Table of Contents

LHCb Software on Fedora Core Systems

Introduction

This page describes how to install the SLC binary software on Fedora Core systems. This is possible using the compatibility packages available for this platform, that provide both the necessary gcc 3.X.Y compilers and also the runtime libraries. The following guidelines have been tested on both Fedora Core 4, 5 and 6 systems, but are fairly general and thus most likely apply to other Fedora Core releases that also have available the necessary compatibility packages.

In addition the principles, if not the details, may apply to other linux distributions.

Preliminaries

First, I recommend making sure your system is fully up to date with security and bug fix releases. This can be done by running (as root)

```
> yum update
```

If the above does not work for you, you may need to setup yum on your system, although with the recent Fedora Core releases it should work out-the-box.

There are many resources available on the web explaining how this can be done. See, for instance :-

Fedora Yum Guide FedoraFAQ

or just "google" for "fedora core yum" and see what you get.

Alternatively, the APT or Smart systems could also be used although these are not really supported within Fedora Core.

You will also require some packages to be installed, such as python. *Note : Depending on what packages you selected to install when you first installed Fedora Core on your machine, there may be other packages that are also required. Unfortunately it is difficult to predict what these are after the fact so unless I manual remove all packages in turn and see what breaks, I cannot say for sure what is needed. If you do discover other rpms are required, please add the information here. If you are about to install Fedora Core on a machine my suggestion is to install all the software development packages offered to you.*

The following is a list of the currently known packages that are required :-

```
> yum install python ipython
```

Once this is done, you must make sure you have the appropriate compatibility packages installed (packages obsolete in main stream Fedora, but available to provide backwards compatibility if needed). This can be done using again the yum updater (again requires root privileges)

Fedora Core 5 and Older

With these releases, it is possible to install a gcc 3.2.3 compatible environment (compatible with SLC3 binaries) with,

```
> yum install compat-gcc-32 compat-gcc-32-c++ compat-gcc-32-g77 \
  compat-libf2c-32 compat-libstdc++-296 compat-libstdc++-33 compat-db \
    compat-readline43
```

Once this is done you should have available the gcc 3.2.3 compiler as "gcc32" and "g++32". The fortran 77 compiler (not available by default in the standard Fedora Core compiler GCC 4.x) is also available as the standard "g77". In addition, the compatible libstdc++ and a few other libraries should also be available.

Fedora Core 6 and Newer

With FC6, the gcc 3.2.3 compat libraries are removed and replaced by the gcc 3.4.6 libraries. Luckily, this corresponds to the general deployment and uptake of the SLC4 platform and binaries, which uses gcc 3.4.6 and thus is compatible.

To install these compat packages run :-

```
  > yum install compat-gcc-34 compat-gcc-34-g77 compat-gcc-34-c++ compat-libstdc++-33 compat-db
```

Once this is done you should have available the gcc 3.4.6 compiler as "gcc34" and "g++34". The fortran 77 compiler (not available by default in the standard Fedora Core compiler GCC 4.x) is also available as the standard "g77". In addition, the compatible libstdc++ and a few other libraries should also be available.

Install the LHCb SLC Binary Software Releases

Follow the instructions using the python script available here.

NOTE : In the above procedure I recommend you use the option "-m do_config".

In the following examples the MYSITEROOT variable is set to "/lhcb/sw" and the example application used is Brunel v28r1, although of course the instructions apply to any LHCb application or release.

Post Install "Fixups"

The rpms you installed during the preliminaries will have installed the compatible GCC compilers as "gccXY" and "g++XY" (in the following 'XY' refers to either '32' or '34' depending on the FC version you are using) instead of the standard and expected "gcc" and "g++" applications. You will therefore need to perform a few fixups after the installation to account for this.

CMT Requirements

You will need to alter the requirements file for CMT (e.g. For Brunel v28r1 this is ${MYSITEROOT}/contrib/CMT/v1r16p20040901/mgr/requirements) and edit the macros :-

- Macro "cc" replacing gcc with gccXY
- Macros "preproc", "cpp" and "shlibbuilder" to replace g++ with g++XY.
- Macros "preproc", "cpp" and replace c++ with g++XY (c++ command does not exist for GCC 3.X.Y but g++XY works just as well).

This fix up will of course need repeating for any new versions of CMT that are subsequently installed.

An example file for CMT v1r16p20040901, with the required edits is available here.

Similarly, you need to make similar changes to the requirements file for GaudiPolicy (e.g. for Brunel v28r1 this is ${MYSITEROOT}/lhcb/GAUDI/GAUDI_v17r2/GaudiPolicy/v5r21/cmt/requirements).

Currently all that is required is to replace g++ with g++XY in the "cpp" macro, although in the future it may be needed for other macros too (just search for gcc and g++ in the requirements file).

Again, this fix needs doing for all versions of Gaudi you wish to install.

An example file for Gaudi v17r2 (GaudiPolicy v5r21) is available here.

Environment Variables

It is also necessary to define a few environment variables, to cope with the non-standard gcc commands. For example for bash like shells :-

```
   > export CC=gccXY
   > export CXX=g++XY
   > export GCC_COMPILER=g++XY
   > export GCCXML_COMPILER=g++XY
```

Or for csh like shells :-

```
   > setenv CC gccXY
   > setenv CXX g++XY
   > setenv GCC_COMPILER g++XY
   > setenv GCCXML_COMPILER g++XY
```

where again XY is either 32 or 34 depending on which SLC binaries you are using.

Issues with SeLinux

SeLinux, a new feature in the latest Fedora Core releases (4 and 5), can cause some problems.

Fedora Core 4

I did not have any major problems with this release and the LHCb software.

However, just before upgrading I did notice that with the latest releases (Boole v11r0) I got a weird traceback message at the end of the job, which I did not see with older Boole releases. However, before sorting this out I upgraded to Fedora Core 5, which had its own problems (see next section).

If you do run into this yourself, and find the solution, please let me know.

Fedora Core 5 and 6

After upgrading to Fedora Core 5 (and subsequently with FC6) I saw messages such as

```
jonesc@localhost ~/cmtuser/Digi/Boole/v11r0/cmt > ../slc3_ia32_gcc323/Boole.exe ../options/v20060
/lhcb/sw/lcg/external/SEAL/SEAL_1_8_0/slc3_ia32_gcc323/lib/liblcg_SealZip.so: cannot restore segm
```

The easiest way to avoid such problems is just to disable SeLinux, but this is unsatisfactory as it circumvents the security advantages of SELinux. On a single user laptop this is not so bad, so if you want this just run the "Security Level and Firewall" utility and turn it off.

Fortunately this is not necessary though, as you can keep the benefits of SeLinux by adjusting the policy to cope.

If you look in the system logs you will see messages like

```
Mar 28 00:24:05 localhost kernel: audit(1143501845.463:1424): avc:  denied  { execheap } for  pid
```

or

```
Mar 28 08:33:26 localhost kernel: audit(1143531206.941:7978): avc:  denied  { execmod } for  pid=
```

Which indicates that you need to allow execheap and execmod. This can be done for the current session (until the next system reboot) by running as root

```
setsebool allow_execheap=true allow_execmod=true
```

To make the change permanent, add the -P option

```
setsebool -P allow_execheap=true allow_execmod=true
```

With this, I've had no further problems (so far).

TODO : The above calls turn off certain SeLinux limits for ALL cases. It should though be possible to develop SeLinux policies specific to the LHCb applications, to allow them to run without having to alter the global policies. I don't know enough about SELinux to do this yet.

Using CVS and Getpack

You will probably want to be able to run the LHCb script "getpack" to install private packages in your "cmtuser" directory, or just use standard cvs commands with the LHCB CERN CVS repository. This will require access to the CERN CVS server, which can be acheived in a few ways.

CVS access via kerberos

The default access method on linux is to use kerberos (kserver). This requires setting up AFS, which is though useful in itself. I personally recommend using the openafs rpms from atpms. You can either just download the rpms appropriate for your kernel and install using "rpm", or (easier) install the yum repo file and install via yum.

```
    > yum install openafs-kmdl openafs openafs-client
```

Also, once I had it running I found it very slow, which turned out to be a known problem. For me, the suggested solution worked OK, although I had to alter my local firewall settings to allow access to port 113.

Following this, you should be able to run getpack just like on lxplus. e.g.

```
     > klog <afs-username> -tmp
     > getpack Rec/Brunel v28r1
```

CVS access via SSH

It is also possible to use the the ssh authentication method. Detailed instructions on how to set things up for the CERN CVS server can be found here.

Once this is done, you then need to set some environment variables. For bash like shells :-

```
    > export CVSROOT=":ssh:username@isscvs.cern.ch:/local/reps/lhcb"
    > export GETPACK_USER=username
    > export CVS_RSH=ssh
```

or csh like shells :-

```
    > setenv CVSROOT ":ssh:username@isscvs.cern.ch:/local/reps/lhcb"
    > setenv GETPACK_USER username
    > setenv CVS_RSH ssh
```

Where you replace

```
 username
```

with your LHCb AFS username.

You can then use the getpack script with an option to force the ssh method :-

```
    > getpack -f ssh Rec/Brunel v28r1
```

Unfortunately, the above command will ask for your AFS password or your SSH key pass-phrase many times ! To avoid this, I recommend using an ssh-agent. This can be done by running the following commands :-

```
    > ssh-agent
    > ssh-add
```

Entering your password as prompted. Then, whenever you open an ssh connection requiring one of the ssh keys, the agent deals with the password and you do not have to enter it again.

Setup Scripts

That should be enough !!

However, it is perhaps convenient to have a single script, which sets up all the things needed as described above. Example scripts for bash like or csh like shells are available for download. Simply edit these files according to you choice of MYSITEROOT, your lxplus username etc.

Note, if you intend using this scripts for multiple user environments, then the environment variables in the script that depend on individual user-names should be moved to the users own .bashrc or .(t)cshrc configuration files.

Problems ?

In case of problems, errors above or just suggestions for improvements, please feel free to edit this page yourself !

Or if you prefer contact ChrisRJones.

ChrisRJones - 31 Jan 2006

This topic: LHCb > LHCbSLCOnFC
Topic revision: r29 - 2006-11-16 - unknown