ECMWF Environment on the CRAY

Cristian Simarro
Cristian.Simarro@ecmwf.int
User Support Section

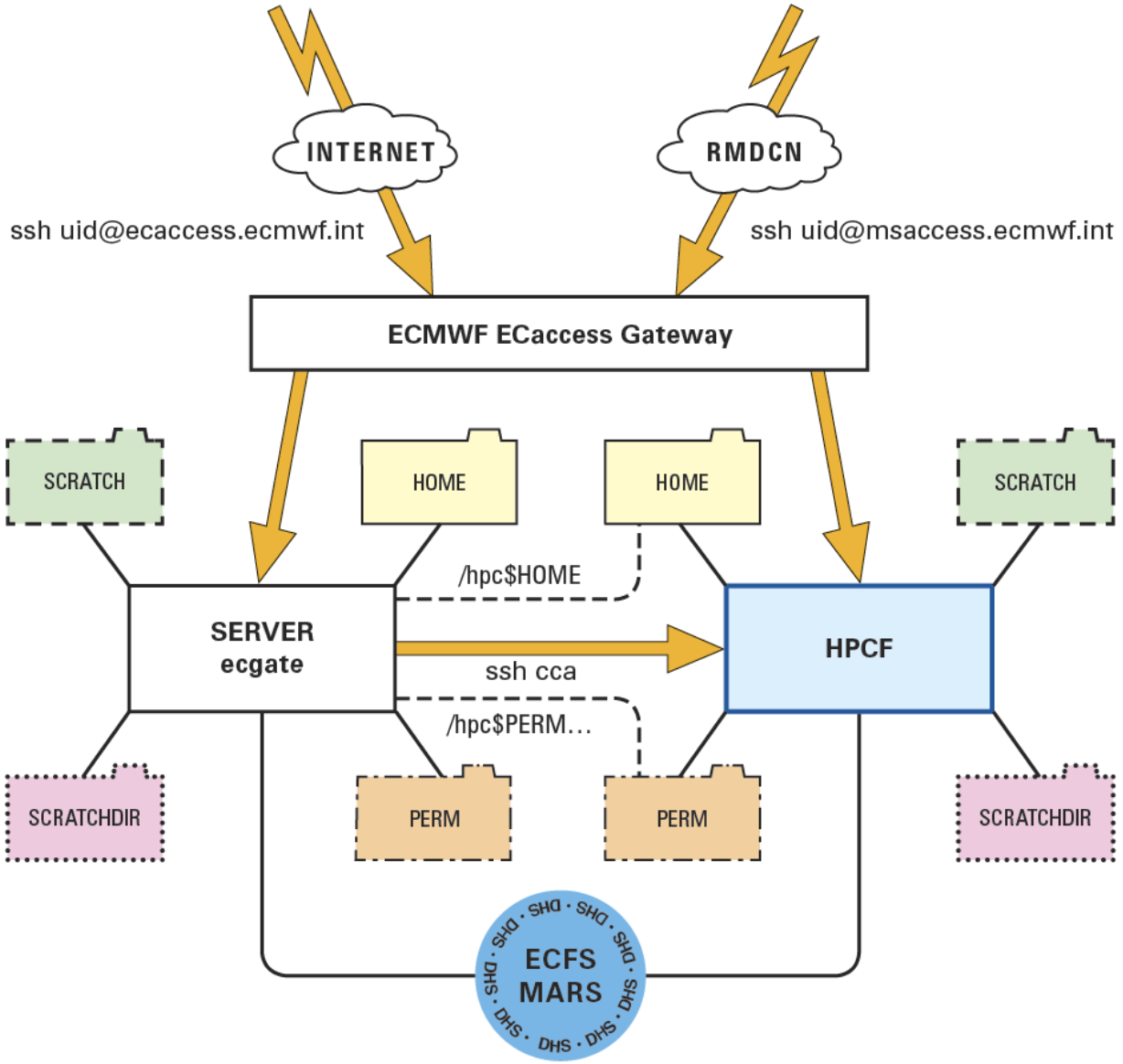
Outline

- Shells
- Filesystems
- Modules
- Practicals

Shells

- Only ksh and bash are supported for interactive shells.
  - csh users are automatically transferred into a ksh
  - use changesh from ecgate to change your login shell if you wish to do so.
- Use the following files to customise your environment:
  - ~/.user_profile for environment variables
  - ~/.user_kshrc or ~/.user_bashrc for shell aliases

Filesystems



Filesystems

File System	Suitable for	Quota
$HOME	permanent files, e. g. .profile, utilities, sources, libraries	512 MB / 22000 files
$PERM	permanent files without the need for automated backups, e. g. smaller input files for model runs, std output etc.	27 GB / 210000 files
$SCRATCH	all temporary (large) files	No quota, 1 month retention
$SCRATCHDIR	data to be automatically deleted at the end of a job	(part of $SCRATCH)

```
trx@cca-login1:/home/ectrain/trx> quota

Quota for $HOME and $PERM:
Disk quotas for user trx (uid 414):
     Filesystem  blocks   quota   limit   grace   files   quota   limit   grace
cnasa1:/vol/home
                 10144    480M    500M             338   20000   22000
Disk quotas for user trx (uid 414):
     Filesystem  blocks   quota   limit   grace   files   quota   limit   grace
cnasa2:/vol/perm
                     0  26624M  27648M               1    200k    210k
```

Filesystems

- Only $HOME is backed up
- $SCRATCHDIR is part of $SCRATCH
- There is a new $TMPDIR on the login nodes (internal drive)
- $HOME, $SCRATCH and $PERM on ecgate and HPCF are different
- Different select/delete policies may apply on temporary file systems
- Do not rely on select/delete. Clear your space as soon as possible!
- All critical files from file systems other than $HOME should be copied to ECFS without delay.
- To "transfers" between files systems which are on the same physical disk storage, e.g. between $SCRATCHDIR and $SCRATCH, use the mv command
- To transfer files between different platforms (e.g. ecgate – HPCF) use scp or rsync

Modules framework

- Utility to manage and control the user environment
  - Software packages and programs
  - Compiler flags and libraries
  - Runtime of applications

One command to rule them all

Why are they important?

- If a module is not loaded or the wrong version is loaded:
  - Your job might fail because it won't find a specific program to run.
  - The compilation of software might fail because it won't find the required libraries.
  - The compilation of software may work, but it might produce binaries linked with undesired versions of libraries.

Be aware of the environment before you start working to avoid surprises…

```
$> module list
```

What is in modules?

- Cray stuff:
  - Cray Programming Environment
  - Compilers
  - Various CRAY packages and libraries
- ECMWF stuff:
  - ECMWF software packages
    - Grib API, ECFS client, sms, ecflow …
  - 3rd party packages and libraries

```
$> module avail
```

Main actions

- See what is loaded and what is available to load

```
$> module list
$> module avail
```

- Load and unload a module

```
$> module load package
$> module load package/version
$> module unload package
```

- Switch/swap an already loaded module by another one

```
$> module switch package/version1
$> module switch package/version1 package/version2
```

Load example

```
usxa@cca-login1:~> cdo -V
If 'cdo' is not a typo you can run the following command to lookup the package that contains the binary:
    command-not-found cdo
-bash: cdo: command not found
usxa@cca-login1:~> module list
Currently Loaded Modulefiles:
  1) modules/3.2.6.7                       17) alps/5.2.0-2.0502.8594.12.4.ari
  2) eswrap/1.1.0-1.020200.1130.0          18) rca/1.0.0-2.0502.49765.5.41.ari
  3) switch/1.0-1.0502.50885.3.4.ari       19) atp/1.7.2
  4) craype-network-aries                  20) PrgEnv-cray/5.2.14
  5) craype/2.1.1                          21) pbs/12.1.400.132424
  6) cce/8.2.7                             22) craype-ivybridge
  7) cray-libsci/12.2.0                    23) cray-mpich/6.3.1
  8) udreg/2.3.2-1.0502.8413.2.9.ari       24) verbose/false
  9) ugni/5.0-1.0502.8670.4.22.ari         25) ecfs/2.0.13rc2
 10) pmi/5.0.3-1.0000.9981.128.2.ari       26) jasper/1.900.1
 11) dmapp/7.0.1-1.0502.8638.9.93.ari      27) grib_api/1.12.3
 12) gni-headers/3.0-1.0502.8554.6.6.ari   28) emos/394-r64
 13) xpmem/0.1-2.0502.50559.4.2.ari        29) sms/4.4.13
 14) job/1.5.5-0.1_2.0502.49000.2.39.ari   30) batch_utils/1.4
 15) csa/3.0.0-1_2.0502.49605.4.45.ari     31) verbose/true(default)
 16) dvs/2.4_0.9.0-1.0502.1696.2.39.ari
usxa@cca-login1:~> module avail cdo

----------------- /usr/local/apps/modulefiles/tools_and_libraries/data_formats -----------------
cdo/1.6.1(default) cdo/1.6.4
usxa@cca-login1:~> module load cdo
load cdo 1.6.1 (PATH, CDO_DIR, CDO)
usxa@cca-login1:~> cdo -V
Climate Data Operators version 1.6.1 (http://code.zmaw.de/projects/cdo)
...
```

Switch example

- Short option (preferred):

```
usxa@cca-login1:~> grib_ls -V
grib_api Version 1.12.3
usxa@cca-login1:~> module switch grib_api/1.13.1
switch1 grib_api 1.12.3 (PATH, MANPATH, GRIB_API_DIR, GRIB_API_VERSION, GRIB_API_INCLUDE, GRIB_API_LIB,
GRIB_API_INCLUDE_DIR, GRIB_API_LIB_DIR)
switch2 grib_api 1.13.1 (PATH, MANPATH, GRIB_API_DIR, GRIB_API_VERSION, GRIB_API_INCLUDE, GRIB_API_LIB,
GRIB_API_INCLUDE_DIR, GRIB_API_LIB_DIR)
usxa@cca-login1:~> grib_ls -V
grib_api Version 1.13.1
```

- Long option:

```
usxa@cca-login1:~> grib_ls -V
grib_api Version 1.12.3
usxa@cca-login1:~> module unload grib_api
remove jasper 1.900.1 (PATH, MANPATH, JASPER_DIR, JASPER_INCLUDE, JASPER_LIB, JASPER_INCLUDE_DIR,
JASPER_LIB_DIR)
remove grib_api 1.12.3 (PATH, MANPATH, GRIB_API_DIR, GRIB_API_VERSION, GRIB_API_INCLUDE, GRIB_API_LIB,
GRIB_API_INCLUDE_DIR, GRIB_API_LIB_DIR)
usxa@cca-login1:~> module load grib_api/1.13.1
load jasper 1.900.1 (PATH, MANPATH, JASPER_DIR, JASPER_INCLUDE, JASPER_LIB, JASPER_INCLUDE_DIR,
JASPER_LIB_DIR)
load grib_api 1.13.1 (PATH, MANPATH, GRIB_API_DIR, GRIB_API_VERSION, GRIB_API_INCLUDE, GRIB_API_LIB,
GRIB_API_INCLUDE_DIR, GRIB_API_LIB_DIR)
usxa@cca-login1:~> grib_ls -V
grib_api Version 1.13.1
```

Advanced options

- See what a module would do (without loading it):

```
usxa@cct-login:~> module show emos
-------------------------------------------------------------------
/usr/local/apps/modulefiles/tools_and_libraries/ecmwf/emos/394-r64:

conflict         emos
prepend-path     PATH /usr/local/apps/libemos/000394/CRAY/82/bin
setenv           EMOS_DIR /usr/local/apps/libemos/000394/CRAY/82
setenv           EMOS_VERSION 394
setenv           EMOS_LIB -L/usr/local/apps/libemos/000394/CRAY/82/lib
-lemos.R64.D64.I32
setenv           EMOSLIB -L/usr/local/apps/libemos/000394/CRAY/82/lib -
lemos.R64.D64.I32
setenv           EMOS_LIB_DIR
/usr/local/apps/libemos/000394/CRAY/82/lib
module-whatis    Set environment variables to enable the usage of the
emos 394. This package is integrated with the CrayPE by default.
```

Advanced options II

- Manage the modules loaded by default

```
$> module initlist

$> module initadd package

$> module initprepend package

$> module initrm package

$> module initswitch package1 package2
```

Advanced options III

- Remove ALL ECMWF default modules loaded at the start

```
$> module initclear
```

- This **will not** change the default CRAY modules loaded by default
- If you want to restore the default set of modules:

```
$> ln -sf /usr/local/etc/.modules ~
```

Integration with the Cray Programming Environment

- Cray compiler wrappers (**cc**, **CC** and **ftn**) are heavily affected by modules:
  - The real compiler (Cray, GNU or Intel)
  - The target architecture
  - The compiler flags and libraries used
- The libraries provided by Cray and some ECMWF packages and 3rd party software are also integrated by default with the CrayPE
  - grib_api, emos, eclib, netcdf, nag, gsl …
  - For those, this integration can be disabled with the env vars:
    - EC_CRAYPE_INTEGRATION=on/off
    - <PACKAGE>_CRAYPE_INTEGRATION=on/off

Integration with the Cray Programming Environment

- Be careful when changing PrgEnv...
  - Manually reload all ECMWF modules such as grib_api after the switch.
  - Special flag to show what the wrapper is doing:

    [cc,CC,ftn] -craype-verbose

  - Alternatively, use the helper shell function:

```
$> prgenvswitchto gnu
remove jasper ...
remove grib_api 1.12.3 ...
remove emos ...
switch PrgEnv-cray PrgEnv-gnu
load jasper 1.900.1 ...
load grib_api ...
load emos 394 ...
```

Let's play…

- Start a **fresh** session on cca, and untar the example tarball:

```
$> ssh trcrayXX@cca2
$> tar xvzf ~trx/modules-example.tar.gz
$> cd modules-example
```

- Have a look at the sample program version.c
- Compile with:

```
$> make
```

Did it work? Why?

What do you need to do to build the program?

Let's play again…

- Once compiled, you can run it:

```
$> ./version
GRIB VERSION: 1.12.3
NETCDF VERSION: 4.3.0 of Feb 18 2014 10:07:17 $
```

- What would you do to get the following result:

```
$> ./version
GRIB VERSION: 1.13.1
NETCDF VERSION: 4.3.2 of Oct 16 2014 10:50:25 $
```

Note: to rebuild the program:

```
$> make clean && make
```

Bonus exercise

- Now change the PrgEnv to use the Intel compilers, and rebuild:

```
$> module switch PrgEnv-cray PrgEnv-intel
$> make clean && make
```

Did it work? Why?

What do you need to do to build the program?

Questions?