How to install Condor-G

Tomasz Wlodek

University of the Great State of Texas at Arlington

Abstract: I describe the installation procedure for Condor-G

Before you start: Go to http://www.cs.wisc.edu/condor/condorg/ site and download the tarfile. You will have to sign the license agreement and register yourself, all the usual legal stuff. Then put the tarfile in your favorite temporary area on the machine where you would like the condor-g to be installed.

The machine where you want to install Condor-G must be able to open windows on the machine which serves as your terminal. So make sure that you have the DISPLAY environment variable set correctly and thet your terminal machine runs Exceed or something like that (if it is Windows) or has the xhost variable set and the incoming connections enabled (if it is Linux/Unix). I assume that you know what I am talking about.

Ok, now we are ready.

1. Unzip and untar the file CondorG-GT2-6.3.2-linux-x86-glibc21.tar.gz
2. It will create directory CondorG-Install, enter it.
3. run `sh setup.sh`
4. It will open a window on your machine. Check in this window the directory where you would like condorG to be installed. (Make sure that you have write permission to it). If you do not have Globus tools installed, check the button "Globus tools".
5. Press "Install" button. It will install condor-g.
6. Set the environment variable CONDOR_LOCATION so that points to the place where you have installed Condor-G `export CONDOR_LOCATION=/home/products/condor-g/CondorG`
7. `export CONDOR_CONFIG=$CONDOR_LOCATION/etc/condor_config`
8. Add condor-G binaries to your path: export PATH=$PATH:$CONDOR_LOCATION/bin
9. `cd $CONDOR_LOCATION/bin`
10. `./condor_master` – This will start your condor master

How to run Condor-G jobs

Create a file `test.sub,` which contains:

```
executable = /bin/date
globusscheduler = hepfm000.uta.edu/jobmanager
```

```
universe = globus
output = test.out
log = test.log
queue
```

The first line tells condor-g what the executable is, second where to send it, third that it should be executed by globus (other options are standard and vanilla universes of condor), next lines specify the standard output file of the executable and condor-g log file.

Now type `condor_submit test.sub`

The job will be submitted to machine hepfm000.uta.edu for execution. You can follow the status of job by `condor_q`. When the job is done, you will find its output in `test.out`.

```
Condor-G March 2002 Release
UW Madison Condor Team
condor-admin@cs.wisc.edu
http://www.cs.wisc.edu/condor/
http://www.cs.wisc.edu/condor/CondorG/

1. Installation
2. Overview
3. Examples

1. Installation
Untar the CondorG.tar.gz file. (You must have done so already
if you're
reading this :) It creates a directory called "CondorG-
Install"

cd into 'CondorG-Install', and type "sh setup.sh". This will
fire up the
full installer. After you agree to the license file, select
which directory
you'd like to install CondorG in. By default, it chooses your
home
directory/CondorG.

Down a section there's are two checkboxes. The first one,
"Base Install", is
already checked (and can't be unchecked.) This installs a
Personal CondorG.
There's also a button to install the Globus Tools. You only
need this if
you don't already have tools like 'grid-proxy-init' and such
installed.

At the bottom of the window there's a "View README" button,
which shows this,
a "Cancel" button, and a "Begin Install" button which is
enabled if the
```

installer is satisifed with your choices. Hit Begin, and the installer will
whirl and grind for a short bit, and then tell you that installation was
successful.

To uninstall Condor-G, just run CondorG/uninstall, which will blow everything
away.

2. Overview
CondorG is a Personal-Condor enhanced with Globus Services.

What's Globus? (from www.globus.org)
  The Globus Provides software tools that make it easier to build computational
grids and grid-based applications. These tools are collectively called the
Globus toolkit. Grids are persistent environments that enable software
applications to integrate instruments, displays, computational and
information resources that are managed by diverse organizations in widespread
locations.

What's a Personal-Condor?
  A Personal Condor is a version of Condor running as a regular user, without
any special privileges. The idea is that you can use your Personal Condor to
run jobs on your local workstations and have Condor keep track of their
progress, and then through "flocking" access the resources of other Condor
pools. Additionally, you can "Glide-in" to Globus-Managed resources, and
create virtual-condor pool by running the Condor daemons on the globus
resoures, and then letting your Personal Condor manage those resources.

CondorG also knows how to speak to Globus Resources via GRAM, so it can
be thought of as a souped-up globusrun.

How it all works:

```
condor_master
       |
       +----->condor_schedd
```

```
                ^     |
                |     +----->condor_gridmanager
                |                  |
condor_submit---|                  +<--~GRAM~--
>modi4.ncsa/jobmanager-lsf
                |                  |
condor_q--------|                  +<--~GRAM~--
>vulture.cs/jobmanager-condor
                |
condor_rm ------|
```

The condor_master is the first thing you start, and it creates the Personal
Condor. All it really does is watch out over the other Condor daemons, and
starts them up and shuts them down as approriate.

The condor_schedd keeps track of all of the jobs in the Personal Condor.
You use condor_submit to add jobs to the queue, condor_q to check the status
of the queue, and condor_rm to remove jobs from the queue.

The condor_gridmanager is roughly the equivalent of the condor_shadow. It
is the interface between the Globus Gatekeeper/Jobmanager and the Personal
Condor running on your machine. There is one GridManager per user.

3. Examples

3.1 Starting everything up

cd to the directory you installed CondorG into. If you installed CondorG
somewhere other than the default place, you'll need to set the CONDOR_CONFIG
environment variable so we can find the configuration file. Here's how in
csh:

```
setenv CONDOR_CONFIG /path/to/CondorG/etc/condor_config
```

Now you should add the CondorG binaries to your path: (csh example again)

```
setenv PATH /path/to/CondorG/bin:$PATH
rehash
```

*Please note that both of these examles do not persitantly set these variables.

Please see your UNIX admin if you're unclear how to persistantly set them)

Now cd into /path/to/CondorG/sbin, and type './condor_master'. Your personal
Condor is now started. You can type "condor_q" and see:

```
[epaulson@localhost etc]$ condor_q

-- Submitter: wireless48.cs.wisc.edu : <128.105.48.148:33012>
: wireless48.cs.wisc.edu
 ID      OWNER            SUBMITTED     RUN_TIME ST PRI SIZE
CMD

0 jobs; 0 idle, 0 running, 0 held
[epaulson@localhost etc]$
```

which, since we've never submitted anything is exactly what we're expecting.

Now for a quick test: let's run a copy of the /bin/date program on our machine
on a remote machine controlled by Globus. (See 4.1 on how to run a pre-staged
executable)

First, create a file called "g-test.sub", and put in it:

```
executable = /bin/date
globusscheduler = biron.cs.wisc.edu/jobmanager
universe = globus
output = globus-test.out
log = globus-test.log
queue
```

(Replace biron.cs.wisc.edu with a machine you've got access to).

Make sure you've got some creditials, and run grid-proxy-init if you don't.
Now, type 'condor_submit g-test.sub' run your job. A condor_q will now show:

```
[epaulson@localhost ~/temp]$ condor_q


-- Submitter: wireless48.cs.wisc.edu : <128.105.48.148:33012>
: wireless48.cs.wisc.edu
 ID      OWNER            SUBMITTED     RUN_TIME ST PRI SIZE
CMD
   7.0   epaulson        3/26 14:08   0+00:00:00 I  0   0.0
date
```

```
1 jobs; 1 idle, 0 running, 0 held
```

In a few moments, once Globus has accepted your job, you'll see:

```
[epaulson@localhost ~/temp]$ condor_q


-- Submitter: wireless48.cs.wisc.edu : <128.105.48.148:33012>
: wireless48.cs.wisc.edu
 ID      OWNER            SUBMITTED     RUN_TIME ST PRI SIZE
CMD
   7.0   epaulson        3/26 14:08   0+00:00:00 R  0   0.0
date

1 jobs; 0 idle, 1 running, 0 held
[epaulson@localhost ~/temp]$
```

Then, very shorly after that, the queue will be empty again:

```
[epaulson@localhost ~/temp]$ condor_q


-- Submitter: wireless48.cs.wisc.edu : <128.105.48.148:33012>
: wireless48.cs.wisc.edu
 ID      OWNER            SUBMITTED     RUN_TIME ST PRI SIZE
CMD

0 jobs; 0 idle, 0 running, 0 held
```

You can check the contents of globus-test.out and see:

```
[epaulson@localhost ~/temp]$ cat globus-test.out
Mon Mar 26 14:09:13 CST 2001
```

Huzzah!

4. Things to watch out for:

4.1 CondorG BY DEFAULT TRANSFERS YOUR EXECUTABLES! For example, if you say

```
executable = /bin/date
```

in your submit file, CondorG will COPY /bin/date from your SUBMIT MACHINE.
If you want to run the remote /bin/date, add

```
transfer_executable = false
```

to your submit file.

4.2 The job exit status is pretty bogus right now - we can't get that back from
Globus, so we just tell you that it exited with status zero. The one time
we say something different is when loose track of a job - this happens when
we can contact the Gatekeeper, but not the job manager. In this case, since
there's no way for us to find out about what happened to that job, we
take it out of the queue with exit status 1. We've got enhancements to the
Globus Job Manager that will let us re-attach to a job that we might have
lost track of later in the future, but they haven't been deployed anywhere
yet. We also don't currently send email when a job completes yet, but
stdout and stderr are constantly streamed back.

4.3 To use a non-standard X509 proxy, you can either specify the path to the
proxy in the submit file by setting the x509userproxy option. For example,
x509userproxy = /home/epaulson/x509up_21002

If the x509userproxy option is not listed, CondorG will use the environment
variable X509_USER_PROXY (as will the other Globus tools.) If that environment
variable does not exist, CondorG will use the default location for the
proxy (currently /tmp/x509up_<unix_uid>)

CondorG will create a single GridManager for each User+Proxy combination.

4.4 The "globusrsl" option can be used to add additional entries in the RSL
string that ultimately gets submitted to Globus. If the RSL string starts
with an '&', the entire "globusrsl" entry of the submit file is used, and
CondorG will not construct one from the other entries of the submit file. If
the RSL string begins with a '+', CondorG will use DUROC to launch your
jobs.

Appendix A
New Entries to the Condor Config File, not present in the current manual:

```
GRIDMANAGER             = $(SBIN)/condor_gridmanager
GRIDMANAGER_LOG         = $(LOG)/GridmanagerLog
MAX_GRIDMANAGER_LOG     = 64000
GRIDMANAGER_DEBUG       = D_SECONDS D_COMMAND
CRED_MIN_TIME_LEFT = 8*60*60;
GRIDMANAGER_CHECKPROXY_INTERVAL = 600
GRIDMANAGER_MINIMUM_PROXY_TIME = 180

DISABLE_AUTH_NEGOTIATION = TRUE
```

GRIDMANAGER is the path to the gridmanager daemon
the GRIDMANGER_LOG and MAX_GRIDMANAGER_LOG entries control where and how log
the logfiles should get. GRIDMANAGER_DEBUG sets some debugging levels for
the GridMananger.

The only really interesting file there is CRED_MIN_TIME_LEFT. Condor-G
will check to make sure that your proxy has at least this much life left in
it before it will submit your job. For example, if your job runs for a
half-hour and there's a 40 minute queue wait time, you'd best make sure you've
got at least 70 minutes of proxy lifetime left. This entry is in seconds, and
defaults to 8 hours. Currently only condor_submit uses this entry.

The next two entries are entries for the Gridmanager, and are only really
uesful if you're using the newest Job Managers from Globus 2.0.

GRIDMANAGER_CHECKPROXY_INTERVAL is, in seconds, how often CondorG should
check for an updated proxy. For example, if you create a 12 hour proxy, and then
6 hours later re-run grid-proxy-init, CondorG will check the proxy within
this time interval, and use the new proxy it finds there. It defaults to 10
minutes.

CondorG also knows when the proxy of each job will expire, and if the proxy
is not refreshed before "GRIDMANAGER_MINIMUM_PROXY_TIME" seconds before the
proxy expires, CondorG will shut down. In other words, if

GRIDMANAGER_MINIMUM_PROXY_TIME is 180, and the proxy is 3 minutes away from
expiring, CondorG will attempt to safely shut down, instead of simply loosing
contact with the remote job because it is unable to authenticate it. The
default setting is 3 minutes (180 seconds)

The last entry is not really a CondorG entry, but affects it anyway.
Condor 6.3.2 will include Kerberos support, but it is not quite ready for
prime-time. The code is already present in Condor, but if
DISABLE_AUTH_NEGOTIATION is set to TRUE, Condor will use the 6.2-style
authencation/authorization methods. CondorG users should use this setting
for this version of Condor.

Please contact condor-admin@cs.wisc.edu with any questions/concerns.

The CondorG installation program is based off the Loki Games installer and
is licensed under the GNU Public License. The source code for the installer
is available from

http://www.cs.wisc.edu/~epaulson/installer_src.tar.gz