A simple, generic print server for distributed LPD queues. Requirements and design specifications

Francesco Prelz, Stefania Alborghetti, Mauro Campanella, Luca Carbone,
Giuseppe Lo Biondo, Marcello Meroni
INFN Milano, Computing Services
June 6^{th}, 1998

Abstract:

The management of printer queues in a multi-OS, multivendor, distributed computing environment often requires a large system integration effort, which entails fixing inessential quirks in a time-consuming, often frustrating process. In this document we try to identify a subset of fundamental printer functions, and describe a client-server architecture which is able to provide a central printer queue service for a number of remote LPD servers. A uniform, general set of print options and queue features are identified in the process.

An implementation of this architecture, written for generic UNIX, was developed and is currently being deployed at the Milano site of the Italian Institute for Nuclear Physics Research, INFN.

1 Introduction: who needs an extra server layer?

Implementations of the print client and server architectures described in RFC 1179 (which will in the following be referred to as *LPD clients* and

LPD servers) are now available for almost all commercial workstations and personal computers. Such implementations, however, do oftentimes provide vendor-specific extensions and features, which in practice translate into mismatches and incompatibilities when LPD servers are accessed remotely from a different incarnation of an LPD client.

Also, the RFC 1179 protocol fails to define certain formatting options which may be applicable to plaintext print files. This task is left to an external formatter. Once again, different print service vendors have implemented this kind of formatting service in different ways and at various levels, either by adding formatting commands that are specific to a particular kind of printer, or by resorting to more general languages such as PCL^{TM} and PostScript$^{(R)}$.

An additional problem is found when more than one queue is defined for a given physical printer. This may be the case when different, separate print queues need to be defined on different operating systems (e.g. VMS, Unix, NT) for lack of a more reliable scheme. Contention and race conditions for accessing the printer by the various queues is a definite possibility in this case. Once again, different and non-uniform formatting options may be available on the different queues.

Race conditions do occur also when some form of embedded LPD server is available in a given physical printer. Quite often such LPD servers do not provide any spooling and queueing capability and they are single-threaded: the server will not respond to connection attempts which are initiated while another print transaction is being served. This can cause (and does cause) service disruptions when many distributed LPD clients are configured to access such an embedded LPD server.

The widespread, almost general availability of Postscript$^{(R)}$ compatible printers, along with the ability to easily create an LPD server queue for any given printer (existing software tools can be used when an embedded LPD server is not available in the printer itself), prompted us to address the problems described above by specifiying a multithreaded server, with queuing, spooling and plaintext formatting features, that acts as an LPD client for a number of local or remote LPD servers. The remote LPD servers are only required to be able to spool and serve one print job at a time, while the bulk of the spool storage is concentrated in one server, and unnecessary file copies are avoided.

2 Common print service needs

2.1 General formatting options

Paper-saving options should be applicable to any and every print job, on any printer, in particular:

- duplex printing,
- n-box-per-page printing.

In particular, it should be possible to define duplex printing as the default mode for a given printer queue. A generally applicable model is to translate every print job into a general purpose high-level formatting language such as Postscript$^{(R)}$, and to apply the required formatting by adding the appropriate directives in such a high-level language.

2.2 Plaintext formatting options

Similar considerations apply to plaintext print jobs. Options that should be provided in addition to the ones listed above are:

- choice of the font type and size,
- choice of landscape/portrait orientation,
- choice of the number of columns per text row,
- printing of file information along with the file contents.

2.3 Queue features

A number of features that apply to a given print queue, and which are not generally available via LPD without vendor-specific extensions can be implemented in a general purpose print server. These include:

- ability to optionally reorder queued print jobs so that shorter jobs are printed before longer ones;
- ability to hold print jobs so that special forms (such as transparent film or letterhead paper) can be fed into the printer;

- ability to define a queue-based access control scheme, either by user or by hostname, by net or subnet;
- ability to print a job banner, for those high-volume printers which serve a large number of users from a displaced location;

2.4 Administration and auditing

Certain features which apply to the server in general, and that perform administrative tasks are also required:

- ability to remove jobs by the remote queue user or by a system administrator: this may sound obvious but cannot be achieved when multiple, cascaded LPD queues are defined in a system;
- ability to limit access to the server to a set of host or networks;
- ability to report a warning to the printer administrator when no print jobs have been successfully printed for a certain amount of time;
- ability to generate an auditing plaintext log file, listing job owner, remote host, printer queue and number of pages.

3 Client-server model

A single, site-wide queue manager (we'll refer to it more generically as the *print server*) needs to be available at any time to properly serialize and handle the flow of print requests. The architecture we describe here provides for a "primary" print server, which handles all print and queue management requests, and a "backup" print server, where the print requests are directed by the print service clients in case of failure of the primary print server. The backup print server is not allowed to accept new print jobs if the primary server is up and running, and the primary server is not allowed to process any print job waiting in a print queue until all the queues in the backup server are empty. This provides for smooth transitions when the primary server stops and starts again.

The site-wide, high traffic nature of the print server requires that all of the IO-intensive tasks (data transfer from the print clients, file conversion and filtering and data transfer to the print servers) be handled by detached

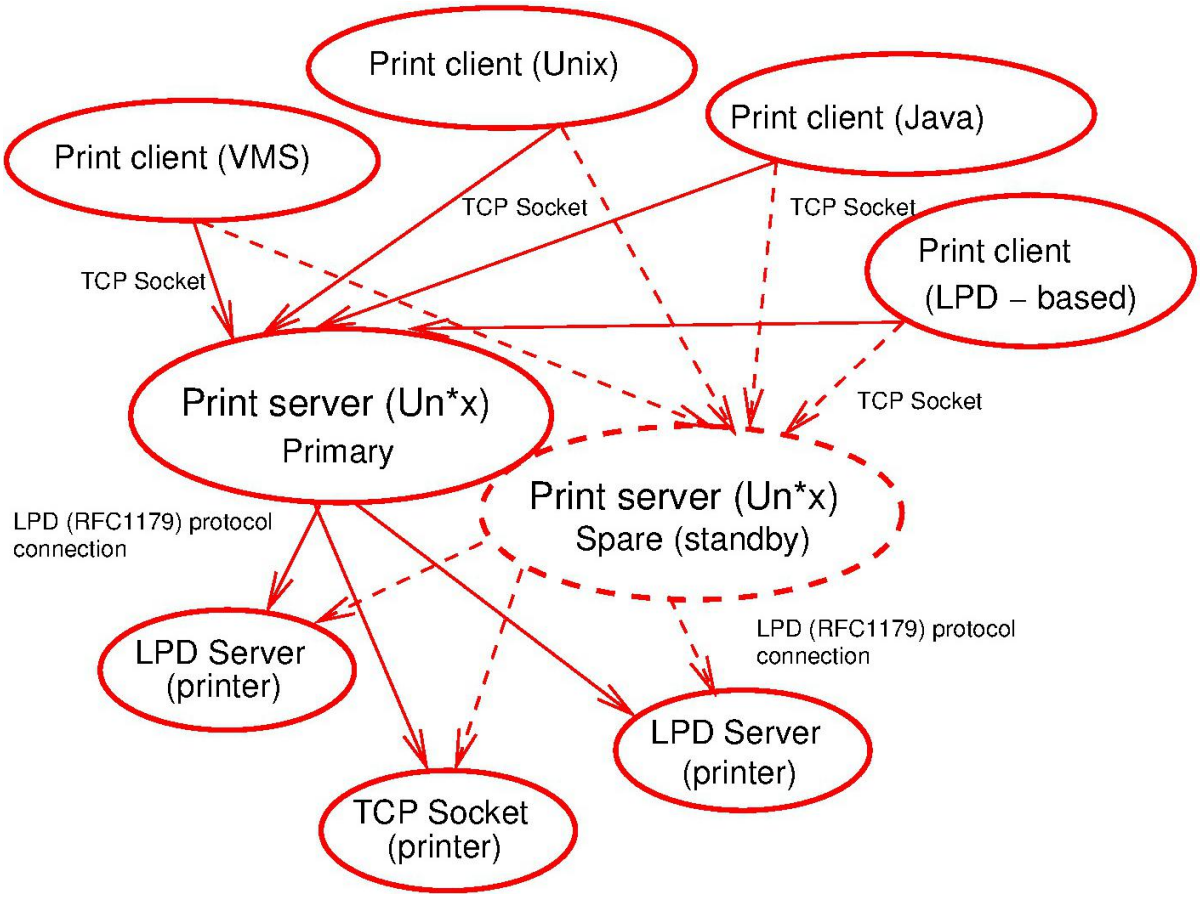

Figure 1: Client-server model for the centralized printer queue tool.

processes. The only function of the print server "parent" process is to handle the bookkeeping of the printer queues and to detach an lpr spool (client) process when the target lpd server has become available.

Given the fact that most of the complexity of this architecture is found in the server, the print client can be kept extremely lightweight, and can thus be provided for a larger number of platforms than the print server. Currently, a server for UNIX and clients for UNIX, VMS and for any platform running secure JAVA^{TM} applets have been written.

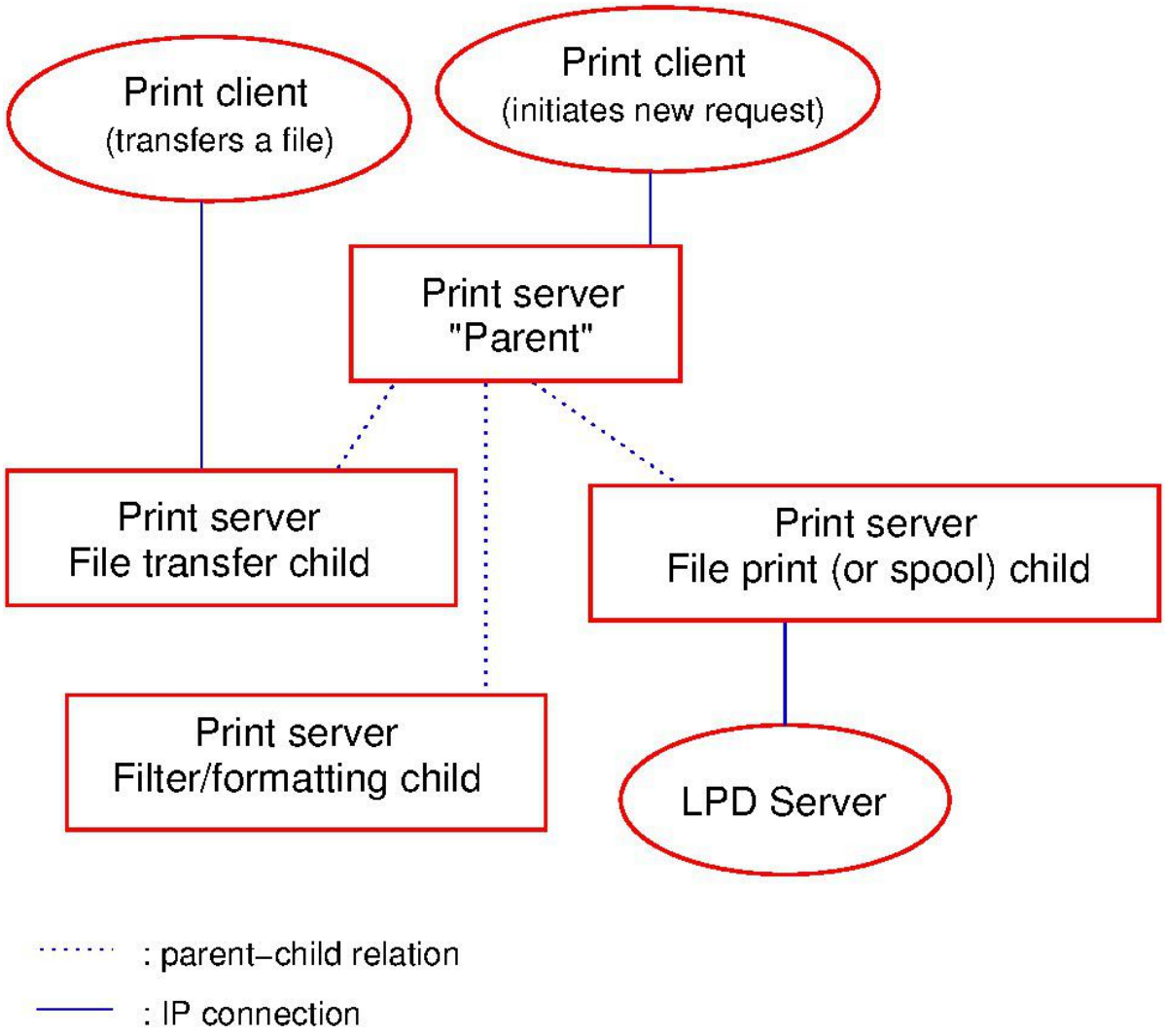

Figure 2: A site-wide print server needs to implement some form of parallel handling of incoming and outgoing print requests. In the proposed scheme, the "parent" print server only deals with queue bookkeeping, while all IO-consuming activities (data transfer and filtering) are carried out by detached children processes.