The FreeBSD Corporate Networker's Guide

Ted Mittelstaedt

The FreeBSD Corporate Networker's Guide
by Ted Mittelstaedt

Table of Contents

List of Figures

Chapter 8 Printserving

Printserving is a complicated topic. There are many different software interfaces to printers, as well as a wide variety of printer hardware interfaces. This chapter covers the basics of setting up a print queue, using Samba to print, and administering print queues and connections.

8.1 PC printing history

In the early days of the personal computer, printing was simple. The PC owner bought a cheap printer, usually a dot matrix that barely supported ASCII, and plugged it into the computer with a parallel cable. Applications would either work with the printer or not, and most did because all they could do was output DOS or ASCII text. The few software applications that supported graphics generally could only output on specific makes and models of printers. Shared *network* printing, if it existed, was usually done by some type of serial port switchbox.

This was the general state of affairs with the PC until the Windows operating system was released. All at once, application programmers were finally free of the restrictions of worrying about how some printer manufacturer would change printer control codes. Graphics printing, in the form of fonts and images, was added to most applications, and demand for it rapidly increased across the corporation. Large, high-capacity laser printers designed for office printing appeared on the scene. Printing went from 150 to 300 to 600 dpi for the common desktop laser printer.

Today organizational network printing is complex, and printers themselves are more complicated. Most organizations find that sharing a few high-quality laser printers is much more cost effective than buying many cheaper dot matrix units. Good network print serving is a necessity, and it can be very well provided by the FreeBSD UNIX system.

8.2 Printer communication protocols and hardware

Printers that don't use proprietary vendor codes communicate with computers using one or more of three major printing protocols. The communication is done over a hardware cable that can be a parallel connection (printer port) or a serial connection (COM port).

8.2.1 ASCII Printing Protocol

The ASCII protocol is the simplest protocol used, as well as the oldest. ASCII is also used to represent text files internally in the DOS, UNIX, and Windows operating systems. Therefore, data taken from a text file or a directory listing generally requires little preparation before being sent to the printer, other than a newline-to-carriage return/linefeed conversion for UNIX. Printers usually follow the DOS text file convention of the print head requiring an explicit carriage return character followed by a linefeed character at the end of a line of text. Since UNIX uses only the linefeed character to terminate text, an additional carriage return character must be added to the end of each line in raw text print output; otherwise, text prints in a *stairstep* output. (Some printers have hardware or software switches to do the conversion.)

8.2.2 PostScript Printing Protocol

Adobe introduced the PostScript language in 1985; it is used to enable the printout of high quality graphics and styled font text. PostScript is now the de-facto print standard in the UNIX community, and the only print standard in the Macintosh community. Numerous UNIX utilities exist to *beautify* and enhance text printing with PostScript.

PostScript can be used to download font files into a printer as well as the data to be printed. PostScript commands can be sent to instruct the printer CPU to image, rotate, and scale complex graphics and images, thus freeing the host CPU. Scaling is particularly important with fonts since the document with the font has been produced on a computer screen with far lower resolution than the printer. For example, a 1024x768 computer screen on a 17-inch monitor allows for a resolution of approximately 82dpi, a modern desktop printer prints at a resolution of 600dpi. Therefore, a font must be scaled at least seven times larger for WYSIWYG output!

PostScript printers generally come with a number of resident fonts. For example, the NEC Silentwriter 95 contains Courier, Helvetica, ITC Avant Garde Gothic Book, ITC Bookman Light, New Century Schoolbook Roman, Palatino Roman, Times Roman, and several symbol fonts. These are stored in Read Only Memory (ROM) in the printer. When a page is printed from a Windows client that contains a font not in the printer, a font substitution table is used. If no substitute can be made, Courier is usually used. The user should be conscious of this when creating documents - documents with fonts not listed in the substitution table may cause other users problems when printing. Avoid use of strange fonts for documents that will be widely distributed.

The user program can choose to download different fonts as outline fonts to the PostScript printer if desired. Fonts that are commonly used by the user are often downloaded to PostScript printers that are connected directly to the user's computer, the fonts are then available to successive print jobs until the printer is turned off. When PostScript printers are networked, the clients must download any fonts desired *with each print job*. Since jobs come from different clients, the clients cannot assume that downloaded fonts will still be in the printer.

PostScript print jobs also contain a header that is sent describing the page layout, among other things. On a shared network printer, this header must also be downloaded with each print job. Although some PostScript drivers allow downloading of the header only once, this usually requires a bi-directional serial connection to the printer, instead of a unidirectional parallel connection.

PostScript print jobs can be sent either as binary data or as ASCII. The main advantage of binary data transmission is that it is faster. However, not all PostScript printers support it. Also, fonts can generally not be downloaded in binary. When FreeBSD is used as a printserver, ASCII PostScript printing should be selected on the clients, this is generally the default with most PostScript drivers.

The Adobe company licenses PostScript interpreters as well as resident fonts to printer manufacturers, and extracts a hefty license fee from any printer manufacturer who wants to use them in its printer. This presents both a benefit and a problem to the end user. Although a single company holding control over a standard can guarantee compliance, it does significantly raise the cost of the printer. As a result, PostScript has not met with much success in the lower-end laser and inkjet Windows printing market, despite the fact that Adobe distributes PostScript software operating system drivers for free.

One issue that is a concern when networking PostScript printers is the selection of banner page, (also known as header page, or *burst page)* printing. UNIX shared printing began with ASCII line printers, and since UNIX is a multiuser system, often many different user print jobs piled up in the printer output hopper. To separate these jobs the UNIX printing system programs support banner page printing if the client program that submits jobs asks for them. These pages print at the beginning or end of every print job and contain the username, submittal date, and so on.. By default, most clients, whether remote (e.g., a Windows LPR client) or local (e.g., the `/usr/bin/lpr` program) trigger a banner page to be printed. One problem is that some PostScript printers abort the entire job if they get unformatted ASCII text instead of PostScript. (In general, PostScript printers compatible with Hewlett-Packard Printer Control Language [HPPCL] handle banners without problems) Banner printing should be disabled for any printers with this problem, unless PostScript banner page printing is set up on the server.

8.2.3 HPPCL Printing Protocol

The Hewlett Packard company currently holds the largest market share of desktop inkjet and office laser printers. Back when Windows was released, HP decided to expand into the desktop laser jet market with the first LaserJet series of printers. At the time there was much pressure on Microsoft to use Adobe Type Manager for scaleable fonts within Windows, and to print PostScript to higher-end printers. Microsoft decided against doing this and used a technically inferior font standard, Truetype. They thought that it would be unlikely that the user would download fonts to the printer, since desktop Publishing was not being done on PC's at the time. Instead users would rasterize the entire page to the printer using whatever proprietary graphics printer codes the selected printer needed. HP devised HPPCL for their LaserJets, and make PostScript an add-on. The current revision of HPPCL now allows for many of the same scaling and font download commands that PostScript does. HP laser jet printers that support PostScript can be distinguished by the letter "M" in their model number. (M is for Macintosh, since Macintosh requires PostScript to print) For example, the HP 6MP has PostScript, the 6P doesn't.

HPPCL has almost no support in the UNIX applications market, and it is very unlikely that any will appear soon. One big reason is the development of the free **Ghostscript** PostScript interpreter. **Ghostscript** can take a PostScript input stream and print it on a PCL printer under UNIX. Another reason is the UNIX community's dislike of reinventing the wheel. HPPCL has no advantage over PostScript, and in many ways there are fewer problems with PostScript. Considering that PostScript can be added to a printer, either by hardware or use of **Ghostscript**, what is the point of exchanging an existing working solution for a slightly technically inferior one? Over the life of the printer, taking into account the costs of toner, paper, and maintenance, the initial higher cost of PostScript support is infinitesimal.

8.3 Network Printing Basics

The most common network printing implementation is a printserver accepting print jobs from clients tied to the server via a network cable.

8.3.1 Printservers

The term "printserver" is one of those networking terms, like *packet,* that has been carelessly tossed around until its meaning has become somewhat confusing and blurred. To be specific, a printserver is simply a program that arbitrates print data from multiple clients for a single printer. Printservers can be implemented in one of the four methods described in the following sections.

8.3.1.1 Printserver on the fileserver

The printer can be physically cabled to the PC running the Network OS. Print jobs are submitted by clients to the printserver software on the fileserver, which sends them down the parallel or serial cable to the printer. The printer must be physically close to the fileserver. This kind of printserving is popular in smaller workgroup networks, in smaller offices.

Figure 8-1. Printserver on the fileserver
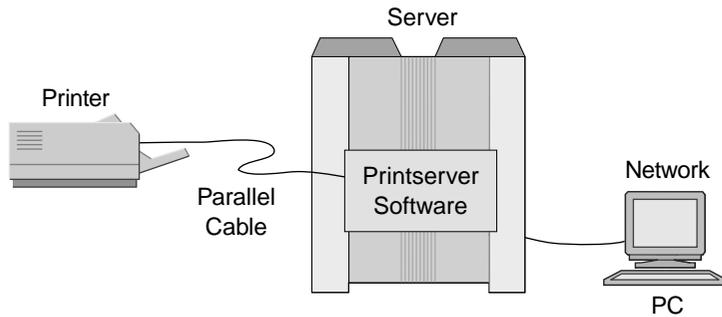
8.3.1.2 Printserver on a separate PC

It is possible to run a print server program on a cheap PC that is located next to the printer and plugged into it via parallel cable. This program simply acts as a pass-through program, taking network packets from the network interface and passing them to the printer. This kind of server doesn't allow any manipulation of print jobs, jobs usually come from a central fileserver, where jobs are controlled.

Figure 8-2. Printserver on a separate PC
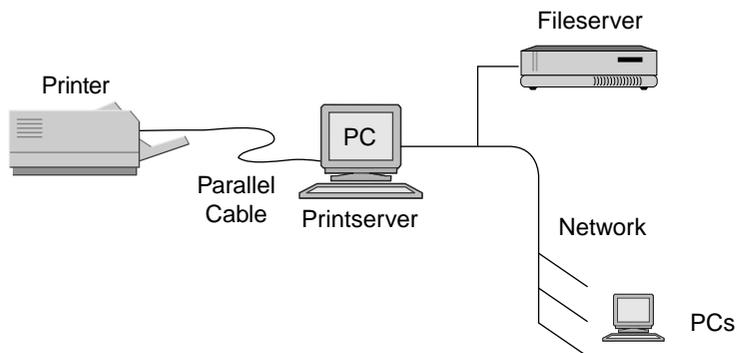
8.3.1.3 Printserver on a separate hardware box

A printserver on a separate hardware box is exemplified by network devices such as the Intel Netport, the HP JetDirect Ex, the Osicom/DPI NETPrint, and the Lexmark MarkNet. Basically, these are plastic boxes with an Ethernet connection on one side and a parallel port on the other. Like a printserver on a PC, these devices don't allow remote job manipulation, and merely pass packets from the network down the parallel port to the printer.

Figure 8-3. Printserver on a separate hardware box
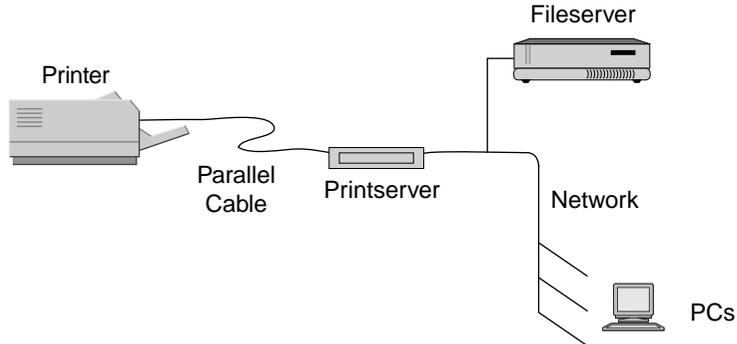
8.3.1.4 Printserver in the Printer

The HP JetDirect Internal is the best known printserver of this type. It is inserted into a slot in the printer case, and it works identically to the external JetDirect units.

Figure 8-4. Printserver in the printer
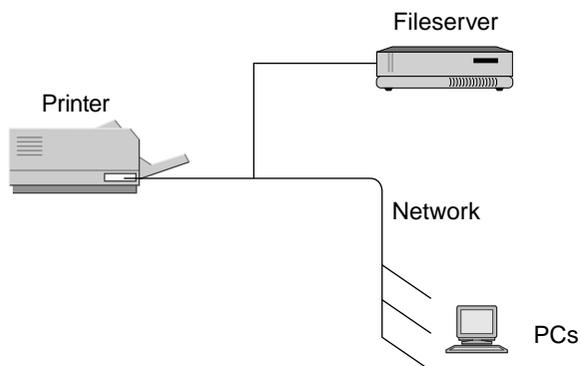
8.3.2 Printspools

Printspooling is an integral part of network printing. Since the PC can spit out data much faster than the printer can accept it, the data must be buffered in a spool at some location. In addition, because many clients share printers, when clients send print jobs at the same time, jobs must be placed on a queue so that one can be printed after the other.

8.3.2.1 Logical location of the print spool

Printspooling can be implemented at one of three locations

1. The client. Clients can be required to spool their own print jobs on their own disks. For example, when a Windows client application generates a print job the job must be placed on the local client's hard drive. Once the remote print server is free to accept the job it signals the client to start sending the job a bit at a time. Client

spooling is popular in peer-to-peer networks with no defined central fileserver. However, it is impossible for a central administrator to perform advanced print job management tasks such as moving a particular print job ahead of another, or deleting jobs.

2. The printserver. If each printer on the network is allocated their own combination print spooler-printserver, jobs can stack at the printer. Many of the larger printers with internal printservers have internal hard disks for this purpose. Although this enables basic job management, it still restricts the ability to move jobs from one printer to another.

3. *A central print spooler on a fileserver*. Print jobs are received from all clients on the network in the spool and then dispatched to the appropriate printer. This scheme is the best for locations with several busy printers and many clients. Administration is extremely simple because all print jobs are spooled on a central server, which is particularly important in bigger organizations. Many large organizations have standardized on PostScript printing for all printing; in the event that a particular printer fails and is offline, incoming PostScript print jobs can be rerouted automatically to another printer. Since all printers and clients are using PostScript, clients don't need to be reconfigured when this happens. Print jobs appear the same whether printed on a 4 page-per-minute NEC Silentwriter 95, or a 24 page-per-minute HP LaserJet 5SiMX if both printers are defined in the client as PostScript printers.

Figure 8-5. Print spool locations
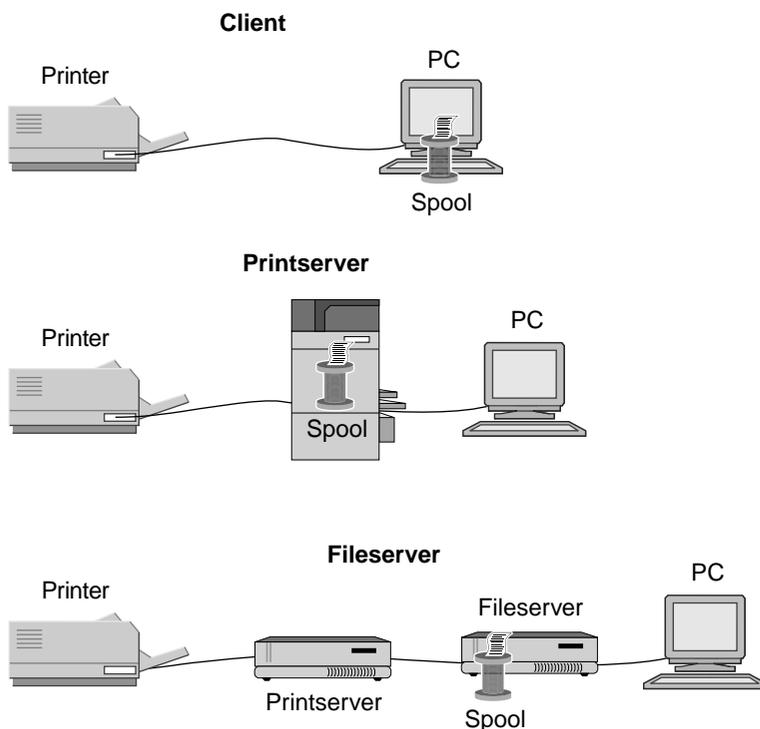
FreeBSD is an excellent platform to implement centralized printserving and print spooling. The rest of this chapter concentrates on the centralized print spooler model. Note that PostScript printing is not a requirement for this model--the HPPCL protocol can be the standard print protocol as well. For transparent printing between printers with HPPCL, however, the printer models must be similar.

8.3.2.2 Physical location of the print spool

In some companies, the central fileserver is often placed in a closet, locked away. Printers, on the other hand, are best located in high traffic areas for ease of use. Network printing works best when the printers are evenly distributed throughout the organization. Attempting to place all the major printers in one location, as technically advantageous as it may seem, merely provokes users to requisition smaller printers that are more convenient for that quick print job. The administrator may end up with a datacenter full of nice, expensive printers that are never used, while the smaller personal laser printers scattered throughout the plant bear most of the printing load.

The big problem with this is that scattering printers through the organization makes it difficult to utilize the 3 possible parallel ports on the fileserver due to parallel port distance limitations. Although high-speed serial ports may extend the distance, not many printers have good serial ports on them. This is where the hardware network print server devices can come into play. I prefer using these devices because they are much cheaper and more reliable than a standalone PC running printserver software. For example, Castelle http://www.castelle.com/ sells the LANpress 1P/10BT printserver for about $170.00. Using these devices a FreeBSD UNIX server can have dozens of print spools accepting print jobs and then route them back out over the network to these remote printserver boxes. If these hardware servers are used, they must support the Line Printer Daemon (LPD) print protocol.

With a scheme like this it is important to have enough disk space on the spool to handle the print jobs. A single large PowerPoint presentation PostScript print job containing many graphics may be over 100MB. When many such jobs stack up in the print spool waiting to print, the print spooler should have several gigabytes of free disk space available.

8.3.2.3 Network Printing to Remote Spools

Although several proprietary network printing protocols such as Banyan Vines and NetWare, are tied to proprietary network protocols, FreeBSD UNIX can use two TCP/IP network printing protocols to print to remote print spools. The two print protocols available on TCP/IP with FreeBSD are the open LPD protocol and the NetBIOS-over-TCP/IP Server Messaging Block (SMB) print protocol first defined by Intel and Microsoft and later used by IBM and Microsoft.

The LPD protocol is defined in RFC1179. This network protocol is the standard print protocol used on all UNIX systems. LPD client implementations exist for all Windows operating systems and DOS. Microsoft has written LPD for the Windows NT versions, the other Windows operating system implementations are provided by third parties.

The Microsoft Networking network protocol that runs on top of SMB can use NetBIOS over TCP/IP as defined in RFC1001 and RFC1002. This protocol has a specification for printing that is the same print protocol used to send print jobs to NT Server by Microsoft clients. To implement this protocol on FreeBSD requires the installation of the Samba client suite of programs discussed in Chapter 7.

8.4 Setting up LPR on Windows clients

The program clients use to print via LPD is the Line Printer Remote, or LPR program. The following instructions cover enabling this program on Windows clients.

8.4.1 Windows 3.1/Windows for Workgroups 3.11

Several commercial TCP/IP stacks are available for Win31, that provide LPR client programs, in addition to the basic TCP/IP protocol to Win31. WfW has TCP/IP networking available for free from Microsoft, but it doesn't include an LPR client. Unfortunately, I have not come across a freeware implementation of a 16-bit Windows LPR client, so with the following instructions I use the Shareware program WLPRSPL available from http://www.winsite.com/info/pc/win3/winsock/wlprs41.zip. This program must be active during client printing, and is usually placed in the Startup group.

Organizations that want to use UNIX as a printserver to a group of Win31 clients without using a commercial or shareware LPR program have another option. The Microsoft Networking client for DOS used underneath Win31 contains SMB-based printing which is covered later in the chapter. DOS networking client setup and use are covered in Chapter 2 and Chapter 7.

If LPR-based client printing is desired and the organization doesn't want to upgrade to Win95, (which has several LPR clients available) the following instructions can be used. WLPRSPL needs a Winsock under Windows 3.1, so for the example I explain the setup of the Novell 16-bit TCP/IP client. The stack can be FTPed from Novell, and is easy to integrate into sites that already use the 16-bit NetWare networking client, usually NW 3.11 and 3.12. In most cases, however, sites that use NetWare + Win31 are probably best off printing through the NetWare server, then loading an LPR spooler as an Netware Loadable Module (NLM) to send the job over to FreeBSD.

As an alternate, the Microsoft Networking DOS 16-bit TCP/IP client under Win31 contains a Winsock, as does Microsoft TCP/IP for WfW. The target machine used here is a Compaq Deskpro 386/33 with 12MB of ram with an operating version of Windows 3.1, and a 3com 3C579 EISA network card. The instructions assume an LPR printserver on the network, named `mainprinter.my.domain.com` with a print queue named RAW.

Use the installation instructions in Exhibit 8.1 for a quick and dirty TCP/IP Winsock for Win31 systems. Administrators who already have the Novell IPX client installed should skip those steps.

8.4.2 Installation of the Novell TCP/IP Winsock client

1. Make sure that the machine has enough environment space (2048 bytes or more) by adding the following line to the `config.sys` file and rebooting:

    `SHELL=C:\COMMAND.COM /E:2048 /P`

2. Obtain the `TCP16.EXE` file from ftp://ftp3.novell.com/pub/updates/eol/nweol/tcp16.exe.

3. Obtain the Network Adapter support diskette for the network card in your machine. This should be supplied with the card, or available via FTP from the network adapter manufacturer's FTP site.

4. Now you need the file `LSL.COM`. This is available on some Network Adapter Driver diskettes, it used to be available from the `VLM121_2.EXE` file from Novell but unfortunately this file is no longer publicly accessible from Novell.

5. If you have `vlm121_2.exe` in a temporary directory, run it. This will extract a number of files.

6. One of the files extracted is `LSL.CO_` extract this file with the command `nwunpack lsl.co_`.

7. Create the directory `c:\nwclient`. Then, copy `lsl.com` from the temporary directory into the directory.

8. Obtain and install the printer driver for the model of printer that you will be spooling to and point it to `LPT1:`. Win31 and WfW 3.11 have an incomplete printer driver list, so if you need a driver Microsoft has many Win16 printer drivers on their FTP site. A list is available at ftp://ftp.microsoft.com/Softlib/index.txt. In addition, if you

are installing a PostScript printer driver for a printer supplied in Win31, it may be necessary to patch the driver. The Microsoft PostScript driver supplied in Win31 is version 3.5. (The patch named `PSCRIP.EXE` which brought the PostScript driver to version 3.58 is no longer publicly available.) WfW already uses the more recent PostScript driver, as does Win31 version A. Installing the Adobe PostScript driver for Win31 is also an option. (see http://www.adobe.com/support/downloads/pdrvwin.htm for the version 3.1.2 Win31 PostScript driver).

9. Look on the network adapter driver disk for the subdirectory `nwclient/` and then look for the ODI driver for the adapter card. For example, on the 3com 3C509/3C579 adapter driver disk, the driver and location are `\NWCLIENT\3C5X9.COM`. Copy this driver to the `c:\nwclient` directory.

10. Create a file called `NET.CFG` in the `c:\nwclient` directory. Often, the network card adapter driver diskette has a template for this file in the same location as the ODI driver. This can be modified, as can the following example:

```
LINK SUPPORT

BUFFERS 4 1600

MEMPOOL 8192

LINK DRIVER
3C5X9

; PORT 300 (these are optional, if needed by card uncomment)

; INT 10 (optional, uncomment and modify if needed)
```

11. Attempt to load the network card driver. First load `lsl`, then the ODI driver. With the 3com card the commands are:

```
lsl
3c5x9
```

If the driver properly loads it will list the hardware port and interrupt settings for the network adapter. If it has loaded properly, unload the drivers in reverse order with the `/u` command:

```
3c5x9 /u
lsl /u
```

12. Go to the temporary directory that contains the `tcp16.exe` file and extract it by running the program.

13. Run the install batch file by typing `installr`. It should list `New Installation detected`. It will then copy a number of files into `nwclient`, add some commented-out sections to `net.cfg`, and call `edit` on `net.cfg`.

14. Read the editing instructions and make the appropriate entries. The sample `net.cfg` file from above would look like this.

```
LINK SUPPORT

BUFFERS 4 1600

MEMPOOL 8192

LINK DRIVER 3C5X9

FRAME ETHERNET_II
```

```
Protocol TCPIP

PATH TCP_CFG c:\nwclient

ip_address 192.168.1.54 LAN_NET

ip_netmask 255.255.255.0 LAN_NET

ip_router 192.168.1.1 LAN_NET

Bind 3C5X9 #1 Ethernet_II LAN_NET
```

Save and exit, the Installer should list `TCP16 installation completed`.

15. Reload the client with the commands:

    lsl
    3c5x9
    tcpip

    The TCP/IP driver should list the IP numbers and other information.

16. Optionally, create either a `HOSTS` file, or a `RESOLV.CFG` file (pointing to a nameserver) in `c:\nwclient`. Check to see this is operating properly by pinging a hostname.

    Add the `c:\nwclient` directory to the `PATH`, as well as the 3 startup commands in step 15 in `autoexec.bat`

8.4.3 Installation of the LPR client on 16-bit Windows with a Winsock installed

The following assumes a running Win31 installation with a Winsock or a running WfW installation with the 32-bit Microsoft TCP/IP protocol installed.

1. Install the printer driver desired. See step 8 of the previous set of instructions.

2. Obtain and extract into a temporary directory the `wlprs41.zip` file from the location mentioned above.

3. Run `setup.exe` from the temporary directory containing the `wlprs` files.

4. In setup, accept default directory, and check Yes to add to its own group. Click Continue when asked for group name, and check whatever choice you want when asked to copy the `doc` files.

5. Click No when asked to add the program to `Startup`.

6. On the UNIX FreeBSD print spooler, make sure that there is an entry in `/etc/hosts.lpd` or `/etc/hosts.equiv` for the client workstation, thereby allowing it to submit jobs.

7. Double-click the Windows LPR Spooler icon in the Windows LPR Spooler group that is opened. When it asks for a valid spool directory, just select the `c:\wlprspl` directory that the program installed its files into.

8. When asked for a valid Queue Definition File, just click OK to use the default filename. The program automatically creates a queue definition file.

9. The program opens up with its menu. Click Setup in the top menu, then select Define New Queue.

10. For a local spool filename, just use the name of the remote queue (RAW) to which the client prints.

11. For the remote printer name, use the same name as the remote queue (RAW) to which the client prints.

12. For the remote hostname, use the machine name of the FreeBSD print spooler. `mainprinter.ayedomain.com`.

13. For the Description, enter a description such as `3rd floor Marketing printer`.

14. For the protocol, leave the default of BSD LPR/LPD selected.

15. Click on the Queue Properties, and make sure that the Print unfiltered is selected. If you're printing PostScript, then also click the Advanced options button. Make sure that Remove trailing Ctrl-D is *unchecked*, and that Remove Leading Ctrl-D is *checked*. Also with PostScript, if the printer cannot print ASCII, uncheck the Send header page box. (PostScript header/banner pages are discussed later in this chapter)

16. Click OK. At the main menu of the program, click File, then Control Panel/Printers to bring up the Printers control panel of Windows.

17. Make sure that the Use Print Manager button is checked, then highlight the printer driver and click the Connect button.

18. Scroll down to the C:\WLPRSPL\RAW entry for the spool that was built and highlight this. Click OK.

19. Minimize the Windows LPR Spooler. Copy the Windows LPR Spooler icon to the Startup group. Click File/Properties with the Windows LPR Spooler icon highlighted in the Startup group. Check the Run Minimized button.

20. Exit Windows, and when the Save queue changes? button comes up, click Yes.

21. Restart windows and make sure that the spooler starts up.

22. Open the Control Panel and look for a new yellow icon named Set Username If you are running the Novell or other Winsock under Win31, click on this icon and put the username of the person using this computer into the space provided. If you are running WfW, this isn't necessary because Windows will supply the username.

23. If the spooler is not started properly in some installations, there may be a bug. If placing the icon in the Startup group doesn't actually start the spooler, the program name can be placed in the `run=` line of `win.ini`.

24. Try printing a print job from an application such as Notepad. If everything goes properly, clicking on the Queues/Show remote printer status" in the Windows LPR menu should show the print job spooled and printing on the remote printserver.

8.4.4 Installation of LPR client on Windows 95/98

The `wlprspl` program also can be used under Windows 95, but as a 16-bit program, it is far from an optimal implementation on a 32-bit operating system. In addition, Win95 and its derivatives fundamentally changed from Windows 3.1 in the printing subsystem. For these reasons I use a different LPR client program for Win95/98 LPR printing instructions. It is a full 32-bit print program, and it installs as a *Windows 32-bit printer port monitor*. The program is called ACITS LPR Remote Printing for Windows 95 and it is located at http://shadowland.cc.utexas.edu/acitslpr.htm.

ACITS stands for Academic Computing and Instructional Technologies Services. The ACITS LPR client includes software developed by the University of Texas at Austin and its contributors, it was written by Glenn K. Smith, a systems analyst with the Networking Services group at the university. The filename of the archive in the original program was ACITSLPR95.EXE and as of version 1.4 it was free for individuals or organizations to use for their internal printing needs. Since that time, it has gotten so popular that the university has taken over the program, incremented the version number (to get out from under the free license) and is now charging a $35 per copy fee for commercial use for the newer versions. The older free version can still be found on overseas FTP servers, such as http://www.go.dlr.de/fresh/pc/src/winsock/acitslpr95.exe.

It is likely that the cost of a shareware/commercial LPR program for Win95 plus the cost of Win95 itself will meet or exceed that of Win2K. As such, users wishing to print via LPR to FreeBSD UNIX systems will probably find it cheaper to simply upgrade to Windows NT Workstation or Win2K.

ACITS LPR and Win95 have a few printing idosyncracies. Most Win95 programs, such as Microsoft Word, expect print output to be spooled on the local hard drive and then metered out to a printer that is plugged into the parallel port. Network printing, on the other hand, assumes that print output will go directly from the application to the remote print server. Under Win95, local ports have a setting under Properties, Details, Spool Settings labeled "Print directly to the printer". If this is checked, the application running on the desktop (such as Microsoft Word) will not create a little Printer icon with pages coming out of it or use other means of showing the progress of the job as it is built. This can be very disconcerting to the user of a network printer, so this option should be checked only with printers plugged directly into the parallel port. Worse, if this is checked with ACITS, it can cause the job to abort if the remote print spooler momentarily goes offline.

Another local setting also should be changed. Generally, with local ports, Win95 builds the first page in the spooler and then starts printing it while the rest of the pages spool. If ACITS starts printing the first page while the rest of the pages are building, timeouts at the network layer can sometimes cause very large jobs to abort. The entire job should be set to completely spool before the LPR client passes it to the UNIX spooler. The problem is partly the result of program design: because ACITS is implemented as a local printer port instead of being embedded into Win95 networking (and available in Network Neighborhood) the program acts like a local printer port in some ways.

The LPR program can be set to deselect banner/burst page printing if a PostScript printer that cannot support ASCII is used. The burst pages referred to here are NOT generated by the Windows machine. Use the instructions in Exhibit 8.3 to install ACITS.

LPR client on Win95/98 installation instructions

1. Obtain the `ACITSLPR95.EXE` file and place it in a temporary directory such as `c:\temp1`.

2. Close all running programs on the desktop. The computer *must* be rebooted at completion of installation or the program will not work.

3. Click Start, Run and type in **c:\temp1\acitslpr95** then click Yes at the InstallShield prompt.

4. Click Next, then Yes. The program will run through some installation and then presents a Help screen that explains how to configure an LPR port.

5. After the help screen closes, the program asks to reboot the system. Ensure that Yes is checked and click Finish to reboot.

6. After the machine comes back up, install a Printer icon in the Start, Settings, Printers folder if one hasn't been created for the correct model of destination printer.

7. With the Printers folder open, right-click over the printer icon that needs to use the LPR program and click on the Properties tab.

8. Under the Details tab, click the Add Port tab, then click Other.

9. Highlight the ACITS LPR Remote Printing line and click OK.

10. The Add ACITS LPR screen opens. Type in the hostname of the UNIX system that the client spools through— *mainprinter.ayedomain.com*.

11. Type in the Printer/Queue name and click OK. (Some versions have a "Verify Printer Information" button.) The LPR program then contacts the UNIX host and makes sure that the selected printer is available.

> **Note:** If this fails the client machine name is probably not in the `/etc/hosts.equiv` or `etc/hosts.lpd` on the FreeBSD printserver. Most sites may simply decide to put a wildcard in `hosts.equiv` to allow printing, especially if DHCP is used, but many security-conscious sites may stick with individual entries in `hosts.lpd`.

12. If the printer is PostScript and cannot print ASCII, make sure that the "No banner page control flag" is checked to turn off banner pages. Accessible under Port settings, this flag is overridden if the `/etc/printcap` file specifies no banner pages.

13. Review how the "send plain text control flag" is set. With this flag unchecked, the LPR code sent is L, (i.e., print unfiltered) meaning that the `if` filter gets called with the `-c` option. This is equivalent to the local invocation of `/usr/bin/lpr -l`. With the flag checked, the code is F, (formatted) meaning that the `if` filter gets called without the `-c` option. This is equivalent to the default invocation `/usr/bin/lpr`. (This is also an issue under Windows NT, which retypes the print job to text if this flag is checked. Some filters understand the `-c` flag, which is used to preserve control characters, so it should generally remain unchecked.

14. Leave the "Send data file before control file" box unchecked. This option is used only in rare mainframe spooling circumstances.

15. Click OK, then click the Spool Settings button at the properties page.

16. Make sure that the "Spool print jobs so program finishes printing faster" box is checked.

17. Make sure that "Start printing after last page is spooled" box is checked.

18. Make sure that "Disable bi-directional support for this printer" is checked, or greyed out.

19. Make sure that the "Spool data format" is set to RAW. Some printer drivers present a choice of EMF or RAW, such as the Generic Text driver, in this case select RAW.

20. Click OK, then OK again to close the Printer Properties. The printer icon now spools through FreeBSD.

8.4.5 Installation of LPR client on Windows NT

Unlike WfW and Win95 TCP/IP, Windows NT—both server and workstation—includes an LPR client as well as an LPD program that allows incoming print jobs to be printed from LPR clients, such as UNIX systems.

To install the LPR client and daemon program under Windows NT 3.51, use the following instructions. The TCP/IP protocol should be installed beforehand and you must be logged in to the NT system as Administrator. This can be done at any time after the NT system is installed, or during OS installation:

1. Double-click on Main, Control Panel, then Network Settings.

2. In the Installed Network Software window, "Microsoft TCP/IP Printing" should be listed as well as "TCP/IP Protocol". If it is, stop here; otherwise continue.

3. Click the Add Software button to get the Add Network Software dialog box

4. Click the down arrow and select TCP/IP Protocol and related components. Click Continue.

5. Check the "TCP/IP Network Printing Support" box and click Continue. LPR printing is now installed. Follow the instructions to reboot to save changes.

To install the LPR client and daemon program under Windows NT 4, use the following instructions. The TCP/IP protocol should be installed beforehand and you must be logged in to the NT system as Administrator. This can be done at any time after the NT system is installed, or during OS installation:

1. Click on Start, Settings, Control Panel, and double-click on Network to open it up.

2. Click on the Services tab. `Microsoft TCP/IP Printing` should be listed. If not, continue steps 3 - 4.

3. Click Add, then select Microsoft TCP/IP Printing and click OK.

4. Click Close. Follow instructions to reboot to save changes.

> **Note:** Any NT Service Packs that were previously installed must be reapplied after these operations.

Once LPR printing has been installed, the Printer icon or icons must be created on the NT system so that applications can print. Since this printer driver does all job formatting before passing the printing to the FreeBSD printserver, the print queues specified should be raw queues on the FreeBSD system, which don't do any job formatting.

To install the printer icon in Print Manager and set it to send print jobs to the FreeBSD UNIX system, use the following instructions under NT 3.51. You must be logged in to the NT system as Administrator. This can be done at any time after the NT system is installed, or during OS installation.

1. Click on Main, and open it. Then click on Print Manager to open it.

2. Click on Printer, Create Printer. Select the appropriate printer driver.

3. Click the down arrow under Print To and select Other.

4. In the Available Print Monitors window select LPR port and click OK.

5. Enter the hostname of the FreeBSD printserver, and the name of the printer queue and click OK

6. Click OK to close the Create Printer window. The Printer icon is created.

To install the printer icon in Print Manager and set it to send print jobs to the FreeBSD UNIX system, use the following instructions under NT 4. You must be logged in to the NT system as Administrator. This can be done at any time after the NT system is installed, or during OS installation:

1. Click Start, Settings, Printers to open the printer folder.

2. Double-click Add Printer to start the wizard.

3. Select the My Computer radio button, not the Network Print Server button and click Next. (The printer *is* a networked printer, it is managed on the local NT system. Microsoft used confusing terminology here.

4. Click Add Port and select LPR Port, then click New Port.

5. Enter the hostname and print queue for the FreeBSD printserver and click OK.

6. Click Next and select the correct printer driver. Continue until the printer is set up.

The LPR client in Windows NT allows DOS print jobs originating in DOS boxes to be routed to the central UNIX print spooler. This is an advantage over the Win95 and WfW LPR programs.

8.4.5.1 Windows NT Registry Changes

Using the LPR daemon program under Windows NT presents one problem. If the NT server is used as an LPR/LPD "relay", for example, to pass jobs from clients to LPR print queues on a UNIX system, to pass jobs from LPR programs on UNIX terminating at NT print queues, or to pass jobs from Appletalk clients to LPR printers, NT retypes the job if the type code is set to P (text). This can wreak havoc on PostScript files printed through HP LaserJet printers with internal MIO cards in them, if the job originates from the `/usr/bin/lpr` program under UNIX, which assigns a P type code. The printserver card treats PostScript jobs as text, and instead of the print job, the raw PostScript codes print. This problem often manifests in the following way: `/usr/bin/lpr` is used to print a PostScript file from UNIX directly to the remote printer printserver, which works fine, but spooling it through NT causes problems.

A registry change that can override the NT Server formatting behavior is detailed in Microsoft Knowledge Base article ID Q150930. With Windows NT 3.51, and 4.0 up to service pack 1 the change is global. Starting with NT 4.0 Service pack 2 the change can be applied to specific print queues, (see Knowledge Base article ID Q168457). This registry change also works for Windows 2000.

Under Windows NT 4.0, the change is:

1. Run Registry Editor (`REGEDT32.EXE`)

2. From the `HKEY_LOCAL_MACHINE` subtree, go to the following key:

   `\SYSTEM\CurrentControlSet\Services\LPDSVC\Parameters`

3. On the Edit menu, click Add Value.

4. Add the following:

Value Name:	SimulatePassThrough
Data Type:	REG_DWORD
Data	1

   Note: The default value is 0, which informs LPD to assign datatypes according to the control commands.

Under Windows NT 3.51, the change is:

1. Run Registry Editor (`REGEDT32.EXE`)

2. From the `HKEY_LOCAL_MACHINE` subtree, go to the following key:

   `\SYSTEM\CurrentControlSet\Services\LPDSVC\Parameters`

3. On the Edit menu, click Add Value.

4. Add the following:

Value Name:	SimulatePassThrough
Data Type:	REG_DWORD
Data	1

> **Note:** The default value is 0, which informs LPD to assign datatypes according to the control commands.

5. Create an LPD key at the same level as the LPDSVC key.

6. Click the LPDSVC Key, click Save Key from the Registry menu, and then save the file as `LPDSVC.KEY`

7. Click the LPD key created in step 5.

8. Click Restore on the Registry menu, click the file created in step 6, and then click OK.

9. A warning message appears. Click OK and then quit the Registry Editor.

10. At a command prompt window, type:

```
net stop lpdsvc
net start lpdsvc
```

8.5 Printing PostScript and DOS command files

One problem with printing under Win31 and Win95 with the LPR methods discussed is the lack of a "raw" `LPT1:` device. This is annoying to the administrator who wants to print an occasional text file, such as a file full of printer control codes, without their being intercepted by the Windows printer driver. Of course this is also an issue with DOS programs, but a commercial site that runs significant DOS software and wants to print directly to UNIX with LPR really only has one option—to use a commercial TCP/IP stack containing a DOS LPR program.

Normally, under Windows printing, virtually all graphical programs print through the Windows printer driver. This is true even of basic programs such as Notepad. For example, an administrator may have a DOS batch file named `filename.txt` containing the following line:

```
echo \033&k2G > lpt1:
```

This batch file switches a HP LaserJet from CR-LF, MS-DOS textfile printing into Newline termination UNIX textfile printing. Otherwise, raw text printed from UNIX on the HP prints with a stairstep effect.

If the administrator opens this file with Notepad and prints it using a regular printer driver, such as an Epson LQ, the Windows printer driver encapsulates this print output into a series of printer-specific control codes that do things such as initialize the printer, install fonts, and so on. The printer won't interpret this output as control code input. Usually, if the printer is locally attached, the user can force a "raw text print" of the file by opening a DOS window and running:

```
copy filename.txt lpt1: /b
```

Since the LPR client program doesn't provide a DOS driver, it cannot reroute input from the `LPT1:` device ports. The solution is to use the Generic / Text Only printer driver in conjunction with Wordpad (under Win95); under Win31 use a different text editor. The Notepad editor supplied with Windows is unsuitable for this - it "helpfully" inserts a 1 inch margin of spaces around all printed output, as well as the filename title. Wordpad supplied with Win95, can be set to use margins of zero, and inserts no additions into the printed output. Also, make sure that banner pages are turned off, and the print type is set to raw.

8.6 Checking PostScript Printer capabilities

Following is a PostScript command file that can be used to get a PostScript printer to output a number of useful pieces of information that are needed to set up a printer icon under Windows properly. It was printed from Wordpad, in Win95, through the Generic / Text Only printer driver with the following instructions:

1. Start, Run, type in **Wordpad** and press **Enter**.

2. File, Open `testps.txt`

3. File, Page Setup, Printer, select Generic / Text Only, click Properties

4. Click Device Options, select TTY custom, click OK.

5. Click OK, then set all four margins to `0`; click OK.

6. Click File, Print, OK.

This could also have been printed with `/usr/bin/lpr` on a UNIX command prompt. The file prints *Test Page* and some printer statistics below that, as follows.

```
% filename: testps.txt
% purpose: to verify proper host connection and function of PostScript
% printers.
/buf 10 string def
/CM {
save statusdict/product get (PostScript) anchorsearch
exch pop {length 0 eq
{1}{2}ifelse
}
{2}ifelse exch restore
}bind def
/isCM {
CM 1 ge
}bind def
/Times-BoldItalic findfont 75 scalefont setfont
150 500 moveto
(Test Page) false charpath
isCM{gsave 0.0 1.0 1.0 0.0 setcmykcolor fill grestore}if
2 setlinewidth stroke
/Times-Roman findfont 10 scalefont setfont
150 400 moveto
(Your PostScript printer is properly connected and operational.)show
150 380 moveto
(The border around the page indicates your printer's printable region.)show
{ vmreclaim } stopped pop
vmstatus exch sub exch pop
150 360 moveto
(Max Available Printer Virtual Memory (KB):)show
150 340 moveto
dup 1024 div truncate buf cvs show
150 320 moveto
(Calculated memory size used for PostScript printer icon properties:) show
150 300 moveto
0.85 mul 1024 div truncate buf cvs show
```

```
150 280 moveto
(Printer Model: )show
statusdict begin product show end
150 260 moveto
(PostScript Level: )show
/languagelevel where
{ languagelevel 3 string cvs show pop }
{(1) show } ifelse
150 240 moveto
(PostScript Version: )show
statusdict begin
version show (.)show
revision 40 string cvs show end
clippath stroke
showpage
```

8.7 Setting up LPR/LPD on FreeBSD

When a FreeBSD system is booted, it starts the LPD spooler control daemon program if the `/etc/rc.conf` file has `lpd_enable="YES"` set. If this is not set, attempts to print through and from the FreeBSD system will fail with an "`lpr: connect: No such file or directory`" error message.

The LPD program manages all incoming print jobs, whether they come in from the network, or from local users on the UNIX system. It transfers print jobs to all locally attached parallel or serial printers, as well as defined remote printers. Several programs also are used to manipulate jobs in the print spools that LPD manages, as well as the user programs to submit them from the UNIX command prompt. All of these programs use the `/etc/printcap` file, which is the master control file for the printing system.

Back when printing was mostly text, it was common to place printers on a serial connection that stretched for long distances. Often, 9600bps was used because it could work reliably up to a block away, which allowed printers to be located almost anywhere on an office high-rise floor. Modern office print jobs, on the other hand, are generally graphics-laden and tend to be rather large. These jobs would take hours to transfer over a slower 9600bps serial printer connection. Today, most printers that are not connected to a remote hardware print server box are directly connected to the server using parallel cables. All of the examples shown here are direct connections that are parallel connections.

The `printcap` configuration file, like most UNIX configuration files, indicates comment lines starting with a hash character. Lines without a hash character are meant to be part of a printer queue description line. Each printer queue description line starts with a symbolic name, and ends with a newline. Since the description lines are often quite long, they are often written to span multiple lines by escaping intermediate newlines with the backslash (\) character. The `/etc/printcap` file, as supplied, defines a single printer queue, `lp`. The `lp` queue is the default queue. Most UNIX-supplied printing utilities send print output to this queue if no printer is specified by the user. It should be set to point to the most popular print queue with *local* UNIX print users, (i.e., users that have shell accounts).

The layout of `/etc/printcap` is covered in the manual page, which is reached by running the **man printcap** command. The stock `/etc/printcap` file at the line defining the spool `lp` shows:

```
#
lp|local line printer:\
    :lp=/dev/lpt0:sd=/var/spool/output/lpd:lf=/var/log/lpd-errs:
#
```

In this example the first line defines the names by which the printer is known, and ends with an escaped newline. The next line defines the physical device, the PC parallel port, by `/dev/lpt0`, and the directory in which the spool files are stored at `/var/spool/output/lpd`, and the error log file. Note that this particular error log file will not show all LPD errors, such as bad job submittals, it usually shows only the errors that originate within the printing system itself.

In general, the administrator creates two print queues for every printer that is connected to the FreeBSD machine. The first queue entry contains whatever additional capabilities UNIX shell users on the server require. The second is a raw queue that performs no print processing on the incoming print job. This queue is used by remote clients, such as Windows clients, that format their own jobs.

If the administrator is setting up the printer to allow incoming LPR jobs from network clients, such as other Windows or UNIX systems, those systems *must* be listed in `/etc/hosts.lpd`.

8.7.1 Creating the spools

Building new print spools is merely a matter of making an entry in the `/etc/printcap` file, creating the spool directories, and setting the correct permissions on them. For example, the following additional line defines a PostScript printer named NEC (in addition to the `lp` definition):

```
#
lp|local line printer:\
    :lp=/dev/lpt0:sd=/var/spool/output/lpd:lf=/var/log/lpd-errs:

NEC|NEC Silentwriter 95 PostScript printer:\
    :lp=/dev/lpt0:sd=/var/spool/output/NEC:lf=/var/log/lpd-errs:
#
```

Because UNIX is case sensitive, NEC is different from `nec` in both the name of the printer and the name of the Spool directory. With the print spooler LPD, the Spool directories *must* be different from each other, or the spooler gets confused and doesn't print.

After the `/etc/printcap` is modified, the root user must create the `/var/spool/output/NEC` directory and assign ownership of it to the `bin` user, assign group ownership to `daemon`, and set permissions with the following commands:

```
% su root
# cd /var/spool/output
# mkdir NEC
# chown bin NEC
# chgrp daemon NEC
# chmod 755 NEC
```

8.7.2 Additional spool capabilities

Because modern print jobs (especially PostScript) can sometimes reach hundreds of megabytes, the `sd` capability entry in the `/etc/printcap` file should always point to a Spool directory on a filesystem that has enough space. The `/var` directory on a default FreeBSD installation is generally set to a fairly small amount, which can easily overflow the spool. There are four ways to handle this problem:

1. During FreeBSD installation, if the administrator knows a lot of print jobs are going to go through the spooler, `/var` should be set to a large amount of free space.

2. Modify the `sd` capability in the `/etc/printcap` file to point to a spool directory in a different, larger filesystem, such as `/usr/spool`.

3. Use soft links to point the `/var/spool/output` directory to directories on a larger filesystem.

4. Don't define a `/var` directory at all during FreeBSD installation; this would make the installer link `/var` to `/usr/var`.

In addition to spools, the following other capabilities are usually placed in a production `/etc/printcap` file.

The entry `fo` prints a form feed when the printer is opened. It is handy for HPPCL (HP LaserJets) or other non-PostScript printers that are located behind electronic print sharing devices. It can also be used for printers that accept input from multiple connections, such as a parallel port, serial port, and localtalk port. An example is an HP LaserJet with an MIO card in it plugged into both Ethernet and LocalTalk networks. It will clear any garbage out of the printer before the job is processed.

The entry `mx` defines the maximum size of a print job, which is a must for modern print jobs that frequently grow far past the default print size of a megabyte. The original intent of this capability was to prevent errant programs from stuffing the spool with jobs so large that they would use up all paper in a printer. Graphics-heavy print jobs have made it impossible to depend on this kind of space limitation, so `mx` is usually set to zero, which turns it off.

The entry `sh` suppresses printing of banner pages in case the printer cannot handle ASCII and the client mistakenly requests them.

The entry `ct` denotes a TCP Connection timeout. This is useful if the remote print server doesn't close the connection properly.

> **Note:** FreeBSD 2.2.5 contains a bug in the LPD system - as a workaround the `ct` capability needs to be set very large, such as 3600, or the appropriate patch installed and LPD recompiled. More recent versions of FreeBSD do not have this bug.

8.7.3 Printing to hardware print server boxes or remote print servers.

Hardware print server boxes, such as the HP JetDirect internal and external cards, need some additional capabilities defined in the `/etc/printcap` entry; `rp`, for remote print spool, and `rm` for remote machine name.

The `rm` capability is simply the DNS or `/etc/hosts` name of the IP number associated with the remote printserver device. Obviously, print server devices, such as the HP JetDirect, must not use a dynamic TCP/IP network numbering assignment. If they get their numbering via DHCP, the IP number should be assigned from the static pool; it should always be the same IP number.

Determining the name used for `rp`, on the other hand, can be rather difficult. Here are some common names:

Windows NT Server: Printer name of the printer icon created in Print Manager

FreeBSD: Print queue name defined in `/etc/printcap`

HP JetDirect: Either the name `TEXT` or the name `RAW`. `TEXT` automatically converts incoming UNIX newline text to DOS-like CR/LF text that the printer can print. `RAW` should be used for PostScript, and HPPCL printing.

HP JetDirect EX +3: External, 3 port version of the JetDirect. Use `RAW1`, `RAW2`, `RAW3`, `TEXT1`, `TEXT2`, or `TEXT3` depending on the port desired.

Intel NetPort: Either use `TEXT` for UNIX text conversion printing or use `PASSTHRU` for normal printing.

DPI: Use `PORT1` or `PORT2` depending on which port the printer is plugged into.

For other manufacturer's print servers refer to the manuals supplied with those devices.

The following is an example printcap that redefines the default `lp` print queue to send print jobs to the first parallel port on a remote HP LaserJet plugged into a JetDirect EX +3 named `floor2hp4.biggy.com`.

```
#
lp|local line printer:\
    :rm=floor2hp4.biggy.com:rp=RAW1:\
    :sd=/var/spool/output/lpd:\
    :lf=/var/log/lpd-errs:
#
```

> **Note:** The `rp` capability *must* be defined or the job goes to the default print queue on the remote host. If the remote device does not have a single print queue, such as another UNIX system, this causes problems. For example, if the remote device was a JetDirect EX + 3 and `rp` was omitted, all queues defined would print out of the first parallel port.

8.7.4 Filters

The last two important printcap capabilities concern print filters, `if` (input filter) and `of` (output filter). If defined, incoming print jobs are run through the filters that these entries point to for further processing.

Filters are the reason that the UNIX print spooling system is so much more powerful than any other commercial server operating system. Under FreeBSD, incoming print jobs are acted on by any filters specified in the `/etc/printcap` *no matter where they originate*. Incoming print jobs from remote Windows, Mac, NT, OS/2 or other clients can be intercepted and manipulated by any program specified as a filter. Want a PostScript Printer? There's a filter that adds PostScript capability to a non-PostScript printer. Want to make a cheap Epson MX 80 dot-matrix emulate an expensive Okidata Microline dot-matrix for some archaic mainframe application? Write a filter that will rewrite the print codes to do it. Want custom-built banner pages? Use a filter. Many UNIX `/etc/printcap` filters on many Internet sites can do a variety of interesting and unique things. Someone may have already written a filter that does what you want!

8.7.4.1 Types of Filters

Three types of filters can be defined in the `/etc/printcap` file. In this book all filter examples are for Input filters.

8.7.4.1.1 Input Filters

Input filters are specified by the `if` capability. Every job that comes into the spool is acted on by any filter specified in the `if` entry for that spool. Virtually all filters that an administrator would use are specified here. These filters can be either shell scripts, or compiled programs.

8.7.4.1.2 Fixed Filters

Fixed filters are specified by separate capabilities, such as `cf`, `df`, and `gf`. Mostly, these exist for historical reasons. Originally, the idea of LPD was that incoming jobs would be submitted with the type fields set to trigger whatever filter was desired. However, type codes are confusing and annoying to the user, who has to remember which option is needed to trigger which type. It is much easier to set up multiple queues with different names, and this is what most sites do these days. For example, originally a DVI fixed filter might be specified in a spool for `lp`, triggered by the `-d` option passed to `lpr`. Jobs without this option aren't acted on by the DVI filter. However, the same thing can be done by creating a queue named `lp` that doesn't have a DVI filter, and a queue named `lpdvi` which has the DVI filter specified in the `if` capability. Users just need to remember which queue to print to, instead of what option needed for this or that program.

8.7.4.1.3 Output Filters

These are specified by the `of` capability. Output filters are much more complicated than input filters and are hardly ever used in normal circumstances. They also generally require a compiled program somewhere, either directly specified or wrapped in a shell script, since they have to do their own signal-handling.

8.7.4.2 Printing Raw UNIX Text with a Filter

One of the first things that a new UNIX user will discover when plugging a standard LaserJet or impact printer into a UNIX system is the *stairstep* problem. The symptom is that the user dumps text to the printer, either through LPR or redirection (by catting it to the parallel device) and instead of receiving the expected Courier 10-point printout, gets a page with a single line of text, or two lines of text "stairstepped", text and nothing else.

The problem is rooted in how printers and UNIX handle textfiles internally. Printers by and large follow the "MS-DOS Textfile" convention of requiring a carriage return, then a linefeed, at the end of every text line. This is a holdover from the early days when printers were mechanical devices, and the print head needed to return and the platen to advance to start a new line. UNIX uses only the linefeed character to terminate a text line. So, simply dumping raw text out the parallel port works on MS-DOS, but not on UNIX.

If the printer is a PostScript printer, and doesn't support standard ASCII, then dumping UNIX text to it doesn't work. But then, neither would dumping MS-DOS text to it. (Raw text printing on PostScript printers is discussed later in this chapter.) Note also that if the printer is connected over the network to an HP JetDirect hardware print server, internal or external, the TEXT queue on the hardware print automatically adds the extra Carriage Return character to the end of a text line.

If the printer is the garden-variety HP LaserJet, DeskJet, or an impact printer, and under DOS the administrator is used to printing raw text from the command line for directory listings, there are two ways to fix stairstep. The first is to send a command to the printer to make it print in "unix textfile" mode, which makes the printer supply its own carriage return. This solution is ugly in a printer environment with UNIX and Windows machines attempting to share use of the same printer. Switching the printer to work with UNIX disrupts DOS/Windows raw text printouts.

The better solution is to use a simple filter that converts incoming text from UNIX style to DOS style. The following filter posted on questions@FreeBSD.org and the sample `/etc/printcap` entry can be used to do this:

```
#!/bin/sh
# /usr/local/libexec/crlfilter
#
# simple parlor trick to add CR to LF for printer
```

```
# Every line of standard input is printed with CRLF
# attached.
#

awk '{printf "%s\r\n", $0}' -
```

An alternative filter posted using sed could be written as:

```
#!/bin/sh
# /usr/local/libexec/crlfilter
#
# Add CR to LF for printer
# Every line of standard input is printed with CRLF
# attached.
#
# Note, the ^M is a *real* ^M (^V^M if your typing in vi)
#

sed 's/$/^M/' -
```

Here is an example of a filter that triggers the printers automatic LF-to-CR/LF converter (this option is only useful on HP LaserJets that support this command):

```
#!/bin/sh
# Simply copies stdin to stdout.  Ignores all filter
# arguments.
# Tells printer to treat LF as CR+LF.  Writes a form feed
# character after printing job.

printf "\033&k2G" && cat && printf "\f" && exit 0

exit 2
```

The printcap file used to trigger the filter is:

```
#/etc/printcap
# The trailer (tr) is used when the queue empties.  I found that the
# form feed (\f) was basically required for the HP to print properly.
# Banners also need to be shut off.
#

lp|local line printer:\
    :lp=/dev/lpt0:sd=/var/spool/output/lpd:lf=/var/log/lpd-errs:
    :if=/usr/local/libexec/crlfilter:sh:tr=\f:mx#0:
#
```

8.7.4.3 The `pr` filter

Although most filters are built by scripts or programs and are added to the UNIX machine by the administrator, there is one filter that is supplied with the FreeBSD operating system is very useful for raw text files: the `pr` filter. It is most commonly used when printing from the UNIX command shell. The `pr` filter paginates and applies headers and

footers to ASCII text files. It is automatically invoked with the −p option used with the lpr program at the UNIX command prompt.

The pr filter is special - it runs *in addition* to any input filters specified for the print queue in /etc/printcap, *if* the user sets the option for a print job. This allows headers and pagination to be applied in addition to any special conversion, such as CR to CR/LF that a specified input filter may apply.

8.7.4.4 Printing PostScript Banner Pages with a Filter.

Unfortunately, the canned banner page supplied in the LPD program prints only on a text-compatible printer. If the attached printer understands only PostScript and the administrator wants to print banner pages, it is possible to install a filter into the /etc/printcap file to do this.

The following filter is taken from the FreeBSD Handbook. I've slightly changed its invocation for a couple of reasons. First, some PostScript printers have difficulty when two print files are sent within the same print job or they lack the trailing Control-D. Second is that the handbook invocation uses the LPRPS program, which requires a serial connection to the printer.

The following filter shows another trick: calling LPR from within a filter program to spin off another print job. Unfortunately, the problem with using this trick is that the banner page always gets printed after the job. This is because the incoming job spools first, and then FreeBSD runs the filter against it, so the banner page generated by the filter always spools behind the existing job.

There are two scripts, both should be put in the /usr/local/libexec directory, and the modes set to executable. The printcap also must be modified to create the nonbanner and banner versions of the print queue. Following the scripts is the /etc/printcap file showing how they are called. Notice that the sh parameter is turned on since the actual printed banner is being generated on the fly by the filter:

```
#!/bin/sh
# Filename /usr/local/libexec/psbanner
# parameter spacing comes from if= filter call template of:
# if -c -w -l -i -n login -h host
# parsing trickiness is to allow for the presence or absence of -c
# sleep is in there for ickiness of some PostScript printers

for dummy
do
  case "$1" in
    -n) alogname="$2" ;;
    -h) ahostname="$2" ;;
  esac
  shift
done

/usr/local/libexec/make-ps-header $alogname $ahostname "PostScript" | \
  lpr -P lpnobanner

sleep 10

cat && exit 0
```

Here is the make-ps-header listing.

```
#!/bin/sh
# Filename /usr/local/libexec/make-ps-header
#
# These are PostScript units (72 to the inch).  Modify for A4 or
# whatever size paper you are using:
#

page_width=612
page_height=792
border=72


#
# Save these, mostly for readability in the PostScript, below.
#

user=$1
host=$2
job=$3
date=`date`

#
# Send the PostScript code to stdout.
#

exec cat <<EOF
%!PS
%
% Make sure we do not interfere with user's job that will follow
%

%
% Make a thick, unpleasant border around the edge of the paper.
%

$border $border moveto
$page_width $border 2 mul sub 0 rlineto
0 $page_height $border 2 mul sub rlineto
currentscreen 3 -1 roll pop 100 3 1 roll setscreen
$border 2 mul $page_width sub 0 rlineto closepath
0.8 setgray 10 setlinewidth stroke 0 setgray

%
% Display user's login name, nice and large and prominent
%

/Helvetica-Bold findfont 64 scalefont setfont
$page_width ($user) stringwidth pop sub 2 div $page_height 200 sub moveto
($user) show

%
% Now show the boring particulars
%
```

```
/Helvetica findfont 14 scalefont setfont
/y 200 def
[ (Job:) (Host:) (Date:) ] {
200 y moveto show /y y 18 sub def
} forall
/Helvetica-Bold findfont 14 scalefont setfont
/y 200 def
[ ($job) ($host) ($date) ] {
270 y moveto show /y y 18 sub def
} forall

%
% That is it
%

showpage
```

Here is the `/etc/printcap` file.

```
#
lp|local line printer, PostScript, banner:\
  :lp=/dev/lpt0:sd=/var/spool/output/lpd:lf=/var/log/lpd-errs:\
  :if=/usr/local/libexec/psbanner:sh:mx#0:

lpnobanner|local line printer, PostScript, no banner:\
  :lp=/dev/lpt0:sd=/var/spool/output/lpd-noban:\
  :lf=/var/log/lpd-errs:sh:mx#0:
#
```

8.8 Printer Accounting

The FreeBSD print spooler can manage accounting statistics for printer usage. The spooler counts each page printed and generates totals for each user. In this manner departments or individuals can be charged money for their use of the printer.

In the academic world, such as student computer labs, accounting is very political. Many schemes have been developed to attempt to gather statistics to charge people (generally students) for printing. Administrators in this environment who deal with printers can have almost as many accounting problems as printer problems. In the corporate environment, on the other hand, accounting is not as important. I strongly recommend against any corporation attempting to implement printer accounting on shared printers for a number of reasons:

1. The entire UNIX accounting system is based on ASCII printouts. It is easy to count the number of ASCII pages, form feeds, or text lines in a print job. In corporations, however, PostScript and HPPCL are generally the order of the day. It is almost impossible to figure out by examining the datastream how many pages it will occupy, and even if this could be done accurately, it wastes significant computational resources.

> **Note:** It is possible to get some PostScript printers to count pages, but doing so requires a bidirectional connection to the printer and additional programming on the UNIX system. This task is beyond the scope of this book.

2. Banner pages aren't included in UNIX printer accounting counts. Therefore, someone submitting 20 two-page jobs uses much more paper than does someone submitting one 40 page job, yet both are charged the same amount.

3. The username of the submitter can be easily forged, if the job is remotely submitted over the network from a client (practically all jobs in a Windows client printing environment are remotely submitted). Although some LPR clients can be set to authenticate, and the `rs` capability can be set to enforce authentication, not all can, especially Windows LPR clients.

4. It is more difficult for a submitter to hide the IP number or machine name of the remote client, but in a Windows environment there is no guarantee that someone was sitting at a particular desktop machine when the job was submitted.

5. A business generates no revenue by monitoring printer usage. In the academic community, however, when a student lab charges for printouts the lab is actually extracting money from an entity (the student) that is separate from the lab. Within a corporation, the concept of department A getting revenue from user B is pointless and doesn't generate a net gain for the corporation as a whole.

   For my printer administration, I have found that I can save more money on printing costs by purchasing supplies wisely than by attempting to discourage printing through "chargebacks". What is the sense of being miserly with printing while spending double on toner cartridges because no one is willing to comparison shop, or signing a "lease" agreement that isn't beneficial for the printer? When you get down to it, corporate users don't care much for print sharing anyway, and they generally only agree to it because the administrator can buy a far bigger, faster, and fancier printer than they can requisition.

6. Worse yet, if usage on a shared printer is charged, it encourages employees to look for other places to print. Inevitably, people run out buy cheap inkjet printers for their own use, and the business ends up spending more on paper and supplies for many poor-quality small printers, than it would for a few decent big ones. Moreover, the inferior output of these printers makes the organization as a whole look bad.

7. The corporate spirit should be one of teamwork, not bickering. The surest way to kill a network in a corporation is to set up a situation that puts the administrator into the policeman position or pits one department against another.

The only justification I've ever seen for running accounting on corporate printers is using the accounting system to automate reminders to the administrator to replace paper, or toner. Aside from this use, a corporation that implements accounting as a way of encouraging employees not to waste paper ends up defeating the purpose of turning on accounting.

8.9 Microsoft Networking Client printing with Samba

Although LPR is a time-tested and truly cross-platform printing solution, sites with a majority of Windows clients running Microsoft Networking have an alternate printing mechanism—Samba. Samba can provide print services to clients running SMB-compatible network clients. With a running Samba installation, the administrator may "share out" printers as well as filesystem directories from the FreeBSD system.

Printers accessed with Samba must be defined both in the `/etc/printcap` file and the `/usr/local/etc/smb.conf` file. If the individual printers are defined in the `smb.conf` file with the `printer driver=` statement set to the exact model name of the printer, the "Auto printer driver install" feature of Windows NT and Win95/98 is activated. This automatically loads the correct printer driver if the user clicks on the print queue in Network Neighborhood under Windows 95 or NT 4.0. The restriction, of course, is that the printer model must be in the Windows client driver database.

The `smb.conf` file also defines the `print` command used to pass jobs to the UNIX print spool. It is a good idea to redefine this via the `print command` option to `lpr -s -P %p %s; rm %s`. This turns on soft linking, so that large print jobs don't get truncated.

In operation, the SMB-networking client builds the print job on itself and then transfers the entire job over the network to the Samba server. On the server, Samba has its own temporary print spool directory to which the job is copied. Once the job has been completely received, it is then passed to the UNIX print spooler.

Figure 8-6. Microsoft Networking Client printing with Samba
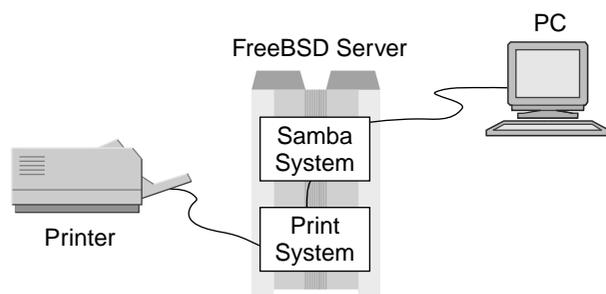
8.9.1 Client access issues

Because a Windows client formats print jobs before sending them to the server, the administrator may want to hide some of the specialty print queues on the server. For example, the queue that converts LF to CRLF for UNIX text printouts would probably not be shared out. To make such queues invisible, the `browseable=no` option can be turned on in the `smb.conf` file. Also, the `load printers` option must be set to no to allow individual printer definitions.

> **Note:** In general, the only print queues that should be visible through Samba are the "raw" print queues that are set up by the administrator to allow incoming preformatted print jobs.

Windows clients that print to Samba print queues on the UNIX system can view and cancel print jobs in the print queue. They cannot pause them, however, which is a difference between Novell and Windows NT Server print queues. They also cannot prioritize print jobs from the print queue window, although the administrator can reprioritize print jobs that are in the queue from a command shell on the FreeBSD server.

8.9.2 Printer entries in configuration files

Following are listings of sample `/etc/printcap` and `smb.conf` files used on the system to provide print services. An explanation of the interaction of these files follows.

Example 8-1. `/etc/printcap`

```
#
#
# The printer in lpt0 is a PostScript printer.  The nec-crlf entry
# is for testing the printer when it is switched into HP LaserJet III
# mode.
#

lp|local line printer:\
  :lp=/dev/lpt0:sd=/var/spool/output/lpd:\
  :lf=/var/log/lpd-errs:sh:mx#0:


#

nec-crlf|NEC Silentwriter 95 in ASCII mode with UNIX text filter:\
  :lp=/dev/lpt0:sd=/usr/lpdspool/nec-crlf:\
  :lf=/var/log/lpd-errs:sh:mx#0:\
  :if=/usr/local/libexec/crlfilter:tr=\f:


#

nec-raw|NEC Silentwriter 95 used for PostScript passthrough printing:\
  :lp=/dev/lpt0:sd=/usr/lpdspool/nec-raw:\
  :lf=/var/log/lpd-errs:sh:mx#0:


#

nec-ps-banner|NEC Silentwriter 95 with PostScript banner page created:\
  :lp=/dev/lpt0:sd=/usr/lpdspool/nec-ps-banner:\
  :lf=/var/log/lpd-errs:sh:mx#0:if=/usr/local/libexec/psbanner:


#
#
```

Example 8-2. `/usr/local/etc/smb.conf`

```
[global]
comment = FreeBSD - Samba %v
log file = /var/log/samba.log
dont descend = /dev,/proc,/root,/stand
print command = lpr -s -P %p %s; rm %s
interfaces = X.X.X.X (the system IP number goes here)
printing = bsd
map archive = no
status = yes
public = yes
read only = no
preserve case = yes
strip dot = yes
security = share
guest ok = no
```

```
password level = 1
dead time = 15
domain master = yes
workgroup = WORKGROUP

[homes]
browseable = no
comment = User Home Directory
create mode = 0775
public = no

[printers]
path = /var/spool
comment = Printers
create mode = 0700
browseable = no
read only = yes
public = no

[lp]
printable = yes
browseable = no

[nec-raw]
comment = Main PostScript printer driver for Windows clients
printer driver = NEC SilentWriter 95
printable = yes
browseable = yes

[wwwroot]
path = /usr/local/www
read only = no
create mode = 0775
comment = Internal Web Server
```

8.9.3 Browsing output

Following is the output of a **net view** command executed at a DOS prompt under Windows 95:

```
Shared resources at \\SERVER

Sharename Type    Comment
---------------------------------------------------------------
nec-crlf  Print   NEC Silentwriter 95 in ASCII mode
nec-raw   Print   Main PostScript printer driver
tedm      Disk    User Home Directory
wwwroot   Disk    Internal Web Server

The command was completed successfully.
```

In the /etc/printcap file four print queues are defined, all tied to the printer plugged into the parallel port on the FreeBSD server. The first is lp, the generic local line printer. Since this print queue generally has a filter placed on it

to format jobs from the UNIX print queue properly, it should not be visible on the SMB network (i.e., visible in Network Neighborhood). The second queue, `nec-crlf`, has a filter that converts UNIX text to text that prints without stairstepping, so it also should be hidden from the SMB network. The third, `nec-raw`, should be visible on the network because this is the spool that the Windows clients use. The last queue, `nec-ps-banner`, is another specialty queue for UNIX local printing and thus should not be visible.

When the `smb.conf` file is parsed, the default entry `[printers]` is first read and used as a set of defaults for printers that are going to be shared out. Next, the `/etc/printcap` file is read to get a list of all printers on the server. Last, each printer is checked for a service name in the `smb.conf` file that contains settings that override the set of defaults.

In the listing of what resources are visible on the network, both `nec-crlf` and `nec-raw` print queues are visible, and `lp` and `nec-ps-banner` is not. `lp` is not visible because there is a specific entry, `[lp]` in the `smb.conf` file that blocks it. `nec-ps-banner` doesn't have such an entry, but because the print queue name is not a legal length for a SMB name, it isn't shared out either.

The `nec-crlf` printer is visible so as to illustrate another point - comments. If a print queue has no entry in the `smb.conf` file and is built by scanning the `/etc/printcap` file and using the `[printers]` defaults, the comment is taken from the `/etc/printcap` file next to the queue definition name. Otherwise, if an entry is made for the printer in the `smb.conf` file the comment is taken from the entry in `smb.conf`.

8.10 Printing between NT Server/NetWare and FreeBSD.

Up to this point in the chapter, our main concern has been FreeBSD and Windows NT printing interoperability with NT as a print client passing jobs to the FreeBSD system. What happens if the situation is reversed and the FreeBSD system is itself a printing client of another LPD server? This situation can arise in a mixed UNIX/NetWare or UNIX/NT environment. The administrator may elect to forgo the use of Samba, and use an NT server to provide print services. Alternatively, the administrator may have existing DOS Novell IPX clients that they don't want to change, printing to an existing IPX Novell NetWare server. Many of the earlier hardware print servers, such as the Intel NetPort 1 and NetPort 2 were IPX only. A site with a large number of these hardware servers may wish to move the clients to TCP/IP, but leave the existing IPX-based printing network intact.

With NetWare it is possible to load an LPD NetWare loadable module (NLM) on the NetWare server that takes incoming LPR print jobs and prints them on IPX print queues. Later versions of NetWare may include this NLM, it was an extra cost add-on with NetWare 3.X

With Windows NT Server, loading the TCP/IP LPR printing support also loads the LPD print server on NT. By using LPR client programs on UNIX, it is possible to submit, view status, and remove jobs remotely from an NT server that has LPR installed as a port for its printers.

Following is a sample `/etc/printcap` file entry that defines a print queue named `tank` on the FreeBSD system pointed to an NT LPD server queue named `sherman` on a NT Server named `big.army.mil` in the DNS. This uses the `rm` printcap capability. Unlike the earlier examples, the output print jobs are sent out not by the PC parallel port but over the network to the NT server.

```
#
tank|sample remote printer:\
  :rm=big.army.mil:rp=sherman:sd=/var/spool/output/lphost:\
  :lf=/var/log/lpd-errs:
#
```

Note: When using an NT server as an LPD server it may be necessary to make the NT registry changes
mentioned under Windows NT Registry Changes, earlier in the chapter.

8.11 Printing from UNIX

Two commands used at the FreeBSD command prompt are intended as general-purpose print commands: `lp` and
`lpr`.

8.11.1 `lp`

The `lp` command is simply a front end command that calls the `lpr` command with appropriate options. Its main use
is to allow the running of precompiled binary programs and scripts that assume that the `lp` command is the *official*
printing command.

8.11.2 `lpr`

The `lpr` command is the main command that is used to print files from the command prompts under the FreeBSD
operating system. It is frequently spawned off as a child program, or used in pipes. For example, when the Netscape
web browser's Print button is clicked, Netscape may create the PostScript output, but the output goes through the
`lpr` command.

The `lpr` command, like many UNIX command-line printing programs, assumes that the default print queue name is
`lp`. When the FreeBSD machine is set up, the administrator usually sets the `lp` queue to print through a filter that
allows raw UNIX text sent to it to print properly. For example, if an HP LaserJet printer that doesn't have PostScript
is connected to the server, the `lp` queue specifies in the `/etc/printcap` file the CRLF filter listed earlier. On the
other hand, if an Apple Laserwriter that doesn't support ASCII is connected to the server, the `a2ps`filter would be
specified in the `/etc/printcap` for the `lp` queue.

When printing raw text files usually the `-p` option is specified to `lpr`. When printing preformatted files, such as
PostScript files, the `-P` option is used, which selects whatever queue is used to handle these job types.

8.11.3 Managing the UNIX Print Queue

Once the print jobs coming in from clients are received on the FreeBSD system and placed in the print spool, they
are metered out at a slower rate to the various printers. If traffic activity is light, and few print jobs get sent through,
the administrator can probably ignore the print queue as long as it continues to work. However, a busy network
printer running at an optimal rate of speed usually has a backlog of unprinted jobs in the queue waiting for print time.
To keep all users happy and to provide for the occasional rush print job, the UNIX LPD/LPR printing system has
several administration commands which are described here.

8.11.3.1 Viewing the queue

On busy printers, and to troubleshoot stopped printers, users sometimes need to view the print jobs in the queue.
Administrators also need to view the queue to see what jobs may need to be expedited. This can be done from the
workstation that remotely submitted the job if the LPR client has the ability to do this. The Windows 3.1 LPR client

discussed earlier has this capability. Unfortunately, many LPR clients don't, which means that the administrator must Telnet into the UNIX machine that the print queues are on and view them there.

The UNIX shell command used to view the queue is the `lpq` command. It is frequently run as **`lpq -a`** which shows jobs in all queues. The following is a sample output of the command:

```
# lpq -a
nec-raw:
Rank Owner Job Files                     Total Size
1st  tedm  19  C:/WLPRSPL/SPOOL/~LP00018.TMP 105221 bytes
2nd  tedm  20  C:/WLPRSPL/SPOOL/~LP00019.TMP 13488 bytes
3rd  root  3   hosts                     1220 bytes
4th  tedm  1   Printer Test Page         765 bytes
5th  tedm  2   Microsoft Word - CHAPTE10.DOC 15411 bytes
```

The first two jobs and the last two jobs came from remote clients, the third came from the command prompt.

8.11.3.2 Removing print jobs

Deleting unwanted print jobs that haven't yet printed from the queue can be done by the remote workstations that submitted the job if their LPR implementations have the necessary commands. The Windows 3.1 LPR client I detailed earlier has this capability. Many LPR clients don't, however, which means that the administrator must Telnet into the UNIX machine that the print queues are on and delete the jobs there.

The administrator can delete any print jobs from any queues by running the `lprm` command followed by the specified print queue and the job number. Below is a sample output of the command:

```
# lprm -P nec-raw 19
dfA019tedmitte dequeued
cfA019dostest dequeued
# lprm -P nec-raw 3
dfA003toybox.placo.com dequeued
cfA003toybox.placo.com dequeued
```

The `lprm` command is also used under UNIX to delete remote print jobs.

8.11.3.3 Advanced management

The administrator logged into the FreeBSD system as the root user can also perform several other operations that ordinary users cannot. These include turning the queues on and off, and moving print jobs within the print queues. The command used to do this is the `lpc` command.

`lpc` has two modes of operation. In the first mode, the command is run by itself, which puts the administrator into an `lpc` prompt. Some general help is available for the commands, such as the following sample output:

```
# lpc
lpc> help
Commands may be abbreviated.
Commands are:
abort enable disable help restart status topq ?
clean exit down quit start stop up
lpc> help disable
```

```
disable turn a spooling queue off
lpc> help status
status show status of daemon and queue
lpc> exit
```

In the second mode of operation the lpc command is just run by itself, followed by the command and the print queue name. Following is a sample output:

```
# lpc disable lp
lp:
queuing disabled
```

Under FreeBSD, there is no command that specifically allows the administrator to move jobs from one queue to another. This can be done, however, by changing into the raw queue directory then rerunning the lpr command. Following is a sample run showing three print jobs moved from a dysfunctional queue to a good one:

```
# lpq -a
lp:
Warning: lp is down: printing disabled
printing disabled
Rank Owner Job Files                      Total Size
1st  root  51  hosts                      1220 bytes
2nd  root  52  services                   60767 bytes
3rd  root  53  printcap                   2383 bytes
# cd /var/spool/output/lpd
# ls
.seq cfA053toybox.placo.com dfA053toybox.placo.com
cfA051toybox.placo.com dfA051toybox.placo.com lock
cfA052toybox.placo.com dfA052toybox.placo.com status
# lpr -P nec-raw dfA051toybox.placo.com
# lpr -P nec-raw dfA052toybox.placo.com
# lpr -P nec-raw dfA053toybox.placo.com
# lprm -P lp -
# lpq -a
nec-raw:
Warning: nec-raw is down: printing disabled
Warning: no daemon present
Rank Owner Job Files                      Total Size
1st  root  5   dfA051toybox.placo.com     1220 bytes
2nd  root  6   dfA052toybox.placo.com     60767 bytes
3rd  root  7   dfA053toybox.placo.com     2383 bytes
```

Note: Moving jobs from queue to queue is feasible only when all printers are similar, as when all printers support PostScript.

8.11.3.4 Remote Management

Just as the root user can manipulate remotely submitted jobs in the print queue, print jobs can be remotely managed by regular users with the LPR clients that created them. Unfortunately, some LPR clients, such as the ACITS LPR client for Win95, don't have enough programming to be able to do this. Others, like the Win31 client, can manipulate the print jobs remotely.

FreeBSD offers some level of protection against inadvertent deletion of print jobs from remote hosts by restricting manipulation of a job to the same host that originated it. Even if the owner of the job matches a local user account on the server, for an ordinary user to delete remotely submitted print jobs, the request still must come from the remote host.

8.12 Advanced Printing Topics

The FreeBSD UNIX LPR/LPD printing system is very flexible, and, with the addition of filters, can be adapted to very unusual printing environments. To enhance this flexibility, several useful printing utilities are supplied on the FreeBSD CDROM which the administrator might wish to install.

8.12.1 Ghostscript

The Ghostscript program, invoked as `/usr/local/bin/gs`, is one of the most useful printing utilities that have been developed for the free software community. Ghostscript reads incoming PostScript data, (or Adobe PDF files) interprets it, and outputs it as a raster image. This can be displayed on screen, for example, with the GhostView program under the X Window system, or printed on most graphics printers, such as Epson dot-matrix, HP DeskJet, or HP LaserJet. In effect, it is a way of adding PostScript printing capability to a printer that doesn't have PostScript firmware code. Ghostscript has been ported to numerous operating systems including Windows.

The Ghostscript home page is located at http://www.cs.wisc.edu/~ghost/ and contains the most current version of the program. A prebuilt FreeBSD binary of Ghostscript is located in the Packages section of the FreeBSD CDROM. This can be installed on the FreeBSD system by selecting the package from the prepackaged software list that is accessed through the `/stand/sysinstall` installation program. Many packaged programs on the CD depend on GhostScript, and so it may already be installed.

Installation of the packaged version of GhostScript is recommended in the FreeBSD ports Section because it has been tested with the other packages that require it. The package creates a directory containing some documentation files in `/usr/local/share/ghostscript/X.XX/doc`. Unfortunately, because of the packaging process on the FreeBSD CDROM not all the useful installation files are copied into this location. So, if the package was version 5.03 (for example) the administrator will also want to get the file ftp://ftp.cs.wisc.edu/ghost/aladdin/gs502/ghostscript-5.02.tar.gz, and unzip and untar it into a temporary directory.

Extracting the archive file creates a directory structure under the `gs5.03` subdirectory. To install ghostscript in the `/etc/printcap` file, read the `gs5.03/devs.mak` file to determine which printer driver definition works with your printer and then use the following instructions:

1. Change to the root user with `su`.

2. In the `gs5.03` directory, copy the `lprsetup.sh`, `unix-lpr.txt`, and `unix-lpr.sh` files to `/usr/local/share/ghostscript/5.03`.

3. Change to the `/usr/local/share/ghostscript/5.03` directory. Edit `lprsetup.sh` with a text editor such as `vi`.

4. Modify the `DEVICES=` entries to list your selected printer driver definitions per the instructions in `unix-lpr.txt`.

5. Modify the `PRINTERDEV=` to `/dev/lpt0`, and the `GSDIR=` to `/usr/local/share/ghostscript`, and the `SPOOLDIR=` to `/var/spool/output`. Save the file.

6. Edit the `unix-lpr.sh` file and change the `PSFILTERPATH=` to `/usr/local/share/ghostscript`.

7. If the printer that you defined in the `lprsetup.sh` file is a monochrome printer, remove the `"-dBitsPerPixel=${bpp}"` and `"$colorspec"` entries on the `gs` invocation line and save the file. Otherwise, if it is a color definition leave them in. For example, the following line is for a monochrome LaserJet:

   `") | gs -q -dNOPAUSE -sDEVICE=${device} \"`

   Don't remove anything else. Exit the editor, and save the `unix-lpr.sh` file.

8. Copy the `unix-lpr.sh` file to the parent directory, `/usr/local/share/ghostscript` and set the execute bit on it.

9. Set the execute bit on `lprsetup.sh` with chmod and run the file by typing **`./lprsetup.sh`**.

10. Follow the instructions on creating the Spool directories. If you will be using accounting and a separate log file, run the `touch` command to create the empty files per directions in script output.

11. The sample `/etc/printcap` is located in the current directory; the filename is `printcap.insert`. Use this as a template to modify the `/etc/printcap` file. A sample `/etc/printcap` file for a LaserJet 3 is below:

```
#
#
ljet3.raw|Raw output device ljet3 for Ghostscript:\
  :rm=big.army.mil:rp=sherman:sd=/var/spool/output/ljet3/raw:\
  :mx#0:sf:sh:rs:

#

ljet3|Ghostscript device ljet3 (output to ljet3.raw):\
  :lp=/dev/null:sd=/var/spool/output/ljet3:\
  :lf=/var/log/lpd-errs:mx#0:sf:sh:rs:\
  :if=/usr/local/share/ghostscript/filt/indirect/ljet3/gsif:\
  :af=/var/spool/output/ljet3/acct:

#
```

8.12.2 a2ps filter

Another handy utility is the `a2ps` filter, short for ASCII-to-PostScript. This program takes an incoming ASCII datastream and converts it into PostScript. It can also print multiple pages on a single sheet of paper by shrinking them down. It is a useful tool for a printer that cannot interpret ASCII, such as a PostScript-only printer.

A2ps is not installed in the FreeBSD system by default; it is located in the ports section `/usr/ports/print/a2ps43`. A prepackaged binary can be installed with `/stand/sysinstall` but I have had problems with that port. It is best to install it by running `make` in the a2ps43 ports directory. A printcap entry and filter using this follow:

Example 8-3. `/etc/printcap`

```
#
lp|local line printer with output dumped through a2ps for raw listings:\
   :lp=/dev/lpt0:sd=/var/spool/output/lpd:lf=/var/log/lpd-errs:sh:mx#0:\
   :if=/usr/local/libexec/ascii2postscript:
#
```

Example 8-4. `/usr/local/libexec/ascii2postscript`

```
#!/bin/sh
#
# Simple filter that converts ASCII to PostScript for basic stuff like
# directory listings.
#

/usr/local/bin/a2ps && exit 0

exit 2
```

Read the system manual page for `a2ps` to see the options available with this program, and remember to set the filter script `ascii2postscript` all-executable.

8.13 Miscellaneous

The large number of other printing utilities cannot be covered here. Some add features such as automatic job type sensing, others handle bidirectional communication between the server and the printer. There are also a few other experimental LPR printing replacement systems. Commands such as ghostscript and `a2ps` can also be used in pipes that create pretty output on an ordinary impact printer.

One last hint - the system manual pages can be printed with the `-t` option which turns their ordinary ASCII output to beautifully formatted PostScript. Try the command `man -t man` and send the output through GhostScript or a PostScript printer for easier to read manual pages.