The Linux Printing Usage HOWTO

by Matt Foster <mwf@engr.uark.edu> v1.0, 5 August 1995

1 Introduction

This document describes how to use the line printer spooling system provided with the Linux operating system. If you are looking for a guide to setting up your print environment, please see the Linux Printing Setup HOWTO.

1.1 Linux Printing HOWTO History

This version of the Linux Printing HOWTO is a complete rewrite of the one originally written by Grant Taylor <grant@god.tufts.edu> and Brian McCauley <B.A.McCauley@bham.ac.uk>. I have tried to keep with the coverage of material presented by Grant and Brian's HOWTO, but I have drastically modified the style of presentation and the depth of material covered. I feel that this makes the HOWTO more complete and easier to read. I can only hope that you agree.

1.2 Copyrights and Trademarks

Some names mentioned in this HOWTO are claimed as copyrights and/or trademarks of certain persons and/or companies. These names appear in full or initial caps in this HOWTO.

The Linux Printing Usage HOWTO v1.0 (c) 1995 Matt Foster.

Unless otherwise stated, Linux HOWTO documents are copyrighted by their respective authors. Linux HOWTO documents may be reproduced and distributed in whole or in part, in any medium physical or electronic, as long as this copyright notice is retained on all copies. Commercial redistribution is allowed and encouraged; however, the author would like to be notified of any such distributions.

All translations, derivative works, or aggregate works incorporating any Linux HOWTO documents must be covered under this copyright notice. That is, you may not produce a derivative work from a HOWTO and impose additional restrictions on its distribution. Exceptions to these rules may be granted under certain conditions; please contact the Linux HOWTO coordinator at the address given below.

In short, we wish to promote dissemination of this information through as many channels as possible. However, we do wish to retain copyright on the HOWTO documents, and would like to be notified of any plans to redistribute the HOWTOs.

If you have questions, please contact Greg Hankins, the Linux HOWTO coordinator, at <gregh@sunsite.unc.edu>. You may finger this address for phone number and additional contact information.

1.3 Downloading the Linux Printing HOWTOs

I recommend that if you want to print a copy of this HOWTO that you download the PostScript version. It is formatted in a fashion that is aesthetically appealing and easier to read. You can get the PostScript version from one of the many Linux distribution sites (such as SunSITE (`ftp://sunsite.unc.edu/Linux/docs/HOWTO/`)), or directly from my World Wide Web page at (`http://www.engr.uark.edu/ mwf/pht/`).

1.4 Feedback

Questions, comments, or corrections for this HOWTO may be directed to <mwf@engr.uark.edu>.

1.5 Acknowledgments

Thanks go out to all of the people who took the time to read the alpha version of this HOWTO and respond with many helpful comments and suggestions–some of you may see your comments reflected in the version.

I would also like to thank Grant and Brian for the loads of information that they have given me during the transition of authorship.

2 Printing Under Linux

This section discusses how to print files, examine the print queue, remove jobs from the print queue, format files before printing them, and configure your printing environment.

2.1 History

The Linux printing system—the *lp* system—is a port of the source code written by the Regents of the University of California for the Berkeley Software Distribution version of the UNIX operating system.

2.2 Basic Printing

By far, the most simplistic way to print in the Linux operating system is to send the file to be printed directly to the printing device. One way to do this is to use the *cat* command. As the `root` user, one could do something like

```
# cat thesis.txt > /dev/lp
```

In this case, `/dev/lp` is a symbolic link to the actual printing device–be it a dot-matrix, laser printer, typesetter, or plotter. (See *ln*(1) for more information on symbolic links.)

For the purpose of security, only the `root` user and users in the same group as the print daemon are able to write directly to the printer. This is why commands such as *lpr*, *lprm*, and *lpq* have to be used to access the printer.

Because of this, users have to use *lpr* to print a file. The *lpr* command is responsible for taking care of all of the initial work needed to print the file, and then it hands control over to another program, *lpd*, the line printing daemon. The line printing daemon then tells the printer how to print the file.

When *lpr* is executed, it first copies the specified file to a certain directory (the spool directory) where the file remains until *lpd* prints it. Once *lpd* is told that there is a file to print, it will spawn a copy of itself (what we programmers call forking). This copy will print our file while the original copy waits for more requests. This allows for multiple jobs to be queued at once.

The syntax of *lpr*(1) is a very familiar one,

```
$ lpr [ options ] [ filename ... ]
```

If a `filename` is not specified, *lpr* assumes that the input should come from standard input (usually the keyboard, or another program's output). This enables the user to redirect a command's output to the printing device. As such,

```
$ cat thesis.txt | lpr
```

or, something more powerful, like

```
$ pr -l60 thesis.txt | lpr
```

The *lpr* command accepts several command-line arguments that allow a user to control how it works. Some of the most widely used arguments are: -**P**printer specifies the printer to use, -**h** suppresses printing of the burst page, -**s** creates a symbolic link instead of copying the entire file to the spool directory (useful for large files), and -#num specifies the number of copies to print. An example interaction with *lpr* might be something like

```
$ lpr -#2 -sP dj thesis.txt
```

This command would create a symbolic link to the file `thesis.txt` in the spool directory for the printer named *dj*, where it would be processed by *lpd*. It would then print a second copy of `thesis.txt`.

For a listing of all the options that *lpr* will recognize, see *lpr*(1).

2.3 Viewing the Print Queue

Sometimes it is useful to know what jobs are currently in a particular printer's queue. This is the sole task of the *lpq* command.

To see what is in the queue of the default printer (as defined by /etc/printcap), use

```
$ lpq
lp is ready and printing
Rank   Owner      Job  Files                              Total Size
active mwf        31   thesis.txt                         682048 bytes
```

2.4 Canceling a Print Job

Another useful feature of any printing system is the ability to cancel a job that has been previously queued. To do this, use *lprm*.

```
$ lprm -
```

The above command cancels all of the print jobs that are owned by the user who issued the command. A single print job can be canceled by first getting the job number as reported by *lpq*, and then giving that number to *lprm*. For example,

```
$ lprm 31
```

would cancel job 31 (*thesis.txt*) on the default printer.

3 Miscellaneous Items

This section discusses some of the miscellaneous things that you may want to know about printing under Linux.

3.1 Formatting

Since most ASCII files are not formatted for printing, it is useful to format them in some way before they are actually printed. This may include putting a title and page number on each page, setting the margins, double spacing, indenting, or printing a file in multiple columns. A common way to do this is to use a print preprocessor such as *pr*.

```
$ pr +4 -d -h"Ph.D. Thesis, 2nd Draft" -l60 thesis.txt | lpr
```

In the above example, *pr* would take the file thesis.txt and skip the first three pages (+4), set the page length to sixty lines (-l60), double space the output (-d), and add the phrase "Ph.D. Thesis, 2nd Draft" to the top of each page (-h). *Lpr* would then print *pr*'s output. See its on-line manual page for more information on using *pr*.

3.2 The PRINTER Environment Variables

All of the commands in the Linux printing system accept the -**P** option. This option allows the user to specify which printer to use for output. If a user doesn't specify which printer to use, then the default printer will be assumed as the output device.

Instead of having to specify a printer to use every time that you print, you can set the PRINTER environment variable to the name of the printer that you want to use. This is accomplished in different ways for each shell. For *bash* you can do this with

```
$ PRINTER="printer_name"; export PRINTER
```

in *csh*, you can do it with

```
% setenv PRINTER "printer_name"
```

These commands can be placed in your login scripts (.profile or .cshrc), or issued on the command-line. (See *bash*(1) and *csh*(1) for more information on environment variables.)

3.3 Printing PostScript files

Printing PostScript files on a printer that has a PostScript interpreter is simple; just use *lpr*, and the printer will take care of all of the details for you. For those of us that don't have printers with PostScript capabilities, we have to resort to other means. Luckily, there are programs available that can make sense of PostScript, and translate it into a language that most printers will understand. Probably the most well known of these programs is Ghostscript.

Ghostscript's responsibility is to convert all of the descriptions in a PostScript file to commands that the printer will understand. To print a PostScript file using Ghostscript, you might do something like

```
$ gs -dNOPAUSE -sDEVICE=deskjet -sOutputFile=|lpr thesis.ps
```

Notice in the above example that we are actually piping the output of Ghostscript to the *lpr* command by using the -**sOutputFile** option.

Ghostview is an interface to Ghostscript for the X Window System. It allows you to preview a PostScript file before you print it. Ghostview and Ghostscript can both be downloaded from (`ftp://prep.ai.mit.edu/pub/gnu/`).

3.4 Printing TeX files

One of the easiest ways to print TeX files is to convert them to PostScript and then print them using Ghostscript. To do this, you first need to convert them from TeX to a format known as DVI (which stands for device-independent). You can do this with the *tex*(1) command. Then you need to convert the DVI file to a PostScript file using *dvips*. All of this would look like the following when typed in.

```
$ tex thesis.tex
$ dvips thesis.dvi
```

Now you are ready to print the resulting PostScript file as described above.

3.5 Printing `troff` formatted files

```
$ groff -Tascii thesis.tr | lpr
```

or, if you prefer,

```
$ groff thesis.tr > thesis.ps
```

and then print the PostScript file as described above.

4 Answers to Frequently Asked Questions

Q1. How do I prevent the staircase effect?

A1. The staircase effect is caused by the way some printers expect lines to be terminated. Some printers want lines that end with a carriage-return/line-feed sequence (DOS-style) instead of the default line-feed sequence used for UNIX-type systems. The easiest way to fix this is to see if your printer can switch between the two styles somehow—either by flipping a DIP switch, or by sending an escape sequence at the start of each print job. To do the latter, you need to create a filter (see Q2 and *Foster*95*b*).

A quick fix is to use a filter on the command-line. An example of this might be

```
$ cat thesis.txt | todos | lpr
```

Q2. What is a filter?

A2. A filter is a program that reads from standard input (*stdin*), performs some action on this input, and writes to standard output (*stdout*). Filters are used for a lot of things, including text processing.

Q3. What is a *magic* filter?

A3. A magic filter is a filter that performs an action based on a file's type. For example, if the file is a plain, text file, it would simply print the file using the normal methods. If the file is a PostScript file, or any other format, it would print it using another method (ghostscript).

5 Troubleshooting

This section covers some common things that can go wrong with your printing system. Send suggestions for this section to <mwf@engr.uark.edu>.

6 References

This is a section of references on the Linux printing system. I have tried to keep the references section of this HOWTO as focused as possible. If you feel that I have forgotten a significant reference work, please do not hesitate to contact me.

Before you post your question to a USENET group, consider the following:

- Is the printer accepting jobs? (Use *lpc*(8) to verify.)
- Is the answer to your question covered in this HOWTO, or the Printing Setup HOWTO?

If any of the above are true, you may want to think twice before you post your question. And, when you do finally post to a newsgroup, try to include pertinent information. Try not to just say something like, "I'm having trouble with lpr, please help." These types of posts will most definitely be ignored by many. Also try to include the kernel version that you're running, how the error occured, and, if any, the specific error message that the system returned.

Welsh, Matt. *Linux Installation and Getting Started*

an excellent introductory text for the beginning Linux user

Foster, Matt. *Linux Printing Setup HOWTO*

the supplement to this HOWTO; covers topics such as setting up, and configuring the print software

On-Line Manual Pages

- *cat*(1) concatenate and print files
- *dvips*(1) convert a TeX DVI file to PostScript
- *ghostview*(1) view PostScript documents using Ghostscript
- *groff*(1) front-end for the groff document formatting system
- *gs*(1) Ghostscript interpreter/viewer
- *lpc*(8) line printer control program
- *lpd*(8) line printer spooler daemon
- *lpq*(1) spool queue examination program
- *lpr*(1) off-line printer
- *lprm*(1) remove jobs from the line printer spooling queue
- *pr*(1) convert text files for printing
- *tex*(1) text formatting and typesetting

USENET newsgroups

- `comp.os.linux.*` a plethora of information on Linux
- `comp.unix.*` discussions relating to the UNIX operating system

7 Colophon

This HOWTO was written using *vi*, the ultimate editor; the human mind, the ultimate computer; and imagination, the ultimate power tool. The computer of choice was a Gateway2000 i486/50 running the Slackware 2.2 distribution of Linux with version 1.3.4 of the kernel. The original document was written in mark-up format using Linuxdoc-SGML (a hacked version of Tom Gordon's QWERTZ DTD). The HOWTO was then emailed to Greg Hankins, the Linux Documentation Project coordinator. There it was converted to PostScript, TeXinfo, DVI, ASCII, and HTML formats. Finally it was posted on USENET, and placed in the Linux archives found on the SunSITE FTP/WWW servers for easy Internet access.