NHR FAU

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

High Performance Computing in a Nutshell

HPC Services, RRZE / NHR@FAU

High Performance
Computing

NHR FAU

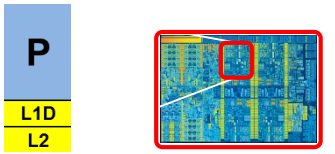
FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

HPC systems at RRZE

https://hpc.fau.de/systems-services/systems-documentation-instructions/

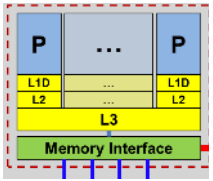
HPC High Performance Computing

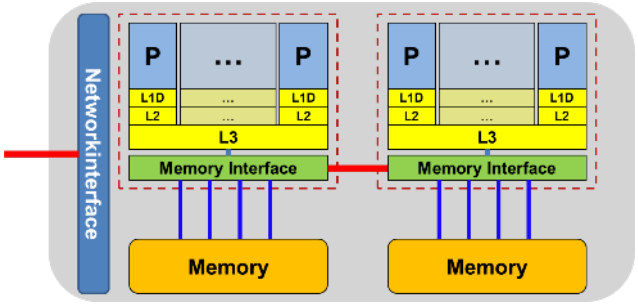
Parallel computing hardware terminology



core

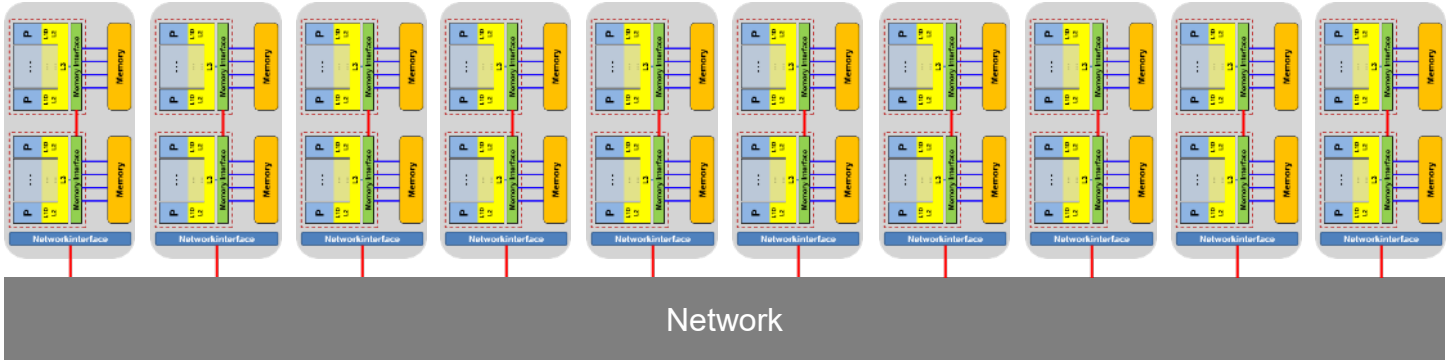
chip/socket
"CPU"





shared-memory compute node

distributed-memory cluster

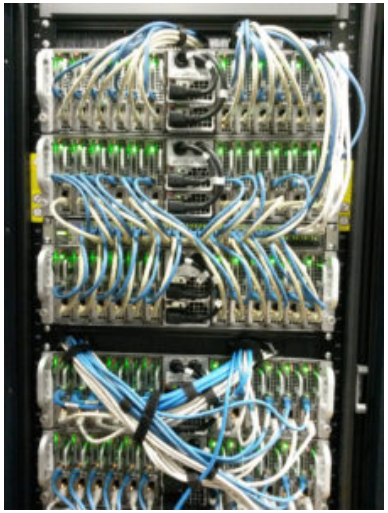


RRZE “Woody” cluster

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

main workhorse for throughput and single-node jobs

- all 246 nodes with 4 cores and high clock frequency (3.5/3.7 GHz) Intel Xeon E3-1240 v? processors
  - 70x Intel Haswell, 8 GB RAM
  - 64x Intel Skylake, 32 GB RAM
  - 112x Intel Kaby Lake, 32 GB RAM
- at least 960 GB local HDD/SSD
- and Gbit network only



RRZE “Emmy” cluster

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

main workhorse for parallel jobs

- 543 compute nodes (10.880 cores)
  - 2 Intel Xeon E5-2660v2 (Ivy Bridge) 2.2 GHz (10 cores)
  - 20 cores/node + SMT cores
  - 64 GB main memory per node
- No local disks
- Full QDR Infiniband fat tree network: up to 40 GBit/s

RRZE “Meggie” cluster

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

for scalable parallel jobs

- 728 Compute nodes (14.560 cores)
  - 2 Intel Xeon E5-2630 v4 (Broadwell) 2.2 GHz (10 cores)
  - 20 cores/node
  - 64 GB main memory
- No local disks
- Intel OmniPath network: Up to 100 Gbit/s

RRZE “TinyGPU” cluster

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

for GPU workloads – not all nodes always generally available

- 7 nodes with 2x “Broadwell” @2.2 GHz, 64 GB RAM, 980 GB SSD, 4x GTX1080
- 10 nodes with 2x “Broadwell” @2.2 GHz, 64 GB RAM, 980 GB SSD, 4x GTX1080Ti
- 12 nodes with 2x “Skylake” @ 3.2 GHz, 96 GB RAM, 1.8 TB SSD, 4x RTX 2080Ti
- 4 nodes with 2x “Skylake” @3.2 GHz, 96 GB RAM, 2.9 TB SSD, 4x Tesla V100
- 7 nodes with 2x “Cascade Lake” @2.9 GHz, 384 GB RAM, 3.8 TB SSD, 8x RTX3080
- 8 nodes with 2x AMD Rome 7662 @2.0 GHz, 512 GB RAM, 5.8 TB SSD, 4x Volta A100

Use different batch system (Torque)

What is each system good for?

Cluster	#nodes	Appl.	Parallel FS	Local HDD	Description
Meggie	728	massively parallel	Yes	No	Newest RRZE cluster, highly parallel workloads
Emmy	560	massively parallel	(Yes)	No	Current main cluster for parallel jobs
Woody	248	serial, single-node, throughput	No	Yes, some w/ SSD	High clock speed single-socket nodes for serial throughput
TinyGPU	48	GPGPU	No	Yes, all w/ SSD	Different types of Nvidia GPGPUs; Access restrictions and throttling policies may apply
TinyFat	47	Large memory	No	Yes, all w/ SSD	256-512 GB memory per node. Access restrictions may apply.

NHR FAU

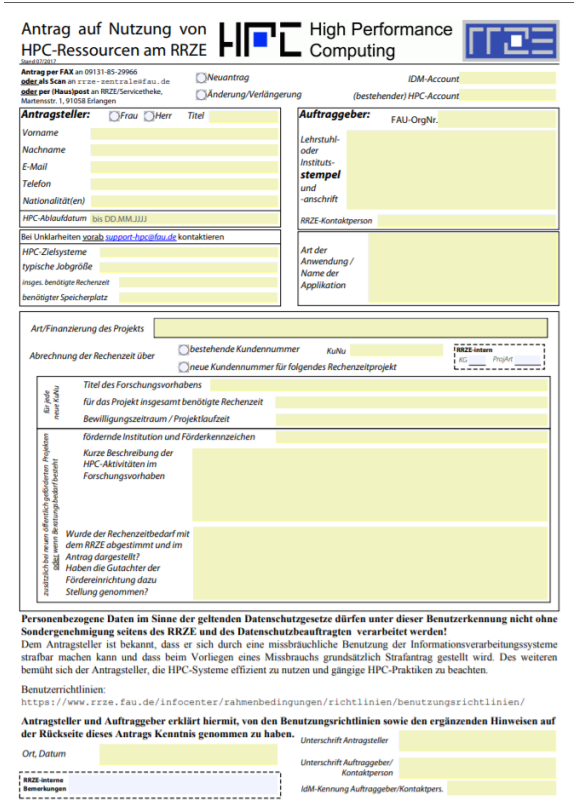
FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

Accessing HPC systems at RRZE

HPC High Performance Computing

HPC account

- You need a separate account (not your IdM account)
- HPC account application form
- Account can access all HPC systems at RRZE!
- Ask your local RRZE contact person for help
- If you change your affiliation, you need a new HPC account. Data migration may be required



Antrag auf Nutzung von HPC-Ressourcen am RRZE — High Performance Computing

Stand 03/2017

Antrag per FAX an 09131-85-29966
oder als Scan an rrze-zentrale@fau.de
oder per (Haus)post an RRZE/Servicetheke, Martensstr. 1, 91058 Erlangen

☐ Neuantrag
☐ Änderung/Verlängerung

IdM-Account
(bestehender) HPC-Account

Antragsteller: ☐ Frau ☐ Herr Titel
Vorname
Nachname
E-Mail
Telefon
Nationalität(en)
HPC-Ablaufdatum bis DD.MM.JJJJ

Bei Unklarheiten vorab support-hpc@fau.de kontaktieren

HPC-Zielsysteme
typische Jobgröße
insges. benötigte Rechenzeit
benötigter Speicherplatz

Auftraggeber: FAU-OrgNr.
Lehrstuhl- oder Instituts-**stempel** und -anschrift
RRZE-Kontaktperson

Art der Anwendung / Name der Applikation

Art/Finanzierung des Projekts
Abrechnung der Rechenzeit über
☐ bestehende Kundennummer KuNu
☐ neue Kundennummer für folgendes Rechenzeitprojekt
RRZE-intern KG ProjArt

bei jeder neuen KuNu
Titel des Forschungsvorhabens
für das Projekt insgesamt benötigte Rechenzeit
Bewilligungszeitraum / Projektlaufzeit

zusätzlich bei neuen öffentlich geförderten Projekten bitte wenn Beratungsbedarf besteht
fördernde Institution und Förderkennzeichen
Kurze Beschreibung der HPC-Aktivitäten im Forschungsvorhaben
Wurde der Rechenzeitbedarf mit dem RRZE abgestimmt und im Antrag dargestellt? Haben die Gutachter der Fördereinrichtung dazu Stellung genommen?

Personenbezogene Daten im Sinne der geltenden Datenschutzgesetze dürfen unter dieser Benutzerkennung nicht ohne Sondergenehmigung seitens des RRZE und des Datenschutzbeauftragten verarbeitet werden!
Dem Antragsteller ist bekannt, dass er sich durch eine missbräuchliche Benutzung der Informationsverarbeitungssysteme strafbar machen kann und dass beim Vorliegen eines Missbrauchs grundsätzlich Strafantrag gestellt wird. Des weiteren bemüht sich der Antragsteller, die HPC-Systeme effizient zu nutzen und gängige HPC-Praktiken zu beachten.

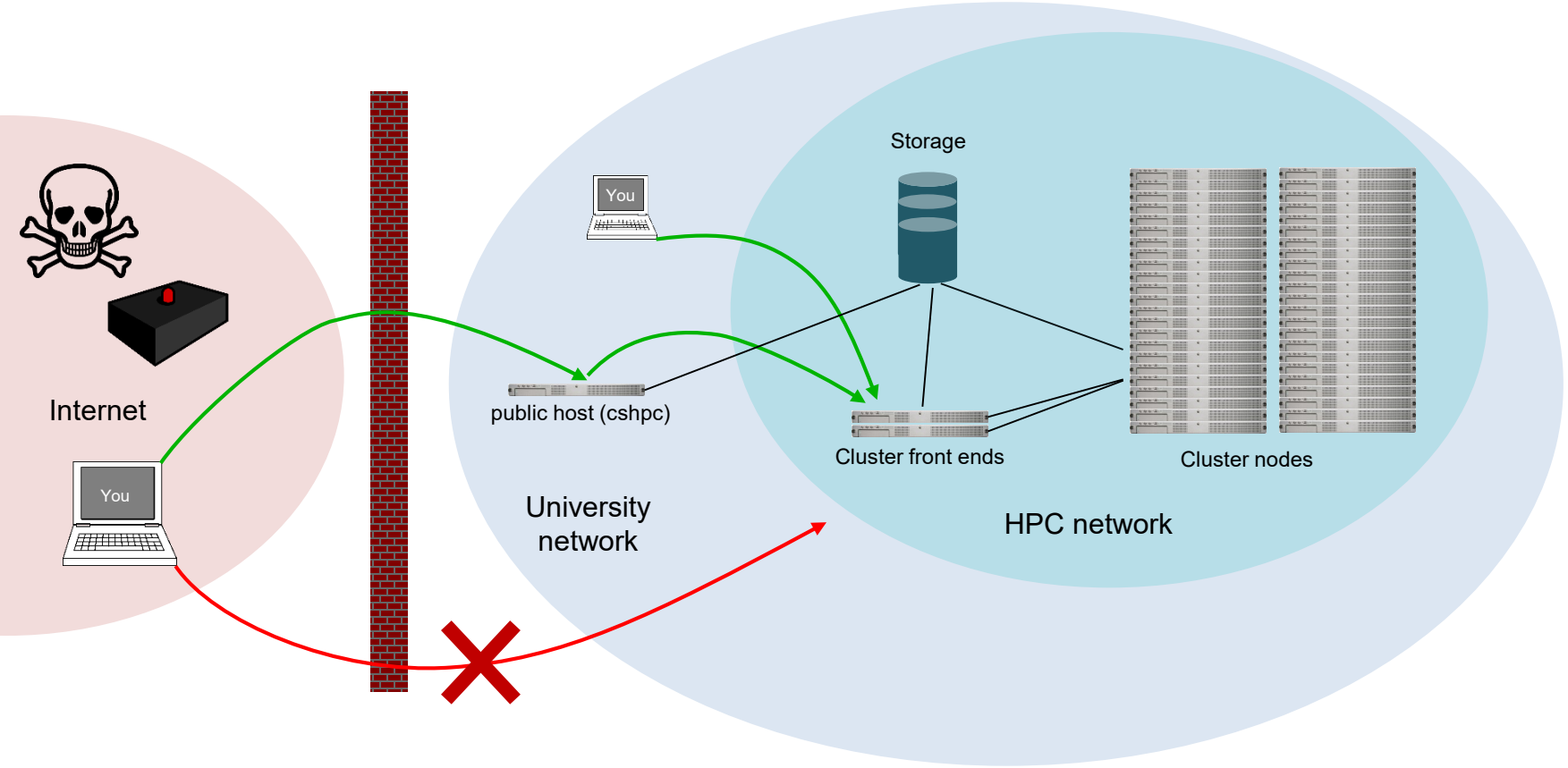
Benutzerrichtlinien:
https://www.rrze.fau.de/infocenter/rahmenbedingungen/richtlinien/benutzungsrichtlinien/

Antragsteller und Auftraggeber erklärt hiermit, von den Benutzungsrichtlinien sowie den ergänzenden Hinweisen auf der Rückseite dieses Antrags Kenntnis genommen zu haben.

Ort, Datum
Unterschrift Antragsteller
Unterschrift Auftraggeber/ Kontaktperson
IdM-Kennung Auftraggeber/Kontaktpers.

RRZE-interne Bemerkungen

Cluster access



Cluster access

- Primary point of contact: cluster frontends
  - **`woody.rrze.fau.de`** (also for TinyX)
  - **`emmy.rrze.fau.de`**
  - **`meggie.rrze.fau.de`**
  - Only available from within FAU (private IP addresses)

- Access from outside FAU: dialog server
  - **`cshpc.rrze.fau.de`**
  - The only machine with a public IP address

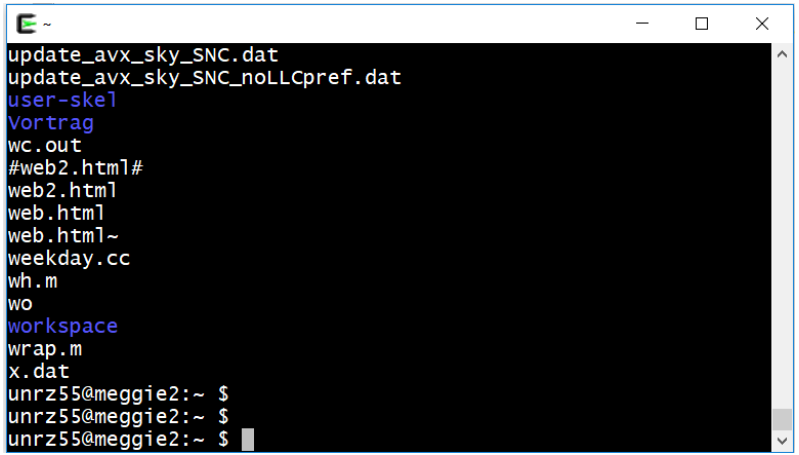
Secure Shell

- By default: text mode only

```
$ ssh ihpc02h@emmy.rrze.fau.de
```

- Basic knowledge of file handling, scripting, editing, etc. under Linux is required
- X11 forwarding with option **-X** or **-Y**
  - Requires local X server

- How to log into HPC systems at RRZE: https://youtu.be/J8PqWUfkCrI



Secure Shell client programs

- Linux: OpenSSH available in any distribution
- Mac: ditto
- Windows
  - Putty (**https://putty.org/**)
  - MobaXterm (**https://mobaxterm.mobatek.net/**)
    - includes an embedded X server
  - OpenSSH via Command/PowerShell
  - Linux Subsystem for Windows
  - WinSCP (data transfer only) (**https://winscp.net**)

NHR FAU

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

Working with data

https://hpc.fau.de/systems-services/systems-documentation-instructions/hpc-storage/

HPC High Performance Computing

File systems

- File system == directory structure that can store files
- Several file systems can be “mounted” at a compute node
  - Similar to drive letters in Windows (C:, D:, …)
  - Mount points can be anywhere in the root file system

- Available file systems differ in size, redundancy and how they should be used

- HPC Café on “Using file systems properly“ (especially for data-intensive applications):
  https://hpc.fau.de/files/2022/01/2022-01-11-hpc-cafe-file-systems.pdf
  https://www.fau.tv/clip/id/40199

RRZE file systems overview

Mount point	Access	Purpose	Technology	Backup	Snap-shots	Data lifetime	Quota
/home/hpc	$HOME	Source, input, important results	NFS on central servers, small	YES	YES	Account lifetime	50 GB
/home/vault	$HPCVAULT	Mid-/long-term storage	Central servers	YES	YES	Account lifetime	500 GB
/home/woody	$WORK	Short-/mid-term storage, General-purpose	Central NFS server	(NO)	NO	Account lifetime	500 GB
/lxfs	$FASTTMP (only within meggie)	High performance parallel I/O	Lustre parallel FS via InfiniBand	NO	NO	High watermark	Only inodes
/???	$TMPDIR	Node-local job-specific dir	HDD/SSD/ ramdisk	NO	NO	Job runtime	NO

File system quotas

- File system may impose quotas on
  - Stored data volume
  - Number of files and directories (inodes, actually)
- Quotas may be set per user or per group (or both)
- Hard quota
  - Absolute upper limit, cannot be exceeded
- Soft quota
  - May be exceeded temporarily (e.g., for 7 days – grace period)
  - Turns into hard quota at end of grace period

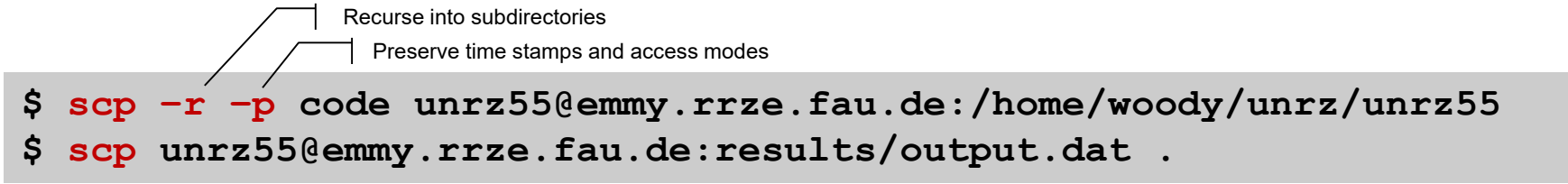
Displaying the quota limits

```
$ quota -s            # generic command
Disk quotas for user unrz55 (uid 12050):
     Filesystem  blocks   quota   limit   grace   files   quota   limit   grace
10.28.20.201:/hpcdatacloud/hpchome/shared
                 5544M  51200M    100G           72041    500k   1000k
wnfs1.rrze.uni-erlangen.de:/srv/home
                 112G   318G      477G            199k       0       0

$ shownicerquota.pl  # only on RRZE systems
   Path              Used     SoftQ    HardQ    Gracetime  Filec    FileQ    FiHaQ    FileGrace

   /home/hpc            5.7G    52.5G    104.9G       N/A      72K     500K   1,000K        N/A
   /home/woody          112G   333.0G    499.5G       N/A     188K                          N/A
```

Data transfer

- Most RRZE file systems are mounted at all HPC systems
  - Exception: parallel FS and node-local storage
- No NFS mounting from or to systems outside of RRZE

- → scp / rsync is the preferred file transfer tool from and to the outside



- Windows: **https://winscp.net/**

NHR FAU

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

Software

https://hpc.fau.de/systems-services/systems-documentation-instructions/environment/

HPC High Performance Computing

The modules system

- Linux standard distro packages available on frontends and to some extend on compute nodes, but might be outdated
- Software provided locally by RRZE via modules system
  - Compilers, libraries, commercial and open software
  - Installed on central server and available on all cluster nodes
- A package must be made available in the user's environment to become usable
  - Command: **`module`**
  - All module commands affect the current shell only!

The `module` command

Show available modules: `module avail`

```
$ module avail
---------------------- /apps/modules/data/applications ----------------------------------------------
amber-gpu/14p13-at15p06-gnu-intelmpi5.1-cuda7.5 gromacs/4.6.6-mkl-IVB
amber-gpu/16p04-at16p10-gnu-intelmpi5.1-cuda7.5 gromacs/5.0.4-mkl-IVB
amber/12p21-at12p38-intel16.0-intelmpi5.1       gromacs/5.1.1-mkl-IVB_d
---------------------- /apps/modules/data/development -----------------------------------------------
cuda/7.5                          intel64/16.0up04                 intelmpi/5.1up03-intel
cuda/8.0                          intel64/17.0up05(default)        llvm-clang/3.8.1
cuda/9.0                          intel64/18.0up02                 opencl/intel-cpuonly-5.2.0.10002
cuda/9.1                          intel64/18.0up03                 openmpi/1.08.8-gcc
$
```

The `module` command

Load a module: `module load <modulename>`

```
$ module load intel64
$ icc -V
Intel(R) C Intel(R) 64 Compiler for applications running on Intel(R) 64, Version 17.0.5.239 Build
20170817
Copyright (C) 1985-2017 Intel Corporation.  All rights reserved.
```

Display loaded modules: `module list`

```
$ module list
Currently Loaded Modulefiles:
  1) torque/current      2) intelmpi/2017up04-intel      3) mkl/2017up05      4) intel64/17.0up05
```

Module command summary

Command	What it does
module avail	List available modules
module whatis	Shows over-verbose listing of all modules
module list	Shows which modules are currently loaded
module load <pkg>	Loads module pkg, i.e., adjusts environment
module load <pkg>/<version>	Loads specific version of pkg instead of default
module unload <pkg>	Undoes what the load command did
module help <pkg>	Shows a detailed description of pkg
module show <pkg>	Shows what environment variables pkg modifies and how

https://hpc.fau.de/systems-services/systems-documentation-instructions/environment/#modules

Using Python

- Use anaconda modules instead of system installation

```
$ module avail python
------------ /apps/modules/modulefiles/tools ------------
python/2.7-anaconda   python/3.6-anaconda   python/3.7-anaconda(default) python/3.8-anaconda
```

- Build packages in an interactive job on the target cluster (especially for GPUs)
- It might be necessary to configure a proxy to access external repositories
- Install packages via conda/pip with **`--user`** option
- Change default package installation path from $HOME to $WORK

- More details: https://hpc.fau.de/systems-services/systems-documentation-instructions/special-applications-and-tips-tricks/python-and-jupyter/

NHR FAU

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

Running jobs

https://hpc.fau.de/systems-services/systems-documentation-instructions/batch-processing/

HPC High Performance Computing

Interactive work on the front-ends

- The cluster frontends are for interactive work
  - Editing, compiling, preparing input,…
  - Amount of compute time per binary is limited by system limits
    - E.g., after 1 hour of CPU time your process will be killed
  - MPI jobs are not allowed on front ends at RRZE
- Front-ends are shared among all users, so be considerate!
- Submit computational intensive work to the batch system to be run on the compute nodes!
- Use interactive batch jobs for debugging and testing.

Batch System

- Users can interact with the resources of the cluster via the “Batch system”
- “Batch jobs” encapsulate:
  - Resource requirements (number of nodes, number of GPUs, …)
  - Job runtime (usually max. 24 hours)
  - Setup of runtime environment
  - Commands for application run

- Batch system will handle queuing of jobs, resource distribution and allocation
- Job will run when resources become available

Example: Simple Slurm batch script

- Most simple batch script (job1.sh):

```
#!/bin/bash -l
~/bin/a.out arg1 arg2 arg3
```

- Submission:

```
iww042@meggie1$ sbatch --nodes=1 --time=01:00:00 job1.sh
1051341.madm
```

Example: Complex Slurm batch script

```
#!/bin/bash -l
#SBATCH --nodes=4 --ntasks-per-node=20 --time=06:00:00
#SBATCH --job-name=Sparsejob_33

#SBATCH --export=NONE
unset SLURM_EXPORT_ENV                  # avoid login shell settings

# create a temporary job dir on $WORK
mkdir ${WORK}/$SLURM_JOB_ID
cd ${WORK}/$SLURM_JOB_ID

# copy input file from location where job was submitted, and run
cp ${SLURM_SUBMIT_DIR}/inputfile .

srun --mpi=pmi2 ${HOME}/bin/a.out -i inputfile -o outputfile
```

Job submission options:
Nodes, cores per node, time, name,…

Job option sentinel

$SLURM_* variables contain job-relevant data

Actual run of your binary

Slurm batch job submission

```
iww042@meggie1$ sbatch job3.sh
Submitted batch job 357074

iww042@meggie1:~ $ squeue -l
Mon Jan 28 17:38:52 2019
             JOBID PARTITION     NAME     USER    STATE       TIME TIME_LIMI  NODES NODELIST(REASON)
            357074      work Sparsejo   iww042  RUNNING       0:35   1:00:00      4 m[0101-0104]
```

Jobs on TinyX

- Nearly all nodes use Slurm
- All jobs are submitted from the woody frontend via wrapper scripts (e.g. sbatch.tinygpu, sbatch.tinyfat)
- TinyGPU:
  - nodes are shared, granularity is one GPU with a corresponding proportion of CPU and main memory
  - Request a specific GPU type by e.g.
    - `sbatch.tinygpu --gres=gpu:1 […]` (if you don‘t care which type you get)
    - `sbatch.tinygpu --gres=gpu:rtx3080:1 […]` (to request a specific type)
    - `sbatch.tinygpu --gres=gpu:a100:1 --partition=a100 […]` (necessary for V100 and A100 GPUs)
- More details and examples:

  https://hpc.fau.de/systems-services/systems-documentation-instructions/clusters/tinyfat-cluster

  https://hpc.fau.de/systems-services/systems-documentation-instructions/clusters/tinygpu-cluster

Interactive batch job with Slurm

- TinyGPU / TinyFat

```
iww042@woody3$ salloc.tinygpu --gres=gpu:1 --time=01:00:00
```

```
iww042@woody3$ salloc.tinyfat --cpus-per-task=10 --time=01:00:00
```

- meggie:

```
iww042@meggie1$ salloc --nodes=1 --time=01:00:00
```

Slurm user commands (non-exhaustive)

Command	Purpose	Options
sbatch [<options>] <job_script>	Submit batch job	--time=HH:MM:SS --nodes=# --ntasks=# --ntasks-per-node=# --mail-user=<address> --mail-type=ALL|BEGIN|END|... --partition=work|devel
squeue [<options>]	Check job status	-j <JobID> show job -t RUNNING show only running jobs
scancel <JobID>	Delete batch job	–
srun <options>	Run program	Many options; see man page

https://hpc.fau.de/systems-services/systems-documentation-instructions/batch-processing/

Example: Torque batch script

```
#!/bin/bash -l
#PBS -l nodes=4:ppn=40,walltime=06:00:00
#PBS -N Sparsejob_33

# jobs always start in $HOME: change to a temporary job dir on $WOODYHOME
mkdir ${WORK}/$PBS_JOBID
cd ${WORK}/$PBS_JOBID

# copy input file from location where job was submitted, and run
cp ${PBS_O_WORKDIR}/inputfile .

/apps/rrze/bin/mpirun -npernode 20 ${HOME}/bin/a.out -i inputfile -o outputfile
```

Job submission options:
Nodes, cores per node, time, name,…

Job option sentinel

$PBS_* variables contain job-relevant data

Actual run of your binary

Example: Managing a Torque job

- Job ID can be used to check and control the job later

```
iww042@emmy1$ qsub job2.sh
1051342.eadm
```

```
iww042@emmy1$ qstat -a
eadm:

                                                                  Req'd    Req'd       Elap
Job ID                  Username    Queue    Jobname          SessID  NDS   TSK   Memory   Time    S   Time
----------------------- ----------- -------- ---------------- ------ ----- ------ ------ --------- - ---------
1051342.eadm            iww042      devel    test.sh             --      1     40    --   00:10:00 R  00:00:02

iww042@emmy1$ qalter -l walltime=02:00:00 1051342
iww042@emmy1$ qdel 1051342
```

- stdout/stderr will be in **<JobName>.[o|e]<JobID>**

Interactive batch job with Torque

```
iww042@emmy1$ qsub -l nodes=2:ppn=40,walltime=01:00:00 -I
qsub: waiting for job 1051378.eadm to start
qsub: job 1051378.eadm ready

Starting prologue... Mon Jan 28 15:55:44 CET 2019
Master node: e0104
Kill all process from other users
Adjust oom killer config
Clearing buffers and caches on the nodes.
Power management available, enabling ondemand governor
End of prologue: Mon Jan 28 15:55:51 CET 2019
iww042@e0104$
```

Some resources reserved for small jobs during working hours

Mostly harmless ☺

Type stuff here

Torque user commands (non-exhaustive)

Command	Purpose	Options
qsub [<options>] [-I|<job_script>]	Submit batch job (-I = interactive)	-l <resource_spec> -N <JobName> -o <stdout_filename> -e <stderr_filename> -M your@email.de –m abe -X X11 fowarding
qstat [<options>] [<JobID>|<queue>]	Check job status	-a friendly formatting -f verbose job info -r only running jobs -n show nodes of each job
qdel <JobID>	Delete batch job	–

NHR FAU

FAU FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

Some Dos and don’ts

HPC High Performance Computing

Good practices

- Be considerate. Clusters are valuable shared resources that have been paid by the taxpayer.
- Use the appropriate amount of parallelism
  - Most workloads are not highly scalable
  - Best to run scaling experiments to figure out “sweet spot”
- Use the appropriate file system(s)
  - #1 mistake: Overload metadata servers by doing tiny-size, high-frequency I/O to parallel FS
  - Delete obsolete data

Good practices

- Check your jobs regularly
  - Are the results OK?
  - Does the job actually use the allocated nodes in the intended way? Does it run with the expected performance?
  - Check if your job makes use of the GPUs
    - Use ssh to log into a node where you have a job running
    - Use e.g. nvidia-smi to check GPU utilization
  - For pytorch/tensorflow, check if GPUs are detected

    https://hpc.fau.de/systems-services/systems-documentation-instructions/special-applications-and-tips-tricks/tensorflow-pytorch/
  - Job Monitoring: https://www.hpc.rrze.fau.de/HPC-Status/job-info.php

    How to use it and what to look out for:

    https://hpc.fau.de/files/2019/11/2019-11-2_HPC_Cafe_monitoring.pdf

Good practices

- Talk to co-workers who are more experienced cluster users; let them educate you
- Do not re-use other people's job scripts if you don't understand them completely
- Look at tips and tricks for various applications (e.g. example batch scripts): https://hpc.fau.de/systems-services/systems-documentation-instructions/special-applications-and-tips-tricks/

Good practices

When reporting a problem to RRZE:

- Use the official contact hpc-support@fau.de – this will immediately open a helpdesk ticket
- Provide as much detail as possible so we know where to look
  - “My jobs always crash” will not do
  - Cluster, JobID, file system, time of event, …
  - Batch script, output files, …

NHR FAU

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

THANK YOU.

HPC@RRZE

https://hpc.fau.de

High Performance
Computing