

Chapter 14: Installing UPS and UPD from Bootstrap

CoreFUE is a bundled product which includes **UPS/UPD** and **perl**. It refers to the core components of the Fermi UNIX Environment (FUE).¹ When we discuss installing **UPS/UPD**, we're generally talking about **coreFUE** since **perl** is a required component. Here we describe how to use automated scripts to bootstrap **coreFUE**; that is, how to use a product called **bootstrap** to install **coreFUE** on a machine on which no prior versions of the **coreFUE** products are installed. Several project-specific configurations of **UPS/UPD** are available.

14.1 Before You Install

14.1.1 Predefined Configuration vs. Manual Install

Automated installs are provided by the Computing Division for UNIX and NT (with **CYGWIN**). You can choose a pre-defined configuration and use it as is or edit it, or you can define your own configuration. Alternatively, it is possible to run a manual installation from a bootstrap tar file (this option is not documented here; see *Bootstrap Core FUE Installation Summary* at ftp://ftp.fnal.gov/products/bootstrap/current/manual_install.html).

14.1.2 Installation Media

The UNIX install requires about 50M, and the NT about 70M. You can download the software from the KITS area at <ftp://fnkits.fnal.gov/products/ups/> (see also the `upd` and

1. Another bundled product you should know about is **FullFUE**. It consists of **coreFUE** plus **systools**, **sectools**, **futil**, **login_shells** and some "courtesy links". FUE is described in the document DR0009, available at <http://www.fnal.gov/docs/Recommendations/dr0009.html>.

perl areas). **CoreFUE (UPS/UPD/perl)** is included in the Fermi Linux package, available as Text or GUI installation over the network or as CDROM, from <http://www-oss.fnal.gov/projects/fermilinux/>.

CoreFUE by itself is not provided on CDROM, however the **bootstrap** product (<ftp://fnkits.fnal.gov/products/bootstrap/current/>) contains a `mkcdrom` script that you can use to make an iso image of **CoreFUE** and any other products you want to include. This is described in section 14.5 *Using mkcdrom (UNIX)*.

14.1.3 Check with your Experiment

Some experiments (e.g., CDF) have integrated **UPS** into their offline environments and provide the code independently. Check with your experiment before installing.

14.1.4 Do you want to run UPS without a Database?

Most users run **UPS** with a database. (How do you know whether you want to? See section 2.5 *Using UPS Without a Database*.) However if you plan to run **UPS** without a database, don't use the bootstrap procedure. Go to section 12.6 *Installing UPS for Use Without a Database*.

14.2 Downloading the Bootstrap and Configuration Files

The bootstrap script for UNIX is called `stage1.sh`. For NT it is called `stage1.bat`. They are both available for download from <ftp://ftp.fnal.gov/products/bootstrap/current/>. Besides that, the only other file you need to download is a configuration file. There are several configuration files from which to choose, as described below.

14.2.1 Predefined Configurations for UNIX

These configurations are intended mainly for on-site users with administrative privileges on their systems. Choose one of the following customized configuration files found under

<ftp://ftp.fnal.gov/products/bootstrap/current/configs/>:

local	puts a products area under /local/ups, creates a <i>products</i> account if needed, and installs fullFUE if it doesn't exist in other standard products areas (e.g., /afs/fnal/ups, /fnal/ups)
local.chmod	just like local, except we chmod/chgrp installed product areas to be group writable by group <i>products</i>
generic	puts a products area under /fnal/ups, creates a <i>products</i> account if needed, and installs fullFUE if it doesn't exist in other standard products areas (e.g., /afs/fnal/ups, /fnal/ups)
generic.chmod	just like generic, except we chmod/chgrp installed product areas to be group writable by group <i>products</i>
D0	sets up the standard three D0 RunII development UPS areas (/usr/products, /d0usr/products, and /d0dist/dist), creates a <i>products</i> account if needed, installs other products to facilitate D0 development (cvs , d0cvs , python) and installs fullFUE
test	sets up a minimal UPS area under /tmp/ups
SDSS	installs under /p/upsdb; installs coreFUE , photo and spectro

14.2.2 User-defined Configuration for UNIX

To create your own configuration file you'll need the `configurator` script, available from `ftp://ftp.fnal.gov/products/bootstrap/current/configurator`. Once it's downloaded, run the script by issuing the command **sh configurator** and answer the questions. This generates a user-customized configuration file called `config.custom`. Here is a sample session:

```
% sh configurator
Should we put symlinks and login shells in /usr/local[Yn]? n
Do you want to use the existing AFS products area[Yn]? n
Do you want to use any other existing products areas[yN]? n
Do you want to create a database local to this system[Yn]? y
What is the path to the database[/fnal/ups/db]?
/scratch/mengel/products/db
```

```

Of the following databases:
    1/scratch/mengel/products/db
From which one do you want to get coreFUE (ups, upd,
etc.)[1-1]? 1
Should we install the full FUE environment in the local
database[yN]? y
Warning -- installing fullFUE without writing in /usr/local
may not work
Can we write in /usr/local after all[Yn]? n
Are you sure we should do fullFUE[yN]? n
Writing config.custom

```

14.2.3 Predefined Configurations for NT

These configurations are mainly targeted at D0 and SVX Run II developers. Download one of the following customized configuration files from <ftp://ftp.fnal.gov/products/bootstrap/current/configs/>:

SVX	puts a products area under C:\products
D0cygC	puts the three D0 Run II products areas under C:\D0RunII
D0cygD	puts the three D0 Run II products areas under D:\D0RunII
D0cygE	puts the three D0 Run II products areas under E:\D0RunII

14.3 Customizing a Bootstrap Configuration



If you plan to use one of the available configuration files as is, skip down to section 14.4 *Running the Bootstrap Procedure*.

The bootstrap process includes only steps listed in the specified configuration file¹. Customizing the bootstrap process therefore only involves editing the configuration file you've chosen. Notice that the configuration file contains several types of instructions. The statements are grouped by type and executed in the order shown below. We recommend that you keep them in this order as you edit your configuration file.

1. It also puts a temporary UPS products area in `$TMPDIR/bootups` (where `$TMPDIR` defaults to `/var/tmp`), which it cleans out when its done. If you change `$TMPDIR`, it puts a symlink in `/var/tmp/bootups` that points to `$TMPDIR/bootups`.

14.3.1 Bootstrap Configuration File Statement Definitions

...

Comments

set_variable <variable>=<string>

Sets variables for the script to use. There are two required variables in the configuration file:

bootbase URL of bootstrap files

upsdb_list UPS database list; separate database paths with a space (used to create \$SETUPS_DIR/upsdb_list). To include the AFS UPS database, include /afs/fnal.gov/ups/db.

You can also define your own variables here and use them in later statements.

check_space </path/to/directory> <size>

Verifies that blocks of the specified size are free in /path/to/directory.

download_file <URL> [<local>]

Pulls down a file for the script to use. It can be any URL, any protocol.

pre_install_command "<shell_command>"

Runs specified shell command after any **download_file** lines and before any **create_user** lines

create_user <name> <uid> <gid> <home>

Adds a userid entry to /etc/passwd. (This has no effect on NT systems.)

**create_db <path> <dbconfig_file> <updconfig_file>
<owner>**

Creates a UPS database. If either of the files is specified as "-", the command uses the corresponding template file within the downloaded UPS product. If the owner is specified as "-", it uses the current userid.

install_coreFUE <database>

Installs the **coreFUE** product in the specified database, with the owner *products* if possible (it checks the /etc/password file for a line starting with *products*:).

install_as <user> <product> <upd_install_options>

Runs a **upd install -G -c <upd_install_options>** of the specified product, as **<user>** if possible (again, it checks the /etc/password file for a line starting with **<user>**:).

do_ups <action>

Runs an action from the table file of the **UPS** product via the command **ups <action>**.

make_courtesy_links

Makes links in `/usr/local/etc` to `setups.sh`, etc.

post_install_command "<shell_command>"

Runs the specified command, after all the preceding steps are complete.

14.3.2 Sample Customization

In this example, we assume that the bootstrap configuration file `D0` has already been downloaded. We want to edit it such that it prevents the bootstrap from downloading the third listed `dbconfig` file, `dbconfig3.D0`.

First we create a replacement `dbconfig` file (we'll call it `/tmp/dbconfig`), and then change the configuration file to refer to it. Let's look at the (abbreviated) file contents prior to the change (affected lines in **bold**):

```

...
#-----
# files to download
#           remote                               local
#           -----                               -----
download_file           $bootbase/downloads/updconfig.D0
updconfig
download_file $bootbase/downloads/dbconfig1.D0
download_file $bootbase/downloads/dbconfig1.D0
download_file $bootbase/downloads/dbconfig3.D0
...
#-----
# databases
#           database                               dbconfig           updconfig
owner
#           -----                               -----           -----
-----
create_db /usr/products/upsdb           dbconfig1.D0           updconfig
products
create_db /d0dist/dist/upsdb           dbconfig2.D0           updconfig
products
create_db /d0usr/products/upsdb           dbconfig3.D0           updconfig
products

```

To make the change, we comment out the last **download_file** statement and change the name of the `dbconfig` file in the third **create_db** statement:

```

#-----
# files to download
#           remote                               local

```

```
#          -----
download_file          $bootbase/downloads/updconfig.D0
updconfig
download_file $bootbase/downloads/dbconfig1.D0
download_file $bootbase/downloads/dbconfig1.D0
# download_file $bootbase/downloads/dbconfig3.D0
...
```

```

#-----
# databases
#      database                dbconfig      updconfig
owner
#      -----                -----      -----
-----
create_db /usr/products/upsdb  dbconfig1.D0  updconfig
products
create_db /d0dist/dist/upsdb   dbconfig2.D0  updconfig
products
create_db /d0usr/products/upsdb /tmp/dbconfig  updconfig
products

```

14.4 Running the Bootstrap Procedure

To run the bootstrap, invoke the `stage1.sh[bat]` script and give it your configuration file as an argument (as shown for both UNIX and NT below). The `stage1` script downloads the bootstrap tar file and unwinds it, then runs a `stage2` script that first verifies the integrity of the configuration script and then executes it.

If `stage2` finds errors, it outputs the information to a log and tells you how to reinvoke the `stage2` script.¹ This allows you to restart the bootstrap process where you left off.

14.4.1 UNIX

To run the bootstrap, issue the command (from any directory):

```
% sh stage1.sh <config-file-name>
```

and the install will either take place, as the following output shows:

```

0% complete
(several minutes pass, the percentage updates...)
100% complete
Bootstrap succeeded.

```

or tell you of any impediments. For example:

```

The ups bootstrap cannot proceed because:
    * ups products database area already exists under
    /local/ups
    * ups setups scripts already exist under /usr/local/etc

```

1. The log file is maintained as `$TMPDIR/bootups.log`, where `$TMPDIR` defaults to `/var/tmp` on UNIX and `%TEMP%\` on NT. The message will be at the end of the (rather long) log file.



* ups setups scripts already exist under /local/ups/etc
 These directories must be cleared before we can proceed.
 If you get error messages, and you want to proceed anyway, you can run:
% sh stage1.sh -F <config-file-name>
 to force the install (but we do not recommend it).

14.4.2 NT

In a DOS command prompt window, issue the command:

U:\> stage1.bat <config-file-name>

If the install succeeds, you will see output like this:

```
Redirecting output to C:\TEMP\bootups.log
(window pops up showing percentage done)
Bootstrap succeeded!
```

A `Cygwin<version>.bat` file appears on your desktop that you can use to start **CYGWIN**.

If the install fails, it should provide error messages, for example:

```
"The bootstrap cannot proceed because:"
* ups products database area already exists under
C:\D0RunII\d0usr\products
* ups products database area already exists under
C:\D0RunII\d0usr\products
* ups products database area already exists under
C:\D0RunII\d0dist\dist
"These directories must be cleaned out before the bootstrap
can run"
```



If you get error messages, and you want to proceed anyway, you can run:
U:\> stage1.bat -F <config-file-name>
 to force the install (again, not recommended).

14.5 Using mkcdrom (UNIX)

CoreFUE by itself is not provided on CDROM, however the **bootstrap** product (<ftp://fnkits.fnal.gov/products/bootstrap/current/>) contains a `mkcdrom` script that you can use to make an iso image of **CoreFUE** and any other products you want to include. We'll show how to do this by way of an example.

14.5.1 Create the ISO Image and Put it on CDROM

Say you want to be able to bootstrap from a CD for the `SDSS` configuration, listed in section 14.2.1 *Predefined Configurations for UNIX*. You want to do this on Linux and IRIX machines, using a Linux box.

First, drop the bootstrap files into `/tmp/cdrom`:

```
% sh mkcdrom -b -f Linux+2 /tmp/cdrom
```

Then add the products:

```
% sh mkcdrom -f Linux /tmp/cdrom photo
```

```
% sh mkcdrom -f Linux /tmp/cdrom spectro
```

Next, make an ISO image of the `/tmp/cdrom` file tree:

```
% cd /tmp/cdrom
```

```
% mkisofs -o /tmp/cdrom.iso -LlNRv .
```

Then you can put that `cdrom.iso` file on an actual CD using **cdrecord** (available for Linux, Solaris and Windows). You may want to refer to the product documentation at

http://freshmeat.net/redirect/cdrecord/1139/url_homepage/ in case the speed or device is different. At least run:

```
% cdrecord -scanbus
```

to verify that it sees a CD-ROM device, and see what the dev value associated with it is (use the triple value in the first column of the same line, e.g., `0,0,0`). Then run the program:

```
% cdrecord -eject speed=2 dev=0,0,0 /tmp/cdrom.iso
```

... and clean up:

```
% rm -rf /tmp/cdrom /tmp/cdrom.iso
```

14.5.2 Run the Bootstrap from CDROM

On your target system, mount that cdrom, and run the bootstrap (version may change):

```
% mkdir /tmp/cdrom
```

```
% mount -r -t iso9660 /dev/cdrom /tmp/cdrom
```

```
% cd /tmp/cdrom/prd/bootstrap/v2_1
```

```
% sh ./stagel.sh configs/SDSS
```

You're done!