

The Challenges of Installing Software on HPC Systems

Santiago Lacalle Puig

slacalle@imperial.ac.uk Github: @slacalle

EUM23 - 8th EasyBuild User Meeting

Overview

- Challenges of installing scientific software.
- Compiling software.
- Other package manager: conda
- System package managers (yum, apt).
- Containers: Singularity.
- Challenges not related to software itself.
- High performance software.
- EasyBuild.



Challenges of installing scientific software

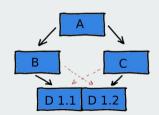
- Non-standard installations. Very time consuming.
- Support Software that is no longer actively developed.
- Lack of documentation and poor software engineering practices from software developers. e.g.
 - Enhancing code readability
 - Keeping code efficient
 - Version control
 - Being descriptive
 - Applying KISS Keep it Simple, Stupid.
- Dependencies

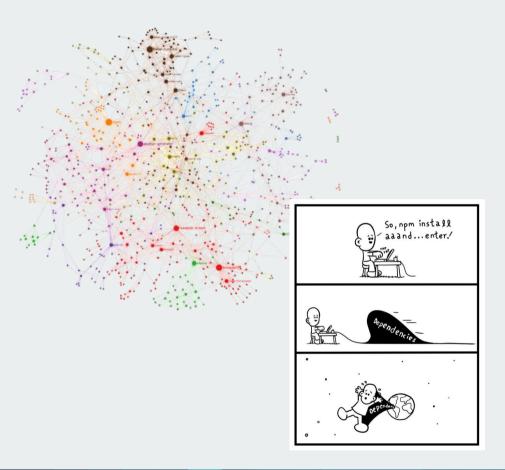


Dependencies Hell...

Long chains of dependencies

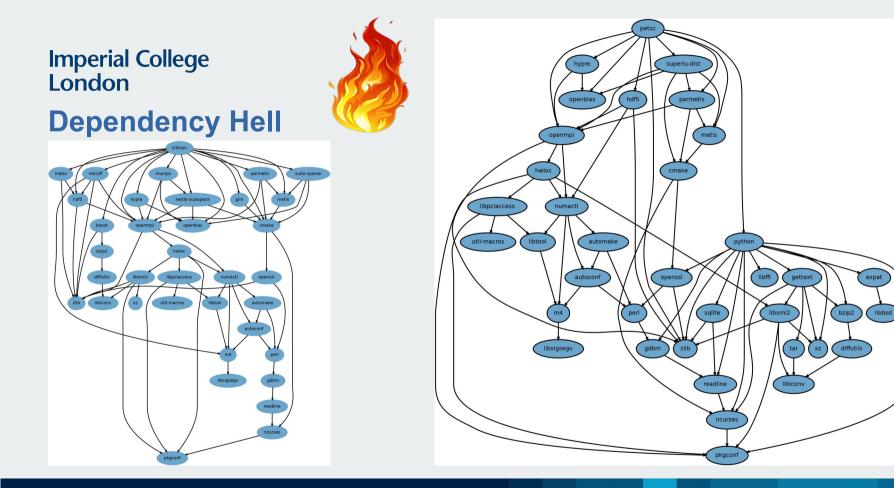
- Conflicting dependencies
- Circular dependencies
- Package manager dependencies
- Diamond dependency





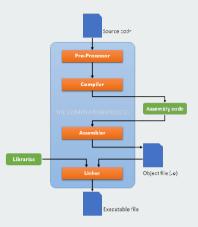






Compiling Software

- Environment modules.
- Environment variables: PATH, LDFLAGS, CPATH, CPPFLAGS, LD_LIBRARY_PATH, RPATH.
- MPI, MKL, BLAS, LAPACK
- Autotools, Make, Cmake



Example: quantum espresso

- \$./install/configure -prefix=/apps/espresso/6.3-new BLAS_LIBS=-L/apps/intel/2017.6/compilers_and_libraries_2017.6.256/linux/mkl/lib/intel64/
- -Imkl_intel_Ip64 -Imkl_sequential -Imkl_core LAPACK_LIBS=-L/apps/intel/2017.6/compilers_and_libraries_2017.6.256/linux/mkl/lib/intel64/
- -Imkl_intel_lp64 -Imkl_sequential -Imkl_core CPPFLAGS=-I/apps/mpi/intel/2018.1.163/include \
- -l/apps/intel/2017.6/compilers_and_libraries_2017.6.256/linux/mkl/include/fftw \
- -l/apps/hdf5/1.8.15/parallel/include LDFLAGS=-L/apps/mpi/intel/2018.1.163/lib \
- -L/apps/hdf5/1.8.15/parallel/lib -L/apps/intel/2017.6/compilers_and_libraries_2017.6.256/linux/mkl/lib/intel64 \
- -L/apps/intel/2017.6/clck/2017.2.019/lib/intel64 --with-hdf5=/apps/hdf5/1.8.15/parallel



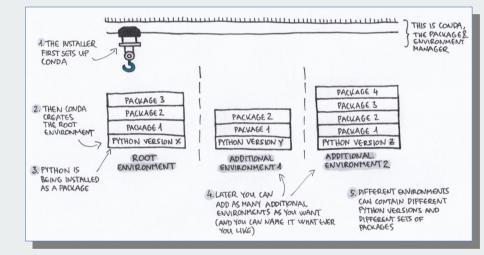


EASVBUUD

Other package managers - Conda

Conda

- Over 7500 packages.
- Aimed at end users/researchers.
- Empower users!
- Reduces support requests.
- Pre-built binaries.
- Relatively quick.
- Plays well with other services (e.g. Jupyter)
- Wide adoption.



and the second second

EasyBuild

Imperial College London

Caveats and considerations of using Conda

- Cannot be used for everything. Mainly python and R.
- Environments can become delicate.
- Increased user support requests to fix environments.
- Requires some playing around to learn a number of quirks.
- Installations for non-conda packages can become tedious and more complicated than using alternatives.
- Doesn't play nice at times with pip or install.packages()

\$ conda create -n test_renv r-base \ r-data.table r-plyr r-ggplot2 r-seurat r-biocmanager \ r-doparallel -c conda-forge -c bioconda

r-base + 6 packages > 247 Packages!



Other Package managers – yum, apt

- Quick solution
- Can cause problems in the future
- Grows the system image
- And most importantly.... Not optimized!

libboost date time.so.1.66.0	libfftw3.so.3	liboxps.so.2.2.2	libnautilus-extension.so.1.5.0	liboo.so.5
libboost fiber.so	libfftw3.so.3.5.5	libhSbzip2.so	Libncurses++.so	libpo, so, 5, 13
libboost fiber.so.1.66.8	libfftw3 threads.a	libbanlib.so.4	libncurses.so	libprocps.so.7
libboost filesystem.so	libfftw3 threads.so	libhamlib.so.4.0.0	Libocurses++.so.5	libprocps.so.7.1.0
libboost filesystem.so.1.66.0	libfftw3 threads.so.3	libhandle.so.1	Libocurses.so.5	libprotobuf-c.so.1
libboost graph.so	libfftw3 threads.so.3.5.5	libhandle.so.1.0.3	Libncurses++.so.5.9	libprotobuf-c.so.1.0.0
libboost graph.so.1.66.0	Libfftw.a	libharfbuzz-icu.so		
libboost iostreans.so	libfftw.so.2	libharfbuzz-icu.so.0	Libncurses++.so.6	libproxy.so.1
libboost iostreams.so.1.66.0		libharfbuzz-icu.so.0.10705.0	Libncurses.so.6	libproxy, so, 1,0,0
libboost locale.so	libfftw threads.a	libharfbuzz.so	libncurses++.so.6.1	libpsl.so.5
libboost locale.so.1.66.0	libfftw threads.so.2	libharfbuzz.so.0	Libncurses.so.6.1	libosl.so.5.3.1
libboost log setup.so	libfftw threads.so.2.0.7	libharfbuzz.so.0.10705.0	libncurses++w.so	libpsm2.so.2
libboost log setup.so.1.66.0	libfipscheck.so.1	libhdf5 cpp.so	libncursesw.so	libpsm2.so.2.2
libboost log.so	libfipscheck.so.1.2.1	libhdf5 cpp.so.103	liborurses++w.so.5	libpspell.so.15
libboost log.so.1.66.0	libFLAC++.so.6	libhdf5 cpp.so.103.1.0	Libocursesw.so.5	libpspell.so,15,1,5
libboost math c99f.so		libhdf5 fortran.so		libpsx.so.2
libboost math c99f.so.1.66.0	LibFLAC.so.8	libhdf5 fortran.so.102		
libboost math c991.so		libhdf5 fortran.so.102.0.0	libncurses++w.so.6	libptexenc.so.1
libboost math c991.so.1.66.0	libflatpak.so.0	libhdf5 hl cpp.so	libncursesw.so.6	
libboost math c99.so		libhdf5 hl cpp.so.100		
libboost math c99.so.1.66.0	libfltk forms.so.1.3	libhdf5 hl cpp.so.100.1.3		libpthread_nonshared.a
libboost math trif.so	libfltk gl, so, 1, 3	libhdfS hl fortran.so	libndr-krbSpac.so.0	libpthread, so
libboost math trif.so.1.66.0	libfltk images.so.1.3	libhdfShl fortran.so		libpthread.so.0
libboost math tril.so		libhdf5hl fortran.so.100	libodr-obt.so.0	libpulse-mainloop-glib.so.0
libboost math tril.so.1.66.0	libfontconfig.so			
libboost_math_tri.so	libfontconfig.so.1	libhdf5 hl.so	libndr.so.2	libpulse-simple.so.0
libboost math tr1.so.1.66.0		libhdf5 hl.so.100		
libboost pro exec monitor.so	libfontembed.so.1		libndr-standard.so.0	libpulse.so.0
libboost pro exec monitor.so.1.66.0		libhdf5.settings		
libboost program options.so	libfontenc.so.1	libhdf5.so	libneon.so.27	libpwquality.so.1
libboost program options.so.1.66.0		libhdf5.so.103		
libboost_random.so			libnetapi.so.1	
libboost_random.so.1.66.0		libhgfs.so.0		libpython3.6m.so
libboost_regex.so			libnetcdf.a	
libboost regex.so.1.66.0	libforn.so.6		libnetcdf c++.a	

75 install postgresql	2019-10-29 11:51 Install	1 EE
74 install p7zip	2019-10-29 11:50 Install	1 EE
73 install opensm	2019-10-29 11:50 Install	1 EE
72 install mesa-*	2019-10-29 11:50 Install	17 EE
71 install lz4	2019-10-29 11:50 Install	1 EE
70 install libunwind	2019-10-29 11:50 Install	1 1 EE
69 install libtar	2019-10-29 11:50 Install	1 EE
68 install liblzf*	2019-10-29 11:49 Install	2 EE
67 install libglade2	2019-10-29 11:49 Install	1 EE
66 install libcgroup	2019-10-29 11:49 Install	1 EE
65 install gtk*-devel	2019-10-29 11:48 Install	12 EE
64 install gstreamer*	2019-10-29 11:48 Install	13 EE
63 install gnutls-*	2019-10-29 11:48 Install	9 EE
62 install gl2ps	2019-10-29 11:48 Install	1 EE
61 install ghostscript*	2019-10-29 11:48 Install	1 EE
60 install freeglut	2019-10-29 11:47 Install	1 EE
59 install fltk*	2019-10-29 11:47 Install	1 EE
58 install expect	2019-10-29 11:47 Install	1 EE
57 install compat-*	2019-10-29 11:47 Install	23 EE
56 install cmake-*	2019-10-29 11:44 Install	2 EE
55 install cfitsio	2019-10-29 11:44 Install	1 EE
54 install cairo-devel	2019-10-29 11:44 Install	2 EE
53 install autogen*	2019-10-29 11:43 Install	1 EE
52 install apache-commons-*	2019-10-29 11:43 Install	5 EE
51 install apr-*	2019-10-29 11:43 Install	14 EE
50 install compat-openssl*	2019-10-21 17:44 Install	1 EE
49 update	2019-10-21 17:28 Upgrade	1 E<
48 install openIdap*	2019-10-16 22:48 Install	4 >E
47 erase teamd libteam network-scripts-team	2019-10-16 22:38 Removed	5 EE
46 install network-scripts	2019-10-16 22:36 Install	2 EE
45 erase at	2019-10-16 22:20 Removed	1 EE
44 erase cups	2019-10-16 22:01 Removed	8 EE
43 install nfs-utils	2019-10-16 21:59 Install	5 EE





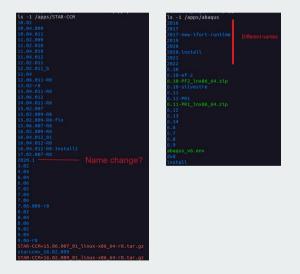
Containers - Singularity

- Containers provide a practical solution for replicating and circulating a pre-existing software stack.
- Some containers are not as easy to create. They can have have numerous dependencies.
- Leveraging the os package manager within a container is not enough.
 - Generic binaries not tailored to specific architectures. Not optimized!
- Containers may require ongoing support or rebuilding to account for updates/changes or accommodate multiple architectures.



Challenges not related to software itself:

- Manage incoming requests and deliver in a timely manner.
- Provide a good level of documentation for service users.
- Efficiently maintain software stacks.
- Ensure organization and standardization across the stack (directory naming, modules).
- Ensure software is optimized for each architecture.





Are we delivering High Performance Software?

- Are we delivering the most efficient software?
- Who should care for optimized software? Package managers/sys admins or end users?
- Considerable performance gains in certain cases.
- Running more efficient code can mean:
 - Saving users time.
 - Reducing compute time.
 - Decreasing carbon footprint.
 - Money saved.









EasyBuild

Misconceptions:

- EasyBuild will make it harder to install software.
 - Adds another layer of complexity on top of the software.
 - Will slow down complex installs.

EasyBuild solves:

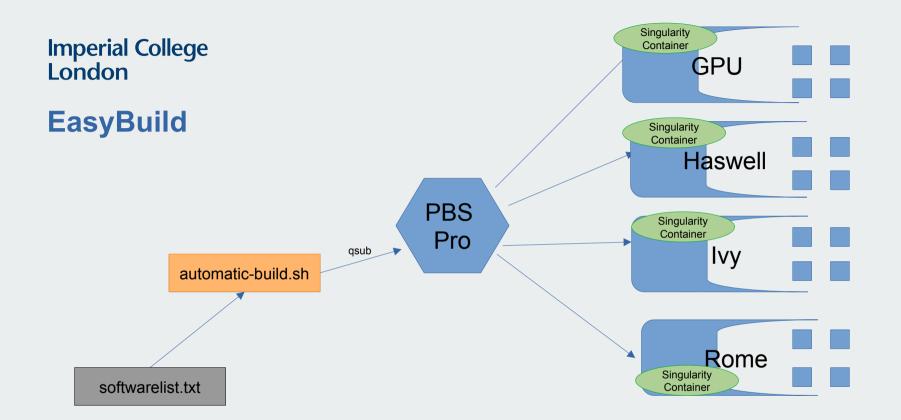
- Used to install multiple versions and manage dependencies
- Automatically installs (less pain for package managers)
- Reduces overall install time
- Ensures standardization in naming and organization in modulefiles
- Optimized software for our hardware. PERFORMANCE!









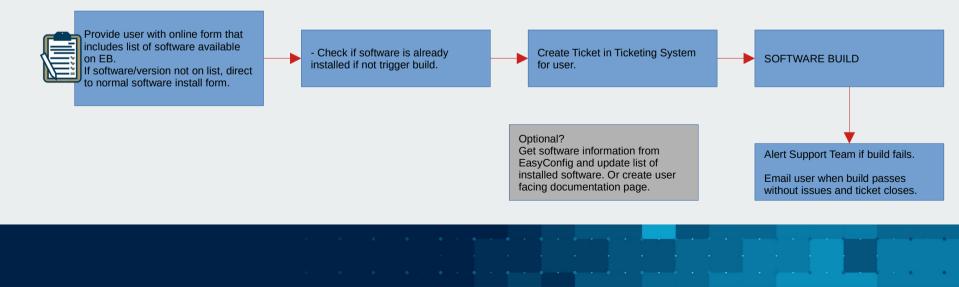






Future plans:

- Automate user software installation requests to trigger EasyBuild



EasyBuild



Old software stack (manual installs) Pre 2013

Total modules: 2776



New Software stack (Easybuild) Since March 2022

> 1585 modules in development stack 1152 modules in production stack Total modules: 2737



.



Thank you!

