Introduction to PDC environment

Thor Wikfeldt

PDC Center for High Performance Computing
KTH Royal Institute of Technology

Introduction to PDC, October 2017
PDC Overview

Outline

1. PDC Overview
2. Infrastructure
   - Beskow
   - Tegner
3. Accounts
   - Time allocations
   - Authentication
4. Development
   - Building
   - Compilers
   - Modules
   - Programming environments
5. Running jobs
   - SLURM
6. How to get help
History of PDC

- In 1988, envisioning that massive parallelism will become important for CS and HPC, a group of scientists from KTH School of Computer Science and Engineering applied for grant to buy a parallel computer.

- Market was surveyed and it was decided that Thinking Machines Corporation (TMC) offered the best choice with its Connection Machine system, CM2.

- What was to be called the Center for Parallel Computers was formed and inaugurated by Janne Carlsson, the President of KTH, on January 15, 1990.

- In January 1991 PDC applied for an upgrade of the CM2 to a CM200. The application was successful and the upgrade was installed in December 1991.
<table>
<thead>
<tr>
<th>Year</th>
<th>rank</th>
<th>procs.</th>
<th>peak gflops</th>
<th>vendor</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>2011</td>
<td>31</td>
<td>36384</td>
<td>305626.00</td>
<td>Cray</td>
<td>Lindgren¹</td>
</tr>
<tr>
<td>2010</td>
<td>76</td>
<td>11016</td>
<td>92534.40</td>
<td>Cray</td>
<td>Lindgren²</td>
</tr>
<tr>
<td>2010</td>
<td>89</td>
<td>9800</td>
<td>86024.40</td>
<td>Dell</td>
<td>Ekman³</td>
</tr>
<tr>
<td>2005</td>
<td>65</td>
<td>886</td>
<td>5670.40</td>
<td>Dell</td>
<td>Lenngren⁴</td>
</tr>
<tr>
<td>2003</td>
<td>196</td>
<td>180</td>
<td>648.00</td>
<td>HP</td>
<td>Lucidor⁵</td>
</tr>
<tr>
<td>1998</td>
<td>60</td>
<td>146</td>
<td>93.44</td>
<td>IBM</td>
<td>Strindberg⁶</td>
</tr>
<tr>
<td>1996</td>
<td>64</td>
<td>96</td>
<td>17.17</td>
<td>IBM</td>
<td>Strindberg³</td>
</tr>
<tr>
<td>1994</td>
<td>341</td>
<td>256</td>
<td>2.50</td>
<td>Thinking Machines</td>
<td>Bellman⁸</td>
</tr>
</tbody>
</table>

¹ XE6 12-core 2.1 GHz  
² XT6m 12-core 2.1 GHz  
³ PowerEdge SC1435 Dual core Opteron 2.2GHz, Infiniband  
⁴ PowerEdge 1850 3.2 GHz, Infiniband  
⁵ Cluster Platform 6000 rx2600 Itanium2 900 MHz Cluster, Myrinet  
⁶ SP P2SC 160 MHz  
⁷ SP2/96  
⁸ CM-200/8k
SNIC

Swedish National Infrastructure for Computing

National research infrastructure that provides a balanced and cost-efficient set of resources and user support for large scale computation and data storage to meet the needs of researchers from all scientific disciplines and from all over Sweden (universities, university colleges, research institutes, etc).
Access to EU Facilities and Experts
PDC and Industry

Working with industrial researchers and developers on major international projects that push high-performance computing to the next level.

Recently established a business development unit that provides consultancy and HPC services to industries.
Broad Range of Training

**Summer School**  Introduction to HPC held every year

**Specific Courses**  Programming with GPGPU, Recent Advances in Distributed and Parallel Computing and/or Cloud Computing, Software Development Tools, etc

**PDC User Days**  PDC Pub and Open House
## Support and System Staff

### First-line support
Provide specific assistance to PDC users related to accounts, login, allocations etc.

### System staff
System managers/administrators ensure that computing and storage resources run smoothly and securely.

### Application Experts
Hold PhD degrees in various fields and specialize in HPC. Assist researchers in optimizing, scaling and enhancing scientific codes for current and next generation supercomputers.
Services

- Access to supercomputers
- HPC training
- Postgraduate degree projects
- Visualization
- Support
- Expertise in HPC software
- Access to international HPC facilities
- Data storage
What is a cluster?

- Cluster
- Racks
- Blades
- Nodes
- Processors
- Cores
- Login nodes
- Compute nodes
- Dedicated nodes
- Transfer nodes
- Service nodes
Beskow - Cray XC40 system

Fastest machine in Scandinavia

- Lifetime: Q4 2018
- 11 racks 2060 nodes
- Intel Haswell Processor 2.30 GHz
  Intel Broadwell, 2.10 GHz
- 67,456 cores - 32/36 cores/node
- Aries Dragonfly network topology
- 104.7 TB memory - 64/128 GB/node

1 XC compute blade
1 Aries Network Chip (4 NICs)
4 Dual-socket Xeon nodes
4 Memory DIMM / Xeon node
Tegner
pre/post processing for Beskow

5 x 2TB Fat nodes
4 x 12 core Ivy Bridge, 2TB RAM
2 x Nvidia Quadro K420

5 x 1TB Fat nodes
4 x 12 core Ivy Bridge, 1TB RAM
2 x Nvidia Quadro K420

65 Thin Nodes
2 x 12 core Haswell, 512GB RAM
Nvidia Quadro K420 GPU

9 K80 Nodes
2 x 12 core Haswell, 512GB RAM
Nvidia Tesla K80 GPU

- Used for pre/post processing data
- Has large RAM nodes
- Has nodes with GPUs
- Has two transfer nodes
- Lifetime: Q4 2018
### Summary of PDC resources

<table>
<thead>
<tr>
<th></th>
<th>Beskow</th>
<th>Tegner</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cores in each node</strong></td>
<td>32/36</td>
<td>48/24</td>
</tr>
<tr>
<td><strong>Nodes</strong></td>
<td>2,060</td>
<td>74 x 24 Haswell/GPU</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10 x 48 Ivy bridge</td>
</tr>
<tr>
<td><strong>RAM (GB)</strong></td>
<td>64 or 128</td>
<td>74 x 512GB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 x 1024GB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 x 2TB</td>
</tr>
<tr>
<td><strong>Allocations</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(core hours per month)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Small</td>
<td>&lt; 5k</td>
<td>&lt; 5k</td>
</tr>
<tr>
<td>Medium</td>
<td>&lt; 200k</td>
<td>&lt; 80k</td>
</tr>
<tr>
<td>Large</td>
<td>≥ 200k</td>
<td></td>
</tr>
<tr>
<td><strong>Allocation via SNIC</strong></td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>AFS</strong></td>
<td>login node only</td>
<td>yes</td>
</tr>
<tr>
<td><strong>Lustre</strong></td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
File Systems

Andrew File System (AFS)
- Distributed file system accessible to any running AFS client
- Home directory
  `/afs/pdc.kth.se/home/[initial]/[username]`
- Access via Kerberos tickets and AFS tokens
- Not accessible to compute nodes on Beskow

Lustre File System (Klemming)
- Open-source massively parallel distributed file system
- Very high performance (5PB storage - 140GB/s bandwidth)
- NO backup (always move data when done) NO personal quota
- Home directory
  `/cfs/klemming/nobackup/[initial]/[username]`
Outline

1. PDC Overview
2. Infrastructure
   - Beskow
   - Tegner
3. Accounts
   - Time allocations
   - Authentication
4. Development
   - Building
   - Compilers
   - Modules
   - Programming environments
5. Running jobs
   - SLURM
6. How to get help
## Access requirements

### User account
Either SUPR or PDC

### Time allocation
Set the access limits

#### Apply for PDC account via SUPR
- [http://supr.snic.se](http://supr.snic.se)
- SNIC database of persons, projects, project proposals and more
- Apply and link SUPR account to PDC
- Valid post address for password

#### Apply for PDC account via PDC
- [http://www.pdc.kth.se/support/accounts/user](http://www.pdc.kth.se/support/accounts/user)
- Electronic copy of your passport
- Valid post address for password
- Membership of specific time allocation
Time Allocations

**Small allocation**
- Applicant can be a PhD student or more senior
- Evaluated on a technical level only
- Limits is usually $5K$ corehours each month

**Medium allocation**
- Applicant must be a senior scientist in Swedish academia
- Evaluated on a technical level only
- On large clusters: $200K$ corehours per month

**Large allocation**
- Applicant must be a senior scientist in Swedish academia
- Need evidence of successful work at a medium level
- Evaluated on a technical and scientific level
- Proposal evaluated by SNAC twice a year
Using resources

- All resources are free of charge for Swedish academia
- Acknowledgement *are* taken into consideration when applying
- Please acknowledge SNIC/PDC when using these resources:

**Acknowledge SNIC/PDC**

The computations/simulations/[SIMILAR] were performed on resources provided by the Swedish National Infrastructure for Computing (SNIC) at [CENTERNAME (CENTER-ACRONYM)]

**Acknowledge people**

NN at [CENTER-ACRONYME] is acknowledged for assistance concerning technical and implementation aspects [OR SIMILAR] in making the code run on the [OR SIMILAR] [CENTER-ACRONYM] resources.
Authentication

**Kerberos Authentication Protocol**

---

**Ticket**

- Proof of users identity
- Users use passwords to obtain tickets
- Tickets are cached on the user’s computer for a specified duration
- Tickets **should be created on your local computer**
- No passwords are required during the ticket’s lifetime

---

**Realm**

Sets boundaries within which an authentication server has authority (**NADA.KTH.SE**)

---

**Principal**

Refers to the entries in the authentication server database (**username@NADA.KTH.SE**)

---
Kerberos commands

Normal commands:

- kinit generates ticket
- klist lists kerberos tickets
- kdestroy destroys ticket file
- kpasswd changes password

On KTH-Ubuntu machines:

- pdc-kinit
- pdc-klist
- pdc-kdestroy
- pdc-kpasswd

Syntax:

```bash
$ kinit --forwardable username@NADA.KTH.SE
$ klist -Tf
```

Credentials cache: FILE:/tmp/krb5cc_500

- Principal: username@NADA.KTH.SE
- Issued: Mar 25 09:45
- Expires: Mar 25 19:45
- Flags: FI
  - krbtgt/NADA.KTH.SE@NADA.KTH.SE
- Flags: FA
  - afs/tdc.kth.se@NADA.KTH.SE
Login using Kerberos tickets

Get a 7 days forwardable ticket on your local system

$ kinit -f -l 7d username@NADA.KTH.SE

Forward your ticket via ssh and login

$ ssh
  -o GSSAPIDelegateCredential=yes
  -o GSSAPIAuthentication=yes
  -o GSSAPIKeyExchange=yes
username@clustername.pdc.kth.se

OR, when using ~/.ssh/config

$ ssh username@clustername.pdc.kth.se

Always create a kerberos ticket on your local system
https://www.pdc.kth.se/resources/software/login-1
Outline

1. PDC Overview
2. Infrastructure
   - Beskow
   - Tegner
3. Accounts
   - Time allocations
   - Authentication
4. Development
   - Building
   - Compilers
   - Modules
   - Programming environments
5. Running jobs
   - SLURM
6. How to get help
Compiling, Linking and Running Applications on HPC clusters

source code  C / C++ / Fortran (.c, .cpp, .f90, .h)

compile  Cray/Intel/GNU compilers
include headers, expand macros (.i, .ii)

assemble into machine code (.o, .obj)

link  Static Libraries (.lib, .a)
Shared Library (.dll, .so)
Executables (.exe, .x)

request allocation  submit job request to SLURM queuing system
salloc/sbatch

run application on scheduled resources
aprun/mpirun
Compiling serial and/or parallel code
specific to Tegner

**GNU Compiler Collection (gcc)**

- $ module load gcc openmpi
- $ gcc -fopenmp source.c
- $ g++ -fopenmp source.cpp
- $ gfortran -fopenmp source.F90
- $ mpicc -fopenmp source.c
- $ mpicxx -fopenmp source.cpp
- $ mpiif90 -fopenmp source.F90

**Intel compilers (i-compilers)**

- $ module load i-compilers
- $ icc -openmp source.c
- $ icpc -openmp source.cpp
- $ ifort -openmp source.F90
- $ module add i-compilers intelmpi
- $ mpiicc -openmp source.c
- $ mpiicpcp -openmp source.cpp
- $ mpiifort -openmp source.F90

**Portland Group Compilers (pgi)**

- $ module load pgi
- $ pgcc -mp source.c
- $ pgcpp -mp source.cpp
- $ pgf90 -mp source.F90

**CUDA compilers (cuda)**

- $ module load cuda
- $ nvcc source.cu
- $ nvcc -arch=sm_37 source.cu
# Modules

The *modules package* allow for dynamic add/remove of installed software packages to the running environment.

## Loading modules

- `module load <software_name>`
- `module add <software_name>`
- `module use <software_name>`

## Swapping modules

- `module swap <software_name_1> <software_name_2>`

## Unloading modules

- `module unload <software_name>`
$ module list

Currently Loaded Modulefiles:
  1) modules/3.2.6.7
  ...
  20) PrgEnv-cray/5.2.56

$ module avail [software_name]

-------------------------- /opt/modulefiles -----------------------------
gcc/4.8.1 gcc/4.9.1(default) gcc/4.9.2 gcc/4.9.3 gcc/5.1.0

$ module show software_name

------------------------- /opt/modulefiles/gcc/4.9.1 ---------------------
conflict gcc
prepend-path PATH /opt/gcc/4.9.1/bin
prepend-path MANPATH /opt/gcc/4.9.1/snos/share/man
prepend-path LD_LIBRARY_PATH /opt/gcc/4.9.1/snos/lib64
setenv GCC_PATH /opt/gcc/4.9.1

-------------------------------------------------------------------
Programming Environment Modules

Specific to Beskow

Cray  $ module load PrgEnv-cray  $ cc source.c
Intel $ module load PrgEnv-intel $ CC source.cpp
GNU   $ module load PrgEnv-gnu  $ ftn source.F90

Compiler wrappers: cc CC ftn

Advantages

- Compiler wrappers will automatically
  - link to BLAS, LAPACK, BLACS, SCALAPACK, FFTW
  - use MPI wrappers

Disadvantage

Sometimes you need to edit Makefiles which are not designed for Cray
Outline

1. PDC Overview
2. Infrastructure
   - Beskow
   - Tegner
3. Accounts
   - Time allocations
   - Authentication
4. Development
   - Building
   - Compilers
   - Modules
   - Programming environments
5. Running jobs
   - SLURM
6. How to get help

Thor Wikfeldt (PDC)

Introduction to PDC environment
How to run programs

- After login we are on a *login node* used only for:
  - submitting jobs,
  - editing files,
  - compiling small programs,
  - other computationally light tasks.

- **Never run calculations interactively on the login node**

- Instead, request compute resources *interactively* or via *batch script*

- All jobs must be connected to a time allocation

- For courses, PDC sets up a *reservation* for resources

- To manage the workload on the clusters, PDC uses a queueing/batch system
SLURM workload manager
Simple Linux Utility for Resource Management

- Open source, fault-tolerant, and highly scalable cluster management and job scheduling system
  - Allocates exclusive and/or non-exclusive access to resources for some duration of time
  - Provides a framework for starting, executing, and monitoring work on the set of allocated nodes
  - Arbitrates contention for resources by managing a queue
- Job Priority computed based on
  - Age the length of time a job has been waiting
  - Fair-share the difference between the portion of the computing resource that has been promised and the amount of resources that has been consumed
  - Job size the number of nodes or CPUs a job is allocated
  - Partition a factor associated with each node partition
Interactive session

**salloc**

Request an interactive allocation of resources

```
$ salloc -A <account> -t <d-hh:mm:ss> -N <nodes>
salloc: Granted job allocation 123456
```

Run application on **Beskow**

```
$ aprun -n <PEs> -d <depth> -N <PEs_per_node> ./binary.x
```

#PEs - number of processing elements

#depth - number of threads (depth) per PE

#PEs_per_node - PEs per node

Run application on **Tegner**

```
$ mpirun -np <cores> ./binary.x
```
Launch jobs in the background

Submit the job to SLURM queue

$ sbatch <script>
Submitted batch job 958287

The script should contain all necessary data to identify the account and requested resources

Example of request to run myexe for 1 hour on 4 nodes

#!/bin/bash -l

#SBATCH -A summer-2017
#SBATCH -J myjob
#SBATCH -t 1:00:00
#SBATCH --nodes=4
#SBATCH --ntasks-per-node=32
#SBATCH -e error_file.e
#SBATCH -o output_file.o

aprun -n 128 ./myexe > my_output_file 2>&1
## Monitoring and/or cancelling running jobs

### `squeue -u $USER`

Displays all queue and/or running jobs that belong to the user

```bash
cira@beskow-login2:~> squeue -u cira
```

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USER</th>
<th>ACCOUNT</th>
<th>NAME</th>
<th>ST</th>
<th>REASON</th>
<th>START_TIME</th>
<th>TIME</th>
<th>TIME_LEFT</th>
<th>NODES</th>
<th>CPUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>957519</td>
<td>cira</td>
<td>pdc.staff</td>
<td>VASP-test</td>
<td>R</td>
<td>None</td>
<td>2016-08-15T08:15:24</td>
<td>6:09:42</td>
<td>17:49:18</td>
<td>16</td>
<td>1024</td>
</tr>
</tbody>
</table>
```

### `scancel [job]`

 Stops a running job or removes a pending one from the queue

```bash
cira@beskow-login2:~> scancel 957519
salloc: Job allocation 957891 has been revoked.
```

```bash
cira@beskow-login2:~> squeue -u cira
```

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USER</th>
<th>ACCOUNT</th>
<th>NAME</th>
<th>ST</th>
<th>REASON</th>
<th>START_TIME</th>
<th>TIME</th>
<th>TIME_LEFT</th>
<th>NODES</th>
<th>CPUS</th>
</tr>
</thead>
</table>
Outline

1. PDC Overview
2. Infrastructure
   - Beskow
   - Tegner
3. Accounts
   - Time allocations
   - Authentication
4. Development
   - Building
   - Compilers
   - Modules
   - Programming environments
5. Running jobs
   - SLURM
6. How to get help
How to start your project

- Proposal for a small allocation
- Develop and test your code
- Run and evaluate scaling
- Proposal for a medium (large) allocation
PDC support

- Many questions can be answered by reading the web documentation: https://www.pdc.kth.se/support
- Preferably contact PDC support by email: support@pdc.kth.se
  - you get a ticket number.
  - always include the ticket number in follow-ups/replies they look like this: [SNIC support #12345]
- Or by phone: +46 (0)8 790 7800
- You can also make an appointment to come and visit.
How to report problems

- Do not report new problems by replying to old/unrelated tickets.
- Split unrelated problems into separate email requests.
- Use a descriptive subject in your email.
- Give your PDC user name.
- Be as specific as possible.
- For problems with scripts/jobs, give an example. Either send the example or make it accessible to PDC support.
- Make the problem example as small/short as possible.
- Provide all necessary information to reproduce the problem.
- If you want the PDC support to inspect some files, make sure that the files are readable.
- Do not assume that PDC support personnel have admin rights to see all your files or change permissions.

support@pdc.kth.se
Questions...?
Hands-on exercise

Login

- Some configuration steps are needed to log in to PDC
- Depends on OS: https://www.pdc.kth.se/resources/software/login-1
- In short, Kerberos and SSH supporting GSSAPI key exchange must be installed
- If needed, you will receive help to connect from your own laptops

Live demo

- We will now demonstrate some key steps in logging in and running at PDC
- We will generate Kerberos tickets and log in with ssh
- We will compile and run MPI, OpenMP and CUDA code on Beskow and Tegner
SSH configuration (Linux and Mac)

kthw@local~$ cat .ssh/config
# Hosts we want to authenticate to with Kerberos
Host *.kth.se *.kth.se.
# User authentication based on GSSAPI is allowed
GSSAPIAuthentication yes
# Key exchange based on GSSAPI may be used for server authentication
GSSAPIKeyExchange yes
# Hosts to which we want to delegate credentials
# Forward (delegate) credentials (tickets) to the server.
GSSAPIDelegateCredentials yes
# Prefer GSSAPI key exchange
PreferredAuthentications gssapi-keyex,gssapi-with-mic
# All other hosts
Host *
Kerberos configuration (Linux and Mac)

kthw@local~$ cat /etc/krb5.conf
[domain_realm]
   .pdc.kth.se = NADA.KTH.SE
[appdefaults]
   forwardable = yes
   forward = yes
   krb4_get_tickets = no
[libdefaults]
   default_realm = NADA.KTH.SE
   dns_lookup_realm = true
dns_lookup_kdc = true
Login and running Kerberos

Create and list tickets

```
kthw@local~$ klist
klist: No credentials cache found

kthw@local~$ kinit -f kthw@NADA.KTH.SE
Password for kthw@NADA.KTH.SE:

kthw@local~$ klist -Tf
Ticket cache: KCM:501
Default principal: kthw@NADA.KTH.SE

Valid starting          Expires            Service principal
08/03/2017 16:39:56     08/04/2017 16:39:50  krbtgt/NADA.KTH.SE@NADA.KTH.SE
Flags: FIA```
## Log in to Beskow, check ticket

```
kthw@local:~$ ssh kthw@beskow.pdc.kth.se
kthw@beskow-login2:~$ klist -f
```

Credentials cache: FILE:/tmp/krb5cc_H26527
Principal: kthw@NADA.KTH.SE

<table>
<thead>
<tr>
<th>Issued</th>
<th>Expires</th>
<th>Flags</th>
<th>Principal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aug 3 16:41:51 2017</td>
<td>Aug 4 16:39:50 2017</td>
<td>FfA</td>
<td>krbtgt/NADA.KTH.SE@NADA.KTH.SE</td>
</tr>
<tr>
<td>Aug 3 16:41:52 2017</td>
<td>Aug 4 16:39:50 2017</td>
<td>fA</td>
<td>afs/pdc.kth.se@NADA.KTH.SE</td>
</tr>
<tr>
<td>Aug 3 16:41:52 2017</td>
<td>Aug 4 16:39:50 2017</td>
<td>fA</td>
<td><a href="mailto:afs@NADA.KTH.SE">afs@NADA.KTH.SE</a></td>
</tr>
</tbody>
</table>
### Inspect module system

```
kthw@beskow-login2:~$ module list
...
kthw@beskow-login2:~$ module avail
...
kthw@beskow-login2:~$ module avail gcc
...
kthw@beskow-login2:~$ CC -V
Cray C++ : Version 8.3.4  Mon Aug 07, 2017  15:04:06
kthw@beskow-login2:~$ module swap PrgEnv-cray PrgEnv-gnu
kthw@beskow-login2:~$ CC --version
 g++ (GCC) 4.9.1 20140716 (Cray Inc.)
```
Interactive job on Beskow

Go to Klemming and start interactive session

```
kthw@beskow-login2:~$ cd /cfs/klemming/nobackup/k/kthw/

# (command line shortened below here)
$ salloc -A pdc-test-2017 -N 1 -t 0:10:0
salloc: Granted job allocation 1733496

$ hostname
beskow-login2.pdc.kth.se

$ aprun -n 1 hostname
nid01610

$ exit
salloc: Relinquishing job allocation 1733497
salloc: Job allocation 1733497 has been revoked.
```
Login and running

Interactive job on Beskow

We compile and run MPI and OpenMP codes

```
kthw@beskow-login2:~$ mkdir -p /cfs/klemming/nobackup/k/kthw/pdc_intro/mpi
kthw@beskow-login2:~$ cd /cfs/klemming/nobackup/k/kthw/pdc_intro/mpi
$ cp ~kthw/Public/pdc_intro/hello_world.f90 .
$ module swap PrgEnv-cray PrgEnv-gnu
$ salloc -A pdc-test-2017 -N 1 -t 0:10:0
salloc: Granted job allocation 1733496
$ ftn -O2 hello_world.f90 -o hello.x
$ aprun -n 32 ./hello.x

$ mkdir ../omp
$ cd ../omp
$ cp ~kthw/Public/pdc_intro/omp_hello.c .
$ cc -fopenmp omp_hello.c -o omp_hello.x
$ export OMP_NUM_THREADS=32
$ aprun -n 1 -d 32 ./omp_hello.x
```
Interactive job on Tegner

**Go to Klemming and start interactive session**

```
kthw@tegner-login-1:~$ cd /cfs/klemming/nobackup/k/kthw/

$ salloc -A pdc-test-2017 -N 1 -t 0:10:0
salloc: job 666061 has been allocated resources
salloc: Waiting for resource configuration
salloc: Nodes t03n03 are ready for job

# either run on allocated node using srun/mpirun, or by logging into node
$ srun -n 1 hostname
t03n03.pdc.kth.se

# to log in to node, must open new terminal:
local-machine $ ssh kthw@t03n03.pdc.kth.se
t03n03 $ mpirun -np 24 my_code.x
```
Interactive job on Tegner

We compile and run a CUDA code

```
kubectl~$ mkdir -p /cfs/nobackup/k/kthw/pdc_intro/cuda
kubectl~$ cd /cfs/nobackup/k/kthw/pdc_intro/cuda
$ cp ~kthw/Public/pdc_intro/hello.cu .
$ module load cuda/8.0
$ salloc -A pdc-test-2017 -N 1 -t 0:10:0 --gres=gpu:K420:1
salloc: job 674692 has been allocated resources
salloc: Nodes t02n17 are ready for job
$ nvcc hello.cu -o hello.x # for K80 nodes, also need -arch=sm_37
$ srun ./hello.x
Hello World!
```
Compile code and write batch script (Beskow)

```bash
$ cp ~/Public/hello_world.f90 .
$ ftn -o hello_world.x hello_world.f90

$ cat <<EOF > submit.bash
#!/bin/bash -l
#SBATCH -A pdc-test-2017
#SBATCH -J myjob
#SBATCH -t 0:10:00
#SBATCH -N 1
#SBATCH -e error_file.e
#SBATCH -o output_file.o

aprun -n 32 ./hello_world.x > my_output_file 2>&1
EOF
```
Login and running Batch job

Submit and monitor job

$ sbatch submit.bash
$ squeue -u kthw

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USER</th>
<th>ACCOUNT</th>
<th>NAME</th>
<th>ST</th>
<th>REASON</th>
<th>START_TIME</th>
<th>TIME_LEFT</th>
<th>NODES</th>
<th>CPUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1735211</td>
<td>kthw</td>
<td>pdc.sta</td>
<td>myjob</td>
<td>R</td>
<td>None</td>
<td>2017-08-07T16:31:01</td>
<td>0:00</td>
<td>1</td>
<td>64</td>
</tr>
</tbody>
</table>

$ cat my_output_file

Hello from rank 31 of 32
Hello from rank 13 of 32
Hello from rank 26 of 32
Hello from rank 10 of 32
Hello from rank 17 of 32
Hello from rank 14 of 32
Hello from rank 1 of 32
...

...
Questions...?
Introducing the unix shell

Login into Beskow

$ ssh beskow.pdc.kth.se
Last login: Fri Feb 13 20:20:06 2016 from example.com
bast@beskow-login2:~$ 

- Command Line Interface often more efficient than GUI
- High action-to-keystroke ratio
- Creativity through pipelines
- System is configured with text files
- Calculations are configured and run using text files
- Good for working over network
- Good for reproducibility
- Good for unsupervised work-flows
**Bash: Files and directories**

**pwd command returns current directory**

```bash
user@machine:~$ pwd
/afs/pdc.kth.se/home/u/user
```

**Change the directory with cd**

```bash
user@machine:~$ cd tmp/talks/
user@machine:~/tmp/talks$ pwd
/afs/pdc.kth.se/home/u/user/tmp/talks
```

**List the contents with ls -l**

```bash
user@machine:~/tmp/talks$ ls -l
```

```
total 237
drwx------ 3 user csc-users 2048 Aug 17 15:21 img
-rw------- 1 user csc-users 18084 Aug 17 15:21 pdc-env.html
```
Bash: Creating and editing files and directories

Command `mkdir` creates a new directory

```
$ mkdir results
$ cd results
```

File editors

```
$ nano draft.txt
$ emacs draft.txt
$ vi draft.txt
$ vim draft.txt # this is Vi "improved"
```
Bash: Copying, moving, renaming, and deleting

$ cp draft.txt backup.txt
$ cp -r results backup
$ mv draft.txt draft_2.txt
$ mv results backup
$ mv results ..
$ rm draft.txt
$ rm -r results

# copy file
# recursively copy directory
# move/rename file
# move/rename directory
# move directory one level up
# remove file
# remove directory and contents
Unix shell

Bash: Finding things

Extract lines which contain an expression with `grep`

```bash
$ grep fixme draft.txt
$ man grep
$ grep energy results.out | sort | uniq
```

Redirecting output

```bash
$ grep energy results.out | sort | uniq > energies.txt
$ grep dipole results.out | sort | uniq >> energies.txt
$ cat results2.txt
$ cat results2.txt >> results_all.txt
```
Bash: Writing shell scripts

Edit script in preferred editor

```bash
#!/usr/bin/env bash
# here we loop over all files that end with *.out
for file in *.out; do
    echo $file
grep energy $file
done
```

Change permissions `chmod` to make the script executable

```bash
# make it executable
$ chmod u+x my_script
# run it
$ ./my_script
```