Filesystems

Sumner Evans and Sam Sartor

December 22, 2017

Mines Linux Users Group

Introduction

What are Filesystems?

- Filesystems manage the storage and retrieval of files from storage media.
- Filesystems are an abstraction layer between storage media (SSDs, HDDs, disk drives, even tape drives).
- Filesystems exist on *partitions*, physically contiguous segments of the disk.

Filesystems are Responsible for...

- **Space management:** filesystems allocate and manage space in discrete chunks. Filesystems must keep track of what data is stored at each chunk.
- **Filenames:** identify a storage location in the file system. Can be case sensitive (ext4) or case insensitive (HFS, NTFS).
- **Directories (folders):** group files into separate collections. Modern filesystems allow arbitrary nesting of directories.
- **Metadata:** filesystems store book-keeping information about their contents (e.g. file sizes, last accessed date, owner and permissions, etc.).
- **Access Control:** prevent unauthorized access to files on disk.
- **Data Integrity:** filesystems must be resilient to failure, some are better at this than others.

A Few Ancient Filesystems

The First Filesystems

The filesystem was originally thought of as part of the operating system. One of the first filesystems that had a name was DECTape. DECTape stored 184 kilobytes (kilo, not mega) of data per tape on the PDP-8.

CP/M $\rightarrow$ FAT

Gary Kildall invented CP/M, an OS with a portable filesystem.

Kildall did not port his system to 16-bits, and Tim Patterson created a clone of CP/M for his new operating system: QDOS. Microsoft bought QDOS from Patterson, and the filesystem was given the name FAT. This first version of FAT was called FAT-12 because it used a 12-bit number to count the clusters.

$2^{12} = 4096$ and given that each cluster could be up to 8KB, so FAT-12 volumes had a maximum size of 32MB.

FAT-12 was superseded by FAT-16 and FAT-32. One limitation was that file paths could only be 255 characters long.

Current Filesystems

Linux

ext4

`ext4` is the successor to `ext3` and is the default filesystem for many Linux distributions.

Pros

- It's everywhere. Everyone and their cat use it.
- It can support volumes with sizes up to 1 exbibyte (EiB) and files with sizes up to 16 tebibytes (TiB).
- Backwards compatible with `ext2` and `ext3`.

Cons

- It does not honor the "secure deletion" file attribute, which is supposed to cause overwriting of files upon deletion
- It's boring. It's not new and fancy. It's the tried and true file system.

Windows & macOS

NTFS

New Technology File System (**NTFS**) is the crap that Windows runs on.

Pros

- It works on Windows (not sure if this is a pro or not...)

Cons

- Filenames are not case sensitive and are limited to 255 UTF-8 code units
- Can resize (post Windows Vista) but cannot move sectors around to do so
- File paths are limited 32,000 characters
- Much less resilient to failure than other filesystems.

HFS and HFS+

Apple has made a ton of filesystems with varying degrees of terribleness.

- **HFS:** Hierarchical File System — Introduced in 1985 with the first Apple computer with a hard drive. Had a limitation of 65,535 files and every file had to take up at least 1 / 65,535th of the disk.
- **HFS+:** Released in 1998 to fix some of the issues with HFS. the core of the filesystem uses case-insensitive NFD Unicode strings, which led Linus Torvalds to say that “HFS+ is probably the worst file-system ever”.

APFS

APFS: Apple Filesystem — Introduced in June 2016 to replace HFS+ and is optimized for SSDs. It fixes some of the problems of HFS+. Basically it replicates the work of other modern filesystems which are actually maintained by large communities.

Apple forcibly upgraded all computers to APFS in macOS High Sierra. So sorry if your data was corrupted. Let me introduce you to an operating system where you have a choice to change you filesystem or not: Linux.

Flashdrives

FAT32

`FAT` is an ancient family of filesystems, going back to floppy disks in 1977. `FAT32` allows for largish files (up to 4GiB) and storage devices. It is mainly used on portable storage devices such as flash drives.

Pros

- Works anywhere
- Works on anything

Cons

- Annoying size limitations
- Very inflexible

exFAT

`exFAT` is the latest variation of the `FAT` family. Although it is far from being fully featured, it is a significant improvement on `FAT32`.

Pros

- Newer
- Supports larger files size
- Supports larger storage devices

Cons

- No journaling
- Not as ubiquitus

Alternative Filesystems

Btrfs

B-tree **f**ile **s**ystem (Btrfs) pronounced “Butter FS” or “better FS” or “b-tree FS” was developed starting in 2007 by Oracle.

Pros

- Copy on Write
- Mostly self-healing
- Can convert from `ext*` to Btrfs

Cons

- It’s being deprecated by Oracle. RHEL 7.4 includes it, but they are transitioning away. The SUSE project will still use and maintain it.

ZFS

ZFS is file system designed by Sun Microsystems for long-term, performant, and reliable data storage.

Pros

- Corruption detection and self-healing
- Integrates well with RAID
- Builtin shapshotting and rollback
- Highly configurable

Cons

- Not for day-to-day use
- Main fork is closed source

TFS

`TFS` is a work-in-progress filesystem for the Redox operating system. Intended as a modern alternative to ZFS, the feature list is mouth-watering.

Pros

- Concurrent & non-blocking
- Lightweight full-disk compression
- Zero-overhead revision history
- Automatic corruption detection
- $O(1)$ recursive directory copies
- Designed for solid state drives
- Perfectly resilient to sudden power loss

Cons

- Not implemented yet

Network Filesystems

What is a network filesystem?

You can access remote storage devices over the internet using a *network filesystem*.

NFS

`NFS` is a common network filesystem protocol for *Nix.

Pros

- Fast
- Mature
- Cross platform

Cons

- Requires setup on both host and client
- Server is (relatively) complex to setup

Samba

`Samba` includes a network filesystem that interpolates between windows network drives and *Nix systems.

Pros

- Talk to Windows networks without having to use them

Cons

- Windows

Virtual Filesystems

What is a virtual filesystem?

A *virtual filesystem* is an abstraction layer that takes in some other source of data and represents it as a mountable structure of files and directories.

Colloquially, any filesystem that is associated with a physical disk is called a virtual filesystem. These might still be backed by other kind of storage, or they might be purely procedural.

tmpfs

`tmpfs` is a filesystem stored in RAM. It appears as a mounted filesystem, but all data is stored in volatile memory. By default, the `/tmp` directory is a `tmpfs`.

Pros

- Useful for storing temporary files such as downloads and program files
- Saves unnecessary disk I/O
- Safe storage for decrypted data, since files are never written to disk

Cons

- Everything in a `tmpfs` will be “deleted” on reboot.

procfs

In *Nix, everything is a file. This includes things such as process information. The way to access this data is through the `proc` filesystem which provides a convenient and standardized method for dynamically accessing process data held in the kernel.

In Linux, `procfs` contains more than just process information including memory information, network utilization statistics, etc.

FUSE

Filesystem in Userspace (FUSE) is an interface for creating filesystems without writing any kernel-level code, which makes it incredibly useful for creating virtual filesystems. It is an available in Linux, FreeBSD, OpenBSD, NetBSD, OpenSolaris, Minix 3, Android, and macOS.

`libfuse`...

- is a C library
- provides a high-level interface for FUSE
- makes creating new filesystems really easy
- has bindings for Python, Rust, etc

sshfs

`sshfs` is a network filesystem implemented through libfuse. You can use it to mount remote directories via ssh.

Pros

- Very easy & quick to setup (one command)
- Remote machine only needs ssh installed

Cons

- Intended to be temporary
- Generally slower than NFS

Questions?

References

- https://en.wikipedia.org/wiki/File_system
- http://www.tldp.org/LDP/sag/html/filesystems.html
- https://arstechnica.com/gadgets/2008/03/past-present-future-file-systems/2/

Copyright Notice

Colorado School of Mines
Linux Users Group