INTEL® HPC DEVELOPER CONFERENCE

FUEL YOUR INSIGHT

intel®

Legal Notices and Disclaimers

- Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.
- No computer system can be absolutely secure.
- Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit **http://www.intel.com/performance**.
- Cost reduction scenarios described are intended as examples of how a given Intel- based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.
- No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.
- Intel, the Intel logo and others are trademarks of Intel Corporation in the U.S. and/or other countries.
- *Other names and brands may be claimed as the property of others.
-

INTEL® HPC DEVELOPER CONFERENCE
FUEL YOUR INSIGHT

SIMPLIFIED SYSTEM SOFTWARE STACK DEVELOPMENT AND MAINTENANCE

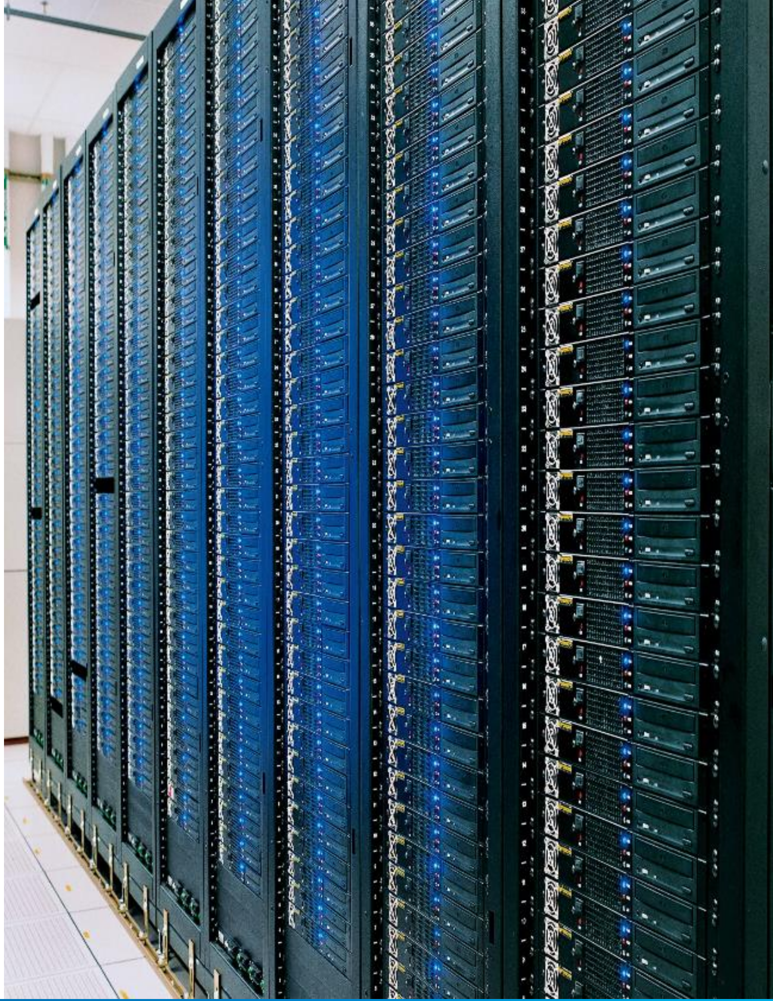
Karl W. Schulz
Technical Project Lead
Datacenter Group, OpenHPC

John Westlund
Systems SW Engineer
Datacenter Group

November 2016

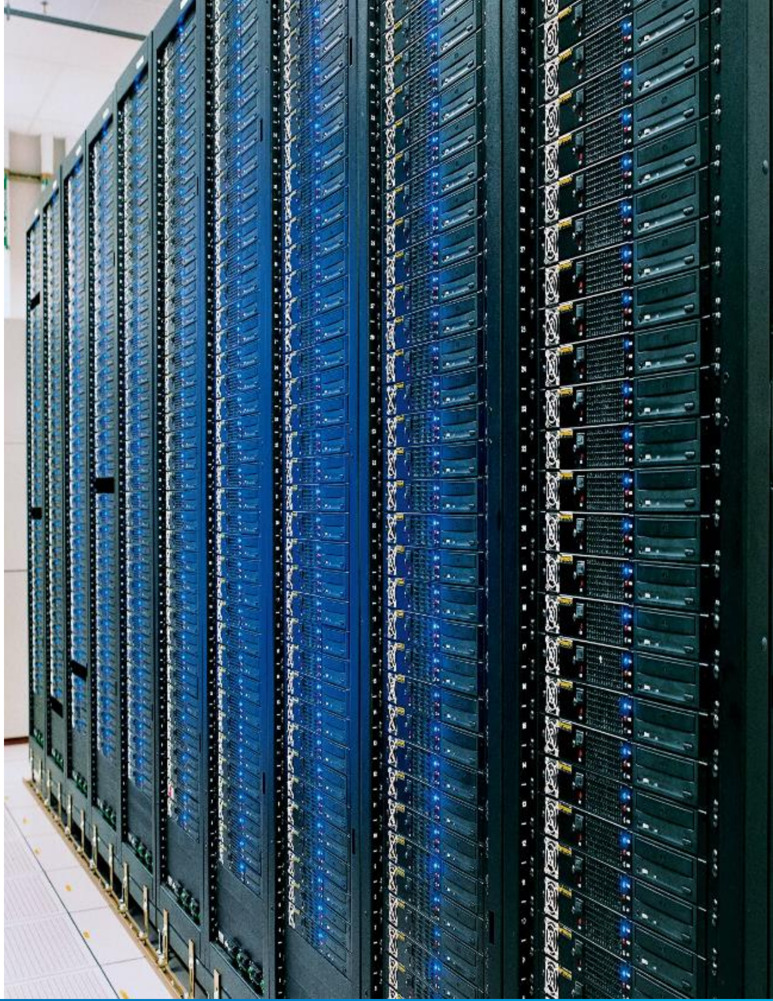
Agenda

- The HPC system software ecosystem problems we all deal with
- OpenHPC* community
- Intel® HPC Orchestrator
- How to make use of these system software solutions



Agenda

- **The HPC system software ecosystem problems we all deal with**
- OpenHPC* community
- Intel® HPC Orchestrator
- How to make use of these system software solutions

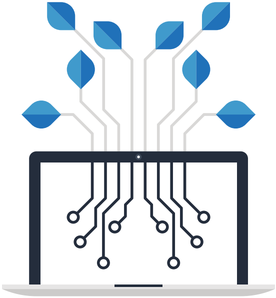


State of System Software Efforts in HPC Ecosystem

Fragmented efforts across the ecosystem – "Everyone building their own solution."

A desire to get exascale performance & speed up software adoption of hardware innovation

New complex workloads (ML[1], Big Data, etc.) drive more complexity into the software stack

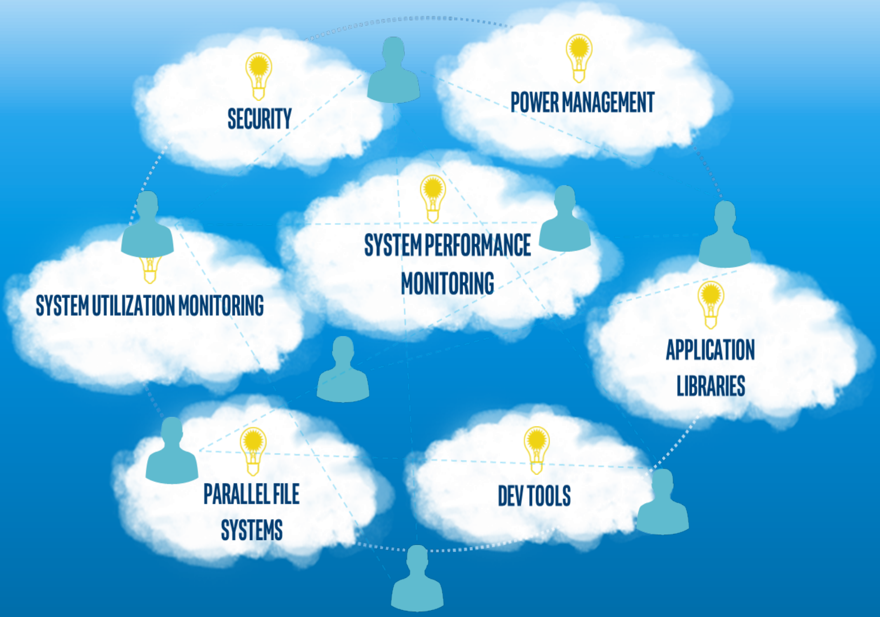


THE REALITY: We will not be able to get where we want to go without a major change in system software development

[1]Machine Learning (ML)

Community Effort to Realize Desired Future State

A Shared Repository



Stable HPC Platform Software that:

- Fuels a vibrant and efficient HPC software ecosystem
- Takes advantage of hardware innovation & drives revolutionary technologies
- Eases traditional HPC application development and testing at scale
- Extends to new workloads (ML, analytics, big data)
- Accommodates new environments (i.e., cloud)

Agenda

- Why a community system software stack?
- **OpenHPC* community**
- Intel® HPC Orchestrator
- How to make use of these system software solutions

A Brief History…

June 2015

ISC '15

- BoF[1] discussion on the merits/ interest in a Community Supported HPC Repository and Management Framework

Nov 2015

SC '15

- Follow-on BoF[1] for a Comprehensive Open Community HPC Software Stack

Nov '15-May '16

Linux* Foundation

- Working group collaborating to define participation agreement, initial governance structure and solicit volunteers

July 2016

Linux Foundation

- announces technical, leadership and member investment milestones with founding members and formal governance structure

Courtesy of openHPC

[1] Birds of a Feather (BoF)

Community 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.

Courtesy of openHPC

OpenHPC* Participation as of Nov 2016

- OpenHPC is a Linux Foundation Project initiated by Intel and gained wide participation right away
- The goal is to collaboratively advance the state of the software ecosystem
- Governing board is composed of Platinum members (Intel, Dell, HPE, SUSE) plus reps from Silver & Academic, Technical committees

29 Members

- Argonne National Laboratory
- Center for Research in Extreme Scale Technologies – Indiana University
- University of Cambridge

Courtesy of openHPC

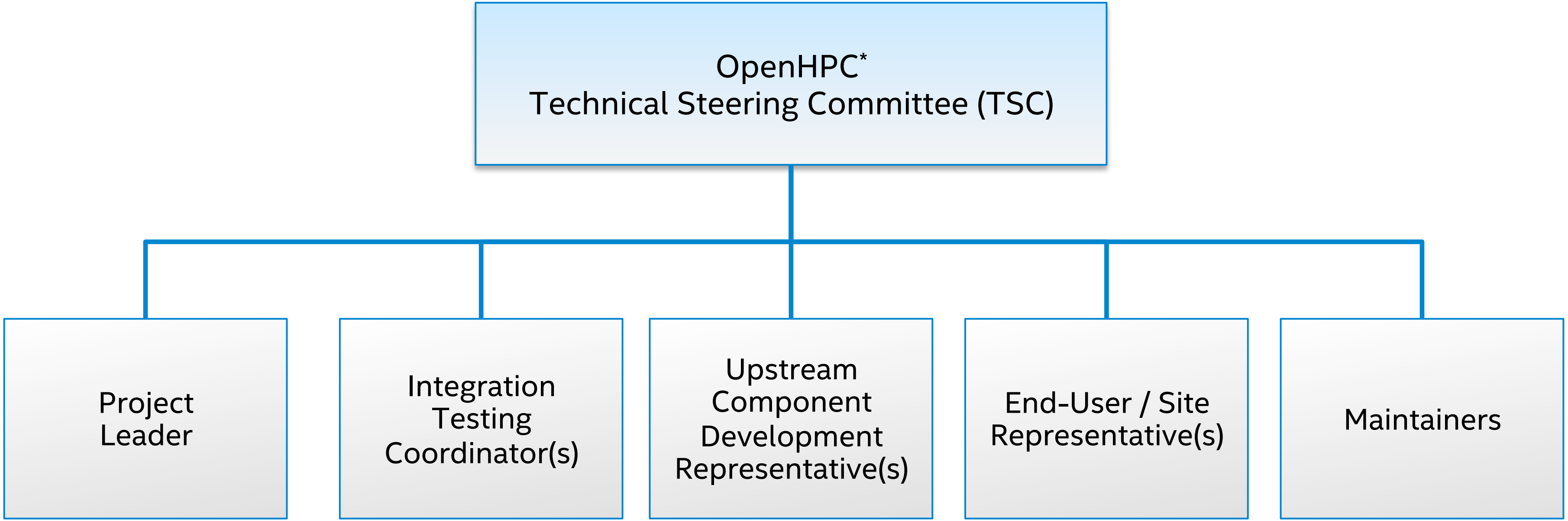
WWW.OpenHPC.Community

Project member participation interest? Please contact Jeff ErnstFriedman: jernstfriedman@linuxfoundation.org

*Other names and brands may be claimed as the property of others.
*Mixture of academics, Labs, OEMs, and ISVs/OSVs

OpenHPC* Technical Steering Committee (TSC)

Role Overview

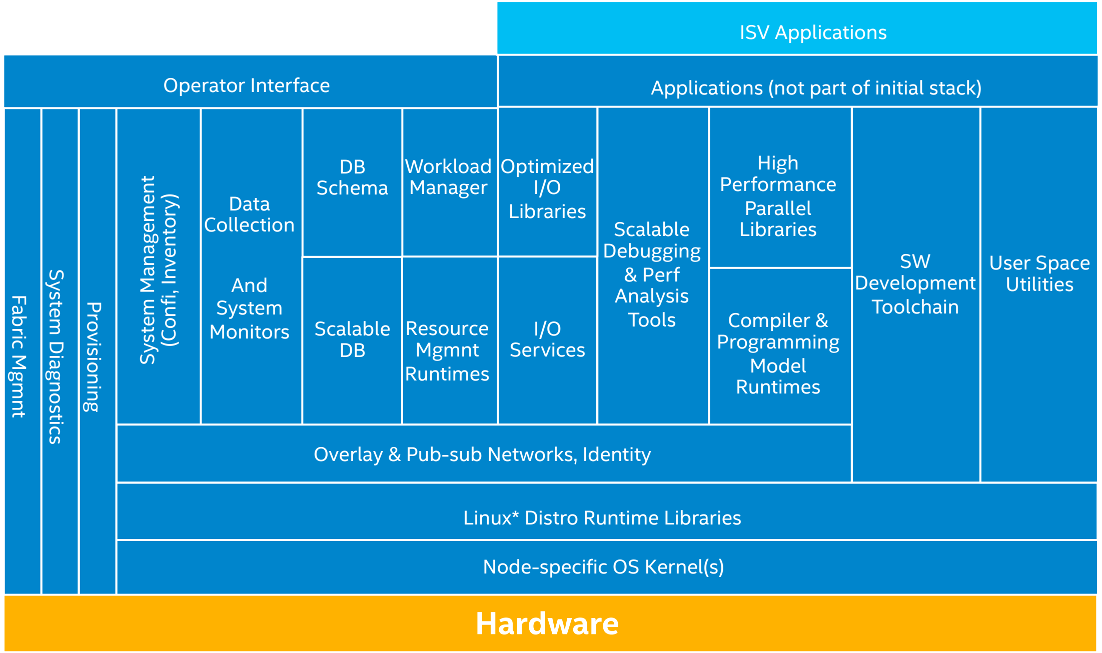


Courtesy of openHPC

https://github.com/openhpc/ohpc/wiki/Governance-Overview

Stack Overview

We have assembled a variety of common ingredients required to deploy and manage an HPC Linux* cluster including provisioning tools, resource management, I/O libs, development tools, and a variety of scientific libraries.



Courtesy of openHPC

*Other names and brands may be claimed as the property of others.

Stack Overview Continued

- Packaging efforts have HPC in mind and include compatible modules (for use with Lmod) with development libraries/tools
- Endeavoring to provide hierarchical development environment that is cognizant of different compiler and MPI families
- Include common conventions for env variables
- Development library install example:

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

- End user interaction example with above install:
  (assume we are a user wanting to build a PETSC hello world in C)

```
$ module load petsc
$ mpicc -I$PETSC_INC petsc_hello.c -L$PETSC_LIB –lpetsc
```

Courtesy of openHPC

Basic Cluster Install Example

- Starting install guide/recipe targeted for flat hierarchy
- Leverages image-based provisioner (Warewulf)
  - PXE[1] boot (stateless)
  - optionally connect external Lustre* file system
- Obviously need hardware-specific information to support (remote) bare-metal provisioning

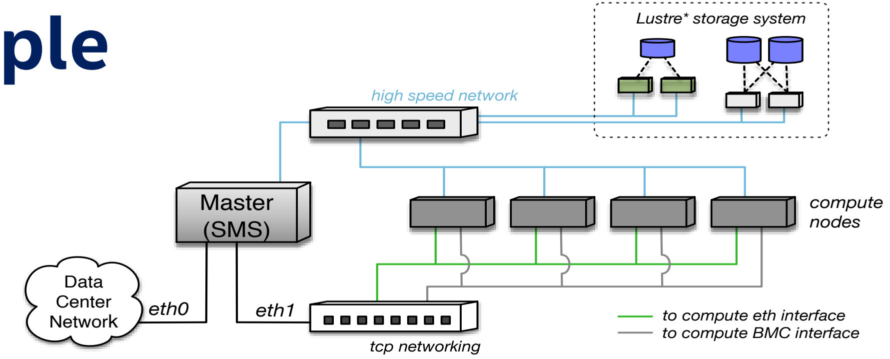


Figure 1: Overview of physical cluster architecture.

- ${sms_name} # Hostname for SMS server
- ${sms_ip} # Internal IP address on SMS server
- ${sms_eth_internal} # Internal Ethernet interface on SMS
- ${eth_provision} # Provisioning interface for computes
- ${internal_netmask} # Subnet netmask for internal network
- ${ntp_server} # Local ntp server for time synchronization
- ${bmc_username} # BMC username for use by IPMI
- ${bmc_password} # BMC password for use by IPMI
- ${c_ip[0]}, ${c_ip[1]}, ... # Desired compute node addresses
- ${c_bmc[0]}, ${c_bmc[1]}, ... # BMC addresses for computes
- ${c_mac[0]}, ${c_mac[1]}, ... # MAC addresses for computes
- ${compute_regex} # Regex for matching compute node names (e.g. c*)

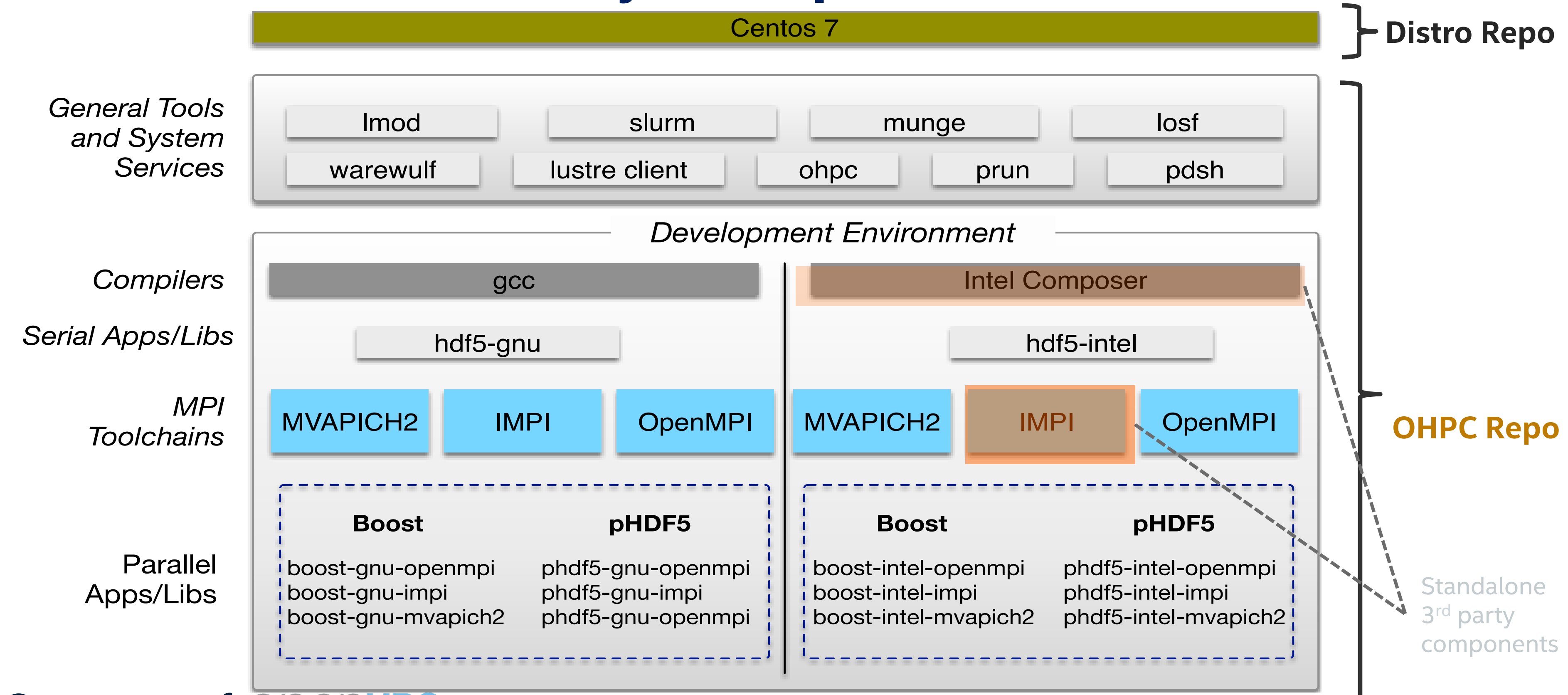
Optional:

- ${mgs_fs_name} # Lustre MGS mount name
- ${sms_ipoib} # IPoIB address for SMS server
- ${ipoib_netmask} # Subnet netmask for internal IPoIB
- ${c_ipoib[0]}, ${c_ipoib[1]}, ... # IPoIB addresses for computes

Courtesy of openHPC

[1]Preboot eXecution Environment (PXE)

Hierarchical Overlay for OpenHPC* Software



Courtesy of openHPC

OpenHPC* 1.1.1 – Current SW Components

Functional Areas	Components
Base OS	CentOS 7.2, SLES12 SP1
Administrative Tools	Conman, Ganglia, Lmod, LosF, Nagios, pdsh, prun, EasyBuild, ClusterShell, mrsh, Genders, Shine, Spack
Provisioning	Warewulf
Resource Mgmt.	SLURM, Munge
Runtimes	OpenMP, OCR
I/O Services	Lustre client (community version)

Functional Areas	Components
Numerical/ Scientific Libraries	Boost, GSL, FFTW, Metis, PETSc, Trilinos, Hypre, SuperLU, SuperLU_Dist, Mumps, OpenBLAS, Scalapack
I/O Libraries	HDF5 (pHDF5), NetCDF (including C++ and Fortran interfaces), Adios
Compiler Families	GNU (gcc, g++, gfortran)
MPI Families	MVAPICH2, OpenMPI
Development Tools	Autotools (autoconf, automake, libtool), Valgrind,R, SciPy/NumPy
Performance Tools	PAPI, IMB, mpiP, pdtoolkit TAU

Courtesy of openHPC

OpenHPC* Development Infrastructure
What are we using to get the job done?

The usual software engineering stuff:

- GitHub* (SCM[1] and issue tracking/planning)
- Continuous Integration (CI) Testing (Jenkins)
- Documentation (Latex)

Capable build/packaging system

- At present: we target a common delivery/ access mechanism that adopts Linux sysadmin familiarity
- Require Flexible System to manage builds
- A system using Open Build Service (OBS) supported by back-end git

https://github.com/openhpc/ohpc

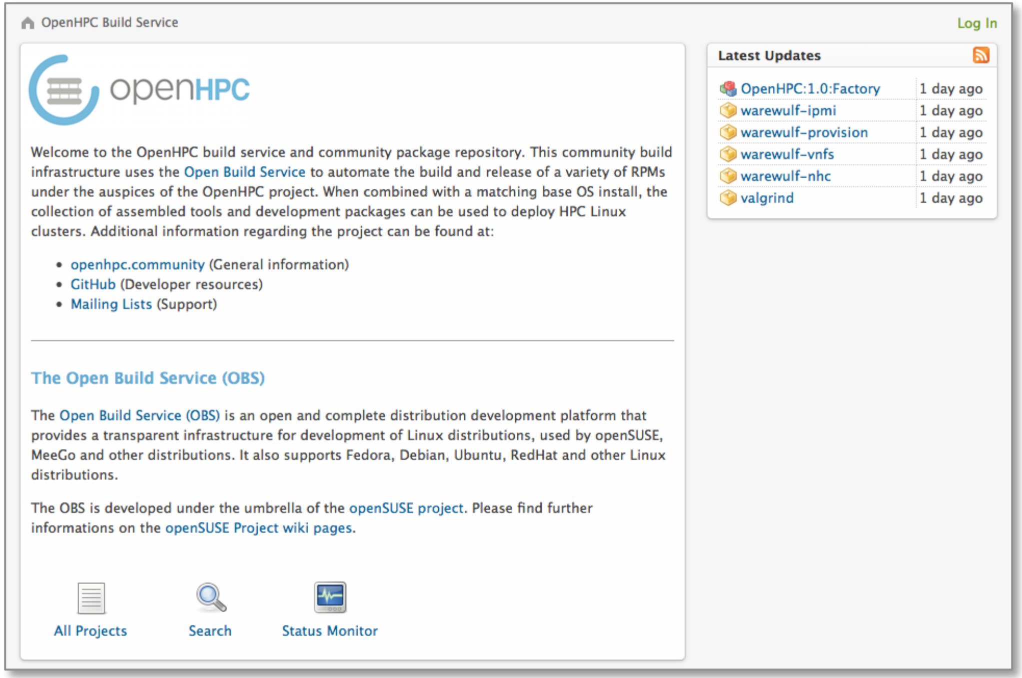
https://build.openhpc.community

Courtesy of openHPC

[1] Software Configuration Management(SCM)

Build System - OBS

- Manages build process
- Drives builds for multiple repositories
- Generates binary and src rpms
- Publishes corresponding package repositories
- Client/server architecture supports distributed build slaves and multiple architectures

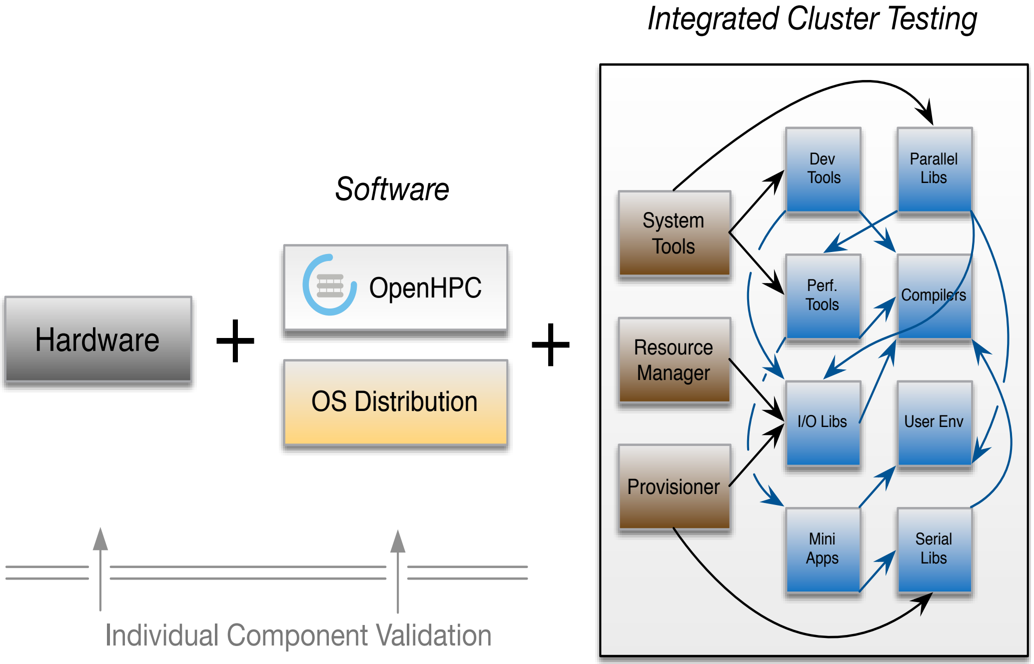


https://build.openhpc.community

Courtesy of openHPC

Integration/Test/Validation

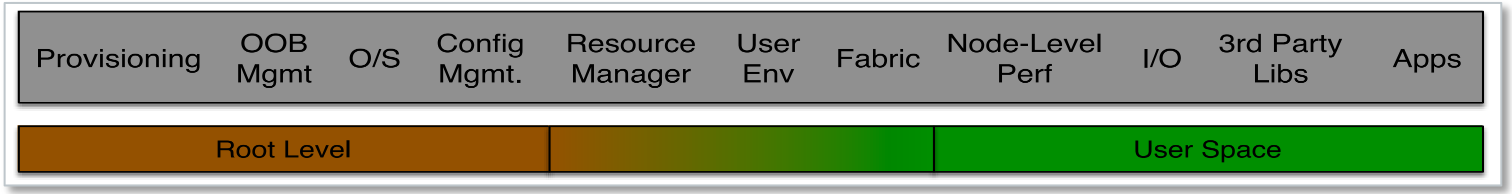
- Install Recipes
- Cross-package interaction
- Development environment
- Mimic use cases common in HPC deployments
- Upgrade mechanism



Courtesy of openHPC

Integration/Test/Validation

- Standalone integration test infrastructure
- Families of tests that could be used during:
  - initial install process (can we build a system?)
  - post-install process (does it work?)
  - developing tests that touch all of the major components (can we compile against 3rd party libraries, will they execute under resource manager, etc.)
- Expectation is that each new component included will need corresponding integration test collateral
- These integration tests are included in GitHub* repo



Courtesy of openHPC

Post Install Integration Tests - Overview

- 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

Example ./configure output (non-root)

```
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.......... :
    Munge..................... :
    Apps...................... :
    Compilers................. :
    MPI....................... :
    HSN....................... :
    Modules................... :
    OOM....................... :
Dev Tools:
    Valgrind.................. :
    R base package............ :
    TBB....................... :
    CILK...................... :
Performance Tools:
    mpiP Profiler........ .... :
    Papi...................... :
    PETSc..................... :
    TAU....................... :
```

```
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
```

Courtesy of openHPC

Note: more than 1,000 jobs submitted to RM as part of the current test suite

New software additions?

- A common question posed to the project is how to request new software components? In response, the TSC has endeavored to formalize a simple submission/review process
- Submission site now exists for this purpose:

  https://github.com/openhpc/submissions

- Expecting to do reviews every quarter (or more frequent if possible)
  - just completed first iteration of the process now
  - next submission deadline: **December 4th , 2016**

Courtesy of OpenHPC

Subset of information requested during submission process

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

How to contribute to OpenHPC*

- Use elements of the stack and provide feedback
- Suggest additional components for selection
- Make software of potential interest for inclusion available as open-source
- Participate in user/developer forums, TSC

http://openhpc.community (General info)

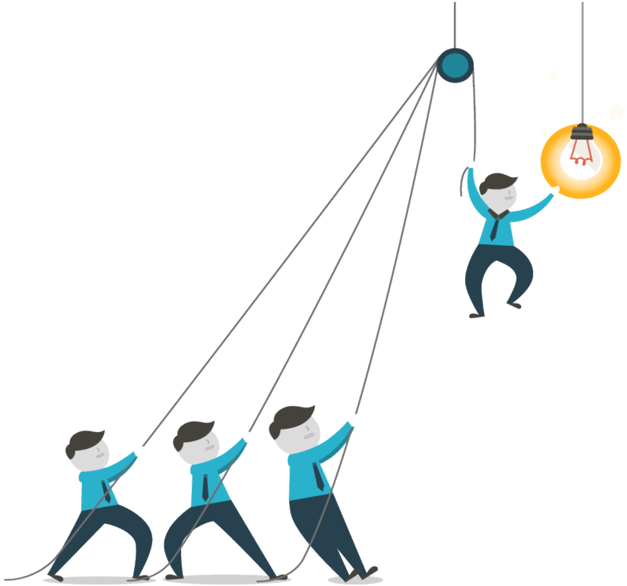
https://github.com/openhpc/ohpc (GitHub site)

https://github.com/openhpc/submissions (Submissions)

https://build.openhpc.community (Build system/repos)

http://www.openhpc.community/support/mail-lists/ (email lists)

- *openhpc-announce, openhpc-users, openhpc-devel*



Courtesy of openHPC

Agenda

- Why a community system software stack?
- OpenHPC* community
- **Intel® HPC Orchestrator**
- How to make use of these system software solutions

OpenHPC* to Intel® HPC Orchestrator to Intel® Scalable System Framework

openHPC

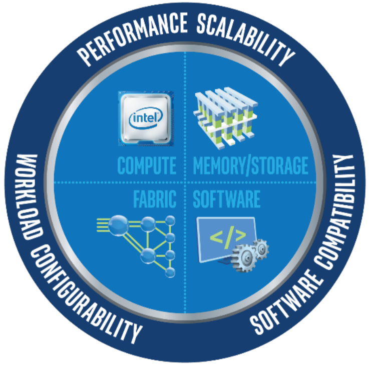
- Open Source Community under Linux Foundation*
- Ecosystem innovation building a consistent HPC SW Platform
- Platform agnostic
- 29 global members
- Multiple distributions

Intel® HPC Orchestrator

- Intel's distribution of OpenHPC*; Intel HW optimized
- Expose best performance for Intel HW
- Advanced testing & premium features
- Product technical support & updates

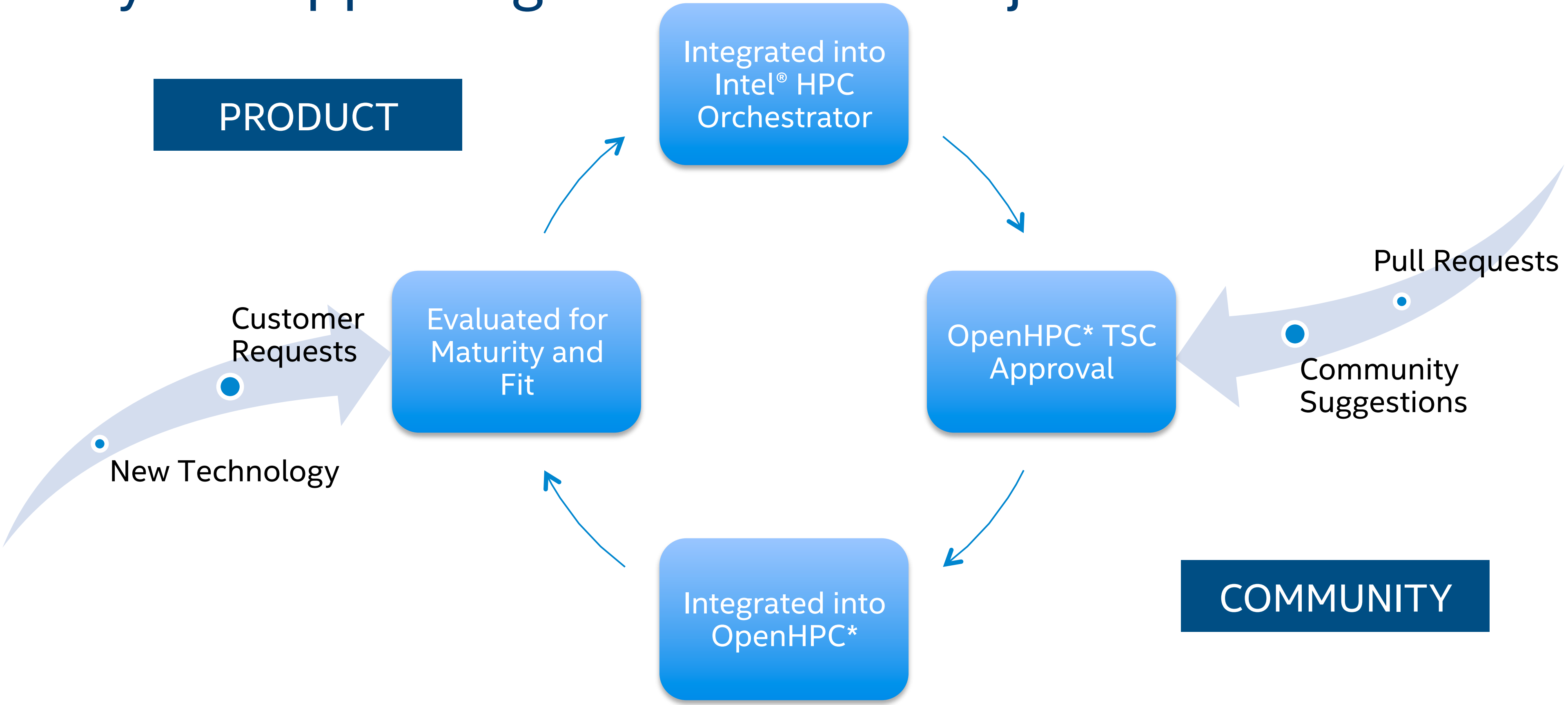
Intel® Scalable System Framework

Holistic Design Solution for All HPC

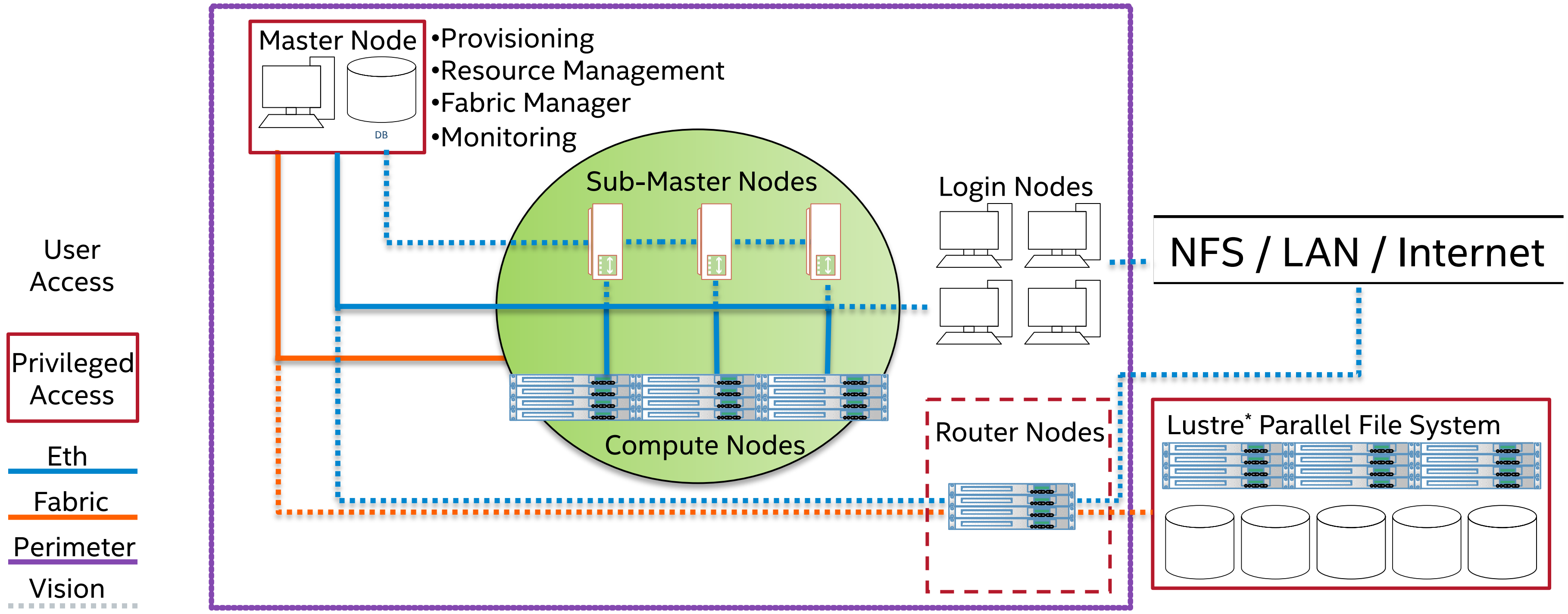


- Small clusters through supercomputers
- Compute and data-centric computing
- Standards-based programmability
- On-Premise and cloud-based

Cycle supporting Product and Project advancement



Intel® HPC Orchestrator System Architecture



Intel® HPC Orchestrator 16.01.004 – Components

Functional Areas	Components
Base OS Compatibility	RHEL 7.2, SLES12 SP1, CentOS 7.2
Administrative Tools	Conman, Powerman, Ganglia, Nagios, Lmod, pdsh, ClusterShell, EasyBuild, Spack, mrsh, Genders, Shine
Provisioning	Warewulf
Resource Management	Slurm, MUNGE
I/O Services	Lustre client (Intel® Enterprise Edition for Lustre)
Numerical/Scientific Libraries	Boost, GSL, FFTW, Metis, PETSc, Trilinos, Hypre, SuperLU, SuperLU_Dist, MUMPS, OpenBLAS, Scalapack
I/O Libraries	HDF5 (pHDF5), NetCDF (including C++ and Fortran interfaces), ADIOS
Compiler Families	GNU (gcc, g++, gfortran), Intel® Parallel Studio XE
MPI Families	MVAPICH2, OpenMPI, Intel® MPI
Developer Tools	Autotools (autoconf, automake, libtool), Valgrind, R, SciPy/NumPy
Performance Tools	PAP, IMB, mpiP, pdtoolkit, TAU

Intel® HPC Orchestrator Enhancements

- Advanced integration testing & extensive validation
- Professional support for
  - All Intel components
  - Components where Intel maintains a support contract
- Best Effort Support for all other components
- Enhanced Documentation
  - Components Description Guide
  - Troubleshooting Guide, including Knowledge Base
  - Readme, Release Notes
  - Technical Update
- Validated security patches & updates

Intel® HPC Orchestrator Enhancements

- Early new hardware integration with System Software
- Inclusion of proprietary Intel Software
  - Intel® Parallel Studio XE 2017 (Cluster Edition)[1]
  - Intel® Solutions for Lustre* (Client) [1]
- Planned Additional components
  - Support for high availability
  - Visualization tools
- SLES 12 SP1 Base OS redistribution avaialble
- Integrated Test Suite
- Intel® Cluster Checker Supportability Extensions

1 – Intel Parallel Studio XE 2017 and Intel Solutions for Lustre client license and support sold separately

Intel® Cluster Checker Supportability Extensions

New set of extensions to Intel® Cluster Checker

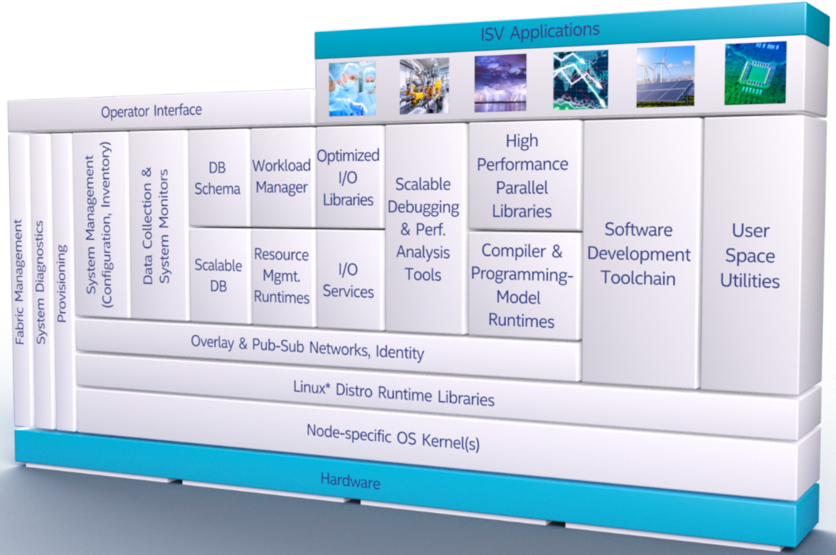
Baseline: system data collected when it is in a good, dependable state

Collects baseline data for:

- RPMs
  - Head Node
  - Virtual Node File System
- Configuration files (along with whitelist/blacklist capabilities)
- Hardware/Firmware

Compare current state of system with baseline

Intel® HPC Orchestrator: Summary



- Integrated open source and proprietary components
- Modular build; Customizable; Validated updates
- Advanced integration testing, testing at scale
- Level 3 technical support provided by Intel
- Optimization for Intel® Scalable System Framework components
- **Available through OEM & Channel Partners in Q4'16**

Benefits

OEMs – reduce R&D

ISVs/Developers – reduce time and man hours constantly retesting apps

IT Admins - reduce R&D to build and maintain a fully integrated SW stack

End Users - hardware innovation reflected in SW faster on path to exascale

Additional Sources of Information

OpenHPC* community – www.openhpc.community

Intel® HPC Orchestrator product page – www.intel.com/hpcorchestrator

Intel® Scalable System Framework – www.intel.com/ssf

THANK YOU!

Backup Slides

Intel® Cluster Checker Supportability Extensions

Collecting **RPM** baseline data

- Create nodefile

  # cat nodefile

```
[sms]# cat nodefile
sms #role: head
c1
c2
```

- Run rpm-baseline command

  # rpm-baseline –f <path-to-nodefile>

- Data captured in

  /var/tmp/rpms-baseline.txt

```
[sms]# cat /var/tmp/rpms-baseline.txt
sms, libpciaccess  0.13.4  2.el7  x86_64
.
.
.
c1, libpciaccess  0.13.4  2.el7  x86_64
.
.
.
c2, libpciaccess  0.13.4  2.el7  x86_64
.
.
.
```

Node name · RPM name · Version · Release · Architecture

Intel® Cluster Checker Supportability Extensions

Collecting **Files** baseline data

 # files-baseline –f <path-to-nodefile>

Data captured in /var/tmp/files-baseline.txt

```
[sms]# cat /var/tmp/files-baseline.txt
sms, /etc/sysconfig/httpd, -rw-r--r--, root, root, 65947590cfc1df04aebc4df81983e1f5
.
.
c1, /etc/os-release, -rw-r--r--, root, root, 1359aa3db05a408808522a89913371f3
.
.
.
c2, /etc/sysconfig/munge, -rw-r--r--, root, root, e0505efde717144b039329a6d32a798f
.
.
```

Permissions

File

Owner

Group

MD5 Sum

Intel® Cluster Checker Supportability Extensions

Collecting **Hardware** baseline data

 # hw-baseline –f <path-to-nodefile>

Data captured in /var/tmp/hw-baseline.txt

```
[sms]# cat /var/tmp/hw-baseline.txt
sms, 00:0d.0, Intel Corporation 82801HM/HEM (ICH8M/ICH8M-E) SATA Controller [AHCI mode]
.
.
c1, 00:03.0, Intel Corporation 82540EM Gigabit Ethernet Controller
c1, 00:07.0, Intel Corporation 82371AB/EB/MB PIIX4 ACPI
.
.
c2, 00:05.0, Intel Corporation 82801AA  AC'97 Audio Controller
.
.
```

Bus:Device.Function   Hardware description

Intel® Cluster Checker Supportability Extensions

Comparing/Analyzing:

- Collect current system state

  # clck-collect –f <path-to-nodefile> –m uname –m files_head –m files_comp

- Analyze current system state against baseline

  # clck-analyze –f <path-to-nodefile> –I files

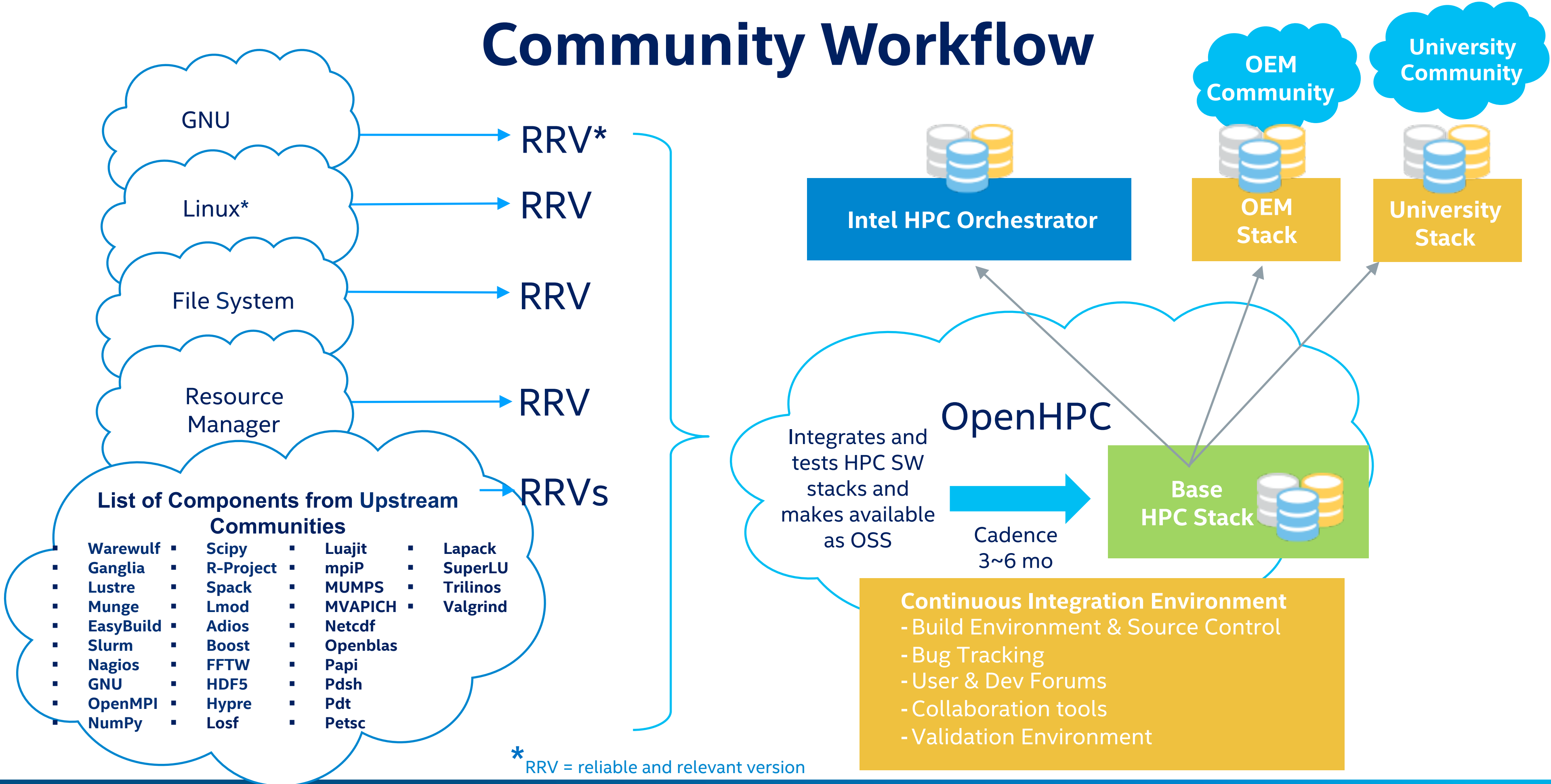
```
1 undiagnosed sign:

  1. The file /etc/pam.d/ppp has been added since the baseline was generated.
     [  Id: files-added  ]
     [  Severity: 25%; Confidence: 90%  ]
     [  Node: RHEL2  ]


This analysis took 0.388902 seconds.

FAIL:  All checks did not pass in health mode.
```

Community Workflow



*RRV = reliable and relevant version