Linux Filesystem Comparisons

Jerry Feldman

Boston Linux and Unix

Presentation prepared in LibreOffice Impress

Boston Linux and Unix 12/17/2014

Background

- My Background. I've worked as a computer programmer/software engineer almost continually since 1972. I have used the Unix Operating System since about 1980, and Linux since about 1993. While I am not an expert in filesystems, I do have some knowledge and experience.
- I have been a member and director of the BLU since its founding in 1994.

Overview

- My talk is intended to focus primarily on the end user. Servers and NA systems have different requirements, so what might be good for you may not be appropriate for a high-traffic NAS system.
- I will try to explain some of the file system issues that you should be aware of when you decide to install Linux.
- While I will mention distributions, all Linux distributions allow you to use just about any file system you want.
- **Please feel free to ask questions at any time.**

Terminology

- RAID:
  redundant array of independent disks
  RAID is used to incorporate stripe sets, and to add redundancy. In general RAID 0 is stripe, and RAID 1 is mirroring. There are other RAID levels that I won't discuss here
- STRIPE – Data is spanned among multiple volumes or devices.
- Mirroring – Data is written to 2 or more volumes

Terminology

- LVM – Logical Volume Manager. This has been around for a while. It allows you to resize and stripe volumes. Recent versions also incorporate RAID
- Journaling. This is a feature of a file system where changes are stored locally in metadata so that in event of a system crash, the data can be restored rapidly.

The File Systems I will talk about

File System	Short Description
EXT2	The venerable Second Extended File System. This was the standard for several years
EXT3	EXT3 simply adds journaling to ext2
EXT4	A substantial upgrade in both performance as well as support for larger files. EXT4 currently is the default for Fedora and Debian.
BTRFS	Btree File System. Adds pooling, snapshots, checksum and multiple device spanning and a few other features
XFS	XFS is a journaling system developed by SGI. It is currently the default file system for Red Hat Enterprise 7
ZFS	ZFS is a tree-based file system developed by Sun (Oracle) for use on Solaris. It has been available in Linux for several years
JFS	JFS is a B-Tree based journaling file system developed by IBM for AIX and OS2.
ReiserFS	I am inclusing ReiserFS for historical purposes. This is a B-Tree based system that was the standard for SuSE for several years

Why the end user should be interested

- You normally keep a lot of personal data on your computer, and you want to feel that the data is reasonably secure.
  - The file system is what reads and writes your data
  - You want to make sure when you save a file it will be where you placed it
  - And its integrity is not compromised

Linux vs Windows

- Linux file systems are very different from Windows file systems
  - In Linux and Unix, when you write to a file, the operating system buffers some of the data.
  - In Windows, every time you write, the file is flushed to disk.
- This difference gives Linux a performance advantage. It also keeps Linux files mostly unfragmented.

Linux vs. Windows

- In Linux and Unix a file name is not an integral part of the file system
  - You can have multiple hard links (or names) for a single file. For instance an empty directory has 2 names, the name you named it, and “.”. A symbolic link is similar to a Windows shortcut.
  - Deleting a Linux file using the rm command may not physically remove the file

Linux vs. Windows

- Only the operating system actually removes a physical file.
  - When you issue the rm command on a file, all that does is bump the use count down by 1. When the use count gets to 0, the OS removes the file.
  - When a program opens a file, it bumps the use count up by one.

Linux vs. Windows

- Let's say you do an update, and libc.so is one of the files. But it is in use by most programs that are running, so you don't want to physically remove it. Since there will be multiple shared instances when you download libc.so, you create a new file, named libc.so. The older version of libc.so does not get removed until all using programs exit. Nothing crashes.

Linux vs. Windows

- Since the file name in Windows is an integral part of the file system, when you download a file, it physically overwrites the existing file
  - So, let's say that Windows has an important DLL called libc.dll. (A DLL is very similar to the Linux .so). Overwriting a DLL that is in use will crash all the users of that DLL.
- This is the main reason you have to reboot Windows every time you do an update

The EXT family

- The original Linux file system was the minix file system.
- EXT was written by Remy Card in 1992 and was loosely based on UFS (Unix File System). It had a 2GB file size max.
- EXT2 replaced EXT in 1993.
- EXT3 emerged in 2001
- EXT4 was available in 2006

Rémy Card

The venerable EXT2

- The file size is dependent on the block size. A 1KB block size allows a 16GB file size
  - By increasing the block size you can have files up to 2TB
- Directories are not indexed
- File compression can be added to the file system
- Files and directories are stored in the Inodes.
- Time stamps are 32-bits which mean they go negative 1/18/2038

The venerable EXT2

- The inode is essentially the root of the file system and contains the following information:

Inode #	Access Control List	Extended Attribute
Direct/Indirect Disk Blocks	Time stamp(3) Access,change, modification	Number of blocks
File Deletion Time	File generation number	File Size
File Type	Group	Number of links
Owner	Permissions	Status flags

EXT3 adds journaling to EXT2

- Stephen Tweedie introduced EXT3 in 1998 and was included in 2.4.15 kernel in 2001.
- EXT3 is an in-place upgrade to EXT2.
- This is essentially still an EXT2 file system.
- The big benefit is that a journal is maintained in metadata for subsequent FSCKs

EXT3 adds journaling to EXT2

- There are 3 types of journaling supported:
  - Journal (Writeback)
  - Ordered (default)
  - Writeback(data)
- Please look in references [13] for the link to journaling file systems.

EXT4 – major upgrade to EXT2/3

- EXT4 was originally part of the Cluster File System
- Ted Tso brought EXT4 into the 2.6.28 kernel for development in 2008.
- Google used ext4 for its storage systems in 2010, and enabled it on Android later that year.

EXT4 – New features

- Large file systems:
  - Volumes up to 1 exbibyte (EiB)
  - File sizes up to 16 tebibytes (TiB)
  - Note that Red Hat recommends using XFS for volumes larger than 100TB. (TB == terabytes)
  - System uses 48 bit addressing
  - Time stamp has a date limitation of April 25, 2514.
  - Online defragmentation

EXT4 – New features

- Additional features
  - Inode is 256KB where EXT3 was 128KB
  - Persistent preallocation. Extra space is reserved.
  - Backwards compatibility. An EXT2 or EXT3 volume can be mounted as an EXT4 volume. This will slightly improve performance.
  - Extent-based storage

EXT4 – Performance Features

- EXT4 has a number of performance-related features
  - Multi-block allocation where multiple blocks can be written at once
  - Delayed Allocation – This allows multiple block writes, reduced fragmentation, reduced processor time for files that are short-term or temporary.
- EXT4 remains the default for Fedora 21 and recent Debian systems

BTRFS – The Linux file system of the future

- 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. Jointly developed at multiple companies, Btrfs is licensed under the GPL and open for contribution from anyone.[1]
- BTRFS was initially developed by Chris Mason who had experience working on the Reiser File System [7]
- Currently default in OpenSuse 13.2.
- Will be the default in Fedora 23

BTRFS - features

- BTRFS has the following features
  - Extent-based storage
  - 16 Exibyte maximum file size (2^64)
  - Space efficient packing for small files.
  - Space efficient indexed directories
  - Dynamic inode allocation
  - Writeable and read-only snapshots
  - Subvolumes

BTRFS - features

- Checksums on data and metadata
- Compression
- Integrated multiple device support
  - Striping
  - Mirroring
  - Striping + mirroring
  - Striping with single and dual parity
- SSD awareness
- Efficient incremental backup

BTRFS - features

- Conversion of EXT3 or EXT4 file systems
- Seed devices – essentially a template
- Subvolume aware quota support
- Send/receive subvolume changes
  - Efficient incremental filesystem mirroring
- Batch deduplication (happens after writing)

BTRFS – my observation

- I think that BTRFS is ready for prime time, but just barely.
  - For home personal use
  - Not for heavy production servers yet
- There are several other non-native file systems (XFS and ZFS) that are probably better for server use today
- But, BTRFS will become the desired Linux file system over the next year or two. It has better performance than a RAID1 EXT4 file system.
- I will upgrade my home system shortly using the EXT4 conversion

BTRFS – a couple of issues

- The GNU utilities DF and DU don't work on BTRFS
  - BTRFS has its own df and du utilities.
  - So, in general, it is impossible to give an accurate estimate of the amount of free space on any btrfs filesystem. Yes, this sucks. If you have a really good idea for how to make it simple for users to understand how much space they've got left, please do let us know, but also please be aware that the finest minds in btrfs development have been thinking about this problem for at least a couple of years, and we haven't found a simple solution yet.[11]

More on BTRFS File System Full

- Read Marc Merlin's blog [12]. I found lots of good data here.
- Preventing a btrfs Nightmare | LAS 320 [13]

ZFS – Commercial File system

- Developed by Sun (now Oracle)
- Tree-based file system
- Designed to focus on data integrity
- Uses storage pools for allocation
- Features
  - Max volume 256 zebibytes
  - Max file 16 exbibytes
  - 2^{48} max files

ZFS on Linux

- ZFS appears to be the favorite file system for some NAS implementations. It is mature, and optimized for data integrity
- Probably will never be directly supported by the major distributions
- Lots of comparisons to BTRFS. Today, I would say that ZFS wins.
- Marc Merlin [10] Why you should consider using BTRFS. Real COW snapshots and file level incremental server OS upgrades

ReiserFS – A look back

- ReiserFS is a B-Tree based file system with journaling. It was the default on SuSE Linux systems until October 2006 when SuSE adopted EXT3.
- Development essentially stopped when Hans Reiser went to prison for killing his wife

ReiserFS features

- Directory contents are B+ Tree
- File allocation is bitmap
- Max volume 16TiB
- Max file size 1 EiB (64-bit)
- Max files 2^{34}
- Supports efficient small file allocation

References

In researching this I have used many references. The following pages are a list of most of the references that provide the best technical details or that I reference or cite directly.

References

1. https://btrfs.wiki.kernel.org/index.php/Main_Page
2. http://en.wikipedia.org/wiki/Ext4
3. http://en.wikipedia.org/wiki/Ext3
4. http://en.wikipedia.org/wiki/Ext2
5. http://www.linux.org/threads/ext-file-system.4365/
6. http://www.linux.org/threads/trees-b-trees-b-trees-and-h-trees.4278/

References

7. http://lwn.net/Articles/342892/
8. https://rudd-o.com/linux-and-free-software/ways-in-which-zfs-is-better-than-btrfs
9. https://btrfs.wiki.kernel.org/index.php/FAQ
10. http://events.linuxfoundation.org/sites/events/files/slides/Btrfs_1.pdf
11. https://btrfs.wiki.kernel.org/index.php/FAQ#Why_is_free_space_so_complicated.3F

References

12. http://www.jupiterbroadcasting.com/61572/preventing-a-btrfs-nightmare-las-320/
13. http://www.linux.org/threads/journal-file-system.4136/