

Introduction to PDC environment

Thor Wikfeldt

PDC Center for High Performance Computing
KTH Royal Institute of Technology

Writing parallel applications with MPI, December 2017



Outline

1 PDC Overview

2 Infrastructure

- Beskow
- Tegner

3 Accounts

- Time allocations
- Authentication

4 Development

- Building
- Modules
- Programming environments
- Compilers

5 Running jobs

- SLURM

6 How to get help



History of PDC

Year	rank	procs.	peak gflops	vendor	name
2011	31	36384	305626.00	Cray	Lindgren ¹
2010	76	11016	92534.40	Cray	Lindgren ²
2010	89	9800	86024.40	Dell	Ekman ³
2005	65	886	5670.40	Dell	Lenngren ⁴
2003	196	180	648.00	HP	Lucidor ⁵
1998	60	146	93.44	IBM	Strindberg ⁶
1996	64	96	17.17	IBM	Strindberg ⁷
1994	341	256	2.50	Thinking Machines	Bellman ⁸

¹XE6 12-core 2.1 GHz

²XT6m 12-core 2.1 GHz

³PowerEdge SC1435 Dual core Opteron 2.2GHz, Infiniband

⁴PowerEdge 1850 3.2 GHz, Infiniband

⁵Cluster Platform 6000 rx2600 Itanium2 900 MHz Cluster, Myrinet

⁶SP P2SC 160 MHz

⁷SP2/96

⁸CM-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).



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.



Outline

1 PDC Overview

2 Infrastructure

- Beskow
- Tegner

3 Accounts

- Time allocations
- Authentication

4 Development

- Building
- Modules
- Programming environments
- Compilers

5 Running jobs

- SLURM

6 How to get help



What is a cluster?



- Cluster
- Racks
- Blades
- Nodes
- Processors
- Cores
- Login nodes
- Compute nodes
- Dedicated nodes
- Transfer nodes
- Service nodes

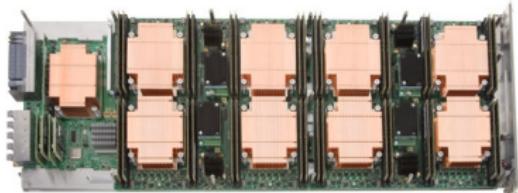


Beskow - Cray XC40 system



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

4x 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

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

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
- Modules
- Programming environments
- Compilers

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>
- 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

- www.pdc.kth.se/support/documents/starting/get_account.html
- 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

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`

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

```
Credentials cache : FILE:/tmp/krb5cc_500
  Principal: username@NADA.KTH.SE
Issued      Expires      Flags  Principal
Mar 25 09:45 Mar 25 19:45 FI  krbtgt/NADA.KTH.SE@NADA.KTH.SE
Mar 25 09:45 Mar 25 19:45 FA  afs/pdc.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/support/documents/login/login.html>



Outline

1 PDC Overview

2 Infrastructure

- Beskow
- Tegner

3 Accounts

- Time allocations
- Authentication

4 Development

- Building
- Modules
- Programming environments
- Compilers

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



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>
```

Modules

Displaying modules

```
$ 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/snobs/share/man  
prepend-path  LD_LIBRARY_PATH /opt/gcc/4.9.1/snobs/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

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
$ mpif90  -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
$ mpiicpc -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
```

Outline

1 PDC Overview

2 Infrastructure

- Beskow
- Tegner

3 Accounts

- Time allocations
- Authentication

4 Development

- Building
- Modules
- Programming environments
- Compilers

5 Running jobs

- SLURM

6 How to get help

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

sbatch

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 edu17.intrompi
#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

```
cira@beskow-login2:~> squeue -u cira
  JOBID      USER ACCOUNT          NAME  ST REASON      START_TIME          TIME  TIME_LEFT NODES  CPUS
957519      cira pdc.staff    VASP-test    R  None  2016-08-15T08:15:24  6:09:42  17:49:18    16  1024
957757      cira pdc.staff    VASP-run     R  None  2016-08-15T11:14:20  3:10:46  20:48:14   128  8192
```

scancel [job]

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

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

```
cira@beskow-login2:~> squeue -u cira
  JOBID      USER ACCOUNT          NAME  ST REASON      START_TIME          TIME  TIME_LEFT NODES  CPUS
957757      cira pdc.staff    VASP-run     R  None  2016-08-15T11:14:20  3:10:46  20:48:14   128  8192
```



Outline

1 PDC Overview

2 Infrastructure

- Beskow
- Tegner

3 Accounts

- Time allocations
- Authentication

4 Development

- Building
- Modules
- Programming environments
- Compilers

5 Running jobs

- SLURM

6 How to get help



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](tel:+46(0)87907800)
- You can also make an appointment to [come and visit](#).



How to report problems

support@pdc.kth.se

- 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.



Questions...?



Hands-on exercise

Login

- Some configuration steps are needed to log in to PDC
- Depends on OS: <https://www.pdc.kth.se/support/documents/login/login.html>
- 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

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
Host *.csc.kth.se *.csc.kth.se. *.nada.kth.se *.nada.kth.se. \
    *.pdc.kth.se *.pdc.kth.se.

# Forward (delegate) credentials (tickets) to the server.
GSSAPIDelegateCredentials yes

# Prefer GSSAPI key exchange
PreferredAuthentications gssapi-keyex,gssapi-with-mic

# All other hosts
Host *
```

Kerberos

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
```



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		

Login

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
```

Issued	Expires	Flags	Principal
Aug 3 16:41:51 2017	Aug 4 16:39:50 2017	FfA	krbtgt/NADA.KTH.SE@NADA.KTH.SE
Aug 3 16:41:52 2017	Aug 4 16:39:50 2017	fA	afs/pdc.kth.se@NADA.KTH.SE
Aug 3 16:41:52 2017	Aug 4 16:39:50 2017	fA	afs@NADA.KTH.SE



Modules

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 edu17.intrompi -N 1 -t 0:10:0 --res=edu-Dec-11  
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.
```

Interactive job on Beskow

We compile and run MPI and OpenMP codes

```
kthw@beskow-login2:~$ mkdir -p /cfs/klemming/nobackup/k/kthw/mpi_course
kthw@beskow-login2:~$ cd /cfs/klemming/nobackup/k/kthw/mpi_course
$ cp ~kthw/Public/intro_mpi_dec2017/hello_world.f90 .
$ module swap PrgEnv-cray PrgEnv-gnu
$ salloc -A edu17.intrompi --res=edu-Dec-11 -N 1 -t 0:10:0
salloc: Granted job allocation 1733496
$ ftn -O2 hello_world.f90 -o hello.x
$ aprun -n 32 ./hello.x

$ cp ~kthw/Public/intro_mpi_dec2017/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
```

Batch job

Compile code and write batch script (Beskow)

```
$ cp ~/Public/intro_mpi_dec2017/hello_world.f90 .
$ ftn -o hello_world.x hello_world.f90

$ cat <<EOF > submit.bash
#!/bin/bash -l

#SBATCH -A edu17.intrompi
#SBATCH --reservation=edu-Dec-11
#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
```

Batch job

Submit and monitor job

```
$ sbatch submit.bash
$ squeue -u kthw
JOBID USER ACCOUNT NAME  ST REASON START_TIME TIME TIME_LEFT NODES CPUS
1735211 kthw pdc.sta myjob R None 2017-08-07T16:31:01 0:00 10:00 1 64

$ 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

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

Change the directory with **cd**

```
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**

```
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
```



Bash: Finding things

Extract lines which contain an expression with **grep**

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

Redirecting output

```
$ 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

```
#!/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

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