HEP_OSlibs

Andrea Valassi (IT-DI-LCG)

WLCG AA Librarian and Integrators Meeting Workshop
30th May 2018

See also my talk at the July 2016 LIM and follow-up discussion at the June 2017 LIM

Further in the past: see Pere Mato’s and Oliver Keeble’s talks at the June 2009 GDB

What is HEP_OSlibs?

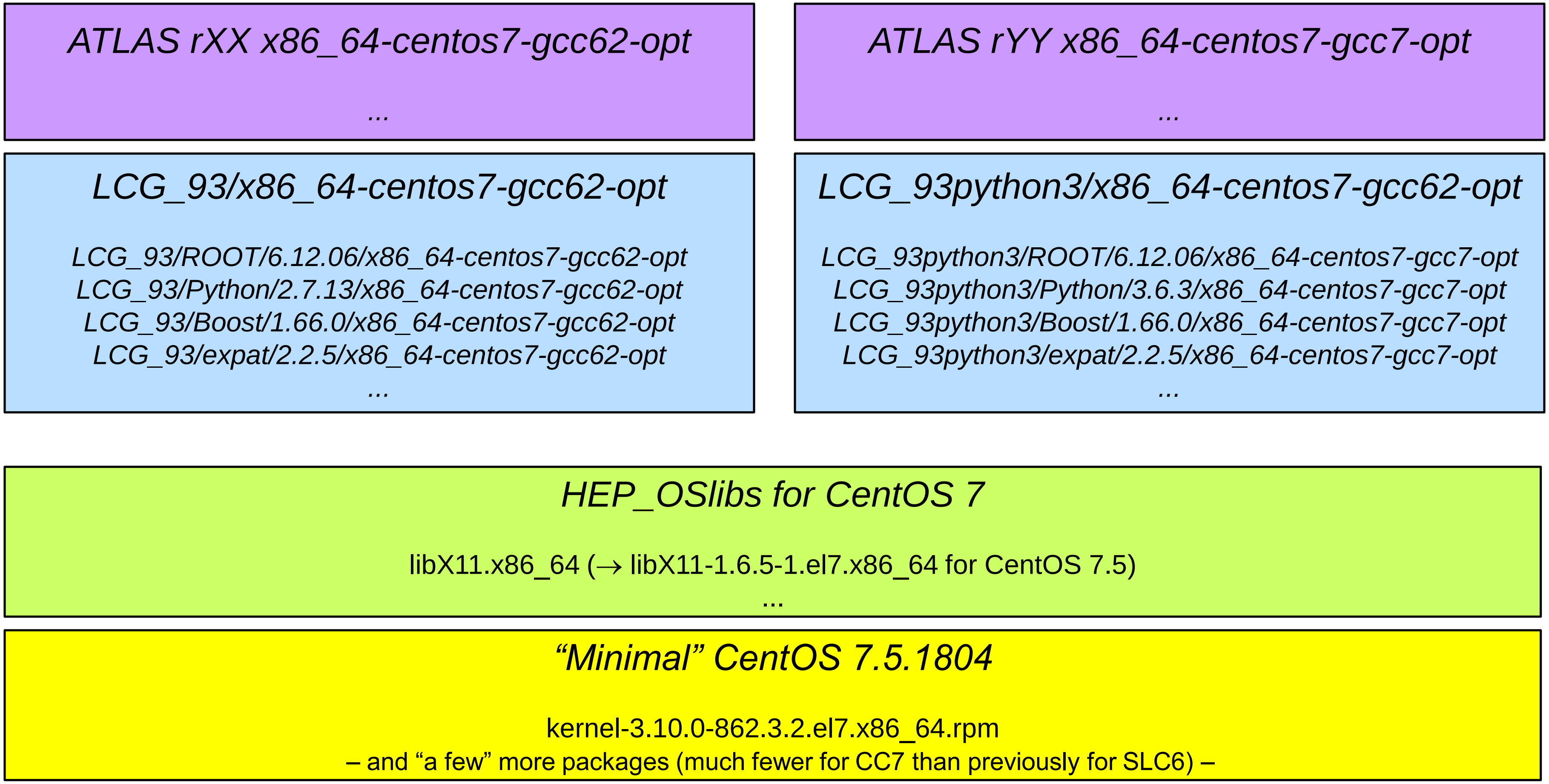
- a meta-package: *‘yum install HEP_OSlibs’* on RedHat is equivalent to
  ‘yum install alsa-lib-devel.x86_64 ... zsh.x86_64’
  - a list of packages
  - without a specific version number
  - taken from the official O/S repos (not from EPEL)
  - “the O/S level dependencies of the LHC experiment software stacks”
  - *a more precise definition of what a “supported O/S” is*

- Initially developed on RedHat (SL5, SL6, CC7)
  - now exists also on Ubuntu (16.04, 17.04, 17.10)

- Initial focus was WLCG deployment (runtime packages) – e.g. SL4 compat
  - now includes also development packages (build time packages)

- See https://gitlab.cern.ch/linuxsupport/rpms/HEP_OSlibs
  - source code repo, build and test (CI+koji), documentation (dependency lists)

Example



ATLAS rXX x86_64-centos7-gcc62-opt

...

ATLAS rYY x86_64-centos7-gcc7-opt

...

LCG_93/x86_64-centos7-gcc62-opt

LCG_93/ROOT/6.12.06/x86_64-centos7-gcc62-opt
LCG_93/Python/2.7.13/x86_64-centos7-gcc62-opt
LCG_93/Boost/1.66.0/x86_64-centos7-gcc62-opt
LCG_93/expat/2.2.5/x86_64-centos7-gcc62-opt
...

LCG_93python3/x86_64-centos7-gcc62-opt

LCG_93python3/ROOT/6.12.06/x86_64-centos7-gcc7-opt
LCG_93python3/Python/3.6.3/x86_64-centos7-gcc7-opt
LCG_93python3/Boost/1.66.0/x86_64-centos7-gcc7-opt
LCG_93python3/expat/2.2.5/x86_64-centos7-gcc7-opt
...

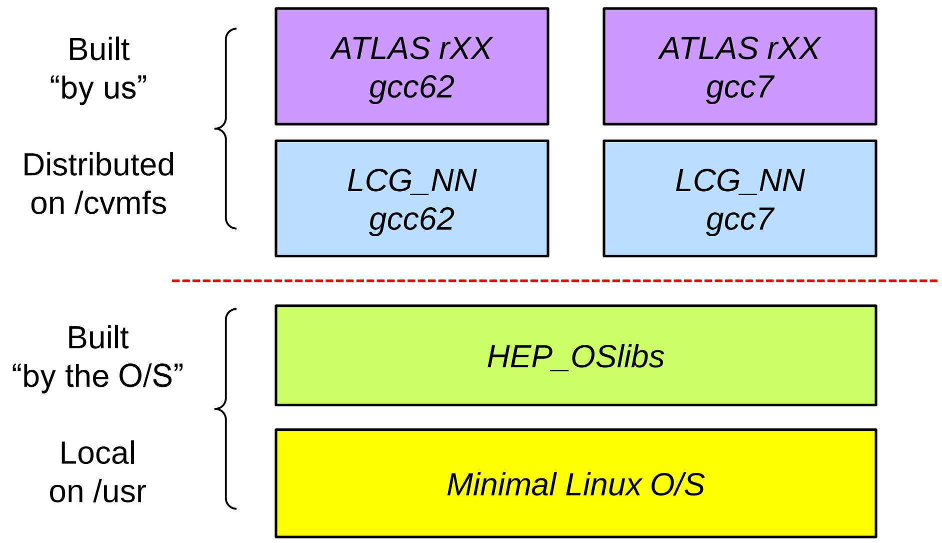
HEP_OSlibs for CentOS 7

libX11.x86_64 (→ libX11-1.6.5-1.el7.x86_64 for CentOS 7.5)
...

“Minimal” CentOS 7.5.1804

kernel-3.10.0-862.3.2.el7.x86_64.rpm
– and “a few” more packages (much fewer for CC7 than previously for SLC6) –

Why HEP_OSlibs?



When a new dependency appears:
- should it be built in the LCG stack?
- should it be taken from the O/S?

Where do you put the boundary?

IMO this question will continue to apply in the future (unless “we do not take anything from the O/S”?)

- My opinion in short:
  - **IF you can take a package from the O/S, why rebuild it yourself?**
  - *Less risk of conflicts (we did observe many conflicts in the past – expat, uuid...)*
  - Less overhead to build and distribute

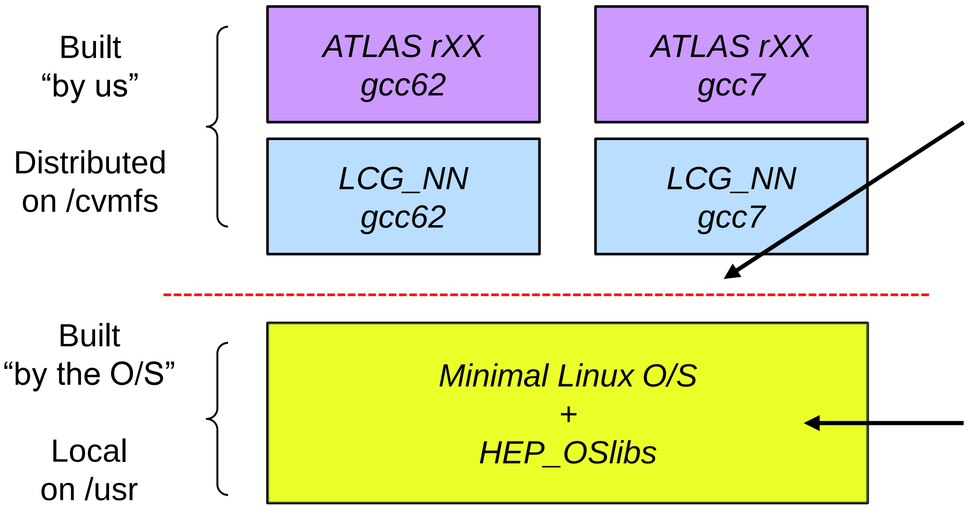
Which packages cannot be in HEP_OSlibs?

- On Linux, I see only 3 reasons to build a package ourselves (LCG stack):
  - 1. Need a different C++ version than the O/S compiler (e.g. Boost)
  - 2. Need a newer package version than the O/S version (e.g. Python)
  - 3. Need a HEP or EPEL package not available in the O/S (e.g. ROOT)
  - In all other cases, I would install missing O/S packages via HEP_OSlibs

- On MacOS, system directories (/bin, /sbin, /usr) may not be modified
  - Linux is an open system, MacOS a closed one (the “Apple walled garden”)
  - Three different policies and locations for Ports, Brew and Fink
  - No HEP_OSlibs on MacOS: all external dependencies should be rebuilt

- *NB: if a package is missing on MacOS but can be included from the O/S on Linux, IMO it should be added to the LCG stack only on MacOS*
  - *do not add unnecessary risks of conflicts on Grid deployment (100% Linux)*

Conflicts?



Built "by us"

Distributed on /cvmfs

ATLAS rXX gcc62

ATLAS rXX gcc7

LCG_NN gcc62

LCG_NN gcc7

Built "by the O/S"

Local on /usr

Minimal Linux O/S + HEP_OSlibs

Conflicts between LCG/experiment and the O/S are possible due to faulty runtime or build time configuration

... here is where development meets deployment...

No conflicts between O/S and HepOS (essentially an extension of the O/S)

- Build time configuration: deploy HEP_OSlibs on build nodes
- Runtime configuration: use dependencies from LCG/experiment stack if an O/S package has been duplicated there (e.g. LD_LIBRARY_PATH)
  - responsibility of LCG/experiment stack
  - *cannot solve these issues by uninstalling packages from the O/S or HepOS*
  - BTW: are the experiments using LCG views or cherry-picking packages?

Where is HEP_OSlibs deployed?

- Current users from LHC: ALICE, ATLAS, LHCb – is this correct?
  - CMS has moved to a different model rebuilding much more than the others
  - SWAN is effectively using HEP_OSlibs through the LCG stack

- Grid nodes (runtime for production jobs) for the above experiments
  - the reason why HEP_OSlibs was introduced in the first place
  - at CERN: lxbatch
  - *NB there are HEP_OSlibs users that do not use the LCG stack (e.g. ALICE)*

- Interactive user environments (development and runtime)
  - at CERN: lxplus, SWAN

- Build nodes (development packages)
  - in particular, the build nodes for the LCG stack and the experiments IIUC

Is HEP_OSlibs too fat?

- Most frequent user criticism on the deployment side

- By design, so far there has been only ONE HEP_OSlibs version
  - dependencies of ALL experiments
  - dependencies for both runtime and build time
  - easier management...
  - easier deployment (e.g. lxplus)... and disk space is cheap after all?

- Of course this could be improved
  - but it would take someone’s effort

Conclusions

- HEP_OSlibs is a meta-package including the runtime and build time O/S dependencies of the LCG and experiment stacks on Linux
  - a simple way to make the O/S fatter now than it's thinner by default...
  - *a more precise definition of what a "supported O/S" is*

- Has been used for 10 years
  - serving its main purpose: runtime environment for Grid jobs
  - and serving its related purpose: build time environment for software stacks

- Infrastructure is now stable and requires minimal maintenance
  - after quite a bit of work last year (move to gitlab, port to Ubuntu...)
  - adding new packages is just changing a few lines and launching a CI

- Do let me know if you plan to keep using it, or instead no longer need it...