Joint laboratory of Optics *PU & IP AS ČR Olomouc*	**Report No.** 367/SLO/2010

Alternative open source implementation of IPP server

Projects MŠMT ČR č. 1M06002, OP VK, č.CZ.1.07/2.2.00/07.0018

Authors: Ondřej Haderka, Martin Hamar, Václav Michálek, Robert Myška

Head of the department: prof. RNDr. Miroslav Hrabovský, DrSc.

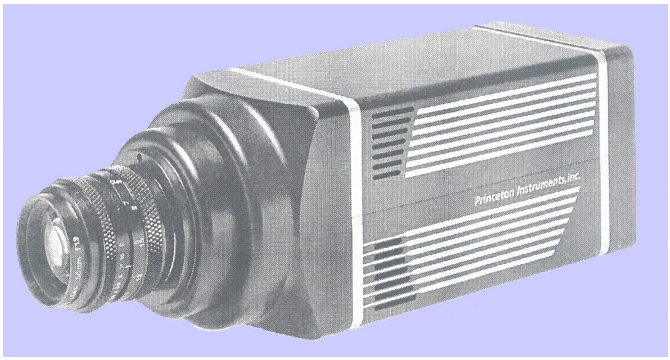
Olomouc, 32.8.2010

Distribution: - SLO

Pages: **11**

Print No.: …

Alternative open source implementation of IPP (Internet Printing Protocol 1.1) server



Index

1. Introduction

This report summarizes implementation of Internet Printing Protocol 1.1 (IPP) server into LPRng [lprng] printing software.

IPP protocol is now the most frequent protocol for printing over Internet, especially in the unix world. In our work/office, we used Samba printing (for Windows machines), but for unix stations, it is better to use IPP protocol, because it enables central administration. Also, one of advantages is the fact that transformation from postscript to specific printer language (for non-postscript printers) needs to be configured only at the server level. Today, there is only one open source IPP implementation, Common Unix Printing System [cups].

Since our existing printing software LPRng does not recognize IPP protocol, we decided to implement it. Other reasons are better cooperation with Windows IPP clients and keeping existing running configuration.

Planned project requirements:

- IPP 1.1 protocol (rfc2910, rfc2911)
- "light" implementation without big dependencies and libraries
- as small as possible changes of existing software
- simple configuration and interoperability with existing windows and unix clients
- no more functionality that can be achieved using existing LPR/LPD protocol, no configuration over IPP

2. Features

- Supported IPP operations:

Operation	Code	Remark
Print-Job	0x0002	rfc2911
Validate-Job	0x0004	rfc2911
Create-Job	0x0005	rfc2911, uses unix OpenOffice
Send-Document	0x0006	rfc2911, uses unix OpenOffice
Cancel-Job	0x0008	rfc2911
Get-Job-Attributes	0x0009	rfc2911
Get-Jobs	0x000a	rfc2911
Get-Printer-Attributes	0x000b	rfc2911
Hold-Job	0x000c	rfc2911
Release-Job	0x000d	rfc2911
Restart-Job	0x000e	rfc2911
Change-Job-Attributes	0x0014	rfc3380
Create-Printer-Subscriptions	0x0016	~rfc3995 not conforming rfc
Get-Subscription-Attributes	0x0018	rfc3995

Get-Subscriptions	0x0019	~rfc3995 not conforming rfc
Renew-Subscription	0x001a	rfc3995
Cancel-Subscription	0x001b	rfc3995
Get-Notifications	0x001c	~rfc3996 not conforming rfc
CUPS-Get-Default	0x4001	Get default printer
CUPS-Get-Printers	0x4002	Get all printers
CUPS-Get-Classes	0x4005	Empty operation
CUPS-Get-Devices	0x400B	Empty operation
CUPS-Move-Job	0x400D	Move job to another printer

- Compatibility with CUPS clients, including many CUPS-specific extensions, attributes and operations.
- Basic authentication, SSL with TLS 1.0.
- Handling printing options in CUPS-compatible way (serving of PPD files via HTTP and passing job options to spooler filter).
- Only local queues are accessible via IPP.
- Supported IPP communication language en_US and UFT-8 encoding only.
- IPP server only (daemon cannot pass jobs to remote IPP printers).
- License: the GNU GPL License Version 2 or the Artistic License
- Used standards:

RFC	Description	Remark
rfc2616	Hypertext Transfer Protocol -- HTTP/1.1	
rfc2910	Internet Printing Protocol/1.1: Encoding and Transport	
rfc2911	Internet Printing Protocol/1.1: Model and Semantics	
rfc3380	Internet Printing Protocol (IPP): Job and Printer Set Operations	settable attributes: job-priority, job-hold-until (*system-config-printer* application uses setting of job-hold-until attribute to hold/release job)
rfc3995	Internet Printing Protocol (IPP): Event Notifications and Subscriptions	simple implemntation satisfying *system-config-printer* application but not conforming rfc
rfc3996	Internet Printing Protocol (IPP): The 'ippget' Delivery Method for Event Notifications	simple implemntation satisfying *system-config-printer* application but not conforming rfc
rfc3510	Internet Printing Protocol/1.1: IPP URL Scheme	

3. Configuration

3.1 LPD server

Compared to the original LPRng version, the software requires additional libraries: PAM (pluggable authentication module, used for HTTP Basic authentication) and iconv (conversion from IPP values (UTF-8) to local names (locales of running LPD daemon) and back); compilation and installation should not be different from the original software.

IPP-related configuration parametres (new and IPP-affected options):

lpd.conf options:

ipp_listen_port (string)

Incoming (listening) TCP port for IPP service. Default value is 631 (standard IPP service port). (The IPP service cannot be accessed via unix socket.)

ipps_listen_port (string)

Listening port for direct HTTPS (TLS/1.0) encrypted IPP requests. Default value is "off" (disabled), because there is no standard port (standard HTTPS port 443 is usually reserved for WWW server).

ipp_getjobs_compat (integer)

This option comprises various compatibility flags for IPP clients. The value can be logical OR of bit values:

0x1: in addition to LPRng-standard URIs (see below), the server accepts:
- printer-uri "ipp://localhost/"
  for Get-Jobs operation (to get all jobs for all printers),
- printer-uri "/"
  for subscription and notification operations (all printers),
- job-uri "ipp://localhost/jobs/*job_number*"
  for job operations

(URIs compatible with newer CUPS clients).

0x2: in addition to standard URIs, the server accepts:
- job-uri "ipp://localhost/jobs"
  for Get-Jobs operation,
- printer-uri "/"
  for subscription and notification operations,
- job-uri "ipp://localhost/jobs/*job_number*"
  for job operations

(compatible URIs with older CUPS clients).

0x100: for "Internet Print Provider" (standard Microsoft Windows IPP print client) and "Novell iPrint Client" (based on User-Agent HTTP header), the Get-Jobs operation with unspecified "which-jobs" IPP attribute

returns all jobs, i.e. completed and non-completed (this is not IPP standard, but enables to show completed jobs in Windows clients – this is pleasant feature, despite of the fact that these clients cannot correctly show the state of jobs: printed jobs are showed OK, but aborted or canceled jobs have wrong or none state).

Default value for *ipp_getjobs_compat* option is 0x1.

ipp_compat_hrcount (integer)

The CUPS servers use unique job number (job-id) within the whole server and the standard CUPS job-uri ("ipp://localhost/jobs/*job_number*") does not contain the printer name; LPRng jobs for different printers can have the same job number. For cooperation with CUPS clients, LPRng transforms job numbers in IPP communication to be unique too. Each printer has reserved number interval <p**ipp_compat_hrcount,* (p+1)**ipp_compat_hrcount*-1> , where p is a printer number. For example, let us suppose default value *ipp_compat_hrcount*=100 . Then job number 3 in the first printer (queue) is transformed to job-id 3, job 3 in the second printer has job-id 103. (LPRng job-uri attributes use always original job numbers, e.g. ipp://example.com:631/printers/printer2/jobs/3).

Any print queue should not contain more than *ipp_compat_hrcount* jobs, otherwise CUPS clients need not work properly. Fortunately LPRng uses as small job numbers as possible.

The job-id transformation is done only if *ipp_getjobs_compat* has set flags 0x1 or 0x2 (CUPS-compatible URIs are turned on).

ssl_ca_path, *ssl_ca_file*, *ssl_server_cert*, *ssl_server_password_file*

These options are used to define parametres for IPP over SSL communication. (For details, see the original LPRng documentation [lprng-doc]). The LPRng IPP server supports two ways to establish encrypted connection: direct encrypted connection to HTTPS port and HTTP upgrade mechanism for unencrypted connection. The server uses TLS/1.0 encryption.

default_printer (string)

Specifies global default printer for CUPS clients. To set other default CUPS printer for particular hosts, use *SERVICE*=d *REMOTEHOST*=xxx in lpd.perms file.

printcap options (printer-specific options):

printer alias

The last alternative printer name in printcap is returned as "printer-make-and-model" and "printer-location" attributes in Get-Printer-Attributes operation.

cm (string)

Commnet identifying printer is returned as “printer-info” attribute in Get-Printer-Attributes operation.

iauth (string)

Allowed authentication methods for the printer (comma-separated case-sensitive keywords). Possible values are *usrname* (use “requesting-user-name” IPP attribute) and *basic* (use HTTP Basic authentication; username/password is validated against “lprng” PAM service). Default value is *usrname*.

ppd (string)

Specify the location for PPD file. CUPS clients can download the file via HTTP protocol and use definitions from the file to offer various printing options. These options are sent to the IPP server in Print-Job or Create-Job operation (as job template attributes) and are passed to the filters as “Z” options (LPRng does not modify the job data, you can use [foomatic] *if* filter to process these options).

Some remote LPD printers cannot accept long control lines with Z options, so use of *control_filter* printcap option to remove “Z” line from control file may be necessary.

mc (integer)

Maximum allowed number of copies. Default value is 1, but CUPS IPP backend does not send document name to the IPP server in this case. Therefore value 2 or more is recommended.

imct (string)

This option controls copies handling. The value “if” causes printing the job only once, for printing of more copies and requested collation is responsible printing filter. Arbitrary other value causes LPRng to print job more times, but only "multiple-documents-collated-copies" collation can be achieved.

IPP-related command-line parametres:

lpd	-i port	- TCP/IP ipp listen port, 'off' disables TCP/IP listening port (*ipp_listen_port*)
	-s port	- TCP/IP https listen port, 'off' disables TCP/IP listening port (*ipps_listen_port*)

Permissions (lpd.perms):

IPP-related permission keywords:

IPP

True if client is connected to *ipp_listen_port* or *ipps_listen_port*.
Example: “REJECT not SERVER not IPP” rule allows only IPP requests from remote hosts.

AUTHTYPE=keyword

IPP authtypes are “*usrname*” or “*basic*”.
Example: “REJECT AUTHTYPE=usrname NOT REMOTEHOST=localnet” requires basic (=disallows usrname) authentication from foreign hosts.

PPATH=url_path

Specifies particular IPP uri path (valid names are “*printers*” or “*raw*”) . It is intended for simultaneous operation of Windows and CUPS clients (see below).

Required permissions for IPP operations:

Operation	Required permission
Print-Job, Validate-Job, Create-Job	SERVICE=R and SERVICE=C for accept job-hold-until job template attribute
Send-Document	SERVICE=R
Cancel-Job	SERVICE=M for job or SERVICE=C for queue
Get-Job-Attributes, Get-Jobs	SERVICE=Q
Get-Printer-Attributes, CUPS-Get-Printers	SERVICE=S
Hold-Job	SERVICE=C LPC=hold
Release-Job	SERVICE=C LPC=release
Restart-Job	SERVICE=C LPC=restart
Change-Job-Attributes	SERVICE=C change job job-hold-until attribute: LPC=hold or LPC=release change job job-priority attribute: LPC=chattr-job-priority
Create-Printer-Subscriptions, Get-Subscription-Attributes, Get-Subscriptions, Renew-Subscription, Cancel-Subscription, Get-Notifications	SERVICE=S

CUPS-Get-Default	SERVICE=S and (SERVICE=d for the default printer (use SERVICE=d REMOTEHOST=host to specify default printer for particular host; first rule is used) or set *default_printer* in lpd.conf globally)
CUPS-Move-Job	SERVICE=C LPC=move PRINTER=current_printer and SERVICE=P PRINTER=destination_printer

3.2 Client configuration

LPRng printer URIs are:

scheme://*lprng_host*[:*port*]/printers/*printername* ("printers" URI, CUPS compatible)

and

scheme://*lprng_host*[:*port*]/raw/*printername* ("raw" URI),

where *scheme* is "http" (for IPP 1.0 clients), "ipp" (IPP 1.1 clients) or "https" (for TLS-encrypted connection).

New jobs received using "printers" URI have 'f' (ordinary text) job format, new jobs sent to "raw" URI have 'l' (binary) job format. (Binary format is also assigned to jobs with "application/vnd.cups-raw") document-format). Two different URIs for the same printer help to achieve simultaneous traffic with unix and Windows clients (various clients also require different configuration for *basic* authentication).

CUPS-specific operations always use and return "printers" URI.

Standard Microsoft Windows client (Internet Print Provider)

Standard Windows clients use IPP/1.0 protocol.

Use "raw" URI with "http" scheme in conjunction with native Windows driver for the specified printer:

http://*lprng_host*:*ipp_listen_port*/raw/*printername*

Windows 2000 support IPP encrypted (HTTPS) communication (TLS/1.0 protocol must be enabled in Internet settings). The printer URI is

https://*lprng_host*:*ipps_listen_port*/raw/*printername*

Windows XP can use HTTPS encrypted communication for IPP, but there is a funny bug here: all print jobs are sent to IPP server twice. Higher version of Windows do not support secured IPP communication.

With "*usrname*" authentication, Windows do not send any username when removing jobs. A workaround is to permit job remove from the same host.

(lpd.perms example – remember that there are more ways to define it …)
```
#windows do not send the user name when removing job !!!
ACCEPT SERVICE=M IPP AUTHTYPE=usrname PPATH=raw SAMEHOST
```

To setup "*basic*" authentication for printing from Windows clients, all operations must use this authentication:

(example of lpd.perms)
```
#windows client
REJECT AUTHTYPE=usrname PPATH=raw PRINTER=passw_protected IPP
DEFAULT ACCEPT
```

Novell iPrint client for Windows

This client use IPP/1.1 protocol and also supports HTTPS communication. Using this printing provider is at your own risk, but it may work, even outside Novell environment. Printer URIs are

http://*lprng_host*:*ipp_listen_port*/raw/*printername* or
ipp://*lprng_host*:*ipp_listen_port*/raw/*printername* (unencrypted connection),
https://*lprng_host*:*ipps_listen_port*/raw/*printername* (encrypted connection).

Use native Windows printig driver like standard Microsoft client. This client can also prompt for username/password if invalid or needed, for example if you try to remove job of another user.

CUPS clients (unix)

Configuration of CUPS clients is straightforward, simply configure LPRng as remote CUPS server (put ServerName *lprng_host*:*ipp_listen_port* in your /etc/cups/client.conf file or something like this). LPRng server does not advertize itself on the network, the server must be specified explicitly. Since CUPS clients often use CUPS-specific (non-portable) URIs (on the contrary to the CUPS library, which allows you to specify arbitrary URIs), accepting of CUPS-specific URIs must be enabled in *ipp_getjobs_compat* option. Encrypted communication (HTTP upgrade) is configured by "Encryption Required" /etc/cups/client.conf option.

To setup "*basic*" authentication for CUPS clients, Get-Jobs operation (and all SERICE=S operations) must be accessible with no restriction. (CUPS authors say that it is not intended to restrict Get-Jobs operation, and since clients use printer-uri "ipp://localhost/" to get all jobs instead of cycling all printers, it is also impossibe to make restrictions, because IPP authentication is per-printer by design.)

(printcap)
```
passw_protected:\
```

```
    iauth=usrname,basic
```

(lpd.perms)

```
#CUPS clients
REJECT SERVICE=R,P,M,C AUTHTYPE=usrname PRINTER=passw_protected IPP
DEFAULT ACCEPT
```

3.3 Other remarks

1. To satisfy requirements for IPP job states and allow IPP notifications (event store), lprm (and Cancel-Job IPP operation) removes jobs in two stages: first lprm call changes job state to “canceled” and finally second lprm completely purges the job.
2. If fifo job order enforced (fifo flag, default on), job create operations create lock file for each host (in printer spool directory). Lock files can be removed by *checkpc* application, usually called by cron (check your installation).
3. The server uses native locales to store job information. Since IPP communication uses utf-8 encoding, the utf-8 locales are recommended, otherwise job names and other information may not be preserved correctly (it does not affect the functionality).

4. Programmer's notes

4.1 Event notifications and subscriptions implementation

To minimize changes to existing software, event notifications and subscriptions are implemented in a simplified stateless way. This implementation also uses low memory and has no extra library requirements; on the other hand, it does not satisfy the rfc standards.

The stateless event notifications use notify-subscription-id and notify-sequence-id attributes to store necessary information: subscribed events and/or subscribed job-id.

notify-subscription-id bit(s)	Printer subscription	Job subscription
31	0	1
30	reserved	reserved
29	1=job-state-changed event subscribed	job-state-changed
28	job-created	job-created
27	job-completed	job-completed
26	printer-state-changed	notify-job-id value
25	printer-stopped	
24-0	not used	

Maximum job-id is limited to 27 bit range (CUPS-compatibile transformed value – *ipp_compat_hrcount*). Subscriptions cannot hold any user data, fortunately clients do not use it now.

As for the notify-sequence-id, unix time is used. The exception is subscription with printer events only, where maximum of 5 low bits are used to store the number of printer which is not in the idle state:

0: all printers in idle state
1: more than one printer not in idle state
>2: single printer in idle state (printer number increased by 2)

Using of timestamp for notify-sequence-id violates the rfc (notify-sequence-id values do not form sequence of natural values and need not be unique, on the other hand clients cannot determine that missed values do not belong to expired events).

Proper implementation of event notification and subscriptions would require shared memory, because CUPS clients use all-printers requests (there is no global space to store data in existing LPRng software) and also notifications and subscriptions are stateful operations. Events would be generated by hooks to Set_job_ticket_file() and Remove_job() (job events) and Set_spool_queue() with child-server hook (for printer events), with respect to control signals. This approach requires more changes in software (holding shared information requires complicated interprocess communication (if programmed from scratch) or additional shared memory library dependencies) and may be done in the future.

4.2 TODO work for volunteers

- Rewrite event notifications and subscriptions to be rfc-conforming.
- Kerberos HTTP authentication, User-Agent controlled permission possibility.
- Enhance the software to support more than one printer-state-reasons and job-state-reasons IPP values.

5. Conclusion

An independent IPP/1.1 implementation was made. The server was tested with several clients, including Microsoft IPP, Novell iprint, cups/gnome, cups/system-config-printer, cups/kde3, cups/kde4, cups/OpenOffice, which required amount of special features (see the source code).

The software is accessible as-is and authors do not provide any support.

The project was supported by grants MŠMT ČR 1M06002, OP VK CZ.1.07/2.2.00/07.0018. and CZ.NIC, z.s.p.o.

References

[lprng] LPRng <http://lprng.sourceforge.net> [cit. 2010-08-02]
[lprng-doc] LPRng documentation <http://www.lprng.com> [cit. 2010-08-02]
[cups] Common UNIX Printing system <http://www.cups.org> [cit. 2010-08-02]
[foomatic] Foomatic
<http://www.linuxfoundation.org/collaborate/workgroups/openprinting/database/foomatic>
[cit. 2010-08-02]