



# GEMMLCA GT4 installation & configuration handson

Gabor Kecskemeti

- Univ. Westminster, UK
- MTA SZTAKI, Hungary  
kecskemeti@sztaki.hu



# Prerequisites

- Software dependencies:
  - Globus Toolkit 4, Ant 1.6, Java 1.5
  - Bash 2
  - Service certificate for the GEMMLCA container
  - Optional packages:
    - Apache Tomcat 5.5 - as a flexible container
    - Condor - G with GT2 access - for GT2 backend
    - LCG2 UI - for EGEE backend
    - P-Grade Portal 2.4.1 - both for GT2 and EGEE backends
- Hardware requirements:
  - 256 MB memory / running GEMMLCA service.
  - 20MB disk + LC configuration store.

# Installation



- Unpack the GEMMLCA tarball
- Internet connection is required to finish the installation!
- If you want to deploy the GEMMLCA in a tomcat container please setup the CATALINA\_HOME environment variable before installation.
- Start the setup process by calling:
  - ant install

# Installer Questionnaire #1



- Managed Job Factory Service's URL?
  - URL of the WS-GRAM where the GEMMLCA will submit the LCs. E.g.: <https://grid-compute-ws.cpc.wmin.ac.uk:8443/wsrf/services/ManagedJobFactoryService>
- Comma separated list of supported jobmanagers?
  - The list of jobmanagers the WS-GRAM service can handle. (GEMMLCA supports fork, condor, pbs, lsf - under development)

# Installer Questionnaire #2



- GEMMLCA root folder (Repository and configuration file storage)?
  - The GEMMLCA is going to store its basic legacy codes and LCID descriptors: e.g.  
`/usr/local/gemmlca/repository`
- “Keyfile?” then “Certificate file?”
  - The GEMMLCA service certificates, e.g.:
    - `/usr/local/gemmlca/servicekey.pem`,  
`/usr/local/gemmlca/servicecert.pem`

# Installer Questionnaire #3



- GT4 Libraries?
  - The location of the CoG, WSRF Core and other libs from the Globus installation: e.g. /usr/local/globus/lib
- Where to install the GEMMLCA services?
  - The location where the installer will create a base Globus WSRF Core container and deploy the GEMMLCA services. E.g.: /usr/local/gemmlca/service
  - The container gets downloaded from the Globus website, then it is compiled and prepared to receive the GEMMLCA service pack.

# Installer Questionnaire #4



- Would you like the GEMMLCA to be deployed in your tomcat installation (/usr/local/tomcat)?
  - Optional question - depends on CATALINA\_HOME
  - Press “y” to deploy it, the installer creates a wsrf webapp.

# Configuration - fine tuning parameters



- Base configuration file is in  
`/usr/local/gemlca/service/gemlca-config.xml`
- It is generated by the install script
- Parameters:
  - `globus-grid-mapfile` - The gridmapfile used for authorization.
  - `gemlca-config-folder`, `gemlca-LC-programs-folder`,  
`gemlca-LC-environment-folder`



# Configuration - plugin setup



- Define plugin classes with the plugin element:
  - `<plugin name="GT4" className="uk.ac.wmin.cpc.gemlca.backend.GT4Job"/>`
- Define plugin instances with the backend element - configure the plugin behaviour site by site.

# Configuration - backend setup



```

<backend id="Westminster">
  <pluginName>GT4</pluginName>
  <contactURL>https://grid-compute-
    ws.cpc.wmin.ac.uk:8443/wsrp/services/ManagedJobFactoryService
  </contactURL>
  <parameter name="gemlca-job-managers-supported"
    value="FORK,CONDOR"/>
  <parameter name="notification-test-timeout-sec" value="20"/>
  <parameter name="notification-cache-max-age-sec" value="3600"/>
  <parameter name="polling-time-for-legacy-codes-sec" value="30"/>
  <parameter name="number-errors-catch-when-contacting-job-manager"
    value="30"/>
  <parameter name="safety-polling-time-for-legacy-codes-sec" value="300"/>
</backend>
  
```

Plugin reference

Generated

Long term experience



# Startup and logging

- Startup:
  - export GLOBUS\_LOCATION=/usr/local/gemlca/service
  - \$GLOBUS\_LOCATION/bin/globus-start-container -p 8443 > \$GLOBUS\_LOCATION/containerlog &
- GEMLCA specific logs can be found in /tmp/GEMLCA.log, contents:
  - Problems with legacy code deployments
  - Problems with legacy code executions
  - Basic GEMLCA accounting information
- GEMLCA log4j appender setup is located in \$GLOBUS\_LOCATION/gemlcalogger.properties.xml
  - Logger examples and formats can be found here

# Legacy code administrators



- GEMMLCA has a filesystem based legacy code store
  - The store can be managed by anyone who can write its files via the FS. FS rights has to be managed by the GEMMLCA administrator - manages the list of legacy code administrators.
  - The individual Legacy Code Descriptors are maintained by their owners. Owner info and authorization is done by the GEMMLCA.



# Recommended FS setup

- `chmod 2775`  
`/usr/local/gemlca/repository/legacycodes`
  - `drwxrwsr-x ...`
- GEMLCA admin user: `gemlca`
- GEMLCA users group: `gridusers`
- GEMLCA LC admins group: `lcadmins`, or `gridusers`
- Permissive setup: `chown -R gemlca:gridusers`  
`/usr/local/gemlca/repository/legacycodes`
- Defensive setup: `chown -R gemlca:lcadmins`  
`/usr/local/gemlca/repository/legacycodes`

# Legacy Code store layout



- The legacy codes are stored under `/usr/local/gemlca/repository/legacycodes`
- The folders inside are the ids of the legacycodes
- Each folder holds a descriptor (called `config.xml`), and the default input files if necessary.



# The Legacy Code descriptor

```

<?xml version="1.0"?>
<LCEnvironment xmlns='http://uk.ac.wmin.cpc.gemlca/schema/legacyCodeConfig'
  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
  xsi:schemaLocation='http://uk.ac.wmin.cpc.gemlca/schema/legacyCodeConfig ../../conf/legacyCodeConfig.xsd'
  id="gemlca_ce" status="system" maximumParallelism="1">
  <description>GEMLCA Helper: Legacy code environment preparation</description>
  <parameter order="0" fixed="false" mandatory="true" file="false" input="true" name="-p" commandline="true" >
    <value></value> <friendlyName>Folders To Be Created</friendlyName>
  </parameter>
  <parameter order="1" fixed="false" mandatory="false" file="false" input="true" name="-f" commandline="true">
    <value></value> <friendlyName>Files to be copied into the folder</friendlyName>
  </parameter>
  <parameter order="2" fixed="false" mandatory="false" file="false" input="true" name="-o" commandline="true">
    <value></value> <friendlyName>Authenticated username</friendlyName>
  </parameter>
  <parameter order="3" fixed="false" mandatory="true" file="false" input="true" name="-e" commandline="true">
    <value></value> <friendlyName>Error output</friendlyName>
  </parameter>
  <backendSpecificData backendId="GT4" executable="LINUX/gemlca_ce.sh" jobManager="FORK"
    jobType="single" output="stdout" error="error"/>
  <authorizationInfo><owner/></authorizationInfo>
</LCEnvironment>

```



# Basic Legacy code info

- id - same as the dirname
- status
  - System - parts of the GEMLCA, cannot be listed
  - Offline - descriptor for future use, not listed
  - Test - descriptor is under development after a successful submit it becomes online, not listed
  - Online - production legacy code, listed
- maximumParallelism - number of legacy code jobs allowed in parallel
- description - Simple description for the service requestor/user





# Parameter setup

- `lcexec $1 $2 -o "inputfilename" ...`
- Parameter can be changed from its initial value
- A new value has to be given before execution
- Parameter is present in the commandline
- Parameter refers to a file which should be transferred from or next to the LC execution

```
<parameter order="2" fixed="false" mandatory="true" file="true" input="true" name="-o" commandline="true">
  <value>inputfilename</value> <friendlyName>Authenticated username</friendlyName>
</parameter>
```



# Backend specific data

- `backendId` = Which plugininstance to use for legacy code submission
- `executable` = absolute or relative path of the legacy code on the local FS - if not present the GEMMLCA tries to execute the LC without transferring the executable
- `jobManager` = which jobmanager to use - has to be chosen from the list of the backend configuration.
- `jobType` = GT4 RSL equivalent jobtype
- `Output, error` = standard outputs



# Current non GT4 Plugins

- EGEE Plugin:
  - Requires the PGrade Portal installed on the same machine as the GEMMLCA.
  - New backend specific data called “ExecutorSites”. List of EGEE sites which can accept the same LC separated by “;”.
- Condor-G Plugin:
  - Requires the Pgrade Portal installed on the same machine as the GEMMLCA.
  - The plugin needs the `/usr/local/gemmlca/repository/conf/condorg.sh`'s GT4 location and `CONDOR_LOCATION` variables to be changed on each installation.