Краткое руководство пользователя вычислителей «Политехник - РСК Торнадо» и «Политехник - РСК Петастрим»

Содержание

1. Доступные ресурсы

Пользователям СКЦ «Политехнический» в настоящее время доступны следующие вычислительные ресурсы:

- 612 узлов кластера "Политехник - РСК Торнадо" (далее узлы *tornado*)
  - 2 x Intel Xeon CPU E5-2697 v3 @ 2.60GHz
  - 64G RAM
- 56 узлов кластера "Политехник - РСК Торнадо" с ускорителями NVIDIA K-40 (далее узлы *tornado-k40*)
  - 2 x Intel Xeon CPU E5-2697 v3 @ 2.60GHz
  - 64G RAM
  - 2 x Nvidia Tesla K40x 12G GDDR
- 288 узлов вычислителя с ультравысокой многопоточностью "Политехник - РСК Петастрим" (далее узлы *mic*)
  - 1 x Intel Xeon Phi 5120D @ 1.10GHz
  - 8G RAM

Все доступные узлы используют сеть 56Gbps FDR Infiniband в качестве интерконнекта. Также, на всех узлах доступна параллельная файловая система Lustre объёмом около 1 ПБ.

По умолчанию пользователю предоставляется доступ к узлам *tornado*. Доступ к остальным типам узлов предоставляется по запросу.

2. Доступ к ресурсам

Зарегистрированные пользователи доступ к вычислительным ресурсам производят с машины login1.hpc.spbstu.ru.

Вход осуществляется с использованием протокола SSH. Для подключения можно использовать любой терминальный клиент, поддерживающий протокол SSH, в том числе:

- PuTTY - для Windows (http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html),
- OpenSSH (http://openssh.org/).

Для копирования файлов терминальных клиентов можно использовать следующие утилиты:

- WinSCP - для Windows, графический интерфейс, (http://winscp.org/);
- pscp/psftp - для Windows, текстовый интерфейс, (http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html).

3. Создание SSH-ключей

Для аутентификации подключений к ресурсам СКЦ используются SSH-ключи, представляющих собою пару файлов. Один из них называется закрытым ключом; он, обычно, защищен паролем, хранится на компьютере пользователя и никому не показывается. Второй файл называется открытым ключом и хранится на том компьютере, к которому осуществляется подключение. Фактически, он дополняет первый ключ, и только при наличии правильной пары ключей можно установить соединение. Поэтому для работы с кластером необходимо сгенерировать такую пару ключей.

3.1. Создание SSH-ключей на Unix-системах

Необходимо наличие пакета OpenSSH (в большинстве дистрибутивов устанавливается по умолчанию). Команда, приведенная ниже, генерирует пару ключей с 4096-битным шифрованием по алгоритму RSA (символ # обозначает приглашение командной строки). В процессе работы сначала запрашивается имя файла, в котором будет сохранен ключ (по умолчанию это /home/user/.ssh/id_rsa), при необходимости его можно задать, чтобы иметь несколько пар ключей. Далее предлагается ввести пароль для защиты ключа (и подтвердить повторным вводом), этот пароль необходимо вводить каждый раз при проверке ключей.

```
# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user/.ssh/id_rsa.
Your public key has been saved in /home/user/.ssh/id_rsa.pub.
The key fingerprint is:
bf:db:05:bc:0d:c9:2c:b4:8c:2a:c2:70:35:82:05:dd user@host
The key's randomart image is:
+--[RSA   4096]---+
|.o..             |
| o. E            |
|. . o     .      |
|   o .   + = .   |
|. .     S + B    |
| +     . . . =   |
|  o . .   . . o  |
|   . .     o .   |
|          o..    |
+-----------------+
```

По завершении работы приложения *ssh-keygen* закрытый ключ будет находиться в файле /home/user/.ssh/id_rsa, а открытый — в /home/user/.ssh/id_rsa.pub. Файл с открытым ключом необходимо отправить в службу регистрации СКЦ.

3.2. Создание SSH-ключей на Windows системах

Пример создания ключа для клиента PuTTY.

Для создания SSH-ключа необходима утилита PuTTYgen (http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html). После ее запуска надо задать тип ключа «SSH-2 RSA», его размер — 2048 бит, и нажать кнопку "Generate" (рис.1).

Рисунок 1. Задание параметров генерации SSH-ключей

Далее необходимо поводить курсором мыши в пустой области окна программы до полного заполнения шкалы. После этого будет создан ключ, для которого предлагается ввести пароль (рис.2).

Рисунок 2. Интерфейс утилиты PuTTYgen

Чтобы сохранить полученные ключи, необходимо нажать кнопку "Save public key" и ввести имя файла id_rsa.pub для открытого ключа, затем нажать кнопку "Save private key" и ввести имя файла для закрытого ключа, например mykey.ppk. Файл с открытым ключом необходимо отправить в службу регистрации СКЦ..

Для того, чтобы программа PuTTY могла использовать эту пару ключей, необходимо в настройках сессии Connection > SSH > Auth в поле "Private key file for authentication" задать путь к файлу с закрытым ключом (рис.3).

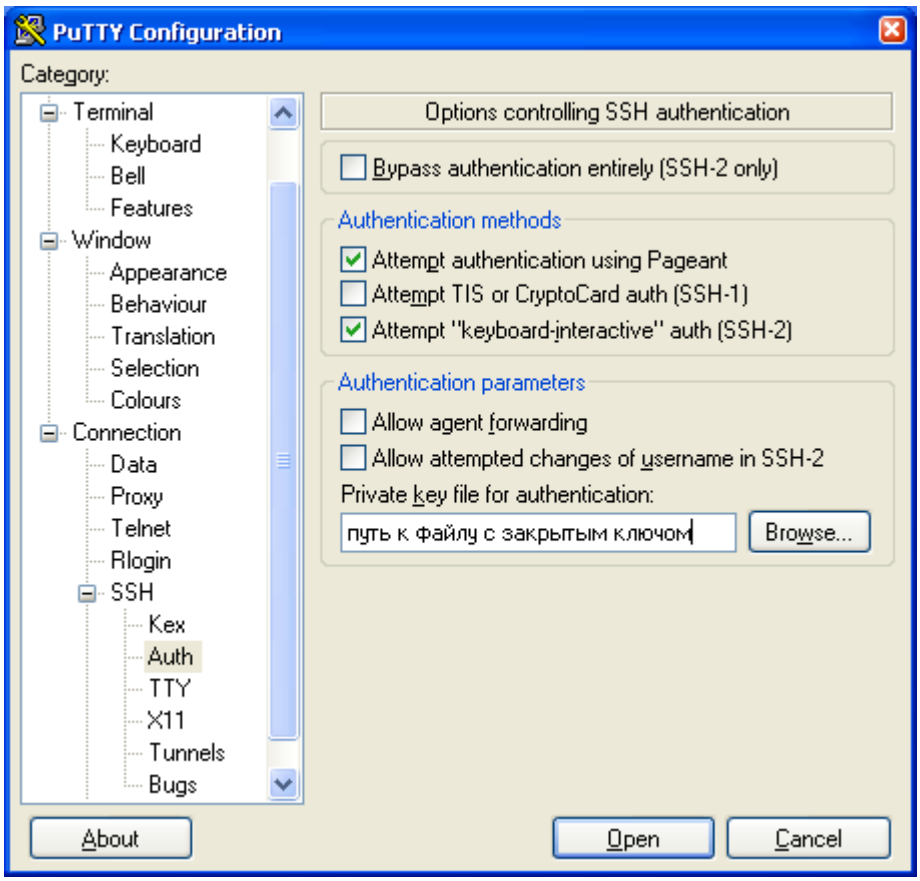

Рисунок 3. Конфигурирование утилиты PuTTYgen

4. Настройка окружения

Для управления окружением на ресурсах СКЦ «Политехнический» используется система так называемых Envirement Modules. С помощью них модифицируется окружение пользователя. Список доступных модулей можно посмотреть так:

$ *module avail*

---------- /usr/share/Modules/modulefiles ----------
dot module-git module-info modules null use.own

---------- /opt/basis/modules ----------
ansys/apdl/16.2 gromacs/2016/gcc mic17
ansys/apdl/17.0 gromacs/2016/intel mic_pmi
ansys/apdl/latest gromacs/2016/mic mic_pmi_2017
ansys/cfx/16.2 gromacs/5.1.2/intel
ansys/cfx/17.0 gromacs/5.1.2/mic
ansys/cfx/latest intel/2016.0.109 mpi/openmpi/2.0.1/gcc/6.2.0

ansys/fluent/16.2 intel/2016.1.150 numeca/fine101
ansys/fluent/17.0 intel/2016.3.210(default) nvidia/cuda-7.5
ansys/fluent/latest intel/2017.0.098 parallel/mpi.intel/2017.0.098
compiler/gcc/6.1.0 parallel/mpi.intel/5.1.1.109
compiler/gcc/6.2.0 intel_license parallel/mpi.intel/5.1.2.150
comsol/52 launcher/mpiexec parallel/mpi.intel/5.1.3.210
espresso/5.4.0 launcher/slurm parallel/mpi.intel/latest
fds/6.4.0 library/fftw/3.3.4/gcc parallel/openmpi/1.10.2/gcc/6.1.0
fftw/3.3.4/gcc library/fftw/3.3.5/gcc parallel/openmpi/1.10.2/intel/2016.3.210
gcc/5.3.0 matlab/2013a python/3.5.2
gcc/6.1.0 mic

Таким образом можно выбрать наборы компиляторов, библиотек а также предустановленного программного обеспечения. На текущий момент доступны следующие компиляторы:

- Intel Compiler 2016.0.109
- Intel Compiler 2016.1.150
- Intel Compiler 2016.3.210
- Intel Compiler 2017.0.098
- GCC 5.3.0
- GCC 6.1.0
- GCC 6.2.0

Доступные библиотеки MPI:

- Intel MPI 5.0.109
- Intel MPI 5.1.150
- Intel MPI 5.3.210
- Intel MPI 2017.0.098
- OpenMPI 1.10.2
- OpenMPI 2.0.1

Загрузка требуемых модулей осуществляется с помощью команды module load

$ *module load compiler/gcc/6.2.0*

Выгрузка произвольного заданного модуля осуществляется с помощью команды *module unload*, например:

$ *module unload compiler/gcc/6.2.0*

Очистка всех загруженных модулей осуществляется с помощью команды

$ *module purge*

5. Запуск задач

Управление ресурсами кластера осуществляется с помощью программного пакета SLURM. Принцип его работы можно описать следующим образом: пользователь запрашивает некоторый ресурс (процессорные ядра, память и т.п.), размещая свою задачу в очереди; система, основываясь на приоритетах пользователя и текущем заполнении очереди, выбирает момент запуска задачи. Под очередью понимается последовательность задач, которая должны решаться на определенном вычислительном ресурсе (группе узлов). На данный момент доступны три очереди для узлов:

- *tornando,*
- *tornado-k40,*
- *mic*

При этом, каждый узел в текущий момент времени может быть занят только одной задачей, одного пользователя; таким образом узел отводится в монопольное использование размещенной на нем задаче, т.е. другие задачи на занятом узле выполнятся не будут.

Наиболее приемлемый способ запуска задач, это использования утилиты *sbatch* и командного файла. Пример такого файла приведен ниже

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=28
#SBATCH -p tornado
#SBATCH -t 10-00:00:00
#SBATCH -J jobname
#SBATCH -o jobname-%j.out
#SBATCH -e jobname-%j.err
if [ -f /etc/profile.d/modules-basis.sh ]; then
        source /etc/profile.d/modules-basis.sh
fi
module purge
```

module load mpi/openmpi/2.0.1/gcc/6.2.0

mpiexec /path/to/my/mpi/app

Разберем параметры указанные в описании командного файла.

--nodes — количество запрашиваемых узлов;

--tasks-per-node – количество mpi ранков на узел;

--cpus-per-task – количество openmp процессов на mpi ранк;

-p – название очереди (типа узлов) куда уйдет задача;

-t – максимальное время выполнения задачи (не должно превышать данный параметр для очереди);

-J — имя задачи (как оно будет отображаться в очереди);

-o – файл куда будет записан весь стандартный вывод (stdout);

-e – файл куда будет записан весь стандартный вывод ошибок (stderr).

Более подробное описание доступно на страницах справки утилит (man sbatch).

Приведенный выше командный файл можно поставить в очередь с помощью команды sbatch:

$ *sbatch run.slurm*

Посмотреть статус задачи можно с помощью утилиты squeue, например:

$ *squeue*

JOBID	PARTITION	NAME	USER	STATE	TIME	NODES
59697	tornado	jobname	user	R	0:01	1

Снять задачу можно с помощью команды scancel:

$ *scancel 59697*

Если задача встала в очередь, то время её запуска можно узнать с помощью команды scontrol

$ *scontrol show jobid 59697*

В выводе будет строка содержащая StartTime, в которой указывается ориентировочное время запуска задачи.