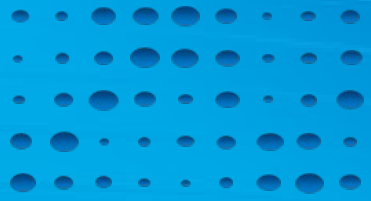


LINKSCEEM

Linking Scientific Computing in Europe and the Eastern Mediterranean

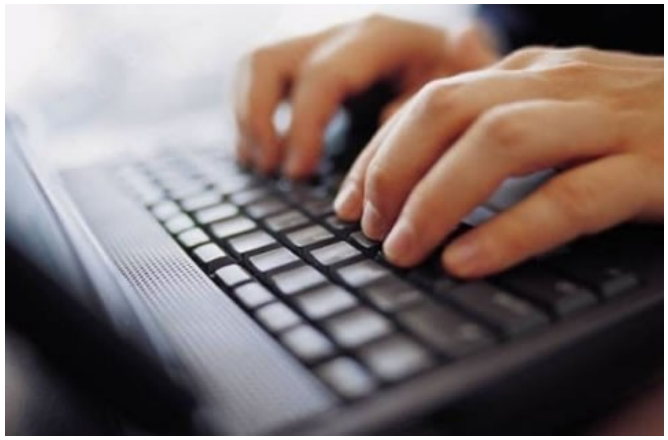
HPC Roadshow

Hands on training session for core skills

LINKSCEEM

Overview

- Obtaining an account
- How to Generate a Public/Private Key Pair
- How to Access the System (ssh)
- Software Environment – Modules
- How to Find Cluster Status and Job Summary
- Compiling and Submitting Jobs - Examples



Obtaining an Account

- **To gain access to one of the LinkSCEEM HPC resources after your application has been accepted:**
  - Print, complete, sign and fax an Acceptable Usage Policy (AUP) for the resource you have been granted access to
  - Can be found in the following link
  - As “Project Reference code” put the project code (eg. lspre100s1) corresponding to your linklings submission. For the educational access projects as "Projects Reference code" type "Educational Access".
  - Request a user account on the allocated HPC resource(s) by completing the Access Issues form
  - You will then be contacted by the user support team to provide your public key

LINKSCEEM

How to generate a Public/Private key pair

Mac OS X/Linux/Unix Keypairs

- **Create a public and private key pair:**

```
ssh-keygen -t rsa
```

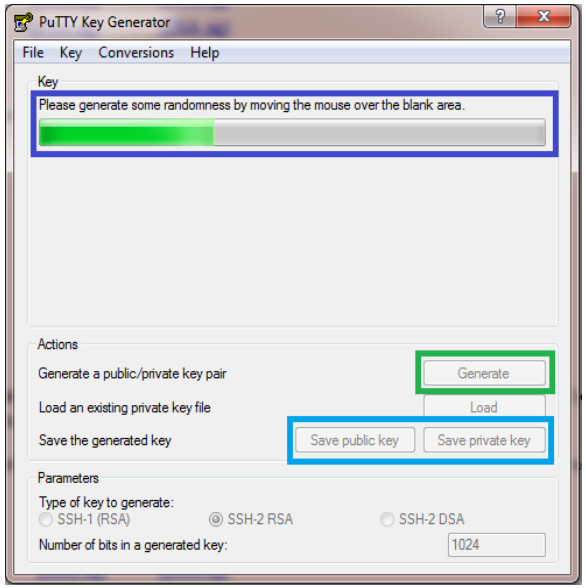
- **You will prompted for a filename for saving the private key. You should press Enter to use the default name**

```
/home/username/.ssh/id_rsa
```

- **You will prompted for a password to protect your private key which you will also confirm**
  - ALWAYS use such a password to protect your private key
- **ssh-keygen will then create a private and public key**
  - The public key will have a .pub extension (e.g., id_rsa.pub)

How to generate a Public/Private key pair

- **Windows**
  - Using Putty
- **You will need to have PuTTy and PuTTygen installed**
  - Download here
- **Run PuTTygen**
  - Click on Generate
  - Run your mouse over the blank area to generate some randomness
  - Save your key pair when done



Important Private/Public Key Notes

- **Always attach a password to your ssh keys**
  - Please note that this is the password which allows access to use the keys
  - To access the HPC resource from any machine you need access to your keys
    - Best to have these on your usb drive
  - ssh –i /media/usbName/keyName (path if in a folder)
    - Will enable you to select the private key you want to use
    - You can thus bypass the default key on the machine you are using
    - By selecting the private key stored on your usb you can access HPC resources from any machine available to you
  - With Putty you just select the key using Browse

How to access the system

- **To access the system you must have access to your private key**
- **With Mac OS X/Linux/Unix:**
  - This should be found in your .ssh folder in your home directory

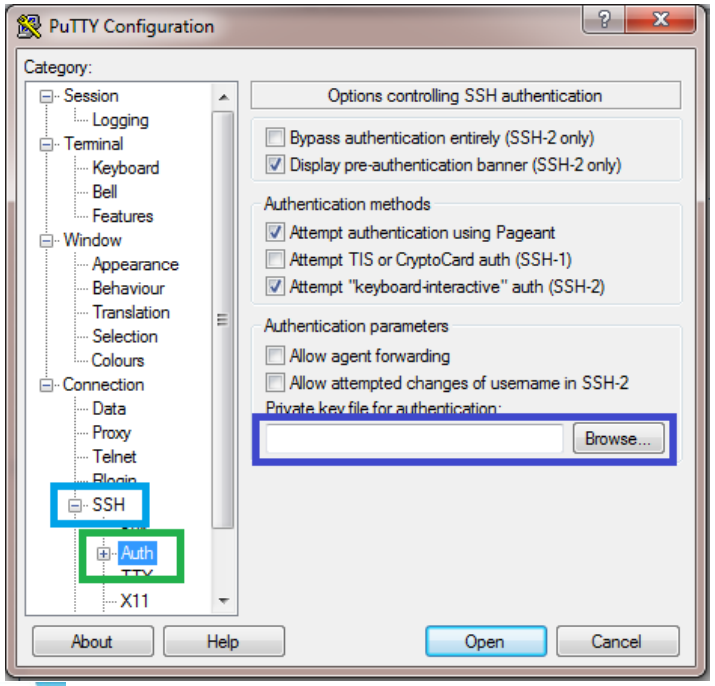
```
ssh username@euclid.cyi.ac.cy
```

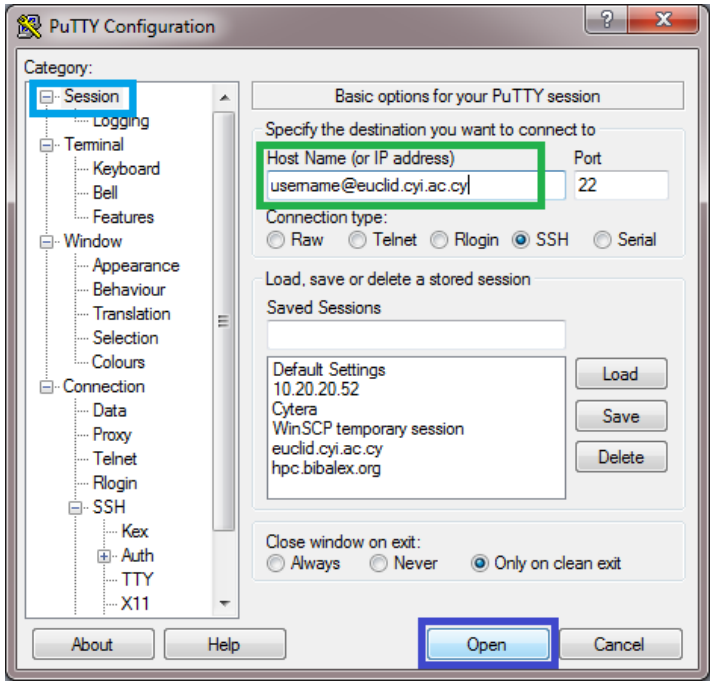
  - Should you encounter a problem with your folder permissions, do the following:

```
chmod 600 .ssh/id_rsa
```

How to access the system

- **With Windows**
  - Use PuTTy
  - First browse for and select your private key file
  - Enter ssh login information



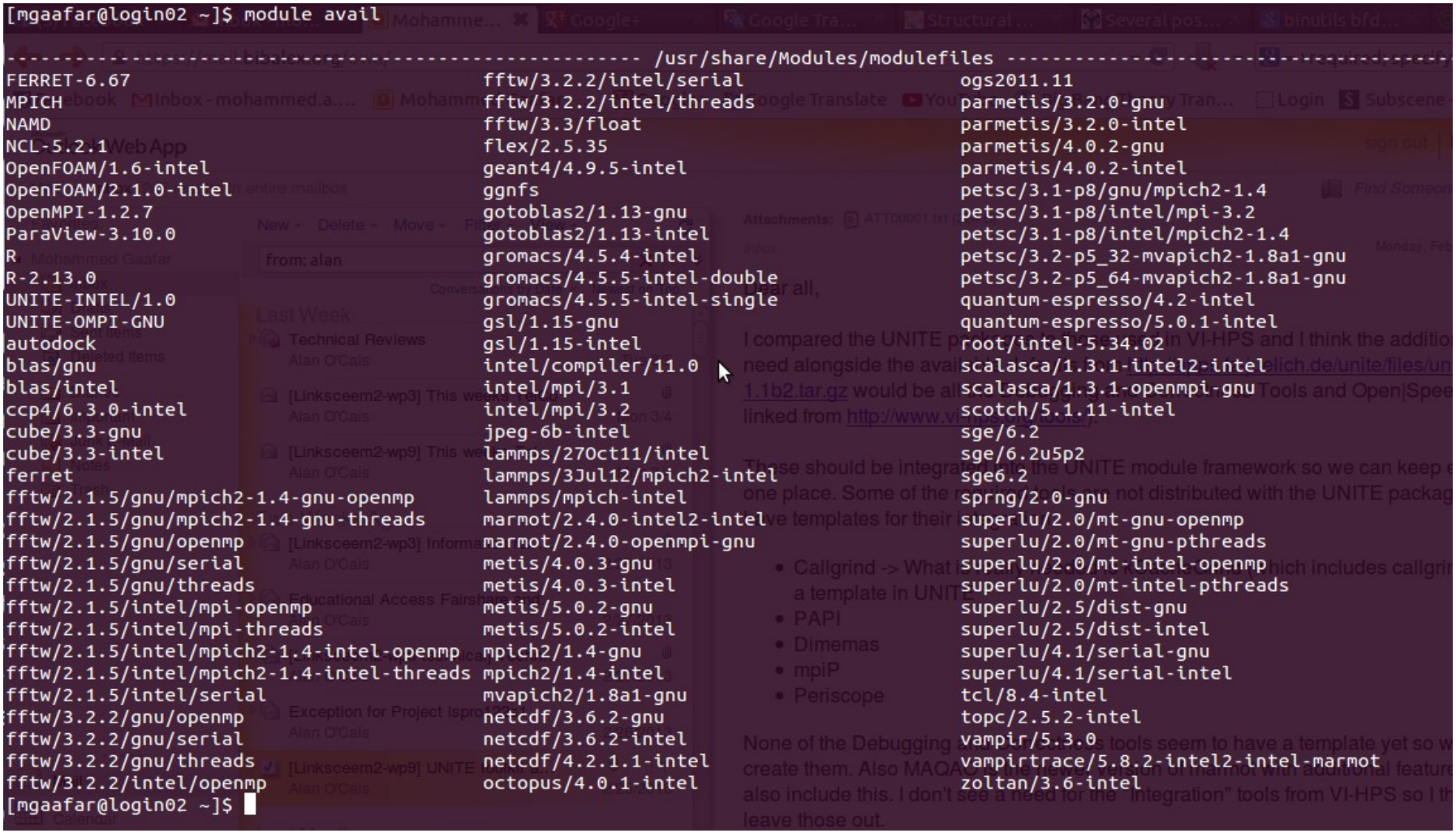


Environment Setup with Modules

- The software environment used on LinkSCEEM systems can be managed via Modules
- Modules facilitate the task of updating applications and provide a user-controllable mechanism for accessing software revisions and controlling combination of versions

```
1 module avail   # lists available modules
2 module list    # lists currently loaded modules
3 module help    # help on a specific module
4 module whatis # brief description of a specific module
5 module display # displays changes by a given module
6 module load X # load a specific module
7 module unload # unloads a specific module
8 module clear   # unloads all modules
```

Common modules and libraries



```
[mgaafar@login02 ~]$ module avail
------------------------------------------------- /usr/share/Modules/modulefiles -------------------------------------------------
FERRET-6.67                                  fftw/3.2.2/intel/serial                      ogs2011.11
MPICH                                        fftw/3.2.2/intel/threads                     parmetis/3.2.0-gnu
NAMD                                         fftw/3.3/float                               parmetis/3.2.0-intel
NCL-5.2.1                                    flex/2.5.35                                  parmetis/4.0.2-gnu
OpenFOAM/1.6-intel                           geant4/4.9.5-intel                           parmetis/4.0.2-intel
OpenFOAM/2.1.0-intel                         ggnfs                                        petsc/3.1-p8/gnu/mpich2-1.4
OpenMPI-1.2.7                                gotoblas2/1.13-gnu                           petsc/3.1-p8/intel/mpi-3.2
ParaView-3.10.0                              gotoblas2/1.13-intel                         petsc/3.1-p8/intel/mpich2-1.4
R                                            gromacs/4.5.4-intel                          petsc/3.2-p5_32-mvapich2-1.8a1-gnu
R-2.13.0                                     gromacs/4.5.5-intel-double                   petsc/3.2-p5_64-mvapich2-1.8a1-gnu
UNITE-INTEL/1.0                              gromacs/4.5.5-intel-single                   quantum-espresso/4.2-intel
UNITE-OMPI-GNU                               gsl/1.15-gnu                                 quantum-espresso/5.0.1-intel
autodock                                     gsl/1.15-intel                               root/intel-5.34.02
blas/gnu                                     intel/compiler/11.0                          scalasca/1.3.1-intel2-intel
blas/intel                                   intel/mpi/3.1                                scalasca/1.3.1-openmpi-gnu
ccp4/6.3.0-intel                             intel/mpi/3.2                                scotch/5.1.11-intel
cube/3.3-gnu                                 jpeg-6b-intel                                sge/6.2
cube/3.3-intel                               lammps/27Oct11/intel                         sge/6.2u5p2
ferret                                       lammps/3Jul12/mpich2-intel                   sge6.2
fftw/2.1.5/gnu/mpich2-1.4-gnu-openmp         lammps/mpich-intel                           sprng/2.0-gnu
fftw/2.1.5/gnu/mpich2-1.4-gnu-threads        marmot/2.4.0-intel2-intel                    superlu/2.0/mt-gnu-openmp
fftw/2.1.5/gnu/openmp                        marmot/2.4.0-openmpi-gnu                     superlu/2.0/mt-gnu-pthreads
fftw/2.1.5/gnu/serial                        metis/4.0.3-gnu                              superlu/2.0/mt-intel-openmp
fftw/2.1.5/gnu/threads                       metis/4.0.3-intel                            superlu/2.0/mt-intel-pthreads
fftw/2.1.5/intel/mpi-openmp                  metis/5.0.2-gnu                              superlu/2.5/dist-gnu
fftw/2.1.5/intel/mpi-threads                 metis/5.0.2-intel                            superlu/2.5/dist-intel
fftw/2.1.5/intel/mpich2-1.4-intel-openmp     mpich2/1.4-gnu                               superlu/4.1/serial-gnu
fftw/2.1.5/intel/mpich2-1.4-intel-threads    mpich2/1.4-intel                             superlu/4.1/serial-intel
fftw/2.1.5/intel/serial                      mvapich2/1.8a1-gnu                           tcl/8.4-intel
fftw/3.2.2/gnu/openmp                        netcdf/3.6.2-gnu                             topc/2.5.2-intel
fftw/3.2.2/gnu/serial                        netcdf/3.6.2-intel                           vampir/5.3.0
fftw/3.2.2/gnu/threads                       netcdf/4.2.1.1-intel                         vampirtrace/5.8.2-intel2-intel-marmot
fftw/3.2.2/intel/openmp                      octopus/4.0.1-intel                          zoltan/3.6-intel
[mgaafar@login02 ~]$
```

Hands on Exercise 1

C code Example

```
#include <mpi.h>
#include <stdio.h>
#include <string.h>

int main (int argc, char **argv) {

        int rank, size, partner;
        int namelen;
        char name[MPI_MAX_PROCESSOR_NAME];

        /* initialize MPI */
        MPI_Init(&argc, &argv);

        /* how many processes */
        MPI_Comm_size(MPI_COMM_WORLD, &size);
        /* my rank */
        MPI_Comm_rank(MPI_COMM_WORLD, &rank);
        /* my name is */
        MPI_Get_processor_name(name, &namelen);

        printf("Hello world: rank %d of %d running on %s \n", rank, size, name);

        /* finalize MPI */
        MPI_Finalize();
        return 0;
}
```

Hands on Exercise 1

Batch Script Example

- Script which can be used to define all commands required to submit a job to a HPC resource

```
#$ -cwd
#$ -j     y
#$ -r     y
#$ -N    hello-mpi
#$ -o    $JOB_NAME.o$JOB_ID
#$ -pe  mpi 16
#$ -l   h_rt=00:30:00

mpirun $1 -np 16 ./hello.bin
```

Hands on Exercise 1

- **From your training account home directory (cd)**
  - Copy the training example with the following command

```
cp -r /lfs01/examples/mpi/hello ./data
```

  - Go the the coreskills directory

```
cd data/hello
```

- **Load the mpi module and compile the code**

```
module load intel/mpi/3.2
make
```

Hands on Exercise 1

- **Submit the job**

```
qsub hello.sge
```

- **Check your job status**

```
qstat
```

- **Check the contents of the output file once your job is completed**

```
cat output_file_name
```

Thank you