Workspace Setup

Charlie Batchelor

LArSoft Workshop 2023

Outline

Task 1: Build our own **sbndcode** (used in following workshops).

Task 2: Create **another** testing area (build) and test some commands.

Before We Begin…

- Have these slides open for your own reference. These should **also** be opened in the VNC session we login to in a moment.
- The $ symbol before something indicates an **environment variable**. If things haven't been setup correctly, we will likely run into problems.
- Commands that should be run by you are in code font and black boxes. For example:

```
echo "hello world"
```

- Screenshots of my own output are included, so you can compare against your own.

Task 1

- **Build our own sbndcode**

Step 1 – Logging In

- Click here to reach the login screen for this workshop. The prompt on the right should be seen.

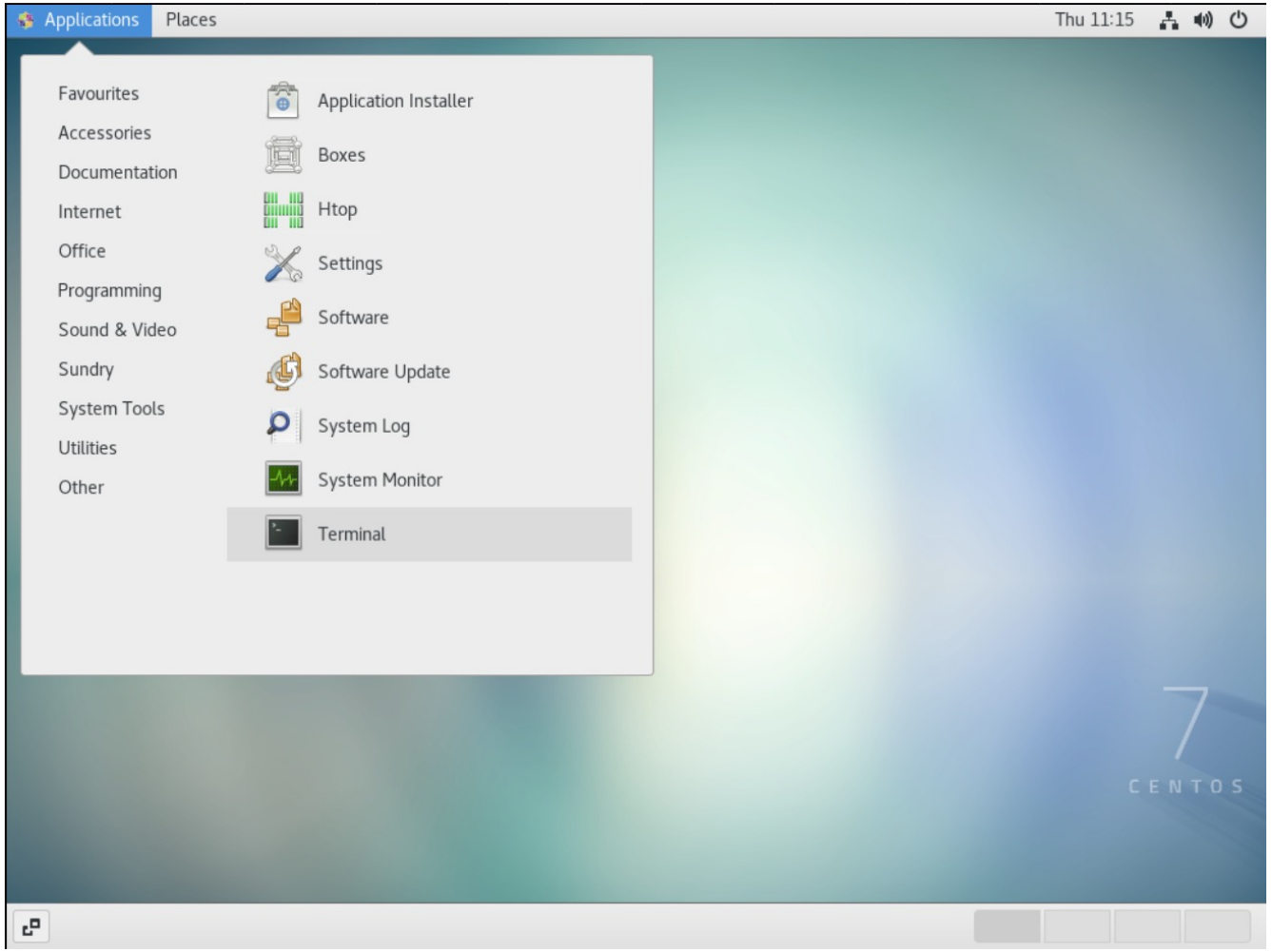
- Enter your Username of format **duneXX** and Password which *should* have been given to you by now. (Shout if it hasn't!)

Step 2 – Open A Terminal

- Follow:

  Applications →
  System Tools → Terminal

  to open a terminal window.



Step 3 – Create A Work Directory

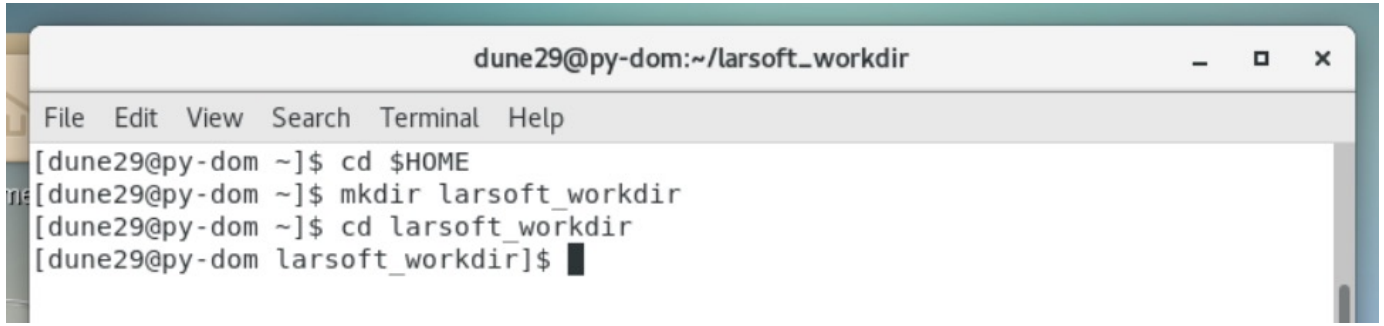
- The terminal opens in your home area. In the terminal, type the following to create a new work directory there:

```
cd $HOME
mkdir larsoft_workdir
```

- And then move into it:

```
cd larsoft_workdir
```

Expected Terminal Output

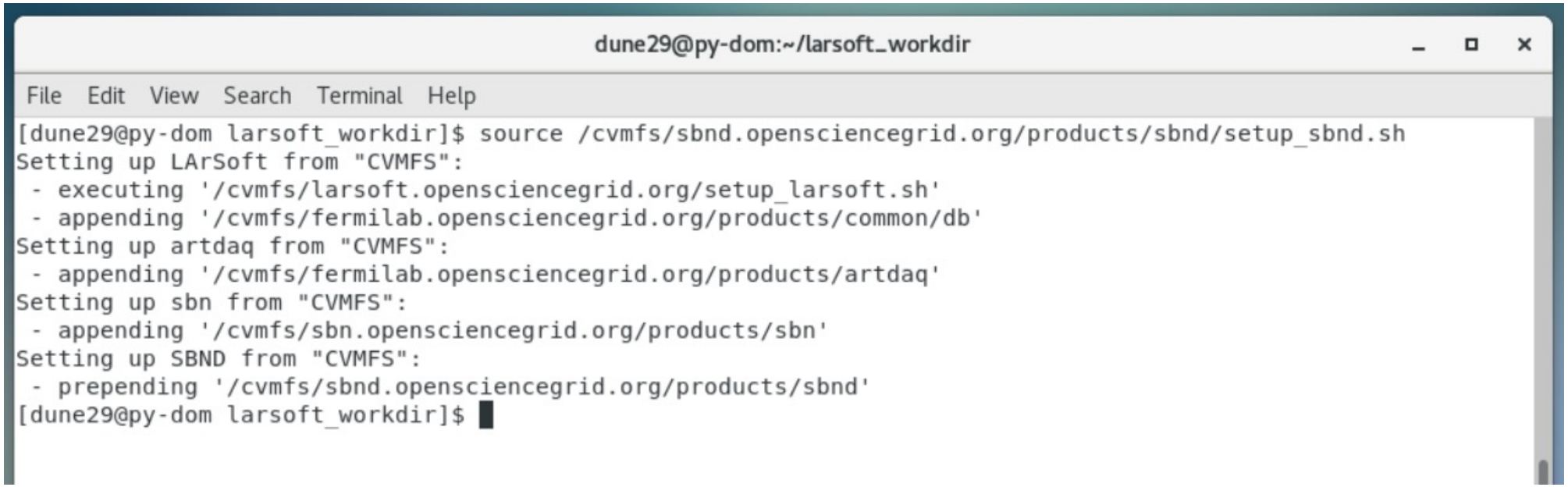


Step 4 – Setup sbndcode Using CVFMS

- CVMFS contains a script that we would like to use to setup sbndcode. Run it with:

```
source /cvmfs/sbnd.opensciencegrid.org/products/sbnd/setup_sbnd.sh
```

Expected Terminal Output



Step 5 – Make A New Development Area

- Executing that script has given us a new command called `mrb`

- Use it to turn the current directory into a new development area:

```
mrb newDev -v v09_78_02 -q prof:e20
```

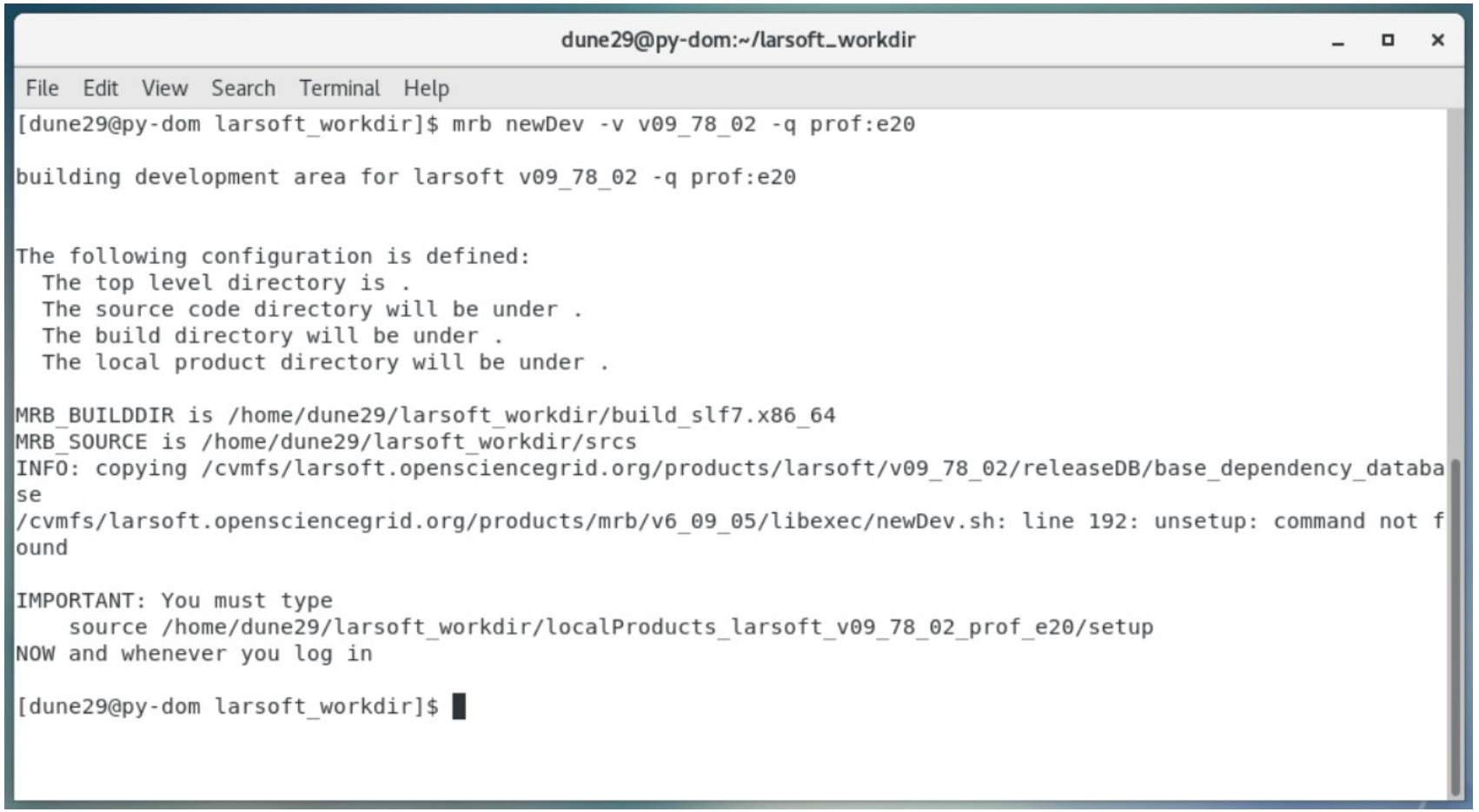
The -v allows us to specify the version of larsoft we'd like to setup a development area for.

v09_78_02 in this case!

The -q specifies a "qualifier". The 'prof' part indicates to build with minimal debug symbols, and the **e20** part determines the compiler to use.

Step 5 – Make A New Development Area

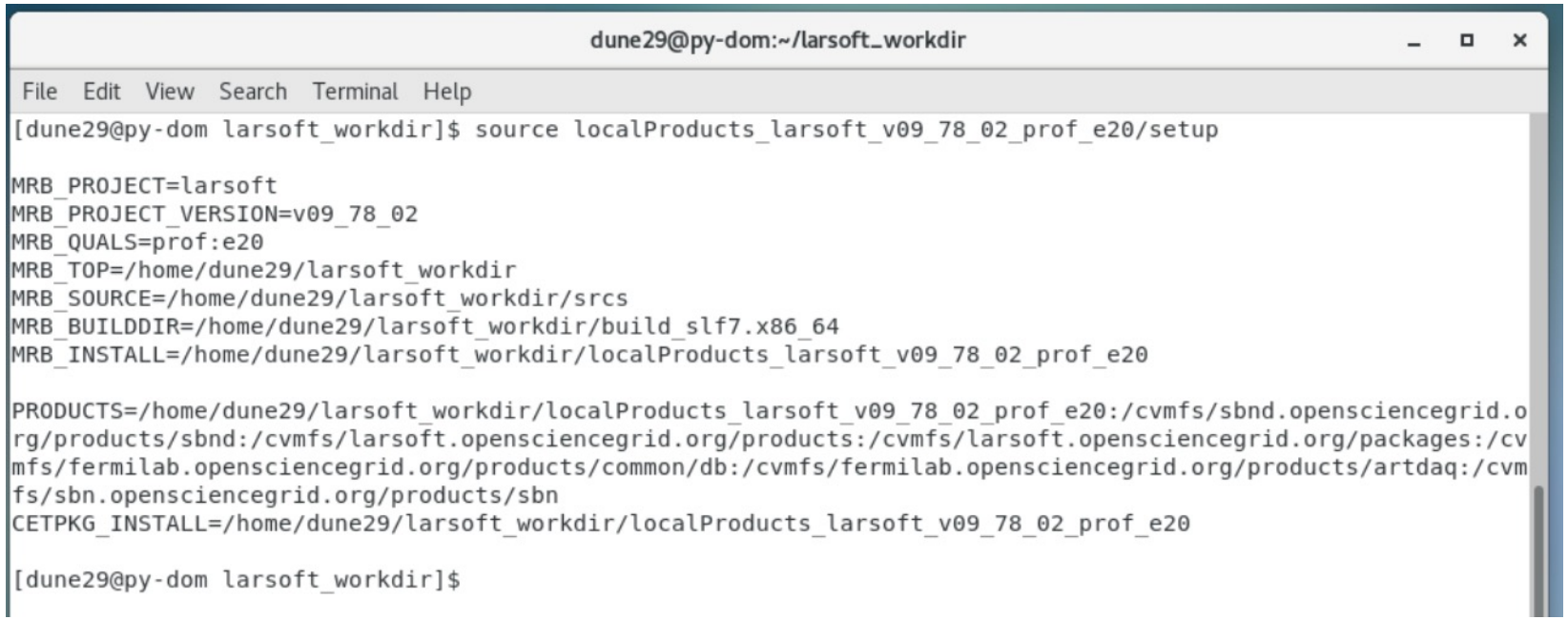
Expected Terminal Output



Step 6 – Source The Local Products

- As the terminal output suggested, we now want to source the local products placed in the current directory.

```
source localProducts_larsoft_v09_78_02_prof_e20/setup
```



Step 7 – Git Clone sbndcode With MRB

- Move to the srcs directory

  `cd srcs`
- And clone the sbndcode repository with

  `mrb g sbndcode`

Expected Terminal Output

```
[dune29@py-dom larsoft_workdir]$ cd srcs/
[dune29@py-dom srcs]$ mrb g sbndcode
Cloning into 'sbndcode'...
remote: Enumerating objects: 35619, done.
remote: Counting objects: 100% (3344/3344), done.
remote: Compressing objects: 100% (1159/1159), done.
remote: Total 35619 (delta 2243), reused 3210 (delta 2160), pack-reused 32275
Receiving objects: 100% (35619/35619), 78.88 MiB | 13.21 MiB/s, done.
Resolving deltas: 100% (24256/24256), done.
NOTICE: Adding sbndcode to CMakeLists.txt file
[dune29@py-dom srcs]$
```

Step 8 – Checkout Workshop Branch

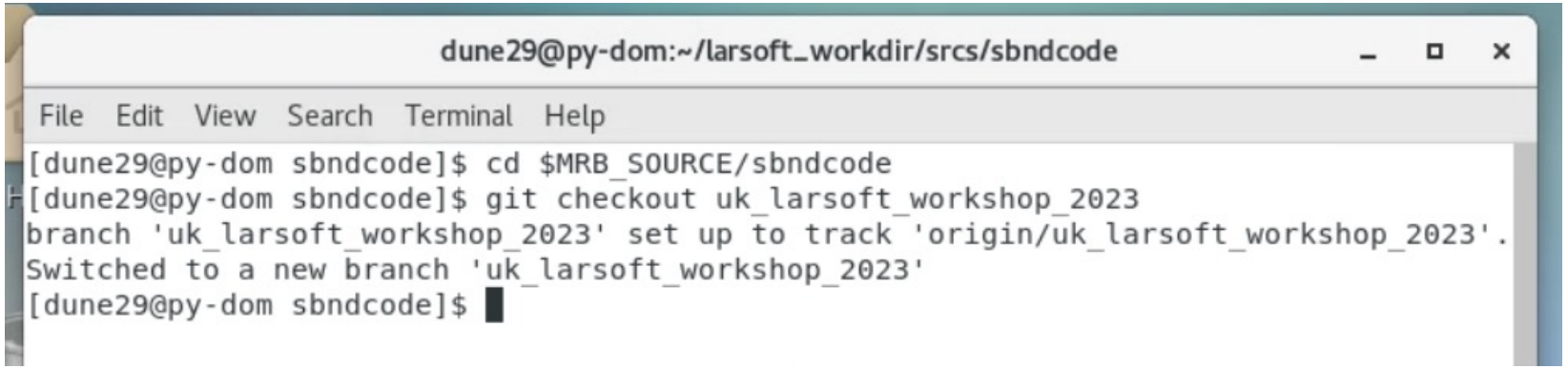
- Move to the srcs directory

  ```
  cd $MRB_SOURCE/sbndcode
  ```

- And clone the sbndcode repository with

  ```
  git checkout uk_larsoft_workshop_2023
  ```

Expected Terminal Output

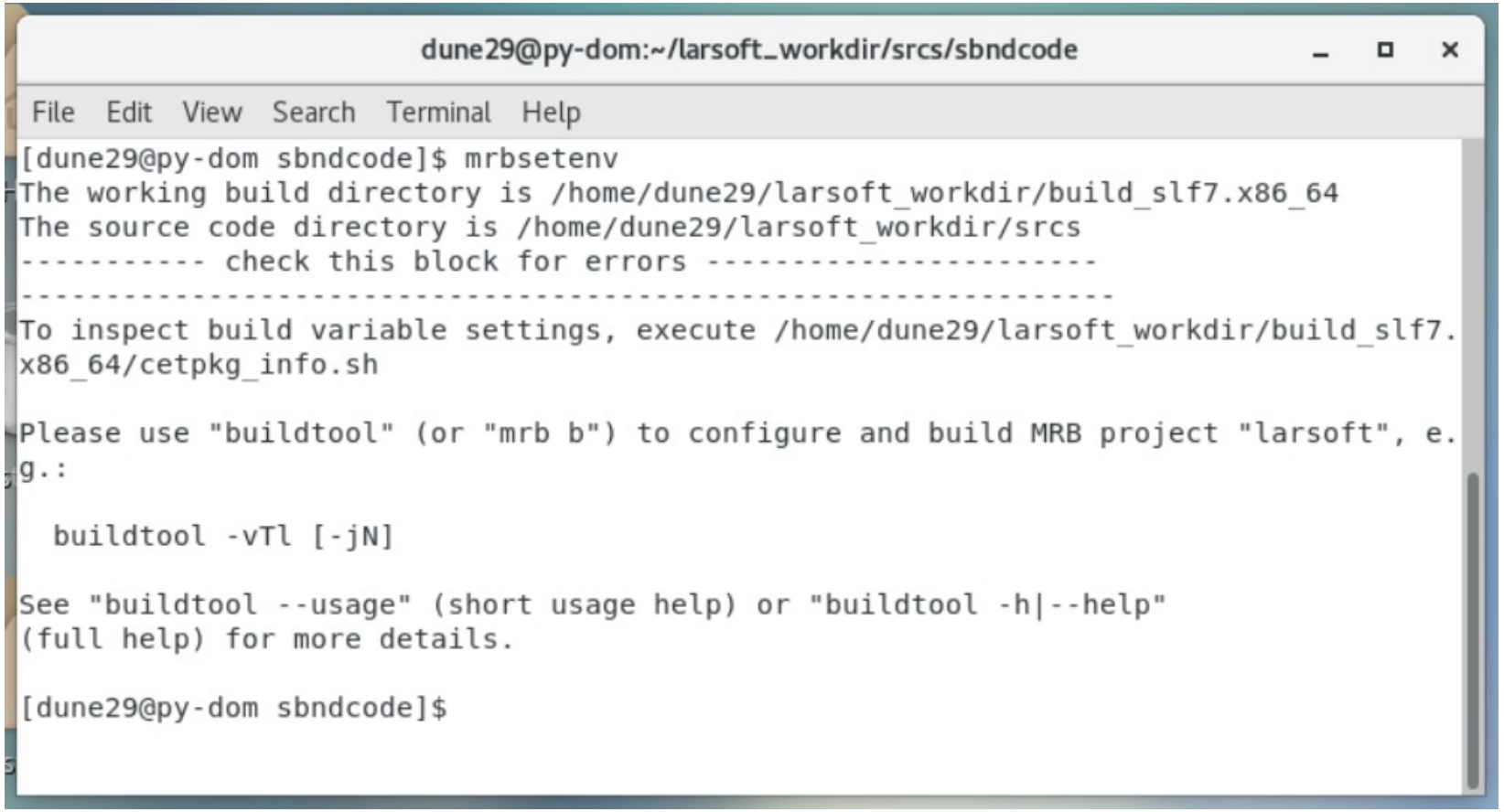


Step 9 – Set The Environment

- Use the following command to set the development envioronment:

```
mrbsetenv
```

Expected Terminal Output



Step 10 – Build & Install

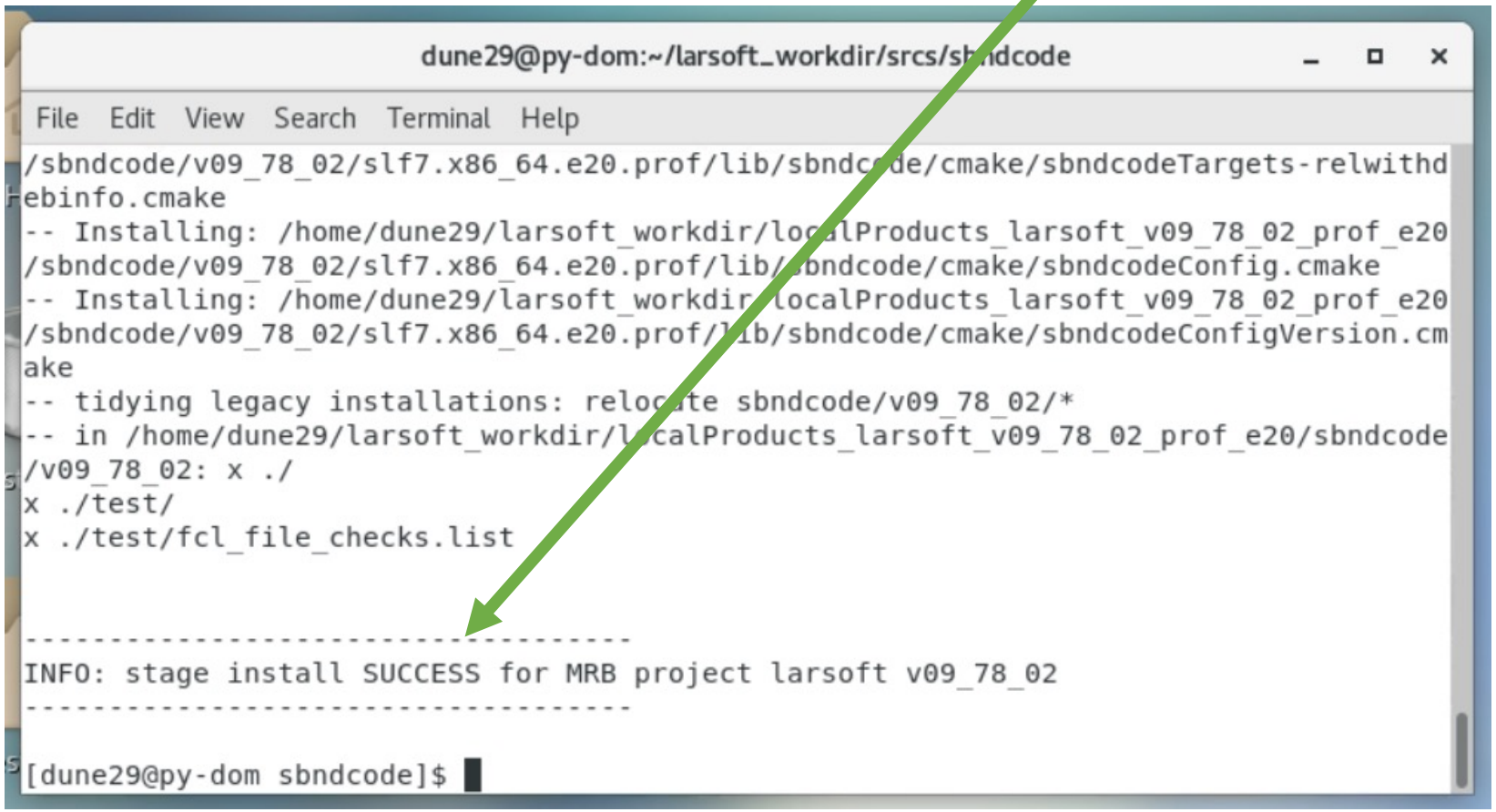
- Use the following command to **build** and **install** the code:

```
mrb i -j4
```

- The ”I” stands for “install” and the “-jX” indicates how many cores X to use for this process.

Expected Terminal Output

IMPORTANT BIT!

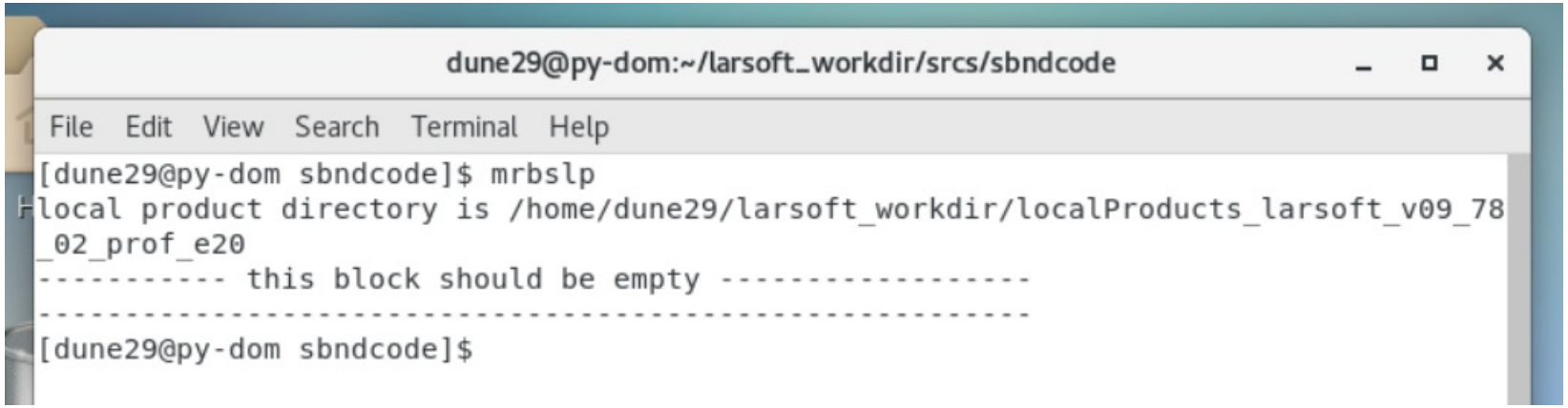


Step 11 – Setup The Local Install

- Use the following command to setup local products ‘slp’ :

```
mrbslp
```

Expected Terminal Output



This will setup all of the things installed in your **localProducts*/** directory.

Step 12 – Create A Setup Script

- Let's make a script to run the setup instructions, so you don't have to do this for each new workshop.

```
cd $MRB_TOP
touch setup.sh
```

- Paste the following lines into this script file using your favourite text editor e.g. **vim**, **emacs…**

```
source /cvmfs/sbnd.opensciencegrid.org/products/sbnd/setup_sbnd.sh
source localProducts_*/setup
mrbsetenv
mrbslp
```

- Save and close this file.

Step 13 – Test Setup Script

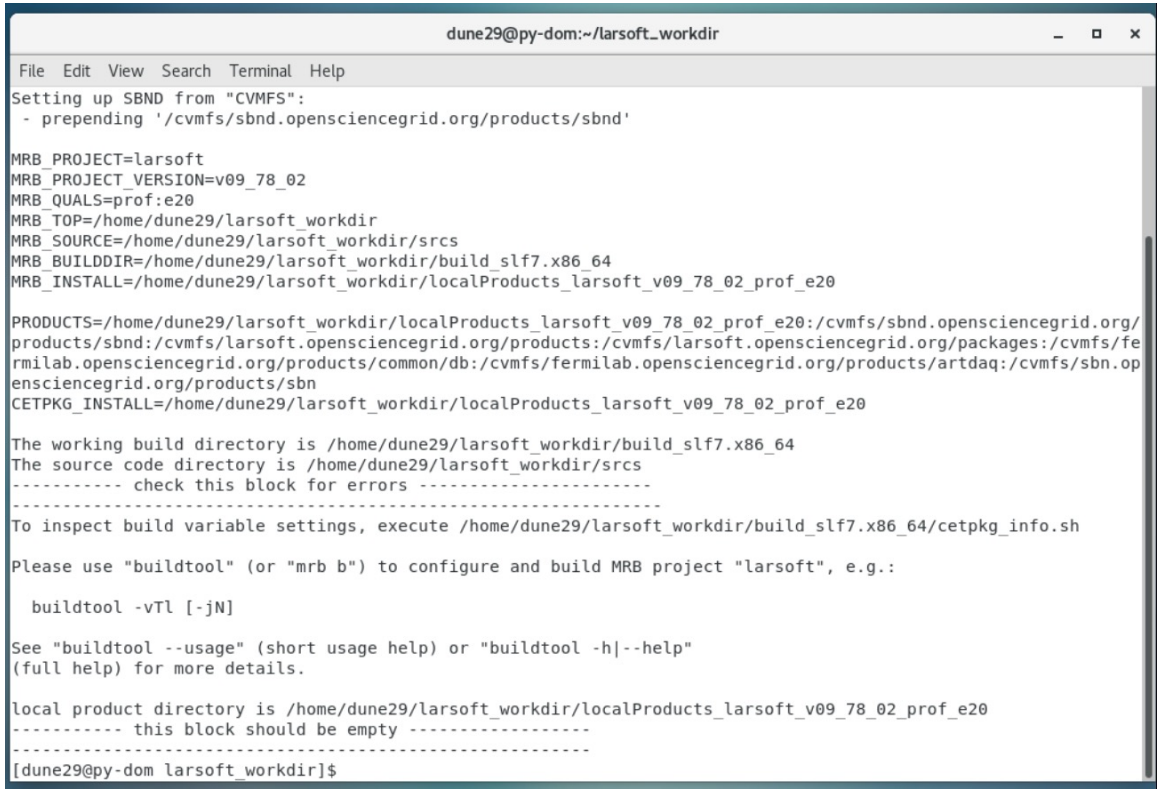
- Open a **new terminal window** and go to your work area.
  `cd $HOME/larsoft_workdir`

- Now try sourcing the script you just made in this directory:

  `source setup.sh`

- If no errors, we are done! You now have your own development area for the remaining workshops.

Expected Terminal Output



Task 2

- **Test LArSoft command(s), whilst practicing the above steps**

Step 1 – Create A New Test Area

- Open a **new terminal**. Create a new testing directory:

```
cd $HOME
mkdir larsoft_testdir
cd larsoft_testdir
```

- Repeat **steps 4 – 11** in Task 1 (Creating a new development area, checking out code, building and installing…)

Step 2 – Make a Data Directory

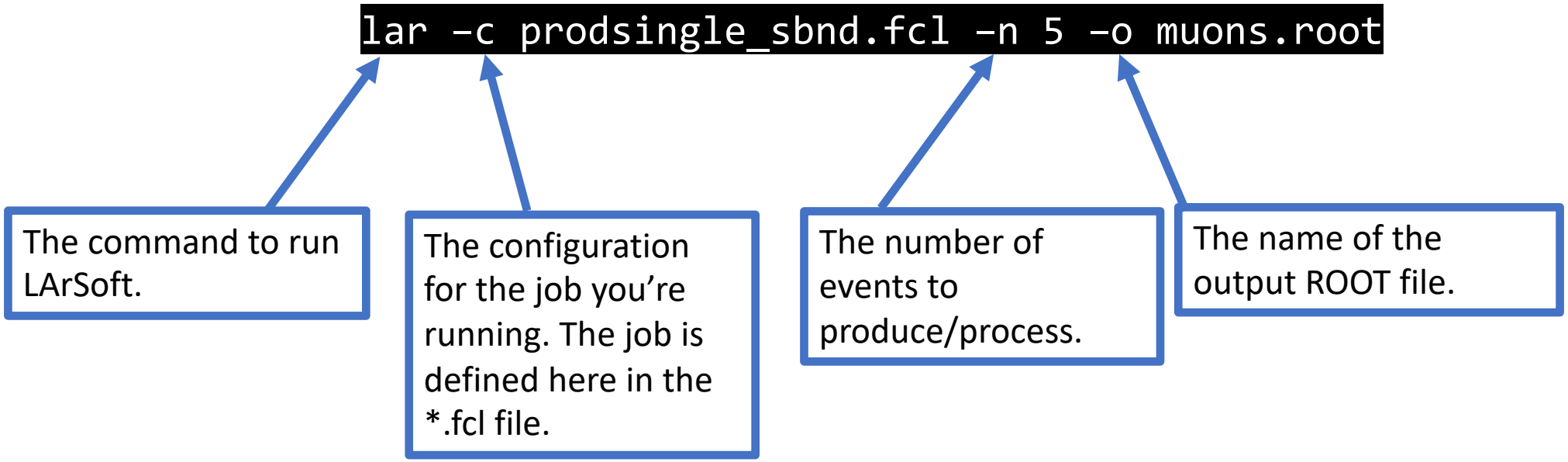
- Use the following commands to create a directory for your data:

```
cd $HOME
mkdir data
cd data
```

- When working on the Fermilab gpvms (General Purpose Virtual Machines), you should use: /EXPERIMENT_NAME/**data**/ directories to store **data** /EXPERIMENT_NAME/**app**/ directories to store **code**, these have a smaller quota but are backed up.

Step 3 – LArSoft Test Run

- The following is an example LArSoft = “lar” command. This is the one you’ll be learning about in the remaining workshops:



Step 3 – LArSoft Test Run

- The following is an example LArSoft = “lar” command. This is the one you’ll be learning about in the remaining workshops:

```
lar -c prodsingle_sbnd.fcl -n 5 -o muons.root
```

Expected Terminal Output

```
TimeTracker printout (sec)                          Min          Avg          Max          Median       RMS          nEvts
Full event                                          14.2137      14.4343      14.67        14.3934      0.159493     5
source:EmptyEvent(read)                             0.000425832  0.000495432  0.000578783  0.000476217  5.87875e-05  5
simulate:rns:RandomNumberSaver                      4.4915e-05   0.000126908  0.000420675  5.7043e-05   0.000146966  5
simulate:generator:SingleGen                        4.8018e-05   0.00121978   0.00589734   5.1163e-05   0.00233878   5
simulate:loader:PhysListLoader                       1.764e-06   2.0002e-06    2.281e-06    2.038e-06   1.80439e-07  5
simulate:largeant:larg4Main                         0.152829     0.165176     0.181004     0.167185     0.0109737    5
simulate:ionandscint:IonAndScint                    0.00755878   0.00853533   0.0098196    0.00824693   0.000771919  5
simulate:pdfastsim:PDFastSimPAR                      2.18741      2.35843      2.53169      2.32895      0.131857     5
simulate:simdrift:SimDriftElectrons                 0.710371     0.775507     0.816528     0.793669     0.0372001    5
simulate:daq:SimWireSBND                             9.73953      9.84461      9.90645      9.85296      0.0611941    5
[art]:TriggerResults:TriggerResultInserter          2.8364e-05   3.74166e-05  5.9642e-05   3.2802e-05   1.1319e-05   5
end_path:out1:RootOutput                             7.515e-06   1.16262e-05  2.3983e-05    9.069e-06   6.22671e-06  5
end_path:out1:RootOutput(write)                      1.24822      1.27896      1.30277      1.28864      0.0199714    5
%MSG-i NuRandomService:  RootOutput:out1@EndJob 19-Oct-2023 13:46:04 BST  ModuleEndJob
```

Complete!

- **You now should have a work area with a working build of sbndcode for the remaining workshops.**