Linus Torvalds & Greg Kroah-Hartman at Kubecon China 2024

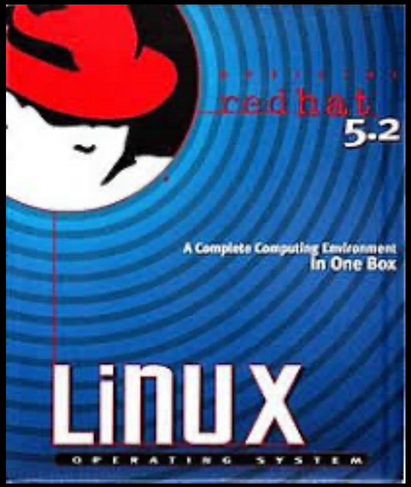
Steven Borrelli 11 Sept 2024

St. Louis UNIX Users Group

About

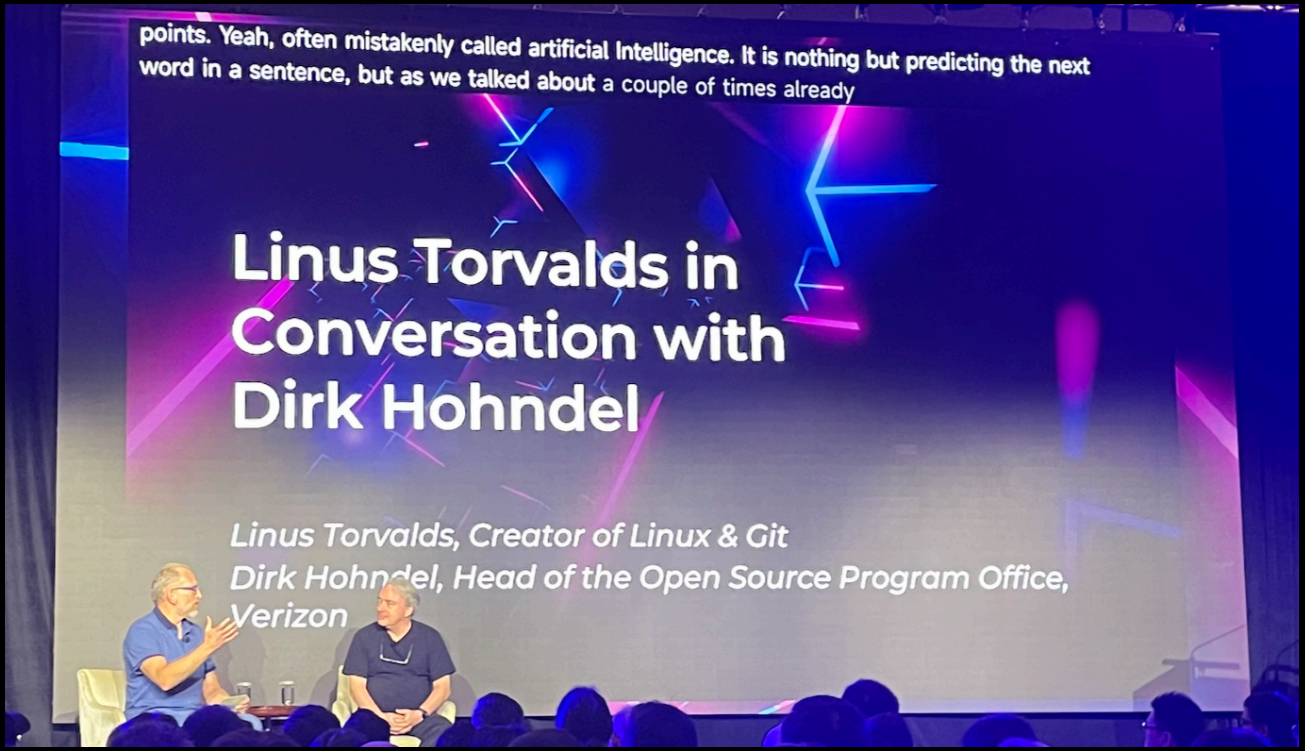
@stevendborrelli

- Intro to Unix: NeXTstep
- AIX Admin
- First Linux: RedHat 4.2 with 2.0.32 Kernel 1997
- Currently focus on Cloud and Kubernetes
- Work for upbound.io as a Principal Solutions Architect



Crossplane

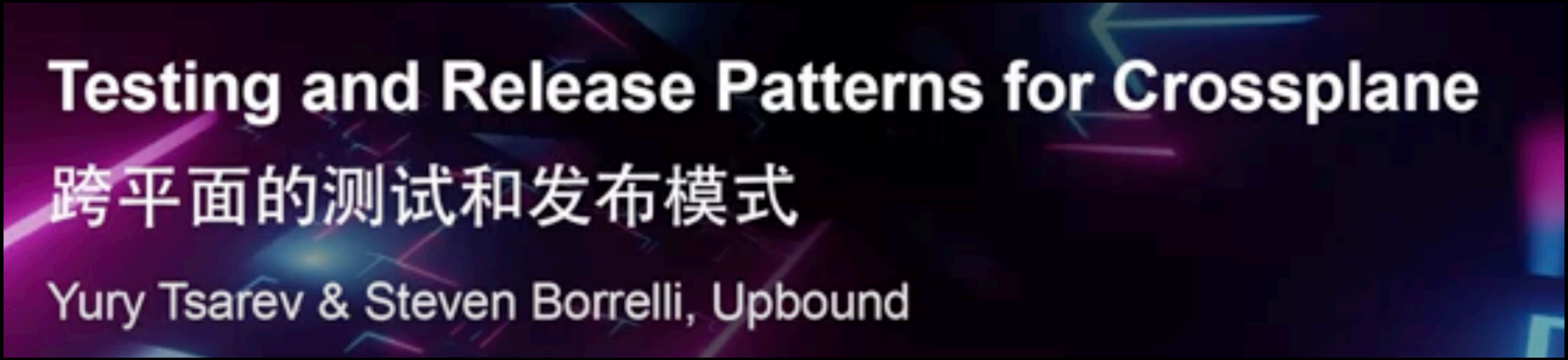
Origin of this Talk


points. Yeah, often mistakenly called artificial Intelligence. It is nothing but predicting the next word in a sentence, but as we talked about a couple of times already

Linus Torvalds in Conversation with Dirk Hohndel

Linus Torvalds, Creator of Linux & Git
Dirk Hohndel, Head of the Open Source Program Office, Verizon

Origin of this Talk



Testing and Release Patterns for Crossplane

跨平面的测试和发布模式

Yury Tsarev & Steven Borrelli, Upbound

https://sched.co/1eYZ7

Linux Kernel Security Process

or

"Why are there so many kernel CVEs now?"

Greg Kroah-Hartman
gregkh@linuxfoundation.org
git.sr.ht/~gregkh/presentation-security

THE LINUX FOUNDATION

Link to Presentation & Video: https://sched.co/1fAqj

Linux is a Massive Project

Most code is in Drivers

Linux size – overall

85,000 files

38,640,000 lines

Linux size – what you use

5%-10%

Complexity

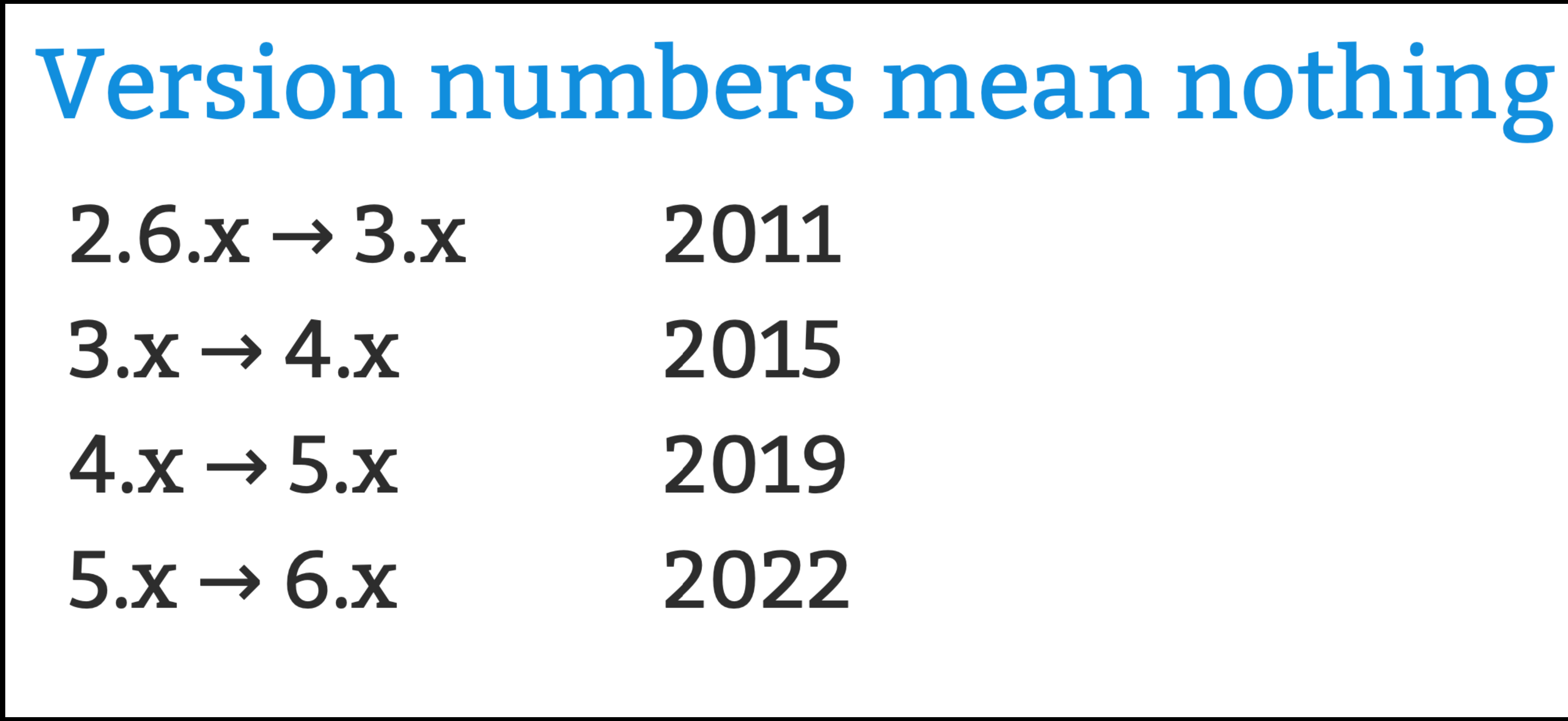
Lines of Code

Linux Core is fairly small, most of the code is in drivers.

- Server ~ 1.5MM
- Desktop/Laptop ~2MM
- Phone ~ 4MM

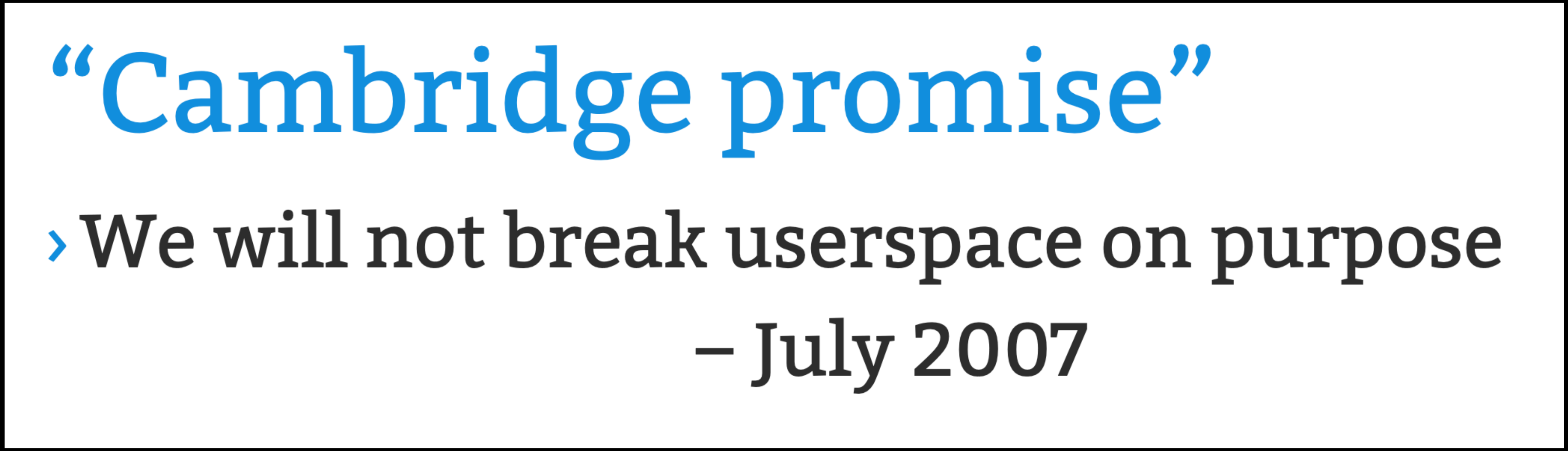
Development Process

- 9 Changes an Hour
- 9 Week Release Cycle
- All releases are stable:



Version numbers mean nothing

2.6.x → 3.x	2011
3.x → 4.x	2015
4.x → 5.x	2019
5.x → 6.x	2022



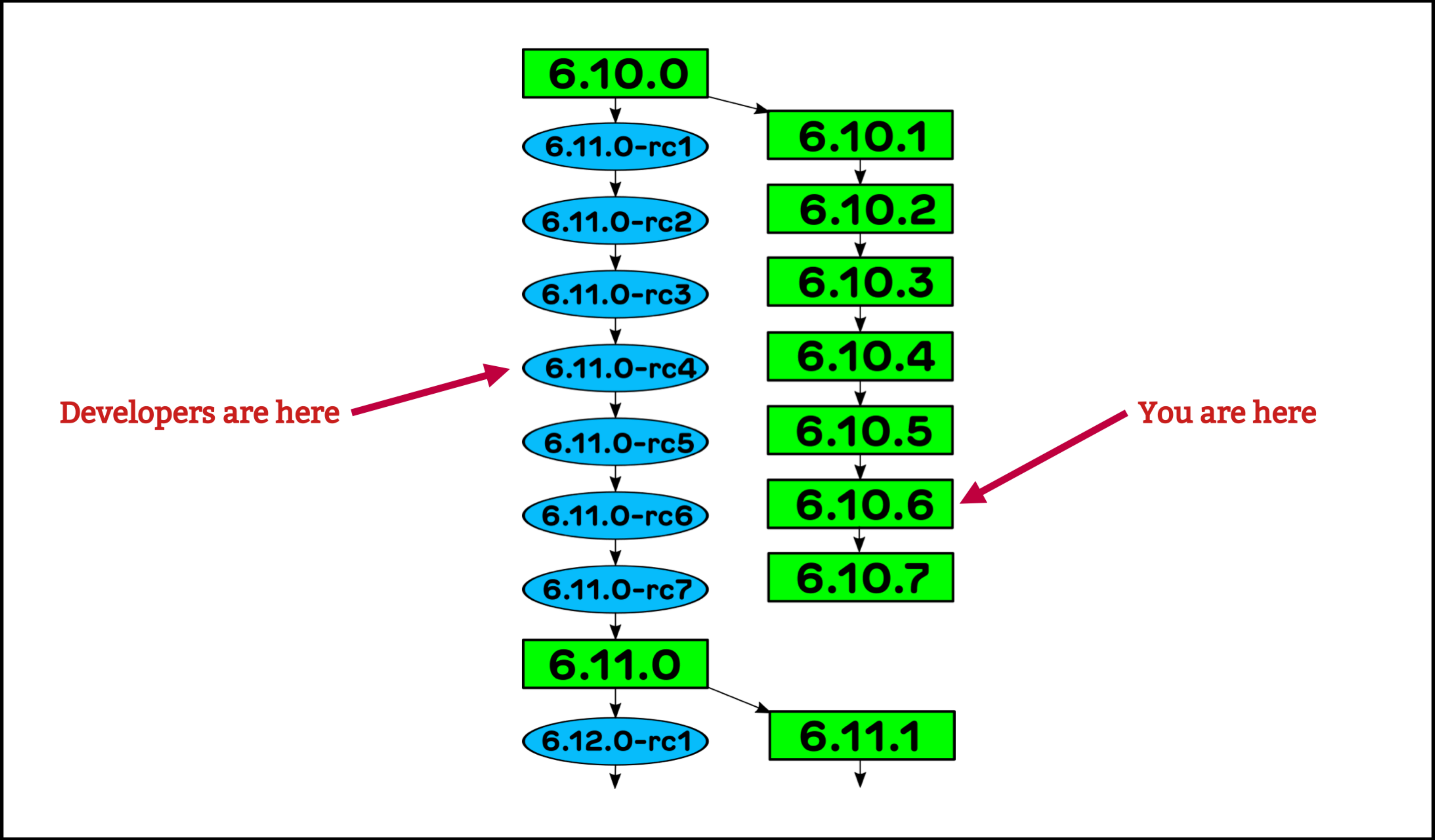
“Cambridge promise”

› We will not break userspace on purpose

– July 2007

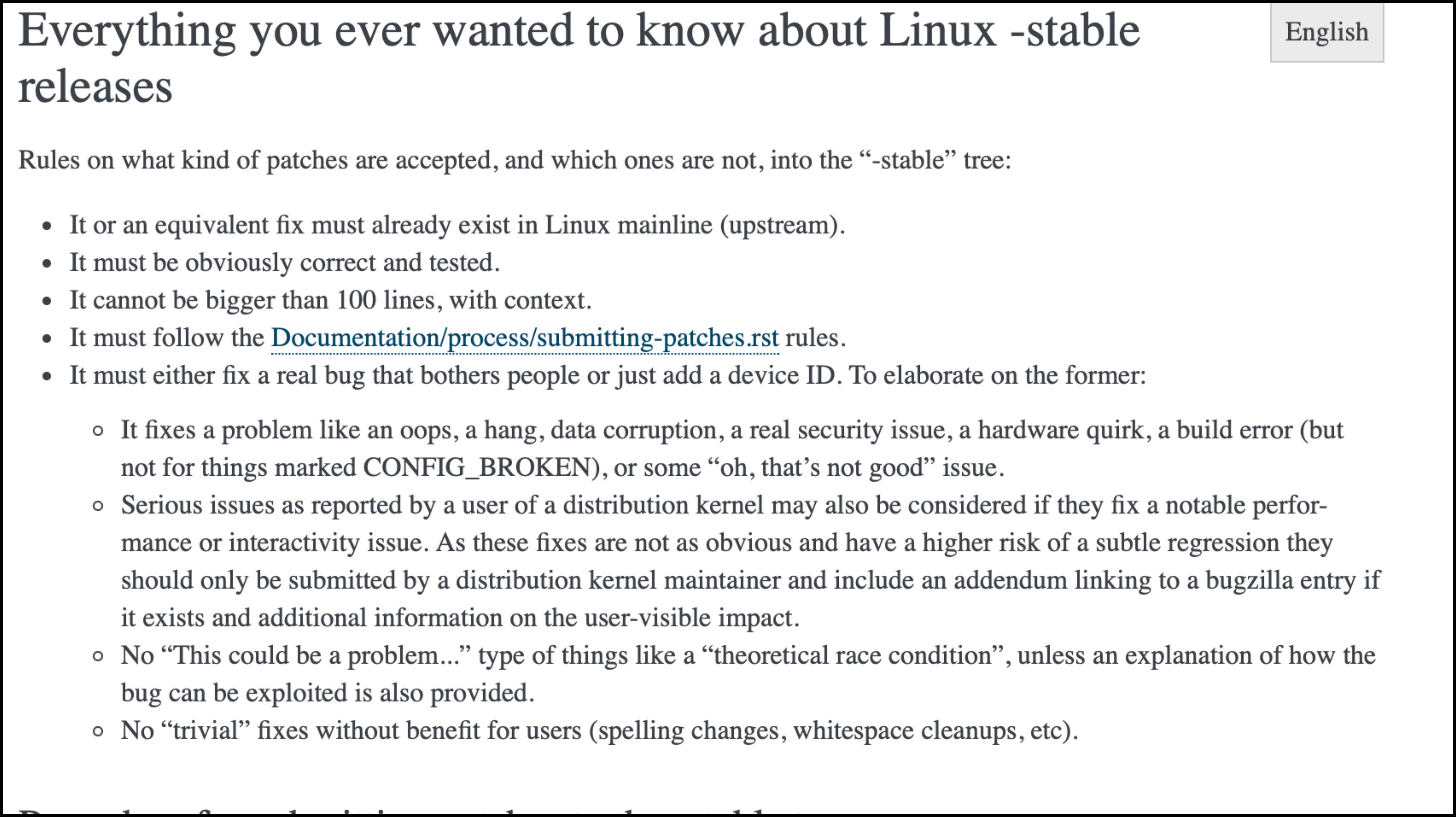
Parallel Development

1-2 Stable Kernels per week



Rules for Stable Kernels

https://kernel.org/doc/html/latest/process/stable-kernel-rules.html



Everything you ever wanted to know about Linux -stable releases

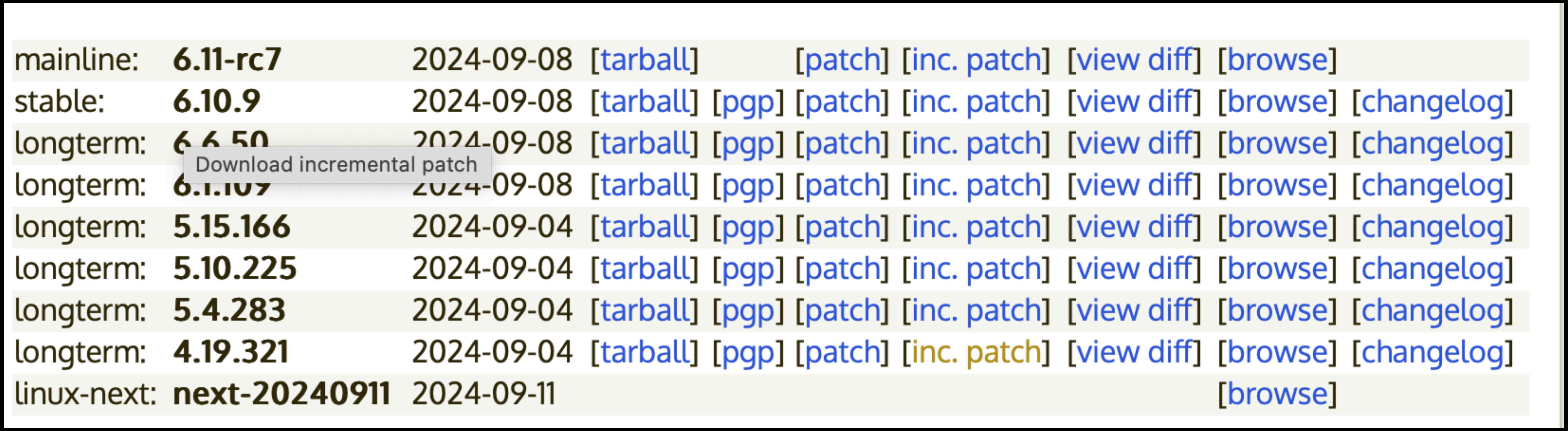
English

Rules on what kind of patches are accepted, and which ones are not, into the "-stable" tree:

- It or an equivalent fix must already exist in Linux mainline (upstream).
- It must be obviously correct and tested.
- It cannot be bigger than 100 lines, with context.
- It must follow the Documentation/process/submitting-patches.rst rules.
- It must either fix a real bug that bothers people or just add a device ID. To elaborate on the former:
  - It fixes a problem like an oops, a hang, data corruption, a real security issue, a hardware quirk, a build error (but not for things marked CONFIG_BROKEN), or some "oh, that's not good" issue.
  - Serious issues as reported by a user of a distribution kernel may also be considered if they fix a notable performance or interactivity issue. As these fixes are not as obvious and have a higher risk of a subtle regression they should only be submitted by a distribution kernel maintainer and include an addendum linking to a bugzilla entry if it exists and additional information on the user-visible impact.
  - No "This could be a problem..." type of things like a "theoretical race condition", unless an explanation of how the bug can be exploited is also provided.
  - No "trivial" fixes without benefit for users (spelling changes, whitespace cleanups, etc).

Longterm Kernels

Usually the December release is selected for LTS, with frequent updates. For example 6.6.50 is the current LTS 6.6.x release, 6.1 is at 6.1.109.



mainline:	**6.11-rc7**	2024-09-08	[tarball]		[patch]	[inc. patch]	[view diff]	[browse]	
stable:	**6.10.9**	2024-09-08	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	**6.6.50**	2024-09-08	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	**6.1.109**	2024-09-08	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	**5.15.166**	2024-09-04	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	**5.10.225**	2024-09-04	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	**5.4.283**	2024-09-04	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	**4.19.321**	2024-09-04	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
linux-next:	**next-20240911**	2024-09-11						[browse]	

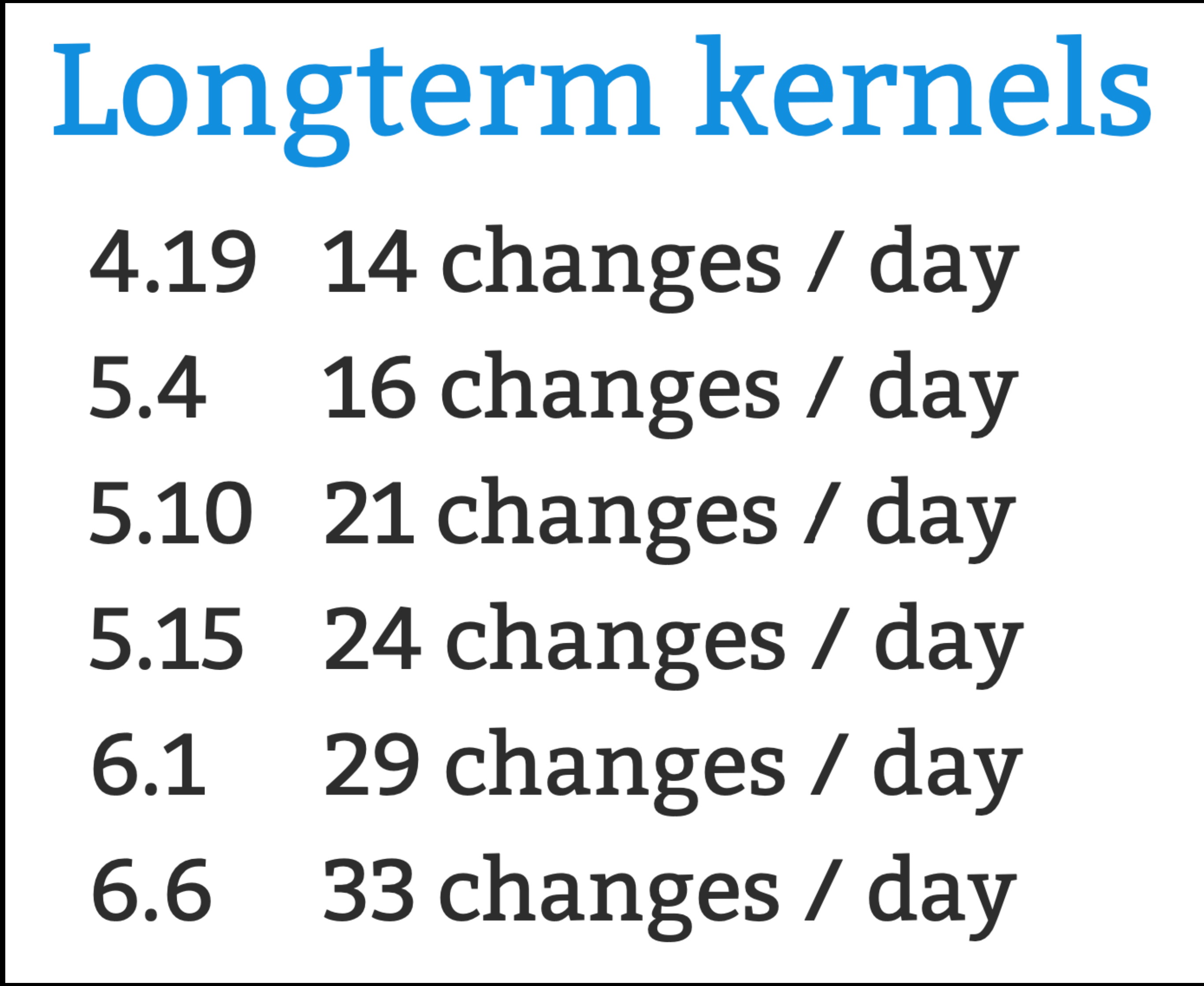
Long Term Support

Long Term Support was cut from 6 Years to 2 in 2023.

Older Kernels are harder to maintain and require Sr. Engineers.

6.6 expires 31 Dec 2026

See https://endoflife.date/linux



Longterm kernels

4.19	14 changes / day
5.4	16 changes / day
5.10	21 changes / day
5.15	24 changes / day
6.1	29 changes / day
6.6	33 changes / day

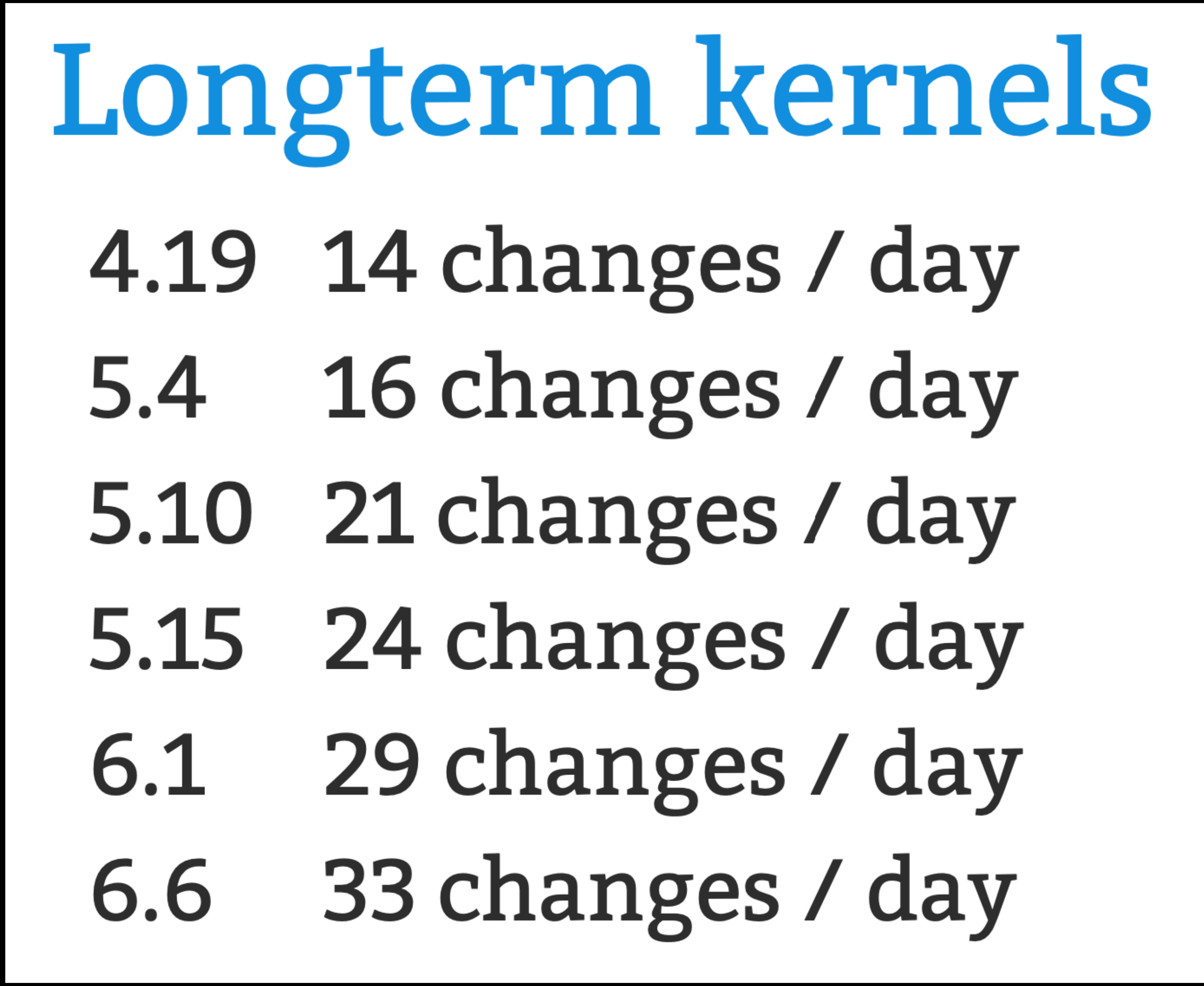
Long Term Support

Long Term Support was cut from 6 Years to 2 in 2023.

Older Kernels are harder to maintain and require Sr. Engineers.

6.6 expires 31 Dec 2026

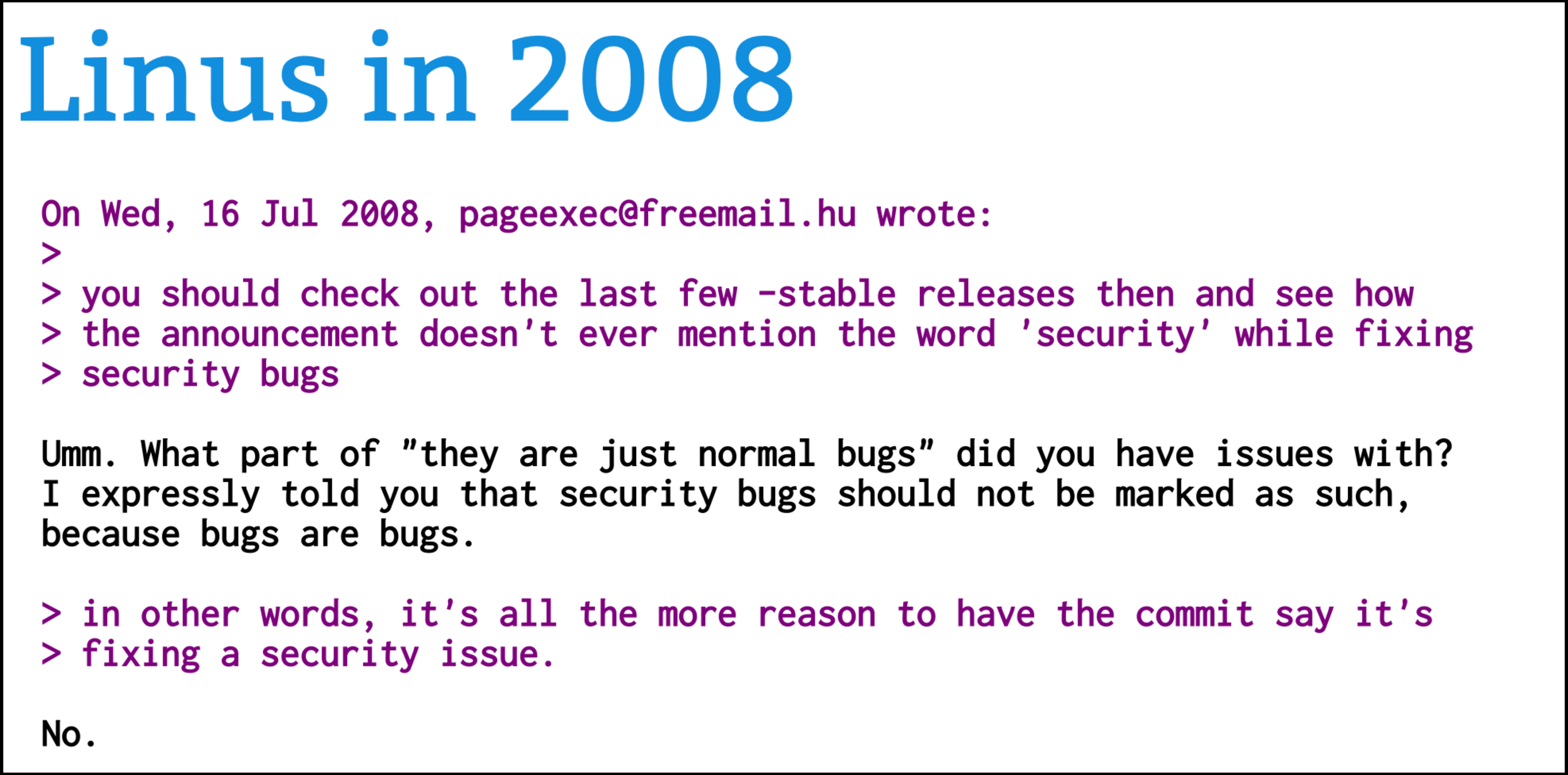
See https://endoflife.date/linux



Longterm kernels

4.19	14 changes / day
5.4	16 changes / day
5.10	21 changes / day
5.15	24 changes / day
6.1	29 changes / day
6.6	33 changes / day

In Linux security bugs are bugs!



Linus in 2008

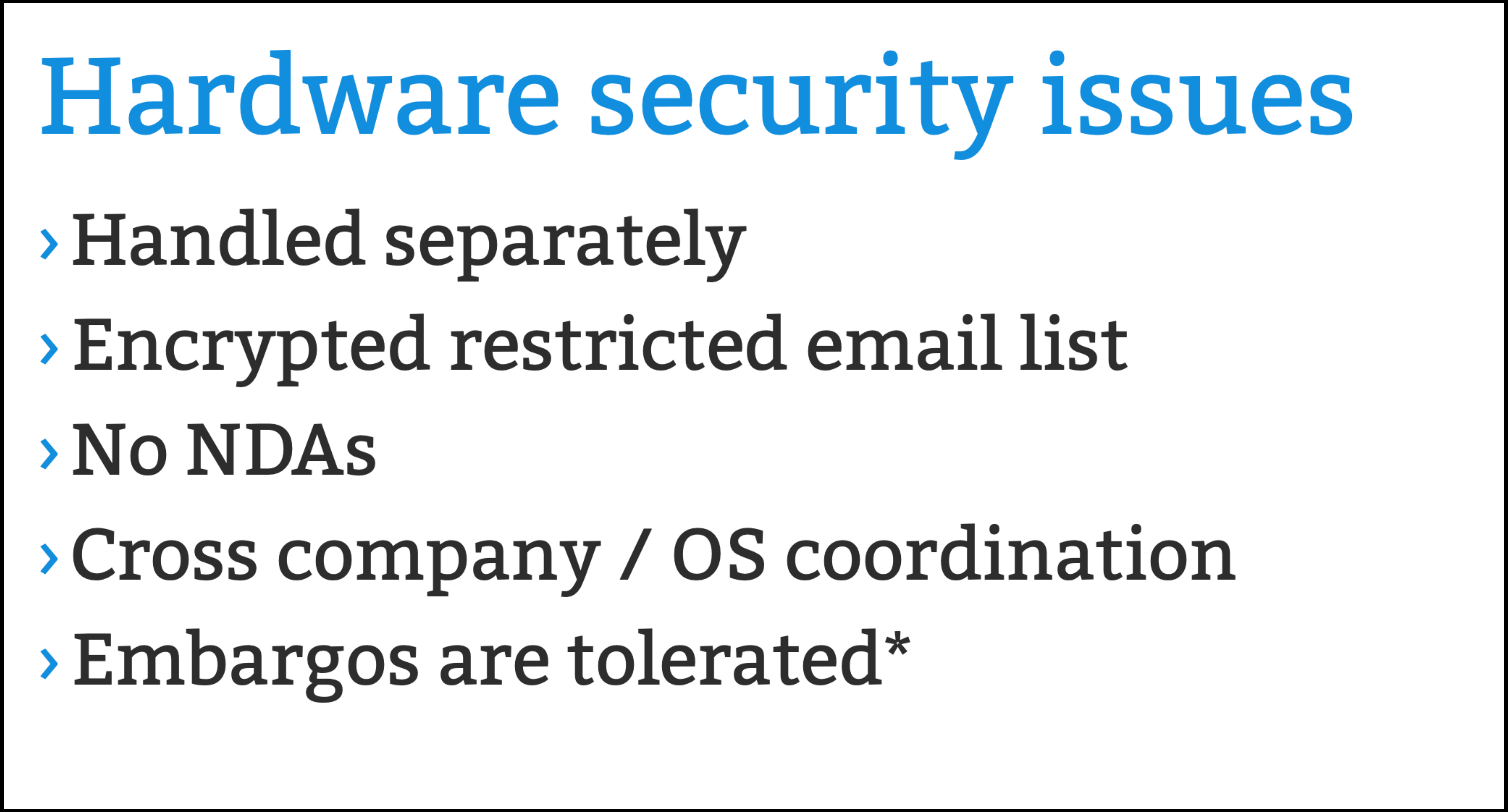
```
On Wed, 16 Jul 2008, pageexec@freemail.hu wrote:
>
> you should check out the last few -stable releases then and see how
> the announcement doesn't ever mention the word 'security' while fixing
> security bugs

Umm. What part of "they are just normal bugs" did you have issues with?
I expressly told you that security bugs should not be marked as such,
because bugs are bugs.

> in other words, it's all the more reason to have the commit say it's
> fixing a security issue.

No.
```

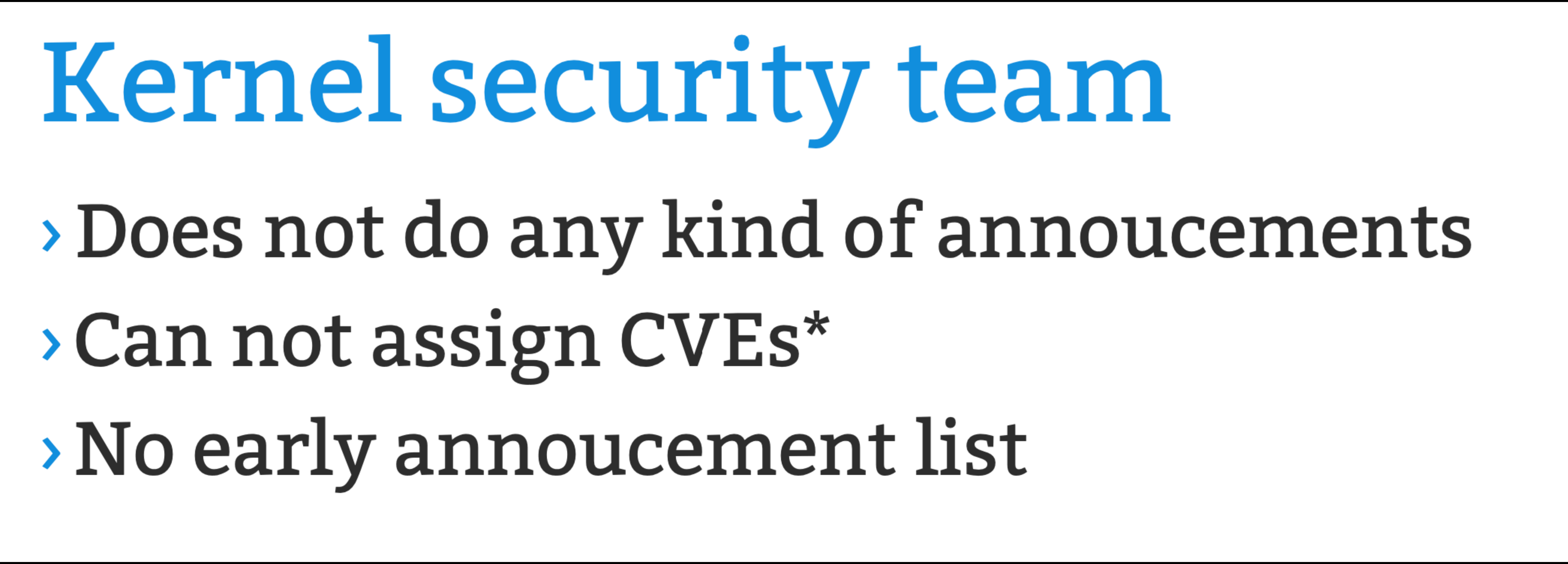
Hardware is Treated Differently


Hardware security issues

- Handled separately
- Encrypted restricted email list
- No NDAs
- Cross company / OS coordination
- Embargos are tolerated*

https://www.kernel.org/doc/html/latest/process/embargoed-hardware-issues.html

No pre-disclosure



Kernel security team

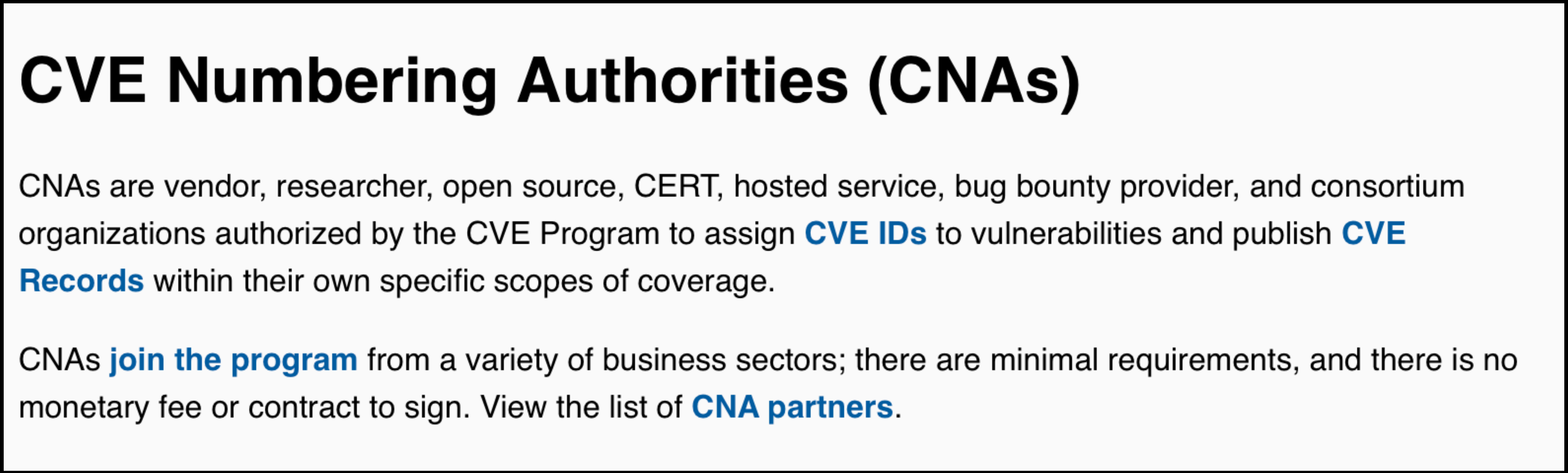
- Does not do any kind of annoucements
- Can not assign CVEs*
- No early annoucement list

Linux is now a CNA

https://www.cve.org/ProgramOrganization/CNAs

EU Regulations for Open Source Project Vulnerabilities

CVEs were being abused by some vendors



CVE Numbering Authorities (CNAs)

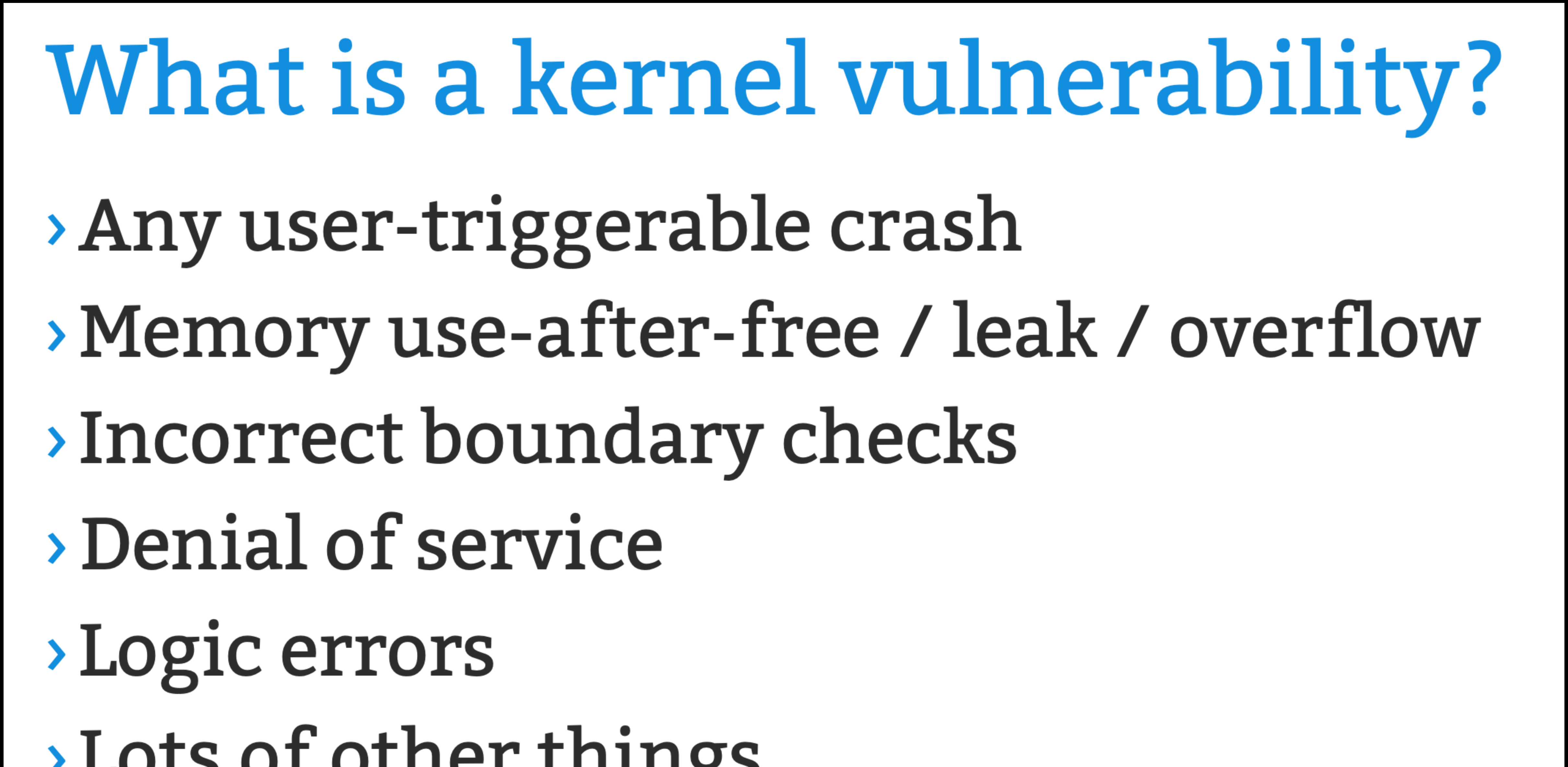
CNAs are vendor, researcher, open source, CERT, hosted service, bug bounty provider, and consortium organizations authorized by the CVE Program to assign **CVE IDs** to vulnerabilities and publish **CVE Records** within their own specific scopes of coverage.

CNAs **join the program** from a variety of business sectors; there are minimal requirements, and there is no monetary fee or contract to sign. View the list of **CNA partners**.

What can be a CVE?

Common Vulnerabilities and Exposures (CVEs) are a way to track security vulnerabilities in hardware and software.

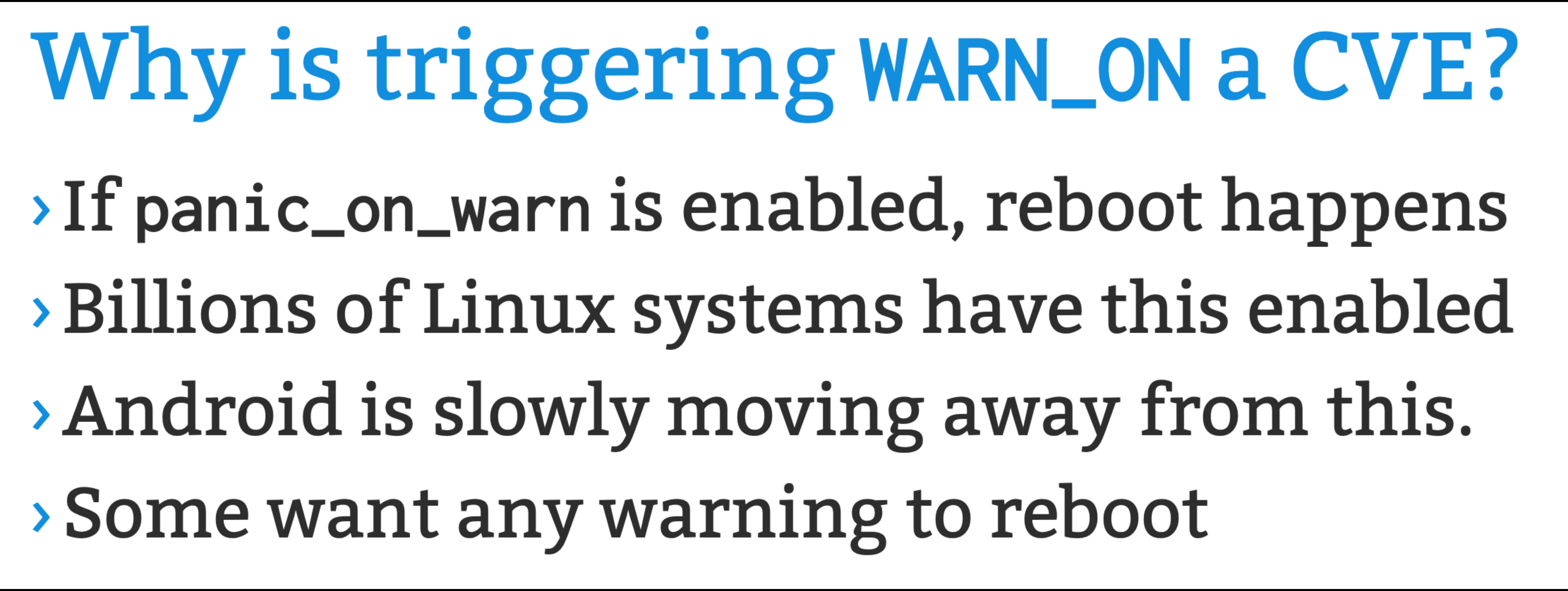
A Vulnerability is broadly defined.



What is a kernel vulnerability?

- Any user-triggerable crash
- Memory use-after-free / leak / overflow
- Incorrect boundary checks
- Denial of service
- Logic errors
- Lots of other things

panic_on_warn



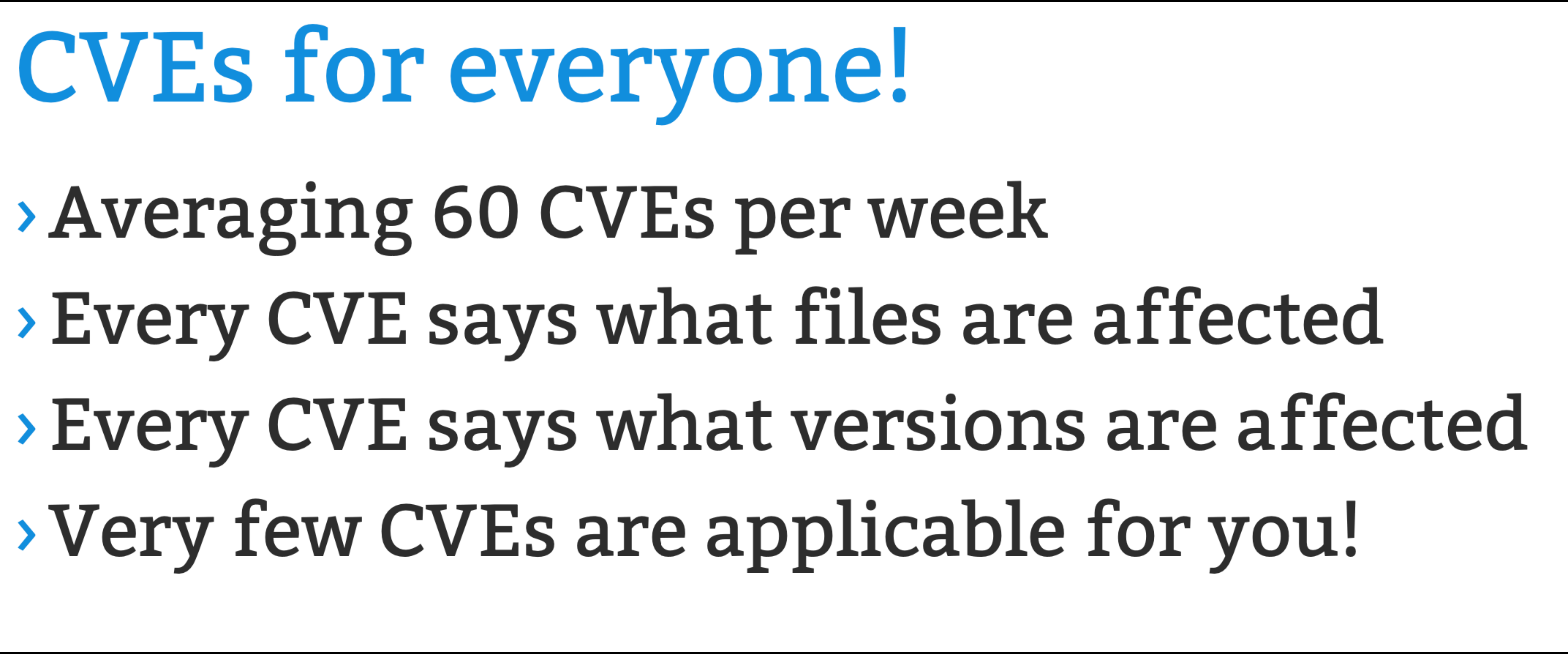
Why is triggering WARN_ON a CVE?

- If panic_on_warn is enabled, reboot happens
- Billions of Linux systems have this enabled
- Android is slowly moving away from this.
- Some want any warning to reboot

CVE Lookups are Available

CVE Reports are in a Standard Format

Multiple Search Engines.



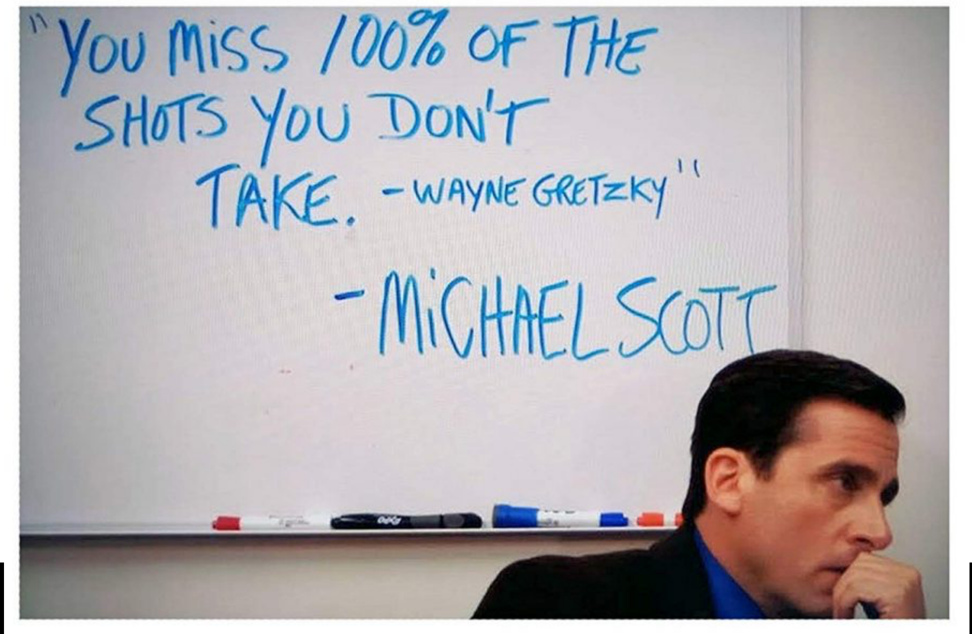
CVEs for everyone!

- Averaging 60 CVEs per week
- Every CVE says what files are affected
- Every CVE says what versions are affected
- Very few CVEs are applicable for you!

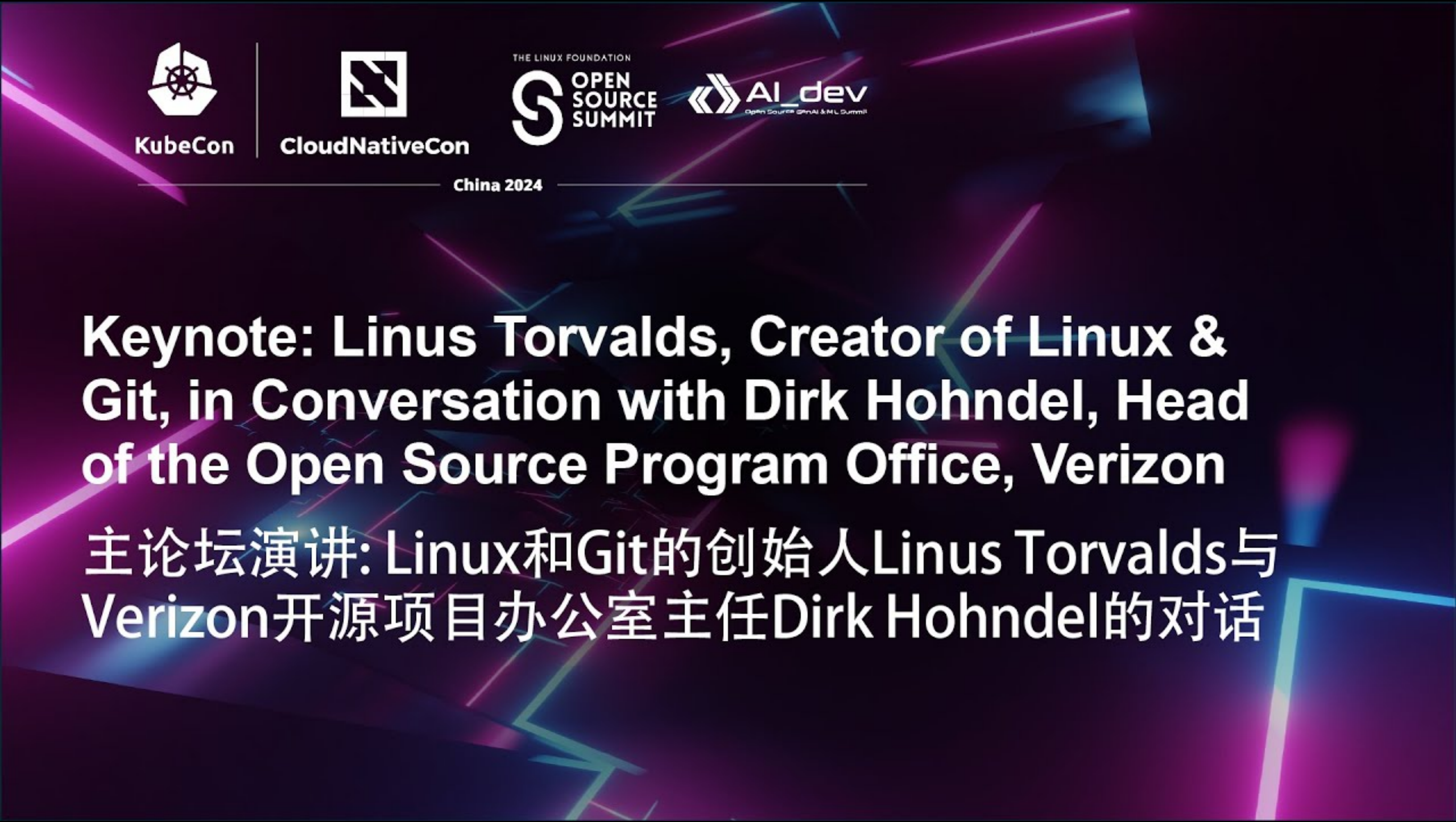
Stay Up to Date

"If you are not using the latest a stable / longterm kernel, your system is insecure"

– me



- Greg Kroah-Hartman



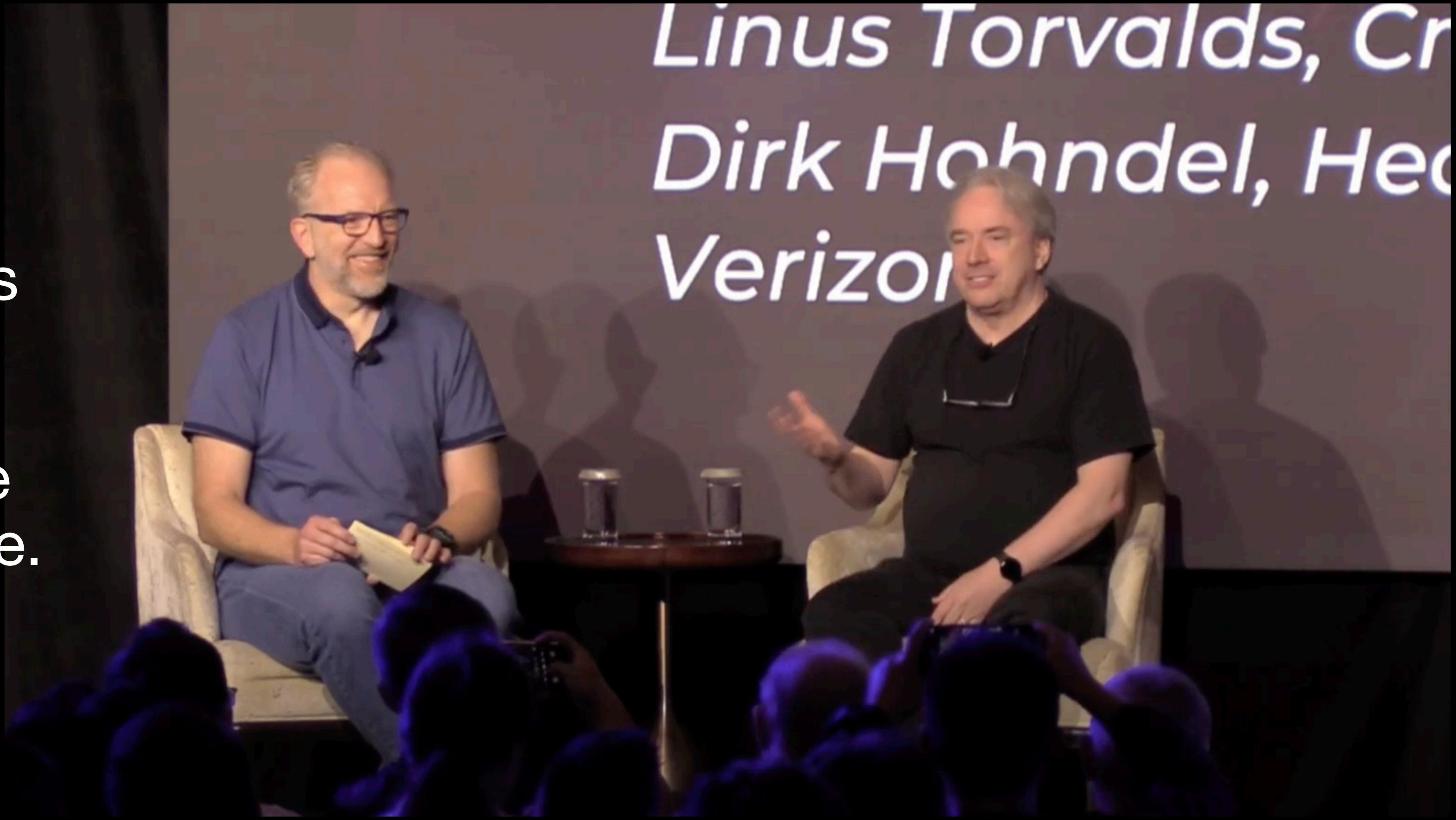
Keynote: Linus Torvalds, Creator of Linux & Git, in Conversation with Dirk Hohndel, Head of the Open Source Program Office, Verizon

主论坛演讲: Linux和Git的创始人Linus Torvalds与Verizon开源项目办公室主任Dirk Hohndel的对话

https://youtu.be/nJMEuBwMD18?si=y_yTKVDrnY_vPKzx

Linus hates public speaking, but he enjoys Q&A sessions with Dirk Hohndel.

He doesn't know the questions in advance.

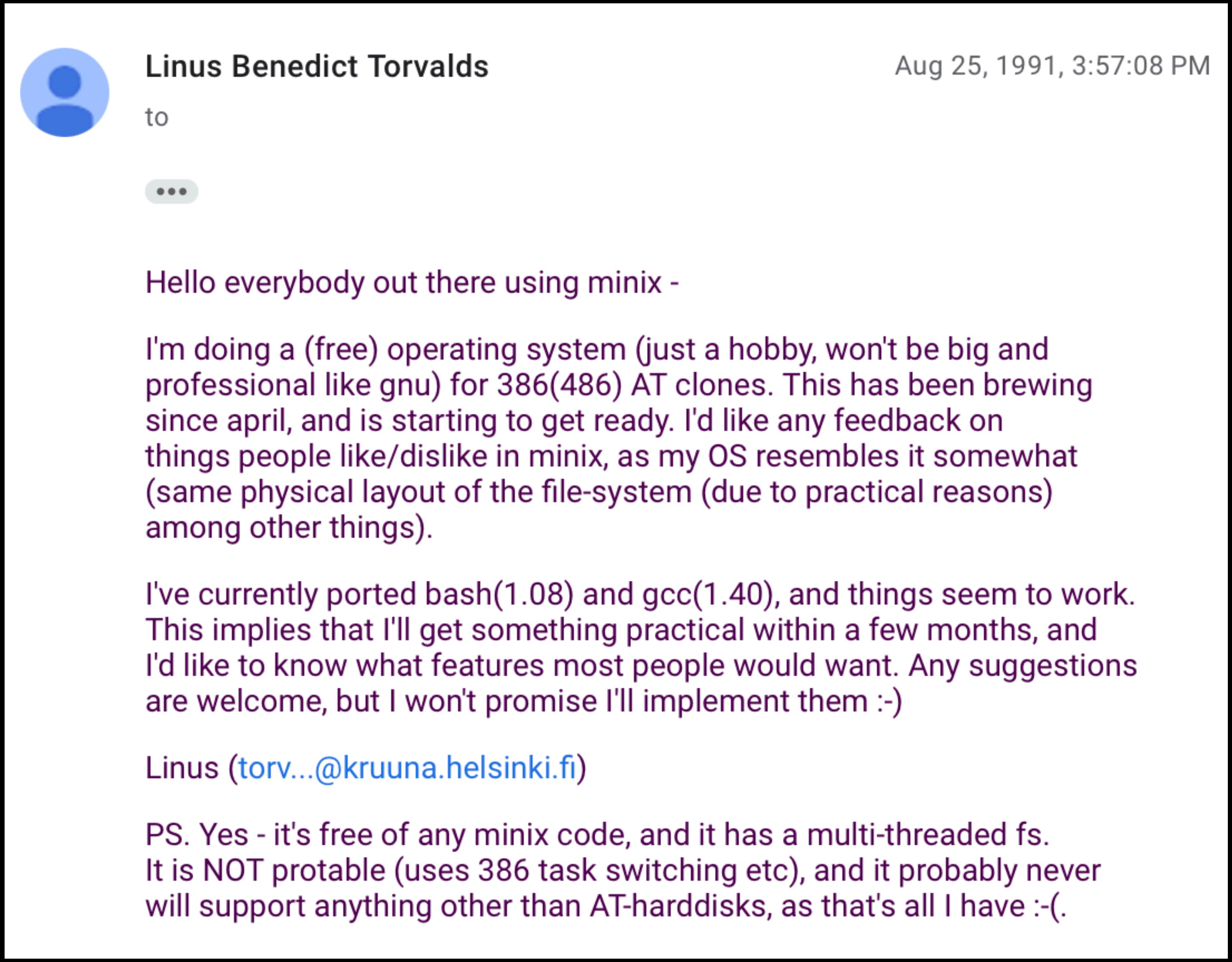


The Start of Linux

The Q&A session takes place 33 years after the first public email about a "(free) operating system".

The project started 4 months earlier, so it is a third of a century old.

"And it's still almost ready now!"



Linus Benedict Torvalds — Aug 25, 1991, 3:57:08 PM

to

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-system (due to practical reasons) among other things).

I've currently ported bash(1.08) and 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 would want. Any suggestions are welcome, but I won't promise I'll implement them :-)

Linus (torv...@kruuna.helsinki.fi)

PS. Yes - it's free of any minix code, and it has a multi-threaded fs. It is NOT protable (uses 386 task switching etc), and it probably never will support anything other than AT-harddisks, as that's all I have :-(.

The Newest Release: 6.11.0-rc4

“We’ve been doing this same release process for almost 20 years, you’d think the basics would have been fixed a long time ago.”

Drivers are 1/2 of each release, but still spends significant time on core kernel operations like memory management.

New behavior patterns mean we still need to tweak things in the core.

Extensible Scheduler in 6.11

A new framework that allows developers to implement host-wide scheduling policies as BPF programs in user space.

Not going to be included in 6.11 due to a death in the developer community.

Now looking to include into 6.12.

Could this be a STL Linux talk? https://www.socallinuxexpo.org/scale/21x/presentations/sched-ext-pluggable-linux-scheduler

On Future Planning

“We don’t look 5 years ahead, we look 1-2 releases ahead.

Later this year we will have the 20th anniversary of the Linux Realtime project. They are almost done.

It seems that kernel development happens quickly, but many features are developed in parallel before they are included into the Kernel.

The Release Process

11 Kernels every 2 years. Kernel major version is updated around every 20.

We have a structured development process that came out of not having a structured process. With companies contributions and multiple distributions, it became very painful for Linus.

Release management is very strict. Code that is not ready by the release window will not be accepted.

“We used to have a release schedule that had a goal of a year and ended up taking multiple years. It was a big painful change.”

Linus wanted to release every 6 weeks, settled on 9 weeks. People are as happy “as kernel developers ever are”

Taking Ownership of CVEs

Six months ago the Linux community (Greg) took ownership of CVE. There are 60 new issues a week: “is it time to panic?”

Bugs are inevitable. Process can be implemented to reduce the impact and number of bugs. Any bug can be a security bug.

Security is important, but we need to move forward so that new hardware and features can be supported.

Greg took over because having someone else maintain a list of Linux CVEs was too painful, and some did not make any sense.

Every Security Issue is Just a Bug

Linux can quickly address and ship fixes to bugs.

“There is a tendency in the IT industry to treat security issues as special and it ends up harming everybody.”

90-day delays/embargoes that can extend to 400 days. It’s demoralizing to the developers to not talk about the issue until a Press Release is released.

No embargoes longer than one week. Otherwise it hurts development.

Hardware issues have much longer embargoes. A few years ago there were multiple HW bugs that the core kernel team could not talk about.

LTS vs. Stable Releases

LTS vs Stable. Some distributions are starting to update Kernels more rapidly.

What makes more sense: Old kernel with back ports or go to a new Kernel?

Old kernels do get fixes, but some get missed because were not considered important.

Companies that have stayed back and wait for a few years may have a painful upgrade process.

Most distributions try the stay back approach but end up migrating to rolling releases.

LTS vs. Stable Releases

A lot of embedded developers are still on 4.9 kernels.

“It’s a bad idea.” Even some the super-long supported Kernels. It’s hard to support older kernels.

Rust in the Kernel

“The slowly increasing footprint of Rust in the Kernel. It has been frustrating, I had expected uptake in the Kernel to be faster.”

A large part of it is experienced Kernel developers are used to C. There has been pushback on Rust.

Rust infrastructure has not been stable. In 6.10 we finally can use the upstream Rust compiler. It has taken more than two years.

Cloud

“I didn’t know we were at Kubecon”.

“The only thing that matters is the Kernel.”

Specialization

Moore’s law is gone. Now scaling is heading to distributed systems like Kubernetes.

One of the The thing that makes it all practical is that people specialize what they are interested in.

So when it comes to things like Cloud and AI, I see myself as a Kernel person who wants to support that, but I am not interested in the end result.

Nvidia

With AI, Nvidia suddenly got much more involved in Kernel side, went from companies not doing good to companies doing really good work.

I end up being interested in what we need to do in the kernel to support AI. There was a lot of memory management that ended up being updated for AI workloads to use accelerators in user space.

“I know linux, I don’t know cloud"

AI

“Autocorrect on Steroids”- Dirk

Interesting use cases for LLM: making work easier for maintainers. I’ve been talking to companies

I don’t like AI in that it’s horrible hype that keeps coming.

I hope in 5 years or sooner we have every tools. We’re not at the point where AI are helping us find patterns in the kernel. I’m optimizing about it.

I’m more interesting in finding bugs and code review and helping maintainers, not in code generation.

LLMs

What are you looking for LLMs?

Something that hopefully takes the Kernel source code history into account and learns what good patterns are, and red flags.

We have a lot of tools that flag red-flags, but they find obvious things. I think AI can do better.

Final Questions

From ChatGPT: “how do you see the future in open source evolving especially with proprietary cloud services and platforms?”

“This is why I like doing sessions with Dirk because doesn’t do the bullshit questions like what is the vision.”

“I never had a big vision, I don’t want to have a big vision. I associate visions with drugs and mind issues. I see myself as a plodding engineer and I’m proud of that”.

Use of LLMs in Kernel Dev

What are you looking for LLMs?

Something that hopefully takes the Kernel source code history into account and learns what good patterns are, and red flags.

We have a lot of tools that flag red-flags, but they find obvious things. I think AI can do better.

Final Thoughts

Linus is a master craftsman with an intense focus and strong management skills.

The regimented software release process allows one of the largest software projects in history to run smoothly.

In an industry full of hype and BS, he happily remains in his niche.