

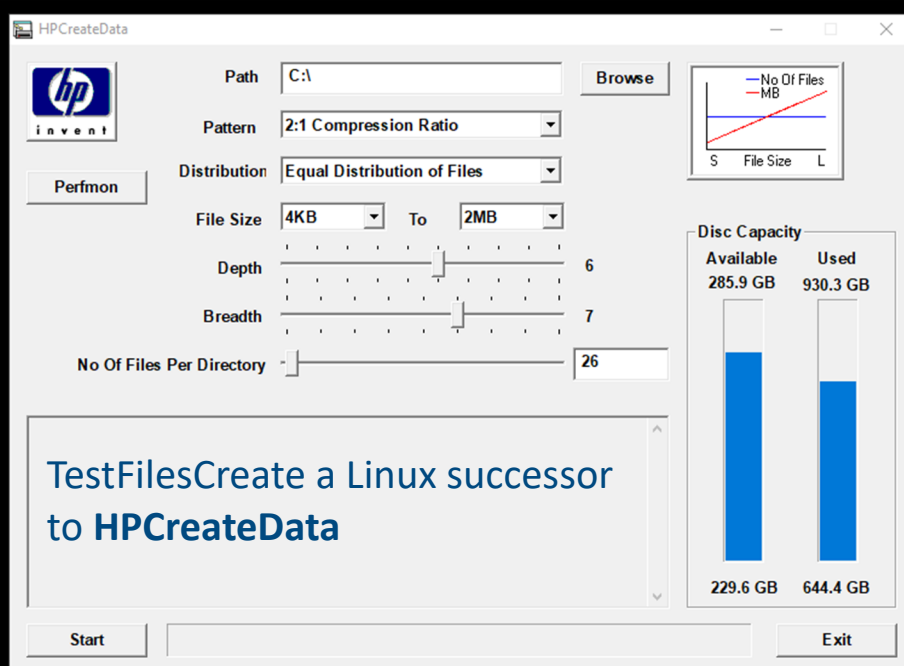
TestFilesCreate

Creates test data files in a single directory or a tree of directories. It is a tool for creating data to test data storage using hard drives and tape, it also can be used evaluating storage compression and deduplication capabilities. Comparative timings of compression software can be performed.

TestFilesCreate can set a benchmark for storage products by creating consistent test data to measure storage efficiency of data compression and data deduplication both at rest in storage systems and inflight during network transmission.

Data is created locally when required. It is not necessary to store test data and transmit between test sites for remote testing.

The complexity of test data is selected from a scale of 1 – 95. A complexity of one has only one printable character repeated, the upper limit is filled with randomized 95 printable characters. The more complex the data is, the more difficult it is to compress.



./TestFilesCreate

```
rFileCreate -P 20 -d 6 -w 7 -s 4K -l 2M -n 26 -o /tmp

DIRECTORY TREE:
Each directory contains 7 directories and 26 files
The tree is 6 levels deep

Output: /tmp/tfc_250518-1136-43

File contents individually generated for each file
Files created in a size range 4K to 2M
Storage used..... 995.72G (max potential)
File Contents..... printable, data complexity of 20
Total data directories.....19607
Total data files.....509808
```

Data creation options

- Selectable data complexity range of 1-95.
- Files are human readable and verifiable.
- Individual file contents can be unique.
- File size can be random, within a size range.
- File size can be fixed, minimum two bytes.
- All files can be identical.

Create a single file or millions.

Output to a single directory or a directory tree.

Filling files with random binary is available.

Sparse files are not available.

Limited to creating 100 million directories and 100 million files.

Calculator mode

```
./TestFilesCreate -C

File size requires units of B, K, M or G, the number must >1
decimals not permitted i.e. 750K 34M 2G and 10B (10 bytes)

NOTES:
Two bytes (2B) file size is the minimum to build a tree
One byte is permitted here for information only
SINGLE DIRECTORY only, set Depth = 1

Data tree directory depth (-d): 6
Data tree directory width (-w): 7
Files per directory (-n): 26
Size per file (-f or -l): 2M

Total number of directories : 19607
Total number of files : 509808
Total file storage usage : 995.72G

Top Level directory (tree depth = 1) total file usage 52.00M
This directory contains 26 files, each 2.00M
Content counts and storage used by this top level is constant
and unaffected by subdirectory count (tree width)

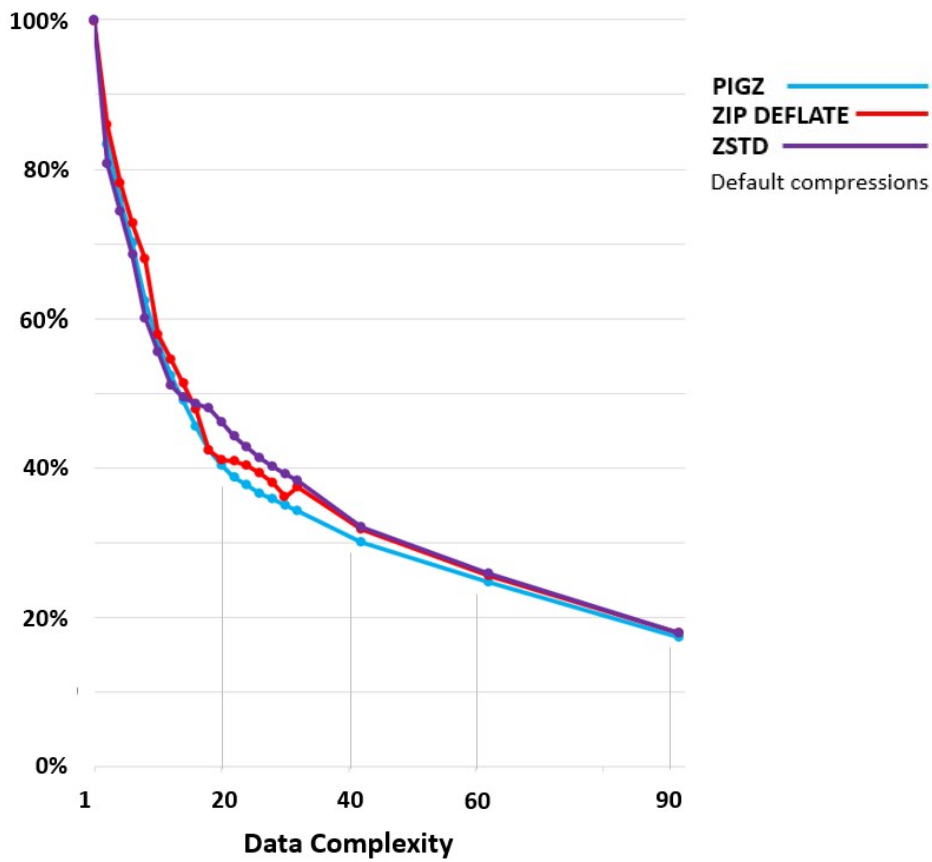
Directory count for each tree
Tree Depth 2 3 4 5 6
Width 1 | 1 2 3 4 5
Width 2 | 2 6 14 30 62
Width 3 | 3 12 39 120 363
Width 4 | 4 20 84 340 1364
Width 5 | 5 30 155 780 3905
Width 6 | 6 42 258 1554 9330
Width 7 | 7 56 399 2800 19607

File count for each tree
Tree Depth 2 3 4 5 6
Width 1 | 52 78 104 130 156
Width 2 | 78 182 390 806 1638
Width 3 | 104 338 1040 3146 9464
Width 4 | 130 546 2210 8866 35490
Width 5 | 156 806 4056 20306 101556
Width 6 | 182 1118 6734 40430 242606
Width 7 | 208 1482 10400 72826 509808

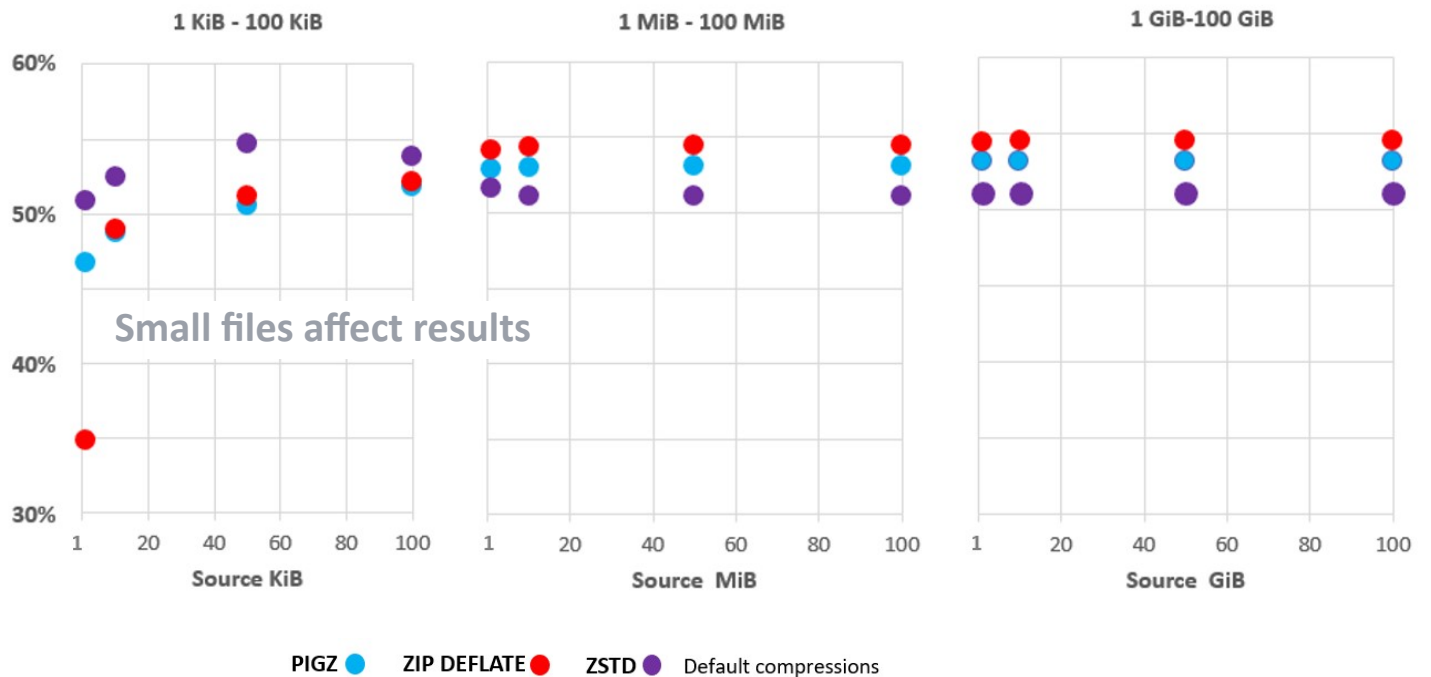
File storage usage for each tree
Tree Depth 2 3 4 5 6
Width 1 | 104.00M 156.00M 208.00M 260.00M 312.00M
Width 2 | 156.00M 364.00M 780.00M 1.57G 3.20G
Width 3 | 208.00M 676.00M 2.03G 6.14G 18.48G
Width 4 | 260.00M 1.07G 4.32G 17.32G 69.32G
Width 5 | 312.00M 1.57G 7.92G 39.66G 198.35G
Width 6 | 364.00M 2.18G 13.15G 78.96G 473.84G
Width 7 | 416.00M 2.89G 20.31G 142.24G 995.72G
```


Benchmark examples

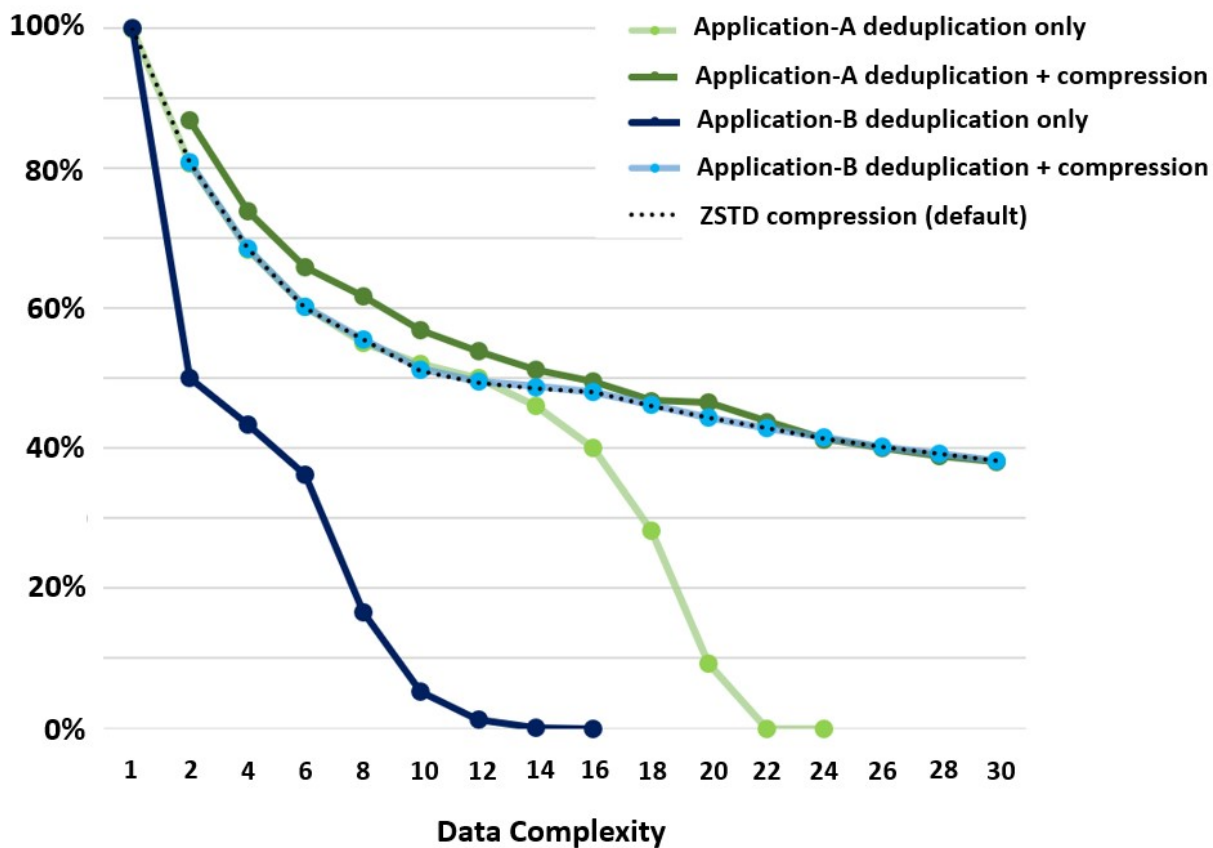
PIGZ, ZIP, ZSTD storage savings (%)



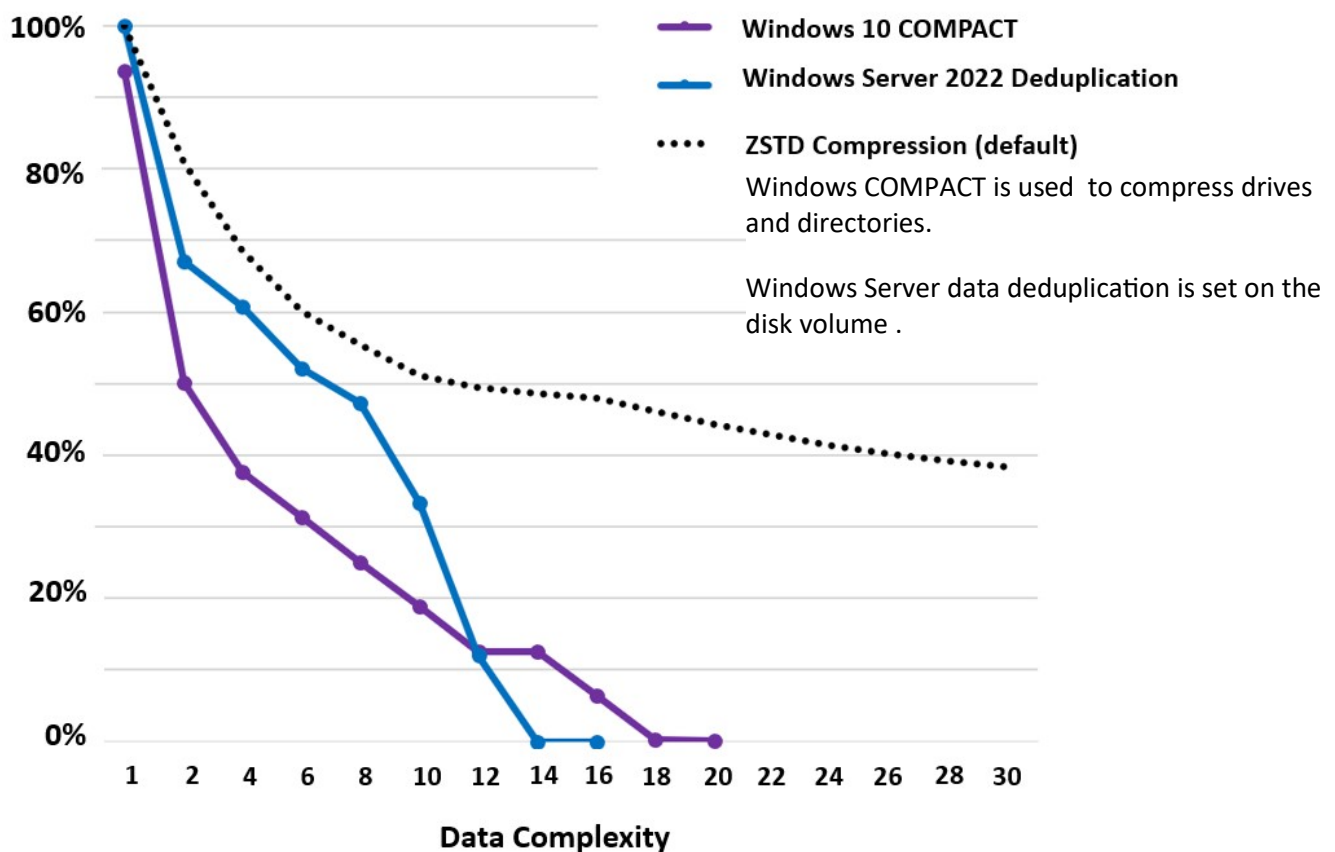
Storage Savings variable file size storage savings (%)



Backup Application storage savings (%)



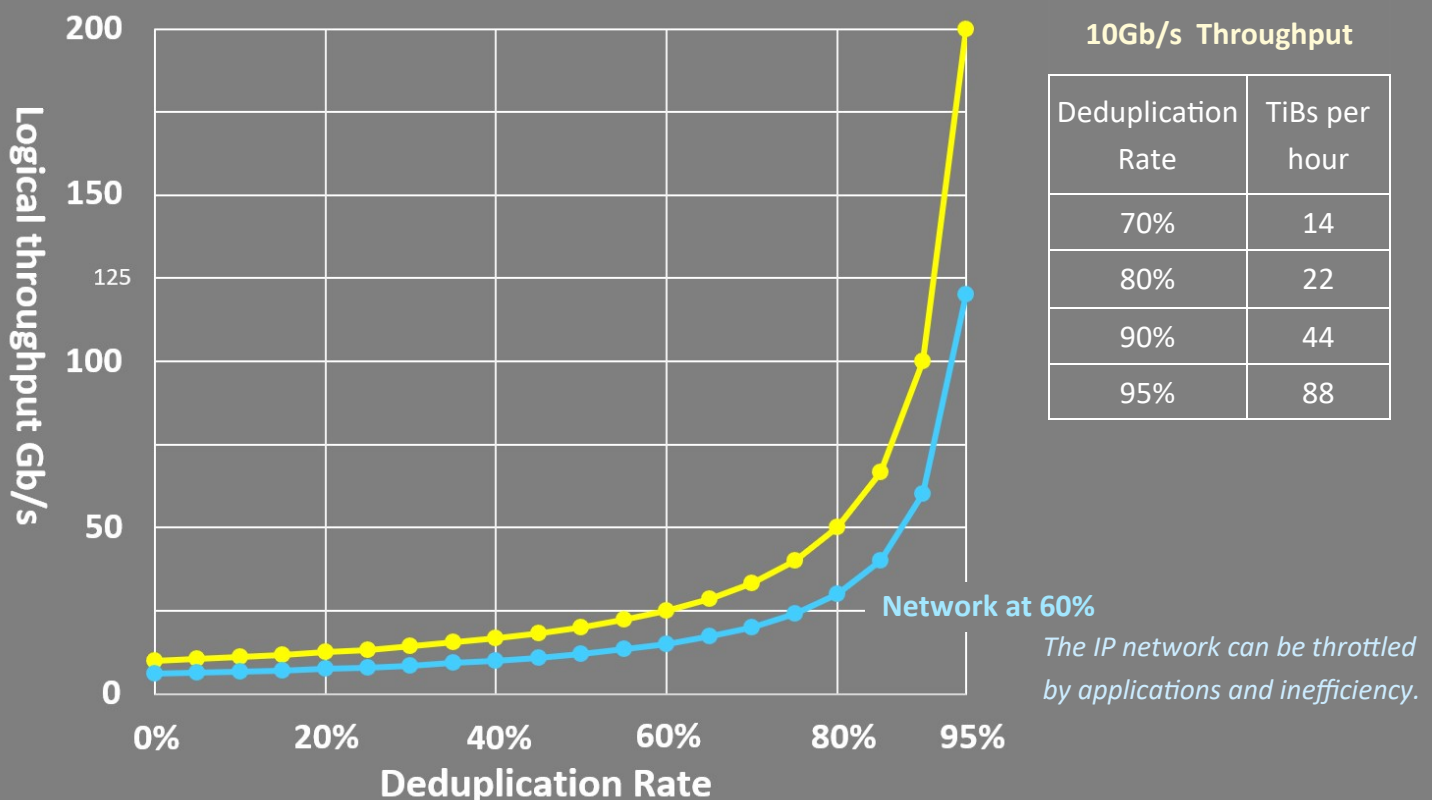
Storage Device storage savings (%)



Testing the IP network performance of deduplication.

Some storage appliances and applications offload data deduplicated to the source device resulting substantial network throughput improvement. During transmission only unique data chunks and the deduplication signatures of repeated data chunks are transmitted.

Deduplication Rate and Logical Network Throughput



Notes.

Storage savings from compression and deduplication is only one factor, the importance of efficiently and quickly producing these savings are often more important in selecting data reduction techniques.

Storage that uses network enabled deduplication may come with guidelines regarding its use because some network devices use deduplication to improve efficiency across WAN and other networks.

The PIGZ and ZSTD **—rsyncable** option is used in PIGZ and ZSTD to create rsync friendly compressed files. The use of the rsyncable option has been found to be beneficial in some deduplication applications and may significantly influence deduplication test results. The rsyncable option in PIGZ and ZSTD has little impact on compressed file size.

Test data for examples.

PIGZ, ZIP, ZSTD, Backup Applications and Device Storage

Each data point is a minimum of two tests using 99.6 GiB directory trees containing 6,800 15 MiB files in 340 directories. If the two tests agreed no further tests was done.

PIGZ, ZIP, ZSTD Variable File Size

Each data point is three tests using a data complexity of 10 i.e. files were filed with a random sequence of *abcdefghij* characters.

TestFilesCreate available from <https://github.com/Jim-JMCD>

TestFilesCreate is a Linux only bash script protected by a *shc* generated executable wrapper.