NUG Monthly Meeting

18 November, 2021
Today's plan

- Interactive - please participate!
  - Raise hand or just speak up
  - **NERSC User Slack** (link in chat), **#webinars** channel

- Agenda:
  - Win-of-the-month
  - Today-I-learned
  - Announcements/CFPs
  - Topic of the day: **Spack at NERSC** by Steve Leak.
  - Coming meetings: topic suggestions/requests?
  - Last (two!) month's numbers
Win of the month

Show off an achievement, or shout out someone else's achievement, e.g.:

- Had a paper accepted
- Solved a bug
- A scientific achievement (maybe candidate for Science highlight, or High Impact Scientific Achievement award)
- An Innovative Use of High Performance Computing (also a candidate for an award) ([https://www.nersc.gov/science/nersc-hpc-achievement-awards/](https://www.nersc.gov/science/nersc-hpc-achievement-awards/))

Please let us know of award-worthy work from you or your colleagues - tell us what you did, and what was the key insight?
### Perlmutter's SC21 Top500 Wins!

#### #5 overall (HPL) 70.87PF
- **Rank 1**
  - **System**: Supercomputer Fujitsu
  - **Rank**: #5 overall (HPL)
  - **Performance**: 70.87PF

#### #7 Green 27.37GF/W
- **Rank 2**
  - **System**: Green 3.19PF 27.37GF/W

#### #3 HPCG 1.9PF
- **Rank 3**
  - **System**: HPCG 1.9PF

### Green500 Data

<table>
<thead>
<tr>
<th>Rank</th>
<th>Name</th>
<th>System</th>
<th>Green500</th>
<th>Rank</th>
<th>Power</th>
<th>Power Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>#5 overall (HPL) 70.87PF</td>
<td>#5 overall (HPL) 70.87PF</td>
<td>70.87PF</td>
<td>5</td>
<td>37.72</td>
<td>27.85</td>
</tr>
<tr>
<td>2</td>
<td>#7 Green 27.37GF/W</td>
<td>#7 Green 27.37GF/W</td>
<td>27.37GF/W</td>
<td>7</td>
<td>27.37</td>
<td>27.37</td>
</tr>
<tr>
<td>3</td>
<td>#3 HPCG 1.9PF</td>
<td>#3 HPCG 1.9PF</td>
<td>1.9PF</td>
<td>3</td>
<td>1.90</td>
<td>1.90</td>
</tr>
</tbody>
</table>

### Top500 List

<table>
<thead>
<tr>
<th>Rank</th>
<th>Name</th>
<th>System</th>
<th>Green500</th>
<th>Rank</th>
<th>Power</th>
<th>Power Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>#5 overall (HPL) 70.87PF</td>
<td>#5 overall (HPL) 70.87PF</td>
<td>70.87PF</td>
<td>5</td>
<td>37.72</td>
<td>27.85</td>
</tr>
<tr>
<td>2</td>
<td>#7 Green 27.37GF/W</td>
<td>#7 Green 27.37GF/W</td>
<td>27.37GF/W</td>
<td>7</td>
<td>27.37</td>
<td>27.37</td>
</tr>
<tr>
<td>3</td>
<td>#3 HPCG 1.9PF</td>
<td>#3 HPCG 1.9PF</td>
<td>1.9PF</td>
<td>3</td>
<td>1.90</td>
<td>1.90</td>
</tr>
</tbody>
</table>
Today I learned

What surprised you that might benefit other users to hear about? (and might help NERSC identify documentation improvements!)

Eg:

- Something you got stuck on, hit a dead end, or turned out to be wrong about
  - Give others the benefit of your experience!
  - Opportunity to improve NERSC documentation
- A tip for using NERSC
- Something you learned that might benefit other NERSC users

"If we knew what it was we were doing, it would not be called research, would it?" - Einstein
Announcements and CFPs

See weekly email for these and more:

- Annual User Survey is now open!
  - Look for an email from NERSC@nbriresearch.com
- In related news: you can see the [NERSC 2020 Annual Report](#) online (one thing the survey contributes to). Also the [2020 User Demographics](#)
- Heads up: User information is transmitted to DOE Office of Science at end of year
  - Includes name, institutional affiliation(s), and project title(s)
- New default python module coming in January
Perlmutter Announcements

- **Breaking news**: Perlmutter User Training originally scheduled for early December will most likely be **rescheduled** to January 5-7, 2022.
- Users with GPU-ready workloads can [request access to Perlmutter by filling out this form](#).
- Prepare your dotfiles for Perlmutter!
  - $HOME is shared across both systems, but each system has its own modules (and module system), etc.
  - Check $NERSC_HOST before making system-specific settings in your .bashrc / .cshrc.
Announcements and CFPs

- Apply for Prestigious **Alvarez & Hopper Postdoctoral Fellowships** in Computing Sciences at Berkeley Lab & NERSC by **Next Monday, November 22**
- Applications for DOE Computational Science Graduate Fellowship are now open
  - For first- and second- year PhD students
- Call for Proposals: Quantum Information Science on Perlmutter
- Nominate someone for the James Corones Award in Leadership, Community Building & Communication
  - Mid-career scientist or engineer making an impact in leadership, community building, and scientific communication
- Others?
Spack at NERSC
"Supercomputing Package Manager"

Automates software installation

Tip: All of today's topic is in our docs! =>
(and the docs have links to other Spack information)
Why use it?

- Building software is laborious
  - And error prone
- Spack automates a lot of the busy-work
  - Including the details of getting the right invocation to build the software with the options you want
- Recommendation: Use Spack as the first option for installing software
Why not use it?

- Scientific software is complex
- Automation is complex
- Complex²!
- The details of what Spack did, and why, are often opaque
- When something fails, finding why (and fixing it) is usually absurdly difficult
- Recommendation: If the fix isn't easy, stop digging
  - Move to a different build method
Today's goal: Demystify Spack and describe NERSC's Spack setup

• Jargon dictionary
• Essential Spack commands
  • Working with environments
  • How Spack decides what to install
• Spack idiosyncrasies
• Spack setup at NERSC
• Workflow for installing software with Spack
• What to do when it doesn't work
• Q&A
Spack words, and what they mean

**Package:** "Source code" describing a piece of software and how to build it (actually a Python class), along with any patches etc that might need to be applied first

```
/global/common/sw/spack/0.16.1/var/spack/repos/builtin/packages/zlib
15:52 sleak@cori04:lib$ ll
total 7
drwxrwxr-x+ 2 swowner swowner 512 Apr 7 10:28 __pycache__
-rw-rw-r--- 1 swowner swowner 2074 Apr 3 13:40 package.py
-rw-rw-r--- 1 swowner swowner 564 Mar 30 19:16 w_patch.patch
15:52 sleak@cori04:lib$ l
```
Spack words, and what they mean

**Repo:** A collection ("repository") of packages. Pretty much everything is in the "builtin" repo, but Spack has a "repos" config section where you can specify locations and order of repos to search.

```
$ ls -l /global/common/sw/spack/0.16.1/var/spack/repos/builtin
total 513
drwxrwxr-x+ 5066 swowner swowner 262144 Apr  3 13:40 packages
-rw-rw-r---  1 swowner swowner   27 Mar 30 19:16 repo.yaml
$ cat repo.yaml
repo:
  namespace: builtin
$ ls -lt packages/ | head
total 5106
drwxrwxr-x+  3 swowner swowner    512 May 11 17:29 clingo
```

```
Spack words, and what they mean

**Spec:** Spack has a DSL for declaratively describing the parameters with which a package should be (or was) built

```
spack spec -Il slate@2020.10.00%nvhpc +cuda ^blaspp cuda_arch=80
```

- Install package "slate"
- "at" version 2020.10.00
- "compiled with" nvhpc
- "enable" variant cuda
- "on" an install of blaspp
- "where" variant cuda_arch is set to 80 (NVidia A100)
Spack words, and what they mean

Spec: (cont'd)
Given a partial spec, and defaults from packages and Spack configuration, Spack uses an ASP solver to work out a detailed ("concrete") spec

```
sleak@cgpu12:~$ spack spec -Ih slate@2020.10.00%nvhpc +cuda ^blaspp cuda_arch=80
Input spec
-------------------------------
   slate@2020.10.00%nvhpc+cuda
       ^blaspp cuda_arch=80
-------------------------------
Concretized
-------------------------------
   how4q4p slate@2020.10.00%nvhpc@20.11+cuda-ipo+mpi+openmp+shared_build_type=RelWithDeb
       ^blaspp@2020.10.02%nvhpc@20.11+cuda-ipo+openmp+shared_build_type=RelWithDeb
       ^cmake@3.18.4%gcc@8.1.1~doc+ncurses+openssl+ownlibs+qt+patches=b695e
       ^pkg-config@0.29.2%nvhpc@20.11+internal_glib arch=linux-opens
       ^nss@6.2%nvhpc@20.11+shared-symlinks+term_lib arch=linux-opens
       ^rhshpcp
       ^jofnia
       ^opencv@2.1.0-fips%nvhpc@20.11+pic+shared+systme cert arch=linux-opens
       ^asufi7l
       ^cuda@11.1.0%nvhpc@20.11 arch=linux-opensuse_leap15-skylake_avx512
       ^bxs47ff
       ^libxml2@2.9.10%nvhpc@20.11+python+shared patches=05f238cf435825
       ^libiconv@1.16%nvhpc@20.11~static arch=linux-opensuse_leap15-skylate_avx
       ^lzop@1.2.5%nvhpc@20.11~pic+shared arch=linux-opensuse_leap15-sky
       ^f2s2ygo
       ^zlib@1.2.11%nvhpc@20.11+optimize+pic+shared arch=linux-opens
       ^kxotamv
       ^openblas@0.3.5%nvhpc@20.11~consistent_fpcsr~ilp64+pic+shared_threa
       ^hargyz6
       ^lapackpp@2020.10.02%nvhpc@20.11~ipo+shared build_type=RelWithDeb
       ^3vba3ib
       ^netlib-scalapack@2.1.0%nvhpc@20.11~ipo+pic+shared build_type=Release pat
       ^dscd243
       ^openmpi@4.0.5%nvhpc@20.11~atomics~cuda~cxx~cxx_exceptions+gpf
```
Spack words, and what they mean

Variant: A selectable build option for a package (defined in Spack in the package definition). Usually corresponds to a `./configure` option or an optional dependency.

You can enable (+), disable (- or ~) or set (name=value) a variant.

```
slate@2020.10.00%gcc@10.2.0+cud@ipo-mpi+openmp-shared build_type=RelWithDebInfo arch=cray-sles15-zen2
```
Hash: Spack calculates a hash of each full concrete spec (including dependencies), and uses it as a key to identify the install. This turns out very handy for distinguishing between the many possible builds of the same software.

```
sleck@perlmutter:login40:~> spack find -l arch=$(spack arch)
  ==> 11 installed packages
-- cray-sles15-zen2 / gcc@10.2.0 ----------------------------------------------
  7vizuc5 clingo@master          rwrjtra py-cffi@1.14.3
  vddikvd cmake@3.18.4           py-pycparser@2.20
  5mrkqr6 libffi@3.3             dnykyom py-setuptools@50.3.2
  qgdrsrj ncurses@6.1.20180317  u4gjofd python@3.8
  kca7yop openssl@1.1.0i-fips    yfkkzua re2c@1.2.1
  pq52cin pkg-config@0.29.2
```
**Install_tree:** The location Spack will install stuff in. The location and the directory-naming convention are defined in a config file.

```
16:04 sleak@cori04:~$ spack config blame config
---
/global/common/sw/spack/0.16.1/etc/spack/config.yaml:16
/global/common/sw/spack/0.16.1/etc/spack/config.yaml:17
/global/common/sw/spack/0.16.1/etc/spack/config.yaml:18
/global/common/sw/spack/0.16.1/etc/spack/config.yaml:19
```

**Tip:** You can override the location for $SPACK_BASE by setting it in your .bashrc or .bash_profile (default is $HOME/sw)
Install_tree: (cont'd)
NERSC setup puts the install_tree in the user's
$HOME, and organizes installs by target architecture

Spack adds some indexing information to each install, and caches it in .spack_db/ at the install_tree root
Upstream: Another install_tree (but read-only) that Spack is allowed to use. Eg if "netcdf" requires "hdf5", and "hdf5" is installed upstream, Spack does not need to build "hdf5" in order to build "netcdf", it can use the upstream install.

NERSC config has an upstream in /global/common/sw/install, so users can build on software that we (via swowner) install

Tip: one upstream is the E4S deployment at NERSC

```bash
16:13 sleak@cori04:/~$ spack config blame upstreams
---
/global/common/sw/spack/0.16.1/etc/spack/upstreams.yaml
16:13 sleak@cori04:/~$ ls -lta /global/common/sw/install
total 24
drwxr-xr-x+ 3 swowner swowner 512 May 11 21:54 .spack-db
drwxr-xr-x+ 9 swowner swowner 512 May 11 21:54 linux-opensuse_leap15-skylake_avx512
drwxr-xr-x+ 8 swowner swowner 512 May 11 21:49 .
drwxr-xr-x+ 78 swowner swowner 4096 May 7 23:45 cray-cn17-haswell
drwxr-xr-x+ 29 swowner swowner 2048 May 6 17:11 ...
drwxr-xr-x+ 34 swowner swowner 2048 Apr 12 17:12 shasta-sles15-zen2
drwxr-xr-x+ 22 swowner swowner 2048 Apr 8 13:35 cray-cn17-x86_64
drwxr-xr-x+ 2 swowner swowner 512 Apr 6 12:06 bin
```
Spack words, and what they mean

**Buildcache**: An installed software package, tarred up and GPG-signed, allowing someone else to install it without redoing the ./configure and build steps

E4S makes builds available via a buildcache, and NERSC has one set up on CFS (where it can be served via https)

16:28 sleak@cori04:~/$ spack config blame mirrors
---
mIRRORS:
/nongnu: file:///global/cifs/cdirs/nstaff/www/spack/cache
/0.16.1/etc/spack/mirrors.yaml:
/global/common/sw/spack/0.16.1/etc/spack/mirrors.yaml:
/global/common/sw/spack/0.16.1/etc/spack/mirrors.yaml:
/global/common/sw/spack/0.16.1/etc/spack/mirrors.yaml:

16:29 sleak@cori04:~/$ ls -lta /global/cifs/cdirs/nstaff/www/spack/cache
 total 4
 drwxrwsr-x+ 5 sleak nstaff 4096 May 11 21:38 build_cache
 drwxrwsr-x+ 4 swowner nstaff 4096 Apr  6 11:48 .
 drwxr-xr-x+ 2 sleak nstaff 4096 Apr  5 14:24 gpgkeys
 drwxr-xr-x  3 swowner nstaff 4096 Nov 23 12:42 ..
Spack words, and what they mean

Environment: A declarative description of an "environment" (ie collection of software specs alongside build/install preferences) that Spack should make available. (So, more like a purchase order than a conda environment)

In practice: a directory with a `spack.yaml` file in it, in which you can declare local Spack preferences and desired installs
Summary of Spack jargon

- **Package** - a unit of software that Spack can build and install
- **Repo** - a collection of packages
- **Spec** - a description of the parameters used when building a package
- **Variant** - a build option you can select for a package
- **Hash** - a unique identifier for an install, calculated deterministically from the concrete spec
- **Install_tree** - the place Spack installs software to
- **Upstream** - an install_tree Spack can use builds from
- **Buildcache** - a pre-built package for faster/easier installation
- **Environment** - a description of zero or more things you would like Spack to build, along with adjustments to Spack's config for them
Summary of Spack jargon

- **Package** - a unit of software that Spack can build and install
- **Repo** - a collection of packages
- **Spec** - a description of the parameters used when building a package
- **Variant** - a build option you can select for a package
- **Hash** - a unique identifier for an install, calculated deterministically from the concrete spec
- **Install_tree** - the place Spack installs software to
- **Upstream** - an install_tree Spack can use builds from
- **Buildcache** - a pre-built package for faster/easier installation
- **Environment** - a description of zero or more things you would like Spack to build, along with adjustments to Spack's config for them
Essential commands

- `spack find` lists packages installed in the install_tree and upstreams.
- `spack list` finds packages (installed or not) in repos - i.e. packages that Spack can build.

1. `module load spack/0.16.1`

2. Check that Spack has a package for the software you want to install, and read about the versions and variants available. `spack list <word1> <word2>` searches where `<word1>` or `<word2>` appear in the name, for example:

```
cori$ spack list gnu octave
===> 12 packages.
dejagnu  gnuradio  octave-optim  perl-term-readline-gnu
gnupg  gnutls  octave-splines  py-gnuplot
gnuplot  octave  octave-struct  ruby-gnuplot
```
Essential commands

• `spack info` shows what Spack knows about a package

In this case, `octave` looks like what we want, and we can find out about it with

`spack info octave`
Essential commands

- **spack spec** computes and shows a concrete spec (given a partial one)

3. Check the list of dependencies Spack will install:

    coris$ spack spec -Il octave

    ...
    - xsuyhgx octave@5.2.0%intel@19.1.2.254~arpack~curl~fftw~fltk~fontc...
    - t7ue4px ^cray-libsci@19.06.1%intel@19.1.2.254~mpi~openmp+shared...
    [+ ] rbctzhm ^pcre@8.44%intel@19.1.2.254~jit+multibyte+shared+utf ...
    [ ^ ] xh3lndn ^pkg-config@0.29.2%intel@19.1.2.254+internal_glib arc...
    [+ ] kvywwaf ^readline@8.0%intel@19.1.2.254 arch=cray-cnl7-haswell
    [ ^ ] 3huqp2j ^ncurses@6.2%intel@19.1.2.254+shared~symlinks+ter...

Things Spack sees already installed in your **installation tree** will have a [+] in the first column, and things Spack found installed **upstream** will have [ ^ ]. A -- means Spack did not find it, although for some packages (such as cray-libsci) Spack will be able to use the external (ie, not installed via Spack) instance.
Essential commands

- `spack install -v <spec>` performs the necessary download, configure, build and install steps to install the package and any needed dependencies.

4. When you're satisfied with what Spack plans to do, install it:

```
spack install octave@gcc
```
Summary

- When it goes smoothly, Spack saves much time and effort
- (When it doesn't, it can be hard to fix, often best to find an alternative)
- NERSC setup defaults to installing software in your $HOME/sw
  - And can use upstream installs such as E4S

- We ran through some of the jargon you'll encounter related to Spack
- And a basic recipe for using it to install software

One final tip: The Spack Slack ([http://spackpm.slack.com/](http://spackpm.slack.com/)) is a really helpful forum!
Q&A
Coming up

December: AY Transition - what to expect

January: (tentatively) A presentation of some of the work of one of our regular participants

Also coming soon:
  NERSC docs
  Queue wait time findings (from one of our regular participants)

We'd love to hear some lightning talks from NERSC users about the research you use NERSC for!
Last month's numbers - Sept/Oct

Cori: 6 outages (4 unscheduled totaling 6 hrs 0 min)
Last month's numbers - Sept/Oct

Cori daily availability:

Cori daily utilization:

New Tickets: 570 (Sept), 724 (Oct)
Closed Tickets: 490 (Sept), 724 (Oct)
Backlog at 1 Nov: 642
Thank You