

**NAME**

**ext2frag** – Report fragmentation statistics for an EXT2 file system

**SYNOPSIS**

**ext2frag** [ **-v**] *IMAGE*

**DESCRIPTION**

The **ext2frag** directly accesses an EXT2 file system, given by the *IMAGE* argument, in order to gather file and free space fragmentation statistics. *IMAGE* may be a regular file or a raw block device which may hold even a mounted file system. The output consists of a series of lines, each prefixed with a key letter followed by a colon, followed by information related to the key. The output format has been designed for easy parsing by machines, rather than for human consumption.

In the rest of this document, the word “**extent**” is used to denote a contiguous range of file system blocks. For example, blocks 618, 619 and 620 comprise a single extent of size 3. See section FRAGMENTATION for explanation of various statistics computed by this program.

**SUMMARY STATISTICS**

In default mode, the program reports only summary statistic as a series of **H:** lines, where each line has **H:key=value** format. See sections below for short explanation of various fragmentation types and metrics that are reported. The following keys are currently defined:

<code>block_size</code>	The file system's block size (EXT2 supports a block size of 1024, 2048 or 4096 bytes).
<code>total_blocks</code>	Total size of file system (in blocks).
<code>free_blocks</code>	Number of free (unallocated) blocks.
<code>free_extents</code>	Number of free extents.
<code>free_extent_size_summary</code>	5-number summary of free extent size distribution.
<code>ext_frag</code>	External fragmentation ratio (0-1); 1 is worst.
<code>total_files</code>	Total number of <i>regular</i> files on the file system.
<code>file_blocks</code>	Number of blocks allocated for files.
<code>file_bytes</code>	Total number of bytes needed for files.
<code>file_fragments</code>	Total number of file data fragments on the file system.
<code>file_size_summary</code>	5-number summary of file size distribution.
<code>data_frag</code>	Data fragmentation ratio ( $\geq 1$ ); 1 is ideal.
<code>int_frag</code>	Internal fragmentation ratio ( $\geq 1$ ); 1 is ideal.
<code>backward_files</code>	Number of files with backward jumps.
<code>backward_jumps</code>	Total number of backward jumps for all files.
<code>file_locations_summary</code>	5-number summary of disk locations used by file data.

The 5-number summary of a size distribution consists of the following 5 numbers: minimum, 1st quartile, median, 2nd quartile and maximum. It is useful for estimating skewness of the distribution because it reflects the clustering trends of the data: 50% of all values fall into the range between 1st and 3rd quartile.

**RAW OUTPUT**

When the **-v** option is given, raw file system information is reported in series of **F:** and **R:** lines. The **F:** line contains data about free extents formatted as a comma-separated list of block and block ranges, for example:

```
F:125892-125908,125910,125913-126975,
```

(block list always ends with a trailing comma). Ranges are *inclusive*.

R: lines report data about *regular files*. One line for each file is output with fields separated by semicolons, for example:

```
R: 30691;1532176;79879,80028-80031,80115-80121,80123-80485,
```

The fields are inode number, file size in bytes and list of block ranges used by file data.

## FRAGMENTATION

**External fragmentation** is the lack of contiguous storage (i.e. large free extents): there may be a lot of free space on disk, but it is chopped up into many small extents. The external fragmentation ratio `ext_frag` is computed as  $\text{ext\_frag} = 1 - M / \text{free\_blocks}$  where `M` is the size (in blocks) of largest free extent. The value is between 0 (best) and 1 (worst). The best value is achieved when  $M == \text{free\_blocks}$  i.e. when all free space is in a single extent. External fragmentation directly contributes to file data fragmentation.

**Internal fragmentation** is loss of usable space that results from block-oriented nature of EXT2 file system. Since all space is allocated in multiples of block size, the unused space at the end of each file (up to the block boundary) is wasted and contributes to internal fragmentation. The internal fragmentation ratio is computed as follows:  $\text{int\_frag} = \text{file\_blocks} * \text{block\_size} / \text{file\_bytes}$  The value is always  $\geq 1$ , 1 being best.

**Data fragmentation** is a consequence of external fragmentation: when a file is too large to be placed in the largest free extent, it must be split into multiple extents (fragments). The ratio reported by the program is  $\text{data\_frag} = \text{file\_fragments} / \text{total\_files}$  (the average number of fragments per file). Best value is 1 (no file is fragmented).

“**Backward files**” are stored in such a way that reading a file sequentially, by increasing *logical* block numbers, sometimes leads to a *backwards* jump in *physical (file system)* block numbers. This may be detected by inspecting the file data block list: if the block numbers are not strictly increasing, there are backward jumps in the file. This is significant performance indicator due to the way hard disks work.

## WARNING

This program accesses the file system directly, which means that the usual OS protection mechanisms are not in place. In the unlikely case of bugs, either in this program or in supporting libraries, data loss may result. **USE AT OWN RISK!**

## BUGS

The program currently does not support file systems with large files.

There is discrepancy between the number of free blocks reported by `dumpe2fs`, which takes the information from `superblock`, and this program which computes the information from available block bitmaps; the discrepancy happens even on *unmounted* file systems. Further checks have revealed that both this program and `dumpe2fs` report the *same* set of free blocks, so the cause of the discrepancy still remains unknown.

## SEE ALSO

The `libext2fs` library is part of the `e2fsprogs` package.