VOMS-aware Middleware Deployment, Work Plan, and Template *

Oliver Keeble, Ronald Starink, Jeff Templon

January 19, 2007

Contents

*$Id: Vdepl.tex, v1.42007/01/1913:31:42templonExp$

Contributors

The templates and strategies discussed here are heavily influenced by the work of the EGEE Job Priorities WG.

1 Introduction

Experiments and sites both want to roll out "VOMS". In practice, VOMS itself as a product already exists, but the rest is missing. Not all middleware is VOMS-aware, and there is no standard for "VOMSifying" a site's resources, which means that experiments and users cannot really exploit groups and roles, even if they have defined them.

This document attempts to provide the 'standard deployment strategy' for sites, and to identify a path to roll out this to the sites, and to make it possible for the middleware to make use of the system.

2 Basic Functionality Desired at Sites

The experiments have expressed a requirement for the following basic functionality:

- Users should be able to function with multiple roles, *e.g.,* it should be possible for a software manager to also do work as a normal user. Some experiments do this with *groups* instead of *roles*.
- Experiments should be able to define different levels of fair shares (or perhaps bare priority) for different classes of users.
- Experiments should be able to control access to various classes of resources, for example certain disk pools should be writeable only for members of the production team.

3 Basic Limitations at Sites

Sites have expressed the following requirements on the implementation:

- it should be as flexible as possible — sites must be allowed to achieve the desired goals as is appropriate for the site, rather than mandating a uniform setup at each site. Note that the consequence is that the information published about the site should be closely related to what the user wants, rather than to the actual implementation.
- along the same lines, as little information as possible about the internal organization should be published, otherwise it becomes very difficult for the site to make internal reorganizations.

A good example of this is the tendency of users and middleware people to "require" use of queues as an organizational tool. In general, contrary to the perception of many people, a queue is nothing but a placeholder, the real work in assiging priorities and shares is

unrelated to queues for many schedulers. Queues increase the amount of organization at a site (increasing the number of places to make an error) without any added value at the site.

4 Basic Template for computing resource VOMSification

The following system is compatible with the experiment requirements.

- A "basic user" class (equivalent to what we already have now in LCG)
- Zero or more "privileged" classes with defined shares. If there is more than zero of these, it means of course that the "basic user" class also gets a defined share.

  Defined share in this context means that a class gets X % of the total share allocated to the VO for the site.
- One or more "special function" classes for which a share is not relevant, but prompt scheduling is. The prime example of this is the SGM user class, SGM jobs should be scheduled before any others from the same VO.

A related issue is that of fast-turnaround jobs. This issue is not directly related to VOMS, please refer to the work of the EGEE Job Priorities Working Group.

5 Configuration for Basic Template

Here we display the configuration for ATLAS. The configuration for other experiments should be straightforward, only the names of the groups and roles, and their number, should change. The principle is exactly the same.

5.1 User accounts

Extra groups of pool accounts, belonging to group `atlb` (with account names `atlb001, atlb002,` *etc.* need to be created. They should have `atlas` as secondary group.

A second group of pool accounts belonging to group `atlsgm` (with account names `atlsgm01, atlsgm02,` *etc.* also needs to be created. This replaces the old single `atlsgm` account. This account should also have `atlas` as secondary group.

A consequence of this is that there should be a note in the release notes informing experiments of this change, they may be using the assumption of "single SGM user" in experiment-specific code.

5.2 LCAS / LCMAPS

Add

```
"/VO=atlas/GROUP=/atlas/ROLE=production"        atlb
"/VO=atlas/GROUP=/atlas/ROLE=software"          atlsgm
```

to the file `/opt/edg/etc/lcmaps/groupmapfile`, and

```
"/VO=atlas/GROUP=/atlas/ROLE=production"        .atlb
"/VO=atlas/GROUP=/atlas/ROLE=software"          .atlsgm
```

to the file `/opt/edg/etc/lcmaps/gridmapfile` . For the moment, the "normal" ATLAS population will be picked up by the usual grid-mapfile mechanism already installed at sites. A short-term action is to determine how to add the "normal" population to the above files, for example does the order matter?

5.3 Generic Information Providers

The relevant VOMS FQANs need to be published in the information system. The information providers will not pick them up unless they are in the static LDIF file for the CE. Here is an example block:

```
dn: GlueCEUniqueID=tbn20.nikhef.nl:2119/jobmanager-pbs-qlong,
  mds-vo-name=local,o=grid
objectClass: GlueCETop
objectClass: GlueCE
objectClass: GlueSchemaVersion
objectClass: GlueCEAccessControlBase
objectClass: GlueCEInfo
objectClass: GlueCEPolicy
objectClass: GlueCEState
objectClass: GlueInformationService
objectClass: GlueKey
GlueCEHostingCluster: tbn20.nikhef.nl
GlueCEName: qlong
GlueCEUniqueID: tbn20.nikhef.nl:2119/jobmanager-pbs-qlong
[ ... ]
GlueCEPolicyAssignedJobSlots: 0
GlueCEAccessControlBaseRule: VO:alice
GlueCEAccessControlBaseRule: VO:astrop
GlueCEAccessControlBaseRule: VO:atlas
GlueCEAccessControlBaseRule: VOMS:/atlas/Role=production
GlueCEAccessControlBaseRule: VOMS:/atlas/Role=software
GlueCEAccessControlBaseRule: VO:biomed
```

Note the `VOMS:` section in the next-to-last line. Further along in the file the `VOView` block must also be included:

```
dn: GlueVOViewLocalID=atlas_role_production,GlueCEUniqueID=
  tbn20.nikhef.nl:2119/jobmanager-pbs-qlong,mds-vo-name=local,o=grid
objectClass: GlueCETop
objectClass: GlueVOView
objectClass: GlueCEInfo
objectClass: GlueCEState
objectClass: GlueCEAccessControlBase
```

```
objectClass: GlueCEPolicy
objectClass: GlueKey
objectClass: GlueSchemaVersion
GlueVOViewLocalID: atlas_role_production
GlueCEAccessControlBaseRule: VOMS:/atlas/Role=production
GlueCEStateRunningJobs: 0
GlueCEStateWaitingJobs: 4444
GlueCEStateTotalJobs: 0
GlueCEStateFreeJobSlots: 0
GlueCEStateEstimatedResponseTime: 2146660842
GlueCEStateWorstResponseTime: 2146660842
GlueCEInfoDefaultSE: teras.sara.nl
GlueCEInfoApplicationDir: /data/esia/atlas/slc3
GlueCEInfoDataDir: unset
GlueChunkKey: GlueCEUniqueID=tbn20.nikhef.nl:2119/jobmanager-pbs-qlong
GlueSchemaVersionMajor: 1
GlueSchemaVersionMinor: 2
```

A similar block is needed for the `software` role.

Note: please do **not** change the values "4444" nor "2146660842" in these blocks. These values are what get printed if the information provider fails to run. If the information provider fails to run correctly, your site is broken from the user's point of view (even though it may be the software that is broken, your site is still broken because it is running broken software). These large values are intended to steer jobs **away** from your broken site. Changing them could create a black hole at your site.

The information provider `lcg-info-dynamic-scheduler` uses these blocks. The version capable of dealing with `VOMS` blocks exists but is not yet in the production release, see 6.1.

This information provider needs to be configured to understand the mapping between unix groups (visible in the batch-system queries) and the VOMS FQANs, this link is needed in order to be able to generate job counts and estimated response times for the various groups. To do so, lines like the following are needed in the configuration file for `lcg-info-dynamic-scheduler`.

```
[Main]
[ ... ]
vomap :
   atlas:atlas
   atlsgm:/atlas/Role=software
   atlb:/atlas/Role=production
   biome:biomed
[ ... ]
```

The syntax is

```
 <unix group name>:<VO name or VOMS FQAN>
```

5.4 Batch System

As of the current writing, ATLAS is assigning 80% of its total allocation to the production group. A first starting point for the scheduler configuration is the following:

- production group, 80% of VO's share
- normal users group, 20% of VO's share
- software administrators group, no share, high priority, limited number of concurrent processes.

5.4.1 Concept in Maui

First an example for "old Maui", *i.e.* the Maui that was distributed with glite 3.0 and earlier. This has fair shares based on absolute numbers.

```
USERCFG[DEFAULT]    FSTARGET=7
GROUPCFG[atlas]     FSTARGET=6    PRIORITY=100                  QDEF=lhcatlas
GROUPCFG[atlb]      FSTARGET=24   PRIORITY=100                  QDEF=lhcatlas
GROUPCFG[atlsgm]                  PRIORITY=200    MAXPROC=2     QDEF=lhcatlas
QOSCFG[lhcatlas]    FSTARGET=30                   MAXPROC=220
```

The last line represents the share of the VO as a whole: `atlas` here is allocated a 30% share of the site's resources. The three `GROUPCFG` lines first implement the two shares, the "normal user" share of 6% (`FSTARGET`, equaling 20% of the VO's 30% of the cluster) and the "production" share for `atlb` which eats the other 80% of the VO's share. These groups both have equal `PRIORITY`, their relative scheduling priority is driven solely by how closely they match their fair share usage.

The SGM group has a higher base priority; assuming that the `FSWEIGHT` and `CREDWEIGHT` are appropriately set, the SGM jobs will always have higher priority than other jobs for the VO, but the system will only run at most two SGM jobs at a time. An appendix will contain a complete copy of a Maui config file showing how these weight factors can be set to achieve the desired effect.

Note that in the older versions of Maui used in EGEE/LCG, the fair share priorities were computed as absolutes: the priority component due to fair share looked like

$$r_{f.s.} = \text{scale factor} \times (t - u)$$

where $r_{f.s.}$ is the fair-share component of the job scheduling rank, t is the target share (fraction of the entire farm) assigned, and u is the share that has been used by the group. This version unfairly biases the system towards VOs with larger fair share allocations; the latest Maui version in EGEE certification (and already being run at some sites) has share components to priority are computed via ratios

$$r_{f.s.} = \text{scale factor} \times \frac{t - u}{t}$$

For the ratio version, the above section could be exactly the same; however the actual values given to `FSTARGET` for `GROUP`s `atlas` and `atlb` could also look as follows:

```
GROUPCFG[atlas]     FSTARGET=20   PRIORITY=100                  QDEF=lhcatlas
GROUPCFG[atlb]      FSTARGET=80   PRIORITY=100                  QDEF=lhcatlas
```

The first version shown (6:24) is better for a number of reasons beyond the scope of this document, but the version above will work fine (especially in the beginning before tuning has taken place) for most sites.

Note as well that in the new Maui from EGEE, one needs the directive

```
FSPOLICY DEDICATEDPS%
```

in order to enable the ratio mode of priority computation; otherwise the old absolute mode will be used.

5.4.2 Concept in LSF

Placeholder for now.

6 Software Gaps

6.1 Information System

`lcg-info-dynamic-scheduler` is now in certification, the proper version to get is 2.0.1 or greater.

There are known problems that this provider is not configured via YAIM for LSF farms out-of-the-box. It is likely that the old `lcg-info-dynamic-lsf` will need to be modified so that it does not print any of the VOView fields, nor the dynamic information for the CE blocks.

6.2 Data Management

DPM supports VOMS ACLs. As of this writing, to the best of our knowledge neither dCache nor CASTOR support VOMS ACLs. We believe that the LFC supports ACLs but it is not clear how well this is integrated with the other data management tools (do they check or care?)

The classic SE almost surely does not support VOMS ACLs, our view is that no effort should be expended here since these SEs are deprecated.

6.3 WMS

The LCG RB does not understand Glue 1.2 and also does not understand VOMS proxies. So unless the policy changes and development on this software resumes, one can forget the LCG RB in this story.

The gLite WMS is now Glue 1.2 aware. Our best information is that the version in certification now has full support for all of this.

6.4 Configuration

None of this is likely to happen on a large scale unless YAIM is adapted with a default VOMS configuration. Work is proceding on this aspect. You can see a presentation on this by Oliver Keeble at the October 8 GDB meeting, archived in the CERN agenda system.

Missing pieces are:

- a definite default group/role scheme for the four LHC experiments, dteam, and ops VOs. This information should perhaps become part of the VO ID card.
- a definite default setup for Maui
- YAIM machinery to handle the following issues for each new group/role:
  - mapping to queues and setting group ACLs in the LRMS
  - adding ACBR and VOView blocks for each of the groups in the static LDIF file for the CE
  - configuring the `vomap` section of the `lcg-info-dynamic-scheduler`

Please let us know if we are missing something. One thing we definitely are missing is a VOMSified storage schema similar to the one for workload described above – contributions are welcome.

7 Technical Decisions

This segment is included from the report of the EGEE Job Priorities Working Group to document certain deployment choices. An example is the syntax with which YAIM should print the VOMS FQANs.

A number of implementation choices need to be made regarding the entire scheme. Here we list those made so far. These are meant to be pragmatic — we don't know enough to make a definitive statement, but we do know that we want to begin quickly. These choices are hence weighted towards easy implementation and need to be reviewed once we have sufficient experience.

7.1 Syntax of ACBRs

All code that needs to parse the AccessControlBaseRules should conform to the following syntax:

`GlueCEAccessControlBaseRule:<OWS><SNC>:<SNC>`

where `<SNC>` means "string containing no colons" and `<OWS>` means "optional whitespace". Whitespace is forbidden between the `GlueCEAccessControlBaseRule` tag and the semicolon that follows, but in general is OK elsewhere. Here are examples of "good" syntax:

`GlueCEAccessControlBaseRule: VO:atlas`

and similarly with VOMS stuff. What is *not* OK is

```
GlueCEAccessControlBaseRule:   VO :atlas
GlueCEAccessControlBaseRule:   VO: atlas
GlueCEAccessControlBaseRule:   VO : cms
```

due to spaces around the colons (basically handling spaces requires a rewrite of various classads matching functions scattered throughout the WMS code). The following

```
GlueCEAccessControlBaseRule:   VOMS:/atlas/Role=sys:admin
```

is also not OK because this has a second colon, and then the split becomes ambiguous. Similarly this is also not OK:

```
GlueCEAccessControlBaseRule: SERVICE:CLASS:lhcb_bronze
```

7.2 Reporting by Information Provider

If a certain queue (`GlueCE`) supports both CMS as a whole, as well as a special CMS share, there are choices to be made in how to report this. One might make the choice that "CMS as a whole" should mean to report everything belonging to CMS, including the jobs belonging to the special share supported by the same queue.

We decided to make the reporting exclusive: "VO-as-a-whole" `VOView` blocks should report numbers corresponding to all jobs / shares for that VO that aren't explicitly reported by more specific blocks.

To be concrete, if we have `VO : atlas` and `VOMS: /atlas/Role=production` supported by a single CE, in the `VOView` block for the production role, job counts and response-time estimates are reported specifically for the production role; for the `VO : atlas VOView`, counts and estimates are reported for all of atlas *except* the production role.

Note there is some work to be done here in matchmaking, since you might have a VOMS proxy that matches more than one of the published FQANs. We need to develop a matching-precedence hierarchy for the long term so that the concept "most specific match" is meaningful.

8 Appendix: NIKHEF maui.cfg

Note that since not all users are using VOMS, the structure here among the various groups is different than sketched above. However, this file does illustrate values for the various WEIGHT parameters that acheive the desired effect. Also note that this version does not use the recommended structure for SGM vs. normal-user submission, because as of this writing there are still people submitting normal work under SGM credentials.

```
# MAUI configuration example

SERVERHOST              tbn20.nikhef.nl
ADMIN1                  root davidg templon ronalds
ADMIN3                  edginfo rgma
ADMINHOST               tbn20.nikhef.nl
RMTYPE[0]           PBS
RMHOST[0]           tbn20.nikhef.nl
```

```
RMSERVER[0]         tbn20.nikhef.nl

# appears to prevent Maui from occasional hangs;
# see
# http://www.clusterresources.com/pipermail/mauiusers/2005-August/001669.html

RMCFG[0]            TIMEOUT=90
SERVERPORT            40559
SERVERMODE            NORMAL
RMPOLLINTERVAL        00:02:00
LOGFILE               /var/log/maui.log
LOGFILEMAXSIZE        50000000
LOGLEVEL              3
LOGFILEROLLDEPTH      30

NODESETPOLICY           ONEOF
NODESETATTRIBUTE        FEATURE
NODESETLIST             dzero halloween ncf
NODESETDELAY            0:00:00

NODESYNCTIME        0:00:30

NODEACCESSPOLICY            SHARED
NODEAVAILABILITYPOLICY      DEDICATED:PROCS
NODELOADPOLICY              ADJUSTPROCS
DEFERTIME                   0
JOBMAXOVERRUN               0
REJECTNEGPRIOJOBS           FALSE
FEATUREPROCSPEEDHEADER      xps

# Policies
BACKFILLPOLICY              ON
BACKFILLTYPE                FIRSTFIT
NODEALLOCATIONPOLICY        FASTEST
RESERVATIONPOLICY           CURRENTHIGHEST
RESERVATIONDEPTH            12

# expire info in maui.ck after five days

CHECKPOINTEXPIRATIONTIME  5:00:00:00

# Weights of various components in scheduling ranking calc

QUEUETIMEWEIGHT         0
XFACTORWEIGHT           1
XFACTORCAP         100000
RESWEIGHT              10

CREDWEIGHT             10
USERWEIGHT             10
```

```
GROUPWEIGHT            10

FSWEIGHT                1
FSUSERWEIGHT            1
FSGROUPWEIGHT          43
FSQOSWEIGHT          2000

# FairShare
# use dedicated CPU ("wallclocktime used") metering
# decays over 24 "days"
# note we don't use the % yet because we use a private Maui build

FSPOLICY              DEDICATEDPES
FSDEPTH               24
FSINTERVAL            24:00:00
FSDECAY               0.99
FSCAP                 100000

#############################################################################
#
# use PRIORITY to define various levels.
# test groups have highest priority, e.g. dteam PRIORITY=5000
# Tier-1 HEP VOs have next PRIORITY, all = 100
# other VOs have less, e.g biomed PRIORITY 10, esr PRIORITY 50,
# geant PRIORITY 80

# USERs in Maui map to real users
# GROUPs in Maui map to unix GIDs which map to VOs or VO subgroups
# QoS in Maui map to VOs (bundle together VO subgroups)

# for fair scare percentages: see spreadsheet on wiki.
#
# installed capacities
#
# note that fair-share competition between users is absolutely disabled
# unless users have a fair share.  A first guess: use 1%.  This didn't
# work since it meant that any user actually using cycles had a
# rather large discrepancy from the target.  Retry: right now there
# are 14 distinct users with jobs in the queue: set to 1/14.  This means
# that everything else being equal, each user should get an equal share.

USERCFG[DEFAULT]    FSTARGET=50          MAXJOBQUEUED=350
GROUPCFG[DEFAULT]   FSTARGET=1    PRIORITY=1      MAXPROC=330

GROUPCFG[tutor]     FSTARGET=1    PRIORITY=200    MAXPROC=40

# the limits applied appear to be a MIN() of all applicable limits

GROUPCFG[users]     FSTARGET=1    PRIORITY=10     MAXPROC=50
GROUPCFG[dteam]     FSTARGET=1    PRIORITY=500    MAXPROC=32
```

```
GROUPCFG[ops]       FSTARGET=1    PRIORITY=500    MAXPROC=32
GROUPCFG[pvier]     FSTARGET=1    PRIORITY=500    MAXPROC=4

GROUPCFG[atlas]     FSTARGET=8    PRIORITY=100            QDEF=lhcatlas
GROUPCFG[atlb]      FSTARGET=32   PRIORITY=100            QDEF=lhcatlas
GROUPCFG[atlsgm]                  PRIORITY=200    MAXPROC=2     QDEF=lhcatlas
USERCFG[atlas081]   FSTARGET=1-   PRIORITY=1      MAXPROC=1     QDEF=lhcatlas
QOSCFG[lhcatlas]    FSTARGET=43                   MAXPROC=330

GROUPCFG[lhcb]      FSTARGET=29   PRIORITY=100            QDEF=lhclhcb
GROUPCFG[lhcbprd]   FSTARGET=29   PRIORITY=100            QDEF=lhclhcb
GROUPCFG[lhcbsgm]                 PRIORITY=200    MAXPROC=2     QDEF=lhclhcb
QOSCFG[lhclhcb]     FSTARGET=29                   MAXPROC=330

######## afwijker
GROUPCFG[alice]     FSTARGET=10   PRIORITY=100            QDEF=lhcalice
GROUPCFG[alicesgm]  FSTARGET=9    PRIORITY=100            QDEF=lhcalice
GROUPCFG[nikalice]  FSTARGET=1    PRIORITY=100            QDEF=lhcalice
QOSCFG[lhcalice]    FSTARGET=10                   MAXPROC=330
########

GROUPCFG[dzero]     FSTARGET=12   PRIORITY=100                QDEF=vledzero
QOSCFG[vledzero]    FSTARGET=12                   MAXPROC=330

GROUPCFG[phicos]    FSTARGET=2    PRIORITY=100                QDEF=vlephicos
GROUPCFG[phicosgm]                PRIORITY=200    MAXPROC=2     QDEF=vlephicos
QOSCFG[vlephicos]   FSTARGET=2                    MAXPROC=330

GROUPCFG[vlemed]    FSTARGET=2    PRIORITY=100                QDEF=vlevlemed
GROUPCFG[vlefi]     FSTARGET=2    PRIORITY=100                QDEF=vlevlefi
GROUPCFG[vlibu]     FSTARGET=2    PRIORITY=100                QDEF=vlevlibu
QOSCFG[vlevlemed]   FSTARGET=2                    MAXPROC=132
QOSCFG[vlevlefi]    FSTARGET=2                    MAXPROC=132
QOSCFG[vlevlibu]    FSTARGET=2                    MAXPROC=132

GROUPCFG[ncf]       FSTARGET=1    PRIORITY=100                QDEF=ncfncf
QOSCFG[ncfncf]      FSTARGET=1                    MAXPROC=94

GROUPCFG[geant]     FSTARGET=1    PRIORITY=200    MAXPROC=2
GROUPCFG[biome]     FSTARGET=5    PRIORITY=100    MAXPROC=171
GROUPCFG[zeus]      FSTARGET=5    PRIORITY=100    MAXPROC=132
GROUPCFG[esr]       FSTARGET=1    PRIORITY=100    MAXPROC=32

GROUPCFG[emutd]     FSTARGET=10   PRIORITY=100    MAXPROC=132
GROUPCFG[vledut]    FSTARGET=10   PRIORITY=100    MAXPROC=132
GROUPCFG[vldbi]     FSTARGET=10   PRIORITY=100    MAXPROC=132

GROUPCFG[asci]      FSTARGET=10   PRIORITY=50     MAXPROC=132

GROUPCFG[cms]       FSTARGET=1-   PRIORITY=20     MAXPROC=10
```

```
GROUPCFG[cmssgm]                  PRIORITY=40     MAXPROC=2


# versto: maxproc=132 because of size of NCF farm
USERCFG[versto]     FSTARGET=1-   PRIORITY=1      MAXPROC=132

USERCFG[davidg]                   PRIORITY=800
USERCFG[templon]                  PRIORITY=800
USERCFG[ronalds]                  PRIORITY=800
USERCFG[janjust]                  PRIORITY=400    MAXPROC=3

CLASSCFG[qinfinite]               PRIORITY=1
```