



PKZIP for Windows Command Line

Users Manual

Copyright © 1997-2005 PKWARE, Inc. All Rights Reserved.

No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any other language in whole or in part, in any form or by any means, whether it be electronic, mechanical, magnetic, optical, manual or otherwise, without prior written consent of PKWARE, Inc.

PKWARE, INC., DISCLAIMS ALL WARRANTIES AS TO THIS SOFTWARE, WHETHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, FUNCTIONALITY, DATA INTEGRITY, OR PROTECTION. PKWARE IS NOT LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES.

Portions of this software include RSA BSAFE ® cryptographic or security protocol software from RSA Security Inc.

This software includes portions that are copyright © The OpenLDAP Foundation, 1998-2003 and are used under the OpenLDAP Public License. The text of this license is indented below:

The OpenLDAP Public License
Version 2.7, 7 September 2001

Redistribution and use of this software and associated documentation ("Software"), with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain copyright statements and notices,
2. Redistributions in binary form must reproduce applicable copyright statements and notices, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution, and
3. Redistributions must contain a verbatim copy of this document.

The OpenLDAP Foundation may revise this license from time to time. Each revision is distinguished by a version number. You may use this Software under terms of this license revision or under the terms of any subsequent revision of the license.

THIS SOFTWARE IS PROVIDED BY THE OPENLDAP FOUNDATION AND ITS CONTRIBUTORS ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OPENLDAP FOUNDATION, ITS CONTRIBUTORS, OR THE AUTHOR(S) OR OWNER(S) OF THE SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The names of the authors and copyright holders must not be used in advertising or otherwise to promote the sale, use or other dealing in this Software without specific, written prior permission. Title to copyright in this Software shall at all times remain with copyright holders.

OpenLDAP is a registered trademark of the OpenLDAP Foundation.

Copyright 1999-2001 The OpenLDAP Foundation, Redwood City, California, USA. All Rights Reserved. Permission to copy and distribute verbatim copies of this document is granted.

PKWARE, the PKWARE logo, the zipper logo, PKZIP, PKUNZIP, and PKSFX are registered trademarks of PKWARE, Inc. SecureZIP is a trademark of PKWARE, Inc.

Trademarks of other companies mentioned in this documentation appear for identification purposes only and are the property of their respective companies.

RSA and BSAFE are registered trademarks of RSA Security Inc.

Table of Contents

| | | |
|----------|--|-----------|
| 1 | WELCOME..... | 1 |
| | Introduction..... | 1 |
| | About This Manual | 1 |
| | Your Work Environment: The Command Line..... | 2 |
| | Entering Commands | 2 |
| | Setting PKZIP in the Path (Windows)..... | 3 |
| | Windows 98/Me | 3 |
| | Windows NT/Windows 2000/Windows XP | 3 |
| | Strong Encryption | 4 |
| | Creating the Tutorial Directory and Files..... | 4 |
| | Setting Up the Tutorial in Windows..... | 5 |
| | Using Help..... | 5 |
| | Getting License Information..... | 6 |
| | Getting Version Information | 6 |
| | Technical Support | 7 |
| 2 | THE BASICS..... | 8 |
| | Introduction..... | 8 |
| | An Overview of What PKZIP Does | 8 |
| | Supported Archive Types..... | 8 |
| | Using the Tutorials | 9 |
| | Using Sample Files and Directories..... | 9 |
| | Preparing Your Workspace..... | 10 |
| | Compressing Files into an Archive: The Basics | 10 |
| | Archive File Naming Conventions..... | 10 |
| | Compressing a Single File in the Current Directory | 11 |
| | Compressing Selected Files in the Current Directory..... | 12 |
| | Compressing Files that Match a Pattern..... | 13 |
| | Compressing All Files in the Current Directory..... | 14 |
| | Compressing Files from a Different Location..... | 14 |
| | Specifying a Different Location for the .ZIP File | 15 |
| | Moving Files into Your .ZIP File..... | 15 |
| | Viewing Files Within a .ZIP File | 16 |
| | Further Information on Compressing Files | 17 |

| | |
|--|-----------|
| Extracting Files: Learning the Basics | 17 |
| Extracting a Single File From a .ZIP File | 18 |
| Extracting Multiple Files From a .ZIP File | 19 |
| Overwriting Files that Already Exist in Your Directory | 21 |
| Extracting Files to a Specified Directory | 22 |
| Further Information on Extracting Files..... | 22 |
| Understanding PKZIP Commands and Options..... | 22 |
| Difference between a Command and Option..... | 23 |
| Including an Option in Your Command Line..... | 23 |
| Abbreviating Commands and Options | 23 |
| Combining Options | 24 |
| Commands and Options That Have Values | 25 |
| Commands and Options: Compression or Extraction? | 26 |
| Setting Defaults..... | 26 |
| More Ways to Select Files | 26 |
| Selecting Files by Date | 26 |
| Selecting Files by Age | 27 |
| Selecting Files by Size..... | 28 |
| Selecting Files to Include or Exclude..... | 28 |
| 3 ADDING FILES TO AN ARCHIVE..... | 30 |
| Conventions in this Chapter..... | 30 |
| Default Values for Commands and Options | 30 |
| Creating and Updating Archives..... | 30 |
| Adding All Files in a Directory..... | 31 |
| Adding New and Modified Files | 31 |
| Adding Only Files That Have Changed | 32 |
| Clearing Archive Attributes | 32 |
| Incremental Archiving | 33 |
| Writing an Archive to STDOUT | 33 |
| Encrypting Files That You Add to an Archive | 34 |
| Encrypting Files with a Password | 34 |
| Encrypting File Names..... | 36 |
| Contingency Keys | 37 |
| Compressing Files in Subdirectories..... | 38 |
| Storing Directory Path Information | 38 |
| Additional Methods for Storing Directory Path Information..... | 39 |
| Storing and Recreating Directory Path Information | 40 |
| Setting the Compression Level..... | 41 |
| Specifying a Compression Level from 0-9 | 42 |
| Specifying a Compression Level by Name | 42 |
| Compressing Files with a List File | 43 |
| Getting a List of Files from Standard Input | 44 |
| Compressing Files in a Specific Format..... | 44 |

| | |
|---|-----------|
| Compressing Files with the Deflate64 Method..... | 45 |
| Compressing Files with the BZIP2 Method..... | 45 |
| Compressing Files Compatible with the Data Compression Library..... | 46 |
| Compressing Files to a Specified Type of Archive..... | 46 |
| Compressing Files to Diskette..... | 47 |
| Creating a Spanned Archive..... | 47 |
| Creating a Split Archive..... | 47 |
| Creating Multiple, Respective Archives..... | 48 |
| Storing File Information..... | 49 |
| Compressing Files with Specified Attributes..... | 49 |
| Extended Attribute Storage..... | 50 |
| Including Additional Information in a ZIP File..... | 51 |
| Including a Text Comment..... | 52 |
| Including a Header Comment..... | 52 |
| Specifying the Date of a .ZIP File..... | 53 |
| Removing File Attributes..... | 54 |
| Sorting Files Within a .ZIP File..... | 54 |
| Moving Files to a .ZIP File..... | 56 |
| Wiping Deleted Files..... | 56 |
| 4 EXTRACTING FILES..... | 58 |
| Introduction..... | 58 |
| Conventions in This Chapter..... | 58 |
| Default Values for Commands and Options..... | 58 |
| Extracting New and Existing Files..... | 58 |
| Extracting All Files from an Archive..... | 59 |
| Extracting Newer Versions of Existing Files and New Files..... | 59 |
| Extracting Only Newer Versions of Files..... | 59 |
| Extracting from an Archive Embedded in An Archive..... | 60 |
| Checking for Viruses when Extracting..... | 60 |
| Extracting Files in Lower Case..... | 61 |
| Preserving File Times..... | 62 |
| Translating End of Line Sequence..... | 62 |
| Retaining Directory Structure while Extracting..... | 62 |

| | |
|---|-----------|
| Sorting Files in the Extract Directory | 63 |
| Extracting Files Only for Display | 63 |
| Extracting Files with a List File | 63 |
| Digital Signatures | 64 |
| 5 MISCELLANEOUS OPERATIONS | 66 |
| Introduction..... | 66 |
| Overwriting Files | 66 |
| Viewing the Contents of a .ZIP File..... | 67 |
| Displaying a Brief View of a .ZIP File..... | 67 |
| Displaying a Detailed View of the .ZIP File..... | 67 |
| Printing the Contents of a .ZIP File..... | 68 |
| Testing the Integrity of a .ZIP File | 69 |
| Checking for Revoked Certificates | 69 |
| Obtaining a CRL | 70 |
| Pausing on Warnings..... | 70 |
| Treating Warnings as Errors | 71 |
| Previewing Command and Option Operations..... | 72 |
| Fixing a Corrupt .ZIP File | 72 |
| Create a Temporary .ZIP File on a Alternate Drive..... | 73 |
| Suppressing Screen Output..... | 73 |
| Setting Internal Attributes | 74 |
| Encoding an Archive to Another Type..... | 74 |
| Removing an Intermediate Archive | 75 |
| Generate a List File | 75 |
| 6 CHANGING DEFAULTS FOR COMMANDS AND OPTIONS..... | 76 |
| Viewing the Configuration File..... | 76 |
| Changing a Default Value | 77 |
| Changing Defaults for Filter Options..... | 78 |
| Changing Defaults for Compression Method..... | 78 |
| Using the Options Dialog to Change Defaults | 79 |

| | |
|---|------------|
| Resetting to Original Defaults | 80 |
| Resetting Individual Defaults | 80 |
| Resetting All Defaults..... | 81 |
| Using an Alternate Configuration File | 81 |
| Creating an Alternate Configuration File..... | 81 |
| Using an Alternate Configuration File | 82 |
| 7 COMMAND CHARACTERISTICS | 83 |
| Introduction | 83 |
| Changing Date and Time Environment Variables | 83 |
| Changing the List Character for List Files | 84 |
| Changing the Command/Option Character | 84 |
| A REFERENCE TO COMMANDS AND OPTIONS | 86 |
| B ERROR AND WARNING MESSAGES | 118 |
| Error Messages | 118 |
| Warning Messages | 123 |
| C FREQUENTLY ASKED QUESTIONS | 128 |
| D HOW DOES PKZIP WORK | 132 |
| Two Processes | 132 |
| Compression | 132 |
| Information Content | 132 |
| Binary Data Representation..... | 133 |
| Speed vs. Size | 136 |
| Archiving | 136 |
| How PKZIP builds a .ZIP File | 136 |
| CRC | 138 |
| Deleting Files from a .ZIP File | 139 |
| Adding to an Existing .ZIP File | 139 |
| INDEX | 140 |

1

Welcome

Introduction

Welcome to *PKZIP for Windows Command Line*. *PKZIP for Windows Command Line* provides a command-line interface to PKZIP for use in creating scripts and batch files. With *PKZIP Command Line*, you execute PKZIP commands and options by entering them at a character-based command prompt and running the resulting command line.

PKZIP Command Line supports the full range of compression and archiving features of the graphical PKZIP for Windows. *PKZIP Command Line* also supports both traditional ZIP encryption and password-based strong encryption. *PKZIP Command Line* decrypts files encrypted with any other version of PKZIP, including files encrypted using a digital certificate, and authenticates digital signatures attached to archives and archived files (see the **extract** and **test** commands.)

About This Manual

This manual describes how to use the various commands and options of *PKZIP Command Line*. Chapter 2 includes brief tutorials and provides an overview of the program functionality described in more detail in the later chapters. See in particular the section “Understanding PKZIP Commands and Options” for an explanation of how commands differ from options.

You can customize just about everything about how *PKZIP Command Line* behaves. Chapter 6 describes how to do this by setting your own default values for commands and options.

Appendix A contains a complete reference to the commands and options of the program. After you become generally familiar with the way that *PKZIP Command Line* operates, you may find most of the information you need in this appendix.

From now on, references to *PKZIP* should generally be understood to mean *PKZIP for Windows Command Line*.

Your Work Environment: The Command Line

In PKZIP *Command Line*, your work area is a character-based command line. You enter a command by typing the command on the command line; to execute the command, you press **Enter**.

To display a command line prompt in Windows, do one of the following:

- Choose **Command Prompt** from the list of programs in the Start menu
- Choose **Run...** from the Start menu, enter `cmd` in the field, and choose **OK**.

Entering Commands

The syntax for commands entered on the command line is shown below. Brackets set off elements that are optional. (Do not type the brackets.)

```
pkzipc [command] [options] zipfile [@list] [files...]
```

Examples:

| <i>To do this</i> | <i>Command line</i> |
|---|---|
| Add specified files to an archive | <code>pkzipc -add zipfile.zip addfile.txt addfile2.doc</code> |
| Add to an archive all files in current directory | <code>pkzipc -add zipfile.zip</code> or: <code>pkzipc -add zipfile.zip *</code> |
| Add to an archive all files in a specified directory | <code>pkzipc -add zipfile.zip subdir*</code> |
| Add files with the fast compression option | <code>pkzipc -add -fast zipfile.zip</code> |
| View list of files in archive | <code>pkzipc zipfile.zip</code> |
| View list of files whose names begin with "f" in archive | <code>pkzipc zipfile.zip f*</code> |
| Extract all files from an archive | <code>pkzipc -extract zipfile.zip</code> |
| Extract specified files from an archive | <code>pkzipc -extract zipfile.zip readme.txt mystuff.doc</code> |

Setting PKZIP in the Path (Windows)

The installation puts PKZIP on your system's search path so that you can access the program from any directory without specifying a path. However, if for any reason you need to specify the path yourself, you can.

The search path in Windows is normally specified in the `autoexec.bat` file, which is typically located in the root directory (`C:\`). To add the PKZIP installation directory to your search path, follow the steps in the appropriate section below.

Windows 98/Me

1. Open a Windows Command Prompt window.
2. Type the following:

```
edit c:\autoexec.bat
```

The contents of your `autoexec.bat` file will appear in a text editor. Find the line in your `autoexec.bat` that starts with the `PATH` command followed by paths separated by semicolons. For example:

```
PATH C:\DOS;C:\WINDOWS;C:\NETWORK;C:\NAV
```

3. At the end of this line, add a semicolon (if one is not there already) and the path, in quotes, to the folder where PKZIP *Command Line* is installed. (The quotes are necessary because the path contains a space.)

For example, assuming that PKZIP *Command Line* (`pkzipc.exe`) is installed in the default location, enter:

```
;"c:\program files\pkware\pkzipc"
```

4. Press `Alt+F+X`. You will be prompted to save the file before exiting. Enter `Y` to save.
5. Close any open applications and restart your PC

You can now access PKZIP *Command Line* from any directory without specifying a path.

Windows NT/Windows 2000/Windows XP

1. Close any open Command Prompt windows.
2. Go to "My Computer" on your desktop and right-click the My Computer icon.
3. Select Settings | Control Panel from the Start Menu.
4. In the Control Panel, double click the System icon. The System (Properties) dialog appears.

5. If you are using NT, select the Environment tab. If you are using XP, click the Advanced tab and then click the Environmental Variables button.
6. Select the PATH variable in the System (Environment) Variables or User (Environment) Variables boxes. If you are unable to locate the PATH variable, enter the following in the Variable box:

path

7. In the Value box, enter (in quotes) the path to the folder where *PKZIP Command Line* is installed. (The quotes are necessary because the path contains a space.)

For example, assuming that *PKZIP Command Line* (`pkzipc.exe`) is installed in the default location, enter:

"c:\program files\pkware\pkzipc"

If necessary to separate the path from another path designation, precede your path with a semicolon.

8. Click the **Set** button.
9. Click the **OK** button.

You may now access *PKZIP Command Line* from any directory without specifying a path. This change will take effect the next time you open an Command Prompt Window to run *PKZIP Command Line*.

If necessary, consult your systems administrator for further information on setting the path environment variable.

Strong Encryption

PKZIP enables you to use either of two kinds of encryption to encrypt files: the older, traditional PKZIP encryption, or strong encryption. Strong encryption is much more secure than traditional PKZIP encryption.

In *PKZIP Command Line*, you use the **password** option to do encryption of either kind. For strong encryption, you additionally use the **cryptalgorithm** option.

People who receive archives strongly encrypted using *PKZIP Command Line* can decrypt them using PKZIP for any platform (version 6 or later) or a copy of the free ZIP Reader by PKWARE.

Creating the Tutorial Directory and Files

The PKZIP installation directory contains a batch file that creates a directory and several practice files for you to work with. These files enable you to do the tutorials in Chapter 2 without accidentally changing or deleting any of your other files. See the section below, "Setting Up the Tutorial in Windows."

Setting Up the Tutorial in Windows

To create the tutorial directory and files in Windows:

1. From a Windows Command Prompt window, change to the directory where you installed PKZIP (C:\program files\pkware\pkzipc by default).
2. Run the `tutorial.bat` batch file: Type the following at the command prompt and press ENTER:

tutorial.bat

The batch file creates a new subdirectory called *tut* in the install directory and copies several practice files. It displays lines on screen like this:

```
.  
. .  
Creating the Tut Directory  
Copying the test files  
Done.
```

3. Change to the `tut` directory created in the previous step. To do so, type the following and press ENTER:

cd tut

4. To confirm that the files were created, type the following and press ENTER:

dir

You should see a list of files similar to the following:

```
07/10/2003 12:19 PM          9,108 black.tut  
07/10/2003 12:19 PM          9,108 blue.fil  
07/10/2003 12:19 PM          9,108 brown.doc  
07/10/2003 12:19 PM          9,108 gold.tut  
07/10/2003 12:19 PM          9,108 green.doc  
07/10/2003 12:19 PM          9,108 orange.fil  
07/10/2003 12:19 PM          9,108 pink.tut  
07/10/2003 12:19 PM          9,108 purple.txt  
07/10/2003 12:19 PM          9,108 red.txt  
07/10/2003 12:19 PM          9,108 tan.txt  
07/10/2003 12:19 PM          9,108 white.doc  
07/10/2003 12:19 PM          9,108 yellow.doc
```

Using Help

Besides the manual you are now reading, PKZIP provides online help for the PKZIP commands and options. The online help describes syntax and shows sample command lines.

You access the online help directly from the command line:

- At the command prompt, type the following and press ENTER:

pkzipc -help

A screen with PKZIP version and usage information appears. You can get help for any PKZIP command or option from here.

- To bypass the command/option menu and go directly to a help file for a particular command or option, type the **help** command followed by an equal sign (=) and the command or option for which you want information.

For example, to access online help for the **add** command, type the following at the command prompt and press ENTER:

```
pkzipc -help=add
```

The help information for the **add** command appears.

Getting License Information

To display the PKZIP license information on your screen, do the following:

- At the command prompt, type the following and press ENTER:

```
pkzipc -license
```

If you have the registered version of PKZIP, the screen displays licensing information.

Getting Version Information

To list the version of PKZIP that you are using, do the following:

- At the command prompt, type the following and press ENTER:

```
pkzipc -version
```

Note: Advanced users of PKZIP may require more detailed version information to assist in automating compression and extraction operations. PKZIP can return the version number as a value to the shell. The version number displays as a positive integer value less than 256. This value is only returned to the shell and is therefore not viewable. This functionality can be useful to verify PKZIP version numbers in the context of a .BAT file or shell script. The default action is to return the major number of the release. For example, for version 2.5 the return value will be 2. You can use sub-options with the **version** option to return precise values of the release number. The available sub-options and the values they return are shown in the following table:

| <i>Sub-Option:</i> | <i>PKZIP Returns:</i> | <i>For example:</i> |
|--------------------|--|-----------------------|
| major | The major release number. For example, if the version number is 2.5, the value returned is <2>. | pkzipc -version=major |
| minor | The minor number of the release. For example, if the version number is 2.5, the value returned is <5>. | pkzipc -version=minor |
| step | The step or patch value. For example, if the version is 2.04.01, the step value returned is <1>. | pkzipc -version=step |

Technical Support

For support, visit our Web site at:

www.pkware.com/support

2

The Basics

Introduction

This chapter shows you how to use PKZIP to do basic compression and extraction operations. It also introduces some command line options and conventions. For example, you will learn how to specify the directory that you want to extract files to and how to specify just which files you want to compress or extract.

PKZIP gives you numerous commands and options that enable you to customize the operations that you use PKZIP to do. Some of these commands and options are discussed in this chapter. The rest are explained in later chapters. The present chapter gets you started. Its tutorials introduce you to major features of PKZIP and enable you to practice using PKZIP on files in a special tutorial directory. See the section, “Creating the Tutorial Directory and Files,” in Chapter 1 to learn how to set up your tutorial workspace.

An Overview of What PKZIP Does

PKZIP does four main sorts of things: It compresses files, and it decompresses them to restore them to their original state. It also authenticates and encrypts files. All the various commands and options relate to how you do these things or to other things that you can optionally do in the process. For example, you have options for picking the files that you want to compress or encrypt and options for how you want to compress or encrypt them.

Supported Archive Types

When PKZIP compresses files, it adds the compressed files to an *archive*. An archive is a kind of file that can contain other files. When PKZIP decompresses files, it extracts them from the archive that contains them. This is why the commands to compress and decompress files are named ***add*** and ***extract***.

Several types of archive files exist. Some can contain only one file, some can contain multiple files, and there can be other differences as well. A ZIP archive can contain multiple compressed files. This is the kind of archive that PKZIP creates by default and the kind that you will probably use most often.

PKZIP enables you to create and extract from other archive types besides ZIP. You do not need to do anything special to use PKZIP with one of these other archive types. PKZIP can tell what type an archive is and will just go ahead and extract its files. If you want to create a new, non-ZIP archive, there are two ways that you can tell PKZIP what type of archive to create:

- Specify a name for the archive file that uses the file name extension customarily associated with that archive type
- Use the ***archivetype*** option to specify the type of archive that you want

The following table lists the types of archives that PKZIP can create or extract from and the file name extensions customarily associated with these types. For some archive types, PKZIP can do extractions but cannot create new archives of that type.

| <i>Archive type</i> | <i>PKZIP can create/extract</i> | <i>Usual file name extensions</i> |
|---------------------|---------------------------------|-----------------------------------|
| ARJ | Extract only | .arj |
| BinHex | Extract only | .hqx |
| BZIP2 | Create and extract | .bz2 |
| CAB | Extract only | .cab |
| GZIP | Create and extract | .gz |
| LZH | Extract only | .lzh |
| RAR | Extract only | .rar |
| TAR | Create and extract | .tar |
| UUEncoded | Create and extract | .uue |
| XXEncoded | Create and extract | .xxe |
| ZIP | Create and extract | .zip, .jar |

Using the Tutorials

To help you learn PKZIP, this chapter contains several tutorials. These tutorials progress from simple to complex features. Each tutorial is preceded by a brief explanation of the topic(s) that the tutorial represents. For each tutorial, you will be asked to perform a specific PKZIP task. The results of what you type are reviewed every step of the way.

Using Sample Files and Directories

The tutorials in this manual take you through compressing and extracting "specific" files from a "specific" tutorial directory. Feel free to use your own files if you choose to practice on them. However, keep in mind that the results that are displayed are specific to the files that these tutorials use as samples.

Note: These tutorials assume that you have installed PKZIP in the default installation directory and that you can access PKZIP from any directory without specifying a file path.

Preparing Your Workspace

To follow the tutorials in this manual step-by-step, you must create a working directory. Consider it a temporary directory to be used only for the tutorials. This helps to ensure that your permanent directories and files are not deleted or damaged while you are practicing with PKZIP. See the section, “Creating the Tutorial Directory and Files,” in Chapter 1 to learn how to set up your tutorial workspace.

Compressing Files into an Archive: The Basics

To compress one or more files into an archive file, you enter a command line that has a minimum of four parts. The parts are listed below. You enter them at your system’s command prompt in the following order:

- The command ***pkzipc***, which indicates the program to run, namely, PKZIP
- A hyphen (-) followed by the word ***add***, which is the PKZIP command to compress files and add them to an archive
- The name of the archive file to receive the files
- The names of the files to compress

For example, to compress a file called test.txt into an archive file called temp.zip, type the following:

```
pkzipc -add temp.zip test.txt
```

You can also encrypt files when you compress them. See “Encrypting Files That You Add to an Archive” in Chapter 3.

In the tutorials that follow, you will learn about other commands and options that you can use and about other ways to specify files to operate on.

Archive File Naming Conventions

Conventionally, archive files (or *archives* for short) are given a name with a *file (name) extension* (the last part of the name, after the dot) that indicates the kind of archive they are. Thus a .ZIP archive generally has a name of the form *myarchive.zip*, where the file extension is *.zip*. Similarly, a BZIP2 archive generally has a file extension of *.bz2*.

Different archive types are used for different things. For example, a BZIP2 archive is used to contain a file compressed using the BZIP2 compression algorithm.

PKZIP can both create and extract from a variety of archive types—including BZIP2. Because the file name extension is generally a good guide to the type of archive, PKZIP sometimes uses this information to determine what sort of archive you want to

create. Here are the rules PKZIP follows with respect to archive types and names when you use the **add** command create an archive:

- If you specify an archive with an extension—for example, *myarchive.zip* or *myarchive.bz2*, or *myarchive.exe*, PKZIP will create an archive of that name. Also, by default, PKZIP will use the file extension to select the type of compression to use. For example,

```
pkzipc -add myarchive.zip
```

results in a ZIP-format archive containing files compressed using standard ZIP-style compression (that is, using the deflate compression algorithm).

- If you specify an archive with *no* file extension, by default PKZIP creates a ZIP archive and adds a *.zip* extension to its name. For example:

```
pkzipc -add myarchive
```

produces a ZIP archive called *myarchive.zip*.

Note: The **archivetype** option lets you explicitly tell PKZIP the type of archive you want to create. See “Compressing Files Compatible with the Data Compression Library” in Chapter 3.

- If you specify an archive that has no file extension but does have a *trailing dot*—that is, a dot as the last character in the file name: for example, “*filename.*”—PKZIP does *not* append an extension to the file name. For example:

```
pkzipc -add myarchive.
```

produces (by default) a ZIP archive called *myarchive* without an extension.

Note: Systems that do not support more than one “dot” in a file name suppress the extension if any dot is present in the file name, even if it is not a trailing dot.

Note: The **noarchiveextension** option suppresses automatic adding of a file name extension on all systems.

Compressing a Single File in the Current Directory

The simplest PKZIP task is compressing a single file from the current working directory, and storing the resulting .ZIP file in that same directory. In this scenario, you are not retrieving files to compress from other directories, nor are you placing the .ZIP file in a different directory.

Tutorial A

Compress a single file (*red.txt*) into an archive file in your current working directory.

Follow these steps:

1. From the command prompt, change to the PKZIP Tutorial (*tut*) directory that you created (see “Creating the Tutorial Directory and Files” in Chapter 1). To

compress the file red.txt into the .ZIP file called test.zip, type the following command and press ENTER:

```
pkzipc -add test.zip red.txt
```

Your file starts to compress, and your screen looks like the following:

```
Creating .ZIP: test.zip
Adding File: red.txt      Deflating    (62.1%), done.
```

The first three lines contain the PKZIP banner information (name of the product, date, version and so on).

The last two lines list the name of the .ZIP file you created, the file(s) that were compressed, and the percentage that the file was compressed.

You have now successfully compressed a file into an archive file. Before you continue with the next section, let's take a look at your **tut** directory.

- View the contents of the *tut* directory by typing the following command:

```
dir
```

A file listing similar to the following appears:

```
.          <DIR>          06-01-01 12:00a .
..         <DIR>          06-01-01 12:00a ..
ORANGE    FIL           561 06-01-01  4:50a orange. fil
YELLOW    DOC           8,369 06-01-01  4:50a yell ow. doc
RED       TXT           8,369 06-01-01  4:50a red. txt
GREEN     DOC           591 06-01-01  4:50a green. doc
PURPLE    TXT           591 06-01-01  4:50a purpl e. txt
WHITE     DOC           8,369 06-01-01  4:50a whi te. doc
BROWN     DOC           8,369 06-01-01  4:50a brow n. doc
PINK      TUT           30,155 06-01-01  4:50a pi nk. tut
TEST      ZIP            1,702 06-01-01  4:50a test. zi p
TAN       TXT           8,369 06-01-01  4:50a tan. txt
BLACK     TUT           30,155 06-01-01  4:50a bl ack. tut
BLUE     FIL           8,369 06-01-01  4:50a bl ue. fil
GOLD      TUT           30,155 06-01-01  4:50a gol d. tut
13 file(s)          36,880 bytes
2 dir(s)            87,945,216 bytes free
```

Note the test.zip file you have created. The file you compressed into the .ZIP file still appears in the current directory. That is because when you compress files, they remain in their original location, as well as inside the .ZIP file, compressed.

You can choose to remove or "move" the files into the .ZIP file so that they exist only in the .ZIP file after compression. The *Moving Files Into Your .ZIP File* section on page 15 shows you how to do this.

Compressing Selected Files in the Current Directory

With PKZIP, you can compress specific selected files into your .ZIP file. To do this, you simply type the files in your command, including a space between each one.

Tutorial B

Compress the following files into the test.zip file:

- green.doc
- blue.fil

Follow these steps:

- From the command prompt, type the following and press ENTER:

```
pkzipc -add test.zip green.doc blue.fil
```

Your file starts to compress, and output similar to following appears:

```
Updating .ZIP: test.zip
Adding File: green.doc      Deflating   (39.3%), done.
Adding File: blue.fil      Deflating   (62.1%), done.
```

This particular example added the specified files to the existing test.zip file. As you can see in the screen output above, PKZIP reports that it is "Updating .ZIP: test.zip" the .ZIP archive now contains any file(s) we added in the previous tutorial as well as the file(s) we added in this tutorial.

Compressing Files that Match a Pattern

PKZIP allows you to compress "only" files of a specific file pattern or extension. For example, you might want to compress only .doc files or .bmp files. On the other hand, you might want to compress all files that begin with the letter "p" or any other character match.

Using PKZIP conventions for typing file patterns (commonly called "wildcards"), follow these guidelines:

- For files of a specific extension, type an asterisk, a period, and the extension. You would type ***.txt** for ASCII text files (.txt extension).
- For files that begin with a specific character or characters, type the character(s), then an asterisk (*). You would type **res*** for all files that begin with res.
- For files that end with a specific character pattern, type an asterisk (*), then the character(s). For example, for all files that end in "all" you would type ***all**.

Note: For other "wildcard" conventions and general command-line conventions, refer to your Operating System documentation.

Tutorial C

Compress all files in your tut directory that end with ".doc"

Follow these steps:

- From your command prompt, type the following and press ENTER:

```
pkzipc -add test.zip *.doc
```

The following appears on your screen:

```
Updating .ZIP: test.zip
```

```

Updating File: green.doc    Deflating    (39.3%), done.
Adding File: yellow.doc    Deflating    (59.4%), done.
Adding File: white.doc     Deflating    (59.4%), done.
Adding File: brown.doc     Deflating    (59.4%), done.

```

When PKZIP is finished, the command prompt reappears.

Compressing All Files in the Current Directory

PKZIP allows you to compress all files in a specified directory. In the previous tutorials you compressed files by specifying either a specific file name (e.g., red.txt) or file pattern (e.g., *.doc). When you compress "all" files, you only have to type the name of the .ZIP file with the **add** command. PKZIP will assume that you wish to compress all files in the current directory. If, however, you wish to specify "all" files in another directory, you must use the convention for specifying "all" files (e.g., "*").

Tutorial D

Compress all the files in your tut directory into the test.zip file.

Follow these steps:

- From your command prompt, type the following and press ENTER:

```
pkzipc -add test.zip
```

Your files start to compress. Because you are compressing all of the files in your tut directory, this will take a little longer.

Output similar to the following appears:

```

Updating .ZIP: test.zip
Updating File: red.txt      Deflating    (62.1%), done.
Updating File: green.doc   Deflating    (39.3%), done.
Updating File: blue.fil    Deflating    (62.1%), done.
Updating File: yellow.doc  Deflating    (59.4%), done.
Updating File: white.doc   Deflating    (59.4%), done.
Updating File: brown.doc   Deflating    (59.4%), done.
Adding File: tan.txt       Deflating    (62.1%), done.
Adding File: orange.fil    Deflating    (39.3%), done.
Adding File: black.tut     Deflating    (61.0%), done.
Adding File: purple.txt    Deflating    (39.3%), done.
Adding File: pink.tut      Deflating    (61.0%), done.
Adding File: gold.tut      Deflating    (61.0%), done.

```

The phrase *Updating File* appears in front of some files, while *Adding File* appears in front of others. This is because in this tutorial you compressed files that were already compressed in the previous tutorials. The phrase *Updating File* appears with those particular files.

Compressing Files from a Different Location

When you compress files, you do not have to be in the directory where those files reside. You can compress them from anywhere on your computer. You just need to specify a path to any files that are not in the current directory so that PKZIP can find them.

Tutorial E

In this tutorial, you will switch locations to the directory that is directly above (parent to) the *Command Line* install directory and compress files that appear in the tutorial directory.

Follow these steps:

1. Go to the directory that is directly above the PKZIP install directory (for example, `c:\program files\pkware` or `/usr/local/pkware`).
2. To compress the file called `purple.txt` from the PKZIP tutorial directory into a file called `test123.zip` file and place the `test123.zip` file into the current directory, type the following command line and press ENTER:

```
pkzipc -add test123.zip pkzipc\tut\purple.txt
```

Your files start to compress, and the following appears:

```
Creating .ZIP: test123.zip
Adding File: purple.txt Deflating (39.3%), done.
```

The `test123.zip` file is created in the current directory as that is the directory in which you typed the command. You can also specify a different location in which to create or update an archive file. The next section shows you how to do that.

Specifying a Different Location for the .ZIP File

Until now, you have created an archive file in the current directory. With PKZIP, you can specify a "different" location for the .ZIP file right in the command line. Simply include the location, or directory path, in your command.

Tutorial F

While remaining in the directory that is parent to the PKZIP install directory, compress the file called `gold.tut`, which is in PKZIP Tutorial directory, and place it in the `test.zip` file, which exists in PKZIP Tutorial directory as well.

Follow these steps:

- To compress `gold.tut` into the `test.zip` file, and place the updated .ZIP file in the PKZIP Tutorial directory, type the following command line and press ENTER:

```
pkzipc -add pkzipc\tut\test.zip pkzipc\tut\gold.tut
```

Your files start to compress, and a screen similar to the following appears:

```
Updating .ZIP: pkzipc/tut/test.zip
Updating File: gold.tut Deflating (61.0%), done.
```

Moving Files into Your .ZIP File

Normally, the original files that you compress remain on your hard drive after you compress them. That is the default action when you use the **add** command. PKZIP allows you to *remove* the files from your hard drive location *after* they are compressed into the .ZIP file.

CAUTION: Be careful when "moving" files from your hard drive to a .ZIP archive. If the .ZIP archive is subsequently damaged, lost, or deleted, its files will be damaged, lost, or deleted with it. Regularly back up your .ZIP archives so that you have a copy.

To remove files after they are compressed, use the **move** option along with the **add** command. To include an option in your PKZIP command, you simply type that option - preceded by a dash - after the **add** command. The following tutorial shows you how to do this.

Note: For basic information on PKZIP options, refer to the *Understanding PKZIP Commands and Options* section on page 22.

Tutorial G

In this tutorial, you are going to compress the file called pink.tut into the test.zip file, and "remove" pink.tut from your tut directory.

Follow these steps:

- Change back to the PKZIP Tutorial directory.

Type the following and press ENTER:

```
pkzipc -add -move test.zip pink.tut
```

Your file starts to compress, and your screen looks similar to the following:

```
Updating .ZIP: test.zip
Updating File: pink.tut      Deflating   (61.0%), done.

Moving 1 files...
Moved File: pink.tut      , done.
```

Note that PKZIP performed two tasks: Updating and Moving. After compressing the file, PKZIP removed it from the tut directory.

- To verify that the file you compressed was removed from your directory, type the following command and press ENTER:

```
dir
```

Verify from the listing that the pink.tut file no longer exists as an individual file in the PKZIP Tutorial directory. The pink.tut file does however exist, compressed, in the test.zip file.

Viewing Files Within a .ZIP File

In the previous section, you learned how to move a file from the hard drive into a .ZIP archive. The file you moved still exists in your test.zip file. You can use the **view** command to verify that this file was indeed archived in the .ZIP file. The **view** command allows you to list the files that exist within a .ZIP file. Because you are not compressing files - only viewing them - you do not have to include the **add** command.

Tutorial H

View the files within the test.zip file.

Follow these steps:

- Verify that you are in the PKZIP tutorial directory.

To view the files within the test.zip file, type the following and press ENTER:

```
pkzipc -view test.zip
```

PKZIP displays a file listing of all of the files contained in the test.zip file. The listing will look similar to the following:

```
Viewing .ZIP: test.zip
```

| Length | Method | Size | Ratio | Date | Time | CRC-32 | Attr | Name |
|--------|----------|-------|-------|------------|-------|----------|-------|------------|
| 8369B | Defl atN | 3084B | 63.2% | 06/01/2001 | 4:50a | 87b3c388 | -a-w- | red.txt |
| 561B | Defl atN | 340B | 39.4% | 06/01/2001 | 4:50a | b6a63b7f | -a-w- | green.doc |
| 8369B | Defl atN | 3084B | 63.2% | 06/01/2001 | 4:50a | 87b3c388 | -a-w- | blue.fil |
| 8369B | Defl atN | 3084B | 63.2% | 06/01/2001 | 4:50a | 87b3c388 | -a-w- | brown.doc |
| 8369B | Defl atN | 3084B | 63.2% | 06/01/2001 | 4:50a | 87b3c388 | -a-w- | white.doc |
| 8369B | Defl atN | 3084B | 63.2% | 06/01/2001 | 4:50a | 87b3c388 | -a-w- | yellow.doc |
| 29KB | Defl atN | 8130B | 73.1% | 06/01/2001 | 4:50a | faf7aa7a | -a-w- | black.tut |
| 29KB | Defl atN | 8130B | 73.1% | 06/01/2001 | 4:50a | faf7aa7a | -a-w- | gold.tut |
| 561B | Defl atN | 340B | 39.4% | 06/01/2001 | 4:50a | b6a63b7f | -a-w- | orange.fil |
| 29KB | Defl atN | 8130B | 73.1% | 06/01/2001 | 4:50a | faf7aa7a | -a-w- | pink.tut |
| 561B | Defl atN | 340B | 39.4% | 06/01/2001 | 4:50a | b6a63b7f | -a-w- | purple.txt |
| 8369B | Defl atN | 3084B | 63.2% | 06/01/2001 | 4:50a | 87b3c388 | -a-w- | tan.txt |
| ----- | | ----- | ----- | | | | | ----- |
| 139KB | | 42KB | 69.2% | | | | | 12 |

For more information on the **view** command, see the *Viewing the Contents of a .ZIP File* section on page 67.

Further Information on Compressing Files

The tutorials in the previous sections presented some of the file compression options available to you. There are other options as well. For example, you can choose to compress only files that are newer than the files already stored in the .ZIP file, or you can choose to compress only files that were created after a certain date.

For information on these and other compression features, refer to Chapter 3.

Extracting Files: Learning the Basics

With PKZIP, you can "extract" files just as easily as you can compress them. This section shows you how to "extract" files back to their original size. You can extract all files from a .ZIP file, or just one file. You can also extract only the files that are newer than the files with the same name on your hard drive.

When you extract a file, you have to include at least the following in your command, including a space between each element:

- The word **pkzipc**, which is the main PKZIP command.
- A dash (-) followed by the word **extract** (which is the base command used to extract files).
- The name of the .ZIP file.

Including these three elements extracts "all" the files contained in the .ZIP file into the current directory. To extract only specific files from the .ZIP file, you would include the following:

- The name of the file(s) you want to extract, or a file pattern. For example, to extract a file called cover.txt from a compressed file called temp.zip into the current directory, you would type the following and press ENTER:

```
pkzipc -extract temp.zip cover.txt
```

The tutorials that follow cover several "extraction" scenarios. For complete information on extracting files, refer to Chapter 4.

Extracting a Single File From a .ZIP File

The simplest task for extracting files is to extract a single file into the current directory. To make it easy, we are going to create an "extract" directory under your tut directory. We will call this directory ext. You will also copy the test.zip file into this directory.

Before you extract files, create a directory called ext and copy the test.zip file into that directory.

Follow these steps:

1. Verify that you are in the PKZIP Tutorial directory.
2. To create the ext directory, type the following and press ENTER:

```
mkdir ext
```
3. To copy the test.zip file into the ext directory, type the following and press ENTER:

```
copy test.zip ext
```
4. Now change to the ext directory. To do so, type the following and press ENTER:

```
cd ext
```

Note: The **cd** command in the preceding line stands for "change directory." It changes the current directory to a directory that you specify on the command line. Many of the following tutorials use this command. The command is an operating system command, not a PKZIP command. Consult your operating system documentation to learn more about the **cd** command.

Tutorial I

Extract the file called red.txt from the test.zip file.

Follow these steps:

1. Verify that you are in the ext directory.
2. Type the following and press ENTER:

```
pkzipc -extract test.zip red.txt
```

PKZIP extracts the file and displays a message similar to the following:

```
Extracting files from .ZIP: test.zip
Inflating: red.txt
```

The first three lines contain the PKZIP banner information (name of the product, date, version and so on).

The fourth line contains the task PKZIP performed (in this case, *Extracting*) followed by the name of the .ZIP file (in this case, test.zip).

The fifth line contains the word *Inflating*, then the name of the file being extracted (in this case, red.txt). *Inflating* is an internal method of extraction that is meaningful only to the program.

3. To confirm that red.txt has been extracted into the ext directory, type the following command and press ENTER:

```
dir red.txt
```

A listing similar to the following appears:

```
RED      TXT      8,369 06-30-97 2:50a red.txt
          1 file(s)      8,369 bytes
          0 dir(s)    88,793,088 bytes free
```

Note that red.txt has been extracted into the ext directory.

Extracting Multiple Files From a .ZIP File

PKZIP allows you to extract multiple files from a .ZIP file at one time. This section shows you how to extract:

- Selected individual files.
- Files that match a specific file pattern. For example, files ending with a .txt extension.
- All files in the directory.

To prepare your ext directory to extract multiple files, first remove the red.txt file that you extracted in the previous tutorial.

Follow these steps:

1. Verify that you are in the ext directory.
2. Type the following command and press ENTER:

```
del red.txt
```

The red.txt file is deleted. The only file that should be in the ext directory is test.zip.

Extracting Selected Files

With PKZIP, you can extract specific selected files from your .ZIP file. To do this, you simply type the files in your command, including a space between each one.

Tutorial J

Extract the following files from the test.zip file:

- green.doc
- blue.fil

Follow these steps:

1. Verify that you are in the ext directory.
2. Type the following and press ENTER:

```
pkzipc -extract test.zip green.doc blue.fil
```

A screen similar to the following appears:

```
Extracting files from .ZIP: test.zip
Inflating: blue.fil
Inflating: green.doc
```

3. To confirm that the files were extracted, type the following command and press ENTER:

```
dir
```

A listing similar to the following appears:

```
.          <DIR>          06-30-97  4:00a .
..         <DIR>          06-30-97  4:00a ..
TEST      ZIP           45,488  06-30-97  3:30a test.zip
GREEN     DOC             561    06-30-97  2:50a green.doc
BLUE     FIL           8,369  06-30-97  2:50a blue.fil
          3 file(s)          54,418 bytes
          2 dir(s)       88,788,992 bytes free
```

Extracting All Files From a .ZIP File

You can extract all of the files from your .ZIP file. When you extract specific files, as before, you must specify those files or file pattern. When you extract "all" files, you only have to type the name of the .ZIP file with the **extract** command. You can also, however, use the convention for specifying "all" files (*).

Tutorial K

Extract all the files from the test.zip file into the ext directory.

Follow these steps:

1. Verify that you are in the ext directory.

- To extract all files from the test.zip file, type the following and press ENTER:

```
pkzipc -extract test.zip
```

Because you are extracting files that already exist in your ext directory, PKZIP asks you if you want to overwrite the file by displaying the following prompt:

```
Overwrite (<Y>es/<N>o/<A>ll/ne<V>er/<R>ename/<Esc>)?
```

- Type "y" for yes if you wish to overwrite the file.

The same prompt will appear for the other files that already exist in the EXT directory. If you wish to overwrite those files as well type "y".

PKZIP extracts the files, and the screen will look similar to the following:

```
Extracting files from .ZIP: test.zip
Inflating: black.tut
PKZIP: (W7) Warning! file: blue.fil already exists.
Overwrite (<Y>es/<N>o/<A>ll/ne<V>er/<R>ename/<Esc>)? y
Inflating: blue.fil
Inflating: brown.doc
Inflating: gold.tut
PKZIP: (W7) Warning! file: green.doc already exists.
Overwrite (<Y>es/<N>o/<A>ll/ne<V>er/<R>ename/<Esc>)? y
Inflating: green.doc
Inflating: orange.fil
Inflating: pink.tut
Inflating: purple.txt
Inflating: red.txt
Inflating: tan.txt
Inflating: white.doc
Inflating: yellow.doc
```

- To confirm that the files were extracted into your ext directory, type the following command and press ENTER:

```
dir
```

A listing similar to the following appears:

```
.          <DIR>          06-01-01 12:00a .
..         <DIR>          06-01-01 12:00a ..
ORANGE    FIL           561 06-01-01 4:50a orange.fil
YELLOW    DOC           8,369 06-01-01 4:50a yellow.doc
RED       TXT           8,369 06-01-01 4:50a red.txt
GREEN     DOC           591 06-01-01 4:50a green.doc
PURPLE    TXT           591 06-01-01 4:50a purple.txt
WHITE     DOC           8,369 06-01-01 4:50a white.doc
BROWN     DOC           8,369 06-01-01 4:50a brown.doc
PINK      TUT          30,155 06-01-01 4:50a pink.tut
TEST      ZIP          45,488 06-01-01 4:50a test.zip
TAN       TXT           8,369 06-01-01 4:50a tan.txt
BLACK     TUT          30,155 06-01-01 4:50a black.tut
BLUE     FIL           8,369 06-01-01 4:50a blue.fil
GOLD     TUT          30,155 06-01-01 4:50a gold.tut
13 file(s)          187,850 bytes
 2 dir(s)           87,945,216 bytes free
```

You have now extracted all of the files contained in the test.zip file.

Overwriting Files that Already Exist in Your Directory

In the previous tutorial, you saw what appears when you extract files into a directory that contains file(s) with the same name. PKZIP offers several choices for handling

the overwriting of files while extracting. Refer to Chapter 4 for complete information on extracting files from a .ZIP file.

Extracting Files to a Specified Directory

The extraction tutorials up to this point had you extract the contents of the test.zip file into the current directory. With PKZIP, you can specify a "different" extract location for the .ZIP file - right in your command. Simply include the location, or directory path.

Tutorial L

Extract the file called red.txt, archived in the test.zip file, into the directory above (parent to) the tutorial directory.

Follow these steps:

1. Verify that you are in the tut directory. If you are still in the ext directory, change to the tut directory.
2. To extract the red.txt file archived in the test.zip file into the tutorial directory's parent directory, type the following command line and press ENTER:

```
pkzipc -extract test.zip red.txt ..
```

The two periods (..) at the end of the command line tell PKZIP to extract the file into the directory up one level in the directory structure. PKZIP runs, and the following appears:

```
Extracting files from .ZIP: test.zip  
Inflating: ../red.txt
```

As you can see from the "Inflating: ../" line above, PKZIP extracted the red.txt file to the parent directory of the current directory.

Further Information on Extracting Files

The tutorials in the previous "extract" sections offer only a small sample of file extraction options. For example, you can extract "only" those files that are "newer" than the files in the extract directory. You can also determine how to handle the overwriting of files in the extract directory.

For information on these and other extract features, refer to Chapter 4.

Understanding PKZIP Commands and Options

The previous sections demonstrated how to complete basic compression and extraction operations. In the tutorials, the **add** command was used to compress files while the **extract** command was used to extract files. These are the two basic operations of PKZIP.

With PKZIP, you can customize how compression and extraction is done. You do this by using additional commands and options on the command line and by configuring

default values for commands and options that tell PKZIP how you want them to behave.

Difference between a Command and Option

A *command* represents the main operation that you want PKZIP to do. For example, the **add** command tells PKZIP to add files to an archive.

An *option* is a qualifier to a command that tells PKZIP to do the operation in a certain way or to do something additional in conjunction with it. For example, to have PKZIP use maximum compression when adding files, use the **maximum** option with the **add** command. To delete the original files after adding copies to an archive, use the **move** option with the **add** command. To extract files in a sorted order, use the **sort** option when you extract.

Including an Option in Your Command Line

In your PKZIP command, an option usually appears immediately following the main command (for example, **add** or **extract**), separated by a space. The following is an example of using the **maximum** option with the **add** command, which instructs PKZIP to compress files at the "maximum level" but lowest speed:

```
pkzipc -add -maximum test.zip white.doc
```

An "option" is usually preceded by a command that represents the main task you are performing.

Tutorial M

Compress the white.doc file into the test.zip file using the highest level of compression (the **maximum** option).

Follow these steps:

1. Verify that you are in the tut directory. If you are in the ext directory, change to the tut directory.
2. To compress files using the **maximum** option, type the following and press ENTER:

```
pkzipc -add -maximum test.zip white.doc
```

PKZIP begins to compress the file using the 'maximum' or level 9 compression. Your screen will look similar to the following:

```
◆ Using compression level 9 - Maximum comp.
```

```
Updating ZIP: test.zip  
Updating File: white.doc Deflating (59.6%), done.
```

Abbreviating Commands and Options

When you include an option in your PKZIP command, you can abbreviate that option, as long as it remains unique. In the preceding example, you could have typed "max"

instead of "maximum" as it is the only option that begins with the letters "max". The same example would look like the following:

```
pkzipc -add -max test.zip white.doc
```

Combining Options

With PKZIP, you can combine options in the same command. For example, if you want to both compress files at the maximum level of compression and add a comment to the header area of the .ZIP file, use both the **header** and **maximum** options in your command:

```
pkzipc -add -header -max test.zip brown.doc
```

The order in which you type the options is not important. You could have typed:

```
pkzipc -add -max -header test.zip brown.doc
```

Not all options can be combined with others; for example, in the previous example, you would not use maximum with another compression level option because you can only select one compression level. See Appendix A for a complete list of commands and options and their functionality.

Tutorial N

In this tutorial, you will compress the brown.doc file into the test.zip file using the **fast** option. Additionally, you will add a header comment to the test.zip file using the **header** option. The **fast** option instructs PKZIP to use the fastest compression speed.

Follow these steps:

1. Verify that you are in the tut directory.
2. To compress files using the **fast** and **header** options, type the following and press ENTER:

```
pkzipc -add -fast -header test.zip brown.doc
```

The **fast** option emphasizes faster compression speed over the compressed file size. The **header** option instructs PKZIP to prompt you to enter a header comment that will then appear in the header area of the .ZIP file.

PKZIP will display the following prompt:

```
Zip Header ?
```

3. Specify a header comment (type ASCII text at the 'Zip Header' prompt) and press ENTER. The PKZIP output screen, including the " Zip Header ?" prompt, will look similar to the following:

```
◆ Using compression level 2 - Fast
```

```
Updating .ZIP: test.zip
```

```
Zip Header ? this is my header
```

```
Adding File: brown.doc Deflating (58.8%), done.
```

As you can see from the screen output, PKZIP is using compression level 2. Also note that we supplied the comment "this is my header" at the "Zip Header ?" prompt.

Commands and Options That Have Values

Some commands and options have different possible values, called sub-options, that let you customize how the command or option is used. For example, the **level** option enables you to specify how much compression you want to use (it takes longer to compress more). When you use **level**, you specify a value for a particular level of compression. For example:

```
pkzipc -add -level=9 myarchive.zip
```

To include a value with a command or option, you separate it from the command/option with an equal sign, as in the last example.

Many *commands* have sub-options too. For example, you can use the **add** command to add all selected files to an archive or to add only files that are newer versions of files that the archive already contains. You control how **add** works by specifying the sub-option you want. To have the command add only newer versions of files that the archive already contains, you use the command with the **freshen** sub-option:

```
pkzipc -add=freshen myarchive.zip *.*
```

With commands and options that have multiple possible predefined values or sub-options, one value is used as the *default* value. The default value is the value that PKZIP uses for the command or option unless you explicitly direct PKZIP to use some other value. For example, by default PKZIP uses the **add** command with the command's **all** sub-option to add all selected files to an archive. By default, PKZIP compresses the files using a compression level of 5.

You can replace original default settings with your own by using the **configuration** command. This command is discussed in Chapter 6.

For a list of all commands and options together with their sub-options, see Appendix A.

Tutorial O

In this tutorial, you will compress files using a command and option, both of which contain sub-options or values. Specifically, you will compress only files that exist in the test.zip file, but that have changed. Additionally, you will specify that the file be minimally compressed while emphasizing maximum compression speed. This is accomplished by using the **add=freshen** command as well as the **level=1** option.

Follow these steps:

1. Verify that you are in the *tut* directory.
2. Open a text editor (e.g., vi; notepad; e). Type anything in the edit screen that you wish. Save the file into your tut directory as test.txt. Exit the text editor. A file called test.txt should now exist in your tut directory.
3. Add the test.txt file to the test.zip archive by typing the following:

```
pkzipc -add test.zip test.txt
```

The test.txt file is added to the test.zip archive.

4. Re-open test.txt file located in the tut directory with your text editor (e.g., vi; notepad; e). Edit or add characters in the edit screen. Re-save the file as test.txt into your tut directory. An updated test.txt file should now exist in your tut directory.
5. Type the following and press ENTER:

```
pkzipc -add=freshen -level=1 test.zip
```

PKZIP adds only those files that have been updated since test.zip was first created or last modified. Since we modified the test.txt file in Step 4, it will be updated in the test.zip archive. In addition, our command line tells PKZIP to compress the files using minimal compression while emphasizing maximum compression speed.

The screen output will look similar to the following:

```
◆ Using compression level 1 - Maximum speed
Freshening .ZIP: test.zip
Updating File: test.txt      Storing      ( 0.0%), done.
```

Commands and Options: Compression or Extraction?

Some command and options in PKZIP apply only to compressing and adding files to an archive while others apply only to extracting. Some commands and options apply to both. Chapters 3 and 4, respectively, describe commands and options specific to adding and extracting files.

Setting Defaults

With PKZIP, you can set up and change default values for many of the commands and options. Chapter 6 explains how to set, view, and edit default settings.

More Ways to Select Files

You can use various criteria to select files to add or extract. Different options enable you to select files by age, date, or size. You can also include or exclude files based on filename pattern.

All the options described in this section work with both the **add** and **extract** commands.

Selecting Files by Date

before, after

The ***before*** option selects files that were modified before a specified date. The ***after*** option selects files that were modified on or after a specified date.

In the United States, enter dates in one of the following formats:

- `mmddy`
- `mmddyyy`

The order in which you enter the month, date, and year depends on your *locale* setting. For more information on the *locale* setting, see Chapter 7.

The following sample command line adds files dated before June 24, 2005:

```
pkzipc -add -before=062405 test.zip
```

The command line below adds files dated June 24, 2005, or later:

```
pkzipc -add -after=062405 test.zip
```

Selecting Files by Age

older, newer

The *older* and *newer* options select files that are older or newer than a specified age. You can list the age in days (the default), hours, minutes, or seconds using the abbreviations shown in the following table.

| <i>Time unit</i> | <i>Abbreviation</i> |
|-----------------------|---------------------|
| Days (default) | d (or nothing) |
| Hours | h |
| Minutes | m |
| Seconds | s |

For example, the following command lines each add files that are no more than five days old:

```
pkzipc -add -newer=5 test.zip *
```

```
pkzipc -add -newer=5d test.zip *
```

The command lines below add files that are older than five days:

```
pkzipc -add -older=5 test.zip *
```

```
pkzipc -add -older=5d test.zip *
```

The following command line uses both options to select files to extract:

```
pkzipc -extract -newer=10 -older=5 test.zip *
```

With a time unit of days, the interval (for example, five days) is measured from the beginning of the current day. So, for example, if it is currently 3:34 p.m. on June 15, setting *newer* or *older* to 5 sets the cutoff to 12:00 a.m. June 10. The *older* option gets files dated earlier than this; the *newer* option gets files dated on or after this.

With time units of hours, minutes, or seconds, the interval is measured from the current system time. So, for example, the following command line selects files modified within the last 48 hours:

```
pkzipc -add -newer=48h test.zip *
```

Selecting Files by Size

larger, smaller

The *larger* and *smaller* options select files that are larger than or equal to, or smaller than or equal to, a size specified in bytes.

The following command line adds files whose size is in the range 5000-7000 bytes, inclusive:

```
pkzipc -add -larger=5000 -smaller=7000 test.zip
```

Selecting Files to Include or Exclude

include

The *include* option has two uses:

- To specify a filename pattern to use by default when selecting files to add or extract
- To override, in the current command line, a configured default setting that excludes files from being selected

Ordinarily, to select files whose names match a pattern (for example, *.doc), simply specify the pattern on the command line:

```
pkzipc -add test.zip *.doc
```

```
pkzipc -extract test.zip *.doc
```

To include one or more file patterns automatically when selecting files, you can configure a default value for *include*. For example, if you want to automatically include all files with the extension of .doc when adding files, enter the following:

```
pkzipc -config -add -include="*.doc"
```

This configured default causes a command line like the following to zip all .doc files in addition to the *.txt files explicitly specified.

```
pkzipc -add test.zip *.txt
```

You can also use *include* to override a default setting of the *exclude* option.

For example, if you have configured PKZIP to exclude *.txt files by default when adding, you can include such files in a particular case with the command line below:

```
pkzipc -add -include="*.txt" test.zip
```

If you do not need to override a default configuration setting, you do not need to specify the ***include*** option in your command: the file pattern by itself is enough.

For more information on modifying default configuration values, see Chapter 6.

exclude

The ***exclude*** option has two uses:

- To specify a filename pattern or list file to use to exclude files by default when selecting files to add or extract
- To override, in the current command line, a configured default setting that includes files

To exclude one or more file patterns automatically when selecting files, you can configure a default value for ***exclude***. For example, if you want to automatically exclude all files with the extension of `.doc` when adding files, enter the following:

```
pkzipc -config -add -exclude="*.doc"
```

This configured default causes a command line like the following to zip all files except `.doc` files.

```
pkzipc -add test.zip *.*
```

To exclude a list of files, specify the list file as the value of the ***exclude*** option:

```
pkzipc -add -exclude=@lst.txt test.zip
```

You can also use ***exclude*** to override a default setting of the ***include*** option. For example, if you have configured PKZIP to include `*.txt` files by default, you can exclude them in a particular case with the command line below:

```
pkzipc -add -exclude="*.txt" test.zip
```

For more information on modifying default configuration values, see Chapter 6.

3

Adding Files to an Archive

This chapter contains detailed information on the features and options available when you add files to an archive.

Conventions in this Chapter

Most commands and options discussed in this and subsequent chapters work on all platforms that PKZIP supports. The cases are noted where a command or option is specific to a platform or operating system.

Where a command has sub-options (also called a *value*), the heading introducing the section on a sub-option shows the command followed by an equal sign and then by the sub-option—for example, ***add=all***.

Default Values for Commands and Options

For each operation in this chapter, the command or option that represents that operation has a default value. The default value determines the way that the command or option is done when the command or option is used on the command line by itself, with no sub-option explicitly specified.

For example, the initial default value for the ***add*** command is ***all***, which causes the command to add all files. See Chapter 6 for information on how to change default settings.

Creating and Updating Archives

add

The ***add*** command adds files to an archive.

You can add files to either a new or existing archive. You specify the name of the archive on the command line, before any list of files to add. If the archive does not already exist, PKZIP creates it.

The command line below adds all `.txt` files in the current directory to `myarchive.zip`.

```
pkzipc -add myarchive.zip *.txt
```

Adding All Files in a Directory

You have the option of compressing all files in a particular directory with a single command. To do this, you do not have to specify each file. Simply type ***pkzipc -add***, and the name of your ZIP file, as shown below:

```
pkzipc -add test.zip
```

In the example above, all files in the current directory are compressed into the `test.zip` file. (To learn how to compress files that appear in subdirectories, see “Compressing Files in Subdirectories,” later in this chapter.)

You can also specify files from a different directory if you wish. For example, if you were in a parent directory to a directory called `temp` and you wanted to compress all the files in the `temp` directory, you could type the following:

```
pkzipc -add test.zip temp/*
```

The resulting `test.zip` file is stored in the current directory (the parent directory to the `temp` directory in our example).

Note: The ***add*** command adds all files in a specified directory to your archive file by default. You do not need to specify the ***all*** sub-option with the ***add*** command to compress all files unless you have used the ***configuration*** command to modify the default setting for ***add***.

For information on how to modify default values for commands and options, see Chapter 6.

Adding New and Modified Files

add=update

PKZIP allows you to specify that only new or modified files are added to an archive. When the ***update*** sub-option is used, dates on the files specified for archiving are compared against dates of files having the same name already present in the archive. A file is added only if no file with the same name is already in the archive or if the file to be added is newer.

The ***update*** sub-option can save time when you repeatedly archive the same files. The sub-option differs from the ***freshen*** sub-option in that it adds files which are not in the archive already.

To compress only updated files or files not already archived in a specific .ZIP file, use the ***update*** sub-option with the ***add*** option, as shown below:

```
pkzipc -add=update test.zip *.doc
```

In this example, a .ZIP file called test.zip is created in the current directory. All files in the current directory matching the file specification (*.doc) will be added or updated into the test.zip archive.

Adding Only Files That Have Changed

add=freshen

The ***freshen*** value allows you to selectively update files archived in a .ZIP file. PKZIP will compress only files that exist in the .ZIP file and that have changed. To update files that have changed, use the ***freshen*** value with the ***add*** option, as shown below:

```
pkzipc -add=freshen test.zip
```

You can also abbreviate the value, so you could type the following instead:

```
pkzipc -add=fre test.zip
```

When you use ***freshen*** with ***add***, only files that already exist in the .ZIP file "and" that have also changed will be compressed. No new files will be added to the .ZIP file.

If you only want to re-compress specific files, simply include those files in your command. For example, if you wanted to re-compress a file called resume.doc, you would type something like this:

```
pkzipc -add=freshen test.zip resume.doc
```

In the above example, only resume.doc will be re-compressed into the test.zip file. This assumes that the version of resume.doc being added is newer than the version of resume.doc that already exists in the .ZIP file.

Clearing Archive Attributes

add=incremental

If you wish to add files to a .ZIP file that have the archive attribute set and subsequently clear the archive attribute on those files, use the ***add*** command with the ***incremental*** sub-option. If you wish to add files to a .ZIP file that have the archive attribute set and *not* clear the archive attribute on those files, use the ***add*** command with the ***-incremental*** sub-option.

The ***incremental*** and ***-incremental*** sub-options can be very useful when backing up files. If, for example, the ***incremental*** sub-option is specified, only files with the archive attribute will be compressed, and the archive attribute will be set to OFF when the ZIP operation is complete for these files.

In the following command line example, PKZIP will add only those files to test.zip with the archive attribute set. Additionally PKZIP will clear the archive attribute on any of the source files that have been added to test.zip.

```
pkzipc -add=incremental test.zip
```

The next time you run this command, only those files that have the archive attribute set (new or updated files) will be added to the test.zip file.

Incremental Archiving

add=archive

By using this option, you can create a complete backup of your disk, while clearing the archive attributes to make the way for incremental archiving.

Incremental archiving makes use of the archive attribute to take only the files which have been modified since the last backup. For this process to work smoothly, you must first have a complete backup and a clearing of the archive attribute for all files.

```
pkzipc -add=archive -dir f:backup.zip
```

This prepares the files set for future incremental. For future incremental backups, use

```
pkzipc -add=incremental test.zip
```

The archive option should only be used if you are preparing your disk for incremental backup (by doing a full backup) or if you are doing a full backup of your disk.

Archive Attribute Explained

A file has various attributes, or items of information about it, such as its date. One such attribute is called the *archive* attribute. This attribute is set ON when a file is created or altered. A backup program that uses this attribute switches the attribute off when the file is backed up. By using the archive attribute to select files, you can get all (and only) files that are new or changed since the last backup. A backup that uses the attribute in this way is called an incremental backup.

Writing an Archive to STDOUT

Ordinarily, when you use the ***add*** command to archive files, you write the resulting archive to a physical file that you name in the command line. For example, the following command line archives text files to the archive `myfiles.zip`:

```
pkzipc -add myfiles.zip *.txt
```

As an alternative to writing an archive to a physical file, you can send it to *standard output*, or STDOUT. Data written to STDOUT appears on your computer screen but is not saved to disk (unless you do something extra to save it).

Data written to STDOUT can be piped to another program or be redirected to (for instance) a file.

To have PKZIP write the output of the ***add*** command to STDOUT, use a hyphen “-” in place of the name of an archive file. You must also use the ***noarchiveextension*** option to prevent PKZIP from outputting to a file named `-.zip` instead of to STDOUT. And finally, you should include the ***silent*** option to suppress normal messages that PKZIP sends to the screen so that these are not inserted in the archive stream. For example:

```
pkzipc -add -noarchiveextension -silent=normal - *.txt
```

The command line below sends output to STDOUT and then redirects that output to archive `myfile.zip`.

```
pkzipc -add -noarchiveextension -silent=normal - *.txt > myfile.zip
```

Caution: When redirecting STDOUT to a file, make sure that file is not in the set of files to be zipped. For example, if archiving all files in a directory, do not redirect output to a file in the same directory. Unlike when writing to an archive file specified on the command line, PKZIP has no way of knowing that it should skip the file to which you redirect output.

Encrypting Files That You Add to an Archive

PKZIP enables you to encrypt files when you add them to an archive. When you encrypt files, only people who know a password that you assign can decrypt and extract the files.

You can encrypt using either strong encryption or traditional ZIP encryption. Strong encryption is far more secure than the older, traditional ZIP encryption, but people are likely to need PKZIP to decrypt your files. Other ZIP utilities generally cannot decrypt strongly encrypted files.

Note: PKZIP *Command Line* only does password-based encryption: to decrypt, users must supply the password that you specified when you encrypted the files.

The graphical, SecureZIP for Windows provides an alternative way to encrypt, namely, using a *recipient list*. With recipient-list encryption, keys belonging to specified digital certificates are used in place of passwords to encrypt, such that only the owners of the digital certificates that are used can decrypt.

PKZIP *Command Line* can decrypt files encrypted using a recipient list, but PKZIP *Command Line* does only password-based, not recipient-list, encryption.

To encrypt files, use the ***password*** option when you add files to an archive. With the ***password*** option, you specify a password to use to decrypt the files. The ***password*** option can be used to do either strong or traditional ZIP encryption.

When you use strong encryption, you also have the option to encrypt not only the contents but the names of files and folders that you add to an archive. When you encrypt file names, you essentially encrypt the archive itself: the archive cannot even be opened for viewing except by someone who can decrypt its contents.

Encrypting Files with a Password

password

Use the ***password*** option (with the ***add*** command) to encrypt files so that users can use a password to decrypt them. You can do either strong or traditional ZIP encryption with the ***password*** option.

To include a password on the command line, use the ***password*** option and enter a password of at least eight characters (preceded by an equal sign). For example (where the password is *mypassword*):

```
pkzipc -add -password=mypassword test.zip
```

For more security, you can enter your password separately from the command line, at a prompt. This method prevents other users who can see the command line from learning your password.

To have PKZIP prompt for a password, include the **password** option in the command line but do not specify a password. For example:

```
pkzipc -add -password test.zip
```

When you press ENTER, a prompt like the following appears:

```
Password?
```

Type your password. The characters appear on your screen as asterisks. Press ENTER. PKZIP asks you to confirm the password:

```
Re-enter password for verification.
Password?
```

Re-enter the password and press ENTER. If your entry matches the original one, PKZIP proceeds and compresses the files. If the passwords do not match, PKZIP prompts you again:

```
Passwords don't match! Please try again.
Password?
```

Specify an Encryption Method

When you use strong encryption, PKZIP gives you a choice of encryption algorithms to use. To list the available algorithms, use the **listcryptalgorithms** command.

```
pkzipc -listcryptalgorithms
```

The following output from **listcryptalgorithms** lists all supported algorithms:

```
AES, 256      AES (256-bit)
AES, 192      AES (192-bit)
AES, 128      AES (128-bit)
3DES, 168     3DES (168-bit)
```

Use the **cryptalgorithm** option to specify a particular algorithm to use.

```
pkzipc -add -password -cryptalgorithm=aes,128 test.zip
```

If you do not use the **cryptalgorithm** option, PKZIP applies traditional PKWARE encryption.

Note: Many other ZIP utilities can decrypt archives encrypted with traditional ZIP encryption, but most cannot decrypt strongly encrypted archives. To decrypt strongly encrypted archives requires PKZIP version 6.0 or later or a copy of PKZIP Reader.

Extracting Password-Protected Files

To extract files from a password-protected archive, use the **extract** command with the **password** option.

- Type the password (preceded by an equal sign) as part of your command. For example:

pkzipc -extract -password=mysecret test.zip

If the correct password, the files are extracted (to the current directory, by default). If the password is incorrect, PKZIP displays a warning message:

```
PKZIP: (W20) Warning! Incorrect password for file: filename.ext
```

Re-type your command line with the correct password.

- If you specify the **password** option without a password, PKZIP prompts for a password. For example:

pkzipc -extract -password test.zip

When you press ENTER, a prompt appears:

```
Password?
```

Type the password. The characters appear on the screen as asterisks, for security. Press ENTER. If you specified the correct password, the files will be extracted to the current directory. If the password you entered is incorrect, a warning message displays:

```
PKZIP: (W20) Warning! Incorrect password for file: filename.ext
```

Retype your command line and when prompted enter the correct password.

- If you do not specify the **password** option when extracting an archive that contains password-protected files, PKZIP warns that the encrypted files are being skipped, and the files are not extracted.

Note: Passwords are case sensitive.

Note: For greater security, enter passwords at the prompt so that asterisks hide the characters you are entering.

Encrypting File Names

cd

Someone who cannot decrypt the contents of an archive may still be able to infer sensitive information just from the unencrypted names of files and folders. To prevent this, you can encrypt the names of files (and folders) in addition to their contents. Encrypted file names can be viewed in the clear—that is, unencrypted—only when the archive is opened by someone who has the password (or by an intended recipient if the archive was encrypted using a recipient list).

Use the **cd** option (stands for “archive *central directory*”) with the **add** command to encrypt file names. The **cd** option encrypts an archive’s central directory, where file names and virtually all other metadata about the archive is stored.

An archive that contains encrypted file names requires PKZIP version 8.0 or later to open it.

The **cd** option has two sub-options:

| <i>Sub-Option:</i> | <i>Effect:</i> | <i>For example:</i> |
|-----------------------|---|--------------------------|
| <i>encrypt</i> | Encrypts file names and the archive's central directory. This is the default sub-option, used if you enter <i>-cd</i> and do not explicitly specify a sub-option. | <code>-cd=encrypt</code> |
| <i>normal</i> | Does not encrypt file names; produces a normal ZIP file. Use to override a configured default setting that would otherwise encrypt file names. | <code>-cd=normal</code> |

You must use strong encryption when you use the ***cd*** option. You cannot encrypt file names using traditional, password encryption.

The sample command line below encrypts file names using a password. When you use the ***cd*** option with a password, PKZIP uses the default strong encryption algorithm (ordinarily AES 256) if you do not explicitly specify an algorithm.

```
pkzipc -add -password=mysecret -cryptalgorithm=aes,256 -cd test.zip
```

Encrypting File Names in an Existing Archive

You can encrypt file names in either a new or an existing archive.

- If you add files to an archive that already contains files with unencrypted file names and specify ***cd*** to encrypt file names, PKZIP encrypts the names of all files in the archive, not just names of newly added files.

If the archive contains files whose contents are already encrypted, PKZIP decrypts these files and then re-encrypts them, and their names, using the currently specified encryption method (password/recipient list) and algorithm.

If PKZIP cannot decrypt the files, PKZIP does not update the archive: no files are added, and file names are not encrypted.

- If you update an archive in which file names are encrypted, PKZIP encrypts the newly added files and their names using the same password or recipient list originally used to encrypt file names in the archive.

Contingency Keys

Note: To configure contingency keys, you must have a license for SecureZIP. A separate Policy Manager tool, which runs on Windows, is used to configure contingency keys. The Policy Manager is not provided for PKZIP.

Contingency keys, if configured, are used whenever either SecureZIP or PKZIP users encrypt.

Contingency keys are digital certificate-based keys that an administrator can have automatically included in the recipient list whenever PKZIP does strong encryption.

Contingency keys enable an organization to decrypt files encrypted by anyone in the organization, whether the files are password encrypted or encrypted for specific recipients. Contingency keys are a safeguard to be sure that important information belonging to the organization does not become inaccessible because no one in the organization can decrypt it.

A contingency key is an ordinary cryptographic key from a digital certificate. The special thing about it is that, once the key is designated as a contingency key, it is automatically included as a recipient whenever PKZIP encrypts files. This enables the owner of the key to decrypt the files.

If defined, contingency keys are used whenever PKZIP or SecureZIP encrypts. They are used even when the user chooses (strong) password-based encryption and does not pick any recipients.

The administrator can set the **config** command to display or suppress a list of contingency keys in use.

The administrator can also optionally cause PKZIP to display a line that states the number of contingency keys in use when encrypting. For example:

```
Using 2 contingency keys
```

Compressing Files in Subdirectories

recurse

PKZIP does not automatically compress files that appear in subdirectories, unless you specify those directories, or use the ***recurse*** option with the ***add*** command. With the ***recurse*** option, all specified files in a directory structure, including files located in subdirectories will be compressed.

If you have a directory called tut with a nested subdirectory called test, to compress all of the files in the tut directory and all files in the tut/test directory, you would type the following in the tut directory:

```
pkzipc -add -recurse test.zip *
```

All files in the tut directory as well as those files in subdirectories of the tut directory are compressed. However, directory path information is not stored within the .ZIP file. If you want to store directory information within your .ZIP file (in addition to compressing all the files in those directories), use the ***path*** option with the ***recurse*** option or simply use the ***directories*** option.

Storing Directory Path Information

path

Normally, when PKZIP compresses files, only the files are stored within the .ZIP file, not the paths of those files. However, you can instruct PKZIP to store the directory

path information of a file within the .ZIP file. This enables you to restore the directory structure of the files when you extract them.

For example, if a file you are compressing appears in the doc/temp directory, you can store the file within the .ZIP file as:

```
doc/temp/<filename>
```

To do this, use the **path** option with the **add** command. For example, the following command line adds all .TXT files in the specified directories and saves the specified path information:

```
pkzipc -add -path test.zip doc/temp/*.txt
```

If path information is saved, you can use the **directories** option with the **extract** command to extract files to the saved paths. PKZIP creates the directories on the saved path if they do not already exist.

Note that the **path** option gets files only from the specified directory. To get files in subdirectories of that directory as well, use the **directories** option instead of the **path** option with the **add** command. Or use the **path** option together with the **recurse** option with the **add** command.

Additional Methods for Storing Directory Path Information

In addition to storing relative path information, PKZIP allows you to further customize path information storage in your .ZIP files. Several sub-options allow you specify exactly what directory and path information is to be stored.

Each sub-option is a value that you include with the path option. The path option and/or sub-option will override the path value in the PKZIP Configuration File. If no sub-option is specified, only relative path information is stored. Examples of each sub-option appear in the table below:

| <i>Sub-option</i> | <i>To</i> | <i>For example</i> |
|-------------------|--|---|
| current | Store the directory path relative to the current location. | pkzipc -add -path=current docs.zip docs/* In this example, only directory information under the docs directory will be stored. Parent directory information will not be stored. |
| root, full | Store the full path, starting from the root directory down. | pkzipc -add -path=root docs.zip docs/* In this example, the entire directory path, starting from "root" directory will be stored. |
| specify | Stores path information for subdirectories under the specified directories | pkzipc -add -path=specify docs.zip temp/docs/* Stores path information for subdirectories under temp\docs. |
| relative | Store the directory path relative to the current working directory of the drive specified. | pkzipc -add -directories=relative docs.zip c:*.doc z:*.doc In this example the path information for those directories recursed under the current working directory (for both the C: and Z: drives) will be stored. |

| | | |
|-------------|--|--|
| none | Turn off the path option. (Used to override configuration file). | pkzipc -add -path=none docs.zip /temp/docs/* In this example, only the file names are stored. |
|-------------|--|--|

Storing and Recreating Directory Path Information

directories

The **directories** option works with both the **add** and **extract** commands.

- With the **add** command, the **directories** option is equivalent to using the **recurse** and **path** options together. It instructs PKZIP to search subdirectories for files and to save the files and their directory path information in the .ZIP file.
- With the **extract** command, the **directories** option extracts any directory tree structure saved with files.

The following example uses the **directories** option with the **add** command to add any files called `whatsnew.htm` in the current directory or in any subdirectory of the current directory:

```
pkzipc -add -dir testdir.zip whatsnew.htm
```

Screen output lists any matching files found in subdirectories:

```
Creating .ZIP: testdir.zip
```

```
Adding File: Win/PK/Whatsnew.htm Deflating (67.0%), done.
Adding File: Win/SZ/Whatsnew.htm Deflating (66.7%), done.
```

On Windows, even if the command line includes the **directories** option, you can turn off the searching of subdirectories for files matching a particular file pattern by including the drive letter (for example, `C:`) in the pattern. The pattern must also not include any wildcard characters (`*` or `?`).

For example, the following command line adds only the specified file; it does not add matching files from subdirectories of `MyFiles`:

```
pkzipc -add -dir testdir.zip C:\MyFiles\whatsnew.htm
```

For information on extracting files saved with directory information, see the section “Retaining Directory Structure while Extracting” in Chapter 4.

As with the **path** option, PKZIP provides several choices for saving directory path information. The following table lists the sub-options you can use with **directories** option:

| <i>Sub-option</i> | <i>To</i> | <i>For example</i> |
|---------------------|--|--|
| current | Store the directory path relative to the current location. | pkzipc -add -directories=current docs.zip docs/* In this example, only directory information under the docs directory will be stored. Parent directory information will not be stored. |
| root or full | Store the full path, starting from the root directory down. | pkzipc -add -directories=root docs.zip docs/* In this example, the entire directory path, starting from "root" directory will be stored. |
| specify | Store path information for subdirectories under the specified directories | pkzipc -add -directories=specify docs.zip temp/docs/* Stores path information for subdirectories under temp/docs. |
| relative | Store the directory path relative to the current working directory of the drive specified. | pkzipc -add -directories=relative docs.zip c:*.doc z:*.doc In this example, the path information for those directories recursed under the current working directory (for both the C: and Z: drives) will be stored. |
| none | Turn off the path option. (Used to override configuration file). | pkzipc -add -directories=none docs.zip /temp/docs/* In this example, only the file names are stored. |

See also the preceding section, "Additional Methods for Storing Directory Path Information."

Setting the Compression Level

Native ZIP compression (which uses the Deflate compression algorithm) and the **bzip2** and **deflate64** compression options each support a range of compression levels from 0 (no compression) to 9 (maximum). By default, each of these options uses level 5, or *normal*, compression. Normal compression strikes a middle balance between compression and performance. In general, greater compression takes more time.

You can use the **level** option to specify a compression level from 0 to 9 when you create or update a ZIP file using one of the compression methods named above.

Alternatively, you can use the options **normal**, **store**, **speed**, **fast**, and **maximum** to specify a desired balance between speed and degree of compression. See "Specifying a Compression Level by Name," later in this chapter.

With the **dclimplode** option, you set the compression level in a different way, namely, by specifying the dictionary type and size as sub-options.

Specifying a Compression Level from 0-9

level

The *level* option enables you to specify a level or degree of compression to use when creating or updating a ZIP archive with the Deflate64, BZIP2, or default Deflate compression methods. (See the *deflate64* and *bzip2* options to learn about using these compression methods.)

To set a compression level with the *level* option, specify a numeric value for the option from 0 to 9. A value of 0 specifies zero compression.

The following command line specifies a compression level of 2 and uses the native Deflate compression method:

```
pkzipc -add -level=2 test.zip *.doc
```

The following command line specifies level 2 compression and the BZIP2 compression method to create or update a ZIP archive:

```
pkzipc -add -bzip2 -level=2 test.zip *.doc
```

Level 5 is the default compression level for *level*. You can use the *configuration* command to set a different default. For example, the following command line sets the default value for *level* to 9:

```
pkzipc -config -level=9
```

For information on changing default settings, see Chapter 6.

Specifying a Compression Level by Name

store, speed, fast, normal, maximum

As an alternative to setting numeric compression levels with *level*, you can use the options *normal*, *store*, *speed*, *fast*, and *maximum*.

These options enable you to use non-numeric names to specify a desired balance between speed and degree of compression. For example, the following command line specifies the *fast* compression option:

```
pkzipc -add -fast test.zip *.doc
```

The non-numeric compression level options are described in the following table:

| <i>Option</i> | <i>Description</i> | <i>Example</i> |
|-----------------------------------|--|---|
| <i>speed</i> | Provides the fastest performance and the least compression: some files are compressed with the Deflate method, using level 1 compression; others* are stored (level 0) uncompressed. | pkzipc -add -speed test.zip *.doc pkzipc -add -bzip2 -speed test.zip *.doc |
| <i>fast</i> | Provides the second fastest compression: some files are compressed with the Deflate method, using level 2 compression; others* are stored (level 0) uncompressed | pkzipc -add -fast test.zip *.doc |
| <i>maximum</i> | Provides the highest level of compression (level 9) | pkzipc -add -max test.zip *.doc |
| <i>store</i> | Provides zero compression: just stores files inside the archive (level 0) | pkzipc -add -store test.zip *.doc |
| <i>normal</i> (Default) | Provides a middle balance of compression and speed (level 5) | pkzipc -add -norm test.zip *.doc You would only need to use this option if you changed the default compression level. See Chapter 6 for information on setting defaults. |

* Types of files that the ***speed*** and ***fast*** options store uncompressed are listed below. The other named options (except ***store***) compress files of these types. You can also use the ***level*** option to compress files of these types.

| | |
|---------|--------|
| *.bz2 | *.jpeg |
| *.bzip2 | *.jpg |
| *.cab | *.mp3 |
| *.gz | *.mpeg |
| *.gzip | *.mpg |
| *.rar | *.sxw |
| *.gif | |

Compressing Files with a List File

Instead of specifying a specific file or file pattern in your command line, you can point PKZIP to a list file that lists all the files or file patterns that you want to operate on. A list file is an ASCII text file that contains file names or file patterns and path information. A list file can be an ideal solution for users who archive specific file sets

on a regular basis. Using a list file saves time in that you do not need to type file names and paths each time you wish to compress these files with PKZIP. A list file may contain wildcard specifications (*,?) as well as exact file names and paths.

A list file in a DOS based environment might look similar to the following:

```
*. exe
*. doc
\tut\*. doc
\tut\?????. *
pkzip.html
```

You identify a list file as such on the command line by prefixing it with the list character—"@" by default. See the *listchar* option if you want to use a different character for the list character.

The following example adds the files listed in list file *lst.txt* to the archive *test.zip*:

```
pkzipc -add test.zip @lst.txt
```

You can also use a list File to specify files to exclude from an archive, based on some criteria, using the *exclude* option. See "More Ways to Select Files" in Chapter 2.. For more information on the *listchar* option, see "Changing the List Character for List Files" in Chapter 7.

Note: The file format for a list file when extracting files may differ from the format referenced above. See the section "Extracting Files with a List File" in Chapter 4 for more information.

Getting a List of Files from Standard Input

You can tell PKZIP to treat as a list a set of files output by another program. PKZIP can then compress the files in the dynamically constructed list.

Use a hyphen (-) prefixed with the list character ("@" by default) to identify a set of files in standard input as a list. For example, in the following command line, PKZIP treats a list of files output from *some program* as a list file and compresses the files into *test.zip*:

```
<some program> | pkzipc -add test.zip @-
```

The special, dynamically constructed list can also be used with the include and exclude options. For example:

```
<some program> | pkzipc -add test.zip -include=@-
```

```
<some program> | pkzipc -add test.zip -exclude=@- *.doc
```

Compressing Files in a Specific Format

shortname

The *shortname* option allows you to convert a file name in *long* file name format to a DOS equivalent *short* (8+3) file name before compressing the file(s). You may

specify how you wish files to be compressed by using the *shortname* option with the *dos* sub-option:

```
pkzipc -add -short=dos save.zip
```

Compressing Files with the Deflate64 Method

deflate64

The *deflate64* option enables you to use the Deflate64 compression method to compress files and create ZIP archives. The Deflate64 method can produce greater compression than the Deflate method that PKZIP uses by default because Deflate64 uses a larger dictionary window (64K compared to 32K).

Not all ZIP-compatible programs from other vendors can extract files compressed with the Deflate64 method.

You can use the *level* option with *deflate64* to specify a level of compression from 0 to 9 (0 is zero compression).

The following command line uses the Deflate64 method with the *level* option set for maximum compression:

```
pkzipc -add -deflate64 -level=9 mydocs.zip *.doc
```

Compressing Files with the BZIP2 Method

bzip2

BZIP2 is an open-source compression algorithm that requires more memory and processing power than standard ZIP compression but provides greater compression. PKZIP can use BZIP2 compression to create either ZIP or BZIP2-format archives (.bz2 files). A BZIP2 archive, unlike a ZIP archive, can contain only a single file.

Files compressed with the BZIP2 method can be extracted with most versions of PKZIP, 4.6 and later, but other ZIP-compatible programs may not be able to extract files compressed with BZIP2.

You can use the *level* option with *bzip2* to specify a level of compression from 0 to 9 (0 is zero compression).

The following command line uses the BZIP2 method with the *level* option set for maximum compression:

```
pkzipc -add -bzip2 -level=9 mydocs.zip *.doc
```

Compressing Files Compatible with the Data Compression Library

dclimplode

The *dclimplode* option enables you to create .ZIP archives that are compatible with the PKWARE Data Compression Library (DCL). Files compressed with this method can be extracted by most versions of PKZIP 2.5x and later, though not by other .ZIP-compatible programs.

When using the Implode compression method, you must specify dictionary type (ASCII or BINARY) and dictionary size (1024, 2048, or 4096). In general, the larger the dictionary, the greater the compression. Use the BINARY dictionary when compressing binary files (for example, executable programs) or when the type of the file is unknown. Use the ASCII dictionary with ASCII (text) files.

For example, to use the DCL Implode method to compress all text files in a directory, type the following:

```
pkzipc -add -dclimplode=ascii,4096 text.zip *.txt
```

Compressing Files to a Specified Type of Archive

archivetype

PKZIP creates ZIP archives by default: When you use the *add* command to create a new archive, PKZIP creates a ZIP archive if you do not specify a file name extension that PKZIP recognizes as associated with a particular archive type.

For example, the following command creates a ZIP archive called *myfile.foo.zip*:

```
pkzipc -add myfile.foo
```

With the *archivetype* option, you can explicitly tell PKZIP what type of archive to create. The following example creates an archive *myfile.foo.bz2* of the BZIP2 archive type. Note that the file name extension *bz2* associated with the specified archive type is added to the file name:

```
pkzipc -add -archivetype=bzip2 myfile.foo
```

A simpler way to create a BZIP2 archive called *myfile.foo.bz2* is to specify the file name extension as part of the file name. In this case, you do not need the *archivetype* option:

```
pkzipc -add myfile.foo.bz2
```

However, when the *archivetype* option is used to specify an archive type, you can use the *noarchiveextension* option with it to tell PKZIP not to add an extension to the specified file name. For example, the following command creates an archive *myfile.foo* of the BZIP2 archive type:

```
pkzipc -add -archivetype=bzip2 -noarchiveextension myfile.foo
```

Compressing Files to Diskette

span

With PKZIP, you can save your .ZIP file or self-extracting file to one or more diskettes when you create it (instead of saving it on your hard disk drive). You can also create a *split* archive that is saved as multiple files on your hard disk. You can also have PKZIP format or wipe your removable media before writing to it.

Creating a Spanned Archive

You can save a ZIP file to multiple diskettes if it is too large to fit on a single one. This is called disk *spanning*. PKZIP prompts you to insert diskettes (or other media) as they are needed.

Depending on the size of the ZIP file, it may be necessary for PKZIP to save the file on multiple diskettes. This process is called "spanning".

To create a spanned archive:

1. Insert a diskette (or other appropriate medium) into its drive.
2. Type your PKZIP command, and press ENTER. Make sure to specify the drive letter or path that corresponds to your destination drive. A sample command line appears below:

```
pkzipc -add -span a:\test.zip *.doc
```

Note: Ordinarily, PKZIP recognizes removable media as such and spans them as necessary automatically, even if you do not specify the *span* option. However, if PKZIP is unable to detect that you are creating your ZIP file on removable media, use the *span* option to tell PKZIP to span.

Creating a Split Archive

The *span* option is also used to create a *split* archive. A split archive is an archive created in segments, all of which are written to your hard disk as separate files.

To create a split archive on your computer disk, specify a size in bytes, or use a predefined size from the following table:

| <i>Predefined size</i> | <i>Comment</i> |
|------------------------|------------------------------------|
| 360 | 360KB floppy disk (362496 bytes) |
| 720 | 720KB floppy disk (730112 bytes) |
| 1.2 | 1.2MB floppy disk (1213952 bytes) |
| 1.44 | 1.44MB floppy disk (1457664 bytes) |
| 2.88 | 2.88MB floppy disk (2915328 bytes) |
| 95.7 | 100MB ZIP disk (100431872 bytes) |
| 650 | 650MB CD-ROM (681574400 bytes) |
| 700 | 700MB CD-ROM (734003200 bytes) |

For example, to create a split archive of size 1.44 Mb to your local system, type the following command:

```
pkzipc -add -span=1.44 c:\test.zip *.doc
```

To have PKZIP format or wipe removable media before writing to it, use the *span* command with *format* or *wipe*. For example, the following command line formats the media prior to creating a ZIP archive:

```
pkzipc -add -span=format a:\test.zip *.doc
```

Creating Multiple, Respective Archives

archiveeach

With the *archiveeach* option, you can create a separate archive for each of multiple files specified in a single command line.

```
pkzipc -add -archiveeach *.*
```

With *archiveeach*, you do not specify names for new archives. PKZIP names each new archive after the file it contains, with an archive-type filename extension (*ZIP* by default) appended to the end. For example, a ZIP archive created for file `mydata.xls` is named `mydata.xls.zip`. An archive created for file `mydata.zip` is named `mydata.zip.zip`.

If an archive with the same name already exists in the target location, PKZIP appends a number to the archived file name before appending the `.zip` (or other filename extension). For example: `mydata.xls2.zip`.

To specify a particular archive type, use the **archivetype** option with the **archiveeach** option. The **archiveeach** option can also be used with the **encode** option, to convert the archive initially created to a different type. By using **archivetype** and **encode** together with **archiveeach**, you can, for example, create multiple `.tar.gz` files:

```
pkzipc -add -archiveeach -archivetype=tar -encode=gz C:\data\*.*
```

You can specify a destination for the new archives in a sub-option to **archiveeach**:

```
pkzipc -add -archiveeach=C:\newzips C:\myfiles\*.*
```

Storing File Information

PKZIP allows you to store specific file attribute/information within your .ZIP file. You can:

- Store file attributes, including hidden, system, archive, and read-only.
- Store extended file attribute information.
- Remove (mask) file attributes.

Refer to the sections that follow for more information.

Compressing Files with Specified Attributes

attributes

PKZIP allows you to compress files based on the attributes that they possess. These attributes are usually assigned either by the creator of a file, a system administrator, or by the operating system. The following are attributes you can store:

- Hidden
- System
- Read-only
- Archive

The attributes set by default for compression are *archive* and *read-only*. With this setting, if you do not use the **attributes** option on your command line, PKZIP compresses all files except any having the attributes *hidden* or *system*.

To specify a file attribute, you must include it with the **attributes** option in your command line. Each attribute is a value for the **attributes** option. You can:

- Specify which file attributes to compress.
- Override values in the Configuration file.
- Turn off the **attributes** option.

The table below lists all of the available sub-options for storing file attribute information:

| <i>Sub-Option:</i> | <i>To:</i> | <i>For example:</i> |
|--------------------|---|---|
| hidden | Compress files including those that contain the "hidden" file attribute. | pkzipc -add -attributes=hid test.zip |
| system | Compress files including those that contain the "system" file attribute. | pkzipc -add -attributes=sys test.zip |
| readonly | Compress files including those that contain the "read-only" file attribute. | pkzipc -add -attributes=read test.zip |
| archive | Compress files including those that contain the "archive" file attribute. | pkzipc -add -attribute=archive test.zip |
| all | Compress files including those that contain the hidden, system, or read-only file attribute. | pkzipc -add -attributes=all test.zip |
| none | Turn off the attributes option in the Configuration file or compress files that do not have any attributes set. | pkzipc -config -attributes=none |

You may use a dash (-) before an **attributes** sub-option on your command line to exclude files with a specific attribute from being added regardless of the default attributes configuration setting. If, for example, the default attributes configuration setting was set to "all", you could enter the following command line to exclude hidden files from being added to the test.zip file.

```
pkzipc -add -attributes=-hidden test.zip
```

Extended Attribute Storage

noextended

When PKZIP adds files to an archive, it automatically stores extended attributes with those files. PKZIP defines extended attributes as any file attributes other than the standard FAT file system attributes (Read-Only, Archive, System, Hidden, Directory). Extended attributes usually represent a characteristic of a file, such as the date and time the file was last modified.

Excluding extended attributes slightly reduces the size of the archive. To exclude extended attribute information, use the **noextended** option, as in the following example:

```
pkzipc -add -noextended test.zip readme.doc
```

Note: The **noextended** option does not affect storage of the offline, temporary, and system attributes.

Extended Attributes and the OS

Extended attributes are automatically added to .ZIP archives when they are created. PKZIP does not display a message indicating that it is saving extended attributes.

PKZIP stores the following extended attributes:

- Create time
- Last modification time
- Last access time

Note: Typically, PKZIP automatically extracts extended attributes with archived files and/or directories and thus overwrites existing files, directories and extended attributes with the ones stored in the .ZIP file.

Extended Attributes and 204g Compatibility

204

By default, PKZIP does not enable PKZIP for DOS 2.04g compatibility. When 204g compatibility is enabled, extended attribute data is stored in both the Local header and Central header records. This will result in a slightly larger .ZIP file size, but improves the chance that extended attribute information can be recovered if the .ZIP file should become damaged. It also ensures the extended attribute information is always retained if the file is generated with a version of PKZIP other than 2.04g. This option is ignored when extracting. The **204** option also limits the number of files that can be added to a .ZIP archive to 16,383. To enable 204g compatibility, use the **204** option as in the following example:

```
pkzipc -add -204 test.zip *
```

Including Additional Information in a ZIP File

With PKZIP, you can include additional information in your .ZIP file, such as a "comment", to identify that .ZIP file.

You can include a:

- Text comment.
- Password to protect your .ZIP file.
- Header comment.
- Date for the .ZIP file (other than the creation date).

Refer to the sections that follow for more information.

Including a Text Comment

comment

With PKZIP, you can include a comment for the individual files within a .ZIP file. There are several options for adding comments to your .ZIP files. To include a comment, use the **comment** option alone or with the **add** command. When you run the command, PKZIP prompts you to enter the comment.

The table below lists the available sub-options for adding comments to your .ZIP archives:

| <i>Sub-Option:</i> | <i>To:</i> | <i>For example:</i> |
|--------------------|---|---|
| all | Comment all of the files and any new files added. | pkzipc -add -comment=all test.zip * |
| unchanged | Comment only files existing in the ZIP file that are not either updated or being added. | pkzipc -add -comment=unchanged test.zip * |
| add | Comment only the new files added. | pkzipc -add -comment=add test.zip * |
| none | Disable the comment option. | pkzipc -add -comment=none test.zip * |
| freshen | Comment all of the files updated in the ZIP file. | pkzipc -add -comment=freshen test.zip * |
| update | Comment all files added and updated in the zip file. | pkzipc -add -comment=update test.zip * |

Note: Comment length is limited to 59 characters.

Including a Header Comment

header

With PKZIP, you can include a general comment for a .ZIP file. This is called a "header" comment because it appears in the header portion of a .ZIP file. This differs from the **comment** option in that the "header" comment applies to the entire .ZIP file, not to individual files within the .ZIP file.

Note: Headers for .ZIP files are limited to 16K in size. PKZIP will automatically truncate headers larger than 16K.

To include a header comment, use the **header** option and the comment or comment file with the **add** command. PKZIP provides several methods to include the comment. You can:

Include an existing file as the header. With this method, you type the **header=@filename.ext** option. If there are no spaces in the file name, it is not necessary to use quotation marks. For example:

```
pkzipc -add -header=@header.txt test.zip *
```

Type the actual comment as part of the command. With this method, you include an equal sign, followed by the comment. If there are no spaces in your comment, it is not necessary to use quotation marks. Our example comment does include spaces so therefore our command line would look like the following:

```
pkzipc -add -header="This is the comment" test.zip *
```

- If you include the header option only, PKZIP will prompt you for text you wish to be the header:

```
pkzipc -add -header test.zip *
```

- When you press enter, the following prompt appears:

Zip Header ?

Type your header comment and press ENTER.

Specifying the Date of a .ZIP File

archivedate

When you create an archive file, PKZIP gives it the current date by default. You can specify a different date for the file by using the ***archivedate*** option with the ***add*** command.

Note: The ***archivedate*** option is the same as the older ***zipdate*** option, which is now deprecated.

PKZIP provides several methods for applying a date to an archive file. The table below lists the available sub-options for applying date information to your archives:

| <i>Sub-Option:</i> | <i>To use:</i> | <i>For example:</i> |
|---------------------------------|--|--|
| <i>retain</i> | The date that the file was created. | pkzipc -add -archivedate=retain test.zip * |
| <i>none</i> (Default) | The current date. | pkzipc -add -archivedate=none test.zip * |
| <i>oldest</i> | The date of the oldest file within the archive file. | pkzipc -add -archivedate=oldest test.zip * |
| <i>newest</i> | The date of the newest file within the archive file. | pkzipc -add -archivedate=newest test.zip * |

Removing File Attributes

mask

If you use the **attributes** option to have PKZIP process files that have attributes, such as *hidden* or *system*, specified with the **attributes** option, you can use the **mask** option to strip those attributes from the files when they are archived or extracted.

You can only use the **mask** option with attributes specified with the **attributes** option. Attributes can be specified with this option either on the command line or as configured defaults,.

The table below lists all of the available sub-options for masking file attribute information:

| <i>Sub-Option:</i> | <i>To:</i> | <i>For example:</i> |
|--------------------|---|---------------------------------------|
| hidden | Remove the hidden file attribute from files. | pkzipc -add -mask=hidden test.zip * |
| system | Remove the system file attribute from files. | pkzipc -add -mask=system test.zip * |
| readonly | Remove the read-only file attribute from files. | pkzipc -add -mask=readonly test.zip * |
| archive | Remove the archive attribute from the file. | pkzipc -add -mask=archive test.zip * |
| none | Turn off file masking. | pkzipc -add -mask=none test.zip * |
| all | Remove all attributes from files. | pkzipc -add -mask=all test.zip * |

The mask sub-options can be used on the command line either individually or in a comma-separated list.

You may use a dash (-) before a mask sub-option on your command line to preserve a file attribute being added or extracted with a file, regardless of the default **mask** configuration setting. For example, if the default mask configuration is set to *all*, you can enter the following command line to preserve the *hidden* attribute associated with any of the files to be added:

```
pkzipc -add -mask=-hidden test.zip
```

Sorting Files Within a .ZIP File

sort

With PKZIP, you can sort the files in an archive in several ways. If you do not change the sort order, the files are automatically sorted in the order in which they were compressed into the archive. This is called the "natural" order.

The **sort** option works with the **add**, **extract**, **test**, and **view** commands. The value you include with **sort** depends on the command you select.

| <i>Sub-Option</i> | <i>To sort by</i> | <i>For example</i> |
|-------------------|---|---|
| date | File date. | pkzipc -add -sort=date temp.zip |
| size | Original uncompressed size of the file ("length" in display). | pkzipc -add -sort=size temp.zip |
| extension | File extension. | pkzipc -add -sort=ext temp.zip |
| name | Sorts files and folders by name in a single series. (Contrast with -sort=none.) | pkzipc -add -sort=name temp.zip |
| none | Groups folders first, sorted by name, and then groups files, sorted by name. (The default.) | pkzipc -view -sort=none temp.zip |
| natural | Preserves the order in which files were added to an archive. | pkzipc -view -sort=natural temp.zip |
| ratio | Ratio of uncompressed size to compressed size. | pkzipc -view -sort=ratio temp.zip Note: The ratio sub-option will not work with the add command. |
| crc | CRC (Cyclic Redundancy Check) number. | pkzipc -view -sort=crc temp.zip Note: The crc sub-option will not work with the add command. |
| comment | File comment. | pkzipc -view -sort=comment temp.zip Note: The comment sub-option will not work with the add command. |

The **name** sub-option sorts entire path names; it does not sort file names directly if folder information is present.

For example, the **name** sub-option sorts the two files *abacus.txt* and *zebra.txt* as follows if they are added to an archive without including any path or folder information:

```
abacus.txt
zebra.txt
```

However, if the files are added with folder information, the name of the outermost folder in the path determines their order of appearance. This is because **name** sorts the entire path name whether or not it includes folder names. For example:

```
al I \junk\zebra.txt
everything\important\abacus.txt
```

By contrast, the **none** sub-option groups path names that contain folder names and sorts this group in a separate series from file names that do not include folder information. The names below are sorted by **none**:

```
al I \junk\zebra.txt
```

```
everything\important\abacus.txt
anotherfile.txt
onefile.doc
somepix.gif
```

If no **sort** option is specified, files are sorted as if **sort=none** was specified (unless you have changed configuration defaults).

If you specify the **sort** option on your command line but do not specify a sub-option value, the **name** sub-option is applied.

Note: Using the **sort** option with the **add** command only works on new archive files. It does not work with an archive that is being updated.

Moving Files to a .ZIP File

move

Normally, when you compress files, you end up with two copies of each file: the original file and the compressed file. With PKZIP, you can choose to remove the original file "after" you compress it into the .ZIP file.

If you want to move only specific files, you must compress them separately since you can only move all or none of the files that you are compressing.

To move files, use the **move** option with the **add** command, as shown below:

```
pkzipc -add -move test.zip *.doc
```

This sample command line tells PKZIP to compress and add to archive `test.zip` all files that end in `.doc` and then to delete the original files.

| |
|--|
| <p>CAUTION: Like any operation that deletes files, the move option should be used with care.</p> |
|--|

Wiping Deleted Files

wipe

A deleted file still remains on your disk and can often be fully or partly recovered. If you want to prevent recovery of certain files that PKZIP deletes, you can use the **wipe** option with the **add** command to have PKZIP *wipe* them. Wiping a file overwrites the file's data so that it cannot be read.

If wiping is turned on, PKZIP wipes deleted files:

- That have been moved into an archive with the **move** option
- That contain the previous version of an archive that has just been updated

The wipe option has three sub-options:

| <i>Sub-Option</i> | <i>Description</i> |
|-------------------|---|
| None | Turns wiping off if it is configured on |
| Random | Overwrites files once with random data (the default) |
| NSA | Overwrites files seven times, to the NSA standard. (Takes much longer.) |

For example:

```
pkzipc -add -wipe test *.doc
```

4

Extracting Files

Introduction

This chapter describes the options PKZIP offers for extracting files from archives. These options give you various ways to choose what files to extract and where to extract them to and help you manage every aspect of the extracting files.

Conventions in This Chapter

In some of the headings in this chapter, a word appears immediately below the heading. That word represents the actual command or option you would type in your PKZIP command (and in many cases, a value or sub-option in addition to the command (e.g., **extract=all**)). If the text under the heading is the command followed by an equal sign and another word, it means that the other word is a value (sub-option) that goes with the command.

Default Values for Commands and Options

For each extraction task in this chapter, the command or option that represents that task contains a default value. A default value represents the action that occurs when only the name of the command or option is included in your PKZIP command. For example, the default for the **extract** command is to unzip or uncompress "all" files in an archive.

See Chapter 6 for information on configuring default values for commands and options.

Extracting New and Existing Files

When you extract files from a .ZIP file, you can select those files you wish to extract and those you do not. If the directory into which you extract the files contains files that have the same name as those being extracted, you have to decide if you want to overwrite those files.

PKZIP provides several ways to choose which files to extract. You can extract:

- All files in an archive (the ***all*** sub-option)
- Files that are not in the target extract directory plus files that are more recent versions of files that are in the extract directory (the ***update*** sub-option)
- Only files that are more recent versions of—that is, have the same names as—files that are already in the extract directory (the ***freshen*** sub-option)

Extracting All Files from an Archive

extract=all

To extract all files from an archive file, type ***pkzipc -extract*** and the name of your archive file, as shown below:

```
pkzipc -extract test.zip
```

In this example, all files in the archive are extracted into the current directory.

The ***all*** sub-option is the original default for the ***extract*** command. You do not need to specify this sub-option unless you have changed the default for ***extract*** to some other sub-option.

The following example explicitly specifies the sub-option. This command does the same thing as the first example but also overrides any changed default setting. The override applies just to this instance of the command; it does not reset the default you have defined.

```
pkzipc -extract=all test.zip
```

Extracting Newer Versions of Existing Files and New Files

extract=update

The ***update*** sub-option extracts to the target, extract directory only files that are not already in the directory or are newer versions of files that are already there. Archive files that are older versions of files already in the directory are not extracted.

```
pkzipc -extract=update test.zip
```

Extracting Only Newer Versions of Files

extract=freshen

The ***freshen*** sub-option extracts only files that are newer versions of files that already exist in the target, extract directory. It does not add any files to the directory that are not already there in an earlier version.

```
pkzipc -extract=freshen test.zip
```

Extracting from an Archive Embedded in An Archive

embedded

An archive can contain other archive files. For example, a ZIP file can contain other ZIP archives, or a GZIP archive might contain a TAR archive. Such contained archives are said to be *embedded* in the archive that contains them.

If PKZIP encounters a lone embedded archive file in another archive whose contents PKZIP is extracting, PKZIP prompts you whether you would like to extract the contents of the embedded archive or just the archive itself. For example, if PKZIP is extracting the contents of *outerarchive.zip*, and *outerarchive.zip* contains *innerarchive.zip*, PKZIP asks you whether you want to extract the files in *innerarchive.zip* or just the inner archive file itself.

The ***embedded*** option can be used with ***extract*** to tell PKZIP to omit the prompt and just go ahead and extract the files contained in any lone embedded archive file of the specified type. You must specify the type.

For example:

```
pkzipc -extract -embedded=zip outerarchive.zip
```

In the example, if *outerarchive.zip* contains *innerarchive.zip* and no other archive files, PKZIP extracts the files from *innerarchive.zip* instead of extracting *innerarchive.zip* itself and does not prompt.

You can also use ***embedded*** to tell PKZIP *never* to extract the contents of an embedded archive of a specified type. This usage of ***embedded***, like the first, causes PKZIP not to prompt for instructions. PKZIP simply extracts the embedded archive file, not its contents. To tell PKZIP *never* to extract the contents of a specified type of embedded archive, prefix the sub-option with a hyphen:

```
pkzipc -extract -embedded=-zip outerarchive.zip
```

Note that PKZIP extracts the contents of an embedded archive, with or without prompting, only if that archive is the *only* embedded archive in the outer archive file. If the outer archive file contains multiple embedded archives, the embedded archive files themselves are extracted.

Checking for Viruses when Extracting

avscan, avargs

PKZIP can use your anti-virus program to scan for viruses when you extract files.

The ***avscan*** option controls whether extracted files are scanned for viruses and specifies the anti-virus program to run to do scans.

When you extract with the ***avscan*** virus scanning option turned on, PKZIP first extracts the specified files and then runs the anti-virus program to recursively scan all

files in the specified destination directory and its subdirectories. PKZIP relays to you any messages returned by the virus scanning program.

If your virus scanner is set up to scan files dynamically as they are read or written, you do not need launch a virus scan from PKZIP. Your virus scanner will automatically scan the files as they are extracted.

How your anti-virus program deals with files infected by a virus is determined by the way the program is configured and by the arguments, if any, included in the PKZIP command line used to run the scanner. The contents of the command line used to run the scanner and the arguments that may be available for it depend on your anti-virus program.

Use the PKZIP ***avargs*** option to specify any anti-virus command line arguments. To tell the anti-virus program what directory to scan, include the variable `%e`. PKZIP replaces this variable with the full path to the extraction directory before passing the command line to the anti-virus program.

The following example shows ***avscan*** used to run a virus-scanning program. The variable `%e` and arguments for the virus-scanning program's command line are given in the ***avargs*** option.

```
pkzipc -extract -avscan=f-prot.exe -avargs="%e /silent /nomem /noboot"  
myfiles.zip
```

In ***avscan***, specify the full path to the anti-virus program if the executable is not on the search path.

PKZIP assumes that the anti-virus program will not launch any graphical interfaces that require user interaction and that the program will automatically clean up any viruses that it finds.

Most virus scanning programs return a value of 0 when a scan completes successfully and finds no viruses. If a program returns any other value as the result of a scan, PKZIP issues a warning that some of the extracted files may not have passed the scan.

Both ***avscan*** and ***avargs*** can be configured for use by default. Configuring ***avscan*** causes PKZIP to do virus scans by default whenever files are extracted, using the specified anti-virus program executable and whatever anti-virus command line arguments, if any, are given in ***avargs***.

Extracting Files in Lower Case

lowercase

The ***lowercase*** option allows you to extract files in lower case regardless of how the file name was originally archived. To force the file names to be extracted in lowercase, use the following example:

```
pkzipc -extract -lowercase test.zip
```

Preserving File Times

times

The *times* option allows you to preserve the access, creation and modification times of the extracted files. Specify the sub option *all* to preserve all times, use *access* to preserve the access times only, use *modify* to restore the time of last modification times or *create* to restore the creation times.

To preserve all the file times, use the following example:

```
pkzipc -extract -times=all test.zip
```

Translating End of Line Sequence

translate

The *translate* option translates text end-of-line characters to the character sequence used by a different platform. The option can be used with *add* or *extract*. Specify a sub-option from the following table to translate line endings to the sequence used by the desired platform.

| <i>Sub-Option</i> | <i>Platform</i> |
|-------------------|---------------------------------------|
| <i>dos</i> | DOS/Windows (carriage return/newline) |
| <i>mac</i> | MacOS (carriage return) |
| <i>unix</i> | UNIX (newline) |

The following command line translates text line endings to UNIX on extraction:

```
pkzipc -extract -translate=UNIX test.zip
```

Retaining Directory Structure while Extracting

directories

If you stored directory path information within a .ZIP file, you can re-create those directory paths when you extract the files. For example, if you compressed a file called apples.doc in the temp/fruit directory, and you stored temp/fruit you can re-create temp/fruit in the location in which you extract the files.

To re-create directories, use the *directories* option with the *extract* command, as in the following example:

```
pkzipc -extract -directories test.zip
```

When you use this command, all directories that were stored in the .ZIP file will be retained during extraction. The directory path stored is appended to the directory in which you extract the files. For example, if your extract directory is /doc, and a directory path stored with the files is temp/fruit, the files would now be extracted to /doc/temp/fruit.

Sorting Files in the Extract Directory

sort

PKZIP allows you to specify the sort order of files that are compressed in a .ZIP file or extracted into a destination directory. For example, if you wish to extract files in a specified sort order (by date), you would type the following and press ENTER:

```
pkzipc -extract -sort=date test.zip
```

In this example, all files that exist in the test.zip file are extracted into the current directory sorted in ascending order by date. For more information on sort options, see Appendix A.

Extracting Files Only for Display

console

PKZIP gives you the option of displaying specific files contained in a .ZIP file to your computer monitor. For example, if you wish to view the contents of all of the .txt files contained in a .ZIP file, type the following and press ENTER:

```
pkzipc -console test.zip *.txt
```

In this example, all files with a .txt extension that exist in the test.zip are displayed on the monitor. Since many .ZIP files contain an information document (e.g., readme.txt), the **console** option is a good way to determine the contents of a .ZIP file without requiring you to extract a file or file(s) to your hard drive.

Note: You can also use the **console** and **silent** options to redirect files to pipe files directly to another program on Windows NT-based systems.

Extracting Files with a List File

The file format of a list file used to extract or exclude certain files is somewhat different than the format used to include files. When compressing files, the list file needs to be in a format that the operating system can understand. For example, in DOS based command lines, it may be necessary to specify a drive letter when compressing files. By contrast, when extracting or excluding files, a drive letter is not necessary and therefore cannot be used in list files.

A list file may contain wild card specifications (*,?) as well as exact file names and paths. An example of a list file used in extract operations follows:

```
*. exe
*. doc
temp/readme. doc
temp/?????. *
text/news. txt
```

Using the @ character in your command line, you can point to the list file. Assume that the list file is called lst.txt and is located in your current directory. An example of such a command line follows:

```
pkzipc -extract test.zip @lst.txt
```

In the example above, PKZIP extracts files from test.zip using file information it retrieves from a file called lst.txt, located in the current directory. It is important that the file format contained in the list file matches the format of the files in the test.zip file. For example, if test.zip contains no path information, specifying **text/news.txt** in your list file will *not* extract the file **news.txt** from the test.zip file. See the section on page 40 for more information on list files. For information on viewing the files as they appear in the .ZIP file, refer to the section “Viewing the Contents of a .ZIP File” on page 67.

Digital Signatures

PKZIP allows you to authenticate .ZIP files that have been signed with a standard X.509 digital certificate. A digital signature attached to a file confirms that the file is unchanged since it was signed.

You can use the **test** command on an archive to check for a signature before extracting files. Testing tells you whether files are signed, authenticates any signatures, and gives you information about certificates used to sign files. PKZIP also authenticates signatures automatically when extracting.

Signatures can be applied to particular files and/or to the central directory of an archive (that is, to the archive itself).

The following table lists warning messages that can be displayed when you test or extract signed files and thus cause PKZIP to authenticate signatures.

| <i>Message</i> | <i>Explanation</i> | <i>What to do?</i> |
|-----------------------------------|---|---|
| Signature is invalid | <p>The file or archive has changed since it was signed.</p> <p>The archive may be corrupt..</p> | <p>You may want to try to obtain the file again (for example, download the file again from the Web site).</p> <p>Contact the archive creator as the file/archive has been compromised. If the file was downloaded from a Web site, you may want to contact a person at that company about the file.</p> <p>If a file has an invalid signature, then the file may have been modified.</p> <p>If the central directory has an invalid signature, then file(s) have been modified, added or deleted from the archive since the archive was signed.</p> |
| Certificate is not trusted | <p>The certificate used to sign is currently not to be trusted.</p> | <p>This message indicates that the certificate is not to be trusted, but there may be no problem with the archive.</p> <p>Contact the issuer of the certificate to validate the certificate/signature.</p> |
| Certificate is expired | <p>The certificate has expired (perhaps because the archive was signed a long time ago).</p> | <p>Contact the owner of the certificate.</p> <p>This message indicates that the certificate is not to be trusted, but there may be no problem with the file or archive.</p> |
| Certificate is revoked | <p>Indicates the issuer has revoked the certificate.</p> | <p>Contact the issuer or owner of the certificate.</p> <p>This message indicates that the certificate is not to be trusted, but there may be no problem with the file or archive.</p> |
| Certificate not found: XXX | <p>The certificate for the signature could not be found on your system.</p> | <p>Check to see if the certificate name was misspelled.</p> <p>Confirm that the certificate is on the system.</p> |

5

Miscellaneous Operations

Introduction

This chapter describes commands and options that are not tied specifically to compressing or extracting or can be done with both of these operations.

Overwriting Files

overwrite

When you add or extract files, the target archive or directory may already contain files that have the same names as the files you are adding or extracting. Use the ***overwrite*** option to tell PKZIP how to proceed. Available choices are represented by the sub-options described in the following table.

| <i>Sub-Option</i> | <i>Description</i> | <i>For example</i> |
|----------------------|--|--|
| <i>all</i> | (Default) PKZIP overwrites all same-named files without prompting first | <code>pkzipc -extract -overwrite=all test.zip *.bmp</code> <code>pkzipc -add -overwrite test.zip *.bmp</code> |
| <i>prompt</i> | PKZIP prompts you whether to overwrite a same-named file before proceeding | <code>pkzipc -extract -overwrite=prompt test.zip *.bmp</code> <code>pkzipc -add -overwrite=prompt test.zip *.bmp</code> |
| <i>never</i> | PKZIP does not overwrite any same-named files | <code>pkzipc -extract -overwrite=never test.zip *.bmp</code> |

If you use the ***add*** or ***extract*** command alone, without the ***overwrite*** option, you are prompted to overwrite same-named files. If you use the ***overwrite*** option but do not specify a sub-option, PKZIP overwrites all files without prompting you.

Viewing the Contents of a .ZIP File

view

PKZIP allows you to view the contents of a .ZIP file, without performing any action on that .ZIP file (for example, compress or extract). To view a .ZIP file, use the **view** option with PKZIP, as in the following example:

```
pkzipc -view test.zip
```

When you type this command, information similar to the following appears:

```
Viewing .ZIP: test.zip
```

| Length | Method | Size | Ratio | Date | Time | CRC-32 | Attr | Name |
|--------|----------|-------|-------|------------|-------|----------|-------|---------|
| 8369B | Defl atN | 3084B | 63.2% | 06/01/2001 | 4:50a | 87b3c388 | -a-w- | red.txt |
| 8369B | Defl atN | 3084B | 63.2% | 06/01/2001 | 4:50a | 87b3c388 | -a-w- | tan.txt |
| 16KB | | 6168B | 63.2% | | | | | 2 |

PKZIP also provides two additional methods for displaying information from a .ZIP file. Specify the desired method as a value in addition to the **view** option. These methods include:

- **brief** - a compact, less informative view of the .ZIP file.
- **detail** - more information than the default view.

Displaying a Brief View of a .ZIP File

To display a more compact (brief) view of a .ZIP file, use the **brief** value with the **view** option, as in the following example:

```
pkzipc -view=brief test.zip
```

When you press ENTER, information similar to the following appears:

```
Viewing .ZIP: test.zip
```

| Length | Method | Size | Ratio | Date | Time | Name |
|--------|----------|-------|-------|------------|-------|---------|
| 8369B | Defl atN | 3084B | 63.2% | 06/01/2001 | 4:50a | red.txt |
| 8369B | Defl atN | 3084B | 63.2% | 06/01/2001 | 4:50a | tan.txt |
| 16KB | | 6168B | 63.2% | | | 2 |

Displaying a Detailed View of the .ZIP File

To display a more detailed view of a .ZIP file, use the **details** value with the **view** option, as in the following example:

```
pkzipc -view=details test.zip
```

When you press ENTER, information similar to the following appears:

```

Viewing .ZIP: test.zip

      FileName: red.txt
      FileType: text
      Attributes: -a-w-----
      Date and Time: Jun 01, 2001  4: 50: 00a
      Compression Method: Defl atN
      Compressed Size: 3084
      Uncompressed Size: 8369
      Compression: 63.2% - 2.948 bits/byte
      32 bit CRC value: 87b3c388
      Version created by: PKZIP: 4.5
      Needed to extract: PKZIP: 2.0 or later

      FileName: tan.txt
      FileType: text
      Attributes: -a-w-----
      Date and Time: Jun 01, 2001  4: 50: 00a
      Compression Method: Defl atN
      Compressed Size: 3084
      Uncompressed Size: 8369
      Compression: 63.2% - 2.948 bits/byte
      32 bit CRC value: 87b3c388
      Version created by: PKZIP: 4.5
      Needed to extract: PKZIP: 2.0 or later

-----

      Total Files: 2
      Compressed Size: 6168
      Uncompressed Size: 16738
      Compression: 63.2% - 2.948 bits/byte

```

Printing the Contents of a .ZIP File

print

PKZIP gives you the option of printing files contained in a .ZIP file to a selected printer. For example, if you wish to print all of the .txt files contained in a .ZIP file, type the following:

```
pkzipc -print=lpt1 test.zip *.txt
```

When you press ENTER, information similar to the following will appear:

```

Extracting files from .ZIP: test.zip

      Inflating: readme.txt <to LPT1>
      Inflating: whatsnew.txt <to LPT1>

```

In this example, all files with a .txt extension that exist in the test.zip are printed to the LPT1 printer. If you do not specify a print device, the 'default' printer is used. Since many .ZIP files contain an information document (e.g., readme.txt), the *print* option is a good way to determine the contents of a .ZIP file without requiring you to extract a file or file(s) to your hard drive.

Testing the Integrity of a .ZIP File

test

You can test an archive to confirm that it is not damaged and that its files can be extracted. Testing also authenticates any digital signatures attached.

Testing extracts the contents of an archive but discards the output instead of saving it to disk.

It's a good idea to test an archive before you delete your only copy of an important file you placed in the archive.

The following sample command line tests `test.zip`:

```
pkzipc -test test.zip
```

When you press ENTER, information similar to the following will appear:

```
Testing files from .ZIP: test.zip
Testing: readme.txt      OK
Testing: whatsnew.txt   OK
```

As each file is tested, an OK is displayed next to the name. If, for some reason, the archive has been damaged, use the **fix** option to repair the .ZIP file.

Checking for Revoked Certificates

crl

Digital certificates used to apply signatures and to do recipient-based encryption are issued by a certificate authority (CA).

Periodically, CAs publish lists of certificates that have been revoked for one reason or another. For example, an employer might request revocation of a certificate that belongs to an employee who has left the company. Or revocation might be requested for a certificate that has been lost or stolen with its private key.

A CA's list of revoked certificates is called a *certificate revocation list* (CRL). It consists of a file that contains serial numbers of certificates that have been revoked and the dates. The CRL is signed by the issuing CA.

The **crl** option tells PKZIP to check to see if any certificate to be accessed by the current command line (to authenticate a signature, for example) appears in a CRL accessible to PKZIP. If it does, PKZIP displays a warning, (*W42*) *Certificate was revoked*.

Note: CAs periodically update CRLs. The fact that you can use the **crl** option and not receive a warning only guarantees that the certificate you accessed is not on a CRL that PKZIP checked. The certificate could still have been revoked subsequent to publication of your list.

The following sample command line checks any certificates used for signatures in an archive to be extracted:

```
pkzipc -extract -crl test.zip
```

You can configure the *crl* option so that it is used by default.

Obtaining a CRL

Certificate authorities commonly make CRLs available for downloading on their Web sites. A CA is apt to provide different CRLs for different series or types of certificates. You must find the CRL for the type of certificate that you want to use it for.

For PKZIP to access a CRL, the CRL must be downloaded and imported into a certificate store that PKZIP checks for certificates. Such a downloaded and imported CRL is called a *static* CRL to distinguish it from a *dynamic* CRL that may be published on the Web. PKZIP does not access CRLs published on the Web.

In Windows, you can import a CRL by double-clicking the downloaded file.

Pausing on Warnings

warning

PKZIP, issues an error or a warning when it encounters a problem or unexpected condition. In general, PKZIP issues a warning when the condition does not prevent PKZIP from completing its operation, and an error when it does. For example, PKZIP issues a warning if a digitally signed file in an archive cannot be authenticated; this condition does not prevent PKZIP from extracting the file. PKZIP issues an error if it cannot find a specified archive or is unable to open it.

The *warning* option causes PKZIP to pause after issuing a warning and to prompt you whether to proceed. The option can be set for specified warning conditions. If used without any specified values, the *warning* option causes PKZIP to pause on every warning. For example:

```
pkzipc -extract -warning save.zip *
```

To have PKZIP pause and prompt on particular warnings, list the warning numbers with the option. For example, the following command line directs PKZIP to pause on warning 41 (*Certificate expired*):

```
pkzipc -test -warning=41 save.zip
```

To specify multiple warning conditions, separate the warning numbers with commas. For example, the following command line tells PKZIP to pause and prompt on either warning condition 40 (*Certificate not trusted*) or 41:

```
pkzipc -test -warning=40,41 save.zip
```

You can use the *configuration* command to specify warning numbers as default values for the *warning* option. If default warning values are specified, you do not

need to explicitly include the **warning** option in a command line to pause on those warnings.

To override a particular configured default warning setting for the **warning** option in the current command line, precede the warning number with a hyphen. For example, the following setting (in a command line) overrides a configured value of warning 43. The example causes PKZIP *not* to pause on warning 43.

```
-warning=42,-43
```

The **warning** option can be used with the **add**, **extract**, **test**, and **view** commands. See Appendix B for a list of error and warning conditions.

Treating Warnings as Errors

error

The **error** option enables you to designate warnings, by number, to treat as errors such that PKZIP halts processing if a specified warning condition is encountered.

A designated warning is treated as error number 73, *Warning configured as an error*.

Multiple warning numbers can be specified, separated by commas:

```
-error=40,41
```

For example, the following command line tells PKZIP to treat the conditions that produce warnings 40 (*Certificate not trusted*) and 41 (*Certificate expired*) as error conditions:

```
pkzipc -extract -error=40,41 save.zip
```

If a specified warning is generated, PKZIP halts processing. Both the triggered warning and an error 73 are issued.

For example, if warning 40 is generated, the display looks like this:

```
PKZIP: (W43) Warning! Certificate not trusted
PKZIP: (E73) Warning configured as an error
```

You can use the **configuration** command to specify warning numbers as default values for the **error** option. If default warning values are specified for the **error** option, you do not need to explicitly include the **error** option in a command line to treat those warnings as errors.

You can override a particular configured default warning setting for the **error** option in the current command line. To override a warning setting, precede the warning number with a hyphen.

The following example (in a command line) overrides a configured value of warning 43. The example causes warning 43 *not* to be treated as an error.

```
-error=42,-43
```

The **error** option can be used with the **add**, **extract**, **test**, and **view** commands. See Appendix B for a list of error and warning conditions.

Previewing Command and Option Operations

preview

PKZIP allows you to preview the results of a set of commands and options. The commands and options specified will be completed and the resulting output will display, but no changes will be made that result in creating a new .ZIP file or in modifying an existing .ZIP file. For example, if you wish to preview an add operation without actually creating or modifying any files, enter the following:

```
pkzipc -add -preview test.zip *.txt
```

When you press ENTER, information similar to the following appears on your console:

◆ Using Preview Option

```
Creating .ZIP: test.zip
Adding File: readme.txt Deflating (62.0%), done.
Adding File: whatsnew.txt Deflating (59.2%), done.
```

The compressed .ZIP file size would be: 2237 bytes

The information, including the size of the resulting .ZIP file, is displayed. However, PKZIP has not actually modified any of your files. The *preview* option will work with the *add*, *delete*, *header*, and *comment* commands.

Fixing a Corrupt .ZIP File

fix

In the event that a .ZIP file becomes damaged, you may find that it is not possible to extract or perform other PKZIP operations on the contents of the file. The *fix* option in PKZIP will attempt to repair damaged .ZIP archives.

For example, if you have determined that test.zip is damaged, type the following to attempt to fix it:

```
pkzipc -fix test.zip
```

When you press ENTER, information similar to the following appears on your console:

```
Enter a new .ZIP file name (pkfixed): test1.zip
Running PKZipFix utility.
Scanning .ZIP file: test.zip
Building new directory.
Writing new .ZIP file: test1.zip
Recovered 2 files.
```

Please note that when you enter the *fix* option, PKZIP will prompt you to enter a new .ZIP file name. In the above example, test1.zip was entered. If you do not enter a file name, the name "pkfixed.zip" will be used. PKZIP scans the original file, attempts to repair the archive, and saves the updated file with the file name provided. The

original, damaged file is not updated. The **fix** option will not repair all damaged .ZIP files. Depending on the degree of damage to the data within a .ZIP file, you may not be able to recover your files from that archive.

Create a Temporary .ZIP File on a Alternate Drive

temp

Every time you update a .ZIP file, PKZIP creates a temporary work file. Before modifying the original file, PKZIP performs all of its compression and extraction operations on the temporary work file. When the modifications to the .ZIP file are successfully completed, the original .ZIP file is replaced with the updated file (temporary work file). This means you must have as much additional disk space available as was used by the original .ZIP file. For example, if you have an existing .ZIP file of 500K and you are adding another file to it that is 10K compressed, you need additional workspace of at least 510K during the update process.

The **temp** option allows you to create the temporary .ZIP file on a drive other than the one on which the original .ZIP file resides. This allows you to update large .ZIP files when space is limited, such as a large .ZIP file on a floppy disk. Furthermore, by setting this temporary drive to point to a RAM drive, you can speed up the operation of PKZIP.

Immediately following the **temp** option, place the drive and/or path you wish to use for the temporary work file as in the following example:

```
pkzipc -add -temp=z:/public test.zip readme.doc
```

Note: It is necessary to specify a path in addition to the drive letter only if you are in a situation where disk space or access is being limited by subdirectory, such as on a local area network.

Suppressing Screen Output

silent

The **silent** option suppresses screen output when compressing or extracting. This option is useful when compressing or extracting files as part of .BAT, .CMD, or shell script operations. Messages that normally appear when compressing or extracting are not displayed. Sub-options provide control over whether to display error messages, warning messages, requests for input, and so on.

```
pkzipc -add -silent test.zip *.doc
```

To suppress confirmation messages printed by the **configuration** command, use the **configuration** command with its own **silent** sub-option.

Setting Internal Attributes

ASCII/BINARY

The **ASCII** and **BINARY** option is used to override the data type of a file. Normally, PKZIP will determine whether the data of a file is ASCII or Binary. If this option is used with no sub option, each file that is added, you will be prompted for the file to be set to ASCII, BINARY or if you want PKZIP to determine the best type. The following examples show the different uses for this option.

To set all the internal attributes to ASCII for each file added:

```
pkzipc -add -ascii="" test.zip
```

To set all the internal attributes for the file test.txt to BINARY and auto detects the other files:

```
pkzipc -add -binary=test.txt test.zip *
```

To prompt the type for each file:

```
pkzipc -add -ascii test.zip *
```

Encoding an Archive to Another Type

encode

With the **encode** option, you can convert an archive from one type to another.

The **encode** option is useful to encode a binary archive type to a text format such as UUEncode or XXEncode. It can also be used to compress a non-compressed archive into a compressed archive type.

For example, a TAR archive can contain multiple files but does not compress them, and a GZIP archive compresses but can contain only one file. You can use **encode** with **add** to create (or update) a TAR archive and encode it to GZIP format:

```
pkzipc -add -encode=gzip myfiles.tar
```

The example creates two archives: a TAR file and a GZIP file `myfiles.tar.gz`.

If you want only the archive created by **encode** (the GZIP archive in the example), you can include the **movearchive** option to delete the intermediate (TAR) archive:

```
pkzipc -add -encode=gzip -movearchive myfiles.tar
```

Removing an Intermediate Archive

movearchive

The *movearchive* option deletes an archive that is created only as an intermediate archive—for example, to be converted by the *encode* option to an archive of a different type.

When you add files with the *encode* option, PKZIP creates two archives: an intermediate archive created by the *add* command, and an archive of the type specified with the *encode* option. The encoded archive is created from the intermediate archive.

If you do not want to keep the intermediate archive, you can include the *movearchive* option to delete it. For example:

```
pkzipc -add -encode=gzip -movearchive myfiles.tar
```

The command line above creates a TAR archive, encodes a copy of this archive as a GZIP archive, and then deletes the intermediate TAR archive.

Generate a List File

listfile

The *listfile* option is used with *add* and *extract* to create a list of the files that would be added or extracted if the command were run without the *listfile* option. With the option, no files are actually added or extracted.

For example, the following command creates a file *mylist.txt* with the names of all the files that would be added to, or updated in, *myarchive.zip* if the *listfile* option were omitted from the command:

```
pkzipc -add=update -listfile=mylist.txt myarchive.zip *
```

Similarly, when the option is used with *extract*, a list file is created containing the names of all the files that would have been extracted if the *listfile* option were not used.

The *listfile* option takes account of any other options used. For example, if files are to be extracted using stored path information, the path information will appear in the list file as well, but if files are to be extracted *without* using stored path information, no path information will appear in the list file either.

6

Changing Defaults for Commands and Options

Default values for PKZIP commands and options are stored in a configuration file. You can view current settings by using the **configuration** command. You also use this command to change default values for any command or option that has a default. Another command—**default**—lets you restore default settings for all commands and options to their original values.

With the **altconfig** option, you can create and use alternate configuration files for special purposes.

Viewing the Configuration File

Use the **configuration** command to view current default values for commands and options. Enter the command by itself, with no options, to view all defaults. For example:

```
pkzipc -configuration
```

A screen similar to the following appears:

```

204 = Di sabl ed
Archi veDate = None
Comment = None
Comp Method = Defl ate
Extract = Extract All Files
Li stChar = @
Lowercase = Di sabl ed
NoArchi veExtensi on = Di sabl ed
NoFix = Di sabl ed
Opti onChar = -
Recurse = Di sabl ed
Sort = None
Test = Extract All Files
Transl ate = None - No Conversi on

Wi pe = None
Add = Add All Files
CD = Normal
Encode = Di sabl ed, UUE
Level = 5
Local e = Di sabl ed
More = Di sabl ed
NoExtended = Di sabl ed
Password = Di sabl ed
Shortname = None
Span = None, Auto-Detect
Times = All
Vi ew = Normal

ASCI I = Di sabl ed
Bi nary = Di sabl ed
CryptAl gori thm = Tradi ti onal
Embedd ed = Di sabl ed
Error = Di sabl ed
Header = Di sabl ed
LDAP = Di sabl ed
Si lent = None
Temp = Di sabl ed

```

Warning = Disabled

Compression Options

After = Disabled
 Attributes = Read-Only, Archive
 Before = Disabled
 Exclude = Disabled
 Include = Disabled
 Larger = Disabled, 0
 Mask = None
 Newer = Disabled
 Older = Disabled
 Overwrite = Always Overwrite
 Path = No Path Information
 Smaller = Disabled, 18, 446, 744, 073, 709, 551, 615

Extraction Options

After = Disabled
 Attributes = Read-Only, Hidden, System, Archive
 Before = Disabled
 Exclude = Disabled
 Include = Disabled
 Larger = Disabled, 0
 Mask = None
 Newer = Disabled
 Older = Disabled
 Overwrite = Prompt
 Path = Full Path
 Smaller = Disabled, 18, 446, 744, 073, 709, 551, 615

In this display, the commands/options are to the left of the equal sign, and the default settings are to the right.

With options that take sub-options, the default setting is generally a sub-option. For example, the **overwrite** option shows a default of **prompt**, which is one of its possible sub-options. Some other options are set to **none**. An option that does not have a **none** sub-option shows as *Disabled* when it is not configured. For example, **After** shows a date if the option is configured and *Disabled* if it is not.

At the bottom of the listing of defaults are two sets of *filter options*, one for compression and one for extraction. These are called filter options because they filter out files that do not meet their criteria. Only files that are not filtered out are selected. For example, the **after** option filters out all files whose date falls before the date specified with the option.

Each of the filter options takes a different default value for compression and for extraction.

For information on changing defaults, refer to the section below, *Changing a Default Value*.

Changing a Default Value

To change a default setting in the configuration file, use the **configuration** command. You can abbreviate this command to: **config**.

To specify a value (sub-option) to use as the default value for a command/option:

- Type **pkzipc –config** and the name of the command/option followed by an equal sign and the sub-option value you want to set as the default.

For example, to change the default for the **add** command to **update** (instead of the original default, **all**), type the following:

```
pkzipc -config -add=update
```

To turn on and use by default an option that has either no sub-options or a sub-option that is used by default:

- Type **pkzipc -config** and the name of the option.

For example, to do virus scanning by default when extracting files, set the **avscan** option on by default:

```
pkzipc -config -avscan
```

To turn on the **silent** option and use its default sub-option:

```
pkzipc -config -silent
```

After you use the **configuration** command to change a default setting, an updated list of settings displays. You can suppress this list so that it is not displayed. To do so, use the **configuration** command with its **silent** sub-option.

For example, the following command line sets a default value for the **overwrite** option and suppresses display of the updated list of settings that the **configuration** command ordinarily prints to the screen:

```
pkzipc -config=silent -overwrite=never
```

Note that the **silent** sub-option of the **configuration** command is different from the **silent** option proper, which suppresses messages when adding or extracting.

See Appendix A for a list of PKZIP commands, options and sub-options, and information about which commands and options have configurable defaults.

Changing Defaults for Filter Options

Options listed as filter options in the display of default settings take separate defaults for compression and extraction. To specify a default for a filter option for one of these operations, include the related command (**add** or **extract**) on the command line. For example:

```
pkzipc -config -add -newer=1d
```

If you specify a default for a filter option without including the related command, as in the following example, PKZIP asks whether you want to specify the default for compression, extraction, or both:

```
pkzipc -config -newer=1d
```

Changing Defaults for Compression Method

The *Comp Method* item in the screen of configuration settings shows the current default setting for compression method. To set a default compression method, specify the compression method that you want to make the default. For example, the following command makes BZIP2 the default compression method:

pkzipc -config -bzip2

The options in the table below set compression method:

| <i>Compression Method Options</i> | <i>Description</i> |
|-----------------------------------|--|
| <i>deflate64</i> | Sets the compression method to Deflate64 |
| <i>bzip2</i> | Sets the compression method to BZIP2 |
| <i>dclimplode</i> | Sets the compression method to DCL Implode |
| <i>store</i> | Sets the compression method to Store (that is, no compression) |

The options in the next table set both compression method and level:

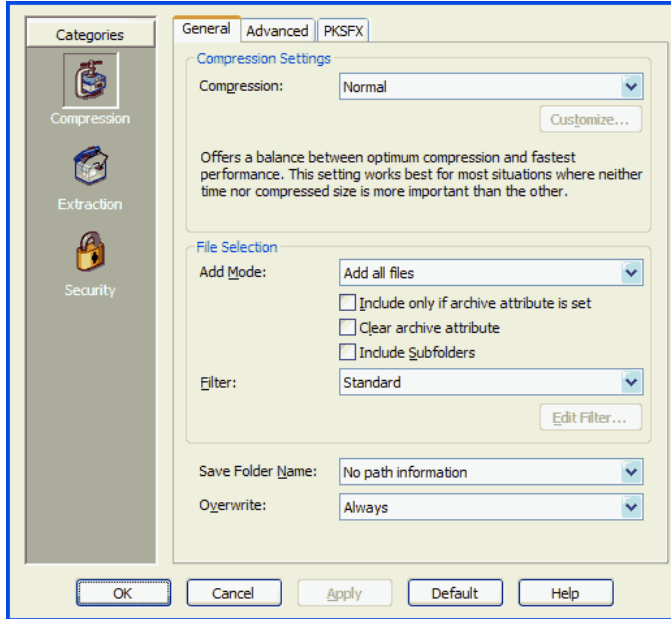
| <i>Option</i> | <i>Description</i> |
|-----------------------|--|
| <i>speed</i> | Sets the compression method to Deflate—the initial PKZIP default method—and the level of compression to 1 (the lowest) |
| <i>fast</i> | Sets the compression method to Deflate and the level of compression to 2 |
| <i>normal</i> | Sets the compression method to Deflate and the level of compression to 5. Normal is the initial default setting for compression method and level for PKZIP. |
| <i>maximum</i> | Sets the compression method to Deflate and the level of compression to 9 |
| <i>level=0</i> | When set to 0, the level option sets the compression method to Store (no compression) |

For example, the following command sets the default compression method to Deflate and the default compression level to 9:

```
pkzipc -config -maximum
```

Using the Options Dialog to Change Defaults

As an alternative to using the command line to change defaults, you can use the graphical Options dialogs if you have PKZIP for Windows installed:



To display the graphical Options dialog:

- Use the **configuration** command with the **gui** sub-option:

```
pkzipc -config=gui
```

In the dialog, the **Help** button opens the online help for the Windows version of PKZIP or SecureZIP. There you can read how to set options in the dialog.

Note that, when you use the **gui** sub-option to open the Options dialog from the command line, settings that you make in the dialog apply only to the command line version of the product, not to the Windows version. Similarly, if you open the Options dialog from the Windows version, options that you set in the dialog apply only to the Windows version.

If you use the **gui** sub-option without having PKZIP for Windows installed, the sub-option is ignored, and the command works as if you had entered it with no sub-option.

Resetting to Original Defaults

Command or option default values that you have changed can be reset back to their original values. You can reset changed defaults either for individual commands and options that you specify, or wholesale, for all.

Resetting Individual Defaults

To reset an individual command or option to its default value in the Configuration file, use the **-config** command and put two hyphens in front of the command or option that you want to reset.

For example, to reset the **add** value back to its default without resetting any other Configuration values that you may have modified, type the following and press ENTER:

```
pkzipc -config --add
```

Notice that there are *two* hyphens in front of the **add** command. The command changes the **update** value we set in a previous example back to **all**.

Resetting All Defaults

To reset default values for all commands and options, use the **default** command. Type the following and press ENTER:

```
pkzipc -default
```

Using an Alternate Configuration File

altconfig

You can create and use alternate configuration files to use for special purposes. The **altconfig** option creates such files and loads them. With an alternate configuration file, you can temporarily change multiple default command or option settings in a single pass just by loading the configuration file that defines them.

Creating an Alternate Configuration File

An alternate configuration file must be in valid format for a *Command Line XML* configuration file. The easiest way to create one is to use the **altconfig** option with the **configuration** command. This creates a copy of the current main configuration file with the file name and at the location specified by the **altconfig** option and updates default settings in the copy with any new settings specified in the **altconfig** command line. If an alternate configuration file of the same name already exists at the specified location, the file is not overwritten. Instead, default settings in that file are updated with the specified new defaults from the command line.

For example, the command line below creates or updates an alternate configuration file `secure.xml` in the root directory of drive C and specifies default values for the **cryptalgorithm**, and **directories** options:

```
pkzipc -config --altconfig=c:\secure.xml --cryptalg=aes,256 --dir=current
```

If you have the graphical *PKZIP for Windows* installed, you can use **config=gui** to configure defaults in the graphical Options dialogs. For example, the following command line opens the Options dialogs:

```
pkzipc -config=gui --altconfig=c:\secure.xml
```

If `secure.xml` exists, PKZIP displays its settings in the graphical Options dialogs. If the file does not already exist, PKZIP displays the settings of your main configuration file. In either case, saving settings from the Options dialog saves to `secure.xml`.

Using an Alternate Configuration File

To use the settings in an alternate configuration file, use the ***altconfig*** option to specify the file in a command line with which you want to use the alternate settings.

You can use the ***altconfig*** option with any command. For example, the following command line loads the alternate configuration file `secure.xml` to use its settings with the ***add*** command. The settings cause PKZIP to use the strong encryption algorithm AES 256 when encrypting and to save path information relative to the current directory.

```
pkzipc -add -altconfig=c:\secure.xml -pass foo.zip *.doc
```

Loading the settings from the alternate configuration file saves the trouble of specifying them all on the command line and does not require changing the main configuration file.

An alternate configuration file must already exist for you to use it in a command line with the ***add*** command or any other command besides ***configuration***. The only time you can use the ***altconfig*** option to specify an alternate configuration file that does not already exist is when you use the option with the ***configuration*** command to create an alternate configuration file.

7

Command Characteristics

Introduction

This chapter describes changes you can make to the PKZIP infrastructure. For example, you can specify different characters to use for the list character and the option character, and you can cause PKZIP to display dates and times using a different format from the one used by default on your system.

Ordinarily, the original values for the settings described in this chapter should be satisfactory. You should not change them without a good reason.

Changing Date and Time Environment Variables

locale

The *locale* option causes PKZIP to use your system's format for displaying dates and times. This option is set on by default.

Formerly PKZIP used a date format of MMDDYY and a 12-hour time format of HH:MM. If you prefer PKZIP to use this format, you can revert to it by turning the *locale* option off.

To disable the option in the Configuration file to turn it off by default, use the *configuration* command and specify the *locale* option prefixed with two hyphens:

```
pkzipc -config --locale
```

If you have turned off the *locale* option in the Configuration file, you can turn it on again for a given command by specifying the option in the command line. For example:

```
pkzipc -add -locale test.zip *.doc
```

This command causes PKZIP to use the system-defined settings regardless of the default settings in the Configuration file.

Changing the List Character for List Files

listchar

PKZIP allows you to specify an ASCII file as a source list of the files to be archived. By default, you specify this ASCII file by pointing to it with the "@" character in your command line. However, if you have files that begin with an "@", you may experience problems when trying to add these files to a .ZIP archive. Fortunately, PKZIP allows you to change the default list character to avoid such problems. This is accomplished using the *listchar* option. For example, if you wish to define the "+" character in place of the "@" as your default list character, type the following and press ENTER:

```
pkzipc -config -listchar=+
```

If you wish to specify an alternate list character on the command line itself, could type a command line similar to the following and press ENTER:

```
pkzipc -add -listchar=+ test.zip +file1.txt
```

When used as a command line option, the *listchar* option only applies to the options that follow it on that particular command line. In our example the *listchar* option allows you to add files that begin with a "+" character (e.g., +file1.txt). For more information on using list files with PKZIP see the section on page 40 and the "Extracting Files with a List File" section on page 63.

Note: Avoid using metacharacters as list characters. Metacharacters have a special significance to the shell and as such their usage may cause unpredictable results. This would include the following characters:

```
; , & ( ) | < > # NEWLINE SPACE TAB
```

Changing the Command/Option Character

optionchar

The *optionchar* option specifies the character to use to identify commands and options as such in command lines. By default, PKZIP uses the hyphen "-" to flag commands and options in a command line. You can use *optionchar* to change this option character to a different character instead. For example, to make it easier to zip files whose names begin with a "-", you might change the option character to a "+".

You can change the option character either just for a single command line or indefinitely, to define a new default character. The following command changes the option character just for the immediate command:

```
pkzipc -opt=+ +add save.zip *.doc
```

In a Windows command line, you can also always use the "/" character to indicate a command or option in a particular command line.

```
pkzipc /add save.zip *.doc
```

You can also use *optionchar* with the *configuration* command to define a different option character to use by default. For example:

```
pkzipc +config -optionchar=+
```

Note that the newly defined option character is used immediately, in the same command line in which it is defined, by every command or option other than *optionchar* itself.

Note: Avoid using metacharacters as option characters. Metacharacters have a special significance to the shell and as such their usage may cause unpredictable results. This would include the following characters:

```
; , & ( ) | < > # NEWLINE SPACE TAB
```

A

Reference to Commands and Options

This appendix contains reference information on every PKZIP command and option. For each command/option, the following information is provided:

| <i>Category:</i> | <i>Represents:</i> |
|-------------------------|--|
| Name/Description | The full name of the command/option and a brief description of what that command/option does. If the command/option can be configured (defaulted) in the Configuration file, the word "Configurable" appears after the description. |
| Value(s) | The value(s) associated with this command/option, including the "default" value for each. If a command/option does not have an associated value, the phrase "no suboptions" appears. |
| Example usage | An example of how to include this command/option in your PKZIP command line, including possible abbreviations. For most options, you can abbreviate the command/option. These abbreviations are illustrated in the examples used in this appendix. |
| Used with | This command can be used for compression, extraction, viewing, testing, a combination, or as a standalone (none of the above). |

Information on each command/option follows:

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|---|---|-----------------------------|-------------------|
| 204 Turns on PKZIP for DOS 204g compatibility Configurable | No sub-options. ----- No default value. | pkzipc -add -204 save.zip * | add |

| Name/Description: | Value(s): | Example usage: | Used with: |
|---|---|--|--|
| <p>add</p> <p>Add files to an archive file</p> <p>Configurable</p> | <p>all – Compress and add files that are new to the archive as well as files that the archive already contains a copy of</p> <p>archive – Turn off archive attribute of all added files (prepares backup file set for incremental archiving).</p> <p>freshen – Add only files that the archive already contains an older copy of</p> <p>update – Freshen files that are in the archive already and add any new ones</p> <p>incremental – Add only files that have the archive attribute on, and then turn off the archive attribute</p> <p>-incremental – Add only files that have the archive attribute on, and do not turn off the archive attribute afterward</p> <p>-----</p> <p>Default = all</p> | <p>pkzipc -add save.zip *.doc</p> <p>pkzipc -add=freshen save.zip *.doc</p> <p>pkzipc -add=increm save.zip *.doc</p> <p>pkzipc -add=-incremental save.zip *.doc</p> <p>Outputs the archive to STDOUT instead of to a file:</p> <p>pkzipc -add -noarchiveextension -silent=normal - *.txt</p> | <p>standalone</p> |
| <p>after</p> <p>Process files that have the specified date or a later one</p> <p>Configurable separately for add and extract operations.</p> | <p>Any date in format specified in Country-Settings or the locale option.</p> <p>For example, the US date format is:</p> <p>mmdyy</p> <p>or</p> <p>mmdyyyy</p> <p>-----</p> <p>No default value.</p> | <p>For compression:</p> <p>pkzipc -add -aft=09152003 save.zip *.doc</p> <p>For extraction:</p> <p>pkzipc -ext -after=09152003 save.zip *.doc</p> | <p>add, extract, delete, test, view, delete, console</p> |

| Name/Description: | Value(s): | Example usage: | Used with: |
|--|--|---|---|
| <p>altconfig</p> <p>Creates or updates an alternate configuration file containing alternate, specified defaults when used with the <i>configuration</i> command; loads the specified alternate configuration file when used in a command line with any command other than <i>configuration</i>.</p> | <p>Path and name of alternate configuration file to create, update, or load</p> | <p>Create or update an alternate configuration file <i>secure.xml</i> with specified defaults. File is created if it does not exist already, or is updated if it does:</p> <pre>pkzipc -config -altconfig=c:\secure.xml -cryptalg=aes,256 -dir=current</pre> <p>Use the default settings specified in alternate configuration file <i>secure.xml</i> when adding files to archive <i>foo.zip</i>:</p> <pre>pkzipc -add -altconfig=c:\secure.xml -pass foo.zip *.doc</pre> | <p>All commands except list-cryptalgorithms, license, and version</p> |
| <p>archivedate</p> <p>Set the file modification date of the archive file.</p> <p>Configurable</p> <p>Note: The <i>archivedate</i> option is the same as the older zipdate option, which is now deprecated.</p> | <p>newest – set to the date of the newest file within the archive file.</p> <p>oldest – set to the date of the oldest file in the archive file.</p> <p>retain – retain the original date of the archive file (the date when the file was created).</p> <p>none – disable the file date in the configuration file and set the archive date as the last modification date.</p> <p>-----</p> <p>Default = the current date.</p> <p>Default if used on command line without a sub-option = retain.</p> | <pre>pkzipc -add=update -archivedate=retain save.zip *.txt</pre> | <p>add, delete, fix, header, comment</p> |

| Name/Description: | Value(s): | Example usage: | Used with: |
|--|--|---|------------|
| <p>archiveeach</p> <p>Creates a separate archive for each of multiple files specified in a single command line.</p> <p>Can be used with <i>archivetype</i> and <i>encode</i> to create .tar.gz archives.</p> | <p><destination> – Directory in which to create the archives</p> <p>By default, archives are created in the current directory.</p> | <p>Creates a separate ZIP archive for each file in the current directory:</p> <pre>pkzipc -add -archiveeach *.*</pre> <p>Creates the archives in a specified destination:</p> <pre>pkzipc -add -archiveeach=C:\newzips *.*</pre> <p>Creates .tar.gz archives:</p> <pre>pkzipc -add -archiveeach -archivetype=tar -encode=gz C:\data*.*</pre> | add |
| <p>archivetype</p> <p>Specifies the type of archive to create. Normally, the archive type can be inferred from the specified name of the file to create. It is necessary to use the <i>archivetype</i> option only if the archive type cannot be inferred (for example, because the file name has no extension).</p> <p>If this option is not specified and the file name does not indicate the archive type, a ZIP archive is created.</p> <p>Configurable</p> | <p>bzip2 – Specifies the Bzip2 archive type.*</p> <p>zip – Specifies the .ZIP archive type. (default)</p> <p>gzip – Specifies the GZIP archive type.*</p> <p>tar – Specifies the TAR archive type.</p> <p>uue – Specifies the UUENCODED archive type.*</p> <p>xxe – Specifies an XXENCODED archive type.*</p> <p>* These archive types can contain only one file. To use with multiple files, create an archive of one of the other archive types and use the <i>encode</i> option to encode this archive as the single-file archive type that you want.</p> | <pre>pkzipc -add -archivetype=tar myfile.foo</pre> <p>Creates a TAR archive named <code>myfile.foo.tar</code></p> | add |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|--|--|---------------------|
| <p>ascii</p> <p>Set the internal attribute bit (ASCII/Binary) to ASCII.</p> <p>Configurable</p> | <p>The file(s) or file pattern whose internal attribute bit you wish to set to ASCII; if no files are specified, PKZIP prompts for each file.</p> <p>-----</p> <p>No default value.</p> | <pre>pkzipc -add -ascii="*.txt" save.zip *</pre> <pre>pkzipc -add -ascii save.zip *</pre> | <p>add</p> |
| <p>attributes</p> <p>Stores files with the specified file attribute information in the archive file.</p> <p>Configurable separately for add and extract operations.</p> | <p>hidden – select hidden files.</p> <p>system – select system files.</p> <p>readonly – select read-only files.</p> <p>archive – select files with the archive bit set.</p> <p>all – select all types of files.</p> <p>none – do not select files that have hidden, system, or read-only attributes; overrides the default attributes setting in configuration file.</p> <p><hex value> –The hex value of an attribute to be selected, or the logical OR of multiple hex values</p> <p>-----</p> <p>Default = readonly, archive</p> | <pre>pkzipc -add -attr=system,hidden save.zip *</pre> | <p>add, extract</p> |
| <p>avargs</p> <p>Specifies any command line arguments to use when running the anti-virus program given in avscan</p> <p>Configurable</p> | <p><command line> – A command line that runs an anti-virus program</p> | <pre>pkzipc -extract -avscan= f-prot.exe -avargs="%e /silent /nomem /noboot" myfiles.zip</pre> | <p>extract</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|---|---|---|
| <p>avscan</p> <p>Turns on virus scanning: runs the specified anti-virus program using the anti-virus command line arguments in avargs</p> <p>Configurable</p> | <p><executable> – The name of the anti-virus program executable—with path, if necessary</p> | <pre>pkzipc -extract -avscan= f-prot.exe -avargs="%e /silent /nomem /noboot" myfiles.zip</pre> | <p>extract</p> |
| <p>before</p> <p>Process files that are older than a specified date.</p> <p>Configurable separately for add and extract operations.</p> | <p>Any date in format specified in Country-Settings or the locale option.</p> <p>For example, the US date format is one of the following:</p> <p>mmddyy mmddyyyy</p> <p>-----</p> <p>No default value</p> | <p>For compression:</p> <pre>pkzipc -add -bef=09152003 save.zip *.doc</pre> <p>For extraction:</p> <pre>pkzipc -ext -bef=09152003 save.zip *.doc</pre> | <p>add, extract, delete, test, view, print, console</p> |
| <p>binary</p> <p>Treats the files to be added as binary files: sets the internal ASCII/Binary attribute bit of the files to binary.</p> <p>Configurable</p> | <p>The file(s) or file pattern whose internal attribute bit you wish to set to binary; if no files are specified, PKZIP will prompt for each file.</p> | <pre>pkzipc -add -binary="*.exe" save.zip *</pre> <pre>pkzipc -add -binary save.zip</pre> | <p>add</p> |
| <p>bzip2</p> <p>Compress files using the BZIP2 method.</p> <p>Note: Files compressed with this method can be extracted with most varieties of PKZIP version 4.6 and later. Other .ZIP programs may not be able to extract files compressed with BZIP2.</p> | <p>No sub-options</p> <p>Default compression level: 5</p> | <p>To compress files using the bzip2 algorithm and level 9 compression:</p> <pre>pkzipc -add -bzip2 -level=9 save.zip doc1.txt</pre> <p>To compress files using the default compression level (level 5):</p> <pre>pkzipc -add -bzip2 save.zip *.doc</pre> | <p>add</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|--|--|----------------------------|
| <p>cd</p> <p>Encrypt file names and other metadata in a ZIP archive's central directory.</p> <p>Requires the <i>password</i> also to be used. Uses strong encryption; does not work with traditional ZIP encryption.</p> <p>Encrypting file names produces an archive that requires PKZIP or SecureZIP version 8.0 or later to open it.</p> <p>Configurable</p> | <p>encrypt – Encrypt file names and the archive's central directory</p> <p>normal – Do not encrypt file names; produces a normal ZIP file. Use to override a configured default setting that would otherwise encrypt file names.</p> <p>-----</p> <p>Default = encrypt</p> | <pre>pkzipc -add -password=mysecret -cryptalgorithm=aes,256 -cd test.zip</pre> <pre>pkzipc -add -password=mysecret -cryptalgorithm=aes,256 -cd=normal test.zip</pre> | <p>add</p> |
| <p>comment</p> <p>Include a text comment for files within an archive file. When you run the command, PKZIP prompts you to enter the comment.</p> <p>Configurable</p> | <p>all – all files within .ZIP file.</p> <p>unchanged – only existing files that have not changed.</p> <p>add – only new files.</p> <p>freshen – only existing files.</p> <p>update – existing and new files.</p> <p>none – turn off comment.</p> <p>-----</p> <p>Default = add</p> | <pre>pkzipc -add -com=all save.zip *.doc</pre> | <p>add, standalone</p> |

| Name/Description: | Value(s): | Example usage: | Used with: |
|--|---|---|---|
| <p>configuration</p> <p>Defines default values for PKZIP commands and options</p> | <p><command or option> – Any configurable command or option</p> <p>GUI – Invokes the configuration dialogs from the graphical PKZIP product. If specified, no other command line arguments are processed for configuration except more and silent, which can be set to govern the screen display of configuration settings.</p> <p>silent – Suppresses list of configured settings that is ordinarily displayed after a command or option is configured.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc –config –extract=freshen</p> <p>To see the current configuration values, type:</p> <p>pkzipc –config</p> <p>To open the Configuration dialogs of the GUI product for use in setting configuration defaults:</p> <p>pkzipc –config=gui</p> <p>Configures overwrite option and suppresses display of settings afterward:</p> <p>pkzipc –config=silent –overwrite=never</p> <p>Configures silent option and suppresses display of settings afterward:</p> <p>pkzipc –config=silent –silent</p> | <p>standalone</p> |
| <p>console</p> <p>Extracts files to the screen (standard output) instead of to disk</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc –console save.zip *.txt</p> | <p>standalone</p> |
| <p>crl</p> <p>Warns if a certificate to be used for digital signing, encryption, or authentication is listed as revoked in an accessible CRL (certificate revocation list).</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc –extract –crl test.zip</p> | <p>add, comment, delete, extract, header, listfile, print, test, view</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|---|---|-------------------|
| <p>cryptalgorithm</p> <p>Encrypts files using the specified strong encryption algorithm.</p> <p>Configurable</p> | <p>The encryption algorithm to use, as listed in the output from the listcryptalgorithms command. This command lists the algorithms available to you.</p> | <p>Encrypt all files added with 128-bit AES using the specified password:</p> <pre>pkzipc -add -cryptalgorithm=aes,128 -password save.zip *.doc</pre> | add |
| <p>dclimplode</p> <p>Instructs PKZIP to use the data compression library compression scheme.</p> <p>Configurable</p> | <p>ascii – use with ASCII files.</p> <p>binary – use with BINARY or unknown data files.</p> <p>You need to also specify the size of the dictionary (1024, 2048, or 4096) by using a comma after the type of dictionary to use (ascii or binary).</p> <p>-----</p> <p>No default value</p> | <pre>pkzipc -add -dclimplode=ascii,4096 text.zip *.txt</pre> | add |
| <p>default</p> <p>Reset the original defaults in the configuration file for all commands and options</p> | <p>No sub-options</p> <p>No default value.</p> | <p>To reset all defaults:</p> <pre>pkzipc -default</pre> | standalone |
| <p>deflate64</p> <p>Compress files using the Deflate64 method.</p> <p>Configurable</p> <p>Note: Files compressed with this method can be extracted by most versions 2.5x and later of PKZIP, but not all ZIP programs from other vendors can extract such files.</p> | <p>No sub-options.</p> <p>No default value.</p> | <p>To compress files using Deflate64 algorithm and level 9 compression:</p> <pre>pkzipc -add -deflate64 -level=9 save.zip doc1.txt</pre> <p>To compress files using the normal, default compression level (level 5):</p> <pre>pkzipc -add -deflate64 save.zip *.doc</pre> | add |

| Name/Description: | Value(s): | Example usage: | Used with: |
|--|---|--|-------------------------|
| <p>delete</p> <p>Remove (delete) files from an archive</p> | <p><files> – Names or file name pattern of files to delete</p> <p>No default value.</p> | <p>For individual files:</p> <pre>pkzipc -del save.zip doc1.txt</pre> <p>For a specific file pattern:</p> <pre>pkzipc -del save.zip *.doc</pre> | standalone |
| <p>directories</p> <p>Store directory path names during compression, or recreate directory path names while extracting.</p> <p>Includes subdirectories (recurse).</p> <p>Configurable</p> <p>Note: Using this command is the same as combining the <i>path</i> and <i>recurse</i> commands.</p> | <p>current – Store the path from the current directory.</p> <p>root or full – Store the entire path beginning at the root of the drive; also referred to as "full" path.</p> <p>specify – Store path information for subdirectories under the specified directories</p> <p>relative – Store the directory path relative to the current working directory of the drive specified.</p> <p>none – No path information stored</p> <p>Default = none</p> <p>Default if used on command line without a sub-option = current</p> | <p>Compression example (assumes you are in "/wp"):</p> <pre>pkzipc -add -dir=root save.zip wp/docs/*</pre> <p>The path stored would be "wp/docs/".</p> <pre>pkzipc -add -dir=current save.zip wp/docs/*</pre> <p>The path stored would be: "docs/".</p> <p>Extraction example:</p> <pre>pkzipc -extract -directories save.zip /*</pre> | add, extract |
| <p>embedded</p> <p>Suppresses prompting whether to extract the contents of a lone embedded archive file of the type specified with the sub-option.</p> <p>The option can be used to tell PKZIP either to extract the contents of an embedded archive automatically, without prompting, or to never extract the contents of an embedded archive automatically.</p> | <p>arj – Extract the contents of an embedded ARJ archive without prompting</p> <p>-arj – Do not extract the contents of an embedded ARJ archive and do not prompt</p> <p>BinHex – Extract the contents of an embedded BinHex archive without prompting</p> <p>-BinHex – Do not extract the contents of an embedded</p> | <p>To extract an embedded ZIP file without prompting:</p> <pre>pkzipc -extract -embedded=zip outerarchive.zip</pre> <p>To suppress the prompt and not extract any embedded ZIP file:</p> <pre>pkzipc -extract -embedded=-zip outerarchive.zip</pre> | extract, console, print |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--------------------------|--|-----------------------|-------------------|
| Configurable | <p>BinHex archive and do not prompt</p> <p>bzip2 – Extract the contents of an embedded BZIP2 archive without prompting</p> <p>-bzip2 – Do not extract the contents of an embedded BZIP2 archive and do not prompt</p> <p>cab – Extract the contents of an embedded CAB archive without prompting (WIN32 ONLY)</p> <p>-cab – Do not extract the contents of an embedded CAB archive and do not prompt (WIN32 ONLY)</p> <p>gzip – Extract the contents of an embedded GZIP archive without prompting</p> <p>-gzip – Do not extract the contents of an embedded GZIP archive and do not prompt</p> <p>lzh – Extract the contents of an embedded LZH archive without prompting</p> <p>-lzh – Do not extract the contents of an embedded LZH archive and do not prompt</p> <p>rar – Extract the contents of an embedded RAR archive without prompting (WIN32 ONLY)</p> <p>-rar – Do not extract the contents of an embedded RAR archive and do not prompt</p> | | |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--------------------------|---|-----------------------|-------------------|
| | <p>(WIN32 ONLY)</p> <p>uue – Extract the contents of an embedded UUENCODED archive without prompting</p> <p>-uue – Do not extract the contents of an embedded UUENCODED archive and do not prompt</p> <p>xxe – Extract the contents of an embedded XXENCODED archive without prompting</p> <p>-XXE – Do not extract the contents of an embedded XXENCODED archive and do not prompt</p> <p>ZIP – Extract the contents of an embedded ZIP archive without prompting</p> <p>-ZIP – Do not extract the contents of an embedded ZIP archive and do not prompt</p> <p>-----</p> <p>No default value. A sub-option must be set or you are prompted.</p> | | |

| Name/Description: | Value(s): | Example usage: | Used with: |
|---|---|--|---------------------------------|
| <p>encode</p> <p>Converts an archive to the archive type specified by the sub-option</p> <p>Configurable</p> <p>Note: PKZIP creates two files when the <i>encode</i> option is invoked: an intermediate archive of the type specified for the <i>add</i> command (ZIP, by default), and an archive of the type specified for the <i>encode</i> option.</p> <p>Use the <i>movearchive</i> option with <i>encode</i> to remove (delete) the intermediate archive.</p> | <p>bzip2 – Creates a BZIP2 file</p> <p>gzip – Creates a GZIP file</p> <p>uue – Creates a UUENCODED file (default)</p> <p>xxe – Creates an XXENCODED file</p> <p>No default value.</p> | <p>Add files to save.zip and encode to the configured default <i>encode</i> format (UUE, initially):</p> <pre>pkzipc -add -encode save.zip *</pre> <p>Add files to a TAR archive and encode to a GZIP archive:</p> <pre>pkzipc -add -encode=gz save.tar</pre> <p>Encode the archive as a GZIP archive and delete the intermediate archive created by the <i>add</i> command:</p> <pre>pkzipc -add -encode=gz -movearchive save.tar *</pre> | <p>add</p> |
| <p>enterlicensekey</p> <p>Displays dialog in which to enter product license key</p> | <p>None</p> | <pre>pkzipc -enterlicensekey</pre> | <p>standalone</p> |
| <p>error</p> <p>Designates warning conditions, by warning number, to treat as error condition 73 (<i>Warning configured as an error</i>)</p> <p>Configurable</p> | <p><warning number> – One or more warning numbers, separated by commas. To override a warning number configured for the option (and thus <i>not</i> treat that warning as an error), precede the number with a hyphen.</p> | <pre>pkzipc -extract -error=42,43 files.zip</pre> <pre>pkzipc -extract -error=42,-43 files.zip</pre> | <p>add, extract, test, view</p> |

| Name/Description: | Value(s): | Example usage: | Used with: |
|---|--|--|---|
| <p>exclude</p> <p>Exclude files from being compressed or extracted.</p> <p>Configurable separately for add and extract operations.</p> <p>Note: You must specify a sub-option (for example, file pattern or list file name preceded by an appropriate list character "@") with the exclude option.</p> | <p>The file(s) or file pattern (for example, *.doc) being excluded.</p> <p>No default value.</p> | <p>Compression example:</p> <pre>pkzipc -add -excl="*.doc" save.zip</pre> <p>Extraction example:</p> <pre>pkzipc -ext -excl="*.txt" save.zip</pre> <p>Setting exclude default:</p> <pre>pkzipc -config -excl="*.txt"</pre> <p>Note: When you use the <i>exclude</i> option with the <i>configuration</i> command, PKZIP prompts you to configure the <i>exclude</i> default for <i>add</i> and/or <i>extract</i> operations.</p> | <p>add, extract, delete, test, view, print, console</p> |
| <p>extract</p> <p>Extracts files from an archive file</p> <p>Configurable</p> | <p>all – Extracts all files in an archive file</p> <p>freshen – Extracts only files in the archive that are newer versions of files that already exist in the target directory</p> <p>update – Extracts files in the archive that are newer versions of files that already exist in the target directory or that do not exist in the target directory</p> <p>Default = all</p> | <pre>pkzipc -ext=up save.zip</pre> | <p>standalone</p> |
| <p>fast</p> <p>Uses the Deflate algorithm and sets the level of compression to level 2 on a scale of 0 - 9. Files having the following extensions are added uncompressed: bz2, bzip2, cab, gz, gzip, rar, gif, jpeg, jpg, mp3, mpeg, mpg, sxw</p> <p>Configurable</p> | <p>No sub-options.</p> <p>No default value.</p> | <pre>pkzipc -add -fast save.zip *.doc</pre> <pre>pkzipc -config -fast</pre> | <p>add</p> |

| Name/Description: | Value(s): | Example usage: | Used with: |
|--|---|--|---|
| <p>fix</p> <p>Attempts to repair a corrupt ZIP archive file</p> | <p><filename> – The name of the ZIP archive to fix</p> <p>No default value.</p> | <p>pkzipc –fix save.zip</p> | <p>standalone</p> |
| <p>header</p> <p>Creates a comment for a ZIP archive file in the header area of the file</p> <p>Configurable</p> | <p><filename> – The file that contains the header comment. The file name must be prefixed with the ListChar symbol ("@" by default) to distinguish it from the other sub-option</p> <p><comment> – The literal comment to be used</p> <p>-----</p> <p>No default value.</p> | <p>To include literal text:</p> <p>pkzipc –add –header save.zip *.doc</p> <p>Note: when you type this command, PKZIP will prompt you for the header text</p> <p>To include an existing file:</p> <p>pkzipc –add –header=@text.doc save.zip *.doc</p> | <p>add, standalone</p> |
| <p>help</p> <p>Displays help screen for PKZIP</p> | <p><command or option> – Any command or option for which help is desired.</p> <p>No default value.</p> | <p>pkzipc –help</p> <p>Display help for the add command:</p> <p>pkzipc –help=add</p> | <p>standalone</p> |
| <p>include</p> <p>Include files to compress or extract.</p> <p>Configurable separately for add and extract operations.</p> <p>Note: You must specify a sub-option (for example, file pattern or list file name preceded by an appropriate list character "@") with the include option.</p> | <p>The file(s) or file pattern (for example, *.doc) being included.</p> <p>No default value.</p> | <p>pkzipc –add –include="*.doc" save.zip</p> <p>In this example, only .doc files will be compressed.</p> <p>pkzipc –config –include="*.txt"</p> <p>In this example, you are setting up .txt files as the files that you always want to compress or extract, until you change the default or override from the command line with the exclude option.</p> <p>Note: When you use the <i>include</i> option with the configuration command, PKZIP prompts you to configure the <i>include</i> default for <i>add</i> and/or <i>extract</i> operations.</p> | <p>add, extract, delete, test, view, print, console</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|---|---|--|--|
| <p>keyfile</p> <p>Specifies a file containing the private key for the certificate specified by the certificate option. The option is most useful when using SSL server certificates, which often have the private key and certificate in separate files.</p> <p>Configurable</p> | <p><filename> – The name and location of the file</p> | <pre>pkzipc -add -certificate=#mycert.pem -keyfile=mykey.key save.zip *.doc</pre> | <p>add, extract, test, view</p> |
| <p>keypassphrase</p> <p>Specifies the passphrase used to decrypt private key information. This can be the passphrase used for your certificate store (UNIX only), for a PKCS#12 file (specified with the certificate option), or a key file specified with the keyfile option.</p> | <p><passphrase> – The passphrase, in quotes</p> | <pre>pkzipc -add -certificate=#mycert.p12 -keypassphrase="my password" save.zip *.doc</pre> <pre>pkzipc -add -certificate=#mycert.pem -keyfile=mykey.key -keypassphrase="my password" save.zip *.doc</pre> | <p>add, extract, test, view</p> |
| <p>larger</p> <p>Process only those files whose size is greater than (in bytes) or equal to a specified file size.</p> <p>Configurable separately for add and extract operations.</p> | <p>Numerical value (in bytes) that indicates a minimum desired file size.</p> <p>No default value.</p> | <pre>pkzipc -add -larger=5000 save.zip *</pre> <p>In this example, PKZIP adds only files that are at least 5000 bytes in size.</p> | <p>add, extract, test, view, delete, print console</p> |
| <p>level</p> <p>Set the level of compression.</p> <p>Configurable</p> | <p>Any digit from 0 through 9, with 0 being no compression at the fastest speed, and 9 being the most compression at the slowest speed.</p> <p>Default = level 5 (normal)</p> | <pre>pkzipc -add -level=9 save.zip *.doc</pre> | <p>add</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|---|---|--|---|
| <p>license</p> <p>Displays the product license information for PKZIP</p> | <p>No sub-options.</p> <p>No default value.</p> | <p>pkzipc -license</p> | <p>standalone</p> |
| <p>listchar</p> <p>Set the list character to the specified ASCII character. Prefixing a file name with the list character identifies it as a list file.</p> <p>Configurable</p> | <p>Any character in the printable ASCII range. Must not be the same as OptionChar and must not be “-”.</p> <p>default = @</p> | <p>pkzipc -config -listchar=+</p> | <p>All commands except list-cryptalgorithms, license, and version</p> |
| <p>listcryptalgorithms</p> <p>Displays a list of the strong encryption algorithms available for use with the CryptAlgorithm option under your PKZIP license</p> <p>Note: This option is only available in versions that have strong encryption.</p> | <p>None</p> | <p>pkzipc -listcryptalgorithms</p> | <p>standalone</p> |
| <p>listfile</p> <p>Generates a file that lists the files to be added to or extracted from an archive. The option causes a list file to be created instead of actually adding or extracting files.</p> | <p>Requires a name for the list file</p> <p>No default value.</p> | <p>pkzipc -add=update -listfile=mylist.txt myarchive.zip *</p> <p>In this example, PKZIP generates an ASCII text file (list.txt) that lists all files in the current directory:</p> <p>pkzipc -add -listfile=list.txt *</p> <p>In this example, PKZIP generates a list file that lists all files in the save.zip archive:</p> <p>pkzipc -extract -listfile=list.txt save.zip</p> | <p>add, extract</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|---|--|---|---|
| <p>locale</p> <p>Sets the default PKZIP time and date settings to match your system time and date formats.</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>The option is enabled and configured On by default. When disabled, PKZIP uses a 12-hour time format and a date format of MMDDYY).</p> | <p>pkzipc -config -locale</p> <p>pkzipc -add -locale test.zip *.doc</p> | <p>All commands except list-cryptalgorithms, license, and version</p> |
| <p>lowercase</p> <p>Extracts file names in lower case regardless of how they appear in the archive</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -extract -lowercase save.zip *</p> | <p>extract</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|---|---|--|---------------------|
| <p>mask</p> <p>Strips file attributes that the attribute option would otherwise cause to be stored or set for extracted files</p> <p>Configurable</p> <p>Note: You can only mask attributes that are specified using the attributes option.</p> | <p>hidden – hidden attributes.</p> <p>archive – archive attribute.</p> <p>system – system attributes.</p> <p>readonly – read-only attributes.</p> <p>none – no attributes (turns off attribute mask in the PKZIP Configurations Settings file for this instance only).</p> <p>all – all attributes</p> <p><hex value> –The hex value of an attribute to be masked, or the logical OR of multiple hex values</p> <p>-----</p> <p>Default (add) = none</p> <p>Default (extract) = all</p> <p>Default if used on command line without a sub-option (add and extract) = all</p> | <p>pkzipc –add –attr=all –mask=hidden save.zip</p> <p>pkzipc –extract –mask=none save.zip</p> <p>pkzipc –config –mask=hidden</p> | <p>add, extract</p> |
| <p>maximum</p> <p>Uses the Deflate compression method and sets the level of compression to level 9, the highest level on a 0 - 9 scale, but gives the lowest speed</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc –add –max save.zip *.doc</p> <p>pkzipc –config –max</p> | <p>add</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|--|--|---|
| <p>more</p> <p>Pauses after one screen of output and prompts to continue.</p> <p>Configurable</p> | <p>The number of rows of information you want to define as a screen</p> <p>-----</p> <p>Default = one screen of information</p> | <p>pkzipc -view -more=22 save.zip</p> <p>pkzipc -config -more</p> | <p>all commands</p> |
| <p>move</p> <p>Removes (deletes) files from the source drive after adding them to an archive.</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -move save.zip *.doc</p> | <p>add</p> |
| <p>movearchive</p> <p>Deletes an archive that is created only as an intermediate archive—for example, to be converted by the <i>encode</i> option to an archive of a different type.</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -encode=gzip -movearchive myfiles.tar</p> | <p>add</p> |
| <p>newer</p> <p>Selects files that are no older than a specified interval, such as “five days”</p> <p>Configurable separately for add and extract operations.</p> <p>Note: Specifying a newer value is functionally equivalent to specifying an after value.</p> | <p><numeric value> A number of days, hours, minutes, or seconds defining the interval, plus a suffix identifying the kind of units used:</p> <p>Suffixes:</p> <p>d – Days (default) h – Hours m – Minutes s – Seconds</p> <p>-----</p> <p>No default value.</p> | <p>Adds files no older than 24 hours:</p> <p>pkzipc -add -newer=24h save.zip *</p> <p>Adds files no older than five days:</p> <p>pkzipc -add -newer=5d save.zip *</p> <p>pkzipc -add -newer=5 save.zip *</p> | <p>add, extract, test, view, print, console</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|--|---|---|
| <p>noarchiveextension</p> <p>Suppresses adding a file name extension to the specified archive file name</p> <p>Configurable</p> <p>Note: This option is identical to nozipextension, which is now deprecated.</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -noarchiveextension file.ibm *.doc</p> | <p>All commands except list-cryptalgorithms, license, and version</p> |
| <p>noextended</p> <p>Suppress the storage of extended attribute information (excluding file permission attributes)</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -noextended save.zip *</p> | <p>add</p> |
| <p>nofix</p> <p>Suppress the attempt to fix any problems PKZIP encounters in extracting from an archive</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -nofix save.zip *.doc</p> | <p>All commands except list-cryptalgorithms, license, and version</p> |
| <p>normal</p> <p>Uses the Deflate algorithm and sets the level of compression to 5 (normal) on a scale of 0 - 9 for a balance of compression and speed. Unlike with the fast option, all files are compressed.</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -normal save.zip</p> <p>pkzipc -config -normal</p> | <p>add</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|--|--|---|
| <p>nozipextension</p> <p>Note: This option is deprecated. Use the option noarchiveextension instead.</p> <p>Suppress PKZIP's adding of an identifying file extension to an archive file name</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -nozipextension file.ibm *.doc</p> | <p>All commands</p> |
| <p>older</p> <p>Selects files that are older than a specified interval, such as "five days"</p> <p>Configurable separately for add and extract operations.</p> <p>Note: Specifying an older value is functionally equivalent to specifying a before value.</p> | <p><numeric value> A number of days, hours, minutes, or seconds defining the interval, plus a suffix identifying the kind of units used:</p> <p>Suffixes:</p> <p>d – Days (default) h – Hours m – Minutes s – Seconds</p> <p>-----</p> <p>No default value.</p> | <p>Adds files older than 24 hours:</p> <p>pkzipc -add -older=24h save.zip *</p> <p>Adds files older than five days:</p> <p>pkzipc -add -older=5d save.zip *</p> <p>pkzipc -add -older=5 save.zip *</p> | <p>add, extract, test, view, print, console</p> |
| <p>optionchar</p> <p>Specifies the prefix character used to identify a command or option as such on the command line</p> <p>Note: On Windows, the "/" (slash) character can also always be used.</p> <p>Configurable</p> | <p>Any valid single character.</p> <p>-----</p> <p>Default = - (hyphen)</p> | <p>pkzipc -opt=+ +add save.zip *.doc</p> <p>pkzipc +config -option=+</p> | <p>all commands</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|---|---|--|---|
| <p>overwrite</p> <p>Specifies whether to overwrite existing files with files being added or extracted. By default, PKZIP prompts before overwriting when extracting but not when adding.</p> <p>Configurable</p> | <p>prompt – Prompt every file individually on whether to overwrite a file that has the same name as the one being added or extracted</p> <p>all – Overwrite all files that have the same name</p> <p>never – Never overwrite a file that already exists in the target directory or archive</p> <p>-----</p> <p>Default if used on command line without a sub-option = all.</p> | <p>pkzipc -ext -over=all save.zip</p> <p>pkzipc -add -overwrite=prompt save.zip</p> | <p>add, extract</p> |
| <p>password</p> <p>Protects an archive with password-based encryption</p> <p>PKZIP prompts for a password if none is specified with the option.</p> <p>Configurable</p> | <p><password> – The password that must be supplied to extract and decrypt the files</p> <p>-----</p> <p>No default value.</p> | <p>To include a password in the command:</p> <p>pkzipc -add -pass=beowulf save.zip</p> <p>To have PKZIP prompt for a password after you type the command:</p> <p>pkzipc -add -pass save.zip</p> <p>To extract password-protected files from an archive:</p> <p>pkzipc -extract -password=beowulf9 save.zip</p> | <p>add, extract, test, print, console</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|---|--|--|-------------------------------------|
| <p>path</p> <p>Stores or restores directory path names for files within a .ZIP file</p> <p>By default, PKZIP does not store path information</p> <p>Configurable</p> | <p>current – Store the path from the current directory.</p> <p>root or full – Store the entire path beginning at the root of the drive; also referred to as "full" path.</p> <p>specify – Store the amount of path information passed on the command line with the file name</p> <p>relative – Same as current</p> <p>none – No path information stored</p> <p>-----</p> <p>Default = none</p> <p>Default if used on command line without a sub-option = current</p> | <p>Assuming in you are in "/temp":</p> <p>pkzipc –add –path=root save.zip docs/*</p> <p>(the complete path is stored including "temp/docs/").</p> <p>pkzipc –add –path=current save.zip docs/wp/*</p> <p>(the path stored would be "docs/wp").</p> | <p>add, extract</p> |
| <p>preview</p> <p>Prints out messages to preview the results of a set of commands or options without actually performing the tasks</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc –add –preview save.zip</p> | <p>add, delete, header, comment</p> |
| <p>print</p> <p>Print a file within a .ZIP file.</p> | <p><print device> – The print device use, for example, "lpt1".</p> <p>-----</p> <p>Default = the default printer on your system.</p> | <p>pkzipc –print=lpt1 save.zip readme.txt</p> <p>If you do not specify a print device, your default printer is used.</p> | <p>standalone</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|---|--|-------------------|
| <p>recurse</p> <p>Search subdirectories for files to compress</p> <p>Use with path to store path information for files in subdirectories. Or use directories, which combines the functionality of recurse and path.</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -recurse save.zip *</p> | <p>add</p> |
| <p>shortname</p> <p>Convert long file names of files added to an archive to WIN32-equivalent "short" file names</p> <p>Configurable</p> | <p>dos – Convert long file names to DOS-equivalent short file names (8+3)</p> <p>none – Do not convert file names</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -short=dos save.zip</p> | <p>add</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|--|--|---|
| <p>silent</p> <p>Suppresses the display of some or all of PKZIP's messages to the user, including warnings and errors. It can also suppress prompts for inputs.</p> <p>Configurable</p> | <p>none – Turns off the silent option; displays all messages</p> <p>banner – Suppresses printing the banner</p> <p>copy – Suppresses "Copy file" messages when updating archives</p> <p>error – Suppresses all error and warning outputs</p> <p>input – Suppresses all requests for input. If any operation requests input, an error is given</p> <p>normal – Suppresses all normal outputs</p> <p>output – Suppresses all normal, error, and warning outputs</p> <p>progress - Suppresses "percent complete" messages</p> <p>all – Same as specifying both Input and Output. (Default)</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -silent save.zip *.doc</p> <p>pkzipc -config -silent</p> | <p>All commands except list-cryptalgorithms, license, and version</p> |
| <p>smaller</p> <p>Process only files that are smaller than or equal to a given file size, specified in bytes</p> <p>Configurable separately for add and extract operations.</p> | <p>Numerical value that indicates a maximum desired file size (in bytes)</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -smaller=5000 save.zip *</p> <p>In this example, PKZIP adds only files no larger than 5000 bytes in size.</p> | <p>add, extract, test, view, delete, print, console</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|---|--|---|---|
| <p>sort</p> <p>Sort files in an archive based on specific criteria (for example, by file size). Files are then viewed, added, and extracted in the order sorted.</p> <p>Configurable</p> <p>Note: The <code>crc</code> and <code>ratio</code> sub-options do not work with the <code>add</code> command and <code>sort</code> option.</p> | <p>crc – sort by CRC value.</p> <p>date – sort by file date of file.</p> <p>extension – sort by file extension.</p> <p>name – alphabetically sort files and folders together in one series by path name.</p> <p>natural – sort in the order files were compressed.</p> <p>ratio – sort by compression ratio.</p> <p>size – sort by the original, uncompressed size of the file ("length" in display).</p> <p>comment – sort by file comment.</p> <p>none – first alphabetically sort path names that contain folders and then separately sort file names that lack folder information. (The default.)</p> <p>-----</p> <p>Default = none</p> <p>Default if used on command line without a sub-option = name</p> | <p><code>pkzipc -add -sort=date save.zip *.doc</code></p> <p><code>pkzipc -config -sort=date</code></p> | <p>add, extract, test, view, delete, print, console</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|---|---|-------------------|
| <p>span</p> <p>Forces PKZIP to create a split archive, even when creating the archive on non-removable media.</p> <p>Also used to format or wipe removable media prior to writing the archive.</p> <p>This option is only available for use with ZIP archives.</p> <p>Note: On Windows, spanning should take place automatically; normally the feature does not need to be enabled from the command line</p> <p>Configurable</p> | <p>Format – fully format media before attempting to write to it.</p> <p>Quick – quick format media before attempting to write to it.</p> <p>Wipe – erase contents of media before attempting to write to it.</p> <p>None – no media format or erase of media contents before attempting to write to it.</p> <p><segment size> – split files into predefined size (see choices below) or a specified size (in bytes) greater than 65536.</p> <p>360K, 720K, 1.2MB, 1.44MB, 2.88MB, 95.7MB, 650MB, 700MB</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc –add –span a:\save.zip *.doc</p> <p>pkzipc –add –span=format a:/save.zip *.doc</p> <p>pkzipc –add –span=1.44 c:/save.zip *.doc</p> <p>pkzipc –add –span=1457664 c:/save.zip *.doc</p> | <p>add</p> |
| <p>speed</p> <p>Uses the Deflate algorithm and sets the level of compression to 1 on a scale of 0 - 9. Some files are stored (level 0) uncompressed.</p> <p>Provides the fastest performance but the least compression. Files having the following extensions are stored uncompressed: bz2, bzip2, cab, gz, gzip, rar, gif, jpeg, jpg, mp3, mpeg, mpg, sxw</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc –add –speed save.zip *.doc</p> <p>pkzipc –config –speed</p> | <p>add</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|--|---|-------------------------------------|
| <p>store</p> <p>Sets the level of compression to 0 (no compression) on a scale of 0 - 9; stores the files in the archive without compressing them</p> <p>Configurable</p> | <p>No sub-options.</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -store save.zip *.doc</p> <p>pkzipc -config -store</p> | <p>add</p> |
| <p>temp</p> <p>Specifies the directory to use for temporary files created by PKZIP</p> <p>Configurable</p> | <p>The drive and/or path. For example: C: or /root/temp</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -add -temp=z:\public test.zip *.txt</p> <p>This example updates the .ZIP file test.zip and uses the z:\public directory location for temporary files.</p> <p>pkzipc -add -temp=/temp test.zip *.txt</p> <p>This example updates the .ZIP file test.zip and uses the /temp directory location for temporary files.</p> | <p>add, delete, header, comment</p> |
| <p>test</p> <p>Tests the integrity of files within a .ZIP file</p> <p>Configurable</p> | <p>all – all files in the archive file are tested</p> <p>freshen – tests only those files in the archive that are newer versions of files that already exist in the extract directory</p> <p>update – tests files in the archive that are newer versions of files that already exist in the extract directory or that do not already exist there</p> <p>-----</p> <p>Default = all</p> | <p>pkzipc -test save.zip</p> | <p>standalone</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|--|--|---|---|
| <p>times</p> <p>Specifies that PKZIP should restore the extended time fields, and/or other dates stored in the archive.</p> <p>Configurable</p> | <p>access – restores the time of last access to file(s) on extraction.</p> <p>modify – restores the time of last modification to files on extraction.</p> <p>create – restores the time of creation to files on extraction.</p> <p>all – all file times are restored.</p> <p>none – file times are not restored.</p> <p>-----</p> <p>Default = none</p> | <p>pkzipc -extract -times=access save.zip</p> | <p>extract</p> |
| <p>translate</p> <p>Translates EOL (“end of line”) characters when adding or extracting files. For .ZIP archives, the translation occurs only for files which are marked as ASCII. For other archive types, the translation may occur on all files, including binary files.</p> <p>Configurable</p> | <p>none – no translation is performed.</p> <p>dos – translates text files so that lines end with a return/newline pair (Win32 default)</p> <p>mac – translates text files so lines end with a single carriage return</p> <p>unix – translates text files so lines end with a single newline</p> <p>-----</p> <p>Default = none</p> <p>Default if used on command line without a sub-option = native operating system compatibility translation.</p> | <p>pkzipc -extract -translate=unix save.zip</p> <p>pkzipc -add -translate=unix scripts.zip *.pl</p> | <p>add, extract, console, print</p> |

| Name/Description: | Value(s): | Example usage: | Used with: |
|--|---|---|---|
| <p>version</p> <p>Identifies the version of PKZIP</p> | <p>major – returns the major version number of the release.</p> <p>minor – returns the minor number of the release. For example, if the version number is 2.5.1, the value returned is 5.</p> <p>step – returns the step, or patch value. For example, if the version is 2.04.01, the step value returned is 1.</p> <p>-----</p> <p>Default = major</p> | <p>pkzipc -version</p> | <p>standalone</p> |
| <p>view</p> <p>Displays information about the files in an archive—for example, the compressed size of a file</p> <p>Configurable</p> | <p>brief – present information in the most compact manner.</p> <p>detail – present information in the most detailed manner</p> <p>normal – present information in the normal manner.</p> <p>-----</p> <p>Default = normal</p> | <p>pkzipc -view save.zip</p> | <p>standalone</p> |
| <p>warning</p> <p>Pauses after every specified warning and prompts whether to continue. If no warning is specified, pauses after every warning.</p> <p>Configurable</p> | <p><warning number> – One or more warning numbers, separated by commas. To override a warning number configured for the option (and thus <i>not</i> pause and prompt on that warning), precede the number with a hyphen</p> <p>-----</p> <p>No default value.</p> | <p>pkzipc -ext -warn=43 save.zip *</p> <p>pkzipc -ext -warning save.zip *</p> <p>pkzipc -ext -warn=-43 save.zip *</p> | <p>add, extract, test, view</p> |

| <i>Name/Description:</i> | <i>Value(s):</i> | <i>Example usage:</i> | <i>Used with:</i> |
|---|--|--|--|
| <p>wipe</p> <p>Overwrites files deleted by PKZIP to prevent recovery of their data</p> <p>Configurable</p> | <p>none – turns off wiping: files are not overwritten</p> <p>random – Overwrites files once with random data</p> <p>nsa – Overwrites files seven times, to the NSA standard</p> <p>-----</p> <p>Default = none</p> <p>Default if used on command line without a sub-option = random</p> | <p>pkzipc -add -move -wipe=nsa myfiles.zip *</p> | <p>add</p> |
| <p>zipdate</p> <p>Note: This option is deprecated. Use the functionally identical option archivedate instead.</p> <p>Set the file modification date of the archive file.</p> <p>Configurable</p> | <p>newest – set to the date of the newest file within the archive file.</p> <p>oldest – set to the date of the oldest file in the archive file.</p> <p>retain – retain the original date of the archive file (the date when the file was created).</p> <p>none – disable the file date in the configuration file and set the archive date as the last modification date.</p> <p>-----</p> <p>Default = the current date.</p> <p>Default if used on command line without a sub-option = retain.</p> | <p>pkzipc -add=update -zipdate=retain save.zip *.txt</p> | <p>add, delete, fix, header, comment</p> |

B

Error and Warning Messages

This appendix contains reference information on all error and warning messages that can occur in PKZIP. An error usually causes the canceling of the task you are performing such as compressing a file. A warning usually indicates that something is wrong, but it is not severe enough to cancel an entire task. It might also be a reminder or query prompt. PKZIP will also return any error codes to the shell. If there were no warnings or errors, 0 is returned.

Error Messages

When an error occurs, PKZIP displays an error message. The following is a description of each error message.

| <i>Error</i> | <i>Potential Cause(s)</i> |
|--|---|
| (E2) Ambiguous option or command specified - XXX. | If you abbreviate an option on your command line, make sure that you are supplying enough characters in the option to delineate it from similarly spelled options. If, for example, you only specify -pr on your command line, PKZIP will generate the (E2) error because it cannot determine whether you are specifying the print or preview option. |
| (E3) Ambiguous sub-option specified - XXX. | If you abbreviate a sub-option on your command line, make sure that you are supplying enough characters in the sub-option to delineate it from similarly spelled sub-options. If, for example, you only specify -sort=na on your command line, PKZIP will generate the (E3) error because it cannot determine whether you are specifying the name or natural sub-option. |
| (E4) Unknown or illegal option - XXX. | The option you specified on the command line is invalid. It does not match any known options. Verify that you typed the option correctly. Check the spelling. |

| <i>Error</i> | <i>Potential Cause(s)</i> |
|--|---|
| (E5) Unknown or illegal sub-option - XXX. | The sub-option you specified on the command line is invalid. It does not match any known sub-options. Verify that you typed the sub-option on your command line correctly. Verify that you are not using an illegal sub-option (-add -sort=crc). Check the spelling. |
| (E6) No .ZIP file specified. | There was not a .ZIP file specified on the command line. PKZIP does not accept wildcards for .ZIP file name when adding files to a .ZIP archive. |
| (E7) Can't create: XXX. | PKZIP could not create a .ZIP file when fixing. PKZIP could not create a volume label on a spanned archive. PKZIP could not create a temporary file for a spanned archive. Verify that you have write access to the drive or diskette on which you are creating these files. |
| (E8) Nothing to do! | You did not do something that is required for a particular task. For example, PKZIP could not find the file you are trying to open or access. You might have specified to update a pattern such as *.txt and PKZIP did not find any files that matched or that needed updating. |
| (E9) No file(s) were processed | PKZIP cannot find the file you are trying to access. For example, you might be trying to extract files from a .ZIP archive that do not exist in that archive. Verify that the file(s) you specify on the command line exactly match the file(s) in the .ZIP file. If, for example, the file in the archive is stored with path information, and you attempt to extract it but specify only the file name, you will get the (E9) error. |
| (E10) No files specified for deletion. | There are no files or file patterns specified for deletion on the command line. In lieu of a specified file or file pattern, PKZIP will not assume that the user wishes to specify all (*) files. |
| (E11) Disk full, file: XXX. | The hard disk or floppy disk you are writing to is full. This error occurs when PKZIP attempts to write a .ZIP file, or extract a file contained in a .ZIP file to a hard or floppy disk that is full. Free up sufficient disk space and try again. |
| (E12) Can't find file: XXX. | PKZIP cannot find the .ZIP file you specified. This error will only occur when you use commands/options/sub-options that work with existing .ZIP files. Verify that the file is specified correctly. If you are adding files to an archive, verify that you place the .ZIP file name before specifying files to be added on the command line. If the .ZIP file is not in the same directory where you typed the command, make sure to include path information. (e.g., pkzipc -add=freshen /temp/test.zip *.txt) |

| <i>Error</i> | <i>Potential Cause(s)</i> |
|---|--|
| (E13) Can't open .ZIP file: XXX. | The named .ZIP file is read-only or locked by another application and can not be modified. This may also occur on a Network drive if you do not have sufficient access rights to the file to allow you to modify it. |
| (E14) Can't create archive: XXX. | PKZIP is not able to create the archive file. Verify that the destination directory is not full, the archive file does not already exist. If you are creating the file on a network drive, confirm that you have the appropriate rights to the network file system. |
| (E15) Renaming temporary .ZIP file, saved as: XXX. | PKZIP could not rename the temporary file to the specified .ZIP file name. Verify that the destination drive is not full. If you are updating a non-spanned .ZIP file on removable media (floppy diskette) and the updated archive exceeds the size available on the removable media, you will receive the (E15) error. You will need to recreate the archive for spanning. Keep in mind that you cannot update a spanned archive. If you are creating the file on a network drive, confirm that you have the appropriate rights to the network file system. |
| (E16) Can't open for write access, file: XXX. | PKZIP is unable to write to the specified file or device. Verify that you have write access to the file or that your printer is configured correctly. Additionally if you are using the PKSFXS.DAT file, verify that you have PKSFXS.DATA environment variable configured correctly. |
| (E17) Error encrypting file data. | PKZIP encountered a problem with the compressed data that it was trying to encrypt. For example, the disk on which the compressed data was located was bad or corrupt. |
| (E18) Can't open list file: XXX. | The named list file could not be found. It does not exist, was spelled incorrectly, is not located in the specified directory, or cannot be accessed because the user does not have the appropriate rights to the file. |
| (E19) Aborted file extract. | Extraction process was terminated by the user while changing disks during a disk spanning operation. |
| (E20) Aborted file compression. | Compression process was terminated by the user while changing disks during a disk spanning operation. |
| (E21) Can't modify a spanned or split .ZIP file | Spanned or split .ZIP files cannot be modified. The archive will need to be recreated. |
| (E22) Cannot format removable media. | The media cannot be formatted. The media may be write-protected. |
| (E23) Suboption is too long | The option is too long. See if you can abbreviate the name of the option or its sub-option to make it shorter. |

| Error | Potential Cause(s) |
|--|---|
| (E24) Insufficient disk space for ZIP comment. | There is not enough space on the system or media to write the ZIP comment. |
| (E25) Insufficient disk space for updated file. | Insufficient disk space for the new archive. If you are adding files to an archive on a removable media, the media may not be large enough to write the modified file (too large). |
| (E26) Device not ready: XXX. | The removable media device is not ready. The disk may not be in the drive properly. |
| (E27) 2.04g compatibility cannot be used with the option - XXX | Option 204 , which creates an archive compatible with PKZIP for DOS v. 2.04g, was used with another option that is not supported for that version of PKZIP |
| (E34) Invalid archive format: <archive name> | The file is not in a format currently supported by PKZIP |
| (E58) Invalid archive - method not supported. | The archive uses a compression method that currently is not supported. |
| (E65) Could not encode archive file: XXX. | The file could not be encoded. |
| (E71) Can't open PKCS#7 file: XXX. | PKZIP cannot open the PKCS#7 because the files does not exist, user cannot read the file, or file is not a valid PKCS#7. |
| (E72) PKZIP wanted user input, but silent=input or silent=all was specified | If PKZIP needs user input—for example, to say whether files should be overwritten—but -silent=input or -silent=all is specified on the command line to hide PKZIP messages and prompts, PKZIP halts processing and issues this error. |
| (E73) Warning configured as an error | The warning immediately preceding this error message has been specified (with the error option) to be treated as a fatal error. |
| (E75) Incorrect password or certificate not found, unable to open ZIP archive: <archive name> | The archive contains encrypted file names that PKZIP cannot decrypt. If the archive is password-protected, you must include the <i>password</i> option with the <i>extract</i> command in the command line. |
| (E76) Cannot open alternate config file: <file name> | The <i>altconfig</i> option was used, but the specified file could not be opened. |
| (E77) Archive can only support one file inside! | You tried to add more than one file to an archive of a type that cannot contain multiple files. For example, a GZIP archive can contain no more than one file. If you try to create a GZIP archive to contain three files, PKZIP displays this error and does not create the archive. |
| (E78) Unable to FTP archive file: <file name> | PKZIP could not transfer the specified file. |
| (E79) Unable to E-mail archive file: XXX | A problem, perhaps with the network or the mail server, prevented PKZIP from emailing the specified file. |

| <i>Error</i> | <i>Potential Cause(s)</i> |
|---|---|
| (E80) Unable to run anti-virus | PKZIP was unable to run the anti-virus scanning program. The anti-virus program did not respond to the command line used to launch the program. |
| (E81) Possible virus detected | The anti-virus program returned a non-zero value after doing a scan. Most anti-virus programs use this return to indicate the possible presence of a virus. |
| (E83) Specified SFX cannot extract archive created with the option – XXXX | You tried to create an SFX that is not able to handle a feature turned on by the option XXXX. For example, you tried to create a strongly encrypted DOS SFX, or an SFX that uses FNE. |
| (84) Fatal policy error - nnnnn, contact your system administrator | <p>A critical problem has occurred with a policy file or policy certificate. The number is a policy error code to help your administrator resolve the problem.</p> <p>On this error, PKZIP goes into read-only mode. In read-only mode, PKZIP will still extract files from archives but will not add files to a new or existing archive and disables the related controls.</p> |
| (E100) Insufficient memory | Insufficient memory is available to process the archive. Try making more memory available to PKZIP. If this does not rectify the problem, then the archive may be corrupted. The -fix option may correct the problem. If you receive this message when you try to create a new archive, possibly you are attempting to compress too many files. Reduce the number of files and try again. If you are using a LIST file in your PKZIP command, the LIST file may be too large. See page 40 for more information on LIST files. |
| (E150) Error reading .ZIP file. | PKZIP cannot read the .ZIP file or is unable to read the central directory record. The file might be located on a corrupt disk or part of a disk. This includes floppy disks. |
| (E155) Too many files in XXX. | PKZIP cannot add or extract files in excess of the limit of 16,383 with the 204 option enabled. Reduce the number of files you are trying to process. |
| (E156) File is now too big for valid zip data. | The .ZIP archive is too large and PKZIP is unable to locate the central end record in the .ZIP file. The file is not a valid .ZIP archive or has been corrupted. The fix option may repair the .ZIP file. |
| (E157) This archive requires a product compliant with ZIP APPNOTE version XX.X | The archive requires a more recent version of PKZIP, or other archiving program, that supports the version of the ZIP file format described in the specified APPNOTE ("application note"). The APPNOTE is a document that is available on the PKWARE Web site. |
| (E158) Errors encountered reading archive | PKZIP was unable to read the archive. |

| <i>Error</i> | <i>Potential Cause(s)</i> |
|---|--|
| (E253) This program is not licensed for use on Windows Server platforms. Please contact PKWARE to obtain an appropriate server product for this machine. | <i>PKZIP for Windows Command Line</i> is intended for single-user desktop use and cannot be run on server platforms such as Windows 2003 Server. |
| (E254) Your evaluation period for PKZIP has expired. Please register to continue using this product. | This copy of PKZIP is an evaluation version. If you have purchased PKZIP and have the serial number, enter it when prompted. |
| (E255) User pressed ctrl-c or control-break. | This error occurs when you press CTRL+BREAK or CTRL+C in the middle of a PKZIP operation. |

Warning Messages

Sometimes a condition occurs that might cause a task to pause temporarily. This could be something that prevents part of a task from happening, or simply a message or reminder. For several of these conditions, PKZIP displays a warning message. When a warning occurs, PKZIP returns a value of 1 to the shell.

The following is a description of each warning message:

| <i>PKZIP Warning</i> | <i>Potential Cause(s)</i> |
|--|--|
| (W1) Can't create: XXX. | PKZIP could not create volume label, file, or directory. Verify that you have appropriate access rights to the file or directory. |
| (W2) Illegal path or drive specified: XXX. | The file being extracted has an invalid name or path. Verify that you have entered the correct path in your command line and that the file does not contain any inappropriate characters such as a colon or leading slash. |
| (W3) Warning! This file requires a product compliant with ZIP APPNOTE version XX.X | The file requires a more recent version of PKZIP, or other archiving program, that supports the version of the ZIP file format described in the specified APPNOTE ("application note"). The APPNOTE is a document that is available on the PKWARE Web site.. |
| (W4) File fails CRC check. | It is likely that the file PKZIP is trying to extract is corrupt, and was not extracted correctly. For more information, see the CRC section in Appendix D. |
| (W7) file: XXX already exists. Overwrite (<Y>es/<N>o/<A>ll/ne<V>er/<R>ename/<Esc>)? | The file(s) you are trying to extract already exists in the location to which you are extracting. By default, PKZIP prompts you before overwriting a file. |
| (W8) Could not open file: XXX. | You may not have the proper permissions to access the file or the file may have been locked by another program while PKZIP was trying to access it. If the file is located on a network file system, consult your System Administrator to verify your access rights. |

| <i>PKZIP Warning</i> | <i>Potential Cause(s)</i> |
|---|--|
| (W9) Could not delete file: XXX. | You do not have the proper permissions to access and delete the file, or another application has the file open. This warning only occurs when the move option is used on the command line. |
| (W12) Unexpected end of compressed data. | Corrupt data caused PKZIP to abort the extraction before it could finish. |
| (W13) Skipping encrypted file: XXX. | PKZIP encountered a file that has been password protected. You need the password to access this file. |
| (W18) Unknown compression method for file: XXX. | An unfamiliar compression method has been used with the current .ZIP file. |
| (W19) Could not clear archive attribute on file: XXX. | PKZIP could not clear the archive attribute on a file. The file will be compressed but the archive bit cannot be cleared. This warning usually occurs when the add=incremental option is used on the command line. |
| (W20) Incorrect password for file: XXX. | Verify that you entered the correct password for the file. When a file is password protected, you can only access its contents with the correct password. Note: Passwords are case sensitive. |
| (W21) Invalid temporary file directory: <dir> | PKZIP creates a temporary file for the file(s) being compressed when updating a .ZIP file. PKZIP was unable to create the temporary .ZIP file in the specified location and so used the default temp directory for your system. |
| (W22) Authenticity Verification Failed! | The Authenticity Verification (AV) information contained in the .ZIP file is corrupt. Failure of AV indicates a file that has been tampered with or damaged. If the file has failed the AV check, the contents are suspect. |
| (W23) Authenticity Verification Failed! | The stored Authenticity Verification (AV) checksum value did not match the calculated checksum value. The .ZIP file has been tampered with or is perhaps corrupt. |
| (W26) directory: XXX already exists. Overwrite (<Y>es/<N>o/<A>ll/ne<V>er/<Esc>)? | Assuming the overwrite option is set to prompt, this warning appears when PKZIP attempts to extract a directory over an existing directory with the same name. Answering Y at this prompt will update any extended attributes (EAs) stored in the .ZIP file. |
| (W29) Can't rename temporary file. Saved as XXX. | PKZIP cannot rename the temporary archive created when updating an archive. The archive was saved under the specified name. |
| (W36) Empty password, files will not be password protected. | When trying to password protect your file, you entered a password containing no letters or numbers. |

| <i>PKZIP Warning</i> | <i>Potential Cause(s)</i> |
|--|--|
| (W37) Can't sign file. | This warning appears when PKZIP fails to sign a file using the specified digital certificate. Common reasons are incorrect password for the certificate (not all certificates have passwords), no private key (certificate needs to have a private key). |
| (W38) Can't sign central directory. | PKZIP failed to sign the central directory. Common reasons are that an incorrect password was supplied to access the certificate (not all certificates have passwords) or the certificate lacks a private key (needed to apply a digital signature). |
| (W39) Signature is invalid. | Someone or something has changed the archive since it was digitally signed. For example, the archive may be corrupt. |
| (W40) Certificate not trusted. | The certificate is currently not to be trusted. |
| (W41) Certificate expired. | The certificate has expired. This does not necessarily mean that the certificate or signatures applied with it are not to be trusted. They may simply be old. |
| (W42) Certificate was revoked. | The issuer has revoked the certificate. |
| (W43) Certificate not found: XXX. | PKZIP was unable to find a certificate of that name on the system. |
| (W45) Bad data in compressed stream. | Something was wrong in the stream of compressed data. The ZIP file is corrupt. |
| (W46) Encryption algorithm is not available. Using: XXX. | PKZIP cannot use the specified algorithm on this system. Use the ListCryptAlgorithms command to view a list of the encryption algorithms that PKZIP can use |
| (W47) No recipients specified. Recipients will not be used. | You specified the <code>--recipient</code> option, but did not include any actual recipients (or specified bogus recipients). When this occurs, PKZIP will not strongly encrypt files for recipients. If you did not tell it to use passwords; that is, you did not use the <code>--password</code> option, it will not encrypt files at all. In addition, if you specify <code>--password</code> and did not also specify <code>-cryptalgorithm</code> , you will not get strong encryption. You will, however, get traditional encryption. |
| (W48) Invalid item name | The name of an item (file) in the archive is invalid. Possible reasons are: The file has the same name as another file in the same folder; the path name of the archive item contains a file or folder name that exceeds the maximum number of characters allowed (254 for Windows, 255 for UNIX); the name contains characters that may not be used in file names on your operating system (the characters <code>: * ? \ " < > "</code> may not be used in file names on Windows). |
| (W52) Certificate verification failed! | Something is wrong with the certificate. |

| PKZIP Warning | Potential Cause(s) |
|---|--|
| (W53) Unknown exception caught: Exception code: XXX | An internal error occurred. Please contact PKWARE Technical Support with the exact command you used and the error code. |
| (W54) Option 'XXX' is not licensed for use in your copy of PKZIP | Your license key does not allow you to use that option. You must purchase an appropriate license key from PKWARE to use it. |
| (W55) Command 'XXX' is not licensed for use in your copy of PKZIP | Your license key does not allow you to use that command. You must purchase an appropriate license key from PKWARE to use it. |
| (W56) Recipient not found for file: XXX | The file was encrypted only for recipients, and PKZIP was unable to find a certificate for any of them. Verify that you have access to the private key for one of the recipients. |
| (W57) Incorrect password or recipient not found for file: XXX | Verify that you entered the correct password for the file. When an archived file is password protected, you can only access the file if you have the correct password. Passwords are case sensitive. If the file is encrypted with a certificate, verify that you have access to the private key for one of the recipients. |
| (W58) Problem reading .ZIP file: <zipfile name> | The .ZIP archive is corrupted. PKZIP can read it, but probably other zipping programs cannot. Use the -fix command to fix the archive so that other programs can read it. |
| (W59) Multiple certificates found | Multiple digital certificates were found that match the same recipient. These certificates may belong to different people. The archive is encrypted using each of the certificates; the owner of any of them can decrypt. |
| (W60) Unable to connect to LDAP server: <server name/address> | PKZIP was unable to access certificates on an LDAP server specified using the ldap option: the server address was bad. |
| (W61) Unable to login to LDAP server: <server name/address> | PKZIP was unable to access certificates on an LDAP server specified using the ldap option: the LDAP login failed. |
| (W62) Central Directory can only be encrypted with strong encryption. Central Directory will not be encrypted. | The cd option was used, which requires strong encryption, but one or both of the following were neither explicitly specified nor configured for use by default: encryption method (<i>password</i> , <i>recipient</i> options), encryption algorithm (<i>cryptalgorithm</i> option). |
| (W63) You must specify -password or -recipient to encrypt files! | You specified -cryptalgorithm or -cd=encrypt but did not specify either the <i>recipient</i> or <i>password</i> option. Files are not encrypted unless one of these options is used. |
| (W68) Must specify MailTo, MailFrom and MailServer to email the archive. | You tried to email an archive without specifying all three options MailTo, MailFrom and MailServer. Values for all three must be specified on the command line or configured for use by default. |

| PKZIP Warning | Potential Cause(s) |
|---|--|
| (W69) Skipping FTP file transfer because of encryption warning XXX. | PKZIP encountered a problem encrypting an archive that you directed to send by FTP, so PKZIP did not send the archive. This warning occurs if, for example, PKZIP can encrypt for only some but not all recipients, or if no password is supplied to use for password encryption. |
| (W70) Skipping mail file transfer because of encryption warning XXX. | PKZIP encountered a problem encrypting an archive that you directed to send by email, so PKZIP did not send the archive. This warning occurs if, for example, PKZIP can encrypt for only some but not all recipients, or if no password is supplied to use for password encryption. |
| (W71) Could not attach unzip instructions to the email message | PKZIP failed to attach instructions on how to unzip, as specified by the MailOptions option |
| (W72) Could not find the unzip instructions | PKZIP could not find the instructions on how to unzip |
| (W73) Some of the encryption recipients do not have email addresses | PKZIP was told to encrypt for recipients but could not find email addresses for some of the recipients |
| (W74) PKZIP is unable to access the default user's private key. | PKZIP is unable to access the private key of the default user. The logon password needed to access the certificate that contains the key may have been reset or changed by an administrator. To fix this warning, the user must change his password from his own computer, rather than let an administrator change it from another system. |
| (W75) Unable to resolve link: XXXX | While updating an archive, PKZIP could not find the original file or the new file |
| (W76) <certificate> does not pass the strict certificate checks, and will not be used. | The certificate did not pass the strict checking applied by the strict option, used in a command line that updates an archive. The certificate will not be used for the intended signing or encryption. |
| (W78) Policy error - <i>nnnn</i> , contact your system administrator | A noncritical problem has occurred with a policy file or policy certificate. Encryption may be disabled. The number is a policy error code to help your administrator resolve the problem. |

C

Frequently Asked Questions

This section lists some commonly asked questions about PKZIP and related subjects. We hope you will find this information helpful.

Why do I get the message "SYS1041: The name specified is not recognized as an internal or external command, operable program or batch file." or "Bad command or file name" or "XXXX: not found"?

These messages tell you that your operating system cannot find the program to which you are referring. This occurs because you are either not spelling the name of the program correctly, or you did not put a space between the program name and its options, or the program has not been properly installed. If you are trying to run PKZIP and you get this error, it may be because pkzipc.exe is not in your search path.

Why didn't the files I zipped get any smaller?

On occasion, you may find that the files you add to a .ZIP file do not compress. These files are "stored". This occurs when a file is either already compressed or encrypted. You will often find that files distributed with commercial applications are already compressed.

I zipped up a bunch of files but now I have LESS disk space?

When PKZIP compresses files, it makes a copy of the original file. The original file(s) still exist. If you wish to recover space that was taken up by the original file(s), you must either delete them yourself, or instruct PKZIP to delete the file(s) with the ***move*** option.

What is the difference between add=freshen and add=update?

The ***freshen*** and ***update*** sub-options are very similar. This may be confusing at first, but the difference between them is easy to understand.

The ***freshen*** option tells PKZIP to archive any files which match those already in the .ZIP file. These files are re-compressed only if they are newer than the files already in the .ZIP file. Each file is evaluated individually.

The **update** sub-option archives all files, with one distinction. If the **update** option is not used, all files specified are compressed and added to the .ZIP file, even if they already exist in the .ZIP file. By using the **update** sub-option, you instruct PKZIP to compare what is already in the .ZIP file against what it was asked to compress. If a file is already present in the .ZIP file as well as the source directory, PKZIP compresses a file only if it is newer than the copy of the file within the .ZIP file. If a file in the source directory is not already present in the target .ZIP file, PKZIP adds it to the .ZIP file.

Is PKZIP compression "lossy" or "lossless"?

PKZIP uses a "lossless" compression scheme. This means that 100% of the original data is preserved and re-created. There is no difference between the data that you put in and the data that you get back out.

There are other compression methods that are known as "lossy". The idea behind these compression methods is that if you throw away some of the data, it becomes less complex and therefore can be compressed more. This type of compression is only useful for data that need not be precise. This applies to some applications that use pictures and sound.

How do I include subdirectory information in my .ZIP file?

In order to include subdirectory information in your .ZIP file, you must recurse the subdirectories and preserve path names. This is done with the **directories** option. For example:

```
pkzipc -add -directories test.zip *
```

In this example, the current directory as well as all subdirectories and files contained therein are archived in a file called test.zip.

When a .ZIP file is created with paths stored, these paths are visible in a view of the file (**view**).

To re-create these subdirectories, or to place files into their original subdirectories, the **directories** option must be used with the **extract** command.

I zipped up some subdirectories, but I cannot get them to come back.

Did you remember to use the **directories** option when you originally created the .ZIP file? Did you use the **directories** option when you extracted the contents of your .ZIP file? To verify that there are paths in the .ZIP file, do a view of the file:

```
pkzipc -view test.zip
```

If you do not see paths as part of the file names within the .ZIP file, then paths are not stored and therefore cannot be recovered. If you do see paths make sure that you are using the **directories** option when you extract the files. For example:

```
pkzipc -extract -directories test.zip
```

How do I unzip a single file that is in a subdirectory in the .ZIP file?

Type **pkzipc -extract** with the name of the .ZIP file and the name of the particular file you want. With a .ZIP file that contains paths, the procedure is the same.

Assume you are working with a file called test.zip that contains the following files:

```
file1.txt
temp/file2.txt
temp/tut/file3.txt
```

To extract only "file3.txt" from this .ZIP file, you must specify the complete name and path.

```
pkzipc -extract test.zip temp/tut/file3.txt
```

If you wanted to extract it with its subdirectory, simply include the **directories** option on the command line.

How do I unzip a directory without also extracting its subdirectories?

Using the test.zip file we discussed in the previous question, we could extract the entire contents of the temp subdirectory easily:

```
pkzipc -extract -directories test.zip "temp/*"
```

If we did it as shown above we would not only extract all the files in the "temp" subdirectory, but also the "tut" subdirectory below it and any files it contains.

To extract only the "temp" subdirectories contents, and nothing else, we must exclude those directories we do not wish to extract:

```
pkzipc -extract -directories test.zip "temp/*" -exclude="temp/tut/*"
```

If the "temp" subdirectory had multiple subdirectories nested in it, you would need to exclude each one individually on the command line.

I forgot my password; what do I do?

- Try to remember the password.
- Try passwords that are "close" to what you think it was.
- Try mixed upper and lower case versions of your password.

Do not forget or lose your passwords! PKWARE has no special means for "getting around" the encryption and may not be able to assist in the recovery of an encrypted file. To help avoid the loss of data, you may wish to keep a written copy of your password(s) in a secure place.

What does "Unknown Compression Method" mean?

There are many different methods of compression. In the history of PKZIP alone, there have been seven different methods to date. The .ZIP file format was designed so that additional methods of compression can be added as they are developed.

Therefore, the .ZIP file format will never need to be abandoned. This means that the .ZIP file in question was created or updated by a newer version of PKZIP than is being used to extract the data. You must use a newer version of PKZIP to extract these files.

How can I make PKZIP run faster?

PKZIP defaults to a compression method that is average in both compression amount and speed. If you want to get the most speed out of PKZIP, try the following:

- Specify a faster compression method with a level sub-option (e.g., -level=0) or with a speed compression option alone (e.g., -speed). See page 41 for more information.
- Compression speeds are highly dependent on the location of files being added, as well as the temporary file PKZIP creates when performing certain compression operations. If these files are located on a network drive, you may want to move them to a local drive before running PKZIP. Be aware of the effects file location can have on PKZIP's speed.

How many files can be in a .ZIP file?

There is no limit to the number of files you can add to a .ZIP file. However, if you use the **204** option for PKZIP 204g compatibility, your .ZIP file may contain no more than 16,383 file entries.

Can I send a .ZIP file to a different type of computer?

As of the publication of this manual, PKWARE supports PKZIP on MS-DOS, Windows(98, NT, Me, 2000, XP), OpenVMS, HP-UX, IBM AIX, Linux, Sun Solaris, MVS/ESA, OS/390, z/OS, VSE, and OS/400 platforms. PKWARE intends to support additional platforms and will announce this support as it becomes available.

Because PKWARE has dedicated the .ZIP file format to the public domain, it is possible for other people to write programs which can read .ZIP files. We are aware of PKZIP compatible programs for a number of different platforms. A .ZIP file can be transferred to any platform for which you can find a compatible extraction program. Extraction and Compression programs not developed by PKWARE may not be completely compatible with the .ZIP file standard. Contact PKWARE for a list of platforms for which PKZIP and PKZIP compatible software is available.

D

How Does PKZIP Work

This Appendix provides a description of how PKZIP actually does its job. It is not necessary for you to know or understand the information presented here, any more than you need to know how your carburetor works to drive a car. It is presented to help you feel more knowledgeable about the software.

Two Processes

PKZIP performs two functions: compression and archiving. Although the two ideas may seem related, they are actually completely separate.

- **Compression** is the process of representing a given piece of information with a smaller piece of information.
- **Archiving** is the process of combining many files into a single unit, usually along with vital information about each file.

Compression

The actual process used by PKZIP for its compression is too complex to explain in detail. Instead, some of the general principles behind information theory and compression are explained.

To understand data compression, you need to understand two ideas: Information Content and Binary Coding.

Information Content

Everything in your computer, everything you ever read, is "information". The more complex a message is, the higher the information content. The less complex, the less "random" a message is, the lower the information content.

If a message contains a low amount of information, it should be possible to represent it in a smaller amount of space. Look at this page, for example. How much of the page is white space with no letters (information) on it? If you took away all of the

white space this page would be significantly smaller. How many times are the words "the", "information" and "compression" on this page? If you could replace each of these words with something smaller, you would save a significant amount of space.

The more frequently the same group of symbols (in this case, letters) appear, the lower the information content of the message.

The "Field of Information Theory" uses the term *entropy* to describe the "true" information content of a message. Formulas can be used to determine the *entropy* of a message. The idea behind data compression is to derive a new smaller message from a larger original message, while maintaining the *entropy* of the original message.

As a simple example, consider this sentence:

she sells sea shells by the sea shore

This sentence is 37 characters long, including spaces. The spaces cannot be simply thrown away as the meaning of the original message would be lost.

There are obvious patterns to the sentence. The combination 'se' appears three times, 'sh' three times, and 'lls' twice. In fact, the 'se' pairs all have a space in front of them, so these can be ' se'.

she sells sea shells by the sea shore

We can replace each of these patterns with a single character:

#=" se"

\$="sh"

%="lls"

Note that the first replacement string includes a space at the beginning. If we reproduce the sentence with these symbols, it now looks like:

\$e#%#a \$e% by the#a \$ore

The new representation is 24 characters long; this is a saving of 13 characters, or 36%.

Binary Data Representation

All information used, stored, and processed by computers is represented by two values, *zero* and *one*. Everything that you see on your screen, everything stored on disk, is represented by combinations of zero and one.

You can think of it as a sort of Morse Code. In Morse Code there are also only two values, dot and dash. When a computer stores a character, it uses a combination of eight zeros and ones.

Having eight positions in which to store a zero or one gives the computer 256 different possible combinations. You arrive at this number of combinations in this way:

If you have one coin, it can be in either of two positions: Heads(0) or Tails(1)

0 or 1

If you have two coins, there are four possible combinations:

00, 11, 10, 01

If you have three coins, there are eight possible combinations:

000, 001, 010, 011, 100, 101, 110, 111

As you can see, each time you add another coin (binary digit), the number of possible combinations doubles: 2, 4, 8, 16, 32, 64, 128, 256.

The computer uses eight binary digits to get 256 possible values. These values are mapped onto a table called ASCII (American Standard Code for Information Interchange). Each different combination has a particular character that is mapped to it, such as a letter, number or symbol. Each of these positions of 0 or 1 is called a **bit**.

she sells sea shells by the sea shore

The sample message above would be represented by 296 bits (37x8 bits).

If we follow standard ASCII, we have 256 different symbols being represented for our use. The sample sentence we are using only contains alphabetical characters, and only 11 of them at that. If we only need 11 different values, we could use a lower number of bits per character.

The closest value to 11 using binary combinations is 16 combinations, using 4 bits per character. If we wrote a new table of our own using four bits per character, and used it to represent the message, we would use only 98 bits. This would be half as many bits, a considerable savings.

We can do better!

It is possible to have binary codes of varying length. To do this we must use codes with unique values that are not repeated as the beginning of another code. In this way, we can find the codes in a long stream of zeros and ones.

If the codes were not constructed to have unique beginnings, it would not be possible to find each individual code within a long stream of zeroes and ones.

There are many types of coding techniques that produce codes of varying length, based upon symbol frequency. Some well-known coding schemes are Huffman and Shannon-Fanno. PKZIP uses Huffman encoding. The scheme is too complex to document here fully, however, we will discuss some rudiments of encoding. It is necessary for you to understand the principles described here.

A table of variable length codes for 11 symbols would look like this:

| | |
|------|-------|
| 11 | 1101 |
| 110 | 0100 |
| 101 | 1000 |
| 001 | 01010 |
| 1011 | 00000 |
| 0010 | |

As you can see, the codes are getting longer and longer. Because of this, we will get the best results if we map the shortest code to the most common symbol in the message. If you know Morse code, or have occasion to look at it, you will notice that frequent characters, such as 'e', 't', 's' and so on have shorter codes assigned to them. Morse code tends to be about 25% more efficient because of this than it would have been had the codes been assigned at random.

A useful idea here is to allow a symbol to be not only a character, but also a group of characters.

Using the common patterns found in the first analysis of the message, we can map the following table:

| Occurrences | Symbol | New Code | Bits in Message |
|-------------|---------|----------|-----------------|
| 4 | e | 11 | 8 |
| 4 | (space) | 110 | 12 |
| 3 | 'se' | 001 | 9 |
| 3 | sh | 101 | 9 |
| 2 | lls | 1011 | 8 |
| 2 | a | 0010 | 8 |
| 1 | b | 1101 | 4 |
| 1 | y | 0100 | 4 |
| 1 | t | 1000 | 4 |
| 1 | o | 01010 | 5 |
| 1 | r | 00000 | 5 |

Our new coding scheme can represent the message with only 74 bits. This is a savings of 222 bits from the 296 bits used in the "natural" encoding. This is one quarter of the original message size.

One important factor that would affect a real situation is the table we are using. In order for the data to be re-created from the "compressed" representation, we must include a copy of the table used to encode the data.

This can be a seriously limiting factor. If the data is too complex, or the encoding scheme too inefficient, the table used can be as big as the space saved by the encoding. In the worst cases, an attempt to re-encode the message using a table results in the encoded message plus the table being larger than the original message.

This is why data which uses a low number of symbols and frequently repeated combinations of symbols, such as a text file, compresses well. Complex, highly random data, such as the information representing a program on disk is difficult to encode efficiently, and therefore compresses less.

Speed vs. Size

Searching for these patterns, and determining an efficient way to encode the data, takes a lot of computer power and time. The more time taken to analyze the data the better the compression will be. To get more speed, you must sacrifice some level of compression.

There are other steps and methods used in powerful compression schemes such as those used by PKWARE products. Hopefully this explanation gives you a better understanding of what happens when PKZIP compresses data.

Archiving

Programs usually rely heavily on associated data files, or may actually consist of several related programs. Some programs may require dozens or even hundreds of files.

In the "dawn" of the PC age, people wanted a way to keep all of these associated files in one location. "Library" programs were created to take a number of files and group them together into a single file. This made them easier to find, easier to store, and much easier to send to someone by modem. It makes much more sense to be able to send someone a single "package" instead of many files. If you forget a file, all sorts of problems arise.

These programs were the birth of Archiving. In order for a single file to hold many files, information about each file also had to be stored in the archive. This information could then be used by the archival software to locate a file and pull it out, or to list information about the files contained within an archive.

Compression was first available as a utility that would take a single file and produce its compressed equivalent. People began to group files together with a Library program and then compress the archive file.

The next and obvious step in this process was to combine the two ideas. Compress the files and archive them. This made storage very simple; the compression was no longer a separate step and could be taken for granted as part of the archiving process.

PKZIP is the second generation of these programs. PKZIP can not only compress and archive files, but also stores a great deal of vital information about the files. PKZIP even stores directory structures.

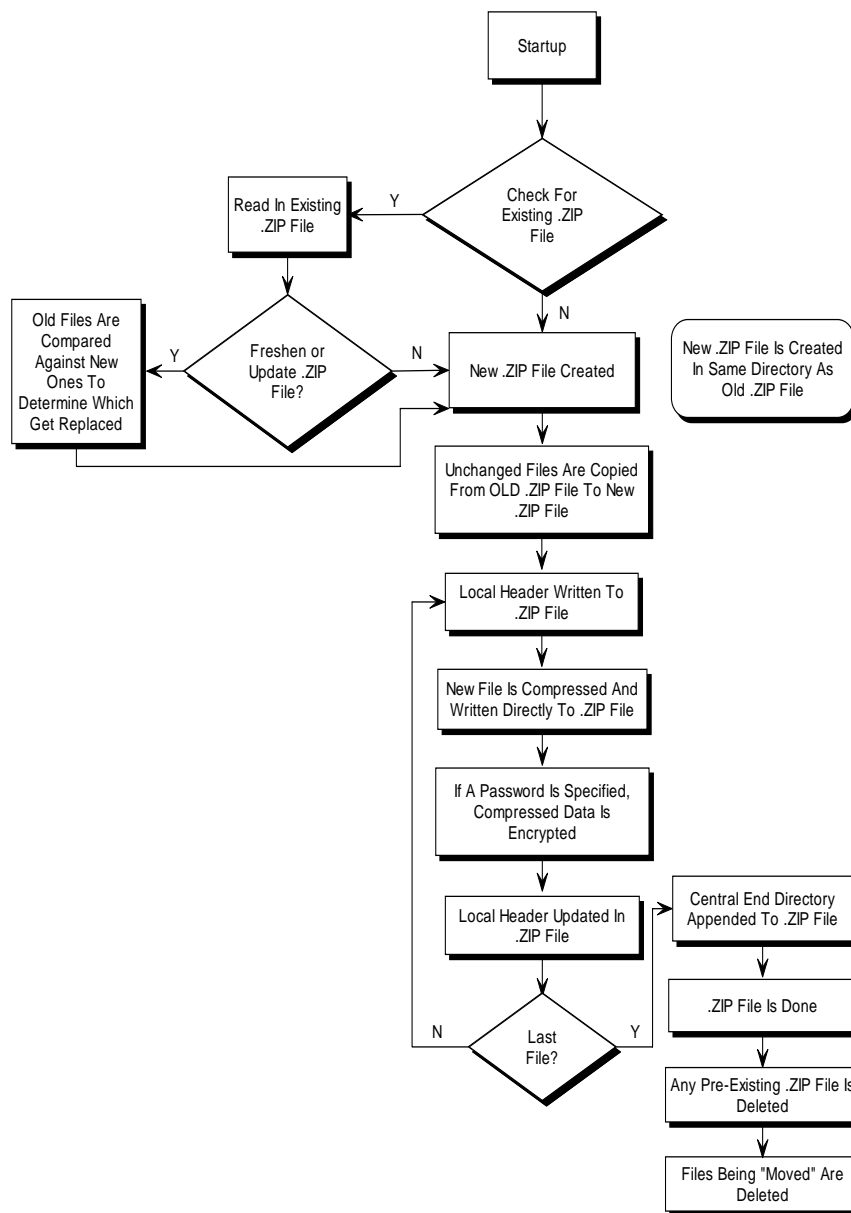
How PKZIP builds a .ZIP File

When you specify a PKZIP command line, PKZIP goes through several steps:

1. Parsing the command line.
2. Reserves the memory it will need to perform the compression, archiving and buffering.

3. Next, PKZIP looks for a .ZIP file with the same name as the one you specified on the command line. If it finds one, PKZIP reads the information on the files that it contains.
4. PKZIP then performs the requested action; it builds a new .ZIP file if none was found.
5. PKZIP reads the information from the command line specifying what files it is supposed to take, what files it should not take, and if there is an **exclude** command.

How PKZIP Builds A .ZIP File



- If a @list file is used, PKZIP reads it, then checks for which files exist. If a pattern is specified in the @list file, PKZIP generates a list of the files which match this pattern.
- If directory recursion has been specified with the **recurse** option, PKZIP next looks for any subdirectories. If it locates subdirectories it goes into them and looks for any files matching the files specified on the command line or in the @list file. If PKZIP finds subdirectories in the subdirectories, it repeats the process. It will continue this process until it finds no additional subdirectories.

Now PKZIP has a list in memory of all the files it should take. The files specified for exclusion are now compared against this list, and any that match are removed. If after this step is complete the list in memory is empty, PKZIP finishes with a message "Nothing to do!".

Now PKZIP reads-in each file, one at a time, and compresses it. When it is finished compressing a file, it adds it to the .ZIP file being created.

6. As PKZIP reads each file, it computes a CRC value for it. This CRC value is stored as part of the information concerning the file.

CRC

This is an acronym for Cyclic Redundancy Check. When a CRC is performed, the data making up a file is passed through an algorithm. The algorithm computes a value based upon the contents resulting in an eight digit hexadecimal number representing the value of the file.

If even a single bit of a file is altered, and the CRC is performed again, the resulting CRC value will be different. By using a CRC value, it can be determined that there is an exact match for a particular file.

PKZIP calculates a CRC value for the original file before it is compressed. This value is then stored with a file in the .ZIP file. When a file is extracted it calculates a CRC value for the extracted data and compares it against the original CRC value. If the data has been damaged or altered, PKZIP can recognize and report this.

1. When PKZIP adds the compressed file to the .ZIP file, it first writes out a "Local Header" about the file. This Header contains useful information about the file, including:
 - The minimum version of PKZIP needed to extract this file.
 - The compression method used on this file.
 - File time.
 - File date.
 - The CRC value.
 - The size of the compressed data.

- The uncompressed size of the file.
 - The file name.
2. After PKZIP has written all of the files to disk, it appends the "Central Directory" to the end of the .ZIP file. This Directory contains the same information as the Local Header for every file, as well as additional information. Some of this additional information includes:
 - The version of PKZIP that created the file.
 - A comment about each file (if any).
 - File attributes (Hidden, Read Only, System).
 - Extended Attributes (If Specified).

Deleting Files from a .ZIP File

PKZIP deletes files from a .ZIP file in the following manner:

1. PKZIP reads in the names of all the files contained in the .ZIP file.
2. PKZIP compares this list against the files you wish to delete.
3. Whatever files remain are moved into a new .ZIP file.
4. The original .ZIP file is superseded by a newer version of the .ZIP file.

This means that in order to delete files from a .ZIP file, you must have enough disk space to hold both the original .ZIP file and the new .ZIP file that lacks the deleted files.

Adding to an Existing .ZIP File

Adding files to a .ZIP file is the same as creating a .ZIP file, but with one difference. Before PKZIP begins to add files, it first reads in the files that were in the existing .ZIP file. These old files and the new files are then both written out to a new .ZIP file, the old files being superseded by the new .ZIP files. This means that there must be enough free space for the old .ZIP file as well as the new .ZIP file to co-exist.

Index

A

About This Manual, 1
 Adding To an Existing .ZIP File, 139
 Additional Methods for Storing Directory
 Path Information, 39
 Archive Attribute, 33
 archiveeach option, 48
 archiving, 132, 136
 Archiving, 136
 ASCII, 134
 Attributes
 Storing, 49
 avargs option, 60
 avscan option, 60

B

Binary, 132
 Binary Data Representation, 133
 BZIP2, 45

C

Clearing Archive Attributes, 32, 33
 Combining Options, 24
 Command Characteristics Chapter, 83
 Command Line
 Including an Option, 23
 syntax, 2
 Command/Option Character
 Changing, 84
 Commands and Options, 22
 Abbreviating, 23
 Default Values, 30
 Difference, 23
 Values, 25
 Commands and Options: Compression or
 Extraction?, 26
 Commands/Options
 204, 51, 86
 add, 30, 31, 87
 after, 26, 87, 90
 altconfig, 81, 88
 archivedate, 53, 88, 117

archiveeach, 89
 archivetype, 89
 ascii, 74
 ASCII/BINARY, 74
 attributes, 49, 90
 avargs, 90
 avscan, 91
 before, 26, 91
 binary, 74, 91
 bzip2, 91
 cd, 36, 92
 comment, 52, 92
 configuration, 93
 configuration command, 76
 console, 63, 93
 curl, 93
 cryptalgorithm, 94
 dclimplode, 94
 default, 94
 deflate64, 94
 delete, 95
 directories, 40, 62, 95
 embedded, 95
 encode, 74, 98
 enterlicensekey, 98
 error, 71, 98
 exclude, 29, 99
 extract, 59, 99
 fast, 42, 99
 fix, 72, 100
 generate list file, 75
 header, 52, 100
 help, 5, 100
 include, 28, 100
 keyfile, 101
 keyphrase, 101
 larger, 28, 101
 level, 42, 101
 license, 6, 102
 listchar, 84, 102
 listcryptalgorithms, 102
 listfile, 102
 locale, 83, 103
 lowercase, 61, 103
 mask, 54

- mask (Win), 104
 - maximum, 42, 104
 - more, 105
 - move, 56, 105
 - movearchive, 75, 105
 - newer, 27
 - noextended, 50, 106
 - nofix, 106
 - normal, 42, 106
 - nozipextension, 106, 107
 - older, 27, 107
 - optionchar, 84, 107
 - overwrite, 66, 108
 - password, 34, 108
 - path, 39, 109
 - preview, 72, 109
 - print, 68, 109
 - recurse, 38, 110
 - shortname, 44, 110
 - silent, 73, 111
 - smaller, 28, 111
 - sort, 54, 63, 112
 - span, 47, 113
 - speed, 42, 113
 - store, 42, 114
 - temp, 73, 114
 - test, 69, 114
 - times, 62, 115
 - translate, 62, 115
 - version, 116
 - view, 67, 116
 - warning, 70, 116
 - wipe, 56, 117
 - zipdate, 53, 88, 117
- Compressing**
- All Files in a Directory, 31
 - All Files in the Current Directory, 14
 - ASCII/BINARY internal attribute, 74
 - Compressing Files Compatible with DCL, 46
 - Compressing Files with Deflate64, 45
 - encode, 74
 - Files From a Different Location, 14
 - Files in a Specific Format, 44
 - Files in SubDirectories, 38
 - Files That Match a Pattern, 13
 - Files With a List File, 43
 - generate list file, 75
 - Incremental Archiving, 33
 - Only Files That Have Changed, 32
 - Only New Files, 31
 - Selected Files in the Current Directory, 12
 - Setting the Compression Level, 41
 - Single File in the Current Directory, 11
 - Specifying a Compression Level by Number, 42
 - Specifying a Compression Level by Option, 42
 - Specifying a Different Location for the .ZIP File, 15
 - Storing and Re-Creating Directory Path Information, 40
 - Storing Directory Path Information, 39
- Compressing Files**
- Learning the Basics, 10
 - That Contain Certain Attributes, 49
- Compressing Files to Diskette, 47**
- Compression, 132**
- Zipfile, 132
- configuration file**
- alternate, 81
- Conventions in this Chapter**
- Compressing Files, 30
- CRC, 138**
- Create a Temporary .ZIP File On a Alternate Drive, 73**
- crl option, 69**
- D**
- Data Compression Library (DCL), 46**
- Date and Time Environment Variables**
- Changing, 83
- Date of the .ZIP File, 53**
- Defaults**
- Changing, 77
 - changing with Options dialog, 79
 - Resetting All Defaults, 81
 - Resetting Individual Defaults, 80
 - Resetting to Original Defaults, 80
- Deleting Files from a .ZIP File, 139**
- Digital Signatures, 64**
- Displaying a Brief View of a .ZIP File, 67**
- Displaying a Detailed View of the .ZIP File, 67**
- E**
- encode, 74**
- encrypting file names, 36**
- encrypting files, 10, 34**
- strong encryption, 35
 - traditional ZIP encryption, 35
 - with passwords, 34
- Entropy, 133**
- Error and Warning Messages, 118**
- Error Messages, 118**
- Extended Attribute Storage, 50**
- Extended Attributes and 204g Compatibility, 51**
- Extended Attributes and the OS, 51**
- Extracting**
- All Files From a .ZIP File, 20

Files Only for Display, 63
 Files With a List File, 63
 lowercase, 61
 Multiple Files From a .ZIP File, 19
 New and Existing Files, 58
 Newer Versions of Existing Files and
 New Files, 59
 Only Newer Versions of Files, 59
 Overriding default settings, 59
 Retaining Directory Structure, 62
 Selected Files, 20
 Single File From a .ZIP File, 18
 Sorting, 63
 times, 62
 translate, 62
 Extracting All Files from an Archive, 59
 Extracting Files
 Further Information, 22
 Extracting Files To a Specified Directory,
 22
 Extracting Files: Learning the Basics, 17

F

Field Of Information Theory, 133
 File Naming Conventions, 10
 Fixing a Corrupt .ZIP File, 72
 format or wipe removable media, 48
 Frequently Asked Questions, 128

G

Getting License Information, 6
 Getting Version Information, 6

H

Header Comment, 52
 How Does PKZIP Work, 132
 How PKZIP Builds a .ZIP File, 136

I

Including a Header Comment, 52
 Including a Password, 34
 Including a Text Comment, 52
 Including Additional Information in a .ZIP
 File, 51
 Incremental Archiving
 Compression, 33
 Information Content, 132
 Introduction, 1
 Introduction Command Characteristics, 83
 Introduction the Basics, 8

L

list files, 43, 63, 75

List Files
 Changing the List Character, 84

M

Moving Files Into Your .ZIP File, 15
 Moving Files to a .ZIP File, 56

O

Overwriting Files, 66
 Overwriting Files That Already Exist in
 Your Directory, 21

P

password, 130
 Passwords, 34
 PKZIP Options Reference, 86
 PKZIP, How Does It Work?, 132
 Preparing Your Workspace, 10
 Previewing Command and Option
 Operations, 72
 Printing the Contents of a .ZIP File, 68

R

Removing File Attributes When
 Compressing, 54

S

selecting files, 26
 Setting Defaults, 26
 Setting Internal Attributes, 74
 Sorting Files Within a .ZIP File, 54
 Spanning/Splitting, 47
 Speed vs. Size, 136
 split sizes, 47
 STDOUT, 33
 Storing File Attribute Information, 49
 Strong Encryption, 4
 subdirectories, 129, 130, 138
 sub-options, 25
 Suppressing Screen Output, 73
 syntax, 2

T

Technical Support, 7
 Testing the Integrity of a .ZIP File, 69
 Two Processes, 132

U

UNIX Command Line, 2
 Using Help, 5
 Using Sample Files and Directories, 9
 Using the Tutorials, 9

V

Viewing Files Within a .ZIP File, 16
Viewing the Configuration File, 76
Viewing the Contents of a .ZIP File, 67

W

Warning Messages, 123
wipe files, 56