



Common Logging Module (CLM)

Log Locator Tool (LLT) User Guide

Version No: 1.0

Last Modified: 11/02/2006

Author : Vijay Parmar
Team : Common Logging Module (CLM)
Purchase Order# 34552
Client : National Cancer Institute – Center for Bioinformatics,
National Institutes of Health,
US Department of Health and Human Services

Document History

Document Location

The most current version of this document is located on the CSM website: <http://ncicb.nci.nih.gov/core/CSM>

Revision History

<i>Version Number</i>	<i>Revision Date</i>	<i>Author</i>	<i>Summary of Changes</i>
1.0	11/02/2006	Vijay Parmar	Initial draft

Review

<i>Name</i>	<i>Team/Role</i>	<i>Version</i>	<i>Date Reviewed</i>	<i>Reviewer Comments</i>
Kunal Modi		1.0	11/03/2006	

Related Documents

More information can be found in the following related CSM documents:

<i>Document Name</i>
CLM Guide for Application Developers
CSM Guide for Application Developers
CLM Enterprise Architect Model

These and other documents can be found on the CSM website: <http://ncicb.nci.nih.gov/core/CSM>

Table of Contents

1.	Introduction to Log Locator Tool	4
1.1.	Purpose	4
1.2.	Scope	4
1.3.	Intended Audience	4
2.	LLT Overview	4
3.	Components	5
3.1.	Login	5
3.2.	Query Section	6
3.3.	Query Results	8
4.	Deployment Models	10
4.1.	Release Contents	10
4.2.	Deployment Checklist	10
4.3.	Deployment Steps	10
5.	Securing Logs in LLT using CSM	12
5.1.	Authentication Settings	12
5.2.	Authorization Settings	12

Log Locator Tool Guide

1. Introduction to Log Locator Tool

1.1. Purpose

This document provides all information application developers need to successfully deploy and use Log Locator Tool to view logs generated via Common Logging Module (CLM). Log Locator is a web application used to browse through the logs that were generated using the CLM APIs. This tool should be pointed to the same database used to store the log messages. This guide explains how to deploy the web locator from start to finish - from uploading the Web Application Archive (WAR) and editing configuration files, to synching the log locator tool with the application. This guide also details the Log Locator release contents and outlines the steps that result in a successful deployment.

1.2. Scope

This document shows how to deploy and integrate the CLM services, including event logging and automated object state logging. It also shows how to deploy and configure the web locator tool for the purpose of browsing through the logs.

1.3. Intended Audience

Log Locator's intended audience includes: system administrators, administrators, management, production support, and anyone else who needs to review the events that occurred within a given system.

2. LLT Overview

The Log Locator tool of CLM is a web application that allows users to view all of the events that occur within a program. It can be used to review changes made by users, reveal login date/times, expose malicious attempts at entering the system, audit a user, and identify and resolve production support issues. Users can filter the log messages by searching for content within any field of the message including: date and time that the event occurred, Log Level/Type, user, etc. By using these filters, users can view only the message relevant to their needs. The Log Locator can only be used by applications that use CLM APIs to look for audit and logging.

Figure 2-1 shows how CLM works with an application and independent entities, such as the credential providers and authorization schema, to perform authentication and authorization.

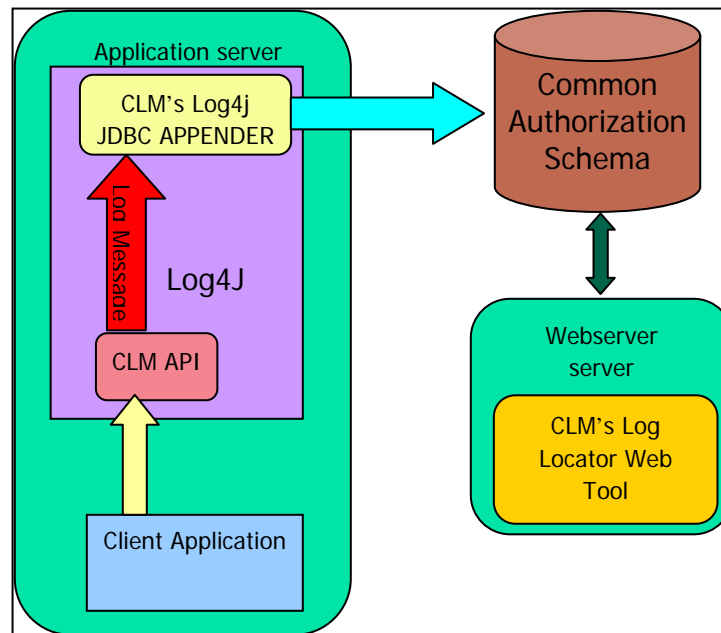


Figure 2-1 CLM interactions Audit Logging and Web Locator Tool (see text)

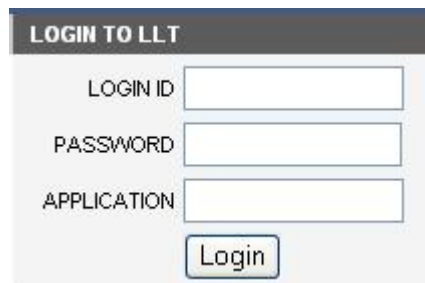
3. Components

There are three visual components to the Log Locator:

- **Login** – Authenticates users and gathers necessary information before allowing the viewing of log messages
- **Query Section** – allows user to provide a Search Criteria to filter the messages that will be retrieved and displayed on the Search Results Page.
- **Query Results** – Displays the log messages retrieved from the log database based on the Search Criteria.. Paging is available for viewing configurable messages per page.

3.1.Login

The login screen is where users identify themselves and select which application's logs will display. Users must provide a valid username and password, select an application, and click the login button to view related log messages. All three of these fields (username, password, and application) are required. Assuming correct credentials were provided, the Search Criteria and Message screens will display.



LOGIN TO LLT

LOGIN ID

PASSWORD

APPLICATION

Login

3.2. Query Section

The Search Criteria area is where you narrow down the search results to find the set of logs you are interested in viewing. Your username and application name that you provided at the login screen are displayed at the top. The next several fields can be used to filter logs to display only the logs you want to see. For every field of a log message, there is a search criteria that can be used to search that field. After providing the desired Search Criteria, click the Submit button. The logs that match ALL of the given criteria will display in the Message area. Below is a description of each field in the Search Criteria section. Searching is not case sensitive. In addition, any field labeled with the word “Contains” does NOT require an exact match to display a message. Various fields can be secured and hence administrators can be restricted to only view those logs allowed permission. LLT can be secured to administrators to only view logs pertaining to certain allowed fields. Refer the ‘Securing Logs in LLT using CSM’ section.

Message Fields:

- **Application** – the application involved in the log message. This shows the application name that you provided at the login screen and can only be changed at the login screen.
- **LogLevel/Type** – the severity or type of message (Debug, Info, Warn, Error, Fatal).

- **Server** – the host that logged the message
- **User** – the user login name of the person that generated the log message. The entire username must be provided.
- **Organization:** The name of the organization to which the user belongs.
- **SessionID** - the session ID of the user that generated the message. Searching in this field requires the user to enter the entire session ID exactly (no partial-string searches).
- **Message Contains** – the actual log message. If something is created, modified, or deleted, the message will contain information about the object that was changed. For example, if searching for something that has been deleted, then type, “Delete” in the search field. This means any message containing the word “delete” anywhere within the message field of the log will display after clicking Submit.
- **NDC Contains** – the Nested Diagnostic Context of the message. This field is not implemented, but can be configured.
- **Thread Contains-** the name of the execute thread that logged the message
- **Throwable Contains:** the throwable (exception) details if available.

Query Section	
* indicates a required field	
Search Criteria	
* Application	<input type="text" value="test"/>
* LogLevel / Type	<input type="text" value="All Levels"/> ▼
* Server	<input type="text" value="ALL"/> ▼
User	<input type="text"/>
Organization	<input type="text"/>
Session ID	<input type="text"/>
Message contains	<input type="text"/>
NDC contains	<input type="text"/>
Thread contains	<input type="text"/>
Throwable contains	<input type="text"/>
Object ID	<input type="text"/>
Object Name	<input type="text"/>
Operation	<input type="text"/>
* Start Date	<input type="text" value="11/14/2006"/> (MM/DD/YYYY)
* Start Time	<input type="text" value="07:56 PM"/> (HH:MM AM/PM)
* End Date	<input type="text" value="11/14/2006"/> (MM/DD/YYYY)
* End Time	<input type="text" value="08:56 PM"/> (HH:MM AM/PM)
* Records Per Page	<input type="text" value="10"/>
<input type="button" value="Submit"/> <input type="button" value="Reset"/>	

- **Object ID:** For object state log message, the Object ID is the value of the key for IdentifierAttributeKey.
- **Object Name:** For the object state log message, the fully qualified class name for the object whose object state is to be retrieved.
- **Operation:** For the object state log message, the operation performed on the Object. Example: Create/Update/Save/Delete.

- **Start Date/Time** – all messages that have a “Created On” date/time that is between the Start Date/Time and End Date/Time will be displayed in the message area. The default time is one hour from the current date/time.
- **End Date/Time** – displayed messages will be prior to this date/time. If no date/time is provided, the current date/time is assumed.
- **Records per Page** – The records per page to be displayed. Since some search results may number in the thousands, users can keep the maximum number of results low by providing a smaller number. The most current messages will always display at the top, and older messages that exceed the Maximum Number of Results become unattainable. LLT doesn’t retrieve all search results at once



3.3. Query Results

After clicking the Search button, the Message area will display all of the log messages that match all of the provided criteria. Below is a description of each of the fields in a log message.

Message Fields:

- **Server** – the host that logged the message
- **Thread** - the name of the execute thread that logged the message
- **Message** – the actual log message. If something is created, modified, or deleted, the message will contain information about the object state before and after the alteration.
- **Throwable** – the associated stack trace message logged with throwable objects. . This field is not implemented, but can be configured.
- **Application** – the application involved in the log message. This shows the application name that you provided at the login screen.
- **NDC** – the Nested Diagnostic Context of the message. This field is not implemented, but can be configured.
- **LogLevel/Type** – the severity or type of message (Debug, Info, Warn, Error, Fatal)
- **Username** – the user who generated the log message
- **Created On** – the date and time that the message was generated
- **SessionID** - the session ID of the user that generated the message.
- **Object ID** – For object state log message, the Object ID is the value of the key for IdentifierAttributeKey.
- **Object Name** – For the object state log message, the fully qualified class name for the object whose object state is to retrieved.
- **Operation** - For the object state log message, the operation performed on the Object. Example: Create/Update/Save/Delete.

- **Object Attribute Details** – For the object state log message, a collapsible table displaying the object attribute name, previous value and current value.

Search Results			
			Page 1 of 32 Next >
Server:	6116-Copene-02	Application:	test
Created On:	10/30/2006 , 4:01 PM	Username:	NAME2
Session ID:	sessionId2	Thread:	Thread-2
Message:			
IIDC:			
Throwable:			
Object ID:	New Item	Object Name:	test.application.domainobjects.Item
		Operation:	INSERT
Object Attribute Details 			
Server:	6116-Copene-02	Application:	test
Created On:	10/30/2006 , 4:01 PM	Username:	NAME2
Session ID:	sessionId2	Thread:	Thread-2
Message:			
IIDC:			
Throwable:			
Object ID:	Bill Burke	Object Name:	test.application.domainobjects.Customer
		Operation:	UPDATE
Object Attribute Details 			
Attribute	Current Value	Previous Value	
items			
last	Burke		
city	Springfield		
street	1 Boston Road		
first	Bill		
zip	02115		
state	MA		
Server:	6116-Copene-02	Application:	test
Created On:	10/30/2006 , 4:01 PM	Username:	NAME2
		Organization:	OrganizationA
		Level:	WARN

4. Deployment Models

4.1. Release Contents

The log locator tool is released as a compressed web application in the form of a WAR (Web Archive) File. Along with the WAR, the release includes sample configuration files that help developers configure the log locator with their application(s).

The release contents of the log locator can be found in the `llt.zip` file or on the NCICB download site(<http://ncicb.nci.nih.gov/download/index.jsp>). The log locator Release contents include the files in Table 4.4.

<i>File</i>	<i>Description</i>
<code>llt.war</code>	The Log Locator Web Application
<code>mysql-ds.xml</code>	This file contains information for creating a datasource. One entry is required for each database connection. Place this file in the JBoss deploy directory.

Table 4- 4 Log Locator release contents

4.2. Deployment Checklist

Before deploying the Log Locator Tool, the following environment and configuration conditions are required.

1. Environment
 - o JBoss 4.0 Application Server
 - o MySQL 4.0 (with an account that can create databases)
2. Log Locator Release Components
 - o `llt.war`

4.3. Deployment Steps

Step 1: Prime MySQL

1. Log into the database using an account id that has permission to add data to the table.
2. If the log database is not already created then run the `CLMSchemaMySQL.sql` script, on the database prompt. This will create the CLM database
3. In the `AuthSchemaMySQL.sql` file replace the `<<database_name>>` tag with the name of the CLM Authorization schema. This will create the CSM tables. Configure security using CSM UPT, refer UPT Deployment Guide for details.
4. In the `DataPrimingMySQL.sql` file, replace:

- a. Replace the <<super_admin_login_id>> with the login id of the user who is going to act as the Super Admin for that particular installation/.
 - b. Also provide the first name and last name by replacing <<super_admin_first_name>> and <<super_admin_last_name >>.
 - c. Finally, replace all four instances of <<application_context_name>> with application entry – CLM.
5. Configure security using CSM UPT. Refer the CSM UPT Deployment/Application Developers Guide for Details.

Step 2: Configure Datasource

1. Modify the mysql-ds.xml or oracle-ds.xml file. These files contain information for creating a datasource. One entry is required for each database connection. Edit the file to provide the following values:
 - The name of the Datasource should always be “CLMDS”
 - The <<database_url>> that is pointing to the same database as the JDBC Appender configured above.
 - Provide the <<database_user_name>> and <<database_user_password>> used to connect to the database.

Shown in Figure 4-1 is an example mysql-ds.xml file.

```
<?xml version="1.0" encoding="UTF-8"?>
<datasources>
  <local-tx-datasource>
    <jndi-name>LogLocatorDS</jndi-name>
    <connection-url><<database_url>></connection-url>
    <driver-class>org.gjt.mm.mysql.Driver</driver-class>
    <user-name><<database_user_name>></user-name>
    <password><<database_user_password>></password>
  </local-tx-datasource>
</datasources>
```

Figure 4-1 Example mysql-ds.xml file

2. Place the mysql-ds.xml file in the JBoss deploy directory.

Step 3: Deploy the LLT WAR

1. Copy the llt.war in the deployment directory of JBoss which can be found at {jboss-home}/server/default/deploy/ where {jboss-home} is the base directory where JBoss is installed on the server.

Step 4: Start JBoss

1. Once the deployment is completed, start JBoss. Check the logs to confirm there are no

errors while the LLT application is deployed on the server.

2. Once the JBoss server has completed deployment, open a browser to access the Log Locator. The URL will be `http://<<jboss-server>>/llt`, where the `<<jboss-server>>` is the IP or the DNS name of JBoss Server. The Log Locator's Login Page will display.
3. Enter the username and password created in Step 1. Then select an Application and click the login button.
4. You should be able to login successfully and the Log Locator Search Page should be displayed.

Note: In case of any errors, follow a debugging and trouble shooting procedure to diagnose and solve the issues.

5. Securing Logs in LLT using CSM

In its current release LLT has been upgraded to use CSM for the purpose of both Authentication and Authorization of the users. So before the user can log into LLT, CSM needs to be configured to point to the correct Authentication Source as well as Authorization Database.

5.1. Authentication Settings

The CSM application context name used for LLT is "CLM". Hence in order to configure the authentication source a entry needs to be made in the JAAS Login Config file (`login-config.xml` in case of JBoss). Based on whether LDAP or RDBMS is used as credential provider, appropriate login module needs to be configured in this file. This entry is only used to validate the user credentials.

5.2. Authorization Settings

Once a user is authenticated, then LLT check to see if the user has Authorization to access logs for that Application. For this purpose, you first need to provision the Authorization Data. The User Provisioning Tool provided by the CSM project should be used to Provision the Authorization data. The detailed steps to configure UPT can be obtained from the CSM's Guide for Application Developers. However following are the high level steps to install UPT

1. Obtain the UPT v3.2 release from the download center
2. Deploy the UPT war on the JBoss server.
3. Make an entry with the datasource name as `csmupt` in the `mysql-ds.xml` file. Use the same database parameter as used for the CLM datasource.
4. Also make an entry in the JAAS login config file for the application context name "csmupt". Copy the same configuration as that for CLM application as mentioned in the step above.

5. If you are using the CSM's User Table as your credential provider than the default password for super admin would be "changeme". This password should be changed to
6. Start the JBoss server and log into the UPT application using the super admin user name and password and the application context name as "csmupt".
7. Now search for the application named "CLM" and on the details screen provide all the database parameters for that application. These parameters can be obtained from the mysql-ds.xml file from the deployment step 2.
8. Also create a new user (or you can use the Super Admin user itself) and assign it as the admin of the "CLM" application.

Now you have a user which is admin for the CLM application. This user can now configure the LLT instance to provide various other users permission to view logs for different applications which host their logs in the log database. Also this user can set up additional security to control attributes values which a user can query up on.

1. Log into UPT as the admin for the CLM Application Context.
2. Create a new protection element for the application whose logs are stored in the log database. The object id for this protection element should be "APPLICATION_NAME:" followed by the name of the application. Eg. APPLICATION_NAME:caxchange. Also create a corresponding protection group and assign this protection element to it.
3. Create a role and assign READ privilege to the role.
4. Now create a user which is going to be the admin for the log viewer. And assign him the above created protection group and role. This way that user has permission to view all the logs for caxchange application.
5. If you want to grant access to another application then create a new protection element as shown above and grant access to the user.

Apart from provided an application level access LLT also provides control to limit the values for certain query criteria for each user within a given application.

1. Log into UPT as the admin for the CLM Application Context.
2. Create a new protection element for the attribute value to which the user has access to. The object id for this protection element should be in the following format.

'APPLICATION_NAME:<<application_name>>&ORGANIZATION:<<organization>>'

The above format has an ampersand '&' symbol to separate the application and organization fields to security. Eg. APPLICATION_NAME:caxchange:ORGANIZATION:lombardi.

Also create a corresponding protection group and assign this protection element to it.

3. Following are the attributes of the query criteria for which security can be provided as explained in step 2. Note that the values in the parenthesis are values that should be prefixed before the actual value of that attribute while creating the protection element
 - a. Organization (ORGANIZATION)
 - b. User (USER)
 - c. Object Name = (OBJECT_NAME)

So for example if you want to restrict the log viewer admin to be able to query only the logs generated by a particular user 'xyz', then you would create a protection element with the object id as "APPLICATION_NAME:caxchange:USER:xyz" and assign it to the user with the READ privilege