EasyBuild @ HPC2N

A clean room installation environment

Our hardware

HPC2N has two separate clusters with different hardware architectures

- Abisko: AMD Bulldozer/Piledriver cores (quad-socket, 2 numa-islands/socket, 6-core/numa, 128GB/512GB)
- Kebnekaise: Intel based with some GPU nodes
    - 432 dual socket 14-core Broadwell (128GB)
    - 52 dual socket 14-core Skylake (192GB)
    - 20 quad socket 18-core Broadwell (3TB)
    - 36 KNL 68-core (192GB)
    - 36 2x14-core Broadwell with dual K80 cards
    - 10 2x14-core Skylake with dual V100 cards

So EasyBuild to the rescue

EasyBuild saves time (four different CPU archs + GPUs on two of them)

We decided on HMNS for two reasons.

- Reduce the number of modules directly visible
- Make sure users can’t mix and match incompatible modules.

Architecture design

- One directory per hardware architecture.
- Only the “current” arch visible to users.
- Fully utilize EasyBuilds optarch when building.
- Takes a bit more disk space, but disk is cheap…

Original directory layout

The directory structure was:

- base of actual installation, **/pfs/software/eb/<basearch_osdistro_hwspec>** (amd64_ubuntu1604_{bdw,skx,bdz,knl})
- a common dir, **/pfs/software/eb/common**, containing site-local easyconfigs/blocks/hooks and top level easybuild configuration file
- EasyBuild top level prefix is **/hpc2n/eb**, which is a node-local softlink to the correct <basearch_osdistro_hwspec>

Original installation environment

- Installation on Lustre (/pfs/software).
- Done by staff with their own account.
- Our Lustre targets high data bandwidth, not IOPS.
- Initially not a big problem.

Build environment

- Building on the arch specific login nodes.
- Lots of dev packages from OS distro installed.
- Hard to make clean builds.
- Cluster installation may change installed OS packages.

Solution part 1

- Compute Canada uses CVMFS for EB software
  - We already use it on batch nodes due to WLCG
  - If they can do it, so can we...
- Move the user “visible” tree from /pfs/software/eb/<basearch_osdistro_hwspec> to **/cvmfs/ebsw.hpc2n.umu.se/<basearch_osdistro_hwspec>**
- Adjust /hpc2n/eb softlink to match

But now the software tree is read-only…

And we do not have enough nodes to dedicate one of each architecture to a build node.

Solution part 2

Singularity???

Yes!!!

Solution part 2 cont.

- Make a container with a minimal set of OS packages.
  - Including IB dev packages.
- Bind mount the NFS exported master tree over the CVMFS mount point.
- Bind mount /hpc2n so the running container points to the correct architecture tree.
- Bind mount nvidia and cuda driver libs on GPU enabled nodes.

Presto, a working container that always installs into the correct architecture tree.

Making it all work

- Use a dedicated user (easybuild) for the actual building.
  - Nobody else have write access to NFS master copy.
- Use a module, eb_builder, that wraps “eb” to run “eb” inside the container.

How does it all work

- The containers “run” script runs an outside script, ~easybuild/bin/eb-build-and-update-spider-cache, which
    - initializes and purges the module environment
    - loads the same version of EasyBuild the container was called with
    - calls the actual eb with whatever arguments the container was called with
    - updates the Lmod spider cache, which is architecture specific, if the build was successful

Finished?

Publishing the builds into CVMFS is still done manually.

We try to make sure we have built for both the broadwell and skylake archs before publishing since jobs can be split over both.

This setup makes building a rather slow and serial process.

Batch building 1

- Write a submit file that loads eb_builder and runs “eb”.
- Submit to required target architecture.

Batch building 2

So how about utilizing EasyBuilds --job functionality to submit batch jobs to handle dependencies in parallel?

EB’s generated batch jobs just calls “eb” again with the specific build target.

Since the eb_builder environment is still active, the submit files will call our wrapper for the container… and it all works out just fine.

This works when running eb from a node with the architecture we want to build for.

Batch building in a clean environment

To make EasyBuild run batch job parallel builds on a part of the cluster without login nodes, all we have to do is submit a batch job that calls eb --job --robot and it will:

- happily resolve dependencies, while looking at the NFS master copy,
- submit build jobs that runs the wrapped eb that starts the container.

Hepp!

Fully batch job compatible container based clean room building.

https://github.com/hpc2n/EasyBuilding_in_Singularity