GT 4.2.0 WS RLS: System Administrator's Guide
GT 4.2.0 WS RLS: System Administrator's Guide

Introduction

This guide contains advanced configuration information for system administrators working with WS RLS. It provides references to information on procedures typically performed by system administrators, including installation, configuring, deploying, and testing the installation.

⚠️ Important

This information is in addition to the basic Globus Toolkit prerequisite, overview, installation, security configuration instructions in the Installing GT 4.2.0. Read through this guide before continuing!
# Table of Contents

1. Building and Installing ................................................................. 1  
   1. Installing WS RLS using Globus Toolkit Binary Bundle ................. 1  
   2. Installing WS RLS using Globus Toolkit Source Bundle ............... 1  
   3. Installing WS RLS using Globus CVS ....................................... 2  
2. Configuring WS RLS ................................................................. 3  
   1. Configuration overview .......................................................... 3  
   2. WS RLS Web Service Deployment Descriptor (WSDD) Configuration 3  
   3. WS RLS Java Naming & Directory Interface (JNDI) Configuration .......... 4  
3. Testing ..................................................................................... 5  
4. Security Considerations ............................................................. 6  
   1. WS Replica Location Service (WS RLS) Security Considerations .......... 6  
5. Debugging .................................................................................. 7  
   1. Logging in Java WS Core ....................................................... 7  
   2. Increasing log verbosity ......................................................... 8  
6. Troubleshooting ......................................................................... 9  
   1. Errors .................................................................................... 9  
7. Usage statistics collection by the Globus Alliance ......................... 10  
   1. WS RLS-specific usage statistics .............................................. 10  
Index .............................................................................................. 11
List of Tables

6.1. WS Replica Location Service (WS RLS) Errors ................................................................................... 9
Chapter 1. Building and Installing

WS RLS is distributed with the Globus Toolkit 4.2.0 and is available in both the binary bundles and the source bundle. For most purposes, the binary bundle provides the simplest means of installing the WS RLS and its requirements. There are three typical ways to access and install the WS RLS:

- Section 1, “Installing WS RLS using Globus Toolkit Binary Bundle”
- Section 2, “Installing WS RLS using Globus Toolkit Source Bundle”
- Section 3, “Installing WS RLS using Globus CVS”

⚠ Important

Before proceeding with installation of WS RLS, please familiarize yourself with the installation procedures for RLS (see Replica Location Service (RLS)).

1. Installing WS RLS using Globus Toolkit Binary Bundle

Follow instructions provided by Installing GT 4.2.0 and Installing GT in order to install the WS RLS from one of the available binary bundles of the Globus Toolkit.

Once you have unpackaged the binary bundle, the following commands may be used to install WS RLS:

% ./configure --prefix=$GLOBUS_LOCATION --enable-rls --enable-wsrls
% make
% make install

2. Installing WS RLS using Globus Toolkit Source Bundle

WS RLS is not installed as part of the default source install. In order to install WS RLS, you will need to ensure that RLS is also installed as part of the build by reviewing its installation procedures. Along with RLS configure settings (e.g., ODBC related settings) the --enable-wsrls flag is required as part of the configuration step.

The following commands will install the WS RLS (optionally you may use the "gt4-wsrls" make target to build only WS RLS and its requirements:

% ./configure --prefix=/path/to/install --enable-rls --enable-wsrls
% make [rls] gt4-wsrls
% make install
3. Installing WS RLS using Globus CVS

Additionally, you may access the WS RLS from the Globus Toolkit CVS repository. You may deploy the 4.1.x WS RLS on top of an existing GT 4.1.x installation using trunk.

⚠️ **Important**

Please review [Accessing the Globus Alliance's Code via CVS](http://www.globus.org/toolkit/docs/development/remote-cvs.html) for the authoritative information on building and installing the Globus Toolkit from source in our CVS.

The following instructions may be used to build and install the WS RLS:

```bash
setenv CVSROOT :pserver:anonymous@cvs.globus.org:/home/globdev/CVS/globus-packages
cvs co packaging/
cd packaging
./make-packages.pl --bundles=gt4-wsrls --deps [--verbose]
setenv GPT_LOCATION /path/to/packaging/gpt-3.2globus2
$GPT_LOCATION/sbin/gpt-build [--verbose] ./bundle-output/custom-deps-1.0-src_bundle.tar.gz
```

🔍 **Note**

Changes to the toolkit (WS Core and WSAA) made between the 4.0.x release and the latest code base, make the GT 4.0.x build environment unsuitable for GT 4.2.0 components. As such the WS RLS cannot be deployed on an existing GT 4.0.x installation. The good news, if you will, is that the WS RLS can use your existing GT 4.0.x RLS without modification. If you have an existing RLS installation, and you want to try out the WS RLS interface layer, you need not reinstall RLS. Of course, you may want to reinstall RLS exclusively for testing with the WS RLS interface, but it is up to your own choosing.

---

Chapter 2. Configuring WS RLS

1. Configuration overview

The WS RLS requires certain WSDD and JNDI settings to be properly configured. The installed JNDI configuration file may be found at $GLOBUS_LOCATION/etc/globus_wsrfs_replicalocation_resource/jndi-config.xml. To view the default JNDI file (current snapshot) from the Globus CVS repository click here. The installed WSDD configuration file may be found at $GLOBUS_LOCATION/etc/globus_wsrfs_replicalocation_service/server-config.wsdd. To view the default WSDD file (current snapshot) from the Globus CVS repository click here. In most cases, you will not need to alter either of these files.

This information is in addition to the basic configuration instructions in the Installing GT 4.2.0.

2. WS RLS Web Service Deployment Descriptor (WSDD) Configuration

The WS RLS's WSDD determines which services will be activated by the GT Java WS Core container. By default, the WS RLS defines two services.

- ReplicaLocationCatalogService. The ReplicaLocationCatalogService service provides the WS interface to the RLS Local Replica Catalog capabilities. If you intend to activate this service, you need not alter the WSDD. If you do not wish to activate this service, simply remove it from the WSDD and restart the container.

  ...
  <service name="ReplicaLocationCatalogService" provider="Handler".
  use="literal" style="document">
  ...
  </service>
  ...

- ReplicaLocationIndexService. The ReplicaLocationIndexService service provides the WS interface to the RLS Replica Location Index capabilities. If you intend to activate this service, you need not alter the WSDD. If you do not wish to activate this service, simply remove it from the WSDD and restart the container.

  ...
  <service name="ReplicaLocationIndexService" provider="Handler".
  use="literal" style="document">
  ...
  </service>
  ...

1 http://viewcvs.globus.org/viewcvs.cgi/ws-replica/location/resource/java/source/deploy-jndi-config.xml
2 http://viewcvs.globus.org/viewcvs.cgi/ws-replica/location/service/java/source/deploy-server.wsdd
3. WS RLS Java Naming & Directory Interface (JNDI) Configuration

The WS RLS's JNDI configuration determines the resource class to be used for each service defined by the WS RLS WSDD. It also determines other critical settings used by these resources. In this file, you will find settings for the two default services defined in the WSDD.

Each service has a JNDI Resource of name RLSConnectionSource. This JNDI Resource is used by the WS RLS service to open a connection to the backend RLS service. In many cases, you may find that the default settings require no modification for your setup. The url parameter sets the endpoint for the backend RLS. The certfile and keyfile parameters set the certificate and key files to be used by the WS RLS to communicate securely with the backend RLS.

You will notice that the default settings use the container's suggested certificate and key file paths. If you have chosen different paths in which to keep these files, please adjust the settings accordingly. At present, the WS RLS uses a single identity (defined by the certificate and key) to authenticate itself with the RLS in order to perform any user requested operations. When the WS RLS interface is finalized, it will most likely use a different approach to authentication between WS RLS and the back end RLS. At this time, we expect that the interface will change such that the caller's credentials will be used to make calls between the WS RLS and the back end RLS on behalf of the originating caller.

...<resource
    name="RLSConnectionSource"
    type="org.globus.replica.rls.connection.impl.PooledRLSConnectionSource">
    <resourceParams>
      <parameter>
        <name>factory</name>
        <value>org.globus.wsrf.jndi.BeanFactory</value>
      </parameter>
      <parameter>
        <name>url</name>
        <value>rls://localhost</value>
      </parameter>
      <parameter>
        <name>certfile</name>
        <value>/etc/grid-security/containercert.pem</value>
      </parameter>
      <parameter>
        <name>keyfile</name>
        <value>/etc/grid-security/containerkey.pem</value>
      </parameter>
    </resourceParams>
  </resource>
...
Chapter 3. Testing

At this time, the WS RLS does not yet provide an automated test suite.
Chapter 4. Security Considerations

1. WS Replica Location Service (WS RLS) Security Considerations

Security recommendations include:

- The following paragraph describes our INITIAL IMPLEMENTATION in this first look of the WS RLS interface -- THIS IS NOT INTENDED TO BE A FINAL SOLUTION.

Users of the WS RLS authenticate themselves to the WS RLS in the usual manner based on their credential. The WS RLS uses the WSAA and Authorization Framework to make authorization decisions. Then, the WS RLS uses a single certificate and key file to identify itself to the RLS irregardless of the user accessing the WS RLS. Thus users that can access the WS RLS are given the fixed WS RLS identity to access the RLS. Users accustomed to using RLS may not feel comfortable with this approach -- if it may be an issue for your environment, we suggest that you not use it with your production RLS. If you think of WS RLS as the gating interface to the RLS, as you should, then you should apply the appropriate authorization restrictions at the WS RLS level, which can be done using WSAA. This will in effect achieve a level of authorization similar to that of the RLS.
Chapter 5. Debugging

The following information is from Java WS Core, on which WS RLS is based.

1. Logging in Java WS Core

The following information applies to Java WS Core and all services built on Java WS Core.

Java WS Core server side has two types of loggers. One logger is used for development logging and by default writes to standard out. The other logger includes system administration information and is CEDPs best practices compliant.

On client side, only developer logging is available and is configured using log4j.properties.

1.1. Development Logging in Java WS Core

The following information applies to Java WS Core and those services built on it.

Logging in the Java WS Core is based on the Jakarta Commons Logging API. Commons Logging provides a consistent interface for instrumenting source code while at the same time allowing the user to plug-in a different logging implementation. Currently we use Log4j as a logging implementation. Log4j uses a separate configuration file to configure itself. Please see Log4j documentation for details on the configuration file format.

1.1.1. Configuring server side developer logs

Server side logging can be configured in $GLOBUS_LOCATION/container-log4j.properties, when the container is stand alone container. For tomcat level logging, refer to Logging for Tomcat. The logger log4j.appender.A1 is used for developer logging and by default writes output to the system output. By default it is set for all warnings in the Globus Toolkit package to be displayed.

Additional logging can be enabled for a package by adding a new line to the configuration file. Example:

```java
# for debug level logging from org.globus.package.FooClass
log4j.category.org.globus.package.name.FooClass=DEBUG
# for warnings from org.some.warn.package
log4j.category.org.some.warn.package=WARN
```

1.1.2. Configuring client side developer logs

Client side logging can be configured in $GLOBUS_LOCATION/log4j.properties. The logger log4j.appender.A1 is used for developer logging and by default writes output to the system output. By default it is set for all warnings in the Globus Toolkit package to be displayed.

---

1 http://cedps.net/index.php/LoggingBestPractices
2 http://jakarta.apache.org/commons/logging/
3 http://logging.apache.org/log4j/
5 http://tomcat.apache.org/tomcat-5.5-doc/logging.html
1.2. Configuring system administration logs

The specific logger to edit will be `log4j.logger.sysadmin` in `$GLOBUS_LOCATION/container-log4j.properties`. There you can configure the following properties:

```
log4j.appender.infoCategory=org.apache.log4j.RollingFileAppender
  log4j.appender.infoCategory.Threshold=INFO
  log4j.appender.infoCategory.File=var/containerLog
  log4j.appender.infoCategory.MaxFileSize=10MB
  log4j.appender.infoCategory.MaxBackupIndex=2
```

Above implies the logging file is rolling with each file size limited to 10MB and the logging information is stored in `$GLOBUS_LOCATION/var/containerLog`.

1.3. Sample log file

The sample log file\(^6\) contains many log entries for various scenarios in the Java WS container.

2. Increasing log verbosity

Generating verbose log output is a critical aid in troubleshooting of the WS RLS and is useful when communicating problems to Globus support lists. To increase logging detail, add the following lines to the `$GLOBUS_LOCATION/container-log4j.properties` file.

```
...
log4j.category.org.globus.replica.location=DEBUG
log4j.category.org.globus.replica.rls=DEBUG
...
```

---

\(^6\) [http://www.globus.org/toolkit/docs/4.2/4.2.0/common/javawscore/sample-container-log.txt](http://www.globus.org/toolkit/docs/4.2/4.2.0/common/javawscore/sample-container-log.txt)
Chapter 6. Troubleshooting

For sys admin debugging information, see Chapter 5, Debugging.

For a list of common errors in GT, see Error Codes.

1. Errors

Table 6.1. WS Replica Location Service (WS RLS) Errors

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Definition</th>
<th>Possible Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error: java.lang.NullPointerException</td>
<td>When invoking the WS RLS command-line clients, a system-level exception like the one above may be encountered.</td>
<td>The admin should check the container logs for the exact error.</td>
</tr>
<tr>
<td>Error: A server error occurred while processing the request</td>
<td>When invoking the WS RLS command-line clients, a server error like the one above may be encountered.</td>
<td>The admin should check the container logs for the exact error.</td>
</tr>
<tr>
<td>java.lang.UnsatisfiedLinkError</td>
<td>This exception when using the WS RLS may indicate that the native RLS libraries that WS RLS depends on could not be located.</td>
<td>To correct this problem, ensure that the $GLOBUS_LOCATION/lib directory is in the library search path (on some systems this is the LD_LIBRARY_PATH variable).</td>
</tr>
<tr>
<td>Unable to connect to localhost:39281</td>
<td>The WS RLS is an interface layer that depends on the RLS for the replica location functionality. You must install and run RLS and configure WS RLS to use the RLS via its JNDI configuration.</td>
<td>Check that RLS is installed, running, and check that the WS RLS JNDI configuration uses the correct hostname and port to connect to the RLS.</td>
</tr>
<tr>
<td>org.globus.common.ChainedIOException: Failed to initialize security context</td>
<td>If this exception occurs while using WS RLS, it may indicate that the user's proxy is invalid.</td>
<td>To correct the error, the user must properly initialize the user proxy. See grid-proxy-init for more information on proxy initialization.</td>
</tr>
<tr>
<td>Error: org.xml.sax.SAXException: Unregistered type: class xxx</td>
<td>If this exception occurs when using the WS RLS, it may indicate that an Axis generated XML type, defined by the WS RLS XSD, was not properly registered. While all the XML types should get registered upon deployment without intervention by the user, sometimes they do not.</td>
<td>To remedy the situation add a typeMapping to the server-config.wadl file under globus_wsrp_replicalocation_service. Use the format shown here.</td>
</tr>
</tbody>
</table>
Chapter 7. Usage statistics collection by the Globus Alliance

1. WS RLS-specific usage statistics

The WS RLS does not collect usage statistics in addition to what the RLS collects. Please consult the RLS documentation to familiarize yourself with usage statistics collected by it.

Also, please see our policy statement\(^1\) on the collection of usage statistics.

\(^1\) ../../Usage_Stats.html
Index

C
configuring ReplicaLocationCatalogService, 3
configuring ReplicaLocationIndexService, 3
configuring WS RLS, 3

D
debugging
   logging, 7

E
errors, 9

I
installing
   from binary, 1
   from CVS, 2
   from source, 1

L
logging, 7
   CEDPS-compliant, 7
   debugging, 7

T
troubleshooting
   errors, 9

U
usage statistics, 10