Getting Buttery with Fedora

An introduction to Btrfs

Neal Gompa, Josef Bacik, Dusty Mabe

Who are we?

Neal Gompa

- Contributor and package maintainer
- Contributor to various system management projects
- DevOps Engineer at Datto, Inc.
- Twitter: @Det_Conan_Kudo
- Email: ngompa@fedoraproject.org

Josef Bacik

- Core Btrfs developer
- Software Engineer at Facebook
- Email: josef@toxicpanda.com

Dusty Mabe

- Fedora Contributor
- Involved in the Fedora Cloud and Fedora CoreOS groups
- Software Engineer at Red Hat
- Twitter: @dustymabe
- Email: dusty@dustymabe.com

Introducing Btrfs

So… What is Btrfs?

From the Btrfs wiki:

- *Btrfs is a new copy on write (CoW) filesystem for Linux aimed at implementing advanced features while focusing on fault tolerance, repair and easy administration.*

Err, what is Copy on Write?

The label Copy on Write (CoW) refers to a type of filesystem optimization strategy where each modification to the filesystem is written in a new location while the original remains preserved.

By doing this, it’s possible to preserve each instance of the filesystem and move back and forth through the instances.

It is ***NOT*** a replacement for proper backups, but it does provide some safety that isn’t possible in traditional filesystems.

How big can Btrfs get?

64-bit filesystem, max volume size is 16 EiB

- Approximately 18,446,744 terabytes!
- At 100GB per 4K (QHD) full-length film, it'd take 184,467,441 copies to fill the whole volume at max size!
- This is more data than what is even possible (or even desired!) to record today on any single disk or disk array.

What are the features of Btrfs?

- Space efficient storage/packing of small files
- Space efficient indexing of directories
- Subvolumes & quota support for subvolumes
- Read-only and writable snapshots
- Sending/receiving volume data with efficient deltas
- SSD awareness and SSD-specific optimizations
- Integrated disk management & multiple disk support
  - RAID 0, 1, 5, 6, 10 support
  - Dynamic resizing (shrink/grow) arrays/volumes after initial array creation
- Transparent on-disk compression
- Seeding from other filesystems
- And much more...

Subvolumes? Snapshots?

Subvolumes are subsections of a volume that can be independently managed. This is useful if you want to have different snapshotting schedules for portions of your volume.

Snapshots are instances of (sub)volumes that are preserved. With the appropriate tools and configuration, snapshots can be used as a means to provide “Time Machine” style data recovery or even to save a system from a bad software install/upgrade.

Who made Btrfs?

It is principally developed by:

FB

SUSE

Western Digital®

ORACLE®

So why use Btrfs?

As a filesystem that is developed within the mainline kernel, it takes advantage of facilities provided in the kernel to be more efficient at doing operations on devices.

Linux distributions also have support for Btrfs out of the box, and can be used with minimal effort.

Who uses Btrfs?

It is used in production by:

facebook

openSUSE

Rockstor

Synology®

Thecus®
Empowering Professionals

NETGEAR

Facebook production use

- 90% of the fleet (millions of machines) have Btrfs root file systems, and have been this way for three years.
- Every container in the fleet is Btrfs based, and has been this way for 4+ years.
- Workloads vary widely, not simply "web servers"
- Every developer VM in the company uses Btrfs

Facebook production wins

- Compression on everywhere, drastically reduced burn rates of our SSDs
- Snapshots drastically improved build and test times for our build system
- Async discard drastically reduced discard related latencies with our SSDs

Facebook production losses

- Still not great use for databases
- Checksumming and general heavier metadata usage results in higher latencies in some workloads
- We find bad hardware much more quickly
- My Mtn Dew consumption is at an all-time high

Btrfs and Fedora

Current state (Fedora 33)

- Anaconda has been configured to install non-server variants with Btrfs
- Disk images of desktop variants provide Btrfs-based images
- VMs created through libvirt have *nodatacow* set on the VM disk images
  - Avoids painful “double-CoW” scenario that impacts performance
- No compression currently
  - Pending discussion with Anaconda developers and tweaks to image build tools
- /boot is not on Btrfs currently by default
  - Pending discussion with bootloader team
  - Installation *is* possible with /boot as Btrfs subvolume or separate Btrfs volume
- Disk encryption uses LUKS
  - LUKS with Btrfs means only full disk encryption is possible

Future plans (Fedora 34/35?)

- Zstd compression by default
- /boot on Btrfs by default
- Online/Live full or partial disk encryption using Btrfs native encryption
    - Pending upstream work
    - Will require moving /boot to Btrfs
- Full support for Btrfs for osbuild
    - Initial work has been done, subvolume creation is missing
- Simpler setup for full system snapshotting and boot-to-snapshot
    - Pending coordination with bootloader team and snapper developers

Demonstration

Btrfs with system snapshots

Demo Setup

- Dusty’s setup he’s been using for years
  - Simple system with a single filesystem (/)
  - Set up btrfs snapshots to be taken on each package update
  - Demonstrate rolling back to previously taken snapshot
  - More information at:
    - https://dustymabe.com/2019/12/29/fedora-btrfs-snapper-the-fedora-31-edition/
- Caveats
  - Lumps /boot into the root filesystem (doesn’t handle UEFI)
  - The state of the art might be better today. I implemented this system years ago

Questions?

Resources

- Btrfs wiki: https://btrfs.wiki.kernel.org/
- openSUSE Btrfs documentation: https://en.opensuse.org/SDB:BTRFS
- Snapper website: http://snapper.io/
- Btrfs + Snapper (F31 Edition): https://dustymabe.com/2019/12/29/fedora-btrfs-snapper-the-fedora-31-edition/