Philip Gales s62j794
Brady Hatton g12m253
James Jacobs d44x441

CSCI 460
Operating Systems

A History Of Linux

History in the making

In early 1991 Linus Torvalds, a University of Helsinki student, set out to buy the best computer he could afford. Unfortunately, he was a broke college student with minimal savings. Undeterred, he took out a student loan and purchased a new computer for $3,500 with sobering specs—an Intel 386 processor, 4 MB of RAM, and a 40 MB hard disc. What excited him most was the Intel 386 processor.

His excitement turned to disappointment when he learned that the DOS operating system that the computer came with did not take full advantage of the 386. Linus regarded the processor as “a lot better than any of the previous chips” and was determined to take full advantage of it’s features.

Instead of waiting for an update from DOS, Linus decided to look into UNIX. Thankfully for us, Linus couldn’t find a copy of UNIX for less than $5,000—almost 50% times more expensive than his computer. As a university student, Linus had access to a free copy of MINIX—a UNIX-like operating system created to teach UNIX to university students—and so he decided to replace his DOS operating system with MINIX since it took advantage of some of the features present in the 386 chip.

With MINIX scheduled to take a month to arrive, Linus decided to pass the time by playing around with his new processor. He did this by writing a task switching

program. The program would switch between two processes that printed “AAAA” or “BBBB” and had a timer that switched tasks. Although a seemingly simple program, he was extremely proud of his work.

Finally, all 16 floppy discs of MINIX arrive. MINIX required manually patching and it took Linus “the better part of a week installing it”. Fairly happy with the change in operating systems, Linus decided to write some programs filling in what he thought were gaps in his new operating system.

Over the summer of 1991, Linus would post questions to the MINIX newsgroup. One question in particular, about POSIX specifications, yielded a response by Ari Lemmke, a volunteer administrator for the FTP server at Helsinki University. Ari informed Linus that POSIX wasn’t free and had to be paid for. He also told Linus “his area of interest was kernels” and that he was putting aside a directory on their FTP for him. The directory was /pub/os/linux.

In an interview with Glyn Moody, Linus addresses the name Linux and how he never wanted to name his operating system after him. Here is Linus’s exact quote “Linux was my working name, so in that sense he didn't really name it, but I never wanted to release it as Linux. Linux was a perfectly good working name, but if I actually used it as the official one people would think that I am egomaniac, and wouldn't take it seriously. So I chose this very bad name "Freax," for "Free Unix." Luckily, Ari Lemmke used this working name instead. And after that he never changed it.”.

Finally, in August of 1991, Linus announced his free operating system to the newsgroup.

> *Hello everybody out there using minix-*
>
> *I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386 (486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-sytem due to practical reasons) among other things.*
>
> *I've currently ported bash (1.08) an gcc (1.40), and things seem to work. This implies that i'll get something practical within a few months, and I'd like to know what features most people want.*
>
> *Any suggestions are welcome, but I won't promise I'll implement them :-)*

The happy accident

After announcing his new operating system, Linus was hard at work implementing the feedback he received. A lot of this worked revolved around file systems, task switching, and interrupts.

In September 1991, a month after announcing Linux, Linus released version 0.01. This version was very crude and was mostly meant for reading, although Linus would help you install it if you were interested. Some notable features of Linux at this time include no message passing which allowed Linus to not have to worry about message queues, a multithreaded file system and better scheduler so you can run several processes concurrently without the performance hit seen in MINIX, and interrupts that aren't hidden. In the release notes, Linus takes a jab at Tanenbaum, the creator of MINIX, for thinking interrupts are ugly and thus deserve to be hidden.

Finally, in October 1991, Linux version 0.02 was released. This was the first stable version as Linus points out in his email to the newsgroup. In his email Linus makes it clear the only people who will care are those who love to code. He even asks why use an operating system where everything works when you can constantly debug code in Linux.

> *Do you pine for the nice days of minix-1.1, when men were men and wrote their own device drivers? Are you without a nice project and just dying to cut your teeth on a OS you can try to modify for your needs? Are you finding it frustrating when everything works on minix? No more all-nighters to get a nifty program working? Then this post might be just for you*

Towards the end of 1991 Linus started to lose interest. He was proud of his work, but got tired of all of the debugging. Below is an excerpt of Linus talking about this period in Linux.

> *I probably would have stopped by the end of 1991. I had done a lot of things I thought were interesting. Everything didn't really work perfectly, but in a software kind of world I find that once you solve the fundamental problems of a project, it's easy to lose interest. And that's what was happening to me. Trying to debug software is not very engaging.*

Luckily for the world though, Linus makes a big mistake and accidentally deletes his MINIX partition. Up until this point, to run Linux you had to have MINIX installed since Linux wasn't a complete operating system. Linus had a tough choice to

make, reinstall MINIX or “bite the bullet and acknowledge that Linux was good enough that I didn’t need Minix”.

In January of 1992, Linus released the first version, version 0.12, of Linux that didn’t need MINIX to bootstrap the system. This change forced Linus to rewrite several key parts of his operating system.

During this transition, Linus received an email from a guy in Germany who asked for help compiling the kernel. His computer did not have enough RAM so he was wondering if he could compile the kernel with a different compiler. Although Linus didn’t need the feature, he decided to add it anyways. This feature wound up being paging. The addition of this feature not only helped the German, it helped Linux rise above its competitors. Linux was no longer compared to just MINIX.

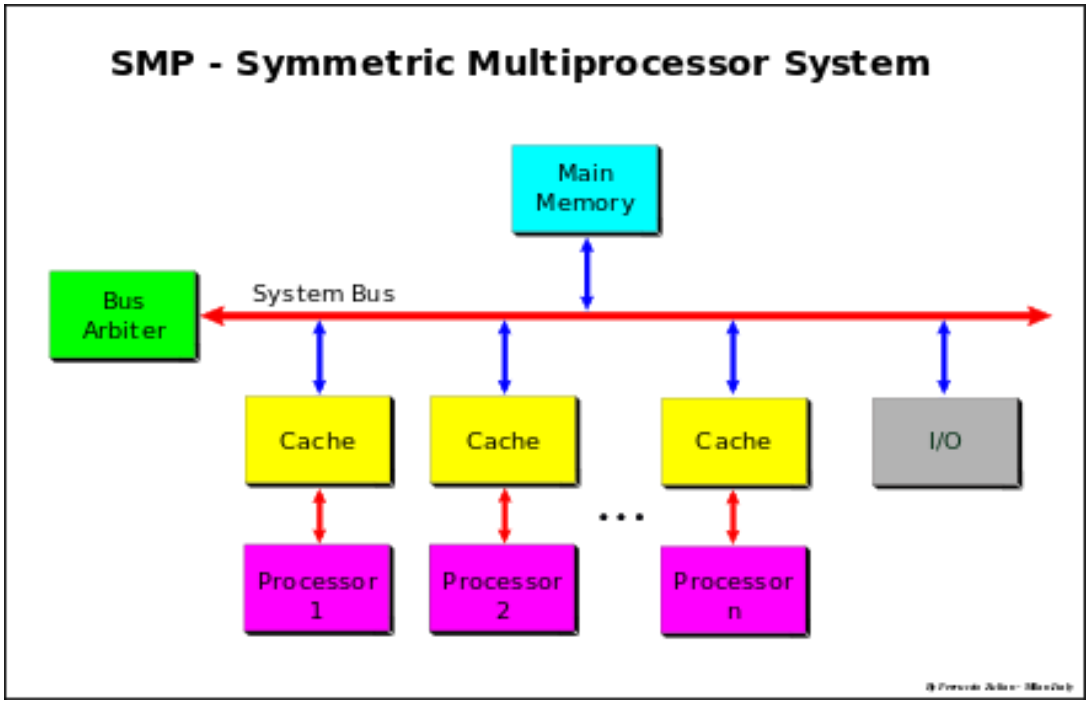
Linux branches out

In 1994 Linus decided the components of Linux to be mature enough to release version 1.0. Shortly after the release, people started distributing their own version of the Linux kernel. Popular distributions of Linux today include Red Hat, Debian, CentOS, and Fedora. Finally, what makes distributions so special is that they take the Linux kernel and package it with other open source software.

A year later, in 1995, Linus was offered an Alpha computer by DEC with the hopes of him porting Linux. At the time, “the Linux kernel isn't written to be portable to any architecture” and was thus a difficult process. After successfully porting Linux, Linus began to port the kernel to other systems. This was made possible because the features that Linux had relied on in the Intel 386 processor were beginning to become standard.

Kernel 2

The kernel 2.0 was released after linus had decided that the kernel had changed enough to warrant a major number change.



In version 2.0 symmetric multiprocessing was supported by the kernel. symmetric multiprocessing is where multiple CPUs can run at the same time running different processes. This was the basis for multithreaded processes in linux because before this was added only one CPU could be run. This was a huge change to the abilities of the os as now multi core processors could greatly increase the speed of computers. In version 2.4 the file system Ext3 was added. This file systems main advantage of the previous was that it supports journaling which keeps track of changes not yet submitted to the main file system. Today os all have some way to spread the load between multiple processors. In version 2.5 a O(1) scheduler was introduced replacing the old schedular. This scheduler was designed to schedule processes in a constant amount of time no matter how many processes were currently running. It also took into account whether or not processes were interactive or not. This means it tried to determine what processes the users would have the most input into and give it more attention when scheduling. In version 2.6 the scheduler was once again changed now it was the Completely Fair Scheduler (CFS). This was introduced in order to fix the shortcomings of the last scheduler. This scheduler is the one that is currently used by the latest linux version. Also in version 2.6 Ext4 file system was first introduced. This change allowed for many things that would future proof the file system such as the following. The new filesystem allowed for 1 EB of maximum filesystem size with a 16 TB maximum file size. This file system is still used in linux

today. Later in version 2.6 another file system was added which was Btrfs. At this point in time the Btrfs was very unstable and was mostly used for testing and development of the filesystem which would come to be updated in later versions. The main idea of this file system was to implement advanced features all while being fault tolerant, repairable and having easy administration.

Version 3

In kernel version 3, Linus decided to change the numbering scheme for linux because he could. The version 3.0 was not a major change like some other companies do. There was no major api or abi changes and no exciting new feature. However the kernel version did have some interesting os additions in various subversions which are the following.

In version 3.2 they add support for bigger block sizes to ext4. This allowed the maximum size of a file system block to be 1 MB instead of 4 KB. This was achieved not just by increasing the block size but by making the file system track clusters of blocks instead of just one. This approach was necessary because otherwise there would need to be major changes to paging and other memory management in the kerel. This change gave a major performance boost to systems that had larger files on them. Ext4 was released back in 2008 about 4 years before this update. In that time the cost per gigabyte of storage has gone down significantly and the average size of storage has raised quite a bit. Also file sizes were increasing and hard disks were getting bigger and it looked like this trend would continue. Thus this file change was released to address the current trend in storage size and file size.

In version 3.8 the f2fs (flash-friendly file system) was added by samsung. This was a file system to handle the hardware such as SSDs and sd cards. This is because most of these storage devices have a flash translation layer that emulates a block based device. F2fs tries to optimise for the flash translation layer by large-scale write gathering when lots of blocks need to be written and then they are put in large sequential writes. This is much easier on the FTL to handle. This version was provided to linux by samsung at a time when ssd were first gaining more mainstream acceptance. This filesystem was likely added to help samsung and other companies better market there ssd to a more tech savvy crowd.

In version 3.14 Kernel address space randomization was added. This was added in 2014 after google had already implemented it in their chrome books. Kernel address space randomizations allow for the random placement of the kernel in memory at the time the kernel is decompressed. This should make it harder for any attacker to get the memory location of a vulnerable function correct by brute force. This also makes it difficult for attacks in another way and that is that the kernel would crash every time the attack fails. This led to attacks directed at the kernel because if vulnerable code was found then an attacker could easily find and exploit it to get root privileges.

Version 4

Kernel version 4 was largely maintained between the 12th of April, 2015 with version 4.0 and the 23rd of December, 2018 with version 4.20 by Greg Kroah-Hartman. Sasha Levin briefly maintained version 4.1 between the 22nd of June, 2015 and the 30th of August, 2015. The early releases of 4.0 to 4.4 covered the bulk of the operating system changes. A large portion of the changes introduced throughout the development of kernel 4, after the early releases, focused more on graphics driver optimizations and memory management improvements to keep up with new graphics cards and hardware during that time span.

The change from version 3 to version 4 holds no weight regarding Linux itself, it is effectively the same kernel. Linus decided to make that change arbitrarily, even though the version could have easily remained in the 3.X naming scheme. The first big new feature to come to the 4.0 kernel is live patching. This feature allows a system to install a security update without needing to reboot. Direct access (DAX) was implemented for persistent memory storage. DAX performs reads and writes directly to the persistent memory storage device, instead of having files copied from the disk and stored to the kernel caches in RAM. Another significant feature added was a kernel address sanitizer, or KASan. This provided a fast and comprehensive solution for user-after-free and out of bounds bugs. Linux already had a feature for this, but KASan was much faster since it used compile-time instrumentation. A few other features introduced in 4.0 include Lazytime, multiple lower layers in overlayfs, support parallel NFS server, dm-crypt scalability improvements and improvements to drivers, file systems, memory management, etc. 4.1 introduces Ext4 encryption support. The user can create a key which will

encrypt both data and file names. Single user support made a return in this version. This provides the option to keep the operating system small for embedded systems that don't need multi-user support. This release implements PMEM, which is a driver that provides a range of memory as a block device and can be used by file systems. In version 4.3 there was a major change to the supported file system as Ext3 was removed from the linux core repository. This was because Ext4 had completely overtaken Ext3 and Ext3 was no longer in use. Version 4.4 adds support for open-channel SSDs with LightNVM. This allows the host to manage garbage collection, data placement and parallelism. 4.15 deals with Spectre and Meltdown, which are security problems involving vulnerabilities within processors. Meltdown breaks the isolation between user applications and the operating system. This allows a program to access the memory and data of the operating system and other applications. Spectre affects the isolation between different applications, which tricks programs into leaking their data. Retpoline is a defense used to mitigate Spectre and Page Table Isolation is a defense that covers Meltdown.

Version 5

There weren't many major changes in 5.0 but there was an interesting addition for low-end android devices i.e. mobile phones or other such devices. This was the inclusion of Adiantum file system encryption. Adiantum uses a combination of different forms of encryptions to secure devices one is the ChaCha stream cipher. This is intended to encrypt phones or other devices that lack AES instructions. Version was released in 2019 and is a response not only to the lack of support for encryption on low end android devices but also for the privacy of the user. This is because more and more personal data is stored on smartphones and the loss of what might very well be someone only electronic device could be devastating if the data was also compromised. In an age where everything is online adiantum attempts to give everyone the security they deserve.

In version 5.1 there was a very interesting time related bug that was partially addressed. In 2038 the unix 32 bit time integer will run out of bits and flip negative. This could cause all sorts of havoc as time suddenly skips back as when the 32 bit signed int flips to negative. This version reworked time keeping on the kernel to enforce 64 bit time keeping even on 32 bit systems. This problem which is being addressed dates back to the start of unix time where the problems that

would arise from using a 32 bit integer. This problem, while not actually solved by the switch to 64 bit integer will be practically solved because it gives programmers billions of years to come up with another solution.

Also in version 5.1 linux added the ability to use persistent memory as RAM. persistent memory is a type of memory that unlike ram it will not lose all data on power loss. While persistent memory may not have the speed of RAM it can be a cost- effective RAM Replacement. To facilitate this version 5.1 creates a new device driver that puts the persistent memory into the system as ordinary ram. This version shows that the history of linux to be ahead of the curve in new future features continues. Just like back in kernel version 3 where they added support for bigger block sizes in Ext4 they are now adding support for another new and emerging technology.

Conclusion and future

In the end we can look at the history of linux, the updates and the bugs to see the past progress of Operating systems in general. Linux has been around long enough that its history and changes mirrors the major revolutions in operating systems as a whole. We have seen in this history that linux has implemented many things to in an attempt keep up with what is the latest practices and new technological advancements. As we look to the future it is almost certain that linux will be a good representation of advancements in Operating systems as a whole and we should keep a close eye on its advancements.

Resources

Corbet, Jonathan. “Persistent Memory for Transient Data.” *[LWN.net]*, 21 Jan. 2019, lwn.net/Articles/777212/.

Corbet, Jonathan. “Approaching the Kernel Year-2038 End Game.” *[LWN.net]*, 11 Jan. 2019, lwn.net/Articles/776435/.

Brown, Neil. “An f2fs Teardown.” *[LWN.net]*, 10 Oct. 2012, lwn.net/Articles/518988/.

Corbet, Jonathan. “Improving ext4: Bigalloc, Inline Data, and Metadata Checksums.” *[LWN.net]*, 29 Nov. 2011, www.lwn.net/Articles/469805/.

Edge, Jake. “Adiantum: Encryption for the Low End.” *[LWN.net]*, 16 Jan. 2019, lwn.net/Articles/776721/.

“Is the ‘Page-to-Disk’ Feature Linus Talks about in His Autobiography Essentially the Concept of Swapping We Use Today?” *Unix & Linux Stack Exchange*, 1 Oct. 1969, unix.stackexchange.com/questions/604123/is-the-page-to-disk-feature-linus-talks-about-in-his-autobiography-essentially.

Edge, Jake. “Kernel Address Space Layout Randomization.” *[LWN.net]*, 9 Oct. 2013, lwn.net/Articles/569635/.

“Linux Exists Only Because of a Happy Accident.” *(August Lilleaas' Blog)*, 5 Dec. 2019, augustl.com/blog/2019/linus_and_linux_happy_accident/.

McMillan, Robert. “Torvalds Kicks Original Linux PC Into Dustbin of History.” *Wired*, Conde Nast, 3 June 2017, www.wired.com/2012/12/linux-386/.

Meltdown and Spectre, 2018, meltdownattack.com/.

Torvalds, Linus. “Open Sources: Voices from the Open Source Revolution.” *O'Reilly Media - Technology and Business Training*, O'Reilly & Associates, Inc., 29 Mar. 1999, www.oreilly.com/openbook/opensources/book/linus.html.