INTRODUCTION TO

EASYBUILD @ BI

PROBLEMS WITH SOFTWARE

- Installing and maintaining software from source requires a lot of time and effort
- Binary package systems focus on a single version compiled against a single set of compilers and libraries
- Installing new versions of software benefits very little from previous efforts
- We needed a way to automate installation of software for both system administrators and users

BRIEF HISTORY

- Started in 2009, EasyBuild reached version 1.0, stable API and GPL license in 2012 (announced at SC'12)
- Developed originally by IT department of Ghent University in Belgium, member of Flemish Supercomputer Center, with monthly releases
- Most recent version 3.3.1 provides:
  - 1,262 different software recipes (easyconfigs) with 25 compiler toolchains
  - 187 software specific and 29 generic methods (easyblocks)

WHAT IS EASYBUILD?

- Software build and installation framework, written in Python
- Supports different compilers and MPI libraries
- Very active and helpful community
- Features necessary to make it work for our environment at BI/VT were implemented by the core development team
- Software recipes are written by many people around the world

SERIOUSLY, WHAT IS IT?

TERMINOLOGY (1 / 2)

- EasyBuild framework
  - core: Python modules and packages
  - provides supporting functionality for building and installing software
- easyblock
  - a Python Module that serves as a build script, ‘plugin’ for the EasyBuild framework
  - implements a (generic) software build/install procedure

TERMINOLOGY (2 / 2)

- easyconfig file
  - (*.eb): build specification, software name/version, compiler toolchain, source URL, dependency list
- (compiler) toolchain
  - set of compilers with accompanying libraries
- extensions
  - additional libraries/packages/modules for a particular application (e.g., Python, R)

HOW DOES IT WORK?

- When EasyBuild is executed to follow an easyconfig recipe:
  - easyconfig is evaluated for dependencies, including toolchains. Those are installed first, if missing.
  - Source files are downloaded and unpacked
  - Relevant configure, build, test and install steps are taken
  - Each installation is verified, and if successful a new set of environment modules are created with full dependency information

Sample easyconfig: ABySS/ABySS-1.9.0-foss-2016a.eb

```
# easy block selection
easyblock = 'ConfigureMake'

# software name and version
name = 'ABySS'
version = '1.9.0'

# software metadata
homepage = 'http://www.bcgsc.ca/platform/bioinfo/software/'
description = """Assembly By Short Sequences - a de novo"""

# toolchain name and version
toolchain = {'name': 'foss', 'version': '2016a'}
toolchainopts = {'usempi': True}

# sources and patches
sources = [SOURCELOWER_TAR_GZ]
source_urls = ['https://github.com/abyss/releases/%(version)s/']

# dependencies
dependencies = [
    ('Boost', '1.60.0'),
    ('sparsehash', '2.0.2'),
    ('SQLite', '3.9.2'),
]

# sanity checks
sanity_check_paths = {
    'files': ["bin/ABYSS", "bin/ABYSS-P"],
    'dirs': []
}

moduleclass = 'bio'
```

WHAT'S DIFFERENT AT BI?

- Environment modules on each cluster pre-set variables for:
  - HPC administrators, managing globally installed software
  - Users, managing their local installations
- Users' environment allows access to ALL global software, and build locally any additional/missing items
  - Users can create their own easyconfigs, or override global ones
- Directory structure takes cluster name and architecture into account
  - a single home directory or /apps dir can be shared between many clusters

HOW DO WE USE EASYBUILD?

Accessing already available software

```
$ ssh discovery1.bi.vt.edu

$ module list
Currently Loaded Modulefiles:
  1) gcc/6.1.0 2) slurm/16.05.8
  3) site/discovery/easybuild/setup

$ module load Valgrind/3.11.0-foss-2016a

$ module list
Currently Loaded Modulefiles:
  1) gcc/6.1.0
  2) slurm/16.05.8
  3) site/discovery/easybuild/setup
  4) GCCcore/4.9.3
  5) binutils/2.25-GCCcore-4.9.3
  6) GCC/4.9.3-2.25
  7) numactl/2.0.11-GCC-4.9.3-2.25
  8) hwloc/1.11.2-GCC-4.9.3-2.25
  9) OpenMPI/1.10.2-GCC-4.9.3-2.25
 10) OpenBLAS/0.2.15-GCC-4.9.3-2.25-LAPACK-3.6.0
 11) gompi/2016a
 12) FFTW/3.3.4-gompi-2016a
 13) ScaLAPACK/2.0.2-gompi-2016a-OpenBLAS-0.2.15-LAPACK-3.6.0
 14) foss/2016a
 15) Valgrind/3.11.0-foss-2016a

$ module load <tab><tab>
Display all 531 possibilities? (y or n)
```

Building new software (1)

```
# do we have 'pbzip2' installed?
dom@discovery1 ~> module av 2>&1 | grep -i pbzip
dom@discovery1 ~>

# let's load EasyBuild
dom@discovery1 ~> module load EasyBuild

# time to search for khmer
dom@discovery1 ~> eb -S pbzip2
CFGS1=/apps/easybuild/ebfiles_repo-vbi/generic/p/PBZIP2
CFGS2=/apps/easybuild/ebfiles_repo-vbi/generic/p/PBZIP2
CFGS3=/apps/easybuild/software/discovery-sandy_bridge/EasyBuild/3.3.1/
lib/python2.7/site-packages/easybuild_easyconfigs-3.3.1-py2.7.egg/
easybuild/easyconfigs/p/PBZIP2
 * $CFGS1/PBZIP2-1.1.12-foss-2016a.eb
 * $CFGS1/PBZIP2-1.1.8-foss-2016a.eb
 * $CFGS2/PBZIP2-1.1.12-foss-2016a.eb
 * $CFGS2/PBZIP2-1.1.8-foss-2016a.eb
 * $CFGS3/PBZIP2-1.1.8-goolf-1.4.10.eb
 * $CFGS3/PBZIP2-1.1.8-ictce-6.2.5.eb

# let's see what's required to build it
dom@discovery1 ~> eb PBZIP2-1.1.12-foss-2016a.eb -Dr
Dry run: printing build status of easyconfigs and dependencies
CFGS=/apps/easybuild/ebfiles_repo
 * [x] $CFGS/discovery-sandy_bridge/M4/M4-1.4.17.eb (module: M4/1.4.17)
[…]
 * [ ] $CFGS/vbi/generic/p/PBZIP2/PBZIP2-1.1.12-foss-2016a.eb (module:
PBZIP2/1.1.8-foss-2016a)
```

Building new software (2)

```
dom@discovery1 ~> eb PBZIP2-1.1.12-foss-2016a.eb —robot

== temporary log file in case of crash /tmp/eb-PSMVAS/easybuild-OiCo6M.log
== resolving dependencies ...
== processing EasyBuild easyconfig /apps/easybuild/ebfiles_repo-vbi/
generic/p/PBZIP2/PBZIP2-1.1.12-foss-2016a.eb
== building and installing PBZIP2/1.1.12-foss-2016a...
== fetching files...
== creating build dir, resetting environment...
== unpacking...
== patching...
== preparing...
== configuring...
== building...
== testing...
== installing...
== taking care of extensions...
== postprocessing...
== sanity checking...
== cleaning up...
== creating module...
== permissions...
== packaging...
== COMPLETED: Installation ended successfully
== Results of the build can be found in the log file(s) /home/dom/
easybuild/software/discovery-sandy_bridge/PBZIP2/1.1.12-foss-2016a/
easybuild/easybui
57.log
== Build succeeded for 1 out of 1
== Temporary log file(s) /tmp/eb-PSMVAS/easybuild-OiCo6M.log* have been
removed.
== Temporary directory /tmp/eb-PSMVAS has been removed.
```

Building new software (3)

```
dom@discovery1 ~> module av 2>&1 | grep -i pbzip2
PBZIP2/1.1.12-foss-2016a

dom@discovery1 ~> module load PBZIP2/1.1.12-foss-2016a

dom@discovery1 ~> module list
Currently Loaded Modulefiles:
  1) gcc/6.1.0
[…]
 17) PBZIP2/1.1.8-foss-2016a

dom@discovery1 ~> which pbzip2
~/easybuild/software/discovery-sandy_bridge/PBZIP2/1.1.12-
foss-2016a/bin/pbzip2
```

```
dom@discovery1 ~> module av 2>&1 | less

[…]
------------ /home/dom/easybuild/modules/discovery-
sandy_bridge/all ------------
PBZIP2/1.1.12-foss-2016a

--------------- /apps/easybuild/modules/discovery-
sandy_bridge/all ---------------
ABySS/1.9.0-foss-2016a
[…]
```

WHAT COMES OUT?

- ~/easybuild/software/<cluster>-<arch>/<app_name>/<app_version>/
  - full application
  - full build and install log
  - markdown report of the configure options, build, environment and install
- ~/easybuild/sources/<app_name_first_letter>/<app_name>
  - source/installer
- ~/easybuild/modules/<cluster>-<arch>/all/<app_name>/<app_version>
  - environment module for the application
- ~/easybuild/ebfiles_repo/<cluster>-<arch>/<app_name>/<app_name>-<app_version>-<toolchain_name-version>.eb
  - new easyconfig, includes build stats

IS THAT ALL?

Building new software - reusing existing easyconfigs

```
# installing different version from the same easyconfig

dom@discovery1 ~> eb PBZIP2-1.1.12-foss-2016a.eb —try-software-version=1.1.13

== temporary log file in case of crash /tmp/eb-wusUzW/easybuild-Eo5LXC.log
== resolving dependencies ...
== processing EasyBuild easyconfig /tmp/eb-wusUzW/tweaked_easyconfigs/
PBZIP2-1.1.13-foss-2016a.eb
== building and installing PBZIP2/1.1.13-foss-2016a...
== fetching files...
== creating build dir, resetting environment...
== unpacking...
== patching...
== preparing...
== configuring...
== building...
== testing...
== installing...
== taking care of extensions...
== postprocessing...
== sanity checking...
== cleaning up...
== creating module...
== permissions...
== packaging...
== COMPLETED: Installation ended successfully
== Results of the build can be found in the log file(s) /home/dom/easybuild/
software/discovery-sandy_bridge/PBZIP2/1.1.13-foss-2016a/easybuild/easybuild-
PBZIP2-1.1.13-20170910.2T3532.log
== Build succeeded for 1 out of 1
== Temporary log file(s) /tmp/eb-wusUzW/easybuild-Eo5LXC.log* have been removed.
== Temporary directory /tmp/eb-wusUzW has been removed.
```

IMPORTANT BITS

HOW TO GET NEW SOFTWARE

- First, check if it's already installed
  - module av 2>&1 | grep -i <app>
- Then, see if easyconfig is available
  - module load EasyBuild
  - eb -S <app>
  - eb <easyconfig.eb> —robot
- If you need a newer/different version, try:
  - eb <easyconfig.eb> —robot —try-software-version=<x.y>

IMPORTANT BITS

- Most easyconfigs target either FOSS or INTEL toolchains, we have standardized on FOSS:
  - foss/2016a
  - foss/2016b
  - foss/2017a
- foss toolchain comprises of:
  - compiler - GCC
  - MPI implementation - OpenMPI
  - libraries - OpenBLAS/LAPACK, ScaLAPACK(/BLACS), FFTW

CREATING NEW EASYCONFIGS

- Find an existing similar easyconfig from foss/2016a or foss/2016b toolchain
- Share the easyconfig between the group, or send to helpdesk@rt.bi.vt.edu and I'll include it in the global installs
- Examples, examples, examples! I constantly browse existing easyconfigs to get ideas
- Be sure to use proper naming scheme
  - pigz-2.3.3-foss-2016b.eb
- Place them in either location:
  - ~/easybuild/ebfiles_repo/generic/
  - ~/easybuild/ebfiles_repo/<cluster>-<arch>/

NOTES

- Always start with a clean module environment, do not preload modules in your ~/.bashrc or ~/.bash_profile
- In your slurm/PBS scripts load module only for the application you are trying to run. EasyBuild generated modules have all the dependency requirements needed
- When re-using easyconfigs from different toolchains, match dependency software versions to the new toolchain
  - module av 2>&1 | grep foss-2016b

GETTING HELP

- EasyBuild documentation is excellent
  - https://easybuild.readthedocs.io/
  - eb –help
- EasyBuild developers and users are available via:
  - mailing list
  - IRC channel
  - Slack channel
- Contact us via the RT system: helpdesk@rt.bi.vt.edu

BI EASYCONFIGS AND ENVIRONMENT SCRIPTS

- Complete and up-to-date list of all easyconfigs and modules installed on BI clusters
  - https://devlab.vbi.vt.edu/HPC/easybuild
  - https://github.com/dominikborkowski/biocomplexity_easybuild
- includes entry level documentation and design

DEMO TIME!