RapidFile Toolkit CLI Reference

Version 2.1.1

PURESTORAGE®

Copyright Statement

Pure Storage, Inc. 650 Castro Street Mountain View, CA 94041 *http://www.purestorage.com*

Direct comments to *DocumentFeedback@purestorage.com*.

Table of Contents

Chapter 1
About this Guide

The Pure Storage FlashBlade® RapidFile Toolkit Reference is written for Linux users who access FlashBlade file systems over NFSv3 and NFSv4.1. Users should have a working knowledge of command line tools commonly found on Linux.

This guide provides information about the commands provided with the FlashBlade RapidFile Toolkit and how to use them.

A Note on Format and Content

FlashBlade technology is evolving rapidly. As with all advanced information technologies, once basic architecture is in place, implementation develops at different rates in its different facets, each preceded by feature-by-feature detailed design.

This edition of the guide describes the properties and behavior of arrays that run the latest Purity//FB release. It may include information about planned capabilities whose external form has been specified at the time of the release. Such material is included to provide users of this release with information for design planning purposes, and is subject to change as new functionality is implemented. Material relating to not-yet-implemented functionality is identified as such in the text.

Documentation and Support

All related guides are or will soon be available in the Pure Storage Knowledge Base, which is located at *http://support.purestorage.com*.

Contact Us

We welcome your feedback about Pure Storage documentation and encourage you to send your questions and comments to *documentfeedback@purestorage.com*.

Product Support

If you are a registered Pure Storage user, you can contact Pure Technical Services at *https://support.purestorage.com/Pure_Storage_Technical_Services/Technical_Services_Information/Contact_Us*.

Alternatively, if you would like to submit product feedback, go to *http://support.purestorage.com/?cid=RapidFileToolkit*.

Chapter 2
FlashBlade RapidFile Toolkit Overview

The FlashBlade RapidFile Toolkit is a set of Linux command-line utilities for finding, managing, and copying millions of files stored on FlashBlade. The RapidFile Toolkit provides fast client-side alternatives for common Linux commands (`ls`, `du`, `find`, `chown`, `chmod`, `rm`, `cp`, and `tar`) that have been optimized for the high level of concurrency supported by FlashBlade.

Note: RapidFile Toolkit v2.1.1 has been validated for use with FlashBlade file systems accessed over NFSv3 and NFSv4.1.

The FlashBlade RapidFile Toolkit currently supports searching, copying, and reporting disk usage, with the following commands: `pls`, `pdu`, `pfind`, `pchown`, `pchmod`, `prm`, `pcopy` and `ptar`. These commands are described in detail in subsequent sections of this guide.

Performance Notes

The power of FlashBlade RapidFile Toolkit comes from combining parallel directory traversal with FlashBlade's massively parallel architecture. RapidFile Toolkit v2.1.1 performs best when used with FlashBlade file systems accessed over NFSv3, particularly on large, wide directory trees with many sub-directories.

Note that performance gains are limited if there is only one directory with millions of files.

Performance Notes

The following notes:

- RapidFile Toolkit v2.1.1 supports an accelerated NFS client only on Pure Storage arrays and only for NFSv3.
- RapidFile Toolkit v2.1.1 supports NFSv4.1 in compatibility mode only.
- When RapidFile Toolkit v2.1.1 is used with a FlashBlade file system mounted in Linux using NFSv3, RapidFile Toolkit bypasses the local cache that operating systems use for mounted file systems. This means that sometimes a RapidFile Toolkit tool may make a change on the NFS drive, but the change will not be visible using the **`ls`** command until the caches are manually dropped, or the OS decides

to refresh the caches, which may take some time. For example, consider the following sequence of commands:

```
touch a
 ls -l
 pchmod +x a
 ls -l a
```

When running the first **`ls -l`** command, the OS caches information about file **`a`**. Then, issuing **`pchmod`** changes its mode, bypassing OS caches and directly communicating with the NFS drive using the NFS backend. The second **`ls -l a`** command may then load the old mode from OS cache, even though the file mode is already changed on the NFS drive. Issuing **`pls -l a`** would work as expected. You can delete the caches manually by issuing **`echo 3 | sudo tee /proc/sys/vm/drop_caches`** command.

- Alternatively, you can run RapidFile Toolkit commands with the **`--backend=os`** parameter to use compatibility mode, which utilizes the standard Linux NFS client and the local cache of the operating system.

Requirements

We recommend a minimum of four cores and 16GB of memory. Note that performance generally increases with more cores.

When simulating behavior of their single-threaded counterparts using synchronization mechanisms (mainly `ptar` and `pls`), RapidFile Toolkit programs store a certain amount of metadata in memory. This can lead to memory consumption exceeding 16GB on processing large directory structures. See `--porcelain` configuration option description and `ptar` section for more details on memory issues.

RapidFile Toolkit runs on most Linux distributions. If you encounter any issues running RapidFile Toolkit, please contact Pure Technical Services.

Downloading the Latest RapidFile Toolkit Release

The latest release is available at *http://support.purestorage.com/?cid=RapidFileToolkit*.

Chapter 3
Using the Pure Storage RapidFile Toolkit

The FlashBlade RapidFile Toolkit v2.1.1 works seamlessly with NFS mounts as well as local file systems. The following sections provide information about the RapidFile Toolkit's auto-detection of NFS and local file systems, command option usage, and how to limit NFS RPC requests. RapidFile Toolkit v2.1.1 supports an accelerated NFS client only on Pure Storage arrays and only for NFSv3.

Automatic Detection

Every tool in the RapidFile Toolkit auto-detects whether specified paths point to a directory on a FlashBlade NFSv3 share, or another type of directory, such as a local directory or an NFSv4.1 share.

For example, issuing **`pcopy -r /tmp/srcdir /mnt/mountpoint/tmp`** detects that **`/tmp/srcdir`** is stored locally and **`/mnt/mountpoint/tmp`** is a FlashBlade file system mounted using NFSv3. It relies on Linux to communicate with the local file system and uses its accelerated NFS backend to communicate with FlashBlade.

Automatic detection can be overriden by issuing the **`--backend=os`** option, which uses the toolkit's compatibility mode. When specified, RapidFile Toolkit will use standard system calls to access the files instead of using its accelerated NFS backend. Additionally, a **`--backend=pure-nfsv3`** option can be used to ensure that the tools are using the accelerated NFS backend. If the **`--backend=pure-nfsv3`** option is used with a non-NFSv3 path such as an NFSv4.1 mount, a local directory, or a non Pure storage array, the tool will return an error.

Limiting NFS RPCs

The **`--max-rpcs`** option sets the maximum number of outstanding RPC requests. In other words, it limits number of NFS RPC requests without response at a given moment. This option can be used to reduce load on FlashBlade. For example, by setting **`--max-rpcs`** to 1, the tool becomes serial (since we are not allowed to wait for more than one RPC in parallel).

What does Rapidfile Toolkit Include

The Rapidfile Toolkit is a parallelized implementation of common Linux commands. The RFT commands only contain a subset of options to help the user manage large amounts of files, and user and group permission manageability.

Table 3.1 Parallelized Commands

Linux	RapidFile Toolkit v2.1.1
`ls`	`pls`
`find`	`pfind`
`du`	`pdu`
`rm`	`prm`
`chown`	`pchown`
`chmod`	`pchmod`
`cp`	`pcopy`
`tar`	`ptar`

Use Cases for Rapidfile Toolkit

The Rapidfile Toolkit maximizes speed and efficiency to help the user manage large amounts of files, including user and group permission management.

Table 3.2 Use Cases for RFT

Use Cases	RFT	Use Case Scenarios
Retrieving File Metadata	`pls`	Supports JSON lines output for application integration
Locating Aged Files Taking Space	`pfind`	pfind includes file age filters
	`pdu`	
Fast Deletion	`prm`	Complements .fast-remove and supports piped input
Permissions and Ownership Management	`pchown`	Updating millions of files' ownership
	`pchmod`	
Rapid Data Movement	`pcopy`	Quickly copy to and from scratch space
Rapid archiving and extraction	`ptar`	Quickly create and extract compressed archives. Supports piped input.

Chapter 4
Installing the RapidFile Toolkit

Note: The following procedure uses `rapidfile-toolkit_2.1.1-1` in examples. Replace this file name with the name of the version that you download.

Perform the following to install the RapidFile Toolkit:

1. Review the Pure Storage EULA for Plugin / Adaptor / Provider / SDK / Management Pack license agreement, which governs the use of this product and is available at *https://www.purestorage.com/legal/productenduserinfo.html*.
2. Navigate to the *http://support.purestorage.com/?cid=RapidFileToolkit* and download the latest RapidFile Toolkit `.rpm`, `.deb`, or `.tgz` file.
3. Uninstall any previous versions of RapidFile Toolkit or PureTools.
4. Install RapidFile Toolkit using the appropriate installation package (`.deb`, `.rpm`, or `.tgz`).
    a. On Debian-based Linux distributions such as Ubuntu that use the `deb` file format, perform the following:

    ```
    $ sudo dpkg -i rapidfile-toolkit_2.1.1-1_amd64.deb
    ```

    b. On RHEL, CentOS, SUSE, Fedora, or other Linux distributions that use the `rpm` file format, perform the following:

    ```
    $ sudo rpm -U rapidfile-toolkit-2.1.1-1.x86_64.rpm
    ```

    c. You can alternatively untar the the RapidFile Toolkit binaries into a shared location such as `/usr/local/bin` by performing the following:

    ```
    $ tar -C /path/to/bin -xzvf rapidfile-toolkit-2.1.1-1-linux-x86_64.tgz
    ```

Chapter 5
RapidFile Commands

This section describes each of the FlashBlade RapidFile Toolkit CLI commands.

The commands listed in this chapter are used to administer directories and files.

pchmod

Synopsis

pchmod [OPTION]... ***MODE***[***,MODE***] ***FILE...***

pchmod [OPTION]... ***OCTAL-MODE FILE...***

Options

`--help`

Can be used with any command or sub-command to display a brief syntax description.

`-c | --changes`

Similar to verbose but only displays diagnostics if the file was modified.

`--max-connections`

The number of NFS connections to use per filesystem (default 8). The ***COUNT*** range can be 1 to 512.

`-R | --recursive`

Change files and directories recursively.

`-v | --verbose`

The verbose output displaying diagnostics for every file.

`--version`

Displays the output version information and exits.

Description

The `pchmod` command quickly changes the mode of a directory's contents.

Example 1

```
$ pchmod -R NEWPERMS path/to/directory
```

The command recursively changes permissions of all files in the directory `path/to/directory`. Reference information for NEWPERMS can be found here: *https://www.gnu.org/software/coreutils/manual/html_node/File-permissions.html*

See Also

- *pchmod*

Author

Pure Storage Inc. *documentfeedback@purestorage.com*

pchown

Synopsis

pchown [OPTION]... ***[OWNER]***[***:***[***GROUP***] ***FILE...***

pchown [OPTION]... ***[OWNER]***[***:***[***GROUP***] ***FILE...***

Options

`--help`

Can be used with any command to display a brief syntax description.

`-c | --changes`

Displays changes for a file.

`--from` ***CUR_USR:CUR_GRP***

Change only files owned by CUR_USR/CUR_GRP.

`-h | --no-dereference`

Affect symbolic links instead of any referenced file. This is the default action.

`--max-connections`

The number of NFS connections to use per filesystem (default 8). The ***COUNT*** range can be 1 to 512.

`-R | --recursive`

Displays files and directories recursively.

`-v | --verbose`

Displays verbose output, showing diagnostics for every file.

`--version`

Displays the version information and exits.

Description

The `pchown` command quickly changes the owner or owning group of a directory's contents.

Example 1

```
$ pchown -R NEWUSER:NEWGROUP path/to/directory
$ pchown -R 1001:1002 path/to/directory
```

The command recursively changes owner and group of all files in the directory `path/to/directory`. Either the name or numeric ID can be used.

Note that when using numeric ID, it is first resolved as name, only then interpreted as ID. To skip the name resolution, use a plus (+) character, for example, `pchown -R +1001:+1002 path/to/directory`. For additional information, see *https://www.gnu.org/software/coreutils/manual/html_node/Disambiguating-names-and-IDs.html*).

See Also

- *pchmod*

Author

Pure Storage Inc. *documentfeedback@purestorage.com*

pcopy

Synopsis

pcopy [OPTION]... [-T] ***SOURCE DEST***

Options

`--help`

Can be used with any command or sub-command to display a brief syntax description.

`-b | --block-size` ***SIZE***

Indicate the chunk size for data copy in bytes. The ***SIZE*** range can be 1024 to 524288 bytes. If not specified, the default is 131072.

`--concurrent-chunks` ***COUNT***

The number of file chunks per file to be copied concurrently (default 18 on local filesystem, 100 on NFS). The ***COUNT*** rane can be 1 to 1,000,000.

`-d | --delete`

Deletes extraneous file systems from the destination directory.

`-f | --force`

If an existing destination file cannot be opened, remove it and try again (This option is ignored when the `-n` option is also used).

`-n | --no-clobber`

Prevents overwriting an existing file.

`-P | --no-deference`

Indicate to never follow symbolic links in ***SOURCE***.

`-p | --preserve=` ***ATTR_LIST***

Preserve the attributes of a file or directory which includes **Mode**, **Ownership**, and **Timestamps**. Must be used with the equal (=) character with no spacing, for example, `--preserve=mode,ownership,timestamps`.

`--max-connections`

The number of NFS connections to use per filesystem (default 8). The ***COUNT*** range can be 1 to 512.

`--no-preserve=` ***`ATTR_LIST`***

Do not preserve the attributes of a file or directory which includes **Mode**, **Ownership**, and **Timestamps**. Must be used with the equal (=) character with no spacing, for example, `--no-preserve=mode,ownership,timestamps`.

`-R | -r | --recursive`

Recursively copy directories. The target must be a directory.

`--remove-destination`

Removes each existing destination file before attempting to open. This option contrasts with the `--force` option.

`--sparse=` ***`WHEN`***

Controls the creation of sparse file systems.

`-t | --target-directory=` ***`DIRECTORY`***

Copies all ***`SOURCE`*** arguments into ***`DIRECTORY`***. Must be used with the equal (=) character with no spacing, for example, `--target-directory=DIRECTORY`.

`-T | --no-target-directory`

Treats ***`DEST`*** as a normal file.

`-v | --verbose`

Displays detailed information.

`--version`

Displays the version information and exits.

By default, sparse ***`SOURCE`*** files are detected by a basic heuristic and the corresponding ***`DEST`*** file is made sparse as well. That is the behavior selected by `--sparse=`***`auto`***. Specify `--sparse=` ***`always`*** to create a sparse ***`DEST`*** file whenever the ***`SOURCE`*** file contains a long enough sequence of zero bytes. Use `--sparse=` ***`never`*** to inhibit creation of sparse files.

Description

The `pcopy` command is used for copying files and directories from a source to a destination. `pcopy` does not support copying files that are being edited while `pcopy` is running. Using `pcopy` for file system migration is not supported.

When copying single large file in NFS mode recommended values are different from the default ones and performance can be improved with the block size set to 512KB and number of concurrently copied chunks to 1024 (i.e. `--block-size 524288 --concurrently-chunks 1024`)

Example 1

```
$ pcopy -r path/to/source/. path/to/target
```

The command recursively copies all files from `path/to/source` directory to `path/to/target` directory. The `target` directory will be created if it does not exist. Note that the trailing `/.` in the source path is important, without it the command would create `path/to/target/source` and copy the contents there.

See Also

- *pdu*
- *pfind*
- *pls*
- *prm*

Author

Pure Storage Inc. *documentfeedback@purestorage.com*

pdu

Synopsis

pdu [OPTION]... ***FILE***

Options

`--help`

Can be used with any command or sub-command to display a brief syntax description.

`-0 | --null`

Indicate the null option to end each output line with 0 byte rather than a new line.

`-a | --all`

Displays write counts for all file systems, not just directories.

`--apparent-size`

Displays the apparent size rather than disk usage.

`-B | --block-size=` ***SIZE***

Displays size of the files as number of blocks of ***SIZE*** bytes. For example, `pdu -B 100000` takes number of bytes for each file and divides it by 100,000. The equal (=) symbol must be used with no spaces, for example, `--block-size=1`.

`-b | --bytes`

Displays size of the files as number of bytes. The equal (=) symbol must be used with no spaces, for example, `--apparent-size --block-size=1`.

`-d | --max-depth=` ***N***

Displays the totals only for directories fewer than N levels deep. The equal (=) symbol must be used with no spaces, for example, `--max-depth=1`.

`-h | --human-readable`

Displays sizes in human readable format, for example, 1K, 234M, 2G.

`-k`

Displays files in kilobyte size. Equal to `--block-size=1K`.

`-m`

Displays files in megabyte size. Equal to `--block-size=1M`.

`--max-connections`

The number of NFS connections to use per filesystem (default 8). The ***COUNT*** range can be 1 to 512.

`-S | --separate-dirs`

Displays only the size of directories. Does not include subdirectory sizes.

`--si`

Displays sizes in human readable format in powers of 1000 and not 1024.

`-s | --summarize`

Displays only the total for each argument.

`--version`

Displays the output version information and exits.

Description

The `pdu` command quickly output disk usage of a directory.

Example 1

```
$ pdu -hs /path/to/mountpoint
```

The command displays the disk usage on a mount.

See Also

- *pfind*
- *pls*
- *prm*

Author

Pure Storage Inc. *documentfeedback@purestorage.com*

pfind

Synopsis

pfind [OPTION]... [***PATH...***] [***EXPRESSION***]

Options

`--help`

Can be used with any command or sub-command to display a brief syntax description.

`-daystart`

Measure times (for `-amin`, `-atime`, `-cmin`, `-ctime`, `-mmin`, and `-mtime`) from the beginning of today rather than from 24 hours ago. This option only affects tests which appear later on the command line.

`--max-connections`

The number of NFS connections to use per filesystem (default 8). The ***COUNT*** range can be 1 to 512.

`-version | --version`

Displays the output version information and exits.

Tests

Numeric arguments can be specified as:

- **+N**: For greater than N.
- **-N**: For less than N.
- **N**: For exactly N.

`-amin` ***+N | -N | N***

File was last accessed N minutes ago.

`-atime` ***+N | -N | N***

File was last accessed N days ago.

`-cmin` ***+N | -N | N***

File status was last changed N minutes ago.

`-ctime` ***+N | -N | N***

File status was last changed N days ago.

`-mmin` ***+N | -N | N***

File was last modified N minutes ago.

`-mtime` ***+N | -N | N***

File was last modified N days ago.

`-links` ***+N | -N | N***

File has N hard links.

`-name` ***PATTERN***

File name matches specified pattern.

`-path` ***PATTERN***

File path matches specified pattern.

`-size` ***SIZE***

Filters your search by file size. 512-byte blocks is the default if no suffix is indicated. Accepts suffixes sizes:

- b - Use for 512-byte blocks.
- c - Use for bytes.
- w - Use for two-byte words.
- k - Use for kibibytes.
- M - Use mebibytes.
- G - Use for gibibytes.

`-type` ***TYPE***

File is of type:

- b - block special
- c - character special
- d - directory
- p - named pipe

- f - regular file
- l - symbolic link
- s - socket

`-group` ***NAME***

File belongs to specified group name.

`-user` ***NAME***

File belongs to specified user name.

`-maxdepth` ***LEVEL***

Descend at most N levels of directories below the command line arguments. **-maxdepth** 0 means only apply the tests and actions to the command line arguments.

Actions

`-ls`

Lists matching files in `ls -dils` format on standard out.

`-print`

Prints matching file names on standard out followed by a new line.

`-print0`

Prints matching file names on standard out followed by a null character. Useful for deleting files by piping `pfind -print0` output into `prm --print0`.

Description

The `pfind` command quickly finds a directory's contents.

Example 1

```
$ pfind /path/to/mountpoint -type f -name FILETOFIND
```

The command finds a file named `FILETOFIND`.

See Also

- *pls*

- *prm*

Author

Pure Storage Inc. *documentfeedback@purestorage.com*

pls

Synopsis

pls [OPTION]... [***PATH***]

Options

`-h | --help`

Can be used with any command to display a brief syntax description.

`-a | --all`

Do not ignore entries starting with a period (.) character.

`-A | --almost-all`

Include '.' prefixed directories in the listing, but exclude '.' and '..'

`-h | --human-readable`

Displays sizes in human readable format, for example, 1K, 234M, 2G.

`I | --ignore` ***PATTERN***

Do not list entries that match the your ***PATTERN***.

`-l`

Displays output in long format.

`-n | --numeric-uid-gid`

Displays numeric user and group IDs.

`--max-connections`

The number of NFS connections to use per filesystem (default 8). The ***COUNT*** range can be 1 to 512.

`--porcelain`

Displays output in RapidFile Toolkit v1.0 compatibile mode; unsorted by default and cannot be used with `-U`. This option allows running `pls` with a constant memory consumption and can be useful when processing directories with a large number of immediate children elements.

`--jsonlines`

Displays output in JSON lines format.

`--zero`

Displays lines terminated by a null character.

`-R`

Display files and directories recursively.

`-U`

Do not sort; list entries in directory order.

`-1`

List one file per line (default).

`--version`

Displays output version information and exits.

Description

The `pls` command quickly lists the contents of a directory.

Known Limitations

In order to simulate single-threaded ls output format, `pls` stores elements of listed directories in memory. This can lead to an increase in its memory consumption over the recommended 16GB when listing a directory with a large number of listed elements (files, directories, etc.). Specific number depends on the selected output format (short, long or JSON lines) and lengths of the path and filenames. Thanks to legacy output format support, `pls` can produce unsorted results, not separated into groups by their parent directories with a constant memory consumption. See `--porcelain` option.

Example 1

```
$ pls -R -l /path/to/directory
```

The command lists all files in directory `/path/to/directory` in long format. Similar to the `ls -l` command.

There is a difference in output of `pls -R` and `ls -R` - the order of the directories is different. The `ls` command sorts it, `pls` command outputs them in random order due to performance reasons. Files within one directory appear in the same order.

Example 2

```
$ pls --jsonlines path/to/directory
```

This command lists files in the directory using JSON lines format.

See Also

- *pdu*
- *pfind*
- *prm*

Author

Pure Storage Inc. *documentfeedback@purestorage.com*

prm

Synopsis

prm [OPTION]... ***PATH...***

Options

`--help`

Can be used with any command to display a brief syntax description.

`-d | --dir`

Removes empty directories.

`-0 | --print0`

Null terminated paths from standard input. Useful for deleting files by piping `pfind -print0` output into `prm --print0`.

`-r | -R | --recursive`

Removes the directories and their contents recursively.

`-f | --force`

Ignores non-existing files and arguments; never prompt (default).

`--max-connections`

The number of NFS connections to use per filesystem (default 8). The ***COUNT*** range can be 1 to 512.

`--verbose`

Displays verbose output, showing diagnostic information for each file.

`--version`

Displays version information and exits.

Description

The `prm` command quickly removes a directory's contents.

Example 1

```
$ prm -r /path/to/mountpoint/dir-to-delete
```

The command deletes the directory named `dir-to-delete` and its contents.

Example 2

```
$ prm -d dir
```

The command deletes the directory named `dir` only if it is empty. Note that `rm -rd dir` is equivalent to the `rm -r dir` command (the `-d` flag is ignored with `-r`) and will delete `dir` regardless if it is empty.

Example 3

```
$ pfind /path/to/mountpoint -type f -name FILETODELETE | prm
```

The command finds and deletes any file named FILETODELETE.

See Also

- *pfind*

Author

Pure Storage Inc. *documentfeedback@purestorage.com*

ptar

Synopsis

ptar -c | --create ***ARCHIVE*** [OPTION]... ***FILE...***

ptar -t | --list ***ARCHIVE*** [OPTION]...

ptar -x | --extract ***ARCHIVE*** [OPTION]...

Options

`--help`

Can be used with any command or sub-command to display a brief syntax description.

`-c | --create` ***FILE...***

Creates an archive of one or more files.

`-t | --list`

Lists the contents of an archive.

`-x | --extract`

Extracts or unpacks an archived file.

`-C | --directory` ***DIRECTORY***

Changes to the directory before processing remaining files. This can only occur once.

`--exclude=` **PATTERN**

Excludes files given as a pattern, where ***PATTERN*** indicates the exclusion match.

`-f | --file` ***ARCHIVE***

Indicates the location of the archived file, where ***ARCHIVE*** is the name of the archived tar file.

`-H | --format=` **FORMAT**

Creates an archive of the given format, where the ***FORMAT*** can be `pax`, `posix`, or `ustar`.

- `pax`: POSIX 1003.1-2001 (pax) format (default).
- `posix`: Same as the `pax` format.

- `ustar`: POSIX 1003.1-1988 (ustar) format.

`-X | --exclude-from=` **`FILE`**

Exclude files that match the patterns specified in the ***`FILE`*** input.

`--same-owner`

Attempt to extract files with the same ownership as exists in the archive. This is the default for 'superuser'.

`--no-same-owner`

Extracts files as yourself. This is the default for ordinary users.

`--no-same-permissions`

Implements the user's umask when extracting permissions from the archive. This is the default for ordinary users.

`--max-connections`

The number of NFS connections to use per filesystem (default 8). The ***`COUNT`*** range can be 1 to 512.

`--numeric-owner`

Use numbers for user and group names.

`--posix`

Creates an archive in the posix format. Same as using the `--format=` **`FORMAT`** option, where ***`FORMAT`*** is set as posix.

`-p | --preserve-permissions | --same-permissions`

Extract information about file permissions. This is the default for 'superusers'.

`-v | --verbose`

The verbose output displaying diagnostics for every file.

`-z | --gzip`

Filters the archive through gzip.

Overwrite Control

The following options control the **`ptar`** actions when extracting a file over an existing copy on the disk.

`--overwrite`

Overwrites existing files when extracting. This is the default action.

`--skip-old-files`

Does not replace existing files when extracting. This will silently skip over the existing files.

`-k | --keep-old-files`

Does not replace existing files when extracting. A warning will be given about kept files.

Description

The **`ptar`** command manages files by compressing, extracting, or modifying tar (tarball) archives. This command is similar to zipping and unzipping groups of files in other operating systems.

Extraction and listing of archives can also be used with standard input (stdin), such as **`cat archive.tar | ptar --list`**.

Known Limitations

In order to simulate single-threaded tar behavior and support different overwrite modes, `ptar` stores metadata above extracted directories to make synchronization possible. Because of that, its memory consumption can exceed the recommended 16GB when extracting structures with more than 60 million directories.

Example 1

```
ptar -c three_files.tar File1 File2 File3
```

Compresses the three files (File1, File2, File3) into an archived file named three_files.tar.

Example 2

```
ptar -t three_files.tar
```

Displays the contents in the three_files.tar archived file.

Example 3

```
ptar -x three_files.tar
```

Extracts the contents of the three_files.tar archived file.

See Also

- *pcopy*

Author

Pure Storage Inc. *documentfeedback@purestorage.com*

Chapter 6
Advanced Configuration Options

These flags allow users to override default settings, such as performance constants. They are not a part of stable CLI and may be changed/removed in future releases. However, they may be useful in certain situations, like performance debugging. The following flags apply to all RFT tools, except as noted:

Note: RapidFile Toolkit v2.1.1 is written in Go and supports Go runtime environment variables, as described here, *https://pkg.go.dev/runtime*.

Note: In case if there is an existing NFSv3 mount set up over RDMA (Remote Direct Memory Access), RapidFile Toolkit will detect such mount, but will try to establish its own connection using plain NFS, i.e. without RDMA. NFS over RDMA is not currently supported.

Options

`--backend` ***MODE***

Sets up a backend mode. Mode can have one the following values: **auto** (default), **os**, **pure-nfsv3**.

`--special-dir-paths`

In recursive mode, do not descend into the specified directories. By default, this value is `"/.snapshot,/.fast-remove"`. Paths are comma-separated and each must start with `/`. When empty, no directories are ignored. Only applies to the **pure-nfsv3** backend.

`--debug-log` ***FILE***

Sets up a file to write debug log records. It can also be used with [`/dev/stdout` or `/dev/stderr`] file. Descriptors to redirect the debug log to standard output or error output respectively.

`--max-rpcs` ***COUNT***

Limits the number of NFS RPCs in flight when using the NFS backend. The default is 1024.

`--porcelain`

Displays output in RapidFile Toolkit v1.0 compatibile mode; unsorted by default and cannot be used with `-U`. This option allows running `pls` with a constant memory consumption and can be useful when processing directories with a large number of immediate children elements.

`--rft-experimental-batch-size`

The number of files or subdirectories to process per batch when reading a directory during a file system walk (default 60).

`--rft-experimental-n-concurrent-batches`

The number of batches per directory to be processed concurrently during a file system walk (default 50 on local file system, 100 on NFS).

`--rft-experimental-n-concurrent-dirs`

The number of directories to process concurrently during file system walk (default 50 on local file system, 1,000 on NFS).

`--rft-experimental-rpc-retry-timeout`

The number of seconds after which NFS RPC message will be repeated in case of failure (e.g. network problem). Affects only NFS backend (default 30).

Pure Storage, Inc.
Twitter: @purestorage

2555 Augustine Drive, 1st Floor
Santa Clara, CA 95054 USA
T: 800-379-7873

Sales: sales@purestorage.com
Support: support@purestorage.com
Media: pr@purestorage.com
General: info@purestorage.com