiid:cache-statistics(cache-
type=QUERY_SERVICE_RESULT_SET_CACHE)

/subsystem=teiid:engine-statistics()

/subsystem=teiid:list-sessions()
/subsystem=teiid:terminate-session(session=sessionid)

/subsystem=teiid:list-requests()
/subsystem=teiid:cancel-request(session=sessionId, execution-id=1)
/subsystem=teiid:list-requests-per-session(session=sessionId)
/subsystem=teiid:list-transactions()

/subsystem=teiid:mark-datasource-available(ds-name=java:/accounts-ds)

/subsystem=teiid:get-query-plan(session=sessionid,execution-id=1)
```

CHAPTER 11. DIRECTORY STRUCTURE

11.1. RED HAT JBOSS DATA VIRTUALIZATION FILE STRUCTURE

The following shows the contents of the Red Hat JBoss Data Virtualization deployment within a Red Hat JBoss EAP instance:

```
EAP-6.4.0/
├── appclient
│   └── configuration
│       ├── appclient.xml
│       └── logging.properties
├── bin
│   ├── add-user.bat
│   ├── add-user.properties
│   ├── add-user.sh
│   ├── appclient.bat
│   ├── appclient.conf
│   ├── appclient.conf.bat
│   ├── appclient.sh
│   ├── client
│   │   ├── jboss-cli-client.jar
│   │   ├── jboss-client.jar
│   │   ├── README-CLI-JCONSOLE.txt
│   │   └── README-EJB-JMS.txt
│   ├── domain.bat
│   ├── domain.conf
│   ├── domain.conf.bat
│   ├── domain.sh
│   ├── init.d
│   │   ├── jboss-as.conf
│   │   ├── jboss-as-domain.sh
│   │   └── jboss-as-standalone.sh
│   ├── jboss-cli.bat
│   ├── jboss-cli-logging.properties
│   ├── jboss-cli.sh
│   ├── jboss-cli.xml
│   ├── jconsole.bat
│   ├── jconsole.sh
│   ├── jdr.bat
│   ├── jdr.sh
│   ├── product.conf
│   ├── run.bat
│   ├── run.sh
│   ├── standalone.bat
│   ├── standalone.conf
│   ├── standalone.conf.bat
│   ├── standalone.sh
│   ├── teiid-oauth-util.bat
│   ├── teiid-oauth-util.sh
│   ├── vault.bat
│   ├── vault.sh
│   ├── wsconsume.bat
│   ├── wsconsume.sh
│   ├── wsprovide.bat
```

```
│       └── wsprovide.sh
├── bundles
│   └── system
│       └── layers
│           └── base
├── cli-scripts
│   ├── disable-welcome-root.cli
│   ├── disable-welcome-root-domain.cli
│   ├── ModeShape-domain.cli
│   ├── ModeShape-ds.properties
│   ├── ModeShape-standalone.cli
│   ├── teiid-add-database-logger.cli
│   ├── teiid-add-database-logger-domain.cli
│   ├── teiid-dashboard-add_datasource.cli
│   ├── teiid-dashboard-domain-add_datasource.cli
│   ├── teiid-deploy-dashboard.cli
│   ├── teiid-deploy-dashboard-domain.cli
│   ├── teiid-domain-auditcommand-logging.cli
│   ├── teiid-domain-install-ds-builder-war.cli
│   ├── teiid-domain-install-vdb-builder-war.cli
│   ├── teiid-domain-mode-install.cli
│   ├── teiid-logger-ds.properties
│   ├── teiid-modeshape-domain.cli
│   ├── teiid-modeshape-standalone.cli
│   ├── teiid-standalone-auditcommand-logging.cli
│   ├── teiid-standalone-ha-mode-install.cli
│   ├── teiid-standalone-install-ds-builder-war.cli
│   ├── teiid-standalone-install-vdb-builder-war.cli
│   └── teiid-standalone-mode-install.cli
├── dataVirtualization
│   ├── dataServiceBuilder
│   │   ├── komodo-rest.war
│   │   ├── vdb-bench-doc.war
│   │   └── vdb-bench-war.war
│   ├── jdbc
│   │   ├── modeshape-client-3.8.4.GA-redhat-64-12.jar
│   │   ├── teiid-8.12.11.6_4-redhat-64-12-jdbc.jar
│   │   └── teiid-hibernate-dialect-8.12.11.6_4-redhat-64-12.jar
│   ├── logging
│   │   └── database-service.jar
│   ├── rest-client
│   │   ├── modeshape-client-3.8.4.GA-redhat-64-12.jar
│   │   ├── README.txt
│   │   ├── restclient.bat
│   │   └── restclient.sh
│   ├── teiid-adminshell
│   │   └── teiid-8.12.11.6_4-redhat-64-12-adminshell-dist.zip
│   ├── teiid-dashboard
│   │   └── teiid-dashboard-builder.war
│   │       ├── admin
│   │       ├── ckeditor
│   │       ├── common
│   │       ├── components
│   │       ├── configuration
│   │       ├── error.jsp
│   │       ├── favicon.ico
```

```
│   │           ├── images
│   │           ├── index.jsp
│   │           ├── js
│   │           ├── js-api
│   │           ├── login_failed.jsp
│   │           ├── login.jsp
│   │           ├── META-INF
│   │           ├── not_authorized.jsp
│   │           ├── panels
│   │           ├── redhat
│   │           ├── robots.txt
│   │           ├── section
│   │           ├── system
│   │           ├── templates
│   │           └── WEB-INF
│   └── vdb
│       ├── ModeShape.vdb
│       ├── teiid-odata.war
│       └── teiid-olingo-odata4.war
├── docs
│   ├── examples
│   │   └── configs
│   │       ├── standalone-genericjms.xml
│   │       ├── standalone-hornetq-colocated.xml
│   │       ├── standalone-jts.xml
│   │       ├── standalone-minimalistic.xml
│   │       ├── standalone-osgi-only.xml
│   │       ├── standalone-picketlink.xml
│   │       └── standalone-xts.xml
│   ├── licenses
│   │   ├── apache software license, version 2.0 - apache-2.0.txt
│   │   ├── common development and distribution license - cddl.txt
│   │   ├── common public license, version 1.0 - cpl-1.0.txt
│   │   ├── eclipse distribution license, version 1.0 - edl-1.0.txt
│   │   ├── eclipse public license, version 1.0 - epl-1.0.txt
│   │   ├── gnu general public license, version 2 - gpl-2.0.txt
│   │   ├── gnu general public license, version 2 with the classpath
exception - gpl-2.0-ce.txt
│   │   ├── gnu lesser general public license, version 2.1 - lgpl-2.1.txt
│   │   ├── gnu library general public license, version 2 - lgpl-2.0.txt
│   │   ├── h2 license, version 1.0 - h2.txt
│   │   ├── jcip-cc-by-2.5.txt
│   │   ├── jdom license - jdom-1.0.txt
│   │   ├── mozilla public license, version 1.1 - mpl-1.1.txt
│   │   ├── osgi-1.0.txt
│   │   ├── the bsd license - bsd.txt
│   │   ├── the jython license - license.html
│   │   ├── the mit license - mit.txt
│   │   ├── the werken company license - license.html
│   │   └── w3c software notice and license - w3c.txt
│   ├── licenses-datavirt
│   │   ├── com.amazonaws,aws-java-sdk,ApacheLicense,Version2.0
│   │   ├── com.beust,jcommander,TheApacheSoftwareLicense,Version2.0
│   │   ├── com.codahale.metrics,metrics-core,ApacheLicense2.0
│   │   ├── com.couchbase.client,core-
io,TheApacheSoftwareLicense,Version2.0
```

```
|   |       ├── com.couchbase.client,java-
client,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.datastax.cassandra,cassandra-driver-core,Apache2
|   |       ├── com.drewnoakes,metadata-
extractor,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.fasterxml,aalto-xml,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.fasterxml.jackson.core,jackson-
annotations,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.fasterxml.jackson.core,jackson-
core,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.fasterxml.jackson.core,jackson-
databind,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.fasterxml.jackson.dataformat,jackson-dataformat-
xml,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.fasterxml.jackson.dataformat,jackson-dataformat-
yaml,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.fasterxml.jackson.datatype,jackson-datatype-
joda,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.fasterxml.jackson.jaxrs,jackson-jaxrs-
base,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.fasterxml.jackson.jaxrs,jackson-jaxrs-json-
provider,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.fasterxml.jackson.module,jackson-module-jaxb-
annotations,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.force.api,force-partner-api,FreeBSDLicense
|   |       ├── com.force.api,force-wsc,FreeBSDLicense
|   |       ├── com.github.junrar,junrar,UnRarLicense
|   |       ├── com.github.virtuald,curvesapi,CDDL+GPLLicense
|   |       ├── com.google.code.findbugs,annotations,GNULesserPublicLicense
|   |       ├── com.google.code.gson,gson,TheApacheSoftwareLicense,Version2.0
|   |       ├── com.googlecode.json-simple,json-
simple,TheApacheSoftwareLicense,Version2.0
|   |       ├──
com.googlecode.juniversalchardet,juniversalchardet,MozillaPublicLicense1.1
(MPL1.1)
|   |       ├── com.googlecode.mp4parser,isoparser,ApacheSoftwareLicense-
Version2.0
|   |       ├── com.google.guava,guava,TheApacheSoftwareLicense,Version2.0
|   |       ├──
com.healthmarketscience.jackcess,jackcess,ApacheLicense,Version2.0
|   |       ├── com.healthmarketscience.jackcess,jackcess-
encrypt,ApacheLicense,Version2.0
|   |       ├── com.jcraft,jsch,RevisedBSD
|   |       ├── com.mchange,c3p0,GNULesserGeneralPublicLicense,Version2.1
|   |       ├── commons-cli,commons-cli,TheApacheSoftwareLicense,Version2.0
|   |       ├── commons-codec,commons-
codec,TheApacheSoftwareLicense,Version2.0
|   |       ├── commons-collections,commons-
collections,TheApacheSoftwareLicense,Version2.0
|   |       ├── commons-fileupload,commons-
fileupload,TheApacheSoftwareLicense,Version2.0
|   |       ├── commons-httpclient,commons-httpclient,ApacheLicense
|   |       ├── commons-io,commons-io,TheApacheSoftwareLicense,Version2.0
|   |       ├── commons-jxpath,commons-
jxpath,TheApacheSoftwareLicense,Version2.0
|   |       ├── commons-lang,commons-lang,TheApacheSoftwareLicense,Version2.0
```

```
│   │       ├── commons-logging,commons-logging,ApacheLicense,Version2.0
│   │       ├── commons-logging,commons-logging-api,ApacheLicense,Version2.0
│   │       ├── commons-logging,commons-logging-
api,TheApacheSoftwareLicense,Version2.0
│   │       ├── commons-logging,commons-
logging,TheApacheSoftwareLicense,Version2.0
│   │       ├── commons-pool,commons-pool,TheApacheSoftwareLicense,Version2.0
│   │       ├── commons-vfs,commons-vfs,ApacheLicense,Version2.0
│   │       ├── commons-vfs,commons-vfs,TheApacheSoftwareLicense,Version2.0
│   │       ├── com.pff,java-libpst,TheApacheSoftwareLicense,Version2.0
│   │       ├── com.rometools,rome,TheApacheSoftwareLicense,Version2.0
│   │       ├── com.rometools,rome-utils,TheApacheSoftwareLicense,Version2.0
│   │       ├── com.squareup,protoparser,Apache2.0
│   │       ├── com.vividsolutions,jts,LesserGeneralPublicLicense(LGPL)
│   │       ├── dom4j,dom4j,BSD2.0+
│   │       ├── edu.ucar,cdm,(MIT-style)netCDFClibrarylicense
│   │       ├── edu.ucar,grib,(MIT-style)netCDFClibrarylicense
│   │       ├── edu.ucar,httpservices,(MIT-style)netCDFClibrarylicense
│   │       ├── edu.ucar,jj2000,TheApacheSoftwareLicense,Version2.0
│   │       ├── edu.ucar,netcdf4,(MIT-style)netCDFClibrarylicense
│   │       ├── edu.ucar,udunits,(MIT-style)netCDFClibrarylicense
│   │       ├── io.hawt,hawtio-core,TheApacheSoftwareLicense,Version2.0
│   │       ├── io.hawt,hawtio-git,TheApacheSoftwareLicense,Version2.0
│   │       ├── io.hawt,hawtio-system,TheApacheSoftwareLicense,Version2.0
│   │       ├── io.hawt,hawtio-util,TheApacheSoftwareLicense,Version2.0
│   │       ├── io.swagger,swagger-annotations,ApacheLicense2.0
│   │       ├── io.swagger,swagger-core,ApacheLicense2.0
│   │       ├── io.swagger,swagger-jaxrs,ApacheLicense2.0
│   │       ├── io.swagger,swagger-models,ApacheLicense2.0
│   │       ├── jakarta-regexp,jakarta-regexp,ApacheLicense,Version2.0
│   │       ├──
javax.activation,activation,COMMONDEVELOPMENTANDDISTRIBUTIONLICENSE(CDDL)V
ersion1.0
│   │       ├── javax.jcr,jcr,AdobeDayJCRLicense
│   │       ├── javax.jcr,jcr,DaySpecificationLicense
│   │       ├── javax.jcr,jcr,DaySpecificationLicenseaddendum
│   │       ├── javax.measure,jsr-275,JScienceBSDLicense
│   │       ├── javax.measure,jsr-275,SpecificationLicense
│   │       ├── javax.validation,validation-api,ApacheLicense,Version2.0
│   │       ├── javax.ws.rs,jsr311-api,CDDLLicense
│   │       ├── javax.xml.stream,stax-
api,COMMONDEVELOPMENTANDDISTRIBUTIONLICENSE(CDDL)Version1.0
│   │       ├── javax.xml.stream,stax-api,GNUGeneralPublicLibrary
│   │       ├── jline,jline,TheBSDLicense
│   │       ├── joda-time,joda-time,Apache2
│   │       ├── licenses.css
│   │       ├── licenses.html
│   │       ├── licenses.xml
│   │       ├── log4j,log4j,TheApacheSoftwareLicense,Version2.0
│   │       ├── net.jcip,jcip-annotations,CreativeCommonsAttributionlicense2.5
│   │       ├── net.oauth.core,oauth,TheApacheSoftwareLicense,Version2.0
│   │       ├── net.sf.opencsv,opencsv,Apache2
│   │       ├── org.antlr,antlr4-runtime,BSD3-ClauseLicense
│   │       ├── org.antlr,antlr-runtime,BSDlicence
│   │       ├── org.antlr,stringtemplate,BSDlicence
│   │       ├── org.apache.accumulo,accumulo-core,ApacheLicense,Version2.0
```

```
│   │       ├── org.apache.accumulo,accumulo-fate,ApacheLicense,Version2.0
│   │       ├── org.apache.accumulo,accumulo-trace,ApacheLicense,Version2.0
│   │       ├── org.apache.avro,avro,PublicDomain
│   │       ├── org.apache.chemistry.opencmis,chemistry-opencmis-commons-
api,Apache2
│   │       ├── org.apache.chemistry.opencmis,chemistry-opencmis-commons-
impl,Apache2
│   │       ├── org.apache.chemistry.opencmis,chemistry-opencmis-server-
bindings,Apache2
│   │       ├── org.apache.chemistry.opencmis,chemistry-opencmis-server-
support,Apache2
│   │       ├── org.apache.commons,commons-
collections4,ApacheLicense,Version2.0
│   │       ├── org.apache.commons,commons-compress,ApacheLicense,Version2.0
│   │       ├── org.apache.commons,commons-
csv,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.commons,commons-exec,ApacheLicense,Version2.0
│   │       ├── org.apache.commons,commons-
lang3,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.commons,commons-
vfs2,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.cxf,cxf-api,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.cxf,cxf-core,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.cxf,cxf-rt-bindings-
xml,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.cxf,cxf-rt-core,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.cxf,cxf-rt-frontend-
jaxrs,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.cxf,cxf-rt-rs-
client,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.cxf,cxf-rt-rs-security-oauth2-
saml,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.cxf,cxf-rt-rs-security-
oauth2,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.cxf,cxf-rt-rs-security-
oauth,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.cxf,cxf-rt-transports-http-
hc,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.cxf,cxf-rt-transports-
http,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache-extras.beanshell,bsh,ApacheLicense,Version2.0
│   │       ├──
org.apache.felix,org.apache.felix.fileinstall,TheApacheSoftwareLicense,Ver
sion2.0
│   │       ├──
org.apache.felix,org.apache.felix.framework,ApacheLicense,Version2.0
│   │       ├── org.apache.geronimo.specs,geronimo-
javamail_1.4_spec,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.httpcomponents,httpasyncclient,ApacheLicense
│   │       ├──
org.apache.httpcomponents,httpasyncclient,ApacheLicense,Version2.0
│   │       ├── org.apache.httpcomponents,httpclient,ApacheLicense,Version2.0
│   │       ├── org.apache.httpcomponents,httpcore,ApacheLicense,Version2.0
│   │       ├── org.apache.httpcomponents,httpcore-nio,ApacheLicense
│   │       ├── org.apache.httpcomponents,httpcore-
nio,ApacheLicense,Version2.0
```

```
│   │       ├── org.apache.httpcomponents,httpmime,ApacheLicense,Version2.0
│   │       ├── org.apache.james,apache-mime4j-
core,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.james,apache-mime4j-
dom,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.lucene,lucene-core,Apache2
│   │       ├── org.apache.lucene,lucene-facet,Apache2
│   │       ├── org.apache.maven.scm,maven-scm-
api,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.maven.scm,maven-scm-provider-svn-
commons,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.maven.scm,maven-scm-provider-
svnexe,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.olingo,odata-client-
api,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.olingo,odata-client-
core,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.olingo,odata-commons-
api,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.olingo,odata-commons-
core,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.olingo,odata-server-
api,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.olingo,odata-server-core-
ext,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.olingo,odata-server-
core,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.opennlp,opennlp-
maxent,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.opennlp,opennlp-
tools,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.pdfbox,fontbox,ApacheLicense,Version2.0
│   │       ├── org.apache.pdfbox,jempbox,ApacheLicense,Version2.0
│   │       ├── org.apache.pdfbox,pdfbox,ApacheLicense,Version2.0
│   │       ├── org.apache.pdfbox,pdfbox-debugger,ApacheLicense,Version2.0
│   │       ├── org.apache.pdfbox,pdfbox-tools,ApacheLicense,Version2.0
│   │       ├── org.apache.poi,poi-ooxml-
schemas,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.poi,poi-ooxml,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.poi,poi,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.tika,tika-core,ApacheLicense,Version2.0
│   │       ├── org.apache.tika,tika-parsers,ApacheLicense,Version2.0
│   │       ├── org.apache.ws.xmlschema,xmlschema-
core,TheApacheSoftwareLicense,Version2.0
│   │       ├──
org.apache.xmlbeans,xmlbeans,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.apache.zookeeper,zookeeper,ApacheLicense,Version2.0
│   │       ├── org.codehaus.jackson,jackson-core-
asl,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.codehaus.jackson,jackson-mapper-
asl,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.codehaus.jettison,jettison,PublicDomain
│   │       ├── org.codehaus.plexus,plexus-
utils,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.codehaus.woodstox,stax2-api,TheBSDLicense
│   │       ├── org.codehaus.woodstox,woodstox-core-
```

```
asl,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.codelibs,jhighlight,CDDL,v1.0
│   │       ├── org.codelibs,jhighlight,LGPL,v2.1orlater
│   │       ├── org.eclipse.jgit,org.eclipse.jgit,EclipsePublicLicense-v1.0
│   │       ├── org.fusesource.jansi,jansi,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.gagravarr,vorbis-java-
core,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.gagravarr,vorbis-java-
tika,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.hamcrest,hamcrest-core,NewBSDLicense
│   │       ├── org.hibernate.common,hibernate-commons-
annotations,GNULESSERGENERALPUBLICLICENSE
│   │       ├── org.hibernate,hibernate-search-
engine,GNULesserGeneralPublicLicense
│   │       ├── org.hibernate,hibernate-validator,ApacheLicense,Version2.0
│   │       ├── org.infinispan,infinispan-cachestore-
leveldb,GNULesserGeneralPublicLicense
│   │       ├── org.infinispan,infinispan-core,GNULesserGeneralPublicLicense
│   │       ├── org.iq80.leveldb,leveldb,ApacheLicense2.0
│   │       ├── org.iq80.leveldb,leveldb-api,ApacheLicense2.0
│   │       ├── org.itadaki,bzip2,MITLicense(MIT)
│   │       ├── org,jaudiotagger,LGPL
│   │       ├── org.javassist,javassist,ApacheLicense2.0
│   │       ├── org.javassist,javassist,LGPL2.1
│   │       ├── org.jboss.aesh,aesh,EclipseLicense,Version1.0
│   │       ├── org.jboss.as,jboss-as-build-config,lgpl
│   │       ├── org.jboss.as,jboss-as-cli,lgpl
│   │       ├── org.jboss.as,jboss-as-controller-client,lgpl
│   │       ├── org.jboss.as,jboss-as-controller,lgpl
│   │       ├── org.jboss.as,jboss-as-protocol,lgpl
│   │       ├── org.jboss.as,jboss-as-version,lgpl
│   │       ├── org.jboss.dashboard-builder,dashboard-
commons,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.jboss.dashboard-builder,dashboard-displayer-
api,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.jboss.dashboard-builder,dashboard-displayer-
core,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.jboss.dashboard-builder,dashboard-provider-
api,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.jboss.dashboard-builder,dashboard-provider-
core,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.jboss.dashboard-builder,dashboard-provider-
csv,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.jboss.dashboard-builder,dashboard-provider-
sql,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.jboss.dashboard-builder,dashboard-
security,TheApacheSoftwareLicense,Version2.0
│   │       ├── org.jboss,jboss-dmr,PublicDomain
│   │       ├── org.jboss,jboss-vfs,asl
│   │       ├── org.jboss.logging,jboss-logging,ApacheLicense,version2.0
│   │       ├── org.jboss.logmanager,jboss-logmanager,ApacheLicenseVersion2.0
│   │       ├── org.jboss.marshalling,jboss-marshalling,PublicDomain
│   │       ├── org.jboss.marshalling,jboss-marshalling-river,PublicDomain
│   │       ├── org.jboss.modules,jboss-modules,lgpl
│   │       ├── org.jboss.msc,jboss-msc,PublicDomain
│   │       ├── org.jboss.oreva,common,TheApacheSoftwareLicense,Version2.0
```

```
│    │    ├── org.jboss.oreva,odata-core,TheApacheSoftwareLicense,Version2.0
│    │    ├── org.jboss.remoting3,jboss-remoting,LGPL2.1
│    │    ├── org.jboss.remotingjmx,remoting-jmx,LGPL2.1
│    │    ├── org.jboss.sasl,jboss-sasl,PublicDomain
│    │    ├── org.jboss.spec.javax.transaction,jboss-transaction-
api_1.1_spec,CommonDevelopmentandDistributionLicense
│    │    ├── org.jboss.spec.javax.transaction,jboss-transaction-
api_1.1_spec,GNUGeneralPublicLicense,Version2withtheClasspathException
│    │    ├── org.jboss,staxmapper,PublicDomain
│    │    ├── org.jboss.teiid.connectors,connector-
accumulo,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-
cassandra,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-
couchbase,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-
file,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-
google,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-
infinispan.6,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-infinispan-
dsl,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-infinispan-
hotrod,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-
ldap,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-
mongodb,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-salesforce-
34,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-
salesforce,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-
simpledb,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-
solr,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,connector-
ws,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,couchbase-
api,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,document-
api,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,google-
api,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,infinispan-
api,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,mongodb-
api,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,simpledb-
api,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,translator-
accumulo,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,translator-amazon-
s3,GNULesserGeneralPublicLicense
│    │    ├── org.jboss.teiid.connectors,translator-
```

```
cassandra,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
couchbase,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
excel,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
file,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
google,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
hbase,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
hive,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-infinispan-
cache,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-infinispan-
dsl,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-infinispan-
hotrod,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
jdbc,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
jpa,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
ldap,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
loopback,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
mongodb,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
object,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
odata4,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
odata,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
olap,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
prestodb,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-salesforce-
34,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
salesforce,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
simpledb,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
solr,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.connectors,translator-
ws,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.extensions,database-logging-
appender,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.extensions,database-
service,GNULesserGeneralPublicLicense
│   │      ├── org.jboss.teiid.modeshape,teiid-modeshape-
core,TheApacheSoftwareLicense,Version2.0
│   │      ├── org.jboss.teiid.modeshape,teiid-modeshape-sequencer-
```

```
dataservice,TheApacheSoftwareLicense,Version2.0
|   |   ├── org.jboss.teiid.modeshape,teiid-modeshape-sequencer-
ddl,TheApacheSoftwareLicense,Version2.0
|   |   ├── org.jboss.teiid.modeshape,teiid-modeshape-sequencer-
vdb,TheApacheSoftwareLicense,Version2.0
|   |   ├── org.jboss.teiid,teiid-admin,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-adminshell,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-api,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-client,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-common-
core,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-engine,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-hibernate-
dialect,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-jboss-
admin,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-jboss-
integration,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-jboss-
security,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-metadata,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-olingo-
common,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-olingo,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-olingo-
patches,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.teiid,teiid-runtime,GNULesserGeneralPublicLicense
|   |   ├── org.jboss.threads,jboss-threads,PublicDomain
|   |   ├── org.jboss.xnio,xnio-api,PublicDomain
|   |   ├── org.jboss.xnio,xnio-nio,PublicDomain
|   |   ├── org.jfree,jcommon,GNULesserGeneralPublicLicence
|   |   ├── org.jfree,jfreechart,GNULesserGeneralPublicLicence
|   |   ├── org.jolokia,jolokia-core,Apache2
|   |   ├── org.json,json,providedwithoutsupportorwarranty
|   |   ├── org.jsoup,jsoup,TheMITLicense
|   |   ├── org.jvnet.mimepull,mimepull,CDDL1.1
|   |   ├── org.jvnet.staxex,stax-
ex,CommonDevelopmentAndDistributionLicense(CDDL)Version1.0
|   |   ├── org.komodo,komodo-core,GNULesserGeneralPublicLicense
|   |   ├── org.komodo,komodo-importer,GNULesserGeneralPublicLicense
|   |   ├── org.komodo,komodo-modeshape,GNULesserGeneralPublicLicense
|   |   ├── org.komodo,komodo-modeshape-sequencer-teiid-
sql,GNULesserGeneralPublicLicense
|   |   ├── org.komodo,komodo-modeshape-vdb,GNULesserGeneralPublicLicense
|   |   ├── org.komodo,komodo-plugin-service,GNULesserGeneralPublicLicense
|   |   ├── org.komodo,komodo-relational,GNULesserGeneralPublicLicense
|   |   ├── org.komodo,komodo-spi,GNULesserGeneralPublicLicense
|   |   ├── org.komodo,komodo-teiid-client,GNULesserGeneralPublicLicense
|   |   ├── org.komodo,komodo-ui,GNULesserGeneralPublicLicense
|   |   ├── org.komodo,komodo-utils,GNULesserGeneralPublicLicense
|   |   ├── org.komodo,komodo-utils-modeshape-
logger,GNULesserGeneralPublicLicense
|   |   ├── org.komodo.plugins,komodo-plugin-
framework,GNULesserGeneralPublicLicense
|   |   ├── org.komodo.plugins.storage,storage-
file,GNULesserGeneralPublicLicense
```

```
│   │    ├── org.komodo.plugins.storage,storage-
git,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-common,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-connector-
git,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-connector-jdbc-
metadata,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-extractor-
tika,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-jbossas-
subsystem,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-jcr-api,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-jcr,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-jdbc-
local,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-
schematic,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-sequencer-
ddl,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-sequencer-
images,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-sequencer-
java,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-sequencer-
mp3,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-sequencer-
msoffice,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-sequencer-
sramp,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-sequencer-
text,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-sequencer-
wsdl,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-sequencer-
xml,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-sequencer-
xsd,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-sequencer-
zip,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-web-cmis,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-webdav,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-web-
explorer,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-web-jcr,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-web-jcr-
rest,GNULesserGeneralPublicLicense
│   │    ├── org.modeshape,modeshape-web-jcr-
webdav,GNULesserGeneralPublicLicense
│   │    ├── org.noggit,noggit,TheApacheSoftwareLicense,Version2.0
│   │    ├── org.opengis,geoapi,OGCcopyright
│   │    ├── org.osgeo,proj4j,TheApacheLicense,Version2.0
│   │    ├── org.osgi,org.osgi.core,PublicDomain
│   │    ├── org.ow2.asm,asm,BSD
│   │    ├── org.quartz-scheduler,quartz,Apache2.0
│   │    ├── org.reflections,reflections,TheNewBSDLicense
│   │    ├── org.reflections,reflections,WTFPL
```

```
│    │    ├── org.slf4j,slf4j-log4j12,MITLicense
│    │    ├── org.springframework,spring-
aop,TheApacheSoftwareLicense,Version2.0
│    │    ├── org.springframework,spring-
beans,TheApacheSoftwareLicense,Version2.0
│    │    ├── org.springframework,spring-
context,TheApacheSoftwareLicense,Version2.0
│    │    ├── org.springframework,spring-
core,TheApacheSoftwareLicense,Version2.0
│    │    ├── org.springframework,spring-
expression,TheApacheSoftwareLicense,Version2.0
│    │    ├── org.springframework,spring-
tx,TheApacheSoftwareLicense,Version2.0
│    │    ├── org.tallison,jmatio,BSD
│    │    ├── org.teiid,vdb-bench-
assembly,TheApacheSoftwareLicense,Version2.0
│    │    ├── org.wildfly,wildfly-core-security-api,lgpl
│    │    ├── org.wildfly,wildfly-core-security,lgpl
│    │    ├── org.wololo,jts2geojson,MITlicense
│    │    ├── stax,stax-
api,COMMONDEVELOPMENTANDDISTRIBUTIONLICENSE(CDDL)Version1.0
│    │    ├── stax,stax-api,GNUGeneralPublicLibrary
│    │    ├── wsdl4j,wsdl4j,CPL
│    │    └── xerces,xercesImpl,TheApacheSoftwareLicense,Version2.0
│    ├── schema
│    │    ├── application_1_2.dtd
│    │    ├── application_1_3.dtd
│    │    ├── application_1_4.xsd
│    │    ├── application_5.xsd
│    │    ├── application_6.xsd
│    │    ├── application-client_6.xsd
│    │    ├── ejb-jar_1_1.dtd
│    │    ├── ejb-jar_2_0.dtd
│    │    ├── ejb-jar_2_1.xsd
│    │    ├── ejb-jar_3_0.xsd
│    │    ├── ejb-jar_3_1.xsd
│    │    ├── j2ee_1_4.xsd
│    │    ├── j2ee_jaxrpc_mapping_1_1.xsd
│    │    ├── j2ee_web_services_1_1.xsd
│    │    ├── j2ee_web_services_client_1_1.xsd
│    │    ├── javaee_5.xsd
│    │    ├── javaee_6.xsd
│    │    ├── javaee_web_services_1_2.xsd
│    │    ├── javaee_web_services_1_3.xsd
│    │    ├── javaee_web_services_client_1_2.xsd
│    │    ├── javaee_web_services_client_1_3.xsd
│    │    ├── java-properties_1_0.xsd
│    │    ├── jboss_1_0.xsd
│    │    ├── jboss-app_3_0.dtd
│    │    ├── jboss-app_3_2.dtd
│    │    ├── jboss-app_4_0.dtd
│    │    ├── jboss-app_4_2.dtd
│    │    ├── jboss-app_5_0.dtd
│    │    ├── jboss-app_7_0.xsd
│    │    ├── jboss-as-cli_1_0.xsd
│    │    ├── jboss-as-cli_1_1.xsd
```

```
│  │    ├── jboss-as-cli_1_2.xsd
│  │    ├── jboss-as-cli_1_3.xsd
│  │    ├── jboss-as-cmp_1_0.xsd
│  │    ├── jboss-as-cmp_1_1.xsd
│  │    ├── jboss-as-config_1_0.xsd
│  │    ├── jboss-as-config_1_1.xsd
│  │    ├── jboss-as-config_1_2.xsd
│  │    ├── jboss-as-config_1_3.xsd
│  │    ├── jboss-as-config_1_4.xsd
│  │    ├── jboss-as-config_1_5.xsd
│  │    ├── jboss-as-config_1_6.xsd
│  │    ├── jboss-as-config_1_7.xsd
│  │    ├── jboss-as-config_1_8.xsd
│  │    ├── jboss-as-configadmin_1_0.xsd
│  │    ├── jboss-as-datasources_1_0.xsd
│  │    ├── jboss-as-datasources_1_1.xsd
│  │    ├── jboss-as-datasources_1_2.xsd
│  │    ├── jboss-as-deployment-scanner_1_0.xsd
│  │    ├── jboss-as-deployment-scanner_1_1.xsd
│  │    ├── jboss-as-ee_1_0.xsd
│  │    ├── jboss-as-ee_1_1.xsd
│  │    ├── jboss-as-ee_1_2.xsd
│  │    ├── jboss-as-ejb3_1_0.xsd
│  │    ├── jboss-as-ejb3_1_1.xsd
│  │    ├── jboss-as-ejb3_1_2.xsd
│  │    ├── jboss-as-ejb3_1_3.xsd
│  │    ├── jboss-as-ejb3_1_4.xsd
│  │    ├── jboss-as-ejb3_1_5.xsd
│  │    ├── jboss-as-infinispan_1_0.xsd
│  │    ├── jboss-as-infinispan_1_1.xsd
│  │    ├── jboss-as-infinispan_1_2.xsd
│  │    ├── jboss-as-infinispan_1_3.xsd
│  │    ├── jboss-as-infinispan_1_4.xsd
│  │    ├── jboss-as-infinispan_1_5.xsd
│  │    ├── jboss-as-jacorb_1_0.xsd
│  │    ├── jboss-as-jacorb_1_1.xsd
│  │    ├── jboss-as-jacorb_1_2.xsd
│  │    ├── jboss-as-jacorb_1_3.xsd
│  │    ├── jboss-as-jacorb_1_4.xsd
│  │    ├── jboss-as-jaxr_1_0.xsd
│  │    ├── jboss-as-jaxr_1_1.xsd
│  │    ├── jboss-as-jaxrs_1_0.xsd
│  │    ├── jboss-as-jca_1_0.xsd
│  │    ├── jboss-as-jca_1_1.xsd
│  │    ├── jboss-as-jdr_1_0.xsd
│  │    ├── jboss-as-jgroups_1_0.xsd
│  │    ├── jboss-as-jgroups_1_1.xsd
│  │    ├── jboss-as-jmx_1_0.xsd
│  │    ├── jboss-as-jmx_1_1.xsd
│  │    ├── jboss-as-jmx_1_2.xsd
│  │    ├── jboss-as-jmx_1_3.xsd
│  │    ├── jboss-as-jpa_1_0.xsd
│  │    ├── jboss-as-jpa_1_1.xsd
│  │    ├── jboss-as-jsf_1_0.xsd
│  │    ├── jboss-as-jsr77_1_0.xsd
│  │    ├── jboss-as-logging_1_0.xsd
```

```
│   │   ├── jboss-as-logging_1_1.xsd
│   │   ├── jboss-as-logging_1_2.xsd
│   │   ├── jboss-as-logging_1_3.xsd
│   │   ├── jboss-as-logging_1_4.xsd
│   │   ├── jboss-as-logging_1_5.xsd
│   │   ├── jboss-as-mail_1_0.xsd
│   │   ├── jboss-as-mail_1_1.xsd
│   │   ├── jboss-as-mail_1_2.xsd
│   │   ├── jboss-as-messaging_1_0.xsd
│   │   ├── jboss-as-messaging_1_1.xsd
│   │   ├── jboss-as-messaging_1_2.xsd
│   │   ├── jboss-as-messaging_1_3.xsd
│   │   ├── jboss-as-messaging_1_4.xsd
│   │   ├── jboss-as-messaging-deployment_1_0.xsd
│   │   ├── jboss-as-mod-cluster_1_0.xsd
│   │   ├── jboss-as-mod-cluster_1_1.xsd
│   │   ├── jboss-as-mod-cluster_1_2.xsd
│   │   ├── jboss-as-naming_1_0.xsd
│   │   ├── jboss-as-naming_1_1.xsd
│   │   ├── jboss-as-naming_1_2.xsd
│   │   ├── jboss-as-naming_1_3.xsd
│   │   ├── jboss-as-naming_1_4.xsd
│   │   ├── jboss-as-osgi_1_0.xsd
│   │   ├── jboss-as-osgi_1_1.xsd
│   │   ├── jboss-as-osgi_1_2.xsd
│   │   ├── jboss-as-pojo_1_0.xsd
│   │   ├── jboss-as-remoting_1_0.xsd
│   │   ├── jboss-as-remoting_1_1.xsd
│   │   ├── jboss-as-remoting_1_2.xsd
│   │   ├── jboss-as-resource-adapters_1_0.xsd
│   │   ├── jboss-as-resource-adapters_1_1.xsd
│   │   ├── jboss-as-sar_1_0.xsd
│   │   ├── jboss-as-security_1_0.xsd
│   │   ├── jboss-as-security_1_1.xsd
│   │   ├── jboss-as-security_1_2.xsd
│   │   ├── jboss-as-threads_1_0.xsd
│   │   ├── jboss-as-threads_1_1.xsd
│   │   ├── jboss-as-txn_1_0.xsd
│   │   ├── jboss-as-txn_1_1.xsd
│   │   ├── jboss-as-txn_1_2.xsd
│   │   ├── jboss-as-txn_1_3.xsd
│   │   ├── jboss-as-txn_1_4.xsd
│   │   ├── jboss-as-txn_1_5.xsd
│   │   ├── jboss-as-web_1_0.xsd
│   │   ├── jboss-as-web_1_1.xsd
│   │   ├── jboss-as-web_1_2.xsd
│   │   ├── jboss-as-web_1_3.xsd
│   │   ├── jboss-as-web_1_4.xsd
│   │   ├── jboss-as-web_1_5.xsd
│   │   ├── jboss-as-web_2_1.xsd
│   │   ├── jboss-as-web_2_2.xsd
│   │   ├── jboss-as-webservices_1_0.xsd
│   │   ├── jboss-as-webservices_1_1.xsd
│   │   ├── jboss-as-webservices_1_2.xsd
│   │   ├── jboss-as-weld_1_0.xsd
│   │   ├── jboss-as-xts_1_0.xsd
```

```
│   │   ├── jboss-client_6_0.xsd
│   │   ├── jboss-common_5_1.xsd
│   │   ├── jboss-common_6_0.xsd
│   │   ├── jboss-deployment-dependencies-1_0.xsd
│   │   ├── jboss-deployment-structure-1_0.xsd
│   │   ├── jboss-deployment-structure-1_1.xsd
│   │   ├── jboss-deployment-structure-1_2.xsd
│   │   ├── jboss-ejb3-2_0.xsd
│   │   ├── jboss-ejb3-spec-2_0.xsd
│   │   ├── jboss-ejb-cache_1_0.xsd
│   │   ├── jboss-ejb-client_1_0.xsd
│   │   ├── jboss-ejb-client_1_1.xsd
│   │   ├── jboss-ejb-client_1_2.xsd
│   │   ├── jboss-ejb-container-interceptors_1_0.xsd
│   │   ├── jboss-ejb-delivery-active_1_0.xsd
│   │   ├── jboss-ejb-iiop_1_0.xsd
│   │   ├── jboss-ejb-pool_1_0.xsd
│   │   ├── jboss-ejb-resource-adapter-binding_1_0.xsd
│   │   ├── jboss-ejb-security_1_0.xsd
│   │   ├── jboss-ejb-security_1_1.xsd
│   │   ├── jboss-ejb-security-role_1_0.xsd
│   │   ├── jboss-jpa_1_0.xsd
│   │   ├── jboss-pojo_7_0.xsd
│   │   ├── jboss-service_7_0.xsd
│   │   ├── jboss-teiid.xsd
│   │   ├── jboss-web_7_0.xsd
│   │   ├── jboss-web_7_1.xsd
│   │   ├── jboss-web_7_2.xsd
│   │   ├── jbossws-jaxws-config_4_0.xsd
│   │   ├── jbossws-web-services_1_0.xsd
│   │   ├── jbxb_1_0.xsd
│   │   ├── jndi-binding-service_1_0.xsd
│   │   ├── jsp_2_0.xsd
│   │   ├── jsp_2_1.xsd
│   │   ├── jsp_2_2.xsd
│   │   ├── module-1_0.xsd
│   │   ├── module-1_1.xsd
│   │   ├── module-1_2.xsd
│   │   ├── module-1_3.xsd
│   │   ├── orm_1_0.xsd
│   │   ├── persistence_1_0.xsd
│   │   ├── persistence_2_0.xsd
│   │   ├── service-ref_4_0.dtd
│   │   ├── service-ref_4_2.dtd
│   │   ├── service-ref_5_0.dtd
│   │   ├── trans-timeout-1_0.xsd
│   │   ├── user-roles_1_0.xsd
│   │   ├── web-app_2_2.dtd
│   │   ├── web-app_2_3.dtd
│   │   ├── web-app_2_4.xsd
│   │   ├── web-app_2_5.xsd
│   │   ├── web-app_3_0.xsd
│   │   ├── web-common_3_0.xsd
│   │   ├── web-facesconfig_1_0.dtd
│   │   ├── web-facesconfig_1_1.dtd
│   │   ├── web-facesconfig_1_2.xsd
```

```
│       │   ├── web-fragment_3_0.xsd
│       │   ├── web-jsptaglibrary_1_1.dtd
│       │   ├── web-jsptaglibrary_1_2.dtd
│       │   ├── web-jsptaglibrary_2_0.xsd
│       │   ├── web-jsptaglibrary_2_1.xsd
│       │   ├── wildfly-picketlink-federation_1_0.xsd
│       │   ├── wildfly-picketlink-federation_1_1.xsd
│       │   ├── wildfly-picketlink-idm_1_0.xsd
│       │   └── wildfly-picketlink-idm_1_1.xsd
│       └── teiid
│           ├── datasources
│           │   ├── accumulo
│           │   ├── actian-vector
│           │   ├── amazon-s3
│           │   ├── cassandra
│           │   ├── couchbase
│           │   ├── db2
│           │   ├── derby
│           │   ├── file
│           │   ├── google
│           │   ├── h2
│           │   ├── hive
│           │   ├── impala
│           │   ├── infinispan
│           │   ├── infinispan-hotrod-7.1
│           │   ├── ingres
│           │   ├── intersystems-cache
│           │   ├── ldap
│           │   ├── mongodb
│           │   ├── mysql
│           │   ├── odbc
│           │   ├── olap
│           │   ├── oracle
│           │   ├── osisoft-pi
│           │   ├── phoenix
│           │   ├── postgresql
│           │   ├── prestodb
│           │   ├── redshift
│           │   ├── salesforce
│           │   ├── simpledb
│           │   ├── solr
│           │   ├── sqlserver
│           │   ├── teiid
│           │   ├── ucanaccess
│           │   ├── vertica
│           │   └── web-service
│           ├── licenses
│           │   ├── apache-2.0 - LICENSE-2.0.txt
│           │   ├── LICENSE-lgpl-2.1.txt
│           │   ├── MPL-1.0.html
│           │   ├── MPL-1.1.html
│           │   └── PostgreSQL-BSD.txt
│           ├── schema
│           │   └── vdb-deployer.xsd
│           └── teiid-releasenotes.html
├── domain
```

```
│   ├── configuration
│   │   ├── application-roles.properties
│   │   ├── application-users.properties
│   │   ├── default-server-logging.properties
│   │   ├── domain.xml
│   │   ├── domain_xml_history
│   │   │   ├── 20180223-095813802
│   │   │   ├── current
│   │   │   ├── domain.boot.xml
│   │   │   ├── domain.initial.xml
│   │   │   ├── domain.last.xml
│   │   │   └── snapshot
│   │   ├── host-master.xml
│   │   ├── host-slave.xml
│   │   ├── host.xml
│   │   ├── host_xml_history
│   │   │   ├── 20180223-095813622
│   │   │   ├── 20180223-095817935
│   │   │   ├── current
│   │   │   ├── host.boot.xml
│   │   │   ├── host.initial.xml
│   │   │   ├── host.last.xml
│   │   │   ├── host-master.boot.xml
│   │   │   ├── host-master.initial.xml
│   │   │   ├── host-master.last.xml
│   │   │   ├── host-slave.boot.xml
│   │   │   ├── host-slave.initial.xml
│   │   │   ├── host-slave.last.xml
│   │   │   └── snapshot
│   │   ├── logging.properties
│   │   ├── mgmt-groups.properties
│   │   └── mgmt-users.properties
│   ├── data
│   │   └── content
│   ├── deployments
│   │   └── integration-platform-console.war
│   │       ├── eap.css
│   │       ├── favicon.ico
│   │       ├── images
│   │       ├── index.jsp
│   │       └── WEB-INF
│   ├── log
│   │   ├── host-controller.log
│   │   └── process-controller.log
│   ├── servers
│   └── tmp
│       └── auth
├── fusepatch
│   ├── repository
│   └── workspace
│       ├── audit.log
│       ├── jboss-dv
│       │   └── 6.4.0
│       └── managed-paths.metadata
├── installation
│   ├── InstallationLog.txt
```

```
│       │       ├── InstallSummary.html
│       │       └── server-logs
│       │           ├── host-controller.log
│       │           ├── process-controller.log
│       │           └── server.log
│       ├── JBossEULA.txt
│       ├── jboss-modules.jar
│       ├── LICENSE.txt
│       ├── modules
│       │   ├── layers.conf
│       │   └── system
│       │       └── layers
│       │           ├── base
│       │           └── dv
│       ├── module.xml
│       ├── pretty-print.xsl
│       ├── quickstarts
│       │   ├── build.metadata
│       │   ├── drools-integration
│       │   │   ├── pom.xml
│       │   │   ├── README.md
│       │   │   └── src
│       │   │       ├── main
│       │   │       ├── modules
│       │   │       ├── scripts
│       │   │       └── vdb
│       │   ├── dynamicvdb-datafederation
│       │   │   ├── pom.xml
│       │   │   ├── README.md
│       │   │   └── src
│       │   │       ├── scripts
│       │   │       ├── teiidfiles
│       │   │       └── vdb
│       │   ├── dynamicvdb-dataroles
│       │   │   ├── pom.xml
│       │   │   ├── README.md
│       │   │   └── src
│       │   │       ├── scripts
│       │   │       └── vdb
│       │   ├── dynamicvdb-materialization
│       │   │   ├── pom.xml
│       │   │   ├── README.md
│       │   │   └── src
│       │   │       ├── scripts
│       │   │       └── vdb
│       │   ├── dynamicvdb-restservice
│       │   │   ├── http-client
│       │   │   │   ├── pom.xml
│       │   │   │   └── src
│       │   │   ├── pom.xml
│       │   │   ├── README.md
│       │   │   ├── resteasy-client
│       │   │   │   ├── pom.xml
│       │   │   │   └── src
│       │   │   └── src
│       │   │       ├── scripts
```

```
│   │           └── vdb
│   ├── hbase-as-a-datasource
│   │   ├── pom.xml
│   │   ├── README.md
│   │   └── src
│   │       ├── scripts
│   │       ├── teiidfiles
│   │       └── vdb
│   ├── hibernate-on-top-of-teiid
│   │   ├── pom.xml
│   │   ├── README.md
│   │   └── src
│   │       ├── main
│   │       └── scripts
│   ├── jdg7.1-remote-cache
│   │   ├── pom.xml
│   │   ├── README.md
│   │   └── src
│   │       ├── scripts
│   │       └── vdb
│   ├── jdg7.1-remote-cache-materialization
│   │   ├── pom.xml
│   │   ├── README.md
│   │   └── src
│   │       ├── scripts
│   │       └── vdb
│   ├── jdg-remote-cache
│   │   ├── kits
│   │   │   ├── jboss-as7
│   │   │   └── jboss-as7-dist.xml
│   │   ├── pom.xml
│   │   ├── README.md
│   │   └── src
│   │       ├── jdg
│   │       ├── main
│   │       └── vdb
│   ├── jdg-remote-cache-materialization
│   │   ├── kits
│   │   │   ├── jboss-as7
│   │   │   └── jboss-as7-dist.xml
│   │   ├── pom.xml
│   │   ├── README.md
│   │   └── src
│   │       ├── jdg
│   │       ├── main
│   │       └── vdb
│   ├── ldap-as-a-datasource
│   │   ├── ldap-add-group-users.md
│   │   ├── pom.xml
│   │   ├── README.md
│   │   └── src
│   │       ├── scripts
│   │       └── vdb
│   ├── mongodb-as-a-datasource
│   │   ├── pom.xml
│   │   ├── README.md
```

```
│   │       └── src
│   │           ├── scripts
│   │           └── vdb
│   ├── pom.xml
│   ├── README.md
│   ├── settings.xml
│   ├── simpleclient
│   │   ├── pom.xml
│   │   ├── README.md
│   │   └── src
│   │       └── main
│   ├── socialMedia-as-a-datasource
│   │   ├── pom.xml
│   │   ├── README.md
│   │   └── src
│   │       ├── scripts
│   │       └── vdb
│   ├── tpch
│   │   ├── change-tablenames.sh
│   │   ├── config-files
│   │   │   ├── datasources.xml
│   │   │   ├── drop-create.sql
│   │   │   ├── modules
│   │   │   ├── postgres-dml.sql
│   │   │   ├── postgres-index.sql
│   │   │   └── tpch-vdb.xml
│   │   ├── generate-data.sh
│   │   ├── generate-one-query.sh
│   │   ├── generate-queries.sh
│   │   ├── javaPerfTest
│   │   │   ├── pom.xml
│   │   │   └── src
│   │   ├── load-data-into-db.sh
│   │   ├── query-templates
│   │   │   ├── 10.sql
│   │   │   ├── 11.sql
│   │   │   ├── 12.sql
│   │   │   ├── 13.sql
│   │   │   ├── 14.sql
│   │   │   ├── 15.sql
│   │   │   ├── 16.sql
│   │   │   ├── 17.sql
│   │   │   ├── 18.sql
│   │   │   ├── 19.sql
│   │   │   ├── 1.sql
│   │   │   ├── 20.sql
│   │   │   ├── 21.sql
│   │   │   ├── 22.sql
│   │   │   ├── 2.sql
│   │   │   ├── 3.sql
│   │   │   ├── 4.sql
│   │   │   ├── 5.sql
│   │   │   ├── 6.sql
│   │   │   ├── 7.sql
│   │   │   ├── 8.sql
│   │   │   ├── 9.sql
```

```
│   │   │   └── dists.dss
│   │   ├── README.md
│   │   ├── run-test.sh
│   │   └── setenv.sh
│   └── webservices-as-a-datasource
│       ├── pom.xml
│       ├── README.md
│       ├── src
│       │   ├── main
│       │   ├── scripts
│       │   └── vdb
│       └── webapp
│           └── WEB-INF
├── standalone
│   ├── configuration
│   │   ├── application-roles.properties
│   │   ├── application-users.properties
│   │   ├── logging.properties
│   │   ├── mgmt-groups.properties
│   │   ├── mgmt-users.properties
│   │   ├── standalone-full-ha.xml
│   │   ├── standalone-full.xml
│   │   ├── standalone-ha.xml
│   │   ├── standalone-osgi.xml
│   │   ├── standalone.xml
│   │   └── standalone_xml_history
│   │       ├── 20180223-095659186
│   │       ├── 20180223-095719975
│   │       ├── current
│   │       ├── snapshot
│   │       ├── standalone.boot.xml
│   │       ├── standalone-full-ha.boot.xml
│   │       ├── standalone-full-ha.initial.xml
│   │       ├── standalone-full-ha.last.xml
│   │       ├── standalone-ha.boot.xml
│   │       ├── standalone-ha.initial.xml
│   │       ├── standalone-ha.last.xml
│   │       ├── standalone.initial.xml
│   │       └── standalone.last.xml
│   ├── data
│   │   └── content
│   ├── deployments
│   │   ├── integration-platform-console.war
│   │   │   ├── consoles
│   │   │   ├── eap.css
│   │   │   ├── favicon.ico
│   │   │   ├── images
│   │   │   ├── index.jsp
│   │   │   └── WEB-INF
│   │   ├── integration-platform-console.war.dodeploy
│   │   └── README.txt
│   ├── lib
│   │   └── ext
│   ├── log
│   │   ├── backupgc.log.current
│   │   ├── gc.log.0.current
```

```
│   │       └── server.log
│   └── tmp
│       ├── auth
│       └── vfs
│           └── temp
├── vault
│   └── VAULT.dat
├── vault.keystore
├── version-dv.txt
├── version.txt
└── welcome-content
    ├── eap.css
    ├── favicon.ico
    ├── fonts
    │   ├── OpenSans-BoldItalic.ttf
    │   ├── OpenSans-Bold.ttf
    │   ├── OpenSans-ExtraBoldItalic.ttf
    │   ├── OpenSans-ExtraBold.ttf
    │   ├── OpenSans-Italic.ttf
    │   ├── OpenSans-LightItalic.ttf
    │   ├── OpenSans-Light.ttf
    │   ├── OpenSans-Regular.ttf
    │   ├── OpenSans-SemiboldItalic.ttf
    │   └── OpenSans-Semibold.ttf
    ├── images
    │   ├── eap_bg.png
    │   ├── header_bg.png
    │   ├── prod_logo.png
    │   ├── prod_name.png
    │   └── product_title.png
    ├── index.html
    ├── index_noconsole.html
    ├── noconsole.html
    └── noredirect.html
```

APPENDIX A. REVISION HISTORY

Revision 6.40-43   **Tues May 29 2018**   **David Le Sage**
   Updates for 6.4.