IPP Client Use Best Practices

Status: Interim

Abstract: This document enumerates a number of tasks that are commonly performed by a client in the process of interacting with a print service, and explores options for how the Internet Printing Protocol (IPP) may be used to perform those tasks, some of which are preferred and others that are less than optimal.

This document is a PWG Working Draft. For a definition of a "PWG Working Draft", see: ftp://ftp.pwg.org/pub/pwg/general/pwg-process30.pdf

This document is available electronically at:

ftp://ftp.pwg.org/pub/pwg/general/templates/tb-ipp-best-practices-20130205.pdf

Copyright © 2013 The Printer Working Group. All rights reserved.

This document may be copied and furnished to others, and derivative works that comment on, or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice, this paragraph and the title of the Document as referenced below are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the IEEE-ISTO and the Printer Working Group, a program of the IEEE-ISTO.

Title: *IPP Client Use Best Practices*

The IEEE-ISTO and the Printer Working Group DISCLAIM ANY AND ALL WARRANTIES, WHETHER EXPRESS OR IMPLIED INCLUDING (WITHOUT LIMITATION) ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

The Printer Working Group, a program of the IEEE-ISTO, reserves the right to make changes to the document without further notice. The document may be updated, replaced or made obsolete by other documents at any time.

The IEEE-ISTO takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights.

The IEEE-ISTO invites any interested party to bring to its attention any copyrights, patents, or patent applications, or other proprietary rights which may cover technology that may be required to implement the contents of this document. The IEEE-ISTO and its programs shall not be responsible for identifying patents for which a license may be required by a document and/or IEEE-ISTO Industry Group Standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention. Inquiries may be submitted to the IEEE-ISTO by e-mail at: ieee-isto@ieee.org.

The Printer Working Group acknowledges that the IEEE-ISTO (acting itself or through its designees) is, and shall at all times, be the sole entity that may authorize the use of certification marks, trademarks, or other special designations to indicate compliance with these materials.

Use of this document is wholly voluntary. The existence of this document does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to its scope.

**About the IEEE-ISTO**

The IEEE-ISTO is a not-for-profit corporation offering industry groups an innovative and flexible operational forum and support services. The IEEE-ISTO provides a forum not only to develop standards, but also to facilitate activities that support the implementation and acceptance of standards in the marketplace. The organization is affiliated with the IEEE (<http://www.ieee.org/>) and the IEEE Standards Association ([http://standards.ieee.org/)](http://standards.ieee.org/%29).

For additional information regarding the IEEE-ISTO and its industry programs visit:

<http://www.ieee-isto.org>

**About the IEEE-ISTO PWG**

The Printer Working Group (or PWG) is a Program of the IEEE Industry Standards and Technology Organization (ISTO) with member organizations including printer manufacturers, print server developers, operating system providers, network operating systems providers, network connectivity vendors, and print management application developers. The group is chartered to make printers and the applications and operating systems supporting them work together better. All references to the PWG in this document implicitly mean “The Printer Working Group, a Program of the IEEE ISTO.” In order to meet this objective, the PWG will document the results of their work as open standards that define print related protocols, interfaces, procedures and conventions. Printer manufacturers and vendors of printer related software will benefit from the interoperability provided by voluntary conformance to these standards.

In general, a PWG standard is a specification that is stable, well understood, and is technically competent, has multiple, independent and interoperable implementations with substantial operational experience, and enjoys significant public support.

For additional information regarding the Printer Working Group visit:

http://www.pwg.org

Contact information:

The Printer Working Group

c/o The IEEE Industry Standards and Technology Organization

445 Hoes Lane

Piscataway, NJ 08854

USA

**About the Internet Printing Protocol Work Group**

The Internet Printing Protocol (IPP) working group has developed a modern, full-featured network printing protocol, which is now the industry standard. IPP allows a print client to query a printer for its supported capabilities, features, and parameters to allow the selection of an appropriate printer for each print job. IPP also provides job information prior to, during, and at the end of job processing.

For additional information regarding IPP visit:

 http://www.pwg.org/ipp/

Implementers of this specification are encouraged to join the IPP mailing list in order to participate in any discussions of the specification. Suggested additions, changes, or clarification to this specification, should be sent to the IPP mailing list for consideration.

Table of Contents

1. Introduction 6

2. Terminology 6

2.1 Conformance Terminology 6

2.2 Other Terminology 6

2.3 Acronyms and Organizations 6

3. Requirements 7

3.1 Rationale 7

3.2 Use Cases 7

3.2.1 Developer Implementing New IPP Client Support 7

3.2.2 Developer Implementing New IPP Printer Support 7

3.3 Out of Scope 7

3.4 Design Requirements 7

4. Tasks and Implementation Alternatives 8

4.1 Create A Relationship With A Printer 8

4.1.1 Discover And Select A Printer Via A Discovery Protocol 8

4.1.2 Select A Printer Via User Provided DNS Hostname Or Raw Ipv4 / Ipv6 Address 9

4.2 Validate User Access to Printer 10

4.3 Get Printer Options 10

4.4 Check constraints between presented options 11

4.5 Submitting a Print Job 12

4.5.1 Submitting a print job with document data 12

4.5.2 Submitting a print job with document references 13

4.6 Monitoring print job status 14

4.7 Canceling a Print Job 15

4.8 Getting printer supplies status 16

5. Attributes and Their Use in Operations 16

5.1 Explicit "document-format" Selection 16

5.2 Prefer "media-col" Attribute To "media" Attribute 16

5.3 Prefer "finishings-col" Attribute To "finishings" Attribute 17

5.4 Using "ipp-attribute-fidelity" 17

5.5 Using "pdl-override" 17

6. HTTP Protocol Usage 17

6.1 HTTP/1.1 Expect Header 17

7. Security Considerations 18

8. References 18

8.1 Informative References 18

9. Authors' Addresses 19

10. Change History 19

10.1 February 5, 2013 19

10.2 March 20, 2013 20

1. Introduction

The use case descriptions below represent stages or sub-tasks that users perform in the process of using a printer. Each of these below include a textual description as well as a series of workflow options for how it might be implemented using IPP. Each workflow option will be informally labeled according to its perceived quality, using the set of labels {"BAD", "POOR", "GOOD", "BETTER", "BEST"}, that are ordered from least desirable to most desirable.

1. Terminology
	1. Conformance Terminology

Capitalized terms, such as MUST, MUST NOT, RECOMMENDED, REQUIRED, SHOULD, SHOULD NOT, MAY, and OPTIONAL, have special meaning relating to conformance as defined in Key words for use in RFCs to Indicate Requirement Levels [RFC2119]. The term CONDITIONALLY REQUIRED is additionally defined for a conformance requirement that applies to a particular capability or feature.

* 1. Other Terminology

*Capitalized Term In Italics*: definition of the term with any references as appropriate.

* 1. Acronyms and Organizations

*IANA*: Internet Assigned Numbers Authority, http://www.iana.org/

*IETF*: Internet Engineering Task Force, http://www.ietf.org/

*ISO*: International Organization for Standardization, http://www.iso.org/

*PWG*: Printer Working Group, http://www.pwg.org/

1. Requirements
	1. Rationale

The Internet Printing Protocol/1.1: Implementor's Guide [RFC3196] was ratified in November 2001. Since that time many extensions to IPP have been ratified, and the scope of use of IPP has grown considerably. Given all these extensions to IPP, implementers would benefit from an updated best practices document that covers the use of these extensions, as well as the core of IPP that has remained unchanged, to assist implementers in their efforts to deliver a quality client experience.

* 1. Use Cases
		1. Developer Implementing New IPP Client Support

Garrett is a developer working on a new client platform that is adding system-level printing support. Many printers support IPP Everywhere [PWG5100.14], so he plans to implement printing support in his client platform using this standard as well. But IPP Everywhere and its related standards don't describe how best to use IPP for the various tasks his software must perform, in order to deliver a quality client user experience. He finds RFC 3196 but its recommendations are insufficient. Using the IPP Use Best Practices document, he is able to avoid some common design pitfalls and quickly deliver a quality IPP client experience.

* + 1. Developer Implementing New IPP Printer Support

Duncan is a firmware developer at a printer vendor creating a new printer that implements IPP Everywhere. In reading the IPP Client Use Best Practices, he can more easily anticipate how some segment of clients implemented according to these practices are likely to behave, and more rapidly understand how the various operations can be used with one another to achieve certain tasks.

* 1. Out of Scope

The following are considered out of scope for this specification:

1. Specifications to extend or replace portions of the Internet Printing Protocol itself
2. Normative requirements regarding user experience
	1. Design Requirements

The design requirements for this specification are:

1. Explore tasks performed by client implementations
2. Enumerate a series of alternatives
3. Rank those options according to a non-numeric qualitative grading scheme
4. Tasks and Implementation Alternatives

For a number of tasks, the set of IPP operations provides a rich enough set of semantics that it is possible to perform those tasks in a few different ways. In this section a number of common tasks will be enumerated, and some alternatives for how those tasks might be performed will be evaluated.

* 1. Create A Relationship With A Printer

You can’t print to a printer if you cannot establish a connection to it. Historically, connecting to a printer to establish a "relationship" with it meant identifying a printer and then creating a persistent local records and resources for that printer relationship with your system’s print spooler. This was called a "print queue", and it involved binding drivers to create the relationships needed to communicate at the different levels, and then keeping record of that set of relationships so that it could be re-used at a later time. The set of printers or other devices the user’s system might encounter was relatively small and fairly static.

More recent re-thinking of this relationship between client and printer has resulted in more "dynamic" relationship creation, where universal drivers can interrogate a device hosting a print service using a standardized protocol solution stack, and using that dynamically ascertain and update print service attributes. In this paradigm, a "persistent" print service record is more like a Web browser bookmark.

Both paradigms still require a method of identifying the target devices. That can be done using dynamic service discovery protocols where the services respond to discovery requests, or explicitly by name (host name or raw IPv4/IPv6 address).

* + 1. Discover And Select A Printer Via A Discovery Protocol

Discovery protocols are used to identify instances of print services or printers by searching the network for service types or device types. This helps the user by making it so that they don’t need to do a physical survey of devices' addresses.

Regardless of the actual discovery protocol used, the APIs driving the protocols generally can be used in either a synchronous or asynchronous fashion. Unfortunately, many legacy software systems (as well as developers) are accustomed to the synchronous model, which is easily identified by the presence of a "refresh button". The synchronous model is not as user friendly as the asynchronous model, but it is somewhat easier to write programs in a synchronous way than an asynchronous way.

Options

* POOR:
	+ Perform network discovery with a synchronous API
		- Show progress bar
		- Discovery.Start()
		- sleep(X) where X is some reasonably short number of seconds
		- Discovery.Stop()
	+ Present the results of the discovery process
	+ "Refresh" button restarts the process
		- Why this is bad:
			* List contents can be stale
			* Results are not "live"
			* "Reset" button is unnecessary and is a crutch
	+ User selects a printer and presses "Continue" or equivalent
* BETTER:
	+ Perform network discovery with an asynchronous API
		- Show List UI widget
		- Discovery.Start() with a callback
		- Callback is called when discovery responses (add or remove) are received
	+ User selects a printer and presses "Continue" or equivalent
		- Discovery.Stop()
		1. Select A Printer Via User Provided DNS Hostname Or Raw Ipv4 / Ipv6 Address

In some cases a discovery protocol is either not adequate or unnecessary. Examples of when this use case is encountered include pre-published names or addresses, and also situations where the target device is not on the local link. (DNS-SD and WS-Discovery are generally used for link-local discovery, though wide-area variants as well as LDAP systems may also be used, but are frequently not for various reasons.)

For each of these options below, the assumption is that the client has been given an address string, and should attempt to connect to the host at that address.

Options

* BAD:
	+ Let each printer model make up its own path, and depend on some other protocol to get the resource path
		- IPP has no defined standard mechanism to enumerate the Printer objects' resource paths
* POOR:
	+ IPP Get-Printer-Attributes with printer-uri set to a URI that was manually entered by the user
		- The "ipp" URI scheme could be used to encode the hostname and the resource path
		- Having the user enter the URI exposes too many details to the user, including the detail about the fact that IPP is actually being used. Users need not be aware of which print protocol is being used.
* GOOD:
	+ IPP Get-Printer-Attributes with printer-uri set to a well-known Printer resource path
		- "/ipp/print"
* BETTER:
	+ IPP Get-Printer-Attributes with printer-uri set to "/"
	+ Examine the "printer-uri-supported" attribute; use the first URI in the list
	+ IPP Get-Printer-Attributes with printer-uri set to first URI
* BEST:
	+ IPP Get-Services operation
		- Coming with System Control Service
		- Is this really going to be better?
			* Yes, expected to have metadata associated with each URI specifying the class of service
	1. Validate User Access to Printer

Selecting a printer is misleading to the user if the user isn’t allowed to use the selected printer. Therefore, access restrictions should be validated before selection confirmation (queue creation, etc.) is done on the client system.

Options

* BAD:
	+ Do Nothing
		- The user may choose a printer but not be able to use it due to not having access credentials (username or password or whatever) to use that printer
* GOOD:
	+ IPP Validate-Job operation
		- Use the defaults, but provide the credentials to allow the user access to be determined
	1. Get Printer Options

Once the user has selected a printer, it is necessary for the print system to understand the capabilities that the printer device's print service provides. This includes what print job payload formats can be consumed by the print service, the available options and default choices, and so forth. It also includes other information about the device itself, such as its location. Some of this is done at relationship creation time (queue creation time), perhaps by consulting information stored statically in the printer. It may be that this information can all be retrieved from the printer itself. This is basically the print dialog's activity between the time that the user performs an action to request that the print dialog be presented, and the time that the dialog is presented to the user, populated with the available option choices.

Options

* SAD:
	+ Depend on a-priori knowledge about a particular model as a way of listing options for the model of device identified as the target
		- Model specific print drivers fall in this bucket
* GOOD:
	+ IPP Get-Printer-Attributes Operation
		- Request includes no printer attributes; only operation attributes
		- Reply will contain the job template attributes for all PDLs
	+ Client guesses at what attributes may work or not work for a given PDL, or uses a-priori knowledge
* BETTER:
	+ IPP Get-Printer-Attributes Operation
		- Any specific attributes?
	+ Process results; decide on a PDL
	+ IPP Get-Printer-Attributes Operation
		- Request includes the document-format attribute with value specifying the chosen PDL
		- Reply will contain the job template attributes appropriately filtered ("colored") for that particular document-format
	1. Check constraints between presented options

Printer features and options are presented typically in a print dialog. Some of these have states that have relationships with other options' states, where one cannot be in a particular state if another one is too. These are known as constraints, and they must be calculated any time the state of a control changes state. There are various ways that this can be done.

Options

* POOR:
	+ IPP Validate-Job
		- Every time a control is changed, the client sends IPP Validate-Job with attribute values corresponding to current state of controls
* GOOD:
	+ IPP Validate-Job
		- When "Print" button is pressed, confirms the job creation / submission will succeed (authentication, etc.)
		- Client depends on this operation to perform constraints validation printer-side
* BETTER:
	+ IPP Get-Printer-Attributes
		- Printer Object implements job-constraints-supported
		- Printer Object implements job-resolvers-supported
	+ <Local processing of constraints>
	+ IPP Validate-Job
		- When "Print" button is pressed, confirms the job creation / submission will succeed (authentication, etc.)
		- Constraints validation already handled client-side
	1. Submitting a Print Job

Once the user has decided on options, the print job is generated and ultimately made available to the printer in some fashion. There are several different ways that this may occur.

* + 1. Submitting a print job with document data

This is the classical way that a print job is sent from the client to the print service: first a job is created, and then the job information and payload content are sent from the client to the print service.

Options

* POOR:
	+ IPP Print-Job
		- No pre-flight checks
		- The printer may reject it but only after it has been transmitted.
		- Better to check ticket and content types first.
* GOOD:
	+ IPP Validate-Job
		- Pre-flight checks the job by validating the job attributes, document type, authentication and transport encryption upgrades (if needed)
	+ IPP Print-Job
		- Creates the job and sends the payload in one operation
		- However, the Job object’s URI isn’t usually known until the job transmission is complete
		- Doesn’t work well with flow-controlled (low-end) printers
* BETTER:
	+ IPP Validate-Job
		- Pre-flight checks the job by validating the job attributes and document type, authentication and transport encryption upgrades (if needed)
	+ IPP Create-Job
		- Returns immediately with the job URI for monitoring and ticket processing status
		- If there is a problem then Create-Job may fail the same as Validate-Job would, but may not, which is why we do a Validate-Job first (so that there isn’t a zombie job there)
		- Once the job is created, the client will receive a list of the actual job processing attributes from the IPP Printer.
		- Response to this operation will include xxx-actual job attributes that could be used to detect substitutions that would be used by the Printer Object. Observing this, the client may decide whether to cancel the job rather than submit the document with this job. If the original job was cancelled, the client could create another job with a new set of attributes submitted, or error out and not submit a job at all
		- Allows an opportunity to perform a Cancel-Job operation during document submission
	+ IPP Send-Document
		- Payload transmission is de-coupled from the creation of the job
		- Multiple documents can be sent to build up a compound job
		- Client MUST check to see if value of "multiple-document-jobs-supported" is "true", to see if it is OK to do multiple Send-Document operations to the same Job object.
		1. Submitting a print job with document references

This is a slightly different way that a print job is sent from the client to the print service: a job is created and made available for retrieval by the print service, and when the print job the job information and job payload content are sent by the client to the print service.

Options

* POOR:
	+ IPP Print-URI
		- No pre-flight checks
		- Printer may reject it but only after it has been transmitted
		- Better to check ticket and content types first
* GOOD:
	+ IPP Validate-Job
		- Pre-flight checks the job by validating the job attributes and document type
	+ IPP Print-URI
		- Creates the job and sends a URL to where the payload can be retrieved in one operation
		- Printer Object "pulls" the document file rather than being given it by the client
		- However, the Job object’s URI isn’t usually known until the job transmission is complete
		- Printer may respond with client-error-document-access-error status code, or might add document-access-error to job-state-reasons
		- URI may not be accessible at time of processing
* BETTER:
	+ IPP Validate-Job
		- Pre-flight checks the job by validating the job attributes and document type
	+ IPP Create-Job
		- Returns immediately with the job URI for monitoring and ticket processing status
		- If there is a problem then Create-Job will fail the same as Validate-Job would
	+ IPP Send-URI
		- Payload URI transmission is de-coupled from the creation of the job
		- Printer may respond with client-error-document-access-error status code, or might add document-access-error to job-state-reasons
		- URI may not be accessible at time of processing
			* (How to handle this appropriately? What recommendations should be provided?)
	1. Monitoring print job status

While the print job is being processed, users may wish to know whether it is proceeding successfully, or whether there are conditions that they need to handle that are preventing processing from proceeding, such as a media jam, open covers, marking agents depleted, and so forth.

For those options below that involve polling the Printer Object, the degree to which the option is better or worse is due in no small part to the polling frequency. The interval should be tuned so that the frequency of queries is not so great that it burdens the Printer Object or Job Object or the network, but not so small that there is an undesirable lag between when an event occurs and when the user is notified. It is certainly NOT a best practice in any case if a client is polling as fast as the network can handle traffic.

Options

* POOR:
	+ IPP Get-Jobs / IPP Get-Printer-Attributes
		- Monitor the value of the printer-state attribute and the state of all jobs
		- Not precise; polling for status without knowing the actual job ID
		- Polling is generally not desirable
			* See above regarding polling intervals
* GOOD:
	+ IPP Get-Job-Attributes / IPP Get-Printer-Attributes
		- Monitor the value of printer-state attribute as well as targeted monitoring of a specific job's status
		- Polling is generally not desirable
			* See above regarding polling intervals
* BETTER:
	+ IPP Create-Printer-Subscriptions / IPP Get-Notifications / IPP Get-Job-Attributes
		- Asynchronous / long running queries for notifications that don’t require polling
		- When you see that a job has completed, query the state of that job at that time
		- Printer state changes will be provided by subscribing to the printer; subscribing to the job will provide less information and not be as useful
	1. Canceling a Print Job

It may be that the user wants to terminate a job before it has been fully processed, for whatever reason. There are things that must be done to ensure that the client has decisively cleaned up the state of the Job Object if the client is responsible for canceling the job. Clients’ leaving broken Job objects on the Print service is bad behavior.

There is also a dependency between the options below and how the job was submitted.

Options

* BAD:
	+ IPP Print-Job operation
	+ Client stops sending chunks
* POOR:
	+ IPP Print-Job operation
	+ Client stops sending chunks
	+ IPP Cancel-Job operation request for the job via a second connection, which for some printers could result in a PDL interpreter hang because the last chunk sent didn’t stop on a "statement" boundary
* GOOD:
	+ IPP Create-Job operation
	+ IPP Send-Document operation
		- Potentially truncating job during Send-Document payload transmission
	+ IPP Cancel-Job operation
	1. Getting printer supplies status

Some administrative tasks, like checking consumables levels, are presented to end users in some cases, such as during print job status or in print dialogs. This is useful to end-users and should be supported.

Options

* POOR:
	+ Don’t use IPP but use some proprietary protocol or platform-specific extension to IPP
		- The point is to use only IPP extensions based on open standards (i.e. PWG standard) and this violates that core principle
* GOOD:
	+ IPP Get-Printer-Attributes
		- Printer must implement JPS3 "printer-supply" attribute
* BETTER
	+ IPP Create-Printer-Subscription operation + IPP Get-Notifications operation
	+ IPP Get-Printer-Attributes operations
1. Attributes and Their Use in Operations

Some attributes that IPP has labeled as optional should always be used as a best practice. Below are some of these attributes and how they should be used in various contexts.

* 1. Explicit "document-format" Selection

While IPP Printer Objects provide a default document format (which is known via the document-format-default attribute), as a general principle, it is much better for a client to explicitly provide the document-format attribute with all operations relating to validating or submitting a document payload to the printer (Validate-Job, Print-Job, Send-Document).

* 1. Prefer "media-col" Attribute To "media" Attribute

Given a Printer Object that supports both "media" and "media-col" attributes, a client should prefer to include the "media-col" attribute with operations that accept one of these attributes. This is true for when "media" and "media-col" are top-level attributes as well as when "media" or "media-col" may be included within other collection attributes, such as "job-sheets", "job-error-sheet", "job-accounting-sheets", and others.

* 1. Prefer "finishings-col" Attribute To "finishings" Attribute

Given a Printer Object that supports both "finishings" and "finishings-col" attributes, a client should prefer to include the "finishings-col" attribute with operations that accept one of these attributes.

* 1. Using "ipp-attribute-fidelity"

TBD

* 1. Using "pdl-override"

TBD

1. HTTP Protocol Usage

IPP currently uses HTTP/1.1 for its transport. IPP/2.0 and other IPP specifications have specified some of the facilities of HTTP that IPP clients and servers should support in order to provide the semantics that IPP needs to provide a great user experience. Even so, there are best practices that should be followed.

* 1. HTTP/1.1 Expect Header

As defined in [RFC 2616 "HTTP/1.1"], the "Expect" header allows the client to check with the server on the HTTP connection negotiation before sending the HTTP request payload.

The IPP client should implement the following:

* On first request to a printer, include the "Expect: 100-continue" header.
* Wait up to 1 second for a response.
* If no response is received, remember this for the next request so that you don't have the 1-second delay; continue sending the request.
* If a 100 (continue) status code is returned, continue sending the request
* If a 301 (moved permanently) or 302 (moved temporarily) status code is returned, redirect the request to the new URI \*or\* fail/report an error depending on the security requirements of the Client (redirection is generally unexpected)
* If a 400 (Bad Request) status code is returned, remember this (don't use Expect header) and re-send the POST request. This Printer is technically non-conforming since it fails RFC 2616 requirements for a HTTP/1.1 server.
* If a 401 status code is returned, re-send the POST request with the requested credentials.
* If a 403 status code is returned, fail/report an error.
* If a 426 status code is returned, send an OPTIONS \* request to upgrade to TLS, then re-send the POST request.

The IPP server should implement the following:

* Return status code 403 for unauthorized client addresses when the HTTP level authentication or authorization is not adequate
* Return status code 200 with an IPP response containing the client-error-not-authorized status code when the IPP level authentication or authorization is not adequate
* Status codes 301 and 302 are not recommended
* Return status code 400 only if problems are detected with the HTTP request itself
* Return status code 200 with an IPP response containing the client-error-bad-request status code if problems are detected with the IPP operation
1. Security Considerations

TBD

* What you might do to ensure that the documents submitted remain private
* Using the [IPPS URI]
1. References
	1. Informative References

[PWG5100.12] R. Bergman, H. Lewis, I. McDonald, M. Sweet, "IPP/2.0 Second Edition", PWG 5100.12-2011, February 2011, ftp://ftp.pwg.org/pub/pwg/candidates/cs-ipp20-20110214-5100.12.pdf

[PWG5100.14] F. Last author list or standards body, "IPP Everywhere", 5100.14-2013, January 2013, ftp://ftp.pwg.org/pub/pwg/candidates/cs-ippeve10-20130128-5100.14.pdf

[RFC2616] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, " Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.ietf.org/rfc/rfc2616.txt

[RFC2910] R. Herriot, S. Butler, P. Moore, R. Tuner, J. Wenn, "Internet Printing Protocol/1.1: Encoding and Transport", RFC 2910, September 2000, http://www.ietf.org/rfc/rfc2910.txt

[RFC2911] R. deBry, T. Hastings, R. Herriot, S. Isaacson, P. Powell, "Internet Printing Protocol/1.1: Model and Semantics", RFC 2911, September 2000, http://www.ietf.org/rfc/rfc2911.txt

[RFC3196] T. Hastings, C. Manros, P. Zehler, C. Kugler, H. Holst, "Internet Printing Protocol/1.1: Implementer's Guide", RFC 3196, November 2001, http://www.ietf.org/rfc/rfc3196.txt

[RFC6762] S. Cheshire, M. Krochmal, "Multicast DNS", RFC 6762, February 2013, <http://www.ietf.org/rfc/rfc6762.txt>

[RFC6763] S. Cheshire, M. Krochmal, "DNS-Based Service Discovery", RFC 6763, February 2013, <http://www.ietf.org/rfc/rfc6763.txt>

[REFERENCE] F. Last author list or standards body, "Title of referenced document", Document Number, Month YYYY, URL (if any)

1. Authors' Addresses

Primary authors:

Smith Kennedy

Hewlett-Packard Co.

11311 Chinden Blvd. MS 506

Boise, ID 83714

smith.kennedy@hp.com

The authors would also like to thank the following individuals for their contributions to this white paper:

Evan Williams - Kentucky State Board of Recreation

1. Change History
	1. February 5, 2013

Initial revision.

* 1. March 20, 2013

Resolved issues from feedback provided during the IPP conference call on February 25, 2013, as documented in teleconference meeting minutes and author's own notes.

1. Added Validate-Job operation as operation to be used during printer selection process to validate access by client / user
2. Replaced previous Section 5 "Conformance Requirements" with new Section 5 "Attributes and Their Use in Operations"
3. Replaced previous Section 6 "Internationalization Considerations" with new Section 6 "HTTP Protocol Usage"
4. Added updated list of references