Required Software

Clusters are accessed via the secure shell programs

ssh used to run commands (secure shell)
scp used to copy files (secure copy)
sftp alternative to copy files (secure file transfer protocol)

from anywhere on the internet using the address `CLUSTER.computecanada.ca` (capitalized terms indicate you should substitute something appropriate).

Windows

Most common options are the GUI

- mobaxterm is a good freeware GUI choice that provides everything in one package including a graphical sftp, X11 server, and local shell
- PuTTY is a basic open-source ssh-only option

Windows 10 has native versions of the command-line tools too

- Windows subsystem for Linux provides full linux text environment including secure shell
- OpenSSH installable component provides secure shell inside Windows command or power shell

MacOS X and Linux

Command-line versions are builtin and available in the shell

- search for and start a *terminal* application

Accessing

Logging in

`ssh USER@CLUSTER.computecanada.ca` login to CLUSTER as USER and start a remote shell session

- for GUI ones (e.g., mobaxterm, putty, etc.), enter `USER` into username field, `CLUSTER.computecanada.ca` into host name field, and connect

Copying files

`scp USER@CLUSTER.computecanada.ca:REMOTE LOCAL` login to CLUSTER as USER and copy REMOTE file to LOCAL file
`scp LOCAL USER@CLUSTER.computecanada.ca:REMOTE` login to CLUSTER as USER and copy LOCAL file to REMOTE file

- For GUI ones (e.g., mobaxterm, etc.) enter `USER` and `CLUSTER`, connect, and then drag and drop files to transfer

Shell

Commands, entered interactively or ran from a file, tell computer what to do

- a bit of an initial learning curve, but
- much easier to automate, document, and share

Basic command

`PROGRAM ARGUMENTS...`

- arguments usually options (switches) followed by strings (e.g., file names, etc.)
- short options are a single dashes followed by letter for each switch (e.g., `rm -fr mydir`)
- long options are a double dash followed by a descriptive string (e.g., `rm --force --recurse mydir`)

Help

`help COMMAND` show help for builtin command (like "cd")
`man COMMAND` show the manual page for command ('q' to quit)

Getting around

`ls` list files
`pwd` what folders you are in
`cd DIRECTORY` change to directory named directory (use ".." for previous one)

Copying/renaming/moving/removing files and directories

`cp SRC DEST` copy file/directory (add `-r` for directory)
`mv SRC DEST` move/rename file/directory
`rm PATH` remove file (add `-r` for directory)

Editing files

`nano PATH` edit file named file

Compute Canada software (scientific packages)

`module load CcEnv StdEnv` enable Compute Canada software stack on VDI systems
`module list` show what software is load environment
`module avail` show what software you can load

`module spider PACKAGE` search for package
`module load PACKAGE` load package
`module unload PACKAGE` unload the package

Software via Nix (utilities)

`module load nix` enable personal Nix software environment
`nix search PACKAGE` search for PACKAGE
`nix run ATTRIBUTES ...` start sub-shell with ATTRIBUTES available‘
`nix-env -q` list software installed in personal environment
`nix-env -iA ATTRIBUTE` install ATTRIBUTE into personal environment
`nix-env -e NAME` remove NAME from personal environment
`nix-env --rollback` undo change to personal environment

Running/submitting, viewing, and killing jobs

`salloc` start interactive job
`sbatch SCRIPT` queue batch job
`squeue` list jobs
`skill JOBID` kill job

Filesystems

Storage Folder	Group	Expiration	HPC	Size	Files
/home/USER	No	No	No	50GB	500K
/scratch/USER	No	60 days	Yes	20TB	1,000K
/project/PROJECT	Yes	No	Yes	1TB	500K

- quick `scratch`, `project` (default), and `projects` (all) links under `home`

Jobs

Check documentation page for desired software on docs.computecanada.ca

- program specifc instructions
- licensing details for commerical software

All significant computations (> few minutes) run as SLURM jobs

- only way to access the 1,000+ compute nodes
- initial interative type-in-commands test runs with `salloc`
- final non-interactive put-commands-in-file runs with `sbatch`

Batch file for `sbatch` have the general format

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

```
#SBATCH OPTION1
#SBATCH OPTION2
...

COMMANDS TO RUN
...
```

Job options

Specify the resources and accounting required to run a job

- as options to `salloc` and `sbatch` commands, and/or
- in start of job files using special `#SBATCH ...` lines

Basic options

`--account def-SPONSOR` account to allocate usage to
`--time D-HH:MM:SS` maximum runtime before killing (add a bit of buffer)
`--mem-per-cpu N` memory per CPU (suffix with `K`, `M`, `G`, or `T` for units)

Multi-threaded jobs

- limited to maximum number of CPUs in one computer (generally 32)
- may need `export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK` in job script

`--cpus-per-task N` number of threads per process (threaded)

Multi-processor jobs (MPI)

- need to start the program with the `mpirun` or `srun` commands

`--ntasks` number of processes (MPI)

GPU jobs

- `TYPE` is cluster dependent (currently `v100`,`v100l`,`p100`,`t4`,`k20`,`k80`,`a100`)

`--gpus-per-task TYPE:N` number of GPUs per process (may not work)
`--grep=gpu:TYPE:N` number of GPUs per node

Breakpoints

`partition_stats` show paritioning of compute nodes eligible for jobs

Exceeding these resources decreases the number of machines that will run a job

- runtime $>$ 3, 12, 24, 72, 168, and 672hrs
- memory/cpu $>$ 12MB