# Installation Instructions for COSMOS

September 18, 2014

## Contents

## 1 System Requirements and User Prerequisites

The general conditions required for operation, and the desirable requirements for best performance are detailed in the following sections.

### 1.1 Operating Systems

The COSMOS system has been ported to all three major Operating Systems; Linux, MacOSX and Windows 7; including variants within these. As a Graphical User Interface application, MOST will only work under Linux if an X11 server is available.

The COSMOS Tools are built using the Qt development environment, and so is basically a 32 bit program. It should, however, work in both 32 and 64 bit environments.

### 1.2 Hardware

While MOST will operate with as little as 512 MiB of RAM, certain features can be quite memory intensive, which can cause significant slow downs. For best performance, it is recommended that you have at least 2 GiB.

MOST has been run on a single core, 32 bit, 1 GHz machine. However, the best performance will be experienced on a 64 bit multi core machine running at at least 2.5 GHz.

While the MOST program takes up only about 50 MiB of disk, its ancillary files take up another 500 MiB. If you also wish to keep the example data on disk, you will use another 500 MiB, so it would be wise to reserve 2 GiB for the entire installation.

Finally, MOST was designed with high resolution displays in mind and takes full advantage of a 1980x1200 pixel WUXGA screen. It will provide scroll bars automatically if used on a smaller display, but WUXGA and above is desirable.

## 2 COSMOS Installation in Red Hat Enterprise Linux 6 (RHEL6)

COSMOS has been installed in Ubuntu, CentOS and Scientific Linux and the licensed Red Hat Enterprise Linux 6 (RHEL6). The following are the instructions to configure and install COSMOS in RHEL6.

Installation Requirements:

- GCC 4.6 or above (for C++11 support)
- Qt 4.8 (for dynamic UI support), COSMOS is not compatible yet with Qt5

### 2.1 Install and configure RHEL6

It is assumed that a new user account is created with the name "*cosmos*". Check the Red Hat installation.

```
# uname -a
Linux cosmos 2.6.32-279.el6.x86_64
1 SMP Wed Jun 13 18:24:36 EDT 2012 x86_64 x86_64 x86_64 GNU/Linux
```

Linux Kernel is 2.6, 64 bit.

Add *cosmos* user to sudoers:

```
# su
# visudo -f /etc/sudoers
```

Add *cosmos* user to *dialout* group (for /dev/tty*) to avoid permission problems when interfacing with hardware:

```
# usermod -a -G dialout cosmos
```

### 2.2 Before anything else ... install all the RHEL6 updates

Make sure to install all the updates before trying to install anything else!!!.

```
# yum list updates
# yum update
or go to system updates in the administration menu
```

*Note that after installing the first updates, there will be more to be installed.*

## 2.3 Install an updated compiler (CGG4.6 or above)

In RHEL6, the default version of the gcc is 4.2.2, that is too old and not compatible with the necessary tools to install COSMOS: GCC 4.6 or above (for C++11 support)

To check the currently gcc version:

```
# gcc --version
>> gcc (GCC) 4.4.6 20120305 (Red Hat 4.4.6-4)
```

Installing gcc is a pain!!! found out that Red hat has a updated Developer Toolset that includes GCC 4.7. Please check the link references 3 and 4. A short description on how to install Red Hat Developer Toolset:

```
# rhn-channel --available-channels
# rhn-channel --add --channel=rhel-x86_64-workstation-dts-6
# yum install devtoolset-1.1
```

This should install gcc 4.7 into: */opt/rh/devtoolset-1.1/root/usr/bin/*. Check again the gcc version. If it's still saying gcc 4.4 then it's because the system path was not updated. There are at least 3 ways to activate the new gcc:

1) create a temporary environment variable

```
#export CC=/opt/rh/devtoolset-1.1/root/usr/bin/gcc
#export CPP=/opt/rh/devtoolset-1.1/root/usr/bin/cpp
#export CXX=/opt/rh/devtoolset-1.1/root/usr/bin/c++
```

2) enable the devtoolset bash

```
# scl enable devtoolset-1.1 bash
```

3) add the folder"/opt/rh/devtoolset-1.1/root/usr/bin/" permanently to the system $PATH environment variable.

```
# cd
# vi .bash_profile
>> add /opt/rh/devtoolset-1.1/root/usr/bin/ to the $PATH varible before other paths
```

Logout your user and login again and check if gcc is properly installed. That should do it! If not then continue reading . . . .

The following are instructions that may help installing gcc from source:

```
# yum install glibc-devel
>> check the gcc version again (if it's 4.6 or above no more installs are needed)
# gcc --version
```

The gcc version should be 4.6 or above, maybe installing qt5.1 will install gcc 4.8

```
# ./qt-linux-opensource-5.1.0-x86_64-offline.run
```

Unfortunately this does not install gcc 4.8 and gives and error that GLIBCXX_3.4.15 is missing.

Next step is to download and install gcc 4.7

```
# ftp://gd.tuwien.ac.at/gnu/gcc/releases/gcc-4.7.3/
# mkdir gcc-4.8.1-build
# /home/cosmos/Downloads/gcc-4.8.1/configure
# --prefix=/opt/gcc-4.8.1 --enable-languages=c,c++
# make
>> this step again gave an error that GLIBCXX\_3.4.15 is missing, it's a cyclic problem.
```

### 2.3.1 References

1. https://access.redhat.com/site/documentation/en-US/Red_Hat_Developer_Toolset/1/html/User_Guide/index.html
2. https://access.redhat.com/site/documentation/en-US/Red_Hat_Developer_Toolset/1/html/1.1_Release_Notes/ch-Usage.html
3. https://access.redhat.com/site/documentation/en-US/Red_Hat_Developer_Toolset/1/html/User_Guide/sect-Red_Hat_Developer_Toolset-Subscribe.html
4. https://access.redhat.com/site/documentation/en-US/Red_Hat_Developer_Toolset/1/html/User_Guide/sect-Red_Hat_Developer_Toolset-Install.html
5. http://preilly.me/2013/05/28/redhat-developer-toolset-1-1/

### 2.3.2 Troubleshooting

One problem that may arise frequently is that GLIBCXX_3.4.15 is missing. To check this type:

```
# strings /usr/lib64/libstdc++.so.6 | grep GLIBC
>> most likely the outcome will be
GLIBCXX_3.4
GLIBCXX_3.4.1
GLIBCXX_3.4.2
GLIBCXX_3.4.3
GLIBCXX_3.4.4
GLIBCXX_3.4.5
GLIBCXX_3.4.6
GLIBCXX_3.4.7
GLIBCXX_3.4.8
GLIBCXX_3.4.9
GLIBCXX_3.4.10
GLIBCXX_3.4.11
GLIBCXX_3.4.12
GLIBCXX_3.4.13
```

```
GLIBC_2.2.5
GLIBC_2.3
GLIBC_2.4
GLIBC_2.3.2
GLIBCXX_FORCE_NEW
GLIBCXX_DEBUG_MESSAGE_LENGTH
```

if this is the case then a gcc version really must be installed

## 2.4 Install Qt 4.8

In RHEL6, the default versions of Qt is 3.3 and is too old and not compatible with the necessary tools to install COSMOS (required Qt 4.8).

Trying to install the latest version of Qt from the RHEL repositories is not enough, because this only installs qt 4.4:

```
#yum install qt-devel
```

The default way to install Qt in Linux is listed in the following links. But for RHEL6 those procedures don't work properly because GLIBCXX_3.4.15 is required and not installed.

We found out that the best way to install Qt is to use the QtSDK 1.2.1 that is no longer available from the `http://qt-project.org` website, but is available from the `http://developer.nokia.com/Develop/Qt/Tools/`. The other option is to download the QtSDK from the HSFL repositories:

```
# rsync --progress user@hsfl.hawaii.edu:/home/shared/software/qt/
#QtSdk-offline-linux-x86_64-v1.2.1.run .
# ./QtSdk-offline-linux-x86_64-v1.2.1.run
>> then follow the instructions
```

If the above does not work then here's listed the default way to install Qt in Linux:

- download source files from `http://qt-project.org/downloads`
  - Qt libraries 4.8.5 for Linux/X11 (230 MB)
  - Qt Creator 2.8.0 for Linux/X11 64-bit (61 MB)
- unzip
- start install qt 4.8.4, Qt will be installed into /usr/local/Trolltech/Qt-4.8.5

  ```
  #./configure
  >> type 'o' for open source edition
  >> type 'yes' to accept licence
  # gmake
  # gmake install
  ```

  Then install qt creator

```
# chmod u+x qt-creator-linux-x86_64-opensource-2.8.0.run
# ./qt-creator-linux-x86_64-opensource-2.8.0.run
>> follow the instructions
```

### 2.4.1 References to install Qt in Linux:

1. http://qt-project.org/doc/qt-5.1/qtdoc/install-x11.html
2. http://qt-project.org/doc/qt-4.8/install-x11.html
3. http://qt-project.org/forums/viewthread/28697
4. http://stackoverflow.com/questions/9450394/how-to-install-gcc-from-scratch-with-gmp-mpfr-mpc-elf-w
5. http://gcc.gnu.org/wiki/InstallingGCC
6. http://qt-project.org/forums/viewthread/25550

### 2.4.2 Troubleshooting

If the following error appears:
"Project ERROR: Package gstreamer-app-0.10 not found gmake[1]: *** [WebCore/Makefile.WebKit] Error 2", then install gstreamer

```
yum install gstreamer
```

## 2.5 Install COSMOS

To install the COSMOS software it is best to have the source files on a cd/dvd, don't use USB at NPS. The best alternative was to download directly the source code from the COSMOS repository:

```
# svn co https://www.hsfl.hawaii.edu/svn/cosmos/trunk/software
# svn co https://www.hsfl.hawaii.edu/svn/cosmos/trunk/cosmosroot
# svn co https://www.hsfl.hawaii.edu/svn/cosmos/trunk/documentation
```

To compile the code make sure GCC 4.6 or above is installed and Qt 4.8 is also installed. Also it will be usefull to have xterm installed when running agents on Qt creator. To install xterm:

```
# yum install xterm
```

### 2.5.1 Install COSMOS Agents

To build the COSMOS agents:

```
# cd software/agents
# make
>> this should also compile all the dependencies (support, device, testbed)
```

### 2.5.2 Install MOST

Next, we need to compile the MOST program. Open the Qt project file in tools/MOST/MOST.pro in Qt Creator. Accept the default configurations (debug and release shadow builds) and hit Build. It will take a few minutes to compile all the code and dependencies. Next, go to project settings and on the run tab add 'hikasat' as an argument, this will load the hiakasat node into MOST. When MOST is running change to 'Archival Mode' to see the hiakasat data displayed in MOST.

## 2.6 Running COSMOS

### 2.6.1 Troubleshooting

If the agents are not working it may be that the ports are being blocked by the firewall. To check if the firewall is on:

```
# service iptables status
>> this will print a lot of stuff
```

To turn of the firewall

```
#service iptables off
```