Preliminary evaluation of Spack as a new backend for LCG releases

R. Bachmann, G. Ganis, D. Konstantinov, I. Razumov

HSF Packaging group meeting, 11 December 2019

Introduction

What is Spack?

Spack is a package management tool designed to support multiple versions and configurations of software on a wide variety of platforms and environments. It was designed for large supercomputing centers, where many users and application teams share common installations of software on clusters with exotic architectures, using libraries that do not have a standard ABI. Spack is non-destructive: installing a new version does not break existing installations, so many configurations can coexist on the same system.

Introduction

Why Spack?

- Is recommended by HSF Packaging group

[a] to build their software on top of existing LCG release

Introduction

Why Spack?

- Is recommended by HSF Packaging group
- After CHEP'19 talk by G. Stewart, the Experiments (ATLAS and LHCb) has expressed interest in evaluating Spack

[a] to build their software on top of existing LCG release

Introduction

Why Spack?

- Is recommended by HSF Packaging group
- After CHEP'19 talk by G. Stewart, the Experiments (ATLAS and LHCb) has expressed interest in evaluating Spack
- Already used by FCC[a], FAIR

[a] to build their software on top of existing LCG release

Introduction

Why Spack?

- Is recommended by HSF Packaging group
- After CHEP'19 talk by G. Stewart, the Experiments (ATLAS and LHCb) has expressed interest in evaluating Spack
- Already used by FCC[a], FAIR
- ALICE has also expressed interest during the LIM of 3 December 2019

[a] to build their software on top of existing LCG release

The Plan

- As a first step, we will try to use Spack to build an equivalent of LCG_96b for one platform (x86_64-centos7-gcc8-opt) for the experiments to test.
- It will contain only the packages for which recipes are already present in Spack
- If the test is successful, we will start implementing missing recipes

Usage model

The foreseen usage model of Spack in LCG is quite different from a typical usage of Spack:

- Take as much from the underlying OS as possible
  - X11 libraries, common tools (diff, patch, tar, ...), etc.

Usage model

The foreseen usage model of Spack in LCG is quite different from a typical usage of Spack:

- Take as much from the underlying OS as possible
  - X11 libraries, common tools (diff, patch, tar, ...), etc.
- Use pre-built compilers

Usage model

The foreseen usage model of Spack in LCG is quite different from a typical usage of Spack:

- Take as much from the underlying OS as possible
  - X11 libraries, common tools (diff, patch, tar, ...), etc.
- Use pre-built compilers
- Build and test the entire stack ($\sim$ 460 packages) every day
  - We don't rebuild every single package, but reuse previously built ones (if they match)

Evaluation

The Good:

- Spack have an easier syntax (Python) for recipes, has templates (e.g. for autoconf or python packages)
  - But the core code of Spack itself is quite complicated
- A lot of "hacks" we have or want in lcgcmake (e.g. setting a fixed version of dependency, using git commit id in place of version, "layered" releases[1]) are already implemented in Spack
- Recipes for many packages are already available in Spack: of 387 "external" packages (i.e. everything but generators), Spack has recipes for about 279.
  - We are missing a bunch of Python packages (machine learning, Jupyter add-ons), all GRID packages, a few 'go' modules.
  - Migrating the recipes for them is not expected to be a difficult task, just time consuming.
  - Detailed comparision: link

[1] a.k.a. chained installations

Evaluation

The Not-so-Good:

- Spack can only build one package at a time
  - Do not know yet the impact of this on regular builds
  - Support for parallel builds is expected by February

Evaluation

The Not-so-Good:

- Spack can only build one package at a time
- In some cases (e.g. openblas, font-util, Geant4) the recipes differ significantly between spack and lcgcmake
  - We can, for now, work around this by overwriting the recipes for such packages with our own (spack allows that)
  - Will try and push the changes upstream.

Evaluation

The Not-so-Good:

- Spack can only build one package at a time
- In some cases (e.g. openblas, font-util, Geant4) the recipes differ significantly between spack and lcgcmake
- The concretizer (a part of Spack that resolves versions and dependencies of packages) has a few problems :
  - In some cases (e.g., python packages like matplotlib) one needs to explicitly specify versions of dependencies, otherwise the concretizer will pick a default one, and will complain about incompatibility
  - "Virtual" packages (i.e. packages that have more than one implementation: blas, lapack, java) are often not resolved correctly — even when you explicitly set what package provides what in packages.yaml.
    - E.g., it tends to use just about anything but Netlib's implementation of LAPACK.
    - Had to edit recipes to fix that.

Evaluation

The Bad:

- Like lcgcmake, Spack uses hashes to uniquely identify a "concretized" package. However, the hashing in Spack seems to be excessive: the entire package.py file is hashed. That means even a slightest change in the recipe file (e.g. adding or removing whitespace or comment) will change the hash of a package. The developers are aware of the issue (see, e.g., #1325, #2247), but no ETA for a fix.
  - This is critical for us: building the stack every night relies on re-using the previous build.

Additional questions

- Views concept
  - probably not the same as in LCGCMake
- Use of variants for restricting instruction sets (AVX512, SSE4.2)
- Use rpath (default), runpath or remove relocation information and use LD_LIBRARY_PATH (possible)
- Setting the Environment: use modules or views

Next steps

- Create/revive dedicated Spack fork
  - Under GitHub/HSF
- Create EOS area for sources, binary hosting
- Provide test build as binary and under CernVM-FS
  - Under sw.hsf.org ?
- Provide documentation page under LCGDocs
- SPI-JIRA Epic to track evolution and issues/wishes