LPI exam 102 prep, Topic 107: Printing

Junior Level Administration (LPIC-1) topic 107

Skill Level: Intermediate

Ian Shields
Senior Programmer
IBM developerWorks

22 Aug 2006

In this tutorial, Ian Shields continues preparing you to take the Linux Professional Institute® Junior Level Administration (LPIC-1) Exam 102. In this third in a series of nine tutorials, Ian introduces you to printing in Linux®. By the end of this tutorial, you will know how to manage printers, print queues, and user print jobs on a Linux system.

Section 1. Before you start

Learn what these tutorials can teach you and how you can get the most from them.

About this series

The Linux Professional Institute (LPI) certifies Linux system administrators at two levels: *junior level* (also called "certification level 1") and *intermediate level* (also called "certification level 2"). To attain certification level 1, you must pass exams 101 and 102; to attain certification level 2, you must pass exams 201 and 202.

developerWorks offers tutorials to help you prepare for each of the four exams. Each exam covers several topics, and each topic has a corresponding self-study tutorial on developerWorks. For LPI exam 102, the nine topics and corresponding developerWorks tutorials are:

Table 1. LPI exam 102: Tutorials and topics

LPI exam 102 topic	developerWorks tutorial	Tutorial summary
Topic 105	LPI exam 102 prep: Kernel	Learn how to install and maintain Linux kernels and kernel modules.
Topic 106	LPI exam 102 prep: Boot, initialization, shutdown, and runlevels	Learn how to boot a system, set kernel parameters, and shut down or reboot a system.
Topic 107	LPI exam 102 prep: Printing	(This tutorial). Learn how to manage printers, print queues, and user print jobs on a Linux system. See detailed objectives below.
Topic 108	LPI exam 102 prep: Documentation	Coming soon.
Topic 109	LPI exam 102 prep: Shells, scripting, programming, and compiling	Coming soon.
Topic 111	LPI exam 102 prep: Administrative tasks	Coming soon.
Topic 112	LPI exam 102 prep: Networking fundamentals	Coming soon.
Topic 113	LPI exam 102 prep: Networking services	Coming soon.
Topic 114	LPI exam 102 prep: Security	Coming soon.

To pass exams 101 and 102 (and attain certification level 1), you should be able to:

- Work at the Linux command line
- Perform easy maintenance tasks: help out users, add users to a larger system, back up and restore, and shut down and reboot
- Install and configure a workstation (including X) and connect it to a LAN, or connect a stand-alone PC via modem to the Internet

To continue preparing for certification level 1, see the developerWorks tutorials for LPI exams 101 and 102, as well as the entire set of developerWorks LPI tutorials.

The Linux Professional Institute does not endorse any third-party exam preparation material or techniques in particular. For details, please contact info@lpi.org.

About this tutorial

Welcome to "Printing in Linux," the third of nine tutorials designed to prepare you for LPI exam 102. In this tutorial, you learn how to configure printers and manage print jobs in Linux.

This tutorial is organized according to the LPI objectives for this topic. Very roughly, expect more questions on the exam for objectives with higher weight.

Table 2. Printing: Exam objectives covered in this tutorial

LPI exam objective	Objective weight	Objective summary
1.107.2 Manage printers and print queues	Weight 1	Configure and monitor print servers. Manage print queues and troubleshoot general printing problems.
1.107.3 Print files	Weight 1	Add and remove jobs from configured printer queues. Convert text files to PostScript for printing.
1.107.4 Printer installation and configuration	Weight 1	Install and configure local and remote printers, including printer daemons and print filters. Use local and remote printers, including PostScript, non-PostScript and Samba printers.

Prerequisites

To get the most from this tutorial, you should have a basic knowledge of Linux and a working Linux system on which to practice the commands covered in this tutorial.

This tutorial builds on content covered in previous tutorials in this LPI series, so you may want to first review the tutorials for exam 101.

Different versions of a program may format output differently, so your results may not look exactly like the listings and figures in this tutorial.

At the time of writing, the published LPI objectives for this topic are largely directed toward the Common UNIX Printing System (CUPS), with vestiges of the earlier LPD (line printer daemon) and LPRng (the next generation line printer, or LPR) printing systems. Accordingly, this tutorial is directed mainly toward CUPS, with only minor mention of the earlier technologies. For more complete preparation, you should also review other material on LPD and LPRng printing technologies.

Section 2. Manage printers and print queues

This section covers material for topic 1.107.2 for the Junior Level Administration (LPIC-1) exam 102. The topic has a weight of 1.

In this section, learn how to:

- Configure and monitor a print server
- Manage user print queues
- Troubleshoot general printing problems

Introduction

In the early days of computers, printing was done by *line printers*, which printed a line of text at a time using fixed-pitch characters and a single font. To speed up overall system performance, early mainframe computers interleaved work for slow peripherals such as card readers, card punches, and line printers with other work. Thus was born *Simultaneous Peripheral Operation On Line* or *spooling*, a term that is still commonly used when talking about computer printing.

In UNIX® and Linux systems, printing initially used the Berkeley Software Distribution (BSD) printing subsystem, consisting of a line printer daemon (lpd) running as a server, and client commands such as `lpr` to submit jobs for printing. This protocol was later standardized by the IETF as RFC 1179, "Line Printer Daemon Protocol".

System V UNIX also had a printing daemon. It was functionally similar to the Berkeley LPD, but had a different command set. You will frequently see two commands with different options that accomplish the same task. For example, `lpr` from the Berkeley implementation and `lp` from the System V implementation are both use to print files.

With advances in printer technology, it became possible to mix different fonts on a page and to print images as well as words. Variable pitch fonts, and more advanced printing techniques such as kerning and ligatures, all became possible. Several improvements to the basic lpd/lpr approach to printing were devised, such as LPRng, the next generation LPR, and CUPS, the Common UNIX Printing System.

Many printers capable of graphical printing use the Adobe PostScript language. A

PostScript printer has an engine that interprets the commands in a print job and produces finished pages from these commands. PostScript is often used as an intermediate form between an original file, such as a text or image file, and a final form suitable for a particular printer that does not have PostScript capability. Conversion of a print job, such as an ASCII text file or a JPEG image to PostScript, and conversion from PostScript to the final raster form required for a non-PostScript printer is done using *filters*.

The rest of this tutorial focuses on the Common UNIX Printing System (CUPS), which supports the traditional commands as well as newer graphical interfaces to printing functions. CUPS 1.1 is assumed; it includes several features not in earlier versions, such as digest passwords for increased security. Many distributions and desktops provide front-end graphical programs for CUPS, so the material covered here is not exhaustive. This tutorial covers the major concepts, but a particular implementation may differ. Note also that physical installation of your printer is beyond the scope of this tutorial.

Print servers

The CUPS server runs as a daemon process, cupsd under control of a configuration file normally located in /etc/cups/cupsd.conf. The /etc/cups directory also contains other configuration files related to CUPS. It is usually started during system initialization, but may be controlled by the cups script located in /etc/rc.d/init.d or /etc/init.d, according to your distribution. As with most such scripts, you can stop, start, or restart the daemon as shown in Listing 1.

Listing 1. Starting and stopping the cups daemon

```
[root@attic4 ~]# /etc/rc.d/init.d/cups
Usage: cups {start|stop|restart|condrestart|reload|status}
[root@attic4 ~]# /etc/rc.d/init.d/cups stop
Stopping cups:                                             [  OK  ]
[root@attic4 ~]# /etc/rc.d/init.d/cups start
Starting cups:                                             [  OK  ]
[root@attic4 ~]# /etc/rc.d/init.d/cups restart
Stopping cups:                                             [  OK  ]
Starting cups:                                             [  OK  ]
```

The configuration file, /etc/cups/cupsd.conf, contains parameters that you may set to control such things as access to the printing system, whether remote printing is allowed, the location of spool files, and so on. On some systems, a second part describes individual print queues and is usually generated automatically by configuration tools. Listing 2 shows some entries for a default cupsd.conf file. Note that comments start with a # character, so entries that were changed from default would have the leading # character removed. Note also that the spool files are stored by default in the /var/spool filesystem as you would expect from the Filesystem Hierarchy Standard (FHS).

Listing 2. Parts of a default /etc/cups/cupsd.conf

```
#
# RequestRoot: the directory where request files are stored.
# By default "/var/spool/cups".
#

#RequestRoot /var/spool/cups

#
# RemoteRoot: the name of the user assigned to unauthenticated accesses
# from remote systems.  By default "remroot".
#

#RemoteRoot remroot

#
# ServerBin: the root directory for the scheduler executables.
# By default "/usr/lib64/cups".
#

#ServerBin /usr/lib64/cups

#
# ServerRoot: the root directory for the scheduler.
# By default "/etc/cups".
#

#ServerRoot /etc/cups
```

You should also be aware of the /etc/printcap file. This was the name of the configuration file for LPD print servers, and many applications still use it to determine available printers and their properties. It is usually generated automatically in a CUPS system, so you will probably not modify it yourself. However, you may need to check it if you are diagnosing user printing problems. An example is shown in Listing 3.

Listing 3. Automatically generated /etc/printcap

```
# This file was automatically generated by cupsd(8) from the
# /etc/cups/printers.conf file.  All changes to this file
# will be lost.
xerox|Xerox Docuprint C20:rm=localhost.localdomain:rp=xerox:
anyprint|Pick any printer:rm=localhost.localdomain:rp=anyprint:
r220|Epson R220:rm=localhost.localdomain:rp=r220:
```

Each line here has a printer name and printer description as well as the name of the remote machine (rm) and remote printer (rp) on that machine. A traditional /etc/printcap file also describes the printer capabilities.

Finally,CUPS 1.1 introduced a passwd.md5 file. This allows CUPS users to be defined using the `lppasswd` command. The CUPS user ids do not have to be system userids.

Print queues

A *print queue* is a logical entity to which users direct print jobs. Frequently, particularly in single-user systems, a print queue is synonymous with a printer. However, CUPS allows a system without an attached printer to queue print jobs for eventual printing on a remote system, and also, through the use of *classes* to allow a print job directed at a class to be printed on the first available printer of that class. These are discussed more in the final section of this tutorial.

Several commands permit inspection and manipulation of print queues. Some of these have their roots in LPD commands, although the currently supported options are frequently a limited subset of those supported by the original LPD printing system. Other commands are new for CUPS. In general, a user can manipulate his or her own print jobs, but root or another authorized user is usually required to manipulate the jobs of others. Most CUPS commands support a `-E` option for encrypted communication between the CUPS client command and CUPS server.

You can check the queues known to the system using the CUPS `lpstat` command. Some common options are shown in Table 3.

Table 3. Options for lpstat

Option	Purpose
-a	Display accepting status of printers.
-c	Display print classes.
-p	Display print status: enabled or disabled.
-s	Display default printer, printers, and classes. Equivalent to `-d -c -v`. Note that multiple options must be separated as values can be specified for many.
-s	Display printers and their devices.

You may also use the LPD `lpc` command, found in /usr/sbin, with the `status` option. If you do not specify a printer name, all queues are listed. Listing 4 shows some examples of both commands.

Listing 4. Displaying available print queues

```
[ian@attic4 ~]$ lpstat -d
system default destination: xerox
[ian@attic4 ~]$ lpstat -v xerox
device for xerox: lpd://192.168.0.10/PS-66D975-P1
[ian@attic4 ~]$ lpstat -s
system default destination: xerox
members of class anyprint:
        r220
        xerox
device for anyprint: ///dev/null
device for r220: smb://MSHOME/DEN/EPSON220
device for xerox: lpd://192.168.0.10/PS-66D975-P1
[ian@attic4 ~]$ lpstat -a r220
r220 accepting requests since Sat 12 Aug 2006 04:01:38 PM EDT
[ian@attic4 ~]$ /usr/sbin/lpc status xerox
xerox:
        printer is on device 'lpd' speed -1
        queuing is disabled
        printing is enabled
        no entries
        daemon present
```

This example shows two printers, xerox and r220, and a class, anyprint, which allows print jobs to be directed to the first available of these two printers.

In the previous example, queuing of print jobs to xerox is currently disabled, although printing is enabled, as might be done in order to drain the queue before taking the printer offline for maintenance. Whether queuing is enabled or disabled is controlled by the `accept` and `reject` commands. Whether printing is enabled or disabled is controlled by the `cupsenable` and `cupsdisable` commands. In earlier versions of CUPS, these were called `enable` and `disable`, which allowed confusion with the bash shell builtin `enable`. Listing 5 shows how to enable queuing on printer xerox while disabling printing. Note that an authorized user must perform these tasks. This may be root or another authorized user. See the cupsd.conf file and the man pages for the `lppasswd` command for more information on authorizing users.

Listing 5. Enabling queuing and disabling printing

```
[root@attic4 ~]# lpc status xerox
xerox:
        printer is on device 'lpd' speed -1
        queuing is enabled
        printing is enabled
        no entries
        daemon present
[root@attic4 ~]# cupsdisable xerox
[[root@attic4 ~]# lpstat -p -a
printer anyprint is idle.  enabled since Sat 12 Aug 2006 04:07:46 PM EDT
printer r220 is idle.  enabled since Sat 12 Aug 2006 04:01:38 PM EDT
printer xerox disabled since Sat 12 Aug 2006 06:43:09 PM EDT -
        Paused
anyprint accepting requests since Sat 12 Aug 2006 04:07:46 PM EDT
r220 accepting requests since Sat 12 Aug 2006 04:01:38 PM EDT
xerox accepting requests since Sat 12 Aug 2006 06:43:09 PM EDT
```

Managing print jobs on print queues

Now that you have seen a little of how to check on print queues and classes, let's look at how to manage print jobs on printer queues. The first thing you might want to do is find out whether any jobs are queued for a particular printer or for all printers. You do this with the `lpq` command. If no option is specified, `lpq` displays the queue for the default printer. Use the `-P` option with a printer name to specify a particular printer or the `-a` option to specify all printers, as shown in Listing 6.

Listing 6. Checking print queues with lpq

```
[ian@attic4 ~]$ lpq
xerox is not ready
Rank    Owner   Job     File(s)                         Total Size
1st     brendan 14      RobotPlayer.java                1024 bytes
2nd     ian     16      .bashrc                         1024 bytes
3rd     ian     17      .bashrc                         1024 bytes
[ian@attic4 ~]$ lpq
xerox is not ready
Rank    Owner   Job     File(s)                         Total Size
1st     brendan 14      RobotPlayer.java                1024 bytes
2nd     ian     16      .bashrc                         1024 bytes
3rd     ian     17      .bashrc                         1024 bytes
[ian@attic4 ~]$ lpq -P r220
r220 is ready
no entries
[ian@attic4 ~]$ lpq -a
Rank    Owner   Job     File(s)                         Total Size
1st     brendan 14      RobotPlayer.java                1024 bytes
2nd     ian     16      .bashrc                         1024 bytes
3rd     ian     17      .bashrc                         1024 bytes
```

In this example, three jobs, 14, 16, and 17, are queued for the printer named xerox. Note that when the `-P` option is included, the output indicates that the printer is not ready. Note also that user ian submitted a job twice, a common user action when a job does not print the first time. You can avoid printing the extra copy by removing a job from the queue with the `lprm` command. The usual authorization setup allows a user to remove his or her own jobs, but not those of other users. The root user or another authorized user can remove jobs for other users. With no options, the current job is removed. With the `-` option, all jobs are removed. Otherwise, you can give a list of jobs to be removed as shown in Listing 7.

Listing 7. Deleting print jobs with lprm

```
[[ian@attic4 ~]$ lprm
Password for ian on localhost?
lprm: Unauthorized
[ian@attic4 ~]$ lprm 17
[ian@attic4 ~]$ lpq
xerox is not ready
Rank    Owner   Job     File(s)                         Total Size
1st     brendan 14      RobotPlayer.java                1024 bytes
2nd     ian     16      .bashrc                         1024 bytes
```

Note that user ian was not able to remove the first job on the queue, because it was for user brendan. However, he was able to remove his own job number 17.

Another command that will help you manipulate jobs on print queues is the `lp` command. You may use it to alter attributes of jobs, such as priority or number of copies. Suppose user ian wants his job to print before that of user brendan, and he really did want two copies of it. The job priority ranges from a lowest priority of 1 to a highest priority of 100 with a default of 50. User ian could use the `-i`, `-n`, and `-q` options to specify a job to alter and a new number of copies and priority as shown in Listing 8. Note the use of the `-l` option of the `lpq` command, which provides more verbose output.

Listing 8. Changing the number of copies and priority with lp

```
[ian@attic4 ~]$ lp -i 16 -n 2
[ian@attic4 ~]$ lpq -l
xerox is not ready

ian: 1st                                [job 16 localhost]
        2 copies of .bashrc                     1024 bytes

brendan: 2nd                                [job 14 localhost]
        RobotPlayer.java                        1024 bytes
```

Finally, the `lpmove` command allows jobs to be moved from one queue to another. For example, we might want to do this because printer xerox is not currently printing. This command requires an authorized user. Listing 9 shows how to move these jobs to another queue, specifying first by printer and job id, then all jobs for a given printer. By the time we check the queues again, two of the jobs have already printed.

Listing 9. Moving jobs to another print queue with lpmove

```
[root@attic4 ~]# lpmove xerox-16 anyprint
[root@attic4 ~]# lpmove xerox r220
[root@attic4 ~]# lpq
xerox is not ready
no entries
[root@attic4 ~]# lpq -a
Rank    Owner   Job     File(s)                         Total Size
active  ian     18      fig1.gif                        26624 bytes
```

If you happen to use a print server that is not CUPS, such as LPD or LPRng, you will find that many of the queue administration functions that we have just looked at are handled as subcommands of the `lpc` command. For example, you might use `lpc topq` to move a job to the top of a queue. Other `lpc` commands may include `disable`, `down`, `enable`, `hold`, `move`, `redirect`, `release`, and `start`.

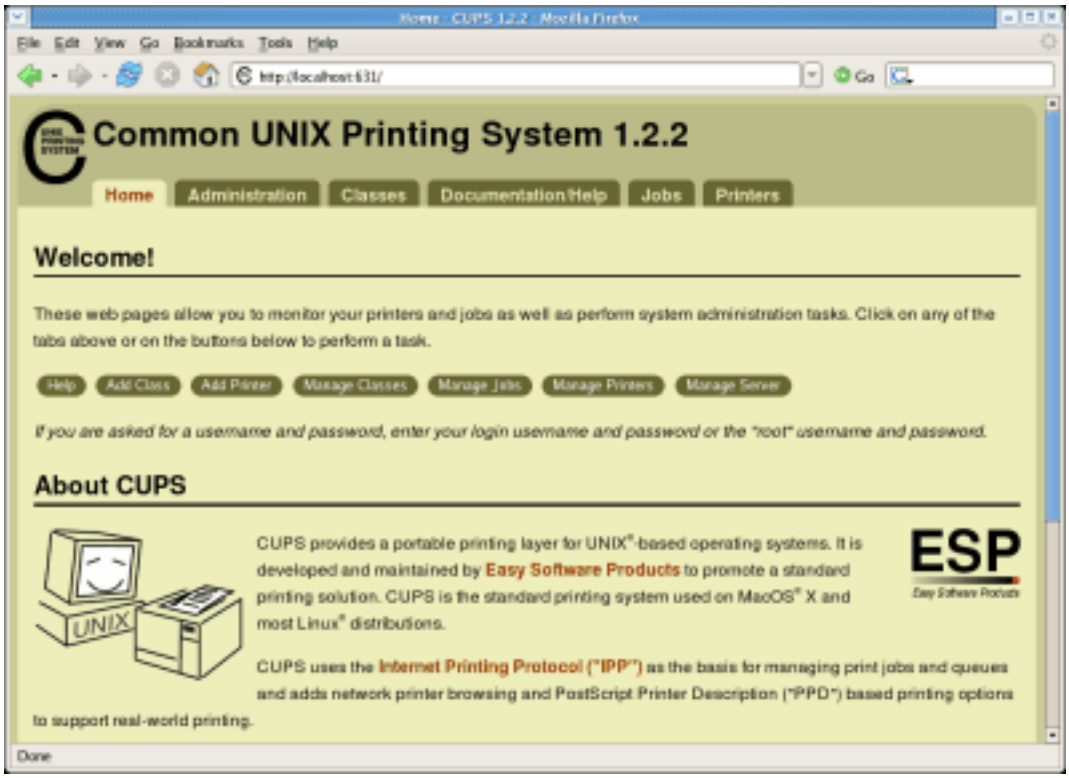
Troubleshooting

If you are having trouble printing, try these tips:

- Ensure that the cups server is running. You can use the `lpstat` command, which will report an error if it is unable to connect to the cupsd daemon. Alternatively, you might use the `ps -ef` command and check for cupsd in the output.
- If you try to queue a job for printing and get an error message indicating that the printer is not accepting jobs results, use `lpstat -a` or `lpc status` to check that the printer is accepting jobs.
- If a queued job does not print, use `lpstat -p` or `lpc status` to check that the printer is accepting jobs. You may need to move the job to another printer as discussed in the next section.
- If the printer is remote, you may need to check that it still exists on the remote system and that it is operational.
- You may need to update the configuration file to allow a particular user or remote system to print on your printer.
- You may need to ensure that your firewall allows remote printing requests, either from another system to your system, or from your system to another, as appropriate.
- You may need to verify that you have the right driver (as discussed in the final section of this tutorial).

As you can see, printing involves the correct functioning of several components of your system and possibly network. In a tutorial of this length, we can only give you starting points for diagnosis. Most CUPS systems also have a graphical interface to the command-line functions that we discuss here. Generally, this interface is accessible from the local host using a browser pointed to port 631 (http://localhost:631 or http://127.0.0.1:631), as shown in Figure 1.

Figure 1. CUPS home page on port 631



Section 3. Print files

This section covers material for topic 1.107.3 for the Junior Level Administration (LPIC-1) exam 102. The topic has a weight of 1.

In this section, learn how to:

- Add and remove jobs from configured printer queues
- Convert text files to PostScript for printing

Print

You learned how to remove files from print queues in the previous section. Here you learn how to print files and change job options.

Many graphical programs provide a method of printing, usually under the **File** menu option. These programs provide graphical tools for choosing a printer, margin sizes, color or black-and-white printing, number of copies, selecting 2-up printing (which is

2 pages per sheet, often used for handouts), and so on. This section shows you the command-line tools for controlling such features, and then a graphical implementation for comparison.

The simplest way to print any file is to use the `lpr` command and provide the file name. This prints the file on the default printer. Listing 10 shows a simple example plus a more complex example. The more complex command is explained below.

Listing 10. Printing with lpr

```
[ian@attic4 ~]$ echo abc>abc.txt
[ian@attic4 ~]$ lpr abc.txt
[ian@attic4 ~]$ lpr -Pxerox -J "Ian's text file" -#2 -m -p -q -r abc.txt
[ian@attic4 ~]$ lpq -l
xerox is ready

ian: 1st                                [job 25 localhost]
        2 copies of Ian's text file             1024 bytes
[ian@attic4 ~]$ ls abc.txt
ls: abc.txt: No such file or directory
```

Table 4 explains the options used on the more complex command above, along with other options that you may use with `lpr`.

Table 4. Options for lpr

Option	Purpose
-C, -J, or -T	Set a job name.
-P	Select a particular printer.
-#	Specify number of copies. Note this is different to the -n option you saw with the `lp` command.
-m	Send email upon job completion.
-l	The print file is already formatted for printing. Equivalent to `-o raw`.
-o	Set a job option.
-p	Format a text file with a shaded header. Equivalent to `-o prettyprint`.
-q	Hold (or queue) the job for later printing.
-r	Remove the file after it

	has been spooled for printing.

So, in our complex example: `lpr -Pxerox -J "Ian's text file" -#2 -m -p -q -r abc.txt`, user ian is requesting a specific printer, giving a name to the job, requesting 2 copies, requesting an email confirmation after printing, holding the job, and having the file abc.txt removed after it has been spooled. The subsequent commands show the held job and the fact that the file has indeed been removed.

In addition to the `lpr` command, the `lp` command covered in the previous section can also be used to print jobs, as well as modify them. Both `lp` and `lpr` also accept a file from stdin if no file name is given on the command line. In contrast to `lpr`, which quietly spools the job, the `lp` default is to display the job number of the spooled job as shown in Listing 11. Note that not all the equivalent options on `lp` and `lpr` have the same name; for example, `-n` on `lp` is equivalent to `-#` on `lpr`.

Listing 11. Printing from stdin with lp

```
[ian@attic4 ~]$ lp
abc
request id is xerox-27 (1 file(s))
```

So we now have a held job in the xerox print queue. What to do? The `lp` command has options to hold and release jobs, using various values with the `-H` option. Listing 12 shows how to release the held job. Check the man page of `lp` for information on other options.

Listing 12. Resuming printing of a held print job

```
[ian@attic4 ~]$ lp -i 25 -H resume
```

Many different printers are available today, but not all of them support the same set of options. You can find out what general options are set for a printer using the `lpoptions` command. Add the `-l` option to display printer-specific options; Listing 13 shows an example. The man page for the `lp` command also lists several common options, particularly relating to portrait/landscape printing, page dimensions, and placement of the output on the pages.

Listing 13. Checking printer options

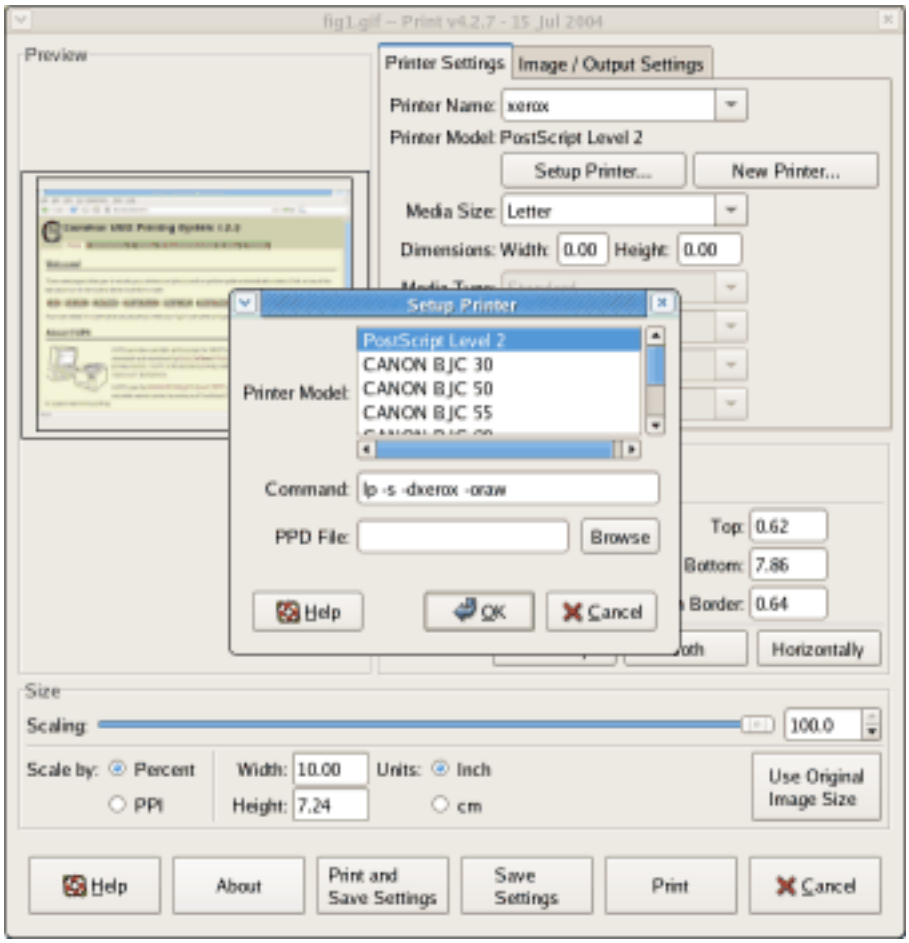
```
[ian@attic4 ~]$ lpoptions -p xerox
job-sheets=none,none printer-info='Xerox Docuprint C20' printer-is-accepting-
jobs=1 printer-is-shared=1 printer-make-and-model='Xerox DocuPrint C20 Foomat
ic/Postscript (recommended)' printer-state=3 printer-state-change-time=115550
6374 printer-state-reasons=none printer-type=143388 cpi=12 scp-fc5=true lpi=7
 page-bottom=86 page-left=57 page-right=57 page-top=72 scaling=100 wrap=true
```

```
[ian@attic4 ~]$ lpoptions -l
PageSize/Page Size: *Letter A4 11x17 A3 A5 B5 Env10 EnvC5 EnvDL EnvISOB5 EnvM
onarch Executive Legal
PageRegion/PageRegion: Letter A4 11x17 A3 A5 B5 Env10 EnvC5 EnvDL EnvISOB5 En
vMonarch Executive Legal
Duplex/Double-Sided Printing: DuplexNoTumble DuplexTumble *None
Resolution/Resolution: *default 150x150dpi 300x300dpi 600x600dpi
PreFilter/GhostScript pre-filtering: EmbedFonts Level1 Level2 *No
```

So far, all our commands have been directed to the local CUPS server. You can also direct most commands to the server on another system, by specifying the `-h` option along with a port number if it is not the CUPS default of 631.

Before moving on to filters, let's look at how all this magic avails itself in a GUI application. Figure 2 shows the illustration of Figure 1 in the GIMP, an image manipulation program. Using the **File > Print** option, you have many choices about how to print the image. In this application, you can also click the **Setup Printer** button to choose a printer and see the command that will be used to print the file, in this case, `lp -s -dxerox -oraw`.

Figure 2. Printing from the GIMP



Convert files

You may have noticed that we were able to print text files above, even though the r220 printer happens to be an Epson photo printer, while the xerox printer is a PostScript Xerox Docuprint C20. This magic feat is accomplished through the use of *filters*. Indeed, a popular filter for many years was named *magicfilter*.

The number of filters included with most CUPS packages allows almost any kind of file to be printed. Additional filters are available commercially, from companies including Easy Software Products, the developers of CUPS.

CUPS uses MIME (Multipurpose Internet Mail Extensions) types to determine the appropriate conversion filter when printing a file. The section on filter installation, later in this tutorial, goes into detail. Other printing packages may use the *magic number* mechanism as used by the `file` command. See the man pages for `file` or `magic` for more details.

The general print flow is to convert the input file to a PostScript format using the appropriate filter for the file type, such as texttops, imagetops, or pdftops. The PostScript format is then filtered through a pstoraster filter to create an intermediate raster format for non-PostScript printers before being filtered through a printer backend, which prepares it for printing on a particular printer. Ghostscript is a popular program that can print PostScript files on many different printers. A companion viewer allows display of the file on a monitor. Many printer backends are derived from Ghostscript printer drivers.

Before all of this was handled so automatically, it was necessary to convert input to PostScript format. Images could be handled with a program such as the GIMP that we saw earlier. ASCII text files were usually converted to PostScript using the `a2ps` command. The default for plain text files is to print 2-up with a header and direct output to the default printer as shown in Listing 14.

Listing 14. Printing text files with a2ps

```
[ian@attic4 ~]$ a2ps -4 abc.txt -o abc.ps
[abc.txt (plain): 1 page on 1 sheet]
[Total: 1 page on 1 sheet] saved into the file `abc.ps'
```

The `a2ps` command can handle a wide range of text file types and make intelligent decisions about the best way to format them. For example, the default for LaTeX files is to first format the file and then print 2-up. Listing 15 uses `a2ps` to print a copy of the sample2e.tex file that is distributed with LaTeX, and then shows a copy renamed to sample2e.txt, and printed 4-up with headers. Both are saved to an output PostScript format file. Figure 3 shows how the output of the second command is formatted.

Listing 15. Saving output from a2ps as a PostScript file

```
[ian@attic4 ~]$ a2ps -4 -E -o fig3.ps sample2e.tex
[sample2e.tex (tex, delegated to texi2dvi): 1 page on 1 sheet]
[Total: 4 pages on 1 sheet] saved into the file `fig3.ps'
[ian@attic4 ~]$ a2ps -4 -E -o fig3.ps sample2e.txt
[sample2e.txt (plain): 4 pages on 1 sheet]
[Total: 4 pages on 1 sheet] saved into the file `fig3.ps'
```

Figure 3. Pretty printed output from a2ps

There are many other filters that you can use to format files for printing in special ways. Most have a range of options. Check the man pages for more details. Some examples are:

mpage
: Formats test files for printing multiple pages on a single page.

psnup
: Performs similar functions for PostScript files as mpage does for text files.

psbook
Rearranges the pages of a PostScript document for printing as a book or booklet, taking into account the number of pages per sheet and how the sheet is folded.

Section 4. Printer installation and configuration

This section covers material for topic 1.107.4 for the Junior Level Administration (LPIC-1) exam 102. The topic has a weight of 1.

In this section, learn how to:

- Install a printer daemon
- Install and configure a print filter
- Access local and remote printers of various kinds

Printer daemons

Install a printer daemon by first installing the printer package, either CUPS, or another such as LPRng, which is usually shipped with a distribution. If you require one that is not shipped with your distribution, you may find a package prebuilt for your distribution, or you may build it yourself from source. Refer to the tutorial for Exam 101 topic 102, " LPI exam 101 prep: Linux installation and package management," if you need help with this task.

Once the printer package is installed, ensure that the printer daemon starts when your system starts. This is covered in the tutorial for Exam 102 topic 106 "LPI exam 102 prep: Boot, initialization, shutdown, and runlevels."

Use the `cups` script located in /etc/rc.d/init.d or /etc/init.d, according to your distribution with the `status` command. You can also use the `lpstat` or `lpc status` command to check whether your daemon is running. If you are using a different printer daemon, use the appropriate script for your package. An example is shown in Listing 16.

Listing 16. Checking CUPS daemon status

```
[root@attic4 ~]# /etc/init.d/cups stop
Stopping cups:                                             [  OK  ]
[root@attic4 ~]# /etc/init.d/cups status
```

```
cupsd is stopped
[root@attic4 ~]# lpstat -d
lpstat: Unable to connect to server
[root@attic4 ~]# /etc/init.d/cups start
Starting cups:                                             [  OK  ]
```

If you ever need to debug CUPS, you can run it in the foreground rather than as a daemon process. You can also test alternate configuration files if necessary. Run `cupsd -h` for more information, or see the man pages.

Listing 17. Running cupsd from the command line

```
[root@attic4 ~]# cupsd -h
Usage: cupsd [-c config-file] [-f] [-F] [-h] [-l] [--ppdsdat]

-c config-file      Load alternate configuration file
-f                  Run in the foreground
-F                  Run in the foreground but detach
-h                  Show this usage message
-l                  Run cupsd from launchd(8)
--ppdsdat           Just build ppds.dat
```

CUPS also maintains an access log and an error log. You can change the level of logging using the LogLevel statement in /etc/cups/cupsd.conf. By default, logs are stored in the /var/log/cups directory. They may be viewed from the **Administration** tab on the Web interface (http://localhost:631).

Print filters

So how does CUPS determine the filter to use for formatting a particular file type? CUPS uses MIME (Multipurpose Internet Mail Extensions) types to determine the appropriate conversion filter when printing a file. Note that other printing packages may use the *magic number* mechanism as used by the `file` command. See the man pages for `file` or `magic` for more details.

MIME types are used for transmitting various files as mail attachments. They consist of a type, such as text or image, and a subtype, such as html, postscript gif, or jpeg. The type and subtype are separated by a semicolon (;). Optional parameters may include information such as character set encoding, or language. CUPS uses rules from /etc/cups/mime.types to determine the type of a file and then uses a suitable filter chosen from those listed in /etc/cups/conv.types for the given MIME type. MIME types are registered with IANA, the Internet Assigned Numbers Authority. If you need a type that is not registered, prefix the subtype with 'x-'. Some image type examples are shown in Listing 18.

Listing 18. Some MIME type entries from /etc/cups/mime.types

```
image/gif                   gif string(0,GIF87a) string(0,GIF89a)
image/png                   png string(0,<89>PNG)
image/jpeg                  jpeg jpg jpe string(0,<FFD8FF>) &&\
                            (char(3,0xe0) char(3,0xe1) char(3,0xe2) char(3,0xe3)\
                             char(3,0xe4) char(3,0xe5) char(3,0xe6) char(3,0xe7)\
                             char(3,0xe8) char(3,0xe9) char(3,0xea) char(3,0xeb)\
                             char(3,0xec) char(3,0xed) char(3,0xee) char(3,0xef))
image/tiff                  tiff tif string(0,MM) string(0,II)
image/x-photocd             pcd string(2048,PCD_IPI)
image/x-portable-anymap     pnm
```

The format of the entries is beyond the scope of this tutorial. Check the files /usr/share/mime/magic or /usr/share/file/magic for some insight on how the *magic numbers* are used to identify files.

One the MIME type of a file has been determined, the correct filter is found using the /etc/cups/mime.convs file. Lines in this file have four entries, a source and destination MIME type, a *cost*, and the name of the filter. The least-cost filter is used. Some examples are shown in Listing 19.

Listing 19. Filter entries from /etc/cups/mime.convs

```
text/plain              application/postscript  33      texttops
text/html               application/postscript  33      texttops
image/gif               application/vnd.cups-postscript 66      imagetops
image/png               application/vnd.cups-postscript 66      imagetops
image/jpeg              application/vnd.cups-postscript 66      imagetops
image/tiff              application/vnd.cups-postscript 66      imagetops
image/x-bitmap          application/vnd.cups-postscript 66      imagetops
```

If a suitable filter cannot be found, your attempt to print a file will result in an error message. If you are using a printer daemon other than CUPS, you may get unexpected output instead. Listing 20 shows how this works with a DVI file (the normal output from TeX and LaTex).

Listing 20. Printing an unsupported file type

```
[ian@attic4 ~]$ lpr samp1.dvi
lpr: Unsupported format 'application/octet-stream'!
```

Fortunately, the tetex package that provides TeX and LaTeX also provides a conversion utility, `dvips` to convert from DVI to PostScript. Unfortunately, it won't work as a filter because it does not know how to handle the arguments that a CUPS filter must handle, namely, job id, user, job title, number of copies, and job options. The first filter in a filter pipeline will also have an additional parameter, the filename, if the input comes from a file.

The solution is to create a wrapper script that will be the filter. The `dvips` command does not accept input from stdin, so the script may need to create a temporary file and copy stdin to that file before calling `dvips`. A possible script is shown in Listing

21.

Listing 21. A CUPS DVI to PostScript filter script

```
#!/bin/bash
# CUPS filter to process DVI files using dvips
# Create a sandbox for working if input on stdin
if [ $# -lt 6  ]; then
    sandbox=${TMPDIR-/tmp}/cups-dvitops.$$
    (umask 077 && mkdir $sandbox) || {
        echo "Cannot create temporary directory! Exiting." 1>&2
        exit 1
    }
    fn="$sandbox/cups-dvitops.$$"
    cat > "$fn"
else
    fn="$6"
fi
# Call dvips quietly, securely and with output to stdout
dvips -R -q -o - "$fn"
# Erase sandbox if we created one
if [ $# -lt 6  ]; then
    rm -rf  "$sandbox"
fi
```

Recall that CUPS uses two files in /etc/cups to determine the MIME type and filter to use. These files will be overwritten whenever you reinstall or upgrade CUPS. Fortunately, CUPS will read **all** files with an extension of .types or .convs whenever it starts or restarts. So you should create a pair of files for your new filter, for example /etc/cups/dvitops.types and /etc/cups/dvitops.convs as shown in Listing 22 shows a partial output listing for Docuprint drivers.

Listing 22. Configuration files for CUPS dvitops filter

```
[ian@attic4 ~]$ cat /etc/cups/dvitops.types
# Local MIME definition for DVI files
application/x-dvi dvi string(0,<F702>)
[ian@attic4 ~]$ cat /etc/cups/dvitops.convs
# Local DVI to PostScript filter for CUPS
application/x-dvi application/postscript 50 dvitops
```

This says that DVI files are identified by having the hexadecimal digits F7 and 02 in the first two positions and that such files should be processed by the dvitops filter.

Next, as root, copy the script above in /usr/lib/cups/filter/dvitops and make sure it is world readable and executable.(-rwxr-xr-x). The name you give the script must match that in the /etc/cups/dvitops.convs file above. If you are running SELinux in enforcing mode, you should also run `restorecon` in the /usr/lib/cups/filter directory to update the security contexts. Otherwise, your `lpr` command will appear to work, but your file will not print.

Finally, use the restart option with the cups script located in /etc/rc.d/init.d or /etc/init.d, to restart CUPS and use your new filter.

If you are using an older print spooler, you will probably use either the magicfilter or apsfilter as input filters to convert various input files to PostScript format for printing to a PostScript printer or, using Ghostscript, to a non-PostScript printer.

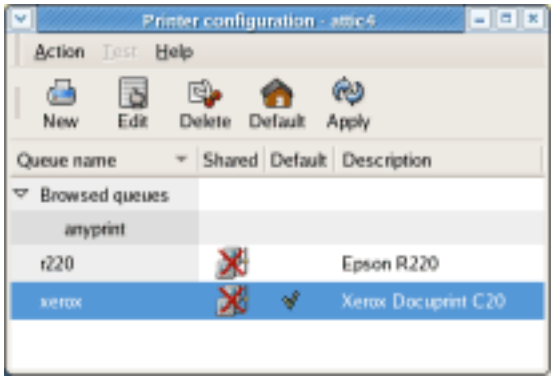
Accessing printers

CUPS supports a variety of printers, including:

- Locally attached parallel and USB printers
- IPP (Internet Printing Protocol) printers
- Remote LPD printers
- Windows® printers using SAMBA
- Novell printers using NCP
- HP JetDirect attached printers

Most systems now attempt to autodetect and autoconfigure local hardware when the system starts or when the device is attached. Similarly, many network printers can be autodetected. You can use the CUPS Web administration tool ((http://localhost:631 or http://127.0.0.1:631) to search for or add printers. Many distributions include their own configuration tools, for example YaST on SUSE systems. Figure 4 illustrates the system-config-printer tool on Fedora Core 5.

Figure 4. Using system-config-printer on Fedora Core 5



You can also configure printers from a command line. The rest of this tutorial shows you how. An understanding of this material will help you answer exam questions on the GUI interfaces.

Before you configure a printer, you need some basic information about the printer and about how it is connected. For illustration, we will use a Xerox Docuprint C20 attached through a D-Link print server. The print server provides LPD printing function. To configure it, we will need the IP address (192.168.0.10, in this case) and

a printer queue name on the LPD server. This is set in the print server and is PS-66D975-P1 in this case. If a remote system needs a user id or password, you will also need that information.

You will also need to know what driver to use for your printer. Check at LinuxPrinting.org (see Resources later in this tutorial for a link) to see if there is a driver for your particular printer. The `lpinfo` command can also help you identify available device types and drivers. Use the `-v` option to list supported devices and the `-m` option to list drivers, as shown in Listing 23.

Listing 23. Available printer drivers

```
lyrebird:~ # lpinfo -m | grep -i "docuprint.c"
Xerox/DocuPrint_C6-cdj550.ppd.gz Xerox DocuPrint C6 Foomatic/cdj550 (recommended)
Xerox/DocuPrint_C8-cdj550.ppd.gz Xerox DocuPrint C8 Foomatic/cdj550 (recommended)
Xerox/DocuPrint_C11-cdj500.ppd.gz Xerox DocuPrint C11 Foomatic/cdj500
Xerox/DocuPrint_C11-hpdj.ppd.gz Xerox DocuPrint C11 Foomatic/hpdj
Xerox/DocuPrint_C11-pcl3.ppd.gz Xerox DocuPrint C11 Foomatic/pcl3 (recommended)
Xerox/DocuPrint_C20-cljet5.ppd.gz Xerox DocuPrint C20 Foomatic/cljet5
Xerox/DocuPrint_C20-hpijs.ppd.gz Xerox DocuPrint C20 Foomatic/hpijs
Xerox/DocuPrint_C20-Postscript.ppd.gz Xerox DocuPrint C20 Foomatic/Postscript
(recommended)
Xerox/DocuPrint_C55-Postscript.ppd.gz Xerox DocuPrint C55 Foomatic/Postscript
(recommended)
```

Several choices are shown here for the Docuprint C20. The recommended driver is the PostScript driver, which is not surprising, since this printer supports PostScript. Again, if you can't find your printer listed, check at LinuxPrinting.org (see Resources) for the appropriate driver. Drivers come in the form of PPD (PostScript Printer Description) files.

Now that you have the basic information, you can configure a printer using the `lpadmin` command as shown in Listing 24. Note that this system did not list the specific Xerox Docuprint driver, so we use the generic PostScript driver instead.

Listing 24. Configuring a printer

```
[root@attic4 ~]# lpinfo -m | grep -i generic
textonly.ppd Generic text-only printer
postscript.ppd.gz Generic postscript printer
[root@attic4 ~]# lpadmin -p xerox1 -E -m "postscript.ppd.gz" \
> -v "lpd:192.168.0.1/PS-66D975-P1" -D "Xerox 1"
[root@attic4 ~]# lpstat -a
anyprint accepting requests since Sat 12 Aug 2006 04:07:46 PM EDT
r220 accepting requests since Tue 22 Aug 2006 11:13:40 AM EDT
xerox accepting requests since Tue 22 Aug 2006 11:13:40 AM EDT
xerox1 accepting requests since Tue 22 Aug 2006 11:17:59 AM EDT
```

If you need to remove a printer, use `lpadmin` with the `-x` option as shown in Listing 25.

Listing 25. Removing a printer

```
[root@attic4 ~]# lpadmin -x xerox1
```

You can also set various printer options using the `lpadmin` or `lpoptions` commands.

Spool files

CUPS uses the /var/spool/cups directory for spooling. This will usually be set up correctly when you install CUPS. If you are using the LPD daemon, spool files are stored in directories such as /var/spool/lpd/xerox, for our printer 'xerox'. Spool directories and files should have permissions set to protect them from being read or written by users other than the printing system.

Other input filters

If you are using LPD, LPRng, or another printing system, you will probably use either magicfilter or apsfilter for converting input files to PostScript format, and you will probably use Ghostscript as the printer driver for non-PostScript printers. Configuration for printers and filters will be in /etc/printcap. If you are using these, consult the man pages or online documentation such as the *Apsfilter handbook* listed in Resources later in this tutorial.

Resources

Learn

- Review the entire LPI exam prep tutorial series on developerWorks to learn Linux fundamentals and prepare for system administrator certification.
- At the LPIC Program, find task lists, sample questions, and detailed objectives for the Linux Professional Institute's Linux system administration certification.
- In "Basic tasks for new Linux developers" (developerWorks, March 2005), learn how to open a terminal window or shell prompt and much more.
- The Linux Documentation Project has a variety of useful documents for system administrators, especially its HOWTOs.
- LinuxPrinting.org provides information on drivers and printer support as well as a CUPS Quick Start.
- The Apsfilter handbook can help you configure printers and filters if you are using LPD, LPRng, or another printing system.
- The Printer Working Group of the IEEE Industry Standards and Technology Organization (IEEE-ISTO) develops standards that make printers -- and the applications and operating systems supporting them -- work better together.
- *CUPS: Common UNIX Printing System* (Sams, 2001) is a detailed reference for CUPS.
- *LPI Linux Certification in a Nutshell, Second Edition* (O'Reilly, 2006) and *LPIC I Exam Cram 2: Linux Professional Institute Certification Exams 101 and 102 (Exam Cram 2)* (Que, 2004) are LPI references for readers who prefer book format.
- Find more tutorials for Linux developers in the developerWorks Linux zone.
- Stay current with developerWorks technical events and Webcasts.

Get products and technologies

- Order the SEK for Linux, a two-DVD set containing the latest IBM trial software for Linux from DB2®, Lotus®, Rational®, Tivoli®, and WebSphere®.
- Download IBM trial software directly from developerWorks.

Discuss

- Participate in the discussion forum for this content.
- Read developerWorks blogs, and get involved in the developerWorks community.

About the author

Ian Shields
Ian Shields works on a multitude of Linux projects for the developerWorks Linux zone. He is a Senior Programmer at IBM at the Research Triangle Park, NC. He joined IBM in Canberra, Australia, as a Systems Engineer in 1973, and has since worked on communications systems and pervasive computing in Montreal, Canada, and RTP, NC. He has several patents. His undergraduate degree is in pure mathematics and philosophy from the Australian National University. He has an M.S. and Ph.D. in computer science from North Carolina State University.

Trademarks