Software management with Spack

Azat Khuziyakhmetov

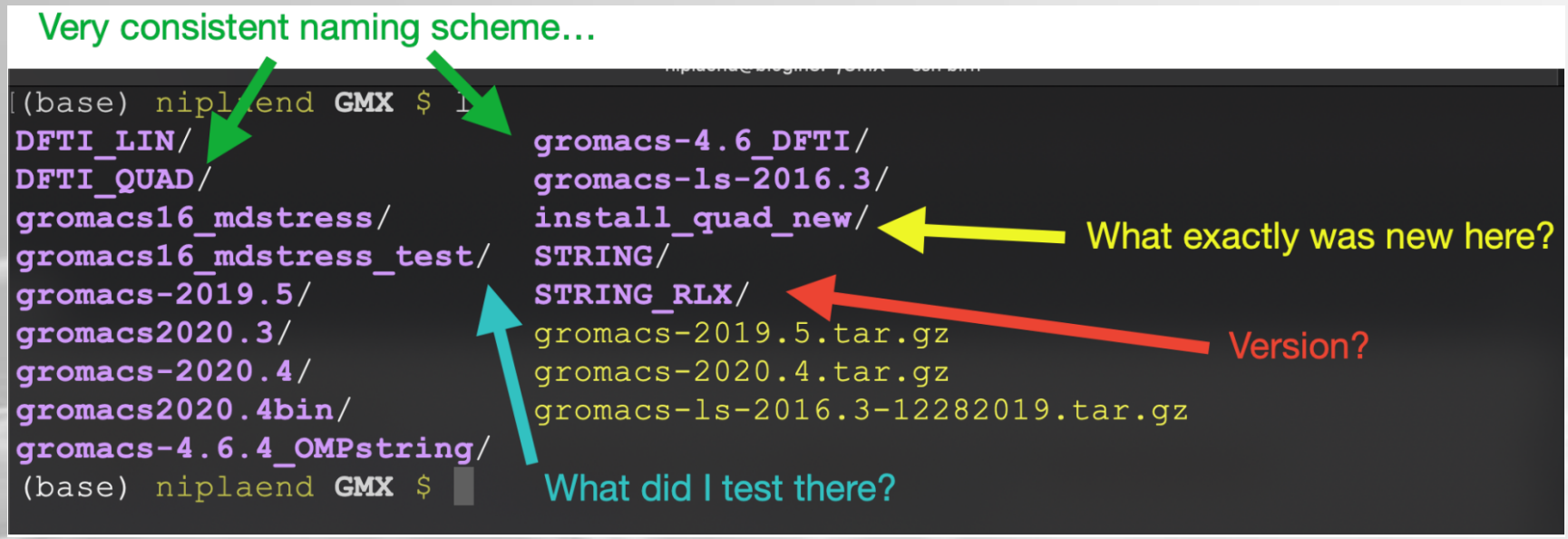
Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen
Burckhardtweg 4, 37077 Göttingen

02, April 2026

Learning objectives

- Understand the challenges of software installations in HPC.
- Formulate valid spack SPECS
- Install packages on the cluster using Spack.
- Manage different software versions.

Introduction



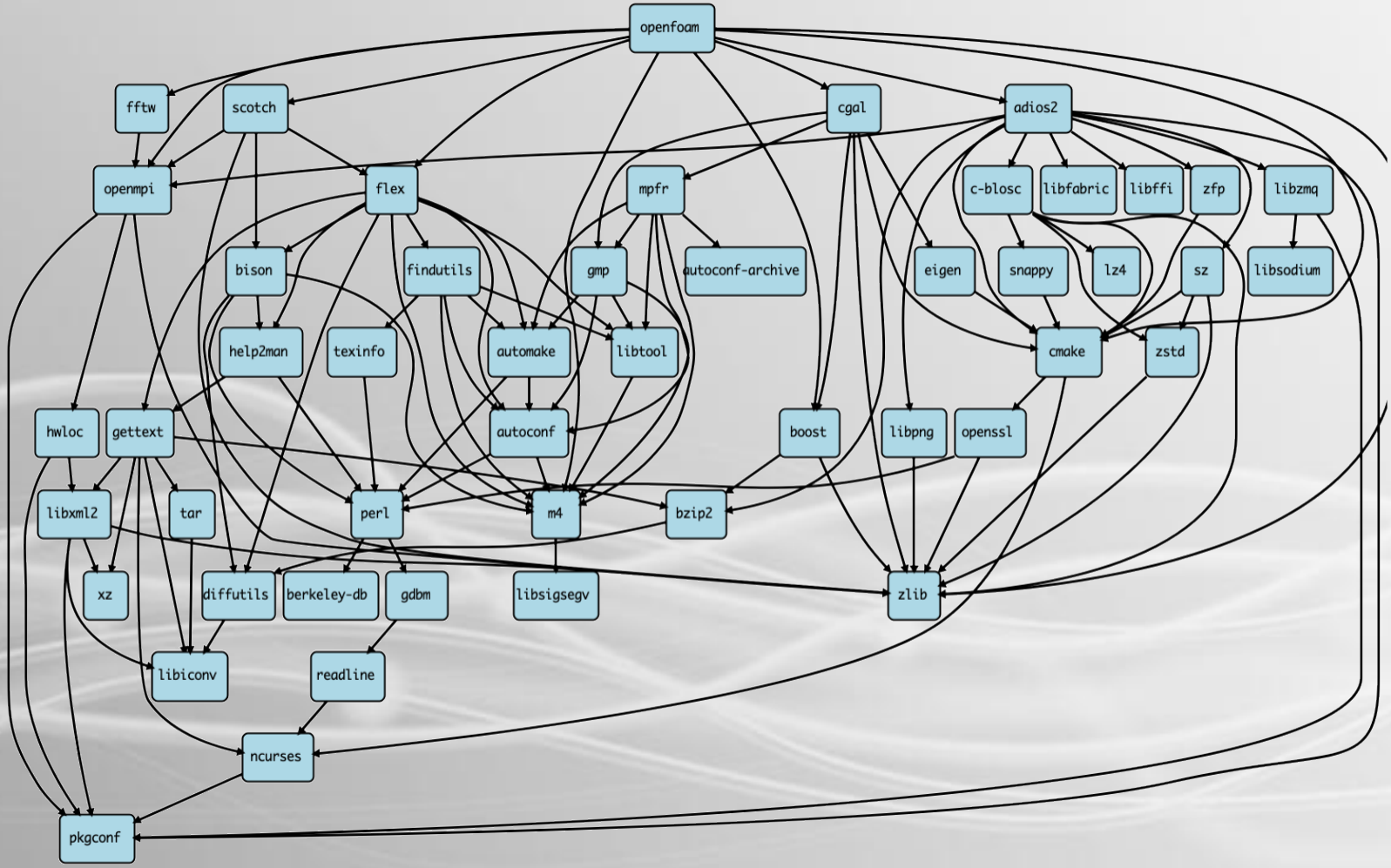
Installing software on a Cluster

- Prepackaged software typically requires administrator (root) privileges
- Admins cannot install all software required by users
- software installation very complex
- Different tools may need different versions
- Often trade reuse and usability for performance
- Compiling on the target system often yields better performance

HPC Software is complex

- Coexistence of several builds
- Specific versions of compilers, MPI, libraries, dependencies
- Often many dependencies

Openfoam dependency tree



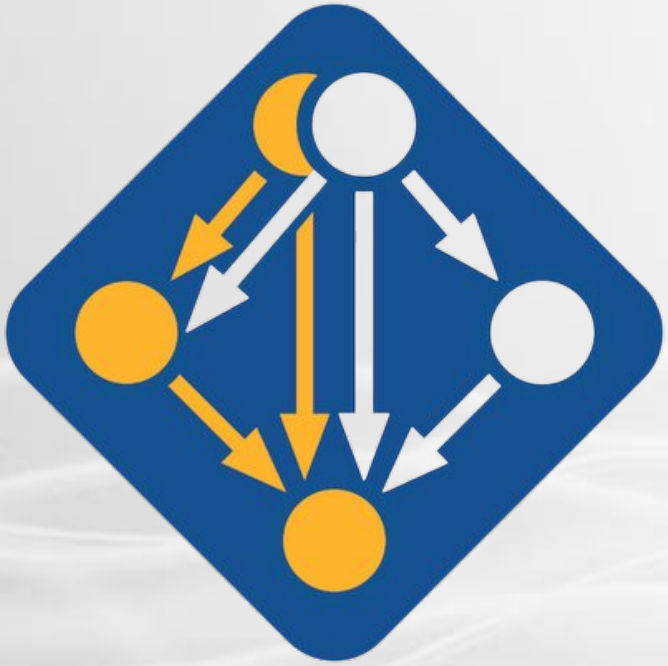
HPC Software is complex

- Specific versions of compilers, MPI, libraries, dependencies
- Often many dependencies
- Many compiling options
- Users active on several clusters

Solution: **Spack**

Spack

Spack: Supercomputer PACKage manager



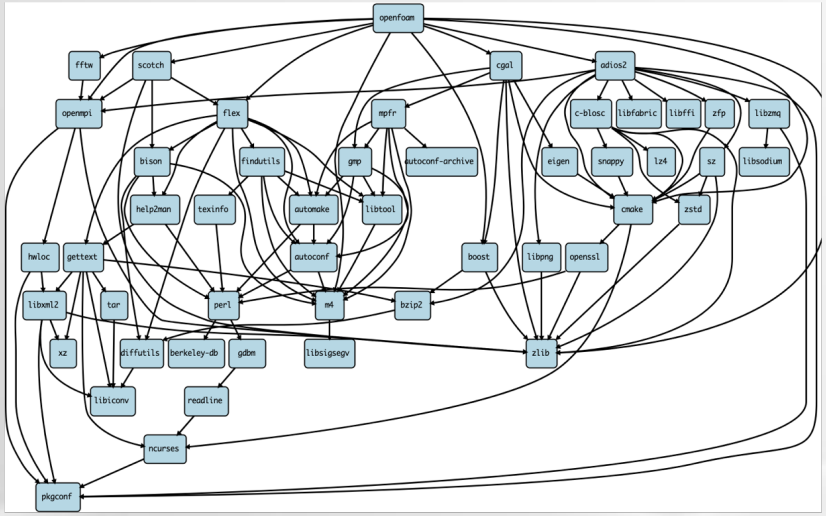
- Manages multiple builds
- Takes care of dependency relationships
- Drives package-level build systems
- Wrapper around built systems (cmake/autotools/make etc.)
- Targeted towards users, admins, and developers

Spack basics

- Bunch of Python scripts
  - 97.1% according to GitHub
- Packages maintained by the developers and the community
  - Currently 7939 available packages
  - https://github.com/spack/spack
  - You can contribute!

Spack basics

- Dependency graphs are translated into a hash
  - unique identifier for each build (if all aspects of a build are identical - same hash)



openfoam-k5hyzc7e6meneqldqfguw2nddghqfz52

- Spack uses RPATH and PATH to ensure dependencies are found

Spack on the SCC

- Most of the modules on the cluster are installed using Spack
- Spack itself is installed as a module `spack-user`
  - Easily find the installed software
  - Reuse packages that are already installed
  - manage your own builds

How to use Spack?

- Load the spack-user module with `module load spack-user`
- Follow the instructions to activate the spack shell support
- Install supported software `spack install SPEC`
- Load the installed software `spack load SPEC`

SPEC syntax

- SPECS specify the software configuration
  - i.e. constraints YOU set for your installation
  - optional: only specify what you need

`$ spack install mpileaks`	Unconstrained
`$ spack install mpileaks@3.3`	@ custom version
`$ spack install mpileaks@3.3 %gcc@4.9.3`	% custom compiler
`$ spack install mpileaks@3.3 %gcc@4.9.3 +threads`	+/-/~ build option
`$ spack install mpileaks@3.3 cppflags="-O3 -g3"`	Set compiler flags
`$ spack install mpileaks@3.3 target=cascadelake`	Set CPU architecture
`$ spack install mpileaks@3.3 ^mpich@3.2 %gcc@4.9.3`	^ dependency information

Example with nano

```
gwdu102:61 23:24:18 ~ > spack install nano
[+] /opt/sw/rev/21.12/haswell/gcc-9.3.0/pkgconf-1.8.0-ktrb7r
[+] /opt/sw/rev/21.12/haswell/gcc-9.3.0/ncurses-6.2-2dvkjs
==> Installing nano-4.9-hyadazgagq6itbtp4dwnrgis3xpyrxky
==> No binary for nano-4.9-hyadazgagq6itbtp4dwnrgis3xpyrxky found: installing from source
==> Using cached archive: /usr/users/ttaboug/.spack/0.17.1/cache/_source-cache/archive/0e/0e399729d105cb1a587b4140db5cf1b23215a0886a42b215efa98137164233
a6.tar.xz
==> No patches needed for nano
==> nano: Executing phase: 'autoreconf'
==> nano: Executing phase: 'configure'
==> nano: Executing phase: 'build'
==> nano: Executing phase: 'install'
==> nano: Successfully installed nano-4.9-hyadazgagq6itbtp4dwnrgis3xpyrxky
  Fetch: 0.02s.  Build: 49.99s.  Total: 50.01s.
[+] /usr/users/ttaboug/.spack/0.17.1/install/haswell/gcc-9.3.0/nano-4.9-hyadaz
```

- spack find -vdl nano
- spack install nano
- spack load nano
- nano –version

The Most Important Command

`spack help`

- overview over all spack commands

`spack help <command>`

- provides information about usage and available options for all spack commands
- usage will help you to get familiar with spack

Basic Spack Commands

spack list

- list of all available packages for spack

spack find

- list of all already installed packages

spack compilers

- list of all installed compilers

spack info

- print information about a spack package

Let's install something!

- Follow the Tutorial
  - `https://pad.gwdg.de/s/rIr_42Bfd`
  - Experiment with the various spack commands!
  - Discuss with your group!
  - Ask us if you've got any questions or need assistance!
- Use our Spack cheat sheet!
  - `https://pad.gwdg.de/s/PvTiJDp3k`