**Hot swappable Filesystems, as smooth as BTRFS**

**By Tedley Meralus**

**Hot swappable Filesystems, as smooth as BTRFS**

Filesystems, like file cabinets or drawers, control how your operating system stores data. They also hold metadata like filetypes, what is attached to data, and who has access to that data. For windows or macOS users

Quite honestly, not enough people consider which file system to use for their computers.

Windows and macOS users have no valid reason to look into filesystems because they have one that’s been widely used since its inception. For Windows that’s NTFS and macOS that’s HFS+. For Linux users, there are plenty of different file system options to choose from. The current default in the Linux field is known as the Fourth Extended Filesystem or ext4.

Currently there is discussion for changes in the filesystem space of Linux. Much like the changes to the default init systems and the switch to systemd a few years ago, there has been a push for changing the default Linux filesystem to the BTRFS. No, I'm not using slang or trying to insult you. BTRFS stands for the B-Tree file system. Many Linux users and sysadmins were not too happy with its initial changes. That could be because people are generally hesitant to change, or because they change may have been too abrupt. A friend once said, "I've learned that fear limits you and your vision. It serves as blinders to what may be just a few steps down the road for you." In this article I want to help ease the understanding of BTRFS and make the transition as smooth as butter. Let’s go over a few things first.

**What do Filesystems Do**

Just to be clear, we can summarize what filesystems do and what they are used for. Like mentioned before filesystems are used for controlling how data is store after a program is no longer using it, how to access that data, where that data is located, and what is attached to the data itself. As a sysadmin, one of the many tasks and responsibilities is to maintain backups and manage filesystems. Partitioning filesystems help with separating different areas in business environments and is common practice for data retention. An example would be taking a 3tb hard disk and partitioning 1tb for your production environment, 1tb for your development environment, 1tb for company related documents and files. When accidents happen to a specific partition, only the data stored in that partition is affected, instead of the entire 3tb drive in this example. A fun example would be a user testing a script in a development application that begins filling up disk space in the dev partition. Filling up a filesystem accidentally, whether it be from an application or a user’s script or anything on the system, could cause an entire system to stop functioning. If data is partitioned to separate partitions, only the data in that partition will be full or affected, so the production and company data partitions are safe.

The ext4 filesystem used in Linux systems, for example, is a journaling file system which means it keeps a journal of where files are located on a drive and journals changes that happen on the drive.

**Advantages of Ext4 filesystem**

The ext4 filesystem used in Linux systems, for example, is a journaling file system which means it keeps a journal of where files are located on a drive and journals changes that happen on the drive. Some advantages of ext4 partitions are that you can do the following:

\* have encryption on specific volumes

\* you can mount volumes on demand

\* you can snapshot, format, or defragment volumes independent of each other.

Some of the drawbacks of ext4 are:

\* more chances of filling up disk partitions with isolation

\* increased administrative overhead for whoever is managing these disks

\* there are difficulties creating a consistent snapshot for applications running on different partitions.

\* ext4 uses more resources than other filesystems.

\* doesn't support transparent compression

\* doesn't support transparent encryption

\* doesn't support data deduplication

**Making change for the BTR... I mean Better**

Btrfs, is pronounced as Butter FS, Better FS, or B-Tree FS, although I don't think people don't use the latter. It is a CoW (copy on write) file system remade from the ground up. Developers wanted to expand the functionality of a file system to include additional functionality for better snapshots and pooling volumes. The author of BTRFS stated that the goal of BTRFS is "to let Linux scale for the storage that will be available". Btrfs expands the functionality of ext4 and adds features like having a seemingly continuous file system across the multiple hard drives for consolidating data across large partitions. Some of the best features of Btrfs are:

\* Data deduplication

\* Data mirroring

\* Self-healing of and checksums for data and metadata

\* Customizable retention policy

\* Secure data consistency of backups.

\* adding new drives to full storages drives for data retention

Data deduplication can reduce the amount of the actual space data could occupy, and the data mirroring could become easier with Btrfs as well. Files are the most important parts of data on your hard drive. Btrfs has other features like scrubbing, self-healing data, and other features for data integrity and security. RAID features can be used to remove devices online, freely convert between RAID levels after the filesystem has been created and mirror across large devices. Having the versatility to update and change the raid levels or verify checksums are verified along with repairing broken copies of files can be enough to consider making the switch.

**Examples**

A few examples and some commands to be used with BTRFS are

**Balance Filesystem**

Balancing a filesystem spreads the data across different disks in an even manner. If disks are not equal in size, equality will be by percentage capacity utilized (all disks will be at the same % utilization). A good time to do this would be after adding more disks to a volume.

$ btrfs balance start [btrfs mount point]

The balance command can take a very long time. Check the progress with the following command:

$ btrfs balance status /path/to/mount

**Changing RAID Levels**

Changing RAID levels is like balancing filesystems. In most cases on an ext4 system you would need to format and backup the filesystem to set it up as for RAID functionality. With BTRFS you can setup a RAID a bit easier.

\* NOTE: I would not suggest doing this if you are not sure what RAID is or have little knowledge of high-level concepts of device storage and mirroring.

For setting up RAID on more than one device you can run this command:

In this example sda1 and sdb1 are the two devices being setup.

$ mkfs.btrfs -m raid1 -d raid1 /dev/sda1 /dev/sdb1

Currently acceptable RAID levels are RAID0, RAID1, and RAID10. Raid10 requires 4 or more disks.

**RAID10**

Creating a RAID10 striped mirror in Btrfs would look something like this.

$ mkfs.btrfs -m RAID10 -d RAID10 /dev/sda1 /dev/sdb1 /dev/sdc1 /dev/sdd1

**Defragment**

Defragment a file or a directory to check its health, type this command:

$ Btrfs filesystem defragment /dev/sda1

**Scrubbing**

Scrubbing a filesystem can help remove errors on a device. To scrub possible errors on

/dev/sda1 you can run this command

$ sudo btrfs scrub start /dev/sda1

Scrubbing is done in the background of a filesystem so you can still use a filesystem while scrubbing is taking place. You can check the scrub status by running this command:

$ sudo btrfs scrub status /dev/sda1

**Adding a Device**

You can add devices to help extend storage on btrfs by running this command

$ btrfs device add -f /dev/sda1

A rule of thumb after adding a device would be to balance the filesystem.

**Removing a Device**

$ btrfs device delete /dev/sda1

**Delete missing devices**

IF you wanted to delete missing devices or devices that may have failed you can run these commands. The first command will mount the device in degraded mode before deleting it.

$ sudo mount -o degraded /dev/sda1

$ btrfs device delete missing

**Sub volumes And Snapshots**

Sub volumes are the BTRFS equivalent of "datasets". They are separate volumes that can be used for snapshotting/restoring. You may want to create a sub volume for any KVM guest you create or mountpoint with critical data in it.

**Create a Sub volume**

$ btrfs subvolume create

**Show sub volumes**

$ btrfs subvolume list

**Take Snapshot**

$ btrfs subvolume snapshot

**Delete Sub volume**

$ btrfs subvolume delete

**In conclusion**

There are many filesystems to choose from for getting the right jobs done, and with Linux you have the option to pick and choose. Btrfs has been a part of the Linux Kernel since 2013 or so and continues to enhanced improvements to data retention and protection. Most distros still use ext4 as their default filesystem but if you are looking to move to an OS that is already using one you can investigate OpenSUSE. OpenSUSE now uses btrfs as their default for their /root partition and use XFS file system for the /home partition currently. For more people, ext4 may be old and outdated but still resilient and reliable. However, there is a new spread to add to your filesystem that can help with better resilience and reliability on multiple drives and partitions. Btrfs is to butter what your filesystem/storage is to bread. John F Kennedy once said "Change is the law of life. And those who look only to the past or present are certain to miss the future.” Try Btrfs for yourself and let me know what you think about it.