Next 10 Big Things in the Linux Kernel

By Aaron Grothe

Introduction

If anybody has any questions or comments at any time please let me know.

If I start to mumble please let me know as well :-)

All of this is just my opinion and I could be totally off. These are just a few of the things that I've seen on the Linux Kernel mailing list lately that have caught my fancy.

#10. 2011 Version of C Used for Kernel.

The Linux kernel has used the 1989 version of C to compile the linux kernel for years.

In February 2022 Linus decided it was time to move to the 2011 version of C to compile the Linux kernel.

This will allow the cleaning up of some code and hopefully bring in more features over time.

C is still an evolving language with C11, C17, and C2X all being developed.

#9. LSM Module Stacking

You've see the various Linux Security Modules (LSM) over the years wax and wane and thought "What can I do to make that more complicated?" Wait I know how about I stack them on top of each other?

If I can have SElinux fight it out with Smack and Apparmor as the referee that would be ideal.

#9. LSM Module Stacking.

This actually is making some progress and you can do that in a limited way now.

The ability to do this for industrial type systems might be worthwhile.

Some of the systems can be pretty complimentary

Smack and Tomoyo have some potential to work together.

Apparmor stacking is in the kernel mailing list recently.

#8. New Random Number Generator

There have been some significant changes to the Linux RNG system.

/dev/random and /dev/urandom are now the same

Quick quiz: what was the difference between /dev/random and /dev/urandom in the past?

#8. New Random Number Generator

Answer

/dev/random - is more secure, but could block if you don't have enough entropy
/dev/urandom - didn't block, but isn't as secure

If you're doing gpg keys - use /dev/random if you need randomness for a game use /dev/urandom

#8. New Random Number Generator.

With the new changes

Both devices return the same data, so no reason to pick/choose between them

Other enhancements

Has been sped up a lot

Detects it is running in a VM and if so remixes the entropy pool, so it won't be the same as another clone of the system

#7. Intel's Software Upgradeable CPU

This one is interesting almost as how quietly it is has gotten into the kernel as anything else.

The idea here is you can buy an CPU from intel and be able to unlock additional features over time. E.g. more cores, more speed, increased cache, iops, whatever

This isn't a new idea. Intel tried something similar back in the early 2000s. IBM has also done something similar with mainframes for decades.

#7. Intel's Software Upgradeable CPU

Technically this is interesting as you're figuring out how to limit a CPU with keys.

Intel's Software Defined Silicon will probably be in the Linux 5.18 kernel

The idea here is you can buy an CPU from intel and be able to unlock additional features over time. E.g. more cores, more speed, increased cache, iops, whatever

#7. Intel's Software Upgradeable CPU.

What is interesting is how little discussion there has been about this. One of the core concepts of Linux is that you own your computer, not licensed it.

#6. Firmware in the Kernel

How “free” do you want to be?

A lot of devices - e.g. wifi cards require proprietary blobs to be able to be used.
You just have to trust the binary blob from the device manufacturer and just load it onto your system.

This has been an on-going discussion for years.

OpenBSD years ago - 3.9 did attack of the Binary Blob which was their fighting against binary blobs in their distro

#6. Firmware in the Kernel.

The discussion continues. Debian goes back and forth on this all the time. Debian Gotham Needs is a pointer to the modified Debian Linux with the proprietary firmware available.

It is only a matter of time until there is a CPU that will require a binary blob to run.

There are projects such as linux-libre which ship a linux kernel without binary blobs - you can add it to a lot of systems without too much work. Interesting thing to try for an afternoon.

#5. Removing Intel x86 support

When do we remove support for the Intel x86 platform?

This one is kind of amazing as the 386 is what helped lead to the creation of Linux. So there are a lot of sentimental feelings here.

There are enough embedded systems with x86 this will probably go on for years to come. The fact it is being discussed is interesting.

#5. Removing Intel x86 support.

Once the distros start dropping the support for x86 the kernel will follow. Both debian and fedora are actively discussing this.

Keep in mind the current discussion is whether to remove 32 bit - x86 a.out support so x86 will continue

#4. C++ In the Linux Kernel

Right now the Linux Kernel is written 95+% in C with a bit of assembler to support everything else.

C has been described as all the power of assembly with all of the convenience of assembly.

There have been attempts to get C++ accepted into the Linux kernel all the way back to the 90s. It never seems to get accepted and it never seems to go away.

Closest it seems to have made it is in various drivers.

#4. C++ In the Linux Kernel.

It is amazing how many times this keeps coming up over the years. People can't let go of this idea.

#3. Rust In the Linux Kernel

Rust is a general purpose language. It has been designed for type-safety, memory and concurrency.

Rust is going to be the second language that is supported by the Linux kernel.

Rust is ideal for writing device drivers and that will probably be the main focus for it.

#3. Rust In the Linux Kernel.

There is a whole OS written in Rust named Redox. It is progressing pretty quickly. Would be interesting if in 10 years we're running the Redox kernel with the GNU userspace.

There is also a project to rewrite coreutils in Rust as well. Will be interesting to see how that evolves.

#2. Lattice Encryption

Quantum computers are happening. It will be 10-15 years probably before they are really available. With Shor's algorithm public key cryptography might be vulnerable.

So there is talk about moving to other encryption algorithms such as Lattice-based encryption that are resistant to Quantum computers.

Why does this matter? If you're a nation state grabbing a bunch of traffic today and being able to decrypt it in 10-15 years (probably much less) could be VERY worthwhile.

#2. Lattice Encryption.

Whatever the standard will be it will probably be available in Linux long before then.

NIST is currently in round 3 of evaluating their post quantum encryption algorithms. They are the people who turned Rijndael into AES.

#1. RISC-V Support

This one is a big one.

RISC-V is analogous to the ARM except for a few things

RISC-V is open source. ARM is available for review, but to implement it you have to get a license. That being said ARM is much further along currently.

RISC-V will be used by countries that don't want to be dependent on Intel/AMD or other manufacturers.

#1. RISC-V Support.

RISC-V might get enough mindshare to actually influence the kernel development.

A supercomputer built out of RISC-V chips are kind of interesting.

Fully supporting RISC-V will shake a lot of bugs out of the Linux kernel and might lead to it being the 4 most popular architecture for the Linux kernel.

MangoPI is a Raspberry Pi competitor that uses a RISC-V cpu.

Summary - Q & A

That's all I've got.

- The Linux Kernel mailing list always has some interesting things going on in it.

Questions???

Links

#10. 2011 Version of C Used for Kernel

https://www.zdnet.com/article/linus-torvalds-prepares-to-move-the-linux-kernel-to-modern-c/

#9. LSM Module Stacking

https://lwn.net/Articles/804906/

https://lwn.net/Articles/891538/#:~:text=LSM%3A%20Module%20stacking%20for%20AppArmor%20This%20patchset%20provides,pseudo-code%20provided%20by%20Paul%20Moore%20as%20a%20basis

Links

#8. New Random Number Generator

https://www.theregister.com/2022/03/21/new_linux_kernel_has_improved/

#7. Intel's Software Upgradeable CPU

https://www.tomshardware.com/news/intel-software-defined-cpu-support-coming-to-linux-518

Links

#6. Firmware in the Kernel

https://www.theregister.com/2022/04/25/debian_firmware_debate/

https://fiendish.github.io/The-Debian-Gotham-Needs/

#5. Removing Intel x86 support

https://www.phoronix.com/scan.php?page=news_item&px=Linux-x86-Remove-A.Out

Links

#4. C++ In the Linux Kernel

https://www.threatstack.com/blog/c-in-the-linux-kernel

https://olegkutkov.me/2019/11/10/cpp-in-linux-kernel/

#3. Rust In the Linux Kernel

https://thenewstack.io/rust-in-the-linux-kernel-good-enough/

https://www.zdnet.com/article/rust-takes-a-major-step-forward-as-linuxs-second-official-language/

Links

#2. Lattice Encryption

https://www.nsa.gov/Cybersecurity/NSAs-Cybersecurity-Perspective-on-Post-Quantum-Cryptography-Algorithms/

https://www.nature.com/articles/d41586-022-00339-5

https://csrc.nist.gov/news/2016/public-key-post-quantum-cryptographic-algorithms

https://csrc.nist.gov/projects/post-quantum-cryptography

Links

#1. RISC-V Support

https://en.wikipedia.org/wiki/RISC-V

https://liliputing.com/?s=risc-v