

Taiwania 1 HPC System User Operation Manual

Version	1.4
Updated on	2020/01/08

Change History

Version	Date	Change details	Changed by	Reviewed by	Approved by
0.1	8 th Feb 2018	Initial Version	Imura	Ishan	Yamada
1.0	28th Mar 2018	- Revised Chapter 2.3, 4.1, 5.2 - Added Chapter 6 - Resource limit in login nodes	Imura	Ishan	Yamada
1.1	12th Apr 2018	- Modified Chapter 4.2.4 MPI parallel program - Modified Chapter 4.3.4 MPI parallel program - Added Chapter 4.4.2 MPI parallel CUDA program - Added Chapter 5.3.4 Specifying E-mail notification - Added Chapter 5.6 Creating and using reservation	Yoshida	Ishan	Yamada
1.2	3rd Jul 2018	- Modify Job Script Example	Oscar	Oscar	Oscar
	25th Jul 2018	- Added ct160 queue	Oscar	Oscar	Oscar
1.3	28th Dec 2018	- Modify Chapter 5.1 - Modified Job queue - Added Queue policy	Oscar	Oscar	Oscar
	5th Jul 2019	- Modify Chapter 2.4.1 - Modify Chapter 4.2.1 - Modify Chapter 4.3.1 - Modify Chapter 2.3.1 - Remove Chapter 5.6 - Remove Chapter 6	Oscar	Oscar	Oscar
	22nd Aug 2019	- Modify Chapter 5.1 - Modified Job queue	Oscar	Oscar	Oscar
1.4	18th Nov 2019	- Modify Cover Page - Modify Chapter 2.2 - Modify Chapter 2.4.1 - Remove Chapter 4.4 - Modify Chapter 5.1	Oscar	Oscar	Oscar
	8th Jan 2020	- Modify Chapter 2.2 - Modify Chapter 4.3.4 - Modify Chapter 5.1	Oscar	Oscar	Oscar

Contents

1. INTRODUCTION.....	6
2. PETA HPC SYSTEM	6
2.1. SYSTEM OVERVIEW.....	6
2.2. AVAILABLE COMPUTE RESOURCES	7
2.3. AVAILABLE STORAGE RESOURCE.....	7
2.3.1. <i>Home area</i>	7
2.3.2. <i>Temporary work area</i>	7
2.3.3. <i>Project storage area</i>	8
2.4. FRONT-END SERVERS FOR USER ACCESS	8
2.4.1. <i>Login nodes</i>	8
2.4.2. <i>Interactive nodes</i>	9
2.4.3. <i>Data transfer nodes</i>	9
3. SYSTEM ACCESS METHOD	10
3.1. MEMBER ACCOUNT & PETA SYSTEM ACCOUNT REGISTRATION	10
3.2. COMMAND LINE LOGIN.....	14
3.3. CHANGING PASSWORD	16
3.4. COMMAND LINE LOGOUT	16
3.5. FILE TRANSFER	16
3.5.1. <i>Linux users</i>	16
3.5.2. <i>Windows users</i>	17
4. COMPILE AND LINK.....	19
4.1. ENVIRONMENT MODULES.....	19
4.2. INTEL COMPILER.....	20
4.2.1 <i>Loading compiler environment</i>	20
4.2.2 <i>Serial program</i>	20
4.2.3 <i>Thread parallel program</i>	20
4.2.4 <i>MPI parallel program</i>	20
4.3. PGI COMPILER	21
4.3.1 <i>Loading compiler environment</i>	21
4.3.2 <i>Serial program</i>	21
4.3.3 <i>Thread parallel program</i>	22
4.3.4 <i>MPI parallel program</i>	22
5. PBS PRO JOB OPERATION.....	23
5.1 JOB QUEUE	23
5.2 QUEUE LIST	24
5.3 JOB SUBMISSION.....	24
5.3.1 <i>PBS job script</i>	24
5.3.2 <i>Batch job submission</i>	26
5.3.3 <i>Array job (Bulk job) submission</i>	26

5.3.4	<i>Specifying E-mail Notification in a Job script</i>	27
5.4	DELETING A JOB	28
5.5	DISPLAYING JOB STATUS	28

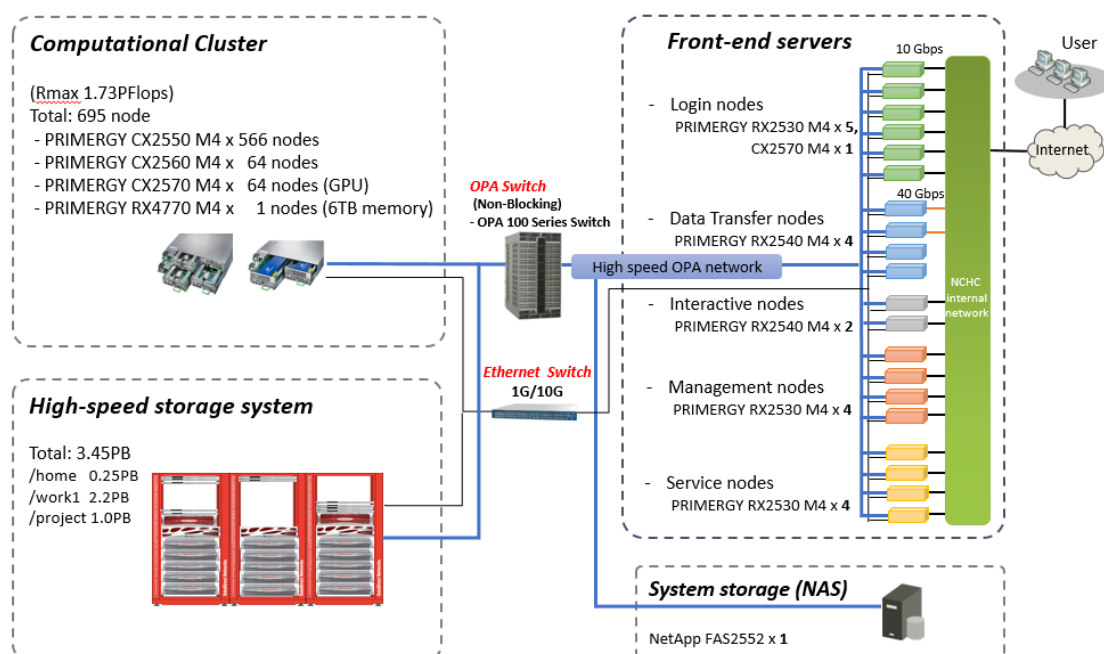
1. Introduction

In this User's Guide, we explain usage of the Peta HPC System installed at the National Center for High Performance Computing (NCHC). Please read this document carefully before using the system and get the latest version of the manual.

2. Peta HPC system

2.1. System overview

An overview of the general architecture of Peta HPC system is shown as below:



This HPC system consists of mainly 4 components:

1. Computational cluster
2. Front-end servers
3. High-speed storage system
4. System storage (NAS)

All these components are connected via Ethernet network and Intel Omni-Path high-speed network.

2.2. Available compute resources

There are 690 compute nodes (1380 processors and 27600 cores) used in Peta HPC system which in total delivers peak compute performance of approximately 1.3 PFLOPS. 690 compute nodes consist of dual CPU sockets, each socket comprises Xeon Gold 6148 CPU (20 cores, 2.4 GHz).

These 690 nodes can be categorized as below:

- **Thin Nodes**
 - For majority of HPC application
- **Fat nodes**
 - For HPC application which needs large memory

The summary of compute nodes and their respective resources are listed below:

Node Type	Node range	Total units (nodes)	Compute resources per unit (node)					
			CPU Sockets	CPU cores	Memory (GB)	Tesla P100	10Gbps interface	480 GB SSD
Thin nodes	cn0101 – cn0673	438	2	40	192	-	-	-
Thin nodes	cn0701 – cn0764	64	2	40	192	-	1	-
Fat nodes	cn0801 – cn0864	64	2	40	384	-	-	-
Fat nodes	cn0901 – cn0964	64	2	40	384	-	-	1
Fat nodes	cn1201 – cn1260	60	2	40	384	-	-	1

2.3. Available storage resource

The following storage resources on high-speed storage system are available to users in this HPC system. They are mounted as lustre file system which is accessible from all the front end servers as well as from the compute nodes via the high-speed OPA network.

	Storage area	Mount point	Capacity
1	Home area	/home	0.25 PB
2	Temporary work area	/work1	2.2 PB
3	Project storage area	/project	1.0 PB

2.3.1. Home area

The total capacity of **0.25 PB** of home area is used by system users to store their private files. Users can compile their program and execute/manage their jobs in this home area. All users by default have 100GB of quota in /home. **Data under /home will be deleted only when our user officially sends us data removal request mail.**

2.3.2. Temporary work area

This area has a total capacity of **2.2 PB** usable storage area. This area is primarily used for storing the active data of running jobs on the compute cluster.

All account holders by default have a /work1 disk space quota of 1.5 TB. This space on this clusters is intended for computing work and not for long term storage. In order to keep /work1 in a stable and efficient

status, we will begin to regularly apply automated purge policy. **There is no system backup for data in /work1, it is the user's responsibility to back up data.** We cannot recover any data in /work1, including files lost to system crashes or hardware failure so it is important to make copies of your important data regularly. **All inactive files that have not been written to or read within the last 28 days will be removed.** We strongly urge users to regularly clean up their data in /work1 to decrease /work1 usage and to back up files you need to retain

e.g. data can be copied in a simple way from /work1 to /home or /project, using cp command as below.

```
[user@clogin1]$ cp /work1/<path to target file> /project/<destination path>
```

Some major options used with cp commands are:

- p Preserves modification times, access times, and modes from the original file.
- r Recursively copy entire directories.

2.3.3. Project storage area

To be announced (TBA).

2.4. Front-end servers for user access

Allow access from Taiwan IP addresses only.

2.4.1. Login nodes

140.110.148.11 clogin1.twnia.nchc.org.tw

140.110.148.12 clogin2.twnia.nchc.org.tw

There are three login nodes which are the primary access point for command line usage of the HPC service. Users access to the login nodes via each nodes' IP address. All login nodes are identically configured and no user data is kept on their disks. Users access their files on high-speed storage system which is mounted on each login node. Therefore it does not matter which login node a user is connected through to.

From login nodes, users are able to perform the following tasks:

- Submit/manage HPC jobs.
- Have full access to files resident on high-speed storage system.
- Compile HPC application.
- Run debugger for code development.

The login nodes have similar technical specification with compute nodes, and this is the reason why they provide complete compatibility for development and testing of application codes.

The summary of compute nodes and their respective resources are listed below:

Node Type	Node range	Total units (nodes)	Compute resources per unit (node)				
			CPU Sockets	CPU cores	Memory (GB)	Tesla P100	480 GB SSD
CPU login	clogin1– clogin2	2	2	40	384	-	1

nodes							
-------	--	--	--	--	--	--	--

Do NOT use the login nodes for computation. If everyone does this, the login nodes will crash keeping other users from being able to login to this cluster.

2.4.2. Interactive nodes

To be announced (TBA).

2.4.3. Data transfer nodes

140.110.148.21 xdata1.twnia.nchc.org.tw

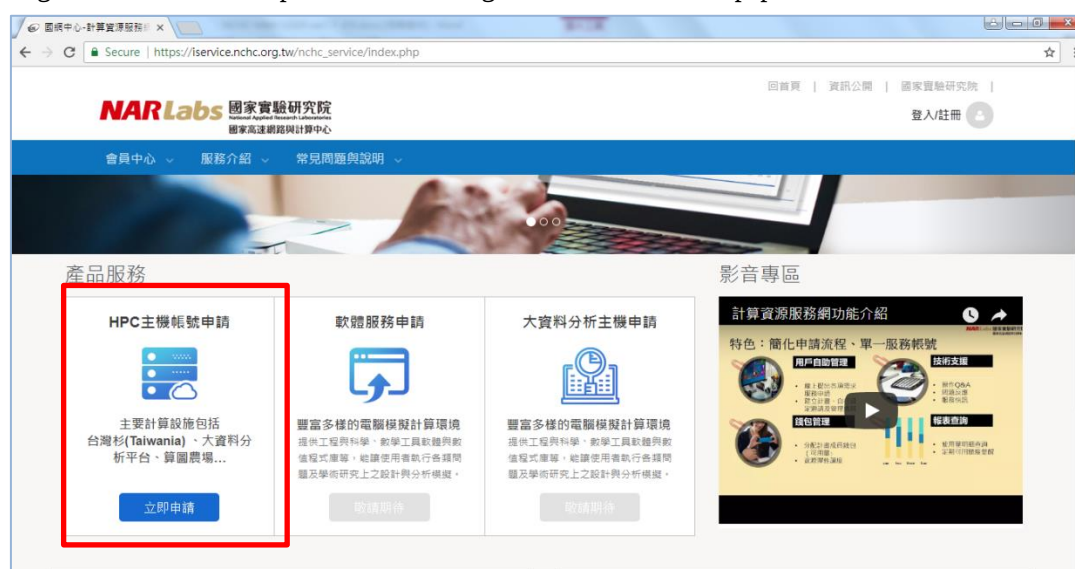
140.110.148.22 xdata2.twnia.nchc.org.tw

There are two data transfer nodes which are configured for transferring data from the external network in/out of the HPC system. Each node consists of a 40Gbps HCA card connected to external network, and an OPA interface connected to high-speed storage system as well as other nodes. By using this configuration, data can be rapidly transferred from/to the high-speed storage to/from users. For this purpose, users are allowed only scp/sftp access and cannot login to the HPC system via this nodes.

3. System access method

3.1. Member account & Peta system account registration

1. Registration website : https://iservice.nchc.org.tw/nchc_service/index.php



2. Member account apply now

A screenshot of the LOGIN form on the NAR Labs website. The form is titled 'LOGIN' and includes a close button (X). It contains the following elements: a link '[還不是會員? 現在就加入會員] [重發認證信] [忘記密碼]', a label '會員帳號(電子郵件地址)', a text input field with placeholder '請輸入您的電子郵件地址', a label '密碼', a text input field with placeholder '密碼', a checkbox labeled '我不是機器人' next to a reCAPTCHA logo, a blue '登入' (Login) button, and a '快速登入' (Quick Login) section with buttons for Facebook and Google.

3. Agreement confirmation

NARLabs 國家實驗研究院
National Applied Research Laboratories
國家高速網路與計算中心

會員中心 ▾ 服務介紹 ▾ 操作說明 ▾ 常見問題 ▾

加入會員

閱讀個資及權利義務聲明 Step 1

填寫會員基本資料 Step 2

收取認證信 Step 3

驗證成功 Step 4

版本2019/06/13 V2

新會員註冊

歡迎您加入並使用國網中心service服務網會員網站所提供之各項服務。為保障您的權益，請詳細閱讀本服務條款所有內容，尤其當您在線上點選「我同意」時，並註冊完成或開始使用本服務時，即視為您已經詳細閱讀、了解本服務條款，並同意遵守以下服務條款之約定。

一、遵守會員規範及法律規定：

您了解您註冊成為會員後，即可使用國網中心service服務網所提供之各項服務(以下稱本服務)。

當會員使用本服務時，即表示除了同意遵守本服務條款及相關法令規定(如：智慧財產權、個人資料保護法、資訊安全管理法等)之約束。

二、服務簡介：當您完成註冊程序後，擁有國網中心service服務網會員資格，亦可以開始使用本服務。

2.1. 服務內容包含：

2.1.1. 計畫申請、帳號申請、購置額度、特殊服務申請

2.1.2. 計畫管理、訂量管理、使用查詢

2.1.3. 成果管理(期刊、研討會論文、專利)

2.1.4. 資料維護、密碼重置

2.2. 若您申請成為本服務網會員，電子郵件信箱將作為您的會員帳號，請務必依據指示完成註冊流程。

2.3. 若申請資格不符，本服務網站有權終止該信箱及其他會員服務使用權利，並自資料庫中刪除申請資料，申請人不得異議。

2.4. 本服務網站有權增加、變更或取消本服務中相關系統或功能之全部或一部份之權利，且無需個別通知會員；且有關現有或將來之各項服務均受本服務條款之規範。

三、真實登錄義務：

4. Fill in your basic information - setup your member account and password

https://computeweb.nchc.org.tw/nchc_service/nchc_member_apply_3.php 90%

NARLabs 國家實驗研究院
National Applied Research Laboratories
國家高速網路與計算中心

回首頁 | 資訊公開 | 國家實驗研究院 |

會員中心 ▾ 服務介紹 ▾ 常見問題與說明 ▾

加入會員

閱讀個資及權利義務聲明 Step 1

填寫會員基本資料 Step 2

收取認證信 Step 3

驗證成功 Step 4

填寫會員基本資料

會員資料 主機帳號資料

會員帳號資料

* 請輸入您的E-mail作為會員帳號

.....@tw.fujitsu.com

因「確認證」為系統自動發出的簡信，部分信箱會誤判為垃圾信，煩請您登入您的信箱並到「垃圾信件匣」查看是否有誤判之情形。

* 會員密碼

.....

* 再次輸入會員密碼

.....

連結 Facebook 帳號登入

連結 Google 帳號登入

連結 EduRoom 帳號

說明：

1. 若已成功連結 Facebook/Google 帳號登入，可不需輸入會員密碼

2. 會員密碼長度至少需8字元，不可過於簡單

3. 會員密碼可為數字、英文字母(大小寫視為2種)、其他特殊字元等4種型式，至少須包含2種

5. Fill in your basic information - setup your Peta system account and password

6. E-mail account authentication

7. Select the [mobile authentication], dialog box to enter the SMS verification code received by the mobile phone.

The screenshot shows the NAR Labs website header with navigation links: 會員中心, 服務介紹, 常見問題與說明. Below the header is a progress bar with four steps: Step 1 (閱讀個資及權利義務聲明), Step 2 (填寫會員基本資料), Step 3 (收取認證信), and Step 4 (驗證成功). Below the progress bar, a message states: "E-Mail驗證成功", "感謝您，您申請的會員帳號E-Mail已經驗證成功。", and "請進行手機認證以開通您的帳號". A blue button labeled "手機認證" is at the bottom.

The screenshot shows the NAR Labs website header with navigation links: 會員中心, 服務介紹, 操作說明, 常見問題. Below the header, a section titled "簡訊驗證" contains a message: "簡訊已發送至您的手機: [redacted]，請於下方輸入驗證碼". Below this is a text input field. A box contains instructions: "當您收到簡訊確認碼後，請務必於10分鐘內完成認證程序，超過10分鐘後，確認碼將失效。若未收到簡訊，請按 重送簡訊。", "若有任何問題，歡迎您隨時透過以下方式與我們連絡，我們將盡快為您服務，謝謝！", and "E-mail: iservice@narlabs.org.tw 電話: 03-5776085-442 呂小姐". At the bottom are two buttons: "確認" and "重新輸入手機號碼".

8. Checking the project code by member account login

The screenshot shows the NAR Labs project management interface. At the top is a search bar with the text "請輸入計畫名稱、計畫編號" and a search icon. To the right are buttons: 新增, 排序, 計畫參與, 審核狀態, and 有效狀態. Below the search bar is a card titled "試用計畫(ISSUE)". The card contains the text: "建立者名稱: [redacted]", "計畫系統代號: [redacted]", and "計畫編號: [redacted]". Below this is a green button labeled "申請通過". To the right of the card, the text "計畫執行期間: 2018-04-19 ~ 2019-04-19" is displayed.

3.2. Command line login

Users login to the system using issued account and One-Time Password. (OTP). Confirm that you have done the following before you start to remote access to the HPC system.

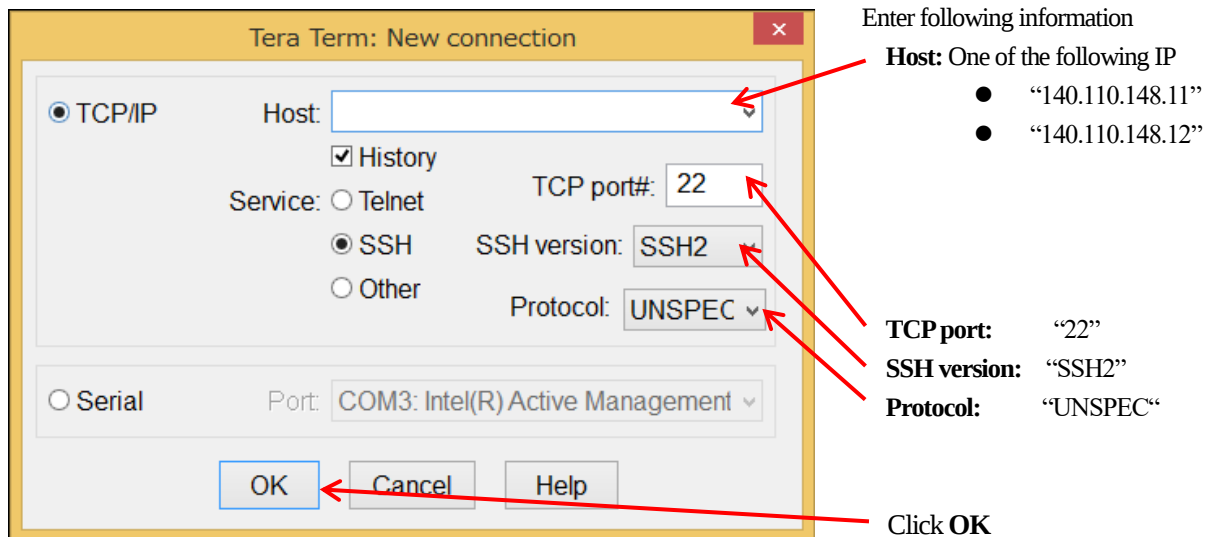
- Connect to member registration website to apply your login account and password for this system.
Note: The password you get here is not OTP. It is an account password. The combination of account password and OTP will be used for login to the system.
- Get OTP QR code from login to member registration website.
- Download “SOPHOS authenticator” or “Google authenticator” apps in your smartphone.
- Scan the generated QR code using the downloaded apps and it will start to generate OTP once in every 30 second.

Incorrect time on your smartphone and PC can cause troubles. Please make sure your device has the right time.

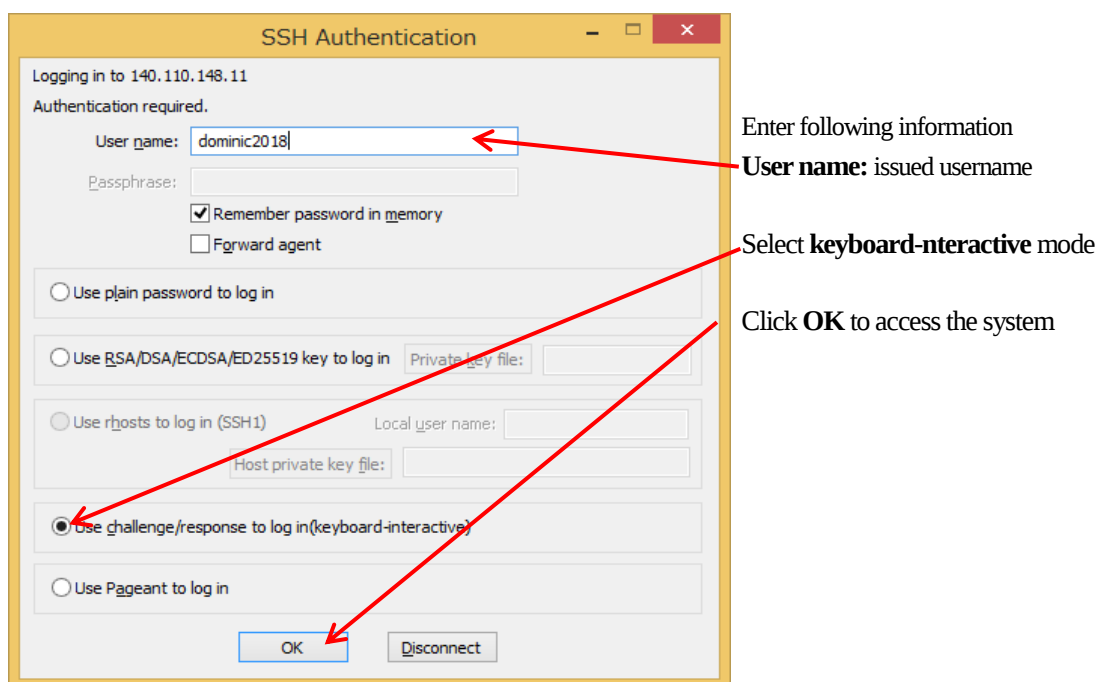
Now, you can follow the steps below to access to the system using SSH client software such as TeraTerm from your PC.

1. Open SSH client software from your PC.
2. Input the Host IP address and port number.

Note: The IP address below are directly accessible from anywhere in Taiwan. If you are outside Taiwan, you will not be able to access to the system.



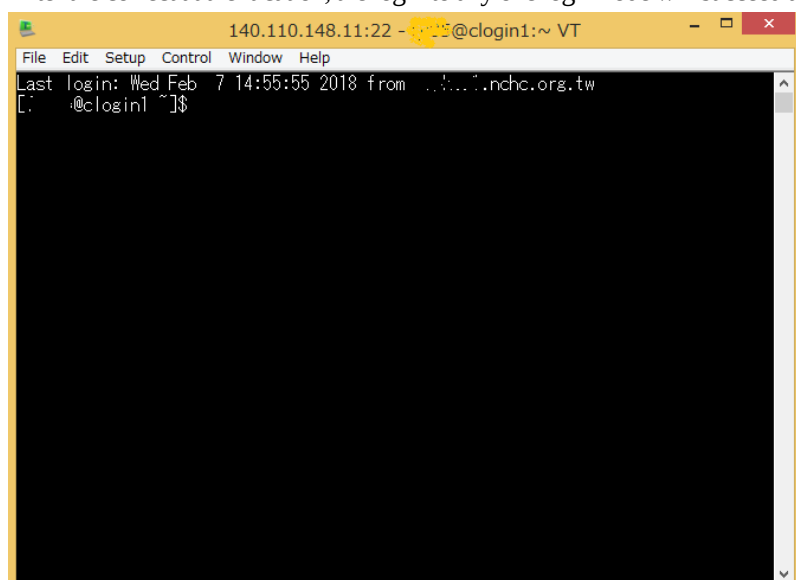
3. Enter your account name in “username” box, select “keyboard-interactive” and click OK.



- Now, you will be asked to enter password. Enter your account password followed by OTP and press Enter.
Password = Account password + OTP

On the first login to this system, a message about a key fingerprint is popped up. In this case, select “yes” and continue.

After the correct authentication, the login to any one login node will succeed as below.



3.3. Changing Password

If you need to change your account password, please go to member registration website.

The screenshot shows the NAR Labs website interface. At the top, there's a navigation bar with links like '會員中心', '服務介紹', '操作說明', and '常見問題'. The main content area is titled '簡訊驗證' (SMS Verification). It shows a message: '簡訊已發送至您的手機 776334, 請輸入驗證碼。' (SMS has been sent to your mobile phone 776334, please enter the verification code). Below this is a text input field for the verification code and a '確認' (Confirm) button. There are also instructions in Chinese about the verification process and a link to '重送簡訊' (Resend SMS). Further down, there's a section for '請輸入新的主機密碼' (Please enter the new main password) with a text input field and a '修改主機密碼' (Change main password) button. At the bottom, there's a '說明' (Notes) section with three points: 1. Password length must be at least 12 characters, not too simple. 2. Password can be numbers, English letters (case-sensitive), and other special characters, at least 4 types, and must contain at least 3 types. 3. If a single character is entered incorrectly 5 times, the account will be locked for 3 minutes, and other characters must be entered or wait 3 minutes before trying again.

3.4. Command line logout

Execute the logout or exit command.

```
[user@clogin1~]$ exit
```

3.5. File transfer

To transfer files from your PC or workstation to this system, use scp/sftp service. Linux/UNIX users use the scp or sftp command and Windows PC users use client software such as WinSCP.

3.5.1. Linux users

Use scp command and access to one of the data transfer nodes.

```
$ scp [option] <source host>:<local path of directory or file> ¥
<destination host>:<remote path of directory or file>
```

Some major options used with scp commands are:

- p Preserves modification times, access times, and modes from the original file.
- r Recursively copy entire directories.

Use sftp command and access to one of the data transfer nodes.

```
$ sftp [option] [username@]<destination host>
```

```
Connected to <destination host>.
sftp> get <remote path of directory or file>
                                     -> download file to local current directory
sftp> put <local path of directory or file>
                                     -> upload file to server current directory
sftp> bye
                                     -> quit sftp
```

Some major options used with sftp commands are:

- p Preserves modification times, access times, and modes from the original file.
- r Recursively copy entire directories.

Some major internal command used in sftp commands are:

- cd <path> Change remote directory to <path>.
- pwd Display remote working directory.
- lcd <path> Change local directory to <path>.
- lpwd Display local working directory.

3.5.2. Windows users

Start WinSCP and access to one of the data transfer node of the system. After the connection is established, you can transfer files just with drag & drop.

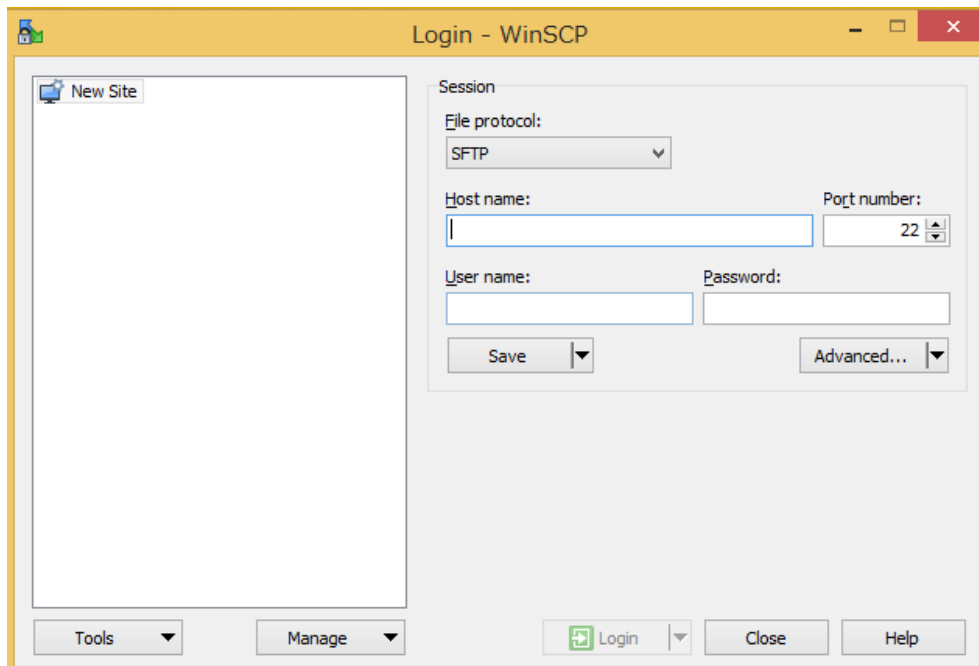
Below is the login window of WinSCP. Enter the following information and click “login” button.

Host name: “140.110.148.21” or “140.110.148.22”

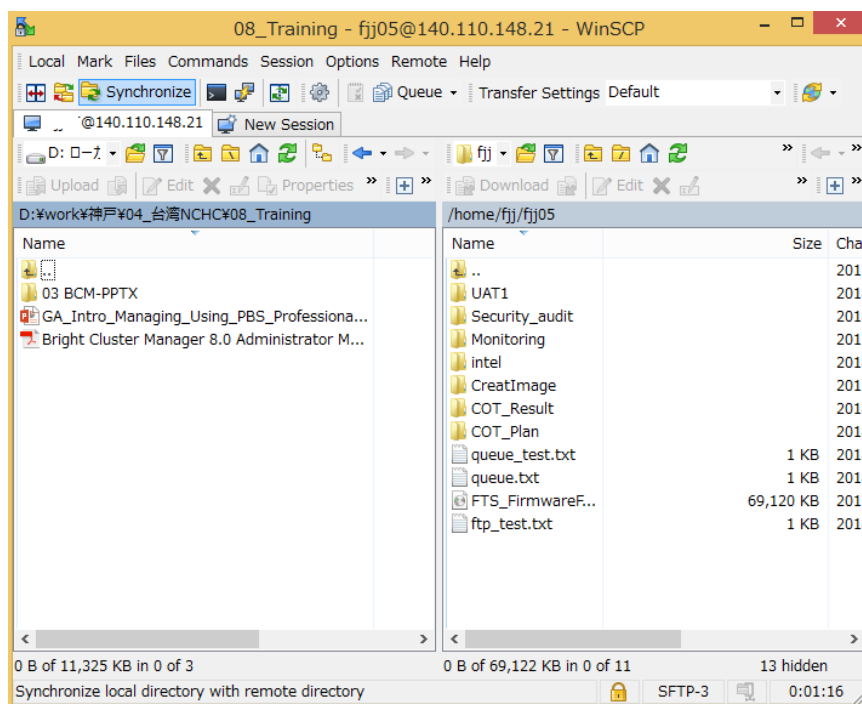
Port number: 22

User name: issued user name

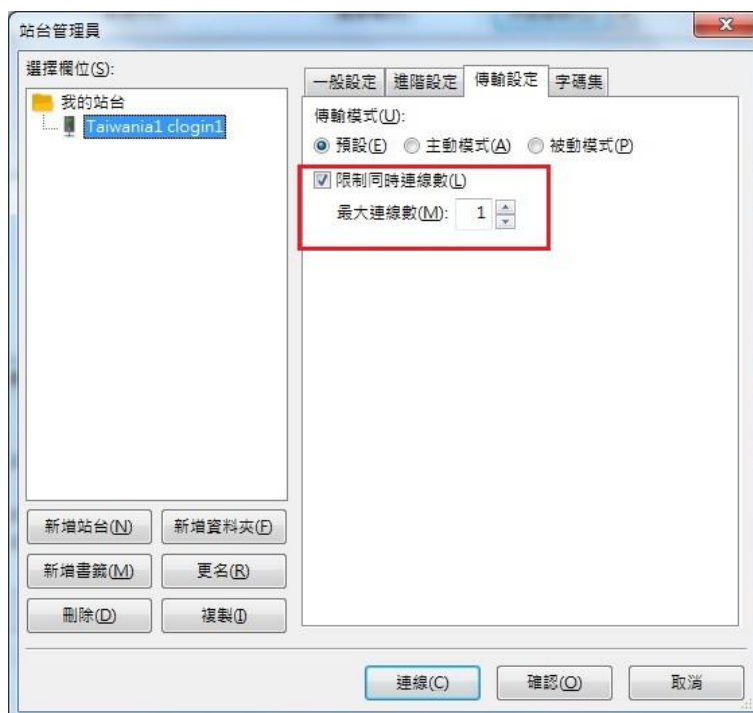
Password: issued username + one-time password



After the connection is established, WinSCP windows appears as below.



If you use FileZilla FTP client to transfer files, you have to go to Site Manager – Transfer Setting – Limit simultaneous connections to 1.



4. Compile and Link

4.1. Environment modules

The environment settings which are required for using the compiler, library, applications can be changed by using module commands.

1. Confirm the available module on login nodes

```
[user@clogin1]$ module avail
```

2. Load the module which are required for using the compiler, library, applications

```
[user@clogin1]$ module load <module name>
```

3. Add another module

```
[user@clogin1]$ module add <module name>
```

4. The following modules are available in this HPC system

Module Name	Description
blacs/openmpi/gcc/64/1.1patch03	Blacs library
blas/gcc/64/3.7.0	Basic Linear Algebra Subprograms for GNU
bonnie++/1.97.1	Bonnie++ library
fftw2/openmpi/gcc/64/double/2.1.5	FFTW library
fftw2/openmpi/gcc/64/float/2.1.5	FFTW library
fftw3/openmpi/gcc/64/3.3.6	FFTW library
gdb/7.12.1	GNU Cross Compilers
hdf5/1.10.1	Hierarchical Data Format
hwloc/1.11.6	Hardware Locality
intel/2017_u4	Intel Parallel Studio XE 2017 update 4
intel/2018_init	Intel Parallel Studio XE 2018 Initial
intel/2018_u1	Intel Parallel Studio XE 2018 update 1
iozone/3_465	File system benchmark tool
lapack/gcc/64/3.7.0	Linear Algebra package
mvapich2/gcc/64/2.2rc1	MVAPICH MPI library
netcdf/gcc/64/4.6.0	Network Common Data Form library
netperf/2.7.0	Network benchmark
petsc/openmpi/gcc/3.8.0	PETSc data structure library
pgi/17.10	PGI compilers and development tools
scalapack/openmpi/gcc/64/2.0.2	Scalable Linear Algebra Library

Note: openmpi/gcc and openmpi/pgi have conflict setting. So, you cannot load both module at a time. Similarly, intel/2017_u4, intel/2018_init and intel/2018_u1 also have conflict setting. So, you can load only one of these module at a time.

5. List all the modules currently loaded

```
[user@clogin1]$ module list
```

6. Unload the module

```
[user@clogin1]$ module unload <module name>
```

7. Unload all loaded modules

```
[user@clogin1]$ module purge
```

4.2 Intel Compiler

4.2.1 Loading compiler environment

1. Load Intel compiler environment

```
[user@clogin1]$ module load intel/2018_u1
```

Choose a module to match the version to use.

4.2.2 Serial program

1. Compile / link C program

```
[user@clogin1]$ icc -o sample.exe sample.c
```

2. Compile / link C++ program

```
[user@clogin1]$ icpc -o sample.exe sample.c
```

3. Compile / link Fortran program

```
[user@clogin1]$ ifort -o sample.exe sample.f
```

4.2.3 Thread parallel program

1. Compile / link C program

```
[user@clogin1]$ icc -qopenmp -o sample_omp.exe sample_omp.c
```

2. Compile / link C++ program

```
[user@clogin1]$ icpc -qopenmp -o sample_omp.exe sample_omp.c
```

3. Compile / link Fortran program

```
[user@clogin1]$ ifort -qopenmp -o sample_omp.exe sample_omp.f
```

4.2.4 MPI parallel program

1. Build C source code called by MPI library

```
[user@clogin1]$ mpiicc -o sample_mpi.exe sample_mpi.c
```

2. Build C++ source code called by MPI library

```
[user@clogin1]$ mpiicpc -o sample_mpi.exe sample_mpi.c
```

3. Build Fortran source code called by MPI library

```
[user@clogin1]$ mpiifort -o sample_mpi.exe sample_mpi.f
```

4. Job script example for running parallel program compiled by Intel library.

```
$ vim example01.sh
#!/bin/bash
#PBS -P TRI654321
#PBS -N sample_job
#PBS -l select=2:ncpus=40:mpiprocs=4
#PBS -l walltime=00:30:00
#PBS -q ctest
#PBS -j oe

module load intel/2018_u1
cd ${PBS_O_WORKDIR:-"."}
export I_MPI_HYDRA_BRANCH_COUNT=-1
mpirun ./sample_mpi.exe
```

5. Please export the environment value `export I\_MPI\_HYDRA\_BRANCH\_COUNT= -1`, before issue the mpirun.

4.3 PGI compiler

4.3.1 Loading compiler environment

1. Load PGI compiler environment

```
[user@clogin1]$ module load pgi/17.10
```

4.3.2 Serial program

1. Compile/link C program

```
[user@clogin1]$ pgcc -o sample.exe sample.c
```

2. Compile/link C++ program

```
[user@clogin1]$ pgc++ -o sample.exe sample.c
```

3. Compile/link Fortran program

```
[user@clogin1]$ pgfortran -o sample.exe sample.f
```

4.3.3 Thread parallel program

1. Compile/link C program

```
[user@clogin1]$ pgcc -mp -o sample_omp.exe sample_omp.c
```

2. Compile/link C++ program

```
[user@clogin1]$ pgc++ -mp -o sample_omp.exe sample_omp.c
```

3. Compile/link Fortran program

```
[user@clogin1]$ pgfortran -mp -o sample_omp.exe sample_omp.f
```

4.3.4 MPI parallel program

1. Load compiler environment

```
[user@clogin1]$ module load pgi/19.4  
[user@clogin1]$ module load mpi/openmpi-2.1.3/pgi194
```

2. Compile/link C program

```
[user@clogin1]$ mpicc -o sample_mpi.exe sample_mpi.c
```

3. Compile/link C++ program

```
[user@clogin1]$ mpic++ -o sample_mpi.exe sample_mpi.c
```

4. Compile/link Fortran program

```
[user@clogin1]$ mpifort -o sample_mpi.exe sample_mpi.f
```

5. Job script example of running parallel program compiled by PGI library.

Refer to Chapter 5 for detail about job script.

```
$ vim example01.sh  
#!/bin/bash  
#PBS -P TRI654321  
#PBS -N sample_job  
#PBS -l select=2:ncpus=40:mpiprocs=4  
#PBS -l walltime=00:30:00  
#PBS -q ctest  
#PBS -j oe  
  
module load pgi/19.4  
module load mpi/openmpi-2.1.3/pgi194  
cd $PBS_O_WORKDIR  
mpiexec -mca pml cm -mca mt1 psm2 ./sample_mpi.exe
```

5. PBS Pro job operation

5.1 Job queue

Queue name	Resource range (CPU cores)	Memory per node	Resource range (GPUs)	Resource range (SSD)	Max walltime per job	High Priority	Max running jobs per user	Max running jobs
<i>serial</i>	1 (1 node)	384GB	—	1	96:00:00		10	
<i>cf40</i>	2-40 (1 node)	384GB	—	1	96:00:00		10	120
<i>cf160</i>	2-160 (1-4 nodes)	384GB	—		96:00:00		8	160
<i>cf1200</i>	161-1200 (5-30 nodes)	384GB	—		48:00:00	V	2	5
<i>ct160</i>	2-160 (1-4 nodes)	192GB	—		96:00:00		4	60
<i>ct400</i>	161-400 (5-10 nodes)	192GB	—		96:00:00		3	22
<i>ct800</i>	401-800 (11-20 nodes)	192GB	—		72:00:00		2	10
<i>ct2k</i>	801-2000 (21-50 nodes)	192GB	—		48:00:00	V	2	4
<i>ct6k</i>	2001-6000 (51-150 nodes)	192GB	—		24:00:00	V	1	2
<i>ctest</i>	1-80 (1-2 nodes)	192GB	—		00:30:00		10	
<i>ct_ind</i>	2-400	192GB	—		72:00:00	V		
<i>cf_ind</i>	2-160	384GB	—		72:00:00	V		

1. Users on this system can submit 50 jobs and use 6000 CPU cores at most; however, each queue has its own limitation of maximum number of running jobs. Your jobs will be placed waiting in the queue when you reach the limit.
2. Some single node jobs such as Gaussian will benefit from using local SSD partition on a compute node as job-specific directory. Both *cf40* and *serial* queues have a local SSD resource

(max size 400GB). You can request this storage space by adding `lscratch=?gb` in select statement and access this temporary directory via the environment variable `$PBS_JOBDIR` in job script.

- Both `ct_ind` and `cf_ind` are express queues and we demand extra charge for using them. After applying these two queues, your jobs will start sooner than others.

5.2 Queue List

```
$ qstat -Q
```

Queue	Max	Tot	Ena	Str	Que	Run	Hld	Wat	Trn	Ext	Type
serial	0	0	yes	yes	0	0	0	0	0	0	Exec
cf40	0	0	yes	yes	0	0	0	0	0	0	Exec
cf160	0	0	yes	yes	0	0	0	0	0	0	Exec
cf1200	0	0	yes	yes	0	0	0	0	0	0	Exec
ct160	0	0	yes	yes	0	0	0	0	0	0	Exec
ct400	0	0	yes	yes	0	0	0	0	0	0	Exec
ct800	0	0	yes	yes	0	0	0	0	0	0	Exec
ct2k	0	0	yes	yes	0	0	0	0	0	0	Exec
ct6k	0	0	yes	yes	0	0	0	0	0	0	Exec
ctest	0	0	yes	yes	0	0	0	0	0	0	Exec

5.3 Job submission

Before job submission, please make sure your Project ID (project name) has positive balance.

```
$ get_su_balance
499023, TRI107693 試用計畫 (ISSUE)
$ get_su_balance TRI107688
-150
$ qsub testjob.sh
qsub: No balance available for the User
```

5.3.1 PBS job script

A PBS job script consists of the following three components.

1. Shell specification
2. PBS directives
3. Programs or commands

e.g.

```
#!/bin/bash                                -> Shell specification
#PBS -l walltime=1:00:00
```

```
#PBS -l select=2:ncpus=16:mpiprocs=16
#PBS -N sample_job
#PBS -q ctest
#PBS -P TRI654321
#PBS -j oe

cd $PBS_O_WORKDIR

module load intel/2018_u1
NODE=`cat $PBS_NODEFILE | wc`

mpirun ./myprogram
```

-> PBS directives

-> Programs or commands

1. Shell specification:

The following line is added in the first line of a job script as shell specification.

```
#!/bin/bash
```

2. PBS directives:

By specifying the PBS directives in a job script, users can set the job property.

Format:

```
#PBS -l <resource name>=<value>      -> specify resources
#PBS -N <job name>                    -> specify job name (optional)
#PBS -q <destination queue>           -> specify the queue
#PBS -P <project name>                 -> specify project name
#PBS -j eo                             -> merge std-err and std-out (optional)
```

e.g.

```
#PBS -l select=1:ncpus=1                -> sequential job (1 core)
#PBS -l select=2:ncpus=8:mpiprocs=8     -> MPI job (2 nodes and 8 proc per node)
#PBS -l select=2:ncpus=8:mpiprocs=1:ompthreads=8
                                           -> MPI/OpenMP Hybrid job (2 MPI and 16 threads)
#PBS -l walltime=1:00:00                -> processing wall time is one hour
```

Note: You have to specify correct resource (**ncpus <= 40 per node**) limits for your job.

3. Program and command

The syntax of a job script is generally same as the syntax of a shell script.

```
cd $PBS_O_WORKDIR

module load intel/2018_u1
NODE=`cat $PBS_NODEFILE | wc`
```

```
mpirun ./myprogram
```

5.3.2 Batch job submission

PBS provides “qsub” command for submitting jobs. Batch jobs can be submitted by (a) job script, or pure (b) command line.

Format:

```
$ qsub <name of job script>
```

(a) Job script

1. Create the job script file.

```
$ vim example01.sh

#!/bin/bash

#PBS -P TRI107693
#PBS -N sample_job
#PBS -l select=2:ncpus=40:mpiprocs=40
#PBS -l walltime=00:30:00
#PBS -q ctest
#PBS -o jobresult.out
#PBS -e jobresult.err

module load intel/2018_u1
cd ${PBS_O_WORKDIR:-"."}

mpirun ./myprogram
```

2. Submit the job.

```
$ qsub example01.sh
```

(b) Command line

Users can specify the PBS directives from command line instead of specifying in a job script.

e.g.

```
$ qsub -l select=1:ncpus=1 -q ctest -P TRI654321 -j oe ./example01.sh
```

5.3.3 Array job (Bulk job) submission

Array is a feature of PBS which allows you to submit a series of jobs using a single submission command described by a single submission script. This feature is used when a large number of identical jobs which has similar inputs and outputs are to be submitted. To submit an array job, use -J option with “qsub” command.

Any normal jobs can be used in array job. Below is an example of a normal job which can be used in array job submission. There is no specific array variables to be used inside job script.

```
$ vim hello_mpi_1.sh

mpirun -np 80 /home/user/array/hello_mpi.exe
```

```

$ vim hello_mpi_2.sh

mpirun -np 80 /home/user/array/hello_mpi.exe

$ vim hello_mpi_3.sh

mpirun -np 80 /home/user/array/hello_mpi.exe


$ vim array.sh

#!/bin/bash

#PBS -l walltime=00:01:00

#PBS -l select=2:ncpus=4:mpiprocs=4

#PBS -N hello-mpi-array-job

#PBS -q ctest

#PBS -P TRI654321

#PBS -J 1-3

#PBS -j oe

echo "Main script: index " $PBS_ARRAY_INDEX

/home/user/array/hello_mpi_${PBS_ARRAY_INDEX}.sh

$ qsub array.sh

```

The command below submits an array job whose sub-jobs are indexed from 1 to 100. It is similar to execute qsub command 100 times without -J option

```
$ qsub -J 1-100 example.sh
```

The command below submits an array job whose sub-jobs are indexed from 100 to 200 with step 2. i.e. 100.102.104 etc.

```
$ qsub -J 100-200:2 example.sh
```

5.3.4 Specifying E-mail Notification in a Job script

For each job, PBS can send email to designated recipients when that job reaches specific points in its lifecycle. There are two steps to realize this PBS functionality.

1. Use “-M” (capital M) option to set E-mail recipients in PBS directives as below.

```
#PBS -M user@example.com
```

2. Use “-m” (small m) option to specify mail point argument as below

```
#PBS -m be
```

Some of the important mail point arguments are listed below:

Mail point argument	Description
a	Send E-mail when job or subjob is aborted by batch system
b	Send E-mail when job or subjob begins execution
e	Send E-mail when job or subjob ends execution

n	Do not send E-mail
---	--------------------

An example of specifying E-mail notification in a job script is as below

```
$ vim example01.sh
#!/bin/bash
#PBS -P TRI654321
#PBS -N sample_job
#PBS -l select=2:ncpus=40:mpiprocs=40
#PBS -l walltime=00:30:00
#PBS -q ctest
#PBS -j oe
#PBS -M user@example.com
#PBS -m be

module load intel/2018_u1
cd ${PBS_O_WORKDIR:-"."}

mpirun ./myprogram
```

5.4 Deleting a job

PBS provides “qdel” command for deleting jobs. Users can delete only your own job.

Format:

```
$ qdel <job ID>
```

e.g.

```
$ qdel 51
$ qdel 1234[] .server
```

job ID can be confirmed by “qstat” command.

Users can forcefully delete an unfinished job using “-W force” option with qdel command.

```
$ qdel -W force <job ID>
```

5.5 Displaying job status

The “qstat” command is for watching the job status. There are 3 types in the statuses which are on S column.

(a) Job status: A job queued in queue “ctest”

```
$ qstat -u user01
```

Job id	Name	User	Time Use	S	Queue

12.localhost	example01	user01	0	Q	ctest
--------------	-----------	--------	---	---	-------

(b) Job status: Running

```
$ qstat -u user01
```

Job id	Name	User	Time Use	S	Queue

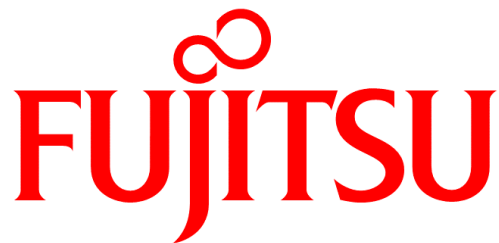
12.localhost	example01	user01	0	R	ctest

(c) Job status: Completed

```
$ qstat -u user01
```

Job id	Name	User	Time Use	S	Queue

12.localhost	example01	user01	00:00:55	C	ctest



shaping tomorrow with you