OpenHPC Community BoF

Karl W. Schulz (University of Texas)
David Brayford (Leibniz Supercomputing Centre)
Adrian Reber (Red Hat)

OpenHPC Technical Steering Committee (TSC) Members

openHPC
https://openhpc.community

Time: Wednesday, **June 19th, 8:30am – 9:30am**
Location: Konstant

ISC High Performance
The HPC Event.

Outline

- Part I: Presentation (~30 mins)
  - membership, growth metrics
  - what's new
  - latest release highlights
  - future work

- Part II: Open Forum (~30 mins)

Current Project Members

Altair

Argonne NATIONAL LABORATORY

arm

BERKELEY LAB

BSC Barcelona Supercomputing Center Centro Nacional de Supercomputación

Bright Computing

cea

CINECA

compute canada

CRAY

DELL EMC

FUJITSU

GENCI

HUAWEI

Indiana University

intel

ift Instituto de Física Teórica UAM-CSIC

KISTI www.kisti.re.kr Korea Institute of Science and Technology Information

Lawrence Livermore National Laboratory

Lenovo

Los Alamos NATIONAL LABORATORY EST. 1943

lrz

New Mexico CONSORTIUM

NREL NATIONAL RENEWABLE ENERGY LABORATORY

NUST "Think in other Terms"

PSC PITTSBURGH SUPERCOMPUTING CENTER

QCT Quanta CLOUD TECHNOLOGY

Red Hat

RIKEN

RUTGERS

Sandia National Laboratories

SUSE

TEXAS The University of Texas at Austin TACC

UNIVA

University of Cambridge

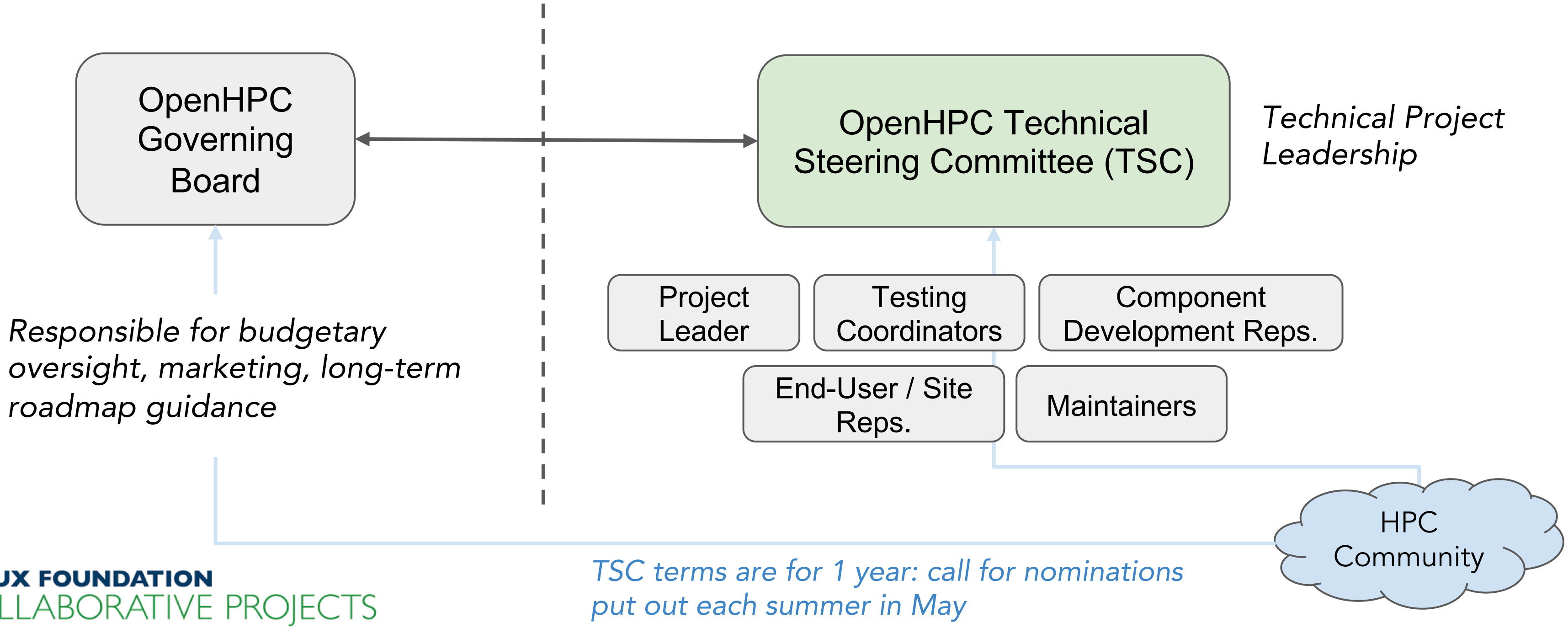
THE UNIVERSITY OF NEW MEXICO

THE LINUX FOUNDATION

Member participation interest? Contact Neal Caidin ncaidin@linuxfoundation.org

Reminder on Project Governance

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



OpenHPC Governing Board

Responsible for budgetary oversight, marketing, long-term roadmap guidance

OpenHPC Technical Steering Committee (TSC)

Technical Project Leadership

Project Leader

Testing Coordinators

Component Development Reps.

End-User / Site Reps.

Maintainers

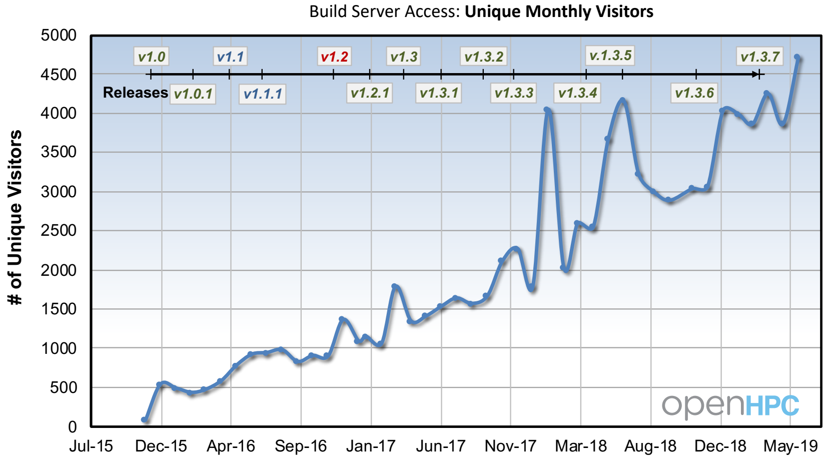
HPC Community

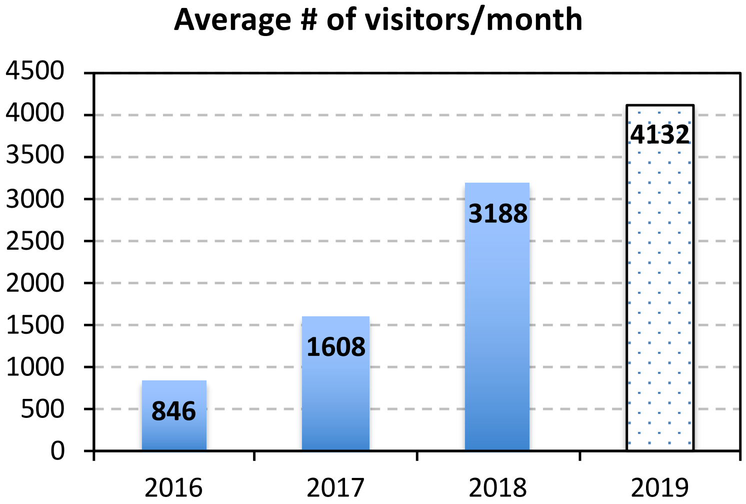
TSC terms are for 1 year: call for nominations put out each summer in May

LINUX FOUNDATION COLLABORATIVE PROJECTS

Adoption / Usage Metrics

Project Adoption Growth





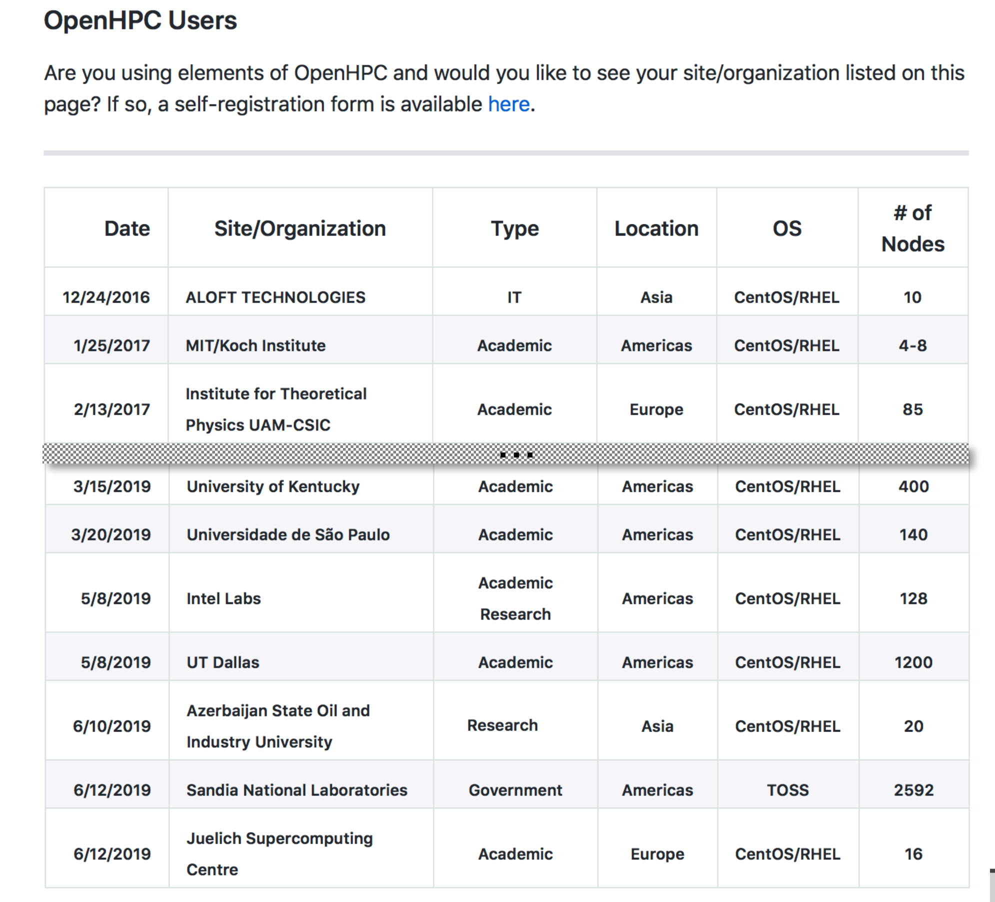
- Continued access/download growth since initial release at SC’15
- Plots highlight number of unique visitors/month to the OpenHPC build server/repo(s)
- **Over 21TB downloaded in 2018**

System Registry Available on OpenHPC GitHub wiki

- Reminder that we maintain a **system registry** on the community wiki site

- Optional, but appreciated to help us know who is using elements from the project

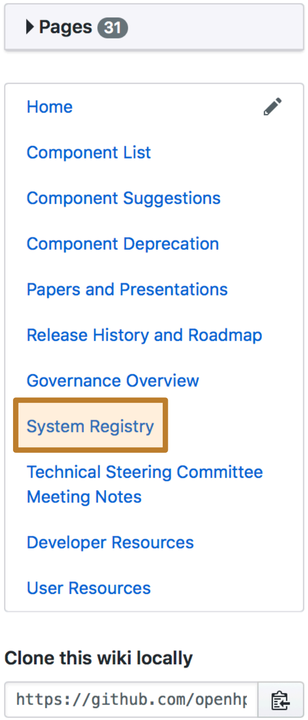
https://github.com/openhpc/ohpc/wiki/System-Registry



OpenHPC Users

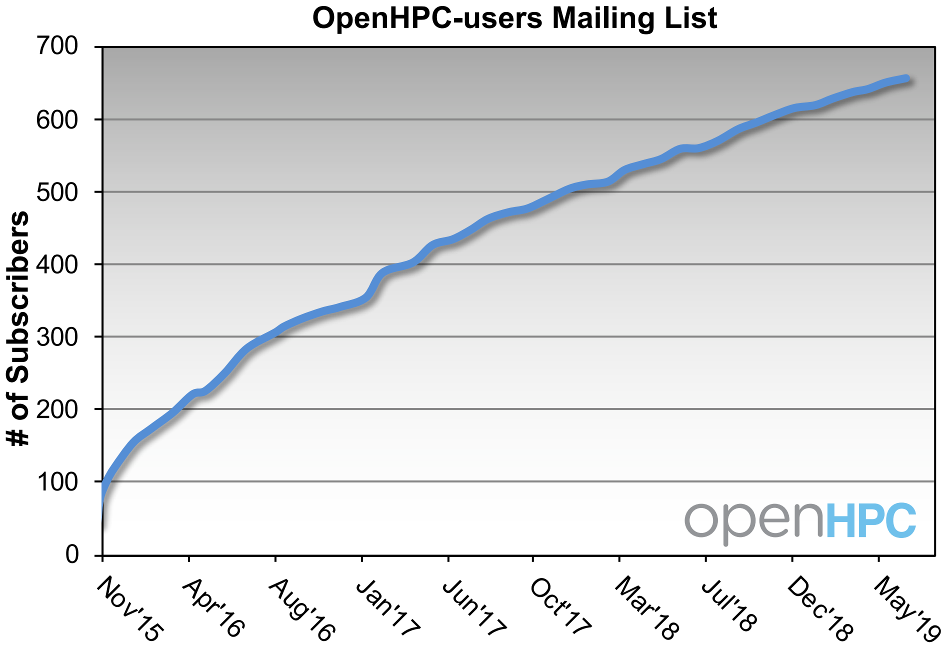
Are you using elements of OpenHPC and would you like to see your site/organization listed on this page? If so, a self-registration form is available here.

Date	Site/Organization	Type	Location	OS	# of Nodes
12/24/2016	ALOFT TECHNOLOGIES	IT	Asia	CentOS/RHEL	10
1/25/2017	MIT/Koch Institute	Academic	Americas	CentOS/RHEL	4-8
2/13/2017	Institute for Theoretical Physics UAM-CSIC	Academic	Europe	CentOS/RHEL	85
3/15/2019	University of Kentucky	Academic	Americas	CentOS/RHEL	400
3/20/2019	Universidade de São Paulo	Academic	Americas	CentOS/RHEL	140
5/8/2019	Intel Labs	Academic Research	Americas	CentOS/RHEL	128
5/8/2019	UT Dallas	Academic	Americas	CentOS/RHEL	1200
6/10/2019	Azerbaijan State Oil and Industry University	Research	Asia	CentOS/RHEL	20
6/12/2019	Sandia National Laboratories	Government	Americas	TOSS	2592
6/12/2019	Juelich Supercomputing Centre	Academic	Europe	CentOS/RHEL	16



Mailing Lists - http://www.openhpc.community/support/mail-lists/

- Three lists currently:
  - openhpc-announce
  - openhpc-users
  - openhpc-devel
- Great place to interact with developers and others using elements from the project
- *Really appreciate the great interaction we've seen on the list*



~90 posts/month in 2018

Updates / New Items

will next highlight some new items/changes since the last BoF at ISC

rysnc is now supported for OpenHPC repos

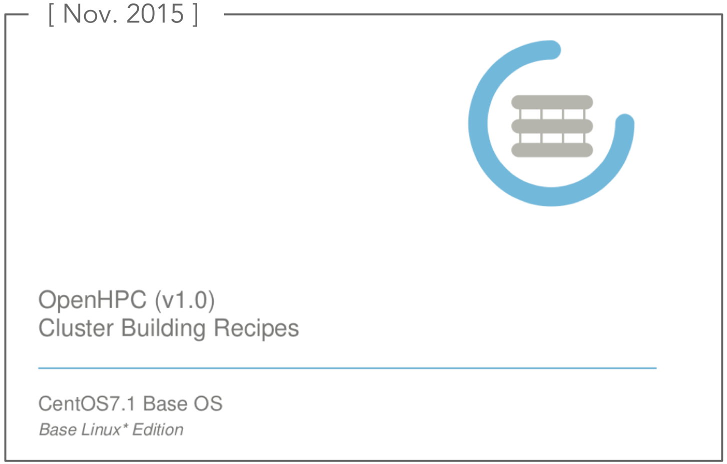
- We now support rsync for local mirroring of OpenHPC repositories

- Three methods now available for package access access:
  - direct repo access via **yum/zypper** (requires external routing)
  - local repo mirroring (download self-contained tarball with binary/src RPMs)
  - local repo mirroring (via rsync)

- Typical rsync mirroring example for binaries in 1.3 release(s):

```
$ rsync -avzH --exclude src --exclude repocache --delay-updates
rsync://build.openhpc.community/OpenHPC/1.3/ 1.3
```

Wider variety of recipes now available



[Nov. 2015]

OpenHPC (v1.0)
Cluster Building Recipes

CentOS7.1 Base OS
Base Linux Edition*

Initially, we started off with only a single recipe with intent to expand

10 recipes now available with v1.3.8 release

excellent place to start if you are new to the project or want to kick the tires

We continue to expand recipe option(s) with multiple resource managers, OSes, provisioners, and architectures:

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

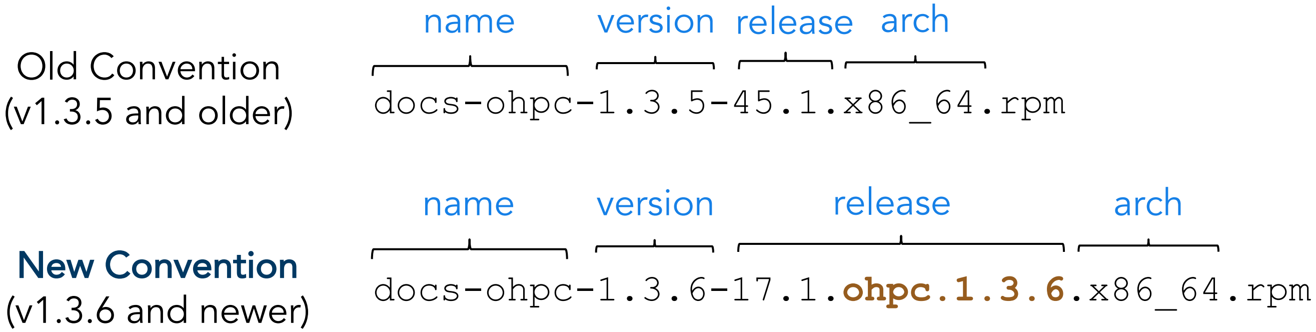
aarch64:

- 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

[we test all of these in a CI environment on bare-metal]

Packaging convention change

- A small change to the package naming schema was introduced in the v1.3.6 release
- This simple change embeds the OpenHPC release version where a package build originates from into the RPM release string
  - motivation: make it easier to look at package name and know which ohpc version it came from



Old Convention
(v1.3.5 and older)

`docs-ohpc-1.3.5-45.1.x86_64.rpm`

New Convention
(v1.3.6 and newer)

`docs-ohpc-1.3.6-17.1.`**`ohpc.1.3.6`**`.x86_64.rpm`

Packaging update: use of custom ohpc RPM plugin

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

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

default RPM dependency analysis

OHPC 1.3.1 RPM

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

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)
```

.so’s contained within ohpc gcc build

.so’s required from base OS supplied packages

Build with Updated Approach

```
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)

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)
```

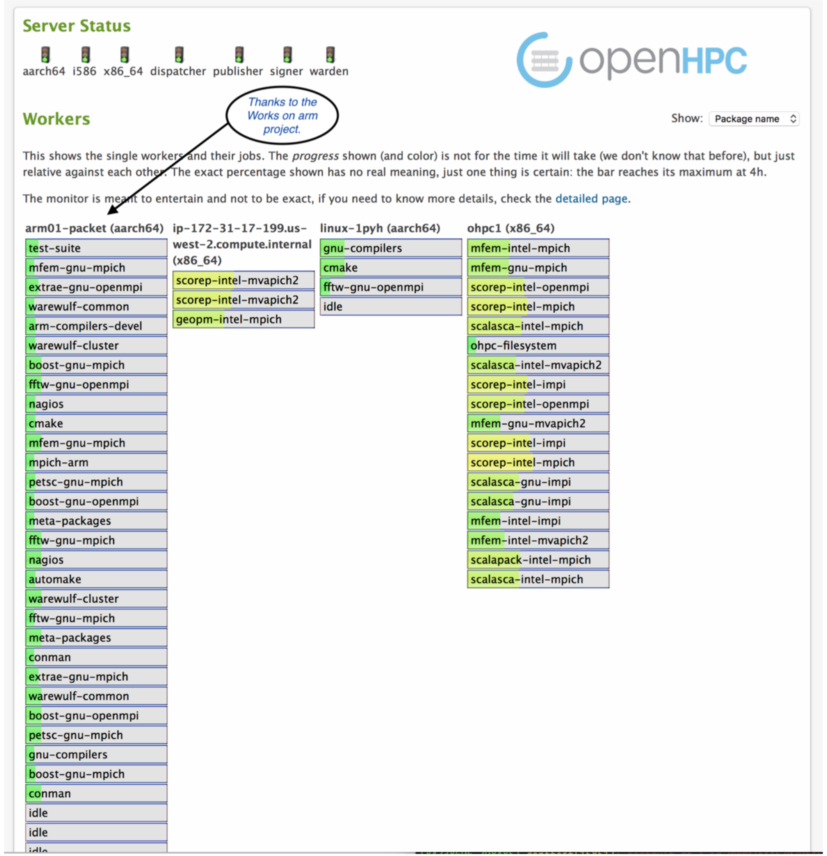
updated RPM dependency analysis using plugin

We introduced this new convention in v1.3.4 release

Build system addition for aarch64

- There is a Works on ARM project that provides CI/CD infrastructure to community projects hosted at Packet
  - https://www.worksonarm.com
  - https://www.packet.net

- OpenHPC community requested CI host to support automated builds
  - we were kindly granted a ThunderX2 node to support our build infrastructure
  - have setup OBS on this host and tied it to our central OBS server
  - significantly expands aarch64 build resources for the project
  - Example build time reductions: **trilinos-mvapich2** build
    - previous build time: 18.79 hours
    - latest build time on TX2: 4.19 hours



Server Status

Workers

Thanks to the Works on arm project.

Show: Package name

This shows the single workers and their jobs. The *progress* shown (and color) is not for the time it will take (we don't know that before), but just relative against each other. The exact percentage shown has no real meaning, just one thing is certain: the bar reaches its maximum at 4h.

The monitor is meant to entertain and not to be exact, if you need to know more details, check the detailed page.

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

Miscellaneous updates

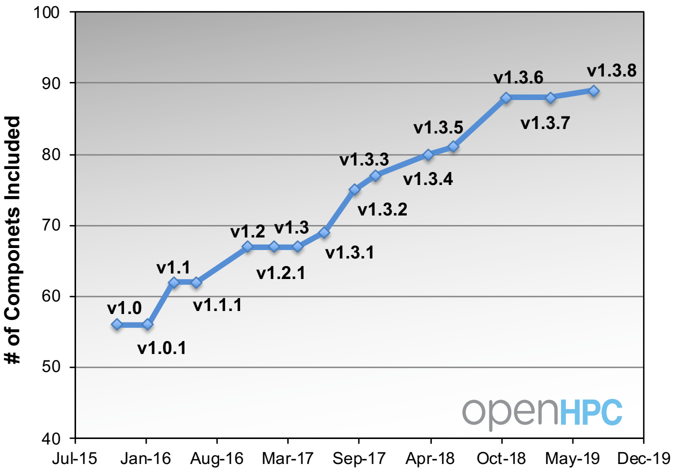
- CentOS7.6 and SLES12 SP4 support introduced (v1.3.7 release)
- packaging go-based Singularity 3.x version (v1.3.7 release)
- gcc8 compiler variant introduced (v1.3.6 release)
- SLURM build updates:
  - enable X11 support
  - enable jobcomp_elasticsearch plugin
  - enabled Lua job submit plugin
- Other installation recipes leveraging OpenHPC available on GitHub wiki
  - XSEDE Compatible Basic Cluster
  - Vanilla recipe using Ansible (LANL)
  - Ansible playbook for OpenHPC (Linaro)

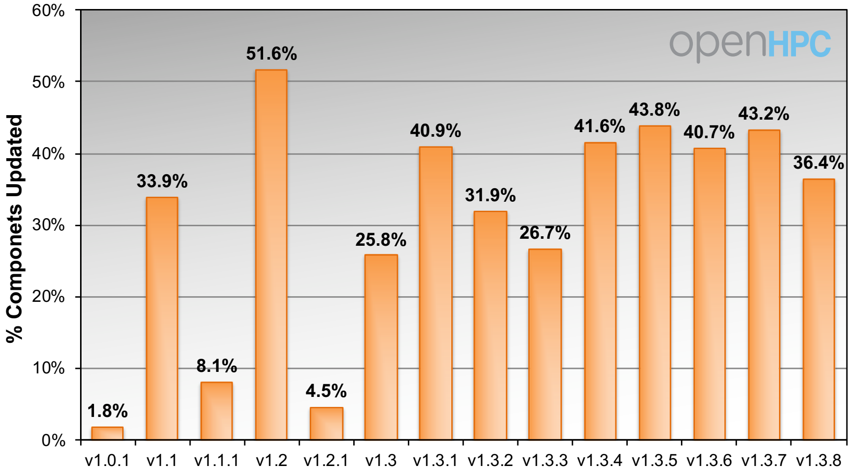
OpenHPC v1.3.8 Release

June 2019

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 continued to do 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

Released June 11, 2019

components available 89 | updates 36%

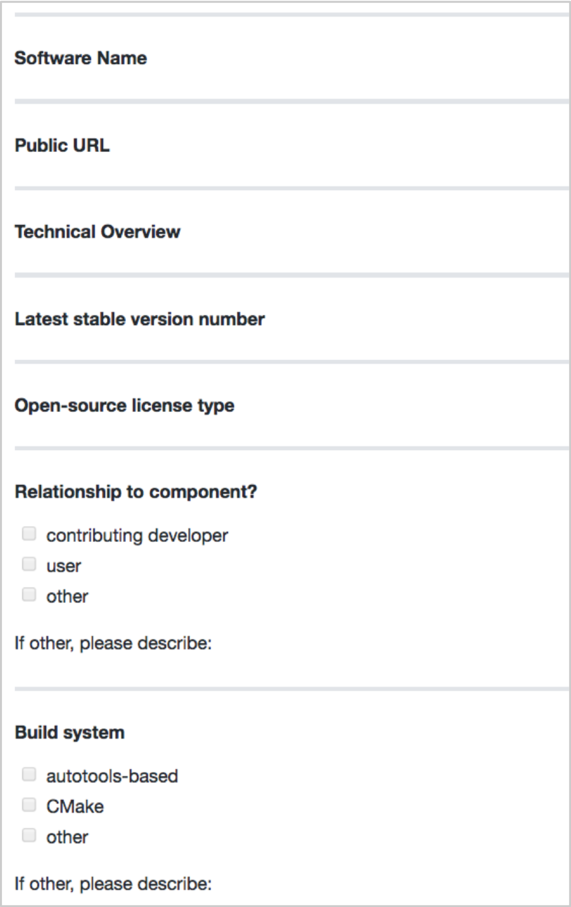
Functional Areas	Components **new in v1.3.8 release**
Base OS	CentOS 7.6, SLES12 SP4
Architecture	aarch64, x86_64
Administrative Tools	Conman, Ganglia, Lmod, LosF, Nagios, NHC, pdsh, pdsh-mod-slurm, prun, EasyBuild, ClusterShell, mrsh, Genders, Shine, Spack, test-suite
Provisioning	Warewulf, xCAT
Resource 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, pdtoolkit 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/Extrae, Paraver
  - OpenCoarrays
  - OSU Benchmarks

Next Submission Deadline: July 31st, 2019
(continue thereafter on a rolling quarterly basis)



Software Name

Public URL

Technical Overview

Latest stable version number

Open-source license type

Relationship to component?

□ contributing developer
□ user
□ other

If other, please describe:

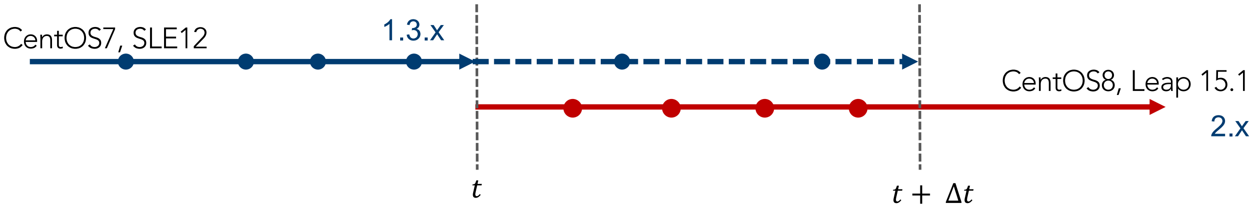
Build system

□ autotools-based
□ CMake
□ 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)
- Targeting two new major 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 **Δt = 1 year**

Additional Future Work

- Compatibility builds with ARM HPC compiler
- Continued s/w updates and releases, OS updates….
- Exploring an archive repository
- RPM macro tweaks to better support localized package rebuild customization (more on that in a sec)

- ***other items based on your feedback…***

Build flag defaults and customization

- RPM build infrastructure generally provides reasonable set of default compilation flags, however these can vary slightly between the distro(s)
  - we're looking to implement same defaults across both distros for 2.x release
  - while we're at it, also looking to expand current macro infrastructure to allow for compiler flag customization without having to edit underlying .spec file
  - have worked thru a prototype of how this might work, will walk thru quick overview here; expect to see future discussion on devel list about this for folks who might be interested in kicking the tires

Alternate build flag support

- General idea is to provide option allowing user to provide custom compile flag overrides (via RPM macro or env variable)
- Exposed to builds leveraging our %{ohpc_setup_compiler} macro
- Look at relevant macro changes (focus on CFLAGS)

example relevant change for OHPC_macros

```
        %global ohpc_setup_compiler %{expand:\
            %if 0%{?OHPC_CFLAGS:1} \
                 export OHPC_CFLAGS=%{OHPC_CFLAGS} \
            %endif \
            %if 0%{?OHPC_CXXFLAGS:1} \
                 export OHPC_CXXFLAGS=%{OHPC_CXXFLAGS} \
    New     %endif \
            %if 0%{?OHPC_FCFLAGS:1} \
                 export OHPC_FCFLAGS=%{OHPC_FCFLAGS} \
            %endif \
            %if 0%{?OHPC_F77FLAGS:1} \
                 export OHPC_F77FLAGS=%{OHPC_F77FLAGS} \
            %endif \
            . %{OHPC_ADMIN}/ohpc/OHPC_setup_compiler %{compiler_family} \
    Orig.   %if 0%{?ohpc_mpi_dependent} == 1 \
                . %{OHPC_ADMIN}/ohpc/OHPC_setup_mpi %{mpi_family} \
            %endif \
        }
```

example relevant change for OHPC_setup_compiler

```
if [ -z "$OHPC_CFLAGS" ];then
    export CFLAGS=$DEFAULT_OPTS
else
    export CFLAGS=$OHPC_CFLAGS
    echo "--> CFLAGS   (user override) = $CFLAGS"
fi
```

Alternate build flag support

- Introduction of **OHPC_CFLAGS**, **OHPC_CXXFLAGS**, and **OHPC_FCFLAGS** will allow fairly simple way for user to override compilation flags without touching .spec file (in most cases)
- user may want to have this build co-installed with default ohpc variant
- in that case, need way to include additional delimiter
- can accommodate this thru change to OHPC_macros and changes to .spec file (leverage fact that we already rely on %{PROJ_DELIM} macro to add "ohpc" delimiter to packages
- triggered via macro setting of **OHPC_CUSTOM_DELIM**

```
# check if user desires to override package and modulefile naming with
# custom delimiter (e.g optimized micro-architecture build)

%global OHPC_CUSTOM_PKG_DELIM %{nil}

%{?OHPC_CUSTOM_DELIM: %global OHPC_CUSTOM_PKG_DELIM -%{OHPC_CUSTOM_DELIM}}
%{?OHPC_CUSTOM_DELIM: %global PROJ_DELIM -%{OHPC_CUSTOM_DELIM}%{PROJ_DELIM}}
```

example relevant change for OHPC_macros

Alternate build flag support

- changes to each .spec file required to accommodate naming override:
    - update install path definition
    - update module name definition
- consider simple autotools package changes below

example changes to typical ohpc-style .spec file

```
# Default library install path
%define install_path %{OHPC_LIBS}/%{compiler_family}/%{pname}%{OHPC_CUSTOM_PKG_DELIM}/%version

...


# OpenHPC module file
%{__mkdir} -p %{buildroot}%{OHPC_MODULEDEPS}/%{compiler_family}/%{pname}
%{__cat} << EOF >
%{buildroot}/%{OHPC_MODULEDEPS}/%{compiler_family}/%{pname}/%{version}%{OHPC_CUSTOM_PKG_DELIM}
```

Alternate build flag support

- Let's see this in action, have a simple autotools package setup in my personal OBS project with these changes in place
  - opt_standard: standard (ohpc-style) package build using default build flags
  - opt_custom: customized build with alternate C compile flags and package name
  - Use our standard _link file overrides to define opt_custom

```
<link project='home:Admin' package='opt_standard'>
<patches>
  <topadd>%define OHPC_CUSTOM_DELIM nonzippy</topadd>
  <topadd>%define OHPC_CFLAGS "-O0 -g"</topadd>
</patches>
</link>
```

example _link file for OBS

- also simple to do via local command-line (e.g user downloads src rpm and wants to rebuild)

```
# rpmbuild -bb --define 'OHPC_CFLAGS "-O0 -g"' --define "OHPC_CUSTOM_DELIM nonzippy" example.spec
```

Alternate build flag support

- Resulting RPM names:
  - example-gnu8-ohpc-1.0-12.2.ohpc.1.3.6.x86_64.rpm
  - example-gnu8-nonzippy-ohpc-1.0-19.1.ohpc.1.3.6.x86_64.rpm
- These two RPMs can be installed simultaneously (since we altered the install_path) and user sees both available in modules:

builds visible to user

```
$ module avail example

--------------------------------- /opt/ohpc/pub/moduledeps/gnu8 ---------------------------------
   example/1.0-nonzippy    example/1.0 (D)

  Where:
   D:  Default Module
```

Open Discussion

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
    - includes links to overview paper and past presentations

- Mailing Lists: http://www.openhpc.community/support/mail-lists/
    - openhpc-announce
    - openhpc-users
    - openhpc-devel