SYNERGY
FOR SMART MULTI-OBJECTIVE OPTIMISATION

GRID'5000 TESTBED : GETTING STARTED TUTORIAL

Jan Gmys, Matthieu Marquillie and Nouredine Melab
(Université Lille 1)

1 Getting the environment

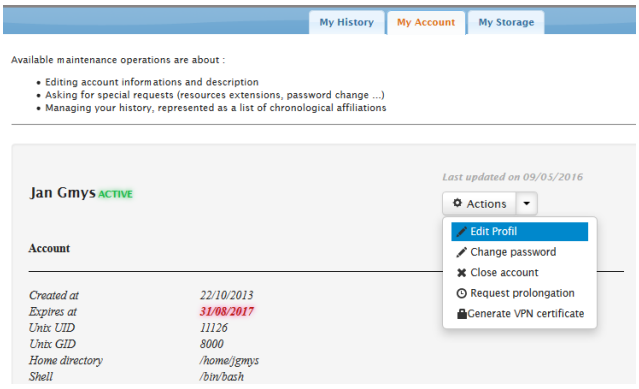
1.1 Authentication / Connection to Grid'5000

> *The Secure Shell (ssh) network protocol is the main tool used to access the Grid'5000 testbed. When connecting to a remote machine, the standard way to authenticate is via the remote account's login and password. However, for security reasons this authentication scheme is disabled on Grid'5000. Instead, Grid'5000 users must use public-key authentication, as follows :*

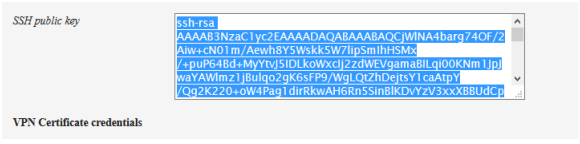
On your local machine, create a key pair using `ssh-keygen`. To do this, enter the following command and save the key in `/home/login/.ssh/id_rsa`.

```
YOUR MACHINE : ssh-keygen -t rsa
```

Now, you should upload the public key (`/home/login/.ssh/id_rsa.pub`) using the Account Management Interface. On the g5k website, go to *Users Portal* > *Manage Account* and choose the option *edit profile* (Fig. 1a). Then, copy-and-paste the public_key to the *SSH public key* field (Fig. 1b). Note that it is possible to enter several keys, for different workstations, one after the other.



(a) Users Portal > Manage Account.



(b) Paste the *whole* ssh public key here.

FIGURE 1 – Enabling your authentication in Grid'5000

> *The Grid'5000 grid (g5k) that we will use during the tutorials is distributed among different sites : Lille, Nancy, Rennes, ... These different sites are connected by the RENATER network. By now, you are ready to access g5k by connecting with* `ssh` *to one of its access points. The command to use to connect to this gateway is :*
>
> `YOUR MACHINE :` `ssh login@access.grid5000.fr`

- Connect to the g5k access site.
- Notice the different information that are given, in particular how to transfer data from outside to g5k.

1.2 Connections to frontends

> *Resources reservation on g5k is performed from a particular machine called the* `frontend`*. You need to connect to the frontend with* `ssh`*. From the access machines you can connect to the frontend of a particular site with the command :*
>
> `ACCESS :` `ssh site`
>
> *You can also use this command when you are already connected to a frontend and you would like to connect to another frontend*

- Connect to the Lille site.
- Disconnect from the Lille site, then try to connect on some other sites.
- Try to connect directly to the frontend of another site.
- Connect to the Lille site and try to connect to one of the compute nodes (for instance `chinqchint-11.lille.grid5000.fr`) with the `ssh` command.

> *Compute nodes are not available with a simple* `ssh`*. You need to reserve them before obtaining any access.*

1.3 Working with files

- From the *access point* create a test file with the following command :

  `ACCESS :` `echo "this is a test file" > lille/test_login.txt`

- Connect to the Lille frontend. Where is the test file located ? How about the one of your neighbour ?
- Connect to another site. Where is the test file located ?

> *The access points and the frontends are available by* `ssh` *from any machine on g5k. Resource access is managed by the LDAP protocol. Your account is duplicated by NFS only on the nodes of a same site (included the compute nodes !).*

🚲 From the access point, copy the test file to another site :

```
ACCESS : cp lille/test_login.txt site/
```

🚲 Connect to this site. Is your file available on this site now ?

☞ *NB : Alternatively one can use* `scp` *or* `rsync` *to copy data between sites (this is preferable for large volumes of data) :*

```
FRONTEND : scp test_login.txt site:
```

```
FRONTEND : rsync test_login.txt site:
```

1.4 Transferring files

> 🛑 *The* `rsync` *command can also be used to transfer files between your local machine and g5k.* For example*, the following command transfers a local file to your home directory on the lille site*
>
> ```
> YOUR MACHINE : rsync file login@access.grid5000.fr:lille/
> ```
>
> 🛑 *And the following command transfers a file from the lille site to your current local directory :*
>
> ```
> YOUR MACHINE : rsync login@access.grid5000.fr:lille/file ./
> ```

☞ *NB : The* `rsync` *command can transfer multiple files at the same time and also full directories. For example, the following command copies the directory* `dir` *to your home directory on lille (check* `man rsync` *for more details) :*

```
YOUR MACHINE : rsync -azvP ./dir login@access.grid5000.fr:lille/
```

2 Supervise and reserve with OAR

> 🛑 *Now you are ready to reserve some resources. Before making a reservation you must check for the grid status by using a supervision tool. We will learn to use* `Monika`*, a web interface dedicated to g5k.*

🚲 Connect to `https://www.grid5000.fr` and go to section : `Users portal` → `Platform status`

🚲 Notice the status of the different nodes on the Lille cluster. Look also for the different job details.

> 🛑 *OAR is a software which allows the users to manage the resources available on g5k site. It allows to :*
> — *Consult the resources (*`oarstat`*)*
> — *Reserve some resources (*`oarsub`*)*
> — *Access to the resources (*`oarsh`*)*

2.1 `oarstat` and `oarnodes`

Use the commands `oarstat` and `oarnodes`. Test the different commands with the options `-j`, `-u` , `-f`. Look for the `man` for further information.

To sum up, `oarstat` *and* `oarnodes` *allow to get in command line the same proposed services by Monika. As we will see, these informations will be very useful for starting a task.*

2.2 `oarsub`

The `oarsub` *scheduler allows to reserve some nodes on g5k. We will discuss about two reservation modes :*

- *The interactive mode : the user is directly connected on one of his reservation nodes. He can after execute his own scripts*
- *The passive mode : a script is specified during the reservation. This script is then executed on one of the reservation nodes. The user is not directly connected to a node. This mode also allows to schedule a task.*

2.3 oarsub : the interactive mode

Reserve a resource in interactive mode by using the command :

```
FRONTEND : oarsub -I
```

`oar` *returns a number :* `JOBID`*. This number corresponds to the reservation that you have just made. Now that you have reserved a resource, you can execute a program on this resource. We will come back on it later.*

To supervise your reservations, use `oarstat/oarnodes` in a second terminal (you could also use Monika ...)

Visualize the reservation that you have just made. Note the attributed resource (node), the duration, the status, the identification, etc...

To finish your reservation, use the following command from the second terminal :

```
FRONTEND : oardel {JOBID}
```

Notice that by default only one machine (possibly a multi-core) has been assigned to you for a duration of one hour. We will now reserve more resources for a shorter duration and execute some scripts.

Reserve two nodes for a duration of five minutes by using the following command :

```
FRONTEND : oarsub -I -l nodes=2,walltime=00:05:00
```

Which nodes have been allocated ? To find out, enter the command :

```
NODE : uniq $OAR_NODEFILE
```

- *Once a node is reserved, you can connect to it using the commands :*

```
FRONTEND : oarsub -C {JOBID}

NODE : oarsh {node}.{site_name}.grid5000.fr
```

- Connect to the second reserved node by using `oarsh` from a second terminal. Then disconnect from it (`exit`). Is the reservation still active ?
- Disconnect from both nodes and verify that your reservation is finished.

- *When you perform an interactive reservation, you are located on a main node. The information of your reservation can be obtained by using the* `oar` *environment variables. Thus from the main node, you can access via* `oarsh` *the others nodes of your reservation and execute some programs. The resources access via* `oarsh` *is limited by your reservation duration : once your reservation is finished, it is not possible to connect on any resources any more. The termination of a reservation is done with* `oardel`*, or when you disconnect yourself from the main node (*`exit`*), or when the reservation time is expired (*`walltime`*). The end of a program execution does not determine the end of an interactive reservation.*

2.4 oarsub : passive reservation

- *The* `oar` *scheduler allows to perform submissions in passive mode. In this mode, the resources are reserved but the user is not directly connected on a reserved node. In the following, we will see the interest of such reservation.*

- *In the passive mode, a script specified during the reservation is executed on the first allocated node. The script must determine the obtained nodes, then it must distribute the work among them. This can be performed by manipulating the* `oar` *environment variables. The following script gives an example.*

- Download the following script from `http://www.lifl.fr/~melab/HTML/Synergy/Tuto-G5K/test_script.sh` (be careful to put the correct ~ sign in the URL) and transfer it to the Lille site. If you prefer, you may also edit it in your work directory - in any case do not forget to do a `chmod +x`.

```
#!/bin/bash

machines=`uniq $OAR_NODEFILE`
for m in $machines
do
    echo Execution on the machine $m .....:
    oarsh $m echo `date` > $m &
    i=0
    while [ $i -ne 50000 ]
```

```
    do
        let "i=i+1"
        echo $i > counting_$m
    done
done
```

- Make a passive reservation by using the following command :

```
FRONTEND : oarsub -l nodes=2,walltime=00:05:00 ./test_script.sh
```

- Using `oarstat`, verify the visibility of your and check the output of the script.

> *In passive mode, the standard output (respectively error) of the script is redirected to the* `OAR.JOBID.stdout` *file (respectively* `OAR.JOBID.stderr`*) in your directory. It is thus possible to retrieve the script execution or to follow it's execution in direct.*

2.5 oarsub : passive planning

> *It is possible to reserve some resources in order to use them at a given time in the future. This is particularly interesting for the planning of long tasks which need lots of resources.*

- Plan the execution of one of your previous script by using the following command :

```
FRONTEND : oarsub -r "AAAA-MM-JJ hh:mm:ss" -l nodes=2,walltime=00:05:00 ./{script}
```

• Follow the execution progress by using in particular `oarstat` and by observing the status of your reservation. Verify the obtained result as before.

• Finish your reservation.

2.6 OpenMP application

- The following file is downloadable at
  `http://www.lifl.fr/~melab/HTML/Synergy/Tuto-G5K/hello_openmp.c`.
  Get it and transfer it to a g5k frontend (or edit it directly on the frontend using `emacs`, `vim` or `nano`)

```
#include <omp.h>
#include <stdio.h>
#include <stdlib.h>

int main (int argc, char *argv[])
{
  int nthreads, tid;

  #pragma omp parallel private(nthreads, tid)
  {
```

```
    /* Obtain thread number */
    tid = omp_get_thread_num();
    nthreads = omp_get_num_threads();

    printf("Hello World from thread %d / %d\n", tid, nthreads);
  }
}
```

- Connect to a node (in interactive mode) and compile the program with the command :

```
NODE : gcc -fopenmp {file_name} -o {exec_name}
```

- Set the environment variable `OMP_NUM_THREADS` and run the program.

```
NODE : export OMP_NUM_THREADS=<N> ; ./{exec_name}
```

2.7 kadeploy3

> *Using the* `oarsub` *command as above allows you to access resources configured in their default environment. However, your experiment might require changing the software environment in some way and therefore it may be useful to have administrative privileges (root access) on the resources you reserved. G5k offers this possibility to its users and the goal of this part of the tutorial is to learn how g5k can be used in this advanced way.*

- Reserve a node, allowing a deployment with kadeploy

```
FRONTEND : oarsub -I -l {...} -t deploy
```

- On each site a library of images is maintained in `/grid5000/images`. You can list the available default environments using the command

```
FRONTEND : kaenv3 -l
```

> *Grid'5000 can be used as a* Hardware-as-a-Service *Cloud, thanks to* `kadeploy3`*, which allows one to deploy a software environment on a pool of nodes for the duration of their reservation. The command* `man kadeploy3` *and the g5k wiki pages provide further information.*

- Deploy an image of your choice on the reserved nodes. For instance, to deploy the image `jessie-x64-base` (a basic version of Debian Jessie) use the command

```
FRONTEND : kadeploy3 -f $OAR_NODEFILE -e jessie-x64-base -k
```

- After the deployment, you can connect to the reserved node as a root user using one of the two following commands

```
FRONTEND : ssh root@{node}
```

```
FRONTEND : ssh root@(head -1 $OAR_NODEFILE)
```

As a root user you are now able to access websites outside Grid5000 (NB : This activity is logged and monitored). For example you can use `wget` *to download packages and* `apt-get` *to install them. In this tutorial we will create a new user.*

Create a new user account using the following command. Notice the creation of a `/home` directory.

```
ROOT@NODE : adduser {username}
```

Now, from a second terminal connect to the node with the login you have chosen

```
FRONTEND : ssh {username}@{node}
```

Now let's save the modified environment, so that we can re-deploy it in the future. We will create an image of the deployed environment and make it compatible with `kadeploy3`

To create an image of the modified environment you can use the `tgz-g5k` script. You can use for example one of the following two commands

```
FRONTEND : ssh root@{node} tgz-g5k > {path_to_myimage}.tgz
```

```
ROOT@NODE : tgz-g5k {login}@frontend:{path_to_myimage}.tgz
```

Now you need to create a description of the environment. The description of the initial image is a good starting point, so print it to a new description file. For instance (assuming that `jessie-x64-base` was used) you can do this by entering

```
FRONTEND : kaenv3 -p jessie-x64-base -u deploy > myjessie-x64-base.env
```

The created environment file (for example `myjessie-x64-base.env`*) must be edited. In particular the* **name, description, author** *and* **image > file** *lines must be changed. Also the line* **visibility** *must be suppressed or changed to* private *or* shared.

Now the newly created environment can be deployed using

```
FRONTEND : kadeploy3 -f $OAR_NODEFILE -a myjessie-x64-base.env
```

Once you have tuned, configured and customized your environment you may want to add it to the `kadeploy` *database. This will allow you to deploy it directly with the* `kadeploy3` *command. The environment can be added using the command*

```
FRONTEND : kaenv3 -a myjessie-x64-base.env
```