

Linux kernel advances

An introduction to what's new in versions 2.6.28 and 2.6.29

Skill Level: Intermediate

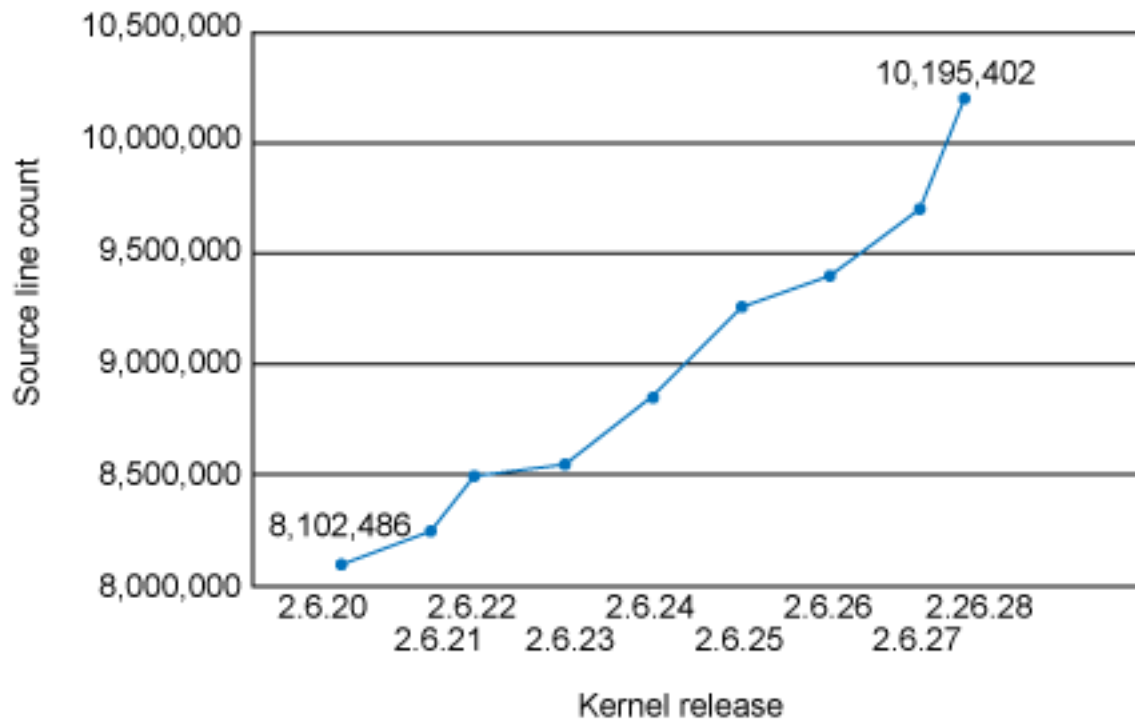
M. Tim Jones (mtj@mtjones.com)
Independent Author

24 Mar 2009

Life's certainties include death and taxes but also the advancement of the GNU/Linux® operating system, and the last two kernel releases did not disappoint. The 2.6.28 and 2.6.29 releases contain an amazing amount of new functionality, such as a cutting-edge enterprise storage protocol, two new file systems, WiMAX broadband networking support, and storage integrity checking. Discover why it's time to upgrade.

At the end of 2008, the 2.6.28 kernel surfaced. Subsequently, the merge window for the next release—2.6.29—opened. As the Linux kernel uses a distributed development process, it's not always clear what's coming (or will be integrated) into a given kernel release, but the last two have been interesting from both a short- and long-term perspective. One interesting milestone for 2.6.28 is that it's the first time Linux has exceeded 10 million lines of source code (see Figure 1, which uses source line count data from Heise Online).

Figure 1. Lines of source code in recent 2.6 kernels



Read more by Tim Jones on developerWorks

- [Tim's Anatomy of... articles](#)
- [All of Tim's articles on developerWorks](#)

These releases have introduced new file systems (one stable, one experimental), new support for graphics and virtualization, and new enterprise storage features. I start with a review of some of the major new features in 2.6.28, then I explore what you might expect from 2.6.29.

What's new in 2.6.28?

Linux kernel 2.6.28 was released on December 24, 2008 (at release 5 as of early February, 2009). This first release of 2.6.28 includes a large number of changes—so large that the change-log text file is itself almost 6MB in size. This release is viewed as so stable that it's the kernel of the next Ubuntu distribution, version 9.04, Jaunty Jackalope.

The fourth extended file system

The fourth extended file system (ext4) file system was renamed from *ext4dev* to *ext4*, which means that it's stable enough for regular use. Ext4 is the successor to the standard third extended file system (ext3) available today, but with better performance, features, and reliability. Ext4 permits exabyte file systems that can

support larger numbers of files, larger files, and deeper directory structures. It also includes extents with multi-block and delayed block allocation for performance. Ext4 is both forward and backward compatible (meaning that you can mount an ext4 file system on an ext3 disk format and vice versa, depending upon the features used). You can also gradually migrate a file system from ext3 to ext4 online with a mass change. For links to more information about the ext4 file system, see [Resources](#).

And although ext4 will be the new standard Linux file system for some time to come, other file systems are coming that offer even better scalability and features. One such file system, Btrfs, is available in an experimental form in the 2.6.29 kernel. Btrfs is a Linux-compatible file system (read GNU Public License [GPL]) that competes in features with the well-known ZFS.

Graphics Execution Manager memory management

One of the areas that has seen solid improvements over the past year is the Linux graphics stack. Not surprisingly, it's also an area where graphics processor units (GPUs) provide useful assists for rendering. In many cases, GPUs are more powerful than the central processing units (CPUs) they assist.

To support the GPUs of today and tomorrow, one area of the Linux graphics stack that needed improvement was memory management, including buffer management, page mapping, placement, and caching. This was necessary because graphics applications—particularly three-dimensional applications—can consume a vast amount of memory. The Graphics Execution Manager (GEM) helps here by providing ways to manage graphics data that blends into the kernel using the existing kernel subsystems (such as using the shared memory file system, or shmf, to manage graphic objects).

Boot tracer

Although the time required to boot Linux has shrunk over time, expectations are still that it takes too long. For that reason, boot times remain under scrutiny. This kernel includes a new feature to measure and record the timings of `init` calls. The timings can be used later to visualize the flow and performance of the boot process. This process is configurable (it requires enabling to collect the data), but once collected, the data can be analyzed using offline scripts (including graphical depictions), which will ultimately lead to better boot times and a more optimized boot process. This update incorporates the process identifier (PID) of the calling thread so that the parallelism of the boot process can be viewed.

Freezer

Based conceptually on the idea of suspending an operating system for the purpose of migrating it to a new host (for example, virtual machine, or VM, migration), a new capability called *freezing* (and *thawing*) has been committed. This new feature

allows either a group of tasks or a file system to be frozen and kept in its freeze-time state, later to be thawed to reintroduce the task group or file system.

You freeze tasks in the context of a *container*, which is a scheme that virtualizes operating systems at the user-space level (a single kernel supports multiple user spaces). This new functionality is a step in the direction of migrating a set of processes between hosts, which can be very useful for load balancing. You can also freeze file systems to support snapshots for file system backup. Currently, file system freezing is achieved through an `ioctl` with an argument of `FIFREEZE` or `FITHAW`.

Outside of containers, this new freeze/thaw scheme can find uses in checkpointing. In this application, you could freeze a collection of related processes at specific intervals (*checkpoints*), then thaw a particular epoch as a way to roll back to a known good state.

Improved virtual memory scalability

As Linux finds increasing use in virtualized systems—particularly those with many processors and vast amounts of memory—the ability to scale memory usage becomes critical to performance. Kernel 2.6.28 includes a number of scalability enhancements related to memory. For example, this kernel maintains separate Least Recently Used (LRU) lists, one for pages backed by files and another for pages backed by swap. This allows the kernel to focus on swap-backed pages, which are more likely to be written to disk, and pay less attention to file-backed pages.

Another change separates the evictable pages from the unevictable pages (such as those that were locked through `mlock`). In this way, the pageout code does not need to iterate unevictable pages in the LRU list, leading to improved performance in systems with very large numbers of pages.

Disk improvements

The 2.6.28 kernel includes a number of advancements for disks. In particular, improved use of solid state drives (SSDs) and protection for ATA devices are detailed in this section.

Improvements for SSD support

SSDs are a fantastic way to improve disk performance over traditional drives with rotating media. SSDs provide reduced latency and better random reads with less power and noise. But SSDs are fundamentally different from hard disk drives (HDDs), so changes to Linux are necessary to exploit them. A fundamental problem with SSDs is that they must wear-level the blocks in their internal flash, and

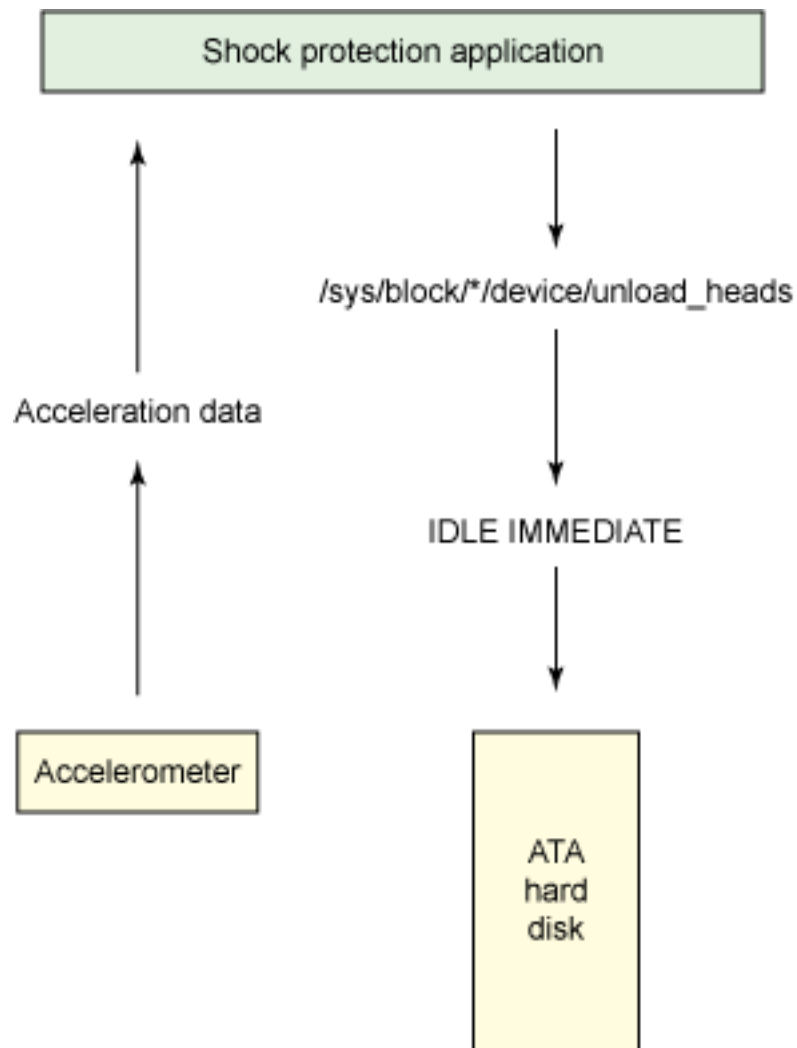
wear-leveling can reduce the lifetime of the SSD. One way to reduce this effect is by telling the SSD when a block is no longer valid (perhaps because of a file deletion). When a block is freed in the SSD, it is no longer a target of wear-leveling and therefore can minimize the overall wear on the device. But to support this operation, the file system must be able to communicate this information to the drive.

The T13 standards committee has created a new ATA protocol extension to support this communication in the context of the `trim` command. When the SSD receives this new command, it can add the blocks defined to its free list and no longer worry about moving them around as part of wear-leveling. The Linux block layer component of this is called a *discard request*, which results from a higher-level file system file deletion. A discard request is part of the block layer and provides the means to discard blocks, eventually resulting in a `trim` command to a supporting ATA drive. Further enhancements include scheduling discard requests intelligently so that they can be coalesced with other requests as well as performing reorder operations on the block request queue to better deal with related discard and write requests.

Improvements for ATA HDDs

One very interesting reliability improvement for ATA HDDs is protection against shock, sudden jolts that can destroy HDDs in laptop computers. The ATAPI specification defines a command called `IDLE IMMEDIATE` that idles the HDD and retracts the disk heads to prevent them from hitting the platters. The kernel makes this command available through a proc file called `/sys/block/*/device/unload_heads`. After a value is written to this file and the subsequent ATA command is written to the respective disk, the heads are retracted and all input/output (I/O) is deferred for a small amount of time based on a timer. When the timer expires, I/O resumes normally. This feature was implemented previously for IBM laptops using access to an accelerometer to measure acceleration (to determine whether the laptop is falling). The generic process is shown in Figure 2.

Figure 2. Generic process of ATA hard disk protection



What's next for 2.6.29?

Where 2.6.28 includes a number of useful new features, 2.6.29 improves even further.

Btrfs

One of the most important advances in 2.6.29 is the addition of the Btrfs (or *B-tree FS*), originally developed by Oracle. The Btrfs was developed as a response to Sun's ZFS, used to build a massively scalable file system with on-disk integrity assurance. In addition to many of the advanced features of ext4, Btrfs supports object-level mirroring and striping, copy-on-write functionality, snapshots (and snapshots of snapshots), integrity, and internal compression. It supports volumes and files up to 16 exabytes in size and up to 2^{64} files in a given volume. To aid in conversion, you can upgrade a file system from ext3 to Btrfs (and also back, but only

to the point at which conversion was done).

Btrfs could define a new standard for massive enterprise file systems that include fault tolerance with online repair and simplified administration. Although still experimental (and not suitable for anything other than review), Btrfs includes a feature set that makes it ideal for future scalable Linux storage.

Networking changes

In addition to a large number of updates to various networking drivers, several enhancements are worth noting. The first is the Generic Receive Offload infrastructure, which offloads network sends (similar to Large Receive Offload, but protocol independent).

You'll also find support for *backward congestion notification* (BCN), which improves congestion management by actively tightening the control loop through messaging. This feature is a staple in new data center Ethernet deployments.

Last but not least, 2.6.29 introduces a WiMAX wireless broadband networking stack into Linux (currently based on the i2400m USB driver). Note that this is different from so-called Wi-Fi, as WiMAX uses licensed spectrum for point-to-point connection and offers improved quality of service.

Kernel mode setting

An interesting cosmetic change to the kernel boot process and graphics mode is called the *kernel-based mode setting* (KMS). This feature allows the kernel to control the graphics hardware after the required components are initialized (such as the PCI bus and graphics card). In this way, the kernel can enter the desired screen resolution much earlier in the boot process, which reduces some issues with flicker and display absence (because of a graphics chip reset) and allows the display to be properly set up before the X server is started.

The other advantage to KMS is improved re-initialization after suspend: because the initialization is done in the kernel, it's much more efficient. Finally, because the kernel manages the graphics chip, the X server may not be required to run with root privileges, hardening the operating system by removing another set of potential exploits.

Going further

At the time of this writing, 2.6.29 is in the stabilization process. But work continues, looking forward to 2.6.30. Other things that you'll find in 2.6.29 include Squashfs (a read-only file system) finally making its way to the mainline kernel, as well as the inclusion of a set of security hooks for path name-based mandatory access control.

A new protocol, Fibre Channel over Ethernet (FCoE), has also found its way into the kernel, in addition to on-disk integrity checking using the standard Data Integrity Field (T10-DIF), which is understood by certain drives for end-to-end and at-rest integrity checking.

So, as Linux moves forward, we find new functionality, improved scalability, and increasing security. Like single-malt scotch, Linux continues to improve with age. For links to more resources on additional kernel changes and what's ahead, check out the [Resources](#) section.

Resources

Learn

- In "[Anatomy of ext4](#)" (developerWorks, February 2009), learn more about the fourth extended file system's advanced features. Ext4 will emerge as the new standard for desktop Linux systems. For very large-scale storage systems that require both scalability and efficiency, Btrfs could be the future. You can learn more about Btrfs from the [Btrfs Wiki](#).
- Learn more about GEM (and the older TTM) from [Memory management for graphics processors](#) at [lwn.net](#). As graphics processors evolve and place new demands on their systems, new ways of managing them become necessary. In the past, translation table maps (TTM) were used for general-purpose memory management, but now GEM can be used for improved memory management performance. You can GEM makes use of the shared memory subsystem for object management. To learn more about shared memory (/dev/shm) and its use, check out this [practical tutorial on /dev/shm](#).
- This [new kernel tracing option](#) provides the means to measure and improve Linux boot times. It seems like no matter how fast Linux boots, someone will complain that it's too slow. Luckily, someone else will find ways to improve it. Measuring where the boot process is spending its time is the first step to optimizing it.
- Learn more about the two sides of wear-leveling in "[Anatomy of Linux flash file systems](#)" (developerWorks, May 2008). Wear-leveling is an important aspect of flash use in SSDs. A key requirement of wear-leveling is the ability to discard blocks that are no longer relevant on the disk (for example, because of deletion). ATA provides a new disk-level command called `trim` to discard on the SSD, which is supported in Linux through [block-layer discard requests](#).
- The ability to [freeze containers and file systems](#), which could support process group checkpointing, is one of the most interesting new features in 2.6.28 (and the upcoming 2.6.29). Also new in 2.6.28 is the ability to pass PCI devices directly through to guest operating systems rather than have them managed by the hypervisor.
- Read "[LXC: Linux container tools](#)" (developerWorks, February 2009) for an overview of Linux containers. Also check out "[Secure Linux containers cookbook](#)" (developerWorks, February 2009) for ways to tighten container security using SELinux.
- Read "[Knock-based commands for your Linux laptop](#)" (developerWorks, July 2006), "[Shake some sense into your Linux ThinkPad](#)" (developerWorks, November 2006), and "[ThinkPad aerobics: Rotate and shake your laptop to control applications](#)" (developerWorks, March 2008) for some creative ways to use a laptop's accelerometers.

- [WiMAX](#), the next-generation broadband wireless technology, is coming to the Linux kernel (2.6.29). This technology will consist of user-space and kernel components based on the Institute of Electrical and Electronics Engineers (IEEE) 802.16e set of standards.
- Read about Generic Receive Offload (GRO) in the [commit log](#) or from [lwn.net](#). GRO is a generalization of the Large Send Offload feature found in many IP networking stacks.
- This [open source article from Oracle](#) documents the introduction of DIF into the Linux kernel, a multi-vendor solution led by Oracle. The T10-DIF is an enterprise-class data integrity feature that's now available in Linux. DIF protects data on the disk and solves the problem of silent data corruption by adding integrity information to each block on the disk.
- In the [developerWorks Linux zone](#), find more resources for Linux developers, and scan our [most popular articles and tutorials](#).
- See all [Linux tips](#) and [Linux tutorials](#) on developerWorks.
- Stay current with [developerWorks technical events and Webcasts](#).

Get products and technologies

- [Order the SEK for Linux](#), a two-DVD set containing the latest IBM trial software for Linux from DB2®, Lotus®, Rational®, Tivoli®, and WebSphere®.
- With [IBM trial software](#), available for download directly from developerWorks, build your next development project on Linux.

Discuss

- Get involved in the [developerWorks community](#) through blogs, forums, podcasts, and spaces.

About the author

M. Tim Jones

M. Tim Jones is an embedded firmware architect and the author of *Artificial Intelligence: A Systems Approach*, *GNU/Linux Application Programming* (now in its second edition), *AI Application Programming* (in its second edition), and *BSD Sockets Programming from a Multilanguage Perspective*. His engineering background ranges from the development of kernels for geosynchronous spacecraft to embedded systems architecture and networking protocols development. Tim is a Senior Architect for Emulex Corp. in Longmont, Colorado.

Trademarks

IBM, the IBM logo, ibm.com, DB2, developerWorks, Lotus, Rational, Tivoli, and WebSphere are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. See the current list of [IBM trademarks](#).

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.