Spack for EIC

2020-07-08

What is Spack

- github.com/spack: “A flexible package manager supporting multiple versions, configurations, platforms, and compilers.”
- spack.io: “Spack is a package manager for supercomputers, Linux, and macOS. It makes installing scientific software easy.”
- Benefits for EIC:
  - Support for environments of scientific software, natively compiled on HPC architectures
  - Entirely controlled by user (think `conda create myenv`, `conda activate myenv`)
- Disadvantages:
  - Yet another package manager when “everyone can just run `cmake .. && make`”
  - Primarily automates build from source; not a binary distribution system without add’l effort
  - Compiling is not a guarantee for valid results; no validation steps currently included

github.com/spack @spackpm

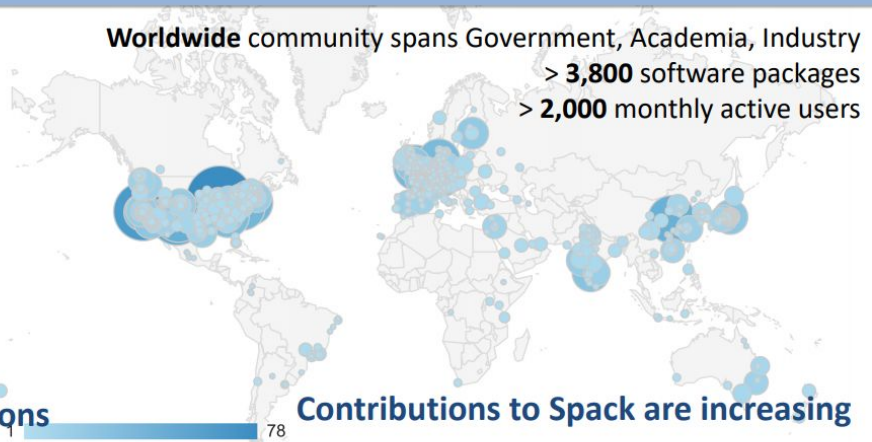
Spack

Building & Deploying the ECP Software Ecosystem

Gregory Becker*, Peter Scheibel*, Tamara Dahlgren*, Mario Melara†, Scott Wittenburg‡, Omar Padron‡, Bryon Bean‡, Zack Galbreath‡, Aashish Chaudhary, and Massimiliano Culpo§, and Todd Gamblin*
*Lawrence Livermore National Laboratory, †Lawrence Berkeley National Laboratory, ‡Kitware, Inc., § Sylabs, Inc.

Spack is enabling delivery of the exascale software stack

- ECP asks us to build a software stack that will have broad impact beyond DOE.
  - Needs to be robust, tested, and reliable
  - Needs to be easy to get up and running
- Spack will provide the infrastructure necessary to make this tractable through automation:
  1. A dependency model that can handle HPC software
  2. A hub for coordinated software releases (like xSDK)
  3. Build and test automation for large packages across facilities
  4. Hosted binary and source software distributions for *all* ECP HPC platforms



Worldwide community spans Government, Academia, Industry
> **3,800** software packages
> **2,000** monthly active users

Easy Installation

```
$ git clone https://github.com/spack/spack
$ . spack/share/spack/setup-env.sh
$ spack install hdf5
```

- Clone from github and you're ready to go!
- Sourcing configuration script is optional

Easily Experiment with Build Options

```
$ spack install mpileaks                    unconstrained
$ spack install mpileaks@3.3                @ custom version
$ spack install mpileaks@3.3 %gcc@4.7.3     % custom compiler
$ spack install mpileaks@3.3 %gcc@4.7.3 +threads   +/- build option
$ spack install mpileaks@3.3 cppflags="-O3 -g3"    set compiler flags
$ spack install mpileaks@3.3 target=skylake        set target microarchitecture
$ spack install mpileaks@3.3 ^mpich@3.2 %gcc@4.9.3 ^ dependency information
```

Contributions to Spack are increasing

Why Spack?

- While containers are the easiest and fastest way to start with a curated environment, some users express strong desire to run software natively.
- This is particularly relevant for large scale simulations on HPC systems (containers provide a solution through singularity, at reduced performance).
- Ideally we would like a single solution that works for general central installations (using CVMFS) as well as individual user systems.

What else exists?

- Scientific and HPC:
  - EasyBuild: non-NP/HEP community (latest are geant4.10.5, root6.14.06)
  - Conda: binary, python-centric (though also support for C++)
  - GuixHPC, NIX: binary, uncommon build recipe languages
- Non-HPC: portage, pkgsrc, homebrew

	CONDA	easybuild	GuixHPC	Nix	Spack
platforms	Linux, macOS, Windows	Linux, Cray	GNU/Linux	Linux, macOS, Unix	Linux, macOS, Cray
implementation	Python 2/3, YAML	Python 2	Scheme, Guile	C++, Nix (DSL)	Python 2/3
supp. software	> 3,500	> 2,000	< 6,500	> 13,000	> 2,300
releases, install & update	★★★½	★★★½	★★½	★★½	★★
documentation	★★★½	★★★	★★★½	★★★½	★★★½
configuration	★★★½	★★★	★★	★★	★★★½
usage	★★★	★★★	★★★	★★★	★★★
time to result	★★★★	★★½	★★★★	★★★★	★★★
performance	★★	★★★★	★★	★★	★★★
reproducibility	★★★	★★	★★★★	★★★★	★★

FOSDEM'18, https://archive.fosdem.org/2018/schedule/event/installing_software_for_scientists

EIC Spack: Status Update

- Spack upstream repository at github.com/spack/spack:
  - Pull requests merged (new or bugfixes): geant4, opencascade, delphes, lhapdf, hepmc
- Spack forked repository at github.com/eic/spack:
  - Default master branch is latest upstream release (v0.15)
  - Upstream develop branch is periodically merged into forked develop
  - Intended for new package and bugfix branches, pull requests to upstream
- EIC Spack repository at github.com/eic/eic-spack:
  - EIC-specific software packages, stable in default branch:
    - `bmf dire eicroot eic-smear eictoymodel ejana escalate g4e geant3-vmc geant4-vmc genfit jana2 lhapdf5 libodbcpp nanocernlib py-pyminuit2 py-qmtest pythia6m tricktrack vgm vmc`
  - Branches with packages in development (still failing to build):
    - `fun4all milou pythia6m pepsi djangoh rapgap`
  - Candidates for submission to upstream repository:
    - `geant3-vmc geant4-vmc vgm vmc libodbcpp lhapdf5`

EIC Spack: Interface to CVMFS

- Goal:
  - Providing single entry point to using all EIC software through CVMFS
    - Taking advantage of spack with environment (for all systems) setup with:
      `source /cmvfs/eic.opensciencegrid.org/packages/setup-env.sh`
    - Next, user can load any software components they need with e.g.
      `spack load geant4@4.10.06.p01`
      `spack load escalate@1.0.1` (loads environment with all dependencies)
  - Support a cross section of frequently used operating systems (with focus on HPC)
- Status:
  - Full software stack with all of Escalate, EicRoot, EicToyModel is ‘working’ on CVMFS
    - ‘Working’ = no known issues, with admittedly only limited testing
- Outlook:
  - Add more MC event generators soon

EIC Spack: Developer Support Welcome

- Releases:
  - While spack packages support e.g. eic-smear@master, released version are preferred (spack package developer can pick a commit hash and tag it by date)
  - Developers: Please release at least one appropriate version for inclusion in spack.
- Testing:
  - A user installing a package wants to rely on the package working as expected. A package maintainer wants to have some more rigorous way of testing the package.
  - Developers: Support a test build target that fails when something obviously went wrong.
- Collaboration:
  - Think about your software in the context of a suite of software packages: requiring ROOT with c++11 conflicts hard with some other software; requiring specific version constrains the dependency graph.

EIC Spack: Versioning

While the discussion of versioning is independent of using spack, we do need some versioning scheme to delivering consistent software to the users.

- We cannot guarantee that all versions of all software will work well together.
  - Using meta package releases provides a target for testing that some versions do work well.
- In the absence of semantic versioning, there is no way a user can distinguish minor and major upgrades in a consistent way for all software packages.
  - If there are no versions at all, the only info they have is time gaps between commits.
  - Using meta package releases provides guidance to the users in a curated way.

EIC Spack: Other Random Rollout Issues

- May need to expand operating systems to support
  - Currently all based on RHEL7, which seems to load just fine on Ubuntu20.04 etc.
- May need build server and binary buildcache
  - Already have docker builders to create binary packages for distribution

Backup slides.

EIC Spack: How To Get Started?

- spack.readthedocs.io:
  - `git clone https://github.com/spack/spack.git`
  - `export SPACK_ROOT=`realpath spack``
  - `export PATH=$SPACK_ROOT/bin:$PATH`
  - `source $SPACK_ROOT/share/spack/setup-env.sh`
  - `spack install root`
- Find packages: `spack list root`
- Info on packages: `spack info root`
- Use variants: `spack install root@6.14.04 +pythia8`
- Load packages: `spack load root` (like `module load root`)
- Load environment (like `conda env`):
  - `spack env create myenv`
  - `spack env activate myenv`
  - `spack env deactivate myenv`

EIC Spack: Writing Packages

- No packages for: lhapdf, genfit, dire, vgm, g4e, eic-smear
- From source location, e.g.
  - `spack create https://gitlab.com/eic/eic-smear`
  - Imports released version, supports git branches (`spack install eic-smear@master`)
  - Autodetection of build system not always successful (eic-smear needed cmake hint)
- Package recipe in repos/builtin/eic-smear/package.py

```
class EicSmear(CMakePackage):
    """Monte Carlo analysis package developed by BNL."""

    homepage = "https://wiki.bnl.gov/eic/index.php"
    url      = "https://gitlab.com/eic/eic-smear"
    git      = "https://gitlab.com/eic/eic-smear.git"

    variant("pythia6", default=False,
        description="Include Pythia6 support")

    version('master', branch='master')
    version('1.0.4',  branch='1.0.4')
    version('1.0.3',  branch='1.0.3')
    version('1.0.2',  branch='1.0.2')
    version('1.0.1',  branch='1.0.1')

    depends_on('root')
    depends_on('cmake', type='build')
    depends_on('pythia6', when='+pythia6')

    def cmake_args(self):
        args = []
        if self.spec.variants['pythia6']:
            args.append('-DPYTHIA6_LIBDIR={0}'.format(
                self.spec['pythia6'].prefix.lib))
        return args
```

EIC Spack: Repositories

- Builtin repository though pull request on github.com/spack
  - “Your PR must pass Spack's unit tests and documentation tests, and must be PEP 8 compliant. We enforce these guidelines with Travis CI.”
- Dedicated repositories with `git repo add`
  - `git clone https://github.com/eic/eic-spack.git`
  - `spack repo add eic-spack`
  - `spack install eic-smear`
- Binary distribution through build caches (with `http` mirror)
  - `spack gpg init`
  - `spack gpg create \`git config --get user.name\` \`git config --get user.email\``
  - `spack buildcache create -d ~/scratch/spack/ root`
  - `spack mirror add data file://$HOME/scratch/spack/`
  - `spack buildcache list`
  - `spack buildcache install`

EIC Spack: Containers

- From environments to Docker containers
  - `spack env create myenv`
  - `spack env activate myenv`
  - `spack install eic-software-stack`
  - `spack env deactivate myenv`
  - `spack containerize myenv > Dockerfile`

```
# Build stage with Spack pre-installed and ready to be used
FROM spack/ubuntu-bionic:latest as builder

# What we want to install and how we want to install it
# is specified in a manifest file (spack.yaml)
RUN mkdir /opt/spack-environment \
&&  (echo "spack:" \
&&   echo "  specs:" \
&&   echo "  - eic-smear" \
&&   echo "  view: /opt/view" \
&&   echo "  concretization: together" \
&&   echo "  config:" \
&&   echo "    install_tree: /opt/software") > /opt/spack-environment/spack.yaml

# Install the software, remove unnecessary deps
RUN cd /opt/spack-environment && spack install && spack gc -y

# Strip all the binaries
RUN find -L /opt/view/* -type f -exec readlink -f '{}' \; | \
    xargs file -i | \
    grep 'charset=binary' | \
    grep 'x-executable\|x-archive\|x-sharedlib' | \
    awk -F: '{print $1}' | xargs strip -s

# Modifications to the environment that are necessary to run
RUN cd /opt/spack-environment && \
    spack env activate --sh -d . >> /etc/profile.d/z10_spack_environment.sh

# Bare OS image to run the installed executables
FROM ubuntu:18.04
COPY --from=builder /opt/spack-environment /opt/spack-environment
COPY --from=builder /opt/software /opt/software
COPY --from=builder /opt/view /opt/view
COPY --from=builder /etc/profile.d/z10_spack_environment.sh /etc/profile.d/z10_spack_environment.sh
ENTRYPOINT ["/bin/bash", "--rcfile", "/etc/profile", "-l"]
```