**DME Prod Deployment Instructions**

# [Optional] Pre-requisites

# Log into the DME Prod API Server

e.g. ssh <hostname> (API Server)

# Switch user to service account

sudo su - ncifhpcdmsvcp

# Tomcat installation

Check if Tomcat is installed and which version. Tomcat 9 should be installed.

/usr/share/tomcat/bin/version.sh

# Java installation

Check Java installation and default version.

alternatives --list

java -version

Switch to Java 11 if not default.

alternatives --config java ; alternatives --config javac; alternatives --config java\_sdk\_openjdk:

Also set the JAVA\_HOME to jdk11 in ~/**setenv.sh**

export JAVA\_HOME=/usr/lib/jvm/java-11-openjdk

# Prepare the build on the Prod Server

# Change directory to Git local directory and get the latest code

cd /opt/HPC\_DME\_APIs/src

* Get the latest code:

git pull

* If this is first time checkout for a new branch, then switch to the new branch:

git pull

git reset --hard origin/releases/<new version>

Example: git reset --hard origin/releases/3.0.0

git checkout releases/<new version>

Example: git checkout releases/3.0.0

# Build the API Server

mvn clean install -Pprod

# [Optional] Build the DME Web Application

cd /opt/HPC\_DME\_APIs/src/hpc-web

cp ~/prod-env.conf src/main/resources/appconfigs/prod-env.conf

mvn clean install -Pprod -DskipTests

scp target/hpc-web-<version>.war <your username>@<DME Prod Web Server>:~/

*(****Note****: If any new env. variable has been added in this release to prod-env.conf, then instead of a blind copy, a merge should be performed)*

# Deployment Steps

***Note****: If this is the first time deploying to the server and the tomcat has not yet been configured, complete* **Step 5** *before proceeding with* **Step 3***.*

# Switch user to root and shutdown the Tomcat server and change directory to the tomcat webapps directory

sudo su  
systemctl stop tomcat  
cd /usr/share/tomcat/webapps

# Update conf/server.xml

If deploying DME Web, in the line

<Context path="" docBase="hpc-web-<prev\_version>" privileged="true"/>

replace <prev\_version> with the value of <new\_version>

e.g.

<Context path="" docBase="hpc-web-3.0.0" privileged="true"/>

If deploying DME API server, in the lines

<Context path="/hpc-server" docBase="hpc-server-<prev\_version>" privileged="true"/>

<Context path="/hpc-scheduler" docBase="hpc-scheduler-<prev\_version>" privileged="true"/>

<Context path="/hpc-scheduler-migration" docBase="hpc-scheduler-migration-<prev\_version>" privileged="true"/>

replace <prev\_version> with the value of <new\_version>

e.g.

<Context path="" docBase="hpc-server-3.0.0" privileged="true"/>

<Context path="/hpc-scheduler" docBase="hpc-scheduler-3.0.0" privileged="true"/>

<Context path="/hpc-scheduler-migration" docBase="hpc-scheduler-migration-3.0.0" privileged="true"/>

Execute any database scripts (if applicable)

# Stage the war files

* Back up the existing war files (if applicable) e.g.

export HPC\_HOME=/opt/HPC\_DME\_APIs

cd /usr/share/tomcat/webapps

mv hpc-web-3.0.0.war hpc-web-3.0.0.war.backup.<date>  
mv hpc-server-3.0.0.war hpc-server-3.0.0.war.backup.<date>

mv hpc-scheduler-3.0.0.war hpc-scheduler-3.0.0.war.backup.<date>

mv hpc-scheduler-migration-3.0.0.war hpc-scheduler-migration-3.0.0.war.backup.<date>

* Delete existing war directory (if applicable) e.g.

rm -rf hpc-scheduler

rm -rf hpc-scheduler-migration

* [API only] Copy the new DME API Server war file

cp $HPC\_HOME/src/hpc-server/hpc-ws-rs-impl/target/hpc-server-<version>.war .

* [API only] Copy the new scheduler war files to Tomcat

cp $HPC\_HOME/src/hpc-server/hpc-scheduler/target/hpc-scheduler-<version>.war .

cp $HPC\_HOME/src/hpc-server/hpc-scheduler-migration/target/hpc-scheduler-migration-<version>.war .

* [Web only] Copy the new DME Web Application war file

cp <Staged directory where you scp to>hpc-web-<version>.war .

# Start tomcat and verify

systemctl start tomcat

*If the web application URL displays Tomcat home page, stop and start Apache Tomcat*

systemctl stop tomcat

systemctl start tomcat

# [Optional] Accessing Tomcat Logs

* Login into the DME Prod server

e.g. ssh <host name>

* Switch user to the DME service account  
  sudo su - ncifhpcdmsvcp
* Change directory to the tomcat logs directory  
  cd /var/log/tomcat
* Access the tomcat logs in the file catalina.out
* In addition, the DME Web application produces the log file:

/var/log/tomcat/hpcweb.log

* In addition, API Server produces the following log file:  
  /var/log/tomcat/hpc-server/hpc-server.log
* To dynamically change the logging level, modify:  
  /usr/share/tomcat/conf/hpc-server/logback.xml

# [First time only] Setup Configuration files on Tomcat Server

# Configure Tomcat

To allow the server to communicate with iRODS and LDAP, we need to deploy/configure a keystore:

* Copy keystore and cacerts into /usr/share/tomcat/conf/hpc-server

mkdir /usr/share/tomcat/conf/hpc-server

cp $HPC\_HOME/src/hpc-server/hpc-ws-rs-impl/src/main/resources/cacerts.jks /usr/share/tomcat/conf/hpc-server

cp $HPC\_HOME/src/hpc-server/hpc-ws-rs-impl/src/main/resources/keystore.jks /usr/share/tomcat/conf/hpc-server

* Create file /usr/share/tomcat/bin/setenv.sh and add the following:

export UMASK=0022   
JAVA\_OPTS="$JAVA\_OPTS -Djavax.net.ssl.trustStore=$CATALINA\_HOME/conf/hpc-server/cacerts.jks -Djavax.net.ssl.trustStorePassword=changeit -Dlogback.configurationFile=$CATALINA\_HOME/conf/hpc-server/logback.xml"

* Copy properties file into /usr/share/tomcat/conf/hpc-server

cp $HPC\_HOME/src/hpc-server/hpc-ws-rs-impl/src/main/resources/WEB-INF/spring/hpc-server.properties /usr/share/tomcat/conf/hpc-server

* Create logback.xml file in /usr/share/tomcat/conf/hpc-server

|  |
| --- |
| <configuration scan="true" scanPeriod="15 seconds">    <include resource="org/springframework/boot/logging/logback/base.xml"/>    <property name="API\_SERVER\_LOG" value="${catalina.base}/logs/hpc-server/hpc-server.log"/>    <appender name="API\_SERVER\_FILE\_ROLLING" class="ch.qos.logback.core.rolling.RollingFileAppender">  <file>${API\_SERVER\_LOG}</file>      <rollingPolicy class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy">  <fileNamePattern>${catalina.base}/logs/hpc-server/hpc-server.log.%i</fileNamePattern>  <minIndex>1</minIndex>  <maxIndex>20</maxIndex>  </rollingPolicy>  <triggeringPolicy class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">  <maxFileSize>500MB</maxFileSize>  </triggeringPolicy>    <encoder>  <pattern>%d %p [%t] %m%n</pattern>  </encoder>  </appender>    <logger name="gov.nih.nci.hpc" level="debug" additivity="false">  <appender-ref ref="API\_SERVER\_FILE\_ROLLING"/>  </logger>  <root level="error">  <appender-ref ref="API\_SERVER\_FILE\_ROLLING"/>  </root>      </configuration> |

* Update /usr/share/tomcat/conf/hpc-server/hpc-server.properties as follows:

1. Set the hpc.integration.ldap.password.
2. Set hpc.dao.oracle.password.
3. Check to see if it is pointed to the Dev Oracle instance
4. Check to see if it is pointed to the Dev iRODS instance

* Configure the SSL port 8080 for DME API Server

To enable SSL on port 8080, add the following connector to /usr/share/tomcat/conf/server.xml:

Comment out the port 8080 http connector and add:

<Connector protocol="org.apache.coyote.http11.Http11NioProtocol" port="8080" maxThreads="200" maxParameterCount="1000" sslEnabledProtocols="TLSv1.2" scheme="https" secure="true" SSLEnabled="true" keystoreFile="${catalina.base}/conf/hpc-server/keystore.jks" keystorePass="changeit" clientAuth="false" sslProtocol="TLS"/>

* To access the Web application directly at port 8080, add the following context in /usr/share/tomcat/conf/server.xml.

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

* To access the hpc-server directly at port 8080, add the following context in $CATALINA\_HOME/conf/server.xml.

<Context path="/ " docBase="hpc-server-<version>" privileged="true"/>

* To avoid war expansion into two folders, update the following in $CATALINA\_HOME/conf/server.xml.

<Host name="localhost" appBase="webapps"  
 **unpackWARs="true"** **autoDeploy="false"** **deployOnStartup="false"**>

* To deploy the scheduler and migration war files, add the following context in $CATALINA\_HOME/conf/server.xml.

<Context path="/hpc-scheduler" docBase="hpc-scheduler-<version>"

<Context path="/hpc-scheduler-migration" docBase="hpc-scheduler-migration-<version>"

# Other Checklists and Troubleshooting

* Check for **~/.aws** folder in service account home for **AWS\_DEFAULT\_REGION**
* Check installation of **Aspera** in service account home for dbGaP transfer
* Make sure the /usr/share/tomcat/conf/hpc-server/**hpc-server.properties** is configured for the proper tier and schedulers are enabled/disabled for that server
* Use the latest Jargon **4.3.3.0-RELEASE** from Eran's branch.
* To resolve "Unable to add resource .... insufficient free space available after evicting expired cache entries" error, added the following in conf/context.xml.

<Resources cachingAllowed="true" cacheMaxSize="100000" />

* To resolve "The scratchDir you specified: [/usr/share/tomcat/work/Catalina/localhost/xxx] is unusable.", changed owner of work/Catalina directory to the service account.
* To address issues with new war file not taking effect right away, disabled caching in conf/server.xml.

<Context path="/ " docBase="hpc-server-3.0.0" **cachingAllowed="false"** privileged="true"/>

<Context path="" docBase="hpc-web-3.0.0" **cachingAllowed="false"** privileged="true"/>

* Ensure there is enough space in /var/log (At least 15GB)