



OpenHPC: Community Building Blocks for HPC

Karl W. Schulz, Ph.D.

Research Associate Professor

Oden Institute for Computational Engineering and Sciences and Dell Medical School



HPC Knowledge Meeting

hpcpk'19 ♦ June 13, 2019 ♦ Barcelona, Spain



openHPC
<https://openhpc.community>

Outline

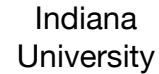
- What is OpenHPC: project overview
 - mission/vision
 - members, governance, adoption
- More technical details
 - key takeaways
 - packaging conventions
 - build/test environment
 - highlights from latest release and what's coming

What is the project's mission and vision?

Mission: to provide a reference collection of **open-source HPC software** components and best practices, lowering barriers to deployment, advancement, and use of modern HPC methods and tools.

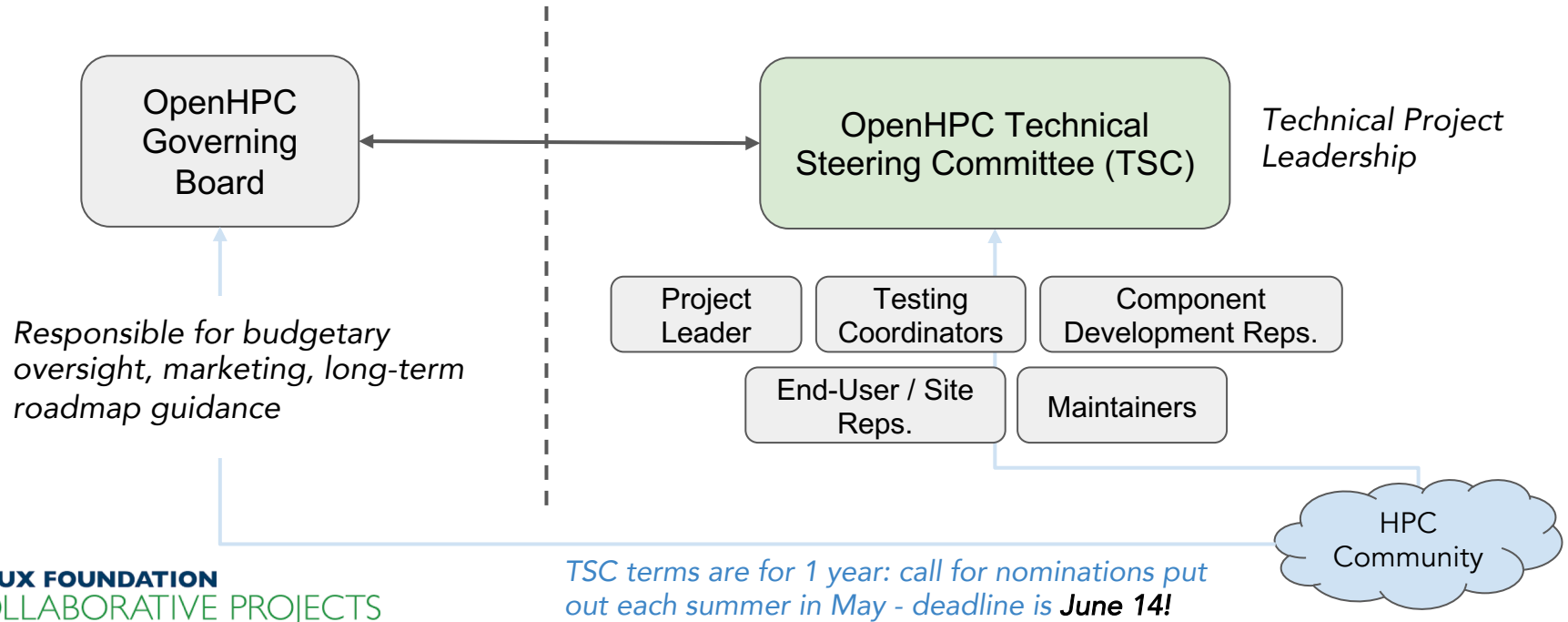
Vision: OpenHPC components and best practices will enable and accelerate innovation and discoveries by broadening access to state-of-the-art, open-source HPC methods and tools in a consistent environment, supported by a collaborative, worldwide community of HPC **users, developers, researchers, administrators**, and **vendors**.

OpenHPC: Current Project Members



How is the community project governed?

Governance is dual-pronged with a Governing Board + Technical Steering Committee

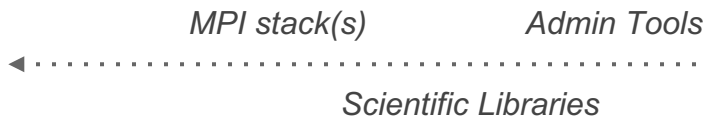


Next: High-level Takeaways
for OpenHPC...

OpenHPC: a building block repository

[Key takeaway]

- OpenHPC provides a collection of pre-built ingredients common in HPC environments; fundamentally it is a software repository
- The repository is published for use with Linux distro package managers:
 - yum (CentOS/RHEL)
 - zypper (SLES)
- You can pick relevant bits of interest for your site
 - if you prefer a resource manager that is not included, you can build that locally and still leverage the scientific libraries and development environment
 - similarly, you might prefer to utilize a different provisioning system
 - *public package repositories also makes it easy to include desired elements in customized containers (e.g. Docker, Singularity, Charliecloud)*



OpenHPC: A Hierarchical HPC Software Environment

[Key takeaway]

- Designation of compiler/MPI variants is **fundamental** to OpenHPC's build, test, and end-user environment
- OpenHPC packaging is architected to allow multiple variants to be added over time
- example: two gcc variants are installed

```
$ module list
Currently Loaded Modules:
  1) autotools  2) prun/1.2  3) gnu7/7.3.0  4) mvapich2/2.2  5) ohpc

$ module avail

----- /opt/ohpc/pub/moduledeps/gnu7-mvapich2 ----- [ MPI tier ]
adios/1.13.1  mpiP/3.4.1  phdf5/1.10.2  py3-scipy/1.1.0  superlu_dist/5.3.0
boost/1.67.0  mumps/5.1.2  pnetcdf/1.9.0  scalapack/2.0.2  tau/2.27.1
fftw/3.3.7  netcdf-cxx/4.3.0  ptscotch/6.0.4  scalasca/2.3.1  trilinos/12.12.1
hypr/2.14.0  netcdf-fortran/4.4.4  py2-mpi4py/3.0.0  scorep/4.0
imb/2018.1  netcdf/4.6.1  py2-scipy/1.1.0  sionlib/1.7.1
mfem/3.3.2  petsc/3.9.1  py3-mpi4py/3.0.0  slepc/3.9.1

----- /opt/ohpc/pub/moduledeps/gnu7 ----- [ compiler tier ]
R/3.5.0  likwid/4.3.2  mvapich2/2.2 (L)  openmpi3/3.1.0  py2-numpy/1.14.3  superlu/5.2.1
gs1/2.4  metis/5.1.0  ocr/1.0.1  pdttoolkit/3.25  py3-numpy/1.14.3
hdf5/1.10.2  mpich/3.2.1  openblas/0.2.20  plasma/2.8.0  scotch/6.0.4

----- /opt/ohpc/pub/modulefiles -----
EasyBuild/3.6.1  clustershell/1.8  gnu7/7.3.0 (L)  ohpc (L)  prun/1.2 (L)
autotools (L)  cmake/3.11.1  hwloc/1.11.10  papi/5.6.0  singularity/2.5.1
charliecloud/0.2.4  gnu/5.4.0  llvms/5.0.1  pmix/2.1.1  valgrind/3.13.0
```

Quick (hierarchical) installation example

- Once an OpenHPC repo is enabled, package installation follows standard package manager semantics
- Package interdependencies maintained within RPMs
- Consider example install of PETSc for gnu7/mvapich2 on bare bones head node

```
# yum install petsc-gnu7-mvapich2-ohpc
```

```
...
```

```
=====
```

Package	Arch	Version	Repository	Size
Installing:				
<code>petsc-gnu7-mvapich2-ohpc</code>	x86_64	3.7.6-104.1	OpenHPC-updates	13 M
Installing for dependencies:				
<code>gnu7-compilers-ohpc</code>	x86_64	7.1.0-29.3	OpenHPC-updates	51 M
<code>mvapich2-gnu7-ohpc</code>	x86_64	2.2-26.2	OpenHPC-updates	8.6 M
<code>openblas-gnu7-ohpc</code>	x86_64	0.2.19-25.1	OpenHPC-updates	3.4 M
<code>phdf5-gnu7-mvapich2-ohpc</code>	x86_64	1.10.0-83.1	OpenHPC-updates	2.9 M
<code>prun-ohpc</code>	noarch	1.1-21.1	OpenHPC	9.5 k
<code>scalapack-gnu7-mvapich2-ohpc</code>	x86_64	2.0.2-39.1	OpenHPC-updates	3.4 M

```
=====
```

```
Transaction Summary
```

```
=====
```

OpenHPC: validated recipes for system install

[Key takeaway]

- In addition to being a package repository, OpenHPC provides validated recipes for bare-metal system installs
- Recipes organized by OS, architecture, and key administrative components
- **10** different recipes available with latest release
- the **docs-ohpc** RPM installs these recipes (along with shell scripts encapsuating all commands)

CentOS 7

SLES12

Warewulf

xCAT

PBS Pro

SLURM

x86_64:

- [Install_guide-CentOS7-Warewulf-PBSPro-1.3.8-x86_64.pdf](#)
- [Install_guide-CentOS7-Warewulf-SLURM-1.3.8-x86_64.pdf](#)
- [Install_guide-CentOS7-xCAT-Stateful-SLURM-1.3.8-x86_64.pdf](#)
- [Install_guide-CentOS7-xCAT-Stateless-SLURM-1.3.8-x86_64.pdf](#)
- [Install_guide-SLE_12-Warewulf-PBSPro-1.3.8-x86_64.pdf](#)
- [Install_guide-SLE_12-Warewulf-SLURM-1.3.8-x86_64.pdf](#)

aarch64:

- [Install_guide-CentOS7-Warewulf-PBSPro-1.3.8-aarch64.pdf](#)
- [Install_guide-CentOS7-Warewulf-SLURM-1.3.8-aarch64.pdf](#)
- [Install_guide-SLE_12-Warewulf-PBSPro-1.3.8-aarch64.pdf](#)
- [Install_guide-SLE_12-Warewulf-SLURM-1.3.8-aarch64.pdf](#)

excellent place to start if you are new to the project

OpenHPC: Multiple Architecture Support

```
[train01@sms001 ~]$ module avail
```

```
x86_64
```

```
----- /opt/ohpc/pub/moduledeps/gnu7-mpich -----  
adios/1.11.0  mpiP/3.4.1      petsc/3.7.6      scorep/3.0  
boost/1.63.0  mumps/5.1.1    phdf5/1.10.0    sionlib/1.7.1  
fftw/3.3.6   netcdf-cxx/4.3.0  scalapack/2.0.2  superlu_dist/4.2  
hypre/2.11.1 netcdf-fortran/4.4.4  scalasca/2.3.1  tau/2.26.1  
imb/4.1      netcdf/4.4.1.1  scipy/0.19.0    trilinos/12.10.1  
  
----- /opt/ohpc/pub/moduledeps/gnu7 -----  
R/3.3.3      metis/5.1.0    numpy/1.12.1    openmpi/1.10.7  
gsl/2.3      mpich/3.2 (L)  ocr/1.0.1      pdtoolkit/3.23  
 hdf5/1.10.0 mvapich2/2.2  openblas/0.2.19  superlu/5.2.1
```

```
EasyBuild/3.2.1  
autotools (L)
```

```
train01@cavium1:~> module avail
```

```
aarch64
```

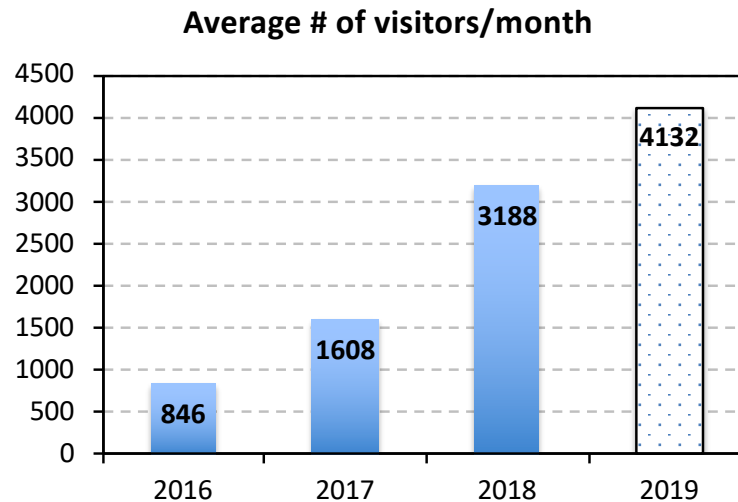
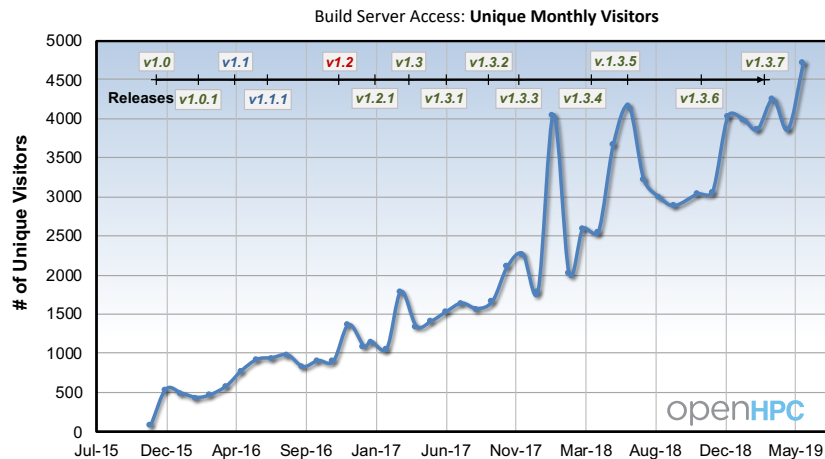
```
----- /opt/ohpc/pub/moduledeps/gnu7-mpich -----  
adios/1.11.0  mpiP/3.4.1      petsc/3.7.6      scorep/3.0  
boost/1.63.0  mumps/5.1.1    phdf5/1.10.0    sionlib/1.7.1  
fftw/3.3.6   netcdf-cxx/4.3.0  scalapack/2.0.2  superlu_dist/4.2  
hypre/2.11.1 netcdf-fortran/4.4.4  scalasca/2.3.1  tau/2.26.1  
imb/4.1      netcdf/4.4.1.1  scipy/0.19.0    trilinos/12.10.1  
  
----- /opt/ohpc/pub/moduledeps/gnu7 -----  
R/3.3.3      metis/5.1.0    ocr/1.0.1      pdtoolkit/3.23  
gsl/2.3      mpich/3.2 (L)  openblas/0.2.19  superlu/5.2.1  
 hdf5/1.10.0 numpy/1.12.1    openmpi/1.10.7  
  
----- /opt/ohpc/pub/modulefiles -----  
EasyBuild/3.2.1  hwloc/1.11.6  singularity/2.3  
autotools (L)   ohpc (L)      valgrind/3.12.0
```

[Key takeaway]

- aarch64 and x86_64 package repositories are available
- benefit is that OpenHPC provides consistent development environment to the end user across multiple architectures

Is anybody using this...?

Project Adoption Growth



- Continued access/download growth since initial release at SC'15
- Plots highlight number of unique visitors/month to the OpenHPC build server/repo(s)
- Over 21TB downloaded in 2018

Additional Technical Details...

More comments on:

- packaging conventions
- infrastructure
- build and test environment
- current release

Packaging Conventions

- OpenHPC tries to play nicely with underlying OS'es and other repositories
- We endeavor to avoid namespace conflict in a variety of ways:

- install paths: end-user oriented components housed under `/opt/ohpc/pub` (also allow for multiple versions to coexist)
- package names: RPMs provided via OpenHPC repos include `"-ohpc"` as the suffix for the package name, for example:

```
lmod-ohpc-7.4.8-11.1.aarch64.rpm
```

- dependencies: compiler/MPI variant dependencies are **explicitly** managed

```
%if "%{compiler_family}" == "gnu7"  
BuildRequires: gnu7-compilers%{PROJ_DELIM} >= 7.2.0  
Requires:      gnu7-compilers%{PROJ_DELIM} >= 7.2.0  
%endif  
%if "%{mpi_family}" == "mvapich2"  
BuildRequires: mvapich2-%{compiler_family}%{PROJ_DELIM}  
Requires:      mvapich2-%{compiler_family}%{PROJ_DELIM}  
%endif
```

[relevant logic from OHPC_macros]

Packaging Conventions

- Packaging adopts a consistent environment variable schema in modulefiles (we rely on `Lmod` implementation of modules):
 - `${PACKAGE_DIR}` - top level install path of package
 - `${PACKAGE_BIN}` - path to binaries supplied by package
 - `${PACKAGE_INC}` - path to include headers for package
 - `${PACKAGE_LIB}` - path to dynamic libraries for package
- Recommend using these vars in build scripts, Makefiles, and interactive execution
- Let's consider an end user interaction example with previous PETSc install: assume we are a user wanting to build a PETSC hello world in C

```
$ module load gnu8 mvapich2
$ module load petsc
$ mpicc -I${PETSC_INC} petsc_hello.c -L${PETSC_LIB} -lpetsc
```

Packaging conventions: custom ohpc RPM plugin

To avoid namespace collision for resolving dynamic libraries (.so's), we apply an "ohpc" color delimiter to libraries installed in `/opt/ohpc/` path

```
# rpm -q --requires gnu7-compilers-ohpc | egrep "libc.so|libgc"
```

.so's self contained
within ohpc gcc build



```
libgcc_s.so.1() (64bit) (ohpc)
libgcc_s.so.1(GCC_3.0) (64bit) (ohpc)
libgcc_s.so.1(GCC_3.3) (64bit) (ohpc)
libgcc_s.so.1(GCC_4.2.0) (64bit) (ohpc)
libgcc_s.so.1(GCC_4.3.0) (64bit) (ohpc)
```

.so's required from base
OS supplied packages



```
libc.so.6() (64bit)
libc.so.6(GLIBC_2.10) (64bit)
libc.so.6(GLIBC_2.11) (64bit)
libc.so.6(GLIBC_2.14) (64bit)
libc.so.6(GLIBC_2.16) (64bit)
libc.so.6(GLIBC_2.17) (64bit)
libc.so.6(GLIBC_2.2.5) (64bit)
libc.so.6(GLIBC_2.3) (64bit)
libc.so.6(GLIBC_2.3.2) (64bit)
libc.so.6(GLIBC_2.3.3) (64bit)
libc.so.6(GLIBC_2.6) (64bit)
libc.so.6(GLIBC_2.7) (64bit)
```

custom RPM
dependency analysis
using ohpc plugin

Convention introduced
in v1.3.4 release

OpenHPC Development Infrastructure

What are we using to get the job done....?

The usual software engineering stuff:

- GitHub (SCM and issue tracking/planning)
- Continuous Integration (CI) Testing (Jenkins)
- Documentation (Latex)
- Bats (unit testing for Bash)

Capable build/packaging system:

- Require flexible system to manage builds for multiple distros, multiple compiler/MPI family combinations, and dependencies across packages
- Have engineered a system using Open Build Service (OBS) which is supported by back-end git
 - git houses .spec files, tarballs, patches, documentation recipes, and integration tests
 - OBS performs automated builds and dependency analysis



<https://github.com/openhpc/ohpc>



<https://build.openhpc.community>



L^AT_EX

Reproducible builds using OBS

<https://build.openhpc.community>

obs

OpenHPC Build Service > Projects > OpenHPC:1.3:Update6:Factory > Overview Log In

Overview Repositories Monitor Requests Users Subprojects Advanced

1.3.6 - Update #6 rolling development build 2 Release Targets is locked

No description set

Packages (248)

Show 25 entries Search:

R

adios-gnu-impi

adios-gnu-mpich

adios-gnu-mvapich2

adios-gnu-openmpi

automake

boost-gnu-mpi **child package(s)**

boost-gnu-mpich

boost-gnu-mvapich2

boost-gnu-openmpi **parent package**

boost-intel-impi

boost-intel-mpich

boost-intel-mvapich2

boost-intel-openmpi

charliecloud

cmake

conman

dimemas-gnu-impi

dimemas-gnu-mpich

dimemas-gnu-mvapich2

dimemas-gnu-openmpi

Build Results

CentOS_7	aarch64	disabled: 147 locked: 101
	x86_64	locked: 248
SLE_12	aarch64	disabled: 148 locked: 100
	x86_64	locked: 248

[OBS _link file]

```
<link project='OpenHPC:1.3:Update6:Factory'  
package='boost-gnu-openmpi '>  
<patches>  
  <topadd>%define compiler_family gnu8</topadd>  
  <topadd>%define mpi_family mpich</topadd>  
</patches>  
</link>
```

- We use the **Open Build Service** (OBS) to manage build processes
- OBS can drive builds for multiple OS's and architectures
- Repeatable builds carried out in chroot environment
- **Generates binary and src rpms**
- Client/server architecture supports distributed build slaves and multiple architectures
- OpenHPC adopts a **single** .spec file input for builds
- Dependencies are self-managed and designed to avoid namespace conflict with distro provided variants

Build system - distributed build servers

- We leverage ability in OBS to have distributed build servers:
 - x86_64: hosted at TACC (thanks to kind hardware donations) and on EC2
 - aarch64: hosted at Packet (thanks to Works on Arm project)

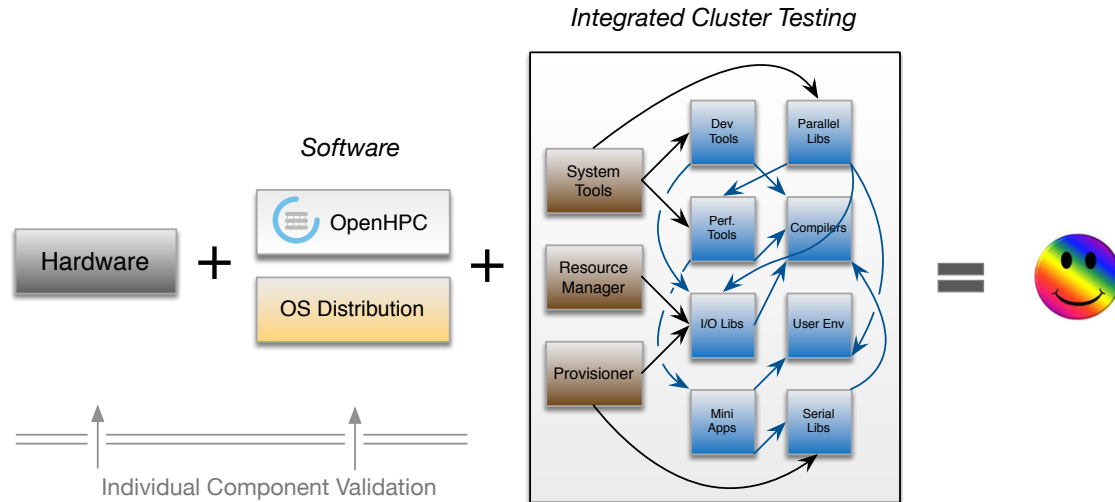
The screenshot shows the 'Server Status' page for openHPC. At the top, it lists server architectures: aarch64, i586, x86_64, dispatcher, publisher, signer, and warden. Below this is the 'Workers' section, which displays a grid of jobs. A callout bubble points to the 'scorep-intel-mvapich2' job on the 'west-2.compute.internal (x86_64)' worker, with the text 'Thanks to the Works on arm project.' The dashboard also includes logos for TACC, Amazon EC2, and Packet.

Worker	Job	Progress
arm01-packet (aarch64)	test-suite	100%
arm01-packet (aarch64)	mfem-gnu-mpich	100%
arm01-packet (aarch64)	extrae-gnu-openmpi	100%
arm01-packet (aarch64)	warewulf-common	100%
arm01-packet (aarch64)	arm-compilers-devel	100%
arm01-packet (aarch64)	warewulf-cluster	100%
arm01-packet (aarch64)	boost-gnu-mpich	100%
arm01-packet (aarch64)	fftw-gnu-openmpi	100%
arm01-packet (aarch64)	nagios	100%
arm01-packet (aarch64)	cmake	100%
arm01-packet (aarch64)	mfem-gnu-mpich	100%
arm01-packet (aarch64)	mpich-arm	100%
arm01-packet (aarch64)	petsc-gnu-mpich	100%
arm01-packet (aarch64)	boost-gnu-openmpi	100%
arm01-packet (aarch64)	meta-packages	100%
arm01-packet (aarch64)	fftw-gnu-mpich	100%
arm01-packet (aarch64)	nagios	100%
arm01-packet (aarch64)	automake	100%
arm01-packet (aarch64)	warewulf-cluster	100%
arm01-packet (aarch64)	fftw-gnu-mpich	100%
arm01-packet (aarch64)	meta-packages	100%
arm01-packet (aarch64)	conman	100%
arm01-packet (aarch64)	extrae-gnu-mpich	100%
arm01-packet (aarch64)	warewulf-common	100%
arm01-packet (aarch64)	boost-gnu-openmpi	100%
arm01-packet (aarch64)	petsc-gnu-mpich	100%
arm01-packet (aarch64)	gnu-compilers	100%
arm01-packet (aarch64)	boost-gnu-mpich	100%
arm01-packet (aarch64)	conman	100%
arm01-packet (aarch64)	idle	100%
arm01-packet (aarch64)	idle	100%
arm01-packet (aarch64)	idle	100%
ip-172-31-17-199.us-west-2.compute.internal (x86_64)	scorep-intel-mvapich2	100%
ip-172-31-17-199.us-west-2.compute.internal (x86_64)	scorep-intel-mvapich2	100%
ip-172-31-17-199.us-west-2.compute.internal (x86_64)	geomp-intel-mpich	100%
linux-1pyh (aarch64)	gnu-compilers	100%
linux-1pyh (aarch64)	cmake	100%
linux-1pyh (aarch64)	fftw-gnu-openmpi	100%
linux-1pyh (aarch64)	idle	100%
ohpc1 (x86_64)	mfem-intel-mpich	100%
ohpc1 (x86_64)	mfem-gnu-mpich	100%
ohpc1 (x86_64)	scorep-intel-openmpi	100%
ohpc1 (x86_64)	scorep-intel-mpich	100%
ohpc1 (x86_64)	scalasca-intel-mpich	100%
ohpc1 (x86_64)	ohpc-filessystem	100%
ohpc1 (x86_64)	scalasca-intel-mvapich2	100%
ohpc1 (x86_64)	scorep-intel-impi	100%
ohpc1 (x86_64)	scorep-intel-openmpi	100%
ohpc1 (x86_64)	mfem-gnu-mvapich2	100%
ohpc1 (x86_64)	scorep-intel-impi	100%
ohpc1 (x86_64)	scorep-intel-mpich	100%
ohpc1 (x86_64)	scalasca-gnu-impi	100%
ohpc1 (x86_64)	scalasca-gnu-impi	100%
ohpc1 (x86_64)	mfem-intel-impi	100%
ohpc1 (x86_64)	mfem-intel-mvapich2	100%
ohpc1 (x86_64)	scalapack-intel-mpich	100%
ohpc1 (x86_64)	scalasca-intel-mpich	100%

Integration/Test/Validation

Post-build testing is another **key element** for us and the intent is to build upon existing component-level validation with targeted cluster-validation and scaling initiatives including:

- install recipes
- cross-package interaction
- development environment
- mimic use cases common in HPC deployments
- upgrade mechanism



Post Install Integration Tests - Overview

Example ./configure output (non-root)

Global testing harness includes a number of embedded subcomponents:

- major components have configuration options to enable/disable
- end user tests need to touch all of the supported compiler and MPI families
- we abstract this to repeat the tests with different compiler/MPI environments:
 - gcc/Intel compiler toolchains
 - Intel, OpenMPI, MPICH, MVAPICH2 MPI families

installable as **test-suite-ohpc RPM**

```
Package version..... : test-suite-1.0.0
Build user..... : jilluser
Build host..... : master4-centos71.localdomain
Configure date..... : 2015-10-26 09:23
Build architecture..... : x86_64-unknown-linux-gnu
Test suite configuration..... : long
```

Submodule Configuration:

User Environment:

```
RMS test harness..... : enabled
Munge..... : enabled
Apps..... : enabled
Compilers..... : enabled
MPI..... : enabled
HSN..... : enabled
Modules..... : enabled
OOM..... : enabled
```

Dev Tools:

```
Valgrind..... : enabled
R base package..... : enabled
TBB..... : enabled
CILK..... : enabled
```

Performance Tools:

```
mpiP Profiler..... : enabled
Papi..... : enabled
PETSc..... : enabled
TAU..... : enabled
```

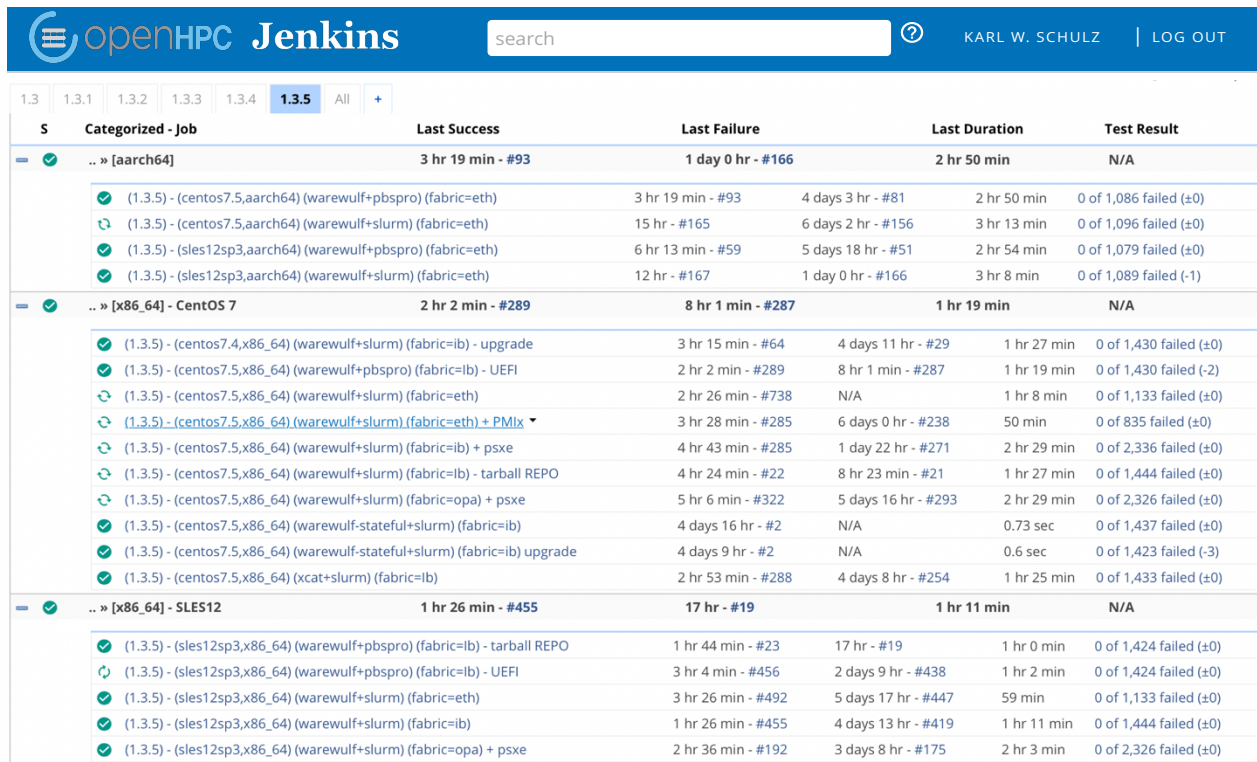
Libraries:

```
Adios ..... : enabled
Boost ..... : enabled
Boost MPI..... : enabled
FFTW..... : enabled
GSL..... : enabled
HDF5..... : enabled
HYPRE..... : enabled
IMB..... : enabled
Metis..... : enabled
MUMPS..... : enabled
NetCDF..... : enabled
Numpy..... : enabled
OPENBLAS..... : enabled
PETSc..... : enabled
PHDF5..... : enabled
ScaLAPACK..... : enabled
Scipy..... : enabled
Superlu..... : enabled
Superlu_dist..... : enabled
Trilinos ..... : enabled
```

Apps:

```
MiniFE..... : enabled
MiniDFT..... : enabled
HPCG..... : enabled
PRK..... : enabled
```

Community Jenkins Instance for CI Testing



The screenshot shows the Jenkins web interface for openHPC. The top navigation bar includes the openHPC logo, the word "Jenkins", a search bar, a help icon, the user name "KARL W. SCHULZ", and a "LOG OUT" button. Below the navigation bar, there are tabs for different versions: 1.3, 1.3.1, 1.3.2, 1.3.3, 1.3.4, and 1.3.5 (selected). There are also "All" and "+" buttons. The main content area displays a table of test jobs with columns for "S", "Categorized - Job", "Last Success", "Last Failure", "Last Duration", and "Test Result".

S	Categorized - Job	Last Success	Last Failure	Last Duration	Test Result
-	.. » [aarch64]	3 hr 19 min - #93	1 day 0 hr - #166	2 hr 50 min	N/A
✓	(1.3.5) - (centos7.5,aarch64) (warewulf+pbspro) (fabric=eth)	3 hr 19 min - #93	4 days 3 hr - #81	2 hr 50 min	0 of 1,086 failed (±0)
⚠	(1.3.5) - (centos7.5,aarch64) (warewulf+slurm) (fabric=eth)	15 hr - #165	6 days 2 hr - #156	3 hr 13 min	0 of 1,096 failed (±0)
✓	(1.3.5) - (sles12sp3,aarch64) (warewulf+pbspro) (fabric=eth)	6 hr 13 min - #59	5 days 18 hr - #51	2 hr 54 min	0 of 1,079 failed (±0)
✓	(1.3.5) - (sles12sp3,aarch64) (warewulf+slurm) (fabric=eth)	12 hr - #167	1 day 0 hr - #166	3 hr 8 min	0 of 1,089 failed (-1)
-	.. » [x86_64] - CentOS 7	2 hr 2 min - #289	8 hr 1 min - #287	1 hr 19 min	N/A
✓	(1.3.5) - (centos7.4,x86_64) (warewulf+slurm) (fabric=ib) - upgrade	3 hr 15 min - #64	4 days 11 hr - #29	1 hr 27 min	0 of 1,430 failed (±0)
✓	(1.3.5) - (centos7.5,x86_64) (warewulf+pbspro) (fabric=ib) - UEFI	2 hr 2 min - #289	8 hr 1 min - #287	1 hr 19 min	0 of 1,430 failed (-2)
⚠	(1.3.5) - (centos7.5,x86_64) (warewulf+slurm) (fabric=eth)	2 hr 26 min - #738	N/A	1 hr 8 min	0 of 1,133 failed (±0)
⚠	(1.3.5) - (centos7.5,x86_64) (warewulf+slurm) (fabric=eth) + PMix	3 hr 28 min - #285	6 days 0 hr - #238	50 min	0 of 835 failed (±0)
⚠	(1.3.5) - (centos7.5,x86_64) (warewulf+slurm) (fabric=ib) + psxe	4 hr 43 min - #285	1 day 22 hr - #271	2 hr 29 min	0 of 2,336 failed (±0)
⚠	(1.3.5) - (centos7.5,x86_64) (warewulf+slurm) (fabric=ib) - tarball REPO	4 hr 24 min - #22	8 hr 23 min - #21	1 hr 27 min	0 of 1,444 failed (±0)
⚠	(1.3.5) - (centos7.5,x86_64) (warewulf+slurm) (fabric=opa) + psxe	5 hr 6 min - #322	5 days 16 hr - #293	2 hr 29 min	0 of 2,326 failed (±0)
✓	(1.3.5) - (centos7.5,x86_64) (warewulf-stateful+slurm) (fabric=ib)	4 days 16 hr - #2	N/A	0.73 sec	0 of 1,437 failed (±0)
✓	(1.3.5) - (centos7.5,x86_64) (warewulf-stateful+slurm) (fabric=ib) upgrade	4 days 9 hr - #2	N/A	0.6 sec	0 of 1,423 failed (-3)
✓	(1.3.5) - (centos7.5,x86_64) (xcat+slurm) (fabric=ib)	2 hr 53 min - #288	4 days 8 hr - #254	1 hr 25 min	0 of 1,433 failed (±0)
-	.. » [x86_64] - SLES12	1 hr 26 min - #455	17 hr - #19	1 hr 11 min	N/A
✓	(1.3.5) - (sles12sp3,x86_64) (warewulf+pbspro) (fabric=ib) - tarball REPO	1 hr 44 min - #23	17 hr - #19	1 hr 0 min	0 of 1,424 failed (±0)
⚠	(1.3.5) - (sles12sp3,x86_64) (warewulf+pbspro) (fabric=ib) - UEFI	3 hr 4 min - #456	2 days 9 hr - #438	1 hr 2 min	0 of 1,424 failed (±0)
✓	(1.3.5) - (sles12sp3,x86_64) (warewulf+slurm) (fabric=eth)	3 hr 26 min - #492	5 days 17 hr - #447	59 min	0 of 1,133 failed (±0)
✓	(1.3.5) - (sles12sp3,x86_64) (warewulf+slurm) (fabric=ib)	1 hr 26 min - #455	4 days 13 hr - #419	1 hr 11 min	0 of 1,444 failed (±0)
✓	(1.3.5) - (sles12sp3,x86_64) (warewulf+slurm) (fabric=opa) + psxe	2 hr 36 min - #192	3 days 8 hr - #175	2 hr 3 min	0 of 2,326 failed (±0)

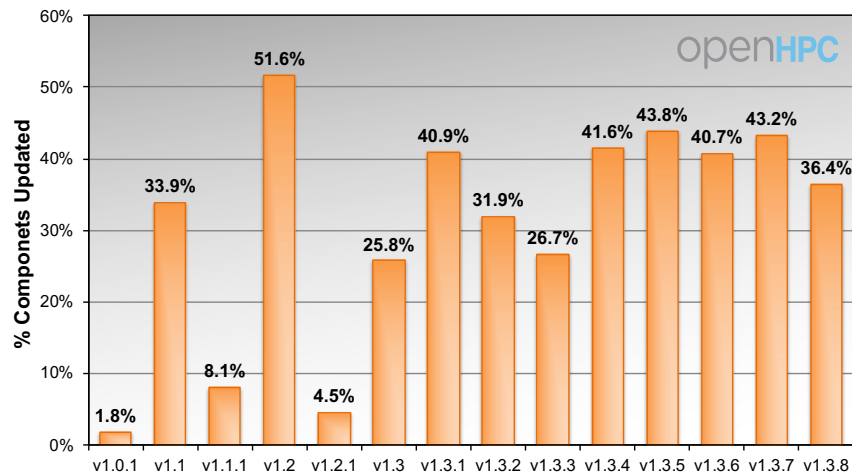
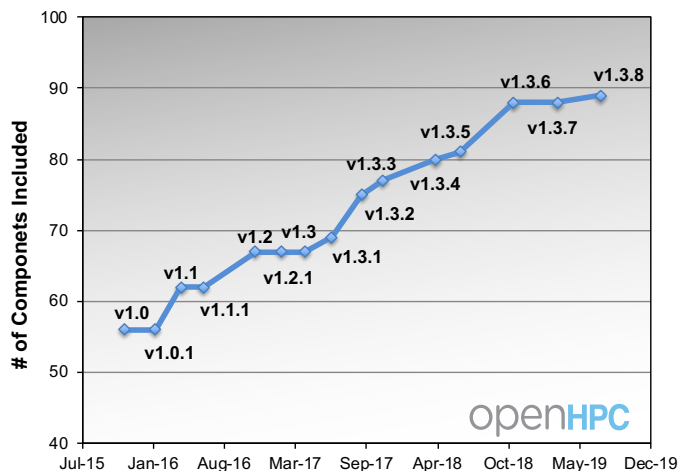
our CI "build" is really a cluster

All recipes exercised in CI system (start w/ bare-metal installs + integration test suite)

Latest Release and Stats

Additions and Upstream Version Changes

- Part of the motivation for community effort like OpenHPC is the rapidity of S/W updates that occurs in our space
- We have been doing releases on a roughly quarterly cadence:
 - convention is to go with latest stable release of upstream components
 - additional components added over time



OpenHPC v1.3.8 - S/W components

Functional Areas	Components	new in v1.3.8 release
Base OS	CentOS 7.6, SLES12 SP4	
Architecture	aarch64, x86_64	
Administrative Tools	Conman, Ganglia, Lmod, LosF, Nagios, NHC, pdsh, pdsh-mod-slurm, prun, EasyBuild , ClusterShell, mrsh, Genders, Shine, Spack , test-suite	
Provisioning	Warewulf, xCAT	
Resource Mgmt.	SLURM, Munge, PBS Professional, PMIx	
Runtimes	Charliecloud, OpenMP, OCR, Singularity	
I/O Services	Lustre client, BeeGFS client*	
Numerical/Scientific Libraries	Boost, GSL, FFTW, Hypre, Metis, MFEM, Mumps, OpenBLAS, OpenCoarrays, PETSc, PLASMA, Scalapack, Scotch, SLEPc, SuperLU, SuperLU_Dist, Trilinos	
I/O Libraries	HDF5 (pHDF5), NetCDF/pNetCDF (including C++ and Fortran interfaces), Adios	
Compiler Families	GNU (gcc, g++, gfortran), Clang/LLVM, Intel Parallel Studio*	
MPI Families	MVAPICH2, OpenMPI, MPICH, Intel MPI*	
Development Tools	Autotools, cmake, hwloc, mpi4py, R, SciPy/NumPy, Valgrind	
Performance Tools	PAPI, IMB, Likwid, mpiP, pdttoolkit TAU, Scalasca, ScoreP, SIONLib, GeoPM, msr-safe, Dimemas, Extrae, Paraver, OSU Microbenchmarks	

- 3rd Party libraries are built for each compiler/MPI family
- Resulting repositories currently comprised of ~700 RPMs

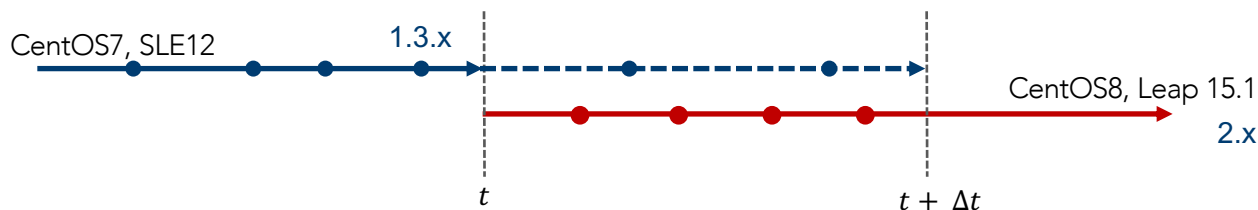
How to Request Additional Software?

- We have a simple submission site for new requests:
 - <https://github.com/openhpc/submissions>
- Example components added via this mechanism since the v1.2. release (Nov' 16)
 - BeeGFS client
 - xCAT recipe
 - hwloc
 - Singularity
 - LLVM/clang
 - PLASMA
 - pNetCDF
 - SCOTCH
 - SLEPc
 - PMIx
 - MPI4py
 - Likwid
 - MFEM
 - NHC
 - Charliecloud
 - GeoPM
 - Dimemas/Extrac, Paraver
 - OpenCoarrays
 - OSU Benchmarks

Software Name
<hr/>
Public URL
<hr/>
Technical Overview
<hr/>
Latest stable version number
<hr/>
Open-source license type
<hr/>
Relationship to component?
<input type="checkbox"/> contributing developer
<input type="checkbox"/> user
<input type="checkbox"/> other
If other, please describe:
<hr/>
Build system
<input type="checkbox"/> autotools-based
<input type="checkbox"/> CMake
<input type="checkbox"/> other
If other, please describe:

What's coming for next big release?

- Will be migrating from 1.3.x series to 2.0 release (targeting SC'19)
- Target major new distro versions:
 - CentOS 8
 - SUSE Leap 15.1 (switching to Leap from SLES)



- Once we release a new major branch (e.g. v2.x) which supports a newer major distro version at time t , we will restrict updates to previous branch (e.g. v1.3.x) to include:
 - security patches (e.g. address known CVEs)
 - significant bugs affecting functionality
 - older branch stays in **maintenance mode** for $\Delta t = 1$ year

Summary

- Provided an overview of the Linux Foundation project along with key takeaways and packaging details
- Community project efforts provides package repository for use with native OS package managers
 - for use on bare-metal or cloud/containerized environments
 - significant companion test effort with documentation recipes for deployment and execution in HPC- runtime environment
 - RHEL8/CentOS8 and SUSE Leap 15.1 coming later this year

Thanks for your time....



- OpenHPC Home: <http://www.openhpc.community/>
- Primary GitHub Site: <https://github.com/openhpc/ohpc>
- Package Repositories: <http://build.openhpc.community/OpenHPC:/>
- Component Submission: <https://github.com/openhpc/submissions>
- System Registry: [System Registration Form](#)
- CI Infrastructure: <http://test.openhpc.community:8080>
- OpenHPC Wiki: <https://github.com/openhpc/ohpc/wiki>

- Mailing Lists: <http://www.openhpc.community/support/mail-lists/>
 - [openhpc-announce](#)
 - [openhpc-users](#)
 - [openhpc-devel](#)