Workload Management: How to install Gold Allocation Manager on top of Bright Cluster

How to install Gold Allocation Manager on top of Bright Cluster

Gold Allocation Manager is an HPC resource accounting tool. It is open source, and released under a license that seems to be a BSD-style license.

The following instructions to install it were tested on Bright 7.1 with CentOS6u6

Download the latest sources:

(one line:)

 # wget -c http://www.clusterresources.com/downloads/gold/gold-2.2.0.5.tar.gz

Untar the files:

 # tar -xzvf /root/tmp-install/gold-2.2.0.5.tar.gz

Configure Gold with MySQL database:

 # cd gold-2.2.0.5

 # ./configure --prefix=/cm/shared/apps/gold/2.2.0.5 --with-db=mysql

Compile the sources:

To compile the program:

 # make

To install the Gold web GUI:

 # make gui

Installation:

Workload Management: How to install Gold Allocation Manager on top of Bright Cluster

To install Gold

 # make install

To install the Gold web GUI

 # make install-gui

***** To delete the files created by the Gold installation, you can use 'make uninstall'.***

You will also need to generate a secret key which enables secure communication between clients and server. This key is a passphrase consisting of up to 80 characters and can include spaces and the regular visible ASCII characters. Note that if you are using Gold with Moab or the Maui Scheduler, they will need both need to use a shared secret key.

 # make auth_key

Enter your secret key (up to 80 characters and can include spaces): mysecret

Install the required Perl modules:

 # yum install perl-Log-Log4perl.noarch

 # yum install perl-Config-Properties.noarch

 # yum install perl-Error.noarch

 # yum install perl-Crypt-CBC.noarch

 # yum install perl-Digest-SHA1.x86_64

 # yum install perl-Digest-HMAC.noarch

 # yum install perl-suidperl.x86_64

***** for list of Perl Modules: http://docs.adaptivecomputing.com/gold/x385.php***

Create a Gold database:

 # mysql -u root -p

mysql> create database gold;

Add database user for Gold:

 # mysql -u root -p

mysql> use mysql;

mysql> create user gold;
mysql> grant all on *.* to 'gold'@'localhost';

Initialize the database:

 # mysql -u root -p gold < bank.sql

Edit the configuration files and add the database information:

 # vim /cm/shared/apps/gold/2.2.0.5/etc/goldd.conf

and inside the file set:

database.user = gold

database.password = Ch@ngeMe

Start Gold:

 # /cm/shared/apps/gold/2.2.0.5/sbin/goldd

*** a startup script can be placed under /etc/init.d with the name gold in order to start/stop Gold as a service.*

Now you are now ready to define users, projects, machines, accounts etc. as necessary for your site.

Workload Management: How to install Gold Allocation Manager on top of Bright Cluster

Define Users:

(On one line:)

 # gmkuser -n "Bright Support" -E "support@brightcomputing.com" cmsupport

Define Machines:

 # for i in `seq -w 1 100`; do gmkmachine -d "cn$i" cn$i; done

Modify Machines:

 # for i in `seq -w 1 100`; do gchmachine -d "cn$i" -m cn$i; done

Remove Machines:

 # grmmachine cn001

Define Projects:

 # gmkproject -d "Bright Test" cmsupport

Successfully created 1 Project

Auto-generated Account 1

Add Users to the Projects

 # gchproject --addUsers cmsupport cmsupport

Successfully created 1 ProjectUser

Make Deposit

Add 360000000 credits to the cmsupport project, to be used from 2015 till 2016:

```
# gdeposit -p cmsupport -z 360000000 -s 2015-01-01 -e 2016-01-01


# glsalloc

Id Account Active StartTime  EndTime    Amount    CreditLimit Deposited Description

-- ------- ------ ---------- ---------- --------- ----------- --------- --------------

1  1       True   -infinity  infinity          0           0         0 Auto-Generated

2  1       True   2015-01-01 2016-01-01 360000000           0 360000000
```

Check Balance:

```
# gbalance -p cmsupport

Id Name      Amount    Reserved Balance   CreditLimit Available

-- --------- --------- -------- --------- ----------- ---------

1  cmsupport 360000000        0 360000000           0 360000000

# gbalance -p cmsupport -u cmsupport

Id Name      Amount    Reserved Balance   CreditLimit Available

-- --------- --------- -------- --------- ----------- ---------

1  cmsupport 360000000        0 360000000           0 360000000

# gbalance -u cmsupport

Id Name      Amount    Reserved Balance   CreditLimit Available

-- --------- --------- -------- --------- ----------- ---------

1  cmsupport 360000000        0 360000000           0 360000000

# gbalance -u cmsupport -m cn001

Id Name      Amount    Reserved Balance   CreditLimit Available
```

Workload Management: How to install Gold Allocation Manager on top of Bright Cluster

-- --------- --------- -------- --------- ---------- ---------

1 cmsupport 360000000 0 360000000 0 360000000

*** Now you are ready to run some jobs. Before doing so you will need to integrate Gold with your Resource Management System. Although the quotation, reservation, and charge steps will most likely be invoked automatically by your resource management system, it is useful to understand their effects by invoking them manually.*

Simulate the lifecycle of a job

Assume the job has the following characteristics:

Job Id:

Job Name:
User Name: cmsupport
Project Name: cmsupport
Machine Name: cn001-cn100
Requested Processors: 24
Estimated WallClock: 3600 seconds
Actual WallClock: 1234 seconds

*** When a job is submitted, it is useful to check that the user's account has enough funds to run the job. This will be verified when the job starts, but by that point the job may have waited some time in the queue only to find out it never could have run in the first place. The job quotation step can fulfill this function. Additionally, the quote can be used to determine the cheapest place to run, and to guarantee the current rates will be used when the job is charged.*

How much it will cost to run the job

gquote -p cmsupport -u cmsupport -m cn001 -P 24 -t 3600

Successfully quoted 86400 credits

Make Reservation:

When a job starts, the resource management system creates a reservation (or pending charge) against the appropriate allocations based on the estimated wallclock limit specified for the job.

greserve -J 130.cn-mgmt -p cmsupport -u cmsupport -m cn001 -P 24 -t 3600

Successfully reserved 86400 credits for job 130.cn-mgmt and created gold job id 1

```
[root@cn-mgmt ~]# gbalance

Id Name      Amount    Reserved Balance   CreditLimit Available

-- --------- --------- -------- --------- ----------- ---------

1  cmsupport 360000000    86400 359913600           0 359913600


[root@cn-mgmt ~]# gbalance -p cmsupport

Id Name      Amount    Reserved Balance   CreditLimit Available

-- --------- --------- -------- --------- ----------- ---------

1  cmsupport 360000000    86400 359913600           0 359913600
```

Charge Jobs:

After a job completes, any associated reservations are removed and a charge is issued against the appropriate allocations based on the actual wallclock time used by the job.

```
# gcharge -J 130.cn-mgmt -u cmsupport -p cmsupport -m cn001 -P 24 -X WallDuration=2400


# gbalance -p cmsupport --total

Balance

---------

359942400

The account balance is 359942400 credits
```

Refund Jobs:

```
# grefund -J 130.cn-mgmt

Successfully refunded 57600 credits for job 130.cn-mgmt
```

Check Project Accounting Statement:

 # gstatement -p cmsupport

Check Project Usage:

 # gusage -p cmsupport

Integrate with WLM

Delayed Accounting

In the absence of a dynamic system, some sites enforce allocations by periodically (weekly or nightly) parsing resource manager job logs and then applying debits against the appropriate project accounts. Although Gold can easily support this type of system by the use of the qcharge command in post-processing scripts, this approach allows users or projects to use resources significantly beyond their designated allocation and generally suffers from stale accounting information.

Dynamic Accounting

Gold's design allows it to interact dynamically with your resource management system. Charges for resource utilization can be made immediately when the job finishes (or even incrementally throughout the job). Additionally, reservations can be issued at the start of a job to place a hold against the user's account, thereby ensuring that a job will only start if it has sufficient reserves to complete. The remainder of this document will describe the interactions for dynamic accounting.

Add an appropriate AMCFG line into maui.cfg to tell Maui how to talk to Gold:

vim /cm/shared/apps/maui/current/spool/maui.cfg

The AMCFG line is added in this kind of format:

AMCFG[bank] TYPE=GOLD HOST=cm-mgmt.cm.cluster PORT=7112 SOCKETPROTOCOL=HTTP WIREPROTOCOL=XML CHARGEPOLICY=DEBITALLWC JOBFAILUREACTION=IGNORE TIMEOUT=15

Add a CLIENTCFG line into maui-private.cfg to specify the shared secret key. This secret key will be the same secret key specified in the "make auth_key" step:

 # cat /cm/shared/apps/maui/current/spool/maui-private.cfg

CLIENTCFG[AM:bank] KEY=Ch@ngeMe AUTHTYPE=HMAC64

*** Gold will need to allow the the user id that Maui runs under to perform scheduler related commands (Job Charge, Reserve, Quote, etc).*

Allow the maui user to use Gold:

gmkuser -d "Maui Scheduler" maui

goldsh RoleUser Create Role=Scheduler Name=maui

Methods of interacting with Gold

http://docs.adaptivecomputing.com/gold/x3877.php

Interaction Points

http://docs.adaptivecomputing.com/gold/x3841.php

References:

http://docs.adaptivecomputing.com/gold/

Downloads:

http://www.clusterresources.com/downloads/gold/

Unique solution ID: #1309
Author: mohamed adel
Last update: 2016-04-28 12:16