compute canada

EasyBuild site presentation: Compute Canada

Bart Oldeman, Maxime Boissonneault, Ryan Taylor
on behalf of
Compute Canada Research Support National Team

compute canada | calcul canada

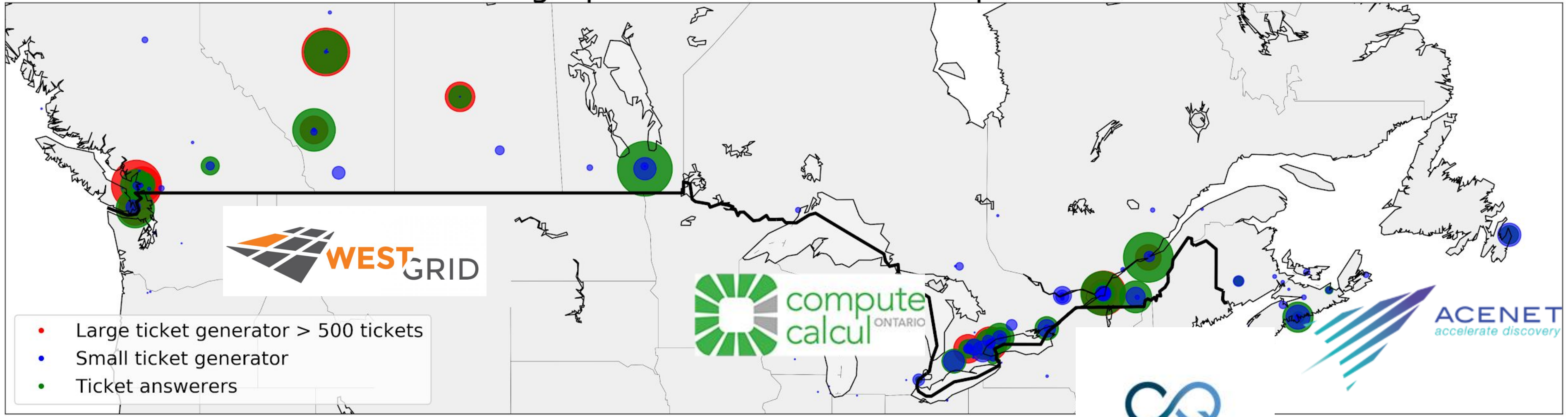
Presentation outline

- Compute Canada, where we were, where we are
- Software installation : Goal and design overview
- Tools used :
  - CVMFS
  - Gentoo ~~Nix~~, EasyBuild
  - Lmod
  - Python
- Monitoring, demo, documentation
- Summary

What is Compute Canada ?

Compute Canada : the people

Network graph of ticket routes Compute Canada

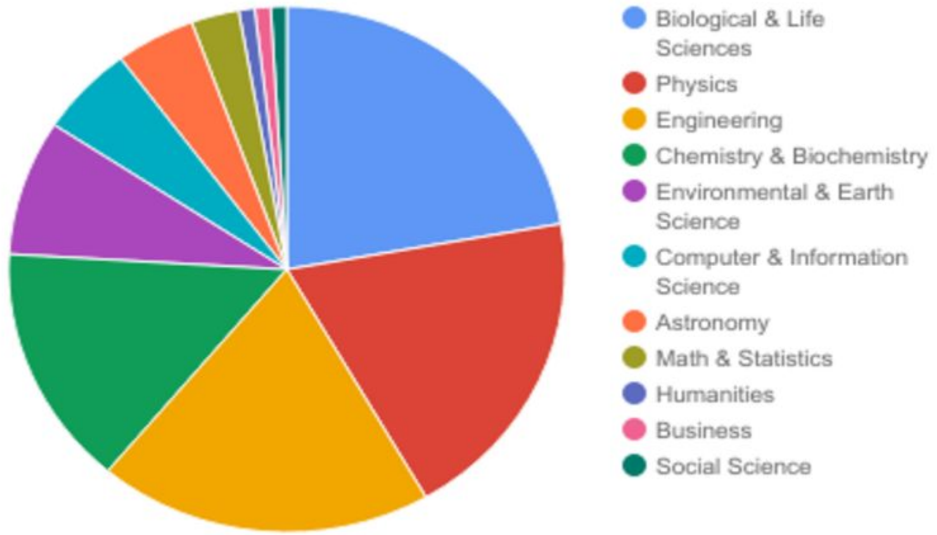


Calcul Québec

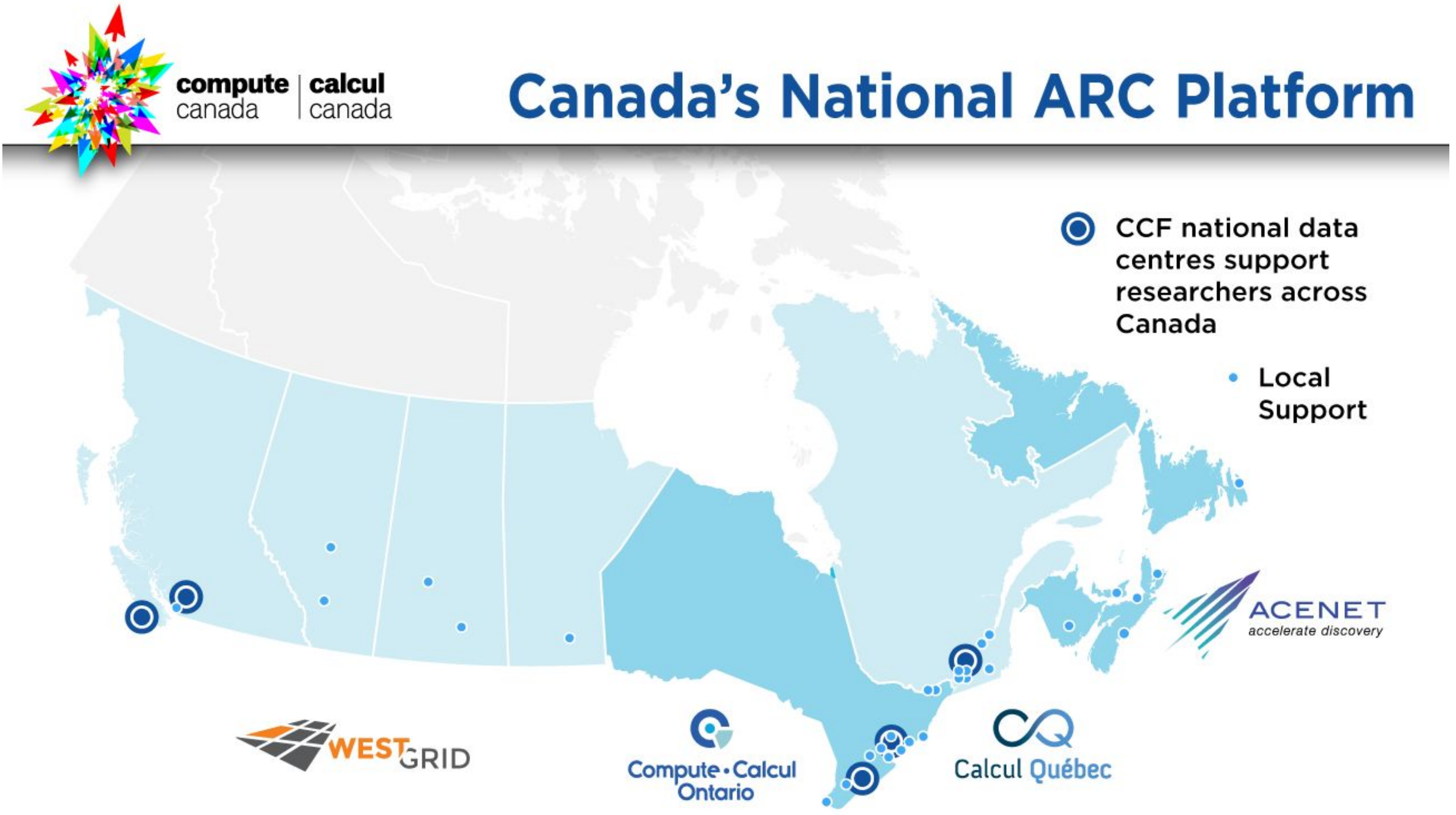
All research disciplines supported

Free access for any researcher at a Canadian institution

- 4 regional consortia
- 35 member institutions
- ~200 technical staff
- ~15,000 user accounts
  - 20% growth per year



Compute Canada : the hardware


compute canada | calcul canada
Canada's National ARC Platform
CCF national data centres support researchers across Canada
Local Support
ACENET accelerate discovery
WESTGRID
Compute • Calcul Ontario
Calcul Québec

5 major national systems
~15 legacy systems
200K cores, 22 PF
70 PB disk, 180 PB tape

System	Type	Network	Production
Arbutus	Cloud	10 GbE	2016 H2
Cedar	General	OPA	2017 H1
Graham	General	EDR IB	2017 H1
Niagara	Large MPI	EDR IB	2018 H1
Béluga	General	EDR IB	2019 H1

compute canada | calcul canada

Goal

Users should be presented with an interface that is as **consistent** and **easy to use** as possible across **all sites**. It should also offer **optimal performance**.

1. All software should be accessible on every site, reliably and performantly.
2. Software should be independent from the underlying OS stack.
3. Software installation should be tracked and reproducible via automation.
4. The user interface should make it easy to use a large and evolving software stack.

compute canada | calcul canada

What this means

All new Compute Canada sites

1. Need a distribution mechanism
   a. CVMFS : CERN Virtual Machine File System

Consistency

2. Independent of the OS (Ubuntu, CentOS, Fedora, etc.)
   a. Gentoo (used to be Nix)
3. Automated installation (humans are not so consistent)
   a. EasyBuild

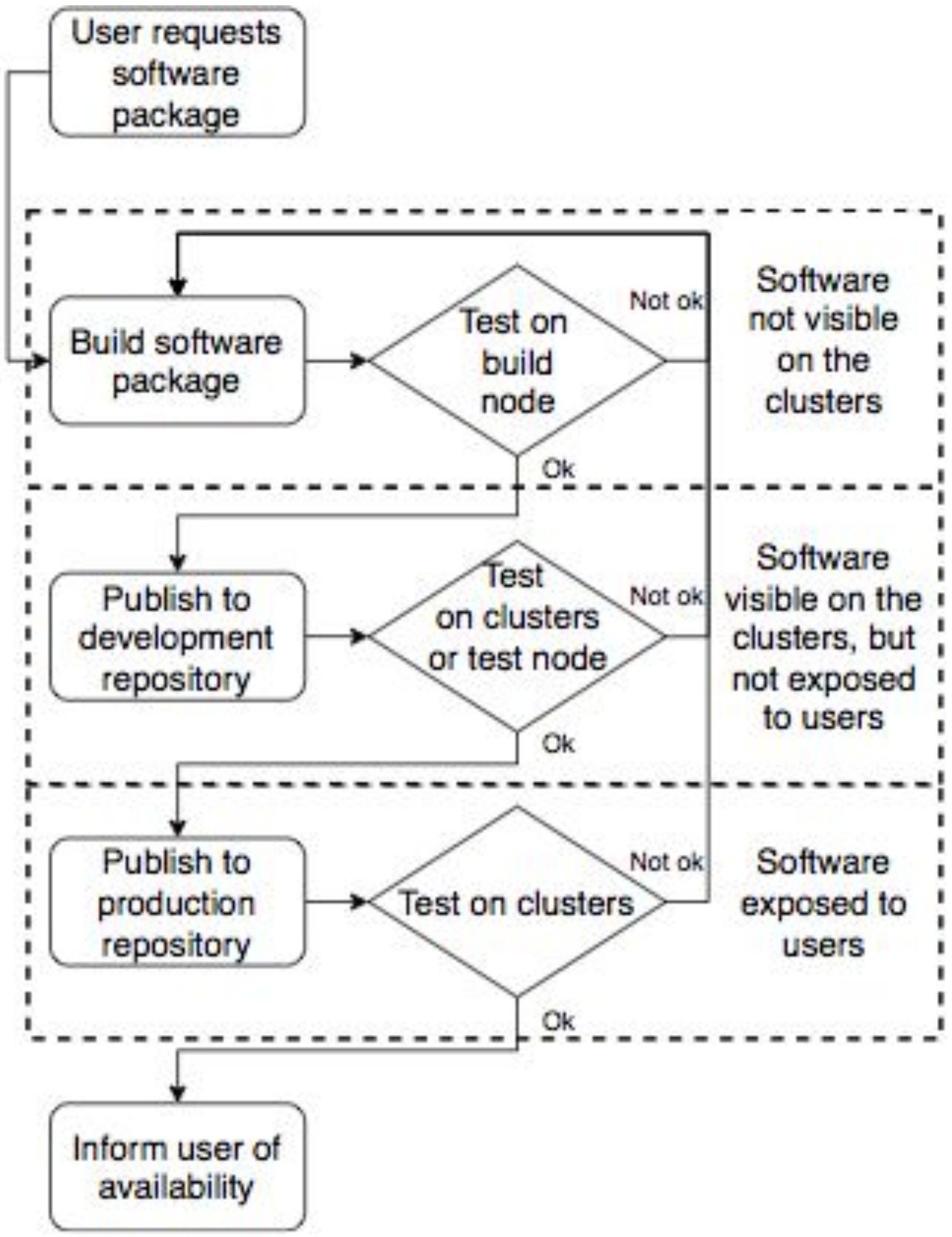
Easy to use

4. Needs a module interface that scale well
   a. Lmod with a hierarchical structure

compute canada | calcul canada

Typical process

1. User requests a software to be installed
2. Staff decides whether it should be installed globally or in the user's account
   a. Globally (default route unless there is a reason not to)
   b. User's account
      i. Custom actions


User requests software package
Build software package
Test on build node
Not ok
Ok
Software not visible on the clusters
Publish to development repository
Test on clusters or test node
Not ok
Ok
Software visible on the clusters, but not exposed to users
Publish to production repository
Test on clusters
Not ok
Ok
Software exposed to users
Inform user of availability

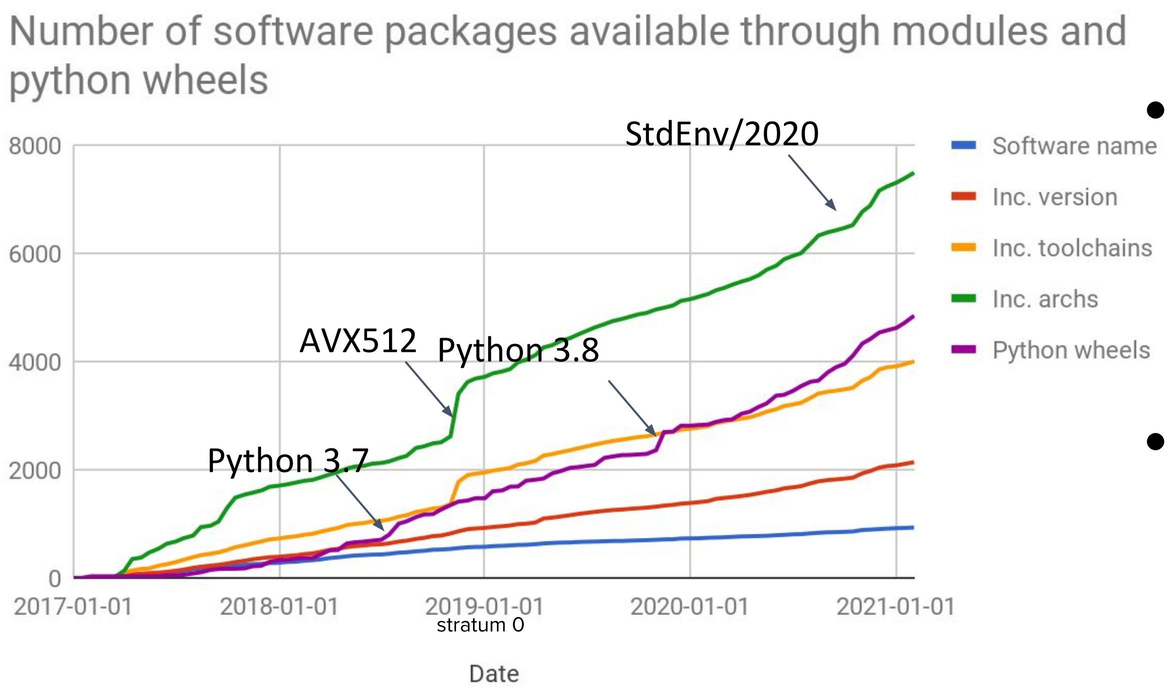
Compute Canada Software Stack

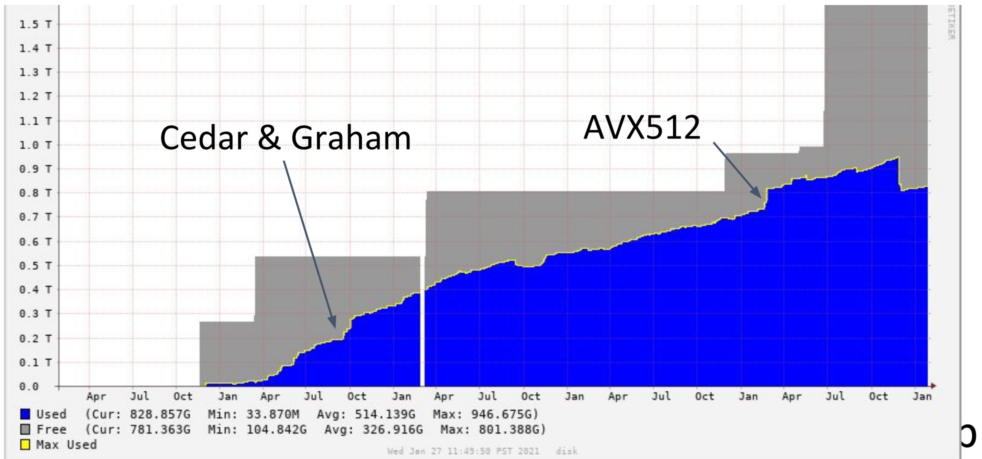
Available software

700+ scientific applications

7,000+ permutations of version/arch/toolchain

Type	Modules
AI	5
Bioinformatics	239
Chemistry	63
Data	19
Geo/Earth	23
Mathematics	82
MPI libraries	7
Physics	48
Various tools	176
Visualisation	28
Misc	38





- Two major new clusters with Skylake CPUs
- Built new modules with AVX512 for most packages
- High deduplication
- Further details

Software: (simpler) design overview

Easybuild layer: modules for Intel, PGI, OpenMPI, CUDA, MKL, high-level applications.
Multiple architectures (sse3, avx, avx2, avx512)
`/cvmfs/soft.computecanada.ca/easybuild/{modules,software}/~~2017~~2020`

Easybuild-generated modules around Nix profiles (GONE):
~~GCC, Eclipse, Qt+Perl+Python no longer~~
~~`/cvmfs/soft.computecanada.ca/nix/var/nix/profiles/[a-z]*`~~

Compatibility: ~~Nix~~ Gentoo Prefix layer: GNU libc, autotools, make, bash, cat, ls, awk, grep, etc.
~~`module nixpkgs/16.09 => $NIXUSER_PROFILE=$EBROOTNIXPKGS=`~~
~~`/cvmfs/soft.computecanada.ca/nix/var/nix/profiles/16.09`~~
```
module gentoo/2020 => $EPREFIX=
/cvmfs/soft.computecanada.ca/gentoo/2020,  $EBROOTGENTOO=$EPREFIX/usr
```

Gray area: ~~Slurm,~~ Lustre client libraries, IB/OmniPath/InfiniPath client libraries (all dependencies of OpenMPI). In Gentoo layer, but can be overridden using PATH & LD_LIBRARY_PATH.

OS kernel, daemons, drivers, libcuda, anything privileged (e.g. the sudo command): always local.
Some legally restricted software too (VASP)

compute canada | calcul canada

Tools used : ~~Nix~~ Gentoo Prefix

- Package, dependency & environment management system

- Builds using bash-like “ebuilds”.

- Used to provide dependencies for scientific applications
  - e.g. glibc, libxml2 (in filter-deps), etc.

- Abstraction layer between the OS and the scientific software stack, using ~~nixpkgs/16.09~~ gentoo/2020 module

- Carries all* the dependencies of scientific software stack

* Exceptions: drivers, kernel modules, etc.

Features of new Gentoo Prefix layer

- Newer versions of almost everything to what was current May 2020 (Gentoo stable 20200504 plus overlay), feels like a Linux distribution upgrade.

- GNU libc: version 2.30 (up from 2.24 -- CentOS-7 has 2.17)
  - Needs at least Linux kernel 3.2: CentOS-6 (kernel 2.6.32) no longer qualifies.
    - CentOS-6 was EOL Nov 30, 2020 so is no longer important.

  - Optimized math functions (exp, log, pow, etc.); older glibc was extremely precise (0.5 ULP) but paid by switching to multi-precision arithmetic when needed. Now at 0.54 ULP but without slowdowns. (https://community.arm.com/developer/tools-software/tools/b/tools-software-ides-blog/posts/update-on-gnu-performance)

- Bash 5.0, Git 2.26.2, Vim 8.2, Emacs 26.2 etc, etc.

compute | calcul
canada | canada

Why Gentoo instead of Nix?

Nix: symlink forest:

```
.../nix/var/nix/profiles/16.09 ->

.../nix/var/nix/profiles/16.09-523-link ->
.../nix/store/cj3f56cgpms7m9fjnbl9vjkmap5fzgsi-user-environment

.../nix/store/cj3f56cgpms7m9fjnbl9vjkmap5fzgsi-user-environment/bin/ls ->
.../nix/store/cn222k5axppndcfbqlckj57939d9h0h9-coreutils-8.25/bin/ls
```

We wrap ld so all rpaths in EB/user code point to $NIXUSER_PROFILE/lib.

Nix components can be upgraded, which changes the store hashes, and allows garbage collect / selective copying.

Sometimes that did not work:

- Python virtualenv: copies the python binary into the virtualenv with store rpaths embedded.
- Qmake: qmake -query QT_INSTALL_BINS /cvmfs/soft.computecanada.ca/nix/store/vxwrgncd38s5prw8qx99rnsfz6lgph52-qtbase-5.6.1-1/bin

Gentoo Prefix : no symlinks, no store leak

compute canada | calcul canada

EasyBuild-generated modules

- https://docs.computecanada.ca/wiki/Standard_software_environments
- module load StdEnv/2016.4, present default on Cedar and Graham
    - Nix + GCC 5.4 + Intel 2016.4 + Open MPI 2.1.1
- module load StdEnv/2018.3, present default on Béluga
    - Nix + GCC 7.3 + Intel 2018.3 + Open MPI 3.1.2
- module load StdEnv/2020 (-> iomkl-2020a + hooks), new default Apr’21
    - Gentoo + GCC 9.3 + Intel 2020.1 + Open MPI 4.0.3
- Multiple x86 architecture flavours: sse3, avx, avx2, avx512, except for “system” toolchain.
- Intel-compiled avx2 binaries are now “fat” binaries and can use avx512 instructions on Skylake+ processors for better performance.

EasyBuild-generated modules (continued)

- Many more modules are now at the “Core” level, compiled using GCCcore-9.3.0 with arch optimizations, e.g. R, Julia, bioinformatics tools. (anything not using MPI, Boost, Fortran, HDF5, FFTW, heavily vectorized)
- Collapsing to GCCcore to Core is possible by using backwards compatible GCC-10 libstdc++, libgfortran, etc from Gentoo layer at runtime.
- Intel MKL also at “Core” level (= all but MPI-FFTW), gcccoremkl toolchain
- Use of RPATH via linker (ld) wrapper to link against libraries from modules (**not** EB’s RPATH support)
- Now uses old-style RPATH instead of RUNPATH, no longer overridable by LD_LIBRARY_PATH; RPATH inherited by run-time plugins.

Caveats of StdEnv/2020

- Fewer modules:
  - about 560 vs 800 different software packages, but catching up.
    - We use module logs to avoid reinstalling software that was not really used
- Newer software is often pickier but still within the specifications:
  - E.g. Open MPI is pickier about tag numbers with UCX, correct memory for one-sided communication.
- Some unresolved issues:
  - Parallel I/O with Open MPI.
  - Need to revalidate the hcoll library to speed up MPI collective communications on clusters with Mellanox IB: it was disabled in 2018.3 because of issues easily reproduced with mpi4py.
  - VirtualGL.

Python extensions with multi_deps

- Heavy multi_deps usage:
  - If python support exists, we install with Python 3.6, 3.7, 3.8 (and sometimes 2.7)
  - Examples:
    - Boost, GEOS, QGIS, thrift, VTK, ParaView, arrow, NLopt, cram, OpenCV, ITK, RDKit,...
- Extensions in the main package
  - HDF5 includes h5py, tables
  - GDAL includes pygdal
  - Qt5 includes pyqt
  - PETSc includes petsc4py
  - mariadb includes mysql-connector-python, PyMySQL, and (Perl) DBD:mysql
  - PLUMED includes plumed (python)
  - PostgreSQL includes psycopg2
  - igraph includes python-igraph
  - Bullet includes pybullet
  - wxWidgets includes wxPython
  - ...

Python wheels

What are wheels?

Wheels are the new standard of Python distribution and are intended to replace eggs. Support is offered in `pip >= 1.4` and `setuptools >= 0.8`.

Advantages of wheels

1. Faster installation for pure Python and native C extension packages.
2. Avoids arbitrary code execution for installation. (Avoids setup.py)
3. Installation of a C extension does not require a compiler on Linux, Windows or macOS.
4. Allows better caching for testing and continuous integration.
5. Creates .pyc files as part of installation to ensure they match the Python interpreter used.
6. More consistent installs across platforms and machines.
7. **You can compile your own wheels, linking against your compiled libraries**

Our supported wheels

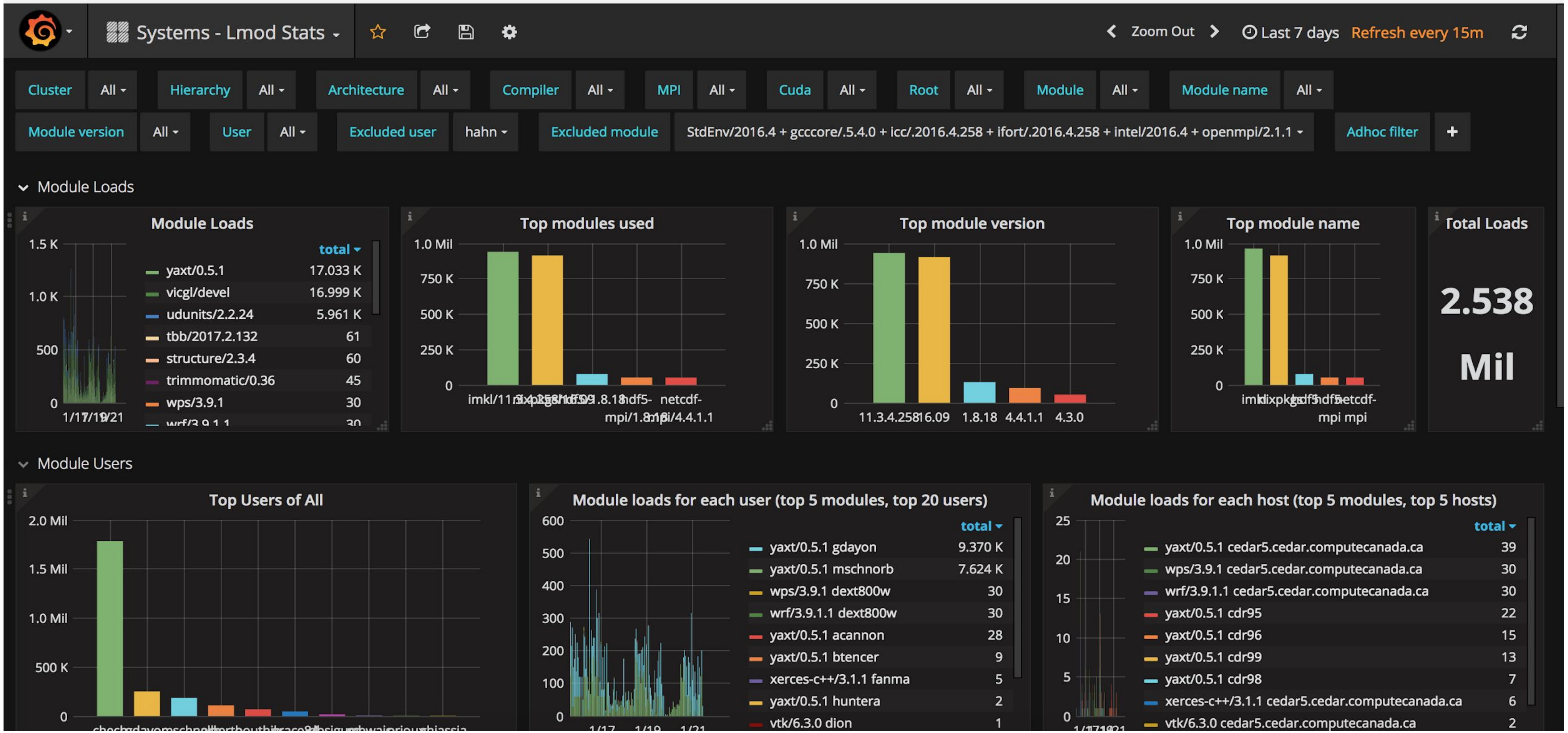
```
$ ls /cvmfs/soft.computecanada.ca/custom/python/wheelhouse/*/* | wc -w
8506
$ avail_wheels tensorflow_cpu
name             version     build          python    arch
--------------   ---------   -------------  --------  ------
tensorflow_cpu   2.3.0             cp38   generic
tensorflow_cpu   2.3.0             cp37   generic
tensorflow_cpu   2.3.0             cp36   generic
$ avail_wheels tensorflow_gpu
...
```

- https://docs.computecanada.ca/wiki/Available_wheels

What modules are our users using ?

- Every “module load” command is sent to syslogs
- Syslogs for all Compute Canada clusters are aggregated into an Elastic Search engine
- Grafana is used to produce dashboards of module usage

Module usage dashboard



Systems - Lmod Stats

Zoom Out | Last 7 days | Refresh every 15m

Cluster: All | Hierarchy: All | Architecture: All | Compiler: All | MPI: All | Cuda: All | Root: All | Module: All | Module name: All

Module version: All | User: All | Excluded user: hahn | Excluded module: StdEnv/2016.4 + gcccore/.5.4.0 + icc/.2016.4.258 + ifort/.2016.4.258 + intel/2016.4 + openmpi/2.1.1 | Adhoc filter

Module Loads

Module Loads

	total
yaxt/0.5.1	17.033 K
vicgl/devel	16.999 K
udunits/2.2.24	5.961 K
tbb/2017.2.132	61
structure/2.3.4	60
trimmomatic/0.36	45
wps/3.9.1	30
wrf/3.9.1.1	30

Top modules used

Top module version

Top module name

Total Loads

2.538 Mil

Module Users

Top Users of All

Module loads for each user (top 5 modules, top 20 users)

	total
yaxt/0.5.1 gdayon	9.370 K
yaxt/0.5.1 mschnorb	7.624 K
wps/3.9.1 dext800w	30
wrf/3.9.1.1 dext800w	30
yaxt/0.5.1 acannon	28
yaxt/0.5.1 btencer	9
xerces-c++/3.1.1 fanma	5
yaxt/0.5.1 huntera	2
vtk/6.3.0 dion	1

Module loads for each host (top 5 modules, top 5 hosts)

	total
yaxt/0.5.1 cedar5.cedar.computecanada.ca	39
wps/3.9.1 cedar5.cedar.computecanada.ca	30
wrf/3.9.1.1 cedar5.cedar.computecanada.ca	30
yaxt/0.5.1 cdr95	22
yaxt/0.5.1 cdr96	15
yaxt/0.5.1 cdr99	13
yaxt/0.5.1 cdr98	7
xerces-c++/3.1.1 cedar5.cedar.computecanada.ca	6
vtk/6.3.0 cedar5.cedar.computecanada.ca	2

Documentation, resources

- List of modules
  - https://docs.computecanada.ca/wiki/Available_software
- List of Python wheels
  - https://docs.computecanada.ca/wiki/Available_wheels
- Technical documentation
  - https://github.com/ComputeCanada/software-stack/

You can use this too

- Mounting our software stack
  - https://docs.computecanada.ca/wiki/Accessing_CVMFS

Questions ?