*HPC Data MANAGEMENT*

adminISTRATION guide

Version *1.9*

*02/26/2018*

**Version History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version Number** | **Implemented**  **By** | **Revision**  **Date** | **Approved**  **By** | **Approval**  **Date** | **Description of Change** |
| 1.0 | Prasad Konka | 3/9/2016 |  |  | Initial Draft |
| 1.0 | Zhengwu Lu | 3/16/16 |  |  | Review |
| 1.0 | Prasad Konka | 3/17/16 |  |  | Updates |
| 1.0 | Prasad Konka | 3/21/16 |  |  | Updates from George Zaki comments |
| 1.0 | Zhengwu Lu | 11/27/16 |  |  | Minor update |
| 1.0 | Prasad Konka | 2/23/2017 |  |  | API Security |
| 1.1 | Prasad Konka | 5/22/2016 |  |  | Service account, database setup |
| 1.2 | Zhengwu Lu/Prasad Konka | 6/14/2017 |  |  | Update for 1.2 release |
| 1.3 | Prasad Konka | 7/13/17 |  |  | Updates addressing Zhengwu comments |
| 1.3 | Prasad Konka | 9/19/2017 |  |  | Updated application/system properties |
| 1.3 | Prasad Konka | 9/26/2017 |  |  | Update, delete, general info |
| 1.4 | Prasad Konka | 10/30/2017 |  |  | Addressed reviewers comments |
| 1.4.1 | William Liu | 11/30/2017 |  |  | Added notes about how SSH connecting to production servers may require first SSH connecting to DEV/UAT server. |
| 1.5 | Prasad Konka | 12/15/2017 |  |  | Reorganizing sections, adding missing configuration steps, adding APPENDIX |
| 1.5.1 | Doug Kanoza | 1/12/2018 |  |  | Added some additional notes to CentOS setup in Appendix A |
| 1.6.0 | Eran Rosenberg | 1/16/2018 |  |  | Added section to check Globus app account transfer |
| 1.6.1 | William Liu | 1/17/2018 |  |  | Added to section about building web GUI application to indicate that proper Maven build profile should be applied based on environment. |
| 1.6.2 | William Liu | 1/18/2018 |  |  | Revised section (3.7) about building and deploying Web UI Application to specify how to configure the Application for various environments and to improve organization. |
| 1.7 | William Liu | 2/2/2018 |  |  | Updates to cover administration duties related to Globus application accounts pooling. |
| 1.8 | Sunita Menon | 2/12/2018 |  |  | Added UAT System Configuration |
| 1.9 | Sunita Menon | 02/26/2018 |  |  | Added section for setting up HPCDME as a service. |
| 1.10 | William Liu | 02/28/2018 |  |  | Updates about SSL certificates generation via Maven build process. |

TABLE OF CONTENTS

[1. Purpose 6](#_Toc507459766)

[2. Introduction 6](#_Toc507459767)

[2.1. What is HPC DME? 6](#_Toc507459768)

[2.2. Intended Users 6](#_Toc507459769)

[3. Setting up HPC DME 6](#_Toc507459770)

[3.1. Pre-requisites for API server 6](#_Toc507459771)

[3.1.1 Java 7](#_Toc507459772)

[3.1.2 Install Maven 7](#_Toc507459773)

[3.1.3 Install Git Client 7](#_Toc507459774)

[3.1.4 Install ServiceMix 7](#_Toc507459775)

[3.2 Set up irods Database server 8](#_Toc507459776)

[3.2.1 Install PostgreSQL 8](#_Toc507459777)

[3.3 Set up irods server 8](#_Toc507459778)

[3.3.1 Install iRODS 8](#_Toc507459779)

[3.3.2 Install iRODS iCommands CLI 8](#_Toc507459780)

[3.3.3 Set up SSL Cert for iRODS 9](#_Toc507459781)

[3.4 Set up HPC DME Environment 10](#_Toc507459782)

[3.4.1 Setup HPC DME Database 10](#_Toc507459783)

[3.4.2 SSL Setup on HPC DME 10](#_Toc507459784)

[3.4.3 Set HPC DME Services properties 12](#_Toc507459785)

[3.4.4 Prepare Initial Data 15](#_Toc507459786)

[3.4.4.1 iRODS Service User 15](#_Toc507459787)

[3.4.4.2 Enable Permissions inheritance on iRODS 15](#_Toc507459788)

[3.4.4.3 Setup DOC 15](#_Toc507459789)

[3.4.4.4 Create Admin User 15](#_Toc507459790)

[3.4.5 NIH AD Service Account 16](#_Toc507459791)

[3.4.6 Password Policy 17](#_Toc507459792)

[3.5 Build and Deploy HPC DME 17](#_Toc507459793)

[3.5.1 Build HPC DME 17](#_Toc507459794)

[3.5.2 Deploy HPC DME into ServiceMix 17](#_Toc507459795)

[3.5.3 Run ServiceMix in the background 17](#_Toc507459796)

[3.5.4 Deploy HPCDME with Servicemix running as a service 18](#_Toc507459797)

[3.5.5 Verification step 19](#_Toc507459798)

[3.6 Set up Service accounts 19](#_Toc507459799)

[3.6.1 Setup or Update Cleversafe service account credentials 20](#_Toc507459800)

[3.6.2 Setup or Update Globus service account credentials 20](#_Toc507459801)

[3.6.3 Setup or Update iRODS Service account credentials 21](#_Toc507459802)

[3.7 Build and Deploy WEB UI Application 22](#_Toc507459803)

[3.7.1. Building Web UI Application 22](#_Toc507459804)

[3.7.2. Deploying Web UI Application 23](#_Toc507459805)

[3.7.3. About LDAP/AD Authentication 24](#_Toc507459806)

[4 Accessing HPC DME 24](#_Toc507459807)

[5 HPC DME Services ADMIN Operations 24](#_Toc507459808)

[5.1 Managing Security 24](#_Toc507459809)

[5.1.1 Roles 25](#_Toc507459810)

[5.1.2 Service API Security 26](#_Toc507459811)

[5.2 Create a User 28](#_Toc507459812)

[5.3 Updating an Existing User 29](#_Toc507459813)

[5.4 Assigning permissions to HPC DME Group administrators 29](#_Toc507459814)

[5.5 Create System Administrator group and assign System Admins 30](#_Toc507459815)

[5.6 Accessing HPC DME Database 31](#_Toc507459816)

[5.7 Register a DOC Or Setting up Base Path 31](#_Toc507459817)

[5.8 Updating Existing DOC Configuration 36](#_Toc507459818)

[5.9 Connecting to iRODS through iCommand 37](#_Toc507459819)

[5.10 Delete or Edit a Collection on iRODS 37](#_Toc507459820)

[5.11 Delete or Edit a Data File on iRODS and Cleversafe 38](#_Toc507459821)

[5.12 Globus Application Account Configuration 38](#_Toc507459822)

[5.13 Access Globus Tasks 43](#_Toc507459823)

[5.14 Accessing Application Log files 43](#_Toc507459824)

[5.15 Clean up and reset initial test data load for Development, UAT Tiers 44](#_Toc507459825)

[5.16 General Configuration 44](#_Toc507459826)

[5.17 Technology Stack 44](#_Toc507459827)

[4. APPENDIX A 45](#_Toc507459828)

[5. APPENDIX B 48](#_Toc507459829)

[6. Appendix C: uat sYSTEM configuration 55](#_Toc507459830)

[7. APPENDIX D: Setting up HPCDME as a Service 56](#_Toc507459831)

# Purpose

The HPC DME Admin guide describes pre-requisites and installation instructions to setup HPC Data Management Services. The Admin guide further details different functions admin users would be performing to support HPC DME user needs.

# Introduction

## What is HPC DME?

The HPC DME, High Performance Computing Data Management Environment (Services), is a highly adaptable and an open ended data storage environment supporting storage and management of data, produced from high performance computing or other research systems. HPC DME provides capabilities for storing, managing, transferring and sharing data across different systems securely and efficiently.

Users can store data objects for a long term on HPC DME, share and transfer their data such that they do not have to redistribute or maintain copies of the data on other systems by eliminating the data integrity issues. HPC DME stores and associates user defined metadata to any registered data at different levels of data life cycle, enabling the system not only to help identify the data but also to enhance the search capabilities and to be able to attach a value factor to each dataset.

HPC DME has two main components, HPC DME API – RESTful API and HPC DME Web – Web UI interacting with the API.

## Intended Users

The HPC DME has been built to cater to the data storage and data management needs of NCI cancer community. Any user with a valid HPC DME user account can run its client interface. This guide intends to guide HPC system administrators in managing user accounts, security and data integrity.

Note: HPC DME supports only NIH active directory account holders at this point of time.

# Setting up HPC DME

## Pre-requisites for API server

General pre-requsities:

* Ensure you have sudo permissions in the new environment.
* Ensure that the ncif-hpcdm-svc account is created and you can su to it.
* Ensure that port 7738 (or whatever port you choose for the API server) is open.
* Ensure that the server is able to communicate with the LDAP server at ldaps://ldapad.nih.gov:636
* Put in a request to the Help Desk to mount the following drives in the new environment:   
  **at-s-is2s:/ifs/projects/IRODsTest/archive   20T 1004G   20T   5% /mnt/IRODsTest  
  at-s-is2s:/ifs/projects/IRODsTest/scratch   20T 1004G   20T   5% /mnt/IRODsScratch**
* Request a firewall exception for port 8080 (tomcat)

Please see APPENDIX A for set up instructions on CentOS 7.4.

### Java

Download and install Java 1.8.x

<https://www.java.com/en/download/>

Please visit the following link for installation instructions

<https://www.java.com/en/download/help/index_installing.xml>

After successful installation of Java, update PATH with Java executable folder (Ex: c:/jdk1.8\_13/bin)

<https://java.com/en/download/help/path.xml>

Open cmd/shell window and type “java -version” to make sure it displays the version information you installed.

### Install Maven

Install Maven 3.3.9: [https://maven.apache.org/download.cgi#](https://maven.apache.org/download.cgi)

After successful installation of Maven, update PATH with Maven executable folder (Ex: c:/apache-maven-3.3.9/bin)

Open cmd/shell window and type “mvn -version” to make sure it displays the version information you installed.

### Install Git Client

For Windows: <https://tortoisegit.org/>

Ubuntu: sudo apt-get -y install git

### Install ServiceMix

Downlad Servicemix 7.0.0: <http://servicemix.apache.org/downloads.html>

Extract ServiceMix to any folder.

## Set up irods Database server

### Install PostgreSQL

Install PostgreSQL 9.5.x for iRODS iCAT database

<https://help.ubuntu.com/community/PostgreSQL>

## Set up irods server

### Install iRODS

Download iRODS 4.2.0 from the following link:

<http://irods.org/download/>

There are two types of iRODS server installation, iCAT and Resource Server. HPC DME uses the simplest iRODS installation consists of one iCAT server and zero Resource servers.

Please visit following link for detailed instructions on how to install iRODS iCAT and Resource servers. Current implementation of HPC DME supports setting up one iRODS zone with no resource servers: <https://docs.irods.org/master/manual/installation/>

### Install iRODS iCommands CLI

If you are planning to interact with iRODS remotely, download iCommands CLI package from <http://irods.org/download/>

The iCommands client is a command line interface that can be installed on Unix-based operating systems (Linux, Solaris, Macintosh, AIX) as well as Windows platforms.

This package installs just the client iCommands. Install this package if you want to have a system-wide installation of the iCommands (in /usr/bin), but do not want the entire iRODS iCAT or resource server. The iCommands are already included in both the iCAT and Resource packages – so installing this package in addition is unnecessary (and will conflict).

Once the iCommands client has been installed, you must then configure the iRODS environment. This can be done by editing $HOME/.irods/irods\_environment.json. You can set the contents of this file using the following configuration.

{

"irods\_host": "<iRODS host name>",

"irods\_port": <iRODS port>,

"irods\_user\_name": "rods",

"irods\_zone\_name": "<Zone name>"

}

The default installation of iRODS comes with a system user account 'rods' with a default password 'irods'. You should change the password before letting anyone else into the system:

iinit - Enter rods user password

irods@hostname:~/ $ iadmin moduser rods password <newpassword>

To make sure everything is successful, you will need to re-authenticate and check the new connection:

irods@hostname:~/ $ iinit

Enter your current iRODS password:

irods@hostname:~/ $ ils

/tempZone/home/rods:

“rods” user is defaulted with “rodsadmin” privileges. You should not use “rods” user account for your system admin functions. It is recommended to create a separate admin user account associated with NIH ID for your system admin functions.

### Set up SSL Cert for iRODS

HPC DME connects to iRODS over SSL to authenticate users. Please follow the instructions below to setup SSL on iRODS.

<https://docs.irods.org/4.1.7/manual/authentication/>

openssl genrsa -out server.key

openssl req -new -x509 -key server.key -out server.crt -days 365

openssl dhparam -2 -out dhparams.pem 2048

vi ./var/lib/irods/.irods/irods\_environment.json

Add following statements with valid path to the certificate files

"irods\_ssl\_certificate\_chain\_file": "/etc/certs/server.crt",

"irods\_ssl\_certificate\_key\_file": "/etc/certs/server.key",

"irods\_ssl\_dh\_params\_file": "/etc/certs/dhparams.pem",

"irods\_ssl\_verify\_server": "cert",

Restart iRODS

/etc/init.d/irods restart

## Set up HPC DME Environment

* Set your $PATH environment variable

If you are using Windows, set the following environment variables

MAVEN\_HOME=<Path to Maven folder>

SERVICE\_MIX\_HOME=<Path to ServiceMix folder>

PATH=%MAVEN\_HOME%/bin;%SERVICE\_MIX\_HOME%/bin;%PATH%

If you are using Unix, add the following to your shell configuration file.

export MAVEN\_HOME=<Path to Maven folder>

export SERVICE\_MIX\_HOME=<Path to ServiceMix folder>

export PATH=$MAVEN\_HOME/bin:$SERVICE\_MIX\_HOME/bin:$PATH

* First, Clone the code from HPC DME GIT repository:

Repository URL: <https://github.com/CBIIT/HPC_DME_APIs>

git clone -b releases/1.5.0 <https://github.com/CBIIT/HPC_DME_APIs.git>

* Set HPC\_HOME environment variable to the ‘src’ directory path in the source tree.

### Setup HPC DME Database

HPC DME users iRODS iCAT database to host its own schema. Once iCAT database is setup, please run the following scripts to create HCP DME custom schema in iCAT Postgres database.

<HPC\_HOME>\hpc\hpc-server\hpc-dao-impl\src\main\scripts

If you are doing a new setup, please run all the scripts from “schema” subfolder. If you have already had HPC DME database schema setup, run scripts from migration subfolder based on your current version.

psql -d ICAT -f <HPC\_HOME>\hpc\hpc-server\hpc-dao-impl\src\main\scripts schema/hpc\_dme\_main.sql

Make sure to grant ALL permissions on the “HPC\_\*” tables in the schema to the iRods database user created in step 3.3.

### SSL Setup on HPC DME

By default, HPC DME API server port is secured through a self-signed certificate. With the proper Maven profile active (named generate-keystore-jks-files), this certificate is generated and stored in a key store during the Maven-based application build process. If you want to setup a CA cert instead of using default self-signed certificate, please follow the instructions from the section 3.4.2.1 below.

HPC DME API server can be setup to communicate over SSL with its integrating components like Cleversafe, Active Directory for authentication, etc. For setup instructions, please see section 3.2.2.2 below.

#### Set up CA Certificate for SSL port

Copy cacerts from <JRE\_HOME>/lib/security to <HPC\_HOME>/certs folder.

1. Locate Certificate files:
   1. Primary certificate – host.domain.crt
   2. Intermediate certification – DigiCertCA.crt
   3. Private Key – host.domain.key
2. Import Intermediate Certification

*keytool –importcert –trustcacerts –file DigiCertCA.crt –alias root –keystore* <HPC\_HOME>/certs/cacerts  
You will be asked to enter a password for the keystore: default password is “changeit”.

1. Combine Primary Certificate and Private Key

*openssl pkcs12 –export –name [domain] –in [host.domain.crt] –inkey [host.domain.key] –out keystore.p12*  
You will be asked to enter password. Enter a password and note it down.

1. Import the Primary Certificate & Private Key into the Keystore

*keytool –importkeystore –destkeystore* <HPC\_HOME>/certs/cacerts *–srckeystore keystore.p12 –srcstoretype pkcs12 –alias [domain]*

You will be asked to provide passwords.

You now have cacerts that contains the cert.

#### Import SSL Certs into Java Keystore

HPC DME API can be setup to communicate with Cleversafe, iRODS, NIH AD over SSL to enable secure communication.

For HPC DME instance at FNLCR: Cleversafe, iRODS and NIH AD certs are located on fr-s-hpcdm-uat-p at /opt/certs

* + 1. Please import NIH AD SSL Cert into Java keystore

keytool -importcert -file NIHDeviceRoot.pem -keystore <HPC\_HOME>/certs/cacerts -alias NIHRoot

keytool -importcert -file NIHDeviceIntermediate.pem -keystore <HPC\_HOME>/certs/cacerts -alias NIHIntermediate

* + 1. Please import Cleversafe SSL Cert into Java keystore

keytool -importcert -file Cleversafe.crt -keystore <HPC\_HOME>/certs/cacerts -alias Cleversafe

* + 1. Please import iRODS SSL Cert into Java keystore

keytool -importcert -file iRODS.pem -keystore <HPC\_HOME>/certs/cacerts -alias iRODS

#### Configure HPC-DME properties to use generated keystore

Edit < SERVICEMIX\_HOME >\etc\system.properties to set SSL properties. Add following lines at the end.

# HPC-DME certs to access iRODS, LDAP, etc

javax.net.ssl.keyStore=${karaf.home}/etc/hpc-server/cacerts.jks

javax.net.ssl.keyStorePassword=changeit

javax.net.ssl.trustStore=${karaf.home}/etc/hpc-server/cacerts.jks

javax.net.ssl.trustStorePassword=changeit

### Set HPC DME Services properties

HPC DME build scripts supports preparing the build for “dev”, “preprod” and “prod” environments.

Edit < HPC\_HOME >\ hpc-server\hpc-features\src\main\resources\<properties file> to set system configuration properties.

To set application property, you would need to uncomment it and set a valid value to it. If a property is commented, it uses its default value.

The following table shows minimum set of properties you would need to change. For full list of properties, please see APPENDIX B

|  |  |  |
| --- | --- | --- |
| Property Name | Description | Default Value |
| hpc.ws.rs.host | This is host name of the machine where HPC DME Rest services will be running | Localhost |
| hpc.ws.rs. port | This is the secured port of the machine where HPC DME Rest services will be running. | 7738 |
| hpc.ws.rs.path | HPC DME Server URL context name | <context name>. This can be left blank |
| hpc.ws.rs.ldapAuthentication | Enable Active Directory/LDAP Authentication. If LDAP authentication is set to false, HPC DME uses authentication based on iRODS accounts | true | false  If false, ignore LDAP settings below |
| hpc.ws.rs.attachment-tmp-directory | Temporary staging directory where servicemix writes files while receiving from a client request |  |
| hpc.ws.rs.ssl.keystore.path | Name of the keystore where HPC DME certificate is stored | Path of the certificate created in section 3.2.2. If you are using default self-signed, use default vlaues |
| hpc.ws.rs.ssl.keystore.type | Type of the keystore where HPC DME certificate is stored | JKS |
| hpc.ws.rs.ssl.keystore.password | Password of the keystore where HPC DME certificate is stored | Key store password. If you are using default self-signed, use default vlaues |
| hpc.ws.rs.ssl.truststore.path | Name of the trust store where HPC DME certificate is stored | Path of the certificate created in section 3.2.2. If you are using default self-signed, use default vlaues |
| hpc.ws.rs.ssl.truststore.type | Type of the trust store where HPC DME certificate is stored | JKS |
| hpc.ws.rs.ssl.truststore.password | Password of the trust store where HPC DME certificate is stored | Key store password. If you are using default self-signed, use default vlaues |
| **LDAP/AD settings** | | |
| hpc.integration.ldap.URL | AD/LDAP Server URL | Ex: ldaps://ldapad.nih.gov:636 |
| hpc.integration.ldap.username | User name to connect to AD Server |  |
| hpc.integration.ldap.password | User password to connect to AD server |  |
| hpc.integration.ldap.base | Base name | Ex: OU=NIH,OU=AD,DC=nih,DC=gov |
| hpc.integration.ldap.userIdFilter | UserId filter | Ex: cn |
| hpc.integration.ldap.userIdDomainName | Email address domain name | Ex: nih.gov |
| hpc.service.notification.mail.host | Email host name to send out email notifications | mailfwd.nih.gov. HPC DME application sends email notifications to <userId>@< hpc.integration.ldap.userIdDomainName>  Ex: jdoe@nih.gov |
| hpc.bus.aspect.systemAdministratorUserId | System error notifications are sent to the email given here. Please mention email name without domain name | Ex:HPC\_DME\_Admin  Email will be sent to  HPC\_DME\_Admin@nih.gov |
| **iRODS Settings** | | |
| hpc.integration.irods.host | iRODS host name | DEV - fr-s-hpcdm-gp-d.ncifcrf.gov  UAT - fr-s-hpcdm-uat-p.ncifcrf.gov  Production - fr-s-hpcdm-irods-p.ncifcrf.gov |
| hpc.integration.irods.port | iRODS host port number | 1247 |
| hpc.integration.irods.zone | iRODS zone name | tempZone |
| hpc.integration.irods.resource | iRODS resource name | demoResc |
| hpc.integration.irods.basePath | iRODS zone base path | / tempZone/home |
| **Database Settings** | | |
| hpc.dao.postgresql.dbName | iRODS iCAT database name |  |
| hpc.dao.postgresql.host | iRODS iCAT database host name |  |
| hpc.dao.postgresql.port | iRODS iCAT database port number |  |
| hpc.dao.postgresql.username | iRODS iCAT database user name |  |
| hpc.dao.postgresql.password | iRODS iCAT database user password |  |
| hpc.dao.postgresql.encryptor.key | Encryptor key used to encrypt integrating system credentials | \*\*\*128-bit-key\*\* |
| **Globus Settings** | | |
| hpc.integration.globus.download.directory | Archive directory on the mounted drive of the Globus endpoint. | /mnt/IRODsTest/FNL\_SF\_Archive/HPCDME\_PROD/Download |
| **Authentication Token** | | |
| hpc.service.security.authenticationTokenSignatureKey | Key to generate authenticate token | Make sure to rotate this key periodically |

### Prepare Initial Data

### iRODS Service User

If you are using LDAP authentication and iRODS PAM authentication, use LDAP UserId to create the account.

Login into iRODS icommands to create service user account.

* 1. SSH to iRODS machine
     + In order to establish SSH connection to production iRODS machine, one must first SSH to either the development (DEV) virtual machine or the user acceptance testing (UAT) virtual machine. Then, from the connected virtual machine, regardless of whether DEV or UAT, one may SSH a second time to the production iRODS machine. This is due to how the production iRODS machine is configured for SSH connectivity.
  2. sudo su
  3. su irods
  4. iadmin mkuser <irodsserviceuser> rodsadmin
  5. iadmin moduser <irodsserviceuser> password <hpcdme123> (For PAM authentication, SKIP this step)

### Enable Permissions inheritance on iRODS

Login onto iRODS through iCommands as listed in the step 3.2.4.1 and type the following command to enable permissions inheritance in iRODS.

ichmod -r inherit /tempZone/home

### Setup DOC

Please follow instructions in section 5.7. Please skip step 6 in that section and get back here.

### Create Admin User

HPC DME users are configured to get authenticated with NIH AD pool. When HPC DME is deployed initially, it does not have any accounts created. In order to set up a system admin account, please perform following steps one time only.

1. Login into iRODS through its iCommands CLI and execute following command.

iadmin mkuser <NCI UserId>#<Zone name> rodsadmin

If you are not using LDAP authentication set user password:

iadmin moduser <UserId> password <Password>

Following are the zone names in different environments:

DEV: tempZone

UAT: ncifHpcZone

Production: ncifprodZone

1. Assign permission on iRODS home to System Admin

ichmod -r own <adminUserId> <iRODS home>

1. Login into PostgreSql database.
   1. psql <userId> -h <host> -d <database>
   2. After successfully login into the database, execute following statement

INSERT INTO public."HPC\_USER"(

"USER\_ID", "FIRST\_NAME", "LAST\_NAME", "CREATED", "LAST\_UPDATED", "DOC", "ACTIVE", "ACTIVE\_UPDATED\_BY", "DEFAULT\_CONFIGURATION\_ID")

VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?);

Ex: INSERT INTO public."HPC\_USER"(

"USER\_ID", "FIRST\_NAME", "LAST\_NAME", "CREATED", "LAST\_UPDATED", "DOC", "ACTIVE", "ACTIVE\_UPDATED\_BY", "DEFAULT\_CONFIGURATION\_ID")

VALUES ('<UserId>', '<First>', '<Last>', '12-14-2017', '12-14-2017',

'<DOC>', true, null, null);

### NIH AD Service Account

HPC DME API integrates with NIH AD for user authentication. To connect to NIH AD Pool, HPC DME API users [ncihpcdmsvcad@nih.gov](mailto:ncihpcdmsvcad@nih.gov) service account to connect to it. Following are the application properties used to connect to NIH AD Pool. Please see section 3.2.3 to set application properties.

hpc.integration.ldap.URL=ldaps://ldapad.nih.gov:636

#hpc.integration.ldap.URL=ldaps://NIHDCADHUB3.nih.gov

hpc.integration.ldap.username=ncihpcdmsvcad@nih.gov

hpc.integration.ldap.password=<Password>

hpc.integration.ldap.base=OU=NIH,OU=AD,DC=nih,DC=gov

hpc.integration.ldap.userIdFilter=cn

hpc.integration.ldap.userIdDomainName=nih.gov

### Password Policy

CleverSafe credentials are generated by ITOG Group and they are in the format of S3 keys. Again, whenever there is an update to these keys, they need to be updated in HPC DME database. Please use HPC DME API specification to update Globus credentials in HPC DME database.

iRODS is setup to use PAM authentication with NIH AD. So iRODS uses NIH AD service account credentials listed in section 5.16 below. Following NIH password policy, this account credentials must be updated using step 3.6.3 mentioned below.

## Build and Deploy HPC DME

### Build HPC DME

Open cmd window and navigate to $HPC\_HOME/

Run “*mvn clean install -P<env>[,generate-keystore-jks-files]*”

The -P option is to declare which Maven profiles to activate.

Valid values for <env> are dev, preprod, prod; each is a profile representing an

deployment environment/tier.

The generate-keystore-jks-files profile is optional and should be

specified when generation of keystore files during the build process is desired.

### Deploy HPC DME into ServiceMix

Run Servicemix from the cmd prompt:

If you did not put SERVICEMIX\_HOME as part of your env PATH, go to SERVICEMIX\_HOME and type

./bin/servicemix

After successful start, type the following commands in the Servicemix Console

*feature:repo-add mvn:gov.nih.nci.hpc/hpc-features/1.5.0-RELEASE/xml/features*

*feature:install hpc-server-rest-services*

### Run ServiceMix in the background

You may use following script to run ServiceMix as a background process. Edit and save the script below into a file. Run it as “sh <filename>”

|  |
| --- |
| *echo 'Stopping ServiceMix..'*  *./bin/stop*  *sleep 3*  *NOW=$(date +"%m-%d-%Y"+"%T")*  *FILE="backup.$NOW"*  *#mkdir backuplogs/$FILE*  *#cp -r data/log backuplogs/$FILE*  *rm -rf ./data*  *#start in background*  *echo 'Starting ServiceMix..'*  *./bin/start*  *# wait till SSH daemon is up*  *sleep 60*  *# run your commands*  *./bin/client "feature:repo-add mvn:gov.nih.nci.hpc/hpc-features/1.5.0-RELEASE/xml/features"*  *sleep 30*  *./bin/client "feature:install hpc-server-rest-services"*  *sleep 30*  *./bin/client "feature:install hpc-server-scheduler"* |

### Deploy HPCDME with Servicemix running as a service

ServiceMix can be setup to run as a service as indicated in Appendix D. The below script can then be run with the <present version> and <new version> replaced with the correct values. Edit and save the script below into a file with ‘sh’ extension and run it as

./<filename>.sh

|  |
| --- |
| *echo 'Restarting ServiceMix....'*  */etc/init.d/KARAF-service restart*  *# wait till SSH daemon is up*  *sleep 30*  *# run your commands*  *./bin/client -h 127.0.0.1 "feature:repo-remove mvn:gov.nih.nci.hpc/hpc-features/<present version>/xml/features"*  *sleep 10*  *./bin/client -h 127.0.0.1 "feature:repo-add mvn:gov.nih.nci.hpc/hpc-features/<new\_version>/xml/features"*  *sleep 30*  *./bin/client -h 127.0.0.1 "feature:install hpc-server-rest-services"*  *sleep 30*  *./bin/client -h 127.0.0.1 "feature:install hpc-server-scheduler"* |

### Verification step

Run following curl command to verify if API is running.

curl -k -G -X GET https://<hpc-server>/<context>/user/<userId> --user <UserId> -H "Accept: application/json"

## Set up Service accounts

HPC DME uses service account to integrate with Globus, Cleversafe and iRODS. While HPC DME security enforces authentication and authorization, the service account is used to perform actions on behalf of the users. For example, the service account is used to upload data into Cleversafe data archive object store. It is used to update system metadata in iRODS. It is also used to create shared endpoints used by HPC DME application account for 2 hop data transfer process. Please section 5.12 for details on application accounts and the setup around 2 hop data transfer. It is also used to manage asynchronous data transfer using Globus. It is important to keep service accout credentials in a safe location. HPC DME stores service accounts (Globus, Cleversafe and iRODS) information in its database in an encrpted manner. HPC DME provides a REST API to register service account with the system initially. This API is accessible to System Administrator role only. Please see API Specification section below for API details.

The following service account credentials are expected to be setup and attached with the service account for HPC DME to function properly.

1. Cleversafe AWS access key and secret access key

2. iRODS user name (rods) and password with “rodsadmin” role and has “own” access to the root folder

3. Globus Application account Id and Key. Please see section 5.12 to setup Globus Application Id.

Globus user name and password with read/write access to “nihfnlcr#gridftp1”

After updating service account credentials, application should be restarted to get changes in effect.

### Setup or Update Cleversafe service account credentials

|  |
| --- |
| Set Cleversafe credentials into a file called Cleversafe.json  {  "account": {  "username": "AWS Access Key Id",  "password": "AWS Secret access key",  "integratedSystem": "CLEVERSAFE"  },  "dataTransferType" : "S\_3"  }  **Production:**  **AWS Access Key Id:** XmddNHnGowAPX7th9eSb  **DEV/UAT:**  **AWS Access Key Id:** 025ff462-07e1-483b-8dbb-1fc26c7eb17e DelCtesWPOLiDL1aRLh5  Execute following curl command with correct parameter values:  curl -k -H "Content-Type: application/json" -d @Cleversafe.json -X PUT <server URL>/systemAccount --user <userId>:<password> -H "Accept: application/json" |

### Setup or Update Globus service account credentials

|  |
| --- |
| Set Globus application account credentials into a file called Globus.json  {  "account": {  "username": "<Globus Application Client Id>",  "password": "<Globus Application Client Secret>",  "integratedSystem": "GLOBUS"  },  "dataTransferType" : "GLOBUS",  "classifier" : "<Pool Classifier>"  }  **Production:**  **Globus Application Client Id:** 07cec4f4-8d99-4b9d-a513-d9a1148ac054  **DEV/UAT:**  **Globus Application Client Id:** 025ff462-07e1-483b-8dbb-1fc26c7eb17e 025ff462-07e1-483b-8dbb-1fc26c7eb17e  Globus service account uses NIH AD service account credentials.  NIH AD service account is used to create application project and its accounts in <https://developers.globus.org>. Please see section 5.12 for details.  Execute following curl command with correct parameter values:  curl -H "Content-Type: application/json" -d @Globus.json -X PUT <server URL>/ systemAccount --user <userId>:<password> -H "Accept: application/json" |

### Setup or Update iRODS Service account credentials

HPC DME application uses NIH AD account to authenticate with iRODS PAM and to perform service account activities. In order to not to lock yourself into the system, before updating the service account password in NIH AD, first update the application with the new password and then update NIH AD. After updating the password, you will need to restart the application.

|  |
| --- |
| Set iRODS credentials into a file called iRODS.json  {  "account": {  "username": "iRODS UserId",  "password": "iRODS Password",  "integratedSystem": "IRODS"  }  Production Service Account: ncifhpcdmsvcp  DEV/UAT Service Account: ncif-hpcdm-svc  This iRODs service account is a NIH AD account with RODSADMIN privileges.  Execute following curl command with correct parameter values:  curl -H "Content-Type: application/json" -d @iRODS.json -X PUT <server URL>/ systemAccount --user <userId>:<password> -H "Accept: application/json" |

Please restart the application after completing this step. This is to inject application service accounts into the application.

If you have created background script to start and stop servicemix, you could do

sh <script name>

If not, type shutdown and repeat 3.5.2.

## Build and Deploy WEB UI Application

### Building Web UI Application

Before executing the build process, edit the appropriate configuration file. All configuration files are located in the directory, $HPC\_HOME/hpc-web/src/main/resources/appconfigs.

The appropriate configuration file to edit is named <environment\_label>-env.conf, where <environment\_label> is a placeholder for some word conveying which environment. The placeholder should match any one of the following words: prod, preprod, dev, and local.

The following table summarizes the properties to set in the configuration file based on your deployment environment. You do not need to change any other properties, and it is recommended to avoid changing any properties not mentioned in the table below unless you know what you are doing.

|  |  |  |
| --- | --- | --- |
| Property Name | Value | Comments |
| gov.nih.nci.hpc.server | Ex: https://fr-s-hpcdm-gp-d.ncifcrf.gov:7738/hpc-server | API Server URL |
| gov.nih.nci.hpc.web.server | https://fr-s-hpcdm-gp-d.ncifcrf.gov | Web application URL |
| server.port | Ex: 8080 | Web application container port number |
| gov.nih.nci.hpc.ssl.cert | Path to API Server public certificate | This keystore is used to make secure connection with the API server |
| gov.nih.nci.hpc.ssl.cert.password | Key store password |  |

After editing the appropriate configuration file, you may execute the build process. To do so, follow these steps:

1. Open a command window
2. cd $HPC\_HOME/hpc-web
3. Issue the Maven build command as shown on next line

mvn clean build -P<environment\_label>

* 1. <environment\_label> is the same placeholder that was mentioned earlier in this section of this document concerning how to build the HPC Web UI Application

### Deploying Web UI Application

To deploy the war into Tomcat application container

1. Stop Tomcat.
2. If applicable, remove the existing web application from Tomcat. This would be old version of the web application.
   1. Delete exploded web app directory from <Catalina\_home>/webapps/ directory.

It would be named hpc-web-<old\_version\_number>.

* 1. Delete war file from <Catalina\_home>/webapps/ directory.

It would be named hpc-web-<old\_version\_number>.war.

1. Copy war file of the new/current version of the web application to <Catalina\_home>/webapps/ directory.
2. Edit <Catalina\_home>/conf/server.xml to include following statement in the <host> element but replacing the placeholder <version\_number> with the correct version number:

<Context path="" docBase="hpc-web-<version\_number>" privileged="true"/>

1. Start Tomcat.
2. Restart Tomcat to ensure that configuration modification in Step #4 is applied.

To run/deploy as executable jar file (not recommended for production)

HPC DME Web application is built with Spring Boot. Therefore, it can be run as executable jar via a java command similar to the following:

java -jar <HPC\_DME>/src/hpc-web/target/hpc-web-<version\_number>.war

### About LDAP/AD Authentication

HPC DME Web UI application uses token-based authentication supported by PC DME API. This token is only be generated by the API server while it is actively using LDAP/AD authentication. If not, the Web UI application will not work.

# Accessing HPC DME

HPC DME can be accessed through its Service API client interface. HPC DME Service APIs are developed based on REST standard. These APIs can be accessed by any interface that is compatible with REST standard. For example, these Service APIs can be accessed through Java, Perl, Ruby, Curl, JavaScript languages programmatically or through UI clients like SOAP UI, REST Client.

The following is the HPC DME Service API URL:

https://<host name>:<port>/<hpc.ws.rs.path>/<Resource Name>

DEV: https://fr-s-hpcdm-gp-d.ncifcrf.gov:7738/hpc-server/

UAT: <https://fr-s-hpcdm-uat-p.ncifcrf.gov:7738>/hpc-server/

Production: https://hpcdmeapi.nci.nih.gov/

HPC DME can also be accessed through its interactive client interface. Please refer to [HPC DME User guide](https://github.com/CBIIT/HPC_DME_APIs/blob/master/doc/guides/HPC_User_Guide.docx) and [API Specification](https://github.com/CBIIT/HPC_DME_APIs/blob/master/doc/guides/HPC_API_Specification.docx) for details.

# HPC DME Services ADMIN Operations

HPC Admin is responsible for HPC user accounts and authorizations setup to facilitate end users in using the solution to manage scientific data/metadata workflow. Managing hardware, storage, network is out of scope for this document.

## Managing Security

HPC DME has different layers of security. First layer of security is enforced at HPC DME API level through NIH AD credentials. By default, HPC DME is setup to work with NIH AD pool server “ldaps://ldapad.nih.gov:636”. NIH AD connection requires user credentials to be set in application properties file. This service User account also follows NIH password policy. Whenever the password is changed, it has to be updated in gov.nih.nci.hpc.cfg file in <SERVICE\_MIX\_HOME>/etc folder.

HPC DME integrates with iRODS for data management and with Globus for data transfer. iRODS instance supports PAM authentication that integrates its authentication with NIH AD server. Please see the following link to setup PAM authentication for iRODS.

<https://github.com/DICE-UNC/jargon/wiki/Setting-up-PAM>

Execute following command to verify PAM authentication setup

/var/lib/irods/iRODS/server/bin/PamAuthCheck

In order to authenticate iRODS user with NIH AD, iRODS users should be created with NIH loginId only, password is not required.

HPC DME stores iRODS, Globus and Cleversafe service account credentials in its PostgreSql database in encrypted format. HPC DME API uses AES encryption in combination of encryption key set in HPC DME services properties. The following is a property in gov.nih.nci.hpc.cfg. Access to this file should be restricted.

hpc.dao.postgresql.encryptor.key=\*\*\*128-bit-key\*\*

### Roles

HPC DME supports the following roles:

1. USER – HPC DME user role with access to the following functions:
   * Update self-profile (Only through API)
   * Register a data object with metadata
   * Register a collection with metadata
   * Update collection metadata
   * Update metadata on data object
   * Download data object
   * Share data object
   * Search collections and data objects
   * Generate reports
2. GROUP\_ADMIN – HPC DME group admin role with access to following functions in addition to all USER role allowed functions:
   * List group info
   * Create or Delete a group
   * Create user
   * Add user to a group
   * Remove Data Object
   * Query All Active Users
3. SYSYEM\_ADMIN - HPC DME system admin role with access to following functions in addition to all GROUP\_ADMIN role allowed functions:
   * Update User
   * Remove User
   * Query All Users (Including inactive users)
   * Generate system usage reports

HPC DME security is integrated with iRODS security. iRODS roles “rodsadmin”, “groupadmin”, “rodsusers” are mapped to HPC DME types as follows:

SYSTEM\_ADMIN 🡪 rodsadmin

GROUP\_ADMIN 🡪 groupadmin

USER 🡪 rodsusers

### Service API Security

Access to each of HPC DME API services is configured through Spring Security configuration at \hpc-server\hpc-ws-rs-impl\src\main\resources\META-INF\spring\ hpc-ws-rs-bundle-context-osgi.xml. Making a change to this file requires rebuilding the API packages and deploying it to run-time environment to have an effect. This configuration is not expected to change without approval from the Project Management.

|  |
| --- |
| <prop key="hpc.ws.rs.auth.security.register-user">SYSTEM\_ADMIN</prop>  <prop key="hpc.ws.rs.auth.security.update-user">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.security.get-user">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.security.set-group">SYSTEM\_ADMIN GROUP\_ADMIN</prop>  <prop key="hpc.ws.rs.auth.security.authenticate-user">SYSTEM\_ADMIN</prop>  <prop key="hpc.ws.rs.auth.security.register-system-account">SYSTEM\_ADMIN</prop>    <prop key="hpc.ws.rs.auth.data-management.register-collection">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-management.get-collection">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-management.register-data-object">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-management.get-data-object">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-management.download-data-object">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-management.set-permissions">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-management.get-data-management-model">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>    <prop key="hpc.ws.rs.auth.data-search.query-collections">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-search.query-data-objects">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-search.add-query">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-search.update-query">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-search.delete-query">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-search.get-query">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-search.get-queries">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-search.get-metadata-attributes">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.data-search.refresh-metadata-views">SYSTEM\_ADMIN</prop>    <prop key="hpc.ws.rs.auth.notification.subscribe-notifications">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>  <prop key="hpc.ws.rs.auth.notification.get-notification-subscriptions">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop>    <prop key="hpc.ws.rs.auth.report.generate-report">SYSTEM\_ADMIN GROUP\_ADMIN USER</prop> |

## Create a User

Both System Admin and Group Admin can create HPC user accounts. System Admin can do that by accessing HPC DME REST API. Any REST client can be used to access the API. This admin function can also be performed through UI. Please see HPC\_User\_Guide for detailed instructions.

For example, a system admin would perform the following using curl.

* Create input.json with the following content

|  |
| --- |
| {  "firstName": "John",  "lastName": "Doe",  "doc" : "FNLCR",  "userRole" : "SYSTEM\_ADMIN"  } |

Execute the following command:

*curl -H "Content-Type: application/json" -d @input.json -X PUT $<SERVER>/user/<UserId> --user <Admin NCIUserId:Password> -H "Accept: application/json"*

Valid values for “dataManagementUserType” are "SYSTEM\_ADMIN", "GROUP\_ADMIN" and "USER". Known issue: If you create a user with "GROUP\_ADMIN" role, iRODS Jargon API has a known issue in creating that. A workaround is to create user with “USER” role and update User role with “groupadmin” through iCommand interface. Once you create the user, you would need to login into iRODS iCommand and execute following command to assign “groupadmin” role. This is a workaround until the issue with iRODS Jargon API is fixed.

*iadmin moduser <userId>#<ZoneName> Type groupadmin*

Please see User registration API specification for more details on the API and any errors.

## Updating an Existing User

System Admin is responsible for updating HPC user accounts. System Admin can do that by accessing HPC DME REST API. Any REST client can be used to access the API.

For example, a system admin would do the following using curl.

* Create input.json with the following content

|  |
| --- |
| {  "firstName": "Eran",  "lastName": "Rosenberg",  "doc" : "FNLCR",  "userRole" : "SYSTEM\_ADMIN"  } |

Execute the following command:

*curl -H "Content-Type: application/json" -d @input.json -X POST $<SERVER>/user/<UserId> --user <Admin NCIUserId:Password> -H "Accept: application/json"*

Please see User registration update API specification for more details on the API and any errors.

## Assigning permissions to HPC DME Group administrators

In order to complete user registration with HPC DME and start using HPC DME API, a system administrator should complete the following steps for each group administrator created for a DOC.

* Group Admin User should be NCI user with account in NIH AD
* Assign “own” permission to archive folder to the group administrator.

|  |
| --- |
| Edit ad save the following json into permission.json  {  "entityPermissionRequests": [  {  "path": "/<DOC Base Path> ",  "userPermissions": [  {  "userId": "<UserId>",  "permission": "WRITE"  }  ]  }  ]  }  curl -H “Content-Type: application/json” -d @permission.json -X POST <server>/acl -H “Accept: application/json” -D <response-header> -o <response-message.json> |

## Create System Administrator group and assign System Admins

Create System Admin users group and add system admin users to the group. Assign read permission to each DOC basepath.

|  |
| --- |
| 1. Create Group and add user   Edit ad save the following json into sysadmin.json  {  "addUserIds": [ "abc"]  }  curl -H “Content-Type: application/json” -d @ sysadmin.json -X PUT <server>/group/HPC\_DME\_SYSTEM\_ADMINS -H “Accept: application/json” -D <response-header> -o <response-message.json>   1. Assign READ permission to a DOC basepath   Edit ad save the following json into permission.json  {  "entityPermissionRequests": [  {  "path": "/<DOC Base Path> ",  "userPermissions": [  {  "userId": "<UserId>",  "permission": "READ"  }  ]  }  ]  }  curl -H “Content-Type: application/json” -d @permission.json -X POST <server>/acl -H “Accept: application/json” -D <response-header> -o <response-message.json |

## Accessing HPC DME Database

HPC DME database holds iRODS iCAT database and its own custom tables and views. To access the database, following are the steps:

* SSH to database machine
  + In order to establish SSH connection to production database machine, one must first SSH to either the development (DEV) virtual machine or the user acceptance testing (UAT) virtual machine. Then, from the connected virtual machine, regardless of whether DEV or UAT, one may SSH a second time to the production database machine. This is due to how the production database machine is configured for SSH connectivity.
* sudo su
* su postgres
* psql
* \c ICAT

## Register a DOC Or Setting up Base Path

System Admin can register a DOC with the application so that its policies and users can be added to access the application. Registering a DOC is currently a manual process with following steps. A DOC can be configured to have multiple base paths. And each base path can be set to have its own metadata policies and storage type. HPC DME currently supports storing data in S3 compatible object store or any file system mounted with API server.

1. Pick an abbreviated name for DOC in consultation with the users.
   1. Example: CCR (Center for Cancer Research)
2. Login into iRODS icommands to create DOC base path.
   1. SSH to iRODS machine
      * In order to establish SSH connection to production iRODS machine, one must first SSH to either the development (DEV) virtual machine or the user acceptance testing (UAT) virtual machine. Then, from the connected virtual machine, regardless of whether DEV or UAT, one may SSH a second time to the production iRODS machine. This is due to how the production iRODS machine is configured for SSH connectivity.
   2. sudo su
   3. su irods
   4. icd .. (You will be on iRODS home path, ex: /tempZone/home
   5. imkdir <Base path name>
   6. ichmod -r own <iRODS service account> <base path name>
3. Define DOC Base path metadata policy
   1. Metadata policy its hierarchy should be defined in JSON format. Please see Section 4.4 in API Specification document for details.
4. Get DOC Base path data store configuration. Each DOC Base path may have its dedicated S3 based vault.
5. Login into Database to insert DOC record
   1. SSH to database machine
      * In order to establish SSH connection to production database machine, one must first SSH to either the development (DEV) virtual machine or the user acceptance testing (UAT) virtual machine. Then, from the connected virtual machine, regardless of whether DEV or UAT, one may SSH a second time to the production database machine. This is due to how the production database machine is configured for SSH connectivity.
   2. sudo su
   3. su postgres
   4. psql
   5. \c ICAT
   6. Following are the details to create DOC record

| **Column** | **Type** | **Comments** |
| --- | --- | --- |
| ID | text | The configuration ID. Generate your unique ID |
| BASE\_PATH | text | The base path to apply this configuration to |
| DOC | text | The DOC that own this configuration |
| S3\_URL | text | The S3 archive (Cleversafe) URL |
| S3\_VAULT | text | The S3 archive (Cleversafe) vault |
| S3\_OBJECT\_ID | text | The S3 archive (Cleversafe) object id prefix |
| S3\_ARCHIVE\_TYPE | text | The S3 archive type (Archive / Temp Archive). Note: Temp Archive is currently not used |
| DATA\_HIERARCHY | text | The data hierarchy policy |
| COLLECTION\_METADATA\_VALIDATION\_RULES | text | The collection metadata validation rules |
| DATA\_OBJECT\_METADATA\_VALIDATION\_RULES | text | The data object metadata validation rules |
| S3\_UPLOAD\_REQUEST\_URL\_EXPIRATION | integer | Expiration time in hours for S3 pre-signed URL |
| GLOBUS\_URL | text | Globus authentication URL (not null) |
| GLOBUS\_ARCHIVE\_ENDPOINT | text | Globus endpoint used to support asynchronous data registration. This endpoint should be accessible to HPC DME Globus application account |
| GLOBUS\_ARCHIVE\_PATH | text | Folder on the Globus endpoint to store registered data before moving it to the Archive storage |
| GLOBUS\_ARCHIVE\_DIRECTORY | text | Physical path of the Globus storage. This storage should be mounted on API server and make it accessible to HPC DME Service account. **For File System storage, this is the mounted path to the POSIX storage.** |
| GLOBUS\_ARCHIVE\_TYPE | text | ARCHIVE or TEMPORARY\_ARCHIVE. For File System type storage, leave the value as ARCHIVE. |
| GLOBUS\_DOWNLOAD\_ENDPOINT | text | Globus endpoint used to support asynchronous data download. This endpoint should be accessible to HPC DME Globus application account |
| GLOBUS\_DOWNLOAD\_PATH | text | Folder on the Globus endpoint to download registered data before moving it to the user Globus endpoint |
| GLOBUS\_DOWNLOAD\_DIRECTORY | text | Physical path of the Globus storage. This storage should be mounted on API server and make it accessible to HPC DME Service account |

* 1. Execute following SQL statement

INSERT INTO public."HPC\_DATA\_MANAGEMENT\_CONFIGURATION"(

"ID", "BASE\_PATH", "DOC", "S3\_URL", "S3\_VAULT", "S3\_OBJECT\_ID",

"S3\_ARCHIVE\_TYPE", "DATA\_HIERARCHY", "COLLECTION\_METADATA\_VALIDATION\_RULES",

"DATA\_OBJECT\_METADATA\_VALIDATION\_RULES", "S3\_UPLOAD\_REQUEST\_URL\_EXPIRATION",

"GLOBUS\_URL", "GLOBUS\_ARCHIVE\_ENDPOINT", "GLOBUS\_ARCHIVE\_PATH",

"GLOBUS\_ARCHIVE\_DIRECTORY", "GLOBUS\_ARCHIVE\_TYPE", "GLOBUS\_DOWNLOAD\_ENDPOINT",

"GLOBUS\_DOWNLOAD\_PATH", "GLOBUS\_DOWNLOAD\_DIRECTORY")

VALUES (?, ?, ?, ?, ?, ?,

?, ?, ?,

?, ?,

?, ?, ?,

?, ?, ?,

?, ?);

If you are setting up S3 bucket as your Archive:

*Example:*

*INSERT INTO public."HPC\_DATA\_MANAGEMENT\_CONFIGURATION"*

*VALUES ('c93e82ba-7c66-4463-8376-1c7cb0b1a598','/FNL\_SF\_Archive','FNLCR','http://fr-s-clvrsf-01.ncifcrf.gov','DSE-TestVault1','FNL\_SF\_Archive','ARCHIVE','{*

*"collectionType": "PI\_Lab",*

*"isDataObjectContainer": false,*

*"subCollections": [*

*{*

*"collectionType": "Project",*

*"isDataObjectContainer": true,*

*"subCollections": [*

*{*

*"collectionType": "Flowcell",*

*"isDataObjectContainer": true,*

*"subCollections": [*

*{*

*"collectionType": "Sample",*

*"isDataObjectContainer": true*

*}*

*]*

*}*

*]*

*}*

*]*

*}','{*

*"metadataValidationRules": [*

*{*

*"attribute":"collection\_type",*

*"mandatory":true,*

*"validValues":[*

*"Project",*

*"PI\_Lab",*

*"Flowcell",*

*"Sample",*

*"Folder"*

*],*

*"ruleEnabled":true*

*},*

*{*

*"attribute":"pi\_name",*

*"mandatory":true,*

*"collectionTypes":[*

*"PI\_Lab"*

*],*

*"ruleEnabled":true*

*},*

*{*

*"attribute":"project\_id\_CSAS\_NAS",*

*"mandatory":true,*

*"collectionTypes":[*

*"Project"*

*],*

*"ruleEnabled":true*

*}*

*{*

*"attribute":"flowcell\_id",*

*"mandatory":true,*

*"collectionTypes":[*

*"Flowcell"*

*],*

*"ruleEnabled":true*

*},*

*{*

*"attribute":"initial\_sample\_concentration\_ngul",*

*"mandatory":false,*

*"collectionTypes":[*

*"Sample"*

*],*

*"ruleEnabled":true*

*}*

*]*

*}','{*

*"metadataValidationRules": [*

*{*

*"attribute":"object\_name",*

*"mandatory":true,*

*"collectionTypes":[*

*"Sample"*

*],*

*"ruleEnabled":true*

*},*

*{*

*"attribute":"data\_compression\_status",*

*"mandatory":true,*

*"collectionTypes":[*

*"Sample"*

*],*

*"defaultValue":"Unspecified",*

*"validValues":[*

*"Unspecified",*

*"Compressed",*

*"Not Compressed",*

*"Not Specified"*

*],*

*"ruleEnabled":true*

*}*

*]*

*}',24,'https://auth.globus.org/v2/oauth2/token','c6790626-aab4-11e7-aef3-22000a92523b','/ServiceFolderFor2Hop','/mnt/IRODsTest/FNL\_SF\_Share/ServiceFolderFor2Hop','TEMPORARY\_ARCHIVE','c6790626-aab4-11e7-aef3-22000a92523b','/FNL\_SF\_S3\_Download','/mnt/IRODsTest/FNL\_SF\_Share/FNL\_SF\_S3\_Download');*

If you are setting up POSIX File system as your Archive:

*INSERT INTO public."HPC\_DATA\_MANAGEMENT\_CONFIGURATION"*

*VALUES(‘41ff9b3a-dfb1-11e7-80c1-9a214cf093ae','/FS\_ARCHIVE','FS\_ARCHIVE',null,null,null,null,null,null,null,,'https://auth.globus.org/v2/oauth2/token','N/A','/FS\_ARCHIVE','/mnt/FS\_ARCHIVE','ARCHIVE',null,null,null);*

1. Restart API Server
   1. Login into API Server machine
   2. sudo su
   3. su <service account>
   4. cd /opt/apache-servicemix-7.0.0.M3
   5. sh starthpcserver.<Version>.sh

## Updating Existing DOC Configuration

System Admin can update a DOC configuration including its metadata hierarchy, policies, data archive connection settings. Updating a DOC configuration is currently a manual process with following steps.

1. Login into Database to update DOC record
   1. SSH to database machine
      * In order to establish SSH connection to production database machine, one must first SSH to either the development (DEV) virtual machine or the user acceptance testing (UAT) virtual machine. Then, from the connected virtual machine, regardless of whether DEV or UAT, one may SSH a second time to the production database machine. This is due to how the production database machine is configured for SSH connectivity.
   2. sudo su
   3. su postgres
   4. Execute following SQL statement

psql ICAT -c “select \* from \” HPC\_DATA\_MANAGEMENT\_CONFIGURATION \” where \”ID\”=’<UUID>’” > <DOC>.txt

* 1. Edit <DOC>.txt and update values as needed and use those value with the following update statement.
  2. psql ICAT -c “

UPDATE public.\"HPC\_DOC\_CONFIGURATION\" SET (

<Column Name>=<Value> …

) WHERE \"DOC\"=’<DOC>’”

1. Restart API Server
   1. Login into API Server machine
   2. sudo su
   3. su <service account>
   4. cd /opt/apache-servicemix-7.0.0.M3
   5. sh start-deploy-servicemix.sh

## Connecting to iRODS through iCommand

Following are the steps to connect to iRODS through its iCommand. On DEV, UAT and Production environments, iRODS is setup to use PAM authentication. To access iRODS directly, you should have your account (with NIH AD UserId) registered with iRODS. iRODS port is enabled with SSL. To connect to iRODS, you need to have its pubic certificate located at /etc/apache2/ssl folder. Please refer to <https://docs.irods.org/4.2.0/icommands/user/> for commands and details.

1. SSH to iRODS server
   1. In order to establish SSH connection to production iRODS server one must first SSH to either the development (DEV) virtual machine or the user acceptance testing (UAT) virtual machine. Then, from the connected virtual machine, regardless of whether DEV or UAT, one may SSH a second time to the production iRODS server. This is due to how the iRODS server is configured for SSH connectivity.
2. From your home folder, type “mkdir .irods” and then “cd .irods”
3. If you have not already setup iRODs client environment, please follow these steps:
   1. From your home folder, type “mkdir .irods” and then “cd .irods”
   2. vi irods\_environment.json
   3. Paste following text and update highlighted text with correct values

{

"irods\_port": 1247,

"irods\_host": "fr-s-hpcdm-gp-d.ncifcrf.gov",

"irods\_user\_name": "NIH ID",

"irods\_zone\_name": "tempZone",

"irods\_authentication\_scheme": "PAM",

"irods\_ssl\_ca\_certificate\_file": "/etc/apache2/ssl/fr-s-hpcdm-gp-d.ncifcrf.gov.crt",

"irods\_ssl\_verify\_server": "none"

}

* 1. Type “iinit” and enter your NIH AD password
  2. Type any iRODS commands

## Delete or Edit a Collection on iRODS

Following are the steps to delete a collection from iRODS directly.

1. Following steps from section 5.9 to connect to iRODS
2. To recursively remove a collection and its contents and metadata “irm -rf <Collection full path>”
3. To move a collection from one location to another location: “imv <source location> <destination location>
4. To update permissions on a collection and its contents: “ichmod -r <null | read | write | own> <UserId> <Collection path>”
5. To update metadata of a collection: “imeta mod -d <Collection path> <Attribute name> <Attribute value>
6. To add metadata of a collection: “imeta add -d <Collection path> <Attribute name> <Attribute value>

## Delete or Edit a Data File on iRODS and Cleversafe

Following are the steps to delete a data from iRODS directly. Removing a data file from iRODS would delete data file logical reference and its metadata is deleted. Physical data file residing on Cleversafe would not be deleted.

1. Following steps from section 5.9 to connect to iRODS
2. To remove a data file and its metadata “irm <Data file full path>”
3. To remove data file from Cleversafe, use “S3 Browser” or “CyberDuck” to connect to Cleversafe vault with appropriate credentials. Navigate to the data file and right click on the file name to get pop-up menu. Click on delete to permanently delete the file from Archive.

## Globus Application Account Configuration

HPC DME Application uses Globus to transfer data asynchronously to and from the archive. At this time, setting up a Globus endpoint CleverSafe object store is not tried or assessed. To enable asynchronous data transfer, our application took 2 hop data transfer approach.

HPC DME uses multiple groups of Globus application service accounts to perform any transfers through our system Globus endpoint. DEV and UAT environments share a common collection of Globus groups each having multiple Globus application service accounts. Each Globus group is essentially a pool of application service accounts. DEV and UAT environments also share one common Globus service endpoint serving as the staging area to keep the data during 2-hop upload or download process.

Production environment has its own separate set of Globus pools each having multiple Globus application service accounts. The production environment also has its own service endpoint.

HPC DME is using Globus endpoint “nihfnlcr#gridftp1” hosted at Frederick Data center. The file system attached to this Globus endpoint is mounted on API Server to move files for 2 hop transfer using application accounts.

**/mnt/IRODsTest** is the Globus endpoint storage path mounted on both DEV/UAT and Production API Servers.

Following are the steps to setup 2-hop transfer configuration:

Create Globus groups serving as application account pools

1. Log into globus.org using the NIH AD Service account (Production/DEV-UAT) credentials.
2. In the blue menu bar across the top of the screen, click **Groups** to open the **Groups** menu, then click the option **Create New Group**.
3. In the **Name** field, type the name of the group using the following format:

*HPCDME-<which environment>-App-Accts-Pool-<classifier>*

where

*<which environment>* is a placeholder for either "DEV" (non-production) or "PROD"

and

*<classifier>* is a placeholder for the pool's classifier (human-friendly name to refer to the pool).

Note that the pool classifier is expected to be either a DOC name (examples: FNLCR, HiTIF) or the word DEFAULT spelled as shown in all capital letters. If the pool classifier is a DOC name, it must be the DOC name spelled correctly to include letter case. This means if the DOC name is HiTIF, then the pool classifier must also be HiTIF and not any of the following variants: HITIF, hitif, HiTif, and hiTif.

1. (Optional) In the **Description** field, type a useful description for the group.
2. For the **Viewable by** field, select **all Globus users** setting.
3. Click the **Create Group** button to create the group.
4. To the left, the new group should appear in the **My Groups** tab. Select the new group, and in the central section of the screen, click the **Settings** tab.
5. On the **Settings** tab, the first sub-tab is **Policies**. Find the **edit** link on this sub-tab toward the right and click it.
6. Leave the default selections for the first several policy settings, and go to the **Users may request membership if they are** policy setting. Change this setting to **invited to join this group** option.
7. Change the **Membership requests are approved** policy setting to **automatically if all policies are met** option.
8. Click the **Save Changes** button to save the adjusted policies.
9. Repeat Steps 2-11 for each Globus application account pool that shall be utilized.
   1. Globus application service accounts pooling started with Version 1.6.0 of HPC DME. For 1.6.0, the pooling shall be split into 3 pools as noted in the following table.

|  |  |
| --- | --- |
| **Pool Classifier** | **Pool Description** |
| FNLCR | Contains application accounts for serving the DOC known as FNLCR. Make sure classifier spelling exactly matches DOC name's spelling including letter case (all upper case in this context). |
| HiTIF | Contains application accounts for serving the DOC known as HiTIF. Make sure classifier spelling exactly matches DOC name's spelling including letter case (all upper case except the i between H and T). |
| DEFAULT | Contains application accounts for serving any DOC that is none of the above DOCs. Make sure DEFAULT is spelled as shown in all upper case letters. |

1. (For reference after groups have been created) To obtain the UUID of a Globus group, click the group's name in the list of groups on the **My Groups** screen. While the selected group's information is in context, the browser's location bar shows the current URL which should of in the following format:

*https://www.globus.org/app/groups/<group UUID>/about*

The UUID can be copied and pasted from the current URL between the last and second-to-last forward slashes.

Create Globus application service accounts

1. Login into developers.globus.org with NIH AD Service account (Production/DEV-UAT) credentials.
2. If necessary, create Project with Project name and description. Otherwise, use pre-existing Project.
3. Click on **Add new app** to create application service account.
4. Enter **App Name** and other details as shown below in screen image.
   1. The **App Name** should follow the format of

*HPCDME-<which environment>-<classifier>-APP-ACCOUNT-<index>*

where

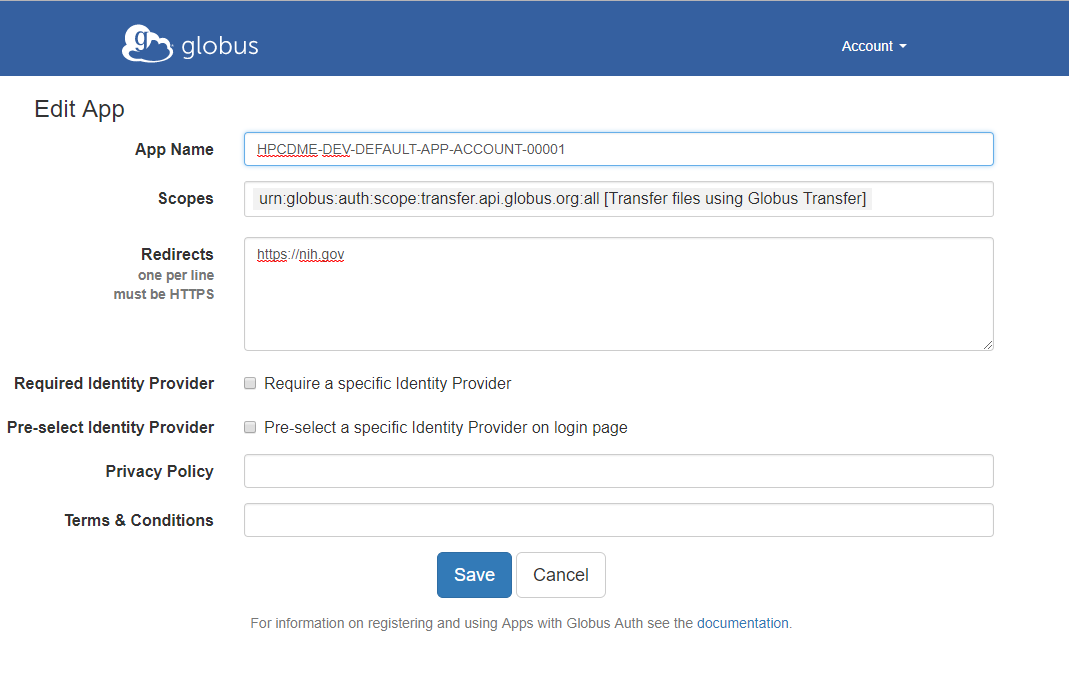
*<which environment>* is a placeholder for either "DEV" (non-production) or "PROD"

and

*<classifier>* is a placeholder for the pool's classifier (human-friendly name to refer to the pool)

and

*<index>* is a placeholder for a 5-digit serial number starting with 00001 and having leading zeros as needed



1. Click on **Generate New Client Secret** to generate secret key for the account.
2. Keep **Client Id** and **Secret Key** recorded in safe place.
3. Repeat Steps 3-6 for each Globus application service account that shall be utilized. Recommendation is minimum of 3 accounts per pool.
4. Use generated **Client Id** and **Secret Key** to setup Globus service account detailed in section 3.6.2.

Add Globus application service accounts to Globus pools

Prerequisites

1. Python installed, preferably 2.7 or newer
2. Text editor or other software tool for editing JSON
3. (Optional) Some comfort with scripting and running Python is desirable

Steps

1. Download the Python script at the following link to HPC DME's GitHub repository, <https://raw.githubusercontent.com/CBIIT/HPC_DME_APIs/master/doc/development/globus-add-app-acct-to-group/process_join_group.py>

Also download the associated JSON input file for the Python script at

<https://raw.githubusercontent.com/CBIIT/HPC_DME_APIs/master/doc/development/globus-add-app-acct-to-group/inputGroupAdds.json>

1. Open the *process\_join\_group.py* script and read the descriptive block comment toward the top. The text indicates how to use the script and the expected JSON input file that the script requires.
2. With a text editor or other software tool of your choice, edit the *inputGroupAdds.json* to suit your needs. In order to do this, you must know:
   1. UUID of any Globus group to which you wish to add application service accounts
   2. client ID and client secret of any Globus application service account that you wish to add to a Globus group
   3. which application service accounts to add to which groups
3. Execute the Python script. If it is successful, you shall see output conveying that every specified application account has been added to the proper group.

Creating a service Endpoint

1. Login into Globus.org with NIH AD Service account (Production/DEV-UAT) credentials.
2. Browse nihfnlcr#gridftp1 and select a folder to be used a stage area for 2 hop transfer and click on **Share**.
3. Enter **Share** display name and click on **Create** button.
4. Update **Share** permissions on this endpoint to grant **Write** permission to every Globus group which is intended to serve as a pool of Globus application service accounts.
   * You may refer to a group by its UUD if you have it handy.
   * You can also search/look up a group by its name.
5. Update **Roles** to grant the **Activity Monitor** role to every applicable Globus group as in previous step.
   * Again, you may refer to a group by its UUID or search/look up a group by its name.
6. Go to **Transfer Files** and browse the endpoint you just created. Create a folder for upload and folder for download
7. Give file system level write permission to NIH AD service account to the upload and download paths created on the endpoint
8. Go to endpoint details to get UUID

Setting up Application Configuration

* Go to HPC\_DME\_APIs\src\hpc-server\hpc-features\src\main\resources and open DEV/UAT/Production properties file based on the environment you are working with.
* Update following properties

hpc.integration.globus.archive.endpoint=< Endpoint UUID >

hpc.integration.globus.archive.path=< Upload folder name >

hpc.integration.globus.archive.directory=< Physical path on the mounted Globus file system >

#hpc.integration.globus.archive.type=TEMPORARY\_ARCHIVE

hpc.integration.globus.download.endpoint=< Endpoint UUID >

hpc.integration.globus.download.path=< Download folder name >

hpc.integration.globus.download.directory=< Physical path on the mounted Globus file system >

## Access Globus Tasks

HPC DME Globus application account submits asynchronous data registration requests to Globus on behalf of users. Users need to share the endpoints they intend to use with the Globus app account. A system admin can use the following method to check what Globus tasks are pending for the app account, and a (Globus) status of a particular task:

**Get Access token:**

Use app-id credentials (you will need the Globus app-id and secret key) to invoke the following service.

POST <https://auth.globus.org/v2/oauth2/token?grant_type=client_credentials&scope=urn:globus:auth:scope:transfer.api.globus.org:all>

Response:

{

   "access\_token": "AQBZrHddAAAAAAAFlJZYSDp4Jq6owlfXPbKdZBp-u70fHsmn8ZbopvABpKVnq4ROqy-ORzBsL\_28jOHSKsg0",

   "expires\_in": 172800,

   "resource\_server": "transfer.api.globus.org",

   "token\_type": "Bearer",

   "other\_tokens": [],

   "scope": "urn:globus:auth:scope:transfer.api.globus.org:all"

}

Use the token from the response to invoke the services listed below

**Get list of Globus tasks:**

GET [https://transfer.api.globusonline.org//v0.10/task\_list?filter=status:ACTIVE,INACTIVE](https://transfer.api.globusonline.org/v0.10/task_list?filter=status:ACTIVE,INACTIVE)

**Get a Globus task status:**

GET [https://transfer.api.globusonline.org/v0.10/task/<globus-task-id](https://transfer.api.globusonline.org/v0.10/task/%3cglobus-task-id)*>*

## Accessing Application Log files

HPC DME API runs on Apache ServiceMix. To access application logs:

1. SSH to API server
   1. In order to establish SSH connection to production API server, one must first SSH to either the development (DEV) virtual machine or the user acceptance testing (UAT) virtual machine. Then, from the connected virtual machine, regardless of whether DEV or UAT, one may SSH a second time to the production API server. This is due to how the production API server is configured for SSH connectivity.
2. Go to /opt/ apache-servicemix-<7.0.0.M3> folder
3. Go to /data/log folder to see application log files

At every restart of the application server, the log files are backed up to /backuplogs folder under /opt/ apache-servicemix-<7.0.0.M3>.

## Clean up and reset initial test data load for Development, UAT Tiers

## General Configuration

VM/Hardware:

* 1. DEV: fr-s-hpcdm-gp-d.ncifcrf.gov
  2. UAT: fr-s-hpcdm-uat-p.ncifcrf.gov
  3. Production:
     1. API: fr-s-dmeapi-t-p.ncifcrf.gov
     2. Database: fr-s-dmedb-t-p.ncifcrf.gov
     3. iRODS: fr-s-hpcdm-irods-p.ncifcrf.gov
     4. Web UI: fr-s-hpcdm-web-p.ncifcrf.gov
  4. Production mirrored VM
     1. API: fr-s-hpcdm-api-p.ncifcrf.gov
     2. Database: fr-s-hpcdm-db-p.ncifcrf.gov
  5. Database Daily Backup files storage: fr-s-hpcdm-gp-p.ncifcrf.gov

Service User Account:

DEV: ncif-hpcdm-svc

UAT: ncif-hpcdm-svc

Production: ncifhpcdmsvcp

NIH AD Account: ncihpcdmsvcad@nih.gov

Globus Endpoint for 2 hop data transfer: nihfnlcr#gridftp1

Mounted drive to access Globus Isilon storage: /mnt/IRODsTest

## Technology Stack

|  |  |
| --- | --- |
| Software/Package | Version |
| Java | 1.8 |
| Apache ServiceMix | 7.0.0 |
| Apace Tomcat | 8.0.x |
| PostgreSQL | 9.5.x |
| iRODS | 4.2 |
| Jargon API | 4.1.10.0.RELEASE |
| S3 API | 1.11.125 |
| Spring | 3.2.17.RELEASE |
| Cleversafe | 3.8.3.117 |

# APPENDIX A

Following instructions were tried on CentOS 7.4 version:

yum -y update

yum -y install yum-plugin-fastestmirror

sudo yum install wget

If there are any issues with yum, contact Frederick support. Occasionally there are unsatisfied dependencies that have to be manually resolved.

**Install Java**

wget --no-cookies --no-check-certificate --header "Cookie: gpw\_e24=http%3A%2F%2Fwww.oracle.com%2F; oraclelicense=accept-securebackup-cookie" <http://download.oracle.com/otn-pub/java/jdk/8u151-b12/e758a0de34e24606bca991d704f6dcbf/jdk-8u151-linux-x64.rpm>

sudo yum localinstall jdk-8u151-linux-x64.rpm

After installing java, open a shell and type:

java –version  
If this does not reflect the correct version, excecute the following command and choose your version.

(sudo) alternatives –config java

In addition, you can enter:

(sudo) alternatives –config javac

This will allow you to set the default version of the Java compiler in your environment, although most applications will look for the value of the JAVA\_HOME environment variable.

Set JAVA\_HOME to the directory where you just installed Java. This is usually accomplished by editing /etc/profile (setting it for all users) or individual .profile files for each account. Adding the following lines will ususally suffice:

export JAVA\_HOME=/path/to/java

export PATH=$JAVA\_HOME/bin:$PATH

**Install Maven**

(Execute the following command in the directory where you want to install maven, such as /usr/local).

sudo wget <http://www-eu.apache.org/dist/maven/maven-3/3.3.9/binaries/apache-maven-3.3.9-bin.tar.gz>

sudo tar xzf apache-maven-3.3.9-bin.tar.gz

(If you didn’t download the file in the directory where you want maven installed, you need to move it there with sudo mv apache-maven-3.3.9 /usr/local)

Execute the following command in the directory where you extracted or moved the maven folder.

sudo ln -s apache-maven-3.3.9 maven

sudo vi /etc/profile.d/maven.sh

Add following content:

export M2\_HOME=/usr/local/maven

export PATH=${M2\_HOME}/bin:${PATH}

source /etc/profile.d/maven.sh

Open another terminal and type ‘mvn -version’ to confirm that maven is configured correctly. If you get an error, or a different version than the one you installed, consult the generic instructions for installing maven here: <https://maven.apache.org/install.html>.

**Install Git client**

sudo yum install git

**Install ServiceMix**

sudo yum install zip unzip -y

sudo wget http://archive.apache.org/dist/servicemix/servicemix-7/7.0.0/apache-servicemix-7.0.0.zip

sudo unzip /home/centos/apache-servicemix-7.0.0.zip

Additional instructions for installing servicemix can be found here:

<https://servicemix.apache.org/docs/7.x/users-guide/installation.html>

Note that ServiceMix expects JAVA\_HOME to be set – you may see an error message if it’s not.

**Setup hostname:**

sudo hostname <hostname>

sudo vi /etc/hostname – Enter host name

sudo vi /etc/hosts – Enter host name

**Install PostgreSQL:**

sudo wget <https://download.postgresql.org/pub/repos/yum/9.5/redhat/rhel-7-x86_64/pgdg-centos95-9.5-3.noarch.rpm>

sudo yum install pgdg-centos95-9.5-3.noarch.rpm

sudo yum install postgresql95 postgresql95-server postgresql95-libs postgresql95-contrib postgresql95-devel

sudo vi /etc/sysconfig/pgsql/postgresql-9.5

PGPORT=5432

PGDATA=/pgdata95

sudo passwd postgres <hpcdme2017>

sudo /usr/pgsql-9.5/bin/postgresql95-setup initdb

sudo systemctl enable postgresql-9.5.service

sudo service postgresql-9.5 start

Detailed instructions for installing PostgreSQL on CentOS can be found here:

<https://linode.com/docs/databases/postgresql/how-to-install-postgresql-relational-databases-on-centos-7/>.

**Create iRODS root user:**

(sudo) su - postgres

postgres$ psql

psql> CREATE USER irods WITH PASSWORD 'testpassword123';

psql> CREATE DATABASE "ICAT";

psql> GRANT ALL PRIVILEGES ON DATABASE "ICAT" TO irods;

Test the connection:

psql -U irods -h 127.0.0.1 -d ICAT -W

If you get error like psql: FATAL: Ident authentication failed for user "irods"

Edit

sudo vi ./var/lib/pgsql/9.5/data/pg\_hba.conf

Set these values

# "local" is for Unix domain socket connections only

local all all md5

# IPv4 local connections:

host all all 127.0.0.1/32 md5

# IPv6 local connections:

host all all ::1/128 md5

sudo service postgresql-9.5 restart

Try login again

**iRODS installation:**

sudo rpm --import https://packages.irods.org/irods-signing-key.asc

wget -qO - https://packages.irods.org/renci-irods.yum.repo | sudo tee /etc/yum.repos.d/renci-irods.yum.repo

sudo yum install epel-release

sudo yum install irods-server irods-database-plugin-postgres

<https://docs.irods.org/4.2.2/getting_started/installation/>

# APPENDIX B

|  |  |  |
| --- | --- | --- |
| Property Name | Description | Default Value |
| **The following system properties are related to REST services** | | |
| hpc.ws.rs.host | This is host name of the machine where HPC DME Rest services will be running | Localhost |
| hpc.ws.rs. port | This is the secured port of the machine where HPC DME Rest services will be running. This port should be secured with CA cert as detailed below in section 3.2.2 | 7738 |
| hpc.ws.rs.path | HPC DME Server URL context name | hpc-server |
| hpc.ws.rs.stackTraceEnabled | Enable stacktrace as part of the error log | True |
| hpc.ws.rs.api-version | Do not change this value. This is updated by developers for any new release | 1.5.0 |
| hpc.ws.rs.ldapAuthentication | Enable NCI LDAP Authentication. If LDAP authentication is set to false, HPC DME uses authentication based on iRODS accounts | True |
| hpc.ws.rs.attachment-tmp-directory | Temporary staging directory where servicemix writes files while receiving from a client request |  |
| hpc.ws.rs.attachment-memory-threshold | Memory threshold before servicemix starts writing into “hpc.ws.rs.attachment-tmp-directory” folder | 104857600 |
| hpc.ws.rs.ssl.keystore.path | Name of the keystore where HPC DME certificate is stored | ${karaf.base}/etc/hpc-server/keystore.jks |
| hpc.ws.rs.ssl.keystore.type | Type of the keystore where HPC DME certificate is stored | JKS |
| hpc.ws.rs.ssl.keystore.password | Password of the keystore where HPC DME certificate is stored | hpc-server-store-pwd |
| hpc.ws.rs.ssl.truststore.path | Name of the trust store where HPC DME certificate is stored | ${karaf.base}/etc/hpc-server/keystore.jks |
| hpc.ws.rs.ssl.truststore.type | Type of the trust store where HPC DME certificate is stored | JKS |
| hpc.ws.rs.ssl.truststore.password | Password of the trust store where HPC DME certificate is stored | hpc-server-store-pwd |
| hpc.ws.rs.ssl.wantClientAuth | Set this property to the value true to configure the server to request a client certificate. This property is ignored when configured for a client component such as SSLSocketAppender. | False |
| hpc.ws.rs.ssl.needClientAuth | Set this property to the value true to configure a server to require a valid client certificate. This property is ignored when configured for a client component such as SSLSocketAppender. | False |
| hpc.ws.rs.jetty-engine.minThreads | Specifies the min size of the thread pool used by the specific Jetty instance | 5 |
| hpc.ws.rs.jetty-engine.maxThreads | Specifies the max size of the thread pool used by the specific Jetty instance | 15 |
| **The following system properties are related to REST services access permissions** | | |
| hpc.ws.rs.auth.register-user | Mapping user registration function to Allowed roles to register user | SYSTEM\_ADMIN GROUP\_ADMIN |
| hpc.ws.rs.auth.security.update-user | Mapping user registration function to Allowed roles to update user | SYSTEM\_ADMIN |
| hpc.ws.rs.auth.get-user | Allowed roles to query user | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.security.get-invoker | Allowed roles to query invoker | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.security.get-active-users | Allowed roles to query active users | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.security.get-all-users | Allowed roles to query all users | SYSTEM\_ADMIN |
| hpc.ws.rs.auth.authenticate | Allowed roles to authenticate user | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.security.register-group | Allowed roles to register a group | SYSTEM\_ADMIN GROUP\_ADMIN |
| hpc.ws.rs.auth.security.update-group | Allowed roles to update group | SYSTEM\_ADMIN GROUP\_ADMIN |
| hpc.ws.rs.auth.security.get-group | Allowed roles to get group | SYSTEM\_ADMIN GROUP\_ADMIN |
| hpc.ws.rs.auth.security.get-groups | Allowed roles to get groups | SYSTEM\_ADMIN GROUP\_ADMIN |
| hpc.ws.rs.auth.security.delete-group | Allowed roles to delete group | SYSTEM\_ADMIN GROUP\_ADMIN |
| hpc.ws.rs.auth.security.register-system-account | Allowed roles to register system account | SYSTEM\_ADMIN |
| hpc.ws.rs.auth.register-collection | Allowed roles to register collections | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.get-collection | Allowed roles to query a collection | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-management.download-collection | Allowed roles to download collection | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-management.set-collection-permissions | Allowed roles to set collection permissions | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-management.get-collection-permissions | Allowed roles to get collection permissions | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.register-data-object | Allowed roles to register a data object | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.get-data-object | Allowed roles to query data object | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-management.download-data-object | Allowed roles to download data object | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-management.set-data-object-permissions | Allowed roles to set data object permissions | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-management.get-data-object-permissions | Allowed roles to get data object permissions | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-management.get-data-management-model | Allowed roles to get data management model | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-management.get-data-management-tree | Allowed roles to get data management tree | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-search.query-collections | Allowed roles to query collections | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-search.query-data-objects | Allowed roles to query data objects | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-search.add-query | Allowed roles to save a named query | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-search.update-query | Allowed roles to update a named query | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-search.delete-query | Allowed roles to delete a named query | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-search.get-query | Allowed roles to get a named query | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-search.get-queries | Allowed roles to get a named queries | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-search.get-metadata-attributes | Allowed roles to get all metadata attributes with levels | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.data-search.refresh-metadata-views | Allowed roles to refresh search views in the database | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.notification.subscribe-notifications | Allowed roles to subscribe to system generated notifications | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.notification.get-notification-subscriptions | Allowed roles to get system generated notifications | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| hpc.ws.rs.auth.report.generate-report | Allowed roles to get reports | SYSTEM\_ADMIN GROUP\_ADMIN USER |
| **The following system properties are related to the application** | | |
| hpc.service.security.authenticationTokenSignatureKey | Signature key for the authentication token | User defined |
| hpc.service.security.authenticationTokenExpirationPeriod | Authentication token expiration in mins | 120 |
| hpc.service.dataTransfer.downloadDirectory | Staging directory for 2 hop data transfer. 1st hop is from client to API (stage). And the 2nd hop is from API (stage) to data archive | ${karaf.base}/data/tmp/S3 |
| hpc.service.dataSearch.searchResultsPageSize | Search page results size | 100 |
| hpc.service.dataSearch.defaultCollectionLevelFilter.level | Default search filter for ANY collection level | 1 |
| hpc.service.dataSearch.defaultCollectionLevelFilter.operator | Default search filter operator for ANY collection level | NUM\_GREATER\_OR\_EQUAL |
| hpc.service.dataSearch.defaultDataObjectLevelFilter.level | Default search filter for ANY data object level | 1 |
| hpc.service.dataSearch.defaultDataObjectLevelFilter.operator | Default search filter operator for ANY data object level | EQUAL |
| hpc.service.notification.formatsPath | Notification formats file path | ${karaf.base}/etc/hpc-server/notificationFormats.json |
| hpc.service.notification.mail.host | Email host name to send out email notifications | mailfwd.nih.gov |
| hpc.service.notification.mail.port | Email host port to send out email notifications | 25 |
| hpc.service.notification.notificationDeliveryReceiptsPageSize | Page size to display notifications on UI Dashboard | 100 |
| hpc.service.dataManagement.systemAdminSubjects | System accounts not allowed to set/update permissions by users | rods rodsadmin |
| hpc.service.event.invokerCollectionUpdateNotification | Flag to generate notification to the invoker of collection update | false |
| **The following system properties are related to business services called by REST API for Asynchronous data transfer.** | | |
| hpc.integration.globus.download.directory | Archive directory on the mounted drive of the globus endpoint. | /mnt/IRODsTest/FNL\_SF\_Archive/HPCDME\_PROD/Download |
| **The following system properties are related to application services called by integration services.** | | |
| hpc.integration.irods.host | iRODS host name | DEV - fr-s-hpcdm-gp-d.ncifcrf.gov  UAT - fr-s-hpcdm-uat-p.ncifcrf.gov  Production - fr-s-hpcdm-irods-p.ncifcrf.gov |
| hpc.integration.irods.port | iRODS host port number | 1247 |
| hpc.integration.irods.zone | iRODS zone name | tempZone |
| hpc.integration.irods.resource | iRODS resource name | demoResc |
| hpc.integration.irods.basePath | iRODS zone base path | / tempZone/home |
| hpc.dao.postgresql.dbName | iRODS iCAT database name | Please refer to prod/preprod/dev properties file |
| hpc.dao.postgresql.host | iRODS iCAT database host name | Please refer to prod/preprod/dev properties file |
| hpc.dao.postgresql.port | iRODS iCAT database port number | Please refer to prod/preprod/dev properties file |
| hpc.dao.postgresql.username | iRODS iCAT database user name | Please refer to prod/preprod/dev properties file |
| hpc.dao.postgresql.password | iRODS iCAT database user password | Please refer to prod/preprod/dev properties file |
| hpc.dao.postgresql.encryptor.key | Encryptor key used to encrypt integrating system credentials | \*\*\*128-bit-key\*\* |
| **The following system properties are related to application scheduler that runs background tasks.** | | |
| hpc.scheduler.cron.summaryreport.delay | Background task to generate daily summary report notification. This runs every day at 11PM. | 0 0 23 1/1 \* ? |
| hpc.scheduler.cron.weeklysummaryreport.delay | Background task to generate weekly summary report notification. This runs on every Sunday at 11PM. | 0 23 \* \* 7 ? |
| hpc.scheduler.cron.processevents.delay | Background task to process application events. Runs at every 10th second of a minute | 10 0/1 \* \* \* ? |
| hpc.scheduler.cron.processDataTranferUploadReceived.delay | Background task to process data transfer upload requests. Runs at every 5th second of a minute | 5 0/1 \* \* \* ? |
| hpc.scheduler.cron.processDataTranferUploadInProgress.delay | Background task to process “In progress” state upload requests . Runs at every 15th second of a minute | 15 0/1 \* \* \* ? |
| hpc.scheduler.cron.processTemporaryArchive.delay | Background task to process files from temporary archive to permanent archive. Runs at every 25th second of a minute | 25 0/1 \* \* \* ? |
| hpc.scheduler.cron.completeDataObjectDownloadTasks.delay | Background task to process download requests. Runs at every 35th second of a minute | 35 0/1 \* \* \* ? |
| hpc.scheduler.cron.processCollectionDownloadTasks.delay | Background task to process download collection requests. Runs at every 50th second of a minute | 50 0/1 \* \* \* ? |
| hpc.scheduler.cron.completeCollectionDownloadTasks.delay | Background task to complete download collection requests. Runs at every 55th second of a minute | 55 0/1 \* \* \* ? |
| hpc.scheduler.cron.refreshMaterializedViews.delay | Background task to refresh search materialized views. Runs at every 15 minutes | 0 0/15 \* \* \* ? |
| hpc.scheduler.cron.refreshReportViews.delay | Background task to refresh reports materialized views. Runs at every 60 minutes | 0 0/60 \* \* \* ? |

# Appendix C: uat sYSTEM configuration

This section describes the system configuration on the UAT server and command for starting and stopping the Tomcat and ServiceMix instances.

There is only one physical machine for UAT and it runs both ServiceMix (to expose the API Services), and Tomcat (for hosting the web client). Tomcat runs standalone, listening on port 443. There is no apache web server setup currently to proxy to it.

The UAT server needs to be accessed from the dev machine fr-s-hpcdm-gp-d.ncifcrf.gov

From the dev machine run the following command:

ssh fr-s-hpcdm-uat-p

****ServiceMix****

Servicemix in installed in the /opt/ apache-servicemix-7.0.0 directory. It is started and stopped with the service account privileges.

Sudo to the service account with the command:

sudo su

su ncif-hpcdm-svc

Goto the servicemix installation directory

cd /opt/apache-servicemix-7.0.0

To stop servicemix run

./bin/stop

To start servicemix, run

./bin/start

**Tomcat:**

Tomcat is installed in the /opt/apache-tomcat-8.0.42 directory. It is started and stopped with the root user privileges as indicated below:.

Sudo to root with the command

sudo su

Goto the tomcat installation directory

cd /opt/apache-tomcat-8.0.42

To stop tomcat, run

./bin/catalina.sh stop

To start tomcat, run

./bin/catalina,sh start

# APPENDIX D: Setting up HPCDME as a Service

HPCDME can to setup with ServiceMix running as a service. The steps are as follows:

* 1. Cd to the ServiceMix installation directory. Start ServiceMix if it is not running.

./bin/start

.

* 1. Run the following commands to set it up as a service

./bin/client -h 127.0.0.1 “feature:install wrapper”

./bin/client -h 127.0.0.1 “wrapper:install -s AUTO\_START -n KARAF -d Karaf -D ‘Karaf Service’"

* 1. Stop ServiceMix

./bin/stop

* 1. Run the following commands

ln -s /opt/apache-servicemix-7.0.0.M3/bin/KARAF-service /etc/init.d/

update-rc.d KARAF-service defaults

The above will install the service and set it up to start whenever the server reboots.

* 1. Start ServiceMix as a service.

/etc/init.d/KARAF-service start

To stop the service if required, run

/etc/init.d/KARAF-service stop