Formatted vs unformatted size of filesystems on linux: ext3, ext4, xfs, zfs and reiserfs

A small comparison of formatted space differences between ext3, ext4, xfs, zfs, btrfs, ntfs and reiserfs.

Every one of us has beliefs. One of mine was that if I finally have the time and courage to start using ZFS on linux[^1] compression and encryption will be some much easier and I'll have so much more space on my disk.

As most beliefs, this proven to be wrong.

First of all, ZFS on linux does not support encryption, so I needed to add a LUKS layer. After that I turned on lz4 compression and let my copy go for the night. I have a few things and copying 845GB takes a while.

The reason I thought I can gain a little space is that I have all kind of data on that disk; what I was unaware of if that how little the compressable data is out of the all. JPEG is compressed, RAW images are compressed, flac, mp3, mp4, xvid - all compressed. So in reality, on your media drive, you won't get any good of the lz4 compression - alltogether I only gained 10GB and I suffered a significant performance loss. If I'd have RAID and I was on BSD, where encryption is an actual option, I would have stayed with ZFS, but for this purpose it did not suite me at all.

After this brief experience, since my data could be formatted again, I tried out a few different filesystems. On ext3 and ext4 I even added the -m 1 magic, to reduce the reserved space to 1% and the results are the following:

	Size	Size -h	Used	Used -h	Free	Free -h
unformatted	999664124	999.7G				
ext3	960783720	917G	73496	72M	911898500	870G
ext3 tune2fs -m 1	960783720	917G	73496	72M	950947880	907G

	Size	Size -h	Used	Used -h	Free	Free -h
ext4	960783720	917G	73364	72M	911882248	870G
ext4 tune2fs -m 1	960783720	917G	73364	72M	950931628	907G
xfs	975757820	931G	34208	34M	975723612	931G
zfs	957874048	914G	128	0	957873920	914G
reiserfs	976204696	931G	32840	33M	976171856	931G
ntfs	976234492	932G	95780	94M	976138712	931G
btrfs	976234496	932G	16896	17M	974107392	929G

The winner is ReiserFS - but it's old and abandoned and there is a bit of a moral issue with it.

The next is XFS, which would have been my choice - but it does not support file creation time[^2], which, for historical reasons, I want to have. My filed used to be on NTFS before my linux times, so I do actually have creation dates as I migrated the data to ext4 from there, also storing creation dates.

NTFS is a brilliant filesystem, a full-fledged beast - sadly not designed for linux and to get POSIX permissions on them is a bit too tricky.

Therefore I decided to go with the new kid on the block: btrfs. I was finally titled stable a while ago and it gives me 22 more GB to play with, and it has a semi-intelligent compression option which, in theory, will skip compressing already heavily compressed data, which is good for performance. I do have a bad feeling about it, but my data is backed up on more mature filesystems as well.

Links

1. http://zfsonlinux.org/
2. https://en.wikipedia.org/wiki/Comparison_of_file_systems

Created by Peter Molnar <mail@petermolnar.net>, published at 2015-07-20 09:08 UTC, last modified at 2021-05-11 11:49 UTC , to canonical URL https://petermolnar.net/article/why-use-btrfs-for-media-storage/ , licensed under CC-BY-4.0 .