LP Administrator's Guide

J. R. Kliegman

Bell Laboratories
Piscataway, New Jersey 08854

1. INTRODUCTION

LP is a system of commands that performs diverse spooling functions under the UNIX† operating system. Because its primary application is off-line printing, this paper focuses mainly on spooling to line printers. LP allows administrators to customize the system to spool to a collection of line printers of any type and to group printers into logical classes in order to maximize the throughput of the devices. Users are provided the capabilities of queuing and canceling print requests, preventing and allowing queuing to and printing on devices, starting and stopping LP from processing requests, changing their configuration of printers and finding the status of the LP system. This memo describes the role of an LP Administrator (LPA) in performing restricted functions and overseeing the smooth operation of LP.

The remainder of this paper is organized as follows: Section 2 presents an overview of the features of LP and defines terms that will be used throughout the memo. See [1] for a detailed description of the implementation of LP. Section 3 tells how to build an LP system. Sections 4-11 describe how to perform administrative functions using LP commands. Section 12 covers how to write printer interface programs, Section 13 indicates how to set up hardwired printers and login terminals to be used with LP and the final section summarizes the role of the LPA.

2. OVERVIEW OF LP FEATURES

2.1 Definitions

We will define several terms before presenting a brief summary of LP commands. LP was designed with the flexibility to meet the needs of users on different UNIX systems. Changes to LP's configuration (see below) are performed by the *lpadmin*(1M) command. (A parenthesized number immediately following a command name refers to that section of the *UNIX User's Manual.*)

LP makes a distinction between printers and printing devices. A *device* is a physical peripheral device or a file and is represented by a full UNIX path name. A *printer* is a logical name that represents a device. At different points in time, a printer may be associated with different devices. A *class* is a name given to an ordered list of printers. Every class must contain at least one printer. Each printer may be a member of zero or more classes. A *destination* is a printer or a class. One destination may be designated as the *system default destination*. The *lp*(1) command will direct all output to this destination unless the user specifies otherwise. Output that is routed to a printer will be printed only by that printer, whereas output directed to a class will be printed by the first available class member.

Each invocation of *lp* creates an output *request* that consists of the files to be printed and options from the *lp* command line. An *interface program* which formats requests must be supplied for each printer. The LP scheduler, *lpsched*(1M), services requests for all destinations by routing requests to interface programs to do the printing on devices. An LP *configuration* for a system consists of devices, destinations and interface programs.

† UNIX is a trademark of Bell Laboratories.

2.2 Commands

2.2.1 Commands for General Use

Lp(1) is used to request the printing of files. It creates an output request and returns a *request id* of the form:

```
    dest−seqno
```

to the user, where *seqno* is a unique sequence number across the entire LP system and *dest* is the destination where the request was routed.

Cancel is used to cancel output requests. The user supplies request ids as returned by *lp* or printer names, in which case the currently printing requests on those printers are canceled.

Disable prevents *lpsched* from routing output requests to printers.

Enable(1) allows *lpsched* to route output requests to printers.

2.2.2 Commands for LP Administrators

Each LP system must designate a person or persons as LP Administrator to perform the restricted functions listed below. Either the super-user or any user who is logged into UNIX as **lp** qualifies as an LP Administrator. All LP files and commands are owned by lp, except for *lpadmin* and *lpsched*, which are owned by root. The following commands will be described in more detail later in this memo.

Lpadmin(1M) modifies the LP configuration. Many features of this command cannot be used when *lpsched* is running.

Lpsched(1M) routes output requests to interface programs which do the printing on devices.

Lpshut stops *lpsched* from running. All printing activity is halted, but the other LP commands may still be used.

Accept(1M) allows *lp* to accept output requests for destinations.

Reject prevents *lp* from accepting requests for destinations.

Lpmove moves output requests from one destination to another. Whole destinations may be moved at once. This command cannot be used when *lpsched* is running.

3. BUILDING LP

All LP commands are built from source code that resides in the **/usr/src/cmd/lp** directory including the make file, **lp.mk**. Unless some of the definitions in **lp.mk** are changed, LP may be installed only by the super-user. Before installing a new LP system, make sure there is a login called **lp** on your system and that the spool directory, **/usr/spool/lp**, does not exist. To install LP, perform the following:

```
    cd /usr/src/cmd/lp
    make −f lp.mk install
```

This builds all LP commands and creates an initial LP configuration consisting of no printers, classes or default destination. LP must be configured by an LPA using the *lpadmin* command in order to create a useful spooler.

In addition, add the following code to /etc/rc:

```
    rm −f /usr/spool/lp/SCHEDLOCK
    /usr/lib/lpsched
    echo "LP scheduler started"
```

This starts the LP scheduler each time that UNIX is restarted.

Several variables in **lp.mk** may be changed before installing LP to customize the system:

Variable	*Default Value*	*Meaning*
SPOOL	/usr/spool/lp	spool directory
ADMIN	lp	logname of LP Administrator
GROUP	bin	group that owns LP commands and data
ADMDIR	/usr/lib	administrator commands reside here
USRDIR	/usr/bin	user commands reside here

If an existing LP spool directory is corrupted (but not the LP programs) or if it needs to be rebuilt from scratch, make sure that *lpsched* is not running and perform the following as super-user:

1. Make copies of any interface programs that are not standard LP software. DO NOT make these copies underneath the spool directory. The path name for printer p is **/usr/spool/lp/interface/p**.
2. rm −fr /usr/spool/lp
3. make −f lp.mk new   (this recreates the bare LP configuration described above).

WARNINGS:

1. Some LP commands invoke other LP commands. Moving them after they are built will cause some commands to fail.
2. The files under the SPOOL directory should be modified *only by LP commands*.
3. All LP commands require set-user-id permission. If this is removed, the commands will fail.

4. CONFIGURING LP − THE LPADMIN COMMAND

Changes to the LP configuration should be made by using the *lpadmin* command and not by hand. *Lpadmin* will not attempt to alter the LP configuration when *lpsched* is running, except where explicitly noted below.

4.1 Introducing New Destinations

The following information must be supplied to *lpadmin* when introducing a new printer:

1. The printer name (−pprinter) is an arbitrary name which must conform to the following rules:
   - It must be no longer than fourteen characters.
   - It must consist solely of alphanumeric characters and underscores.
   - It must not be the name of an existing LP destination (printer or class).
2. The device associated with the printer (−vdevice). This is the path name of a hardwired printer, a login terminal, or other file that is writable by lp.
3. The printer interface program. This may be specified in one of three ways:
   - It may be selected from a list of model interfaces supplied with LP (−mmodel).
   - It may be the same interface that an existing printer uses (−eprinter).
   - It may be a program supplied by the LPA (−iinterface).

Information that need not always be supplied when creating a new printer includes:

1. The user may specify **−h** to indicate that the device for the printer is hardwired or the device is the name of a file (this is assumed by default). If, on the other hand, the device is the path name of a login terminal, then **−l** must be included on the command line. This indicates to *lpsched* that it must automatically disable this printer each time *lpsched* starts running. This fact is reported by *lpstat* when it indicates printer status:

   ```
   $ lpstat −pa
   printer a (login terminal) disabled since Oct 31 11:15 −
            disabled by scheduler: login terminal
   ```

   This is done because device names for login terminals can be (and usually are) associated with different physical devices from day to day. If the scheduler did not take this action, somebody might log in and be surprised that LP is spooling to his/her terminal!

2. The new printer may be added to an existing class or added to a new class (**−c**class). New class names must conform to the same rules for new printer names.

Examples:

The following examples will be referenced by further examples in later sections:

1. Create a printer called pr1 whose device is /dev/printer and whose interface program is the model hp interface:

   ```
   $ /usr/lib/lpadmin −ppr1 −v/dev/printer −mhp
   ```

2. Add a printer called pr2 whose device is /dev/tty22 and whose interface is a variation of the model prx interface. It is also a login terminal:

   ```
   $ cp /usr/spool/lp/model/prx xxx
      < edit xxx here >
   $ /usr/lib/lpadmin −ppr2 −v/dev/tty22 −ixxx −l
   ```

3. Create a printer called pr3 whose device is /dev/tty23. pr3 will be added to a new class called cl1 and will use the same interface as printer pr2:

   ```
   $ /usr/lib/lpadmin −ppr3 −v/dev/tty23 −epr2 −ccl1
   ```

4.2 Modifying Existing Destinations

Modifications to existing destinations must always be made with respect to a printer name (**−p**printer). The modifications may be one or more of the following:

1. The device for the printer may be changed (**−v**device). If this is the only modification, than this may be done even while *lpsched* is running. This facilitates changing devices for login terminals.

2. The printer interface program may be changed (**−m**model, **−e**printer, **−i**interface).

3. The printer may be specified as hardwired (**−h**) or as a login terminal (**−l**).

4. The printer may be added to a new or existing class (**−c**class).

5. The printer may be removed from an existing class (**−r**class). Removing the last remaining member of a class causes the class to be deleted. No destination may be removed if it has pending requests. In that case, *lpmove* or *cancel* should be used to move or delete the pending requests.

Examples:

These examples are based on the LP configuration created by those in the previous section.

1. Add printer pr2 to class cl1:

```
$ /usr/lib/lpadmin −ppr2 −ccl1
```

2. Change pr2's interface program to the model prx interface, change its device to /dev/tty24, and add it to a new class called cl2:

```
$ /usr/lib/lpadmin −ppr2 −mprx −v/dev/tty24 −ccl2
```

   Note that printers pr2 and pr3 now use different interface programs even though pr3 was originally created with the same interface as pr2. Printer pr2 is now a member of two classes.

3. Specify printer pr2 as a hardwired printer:

```
$ /usr/lib/lpadmin −ppr2 −h
```

4. Add printer pr1 to class cl2:

```
$ /usr/lib/lpadmin −ppr1 −ccl2
```

   The members of class cl2 are now pr2 and pr1, in that order. Requests routed to class cl2 will be serviced by pr2 if both pr2 and pr1 are ready to print, otherwise they will be printed by the one which is next ready to print.

5. Remove printers pr2 and pr3 from class cl1:

```
$ /usr/lib/lpadmin −ppr2 −rcl1
$ /usr/lib/lpadmin −ppr3 −rcl1
```

   Because pr3 was the last remaining member of class cl1, the class is removed.

6. Add pr3 to a new class called cl3:

```
$ /usr/lib/lpadmin −ppr3 −ccl3
```

4.3 Specifying the System Default Destination

The system default destination may be changed even when *lpsched* is running.

Examples:

1. Establish class cl1 as the system default destination:

```
$ /usr/lib/lpadmin −dcl1
```

2. Establish no default destination:

```
$ /usr/lib/lpadmin −d
```

4.4 Removing Destinations

Classes and printers may be removed only if there are no pending requests that were routed to them. Pending requests must either be canceled using *cancel* or moved to other destinations using *lpmove* before destinations may be removed. If the removed destination is the system default destination, then the system will have no default destination until it respecified. When the last remaining member of a class is removed, then the class is also removed. The removal of a class never implies the removal of printers.

Examples:

1. Make printer pr1 the system default destination:

   ```
   $ /usr/lib/lpadmin −dpr1
   ```

   Remove printer pr1:

   ```
   $ /usr/lib/lpadmin −xpr1
   ```

   Now there is no system default destination.

2. Remove printer pr2:

   ```
   $ /usr/lib/lpadmin −xpr2
   ```

   Class cl2 is also removed, because pr2 was its only member.

3. Remove class cl3:

   ```
   $ /usr/lib/lpadmin −xcl3
   ```

   Class cl3 is removed, but printer pr3 remains.

5. MAKING AN OUTPUT REQUEST − THE LP COMMAND

Once LP destinations have been created, users may request output by using the *lp* command. The request id that is returned may be used to see if the request has been printed or to cancel the request.

Lp determines the destination of a request by checking the following list in order:

- If the user specifies **−d***dest* on the command line, then the request is routed to *dest*.
- If the environment variable **LPDEST** is set, the request is routed to the value of *LPDEST*.
- If there is a system default destination, then the request is routed there.
- Otherwise, the request is rejected.

Examples:

1. There are at least four ways to print the password file on the system default destination:

   ```
   lp /etc/passwd
   lp < /etc/passwd
   cat /etc/passwd | lp
   lp −c /etc/passwd
   ```

   The last three ways cause copies of the file to be printed, whereas the first way prints the file directly. Thus, if the file is modified between the time the request is made and the time it is actually printed, then the changes will be reflected in the output.

2. Print two copies of file abc on printer xyz and title the output "my file":

   ```
   pr abc | lp −dxyz −n2 −t"my file"
   ```

3. Print file xxx on a Diablo 1640 printer called zoo in 12-pitch and write to the user's terminal when printing has completed:

   ```
   lp −dzoo −o12 −w xxx
   ```

   In this example, **12** is an option that is meaningful to the model Diablo 1640 interface program that prints output in 12-pitch mode (see *lpadmin*(1M)).

6. FINDING LP STATUS — LPSTAT

The *lpstat* command is used to find status information about LP requests, destinations and the scheduler.

Examples:

1. List the status of all pending output requests made by this user:

   ```
   lpstat
   ```

   The status information for a request includes the request id, the logname of the user, the total number of characters to be printed and the date and time the request was made.

2. List the status of printers p1 and p2:

   ```
   lpstat —pp1,p2
   ```

7. CANCELING REQUESTS — CANCEL

LP requests may be canceled using the *cancel* command. Two kinds of arguments may be given to the command — request ids and printer names. The requests named by the request ids are canceled and requests that are currently printing on the named printers are canceled. Both types of arguments may be intermixed.

Example:

Cancel the request that is now printing on printer xyz:

```
cancel xyz
```

If the user that is canceling a request is not the same one that made the request, then mail is sent to the owner of the request. LP allows *any* user to cancel requests in order to eliminate the need to find LP Administrators when unwanted output is be purged.

8. ALLOWING AND REFUSING REQUESTS — ACCEPT AND REJECT

When a new destination is created, *lp* will reject requests that are routed to it. When the LP Administrator is sure that it is set up correctly he or she should allow *lp* to accept requests for that destination. The *accept* command performs this function.

Sometimes it is necessary to prevent *lp* from routing requests to destinations. If printers have been removed or are waiting to be repaired or if too many requests are building for printers then it may be desirable to cause *lp* to reject requests for those destinations. The *reject* command performs this function. After the condition that led to the rejection of requests has been remedied, the *accept* command should be used to allow requests to be taken again.

The acceptance status of destinations is reported by the **—a** option of *lpstat*.

Examples:

1. Cause *lp* to reject requests for destination xyz:

   ```
   /usr/lib/reject —r"printer xyz in need of repair" xyz
   ```

   Any users that try to route requests to xyz will encounter the following:

   ```
   $ lp —dxyz file
   lp: can't accept requests for destination "xyz" —
             printer xyz in need of repair
   ```

2. Allow *lp* to accept requests routed to destination xyz:

   ```
   /usr/lib/accept xyz
   ```

9. ALLOWING AND INHIBITING PRINTING — ENABLE AND DISABLE

The *enable* command allows the LP scheduler to print requests on printers. That is, the scheduler routes requests only to the interface programs of enabled printers. Note that it is possible to enable a printer but to prevent further requests from being routed to it.

The *disable* command undoes the effects of the *enable* command. It prevents the scheduler from routing requests to printers, independently of whether or not *lp* is allowing them to accept requests. Printers may be disabled for several reasons including malfunctioning hardware, paper jams and end of day shutdowns. If a printer is busy at the time it is disabled, then the request that it was printing will be reprinted in its entirety either on another printer (if the request was originally routed to a class of printers) or on the same one when the printer is re-enabled. The −c option causes the currently printing requests on busy printers to be canceled in addition to disabling the printers. This is useful if strange output is causing a printer to behave abnormally.

Example:

Disable printer xyz because of a paper jam:

```
$ disable −r"paper jam" xyz
printer "xyz" now disabled
```

Find the status of printer xyz:

```
$ lpstat −pxyz
printer "xyz" disabled since Jan 5 10:15 −
        paper jam
```

Now, re-enable xyz:

```
$ enable xyz
printer "xyz" now enabled
```

10. MOVING REQUESTS BETWEEN DESTINATIONS — LPMOVE

Occasionally, it is useful for LP Administrators to move output requests between destinations. For instance, when a printer is down for repairs it may be desirable to move all of its pending requests to a working printer. This is one way to use the *lpmove* command. The other use of this command is to move specific requests to a different destination. *Lpmove* will refuse to move requests while the LP scheduler is running.

Examples:

1. Move all requests for printer abc to printer xyz:

   ```
   $ /usr/lib/lpmove abc xyz
   ```

   All of the moved requests are renamed from abc−nnn to xyz−nnn. As a side effect, destination abc is no longer accepting further requests.

2. Move requests zoo−543 and abc−1200 to printer xyz:

   ```
   $ /usr/lib/lpmove zoo−543 abc−1200 xyz
   ```

   The two requests are now renamed xyz−543 and xyz−1200.

11. STOPPING AND STARTING THE SCHEDULER — LPSHUT AND LPSCHED

Lpsched is the program that routes the output requests that were made with *lp* through the appropriate printer interface programs to be printed on line printers. Each time the scheduler routes a request to an interface program, it records an entry in the log file, **/usr/spool/lp/log**. This entry contains the logname of the user who made the request, the request id, the name of the printer that the request is being printed on and the date and time that printing first started. In the case that a request has been restarted, more than one entry in the log file may refer to the request. The scheduler also records error messages in the log file. When *lpsched* is started, it renames **/usr/spool/lp/log** to **/usr/spool/lp/oldlog** and starts a new log file.

No printing will be performed by the LP system unless *lpsched* is running. Use the command:

```
lpstat —r
```

to find the status of the LP scheduler.

Lpsched is normally started by the **/etc/rc** program as described above and continues to run until UNIX is shut down. The scheduler operates in the **/usr/spool/lp** directory. When it starts running, it will exit immediately if a file called **SCHEDLOCK** exists. Otherwise, it creates this file in order to prevent more than one scheduler from running at the same time.

Occasionally, it is necessary to shut the scheduler in order to reconfigure LP or to rebuild the LP software. The command

```
/usr/lib/lpshut
```

causes *lpsched* to stop running and terminates all printing activity. All requests that were in the middle of printing will be reprinted in their entirety when the scheduler is restarted.

To restart the LP scheduler, use the command

```
/usr/lib/lpsched
```

Shortly after this command is entered, *lpstat* should report that the scheduler is running. If not, it is possible that a previous invocation of *lpsched* exited without removing **SCHEDLOCK**, so try the following:

```
rm —f /usr/spool/lp/SCHEDLOCK
/usr/lib/lpsched
```

The scheduler should be running now.

12. PRINTER INTERFACE PROGRAMS

Every LP printer must have an interface program which does the actual printing on the device that is currently associated with the printer. Interface programs may be shell procedures, C programs, or any other executable programs. LP's model interfaces are all written as shell procedures and can be found in the **/usr/spool/lp/model** directory. At the time *lpsched* routes an output request to a printer P, the interface program for P is invoked in the directory **/usr/spool/lp** as follows:

```
interface/P id user title copies options file ...
```

where

id is the request id returned by *lp*
user is the logname of the user who made the request
title is the optional title specified by the user
copies is the number of copies requested by the user
options is a blank-separated list of class- or printer-dependent options specified by the user
file is the full path name of a file to be printed

Examples:

The following examples are requests made by user "smith" with a system default destination of printer "xyz". Each example lists an *lp* command line, followed by the corresponding command line generated for printer xyz's interface program:

1. lp /etc/passwd /etc/group
   interface/xyz xyz−52 smith "" 1 "" /etc/passwd /etc/group
2. pr /etc/passwd | lp −t"users" −n5
   interface/xyz xyz−53 smith users 5 "" /usr/spool/lp/request/xyz/d0−53
3. lp /etc/passwd −oa −ob
   interface/xyz xyz−54 smith "" 1 "a b" /etc/passwd

When the interface program is invoked, its standard input comes from /dev/null and both the standard output and standard error output are directed to the printer's device. Devices are opened for reading as well as writing when file modes permit. In the case where a device is a regular file, all output is appended to the end of the file.

Given the command line arguments and the output directed to a device, interface programs may format their output in any way they choose. Interface programs must ensure that the proper stty modes (terminal characteristics such as baud rate, output options, etc.) are in effect on the output device. This may be done as follows in a shell interface only if the device is opened for reading:

```
stty mode ... <&1
```

That is, take the standard input for the stty command from the device.

When printing has completed, it is the responsibility of the interface program to exit with a code indicative of the success of the print job. Exit codes are interpreted by *lpsched* as follows:

CODE	*MEANING TO LPSCHED*
zero	The print job has completed successfully.
1 to 127	A problem was encountered in printing this particular request (e.g., too many non-printable characters). This problem won't affect future print jobs. *Lpsched* notifies users by mail that there was an error in printing the request.
greater than 127	These codes are reserved for internal use by *lpsched*. Interface programs must not exit with codes in this range.

When problems that are likely to affect future print jobs occur (e.g., a device filter program is missing), the interface programs would be wise to disable printers so that print requests are not lost. When a busy printer is disabled, the interface program will be terminated with signal 15.

13. SETTING UP HARDWIRED DEVICES AND LOGIN TERMINALS AS LP PRINTERS

13.1 Hardwired Devices

As an example of how to set up a hardwired device for use as an LP printer, let us consider using tty line 15 as printer xyz. As super-user, perform the following:

1. Avoid unwanted output from non-LP processes and ensure that LP can write to the device:

   ```
   $ chown lp /dev/tty15
   $ chmod 600 /dev/tty15
   ```

2. Change /etc/inittab so that tty15 is not a login terminal. In other words, ensure that /etc/getty is not trying to log users in at this terminal. Change the entries for line 15 to:

   ```
   1:15:o:
   2:15:o:
   ```

   Enter the command:

   ```
   $ init 2
   ```

   If there is currently an invocation of /etc/getty running on tty15, then kill it. Now, and when UNIX is rebooted, tty15 will be initialized with default stty modes. Thus, it is up to LP interface programs to establish the proper baud rate and other stty modes for correct printing to occur.

3. As explained above in Section 4.1, introduce printer xyz to LP using the model Printronix interface program:

   ```
   $ /usr/lib/lpadmin −pxyz −v/dev/tty15 −mprx
   ```

4. When xyz is created, it will initially be disabled and *lp* will be rejecting requests routed to it. If it is desired, allow *lp* to accept requests for xyz:

   ```
   /usr/lib/accept xyz
   ```

   This will allow requests to build up for xyz and they will be printed when it is enabled at a later time.

5. When it is desired for printing to occur, be sure that the printer is ready to receive output. For several printers, this means that the top of form has been adjusted and that the printer is on-line. As explained above in Section 9, enable printing to occur on xyz:

   ```
   enable xyz
   ```

   When requests have been routed to xyz, they will begin printing.

13.2 Login Terminals

Login terminals may also be used as LP printers. To do this for a Diablo 1640 terminal called abc, perform the following:

1. As explained above in Section 4.1, introduce printer abc to LP using the model 1640 interface program:

   ```
   $ /usr/lib/lpadmin −pabc −v/dev/null −m1640 −l
   ```

   Note that **/dev/null** is used as abc's device because we will specify the actual device each time that abc is enabled. This device may be different from day to day. When abc is created, it will initially be disabled and *lp* will be rejecting requests routed to it. If it is desired, allow *lp* to accept requests for abc:

   ```
   /usr/lib/accept abc
   ```

   This will allow requests to build up for abc and they will be printed when it is enabled at a later time. It is not advisable to enable abc for printing, however, until the following steps have been taken.

2. Log the terminal in if this has not already been done.

3. Assuming the *tty*(1) command reports that this terminal is /dev/tty02, associate this device with printer abc:

   ```
   $ /usr/lib/lpadmin −pabc −v/dev/tty02
   ```

   Note that *lpadmin* may be used only by an LPA. If it is desired for other users to routinely perform this step, then an LPA may establish a program owned by **lp** or by **root** with set-user-id permission that performs this function.

4. When it is desired for printing to occur, be sure that the printer is ready to receive output. For several printers, this means that the top of form has been adjusted. As explained above in Section 9, enable printing to occur on abc:

   ```
   enable abc
   ```

   When requests have been routed to abc, they will begin printing.

5. When all printing has stopped on abc or when you want it back as a regular login terminal, you may prevent it from printing more output:

   ```
   $ disable abc
   printer "abc" now disabled
   ```

   If abc is enabled when UNIX is rebooted or when *lpsched* is restarted, it will be disabled automatically.

14. SUMMARY

The administrative functions of the LP Administrator have been described in detail. They include configuring and re-configuring LP, maintaining printer interface programs, accepting, rejecting and moving print requests, stopping and starting the LP scheduler and enabling and disabling printers. LP offers administrators the following advantages over other centrally supported printer packages:

- Printers may be grouped into classes.
- LP may be configured to meet the needs of each site.
- Administrators may supply interface programs to format output in any way desirable.
- LP functions are performed by simple commands and not by hand.

REFERENCE

[1] Kliegman, J. R. *The Implementation of the LP Spooling System*, Bell Laboratories.

January 1981