HP Client Automation

Batch Publisher

for Linux® and Windows® operating systems

Software Version: 7.50

Installation and Configuration Guide

Manufacturing Part Number: none

Software Release Date: May 2009

Document Release Date: May 2009

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notices

Trademark Notices

Linux is a registered trademark of Linus Torvalds.

Microsoft®, Windows®, and Windows® XP are U.S. registered trademarks of Microsoft Corporation.

OpenLDAP is a registered trademark of the OpenLDAP Foundation.

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
  - The number before the period identifies the major release number.
  - The first number after the period identifies the minor release number.
  - The second number after the period represents the minor-minor release number.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition, visit the following URL:

http://h20230.www2.hp.com/selfsolve/manuals

This site requires that you register for an HP Passport and sign in. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

Or click the **New users - please register** link on the HP Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Table 1 indicates changes made to this document.

Table 1 Document Changes

Chapter	Version	Changes
All	7.50	Changed version number to 7.50.
All	7.50	The Batch Publisher was rebranded from Configuration Management to Client Automation.
All	7.50	Removed support for all Unix-based platforms except Linux.
All	7.50	Removed all information about CM legacy Source Control Management adapters, including PVCS and ClearCase.
All	7.50	Removed all information about object-based publishing.
Chapter 4	7.50	Removed the Object-Based Publishing (SCMAdapt.tkd) chapter.

Chapter	Version	Changes
Chapter 4	7.50	Removed support for HP-UX, Solaris, and AIX throughout this chapter.
Chapter 4	5.00	Page 50, HPCA Native Packaging Command-Line Interface: Added –depth and -dist options.
Chapter 4	5.10	Page 51, Table 6, Command-line parameters: Added HP-UX SD packages and SD bundles to –dist command line parameter. Added HP-UX SD packages to list of supported formats for dependency checking (–depth). Note: This was removed for 7.50.
Chapter 4	5.00	Page 60, Publishing with Interactive Mode: Updated to include new continue option behavior.
Chapter 4	5.00	Page 66, Operational Notes: Included notes for new command line options and multi-level dependency support.
Chapter 4	5.00	Page 65, Automatic Inclusion of Required Packages: Updated information for default behavior and RPM packages.
Chapter 4	5.00	Page 68, NOCHECK information added to Operational Notes and class instance tables.
Chapter 4	7.50	Table 12, Table, SD Class instance attributes modified: Removed this entire table.
Chapter 4	3.1	Table 13, SVR4 Class instance attributes modified by CM Native Packaging: Added new SVR4 class instance attributes: ADMIN, AMDINOBJ, CONTENTS, PKGVER, PKGREV.
Chapter 4	7.50	Table 13, SVR4 Class instance attributes modified by CM Native Packaging: removed this entire table.
Appendix	7.50	Appendix A, Product Name Changes: Removed this appendix.

Support

You can visit the HP Software support web site at:

www.hp.com/go/hpsoftwaresupport

This Web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online software support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract.

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

Contents

1 Introduction

At the end of this chapter, you will:

- Be familiar with the HPCA Batch Publisher.
- Understand the different publishing modes available with the HPCA Batch Publisher.
- Understand the HPCA Batch Publisher system requirements.

What is the HPCA Batch Publisher?

The HPCA Batch Publisher is a command-line-driven content publishing tool that identifies a set of files and components (and their relationships) and publishes them in a controlled, automated, repeatable manner, to the Configuration Server Database (CSDB), where they are stored as objects. The HPCA Batch Publisher can:

- scan for files on multiple drives or file systems,
- scan and publish files from any mapped drive or file system,
- be configured to limit the subdirectories that are scanned,
- include or exclude at the file level, and
- select files by type.

Also, the HPCA Batch Publisher can accommodate frequent patching of internal applications. Its capacity to revise content material is reliable, and can be designed to perform continuously, at designated times, and in pre-determined intervals, and can be easily executed from within any script or code capable of calling a command prompt.

Why Use the HPCA Batch Publisher?

The HPCA Batch Publisher offers a means of reliable and instant data updates to information that must be posted in an automated fashion.

The primary function of the HPCA Batch Publisher is to distribute updates to content, data, and applications rather than the initial application packaging. Typically, these types of data updates require a repeatable process. Digital content, such as file sets, graphics, price lists, and interest rates, are types of managed lists that might require an automated update process that the HPCA Batch Publisher can provide.

Since the HPCA Batch Publisher is a repeatable process, it dynamically creates package instances and names them (with date and sequence number)

to accommodate multiple publishing sessions. The user can select from two input modes: files and a configuration file. An HPCA agent is not required.

The HPCA Batch Publisher vs. Standard HPCA Publishing

The Batch Publisher provides a command-line alternative to the Component Selection Mode of the graphical user interface of the Administrator Publisher. The Batch Publisher is an automated, repeatable command-line process, whereas the HPCA Admin Publisher must be monitored from start to finish. For more information on the HPCA Admin Publisher tool, refer to the *HP Client Automation Administrator User Guide* or the *HP Client Automation Application Manager and Application Self-service Manager Installation and Configuration Guide.*

Overview

The Batch Publisher default operation creates standard instances of the PACKAGE, FILE, PATH, DESKTOP, and REGISTRY Classes in the SOFTWARE Domain of the HPCA-CSDB. Three additional features of the HPCA Batch Publisher are the ability to:

- publish into other classes, as well as a different domain.
- optionally create (and update, as needed) a ZSERVICE Class instance connection to a published package.
- automatically generate the path information that is required for the distribution of a package. The path information is generated dynamically by a combination of configuration options and the location of the files being published.

You can run the HPCA Batch Publisher in the following way:

- Specify in the configuration file the targeted files to be published.

Table 1 shows how to apply each of these methods.

Table 1 HPCA Batch Publisher method applications

method	promote.tkd (configuration file-based publishing)
scan	intype=SCAN
file	intype=FILE (files specified in the insource file)
filtering	Available

Publishing Modes

Configuration File-Based Publishing

Configuration file-based publishing allows for multiple publishing modes that are dictated by the information contained in a configuration file. Multiple configuration files can be maintained and used for different publishing jobs, providing there is an administrator with the ability to repeat a publishing session as needed.

Use the HPCA Batch Publisher to publish files to the HPCA-CSDB with either of two methods: scanning a directory or publishing files listed in an input file.

- The scanning method enables you to scan one or more directories. This method also lets you specify:
  - the depth of the scan (that is, the number of subdirectories),
  - filters as selection criteria, and
  - criteria for the inclusion/exclusion of files.
- The files listed method is more efficient if you want to publish a set of files. You can also identify and target files to be published to specific classes of the HPCA-CSDB. For example, you can designate files with the "lnk" extension to be published to the DESKTOP Class on the HPCA-CSDB.

In configuration file-based publishing, when a name is designated in the service option and `addtosvc=1`, a new connection is made to the service. If the service does not exist, it is created and the connection is made. In either case, this connection will occupy the first available CONNECT_TO field. When a name for a package is specified with an asterisk (*), the package name is sequentially generated (prefixYYYYMMDD#) with the same prefix (*prefix**). Multiple packages with the same name (identical prefix*) are linked to one another as REQUIRES connections within the service. The first package promoted is linked directly (as an INCLUDES connection) to the service in the first available CONNECT_TO field. See the following example.

```
SERVICE ---> INCLUDES connection ---> PCKG01
```

Packages (with the same prefix) that are promoted subsequently override the previous package, and assume the direct link to the service, forcing that previous package to adopt a REQUIRES link to it. And so it continues, with each new same-named package breaking its predecessor's INCLUDES connection to the service, and "demoting" that previous package to a REQUIRES link to itself. See the following example.

```
SERVICE--->INCLUDES--->PCKG03
                       |
                       |--->REQUIRES conn--->PCKG02
                                             |
                                             |-->REQUIRES-->PCKG01
```

The prefix used to create a sequentially generated service name must be a unique name and cannot match any existing service names. For example, if the service name SAMPLE exists, the prefix SAMPLE* cannot be used to create sequentially generated service names using the `addtosvc` parameter.

Only in this scenario are the packages connected to the service as REQUIRES, with the second package requiring the first, the third package requiring the second, and so on.

Multiple packages with different names are linked to the service independently at subsequent available connects. Each of these packages will be added in the order in which it is received by the HPCA Configuration Server, and placed in the first available CONNECT_TO field.

The HPCA Batch Publisher performs a CRC (cyclical redundancy check) on the fully qualified path, not just the file name. In order for the file to be recognized as a duplicate, it must consistently be promoted from the same location. The HPCA Batch Publisher does not delete connections, except in the case of multiple promotes having an identical prefix*, nor does it remove REQUIRES links.

Native Packaging

HPCA Native Packaging is a feature of the HPCA Batch Publisher specifically designed to publish Linux native software packages. Native Packaging is installed with the Batch Publisher on Linux systems. See Chapter 4, HPCA Native Packaging for more information.

Summary

- The HPCA Batch Publisher is a command-line-driven content publishing tool.
- The HPCA Batch Publisher offers two publishing modes: Configuration File-Based and HPCA Native Packaging.
- The HPCA Batch Publisher requires connectivity to a HPCA-CSDB.

2 Installing the HPCA Batch Publisher

At the end of this chapter, you will:

- Know how to install the HPCA Batch Publisher

System Requirements

The HPCA Batch Publisher is available for Windows and Linux operating systems. It has these system requirements:

- Network connectivity to the HPCA Configuration Server.
- A minimum of 2 MB of hard disk space.
- Access to any directories from which you want to publish.

Operating System Considerations

Windows Platforms

Registry files that are published into the REGISTRY class need to be converted from the REGEDIT4 registry export format to the HPCA EDR format required by the HPCA agent. The HPCA Batch Publisher will perform this conversion automatically, unless the file has an EDR extension. In this case, `promote.tkd` assumes that the file has already been converted to the EDR format.

The HPCA Batch Publisher will *not* convert files from the REGEDIT5 registry export format.

Linux Platforms

Before using the HPCA Batch Publisher in a Linux environment, you need to modify the `filters all` parameter in the configuration file. This is specific to the configuration file-based publishing method (`promote.cfg`).

As you can see below, the default values are:

```
filters all {

type            file
class           file
exclude   "*.log *.bak"
```

```
include   "*"
distroot  {}

}
```

You will need to change the `class` parameter from its default of `file` to `unixfile`.

```
filters all {

type             file
class            unixfile
exclude   "*.log *.bak"
include   "*"
distroot  {}

}
```

Make sure that the new class, UNIXFILE, is included in the HPCA-CSDB. If your HPCA Configuration Server is version 4.3 or earlier, contact HP Support in order to get the class definition.

The exclude, include, and distroot parameters should be set to the values appropriate to the user's requirements.

Platform Support

For detailed information about supported platforms, see the release note document that accompanies this release.

Recommendations

Stop any programs that are running before installing the HPCA Batch Publisher.

Installing the HPCA Batch Publisher for Windows

To install the HPCA Batch Publisher for Windows

1. From the installation media, `/management_extensions/publishing_adapter/publisher/win32` folder, double-click **setup.exe**. The Welcome window opens.
2. Click **Next**. The HP Software License Terms window opens.
3. Read the license terms and click **Accept**. The Directory Location window opens.
4. Type the name of the directory where you would like to install the HPCA Batch Publisher (default is `C:\Program Files\Hewlett-Packard\CM\BatchPublisher`), or click **Browse** to navigate to it.
5. Click **Next**. If the directory you specified already exists, you are prompted to replace it.
6. Click **OK**. The license file window opens.
7. Enter the location of your license file, or click **Browse** to navigate to it.
8. Click **Next**. The Installation Settings window opens.
9. Click **Install**.
10. When the installation is complete, click **Finish**.

You have successfully installed the Batch Publisher for Windows.

Installing the HPCA Batch Publisher for Linux

If you are installing the HPCA Batch Publisher on a Linux system that supports graphics, the graphical installation will automatically begin after you run the installation program. For Linux systems that support graphics,

see Linux Graphical Installation below. For Linux systems that do not support graphics, the non-graphical installation program is automatically started. See Linux Non-Graphical Installation on page 24.

If you are installing the HPCA Batch Publisher onto a Linux system that supports graphics, but you would like to use the non-graphical mode instead, change your directory to the location of the install program on the installation media and type:

```
./install –mode text
```

This will start the non-graphical installation of the HPCA Batch Publisher. See Linux Non-Graphical Installation on page 24 for instructions.

Linux Graphical Installation

This section guides you through the graphical installation of the HPCA Batch Publisher.

To install the HPCA Batch Publisher using the graphical interface

1. From the installation media, `/management_extensions/publishing_adapter/publisher/linux` folder
2. Type **`./install`**, and then press **Enter**. The Welcome window opens.
3. Click **Next**. The HP Software License Terms window opens.
4. Read the agreement and click **Accept**. The Directory Location window opens.
5. Type the name of the directory to which you would like to install the Batch Publisher (default is `/opt/HP/CM/BatchPublisher`), or click **Browse** to select a location.
6. Click **Next**.
7. If the directory you specified already exists, you are prompted to overwrite the existing directory. To specify a new directory, click **Cancel** to return to the previous step, or click **OK** to proceed. The license file window opens.

8 Enter the location of your license file or click **Browse** to select the location manually.

9 Click **Next**. The Installation Settings window opens.

10 Click **Install**.

11 When the installation is complete, click **Finish**.

You have successfully installed the Batch Publisher for Linux.

Linux Non-Graphical Installation

This section guides you through the non-graphical installation of the HPCA Batch Publisher for Linux.

To install the HPCA Batch Publisher using the non-graphical installation

1 From the installation media, `/management_extensions/publishing_adapter/publisher/linux` folder

2 Type **`./install -mode text`** and then press **Enter**. The HPCA Batch Publisher installation begins.

3 Type **`C`**, and then press **Enter**.

4 Press a key to view the End User License Agreement.

5 When you are finished viewing the agreement, type **`Accept`** and press **Enter**.

6 Accept the default location for the HPCA Batch Publisher (`/opt/HP/CM/BatchPublisher`) by pressing **Enter**, or specify a different location.

If the directory you specify already exists, you will be prompted to continue. If the directory does not exist, the installation program will display the Installation Settings.

7 Type **`Y`**, and then press **Enter**.

8 Enter the location and name of your license file and press **Enter**.

9 Press **Enter** to accept the default (Y) and begin the installation. If you do not want to begin the installation, type **N**, and then press **Enter**.

10 To complete the configured installation process, press **Enter**.

You have successfully installed the HPCA Batch Publisher for Linux.

Summary

- The HPCA Batch Publisher is available for Windows and Linux operating systems.
- Before installing the HPCA Batch Publisher, we recommend that you stop any running programs.

3 Configuration File-Based Publishing (promote.tkd)

At the end of this chapter, you will:

- Be familiar with configuration file-based publishing.
- Understand the command-line parameters needed for `promote.tkd`.
- Understand the `promote.cfg` parameters.
- Understand how to specify additional attributes.

Using Configuration File-Based Publishing

Configuration file-based publishing uses a configuration file (`promote.cfg`) that contains your publishing specifications. The publishing session is then executed from the command line. Command-line parameters are described in Table 2 below, and the configuration file is described in The PROMOTE Configuration File on page 29.

Execute the command line from the directory where you installed the HPCA Batch Publisher (default is `C:\Program Files\Hewlett-Packard\CM\BatchPublisher\`). The command line is: `nvdkit promote.tkd`. Additional parameters can be added to modify this command line. These parameters are described in Table 2 below. All files needed were installed during the HPCA Batch Publisher installation including the HP runtime Tcl interpreter and configuration file-based publishing code.

Example

```
nvdkit promote.tkd -cfg promote.cfg -user rad_mast -pass
secret
```

Table 2 Command-line parameters for promote.tkd

Parameter	Description
`-cfg` *`filename`*	Specifies the file that contains the configuration options for this execution of the HPCA Batch Publisher. The file `sample.cfg` is provided as a sample configuration file, and can be used to model the promote.cfg. The configuration file can be custom named. You can maintain multiple configuration files to facilitate a variety of publishing jobs. This parameter is optional. If no configuration file is specified, `promote.cfg` in the current working directory is used.
`-user` *`userid`*	HPCA administrator user ID. The default is RAD_MAST. This parameter is optional.
`-pass` *`password`*	HPCA administrator password. This parameter is optional.

Parameter	Description
`-phase input`	If present and the value is `input` (not case-sensitive), the database will be created, but the files will not be published. This is useful for testing filters, debugging, and verifying that your selected criteria are producing the expected results (the results are sent to the log and displayed on the screen). This parameter is optional. Note: Any value other than input will be ignored.

The PROMOTE Configuration File

Table 3 below describes the configuration file parameters.

The PROMOTE Configuration File Format

Table 3 PROMOTE configuration file format (`promote.cfg`)

Option	Description
`Package`	Defines the PACKAGE Class instance name or prefix. If specified without a trailing asterisk (*), the value is used as the absolute PACKAGE Class instance name. If specified with a trailing asterisk (*), the value is used as a prefix to dynamically generate the PACKAGE Class instance name. When used as a prefix, the PACKAGE Class instance name is generated as: `<pkgprfx>`*`YYYYMMDDs`* where *`YYYYMMDD`* is the current date, and `s` is a sequence number used to guarantee uniqueness.
`pkgname`	Specifies the friendly name of the PACKAGE Class instance (NAME).
`pkgdesc`	Specifies a description of the PACKAGE Class instance (ZPKGDESC) attribute on the package that gets populated.

Option	Description
service	Defines the name of the ZSERVICE Class instance that will be optionally created (or updated) in the HPCA-CSDB during the publishing session. The publishing session will create a ZSERVICE Class instance if one does not exist. Note: This option will work only if addtosvc=1.
svcname	Specifies the friendly name of the ZSERVICE Class instance (NAME). This command is optional.
svcdesc	Specifies a description of the ZSERVICE Class instance (ZSVCNAME) attribute on the service that gets populated. This command is optional.
addtosvc	Tells the HPCA Batch Publisher whether to update a ZSERVICE Class instance with a connection to the newly published package. 1 = Add connection to ZSERVICE. 0 = Do not add connection to ZSERVICE. Note: If set to 1, the service command must have a value specified.
compress	Tells the HPCA Batch Publisher whether to use compression. 1 = Use compression. 0 = Do not use compression.
intype	Defines the type of the input source. Values are FILE and SCAN. FILE - Use when the list of files to be published is contained in a file. Note: The insource option must be used if intype=FILE. SCAN - Use when the list of files to be published is to be scanned on a drive/file system. Note: The filescan option must be used if intype=SCAN.

Option	Description
insource	Specifies the name of the source file. The specified file should contain a list of qualified file names, one per line, to be published. Also, numsplit and distroot can be specified in the file. These options behave in the same manner as described in the filescan row of this table. Note: Relevant only when intype=FILE. The formats that are accepted for the lines in the file are: • global distroot <value> - Specifies the distroot value to be used for the files listed on the lines that follow it. If not specified, the original location of the file will be used as the distribution directory. • global numsplit <value> - Specifies the numsplit position to be used for the files listed on the lines that follow it. The default value is 1. • <filename> - Specifies the fully qualified name of a file to be published. Notes: Filters will still be applied to the files before publishing. If a file does not match any filters, it will not be published. The commands global distroot and global numsplit can be specified at any point in the insource file. Their values affect only the lines that follow them, and remain in effect until the next global command is encountered. Therefore, group together files by their common distroot and numsplit values. In the examples below, note the values of numsplit (3 and 2) and distroot (d:/myapps and d:/place). The resulting outputs are also shown.

Example A	Example B
global numsplit 3 global distroot d:/myapps d:/temp/src/apps/a.dat d:/temp/src/apps/test2.tcl	global numsplit 2 global distroot d:/place d:/temp/list.pdf d:/temp/mymk.tcl
Output: (distroot) | (stem) d:/myapps/apps/a.dat d:/myapps/apps/test2.tcl	Output: (distroot) | (stem) d:/place/list.pdf d:/place/mymk.tcl

Option	Description
mgrdiff	Reserved for future use. 1 = to activate comparison with existing resources for service. 0 = to turn off.
loglvl	Defines the log tracing level. A value of 3 will show informational log messages. A value greater than 3 will show debugging log messages.
logfile	Specifies the name of log file.
host	Defines the name and port (in URL format) of the host Configuration Server. For example: cmcs://localhost:3464
path	Defines the HPCA-CSDB path to the file and domain to which the package will be published, for example, PRIMARY.SOFTWARE.
filescan {body}	Specifies the control information for file scanner. The configuration file sample shows two filescan sections, to indicate that multiple filescan functions are supported. However, if you are performing only one filescan function, you must delete the additional section. Note: This applies only when intype is set to SCAN. Each filescan must contain the following options:
	dir - Directory to scan.
	distroot - Optional root directory for distribution to be used in the creation of PATH Class instance. If omitted, the root is derived by applying the value of numsplit to dir.

<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td></td><td>numsplit - Ordinal position in which to split file paths into root and stem (starting with the drive letter on Win32 systems, and the first directory on Linux platforms). The root that results from the split will be used in the creation of PATH Class instances, unless distroot is specified. The resulting stem is used to create the class instances as specified in the filters.{class} option.
<table>
<tr><th>Value</th><th>Full path</th><th>Root</th><th>Stem</th></tr>
<tr><td>0</td><td>c:/program files/my app</td><td>empty</td><td>c:/program files/my app</td></tr>
<tr><td>1</td><td>c:/program files/my app</td><td>c:/</td><td>program files/my app</td></tr>
<tr><td>0</td><td>/work/myapp</td><td>empty</td><td>/work/myapp</td></tr>
<tr><td>1</td><td>/work/myapp</td><td>/work</td><td>/myapp</td></tr>
</table>
Important Note: We recommend that you specify a minimum value of 1 on Win32 platforms, because a value of 0 will result in the drive letter being included in the stem, rather than the root.</td></tr>
<tr><td></td><td>depth - Defines how may directory levels the file scanner will scan, starting with (and including) the directory specified for dir. A value of -1 is a special case that tells the file scanner to scan to any depth. Scan depth cases are:
<table>
<tr><th>depth</th><th>result</th></tr>
<tr><td>-1</td><td>root directory and all of its subdirectories</td></tr>
<tr><td>0</td><td>root directory only</td></tr>
<tr><td>1</td><td>root directory and its files</td></tr>
<tr><td>>1</td><td>root directory and its files down to the specified depth</td></tr>
</table></td></tr>
<tr><td>filters {body}</td><td>Filters to use as selection criteria during the scan process. Multiple filters are supported. Priority of filters is the order in which they are specified. Therefore, filters for desktop links should be placed before filters for regular files. Once a file meets the selection criteria of a filter, the remaining filters do not evaluate it.</td></tr>
</table>

Option	Description
	`type` - Identifies the type of Configuration Server file being filtered. This value tells the publishing session how to create the instance in the HPCA-CSDB for a given file that matches the filtering criteria. Accepted values are FILE, DESKTOP, and REGISTRY. `class` – HPCA-CSDB class used for files selected by `filters`. For example: FILE, DESKTOP, and REGISTRY. Note: Refer to the section, Operating System Considerations on page 20 for more information. `exclude` - Specifies a file to be excluded. Values should be enclosed in quotes, with multiple values separated by a space, as in, `"*.lnk .exe"`. This option will accept an asterisk (*) wildcard. `include` – Specifies a file to be included. Values should be enclosed in quotes, with multiple values separated by a space, as in, `"*.lnk *.exe"`. This option will accept an asterisk (*) wildcard. `distroot` - Optional root directory (for distribution) to be used in the creation of PATH Class instances for any files that match this filter. Note: This setting overrides the `distroot` value specified in `filescan`. `value(s)` - Optional ZSTOP expression to be used in PACKAGE Class instance. Multiple expressions are supported, and should be arranged as one expression per line.
`expression`	The ZSTOP expression to be used in the PACKAGE Class instance. Multiple expressions are supported, but should be arranged one per line. This parameter is optional. `Note:` Although the expression is optional, the variable `expression` must be specified in the `*.cfg` file. Its value will be set in ZSTOP in the published package.

Option	Description
`replacepkg`	Replace existing package with new package. This parameter works only for packages that do not contain a PACKAGE connection. If the new package promote session does not complete, the original package remains available renamed with a leading underscore (`_packageName`). If promote session completes successfully, the original package is deleted. `1` = Replace existing package with new package. `0` = Do not replace existing package. If package exists, the HPCA Batch Publisher session is aborted.

`attr {body}`	Additional instance attribute values to be added during the promote process. The instance names and values should be enclosed in brackets, one per line. Use only valid instance names.
	
	When specifying connection type instances, use an enumerated instance name, with the exception of the first instance, for example, ALWAYS connections should be designated as: `_ALWAYS_`, `_ALWAYS_#2`, `_ALWAYS_#3`. Alternatively, you can specify a connection as `CONN0001`. The enumerated instance names are defined as follows:
	
	METHOD Connections:
	`METH0001, METH0002, METH0003…`
	
	ALWAYS Connections:
	`CONN0001, CONN0002, CONN0003…`
	
	INCLUDES Connections:
	`INCL0001, INCL0002, INCL0003…`
	
	REQUIRES Connections:
	`REQU0001, REQU0002, REQU0003…`
	
	Refer to the section Specifying Additional Attributes on page 36 for more information.

Specifying Additional Attributes

Use the HPCA Batch Publisher `attr` parameter to automatically create Service, Package, and Component instances for individual applications via a publishing session. These additional attribute values can be specified in the configuration file or directly on the command line as command-line arguments.

When specifying additional attributes, the following rules apply:

- The attributes and their values only affect the instances being created or promoted during that publishing session. For example, if the ZRSCVRFY attribute and its value for the UNIXFILE Class are specified as input to the publishing session, only instances of the UNIXFILE Class created during that publishing session are affected. No other instances of the UNIXFILE Class or any other class are affected.

- The value of the attributes, which may share an identical name with attributes in other classes, will not be contaminated by the value specified for a named class. For example, if a Batch Publisher execution will create both FILE and UNIXFILE instances in the same publishing session, it is possible to specify an altered value of the ZRSCVRFY attribute for UNIXFILE without altering the default value to be applied to the ZRSCVRFY attribute of the FILE class.

- No new attributes will be added to a class using the HPCA Batch Publisher. If an additional attribute is specified that is not defined in the class template, the attribute will not be included with the promote object and a warning will be issued in the log file (`promote.log`) as follows:

  ```
  Warning: Invalid Attribute: XYZ!
  Warning: Not defined in class template
  Warning:       -zservice-attr-XYZ discarded
  ```

- Attributes defined in the configuration file will overwrite the attributes inherited from the base instance.

- Attributes defined on the command line will overwrite the attributes defined in the configuration file and the attributes inherited from the base instance.

- The following attributes are generated by the promote process and cannot be specified in the configuration file or on the command line:

  ZRSCDATE
  ZRSCTIME
  ZRSCSIZE
  ZCMPSIZE
  ZRSCSIG
  SIGTYPE

  The following message will be issued to the log if one of these attributes is specified:

  ```
  Warning: Restricted Attribute: ZRSCDATE!
  Warning: ZRSCDATE is set during promote
  Warning:       -all-attr-ZRSCDATE discarded
  ```

- The ZRSCCRC represents a special case. The ZRSCCRC will be calculated if the additional attribute ZRSCCRC is set to YES. Not including the additional attribute will leave the ZRSCCRC field blank.

- There is no error checking of attribute values specified in the configuration file or on the command line. If a value specified is too large for its field or the character type is incorrect, the value will be truncated and the incorrect character type will be promoted. For example, specifying a two-character numeric field such as ZOBJPRI with the value ABCD will result in a value of AB after promotion.

Specifying Additional Attributes in the Configuration File

To specify an additional attribute with its associated value, an `attr` section must be added to the appropriate filter section or class section of the configuration file. Attributes are specified in the filter section for the components they apply to using a unique filter name. Additional Package, Service, and Path attributes are specified in a separate `attr` section.

The sample code below displays an excerpt from a configuration file containing the `all` filter with an additional attribute section (`attr`):

```
filters all {
             type         file
             class        unixfile
             exclude      ""
```

```
            include      "*"
            distroot     {/xyz/test}

             attr        {
              ZCREATE    {PKUNZIP &ZRSCCFIL}
              ZPERUID       (&(USER)/&(GRP))
             }
}
```

Within each appropriate filter section an `attr` section is added. The arguments of the `attr` section must be included within curly brackets ({ }). These arguments make up the attribute name and value list for that filter.

The Package, Service, and Path Class instances that are created by the HPCA Batch Publisher do not have filters associated with them. To specify attributes for these class instances, use the format below, with the attributes and their values specified between the curly brackets ({}).

```
attr PACKAGE {
            RELEASE 3.5.6
}
```

There is only one attribute and its associated value or value list allowed per line. If the value of the variable is multiple words the value must be enclosed in curly brackets ({ }) or double quotes as in the value `{PKUNZIP &ZRSCCFIL}`. Attribute names are not case-sensitive; the values are promoted in the same case in which they are specified.

If an attribute is specified and it is not part of the PACKAGE, ZSERVICE, or PATH Class or it is not part of a recognized filter, the attribute is deleted and the following message is written to the log:

```
Warning: Invalid Filter: abc !
Warning:       -abc-attr-ZUSERID discarded
```

If an attribute specified does not exist in the class template, when this attribute is processed the attribute is discarded and the log will display:

```
Warning: Invalid Attribute: NOTGOOD!
Warning: Not defined in class template
Warning:       -all-attr-NOTGOOD discarded
```

There is no limit to the number of additional attributes that can be specified or the order in which they can be specified.

Specifying Connection Types

INCLUDES, REQUIRES and ALWAYS connections can be specified for all classes that contain these type of connections. There are two methods of specifying connection types.

- Specify the explicit connection type with a sequential number appended such as `_ALWAYS_#3`.
- Specify the numbered type connection such as `CONN0001`.

REGISTRY, DESKTOP, FILE, PACKAGE, and ZSERVICE Classes contain INCLUDES, REQUIRES, and ALWAYS connections defined in the default database. The connection must be specified with the name and the number.

This sample code, displays an example of specifying connections for the ZSERVICE instance.

```
attr zservice {
      _ALWAYS_#3   SOFTWARE.ZSERVICE.REDBOX
      _ALWAYS_#2   SOFTWARE.ZSERVICE.DRAGVIEW
}
```

The connection takes the slot number specified with one exception. The `_ALWAYS_` connection of the ZSERVICE Class is reserved for use by the package instance created by the HPCA Batch Publisher session. If this connection is specified on the command line or in the configuration file, the value specified in the configuration file or on the command line will overwrite the package connection created from the promote process.

The formats for specifying additional attributes using connection types are as follows:

- Method Connections:
  `METH0001, METH0002, METH0003`
- Always Connections:
  `CONN0001, CONN0002, CONN0003`
- Includes Connections:
  `INCL0001, INCL0002, INCL0003`
- Requires Connections:
  `REQU0001, REQU0002, REQU0003`

The following is an excerpt of the configuration file with the connection type attributes specified.

```
filters all {
        type         file
        class        file
        exclude      "*.log *.bak"
        include      "*"
        distroot     {}
        attr {
   meth0001     notepad
        CONN0003 test123
        }
}
```

A table is printed in the `promote.log` that shows:

- All attributes in the class.
- The connection type (V=variable, M=method, C=class, I=includes, R=requires).
- The connection type name.
- The value inherited from the base instance.
- The values set for the HPCA Batch Publisher promote.

The following is an excerpt of the table presented in the log file.

```
Info: ------------------------------------------------------
Info:   filter = all   classname = FILE
Info:
Info: Name       Type Connection     BaseInst        RPA
Info: ------------------------------------------------------
Info: ZOBJDATE   V                   20010910
Info: ZOBJTIME   V                   17:04:57
Info: ZOBJID     V                   D0010BE54B1E
Info: ZRSCMO     V                   M               O
Info: ZINIT      M  METH0001                          notepad
Info: _ALWAYS_#3 C  CONN0003                          test123
```

If the same attribute is set using an explicit connection (for example, `ZINIT = {pzunzip &zrsccfil}`) and a connection type connection (for example, `meth0001 = notepad.exe`), the following error is generated and the HPCA Batch Publisher session is halted.

```
Error:!!!Conflict of Additional Attributes
Error:   Specify either Explicit or Connection type for Attribute
Error:   Explicit type: -all-attr-ZINIT = pzunzip &zrsccfil
Error:   Connection type: -all-attr-METH0001 = notepad.exe
```

Specifying Additional Attributes on the Command Line

Additional attributes can also be specified directly on the command line. Attributes added using the command line take the following format:

```
-(filter name)-attr-(variable name) value
```

or

```
-(class name )-attr-(variable name) value
```

Example

```
-all-attr-zinit        "PKUNZIP &ZRSCCFIL"
-package-attr-release        1.2.3
```

Therefore an example of a HPCA Batch Publisher command line with additional attributes specified would be as follows:

```
nvdkit promote.tkd cfg promote.cfg -all-attr-zinit "PKUNZIP
&ZRSCCFIL"
```

Additional attribute command-line arguments are specified in lowercase with the exception of the attribute values. The attribute values will retain the case they were specified in when promoted. If the value of the attribute contains multiple words, the value should be surrounded by double quotes as in the example above.

The filter name, attr keyword, and variable name must be separated by hyphens.

If the second element of the string is not attr, a warning is issued to the promote.log:

```
Warning: Problem command line attribute !
Warning:       -zservice-axxt-zinit discarded
```

If the configuration file is specified and the `.cfg` file exists, no new configuration file is unpacked. If the configuration file does not exist, a blank configuration file is unpacked with the name specified for the `.cfg` file. If no `.cfg` file is specified, the default name of `promote.cfg` is used for the blank configuration file that is unpacked.

When the `promote.tkd` is run, a sample `.cfg` file is unpacked.

Filters and Filescans

To specify filters and filescan configuration on the command-line use the following formats.

Filescans

Only one filescan can be specified on the command line. If additional filescans are needed they must be specified in the configuration file. The command-line options for filescan are:

```
-fs-dir
-fs-distroot        {}
-fs-numsplit        1
-fs-depth           -1
```

Filters

To specify a filter on the command line use the following argument format:

```
-filters <filtername>
-<filtername>-type        value
-<filtername>-class       value
-<filtername>-exclude     value
-<filtername>-include     value
```

You must use the filters argument to specify the unique name of the filter. There can be multiple filter entries each specifying a unique filter name. Multiple filters can be defined on the command line.

Command-line example:

```
nvdkit promote.tkd -filters testrpa -testrpa-type file -
testrpa-class file -testrpa-exclude "" -testrpa-include "*"
```

The filter executed on the command line above is displayed in the `promote.log` excerpt below:

```
20020918 11:42:05 Info: Filter[testrpa]:
20020918 11:42:05 Info: filtername = testrpa
20020918 11:42:05 Info:       type = file
20020918 11:42:05 Info:      class = file
20020918 11:42:05 Info:    include = *
20020918 11:42:05 Info:    exclude = {}
```

There is no limit to the number of additional attributes that can be specified or the order in which they can be specified. The same rules that apply to the configuration file for valid attributes also apply to the command-line attributes.

Specifying attributes on the command line, the attribute must be in a recognized filter or in the zservice, package, or path class. If not, the following message is written to the log:

```
Warning: Invalid Filter: abc !
Warning:       -abc-attr-ZUSERID discarded
```

If a package name is not specified on the command line, the default package name of `rpadefault*` is used.

```
#
# package       - package instance name or prefix (i.e. foo or foo_*)
# pkgname       - to be used as friendly name of package (NAME)
# pkgdesc       - to be used as description of package (DESCRIPT)
# service       - zservice instance name
# svcname       - to be used as friendly name of the service (ZSVCNAME)
# svcdesc       - to be used as a description of the service (NAME)
# addtosvc      - connect package to service
# compress      - 1 to request compression
# intype        - source type for list of resources (FILE/SCAN)
# insource      - file path for input if type is FILE
# mgrdiff       - 1 to activate comparison with existing resources for service - not implemented
#
#
    package     "attr_test"
    pkgname     "attr_test"
    pkgdesc     "attr_test"

    service     "attr_test"
    svcname     "attr_test"
    svcdesc     "attr_test"
    addtosvc    1
```

```
    compress    1
    intype      SCAN
    insource    ""

    mgrdiff     0

    loglvl      3
    logfile     promote.log
    hostcmcs://localhost:3464
    pathPRIMARY.SOFTWARE
#
    # File Scanner Control Info
    # depth     - number of subdirs to traverse (-1 = all)
    # numsplit  - number of subdirs (includes drive in win) to use in root
    # distroot  - distribution root to be used to create path instance
    #             if left blank, root of dir is used
    #
    filescan {
        dir             {c:/attr/test}
        distroot        {}
        numsplit        2
        depth           2
    }
    #
    # Priority of the component classes as receiving bucket is based on
    # filter order
    # Specialized (like desktop) should be put before file class filters
    #
    # Abstract Filters (multi-type)
    # class     - database class used for files that satisfy this filter
    # expression - expression strings for ZSTOPs in package instance
    #
filters reg {
            type         registry
            class        registry
            exclude      ""
            include      "*.reg *.edr"
            distroot     {}
}

filters lnk {
        type         desktop
        class        desktop
        exclude      ""
        include      "*.lnk"
        distroot     {}
        attr    {
        MACHUSER    TESTUSER
        ZCREATE     {PKUNZIP &ZRSCCFIL}
```

```
        }
    }
filters all {
                type         file
                class        file
                exclude      ""
                include      "*"
                distroot     {/john/test}
                attr    {
                ZCREATE     TESTSTART
                ZDELETE     TESTOVER
                }
    }
  expression {
    }
attr package {
            releASE     3.5.6
            wrong       thisiswrong
            includes    SOFTWARE.PACKAGE.ADAPT
            includes#2  SOFTWARE.PACKAGE.RAPILINK
        }
        attr  zservice {
        ZSVCMO          m
        URL             {WWW.HP.COM}
        _ALWAYS_#3      SOFTWARE.ZSERVICE.REDBOX
        _ALWAYS_#2      SOFTWARE.ZSERVICE.DRAGVIEW
        }
     attr path {
                zrscmo  O
 }
```

Summary

- Execute configuration file-based publishing from the command line.
- Edit `promote.cfg` to include your required publishing parameters.
- Use the `attr` parameter to specify additional attributes.

4 HPCA Native Packaging

At the end of this chapter, you will:

- Be familiar with HPCA Native Packaging.
- Understand HPCA Native Packaging system requirements.
- Understand the HPCA Native Packaging command-line interface.
- Know how to publish using HPCA Native Packaging.

What is HPCA Native Packaging?

HPCA Native Packaging is a feature of the HPCA Batch Publisher specifically designed for Linux environments. It is a command-line-driven content-publishing tool that:

- Supports native Linux software.
- Is neither a graphical publishing tool nor a mainstream publishing tool.
- Is installed during the regular installation of the HPCA Batch Publisher on a Linux system.
- Explores Linux native software depots and searches for available native packages
- Publishes wrapped native packages to the HPCA Configuration Server. It will publish all necessary information that will allow you immediate installation of native software to end clients including, if necessary, information about native package dependencies.

Why use HPCA Native Packaging?

HPCA Native Packaging supports RedHat Linux RPM software package formats. With the use of HPCA Native Packaging you can easily publish wrapped native Linux software, updates, and patches without any need for re-packaging. Wrapped Linux native software enables policy-based centralized software management of your Linux agents.

This document assumes that the system administrator who uses the HPCA Native Packager possesses packaging/publishing knowledge for a HPCA Configuration Server Database.

Overview

HPCA Native Packaging creates the standard instances of ZSERVICE, PACKAGE, and PATH in the SOFTWARE Domain of the HPCA-CSDB. HPCA Native Packaging creates instances of RPM classes for each published wrapped native package for RedHat Linux.

For each native software package selected, HPCA Native Packaging will create an instance of the RPM class. This instance holds actual content (software depot) and native method calls that will do actual install/removal/update on the client. It will also create an instance of the PACKAGE Class that contains the newly created instance and an instance of ZSERVICE Class that contains the new PACKAGE instance.

Publish native packages from the specific Linux platform to which you will be deploying. For example, you cannot use HPCA Native Packaging on a non-Linux platform to promote Linux RPM packages – HPCA Native Packaging would be unable to use the native Linux utilities to interrogate details of the package.

HPCA Native Packaging System Requirements

HPCA Native Packaging is available for the RedHat Linux operating systems. It has these system requirements:

- Root permissions are required to use HPCA Native Packaging.
- Network connectivity to the HPCA Configuration Server.
- Space on `/tmp` file system for temporary depot files used for publishing.

Required Class

HPCA Native Packaging requires a specific class for the operating system. Make sure your HPCA-CSDB includes these SOFTWARE Domain classes before using HPCA Native Packaging.

Table 4 Required SOFTWARE Domain Class

Operating System	Class
RedHat Linux	Linux RPM Packages (RPM)

HPCA Native Packaging and the HPCA Agent

During the installation of the HPCA agent, a Tcl script is installed into the `IDMSYS` directory along with the HPCA agent components. This script is required for deployment of packages published using HPCA Native Packaging. The actual Tcl script installed is customized for the Linux environment. The script `rpm.tcl` for RedHat Linux contains native command calls to deploy the software.

A common helper Tcl script `method_utils.tcl` is also installed with the HPCA agent.

Supported Native Package Types

Table 5 below lists the native package type supported by the HPCA Native Packaging and its expected format.

Table 5 Native Package and Supported Format

Native Package	Supported Format
RedHat Linux RPM Package	`*.rpm` file

HPCA Native Packaging Command-Line Interface

HPCA Native Packaging is run from the command line. The base input parameter for HPCA Native Packaging is the source depot containing the RedHat Linux software. The native packages must be in a disk depot format (the native software packages are resident on disk in a format that can be utilized immediately by the native operating system's software management

tools). HPCA Native Packaging is capable of publishing one or more packages in a single publishing session.

In addition, you can specify the selection of the software you want to publish, and in the event HPCA-CSDB user verification is enabled, an optional user ID and password can be designated. Here is an example of command-line usage for HPCA Native Packaging:

```
20070201 18:18:40 Info: Message catalog for en_us loaded.
Usage: rnp -d depot_path -m manager_ip:manager_port
    [-v] [-debug type] [-tmp directory]
    [-user user_id] [-pass password] [-admin]
    [-depth level] [-dist dist_depot_path]
    [-domain domain] [-l logfile] [-help]
    [-i] [-coreq] [-I] [-M] [-S]
    [-a | -A type | -p
package1[,r=revision][,a=architecture][,v=vendor]
                -p
package2[,r=revision][,a=architecture][,v=vendor]...]
    [-P] [-r] [-f prefix]
    [-s] [-t service_type] [-c flag] [-relyondb]
```

The table below contains the description of the command-line arguments for HPCA Native Packaging.

Table 6 Command-line parameters

Parameter	Description
`-a`	Specifies to publish all native software available in the depot. This parameter is optional. You cannot use this parameter with `-p`.

Parameter	Description

`-A` *type*	Select and publish all packages of specific *type*.
	
	type can also be one of the following:
	
	`help`
	for a list of valid types for the running platform.
	
	`all`
	to select all package types. This option would then behave like the `-a` option.
	
	`none`
	to select none of the package types. This would then behave like having neither the `-a` or `-A` options specified.
	
	Multiple package types can be specified and separated by commas.
	
	This parameter is optional.

`-c flag`	This option enables or disables compression on all packages to be published. *flag* can be one of the following: • `yes` Enable compression for all packages. • `no` Disable compression for all packages. Default behavior is dependent on each package type being published. This parameter is optional.
`-d depot` *path*	Specifies the path to the depot directory containing native software packages. Software contained in this depot will serve as an input to Native Packaging. This parameter is required.

Parameter	Description
`-debug type`	Specify the level of debugging desired. *type* can be one of the following: `init` for initialization data `func` for detailed function debugging `trace` for function tracing `cmd` for native command executions `pub` for publishing information `rapi` for HPCA Batch Publisher details `all` for all the above `none` to disable debugging Multiple types can be specified and separated by commas. The default behavior is that debugging is disabled. This parameter is optional.
`-depth`	For Linux RPM packages. Determines the level of dependency processing for the target package. 0 – Process all dependencies. 1 – (Default) process 1 level of dependency. You can specify any number of level dependencies you require. If no level is defined (-depth is not part of the rnp command), then one level of dependencies is processed (assuming –i option used). This option allows for HPCA Batch Publisher backwards compatibility.
`-dist distribution_depot`	For Linux RPM packages. Specifies the path to a distribution depot directory. Published packages will contain DIST=distribution_depot in the CONTENTS field of their package class instance.

Parameter	Description
`-domain` *domain*	Specify which HPCA Configuration Server domain the packages are to be published to. The default domain used is PRIMARY.SOFTWARE. This parameter is optional.
`-f prefix`	Instructs HPCA Native Packaging to prefix the PACKAGE Class and service class instance names used for the new published package with this prefix. This parameter is optional.
`-help`	Display help on the command-line usage and the `rnp.cfg` configuration file format.
`-i`	Instructs HPCA Native Packaging to include prerequisite software package (supported for RPM packages) with the package you have selected if prerequisite software is present in the source depot. Dependency information is published regardless of this parameter. This parameter is optional.
`-I`	Interactive mode. Allows user to select more required packages (dependency). Ignored if neither –i nor –coreq are present or no additional package is required. Note: Available for RPM packages.
`-l` *logfile*	Instructs HPCA Native Packaging to store the log details in the *logfile* specified. If this option is omitted, the default log file created is `publish.log`. This parameter is optional.
`-m` *ip:port*	Specifies the host name or IP address and port of the HPCA Configuration Server to which you intend to publish software. This parameter is required.
`-M`	Multiple. If `-i` or `-coreq` is present (so additional packages are required), and there are several versions of an additional package, then all of them will be displayed in the additional packages menu. Otherwise, only one version of each additional package will be displayed (default). It is ignored if `-I` is not present.

Parameter	Description
-p *package* [,r=revision] [,a=arch] [,v=vendor]	Specifies a software package to publish to the HPCA Configuration Server. Specify the following: an RPM package on RedHat Linux (software selection with optional revision, architecture and vendor. Specifying the software selection alone will work, but if there are multiple products with the same identifier, they will all be published). This parameter is optional. You can specify multiple -p *package* parameters for multiple package selections. Note: If a package is not specified on the command line, you will be presented with a list of all available packages within the specified depot.
-pass *password*	HPCA administrator password. This parameter is optional.
-relyondb	For Linux RPM packages. Use this option to rely on package database information instead of a native command return code when installing packages. A native database query command is used to verify the package was installed (for RPM, rpm –q) Promoting a package with this option sets the HPCA Configuration Server Database attribute RELYONDB to Y. Default is blank (N).
-s	Instructs HPCA Native Packaging to skip the creation of services for the packages to be published.
-S	Strict mode. If any requirements for a package are not met (for example, if -i or -coreq option are present and not all additionally required packages are in the depot), the package will not be promoted. It is ignored if -I option is present.

Parameter	Description
-t *svc_type*	Use this option to specify the type of service to create. Available values: M for Mandatory O for Optional Default Service type created is M. This parameter is ignored when the -s option is specified.
-tmp *dir*	Specify an alternative temporary directory to use when creating packages. The default value is /tmp. This parameter is useful when */tmp* on the machine where publishing is performed has limited available disk space. This parameter is optional.
-user *user ID*	HPCA administrator user ID. The default is RAD_MAST. This parameter is optional.
-v	Displays the version and build number of the HPCA Native Packager rnp command. This parameter is optional.

When no packages are specified with the -p option or by selecting all packages with the -a or -A options, the HPCA Native Packaging command will present a text based menu of native packages found in the depot directory specified. You can then select individual or all packages from the menu to be published.

HPCA Native Packaging Options File

If you usually use the same source depot, or publish to the same HPCA Configuration Server, you can create a file, rnp.cfg, in the same directory where you have the HPCA Native Packaging components installed. Use of this configuration file allows you to preset default option values in the following format:

parameter=value

Example:

```
depot=<depot path>
manager_ip=<HPCA configuration server IP or hostname>
manager_port=<port number that the HPCA configuration server
uses>
```

By default, `rnp.cfg` is not supplied.

Table 7 Supported `rnp.cfg` settings and default values

Setting	Expected Values	Default Value
`depot`	Fully qualified path to the depot directory	None
`manager_ip`	IP address or hostname of the HPCA Configuration Server	None
`manager_port`	Port number of the HPCA Configuration Server	`manager_port=3464`
`user`	`user=userid` Administrator ID used for authentication with the HPCA Configuration Server.	`User=RAD_MAST`
`create_service`	`create_service=[yes/no]` A value of `yes` will create a ZSERVICE instance for each of the promoted packages. A value of `no` will not automatically create a ZSERVICE instance for each of the promoted packages	`create_service=yes`
`service_type`	`service_type=[M/O]` A value of `M` will cause the promoted ZSERVICE instance to be set as a mandatory service. A value of `O` will cause the promoted ZSERVICE instance to be set as an optional service.	`service_type=M`

Setting	Expected Values	Default Value
include_responses	include_responses=[yes/no] A setting of yes will include SVR4 response files when they are found in the Solaris depot. Value of no will not include response files for Solaris SVR4 packages.	include_responses=no
include_dependencies	include_dependencies=[yes/no] A value of yes will attempt to publish RPM dependent packages if they are in the specified depot. A value of no will not attempt to publish RPM dependent packages.	include_dependencies=no
include_adminfile	include_adminfile=[yes/no]	include_adminfile=no
select_types	select_types=[type1,type2,…] Publish all packages of specific types found in the depot directory. Run rnp with the -A help option to get a complete list of supported types.	select_types=none
debug	debug=[type1,type2,…] List specific types of debugging to enable. valid types are: init, func, trace, cmd, pub, rapi, all or none.	debug=none
temp_dir	temp_dir=[dir] Specify an alternate temporary directory to use for creating the packages to publish.	temp_dir=/tmp

Setting	Expected Values	Default Value
domain	domain=FILE.DOMAIN Specify the target FILE.DOMAIN in the HPCA-CSDB where to publish the packages.	domain=PRIMARY.SOFTWARE
compress	compress=[yes/no] Enable or disable compression for all packages to be published. The default behavior is that compression depends on the package type being published.	Package Dependent
password	password=pass Administrator password, used for authentication with the HPCA Configuration Server.	Blank
interactive	interactive=[yes/no] Publish using interactive mode. Interactive mode allows you to choose whether or not to include required packages.	interactive=no
strict	strict=[yes/no] Publish using strict mode. Strict mode will not publish packages missing required components.	strict=no
multiple	multiple=[yes/no] Display multiple versions of additional required packages,	multiple=no
rely_on_db	rely_on_db=[yes/no]	rely_on_db=no
requisite_depth	Set to 0 to include all dependencies or set to a number of levels.	requisite_depth=1
distribution_depot	Distribution depot path.	Blank

Publishing with HPCA Native Packaging

Example

See Table 6 on page 51 for an explanation of the HPCA Native Packager command-line parameters.

To publish a specific Xchat RPM package residing in the specified depot on RedHat Linux

1 Change your current working directory to the HPCA Batch Publisher directory.

2 On the command line, type:

```
./rnp -d /home/rpmadmin -p xchat-1.4.0.2.i386.rpm
```

Or simply:

```
./rnp -d /home/rpmadmin -p xchat
```

If a package is not supplied on the command line via the `-p` parameter, you will be presented with a list of all available packages within the specified depot.

Publishing with Interactive Mode

When you specify the parameter`-I` on the command line, the HPCA Native Packager interactive mode is invoked. This allows you to select which of the *available* required software you would like to include with your current package. You will also see which required prerequisite software is not available in the current depot.

The interactive mode option is ignored if neither the `-I` nor `-coreq` or `-i` parameters are specified on the command line (indicating prerequisite software is required for the current package). Here is an example of Interactive Mode:

```
------------------------------------------------------
Processing additional software required for QA_MASTER_1-1.0.0-0.i386.rpm
```

```
Following additionally required software is found in software depot
and selected to be included in to promote package:
      1. prereqs:  - QA_RPM2-1.2.0-0.i386.rpm - included
      2. prereqs:  - QA_RPM3-1.0.0-0.i386.rpm - included
      3. prereqs:  - QA_RPM4-1.0.0-0.i386.rpm - included

Please toggle the selection:
Select (a to include all; d to exclude all; c to continue; s to skip current package; q to
quit entire session; a number to toggle its selection):
```

You can exclude any of the required software by entering the corresponding number. A message at the end of each line (included or not included) lets you know whether or not the required software will be included with the current package.

- Enter the number of the required software or type another option available in the interactive mode menu and press **Enter** to continue the native packaging process.

Table 8 Interactive Mode Selections

Selection	Description
a	Selects all available required software to include with the current package. Available required software is included by default. (Set all available required software to included)
c	Continue the native packaging process. If you have made changes to the list of packages to be processed, the continue option will update this list. Use the continue option again to move on to the next processing step. This behavior accommodates multi-level dependency processing.
d	Deselects all included software. (Set all available required software to not included)
q	Quit the HPCA Native Packaging process.
s	Skip the current package.

Wrapped Native Packages

The following section lists all HPCA-CSDB class instances and their attributes that are created when you publish native Linux software with HPCA Native Packaging.

HPCA Native Packaging utilizes a **method harness** to invoke agent methods, therefore when a package is published to the HPCA-CSDB, populated method attributes such as ZCREATE, ZDELETE, ZUPDATE, ZVERIFY, and ZREPAIR will contain the text "hide nvdkit method."

The supplied agent methods are designed to invoke the native software management utilities, therefore, the methods are specific for the agent platforms.

When publishing native Linux packages using the HPCA Native Publisher, the software packages are published to the HPCA-CSDB (in compressed format). The table below lists the modified attributes:

Table 9 RPM Class Instance Attributes

Attribute	Description
ZRSCNAME	Specifies a string that is used by native methods to identify software contained in the published depot. This is the RPM Package Name.
ZRSCCFIL	Specifies the path to the file that is included in this instance. This file contains the native packaged software.
ZCREATE	Uses method "Harness" call. The HPCA agent method `rpm.tcl` script contains a native command call to install the software package: `hide nvdkit method`
ZDELETE	Uses method "Harness" call. The HPCA agent method `rpm.tcl` script contains a native command call to remove the software package: `hide nvdkit method`
ZUPDATE	Uses method "Harness" call. The HPCA agent method `rpm.tcl` script contains a native command call to update the software package: `hide nvdkit method`

Attribute	Description
ZVERIFY	Uses method "Harness" call. The HPCA agent method `rpm.tcl` script contains a native command call to verify the installed software package: `hide nvdkit method`
ZREPAIR	Uses method "Harness" call. The HPCA agent method `rpm.tcl` script contains a native command call to repair the installed software package (reinstall): `hide nvdkit method`
PKGVER	Package Version. Informational attribute only.
PKGREL	Package Release. Informational attribute only.
PKGARCH	Package Architecture. Informational attribute only.
PKGSUMM	Package Summary. Informational attribute only.
REQPKGS	Required Packages. Informational attribute only.
REQCMDS	Required Commands. Informational attribute only.
REQLIBS	Required shared libraries. Informational attribute only.
CONTENTS	Required packages included in Tar file.
PKGEPOCH	RPM Package EPOCH.
INSTOPTS	Package installation options. (For example, --oldpackage , --replacepkgs). NOCHECK keyword can be added to installation options. See Operational Notes for more information.
VRFYOPTS	Package verify options. (For example, --nomode , --nogroup)

An instance of PACKAGE class is created that contains the instance of the RPM class. Table 10 below describes how HPCA Native Packaging maps native package information into PACKAGE class attributes.

Table 10 PACKAGE Class attributes

Attribute	Description
Instance Name	For RPM, the `RPM_` prefix is added to the RPM Package Name and date and sequence number is appended

Attribute	Description
	(RPM_<PKG>_yyyymmddn). Note: When instance names generated are longer than 32 characters, the package/patch names parts of the instance names shall be truncated.
RELEASE	The RPM version native attributes are mapped into RELEASE.
NAME	The RPM Packages, RPM_ prefix is added to the RPM package name and suffixed with the package version , release and architecture (RPM_<PKG>,ver=<VERSION>,rel=<RELEASE>,arch=<ARCH>).
DESCRIPT	The RPM Packages, the package summary is mapped into DESCRIPT.
ZSTOP000	Contains an expression that contains target operating system information.
FILE	Holds reference to respective instance of RPM class.

HPCA Native Packaging also creates an instance of ZSERVICE Class holding previously created instance of PACKAGE Class. Table 11 on page 63 lists the modified attributes.

Table 11 ZSERVICE Class attributes

Attribute	Description
Instance Name	For RPM, the RPM_ prefix is added to the RPM Package Name and a date and sequence number is appended (RPM_<PKG>_yyyymmddn). Note: When instance names generated are longer than 32 characters, the package/patch names parts of the instance names shall be truncated.
VERSION	For RPM Packages, the RPM Package Version is mapped into VERSION.

Attribute	Description
NAME	For RPM Packages, the RPM_ prefix is added to the RPM package name and suffixed with the package version, release and architecture (`RPM_<PKG>,ver=<VERSION>, rel=<RELEASE>,arch=<ARCH>`).
ZSVCNAME	For RPM, the RPM Package Name is mapped into ZSVCNAME.
VENDOR	Specifies vendor of the native Linux package.
ZSVCMO	Service is set to mandatory by default. Valid values of this attribute are: • `M` for mandatory • `O` for optional
ALWAYS	Holds reference to the respective instance of PACKAGE Class.

If a package requires a system reboot after an agent connect, make sure the `hreboot radskman` parameter is set to `Y`. Refer to the HPCA *Application Manager Guide* for more information.

Automatic Inclusion of Required Packages

If you specify the `-i` command-line option, HPCA Native Packaging will include prerequisite packages into the depot with the (main) package you are publishing to HPCA. The prerequisite package needs to exist in the depot HPCA Native Packaging is using as a source. For RPM packages, all prerequisite packages as well as any additional prerequisite packages they might require are included (this can be determined by using the –depth option).

When using the `-i` or `-coreq` options, the promotion of native software packages will not fail because of a missing prerequisite or corequisite package (unless the `-S` option is specified). Installation will fail only if prerequisite or corequisite packages are missing from both the promoted native software package *and* from the target machine.

Alternatively, if a prerequisite or corequisite package is already installed on the target machine, including these in a native software package for

promotion will result only in using more network bandwidth and disk space than necessary.

Publishing packages with prerequisites included may take an extended amount of time. About every thirty seconds, a progress message is displayed:

```
Info:    Compiling extended info about all packages in the depot.
Please wait...
```

Troubleshooting HPCA Native Packaging

Should you encounter problems publishing native Linux software packages, please perform the following steps before contacting technical support:

- Enable full diagnostic tracing by appending the text `-debug all` to your command line and re-run the publishing session.
- Have the log file produced by the HPCA Native Packager publishing readily accessible to provide to support. By default, the log file would be called `publish.log` located in the directory where you installed the HPCA Batch Publisher.

You should only use the command-line option `-debug all` to diagnose publishing problems.

Operational Notes

The following describes the operations involved during the publishing and deployment of native packages. This gives you a better understanding of the current processes and capabilities provided to manage these packages.

Publishing

- All packages are selected from the software depot specified by using the -d option. We recommend that you build a depot with packages of the

same architecture only (for example, separate depots of RPMs for installation of i386 or x86_64 machines).

- "-dist *a_distribution_depot*" option (RPM packages bundles only) where *a_distribution_depot* is the location of packages used during deployment. If this option is present, then the -d option can be omitted and *a_distribution_depot* location will be used instead. Packages promoted with the -dist option will contain "DIST=distribution_depot_path" in the CONTENTS field of their package class instances.

- Dependency checking is performed on the target (selected) package as well as on all its dependent packages (currently, multi-level dependency checking is implemented for RPM). Use "–depth N" option to control the depth of dependency processing. If the –depth option is not defined, only one level of dependencies (of the target package only) are processed. To include all dependencies, use –depth 0.

- On Linux, a dependency's release level can be specified as conditional (>= version 2, release 1). If multiple dependencies are found to satisfy this condition, by default the newest package is selected for inclusion. If a specific version is desired, one can use the –M option in interactive mode to list all possible matches, and select the one desired.

- Use the `-s` (strict) option to ensure that all identified dependencies are included in the deployment. If required dependencies are not found in the software depot, an error message will be displayed and publishing will be terminated.

- Using Interactive mode allows you to:

  — See all packages in the software depot available for selection

  — Review all dependencies found for a selected package.

  — Select / de-select dependencies. This allows administrators who have knowledge of their target machines to tailor the deployment to fit their environments and needs. Some dependencies can be large, and rather than waste bandwidth and client processing, if not needed, it can be removed from the deployment.

Deployment

- If a package was promoted with the –dist option, it will be installed from the distribution depot specified in the CONTENTS field.

- If the target package is already installed on the machine and is newer than the one to be deployed, no further processing is done, and the deployment is viewed as successful. However, since it was not deployed, it will not be removed when the service is deleted.

  

  If back leveling of the package is required, this behavior can be overridden by specifying the appropriate native command-line option in the attribute INSTOPTS for RPM packages. This requires the use of the HPCA-CSDB Editor

- If (during installation) the target package already exists and is the same release level, it is first verified. If verification fails, it will be re-installed. Subsequent verify or delete processing would occur as usual. This behavior can be changed by adding NOCHECK keyword into INSTOPTS attribute of the package (this requires the use of the HPCA-CSDB Editor). If present, the package will be re-installed even if it does not fail verification. It is valid for RPM packages only. Appropriate options for the installation are still required.

- During install/update, the release levels of already installed dependencies are individually checked, and if newer on machine, they are not installed as this may cause conflicts for other packages. This behavior can be changed by adding the NOCHECK keyword to the INSTOPTS attribute of the package (this requires the use of the HPCA-CSDB Editor). For example, if INSTOPTS for an RPM package is set to “NOCHECK --oldpackage” or “NOCHECK --force”, then required packages for the package can be back leveled; while without NOCHECK, only the main package can be back leveled.

- During verify, only the target package is verified and not its dependencies.

- After installation, the native package database is queried to make sure the target package was properly installed and registered in the database.

- During removal, the package is checked to make sure it exists (as it may have been upgraded or superseded). If it does not exist, no attempt to delete it is made, and the process is viewed as successful. Only the target

package is deleted. Dependent packages are not deleted, as they may be required for other packages.

- If the verify attribute (ZRSCVRFY) of the package instance is set to N, the source depot (file actually deployed from server) is deleted after a successful installation. If a subsequent verification of the installed target package fails, this file is again downloaded and used to repair the damaged package.

Event Reporting

Use the RNPEVENT object to report events to the HPCA Configuration Server. Similar to the APPEVENT object, RNPEVENT uses the same variable set and is created if the HPCA administrator has enabled the reporting flags for a particular event in the EVENTS variable of the ZSERVICE Class. The RNPEVENT variables are listed in the table below.

Table 12 RNPEVENT variables

Variable	Description	Sample Value
EVENT	Text description of the current event.	create
STATUS	Error messages.	Successful
CMDRC	Return code from native command.	0
CMDMSG	Message from native command.	Main package <Regina> on desktop <2.0> is newer than in CM-CS <1.0>. Skipping further processing.
POSTRC	Return code from RNP post-processing check.	0
POSTMSG	Message from RNP post-processing check	Post installation is successful
ZOBJDOMN	The domain name for the application.	SOFTWARE

Variable	Description	Sample Value
ZOBJCLAS	The class name for the application.	ZSERVICE
ZOBJNAME	The instance name for the application.	RPM_GAIM_200504123
ZOBJID	The objects ID for the instance.	D123ACD45F67
ZUSERID	HPCA user that installed the application.	RPMUSER_LINUX
DELDATE	ISO8601 date time when the delete event occurred.	2005-05-10T16:45:00Z
VERDATE	ISO8601 date time when the verify event occurred.	2005-06-10T16:47:00Z
INSTDATE	ISO8601 date time when the install event occurred.	2005-07-10T16:44:00Z
FIXDATE	ISO8601 date time when the repair event occurred.	2005-08-10T16:43:00Z
UPGDATE	ISO8601 date time when the update event occurred.	2005-09-10T16:42:00Z
JOBID	Session identifier	MachineConnect
CJOBID	Session identifier	11122:3

Viewing Event Details

Use the Reporting Server to view the details of your Native Package Events. View the details of the HPCA Managed Service, then select the HPCA Native Package Events report. Refer to the *HPCA Reporting Server Guide* for details on using the Reporting Server.

Summary

- HPCA Native Packaging is a feature of the HPCA Batch Publisher specifically designed for Linux environments.
- HPCA Native Packaging requires specific classes for Linux.

Index

E

F

G

H

I

J

L

M

N

O

P

R

S

U

V