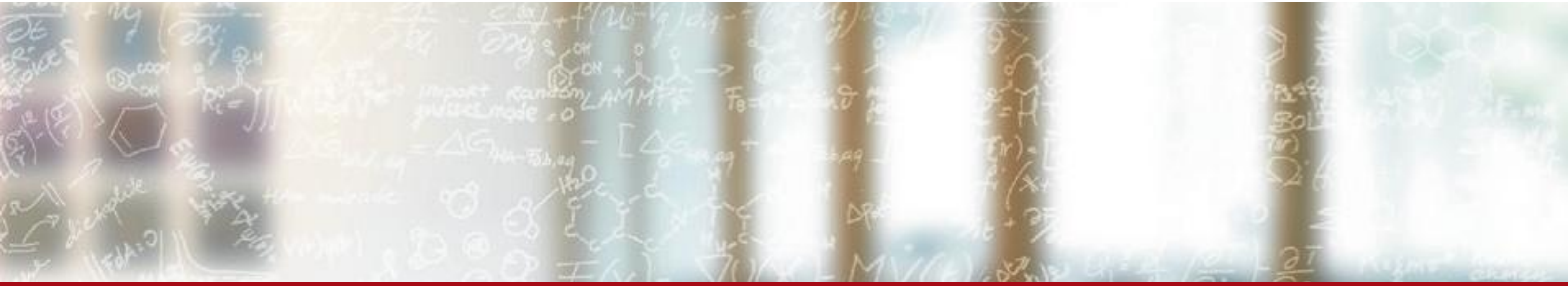
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich



Building and Using HPC Software on Alps: CPE and uenv

CSCS User Day

Ben Cumming
September 2, 2024

Today’s Talk

I will present an overview of how to build and find HPC software on Alps

- Scientific software provided by CSCS on Alps
  - Cray Programming Environment
  - uenv
- uenv introduction
- uenv hands on
  - Getting started: exploring and managing uenv

We won’t cover containers in this presentation, see:
“ML and PyTorch in Containers” by Dr Nicholas Browning at 14:30 today

Nomenclature

- Daint: the new Alps vCluster with GH200 nodes
- Daint-XC: “old daint”
- CPE: Cray Programming Environment

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Cray Programming Environment

CPE: The Cray Programming Environment

HPE/Cray provide a programming environment that is familiar to all users:

- Daint XC - Cray Development Toolkit (CDT)
- Alps - Cray Programming Environment (CPE)

On Alps it is installed in `/opt/cray/`

- Delivered as a collection or RPMs - system engineers select the set of RPMs to install (e.g. do we install PrgEnv-Intel on AMD CPU nodes?)
    - CSCS can choose which RPMs to install, but we don't have much flexibility to modify the packages or their configuration
- Any change requires rebuilding the node image and rebooting nodes
- Vertically integrated
- Released on a 3 month cadence

Users configure the environment using modules

Using the CPE

On Daint-XC the CPE is loaded by default with a default configuration.

On Alps you get a clean environment on login, and load CPE with a module

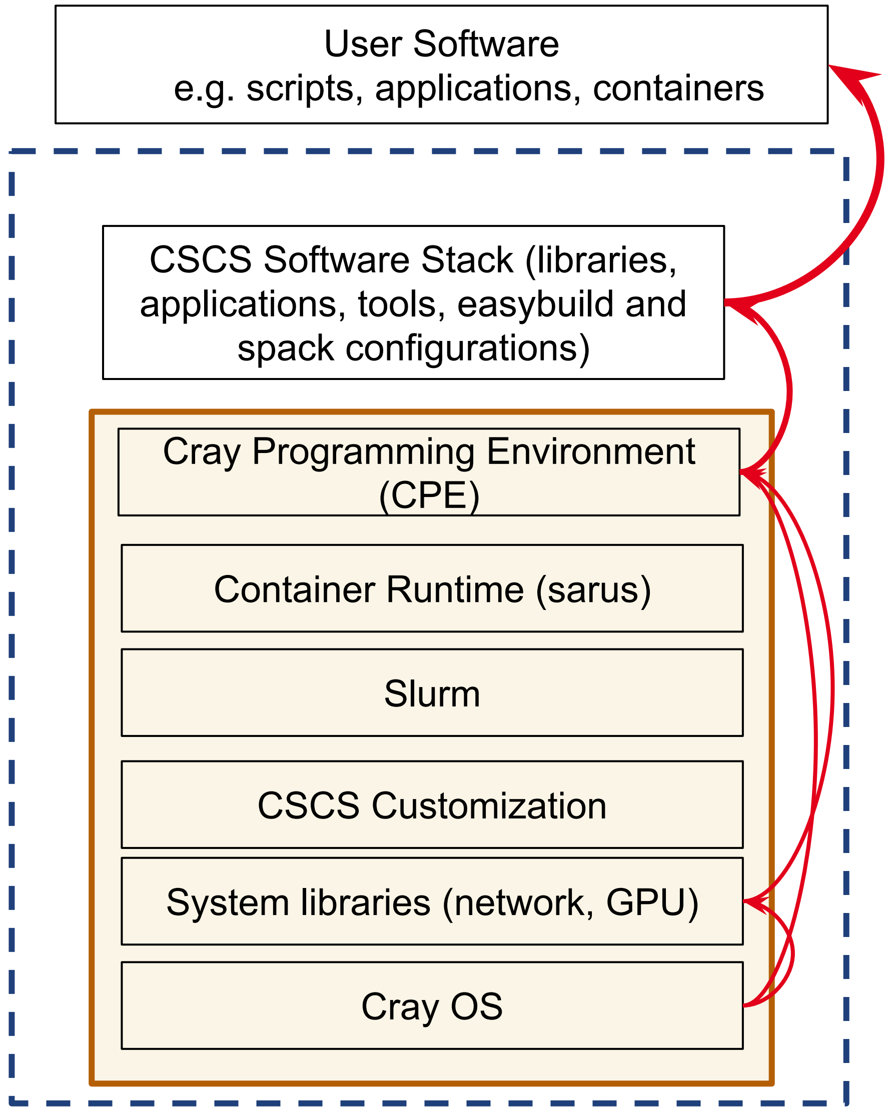
```
>> module list
No modules loaded
>> module avail
--------------------- /etc/cscs-modules ---------------------
   cray/23.12
>> module load cray
>> module list

Currently Loaded Modules:
  1) craype-arm-grace
  2) libfabric/1.15.2.0
  3) craype-network-ofi
  4) xpmem/2.8.2-1.0_5.1__g84a27a5.shasta
  5) PrgEnv-cray/8.5.0
  6) cce/17.0.0
  7) …

>> module avail cray-mpich

---- /opt/cray/pe/lmod/modulefiles/comnet/crayclang/17.0/ofi/1.0 ----
   cray-mpich-abi/8.1.28      cray-mpich/8.1.28 (L)
```

Providing software via CPE is challenging


User Software
e.g. scripts, applications, containers
CSCS Software Stack (libraries, applications, tools, easybuild and spack configurations)
Cray Programming Environment (CPE)
Container Runtime (sarus)
Slurm
CSCS Customization
System libraries (network, GPU)
Cray OS

The Cray Programming Environment is complex by necessity:

- Modules provide a combinatorial set of libraries and tools that serve as many use cases as possible on an increasing number of hardware types.
- Integration is provided by HPE: once an issue is identified HPE have to fix the issue in a future release
    - Long latency between issues reporting and the fix available on Daint.
- Each new release requires extensive testing to check that issues have been fixed
    - And to identify the inevitable new issues

The Cray PE is the best configuration from any vendor in my experience.

These challenges affect all HPC clusters.

Stability vs. Bug Fixes and New Features

Any feedback that in your opinion can help us improve the HPC environment?

Would it be possible to keep older versions?

I am very satisfied with the HPC environment on Piz Daint

Compilers that support the newest C++ standards as well as possible

I need a stabler environment: older versions of the software tools disappear too quickly, which means I have to rebuild my stack every few months.

Please regularly update C++ and CUDA compilers

By providing an environment on CPE it is very difficult to meet all requirements

- Regular updates are required to fix bugs, maintain security and provide updated versions of tools.
- The latest versions of compilers can't be installed before they are packaged by HPE and tested by CSCS.
- It is impractical to maintain:
  - Full stacks on top of more than one CPE
  - More than 2-3 CPE on a system

CPE Support at CSCS

CSCS will no longer provide software built using CPE for users

CPE will still be provided on Daint

- It is provided on vClusters for users who have a hard dependency on Cray tools
  - contact me if you have concerns about dependencies on CPE
- If there is a bug or performance issue with CPE, CSCS will forward the HPE
  - this is a similar level of support to what CSCS provides for CPE in the past

CSCS is focussing on uenv for deploying software on Alps vClusters

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

uenv

uenv are self-contained software stacks

uenv are built for a **(system, uarch)** pair.

There are 5 target microarchitecture (**uarch**):

- `zen2/zen3` CPU-only 2 x AMD Rome/Milan
- `a100` 4 NVIDIA A100 GPUs + 1 AMD CPU socket
- `mi200` 4 AMD Mi250x GPUs + 1 AMD CPU socket
- **`gh200` 4 x GH200**

Each **system** provides some key dependencies

- `libfabric` and `xpmem`: network libraries
- slurm
- customization points for the target audience

uenv are self-contained software stacks

A uenv is two components:

1. A squashfs file
    - *"a read-only file system that lets you compress whole file systems or single directories, write them to ordinary files, and then mount them using a loopback device."* The Linux Documentation Project
    - A single file that contains everything in a working environment
2. Meta data:
    - information about the uenv build (when, where, who)
    - the recipe that was used to build
    - information about the contents of the uenv
    - **environment configurations**

uenv have to be available on the local file system to be used

Store in repository, which is a directory with:

- A database: `index.db`
- A **hashed path** for each uenv that contains
  - the squashfs image `store.squashfs`
  - meta data: `env.json`

```
index.db
images/41fb...49c9/store.squashfs
images/41fb...49c9/meta/configure.json
images/41fb...49c9/meta/env.json
images/41fb...49c9/meta/recipe/
images/95fc...f8d6/meta/configure.json
images/95fc...f8d6/meta/env.json
images/95fc...f8d6/meta/recipe/
images/95fc...f8d6/store.squashfs
```

uenv can be deployed at any time

uenv are a single file, that is mounted dynamically.

There is minimal coupling between each uenv and the underlying OS

1. use cases that need stability can continue to use the same uenv
   - in extreme cases where a low level upgrade breaks the uenv the same recipe can be rebuilt
2. updates and bug fixes can be deployed immediately without affecting old images or rebooting nodes
   - we can deploy the latest version of cray-mpich in a uenv within an hour of it being released

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Hands On

A picture is worth a thousand words

and / or

Watching somebody debug their own tool live is a great way to learn

- follow along in the terminal
- … or watch the live demo
- ask questions!

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Documentation

Uenv documentation

The most accessible docs are available on the command line

```
> uenv --help
> uenv image --help
> uenv image pull --help
# help for the slurm plugin arguments is available via srun/sbatch:
> srun --help | less
```

The CSCS knowledge base has more information:

https://confluence.cscs.ch/x/bYDTKQ

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Future work

we are very busy improving the uenv experience and features

- user-managed uenv images in repos
- community-managed uenv deployment
    - separate repositories with uenv recipes and deployment pipelines
    - managed by communities, e.g. weather and climate
    - deployed directly to vClusters
- storage policies:
    - ensure that uenv are not affected by scratch deletion policies

There are corner cases that we are always discovering, e.g. running heterogenous slurm jobs:

```
srun -n8 -N2 --uenv=cp2k    ./wrap-cp2k.sh : \
     -n4 -N1 --uenv=gromacs ./wrap-gmx.sh
```

please give your feedback

The uenv development team is aware of two types of uenv users:

- those who have simple problems and questions that are fixed quickly
- those who have complicated challenges that take significant effort to understand and fix

We don’t have much information about two key groups

- happy users who don’t need help or give feedback
- unhappy users who suffer in silence.

Please give feedback about what works and what doesn’t

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Questions