Installing LOFAR Software on a CentOS 6.5 (Final) system (cluster Newton of the AIP Potsdam)

— *Frank Breitling* *2015/06/26 18:47*

I was mainly following the previous installation notes provided by others:

http://www.lofar.org/operations/doku.php?id=public:user_software:lofar
http://www.lofar.org/operations/doku.php?id=public:user_software:ubuntu_12_4

However a few problems occured and had to be solved as described below.
Ideas and support by **Arno Schoenmakers**, **Ger van Diepen** and **Marcel Loose** were very helpful and highly appreciated.

1. Setup of build environment

The following packages are already provided on the cluster:

- BLAS
- LAPACK
- Boost 1.41.0 (but too old for AOFlagger v2.8.0 and later)
- cfitsio 3.240
- HDF5
- OpenMP

The cluster also provides a *module system* to load newer versions of the installed software.
We will activate it and replace the default version of gcc 4.4.7 with gcc 4.8.3 for C++11 support.

1.1 Paths

This instruction will install all packages in $HOME/local/.
And for the installation of Pyrap (python-casacore) we need a Python path with write access.
So I added the following lines to $HOME/.bashrc:

```
source /usr/share/Modules/init/sh  # activate module system, needs to be
done first
module load gcc  # use gcc 4.8.3 (instead of 4.4.7) for C++11 support

export PYTHONPATH=$HOME/local/lib64/python2.6/site-packages:$PYTHONPATH
export LD_LIBRARY_PATH=$HOME/local/lib:$LD_LIBRARY_PATH
export PATH=$HOME/local:$PATH
```

1.2 CMake

The 4 Newton head nodes provide CMake 2.6 by default, but the LOFAR Software requires at least 2.6.6. A sufficiently recent version of CMake 3.0.0 is provided via Environment Modules (see also http://modules.sourceforge.net/).
The environment modules also provide more recent HDF5 libraries. Both could be loaded by adding `module load cmake hdf5` to $HOME/.bashrc.
But unfortunately the CMake 3.0.0 loaded on the head node newl1 is broken. Furthermore loading the HDF5 libs has no effect on the libs found by CMake, which are still the system's default. So we don't load any module and don't add this line to the .bashrc. Instead we build our own CMake as follows:

```
mkdir -p $HOME/local/src
cd $HOME/local/src
wget http://www.cmake.org/files/v3.2/cmake-3.2.3.tar.gz
tar xf cmake-3.2.3.tar.gz
cd cmake-3.2.3
.bootstrap --prefix $HOME/local
make -j12
make install
```

1.3 Boost

Also a newer version of Boost is required since AOFlagger v2.8.0 and later:

```
cd $HOME/local/src
V_BOOST=1_63_0
wget wget
https://sourceforge.net/projects/boost/files/boost/$V_BOOST/boost_$V_BOOST.t
ar.bz2
tar xf boost_$V_BOOST.tar.bz2
cd boost_$V_BOOST
./bootstrap.sh
time ./b2 install --prefix=${HOME}/local
```

2. Installation of dependencies

Next we need to build and install these other packages as follows:

2.1 WCSLIB

The WCSLIB:

```
cd $HOME/local/src/
wget ftp://ftp.atnf.csiro.au/pub/software/wcslib/wcslib.tar.bz2
```

```
tar xf packages/wcslib.tar.bz2
cd wcslib-5.15/
./configure --prefix=$HOME/local/
time make -j install
```

2.2 FFTW

The FFTW lib:

```
V_FFTW=3.3.5
cd $HOME/local/src/
wget http://www.fftw.org/fftw-$V_FFTW.tar.gz
cd fftw-$V_FFTW
./configure --prefix=$HOME/local --enable-threads --enable-shared --enable-
float --enable-sse --enable-sse2 --enable-avx
time make -j install
```

3. Installation the LOFAR software

Now we can build the LOFAR software.

We will need the following CMake switches:

```
 1. -DCMAKE_INSTALL_PREFIX:PATH=$HOME/local
 2. -DBOOST_ROOT=$HOME/local/src/boost_1_63_0
 3. -DCMAKE_CXX_FLAGS="-lboost_thread -lboost_filesystem -L$HOME/local/lib"
 # 4. -DBoost_NO_BOOST_CMAKE=YES (not necessary with new installed Boost)
```

1.) Sets the installation path to $HOME/local.

2.) Necessary to find the new Boost installation

3.) Necessary to link against the new Boost installation

4.) Was required by some older versions of boost e.g. 1.41 with cmake-2.8.6-rc2, producing this problem:
https://stackoverflow.com/questions/9948375/cmake-find-package-succeeds-but-returns-wrong-path
resulting in

```
make[2]: *** No rule to make target
`/usr/lib64/lib64/libboost_program_options-mt.so.5', needed by `segment'.
Stop.
make[1]: *** [CMakeFiles/segment.dir/all] Error 2
make: *** [all] Error 2
```

It is not necessary any more with the new boost installation.

3.2 Casacore

Casacore also requires the CASA data. This is for example provided with CASA.
Here we assume CASA is already installed and $HOME/local/casa is a link to the CASA folder. Then the data is in $HOME/local/casa.
We will specify this via -DDATA_DIR, otherwise we need a ~/.casarc file with the line:
`measures.directory: $HOME/local/casa/data` .

`-DENABLE_TABLELOCKING=NO` switches off table locking, since read locks could not be acquired from the AIP Lustre file system leading to errors like:

```
msoverview: Version 20110407GvD
2015-06-30 19:32:19     INFO            Process 13592: waiting for read-lock
on file
/lustre/fkbreitl/data/L206894/L206894.cal/L206894_SAP000_SB200_uv.MS.dppp_00
001~00040/table.lock
Error: Error (Function not implemented) when acquiring lock on
/lustre/fkbreitl/data/L206894/L206894.cal/L206894_SAP000_SB200_uv.MS.dppp_00
001~00040/table.lock
```

For practical work there is no disadvantage in disabling table locking.

```
cd ~/local/src/
git clone https://github.com/casacore/casacore
mkdir build/casacore
cd build/casacore
time cmake ../../casacore -DBUILD_PYTHON=YES -DCMAKE_INSTALL_PREFIX=~/local
-DUSE_FFTW3=YES -DENABLE_TABLELOCKING=NO \
  -DUSE_OPENMP=YES -DDATA_DIR=~/local/casa/data -DUSE_HDF5=YES -
DCMAKE_PREFIX_PATH=$HOME/local
time make -j
make install
```

3.3 Pyrap

```
cd ~/local/src/
git clone https://github.com/casacore/python-casacore.git
cd python-casacore
python setup.py build_ext -I/usr/include/cfitsio:$HOME/local/include -
L$HOME/local/lib
python setup.py install --prefix=$HOME/local
```

3.3.1 Alternative Pyrap installation

Alternatively we can install Pyrap with *pip* if we install pip before:

```
wget https://bootstrap.pypa.io/get-pip.py
python get-pip.py --user
~/.local/bin/pip install python-casacore --global-option=build_ext --global-
option=-I/usr/include/cfitsio:$HOME/local/include \
   --global-option=-L$HOME/local/lib --install-option=--prefix=$HOME/local
```

3.4 Casarest

Now we continue with the rest:

```
cd ~/local/src/
svn co https://svn.astron.nl/casarest/trunk/casarest
mkdir build/casarest
cd build/casarest
cmake ../../casarest -DCMAKE_INSTALL_PREFIX:PATH=$HOME/local -
DCASACORE_ROOT_DIR=$HOME/local -DCMAKE_PREFIX_PATH=/usr/include/cfitsio -
DBUILD_ALL=1 \
  -DBOOST_ROOT=$HOME/local/src/boost_1_63_0 -DCMAKE_CXX_FLAGS="-
lboost_thread -lboost_filesystem -L$HOME/local/lib"
# not necessary any more -DBoost_NO_BOOST_CMAKE=YES
time make -j
make install
```

3.5 AOFlagger

The AOFlagger is also required since the LOFAR Offline release 2.15.0.
For further details see the LOFAR release link or the installation instructions.
The AOFlagger v2.8.0 and later requires a version of Boost later than 1.41.0.

```
V_AOF=2.9.0
cd ~/local/src/
wget
http://downloads.sourceforge.net/project/aoflagger/aoflagger-$AOVER/aoflagge
r-$V_AOF.tar.bz2
tar xf aoflagger-$V_AOF.tar.bz2
mkdir aoflagger-$V_AOF/build/
cd aoflagger-$V_AOF/build/
time cmake ../ -DCMAKE_INSTALL_PREFIX:PATH=$HOME/local -
DBUILD_SHARED_LIBS=YES -DBOOST_ROOT=$HOME/local/src/boost_1_63_0 \
  -DCMAKE_CXX_FLAGS="-lboost_thread -lboost_filesystem -L$HOME/local/lib"
time make -j
make install
```

3.6 The LOFAR packages

Finally we can build the LOFAR packages. We can chose between a release, an offline release or the

latest version in trunk/ by commenting in and out the corresponding RELEASE lines.

-DUSE_BACKTRACE=NO is necessary for building shared libraries (the default), but libiberty.so is not installed.
(See https://support.astron.nl/lofar_issuetracker/issues/8046).

```
cd ~/local/src/
RELEASE=LOFAR-Release-2_19 && svn checkout --ignore-externals -N
https://svn.astron.nl/LOFAR/branches/$RELEASE
#RELEASE=LOFAR && svn checkout --ignore-externals -N
https://svn.astron.nl/LOFAR/trunk $RELEASE
svn update $RELEASE/CMake
mkdir -p build/gnu_opt
cd build/gnu_opt
time cmake ../../$RELEASE -DCASACORE_ROOT_DIR=$HOME/local -
DBUILD_PACKAGES="DP3 Calibration" -DCMAKE_INSTALL_PREFIX:PATH=$HOME/local -
DUSE_BACKTRACE=NO
# -DUSE_LOG4CPLUS=NO was required in absence of the Log4cplus lib
time make -j
make install
```

Done!

Now the commands NDPPP and `calibrate-stand-alone` should be available on this system.

Enjoy!