Spack in Shared Environment

2018 Scalable Tools Workshop

LANL Programming and Runtime Environments Team
Paul Ferrell
The Program Runtime Env. Team

We support:

- 4 Compilers (GCC, Intel, PGI, cray)
- 4 MPI libraries (openmpi, mvapich2, intel-mpi, cray-mpich)
- ~64 packages total in production
- Continuous release schedule
- Package Life = System Life

No root access

Multiple Customers, Shared Environment
About our Environment

- > 12 systems
- 25000 Compute Nodes
- 3 OS's
- 5 Interconnects
- Many CPU architectures
- GPU clusters
- Lmod and Env Mod module systems
It’s all under control

• Solid, readable build scripts for everything
• Smart modules
  – Hide and manage the dep chain automatically, even under Env. Mod
• Shared builds (when it’s easy)
What Do We Lack?

- No automated inventory
- Each project is an island.
- Many (mostly) identical builds
- **No definition of what ‘should be’**
What Should be in our Environment?

Environment Definition → Build System → Programming Environment Q3 2018
Spack Package Manager

- Automated building
  - Multiple configurations of the same thing
- Dependency management
- Module management/auto-generation
- Comes with a large library of packages.
Does Spack Actually Help?

✓ Automated inventory → Awareness
✓ Each project is an island → Uniformity
✓ Many (mostly) identical builds → Efficiency
✗ No definition of what ‘should be’
The Problem with Spack...

- Mvapich2 v2.2
  - GCC v6.4.0
    - ...

- Mvapich2 v2.2
  - Intel v17.0.4
    - ...

- Mvapich2* v2.2
  - Intel v17.0.4
    - ...

...
Non-Conflicting Environments?

It’s Multiplicative:

12 systems
* 3 compilers
* 3 MPI’s
* (assorted versions)
* (other oddities)
= Just too much
Spack, as the Foundation

Spack is good at what it does.

Build everything else around it.
Wrapping Spack, Requirements

- Per (Environment Version) environment definitions
- Automated building of each environment
  - Host awareness
  - Scheduler awareness
- Separation of User Modules from Spack Modules
Spack Environment Wrapper

Environment Definition → Spack Wrapper → Spack (shared) → Programming Environment Q3 2018
Module Separation

bash $ module load mvapich2

Module System

Mvapich2.tcl

Spack Wrapper

- Which environment?
- What dependencies?
- Dependency defaults?
- Module default?

(The Real) mvapich2_(hash).tcl
Famous Last Words

• This should be easy!
Questions