

**Title:** network printing in a heterogeneous environment  
**Authors:** Christoph Beyer, Gerhard Schroth  
**Keywords:** printing, RFC1179, SAMBA, LPRng  
**Abstract:** 6-001 "network printing in heterogeneous environments"  
**Inst./Univ.:** DESY Hamburg

**[Abstract]**

Mail and printing are often said to be the most visible services for the user in the network. Though many people talked about the paperless bureau a few years ago it seems that the more digital data is accessible, the more it gets printed. Print management in a heterogeneous network environments is typically crossing all operating systems. Each of those brings its own requirements and different printing system implementations with individual user interfaces. The scope is to give the user the advantage and features of the native interface of their operating system while making administration tasks as easy as possible by following the general ideas of a centralised network service on the server side.

**[Overview, printing at DESY]**

At DESY Hamburg ~400 queues serve ~280 active printers. Central print service is mainly provided for UNIX (all flavours including LINUX) and Windows (NT / 2000) in addition there is still MACOS and NOVELL with a pending service level and no further development.

The hardware consists of two SUN E250 running SOLARIS2.7 as central spooling hosts and mainly HP (b&w laser) and TEKTRONIX (colour laser) printer.

For historic reasons and due to the non-existence of an integration tool WINDOWS NT clients have been printing to a native NT spooler which was forwarding the jobs to the central UNIX machines. We are about to change this and use SAMBA 2.2 on the UNIX machines instead.

**[Some common strategies on network printing]**

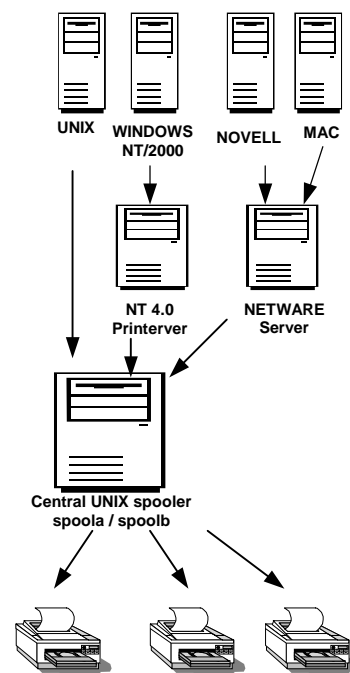
Take advantage of the flexibility of UNIX as a backend for queuing jobs and talking to the printer. Use software to present the UNIX queues to the clients using their native protocols.[1]

We make heavy use of the 'host access lists', access to the printer is only given to the central spooler and a few test machines. Like this every printjob has a fair chance to be printed.

The server should understand every protocol the clients are capable of in order to keep those as 'vanilla' as possible. We prefer to have some strange stuff running on two machines then running it on 2000 clients.

The system as a whole should be robust, printjobs should be redirected to a working machine in case of server failures.

Some nice features like accounting and loadbalancing between a couple of printers make life much easier.



*figure 1*

**[The Berkley print spooler architecture]**

A print spooler is a program that accepts print jobs (which are usually one or more files) from a program or network interface, stores them in a spool queue, and then sends them to a printer or another print spooler. Usually there are facilities to submit jobs, check on the current job status, remove jobs from spool queues, and perform administrative functions such as starting or stopping printing.

A print spooler is a client/server application. The client programs are used to submit jobs to the print spooler program which performs the actual printing operations. In order to carry out these operations, the server may need to use other programs to convert print job files into a format acceptable to a printer, or perform various accounting or administrative functions.

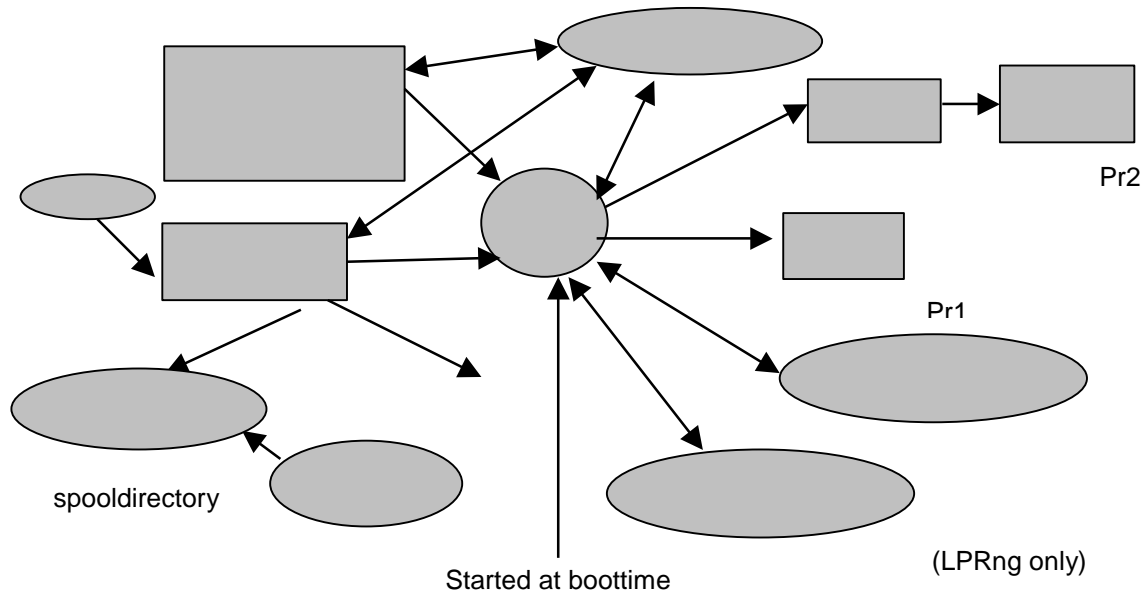


Figure 2 shows the flow of data between the individual components of the LPRng print spooling system. A program (or user) will use the lpr program to send a file to the lpd server over a TCP/IP connection. The lpd server will store the file temporarily in a spool queue directory. The information needed by the lpr and lpd programs to carry out this activity is stored in the printcap (usually called the /etc/printcap) database file.

The lpd server sorts the queue entries and determines the print order. It will select a job to be printed, open a connection to the printer, and then use a filter program to convert the file contents into a format suitable for the printer. If the file does not need conversion, then the lpd server will send the file directly to the printer.

The lpd server can also forward jobs to another print server over a network connection, optionally sending them through a filter as well. The destination server can in turn forward the job or send it to a printer.

The protocol or commands used to forward the job and transfer are specified by RFC1179. This protocol specifies how the lpr client program sends a job to the lpd server, as well as how the lpd server forwards jobs to another server. In addition to job submission, RFC1179 specifies commands to obtain queue status, to remove jobs from the queue, and to start and stop print queues. [2]

### [Why use LPRng (?)]

LPRng is probably the most powerful spooler software today:

built in failover: 'rm=<first\_spooler>,<second\_spooler>:'

stand alone functionality 'lpq -P<port>@<printer>' to obtain the exact status of the printer network interface. 'lpr -P<printer>@<spooler>' for forwarding jobs from one spooler to another, 'lpr -P<port>@<printer>' to check if the printer itself is accepting jobs on a certain port.

The 'lpc' command to control the 'lpd' daemon is not mentioned in RFC1179, therefore there are many nice features implemented for admins: topq, hold, redirect queues, set status message, reread conf files, redirect jobs etc.

Probably 98% of all printing problems can be tracked down by simply typing 'lpq -L -P<printer>' as the whole transfer and filter status from the client and spooler is displayed:

```
...
Filter_status: sending job file at 11:45:56.400
Filter_status: starting transfer at 11:45:56.400
Filter_status: file program = '/usr/bin/file -' at 11:45:56.400
Filter_status: started FILE_UTIL- 'file' at 11:45:56.403
```

```

Filter_status: file information = 'jpeg_image_data,_jfif_standard' at 11:45:56.471
Filter_status: initial job type 'jpeg_image_data,_jfif_standard' at 11:45:56.472
Filter_status: decoded job type 'jpeg_image_data,_jfif_standard' at 11:45:56.472
Filter_status: cannot process language 'jpeg_image_data,_jfif_standard' at 11:45:56.473
Filter_status: sent job file at 11:45:56.473
Filter_status: getting end using 'pjl job/eoj' at 11:45:56.474
Filter_status: end of job detected at 11:45:58.701
Filter_status: pagecounter 32442 after 1 attempts at 11:46:00.398
Filter_status: pagecounter 32442, pages 0 at 11:46:00.398
Filter_status: done at 11:46:00.399
...

```

4 ascii files (printcap, lpd.conf, lpd.perms, ifhp.conf) make the only difference to any other workgroupserver in our network, which is nice in cases of a hardware failure.

Locally attached printers can be defined in a second printcap, SysV and BSD binaries are provided for all UNIX platforms.

LPRng is compatible with all known printing implementations even with those like violating the RFC1179 while being as close to the RFC1179 as possible.

The permission file 'lpd.perms' allows different access to the functions of 'lpr', 'lpq', 'lprm', 'lpc' for classes of users.

Global definitions in the 'lpd.conf' file greatly simplify management and printcap.

The documentation for LPRng and IFHP is very good and there is a strong support on the mailing list

Accounting per page and printer is possible as well as error notification via e-mail in case of printer failures.

#### [What is a filter]

A filter is typically run on the server to convert the data files into the printer specific language, to create bannerpages (bp) or to query the printer (of). It is called by the lpd daemon, reads from <STDIN> and writes to <STDOUT>

Using LPRng with the 'filter=' printcap entry defines a filter for all dataformats determined in the controlfile except those explicitly defined in the printcap.

#### [The IFHP filter package]

The IFHP program is an enhanced, extended, highly configurable, and portable implementation of a print filter for use with LPRng. IFHP supports network, serial, and parallel printers, does page accounting and job recovery, and allows an extremely high level of configuration and tuning. IFHP gets its flexibility by using a configuration file to set its operational characteristics. The filter supports text, PostScript, PCL and PJI printers.[3]

#### [File conversion with the IFHP package]

The IFHP filter uses the file tool to determine the filetype and then calls different format conversion filters which are defined in the ,file\_output\_match':

```

file_output_match = [
  *postscript*  ps  \${ps_converter}
  *pcl*         pcl \${pcl_converter}
  *pjl*        pjl \${pjl_converter}
  *printer*job*language* pjl
  *text*       ps  /usr/libexec/filters/ascii host=\${H} user=\${n} Z=\${Z}
  *gzip_compressed* filter \${gzip_decompress}
  *PDF*        ps  /usr/libexec/filters/pdf
]

```

This makes it extremely easy to implement support for whatever type of files.

#### [Server & clients @ DESY]

On the servers there are mainly three pieces of software running: LPRng, IFHP and SAMBA. Some tools are needed as well: A2PS, FILE and some converters like PDF2PS etc. Normally every printer is served by one spooler, for failover reasons each printer is known on both machines.

#### [UNIX clients]

LPRng is installed on all centrally maintained clients by default. Though it would be possible to run the clients without any printer database and daemon we do it the old fashioned way with a complete installation. A lpd is started at boottime, reading the /etc/printcap file, scheduling jobs in the spooldirectories. As 99% of the centrally maintained clients are using AFS, a simple lp crontab entry does the regular printcap update. The LPRng 'checkpc' tool is used to create automatically the needed spooldirectories, remove jobs that don't get printed after 3 days and to keep logfiles small.

### [MACOS and NOVELL clients]

Both systems are not supported anymore, the NOVELL printserver is talking appletalk to the mac clients and LPR to the central spooler. An appletalk support on the UNIX spooler would be possible if desired.

### [Failover/availability]

All UNIX clients take advantage of the LPRng built in failover, if a spooler is not responding the second address is tried. If no spooler is available, the client lpd keeps polling for a few days. Due to the fact that NT jobs were forwarded from a NT spooling host we had a nice failover switching the nameserver alias in case of downtimes of the UNIX servers. Running SAMBA is different because each queue is just hosted on one of both machines. We will try to keep a backup of the SAMBA specific files and prepare a standby machine to join the NT domain using the mac key and the netbios alias from the backup in case of downtimes.

### [Using SAMBA 2.2 as a front-end interfaces to the UNIX queues]

Talking about network printing often means talking about gateways: The forwarding of jobs from the NT spooler works fine, the NT desktops install generic drivers over their spooler and then bind via RPC calls for every printjob to get the current configuration. This makes sure that everyone on

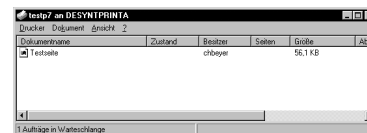


figure 3

the campus always uses the same and latest driver with the preconfigured 'standard options for documents' which makes it easy to track down the typical 'I can not print' problems from NT users. Unfortunately the NT spooler just forwards the jobs via 'lpr' but there is no queue inquiring function or job-remove functionality.

For a while SAMBA [4] has released version 2.2 which implements the whole NT printspooler functionality on a UNIX machine. There is no need for additional installations on the NT desktops or a WINDOWS printspooler in the domain. The desktops simply use their common mechanism to: download generic informations at install time, get the printer capabilities by using RPCs at printtime and to query the queue (figure 3). For installation through the common 'install printer wizard' the SAMBA server shares the domain browsing like any NT/2000 server would (figure 4).



figure 4

The server can be administrated from a NT desktop as well as from the UNIX side.

According to our investigation working with up to ~400 queues on a UNIX machine using SAMBA does not result in major problems. Only the admin mode, using the NT desktop gets very slow due to a lot of RPCs being handled. This can be avoided by using a little VB script to avoid passing through the browsing windows to get into the admin mode for a single printer. The SAMBA server at DESY will probably be the only printspooler in the DESY NT domain in the near future!

### [References]

- [1] Todd Rademacher & Matthew Gast, *Network Printing*, O'Reilly
- [2] Patrick A. Powell, *LPRng-HOWTO*, <http://www.lprng.com>
- [3] Patrick A. Powell, *IFHP-HOWTO*, <http://www.lprng.com>
- [4] The SAMBA group, <http://www.samba.org>