LIM

17 March 2020

Status of LCG_97

- Since last meeting
  - Fixes in packages
    - Fixed issue with Tensorflow 1.14 and Python 3.7:
      - Managed to build our own wheel (thanks to B Hegner)
    - Fixed some consistency issues related to the LHCb layer
    - Included patch to allow NA61/SHINE to build w/ Geant 10.3 and ROOT 5
    - Fixed issues to build on Mac
  - LCG_97rc4, LCG_97rc4_FCC_1
    - Centos7, gcc{8,9}, Python{2,3}, on cvmfs and single-meta-RPMs
    - Mac 10.15, on cvmfs
  - ROOT tag v6-20-02 (available on Sunday 15/3/2020)
    - Ongoing build of LCG_97rc4_LHCB_1
  - RPMs
    - Discussions on how to better handled the changes in RPM packaging implied by the layered stacks (see next)

LCG releases and RPMs

- A LCG release is a set of packages with an hash tracking its dependencies
- The details are saved into a '*${release}_${platform}.txt*' file
  - Located under http://lcgpackages.web.cern.ch/lcgpackages/tarFiles/releases/
    - Pointing to /eos/project/l/lcg/www/lcgpackages/tarFiles/releases/
  - E.g. for 96b, centos7, gcc8, opt: LCG_96b_x86_64-centos7-gcc8-opt.txt
- Core RPMs named *${package}-${hash}* are provided
- The '.txt' files provide *package-to-hash* resolution for a given release
- Link RPMs in the form *release_package_platform* facilitate resolution using RPM technology, providing, for each release and platform
  - *package-to-hash* resolution, including dependency resolution
  - Creation of symlinks

LCG RPMs in numbers

- For each release and platform there are
  - $N_{packages}$ *package-to-hash*
  - one *global*

  additional meta RPMs
- For example, for LCG_96b
  - **509** meta-RPMs for x86-64-centos7-gcc8-opt
  - **503** meta-RPMs for x86-64-centos7-gcc9-opt
- The LCG_96 release (LCG_96, LCG_96b) includes **8384** meta-RPMs
- One package-hash RPM per meta-RPM, with overlap between releases
- For example, rpms_updates (basically last 15 months) contains **13118** package-hash RPMs

LCG meta RPMs: the problem with layers

- Each layer is a complete release, so a complete stack of link-RPMs
- Each change, e.g. a different MC version, brings a new layer
- The number of combinations, hence the number of link-RPMs, can be huge, putting at risk the stability and manageability of RPM databases

- The initial idea to face this was to have only the meta-RPM, installing all packages in the layer, and creating related symlinks
- We learned at last meeting (minutes) that this does not work for ATLAS
  - ATLAS uses a modified version of yum, ayum, including **relocation** support
  - Possibility to **install single package** using meta-RPM very important
- LHCb is not affected as they use the '.txt' files directly

RPMs and layers: possible solutions considered

- Provide a replacement of yum, e.g. lcgyum, to install/remove packages, based on the above scripts
  - Pros: minimal changes, at least in simple cases
  - Cons: complicated to get all functionality, e.g. relocation support
- Provide official tools to handle ‘.txt’ files to extract the relevant information
  - Pros: may be adapted to complex cases
  - Cons: changes may be important
- Better partition the RPM repository to reduce the size of databases, for example at level of release or even of layer
  - Pros: continue to use RPMs machinery
  - Cons: a new repo file per database partition; changes can be minimised using, for example, the YUM0...YUM9 variables to tailor repository definition (see next)

Example of YUM0...YUM9 usage

Repo file:

```
[lcgrepo]
name=LCG Releases
baseurl=https://lcgpackages.web.cern.ch/lcgpackages/lcgrepo/$YUM1/$YUM2
gpgcheck=0
enabled=1
protect=0
```

Usage:

```
$ YUM1=7 YUM2=96 yum install -y LCG_96b_x86_64_centos7_gcc8_opt
```