[MS-PSSO]:
Print Services System Overview

This document provides an overview of the Print Services System Overview Protocol Family. It is intended for use in conjunction with the Microsoft Protocol Technical Documents, publicly available standard specifications, network programming art, and Microsoft Windows distributed systems concepts. It assumes that the reader is either familiar with the aforementioned material or has immediate access to it.

A Protocol Family System Document does not require the use of Microsoft programming tools or programming environments in order to implement the Protocols in the System. Developers who have access to Microsoft programming tools and environments are free to take advantage of them.

Abstract

The Print Services System Overview document describes a distributed system of Print Servers that manage printers and make them available to Print Clients. One or more servers may be used, each server independently managing one or more printers. Clients use the component protocols to submit print jobs, manage jobs, receive job notifications, and administer printer drivers and Print Queues.

Revision Summary

Date	Revision History	Revision Class	Comments
08/14/2009	0.1	Major	First Release.
09/25/2009	1.0	Major	Updated and revised the technical content.
11/06/2009	1.1	Minor	Updated the technical content.
12/18/2009	2.0	Major	Updated and revised the technical content.
01/29/2010	3.0	Major	Updated and revised the technical content.
03/12/2010	3.1	Minor	Updated the technical content.
04/23/2010	4.0	Major	Updated and revised the technical content.
06/04/2010	4.0.1	Editorial	Revised and edited the technical content.
07/16/2010	4.1	Minor	Clarified the meaning of the technical content.
08/27/2010	4.2	Minor	Clarified the meaning of the technical content.
10/08/2010	4.2	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	4.2	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	4.2	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	4.2	No change	No changes to the meaning, language, or formatting of the technical content.

Contents

1 Introduction

This Protocol Family System Document (PFSD) is primarily intended to cover the Protocol Family as a whole. In conjunction with Member Protocol Technical Documents (TDs), which are intended to cover Member Protocols, it presents the rules for information exchange relevant to those Member Protocols and the Protocol Family that are used to interoperate or communicate with a Windows operating system in its various environments.

This document describes the Print Services System, which supports the use and management of a distributed print infrastructure. This document describes how the Print Services System can be scaled from workgroup use to domain-based networks, and how Print Servers and Print Clients can use the member protocols of the system to share one or more printers between one or more Print Clients.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

access control list (ACL)
Active Directory
authentication
device
domain
DNS
driver package
driver store
endpoint
enhanced metafile format (EMF)
enhanced metafile spool format (EMFSPOOL)
Graphics Device Interface (GDI)
page description language (PDL)
PostScript
print job
Printer Control Language (PCL)
printer driver
registry
remote procedure call (RPC)
security descriptor

The following terms are defined in [MS-RPRN]:

language monitor
port monitor
print processor
print provider

The following terms are specific to this document:

deployed printer connection: A computer connection or user connection as specified in [MS-GPDPC].

IHV: Independent Hardware Vendor. In the context of this document, an IHV indicates a printer manufacturer, such as Canon or Hewlett Packard.

print spooler: The component (service) that implements the Print Services System on Windows Print Clients and Print Servers. The spooler buffers and orders **print jobs**, as well as converts print job data to printer-specific formats.

The following protocol abbreviations are used in this document:

EMFSPOOL: Enhanced Metafile Spool Format

GPDPC: Group Policy: Deployed Printer Connections Extension

GPOL: Group Policy: Core Protocol

IPP: Internet Printing Protocol

LDAP: Lightweight Directory Access Protocol

LPR: Line Printer Daemon Protocol

PAN: Print System Asynchronous Notification

PAR: Print System Asynchronous Remote Protocol

RAP: Remote Administration Protocol

RPRN: Print System Remote Protocol

SMB: Server Message Block

SMB2: Server Message Block Version 2.0

WPRN: Web Point-and-Print Protocol

XPS: Open XML Paper Specification

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [RFC2119]. Note that in [RFC2119] terms, most of these specifications should be imperative, to ensure interoperability. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

Any specification that does not explicitly use one of these terms is mandatory, exactly as if it used MUST.

1.2 References

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[C706] The Open Group, "DCE 1.1: Remote Procedure Call", C706, August 1997, http://www.opengroup.org/public/pubs/catalog/c706.htm

[MS-BRWS] Microsoft Corporation, "Common Internet File System (CIFS) Browser Protocol Specification", July 2007.

[MS-CIFS] Microsoft Corporation, "Common Internet File System (CIFS) Protocol Specification", September 2009.

[MS-FSCC] Microsoft Corporation, "File System Control Codes", July 2007.

[MS-FSSO] Microsoft Corporation, "File Access Services System Overview", August 2009.

[MS-GPDPC] Microsoft Corporation, "Group Policy: Deployed Printer Connections Extension", August 2007.

[MS-GPOL] Microsoft Corporation, "Group Policy: Core Protocol Specification", June 2007.

[MS-PAN] Microsoft Corporation, "Print System Asynchronous Notification Protocol Specification", January 2007.

[MS-PAR] Microsoft Corporation, "Print System Asynchronous Remote Protocol Specification", June 2007.

[MS-RAP] Microsoft Corporation, "Remote Administration Protocol Specification", March 2007.

[MS-RPCE] Microsoft Corporation, "Remote Procedure Call Protocol Extensions", January 2007.

[MS-RPRN] Microsoft Corporation, "Print System Remote Protocol Specification", June 2007.

[MS-SMB] Microsoft Corporation, "Server Message Block (SMB) Protocol Specification", July 2007.

[MS-SMB2] Microsoft Corporation, "Server Message Block (SMB) Version 2 Protocol Specification", July 2007.

[MS-WPRN] Microsoft Corporation, "Web Point-and-Print Protocol Specification", August 2007.

[RFC1179] McLaughlin III, L., "Line Printer Daemon Protocol", RFC 1179, August 1990, http://www.ietf.org/rfc/rfc1179.txt

[RFC1831] Srinivasan, R., "RPC: Remote Procedure Call Protocol Specification Version 2", RFC 1831, August 1995, http://www.ietf.org/rfc/rfc1831.txt

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, http://www.ietf.org/rfc/rfc2818.txt

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

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

[RFC4511] Sermersheim, J., "Lightweight Directory Access Protocol (LDAP): The Protocol", RFC 4511, June 2006, http://www.rfc-editor.org/rfc/rfc4511.txt

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.ietf.org/rfc/rfc2119.txt

1.2.2 Informative References

[MS-ADLS] Microsoft Corporation, "Active Directory Lightweight Directory Services Schema", June 2007.

[MS-ADSC] Microsoft Corporation, "Active Directory Schema Classes", June 2007.

[MS-ADSO] Microsoft Corporation, "Active Directory System Overview", July 2009.

[MS-AUTHSO] Microsoft Corporation, "Windows Authentication Services System Overview", August 2009.

[MS-DRDM] Microsoft Corporation, "Directory Replication and Data Management (DRDM) Remote Protocol Specification", July 2007.

[MS-EMFSPOOL] Microsoft Corporation, "Enhanced Metafile Spool Format", July 2007.

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

[MS-GPSO] Microsoft Corporation, "Group Policy System Overview", February 2009.

[MS-NRPC] Microsoft Corporation, "Netlogon Remote Protocol Specification", March 2007.

[MS-SPNG] Microsoft Corporation, "Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) Extension", January 2007.

[MS-WSO] Microsoft Corporation, "Windows System Overview", January 2010.

[MS-WUSP] Microsoft Corporation, "Windows Update Services: Client-Server Protocol Specification", September 2007.

[MSDN-XMLP] Microsoft Corporation, "A First Look at APIs For Creating XML Paper Specification Documents", http://msdn.microsoft.com//msdnmag/issues/06/01/xmlpaperspecification/default.aspx.

2 Overview

Section 1, "Introduction", describes this Protocol Family System Document. This section introduces the system that is being documented.

2.1 System Summary

The Print Services System provides functionality for managing a distributed print infrastructure of Print Clients, **print jobs**, and shared printers. One or more printers are shared among multiple Print Clients using a Print Server. Printers are represented as Print Queues located on Print Servers. A Print Client has one or more connections to Print Queues shared by one or more Print Servers, comprising a simple hub-spoke server-client model.

The Print Services System addresses a number of challenges that exist in distributed environments, including the following:

- The system enables a large number of people to print to a small number of printers.
- The system enables an administrator to enforce which printers can be used by each user, and when they can be used.
- The system automates the process of installing and updating necessary **printer drivers** on numerous computers that send jobs to remote printers.

The Print Services System can be deployed within a **domain**-based network, or in a home or workgroup environment. With Windows implementations, each Print Client can also act as a Print Server and share locally connected printers with other Print Clients.

Using the Print Services System, Print Clients can enumerate the shared Print Queues on a Print Server, establish connections to shared Print Queues, and download printer drivers. Print Clients submit print jobs to the Print Queues, and can obtain notifications about printer status, job status, and job progress. Print Clients can also remotely manage print jobs. For example, after a Print Client submits a print job, the Print Client can cancel the print job, alter the priority of the print job, or resume the print job if it is paused.

Administrative clients (those with server administration authorization) can use the Print Services System to manage Print Client connections to shared Print Queues, manage Print Queues, manage jobs submitted by other clients, and manage printer drivers and all other aspects of the Print Services System.

2.2 List of Member Protocols

The member protocols of the Print Services System include the protocols described in this section, as well as the standard protocols IPP and LPR. The Print Services System uses the print data formats of the XML Paper Specification (see [MSDN-XMLP]) and the enhanced metafile spool format (see [MS-EMFSPOOL]).

Print System Remote Protocol, as specified in [MS-RPRN]. This protocol supports synchronous printing and spooling operations between a client and server, including print job control and Print Services System management. This protocol also provides status notifications (defined by the Print Services System) to the Print Client. An enhanced replacement for this protocol is specified in [MS-PAR].

Print System Asynchronous Remote Protocol, as specified in [MS-PAR]. This protocol supports printing and spooling operations between a client and server, including print job control and Print Services System management. This protocol also provides status notifications (defined by the Print

Services System) to the Print Client. This protocol is designed to be used asynchronously by Print Clients whose implementations permit them to continue execution without waiting for a **remote procedure call (RPC)** method call to return. This protocol is an enhanced replacement for the Print System Remote Protocol [MS-RPRN].

Print Asynchronous Notification Protocol, as specified in [MS-PAN]. This protocol is used by Print Clients asynchronously to receive print status notifications from a Print Server and to send back responses to those notifications. A set of notifications and responses are defined together as a notification type. In contrast to the status notification capabilities included in the Print System Remote Protocol ([MS-RPRN]) and the Print System Asynchronous Remote Protocol ([MS-PAR]), the RPC interfaces and methods defined by this protocol provide a transport mechanism for arbitrary, **IHV**-extensible notification types. This protocol is used by IHV-provided components running on the Print Server to trigger the display of a user interface on the Print Client.

Web Point-and-Print Protocol, as specified in [MS-WPRN]. This protocol is used in conjunction with IPP and enables a client to download printer driver software from a Print Server in the client network, from a Web site, or directly from a printer. Based on the Hypertext Transfer Protocol (HTTP).

Enhanced Metafile (EMF) Spool Format, as specified in [MS-EMFSPOOL]. The EMFSPOOL structure specifies a metafile format that can store a print job in portable form. The stored print job contains information for printing a document outside the control of the original application, either on the same computer or on another computer. An EMFSPOOL metafile is "played back" when its records are parsed and processed and the print job is sent to its destination.

SMB Access Protocols, as specified in [MS-CIFS], [MS-SMB], [MS-SMB2], and [MS-FSCC]. Used in the Print Services System for communication with legacy Windows Print Clients and Print Servers (those that do not support [MS-RPRN]/[MS-PAR]) to submit print job information. This group of protocols is used by command-line-based "copy to printer share" operations.

Remote Administration Protocol, as specified in [MS-RAP]. This protocol performs remote administrative functions, including share maintenance and printer maintenance on LAN Manager servers. It is used in the Print Services System for communication with legacy Windows Print Clients or Print Servers ("legacy" collectively used for those Print Clients and Print Servers that do not support [MS-RPRN]/[MS-PAR]) to manage Print Queues.

Group Policy: Deployed Printers Connection Extension, as specified in [MS-GPDPC]. This protocol supports managing connections to printers that are hosted by Print Servers and shared by multiple users. The Print Server component of this protocol enables an administrator to configure printer connections. The Print Client component of this protocol enables a user to discover the printer connections that have been configured.

2.3 Relevant Standards

CIFS: Common Internet File System (CIFS/1.0) Protocol, as specified in [MS-CIFS]. The basis for the SMB protocol family used by the Print Services System for some file transfer operations.

HTTP over TLS: Hypertext Transfer Protocol, as specified in [RFC2818]. Used in the Print Services System as the transport protocol for member protocols IPP and [MS-WPRN].

IPP: Internet Printing Protocol, as specified in [RFC2910] and [RFC2911]. Used in the Print Services System for printing from non-Windows clients, or in cases where network security requirements mandate firewall settings that preclude use of the protocols described in [MS-PAR] or [MS-RPRN].

LDAP: Lightweight Directory Access Protocol, as specified in [RFC4511]. Used by Print Server role to publish shared printer information to directory, used by Print Client role to discover available shared printers in domain.

LPR: Line Printer Daemon Protocol,as specified in [RFC1179]. Used in the Print Services System for printing from non-Windows clients, mainly Unix and Linux systems (LPD Clients).

RPC: Remote Procedure Call Protocol, as specified in [RFC1831], [C706] and [MS-RPCE]**.**Used as transport for the protocols described in [MS-RPRN] (RPC over SMB), [MS-PAR] (RPC over TCP/IP), and [MS-PAN] (RPC over TCP/IP).

3 Foundation

This section describes the theoretical and practical information needed to understand this document and this system.

3.1 Background Knowledge and System-Specific Concepts

This section summarizes:

- Background knowledge required to understand this document.
- Concepts specific to this system.

3.1.1 Background Knowledge

Background knowledge that is helpful in understanding the Print Services System Overview includes the following:

- RPC concepts
- Data encryption using Secure Socket Layers (SSL) and Transport Layer Security (TLS)
- Windows **Graphics Device Interface (GDI)**
- Windows Security (see [MS-WSO])

Additionally, having knowledge of the following components and concepts is important:

- **Print spoolers**
- Print queues
- Printers and print data formats
- Printer drivers

3.1.1.1 Print Spoolers

A Print Spooler running on a Print Client exposes local printing APIs to applications, receives the print output of an application, and sends it to a shared Print Queue on a Print Server. The print output is a data stream in a format that can be processed on a specific printer, or it is a generic metafile that can be translated to a format that can be processed on a specific printer.

A Print Spooler running on a Print Server shares Print Queues to expose them for use by Print Clients; it receives the print output data stream sent by a client and buffers the data stream and sends it to a target printer.

On both a Print Server and Print Client, the print spooler manages printer driver components and other components that aid in the creation of application print output and translation of metafiles, and allows applications to obtain metrics and status information about printers.

The Windows implementations of the Print Client and Print Server roles are provided by the print spooler component. Each Windows system runs a print spooler. Therefore, each Windows computer can act as a Print Client or a Print Server.

3.1.1.1.1 Print Spooler Service

The print spooler service is a service running on each computer participating in the Print Services System. The print spooler service implements the Print Client and Print Server roles, allowing each participating system to act as a Print Client, Administrative Client, or Print Server for the Print Services System.

Implementation of the Print Client role requires implementation of the Print Server role in the print spooler service due to the dual client/server nature of the protocol described in [MS-RPRN], which necessitates an **endpoint** on the Print Client to receive notifications.

If an implementation acts as Print Client only, it MUST support all methods described in [MS-RPRN] and [MS-PAR], but can return 0 for the number of supported objects from all enumeration methods, and fail all open printer calls.

For the Print Server role, the print spooler service registers the RPC endpoints for the protocols described in [MS-PAR] <1>, [MS-RPRN] and [MS-PAN]. For the Print Server role, the Print Spooler service registers the LPD server endpoint, if configured to support LPD.

For the Print Server role, the Print Spooler service exposes local interfaces extending Internet Information Server to support the IPP and Web Point-and-Print ([MS-WPRN]) protocols, if configured to support IPP.

For the Print Client role, the Print Spooler service handles the [MS-RPRN] notification callback functions exposed in the [MS-RPRN] RPC endpoint. Additionally, the print spooler service exposes local interfaces used by client applications to print, obtain Print Queue status, administrate Print Queues, or perform other print specific actions.

3.1.1.2 Print Queues

A Print Queue is an abstract component residing on a Print Server to which print jobs are submitted. In many component protocol documents, the phrase Print Queue is used interchangeably with the word Printer. However, in this system document, Print Queue will always refer to the abstract component, and Printer will refer to the printers.

3.1.1.3 Printers and Print Data Formats

A printer is a physical **device** provided by an IHV. A printer consumes a data stream representing a description of a document to be printed. The data format of that data stream is defined or selected by the IHV that designed the printer.

There are several de facto standards for these data formats, including **PostScript**, Adobe PDF, Hewlett-Packard (HP) **Printer Control Language (PCL)**, and the XML Paper Specification. Other vendor-specific print data formats are in use as well.

In order for applications to use different printer technologies in a uniform manner, the Microsoft Windows® graphical subsystem abstracts the details of the print data formats. Windows relies on the services provided by printer driver and **print processors** to abstract details of data formats. Regardless of the data format, all print data streams sent from a print client to a print server are collectively referred to as "payloads." Print payloads are sent from a print client to a print server using one of the member protocols of the Print Services System.

3.1.1.4 Printer Drivers and Print Processors

Printer drivers and print processors are printer specific and implement a set of functions that Windows calls to perform the following tasks:

- Convert application document creation and drawing commands (including drawing of text) into one of the data formats understood by the respective printer.
- Provide Windows with information about metrics and capabilities of the printer, such as physical paper dimensions, color capabilities, paper path options, duplexing options, and so on.
- Provide user interface functionality enabling users to configure settings and default behavior for a printer.

3.1.2 System-Specific Concepts

System-specific concepts that are helpful in understanding this system document include the following:

- Windows printing architecture
- Print Client communication with a Print Server
- Protocols supporting different Print Clients
- Protocol redirectors on Print Servers
- Enabling Print Queues to be discoverable
- Translating application content to a print data format
- Supporting client-side rendering and server-side rendering
- Sending print data to a printer via a **port monitor**

3.1.2.1 Windows Printing Architecture

Applications built using Windows interfaces can use device-independent functions for creating print jobs and sending them to many types of printers. This device independence is based on the Windows architecture provided by print spoolers and printer drivers. Print spoolers and printer drivers are replaceable and can be written targeting specific client and printer hardware, while allowing applications to continue to use known interfaces.

3.1.2.1.1 Print Client Communication with Print Server

Applications running on Print Clients typically access the local print spooler, which then redirects the application requests to the print services provided by the Print Server. This indirection isolates applications from protocol and capabilities negotiation and other complexities of the protocol, and keeps application code portable across multiple versions of the Print Services System.

The print spooler residing on the Print Server implements Print Queues. These Print Queues are associated with printer drivers, printer ports, and the printers to which they route print data. Print Clients make requests of Print Servers to perform a variety of functions, as shown in the following diagram.

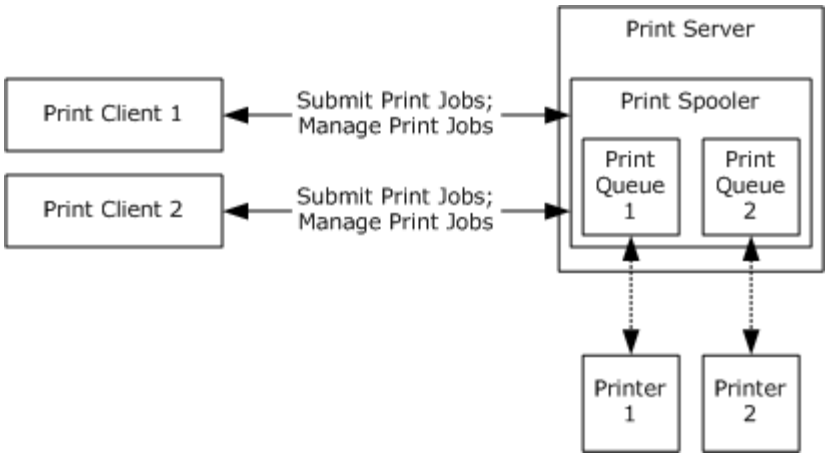

Figure 1: Print Clients communicating with Print Server

3.1.2.1.2 Protocols Supporting Different Print Clients

Different Print Services System protocols and externally defined system protocols are used to allow a variety of Print Clients to communicate with a Print Server as shown in the following diagram. The Print Services System versions, capabilities, and protocol support are fully described in section 4.5.

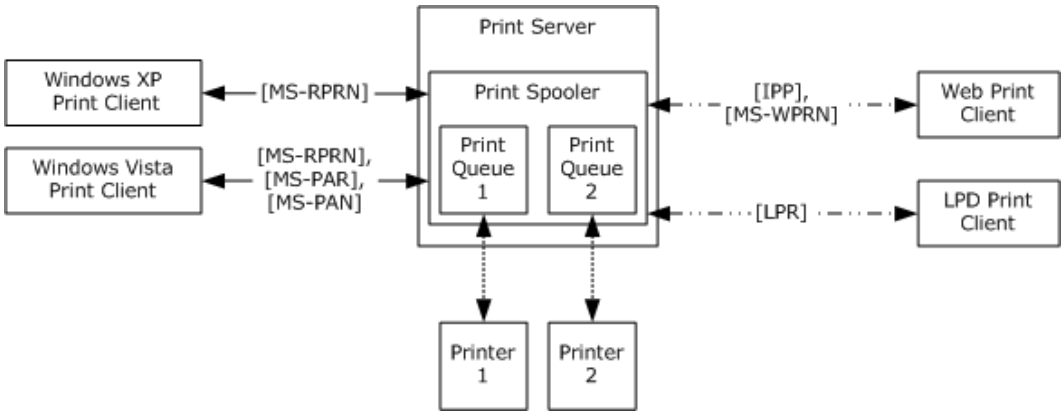

Figure 2: Various implementations of Print Clients interacting with a Print Server

3.1.2.1.3 Protocol Redirectors on Print Servers

Simultaneous server-side support for printing services and file access services using the SMB Access Protocols and the protocol specified in [MS-RAP] require that redirector components be implemented on the server because each of these protocols exhibits print-specific and file access-specific functionality on a single protocol end point, and no common abstract data model (ADM) is used between the print services and file access services.

These redirector components transparently route file access-specific requests to local calls to the file access services, and translate print-specific requests to local calls to the print spooler component implementing the Print Server role. The redirector components then translate the results of these local calls back to the appropriate protocol response described in [MS-SMB] or [MS-RAP]. These translations are described in section 5.

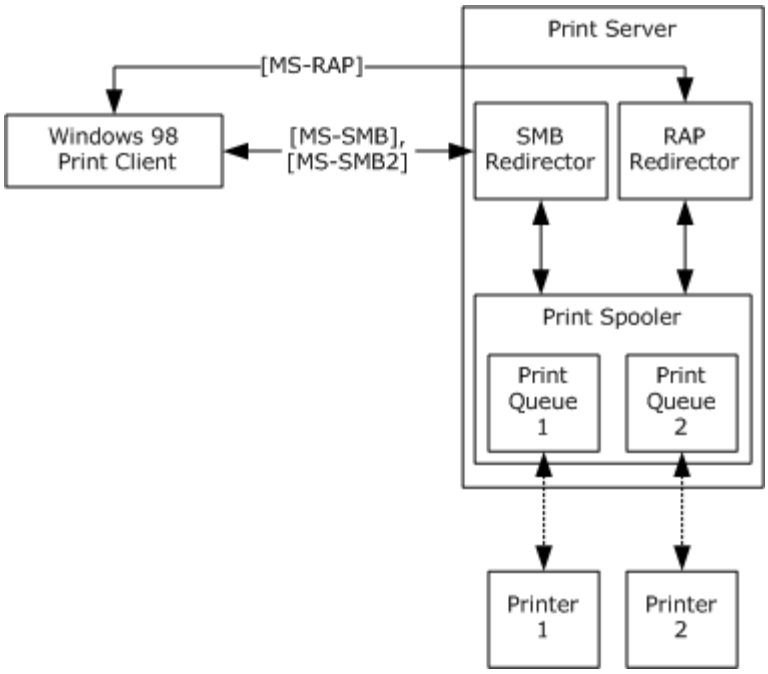

Figure 3: Redirectors routing data from a Print Client to the Print Spooler

3.1.2.1.4 Enabling Print Queues to be Discoverable

To enable Print Queues to be discoverable by Print Clients, the Print Services System uses the **Active Directory** System, as described in [MS-ADSO], to store and provide information about shared queues. Although a Print Client can contact a Print Server directly to discover the Print Queues on the Print Server, using the Active Directory System implemented on the LDAP Server allows all Print Queues on all Print Servers to be enumerated. Furthermore, the Active Directory System isolates Print Clients from the requirement to locate Print Servers in the domain. The details of the interactions between the Print Services System and the Active Directory System are described in [MS-RPRN] section 2.3.

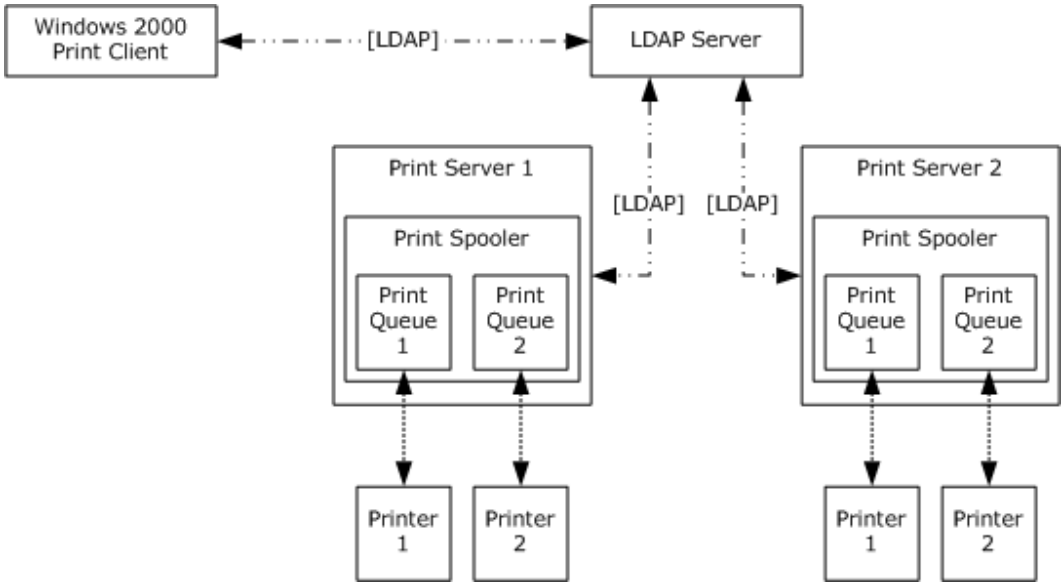

Figure 4: Obtaining information about Print Queues from an LDAP Server

3.1.2.2 Translating Application Content to a Print Data Format

Windows works with the print spooler to manage the translation of application content to printer-specific data formats using two configurations, as shown below. The print spooler determines the data format for the print job using the mechanism described in section 3.1.4.9.1 of [MS-RPRN].

In one configuration, Windows loads the printer driver directly into the application process. When the application prints, the data stream produced is already converted to the printer-specific data format in the application process, and is then passed to the print spooler service for buffering and further processing. This configuration is independent of the actual data format produced, and is referred to as producing a "RAW" data format.

In another configuration, Windows does not load the printer driver into the application process, isolating the application from the actual translation work. Instead, Windows records document creation and drawing commands in a metafile, which is then passed to the print spooler service for buffering and conversion to a printer-specific data format. The metafile can have various data formats, such as the format described in [MS-EMFSPOOL], the XML Paper Specification, or a vendor-defined format. The print spooler service uses a print processor (usually part of the printer driver) to assist in the translation of the metafile to the printer-specific data stream. A print processor loads the metafile and replays the contained commands to the printer driver, which then performs the actual translation process to the printer-specific data stream.

3.1.2.3 Supporting Client-Side Rendering and Server-Side Rendering

Client-side rendering is one configuration of a client-server printing model. In this configuration, the print spooler service on the Print Client translates the metafile to the printer-specific data stream before sending it to the print spooler service on the Print Server.

In a server-side rendering configuration, the print spooler service on the Print Client sends the metafile without modification to the print spooler service on the Print Server, where the metafile is subsequently converted to the printer-specific data format.

In order to support both client-side rendering and server-side rendering, it is a requirement that print processors and printer drivers are available to Print Servers and Print Clients. Windows addresses this requirement in part by using a mechanism called "Point-and-Print," which allows a Print Client to download a printer driver directly from a Print Server.

3.1.2.4 Sending Print Data to a Printer via a Port Monitor

In order to send print data to a printer, a Print Server uses a port monitor module. Port monitors implement an abstraction API that allows the print spooler service to send and receive data through a printer port (such as USB or parallel ports) on the computer acting as the Print Server and connected to a printer, or through a network connection (such as TCP/IP) to a printer.

3.2 System Purposes

Printers are often acquired and deployed in an organization over time, resulting in a mixed infrastructure comprised of many different printers (with varying color capabilities, paper sizes, resolutions, features) and capacities. Some printers are intended for heavy use by large groups of users, and others for lighter use by smaller groups. The Print Services System provides a method for standardizing access and maintenance of multiple printers in a network, workgroup, or home environment. These features benefit administrators responsible for printer deployment, administration, and maintenance. These features also benefit users of Print Clients by allowing printer resources to be shared, and allowing users to choose the most appropriate printers to use for different print jobs.

3.3 System Use Cases

The Print Services System is designed to support scenarios that allow users shared access to printers connected to Print Servers or to computers used by other users. When a user wants to print to a shared printer, the system facilitates this action by providing the following:

- Lists of shared printers
- Connecting the user to the Print Queue associated with the desired printer and enabling the user to install the appropriate printer driver
- Transporting the print data from the Print Client to the Print Queue
- Processing multiple print jobs in order of their arrival or priorities
- Keeping the user apprised of the status of the print job

The Print Services System also facilitates the management of a large number of shared printers across multiple Print Servers by enabling an administrator to remotely install and configure shared Print Queues on Print Servers, to manage the use of shared Print Queues on Print Servers, and to view and manage print jobs that have been submitted by all users to shared Print Queues on Print Servers.

Although there is a distinction between the roles and capabilities of Administrator and User when participating in the Print Services System, both roles can be conducted on the same computer. And although there is a distinction between a Print Client and a Print Server as actors in scenarios using the Print Services System, a Print Client can assume the role of a Print Server if it enables other Print Clients to discover and use (or Administrative Clients to administer) a Print Queue installed on itself.

This section provides the use cases that describe the functionality of the Print Services System in terms of actors that participate in this system and their goals. The use case participants most often

include a Print Client and a Print Server, and they are triggered by a User performing print operations, or an Administrator performing administrative operations.

Section 3.3.3 provides a table that summarizes the use cases described in section 3.3.4.

3.3.1 Stakeholders and Interests Summary

Print Client: The Print Client connects to a Print Queue on a Print Server, sends print data to the Print Queue, displays notifications about the printer or print job, and disconnects from the Print Queue. The Print Client may take the form of an application that the User is printing from, or as a component of an operating system.

Administrative Client: The Administrative Client connects to Print Servers to establish, configure, and manage Print Queues hosted on the Print Servers. The Administrative Client can also be used to manage print jobs sent to the Print Queues. The Administrative Client takes the form of an administrative tool application.

Print Server: The Print Server hosts Print Queues, each Print Queue corresponding to a shared printer. The Print Server stores and uses printer drivers that are necessary for processing print data before it is sent to printers. Depending on the configuration of the Print Server and Print Client, the Print Server may supply printer drivers to Print Clients, so Print Clients can perform print data processing locally. The Print Server processes requests from Print Clients and the Administrative Clients (such as connect, disconnect, upload driver, download driver, route print data). The Print Server manages access from Users and Administrators to the Print Queues, sends notifications about Print Queues and print jobs to Users and Administrators, and routes print data to the appropriate Print Queues.

Internet Browser: An Internet Browser can be used to discover shared Print Queues on a Print Server. The internet server component on the Print Server uses local print spooler APIs to obtain a list of Print Queues and presents the list to the client as an HTML page. An Internet Browser can be used for this purpose if a firewall or other network restrictions do not allow the core Print Services System protocols to be used. Once a User knows the name of the Print Queue, the User can enter the address in the Print Client user interface to connect to the printer.

User: The individual who uses a Print Client to choose a Print Queue from a list of available shared Print Queues, sends a print job to the Print Queue, and monitors the progress of the print job.

Administrator: The individual responsible for installing and configuring printers within a deployed Print Services System, as well as monitoring, managing, and troubleshooting printing functions. The Administrator uses an Administrative Client.

Architect: The individual who designs systems or applications that support distributed printing features.

Developer: The individual who designs and implements a Print Client, a Print Server, a printer driver, an application that prints, or an administrative tool for managing Print Servers and print jobs. The developers of a Print Services System or a subcomponent of the system are responsible for implementing the incoming and outgoing interfaces of the system as documented in this document and the respective technical documents of the member protocols of the Print Services System.

Tester and QA Personnel: The individuals who determine if the Print Services System meets design and functionality requirements.

3.3.2 Supporting Actors and System Interests Summary

The Print Services System has the following supporting actors with noted interests:

Active Directory System: The Active Directory System is used to store a list of shared Print Queues in a domain environment, The Print Services System uses the LDAP protocol to publish information about each shared Print Queue to the Active Directory System. Print Clients can then discover the available shared Print Queues, allowing a User to choose a Print Queue to which to send a print job.

Group Policy System: The Print Services System makes use of the Group Policy System, as specified in [MS-GPSO], to restrict Print Clients from accessing specified Print Servers, as well as to remotely push pre-configured Print Queue connections to Print Clients. The Print Services System uses a Group Policy Services extension protocol, as described in [MS-GPDPC], to distribute these pre-configured Print Queue connections to Print Clients.

3.3.3 Use Case Diagrams

The following table provides an overview of use cases which span the functionality of the Print Services System. Use case extensions are noted within each use case. Detailed descriptions for these use cases are provided in section 3.3.4.

Use case group	Use cases
Provision or Delete a Print Queue	Provision a Print Queue using [MS-RPRN] - Administrative Client a. Install print processor on Print Server Provision a Print Queue using [MS-PAR] - Administrative Client a. Install print processor on Print Server Delete a Print Queue using [MS-RPRN] - Administrative Client a. Delete objects no longer referenced on Print Server Delete a Print Queue using [MS-PAR] - Administrative Client a. Delete objects no longer referenced on Print Server
Locate and Establish a Connection to a Print Queue	Locate and Establish a Connection to a Print Queue in a Domain Environment using [MS-RPRN] - Print Client a. Print Client downloads printer driver b. Print Client connections are restricted by Group Policy settings c. Print Client registers for notifications of Print Queue status Locate and Establish a Connection to a Print Queue in a Domain Environment using [MS-PAR] - Print Client a. Print Client downloads printer driver b. Print Client connections are restricted by Group Policy settings c. Print Client registers for notifications of Print Queue status Locate and Establish a Connection to a Print Queue in a Workgroup Environment using [MS-RPRN] - Print Client a. Print Client downloads printer driver b. Print Client registers for notifications of Print Queue status Locate and Establish a Connection to a Print Queue in a Workgroup Environment using [MS-PAR] - Print Client a. Print Client downloads printer driver b. Print Client registers for notifications of Print Queue status

Use case group	Use cases
	Locate and Establish a Connection to a Print Queue from an Internet Client using IPP; download a printer driver using [MS-WPRN] - Print Client
Manage a Print Queue	Set Permissions on a Print Queue using [MS-RPRN] - Administrative Client Set Permissions on a Print Queue using [MS-PAR] - Administrative Client
Submit a Print Job	Submit a Print Job using [MS-RPRN] - Print Client a. Print Client obtains notifications about print job status b. Print Client obtains notifications for IHV-defined components on Print Server Submit a Print Job using [MS-PAR] - Print Client a. Print Client obtains notifications about print job status b. Print Client obtains notifications for IHV-defined components on Print Server Submit a Print Job using IPP - Print Client Submit a Print Job using SMB Protocol Family - Print Client
Manage a Print Job	Manage Print Jobs Submitted by Self using [MS-RPRN] - Print Client Manage Print Jobs Submitted by Self using [MS-PAR] - Print Client Manage Print Jobs Submitted by All Users using [MS-RPRN] - Administrative Client Manage Print Jobs Submitted by All Users using [MS-PAR] - Administrative Client Manage a Print Job Submitted from Command Line using [MS-RAP] - Print Client

The following use case diagrams illustrate the use cases described in this section, dividing them between those initiated by an Administrative Client and those initiated by a Print Client.

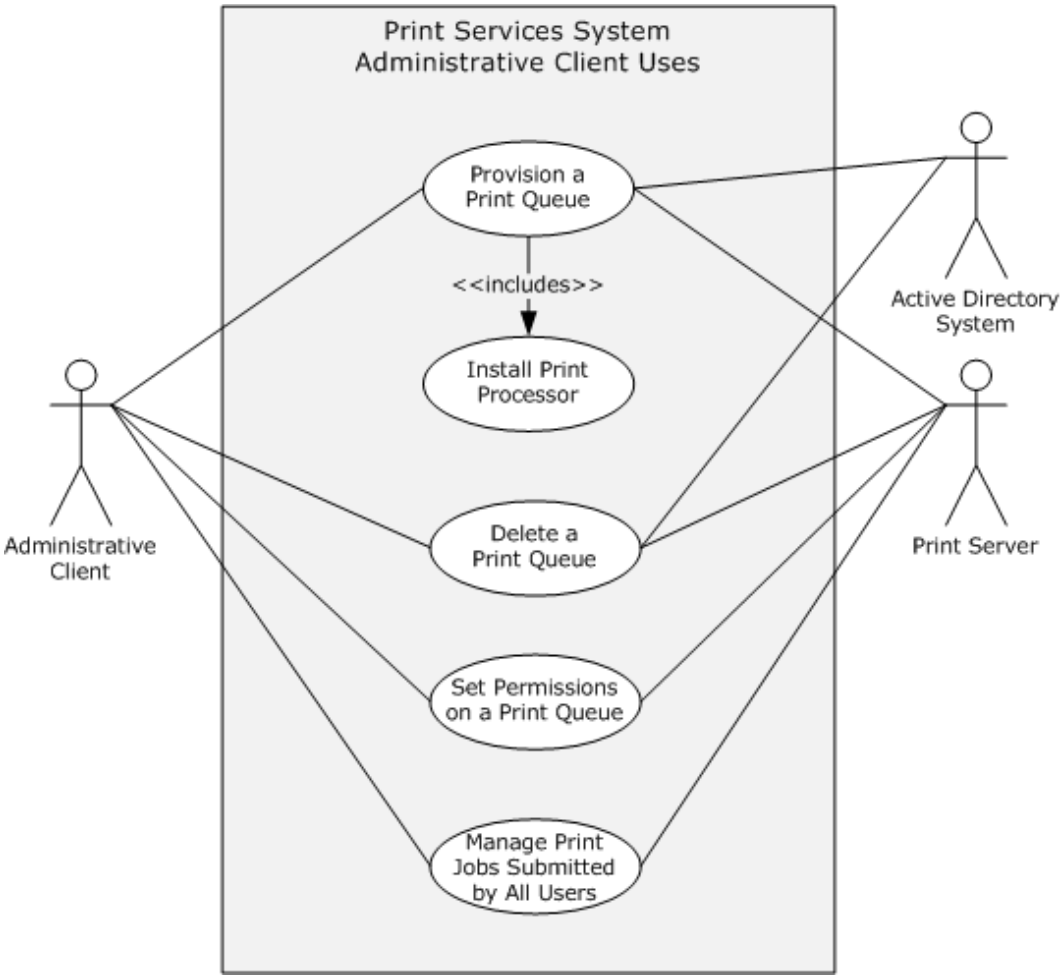

Figure 5: Print Services System use cases initiated by an Administrative Client

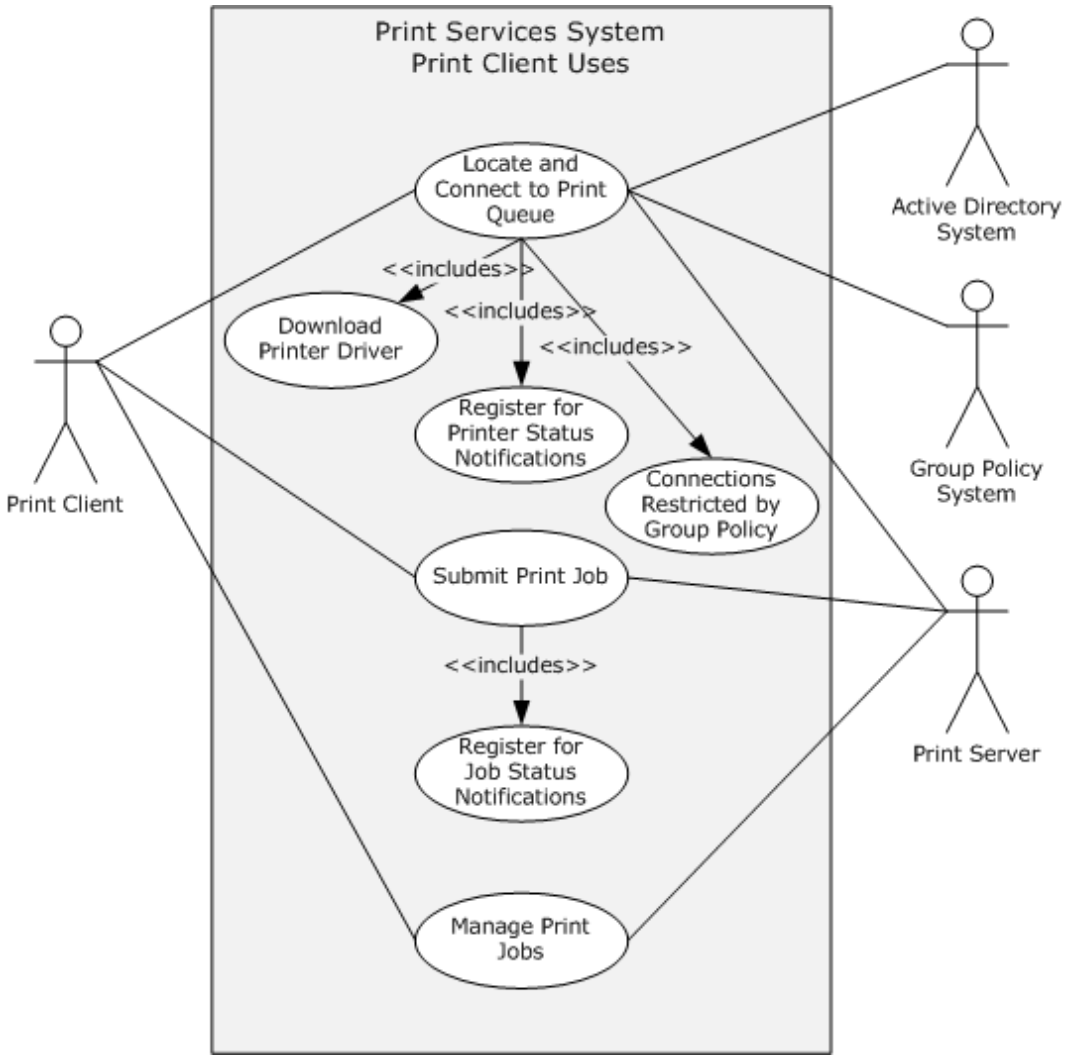

Figure 6: Print Services System use cases initiated by a Print Client

3.3.4 Use Case Descriptions

3.3.4.1 Provision a Print Queue -- Administrative Client

This use case has two variations:

(a) Performing the use case using the protocol described in [MS-RPRN]

(b) Performing the use case using the protocol described in [MS-PAR]

Variation (a) - Performing the use case using the protocol described in [MS-RPRN]

Goal: To make a Print Queue available on a Print Server, subsequently allowing a User to select the Print Queue, establishing all attributes and components necessary to make the Print Queue discoverable and accessible to a Print Client.

Context of Use: Prior to a User being able to select a shared Print Queue to which to send a print job, an Administrator needs to use an administrative tool to create and share a Print Queue on a Print Server. In Windows, this tool is the Print Management Console. This shared Print Queue corresponds to a printer. In addition to creating the Print Queue, attributes and components of the Print Queue MUST be defined (such as the port to which the printer is connected, and the printer driver and print processor component that will convert the print job data to drawing commands that the printer uses). After an Administrator provisions the Print Queue with the necessary information, a User using a Print Client (an application or operating system) will be able to send a print job to the printer.

Direct Actor: The direct actor is the Administrative Client.

Primary Actor: The primary actor is an Administrator.

Supporting Actors: The supporting actors are the Print Server and the Active Directory System.

Stakeholders and Interests:

- Administrative Client: The Administrative Client will be used by an Administrator to participate in this use case to add the necessary information to a Print Server for allowing a printer to be shared by multiple Print Clients.
- Administrator: An Administrator will participate in this use case using an Administrative Client to configure a Print Server to share a printer among multiple Users.
- Print Server: A Print Server will be configured by this use case so that a printer connected to the Print Server can be shared among multiple Users via a queue located on the Print Server.
- Active Directory System: The Active Directory System will be updated by this use case so that a listing of the Print Queue is available via a Print Client when a User attempts to locate a Print Queue to which to send a print job.

Preconditions: Print Server and Administrative Client are connected using a network connection and both are members of the domain. The print spooler service MUST be operational on the Print Server and Administrative Client. The Active Directory System is available and operational.

Minimal Guarantee: The Administrative Client is denied access and no provisioning operations occur.

Success Guarantee: A Print Queue corresponding to a printer is established on a Print Server and the Active Directory System is updated to allow multiple Print Clients to locate and share the printer.

Trigger: An Administrator initiates the process of provisioning a Print Queue using an administrative tool to add a new Print Queue to a Print Server.

Main Success Scenario:

Using the SMB protocol family, the Administrative Client copies the files for the printer driver to a directory on the Print Server that is accessible via an SMB share.

The Administrative Client uses the protocol described in [MS-RPRN] to install the printer driver on the Print Server. The driver is associated with a Print Queue in a later step of this use case. In a network of Print Clients having different architectures, different versions of the same driver are uploaded and installed by the Administrative Client, allowing the Print Server to provide printer drivers to Print Clients whose architecture differs from the Print Server's architecture.

The Administrative Client ensures that an executable module for a port monitor is present on the Print Server (copying it there using the SMB protocol family, if necessary), and then installs the port monitor on the Print Server using the protocol described in [MS-RPRN]. A port monitor implements and exports methods the Print Server calls locally to interface with a port to which a printer is connected. Ports for the port type implemented by the port monitor can only be added to the Print Server after the port monitor is added.

The Administrator provides information about the port to which the printer is connected to the Administrative Client, which then uses the protocol described in [MS-RPRN] to direct the Print Server to add a specified printer port. A port is either a physical hardware port (such as a parallel or USB port), or a network port (such as a WSD port or a TCP/IP port). The port is assigned to the Print Queue in a later step.

If the printer supports custom paper formats, called "forms," the Administrative Client can use the protocol described in [MS-RPRN] to add definitions for those custom forms to the Print Server. Once added, such forms can be made available for use with the Print Queue.

The Administrative Client uses the protocol described in [MS-RPRN] to create a Print Queue on the Print Server. The Administrator uses the Administrative Client to assign a printer driver, a printer port, and a print processor to the Print Queue during this step, and also sets access permissions, metadata (name of printer, physical location, and comments), time of availability, name under which the printer is shared, separator page settings, and processing priority.

The Administrative Client uses the protocol described in [MS-RPRN] to direct the Print Server to publish the Print Queue to the Active Directory System, which the Print Server satisfies using the LDAP protocol to create a directory object for the Print Queue, supplying the UNC name (server name plus share name) of the Print Queue. (Print Clients can then use the LDAP protocol to discover the Print Queue.)

Extension (a) - Install print processor on Print Server: Between steps 4 and 5 previously described, if the installed printer driver does not contain a print processor, the Administrative Client may use the protocol described in [MS-RPRN] to install a print processor on the Print Server.

Variation (b) - Performing the use case using the protocol described in [MS-PAR]: All details identical to variation (a) except that the protocol described in [MS-PAR] is used to locate and establish the connection to the Print Queue.

3.3.4.2 Delete a Print Queue -- Administrative Client

This use case has two variations:

(a) Performing the use case using the protocol described in [MS-RPRN]

(b) Performing the use case using the protocol described in [MS-PAR]

Variation (a) - Performing the use case using the protocol described in [MS-RPRN]

Goal: To delete a previously provisioned Print Queue available on a Print Server, subsequently no longer allowing a User to select the Print Queue, removing all attributes and components necessary to make the Print Queue discoverable and accessible to a Print Client.

Context of Use: When a Print Queue is no longer needed, an Administrator uses an administrative tool to delete a Print Queue from a Print Server. In Windows, this tool is the Print Management Console. After an Administrator deletes the Print Queue, a User using a Print Client (an application or operating system) will no longer be able to send a print job to the printer.

Direct Actor: The direct actor is the Administrative Client.

Primary Actor: The primary actor is an Administrator.

Supporting Actors: The supporting actors are the Print Server and the Active Directory System.

Stakeholders and Interests:

- Administrative Client: The Administrative Client will be used by an Administrator to participate in this use case to delete a previously provisioned Print Queue.
- Administrator: An Administrator will participate in this use case using an Administrative Client to configure a Print Server to delete a previously provisioned Print Queue.
- Print Server: A Print Server will be configured by this use case so that a printer connected to the Print Server can no longer be shared among multiple Users via a queue located on the Print Server.
- Active Directory System: The Active Directory System will be updated by this use case so that the listing of the Print Queue will no longer be reported to a Print Client when a User attempts to locate a Print Queue to which to send a print job.

Preconditions: Print Server and Administrative Client are connected using a network connection and both are members of the domain. The print spooler service MUST be operational on the Print Server and Administrative Client. The Active Directory System is available and operational.

Minimal Guarantee: The Administrative Client is denied access and no delete operations occur.

Success Guarantee: A Print Queue is deleted from a Print Server and the Active Directory System is updated to no longer return the Print Queue in directory queries.

Trigger: An Administrator initiates the process of deleting a Print Queue using an administrative tool to delete a Print Queue on a Print Server.

Main Success Scenario:

The Administrative Client uses the protocol described in [MS-RPRN] to direct the Print Server to unpublish the Print Queue from the Active Directory System, which the Print Server satisfies using the LDAP protocol to delete the directory object for the Print Queue.

The Administrative Client uses the protocol described in [MS-RPRN] to delete the Print Queue on the Print Server.

Extension (a) - Deleting objects no longer referenced on Print Server: After deleting the Print Queue, the Administrative Client may use the protocol described in [MS-RPRN] to determine if the port, the print processor, and the printer driver previously used by the deleted Print Queue are still in use by other Print Queues; and if that is not the case, the Administrative Client may use the protocol specified in [MS-RPRN] to delete said objects from the server.

Variation (b) - Performing the use case using the protocol described in [MS-PAR]: All details identical to variation (a) except that the protocol described in [MS-PAR] is used to delete the Print Queue.

3.3.4.3 Locate and Establish a Connection to a Print Queue in a Domain Environment -- Print Client

This use case has two variations:

(a) Performing the use case using the protocol described in [MS-RPRN]

(b) Performing the use case using the protocol described in [MS-PAR]

Variation (a) - Performing the use case using the protocol described in [MS-RPRN]

Goal: To make a connection to a shared Print Queue in a Domain environment so documents can be printed by the User.

Context of Use: The User wants to print a document and does not have a printer attached to the local computer, but is member of a domain and knows that there are shared Print Queues in the enterprise available for use. The User decides to use one of the share Print Queues in the enterprise and initiates this use case.

Direct Actor: The direct actor is the Print Client.

Primary Actor: The primary actor is a User.

Supporting Actors: The supporting actors are Print Server, the Active Directory System, and Group Policy System (in Extension (b)).

Stakeholders and Interests:

- Print Client: The Print Client will display a list of shared Print Queues from which a user will select a Print Queue, and then the Print Client will establish a connection to the selected Print Queue.
- Print Server: The Print Server provides information about the shared Print Queue to the Print Client, makes the printer driver available to the Print Client for download if necessary, and allows the Print Client to connect to the Print Queue.
- Active Directory System: The Active Directory System store and provide a list of shared Print Queues that the User will view in the Print Client user interface.
- Group Policy System: The Group Policy Services Deployed Printer Connections Extension is a Supporting Actor in Extension (b). The Group Policy System is used to push a list of shared Print Queues to the Print Client so that the Print Client has pre-established connections to specified Print Queues. Additionally, Group Policy settings are inspected by the Print Client to restrict access to certain Print Servers.
- User: The User wants to view a list of available shared Print Queues and select one to which to send a print job.

Preconditions: The print spooler services are operational on the Print Client and Print Server. Both are members of the domain and connected with a network. The network is operational. the Active Directory System are available and operational.

Minimal Guarantee: The Print Queue is located but the user is informed via the Print Client user interface that a connection cannot be established due to insufficient permissions.

Success Guarantee: A connection to a shared Print Queue has been established by the Print Client and the connection can be used to submit print jobs to the Print Server.

Trigger: A user initiates this use case by starting the Add a printer command in the Print Client's user interface.

Main Success Scenario:

1. The Print Client queries the Active Directory System for a list of shared Print Queues in the domain.

2. The Active Directory System replies with a list of shared Print Queues.

3. The user of the Print Client selects a Print Queue from the query results.

4. The Print Client opens a handle to the selected Print Queue using the protocol described in [MS-RPRN].

5. The Print Client queries the Print Server for information about the Print Queue, such as which printer driver (and version) is being used, and the hardware ID of the printer represented by the Print Queue.

6. If the Print Client already has a copy of the printer driver installed, the Print Client creates a local Print Queue Proxy object representing the connection to the Print Queue on the Print Server (see Extension (a) for the case of when the Print Client does not already have a copy of the printer driver installed).

   Extension (a) - Print Client downloads printer driver:

   Step 6: If the Print Client does not have a copy of the printer driver installed, the Print Client obtains the printer driver from one of the following sources, identifying the printer driver using the hardware ID of the Print Queue obtained earlier:

   - From an administrative share on the Print Server
   - From Windows Update Services
   - From the User (for example, from a CD)

   The Print Client then locally installs the printer driver.

   Extension (b) - Print Client makes connections only to Print Queues specified by Group Policy [MS-GPDPC] deployed printer connections:

   Replace previous steps 1 through 3 with the following steps:

   1. An administrator defines one or more **deployed printer connection** Group Policy settings for one or more domain users or user groups - each Group Policy setting includes details of one or more Print Queues shares by the Print Server.

   2. Using the protocol specified in [MS-GPDPC], the Print Client retrieves the deployed printer connections for the logged on user and the current computer and proceeds to automatically make connections to all Print Queues specified in the Group Policy settings, without user interaction (steps 4 through 6 for each Print Queue connection).

   Extension (c) - Print Client registers for notifications of Print Queue status:

   Step 6 is followed by:

7. Once the Print Queue Proxy representing the connection has been created, the Print Client opens a handle to the selected Print Queue using the protocol described in [MS-RPRN]. The Print Client then registers with the Print Server for some or all supported notifications on that handle, which will then be sent to the Print Client when available. The Print Client will reflect state changes of the server Print Queue in the local Print Queue Proxy object (such state changes can be, among others: error state, online/offline state, printer paused, list of queued jobs, associated printer driver - fetching and installing a different or updated printer driver if necessary, and so on.)

Variation (b) - Performing the use case using the protocol described in [MS-PAR]: All details identical to Variation (a) except that the protocol described in [MS-PAR] is used instead of the protocol described in [MS-RPRN].

3.3.4.4 Locate and Establish a Connection to a Print Queue in a Workgroup Environment -- Print Client

This use case has two variations:

(a) Performing the use case using the protocol described in [MS-RPRN]

(b) Performing the use case using the protocol described in [MS-PAR]

Variation (a) - Performing the use case using the protocol described in [MS-RPRN]

Goal: To make a connection to a shared Print Queue in a workgroup environment so documents can be printed by the User.

Context of Use: The User wants to print a document and does not have a printer attached to the local computer, but is member of a workgroup and knows that there are shared Print Queues in the workgroup available for use. The User decides to use one of the shared Print Queues in the workgroup and initiates this use case.

Direct Actor: The direct actor is the Print Client.

Primary Actor: The primary actor is the User.

Supporting Actors: The supporting actors are Print Clients performing a Print Server role.

Stakeholders and Interests:

- Print Client: The Print Client will display a list of shared Print Queues from which a User will select a Print Queue, and then the Print Client will establish a connection to the selected Print Queue.
- Print Clients performing a Print Server role: In this use case, the Print Server role is performed by Print Clients that have shared Print Queues. After a User shares a Print Queue on the computer, the computer performs Print Server functions, including providing printer drivers to Print Clients, and accepting and printing print jobs from Print Clients. These Print Servers announce themselves to all other computers acting as Print Servers in the workgroup, as well as storing a list of all the other computers performing the Print Server role.
- User: The User wants to view a list of available shared Print Queues and select one to which to send a print job.

Preconditions: The Print Server has used the protocol described in [MS-BRWS] to find other computers (Print Servers and Print Clients) in the workgroup, and then used the protocol described in [MS-RPRN] to push the list of shared Print Queues to all Print Servers in the workgroup. The Print Server has also notified all other computers in the workgroup that it is a Print Server itself, using the protocol described in [MS-BRWS]. The print spooler service is operational on Print Server and Print Client. Both are connected with a network connection.

Minimal Guarantee: The Print Queue is located but the user is informed via the Print Client user interface that a connection cannot be established due to insufficient permissions.

Success Guarantee: A connection to a shared Print Queue has been established by the Print Client and the connection can be used to submit print jobs to the Print Server.

Trigger: A user initiates this use case by starting the Add a printer command in the Print Client user interface.

Main Success Scenario:

1. The User initiates the Add a printer action and selects the option to look for a printer.
2. The Print Client queries Print Servers in the workgroup for lists of shared Print Queues.
3. The user of the Print Client selects a Print Queue from presented list of shared Print Queues in the workgroup.
4. The Print Client opens a handle to the selected Print Queue using the protocol described in [MS-RPRN].
5. The Print Client queries the Print Server for information about the Print Queue, such as which printer driver (and version) is being used, and the hardware ID of the printer represented by the Print Queue.
6. If the Print Client already has a copy of the printer driver installed, the Print Client creates a local Print Queue proxy object representing the connection to the Print Queue on the Print Server (see Extensions for the case of when the Print Client does not already have a copy of the printer driver installed).

Extension (a) - Print Client downloads a printer driver:

Step 4: After opening a handle to the selected Print Queue, if the Print Client does not have a copy of the printer driver installed, the Print Client downloads the printer driver from either (a) Windows Update, using the hardware ID of the Print Queue obtained earlier, or (b) from an administrative printer share of the Print Server (or other Print Client) in the workgroup. The Print Client then locally installs the printer driver before proceeding with the rest of the use case.

Extension (b) - Print Client registers for notifications of Print Queue status:

Step 4. Once the Print Queue representing the connection has been created, the Print Client opens a handle to the selected Print Queue using the protocol described in [MS-RPRN]. The Print Client then registers with the Print Server (or other Print Client) for some or all supported notifications on that handle, which will then be sent to the Print Client when available. The Print Client will reflect state changes of the server Print Queue in the local Print Queue object (such state changes can be, among others: error state, online/offline state, printer paused, list of queued jobs, associated printer driver - fetching and installing a different or updated printer driver if necessary, and so on.) Once the Print Client is no longer interested in notifications (for example, because the User closes the user interface that shows status changes), the Print Client unregisters for notifications.

Variation (b) - Performing the use case using the protocol described in [MS-PAR]: All details identical to Variation (a) except that the protocol described in [MS-PAR] is used instead of the protocol described in [MS-RPRN].

3.3.4.5 Locate and Establish a Connection to a Print Queue from an Internet Client using IPP; Download a Printer Driver Using [MS-WPRN] -- Print Client

Goal: To make a connection to a shared Print Queue from an Internet client so documents can be printed by the User.

Context of Use: The User wants to print a document and does not have a printer attached to the local computer. The environment blocks connections using protocols described in [MS-RPRN] and [MS-PAR] (for example, via firewall). As an alternative, IPP and the protocol described in [MS-

WPRN] are available since they use HTTP/HTTPS and are typically allowed by most firewall administrators. Using these protocols, the Print Client allows the User to choose a shared printer via the Internet.

Direct Actor: The direct actor is the Print Client.

Primary Actor: The primary actor is the User.

Supporting Actors: The supporting actors are the Print Server and Internet Browser.

Stakeholders and Interests:

- Print Client: The Print Client can make a connection to a Print Queue when the User enters the name of the Print Queue, as well as downloading a printer driver using the protocol described in [MS-WPRN]. The Print Client sends a print job to the Print Queue using IPP.
- Print Server: The Print Server receives an HTTP GET request from an Internet Browser, generates a list of shared printers in HTML format, and responds to the HTTP GET request with the list to the Print Client. The Print Server also provides a printer driver to the Print Client.
- Internet Browser: The Internet Browser is used to connect to a Print Server and view shared Print Queues on the Print Server. The Internet Browser can also be used to connect the Print Client to a selected Print Queue by executing script code which calls local APIs of the print spooler. Alternately, the User may use the Internet Browser only for finding out the name of the Print Queue, and can then enter that name in the Print Client user interface to connect to the Print Queue.
- User: The User wants to send a print job to a Print Queue but is unable to use the Print Client to discover the shared Print Queues due to network restrictions.

Preconditions: The optional Internet Printing server role component has been installed on the Print Server. The print spooler service is operational on the Print Server and Print Client. An HTTP/HTTPS connection can be established between the Print Client and the Print Server. The User knows the URL of the Print Server.

Minimal Guarantee: A connection cannot be made.

Success Guarantee: A connection can be made.

Trigger: A User initiates this use case by starting the Add a printer command in the Print Client user interface.

Main Success Scenario:

1. The User of an Internet Browser browses to the "http://<HOST>/printers" URL of the Print Server with the address <HOST>.
2. The Print Server returns an HTML page showing the available Print Queues and their status.
3. The User selects a Print Queue to which to connect.
4. The Print Client uses the protocol described in [MS-WPRN] to download the printer driver for the Print Queue from the Print Server.
5. The Print Client installs the downloaded Printer Driver.
6. The Print Client creates a local Print Queue Proxy object that represents the IPP connection to the Print Queue on the Print Server.

Extensions: None.

3.3.4.6 Set Permissions for a Print Queue -- Administrative Client

This use case has two variations:

(a) Performing the use case using the protocol described in [MS-RPRN]

(b) Performing the use case using the protocol described in [MS-PAR]

Variation (a) - Performing the use case using the protocol described in [MS-RPRN]

Goal: To set permissions for a Print Queue, such as the priority of the Print Queue and the times it is available for shared use, as well as who may access it.

Context of Use: The Administrator wants to restrict use of a shared Print Queue to a selected group of users or individual users. Additionally, the Administrator wants to restrict the times of day when the shared Print Queue is available for use.

Direct Actor: The direct actor is the Administrative Client.

Primary Actor: The primary actor is the Administrator.

Supporting Actors: The supporting actors are the Print Server and the Active Directory System.

Stakeholders and Interests:

- Administrative Client: The Administrative Client connects to the Print Server to send information regarding restrictions on the use of shared Print Queues.
- Print Server: The Print Server restricts the use of shared Print Queues as specified by the Administrative Client.
- Administrator: The Administrator wants to restrict the use of a shared Print Queue to a selected group of users.

Preconditions: The Print Server and Administrative Client are connected using a network connection and both are members of the domain. The print spooler service MUST be operational on the Print Server and Administrative Client.

Minimal Guarantee: The Administrative Client denies the administrator the privilege to set permissions based on the Administrator's authorization.

Success Guarantee: The permissions and time restrictions are set as requested.

Trigger: An Administrator initiates the process of setting the permissions for a Print Queue by selecting a Print Queue using an Administrative Client.

Main Success Scenario:

1. The Administrative Client uses the protocol described in [MS-RPRN] to enumerate the Print Queues on the Print Server.
2. The Administrator selects one of the enumerated Print Queues.
3. The Administrative Client uses the protocol described in [MS-RPRN] to open a handle to an existing Print Queue on the Print Server.

4. The Administrative Client uses the LDAP protocol to obtain a list of domain users and groups from the Active Directory System.

5. The Administrator selects users and groups from the list obtained in step 4 and assigns access permissions to form an access control list (ACL) and security descriptor.

6. The Administrative Client uses the protocol described in [MS-RPRN] to set the desired security descriptor on the Print Queue.

7. The Administrator further chooses a time interval in which the Print Queue will accept print jobs from Print Clients.

8. The Administrative Client uses the protocol described in [MS-RPRN] to set the desired availability time interval on the Print Queue.

9. The Administrative Client closes the printer handle.

Extensions: None.

Variation (b) - Performing the use case using the protocol described in [MS-PAR]: All details identical to Variation (a) except that the protocol described in [MS-PAR] is used instead of the protocol described in [MS-RPRN].

3.3.4.7 Submit a Print Job -- Print Client

This use case has four variations:

(a) Performing the use case using the protocol described in [MS-RPRN]

(b) Performing the use case using the protocol described in [MS-PAR]

(c) Performing the use case using the IPP protocol

(d) Performing the use case using the SMB protocol family

Variation (a) - Performing the use case using the protocol described in [MS-RPRN]

Goal: To print a document.

Context of Use: The User using an application capable of printing a document wants to print. A connection to a shared Print Queue has been previously established. The User initiates the Print command from the application.

Direct Actor: The direct actor is the Print Client.

Primary Actor: The primary actor is the User.

Supporting Actors: The supporting actor is the Print Server.

Stakeholders and Interests:

- Print Client: The Print Client sends the print job data to the Print Queue on the Print Server.
- Print Server: The Print Server buffers the print job data sent to the Print Queue to which the Print Client is connected, optionally processes the print job data further, and sends it to the printer associated with the Print Queue.

- User: The User desires a printed copy of content on the computer and chooses the Print function from the Print Client user interface.

Preconditions: The print spooler services are operational on the Print Client and Print Server. Both are members of the domain and connected with a network. The network is operational. The Active Directory System is available and operational.

Minimal Guarantee: The print job was submitted to the Print Server, but no job status feedback could be initiated, so the print job might print, but the Print Client cannot show feedback.

Success Guarantee: The job is submitted and job progress status was received and displayed to the User.

Trigger: A User initiates this use case by selecting the Print command from a printing capable application.

Main Success Scenario:

1. The Print Client opens a printer handle using the protocol described in [MS-RPRN].
2. The Print Client starts a new print job to the Print Server using the protocol described in [MS-RPRN].
3. The Print Client indicates the start of a new logical page to the Print Server, repeatedly sends data for the page, and signals the end of a logical page to the Print Server using the protocol described in [MS-RPRN]. The Print Client repeats this step for all pages in the document.
4. After sending all the pages of the print job to the Print Server, the Print Client ends the print job using the protocol described in [MS-RPRN].
5. The Print Client closes the printer handle using the protocol described in [MS-RPRN].

Extension (a) - Print Client obtains notifications about print job status:

Following the previous step 1 and before step 2: The Print Client registers for change notifications using the protocol described in [MS-RPRN].

In parallel with the remaining steps, the Print Server sends change notifications as the processing of the print job proceeds. The Print Client provides feedback to the User.

Extension (b) - Print Client obtains notifications for IHV-defined components on the Print Server and shows the user interface to the User:

Prior to the previous step 1, the Print Client uses the protocol described in [MS-RPRN] to register for change notifications from the Print Server.

In parallel with the other steps of the use case, when the Print Client receives change notifications signaling the arrival of a new print job, the Print Client will attempt to listen to the Print Queue for connection requests from the Print Server using the protocol described in [MS-PAN]. An IHV-defined printer driver or other IHV-defined component running on the Print Server can act as a notification source using the protocol described in [MS-PAN] and send unidirectional or bidirectional notification messages to the Print Client. One important notification message type for IHV-defined components is used to request that the Print Client shows a user interface to the User.

After step 4, if a connection has been established between the Print Client and the IHV-defined components using the protocol described in [MS-PAN], then the Print Client continues to listen for Print Server change notifications signaling the end of the print job for the monitored Print Queue. The Print Client will close the [MS-PAN] connection once all the jobs have been processed.

Variation (b) - Performing the use case using the protocol described in [MS-PAR]: All details identical to Variation (a) except that the protocol described in [MS-PAR] is used instead of the protocol described in [MS-RPRN].

Variation (c) - Performing the use case using the IPP protocol: All details are identical to Variation (a) except the Main Success Scenario differs as noted below, and there are no extensions addressing job status notifications:

1. User initiates a print job from a Windows application to a Print Queue that is a connected to an IPP Print Queue on the Print Server.

2. The Print Client uses IPP to submit the print job to the IPP Print Queue on the Print Server.

Variation (d) - Performing the use case using the SMB protocol family: All details are identical to Variation (a) except the Main Success Scenario differs as noted below, and there are no extensions addressing job status notifications:

1. The User initiates a print job from the command line by executing a copy /b FILE \\SERVER\PRINTQ command (where FILE is a local file containing print job data, SERVER is the name of the server, and PRINTQ is the name of a shared Print Queue).

2. The Print Client uses the SMB protocol family to submit the file containing the print job data to a printer share on the Print Server.

3.3.4.8 Manage Print Jobs -- Print Client

This use case has five variations:

(a) Performing the use case for jobs submitted by self using the protocol described in [MS-RPRN].

(b) Performing the use case for jobs submitted by self using the protocol described in [MS-PAR].

(c) Performing the use case for jobs submitted by all users using the protocol described in [MS-RPRN].

(d) Performing the use case for jobs submitted by all users using the protocol described in [MS-PAR].

(e) Performing the use case from a command line using the protocol described in [MS-RAP].

Variation (a) - Performing the use case for jobs submitted by self using the protocol described in [MS-RPRN]:

Goal: For a User to manage his or her own submitted print jobs, including pausing them, resuming them, canceling them, changing their priority, changing their order in the queue, or restarting them.

Context of Use: After submitting a print job, a User might need to manage the job for a variety of reasons. For example, a User might want to pause a job in order to print another job of higher priority; or a User might cancel a job when it becomes apparent that the content is flawed and printing would only waste paper.

Direct Actor: The direct actor is the Print Client.

Primary Actor: The primary actor is the User.

Supporting Actors: The supporting actor is the Print Server.

Stakeholders and Interests:

- Print Client: The Print Client is used to display the print jobs submitted by the User, allow the User to select a print job to manage, and allow the User to select management functions.
- Administrative Client (in Variations (c) and (d)): The Administrative Client is used to display the print jobs submitted by all Users, allow the Administrator to select print jobs to manage, and allow the Administrator to select management functions.
- Print Server: The Print Server receives and executes job management functions requested from a Print Client or Administrative Client.
- User: A User wants to manage the printing of a print job, pausing, canceling, or changing the priority of the print job, for example.
- Administrator (in Variations (c) and (d)): An Administrator wants to manage all the print jobs that have been submitted to a Print Queue in order to service the printer, free up the printer for alternate uses, or to intervene when discovering that a very large print job has been sent to a printer not capable of handling such volume.

Preconditions: The print spooler services are operational on the Print Client and Print Server. Both are members of the domain and connected with a network. The network is operational.

Minimal Guarantee: The User or Administrator attempting to perform management functions on a print job will be denied permission via the user interface on the Print Client or the Administrative Client.

Success Guarantee: The User or Administrator attempting to perform management functions on a print job will successfully be able to pause a job, resume a job, cancel a job, or perform other functions.

Trigger: The User opens the Print Queue user interface and looks at queued jobs prior to modifying them (cancel, reorder, pause, resume, and so on).

Main Success Scenario:

1. The Print Client opens a printer handle using the protocol described in [MS-RPRN].
2. The Print Client enumerates jobs scheduled for printing on the printer using the protocol described in [MS-RPRN].
3. The Print Client opens a handle to a specific job using the protocol described in [MS-RPRN]. The Print Server will refuse this request for any job not submitted by the User of the Print Client.
4. The Print Client modifies job settings or job priority using the protocol described in [MS-RPRN].
5. The Print Client closes the printer handle using the protocol described in [MS-RPRN].

Variation (b) - Managing print jobs submitted by self using the protocol described in [MS-PAR]: All details identical to Variation (a) except that the protocol described in [MS-PAR] is used instead of the protocol described in [MS-RPRN].

Variation (c) - Managing print jobs submitted by all Users using the protocol described in [MS-RPRN]:

Goal: For an Administrator to manage print jobs submitted by all Users, including pausing them, resuming them, canceling them, changing their priority, changing their order in the queue, or restarting them.

Context of Use: An Administrator may need to override or reconfigure print jobs that various Users have submitted to multiple Print Queues in order to effectively manage or maintain printing resources. An Administrator may use an Administrative Client or a Print Client to manage print jobs submitted by all Users.

Direct Actor: The direct actor is the Administrative Client or Print Client.

Primary Actor: The primary actor is the Administrator.

Supporting Actors: None.

Preconditions: An Administrator possesses administrative permissions for the Print Queue, enabling the Print Server to permit the request to open a job handle of any job, regardless of who submitted the job.

Variation (d) - Managing print jobs submitted by all Users using the protocol described in [MS-PAR]: All details identical to Variation (c) except that the protocol described in [MS-PAR] is used instead of the protocol described in [MS-RPRN].

Variation (e) - Managing print jobs from a command line using the protocol described in [MS-RAP]:

1. The User uses the 'net print' command to enumerate print jobs on the Print Queue.<2>
2. The Print Client uses the protocol described in [MS-RAP] to enumerate print jobs on the Print Queue of the Print Server.
3. The User uses the 'net print' command to pause, resume or cancel print jobs on the Print Queue.<3>
4. The Print Client uses the protocol described in [MS-RAP] to pause, resume, or cancel a print job.

4 System Context

This section describes the relationship between this system and its environment.

4.1 System Environment

The Print Services System member protocols require the following protocols:

- SMB Access Protocols:
  - Used as transport for the protocol defined in [MS-RPRN] (uses RPC over named SMB pipes)
  - Used for driver download from Print Server (using copy file functionality of the SMB Access Protocols).
- SMB Access Protocols plus [MS-RAP]:
  - Used for managing SMB print scenarios, using the protocol defined in [MS-RAP].
  - Used for managing SMB file copy functionality of the Print Server, using the protocol defined in [MS-RAP].
- HTTP / HTTPS:
  - Used as transport for the protocol defined in [MS-WPRN]
  - Used as transport for IPP
- RPC over TCP/IP:
  - Used as transport for the protocol defined in [MS-PAR]
  - Used as transport for the protocol defined in [MS-PAN]
- Group Policy: Core Protocol [MS-GPOL] (requires prerequisites [MS-NRPC] and [MS-DRDM])
  - Used for the protocol defined in [MS-GPDPC]
- LDAP
  - Used for publishing and query of shared Print Queues
  - Used by the protocol defined in [MS-GPDPC]
- TCP/IP
  - Used as transport by LDAP
  - Used as transport by LPR
  - Used as transport by HTTP
  - Used as transport by HTTPS
  - Used as transport by "RPC over TCP/IP"
  - Used as transport by the SMB protocol family (unless on an IPX or NetBEUI network)
- Common Internet File System (CIFS) Browser Protocol Specification [MS-BRWS]

- Used for workgroup preconditions

The Print Services System member protocols, as defined in [MS-RPRN], [MS-PAR] and [MS-PAN], are RPC-based and require RPC bindings between Print Servers and Print Clients to registered RPC endpoints. If a Print Client or Print Server is unable to register an RPC endpoint or create RPC bindings, then Print Services provided by the local print spooler will only allow the management and use of locally connected printers. Firewalls implemented on Print Clients or Print Servers MUST be configured so that all ports required by member protocols are open for RPC-based communications, or at least open for HTTP-based communication to support the internet printing scenario using IPP and the protocol described in [MS-WPRN].

The Print Services System can run in a domain-based network and workgroup environment. Print Services require that File Share services are installed and enabled (for Point and Print driver download and for SMB and Remote Administration Protocol [MS-RAP] support). Upon creation of a printer queue connection to a Print Server, the Print Client tries to copy the driver from the Print Server (see Use Cases in sections 3.3.4.2 and 3.3.4.3), and if that fails and the Print Client cannot obtain a suitable printer driver otherwise, the Print Queue connection cannot be created.

The Print Services System requires the protocol defined in [MS-BRWS] for Workgroups. Availability of this protocol and the Active Directory System is not determined at startup, but only upon request. If neither the Common Internet File System (CIFS) Browser Protocol ([MS-BRWS]) nor the Active Directory System are available, a Print Client will not list any available shared Print Queues in the network, and as a result, only local functionality of the print spooler is available, unless the user of the Print Client knows the name of the Print Server and enters it manually.

The Print Services System makes use of the Windows Update service, as described in [MS-WUSP], if available. Windows Update is used to find the most up to date or best matching printer drivers for a Print Queue connection. If Windows Update services are unavailable, print services uses other methods to find an appropriate printer driver (see Use Cases in sections 3.3.4.2 and 3.3.4.3).

All member protocols are used between Print Clients and Print Servers.

The Print Services System uses the Active Directory System in domain-based networks. The LDAP protocol is used between Print Server and the Active Directory System and between Print Client and the Active Directory System. In workgroup environments, the Common Internet File System (CIFS) Browser Protocol ([MS-BRWS]) is used for communication between Print Clients and Print Servers, and between Print Servers.

The file system access used by clients (to download printer drivers only) assumes a directory structured file system, and the Print Server SHOULD restrict access using ACLs, so that Print Clients have only read access to the printer driver files. The file system SHOULD support file date/time stamps for driver version comparison purposes.

Any number of Print Servers and Print Clients can be operated in a domain-based network. The number of shared Print Queues clients retain knowledge of in a Windows workgroup is limited to an implementation-defined maximum <4>, which limits the useful number of shared printers in a Windows workgroup network. From the system perspective, the entirety of Print Clients and Print Servers in a given network represents a single instance of the system.

4.2 System Assumptions and Preconditions

The following assumptions and preconditions MUST be satisfied for the Print Services System to operate successfully:

- A network which provides a viable transport for communications between the server and its clients MUST be available.

- The transport protocol for that network MUST be available and configured (for example, the TCP transport MUST be configured with a valid IP address).
- Security package providers MUST be available to the system.
- The durable storage devices used to store the system's state MUST be available on all participating computers.
- The print spooler service MUST be in the Running state on the server and each of the clients participating in the interaction.
- The print spooler MUST be installed on all the computers involved.
- The print spooler is run as a local system and impersonates users on method calls.
- The file (SMB Protocol Family support) server service MUST be in the Running state on the server.
- The Print Client and Print Server SHOULD support the protocol defined in [MS-RPRN].
- The system MUST be configured so that participants can access its services locally or remotely.
- It is assumed that each participant is trusted by the system.
- If Member Protocols supported by the system, as listed in section 2.2, have additional assumptions and preconditions for when that protocol is in use, please see the relevant member protocol specification for details.
- Print Servers MUST have a durable store to place the following objects in:
  - Printer drivers
  - Print jobs
- Print Clients MUST have a durable store to place the following objects in:
  - Printer drivers
- Print Clients and Print Servers MUST have access to local **registry** storage to persist ADM and state information.
- In a domain configuration, Print Clients and Print Servers MUST have access to the Active Directory System provided by the domain.
- **Authentication** services supporting Simple and Protected Negotiate (SPNEGO, as described in [MS-SPNG]) are available to the Print Servers and Clients.
- The Print System Remote Protocol, as described in [MS-RPRN], is a remote procedure call (RPC) interface and therefore has the prerequisites specified in [MS-RPCE] section 1.5 as being common to RPC interfaces.
- The Print System Asynchronous Remote Protocol, as described in [MS-PAR], is a RPC interface and therefore has the prerequisites specified in [MS-RPCE] section 1.5 as being common to RPC interfaces.
- The Print Asynchronous Notification Protocol, as described in [MS-PAN], is a RPC interface and therefore has the prerequisites specified in [MS-RPCE] section 1.5 as being common to RPC interfaces.

- The Print Client has obtained the name of the Print Server that supports the Print Services System. Various protocols may be used to accomplish this. For more information, see [MS-ADLS], [MS-ADSC], and [MS-BRWS].

4.3 System Relationships

4.3.1 Black Box Relationship Diagram

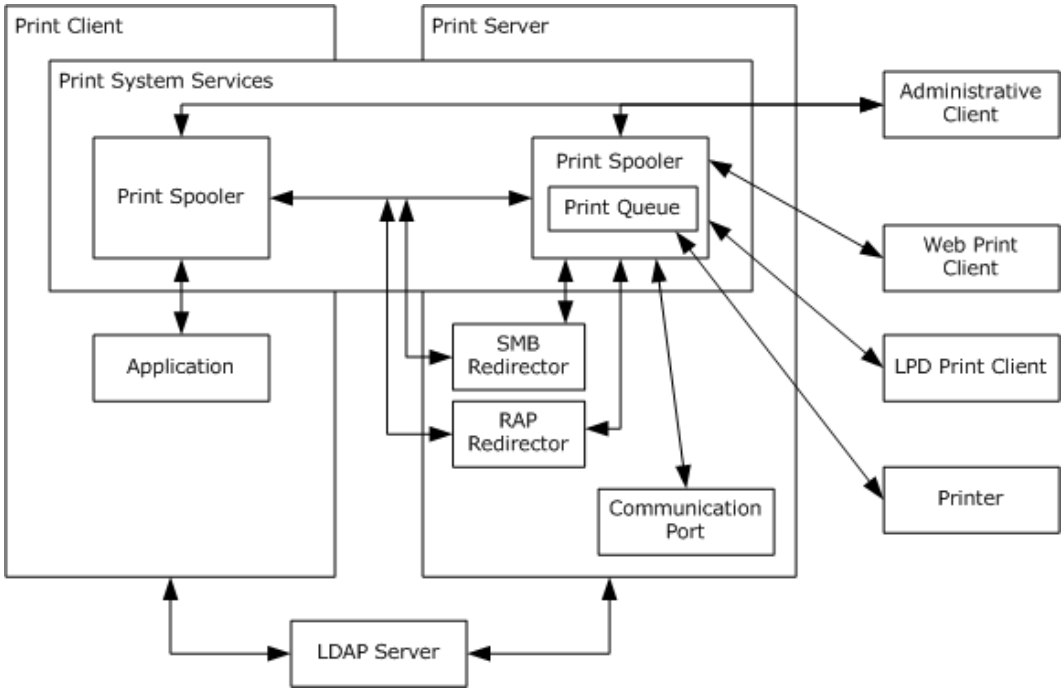

Figure 7: Black Box diagram illustrating components of the Print Services System

4.3.2 System Dependencies

The Print Services System is dependent on the following components and infrastructure:

- Windows Authentication Services System, as described in [MS-AUTHSO].
- Active Directory System (in a domain environment): If the Print Services System is deployed within a domain, the Print Server requires access to the Active Directory System. Shared printers are published to the Active Directory System using LDAP. Print Clients joined to domains also require access to the Active Directory System to discover published printers.
- Networking system to connect Print Server, print clients, and the Active Directory System.
- Printers.
- Application software that provides printing functionality, such as Microsoft Word.

4.3.3 System Influences

The Print Services System is influenced by the following:

- The Active Directory System can propagate policy settings controlling local spooler behavior to Print Clients and Print Servers <5> using the protocol defined in [MS-GPOL].
- These policies control print spooler implementation aspects, for example which Print Servers are considered trusted for printer driver download in Point-and-Print scenarios.
- The protocol described in [MS-GPDPC] influences the Print Services System in regards to deployed printer connections.

4.4 System Applicability

The Print Services System supports the use and management of a distributed print infrastructure. Using Print Servers and Print Clients that implement the Member Protocols described in this document, one or more printers may be shared between one or more Print Clients.

The Print Services System scales from workgroup use, in which printers are shared between computers; to domain-based networks, in which multiple Print Servers are employed in a cluster configuration, and the Print Client configuration is managed by the Active Directory System. The Print Services System also provides a subset of functionality for managing a single printer connected to a single computer.

In addition to protocols supporting communication between Print Clients and Print Servers, the Print Services System also supports externally defined system protocols for the Group Policy System and the Active Directory System. The Print Services System uses the local File System Services of the Print Server to store print jobs and printer drivers. The Print Services System also uses the local registry on the Print Server to persist the ADM of the Print Server. The Print Services System has minimal interaction with other components of Windows.

Managed printers are reflected in the Print Services System as Print Queues on a Print Server. Each Print Queue has an associated printer driver used by the Print Services System and applications to learn about printer capabilities, such as paper formats, color capabilities, and print quality. The associated printer driver is also responsible for conversion of application commands into the vendor-defined **page description language (PDL)** used to print a job on a printer.

Print clients submit print jobs to the Print Queues. The Print Queues are managed by a print spooler component running on the Print Server that buffers and orders print jobs arriving from many Print Clients simultaneously, or jobs arriving at a higher speed than the printers are capable of handling.

4.5 System Versioning and Capability Negotiation

The Print Services System (PSS) evolved over multiple releases of Windows client and server products. Each version of the Print Services System provides additional functionality, while maintaining interoperability with previous versions. For the purposes of this document, the Print Services System functionality is grouped using the following versions:

- PSS 1.0
- PSS 2.0
- PSS 3.0
- PSS 4.0

The functionality in these versions is incremental. In order for a third party Print Server to interoperate with a Windows Print Client, the Print Server MUST support all the functionality for a given version and the versions that preceded it.

For example, if PSS 3.0 included a protocol defining opnums 1 through 10 and PSS 4.0 included a new protocol and an extension to the PSS 3.0 protocol defining a new opnum 11, then Windows Print Clients require that the Print Server supports the new PSS 4.0 protocol as well as the opnum 11 extension to the PSS 3.0 protocol.

Support for some protocols is optional. An implementation can be fully functional in a typical Windows network without implementing the optional protocols.

The Print Client or Administrative Client determines the Print Server's PSS version support using the following sequence of operations, calling the methods described in [MS-RPRN]:

1. The Print Client calls the RpcOpenPrinter method on the Print Server using the Print Server's name to obtain a handle.

2. The Print Client calls the RpcGetPrinterData method on the Print Server using the key "OSVersion" to obtain the Print Server's operating system version (and maps the OSVersion of the Print Server to the supported PSS version).

3. The Print Client calls the RpcClosePrinter method on the Print Server to close the Print Server handle.

In addition to following the process previously described to determine the capabilities of the Print Server, Print Clients MUST also follow the processes defined by the protocols of the Print Services System for negotiating capability support.

The tables below describe how the protocols used in the Print Services System are supported by the different versions of the system. The Windows product releases that correspond to each version of the Print Services System are listed in the section 8, Product Behavior.<6>

Microsoft-defined protocols, standard protocols, and print payloads are described in the following tables.

PSS 1.0

Protocols supported by Print Server	Print payloads supported by Print Server	Protocols used by Print Client	Print payloads used by Print Client	Protocol notes
RAP, SMB, LPR	Raw	SMB, LPR	Raw	LPR (support optional): Print Servers provide support for interoperability with Unix clients. Print Clients use this protocol only if specifically configured to do so when SMB is blocked by a firewall.

PSS 2.0

Protocols supported by Print Server	Print payloads supported by Print Server	Protocols used by Print Client	Print payloads used by Print Client	Protocol notes
PSS 1.0 (RAP, SMB, LPR)	PSS 1.0 (Raw)	PSS 1.0 (SMB, LPR)	PSS1.0 (Raw) and EMFSPOOL	LPR (support optional): Print Servers provide support for interoperability with Unix clients. Print Clients support only if specifically

Protocols supported by Print Server	Print payloads supported by Print Server	Protocols used by Print Client	Print payloads used by Print Client	Protocol notes
and RPRN	and EMFSPOOL	and RPRN		configured to do so when SMB and RPRN protocols are blocked by firewall. SMB: Print Clients use this protocol for driver file download operations or if RPRN is not available on a Print Server. EMFSPOOL (support optional): Print Clients can use this payload format only if the Print Server advertises support through RPRN::RpcEnumPrintProcessorDatatypes. RPRN: This protocol is supported in different versions of Windows as indicated in Appendix A: Product Behavior (section 8).

PSS 3.0

Protocols supported by Print Server	Print Payloads supported by Print Server	Protocols used by Print Client	Print payloads used by Print Client	Protocol notes
PSS 2.0 (RAP, SMB, LPR, RPRN) and WPRN, IPP, GPDPC, LDAP, GPOL	PSS 2.0 (Raw, EMFSPOOL)	PSS 2.0 (SMB, LPR, RPRN) and WPRN, IPP, GPDPC, LDAP, GPOL	PSS 2.0 (Raw, EMFSPOOL)	LPR (support optional): Print Servers provide support for interoperability with Unix clients. Print Clients support only if specifically configured to do so when SMB and RPRN protocols are blocked by firewall. WPRN (support optional): Print Servers support to provide printer drivers for Print Clients connecting through IPP. Print Clients use this protocol to download printer drivers from a Print Server when using the IPP protocol to connect. IPP (support optional): Print Servers provide support to allow printing from Print Clients on the extranet via HTTP. Print Clients use this protocol only if specifically requested to connect using an HTTP printer connection, which is done when a firewall blocks the other printing protocols. GPDPC (support optional): Support for this protocol by Print Servers and Print Clients was provided in this version by a downloadable utility. SMB: Print Clients use this protocol for driver file download operations or if RPRN is not available on a server. EMFSPOOL (support optional): Print Clients can use this payload format only if the Print Server advertises support through RPRN::RpcEnumPrintProcessorDatatypes.

PSS 4.0

Protocols supported by Print Server	Print payloads Supported by Print Server	Protocols used by Print Client	Print payloads used by Print Client	Protocol notes
PSS 3.0 (RAP, SMB, SMB2, LPR, RPRN, WPRN, IPP, GPDPC) and PAN, PAR, LDAP, GPOL	PSS 3.0 (Raw, EMFSPOOL), XPS	PSS 3.0 (SMB, LPR, RPRN, WPRN, IPP, GPDPC, LDAP, GPOL) and PAN, PAR	PSS 3.0 (Raw, EMFSPOOL), XPS	LPR (support optional): Print Servers provide support for this protocol for interoperability with Unix clients. Print Clients use this protocol only if specifically configured to do so when SMB and RPRN protocols are blocked by a firewall. WPRN (support optional): Print Servers support to provide printer drivers for clients connecting through IPP. Print Clients use this protocol to download printer drivers from a Print Server when using the IPP protocol to connect. IPP (support optional): Print Servers provide support to allow printing from Print Clients on the extranet via HTTP. Print Clients use this protocol only if specifically requested to connect using an HTTP printer connection, which is done when a firewall blocks the other printing protocols. XPS (support optional): Print Servers support this payload format only with third-party components installed, or with Terminal Server Easy Print installed. SMB: Print Clients use this protocol for driver file download operations, or if neither RPRN nor PAR are available on a Print Server. RPRN: Windows Vista and Windows 7 Print Clients use this protocol to check if PAR is available on the Print Server (by comparing the server's OS build number retrieved by using RPRN methods and to confirm a printer's presence by using a single RPRN RpcOpenPrinter/RpcClosePrinter sequence), and then uses PAR subsequently. Windows Vista and Windows 7 Print Clients also use RPRN if PAR is unavailable on the Print Server. EMFSPOOL (support optional): Print Clients can use this payload format only if the Print Server advertises support through RPRN::RpcEnumPrintProcessorDatatypes or PAR::RpcAsyncEnumPrintProcessorDatatypes.

4.6 System Vendor-Extensible Fields

Vendor specific extensions are supported as noted in section 1.8 of the member protocol TDs.

5 System Architecture

This section describes the basic structure of the system and the interrelationships among its parts, consumers, and dependencies.

5.1 Abstract Data Model

The Print Services System depends on an abstract data model (ADM) that maintains information about printers and related objects. These objects represent printers, and the objects are used in the protocol to communicate with printers, to print to them, and to manage their configurations.

A Print Server MUST behave as if it hosted the objects as specified in this section.

The abstract data model specified for the protocol described in [MS-RPRN] is identical to that for the protocol described in [MS-PAR]. A Print Server MUST maintain only one copy of the data underlying the implementation that exposes the protocol described in [MS-RPRN] or [MS-PAR].

For ease of presentation, a set of client-only structures called proxy objects, as indicated in the name "proxy" in the name of the object, is used in this document. Unless noted otherwise, proxy objects are not visible on the wire, but are used in this document to more easily describe offline operation of Print Clients. Proxy objects generally reflect all characteristics of the object they proxy for, including operations, persistence and failure characteristics. Proxy objects communicate with the objects they represent on the Print Server using the protocols described in [MS-RPRN] or [MS-PAR].

In this section, the descriptions of objects and elements in the Print Services System ADM are organized as follows:

- Operations and Rules
- Persistence
- Persistence Details Visible on the Wire
- Data Access Rules and Coherency
- Implication on Print Services System as a Whole
- Relationship with other ADM Elements
- General Constraints on ADM Object
- Access Permissions
- System Failure Behavior
- Usage/Interpretation Differences between Member Protocols
- Global Impact on System
- Sharing Between Member Protocols
- Data Visible to Callers/Callees
- Data Shared between Print Services System and External Entities

5.1.1 Print Server Object

The Print Server object represents the single instance of the print spooler running a computer acting as Print Server and Print Client. It is the root container for all other Print Services System objects.

The following operations are supported on a Print Server object:

- Set, Get, and Enumerate configuration data
- Add and Enumerate Print Queues (Delete is an operation on the Print Queue itself)
- Add, Delete, Get, and Enumerate Printer Drivers
- Add, Delete, Get, and Enumerate Print Server-specific Forms
- Add, Delete and Enumerate Ports
- Add, Delete, and Enumerate Print Processors, and Enumerate data types supported by a Print Processor
- Add, Delete, and Enumerate Print Monitors
- Add, Delete, and Enumerate machine global connections to Print Queues on other Print Servers

Persistence

Configuration data for the Print Server object is persisted immediately upon change.

Persistence Details Visible on the Wire

Configuration data is visible on the wire.

Data Access Rules and Coherency

Configuration data changes are immediately visible by all clients.

Implication on Print Services System as a Whole

In order to establish Print Services System communication, Print Clients need to determine the name of the computer hosting the print spooler/Print Server object. That name is also referred to as a Print Server Name. The Print Server Name is subsequently used to establish an RPC binding or for the protocol endpoint address in the non-RPC protocols.

Relationship with other ADM Elements

The Print Server is the root container for all other ADM elements and directly holds and manages:

- List of Print Server Names
- List of Print Queues and Print Queue Proxies
- Bidirectional Notification Channel
- List of Forms and Form Proxies
- List of Printer Drivers
- List of Language Monitors

- List of Port Monitors and Port Monitor Proxies
- List of Print Processors
- List of Known Printers
- List of Warned Printer Drivers
- A Security Descriptor controlling access to the Print Server

General Constraints on Print Server Object

Only one wire-visible Print Server object exists on a computer.

Access Permissions

Printer Driver files for each Print Queue MUST be stored on a Print Server in local file system directories that are shared with read access for Everyone. Print Clients will copy Printer Driver files from that location if they are compatible with the Print Client's operating system platform.

System Failure Behavior

The Print Server object has no specific system failure semantics. All configuration data of the Print Server object is present in the registry, and Windows restarts the print spooler service/Print Server upon restart after recovering from a failure.

A Print Server failure prematurely aborts all pending Print Clients' current operations. Print Clients report failure to client applications. Client applications can initiate a retry.

Usage/Interpretation Differences between Member Protocols

All member protocols acting on the Print Server object share the same interpretation.

Global Impact on System

The Print Server object is the per-system root container for all other Print Services System objects.

Sharing Between Member Protocols

The protocols described in [MS-PAR], [MS-RPRN], and [MS-PAN] share the Print Server object.

Data Visible to Callers/Callees

Data is visible to callers/callees as described in [MS-RPRN], methods for accessing configuration data and contained ADM objects.

Data Shared between Print Services System and External Entities

The Print Server stores the following data in the registry of the computer on which it is running:

- Persisted ADM hierarchy
- Server configuration data

The Print Server stores the following data in the file system of the computer on which it is running:

- Print Job Data
- Printer Driver modules/files

- Port Monitor modules/files
- Language Monitor modules/files
- Print Processor modules/files
- **Print Provider** modules/files

5.1.1.1 List of Print Server Names

In a workgroup environment, each Print Server stores a list of Print Server Names in the workgroup, synchronized using the protocol described in [MS-BRWS]. The protocols described in [MS-BRWS], [MS-ADLS], and [MS-ADSC] specify various ways a client can build a list of Print Server names.

The Print Server name that is passed to the Print Server by the Print Client can differ from the Print Server name that the Print Server determined upon its own initialization; for example, Print Server names may be aliased by **DNS** or the Active Directory System.

For workgroup environments, the list of Print Server Names is used to populate a list of choices the user can select from to see shared Print Queues on the selected Print Server. In workgroup environments, this ADM element is also used to identify Print Servers on the local network in order to notify them about printers shared on the local system. In domain environments, this ADM element is used when browsing the network for shared Print Queues to connect to.

This ADM element is also used to validate the Print Server name parameter on methods such as RpcAddPerMachineConnection as described in [MS-RPRN].

Print Clients MUST choose a name from the list of Print Server names to create a binding to the server.

Operations and Rules

The list of Print Server Names is dynamically populated before the Add Printer Connection dialog is displayed. The list can be populated at any time.

This ADM element is important for system operation, but is not visible on the wire.

The list of Print Server Names can be implemented as a transient data structure that is only created when needed to perform an operation (such as browsing for printers on the local network); however, any efficient implementation retains this ADM element for the lifetime of the print spooler process.

When implemented as a retained ADM element, the local print spooler SHOULD refresh this ADM element periodically to provide an up-to-date view of Print Servers in the local network.

Persistence

The list of Print Server names is not persisted. It is always dynamically detected at run time.

Persistence Details Visible on the Wire

None (ADM element not wire visible)

Data Access Rules and Coherency

None (ADM element not wire visible)

Implication on Print Services System as a Whole

This ADM element is used for user guided discovery of printers on the local network, as well as identification of other Print Servers in the local network. Since this ADM element is not wire visible, there are no protocol implications.

It is recommended to instantiate this ADM element upon print spooler startup so it can be used when required by user interaction without delay.

Relationship with other ADM Elements

None

General Constraints

This object MUST contain valid Print Server names in the local network and SHOULD be refreshed periodically. Different forms of Print Server names (for example, canonical, non-canonical, FQDN, IPv4 address, IPv6 address) MUST be treated equivalently if they resolve to the same Print Server address on the local network.

System Failure Behavior

Since this ADM element is not persisted, there is no specific system failure behavior. The ADM element will be reinitialized when the system is reinitialized after a failure.

Usage Differences between Member Protocols

None

Global Impact on System

In a workgroup scenario, this ADM element is used to determine other Print Servers on the local network in order to advise them of shared printers, resulting in occasional network traffic at arbitrary times.

Sharing Between Member Protocols

This ADM element is not wire visible.

Data Visible to Callers/Callees

This ADM element is not wire visible.

Data Shared between Print Services System and External Entities

This ADM element is initialized by using the protocol defined in [MS-BRWS] in a workgroup environment, but data is not shared with other entities.

To correctly resolve Print Server names, each Print Server MUST maintain a mapping between Print Server names and Print Server addresses. When composing a response requiring a Print Server name to the client, the Print Server, in order to identify itself, MUST use the same name that the client passed as the parameter in the method call.

5.1.1.2 Print Queue

Print Queues represent the central object of interest for the Print Services System. Print Queues are used on Print Servers. Each Print Queue represents a printer or a number of homogeneous printers installed on the Print Server. Print Queues manage the status of the printers. Print Queues receive print jobs, buffer print jobs, and send them to printers. A Print Queue associates a Printer Driver

and port (communication channel) with the printer represented, and maintains global and user-specific settings for that printer.

Each Print Queue object MUST maintain the following data elements:

- A name that uniquely identifies the Print Queue.
- A reference to a Printer Driver object for the printer.
- A reference to a print processor object.
- References to one or more port objects. <7>
- A list of queued or printing print jobs.
- A security descriptor controlling access to the Print Queue.
- Global _DEVMODE settings.
- Per-user _DEVMODE settings.
- The name of the default data type for the Print Queue.

A Print Server typically initializes the list of Print Queues to an empty list. The Print Server MUST persist the list of Print Queues between restarts.

Operations and Rules

The following operations are defined on Print Queues:

- Delete Print Queue.
- Set and query Printer Driver associated with Print Queue.
- Retrieve printer status from Print Queue.
- Receive Print Job data, buffer Print Job data, queue print jobs, send Print Job data to printer.
- Notify client of status changes.
- Manage print jobs queued on a Print Queue.
- Set, retrieve and enumerate data associated with Print Queue.
- Retrieve detailed information about printer managed by Print Queue.
- Add, Delete, Get, Enumerate Print-Queue-specific Forms.

Persistence

Print Queues and associated data are persisted to the registry upon change, for immediate availability upon system restart. See [MS-RPRN] section 3.1.1 for the details of Print Queue persistence.

Persistence Details Visible on the Wire

- [MS-RPRN] and [MS-PAR] describe methods to transparently set, get and enumerate data elements persisted in the registry.

- Printer drivers use these methods to associate driver specific data with a Print Queue.
- The print spooler persists those data elements to the registry for later use by the Printer Driver.

Data Access Rules and Coherency

- The ADM element coherency MUST be satisfied according to the requirements in this section, regardless of the actual protocol used to access the ADM element.
- Multiple Print Clients or multiple threads on the same Print Client can hold open Print Queue handles for the same Print Queue at the same time.
- Setting or deleting data associated with a Print Queue are atomic operations, and the changed data is visible immediately to Print Clients.
- Adding Print Queues behaves atomically, and the added Print Queue is visible immediately to Print Clients.

  Because a Print Queue is immediately visible upon addition, and there is no means to bundle "add Print Queue and set Print Queue data" into an atomic transaction, special care must be taken by the following stakeholders:

  - Printer Driver writer: The Printer Driver SHOULD NOT rely on any non-default settings associated with the Print Queue for correct and reliable operation.
  - ISV monitoring Print Queues using notification methods: Print Queues can change their settings immediately after they appear, before the monitoring client has had a chance to register a notification listener for the Print Queue. The monitoring software needs to be designed to follow these steps when monitoring the appearance of new queues (order of steps important):

    1. Listen for new queue to arrive.
    2. Register a change notification for the new queue.
    3. Retrieve settings for new queue.

       The Print Client MUST treat data received through notifications as most current, even though change notifications can arrive at any time after step 2, even before step 3 has been completed.

Deleting a Print Queue immediately removes the Print Queue from the list of Print Queues visible through enumeration, but does not affect open Print Client handles to Print Queues. The Print Queue is set into a state indicating that deletion is pending and the actual deletion is deferred until the last Print Client handle to the Print Queue is closed and the last queued Print Job has been processed by the Print Server.

Methods for job data submission guarantee coherency by allowing only a single open job submission context (created using RpcStartDocPrinter, closed using RpcEndDocPrinter) per open Print Queue handle. Print Server implementation-specific policy or settings determine the order in which print jobs submitted simultaneously on two different open Print Queue handles for the same Print Queue are being sent to the printer. Windows servers allow two different settings:

- Print when job complete: The server will wait until the client closes the Print Job submission context before processing the Print Job and sending any data to the printer

- Start printing immediately: The server will start processing print jobs as soon as data arrives, and send data to the printer. If two or more clients have open Print Job submission contexts, the print jobs are processed in the order they were started.

All Print Services System methods for enumerating and getting (of Print Queues, associated data, and so on) follow a two-call pattern in which the first call returns the number of bytes required for the result of the enumeration or get. Next the Print Client allocates the required buffer, and then calls the Print Server again with the requested buffer, into which the Print Server writes the result of the enumeration or get. Because methods to modify and delete data and Print Queues behave atomically, but enumeration and get do not (because there are at least two calls required), the Print Client needs to be prepared for the required buffer increasing between the two calls of this two-step pattern. The Print Client MUST therefore always check the result of the second call, and if the Print Server indicates that the size of the buffer passed in was not sufficient, the Print Client needs to allocate more memory and try again. Implementation-specific policy on the Print Client determines how often the Print Client will retry before failing the enumeration request.

Implication on Print Services System as a Whole

In Windows, the protocols described in [MS-PAR], [MS-RPRN], [MS-PAN], and the LPR protocol operate on aspects of the ADM element.

Protocol handlers for [MS-SMB] and [MS-RAP] do not have a representation of or direct access to the ADM. Instead they are implemented by translating and redirecting calls from [MS-SMB] and [MS-RAP] to local calls to the print spooler service (acting as Print Server).

Protocol handlers for IPP and the protocol described in [MS-WPRN] are Internet Information Server extensions and do not have a representation of or direct access to the ADM. Instead, Internet Information Server implements the HTTP server process listening for connections using these protocols, but calls into the IPP and [MS-WPRN] extensions to do the actual work; those extensions in turn call local print spooler APIs to perform the actual work.

It is possible to implement Print Server support for LPR either directly in the print spooler or by providing a redirector module that calls a local print spooler interface. In Windows, the LPR support is implemented as a redirector module that does not have a representation of or direct access to the ADM, and instead calls local print spooler APIs to perform the actual work.

The methods of the local print spooler interface used by redirector modules is implementation specific, but in Windows are a nearly identical subset of the [MS-RPRN] interface called via LPC.

Relationship with other ADM Elements

Each Print Queue ADM element holds a reference to the following:

- Reference to one or more Port ADM elements.
- Reference to one Print Processor ADM element. The Print Processor ADM element MUST be present on the same computer as the Print Queue ADM element.
- Reference to one Printer Driver object, whereas the Printer Driver MUST be compatible with the operating system platform of the Print Server. The Printer Driver ADM element MUST be present on the same computer as the Print Queue ADM element.
- Reference to zero or more Printer Driver objects, whereas the Printer Driver is compatible with an operating system platform different from the Print Server. The Printer Driver ADM element MUST be present on the same computer as the Print Queue ADM element.

- Reference to zero or more Bidirectional Notification Channels associated with this Print Queue ADM element.
- Reference to zero or more Print Job ADM elements. The Print Job ADM elements are on the same computer as the Print Server role. For the Print Client role, the implementation can chose to create a local Print Job ADM element which acts as a proxy to the Print Server's Print Job ADM element, or it can use calls defined in [MS-RPRN]/[MS-PAR] to the Print Server to satisfy Print Job-specific requests on demand, without keeping a local copy of the Print Job ADM element's state.

Additionally, each Print Queue ADM element owns the following:

- A global DEVMODE structure.
- Zero or more per-user DEVMODE structures.
- One SECURITY_DESCRIPTOR controlling access to the Print Queue object.
- The name of the default data type for the Print Queue.

General Constraints

The name of Print Queue MUST be unique across all Print Queues hosted by Server.

System Failure Behavior

Print Queue changes are persisted to the registry upon change. After system failure, Print Queues are reinitialized from the persisted store. Pending and currently processing print jobs which have been submitted completely by the client associated with a Print Queue will be added to the Print Queue upon system restore. Printing of these jobs is restarted after system restore.

Global Impact on Print Services System

The Print Queue object is operated on by all member protocols.

Sharing Between Member Protocols

For historical reasons, member protocol TDs refer to a Print Queue object as "Printer." However, within this system document, printers (physical devices) are delineated from Print Queues (logical objects). The Print Queue object corresponds to a single printer in the "List of Printers" object identified in the member protocol TDs.

The Print Queue object is operated on by all member protocols.

5.1.1.2.1 Print Queue Proxy

Methods for remotely adding Print Queue connections (also known as Print Queue Proxies) to a Print Client are described in [MS-RPRN] section 3.1.4.2.24 through 3.1.4.2.26 and [MS-PAR] sections 3.1.4.1.22 through 3.1.4.1.24. The Print Queue Proxy object represents the Print Client side of a connection to a Print Server Print Queue. Print Queue Proxies are used only on Print Clients and they affect only the behavior of the print client; not the protocol.

The Print Queue Proxy object forwards all operations defined for the Print Queue object to the Print Server (using the protocols defined in [MS-RPRN] and [MS-PAR]) if a connection can be established. The Print Queue Proxy MAY<8> provide caching mechanisms that allow client applications to print even when no network connection to the Print Server can be established. In this case the operations

required for printing are serviced by the Print Queue Proxy on the client and forwarded to the Print Server when the connection is reestablished.

5.1.1.2.2 Print Job

A Print Job represents data sent by a Print Client. The Print Job data can be in a format directly usable by a printer. This format is called Page Description Language (PDL); HP PCL and Adobe PostScript are examples of common PDLs. Applications utilize the Printer Driver associated with the Print Queue to convert graphical primitives into PDL formatted data. PDL formatted data is generically referred to as RAW data in the Print Services System.

Alternatively, the Print Job data can be in a metafile format, representing application graphical primitives. The metafile format acts as an application-independent and printer technology-independent intermediary. The format defined in [MS-EMFSPOOL] is an example of a metafile format. The XML Paper Specification is an example of a data format which can be used both as PDL and as metafile format.

Print Jobs using RAW data are typically not further processed by the Print Server; instead they are directly forwarded to the printer managed by the Print Queue. Extensibility mechanisms in the print spooler, such as Language Monitors, Port Monitors or Print Processors are sometimes utilized by IHVs or ISVs to provide limited additional processing of RAW Print Jobs; such processing can include simple page counting, addition of some high-level print control commands (for example, finishing, stapling, enveloping, or routing commands) or even more complex task such as ink density adjustment through image processing algorithms.

On the other hand, Print Jobs using one of the metafile formats (as defined in [MS-EMFSPOOL]) or the XML Paper Specification require further processing unless the printer has native metafile processing capabilities.

Further processing is done by a Print Processor associated with the Print Queue. The Print Processor reads the metafile data from the Print Job and utilizes the Printer Driver associated with the Print Queue to translate the commands in the metafile to RAW data, which is then sent to the printer.

The Print Processor associated with a Print Queue can be queried by the Print Client for the Print Job data formats supported. If only the RAW format is supported, the Print Client MUST translate the Print Client application's graphics primitives to RAW data using the Printer Driver for the Print Queue. All Print Queues MUST support at least the RAW format.

If one or more metafile formats are additionally supported, the Print Client makes an implementation defined choice between sending RAW data or metafile formatted data. The behavior of Windows Print Clients can be configured to use either RAW or metafile formatted data.

An important factor for this choice is the considerable processing required to do the translation from graphics primitives (either produced directly by the application, or read from a metafile) to RAW data. The Print Client can be freed from this processing burden at the expense of the Print Server and vice versa.

Operations and Rules

The following operations are defined on Print Jobs:

- Start and end Print Job.
- Start and end pages in a Print Job allowing the Print Server to make statements about printing progress that are meaningful to the user.
- Write data to a Print Job.

- Abort a Print Job.
- Read data from a Print Job (this functionality is used by Print Processors locally to read the metafile data from a Print Job; although it is possible to use this functionality remotely through the **RpcReadPrinter** method defined in [MS-RPRN], Windows Print Clients do not do so.)
- Flush Print Job data to a printer.
- Change status of a Print Job (Pause, Resume, Cancel, Restart).
- Change Print Job retention. (The Print Server will keep a retained Print Job after sending it to the printer; it will delete a non-retained Print Job after sending it to the printer.)
- Change other attributes of Print Job (Priority, Start and other times, Description strings, order in Print Queue).

Persistence

- Print jobs are persisted to the local file system of the system on which the Print Server is running.
- Print job files are deleted from the file system when the job has been sent to the printer, unless Print Job retention has been specified by the Print Client.

Persistence Details Visible on Wire

None.

Data Access Rules and Coherency

The structure of methods defined in [MS-RPRN] and [MS-PAR] guarantees serialized access to a Print Job, resulting in coherency. Only a single Print Job submission context can be active for an open Print Queue handle at any time. It is the responsibility of the Print Client to serialize Start/End/Write operations on the Print Job, which is typically trivial because only one thread writes to a Print Job at a time.

Relationship with other ADM Elements

- Owned by a Print Queue object.
- Holds a reference to a Printer Driver, a Print Processor, and a Port.
- Owns a SECURITY_DESCRIPTOR controlling access to the Print Job. The security descriptor is inherited from the Print Queue containing the Print Job.

General Constraints

The JobID of a Print Job MUST be unique across all Print Jobs in a Print Server.

System Failure Behavior

If the system on which the Print Server runs fails, the behavior for Print Jobs is as follows:

- All Print Jobs that have not been ended by the Print Client (using RpcEndDocPrinter/RpcAsyncEndDocPrinter) at the time of failure will be irrevocably lost, and the Print Client can resend them.

- Print Jobs that have been ended by the client and are queued or currently being processed at the time of failure will be added to the Print Queue for subsequent processing upon system restart. The Print Server will find these jobs by scanning the file system locations it stores Print Job files at for remaining Print Jobs. A control file accompanies each Print Job file in these locations, and the control file contains information about the completion status of the Print Job at the time of failure. The control file is locally generated and managed by the print spooler and not visible on the wire.

Usage Differences Between Member Protocols

The protocol described in [MS-RAP] operates only on a subset of Print Job attributes.

Sharing Between Member Protocols

- The protocols defined in [MS-RPRN] and [MS-PAR] share the Print Job object.
- Print Clients using the SMB Access Protocols perform write operations to a shared Print Queue identical to a write operation to a write-only file share.

5.1.1.2.3 Print Job Proxy

The Print Job Proxy object represents the client side of an active Print Job on the Print Server.

The Print Job Proxy object SHOULD be implemented to allow Print Job submission by a client application in case the network connection to the Print Server is unavailable (offline printing).

As such, the Print Job Proxy object SHOULD implement all functionality called by the local print spooler, keeping track of resulting state changes, and use the protocols defined in [MS-RPRN] and [MS-PAR] to forward the calls to the Print Server's Print Job object when a connection can be established. The form of the interaction with the local print spooler is implementation-specific.

5.1.1.3 Unidirectional Notification Queue

Characteristics of the Unidirectional Notification Queue are as follows:

- Unidirectional Notification Queues are only visible on the wire using the PAN protocol [MS-PAN].
- Unidirectional Notification Queues are specified in detail in [MS-PAN] section 3.1.1.1.
- Unidirectional Notification Queue objects are not persisted.
- Upon system failure, Unidirectional Notification Queues are no longer valid and are not restored upon system restart.
- The name specified when registering a Print Client for unidirectional notifications with a Print Server MUST either be NULL, in which case the Print Client is registered with the Print Server object itself; or it MUST be the name of a Print Queue object managed by the Print Server, in which case the Print Client is registered with the Print Queue object.
- When registering a Print Client with a Print Server, the Print Server's **security descriptor** is used for access control to determine if the registration is allowed for the user of the Print Client that is making the registration call.
- When registering a Print Client with a Print Queue, the Print Queue's security descriptor is used for access control to determine if the registration is allowed for the user of the Print Client that is making the registration call.

5.1.1.4 Bidirectional Notification Channel Objects

Characteristics of the Bidirectional Notification Channel objects are as follows:

- Channel objects are only wire visible using the protocol described in [MS-PAN].
- Channel objects are specified in detail in [MS-PAN].
- Channel objects are not persisted.
- Upon system failure, channel objects are no longer valid and are not restored upon system restart.
- The name specified when registering a channel object with a Print Server MUST either be NULL, in which case the channel is registered with the Print Server object itself, or MUST be the name of a Print Queue object managed by the Print Server, in which case the channel is registered with the Print Queue object.
- When registering a channel object with a Print Server, the Print Server's SECURITY_DESCRIPTOR is used for access control to determine if the registration is allowed for the Print Client's user making the registration call.
- When registering a channel object with a Print Queue, the Print Queue's SECURITY_DESCRIPTOR is used for access control to determine if the registration is allowed for the Print Client's user making the registration call.

5.1.1.5 Form Object

Form objects represent information about physical sheets of paper available to Print Clients.

Operations and Rules

Form objects actions are set, get, enumerated and deleted.

Persistence

Form objects are persisted upon change.

Persistence Details Visible on the Wire

The member protocols described in [MS-RPRN] and [MS-PAR] specify which Form object fields are visible on the wire. Since a Form is a static object, all persisted details are visible on the wire.

Data Access Rules and Coherency

Forms can be deleted at any time, and from then on will not be visible to Print Clients.

Since the _DEVMODE structures for print jobs redundantly store the physical form dimensions, deletion of a form does not affect active print jobs using the form.

System Failure Behavior

Changes to forms are persisted to the registry upon change. After system failure, forms are reinitialized from the persisted store.

Form Proxy

A form proxy represents a form on a remote Print Server, which locally caches all the information for the form.

5.1.1.6 List of Printer Drivers

Member protocol TDs refer to this object as "List of Drivers." However, this system document uses the more precise name "List of Printer Drivers" for this object. Each Printer Driver represents the software component responsible for converting print content submitted by applications into printer-specific commands. Each Printer Driver object MUST maintain the following data elements:

- A name that uniquely identifies the Printer Driver.
- A list of well-known modules (rendering module, configuration module, and data module).

Additionally, each Printer Driver object SHOULD maintain the following optional data elements:

- A list of dependent files.
- Information about the Printer Driver manufacturer, Printer Driver timestamp, and version.

The modules making up a Printer Driver are platform and operating system specific executable modules and data files. The interaction between the local print spooler and the Printer Driver is implementation specific. Printer vendors (IHV's) provide Printer Drivers compatible with their printers for a number of operating system platforms. As a convenience to the user and to IHVs, Microsoft ships Windows with a set of Printer Drivers for common printers.

Printer Drivers are packaged by IHVs into **driver packages**. The format and layout of a driver package is operating platform specific. A driver package can optionally contain and install other components, such as print processors, language monitors and port monitors.

Printer Drivers MUST be present on each system participating in the Print Services System. There are no Printer Driver Proxy objects. Instead, a Print Client MUST implement a mechanism to install an appropriate Printer Driver for use by a Print Queue Proxy. Windows Print Clients make an effort to ensure the closest possible match between the Printer Driver used by a Print Queue Proxy and its Print Queue on the Print Server, and can install the driver from different sources to achieve this (Print Server, Windows Update, or the user).

If the Printer Driver for the Print Queue Proxy and the Printer Driver for the Print Queue on the Print Server match (or otherwise are designed to be compatible), they can register or use Bidirectional Notification Channel objects to communicate with each other using the protocol described in [MS-PAN]. Likewise, IHVs can provide application software compatible with their printer drivers that uses Bidirectional Notification Channel objects to communicate with their printer drivers.

A Print Server typically initializes the list of drivers to an empty list. The server MUST persist the list of drivers between restarts. Drivers are added, removed, enumerated and managed using Printer Driver management methods.

Operations and Rules

The add, delete, get, and enumerate operations are defined on Printer Driver objects.

Persistence

Printer Driver objects are persisted to the registry upon addition or change.

Persistence Details Visible on the Wire

None

Data Access Rules and Coherency

All operations on Printer Driver objects are atomic. There are no operations to modify Printer Driver objects, so coherency is not a concern.

A Printer Driver object can only be deleted if no Print Queue object holds a reference to it.

Implication on Print Services System as a Whole

When a Print Client connects to a Print Queue shared by a Print Server, it needs to make sure to obtain and associate a Printer Driver compatible with the printer managed by the Print Queue.

Print Clients use the following steps to accomplish this task:

1. First, using methods described in [MS-RPRN] or [MS-PAR], obtain detailed information about all operating system platform versions for which the server Print Queue has a Printer Driver available, as well as the Print Queue's printer and detailed Printer Driver information.

2. Subsequently:

   - If the Print Queue has a Printer Driver that matches the Print Client's operating system platform, obtain the location of the Printer Driver files on the Print Server's local file system using an appropriate [MS-RPRN] or [MS-PAR] method, and copy the files from there (the Print Server MUST make these locations accessible for Read for Everyone).

   - Use printer's model name, manufacturer name or printer hardware ID to locate a compatible Printer Driver using Windows Update, or from the IHV.

   - Prompt the user for a location containing a compatible Printer Driver (such as a CD-ROM).

Administrative Clients use methods of the protocol described in [MS-PAR] to upload driver packages to the Print Server for later use by Print Queues. The Print Server puts these driver packages into its local **driver store** without actually installing them.

Relationship with other ADM Elements

Printer Driver objects have a reference to a Print Processor. This can be a weak reference; a Printer Driver SHOULD be designed to work with any Print Processor. Printer Driver objects also have a reference to a Language Monitor.

General Constraints

In Windows, printer drivers MUST consist of operating system specific data files and of executable modules implementing specific local APIs. Printer driver modules are loaded into application processes as well as into the print spooler service. Since Printer Driver modules are loaded in various security contexts, including local system, special care has to be applied when developing a Printer Driver to not compromise system security. Of special significance is the fact that a weakness in the Printer Driver might serve as an entry point for a security attack, despite the best intentions of the Printer Driver developer.

It is strongly recommended that Print Servers and Print Clients independently assure that only trusted Printer Drivers are installed and used. Confirmation by an Administrator SHOULD be a requirement before installing a Printer Driver whose trustworthiness cannot be established automatically.

Names of Printer Drivers MUST be unique for a Print Server.

System Failure Behavior

Printer Driver data is persisted to the registry upon addition or change.

Upon installation of a Printer Driver, Printer Driver files are copied by the print spooler from the driver package to a location on the local file system.

5.1.1.7 Language Monitor

A **language monitor** is a module tightly coupled to a printer driver. It filters RAW data as it is being sent from the printer driver to the port monitor, which sends the data to the port. A language monitor also controls aspects of the communication with a printer. Language monitor characteristics include the following:

- A language monitor is only used by the print server role.
- Additionally, a language monitor can implement a bidirectional communication method which allows status and configuration feedback from the printer to the printer driver and print queue. This method is called by the print spooler when the print client issues an **RpcGetPrinterData** or **RpcAsyncGetPrinterData** call on a printer handle that has been opened with administrative access rights. No other aspects or data or result of operation of a language monitor is visible on the wire.
- Language monitors are provided by IHVs and by the operating system.<9>
- Language monitors are added, deleted and enumerated using port monitor management methods of the protocols described in [MS-RPRN] or [MS-PAR].
- Language monitors do not retain state beyond contexts for print jobs currently being processed by the print queue. Consequently, there are no specific failure semantics.

Persistence

Language monitor objects are persisted on creation or change.

Persistence Details Visible on the Wire

None

Data Access Rules and Coherency

All operations on language monitor objects are atomic. There are no operations to modify language monitor objects, so coherency is not a concern.

A language monitor object can only be deleted if no printer driver object holds a reference to it.

General Constraints

Names of language monitors MUST be unique for a print server.

5.1.1.8 Port Monitor

A port monitor is a component that can send buffers of data to printers using supported protocols. Port monitors can manage extended communication with the printer, such as collection status information from the printer. Port monitors expose zero or more ports. Port Monitor characteristics include the following:

- Port monitor modules are implementation-specific for a given port type. A port exposed by a port monitor represents a communication channel to a printer connected to the computer via an implementation-specific protocol understood by the port monitor.
- Port Monitors manage addition and removal of ports.
- Port Monitors provide a port abstraction by implementing transport specific communication protocols. Port Monitors can use other protocol stacks, such as TCP/IP or the local USB stack to communicate with the printer. An example is TCPMON, which provides communication with a printer connected to the local network using TCP/IP port 9100 to send Print Job data to the printer.
- A Print Server typically initializes the list of port monitors to an empty list. The server MUST persist the list of port monitors between restarts. Port monitors are added, removed or enumerated using Port management methods.
- Windows ships with a Port Monitor supporting the TCP/IP Port 9100 and SNMP connections to Printers, with a Port Monitor supporting Web Services for Devices (WSD) printers, and with a port monitor supporting local printer connections (such as USB, parallel, COM, Bluetooth, Firewire).
- Other Port Monitors are provided by IHVs.
- Port Monitors do not retain state beyond contexts for Print Jobs currently being processed by the Print Queue. Consequently, there are no specific failure semantics.
- Port Monitors handle RpcXcvData/RpcAsyncXcvData methods defined in [MS-RPRN]/[MS-PAR]. These methods provide a transparent conduit between an Administrative Client (or Print Client) to control aspects of the Port Monitor, and to query bidirectional data obtained from the Printer.

Operations and Rules

- A handle to a Port Monitor can be opened using RpcOpenPrinter (see details in [MS-RPRN]).
- RpcXcvData can be used to control functions defined by a specific Port Monitor.

Persistence

The Port Monitor object is persisted upon creation and change.

Persistence Details Visible on the Wire

None

Data Access Rules and Coherency

All operations on Port Monitor objects are atomic; there are no operations to modify Port Monitor objects, so coherency is not a concern. If a Port Monitor implements methods to handle RpcXcvData / RpcAsyncXcvData, it MUST ensure coherency between callers, and it MUST also ensure that it does not delete a managed Port while that Port is still in use.

A Port Monitor object can only be deleted if no Port object holds a reference to it.

General Constraints

Names of Port Monitors MUST be unique for a Print Server.

5.1.1.8.1 Port Monitor Proxy

Port Monitor Proxy characteristics include the following:

- A Port Monitor Proxy represents a Port Monitor on a remote Print Server.
- A Port Monitor Proxy SHOULD cache enough information locally to reflect an offline state for a remote Port in case the network connection to the Print Server is unavailable.
- The Print Job Proxy SHOULD be designed, so it caches Print Job information locally if the Port Monitor Proxy indicates an offline state for a remote Port.

5.1.1.8.2 Port

A Port represents a connection to an actual printer. Ports are exposed and managed by Port Monitors.

Operations and Rules

A handle to a Port can be opened using RpcOpenPrinter. That port object handle allows sending print data directly to a Port without intermediate buffering or spooling. The following operations are defined on a port object handle:

- Start/End document
- Write/Read data
- Send and receive bidirectional status data
- RpcXcvData

Persistence

The Port object is persisted upon creation and change.

Persistence Details Visible on Wire

None

Data Access Rules and Coherency

The Port Monitor managing the Port MUST ensure serialized access to each Port, so that only one communication context can be open for a single Port at any given time. A Port cannot be deleted while being in use by a Print Queue or Print Job.

General Constraints

Names of Ports MUST be unique for a Print Server.

5.1.1.9 Print Processor

Print processors, provided by printer manufacturers or generic suppliers, perform additional manipulation of print content before it is sent to the printer.

Windows includes a Print Processor capable of processing Print Jobs in the following formats:

- XML Paper Specification

- Enhanced Metafile Spool Format [MS-EMFSPOOL]
- RAW

Any Print Services System implementation MUST provide a default Print Processor capable of handling at least RAW format. Other than their name, and supported Print Job data formats, there are no wire visible details of Print Processors.

Operations and Rules

Print Job data formats supported by a Print Processor can be queried using RpcEnumPrintProcessorDatatypes.

Persistence

The Print Processor object is persisted upon creation.

Persistence Details Visible on the Wire

None

Data Access Rules and Coherency

All operations on Print Processor objects (Add, Delete, Enumerate) are atomic; there are no operations to modify Print Processor objects, so coherency is not a concern.

A Print Processor object can only be deleted if no Print Queue or Print Job object holds a reference to it.

General Constraints

Print Processor names MUST be unique per Print Server.

System Failure Behavior

Print Processors do not maintain state beyond Print Jobs currently being processed.

5.1.1.10 List of Known Printers

Within this system document, this element can be more accurately described as a "list of known Print Queues." However, for consistency with the ADM described in [MS-RPRN], the element is named here as "List of Known Printers," as it is in [MS-RPRN].

The Print Server SHOULD maintain a List of Known Printers that includes printers not installed on the Print Server but installed on other Print Servers reachable on the network. The printers in this list MUST only be used when composing a response for RpcEnumPrinters with the appropriate flags set, as specified in [MS-RPRN]. This list facilitates printer detection in networks without the Active Directory System.

The List of Known Printers is populated when other Print Servers on the local network call RpcAddPrinter with Level=1, thus announcing shared Print Queue objects to this Print Server.

5.1.2 Instantiation of the ADM Hierarchy

The print spooler SHOULD instantiate the ADM hierarchy in the following order:

1. Print Server object

1. Asynchronously: List of Known Printers
2. Asynchronously: List of Known Print Servers

2. Form objects
3. Port Monitor objects
   - Port objects
4. Language monitor objects
5. Print Processor objects
6. Printer Driver objects
7. Print Job objects

The order of creation is enforced by the appropriate error codes returned by methods defined in [MS-RPRN]/[MS-PAR] if a required object is unknown when an object is created. For example, an attempt to create a Print Queue object referring to a non-existent Print Processor will fail.

5.2 White Box Relationships

The diagrams in this section illustrate the relationship of the components within Print (or Administrative) Clients, and the relationship of the components within Print Servers in the Print Services System. Additional details are available in section 5.4, System Internal Architecture.

5.2.1 Print and Administrative Clients

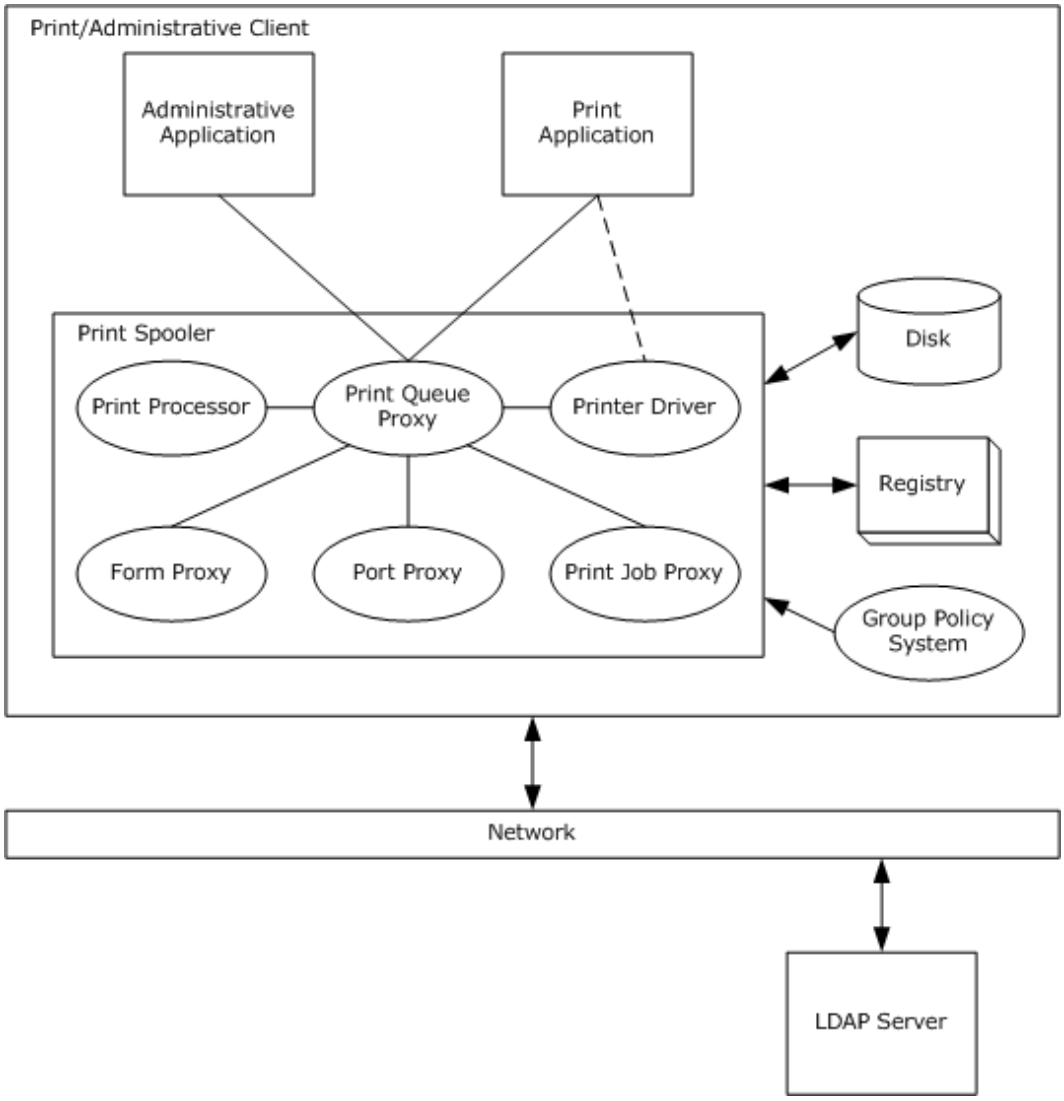

Figure 8: White box diagram of Print and Administrative Clients

5.2.2 Print Server

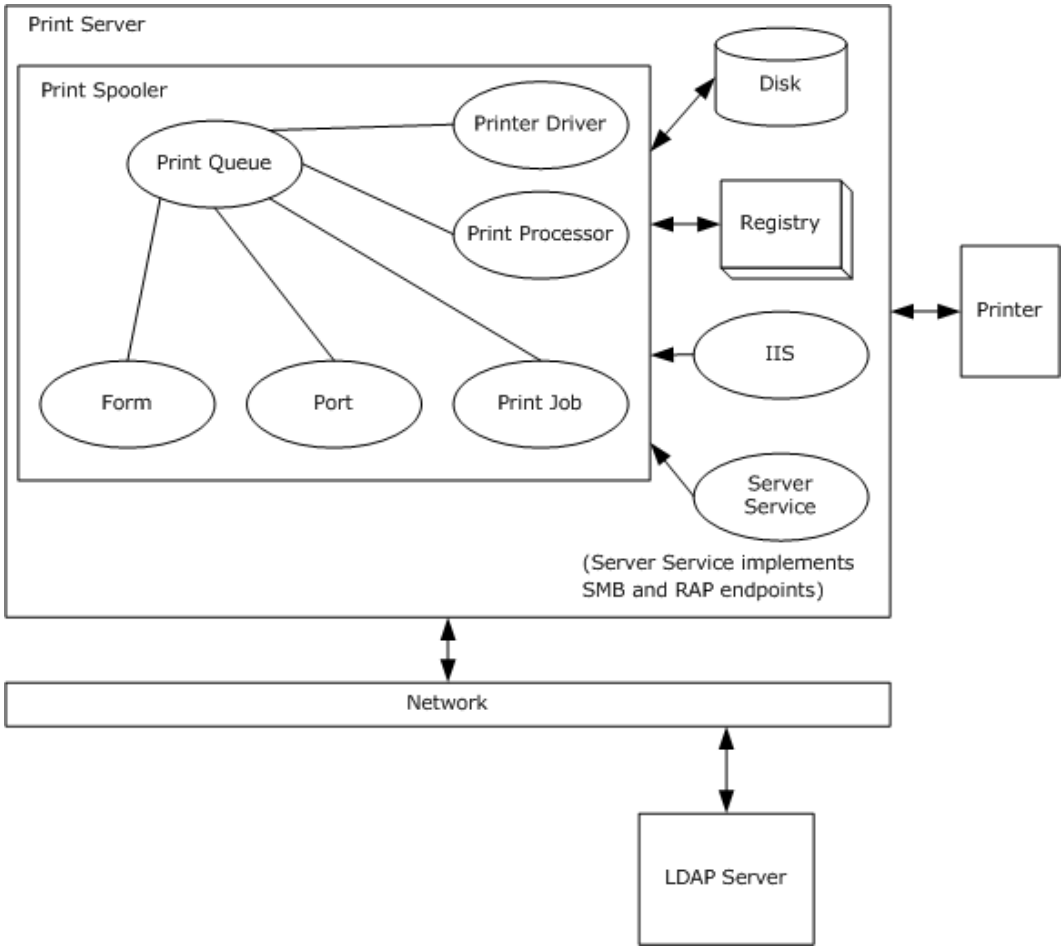

Figure 9: White box diagram of Print Server

5.3 Member Protocol Functional Relationships

The following diagrams illustrate the protocol relationships in the Print Services System.

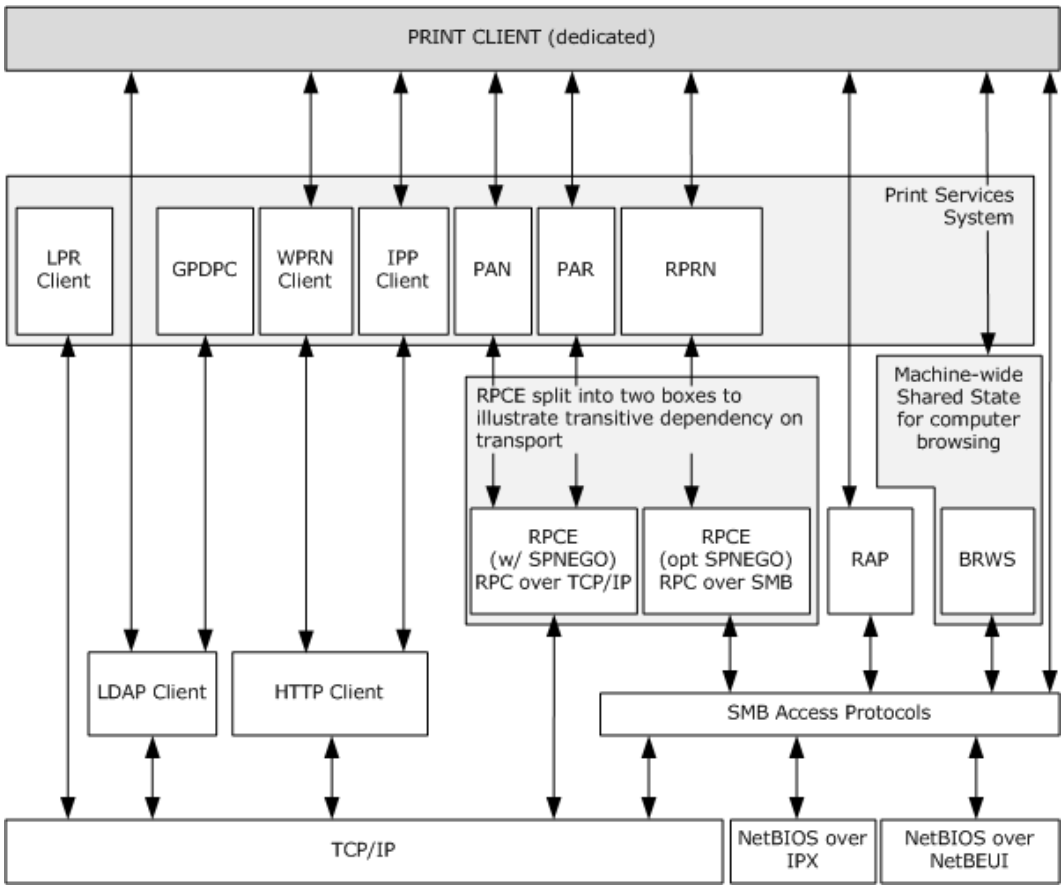

Figure 10: Full protocol layering for a Print Client depicting all protocols, including those that are optional

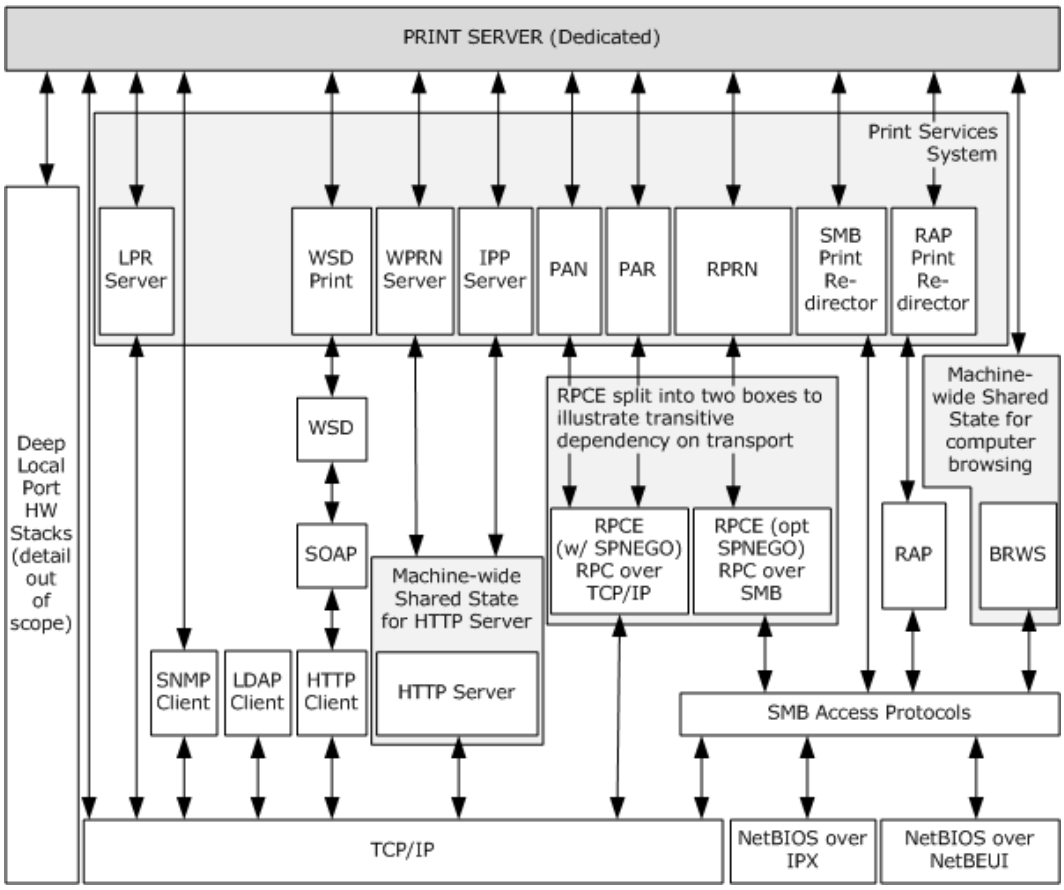

Figure 11: Full protocol layering for a Print Server depicting all protocols, including those that are optional

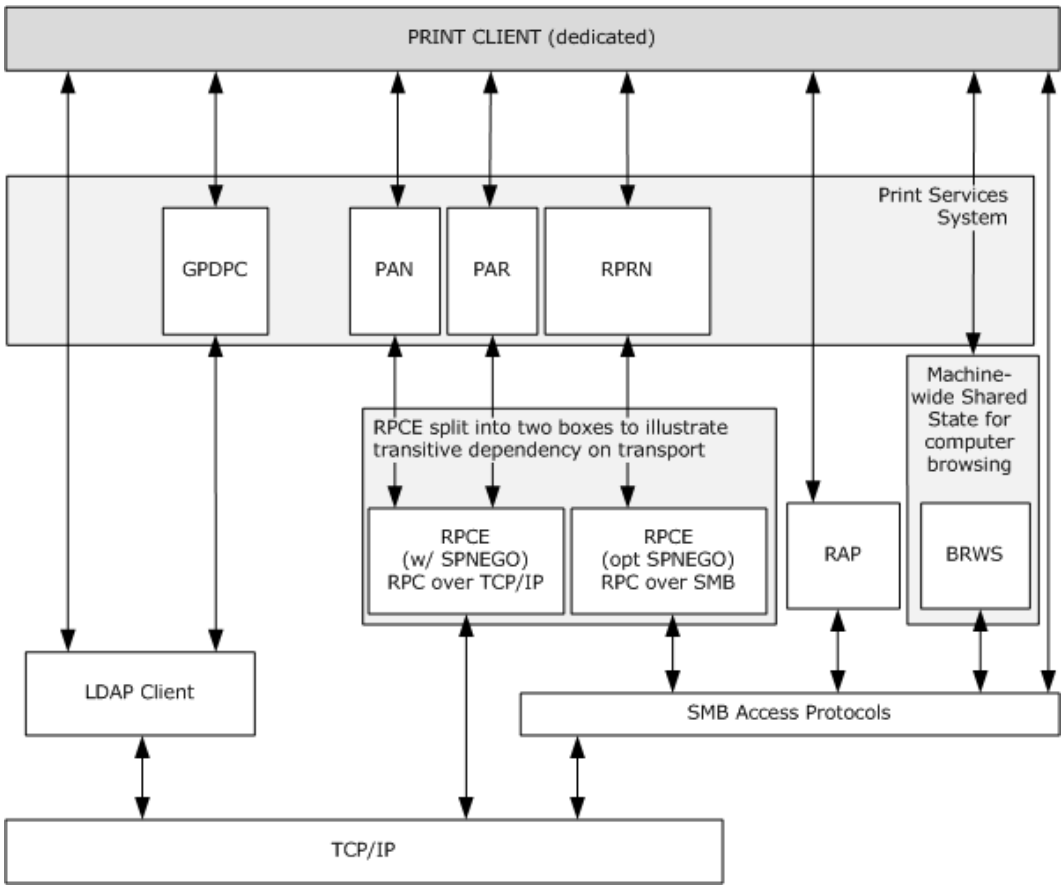

Figure 12: Typical Windows protocol configuration for a Print Client, without optionally supported protocols

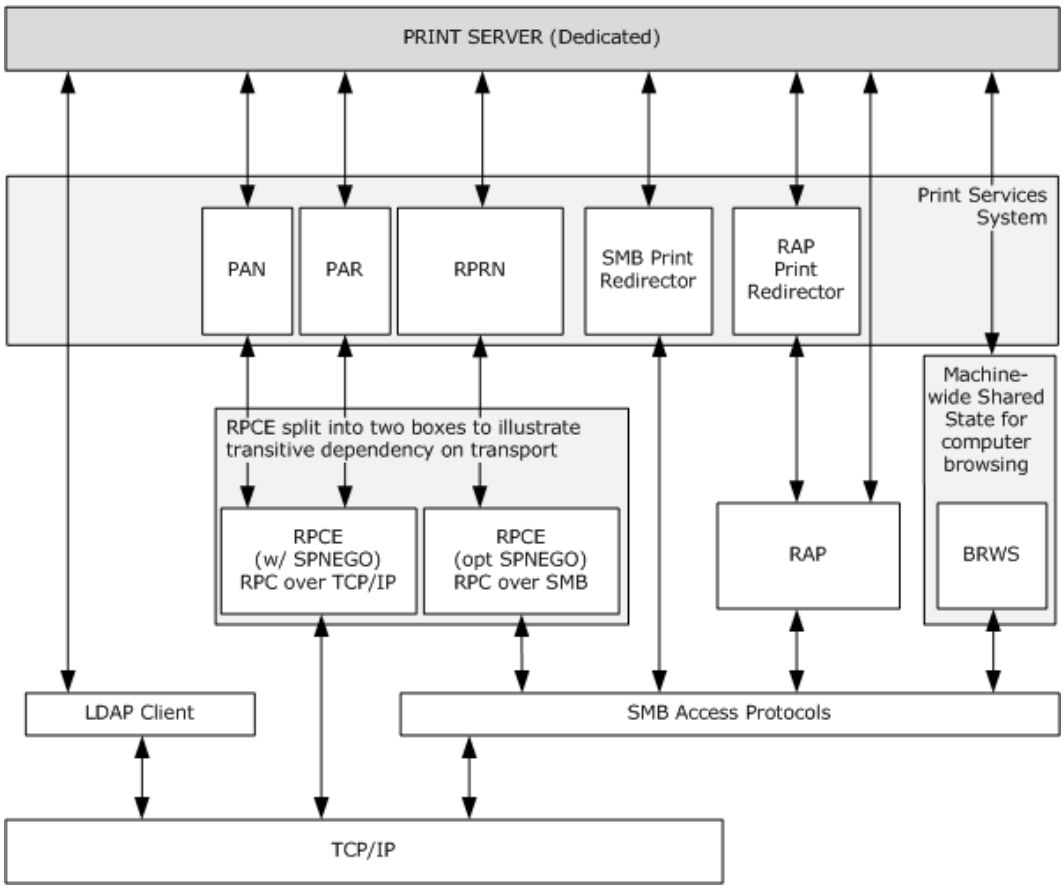

Figure 13: Typical Windows protocol configuration for a Print Server, without optionally supported protocols

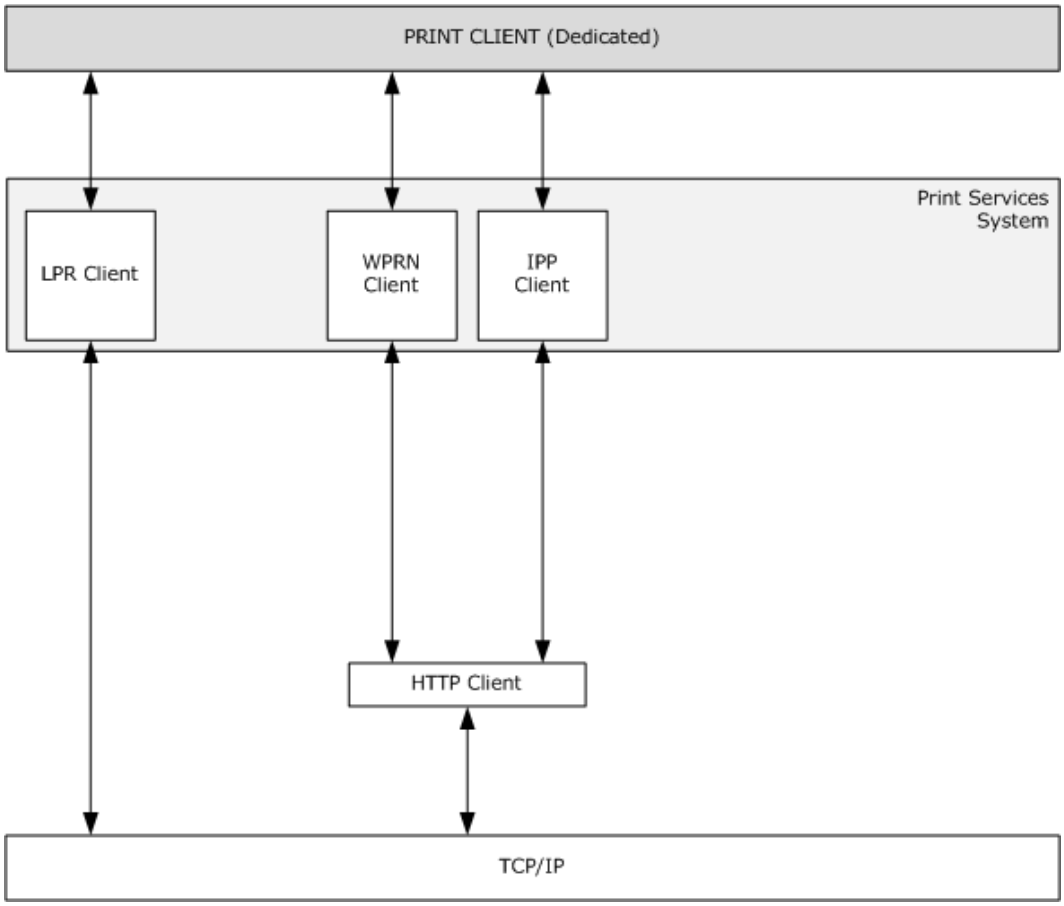

Figure 14: Optional protocol extensions for a Print Client

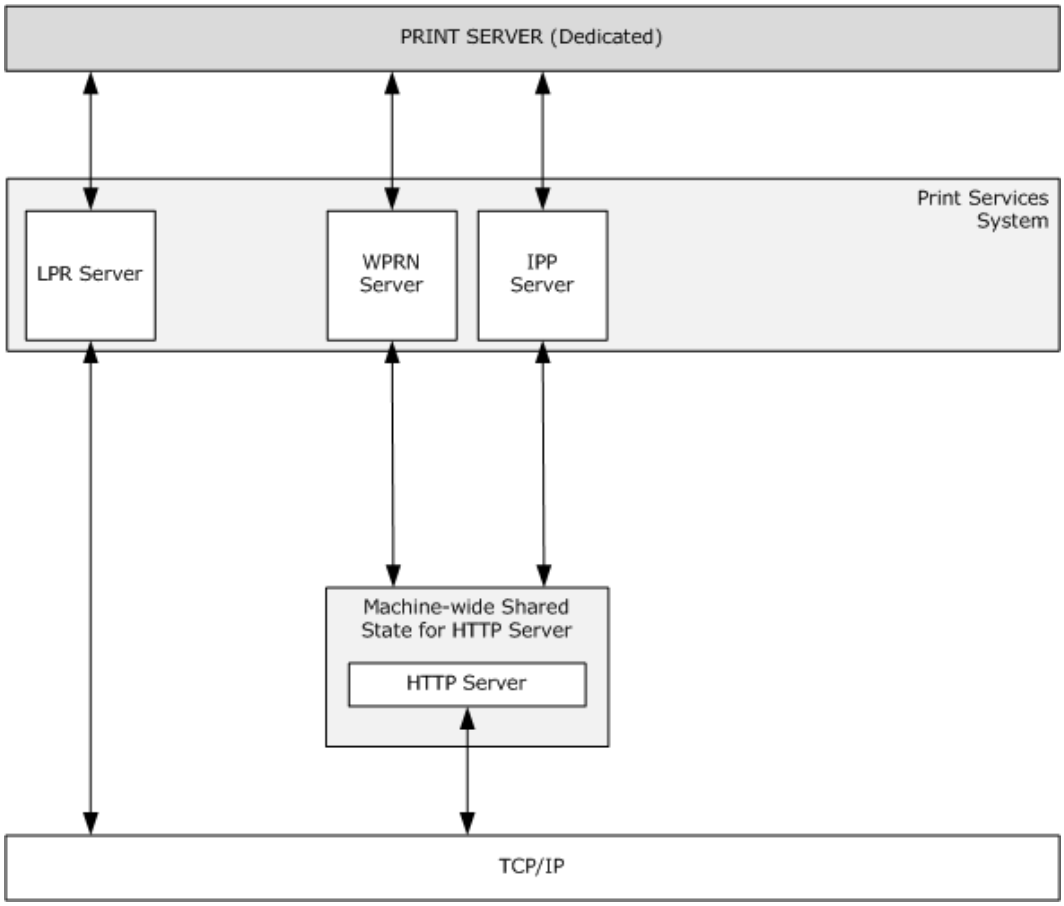

Figure 15: Optional protocol extensions for Print Server enabling Print Server to support internet and LPD Print Clients

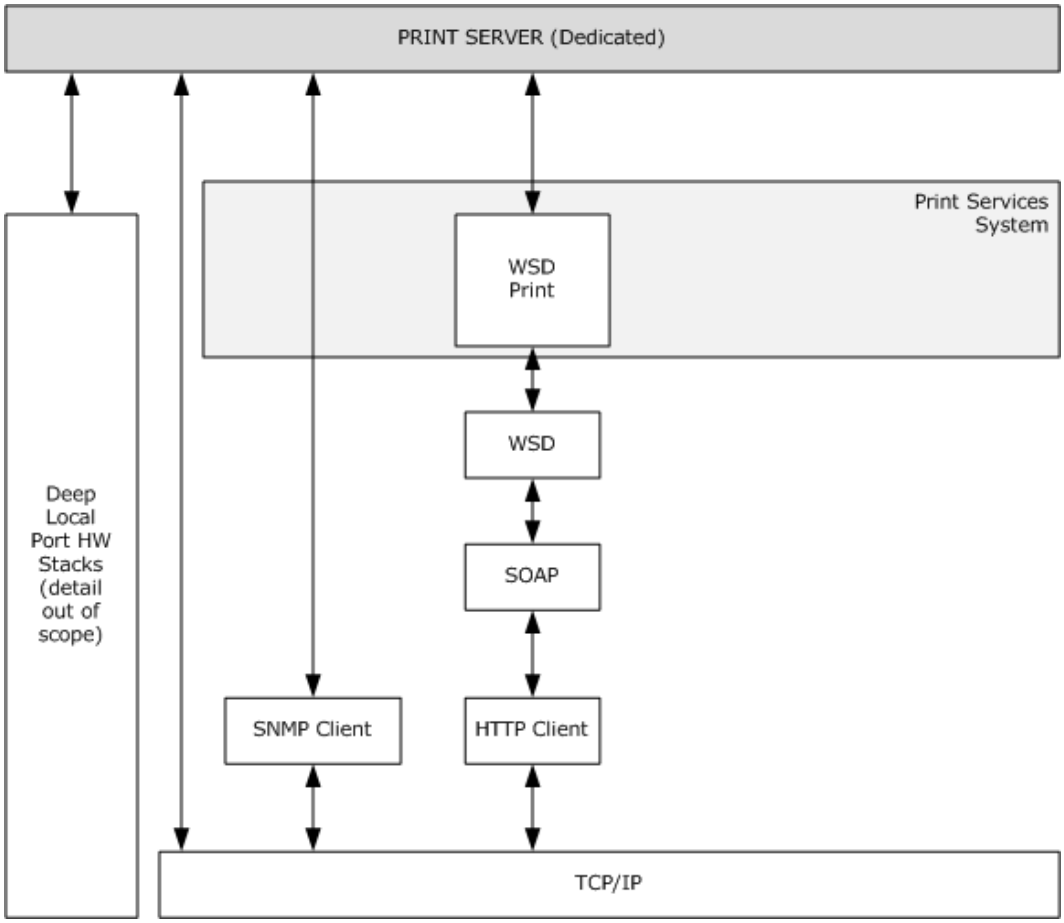

Figure 16: Protocol extensions used by Print Server to communicate with printers

5.3.1 Member Protocol Roles

The protocols used by the Print Services System perform roles as follows:

Printing: The protocols described in either [MS-RPRN] or [MS-PAR], or IPP, LPR, or the SMB Access Protocols are used for this role.

Managing print jobs: The protocols described in [MS-RPRN] or in [MS-PAR] are used for this role. The protocol described in [MS-RAP] is used for this role for Print Clients and Print Servers that do not support the protocols described in [MS-RPRN] or [MS-PAR].

Managing the Print Services System: The protocols described in [MS-RPRN] or in [MS-PAR] are used for this role. The protocol described in [MS-RAP] is used for this role for Print Clients and Print Servers that do not support the protocols described in [MS-RPRN] or [MS-PAR].

Receive notifications about general printing status: The protocol described in [MS-PAR] or [MS-RPRN] is used for this role.

Receive notifications about specific printing status from IHV-defined components: The protocol described in [MS-PAN] is used for this role.

Respond to notifications about specific printing status to IHV-defined components: The protocol described in [MS-PAN] is used for this role.

Download printer drivers to a client: The protocols described in the SMB Access Protocols or [MS-WPRN] (based on HTTP) are used to download printer drivers from a Web site, a printer, or a Print Server in the same network as the Print Client.

Store print jobs: The protocol described in [MS-EMFSPOOL] or the XML Paper Specification is used as a payload for print jobs.

Configure Print Clients: The protocol described in [MS-GPDPC] is used for this role.

5.3.2 Member Protocol Groups

Member protocols are grouped to perform the following functions:

- IPP and [MS-WPRN]: Form a group to allow print queue connection and printing from internet clients.
- [MS-PAR], [MS-RPRN] and [MS-PAN]: Form a group to allow printing and IHV plug in-defined communication.
- [MS-RAP] and the SMB protocol family: Form a group that allows printing from clients only capable of PSS version 1.0 support.

5.4 System Internal Architecture

The following diagram illustrates the internal architecture of the Print Services System.

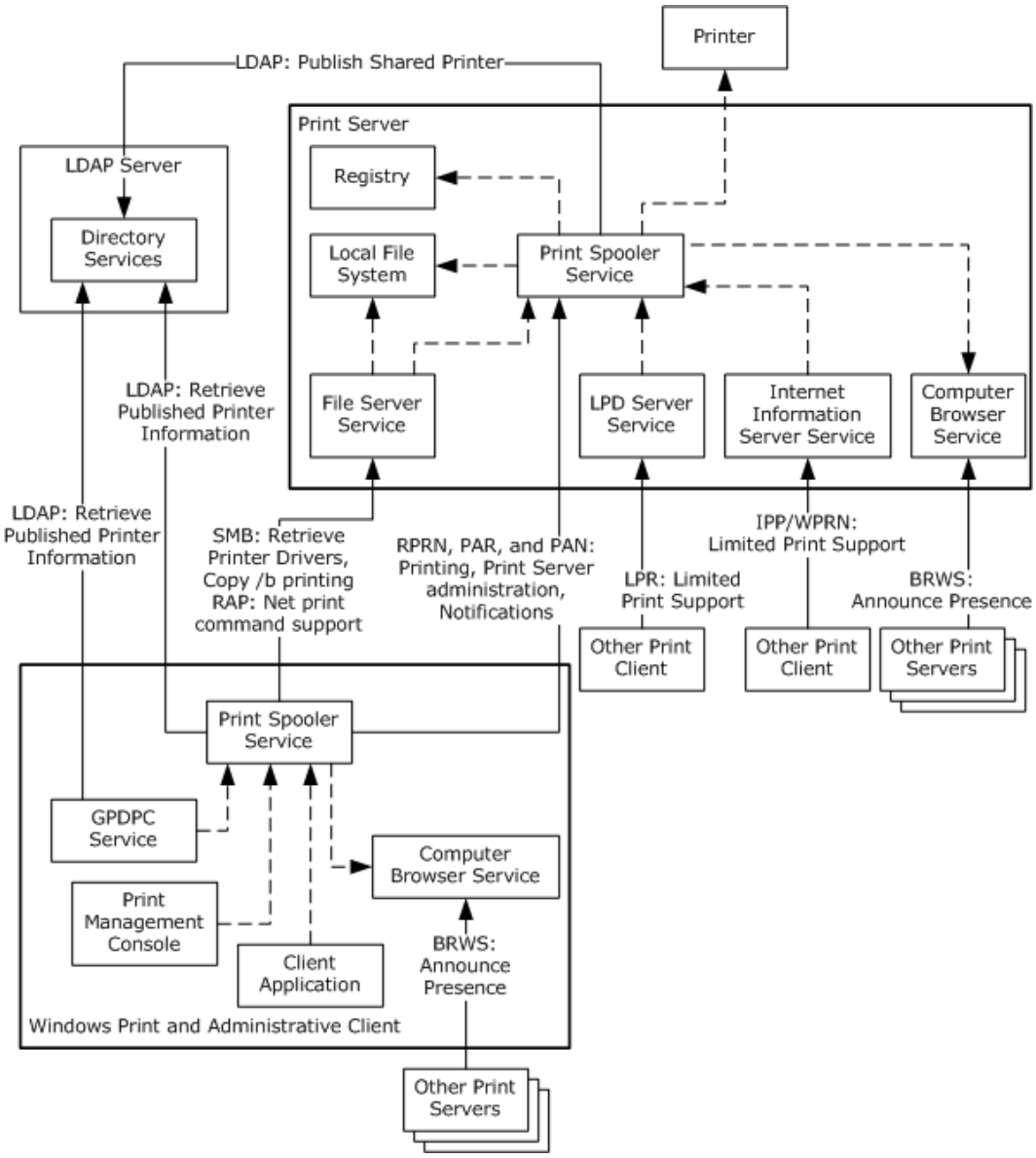

Figure 17: Print Services System internal system architecture

The Windows Print Client and Windows Administrative Client are shown in this diagram as a single client object, contained by a bolded rectangle. Although they are often separate entities, these clients can share many of the same components and are mainly differentiated by whether a client application or the Print Management Console (administrative application under an administrator user account) is running.

Within the Print Client, the Group Policy Service component (GPDPC Service) retrieves Group Policy settings that restrict Print Client connections to specified Print Servers. The GPDPC Service also creates Print Queue connections per-computer and per-user without user interaction.

To locate a Print Queue in a workgroup environment, the Print Spooler of the Print Client uses the local Computer Browser Service component, which continually listens for Print Server announcement

messages. To locate a Print Queue in a domain environment, the Print Spooler of the Print Client accesses an LDAP Server of the Active Directory System. The Print Spooler then sends a print job to the Print Spooler installed on the Print Server.

The Print Server contains a Print Spooler that receives communications from the Print Client's Print Spooler using the core protocols of the Print Services System (RPRN, PAR, and PAN).

Three Server Service components provide Print Server support for Print Clients that do not use the core protocols:

- The File Server Service (using SMB/RAP) provides support for Windows 98/ME Print Clients, as well as users entering print commands at the command line of more current versions of Windows.
- The LPD Server Service (using LPR) provides support for Unix/Linux Print Clients.
- The Internet Information Server Service (using IPP/WPRN) provides support for Print Clients when the Print Server is behind a firewall that prevents communications using the core protocols.

In a domain environment, the Print Server's Print Spooler publishes shared Print Queues (that correspond to shared printers) to an LDAP Server of the Active Directory System. In a workgroup environment, the Print Server's Print Spooler finds other Print Servers using the local Computer Browser Service, and uses the RPRN and PAR member protocols to announce shared Print Queues to each Print Server's local Computer Browser Service, which is continually listening for Print Server announcement messages.

The Print Server's Print Spooler persists system data in a Registry component. The Print Spooler loads print driver, port monitor, print processor and language monitor modules using the Local File System. The Print Spooler also uses the Local File System of the Print Server to buffer print jobs.

5.4.1 Mapping [MS-RAP], [MS-SMB] and [MS-BRWS] to the Print Services System

Print specific methods described in [MS-SMB] and [MS-RAP] are implemented on the server as transparent translations (that is, they are redirected) to calls to a local API of the print spooler. These translations are performed in a transient manner, and neither [MS-SMB] nor [MS-RAP], therefore, share ADM elements with the print services protocols.

The protocol described in [MS-BRWS] specifies a ServerType in ANNOUNCEMENT frames. The ServerType contains a bit indicating if the computer sending the ANNOUNCEMENT frame has shared printers. This section also contains details on how to obtain that bit.

This section specifies these translations in terms of equivalent methods described in [MS-RPRN] and data structures because [MS-RPRN] methods and data structures are equivalent to the local print spooler API.

In the following table, names for structures that have more than one version or level are written with an "X" suffix, and the mapping specifies rules for the entirety of member fields that are used in any of the different versions or levels. For example, "NetShareInfoX" refers to "NetShareInfo1" and "NetShareInfo2," and the mapping for the Type member applies equally for NetShareInfo1 and NetShareInfo2.

5.4.1.1 [MS-RAP] Mapping

[MS-RAP]	Equivalent [MS-RPRN] mapping
PrintQueueX response	RAP response's PrintQName MUST be set to same string value as

[MS-RAP]	Equivalent [MS-RPRN] mapping
	PRINTER_INFO_2.pShareName, truncated if necessary to fit. RAP response's Priority MUST be set to value of PRINTER_INFO_2.Priority. RAP response's StartTime MUST be set to value of least significant 16 bits of PRINTER_INFO_2.StartTime. RAP response's UntilTime MUST be set to value of least significant 16 bits of PRINTER_INFO_2.UntilTime. RAP response's SeparatorPageFileNameLow/High MUST be set to address a string in the response block, the value of which MUST be the same as PRINTER_INFO_2.pSepFile. RAP response's PrintProcessorDllNameLow/High MUST be set to address a string in the response block, the value of which MUST be the same as PRINTER_INFO_2.pPrintProcessor. RAP response's PrintDestinationsNameLow/High MUST be set to address a string in the response block, the value of which MUST be the same as PRINTER_INFO_2.pPortName. RAP response's PrintParameterStringLow/High MUST be set to address a string in the response block, the value of which MUST be the same as PRINTER_INFO_2.pParameters. RAP response's CommentStringLow/High MUST be set to address a string in the response block, the value of which MUST be the same as PRINTER_INFO_2.pComment. RAP response's DriverNameLow/High MUST be set to address a string in the response block, the value of which MUST be the same as PRINTER_INFO_2.pDriverName. RAP response's PrintersLow/High MUST be set to address a string in the response block, the value of which MUST be the same as PRINTER_INFO_2.pPrinterName. RAP response's PrintQStatus MUST be set to PRQ_PAUSED if PRINTER_INFO_2.Status has the PRINTER_STATUS_PAUSED bit set, or else be set to PRQ_ERROR if PRINTER_INFO_2.Status has the PRINTER_STATUS_ERROR bit set, or else be set to PRQ_PENDING if PRINTER_INFO_2.Status has the PRINTER_STATUS_PENDING_DELETION bit set, or else be set to PRQ_ACTIVE. RAP response's PrintJobCount MUST be set to value of PRINTER_INFO_2.cJobs. RAP response's DriverDataOffsetLow/High MUST be set to 0.
PrintJobInfoX response If NetPrintQEnumRequest.InfoLevel or NetPrintQGetInfoRequest.InfoLevel is 2 or 4 All NetPrintJobGetInfo	Obtain array of JOB_INFO_2 structures for printer identified by PRINTER_INFO_2 by following these steps: 1. RpcOpenPrinter(PRINTER_INFO_2.pPrinterName). 2. RpcEnumJobs with NoJobs set to 0xFFFFFFFF (enumerate all jobs), and Level set to 2. 3. RpcClosePrinter. If the RAP request operates on a single job, use the JobID field from the request and find JOB_INFO_2 with matching JOB_INFO_2.JobId. RAP response's JobID MUST be set to the value of

[MS-RAP]	Equivalent [MS-RPRN] mapping
	JOB_INFO_2.JobId. RAP response's UserName MUST be set to same string value as JOB_INFO_2.pUserName, truncated if necessary to fit. RAP response's NotifyName MUST be set to same string value as JOB_INFO_2.pNotifyName, truncated if necessary to fit. RAP response's DataType MUST be set to same string value as JOB_INFO_2.pDatatype, truncated if necessary to fit. RAP response's PrintParameterStringLow/High MUST be set to address a string in the response block, the value of which MUST be the same as JOB_INFO_2.pParameters. RAP response's PrintProcessorStringLow/High MUST be set to address a string in the response block, the value of which MUST be the same as JOB_INFO_2.pPrintProcessor. RAP response's QueueNameLow/High MUST be set to address a string in the response block, the value of which MUST be the same as PRINTER_INFO_2.pShareName. RAP response's PrinterNameLow/High MUST be set to address a string in the response block, the value of which MUST be the same as JOB_INFO_2.pPrinterName. RAP response's DriverNameLow/High MUST be set to address a string in the response block, the value of which MUST be the same as JOB_INFO_2.pDriverName. RAP response's JobPosition MUST be set to the value of JOB_INFO_2.Position. RAP response's JobStatus MUST be set to a bitwise OR combination of the following values: ▪ PRJ_QS_SPOOLING if JOB_INFO_2.Status has the JOB_STATUS_SPOOLING bit set ▪ PRJ_QS_PAUSED if JOB_INFO_2.Status has the JOB_STATUS_PAUSED bit set ▪ PRJ_QS_PRINTING if JOB_INFO_2.Status has the JOB_STATUS_PRINTING bit set ▪ PRJ_ERROR if JOB_INFO_2.Status has the JOB_STATUS_ERROR bit set RAP response's JobStatusStringLow/High MUST be set to address a string in the response block, the value of which MUST be the same as JOB_INFO_2.pStatus. RAP response's StatusStringLow/High MUST be set to address a string in the response block, the value of which MUST be the same as JOB_INFO_2.pStatus. RAP response's TimeSubmitted MUST be set to the value of JOB_INFO_2.Submitted, converted to GmtTime in seconds since 1/1/1970. RAP response's JobSize MUST be set to the value of JOB_INFO_2.Size. RAP response's JobCommentStringLow/High MUST be set to address a string in the response block, the value of which MUST be the same as JOB_INFO_2.pDocument.

[MS-RAP]	Equivalent [MS-RPRN] mapping
	RAP response's UserNameLow/High MUST be set to address a string in the response block, the value of which MUST be the same as JOB_INFO_2.pUserName. RAP response's DocumentNameLow/High MUST be set to address a string in the response block, the value of which MUST be the same as JOB_INFO_2.pDocument. RAP response's Priority MUST be set to the value of JOB_INFO_2.Priority. RAP response's NotifyNameLow/High MUST be set to address a string in the response block, the value of which MUST be the same as JOB_INFO_2.pNotifyName. RAP response's DataTypeLow/High MUST be set to address a string in the response block, the value of which MUST be the same as JOB_INFO_2.pDatatype. RAP response's DriverDataOffsetLow/High MUST be set to 0.
NetPrintJobSetInfo	The Windows implementation only supports setting JobComment, as specified in RAP NetPrintJobSetInfoRequest ([MS-RAP] section 2.5.7.3.1). Obtain JOB_INFO_1 and modify job following these steps: 1. RpcOpenPrinter with name equal to NULL, which will return the handle of the local Print Server. 2. RpcGetJob with Level set to 1 and JobId as specified by NetPrintJobSetInfo. 3. If the request's ParamNum is JobComment, as specified in RAP NetPrintJobSetInfoRequest, then set JOB_INFO_1.pDocument to the string specified in the request. 4. If any changes are made to the JOB_INFO_1 structure, call RpcSetJob with Level set to 1, JobId as specified by the NetPrintJobSetInfo request, and a JOB_CONTAINER that holds the modified JOB_INFO_1. 5. RpcClosePrinter.
NetPrintJobPause	RpcOpenPrinter with name equal to NULL, which will return the handle of the local Print Server. RpcSetJob with Level set to zero, JobId as specified by the request, and a Command value of JOB_CONTROL_PAUSE. RpcClosePrinter.
NetPrintJobContinue	RpcOpenPrinter with name equal to NULL, which will return the handle of the local Print Server. RpcSetJob with Level set to zero, JobId as specified by the request, and a Command value of JOB_CONTROL_RESUME. RpcClosePrinter.
NetPrintJobDelete	RpcOpenPrinter with name equal to NULL, which will return the handle of the local Print Server.

[MS-RAP]	Equivalent [MS-RPRN] mapping
	RpcSetJob with Level set to zero and JobId as specified by the request, and a Command value of JOB_CONTROL_CANCEL. RpcClosePrinter.
NetShareEnum	Locally shared Print Queues (RpcEnumPrinters(PRINTER_ENUM_LOCAL | PRINTER_ENUM_SHARED)) MUST be returned in addition to other shares. The STYPE_PRINTQ bit ([MS-RAP] section 2.5.6.3.3) of_NetShareInfoX.Type MUST be set for each enumerated shared Print Queue in the response.
NetShareGetInfo	If this is called for a share name that matches a shared local Print Queue, the STYPE_PRINTQ bit ([MS-RAP] section 2.5.6.3.3) of NetShareInfoX.Type MUST be set in the response.
NetPrintQEnum	RpcEnumPrinters with Level set to 2 and Flags set to (PRINTER_ENUM_LOCAL | PRINTER_ENUM_SHARED), resulting in an array of PRINTER_INFO_2 structures for all shared printers, then populate PrintQInfoX for response in accordance with requested InfoLevel. Append the PrintJobX structures in accordance with the requested InfoLevel.
NetPrintQGetInfo	Obtain PRINTER_INFO_2 using the following steps, then populate the PrintQInfoX response structures according to the requested RAP InfoLevel. 1. RpcOpenPrinter with name specified by the RAP request's PrintQueueName. 2. RpcGetPrinter with Level set to 2 to retrieve PRINTER_INFO_2. 3. RpcClosePrinter. 4. Append the PrintJobX structures in accordance with the requested InfoLevel.
NetPrintJobGetInfo	Obtain JOB_INFO_2 using the following steps, then populate the PrintJobInfoX response structures according to the requested RAP InfoLevel. 1. RpcOpenPrinter with the name of the Print Server, which will return the handle of the Print Server. 2. RpcGetJob with Level set to 2 and JobId as specified by the request to retrieve JOB_INFO_2. 3. RpcClosePrinter.

5.4.1.2 [MS-SMB] Mapping when Sending Data to an SMB Share for a Print Queue

When the Print Server receives an [MS-SMB] file open request with write access on a share that represents a Print Queue, it will perform a sequence of actions equivalent to the following:

1. RpcOpenPrinter on share name: access = PRINTER_ACCESS_USE
2. RpcStartDocPrinter: specify an implementation defined string in the DOC_INFO_1. Windows uses Local Downlevel Document.
3. Return a file handle to the Print Client representing this open Print Queue.

Subsequent [MS-SMB] write requests from the Print Client will result in RpcWritePrinter calls to the open Print Queue.

Upon the Print Client calling [MS-SMB] close handle, the Print Server calls RpcEndDocPrinter and RpcClosePrinter on the open Print Queue.

5.4.1.3 [MS-BRWS] Mapping

When sending any ANNOUNCEMENT frame, the SV_TYPE_PRINTQ_SERVER bit ([MS-RAP] section 2.5.5.2.1) of ServerType MUST be set if RpcEnumPrinters(PRINTER_ENUM_LOCAL | PRINTER_ENUM_SHARED) returns a non-zero number of Print Queues.

5.4.2 [MS-PAR], [MS-RPRN], and [MS-PAN] Mapping

There is no relationship between the notifications delivered by the RPRN and PAR protocols [MS-RPRN] [MS-PAR] and the PAN protocol [MS-PAN].

The RPRN/PAR notification mechanism delivers a well-known, well-defined set of notifications ([MS-RPRN] sections 2.2.1.13, 2.2.3, 3.1.4.10, 3.2.4.1, and 3.2.4.2.4) about status changes in Print System ADM elements, including print jobs, printer devices, and printer properties. The print spooler uses this notification mechanism to provide a user with print job and printer device feedback, and to reflect the printer property changes of a shared server print queue on the print client connected to it.

The PAN notification mechanism provides a method for IHV-supplied components, including printer drivers and monitor objects, to communicate between the print client and print server. IHVs use the mechanism provided by PAN in custom components to provide feedback and instructions to the user. PAN provides a set of pre-defined messages and also allows IHVs to define custom messages. A print server does not send any PAN messages to a print client unless an IHV-defined component is being used, which is using the PAN notification mechanism.

PAN uses the print server object or print queue objects to define the intended visibility and scope of a communication.

To check security permissions, RPRN, PAR and PAN use the security descriptor of the print server object or print queue objects.

5.5 Failure Scenarios

The following failure scenarios are discussed in this section:

- Abnormal termination of the print spooler service
- Loss of network connectivity
- Loss of system disk storage
- Print spooler service out of system resources
- Authentication issues

- Expected failures

5.5.1 Abnormal Termination of the Print Spooler Service

This problem is typically caused by a malfunctioning plug-in module, such as an IHV-supplied printer driver. Other causes include internal malfunction of the print spooler service and sudden loss of system power. The print spooler service terminates immediately without orderly shutdown. The system will attempt to restart the print spooler service multiple times before giving up, at which point administrative intervention locally on the system experiencing the failure is required to diagnose the problem and take corrective action.

All current completely submitted print jobs remain queued and their processing will be restarted from the beginning upon print spooler restart.

Print clients will not be informed of an abnormal termination event, and will experience failures on currently executing and all subsequent calls through member protocols. Print clients MUST implement appropriate RPC rundown handlers to free RPC resources (handles) that are in use at the time of failure. Print clients need to be designed to tolerate such failures and to retry until the print server is available again.

In a system configuration in which printer drivers are run by the print spooler in the print spooler's address space, an abnormal termination or memory corruption in a printer driver can abnormally terminate the print spooler process.

In a system configuration in which printer drivers are run in isolation processes (sandbox),<10> a malfunctioning printer driver will only cause the isolation process to terminate. Print clients will not be aware of such a failure. All current completely submitted print jobs for all printer drivers sharing the terminated isolation process remain queued and their processing will be restarted from the beginning upon restart of the isolation process. The print spooler will attempt to restart the isolation process for a malfunctioning printer driver two times before giving up, in which case administrative intervention locally on the system experiencing the failure is required to diagnose the problem and take corrective action.

5.5.2 Loss of Network Connectivity

In case of connectivity loss, each component protocol handler will retry independently to reconnect. There is no system-wide detection of connectivity loss.

5.5.3 Loss of System Disk Storage

Loss of system disk storage is a fatal, non-recoverable event. The behavior of the system in this case is undefined and unpredictable. The print spooler service of the affected system MUST be restarted manually once system storage comes back online.

5.5.4 Print Spooler Service Out of System Resources

Methods on component protocols return a non-zero error code if the server cannot process the request due to lack of system resources (such as memory, handles, or disk space). There is no system-wide mechanism for dealing with a server that is out of system resources.

5.5.5 Authentication Issues

Authentication is handled independently for each component protocol. Credential storage on Windows makes a distinction between session based transports (RPC over SMB) and non-session based transports (RPC over TCP/IP). If secondary credentials have been added by the user or

another subsystem to the credential manager, these will only be used for protocol requests using session based transport, and be unavailable for non-session based transport. Specifically, that means that requests over protocols defined in [MS-PAR] and [MS-PAN] can fail due to failed authentication even if requests over the protocol defined in [MS-RPRN] operate correctly. Print Clients MUST be designed to handle these failures gracefully, and fall back to using [MS-RPRN] if [MS-PAR]/[MS-PAN] fail due to authentication issues. As a consequence, functionality specific to [MS-PAR] or [MS-PAN] may be unavailable in such a fall back scenario.

5.5.6 Expected Failures

A group of failures caused by malfunctioning printers are considered expected failures and the member protocols contain facilities to report the type of failure to the Print Client. The Print Client responds to failures typically in an implementation-specific manner, such as by retrying (for example, in a printer offline scenario) and offering visual feedback to the user. The user can then make a choice of further action, such as canceling the job, and reprinting to a different Print Queue, or walking up to the printer and fixing a paper jam.

6 System Details

This section contains the details that complete the descriptions in earlier sections of the document. These details are needed to understand and implement this system.

6.1 Architectural Details

The following examples illustrate the protocols used by the Print Services System:

- Discover and utilize a Print Queue in a domain
- Discover and utilize a Print Queue in a workgroup
- Locate a Print Queue in a domain and establish a connection, then submit a print job to a manual duplex printer using the protocol defined in [MS-PAN] and enable bidirectional IHV-defined communication between the Print Client and the Print Server
- Enumerate print jobs from all users, then cancel several print jobs
- Provision a Print Queue using the protocol defined in [MS-RPRN] from an Administrative Client, then delete the same Print Queue using the protocol defined in [MS-PAR] from a different Administrative Client
- Send a print job to an SMB share

In all the examples discussed in this section, the Print Client will open, close, and reopen the handle to the Print Queue multiple times. This is because the Windows API implementation for the print spooler allows and uses different processes for the Print Client user interface, Print Client applications, and internal features of the print spooler. Each of these processes can have multiple threads (for example, an application can have one thread printing in the background and another thread computing layout in the foreground). Other queries to the Print Server (for example, about print jobs) may happen in parallel from different threads, causing additional exchanges unrelated to a specific scenario on the wire.

In all the examples discussed in this section, the Print Client will retrieve information from the Print Server in arbitrary order, and often will retrieve the same information redundantly. Methods that may be called in arbitrary order and redundantly are marked with (*) in the examples.

Because handles are not generally portable between threads (and are never portable between processes), the Print Client may also use different handles to the same Print Queue for different functions, such as listening for notifications and submitting a print job.

6.1.1 Discover and Utilize a Print Queue in a Domain

This example demonstrates the use cases described in sections 3.3.4.3 (variation (a) with extensions (a) and (c)) and 3.3.4.7 (variation (a) with extension (a)).

Prior to the beginning of this example, an Administrator has installed a printer on a print server and provisioned a print queue corresponding to the printer. The print server has published the name of the print queue to the directory service using the LDAP protocol. During this scenario, LDAP is used to discover the available shared print queues, after which the RPRN protocol [MS-RPRN], as well as the SMB Access Protocols, are used for subsequent operations.

The sequence of steps for this example is organized into the following sections:

- A. Locate print queue in a domain

- B. Establish a connection, download a driver, and register for notifications
- C. Submit a print job and receive notification
- D. Unregister for notifications and disconnect from print queue

Initial System State and Prerequisites

The example described in this section applies under the following conditions:

- The print client communicates with the print server using the RPRN protocol, and RPRN includes support for **RpcGetPrinterDriverPackagePath**.<11>
- The print queue has been provisioned and is located on a print server in a domain.
- The print server has published a list of the shared print queues to the Active Directory System.
- The user of the print client does not know the name of the print server and the print client does not yet have a list of the available print servers or print queues.
- The print client does not have the printer driver for the print queue.
- The print client does not have the printer driver package for the printer driver in its local driver store. (This precondition is present to illustrate a printer driver download from the print server.)
- The print client is configured not to use Windows Update to obtain the printer driver. (This precondition is present to illustrate a printer driver download from the print server.)
- The printer driver is available from the print server.

Sequence

A. Locate print queue in a domain

The following diagram illustrates the first part of this example in which the print client locates a print queue in a domain using LDAP.

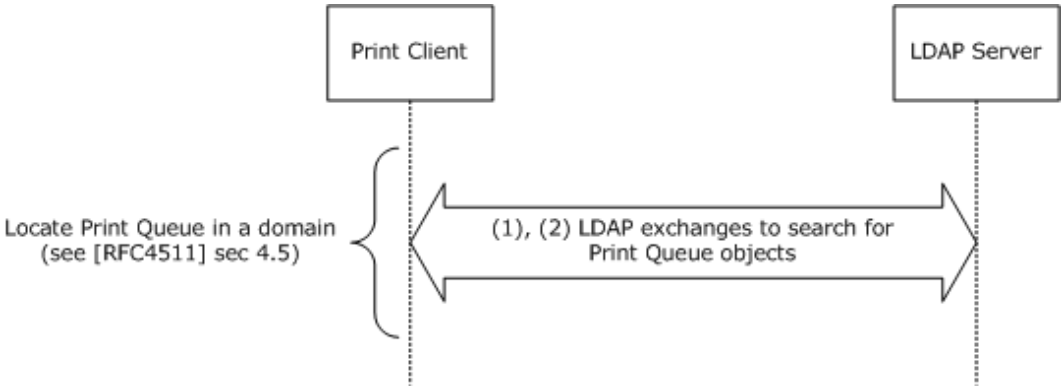

Figure 18: Print client locating a print queue in a domain using LDAP

The following steps describe this sequence.

1. The print client uses LDAP to query the LDAP server for print queues, optionally specifying criteria, such as functionality, that the print queues MUST satisfy. The print client obtains criteria

from the user interface and makes several queries to obtain criteria, such as the physical location of the print client.

2. The LDAP Server returns a list of print queues using LDAP. The print client user interface then presents the list to the user, who selects a print queue.

B. Establish a connection, download a driver, and register for notifications

The following diagram illustrates a print client connecting to a print queue, downloading a driver, and registering for notifications. The sequence is described in the steps that follow the diagram.

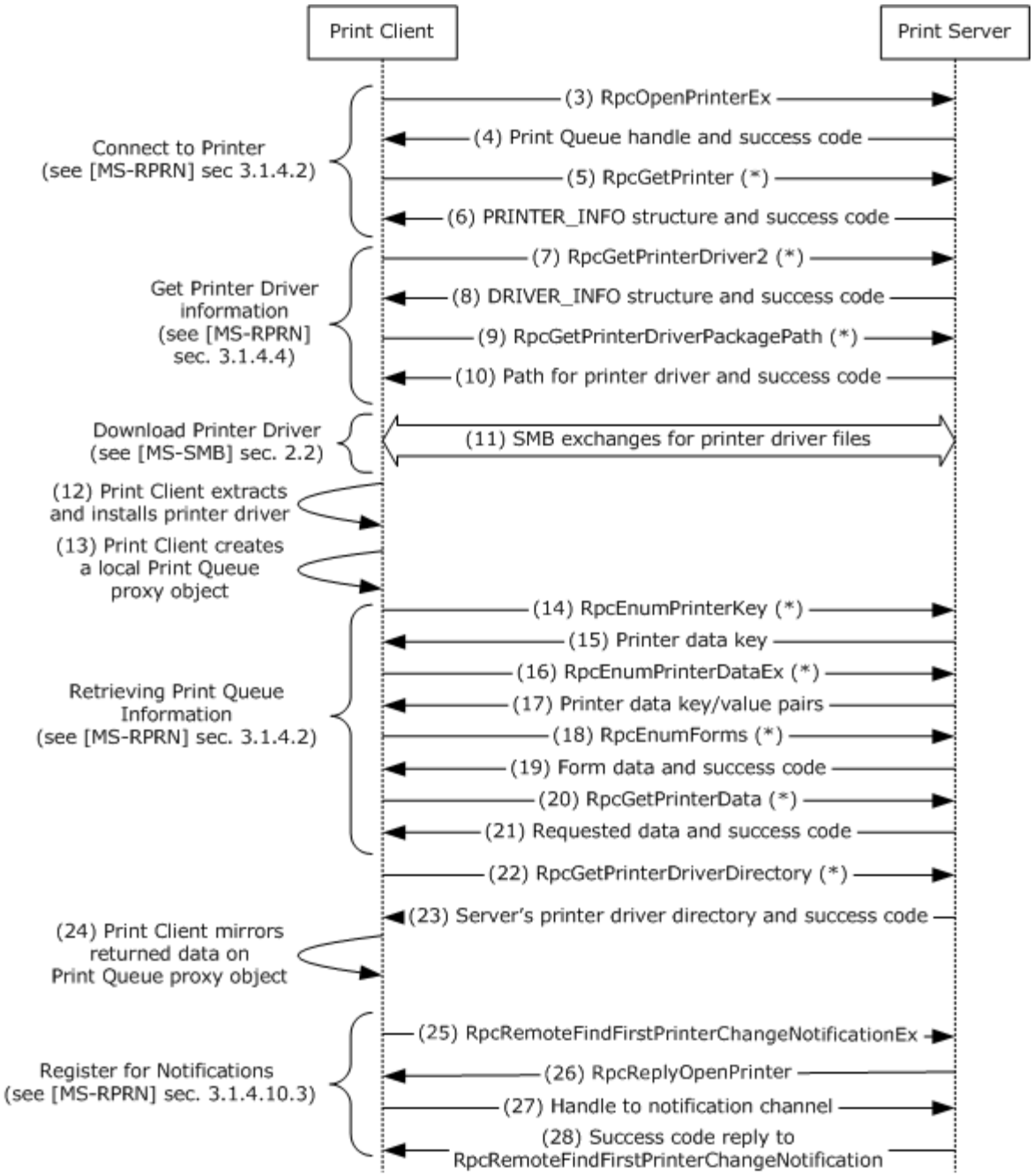

Figure 19: Print client connecting to a print queue, downloading a driver, and registering for notifications

The following steps describe this sequence, which is a continuation of sequence A.

3. The print client calls the **RpcOpenPrinterEx** method on the print server for the designated print queue.

4. The print server returns a handle to the print queue and a success code.

5. The print client calls the **RpcGetPrinter** (*) method on the print server for the print queue handle.

6. The print server returns **PRINTER_INFO** structure with details about the print queue and a success code.

7. The print client calls the **RpcGetPrinterDriver2** (*) method on the print server for the print queue handle.

8. The print server returns the requested **DRIVER_INFO** structure, which contains details about the print queue's printer driver, such as driver file name and driver version, and a success code.

9. The print client calls the **RpcGetPrinterDriverPackagePath** (*) method on the print server for the print queue handle.

10. The print server returns the path for the printer driver package cabinet file and a success code.

11. The print client downloads the printer driver using the [MS-SMB] protocol family, utilizing the following File Services System operations, as documented in [MS-FSSO]: (a) Open File SMB (see [MS-FSSO] section 6.1.5.13), (b) Perform SMB Query Information (see [MS-FSSO] section 6.1.5.18), (c) Perform File Operations SMB Read (see [MS-FSSO] section 6.1.5.19), (d) Perform File Operation SMB Close (see [MS-FSSO] section 6.1.5.16).

12. The print client extracts the printer driver files contained in the downloaded data file and installs the printer driver.

13. The print client creates a connection to the print queue by creating a local print queue proxy object.

14. The print client calls the **RpcEnumPrinterKey** (*) method on the print server for the designated print queue handle.

15. The print server returns the requested printer data keys and a success code.

16. The print client calls the **RpcEnumPrinterDataEx** (*) method on the print server for the designated print queue handle.

17. The print server returns the requested printer data key/value pairs and a success code.

18. The print client calls the **RpcEnumForms** (*) method on the print server for the designated print queue handle.

19. The print server returns the requested form data and a success code.

20. The print client calls the **RpcGetPrinterData** (*) method on the print server for the designated print queue handle to obtain values of specific keys.

21. The print server returns the requested data and a success code.

22. The print client calls the print server's **RpcGetPrinterDriverDirectory** method.<12>

23. The print server responds with the name of its printer driver directory and a success code.

24. The print client mirrors the returned data on the local print queue proxy object.

25. The print client calls **RpcRemoteFindFirstPrinterChangeNotificationEx** on the print server with a print queue handle to begin the process for registering to receive print status notifications.

26. Print server calls **RpcReplyOpenPrinter** on the print client to establish a notification channel.

27. Print client returns the **RpcReplyOpenPrinter** call with a handle to the notification channel.

28. Print server returns the **RpcRemoteFindFirstPrinterChangeNotificationEx** call with a success code.

C. Submit a print job and receive notifications

The following diagram illustrates the print client submitting a print job to the print queue and receiving notifications.

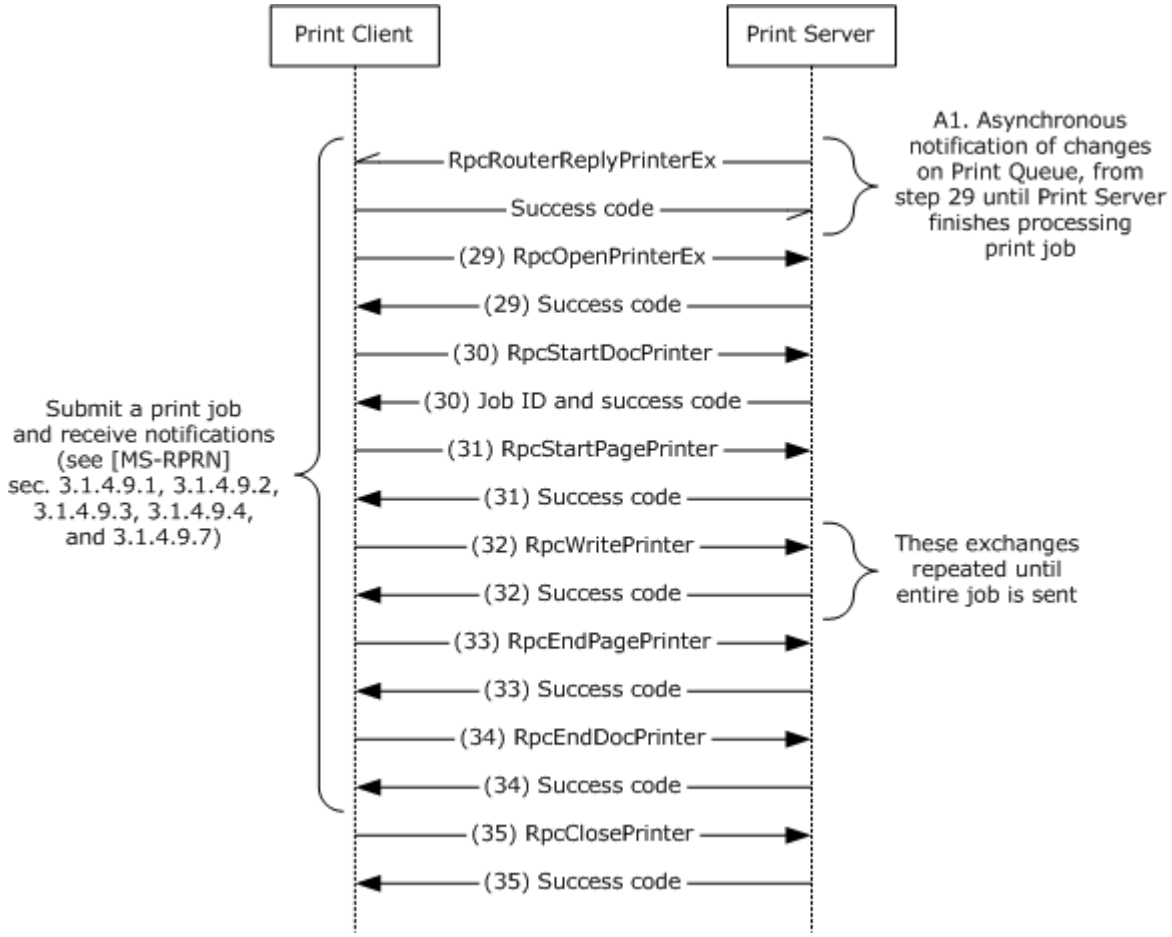

Figure 20: Print client submitting a print job to the print queue

The following steps describe this sequence, which is a continuation of sequence B.

A1. Asynchronously (on a separate thread), starting with step 29 and continuing until the print server is done processing the print job, the print server calls the **RpcRouterReplyPrinterEx** method on the print client to notify the print client of changes on the print queue, such as job processing progress, status changes, or other changes. The print client responds to each notification with a success code.

29. The print client calls **RpcOpenPrinterEx** method on the print server to open a handle to the print queue for submitting a print job, and the print server returns a success code.

30. The print client calls **RpcStartDocPrinter** method on the print server, and the print server returns a job ID and a success code.

31. The print client calls the **RpcStartPagePrinter** method on the print server, and the print server responds with a success code.

32. The print client repeatedly calls the **RpcWritePrinter** method on the print server, sending successive portions of the print job. The print server responds to each call with a success code. This step is repeated until all the print data has been sent.

33. The print client calls the **RpcEndPagePrinter** method on the print server, and the print server responds with a success code.

34. When all print data has been sent, the print client calls **RpcEndDocPrinter** method on the print server and the print server responds with a success code.

35. The print client calls the **RpcClosePrinter** method on the print server and the print server returns a success code (this step will only occur if the print client does not proceed to the next portion of this example).

D. Unregister for notifications

The following diagram illustrates the print client unregistering for notifications.

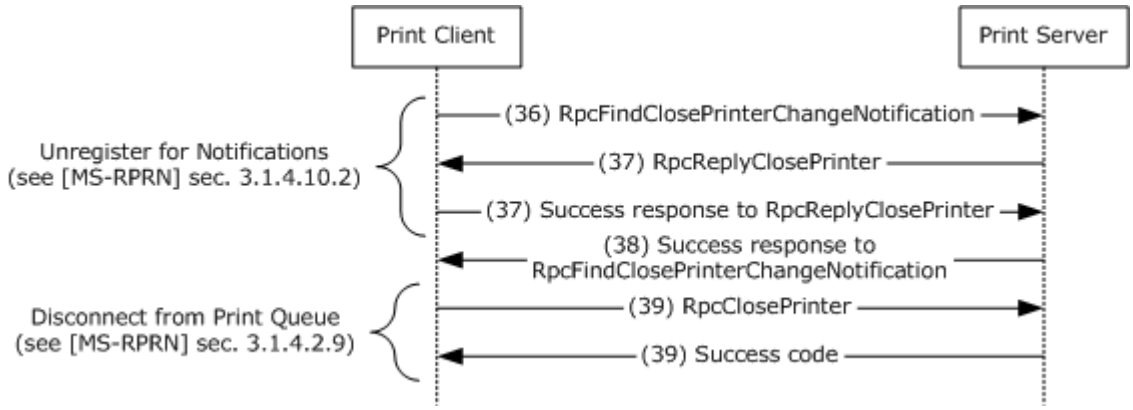

Figure 21: Print client unregistering for notifications

The following steps describe this sequence, which is a continuation of sequence C.

36. Once a print client is no longer interested in notifications (for example, the user closes the print queue dialog user interface), the print client calls the **RpcFindClosePrinterChangeNotification** method on the print server.

37. The print server calls the **RpcReplyClosePrinter** method on the print client and the print client returns a success code.

38. The print server responds to the **RpcFindClosePrinterChangeNotification** method with a success code.

39. The print client calls the **RpcClosePrinter** method on the print server using the handle to the print queue specified during registration in step 23, and the print server returns a success code.

It is common for clients to listen for notifications from more than one thread, so the sequence of steps for registration for notifications (23 through 26) and unregistration (34 through 37) can be observed on the wire multiple times.

6.1.2 Discover and Utilize a Print Queue in a Workgroup

This example demonstrates the use cases described in sections 3.3.4.4 (variation (a)) and 3.3.4.7 (variation (a)).

In this example, a computer running Windows® XP operating system and located in a workgroup locates a shared print queue on another computer in the workgroup and sends a print job to the shared print queue.

The sequence of steps for this example is organized into the following sections:

- A. Locate print queue in a workgroup
- B. Establish a connection and register for notifications
- C. Submit a print job and receive notifications
- D. Unregister for notifications

In this example, there is no dedicated print server, no domain has been established, and no Directory System is available. The clients in the workgroup are distinguished by their roles, already established, as follows:

- Print server: One client in the workgroup is established as a print server and has already announced the shared print queues it hosts to all other print servers in the workgroup (if there are any), using the RPRN protocol [MS-RPRN].
- Print client: One client is distinguished as a print client in the workgroup when directed by the user to send a print job to a print server. This print client can behave in other roles, such as print server or print browser server, but for this example the client will behave in the role of a print client only.
- Print browser server: One client in the workgroup is established as the print browser server, which enumerates available print servers and print queues to print clients. The print browser server has already announced this role to all the clients in the workgroup using the protocol described in [MS-BRWS]. In order to act in the role of the print browser server, the print client MUST also have the role of a print server with at least one shared print queue available.

Only one print server in the workgroup behaves as the active print browser server at any given time. Other print servers (if any) behave as backup print browser servers, and one of them will become the new active print browser server if the current active print browser server becomes unavailable. The print browser server and the backup print browser servers exchange information about shared print queues in the workgroup by calling **RpcAddPrinter** [MS-RPRN] with **PRINTER_INFO_1** structures on each other.

Typically, the first workgroup client that takes on the role of a print server sharing a print queue also becomes the print browser server for the workgroup.

Initial System State and Prerequisites

- The general requirements as set forth in System Assumptions and Preconditions, section 4.2, are met.

- Participating clients are configured to be members of the same workgroup.
- The browser service MUST be in the Running state on all participating clients; that is, the protocol described in [MS-BRWS] MUST be supported by the participating clients.
- One of the clients MUST have at least one shared print queue, so that at least one client will be of type **SV_TYPE_PRINTQ_SERVER**, which is a requirement for this example.
- Either the print client or print server supports the RPRN protocol [MS-RPRN] but not the PAR protocol [MS-PAR]. This precondition was chosen to illustrate the use of RPRN.<13>
- The print client does not have the printer driver for the print queue.
- The print client does not have the printer driver package for the printer driver in its local driver store. This precondition is present to illustrate printer driver download from the print server.
- The print client is configured to not use Windows Update to obtain the printer driver. This precondition is present to illustrate printer driver download from the print server.
- The printer driver is available from the print server's print$ share.

Sequence

A. Locate print queue in a workgroup

The following diagram illustrates the first part of this example, in which a print client locates a print queue in a workgroup and establishes a connection to the print queue. The sequence is described in the steps which follow the diagram.

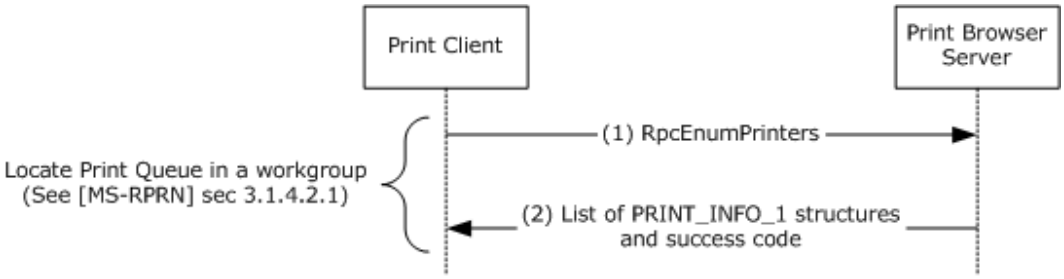

Figure 22: Print client locating a print queue in a workgroup

The following steps describe this sequence.

1. The print client calls the **RpcEnumPrinters** method on the print browser server using RPRN, requesting **PRINTER_INFO_1** structures.
2. The print browser server returns a list of **PRINTER_INFO_1** structures for all shared print queues in the workgroup. The user then selects a print queue from the user interface.

B. Establish a connection and register for notifications

The following diagram illustrates the second part of this example, in which a print client establishes a connection to a print queue and registers for notifications.

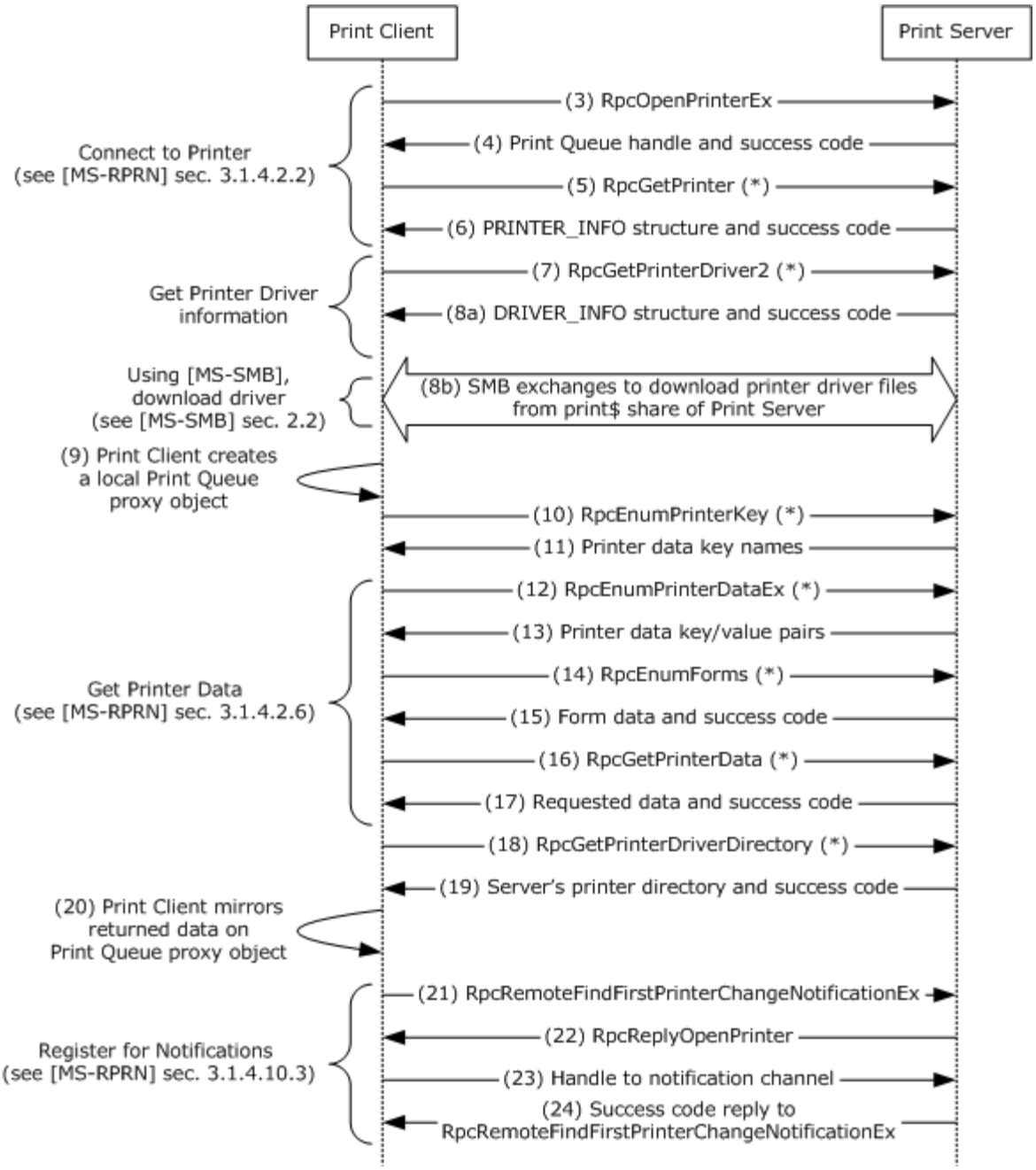

Figure 23: Print client establishing a connection and registering for notifications

The following steps describe this sequence, which is a continuation of sequence A.

3. The print client calls the **RpcOpenPrinterEx** method on the print server for the designated print queue.

4. The print server returns a handle to the print queue and a success code.

5. The print client calls the **RpcGetPrinter** (*) method on the print server for the print queue handle.

6. The print server returns **PRINTER_INFO** structure with details about the print queue and a success code.

7. The print client calls the **RpcGetPrinterDriver2** (*) method on the print server for the print queue handle.

8. This step consists of the following two parts:

   a. The print server returns the requested **DRIVER_INFO** structure, which contains details about the print queue's printer driver, such as driver file name and driver version, and a success code.

   b. The print client downloads the printer driver files from the print$ share of the print server using the SMB protocol family.

9. The print client creates a connection to the print queue by creating a local print queue proxy object.

10. The print client calls the **RpcEnumPrinterKey** (*) method on the print server for the designated print queue handle.

11. The print server returns the requested printer data key names and a success code.

12. The print client calls the **RpcEnumPrinterDataEx** (*) method on the print server for the designated print queue handle.

13. The print server returns the requested printer data key/value pairs and a success code.

14. The print client calls the **RpcEnumForms** (*) method on the print server for the designated print queue handle.

15. The print server returns the requested form data and a success code.

16. The print client calls the **RpcGetPrinterData** (*) method on the print server for the designated print queue handle to obtain values of specific keys.

17. The print server returns the requested data and a success code.

18. The print client calls the print server's **RpcGetPrinterDriverDirectory** method.<14>

19. The print server responds with the name of its printer driver directory and a success code.

20. The print client mirrors the returned data on the local print queue proxy object.

21. The print client calls **RpcRemoteFindFirstPrinterChangeNotificationEx** on the print server with a print queue handle to begin the process for registering to receive print status notifications.

22. The print server calls **RpcReplyOpenPrinter** on the print client to establish a notification channel.

23. The print client returns the **RpcReplyOpenPrinter** call with a handle to the notification channel.

24. The print server returns the **RpcRemoteFindFirstPrinterChangeNotificationEx** call with a success code.

C. Submit a print job and receive notifications

The following diagram illustrates the print client submitting a print job to the print queue.

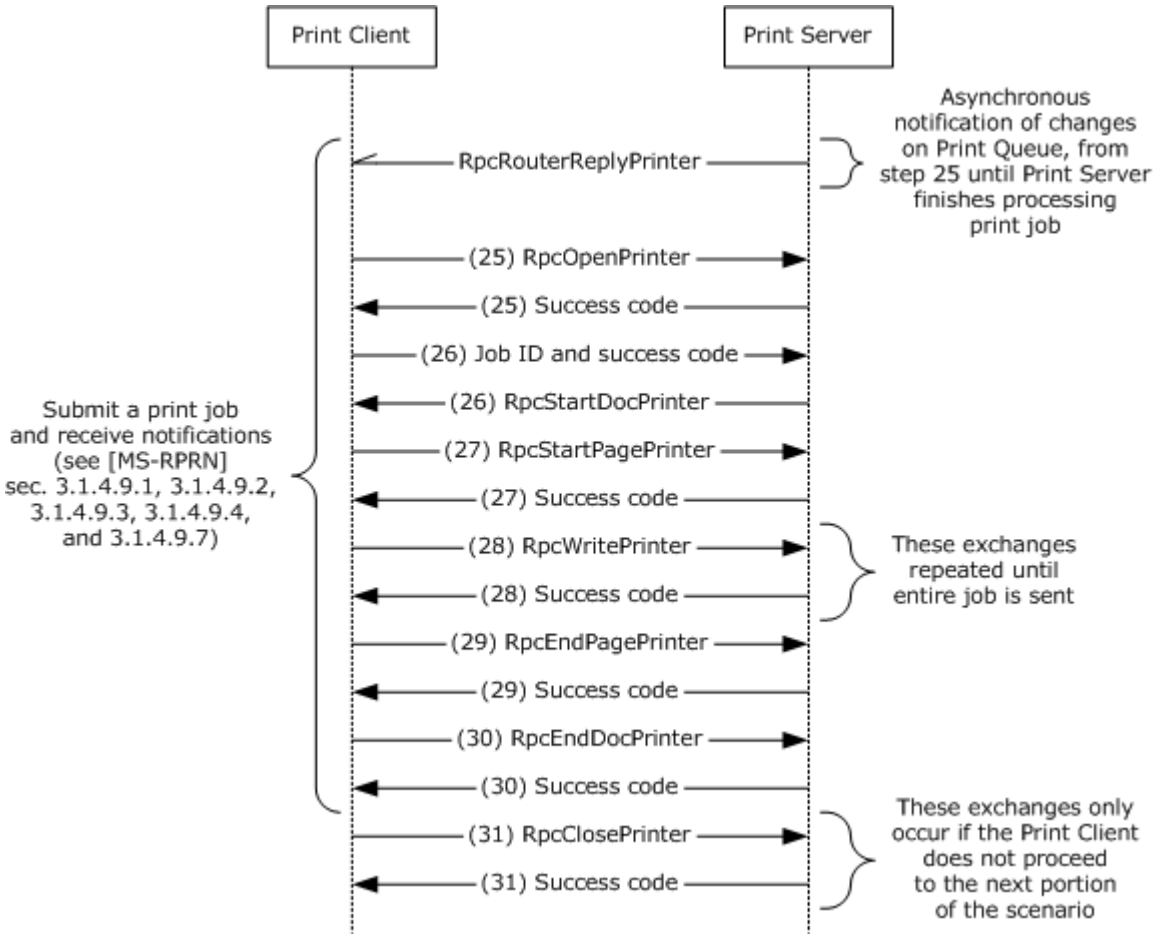

Figure 24: Print client submitting a print job to the print queue

The following steps describe this sequence, which is a continuation of sequence B.

Asynchronously (on a separate thread), starting with step 25 and continuing until the print server is done processing the print job, the print server calls the **RpcRouterReplyPrinterEx** method on the print client to notify the print client of changes on the print queue, such as job processing progress, status changes, or other changes.

25. The print client calls **RpcOpenPrinter** method on the print server to open a handle to the print queue for submitting a print job, and the print server returns a success code.

26. The print client calls **RpcStartDocPrinter** method on the print server and the print server returns a job ID and a success code.

27. The print client calls the **RpcStartPagePrinter** method on the print server, and the print server responds with a success code.

28. The print client repeatedly calls the **RpcWritePrinter** method on the print server, sending successive portions of the print job. The print server responds to each call with a success code. This step is repeated until all the print data has been sent.

29. The print client calls the **RpcEndPagePrinter** method on the print server, and the print server responds with a success code.

30. When all print data has been sent, the print client calls **RpcEndDocPrinter** method on the print server and the print server responds with a success code.

31. The print client calls the **RpcClosePrinter** method on the print server and the print server returns a success code (this step will only occur if the print client does not proceed to the next portion of this example).

D. Unregister for notifications

The following diagram illustrates the print client unregistering for notifications.

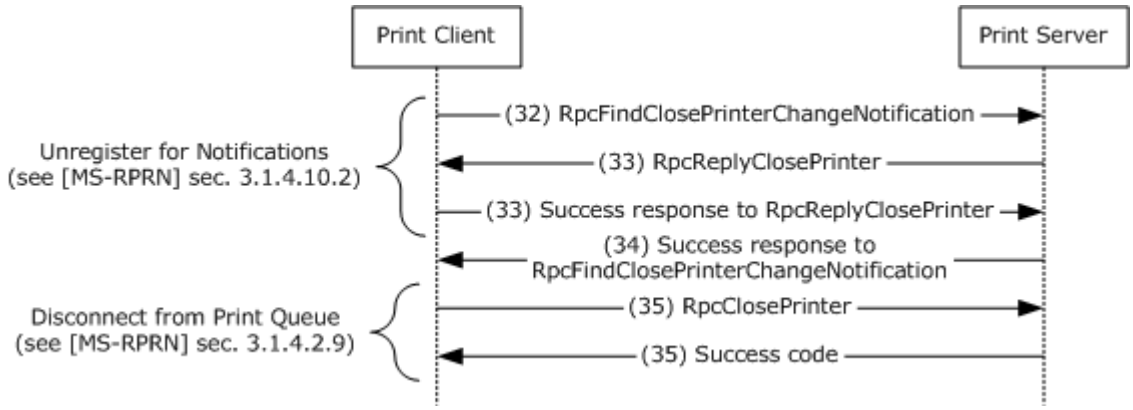

Figure 25: Print client unregistering for notifications

The following steps describe this sequence, which is a continuation of sequence C.

32. Once a print client is no longer interested in notifications (for example, the user closes the print queue dialog user interface), the print client calls the **RpcFindClosePrinterChangeNotification** method on the print server.

33. The print server calls the **RpcReplyClosePrinter** method on the print client and the print client returns a success code.

34. The print server responds to the **RpcFindClosePrinterChangeNotification** method with a success code.

35. The print client calls the **RpcClosePrinter** method on the print server using the handle to the print queue specified during registration in step 17 and the print server returns a success code.

It is common for clients to listen for notifications from more than one thread, so the sequence of steps for registration for notifications (17 through 20) and unregistration (28 through 31) can be observed on the wire multiple times.

Final State

For extension (a), the print queue has been located and the print client has established a connection to the print queue. The print client can now submit print jobs to the print queue.

For extension (b), a print job has been successfully submitted to a print queue and the print job is being processed by the print server.

6.1.3 Locate a Print Queue in a Domain and Establish a Connection, then Submit a Print Job to a Manual Duplex Printer Using [MS-RPRN] and [MS-PAR] and Enable Unidirectional IHV-defined Communication Between Print Client and Print Server Using [MS-PAN]

This example demonstrates the use cases described in sections 3.3.4.3 (variation (a)) and 3.3.4.7 (variation (b) with extension (d)).

The sequence of steps for this example is organized into the following sections:

- A. Locate a Print Queue in a domain
- B. Establish a connection, download a driver, and register for notifications
- C. Submit a print job and receive notifications

Initial System State and Prerequisites

- The Print Client and Print Server support the protocols described in [MS-RPRN], [MS-PAR], and [MS-PAN].<15>
- There is an IHV-defined print processor installed on the Print Server that serves as a [MS-PAN] notification source and implements a Manual Duplex operation, which prompts the user to walk up to the printer and reinsert printed pages.
- The Print Queue has been provisioned and is located on a Print Server in a domain.
- The Print Server has published a list of the shared Print Queues to the Directory System.
- The user of the Print Client does not know the name of the Print Server and the Print Client does not yet have a list of the available Print Servers or Print Queues.
- The Print Client does not have the printer driver for the Print Queue. This precondition is present to illustrate printer driver download from the Print Server.
- The Print Client does not have the printer driver package for the printer driver in its local driver store. This precondition is present to illustrate printer driver download from the Print Server.
- The printer driver is available from the Print Server.

Sequence

A. Locate a Print Queue in a domain

The following diagram illustrates the first part of this example in which the Print Client locates a Print Queue in a domain using the LDAP protocol.

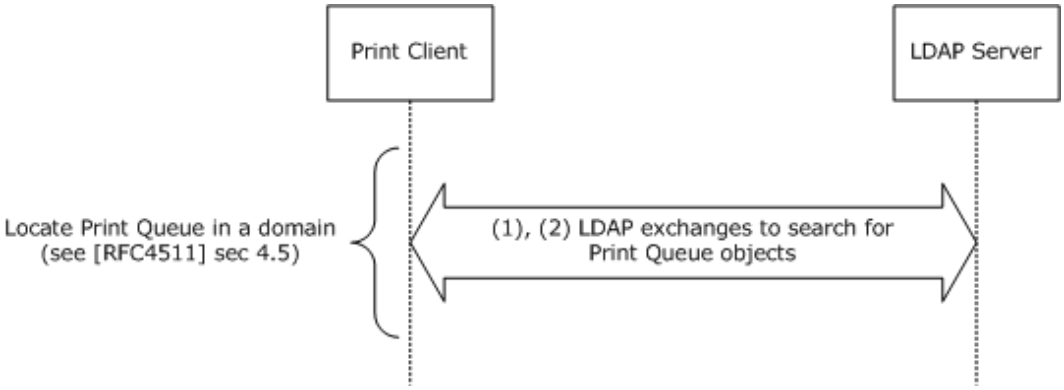

Figure 26: Print Client locating a Print Queue in a domain using the LDAP protocol

The following steps describe this sequence.

1. The Print Client uses LDAP to query the LDAP Server for Print Queues, optionally specifying criteria, such as functionality, that the Print Queues are required to meet. The Print Client obtains criteria from the user interface and also makes several queries to obtain criteria, such as physical location of the Print Client.

2. The LDAP Server returns a list of Print Queues by using LDAP. The Print Client user interface then presents the list to the user, who selects a Print Queue.

   B. Establish a connection, download a driver, and register for notifications

   The following diagram illustrates the second part of this example.

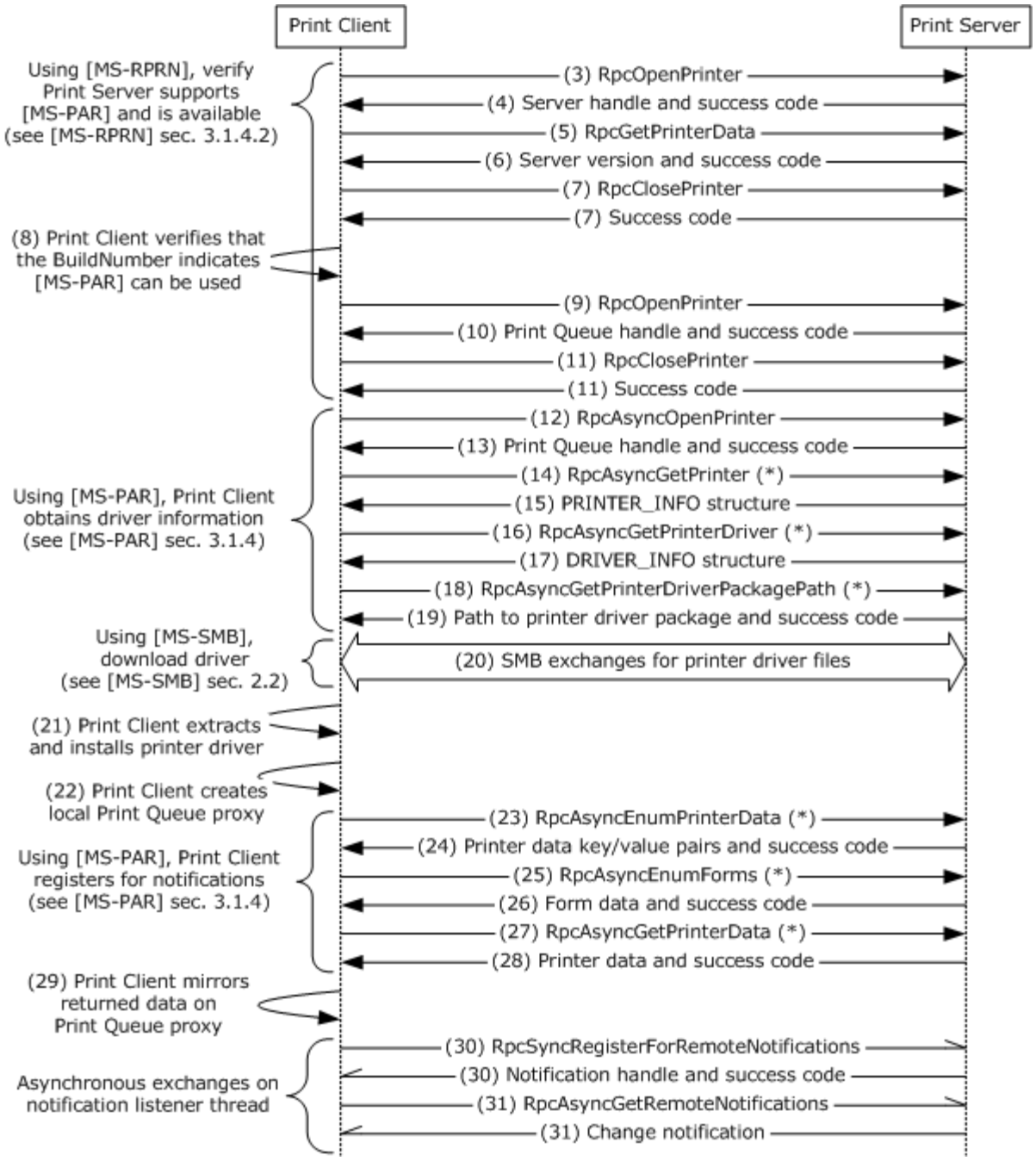

Figure 27: Print Client establishing a connection to a Print Queue, downloading a driver, and registering for notifications

The following steps describe this sequence, which is a continuation of sequence A.

3. The Print Client uses the RPRN protocol [MS-RPRN] to call the RpcOpenPrinter method on the Print Server to obtain a Print Server handle.

4. The Print Server returns the server handle and success code.

5. The Print Client uses RPRN to call the RpcGetPrinterData method on the Print Server, specifying the server handle and the "OSVersion" value name.

6. The Print Server returns the server version (contains the BuildNumber) and a success code.

7. The Print Client uses RPRN to call the RpcClosePrinter method on the Print Server for the server handle, and the Print Server returns a success code.

8. The Print Client verifies that the BuildNumber (OSVersion value) is 3791 or greater, indicating that the Print Server supports the PAR protocol [MS-PAR] for managing Print Queues.<16>

9. The Print Client uses RPRN to call the RpcOpenPrinter method on the Print Server for the Print Queue to obtain the Print Queue handle. This step and the next two are used to verify that the Print Server is still available.

10. The Print Server returns the Print Queue handle and a success code.

11. The Print Client uses RPRN to call the RpcClosePrinter method on the Print Server for the Print Queue handle used to complete verification that the Print Server is still available, and the Print Server returns a success code.

12. The Print Client calls the RpcAsyncOpenPrinter method on the Print Server for the specified Print Queue.

13. The Print Server returns the Print Queue handle and a success code.

14. The Print Client calls the RpcAsyncGetPrinter (*) method on Print Server for the Print Queue handle.

15. The Print Server returns a PRINTER_INFO structure with details about the designated Print Queue.

16. The Print Client calls RpcAsyncGetPrinterDriver (*) method on the Print Server for the designated Print Queue handle.

17. The Print Server returns the requested DRIVER_INFO structure, which contains details about the Print Queue's printer driver, such as driver file name and driver version.

18. The Print Client calls RpcAsyncGetPrinterDriverPackagePath (*) method on the Print Server for the Print Queue handle.

19. The Print Server returns the path for the printer driver package cabinet file and a success code.

20. The Print Client downloads the printer driver using the SMB protocol family.

21. The Print Client extracts the printer driver package contained in the downloaded cabinet file and installs the printer driver.

22. The Print Client creates a connection to the Print Queue by creating a local Print Queue proxy object.

23. The Print Client calls the RpcAsyncEnumPrinterData (*) method on the Print Server for the designated Print Queue handle.

24. The Print Server returns the requested printer data key/value pairs and a success code.

25. The Print Client calls the RpcAsyncEnumForms (*) method on the Print Server for the designated Print Queue handle.

26. The Print Server returns the requested form data and a success code.

27. The Print Client calls the RpcAsyncGetPrinterData (*) method on the Print Server for the designated Print Queue handle to obtain values of specific keys.

28. The Print Server returns the requested data and a success code.

29. The Print Client mirrors the returned data on the local Print Queue proxy object.

30. Asynchronously, on a "notification listener thread," the Print Client calls the RpcSyncRegisterForRemoteNotifications ([MS-PAR]) method on the Print Server, and the Print Server returns a notification handle and a success code.

31. The Print Client asynchronously calls the RpcAsyncGetRemoteNotifications ([MS-PAR]) method on the Print Server using the notification listener thread. When a change notification is available, the Print Server returns the call with the change notification on the same thread.

C. Submit a print job and receive notifications

The following diagram illustrates the final part of this example.

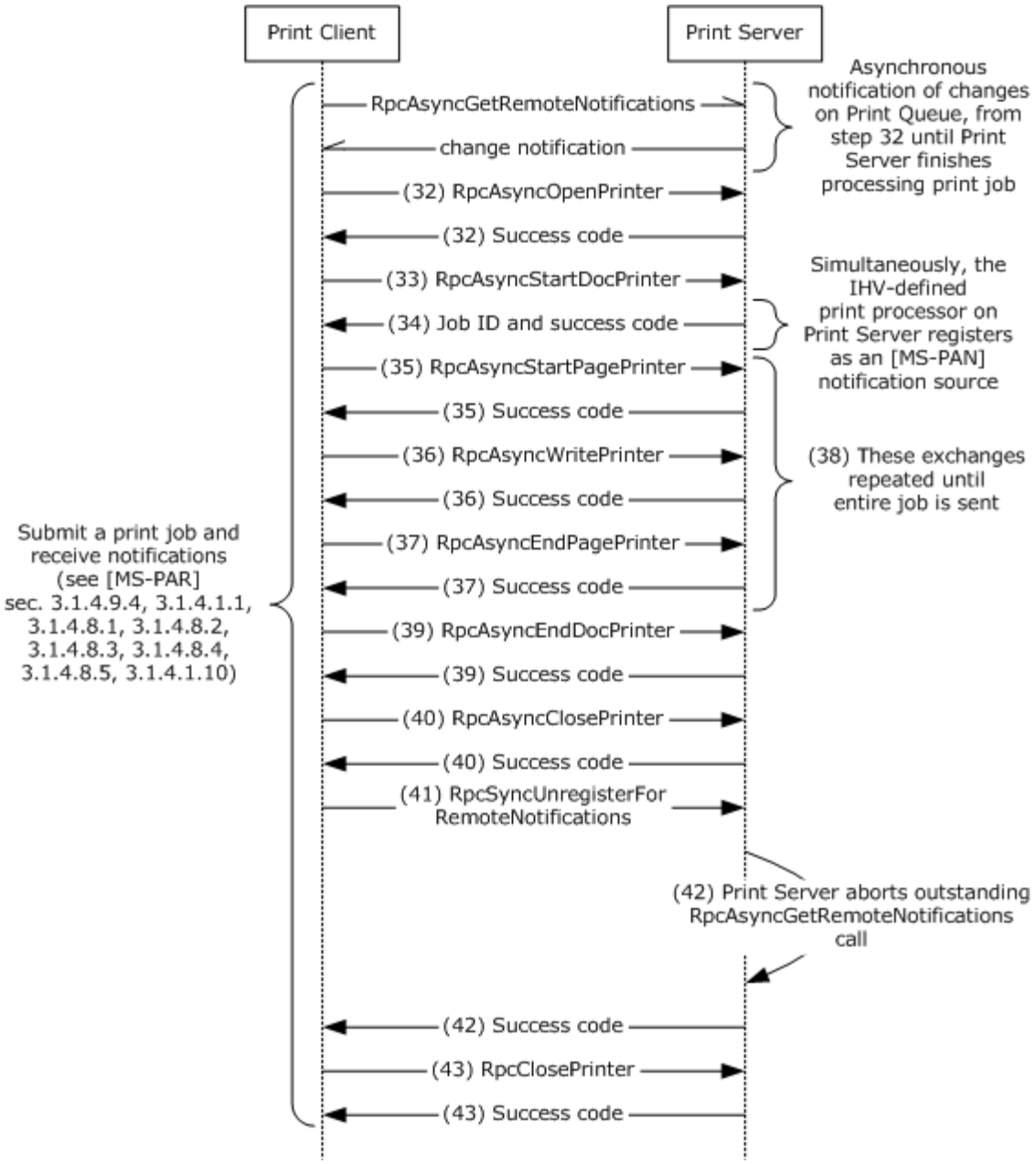

Figure 28: Print Client submitting a print job and receiving notifications

The following steps describe this sequence, which is a continuation of sequence B.

Asynchronous Communication 1: Asynchronously, from step 32 until the Print Server finishes processing the print job, the Print Server returns the RpcAsyncGetRemoteNotifications method to the Print Client to notify the Print Client of changes on the Print Queue, such as job processing progress, status changes, or other changes. The Print Client will call RpcAsyncGetRemoteNotifications again from the notification listener thread to wait for the next notification.

32. The Print Client calls the RpcAsyncOpenPrinter method on the Print Server to open a handle to the Print Queue for submitting a print job, and the Print Server returns a success code.

33. The Print Client calls the RpcAsyncStartDocPrinter method on the Print Server.

34. The Print Server returns a job ID and a success code and the IHV-defined print processor on the Print Server registers as an [MS-PAN] notification source with the Print Server through a local print spooler API.

35. The Print Client calls the RpcAsyncStartPagePrinter method on the Print Server to indicate a new page, and the Print Server responds with a success code.

36. The Print Client calls the RpcAsyncWritePrinter method on the Print Server any number of times to send data for a page, and the Print Server responds with a success code to each call.

37. The Print Client calls the RpcAsyncEndPagePrinter method on the Print Server to end the page, and the Print Server responds with a success code.

38. Steps 31-33 are repeated until all the pages of the print job have been sent.

39. When all print data has been sent, the Print Client calls the RpcAsyncEndDocPrinter method on the Print Server and the Print Server responds with a success code.

40. The Print Client calls the RpcAsyncClosePrinter method on the Print Server and the Print Server returns a success code.

41. Once a Print Client is no longer interested in notifications (for example, the user closes the Print Queue dialog user interface), the Print Client calls the RpcSyncUnregisterForRemoteNotifications method on the Print Server.

42. The Print Server aborts the outstanding RpcAsyncGetRemoteNotifications call (see Asynchronous Communication 1) and returns a success code.

43. The Print Client calls the RpcAsyncClosePrinter method on the Print Server using the handle to the Print Queue specified during registration in step 31, and the Print Server returns a success code.

    Asynchronous Communication 2: Additionally, when the Print Client's notification listener thread receives a notification announcing a new print job, the Print Client starts another listener thread to register and listen for notifications. The notification process based on the PAN protocol [MS-PAN] is illustrated in the following diagram, with further information in [MS-PAN] section 4.2.

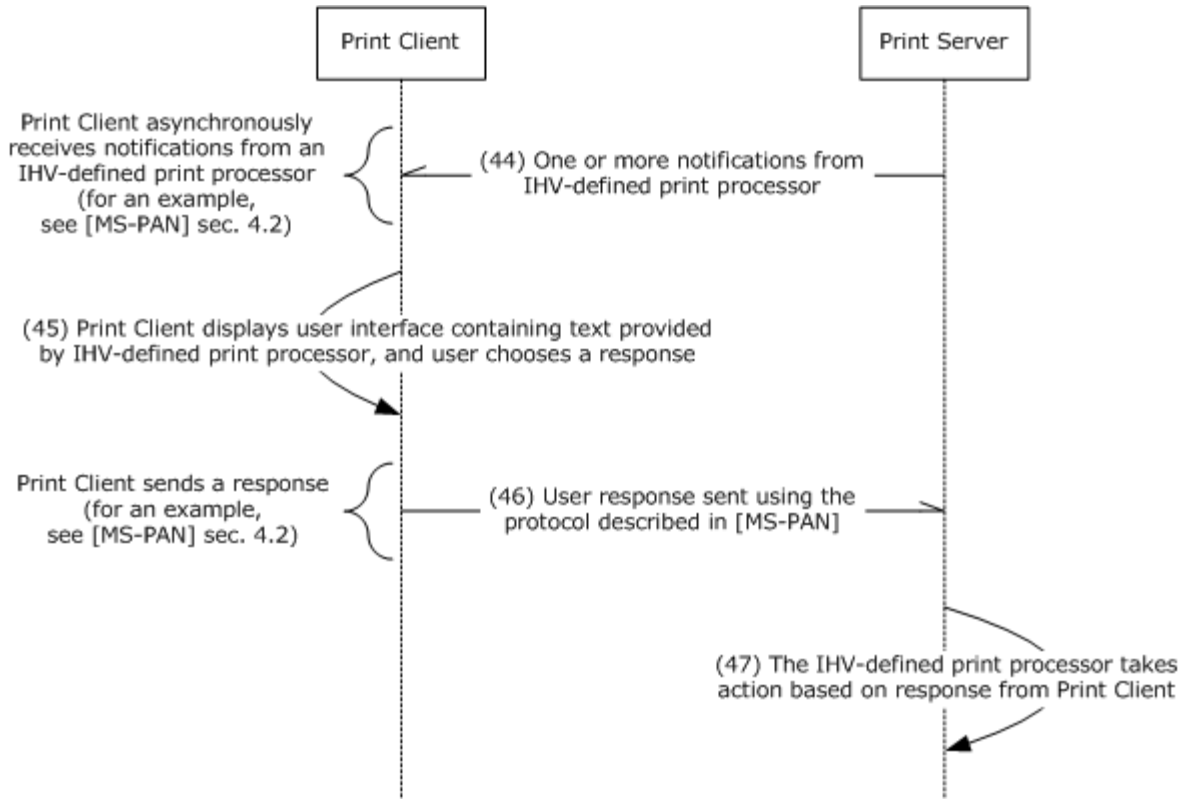

Figure 29: Print Client receiving notifications

44. During processing of the Print Job on the Print Server, the IHV-defined print processor sends one or more notifications to the Print Client using PAN. The IHV-defined print processor can send these notifications at arbitrary times.

45. When receiving the notification from the Print Server, the Print Client shows user interface to the user displaying custom text provided by the IHV-defined print processor. The user takes appropriate action and confirms the user interface.

46. The Print Client sends a response using PAN.

47. The IHV-defined print processor takes appropriate actions based on the response received from the Print Client.

It is common for Print Clients to listen for notifications from more than one thread, so the steps for registration for notifications (step 31) and unregistration (step 42) can be observed on the wire multiple times.

6.1.4 Enumerate Print Jobs from All Users, Then Cancel Several Print Jobs

This example demonstrates the use case described in section 3.3.4.8 (variation (c)).

An Administrator with appropriate privileges has the ability to view and override print jobs submitted by other users. This example illustrates this administrative procedure, using only the RPRN protocol [MS-RPRN].

Initial System State and Prerequisites

- Multiple users have submitted print jobs to the same Print Queue.
- The Administrative Client is being used by an Administrator who has privileges for overriding the print operations of other users.
- Either Print Client or Print Server support RPRN, but not the PAR protocol [MS-PAR]. This precondition was chosen to illustrate the use of RPRN.<17>

Sequence

The following diagram illustrates this example.

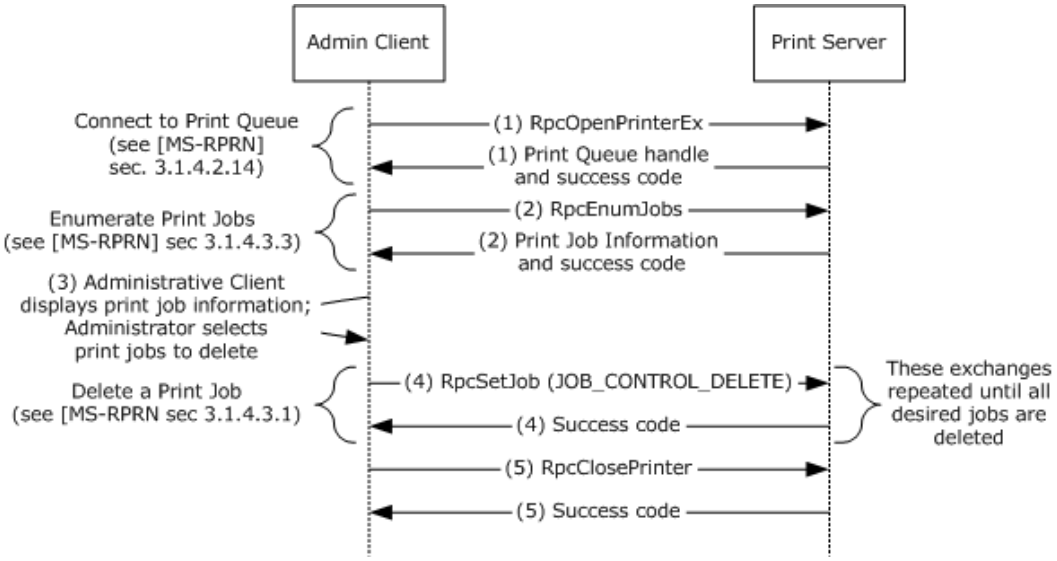

Figure 30: Administrative Client enumerating and canceling print jobs on a Print Queue

The following steps describe this sequence.

1. The Administrative Client calls the RpcOpenPrinterEx method on the Print Server to obtain the handle for the Print Queue. The Print Server returns the Print Queue handle and a success code.
2. The Administrative Client calls the RpcEnumJobs method on the Print Server using the Print Queue handle and the Print Server enumerates all the print jobs assigned to that Print Queue, and information about the print jobs (including names of the jobs, IDs of the jobs, and names of the users who submitted the jobs), and a success code.
3. The Administrative Client displays the information about the print jobs. The Administrator selects the print jobs to cancel.
4. The Administrative Client repeatedly calls the RpcSetJob method (with the JOB_CONTROL_DELETE command) on the Print Server using the Print Queue handle, canceling one job at a time. The Print Server terminates each designated print job, deleting it from the Print Queue and responding to each method call with a success code.
5. The Administrative Client calls the RpcClosePrinter on the Print Queue handle, and the Print Server responds with a success code.

Final State

The Print Queue in this example contains only print jobs that were not deleted by the Administrator.

6.1.5 Provision a Print Queue Using [MS-RPRN] from an Administrative Client, then Delete the Same Print Queue Using [MS-PAR] from a Different Administrative Client

This example demonstrates the use case described in section 3.3.4.1 (variation (a)) and section 3.3.4.2 (variation (b) with extension (a)).

In this example, two different Administrative Clients are involved in managing Print Queues: the first client provisions a Print Queue and the second client subsequently deletes the same Print Queue. The first client uses the RPRN protocol [MS-RPRN]; the second client uses the PAR protocol [MS-PAR]. Although both management functions can be accomplished by either protocol, this example demonstrates how the parallel functionality can be accomplished using different implementations or different versions of Administrative Clients.

Windows Administrative Clients using PAR to manage Print Queues will first use RPRN to ensure that PAR is supported by the Print Server and that the Print Server is available.

The sequence of steps for this example is organized into the following sections:

- A. Provision a Print Queue using RPRN
- B. Delete the same Print Queue using PAR

Initial System State and Prerequisites

- Administrative Clients and Print Servers are located within a domain configuration.
- Administrative Clients and Print Servers MUST have access to the Active Directory System provided by the domain.
- Administrator of section A has physically connected a printer to a Print Server or known network location.
- Administrator of section A knows the location from which to obtain a printer driver for the newly attached printer.
- The Administrative Client used to provision the Print Queue supports RPRN, but not PAR.<18>
- The Administrative Client used to delete the Print Queue and Print Server supports RPRN and PAR.

Sequence

A. Provision a Print Queue using RPRN

The following diagram illustrates the first part of this example.

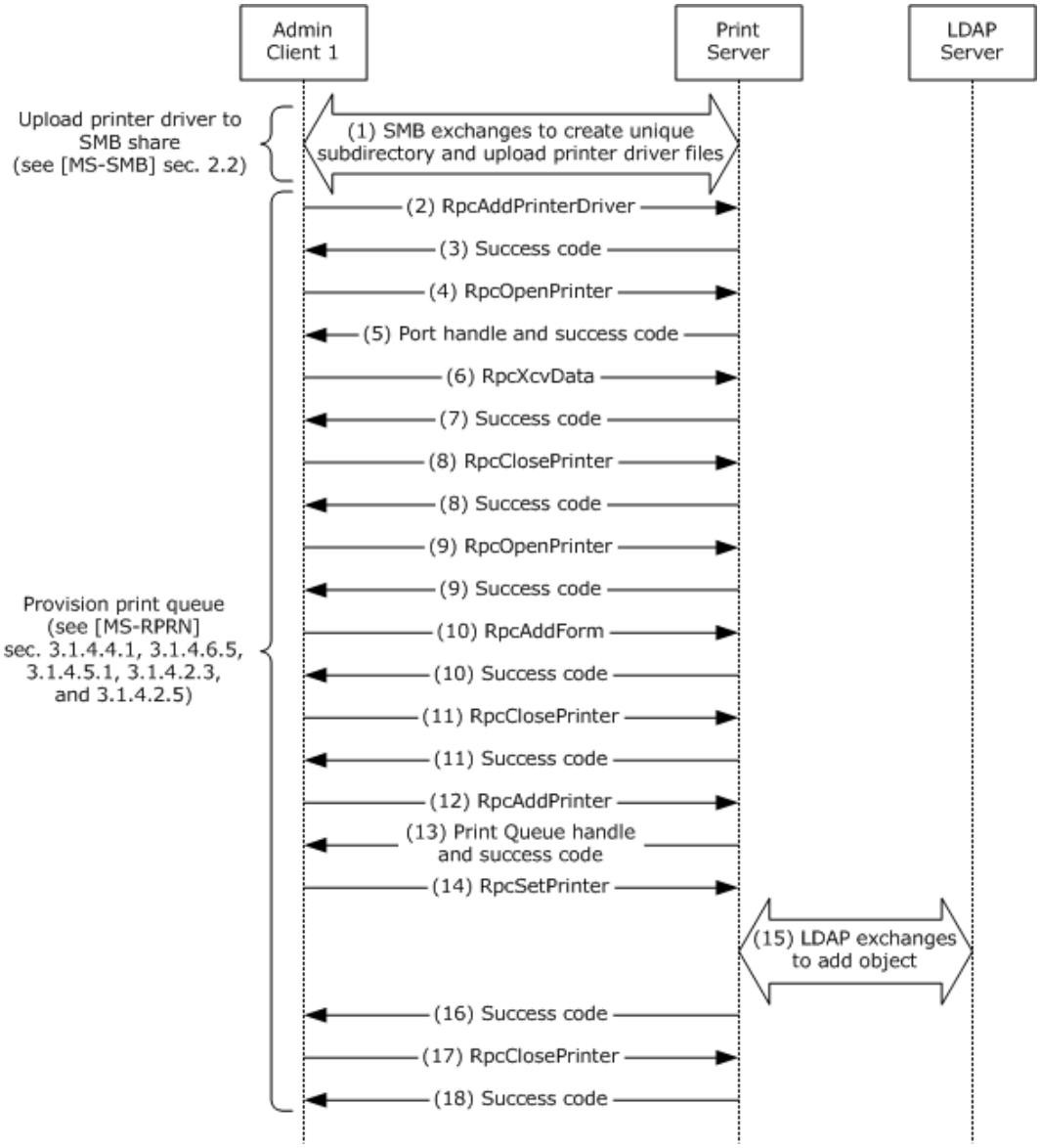

Figure 31: Administrative Client provisioning a Print Queue using RPRN

The following steps describe this sequence.

1. Administrative Client 1 uses the protocol family described in [MS-SMB] to create a unique subdirectory in the print share of the Print Server and uploads the files comprising the printer driver to that location.

2. Administrative Client 1 calls the RpcAddPrinterDriver method on the Print Server, specifying the unique directory created in Step 1.

3. The Print Server returns a success code.

4. Administrative Client 1 calls the RpcOpenPrinter method on the Print Server to open a port handle for the port monitor.

5. The Print Server returns a port handle and a success code.

6. The Administrative User initiates an "Add Port" action in the Administrative Tool and selects to add a TCP/IP port to the server. In response, the Administrative Client 1 calls the RpcXcvData method on the Print Server, specifying an "AddPort" action.

7. The Print Server adds a port and returns a success code.

8. Administrative Client 1 calls the RpcClosePrinter method on the Print Server to close the port handle, and the Print Server returns a success code.

9. Administrative Client 1 calls the RpcOpenPrinter method on the Print Server to open a server handle to the Print Server, and the Print Server returns a server handle and a success code.

10. The Administrative User initiates an "Add Form" action in the Administrative Tool and adds a form. In response, the Administrative Client 1 calls the RpcAddForm method on the Print Server to add a form. The Print Server adds the form and returns a success code.

11. Administrative Client 1 calls the RpcClosePrinter method on the Print Server to close the server handle, and the Print Server returns a success code.

12. Administrative Client 1 calls the RpcAddPrinter method on the Print Server, using the PRINTER_INFO_2 structure, the DEVMODE_CONTAINER structure, and the SECURITY_CONTAINER structure.

13. The Print Server returns a handle to the new Print Queue and a success code.

14. Administrative Client 1 calls the RpcSetPrinter method on the Print Server, using the DEVMODE_CONTAINER structure, the SECURITY_CONTAINER structure, and the PRINTER_INFO_7 structure (specifying that the Print Queue be published).

15. The Print Server uses the LDAP protocol to add the Print Queue as a new object in the Active Directory System, indicating the location of the Print Queue.

16. The Print Server returns a success code to the Print Client.

17. Administrative Client 1 calls the RpcClosePrinter method on the Print Server.

18. The Print Server returns a success code to the Administrative Client.

B. Delete the Print Queue using PAR

The following diagram illustrates the second part of this example.

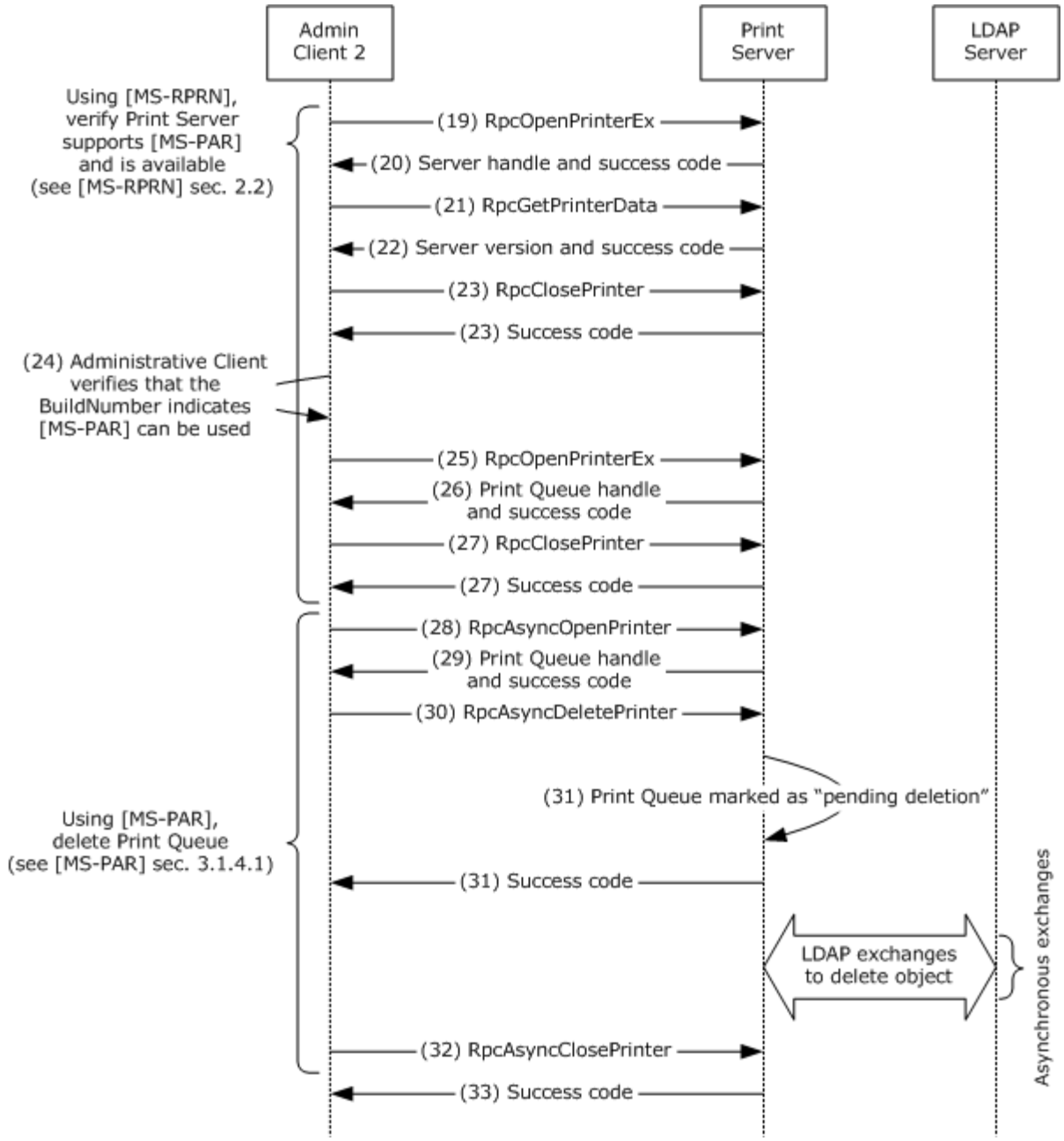

Figure 32: Administrative Client deleting a Print Queue using PAR

The following steps describe this sequence, which is a continuation of sequence A.

19. Administrative Client 2 uses RPRN to call the RpcOpenPrinterEx method on the Print Server to obtain a Print Server handle.

20. The Print Server returns the server handle and success code.

21. Administrative Client 2 uses RPRN to call the RpcGetPrinterData method on the Print Server, specifying the server handle and the "OSVersion" value name.

22. The Print Server returns the server version, which contains the BuildNumber, and a success code.

23. Administrative Client 2 uses RPRN[MS-RPRN] to call the RpcClosePrinter method on the Print Server for the server handle, and the Print Server returns a success code.

24. The Administrative Client 2 verifies that the BuildNumber (OSVersion value) is 3791 or greater, indicating that PAR can be used for managing Print Queues.

25. Administrative Client 2 uses RPRN to call the RpcOpenPrinterEx method on the Print Server for the Print Queue to obtain the Print Queue handle. This step and the next two are used to verify that the Print Server is still available.

26. The Print Server returns the Print Queue handle and a success code.

27. Administrative Client 2 uses RPRN to call the RpcClosePrinter method on the Print Server for the Print Queue handle used for verification that the Print Server is still available, and the Print Server returns a success code.

28. Administrative Client 2 uses PAR to call the RpcAsyncOpenPrinter method on the Print Server for the specified Print Queue.

29. The Print Server returns the Print Queue handle and a success code.

30. Administrative Client 2 uses PAR to call the RpcAsyncDeletePrinter method on the Print Server for the Print Queue handle.

31. The Print Server marks the specified Print Queue as "pending deletion" and returns a success code.

    Following step 31, on a separate thread of execution, the Print Server does the following:

    (a) Uses LDAP to remove the published Print Queue from the LDAP Server

    (b) Waits until the Print Queue marked "pending deletion" is no longer in use by any other Print Client or active print job, and then deletes the Print Queue from the Print Server

32. Administrative Client 2 uses PAR to call the RpcAsyncClosePrinter method on the Print Server for the Print Queue handle.

33. The Print Server returns a success code.

Final state

The Print Queue is deleted from the Print Server.

6.1.6 Send a Print Job to an SMB Share

This example demonstrates the use cases described in sections 3.3.4.7 (variation (d)) and 3.3.4.8 (variation (e)).

This example involves the SMB protocol family [MS-SMB] and the RAP protocol [MS-RAP], supporting interoperability with non-Microsoft Windows® platforms.<19> This example also applies to the rare case when a Windows Print Client installs a local Print Queue with a local port specifying the UNC path of a shared Print Queue on a Print Server.<20>

In this example, when a user performs commands from the command line the user is interacting with the SMB Server Service installed on the Print Server. A user sends a print job to an SMB share by using the command **copy /b file ***server******printer* at the command line. The Print Client subsequently uses the SMB protocol family to copy the file to the specified SMB print share of the Print Server. The SMB/RAP Print Redirector module (part of the SMB protocol family Server Service) on the Print Server translates SMB calls from the Print Client to local calls to the print subsystem of the Print Server. See section 5.4.1.2 for details of this translation.

After sending a print job to the SMB share of the Print Server, the user can obtain a list of the active jobs on the specified Print Queue by using the command **net print ***server******printer* at the command line. The Print Client uses RAP to obtain the list of active jobs. The SMB/RAP Print Redirector module on the Print Server translates the RAP calls from the Print Client to local calls to the print subsystem of the Print Server (see section 5.4.1.1 for details of this translation).

Initial System State and Prerequisites

- The requirements in the System Assumptions and Preconditions (section 4.2) are met.
- The SMB Server Service is installed and active on the Print Server.

Sequence

The following diagram illustrates this example.

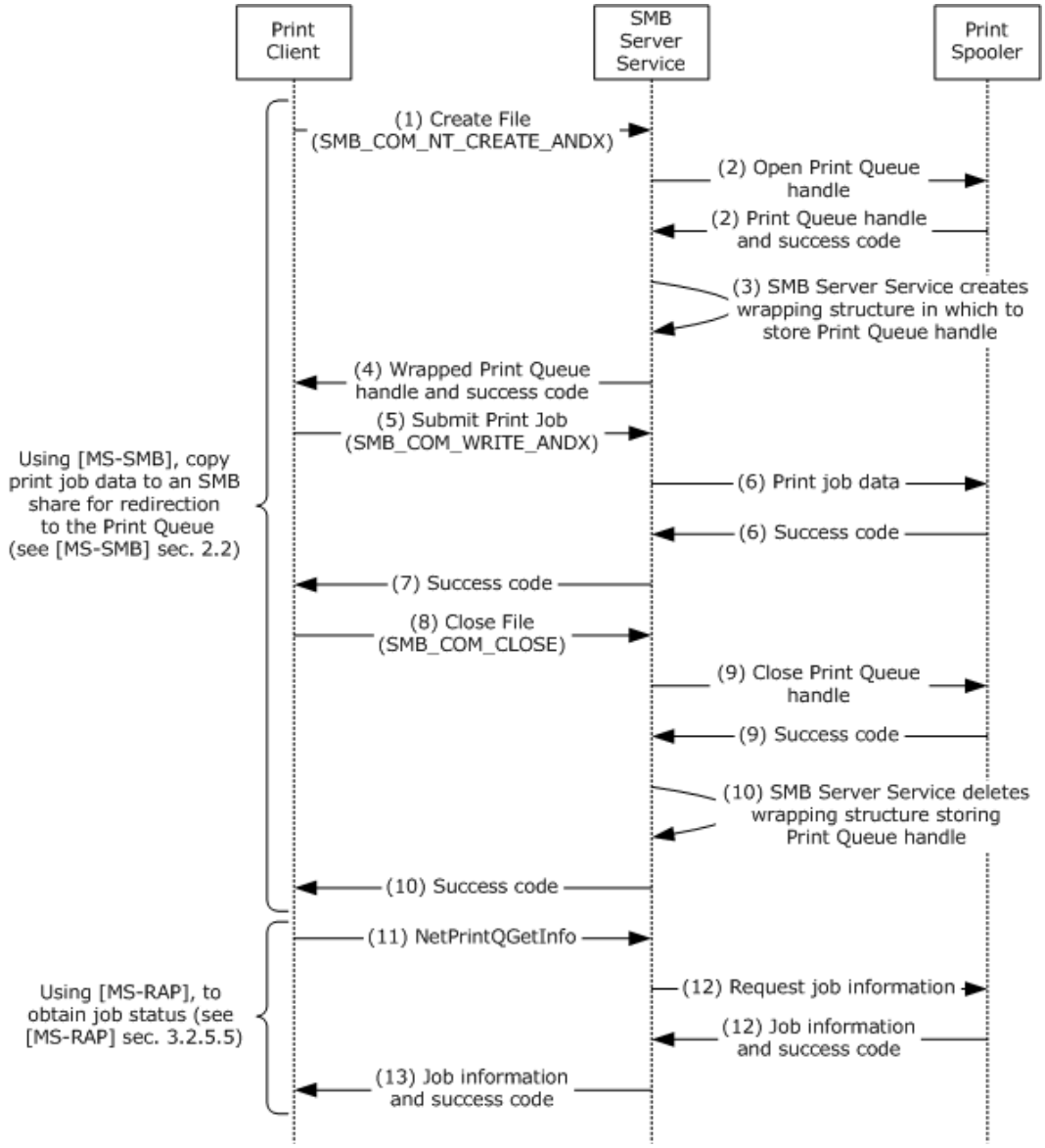

Figure 33: Print Client sending a print job to an SMB Share

The following steps describe this sequence.

1. The Print Client calls the Create File method (SMB_COM_NT_CREATE_ANDX) on the Print Server for the UNC name of the Print Queue.

2. The SMB Server Service on the Print Server redirects the call to a local API of the print spooler to open a handle to the Print Queue, and the local print spooler API returns the Print Queue handle and a success code to the SMB Server Service.

3. The SMB Server Service on the Print Server creates a wrapping structure in which to store the Print Queue handle and starts the print job by invoking a local print spooler API.

4. The SMB Server Service returns the handle of the wrapping structure and a success code to the Print Client as though it were a file handle.

5. The Print Client calls the method (SMB_COM_WRITE_ANDX) on the Print Server to write the print job data to the file handle.

6. The SMB Server Service redirects the print job data to a local print spooler API using the Print Queue handle stored in the wrapping structure, and the local print spooler API returns a success code to the SMB Server Service.

7. The SMB Server Service returns a success code to the Print Client.

8. The Print Client ends the print job by calling the Close File method (SMB_COM_CLOSE) on the SMB Server Service.

9. The SMB Server Service calls a local print spooler API to close the Print Queue handle designated in the wrapping structure, and the local print spooler API returns a success code to the SMB Server Service.

10. The SMB Server Service frees the wrapping structure and then returns a success code to the Print Client.

    To then obtain information from the SMB Server Service about print job status, the user can enter the **net print** command at the command line, to which the Print Client responds as follows:

11. The Print Client sends the NetPrintQGetInfo command defined by [MS-RAP] to send the share name of the Print Queue on the Print Server.

12. The SMB Server Service on the Print Server redirects the call to a local API of the print spooler to obtain information about the jobs queued for the specified Print Queue, and the local print spooler API returns job information and a success code.

13. The SMB Server Service responds to the NetPrintQGetInfo request with print job information and a success code.

Final State

A new print job has been added to the Print Queue on the Print Server and is being sent to the associated port.

6.2 Communication Details

The system does not define any communication constraints or additional message types beyond those described in the specifications of the protocols supported by the system, as listed in section 2.2.

6.3 Transport Requirements

The Print Services System uses transports as described in the component protocol documentation. There is no system wide transport security. Each component protocol specifies its own transport security.

6.4 Timers

There are no system level timers in the Print Services System.

6.5 Non-Timer Events

When a system shutdown event occurs:

- The Print Services System MUST close all open handles to remote objects.
- The Print Services System MUST unregister endpoints.

When a local Plug and Play event occurs:

- When new printer hardware is detected on a local port of a computer acting as Print Server, it automatically locates a matching driver for the detected printer and installs a Print Queue. The created Print Queue can optionally be set up as a shared Print Queue (per policy setting).

6.6 Initialization and Reinitialization Procedures

Initialization occurs at system startup, which will start the print spooler service.

The print spooler service SHOULD start listening on endpoint for protocols in the following order:

- At any time: Start listening to LPR.
- At any time: Start listening to WPRN and IPP simultaneously, or in the listed order.
- At any time: Start listening to PAN, PAR, and RPRN simultaneously, or in the listed order.

Reinitialization occurs up to two times only upon abnormal termination of the print spooler service.

The Print Services System starts up automatically when the server hosting the Print Services System boots. The Print Services System stays in the running state until a shutdown of the Print Server occurs, or it is manually stopped by an administrator.

The Print Services System persists elements of the ADM in the system registry.

On initialization, print services instantiate in-memory objects according to persisted information.

6.7 Status and Error Returns

Status and error returns are defined by member protocols of the Print Services System. The Print Services System as a whole does not define status codes or error codes in addition to those described in the component TDs.

7 Security

This section documents system-wide security issues that are not otherwise described in the Technical Documents (TDs) for the Member Protocols. It does not duplicate what is already in the Member Protocol TDs unless there is some unique aspect that applies to the system as a whole.

Windows Print Services allow installation of printer drivers on a Print Client from local storage media, or via copying from the Print Server for the shared printer. Windows printer drivers contain executable modules which are loaded and executed in the local system account by the print spooler. This mechanism poses a potential security threat. As a mitigation, Windows warns users before installing a printer driver which has not been signed by Microsoft, and the user can choose to abort the installation.

As another mitigation, a system administrator can restrict printer driver installation by application of Group Policy settings respected by the Print Client.

Print Services secures access to Print Queues and ports shared by a Print Server using ACLs. However, printers on the network connected via TCPMON or WSDMON (Web Services for Devices Port Monitor) do not typically offer a mechanism for access protection; therefore, Print Clients can bypass the Print Server's protection mechanism by creating a direct TCPMON or WSDMON connection to a printer. The address of the printer can be determined by scanning the local network for ports used by TCPMON or WSDMON connections. Often, it is also possible to find printers by examining DNS entries for the local network, or by printing a printer status sheet on the printer itself (the status sheet typically shows the DNS name or the TCP/IP address of the printer). Because of these limitations, in a Print Services installation with heightened access security requirements for printers it is recommended to connect the printers to a private network only accessible by the Print Server. In such a configuration the Print Server always acts as an intermediary between the Print Client and the printer, and therefore can effectively enforce access security.

An additional security concern is the actual security and accessibility of the physically printed pages resulting from a print job. The Print Services System does not offer any solution for this problem; however, multiple IHV specific solutions exist, in which the IHV provided printer driver running on the Print Client will embed a PIN only known to the user into the IHV specific command stream sent to the printer. The printer will only print or release the physical print job after the user is identified with the PIN at the printer.

Another security concern is that none of the member protocols of the Print Services System require the use of heightened transport security (RPC_C_AUTHN_LEVEL_PKT_INTEGRITY or RPC_C_AUTHN_LEVEL_PKT_PRIVACY). Windows Print Clients do not negotiate for heightened transport security. Consequently print job data and other data transmitted using member protocols is not considered secure. It can be snooped or tampered with. To achieve a secure print environment, underlying transport security mechanisms, such as IPSEC, need to be employed.

7.1 Security and Authentication Methods

Versioning of security is handled by the underlying RPC transport; see [MS-RPCE] section 3.3.3.3 for more information.

Policy settings controlling local print spooler behavior can be pushed using the GPOL protocol [MS-GPOL] from the directory server to Print Clients and Print Servers.<21> This functionality allows the following:

- Driver installation security settings
- Sandbox grouping settings

7.2 Securable Objects

The securable objects in the Print Services System are the print job object, Print Queue object, and Print Server object. Security for these objects is discussed in [MS-RPRN], [MS-PAR], and [MS-PAN].

Print services apply ACLs to elements of the component TDs' ADM. In a typical managed network, default ACLs allow users the Use access right for all elements visible through the component TD protocols, and allow administrators unrestricted access for all elements visible through the component TD protocols.

7.3 Internal Security

The Print Services System applies ACLs to elements in its ADM. When processing calls, the system uses the user's token to verify access to a secured element.

7.4 External Security

The Print Services System impersonates the user when processing calls. The permissions in the user's token determine the type of access this system has to external resources while processing calls.

Special care must be taken when implementing a Print Client or Print Server for print services whenever printer drivers or other plug-ins are referenced on a remote system, or copied from a remote system. Printer drivers or other plug-ins can contain executable code, and therefore SHOULD only be executed in the context of the local caller if they are trusted.

The Windows print spooler will call printer drivers or other plug-ins as local system calls and impersonate the calling user. As a consequence, a system implementer must exercise utmost care to protect the system from harm by untrusted printer drivers.

The Windows implementation performs none or more of the following:

- Restrict non-administrative users from installing printer drivers.
- Check the digital signatures of drivers.
- Prompt the user for consent before downloading such components or executing the component for the first time.

8 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft Windows NT® 3.1 operating system
- Microsoft Windows NT® 3.5 operating system
- Microsoft Windows NT® 3.51 operating system
- Microsoft Windows NT® 4.0 operating system
- Microsoft Windows® 98 operating system
- Microsoft Windows® 2000 operating system
- Microsoft Windows® 2000 Server operating system
- Microsoft Windows® Millennium Edition operating system
- Windows® XP operating system
- Windows Server® 2003 operating system
- Windows Server® 2003 R2 operating system
- Windows Vista® operating system
- Windows Server® 2008 operating system
- Windows® 7 operating system
- Windows Server® 2008 R2 operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 3.1.1.1.1: Print Server support for the protocol specified in [MS-PAR] can be disabled using a local Windows registry setting.

<2> Section 3.3.4.8: The net print command is no longer present in Windows 7.

<3> Section 3.3.4.8: The net print command is no longer present in Windows 7.

<4> Section 4.1: The maximum number of shared Print Queues allowed in a Windows workgroup is 256.

<5> Section 4.3.3: Group Policy settings can be applied to Print Clients and Print Servers running Windows 2000, Windows 2000 Server, Windows XP, Windows Server 2003, Windows Server 2003 R2, Windows Vista, Windows Server 2008, Windows 7, Windows Server 2008 R2.

<6> Section 4.5: The Print Services System versions described in section 4.5 are supported in Windows releases as noted in the following table.

Print Services SystemVersion	Print Server Release	Print Client Release	Method used by Client to Determine Support Level
PSS 1.0	Windows NT 3.1 Windows NT 3.5 Windows NT 3.51 Windows NT 4.0 Windows 2000 Windows 2000 Server Windows XP Windows Server 2003 Windows Server 2003 R2 Windows Server 2008 Windows Server 2008 R2 Windows Vista Windows 7	Windows 98 Windows Millennium Edition Windows NT 3.1 Windows NT 3.5 Windows NT 3.51 Windows NT 4.0 Windows 2000 Windows 2000 Server Windows XP Windows Server 2003 Windows Server 2003 R2 Windows Server 2008 Windows Server 2008 R2 Windows Vista Windows 7	The protocol described in [MS-RPRN] is not supported. The client detects this if the RPC binding to the endpoint for the protocol described in [MS-RPRN] fails.
PSS 2.0	Windows NT 3.1 Windows NT 3.5 Windows NT 3.51 Windows NT 4.0 Windows 2000 Windows 2000 Server Windows XP Windows Server 2003 Windows Server 2003 R2 Windows Server 2008 Windows Server 2008 R2 Windows Vista Windows 7	Windows NT 3.1 Windows NT 3.5 Windows NT 3.51 Windows NT 4.0 Windows 2000 Windows 2000 Server Windows XP Windows Server 2003 Windows Server 2003 R2 Windows Server 2008 Windows Server 2008 R2 Windows Vista Windows 7	The protocol described in [MS-RPRN] is supported except for the RpcOpenPrinterEx method.
PSS 2.0 Additional capabilities in [MS-RPRN]	Windows NT 4.0 Windows 2000 Windows 2000	Windows NT 4.0 Windows 2000 Windows 2000	The protocol described in [MS-RPRN] is supported including the RpcOpenPrinterEx method. AND

Print Services SystemVersion	Print Server Release	Print Client Release	Method used by Client to Determine Support Level
	Server Windows XP Windows Server 2003 Windows Server 2003 R2 Windows Server 2008 Windows Server 2008 R2 Windows Vista Windows 7	Server Windows XP Windows Server 2003 Windows Server 2003 R2 Windows Server 2008 Windows Server 2008 R2 Windows Vista Windows 7	OSVERSIONINFO.BuildNumber is less than or equal 1381.
PSS 3.0	Windows 2000 Windows 2000 Server Windows XP Windows Server 2003 Windows Server 2003 R2 Windows Server 2008 Windows Server 2008 R2 Windows Vista Windows 7	Windows 2000 Windows 2000 Server Windows XP Windows Server 2003 Windows Server 2003 R2 Windows Server 2008 Windows Server 2008 R2 Windows Vista Windows 7	OSVERSIONINFO.BuildNumber is greater than 1381 but less than or equal 3790.
PSS 4.0	Windows Server 2008 Windows Server 2008 R2	Windows Vista Windows 7	OSVERSIONINFO.BuildNumber is greater than 3790.

<7> Section 5.1.1.2: Windows implements port pooling. A printer object can manage references to multiple port objects. A physical print device is connected to each of the port objects, but the physical print devices are substantially the same. Windows transparently distributes incoming jobs to the multiple port objects to balance the workload.

<8> Section 5.1.1.2.1: Windows Vista, Windows Server 2008, Windows 7, and Windows Server 2008 R2 implement such caching Print Queue Proxy objects that enable client applications to print to and interact with Print Queues even if no network connection to the Print Server can be established.

<9> Section 5.1.1.7: Windows ships with a language monitor supporting the HP PJL Job Control language. (PJL supports remote configuration and the control of print job processing.)

<10> Section 5.5.1: Only Windows 7 and Windows Server 2008 R2 support this configuration.

<11> Section 6.1.1: This precondition requires that the print client is implemented on Windows 7, the print server is implemented on Windows 7 or Windows Server 2008 R2, and the print client does

not negotiate to use the PAR protocol [MS-PAR], which can be enforced by a registry setting on the print server that disables the server endpoint for PAR.

<12> Section 6.1.1: The printer driver directory is not needed by the Windows print client to complete this scenario; the print client merely preemptively queries this piece of print server configuration data.

<13> Section 6.1.2: This precondition applies when the print client and print server are implemented on any Windows version except Windows 95, Windows 98, and Windows Millennium Edition. If the print server is implemented on Windows Vista, Windows Server 2008, Windows 7, or Windows Server 2008 R2, a registry setting has to be used to disable the PAR protocol [MS-PAR].

<14> Section 6.1.2: The printer driver directory is not needed by the Windows print client to complete this scenario; the print client merely preemptively queries this piece of print server configuration data.

<15> Section 6.1.3: The example described in this section applies when the Print Client is implemented on Windows Vista or Windows 7, and the Print Server is implemented on Windows Server 2008 or Windows Server 2008 R2.

<16> Section 6.1.3: This applies if the Print Server runs on Windows Vista, Windows Server 2008, Windows 7, or Windows Server 2008 R2.

<17> Section 6.1.4: This precondition applies when the Print Client and Print Server are both implemented on any Windows version except Windows 95, Windows 98, and Windows Millennium Edition. If the Print Server is implemented on Windows Vista, Windows Server 2008, Windows 7, or Windows Server 2008 R2, a registry setting is used to disable PAR.

<18> Section 6.1.5: This precondition applies when the Administrative Client and Print Server are both implemented on any Windows version except Windows 95, Windows 98, and Windows Millennium Edition. If the Print Server is implemented on Windows Vista, Windows Server 2008, Windows 7, or Windows Server 2008 R2, a registry setting is used to disable PAR.

<19> Section 6.1.6: The example also applies to Print Clients running on Windows 95, Windows 98, and Windows Millennium Edition, as well as when a user sends a print job to a Print Queue by using the command line to copy a file to a printer share.

<20> Section 6.1.6: Windows 7 Print Clients and Windows Server 2008 R2 Print Servers no longer support the "copy /b" or "net print" command line functionality.

<21> Section 7.1: This functionality requires Windows 2000, Windows XP, Windows Server 2003, Windows Server 2003 R2, Windows Vista, Windows Server 2008, Windows 7, or Windows Server 2008 R2.

9 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

10 Index

T

U

V

W