Linux Kernel Development

Greg Kroah-Hartman
gregkh@linuxfoundation.org

github.com/gregkh/kernel-development

61,000 files
25,000,000 lines

Kernel release 4.14.0

4,394 developers
≈530 companies

Kernel releases 4.9.0 – 4.14.0
December 2016 – November 2017

10,000 lines added

2,700 lines removed

2,000 lines modified

Kernel releases 4.9.0 – 4.14.0
December 2016 – November 2017

10,000 lines added

2,700 lines removed

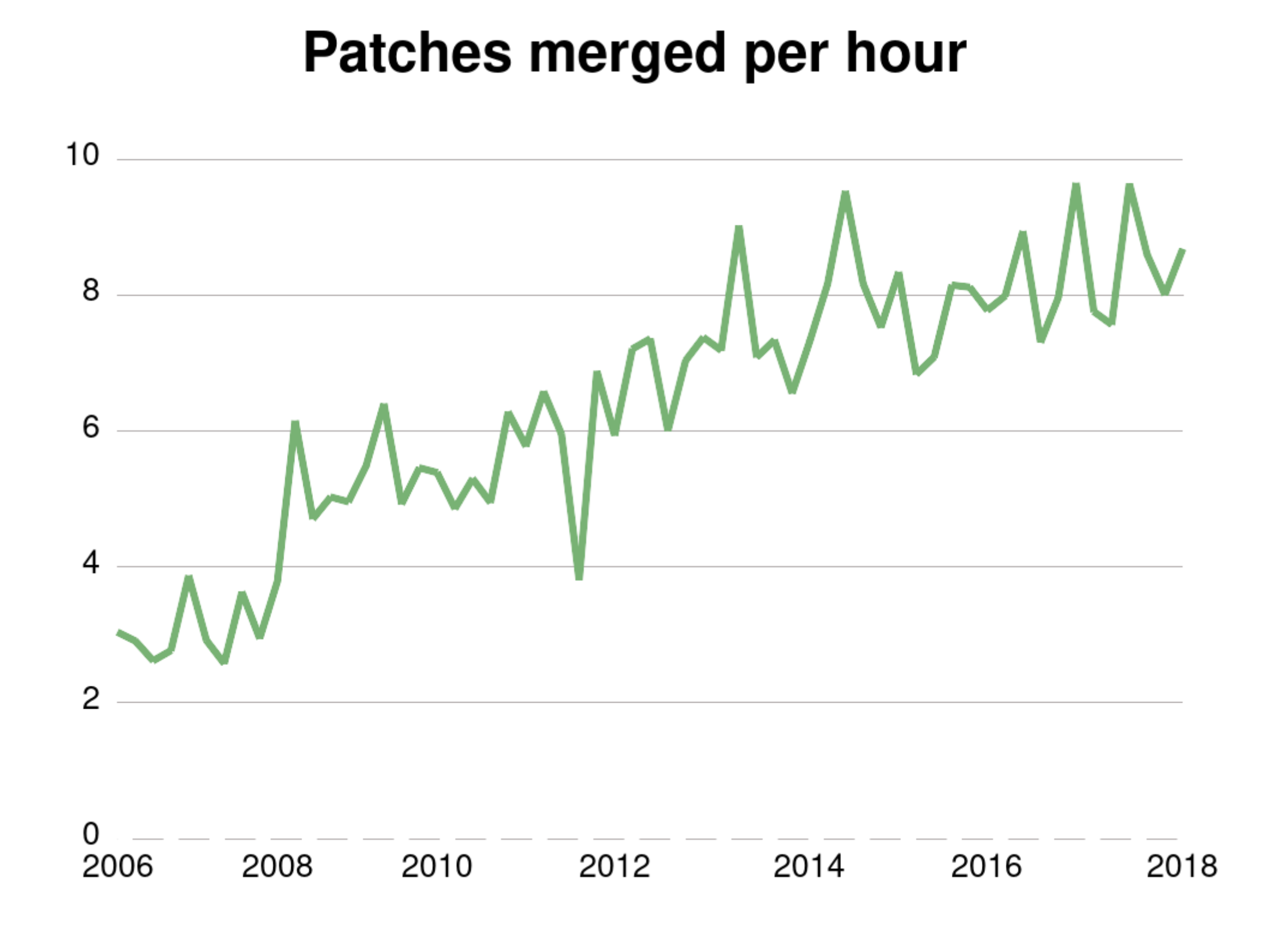
2,000 lines modified

Every day

Kernel releases 4.9.0 – 4.14.0
December 2016 – November 2017

8.5 changes per hour

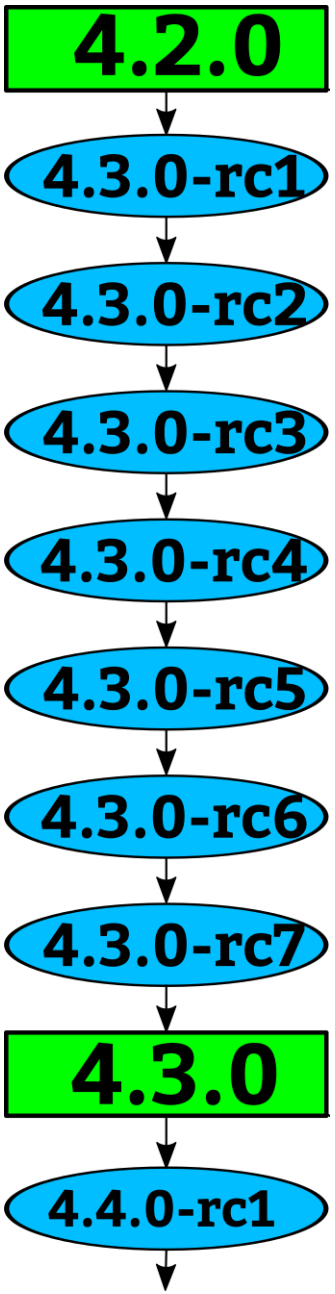
Kernel releases 4.9.0 – 4.14.0
December 2016 – November 2017

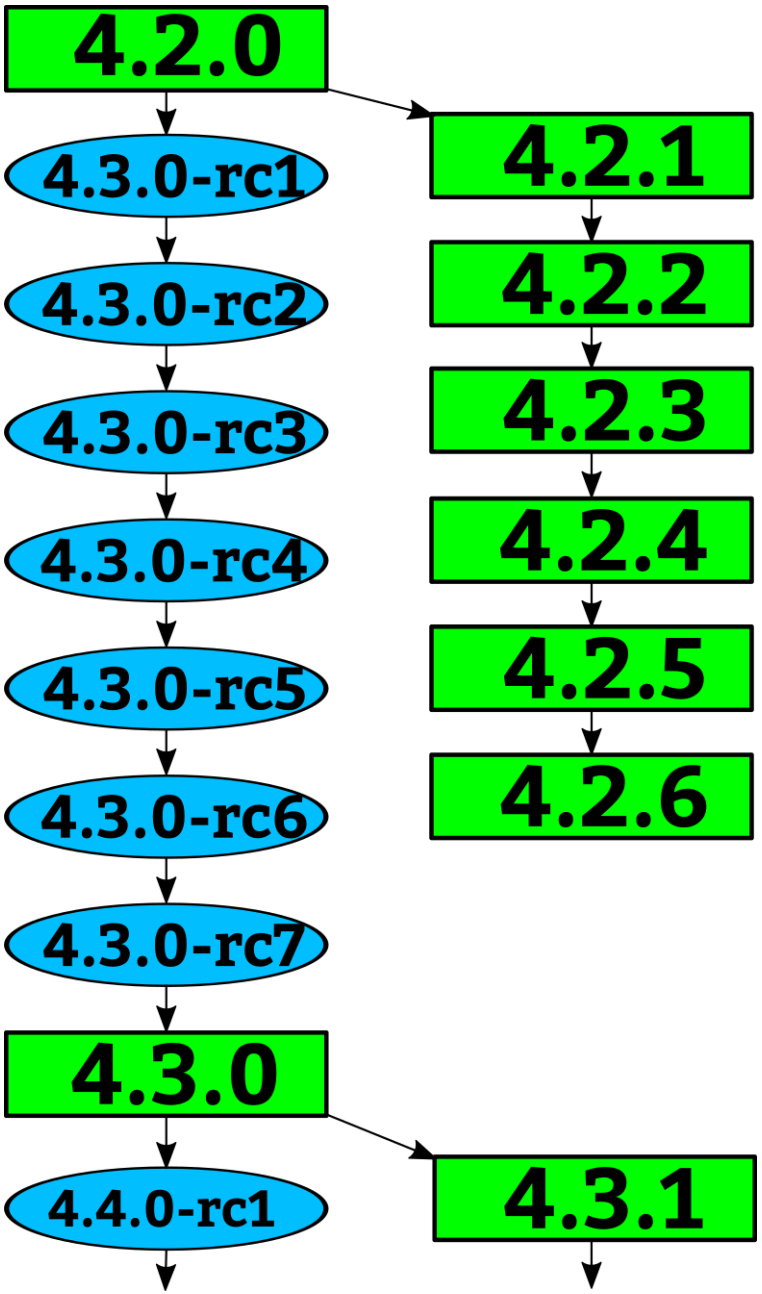

Patches merged per hour
10
8
6
4
2
0
2006
2008
2010
2012
2014
2016
2018

How we stay sane

Time based releases
Incremental changes


New release every
2½ months
Kernel releases 3.10.0 – 3.14.0


4.2.0
4.3.0-rc1
4.3.0-rc2
4.3.0-rc3
4.3.0-rc4
4.3.0-rc5
4.3.0-rc6
4.3.0-rc7
4.3.0
4.4.0-rc1

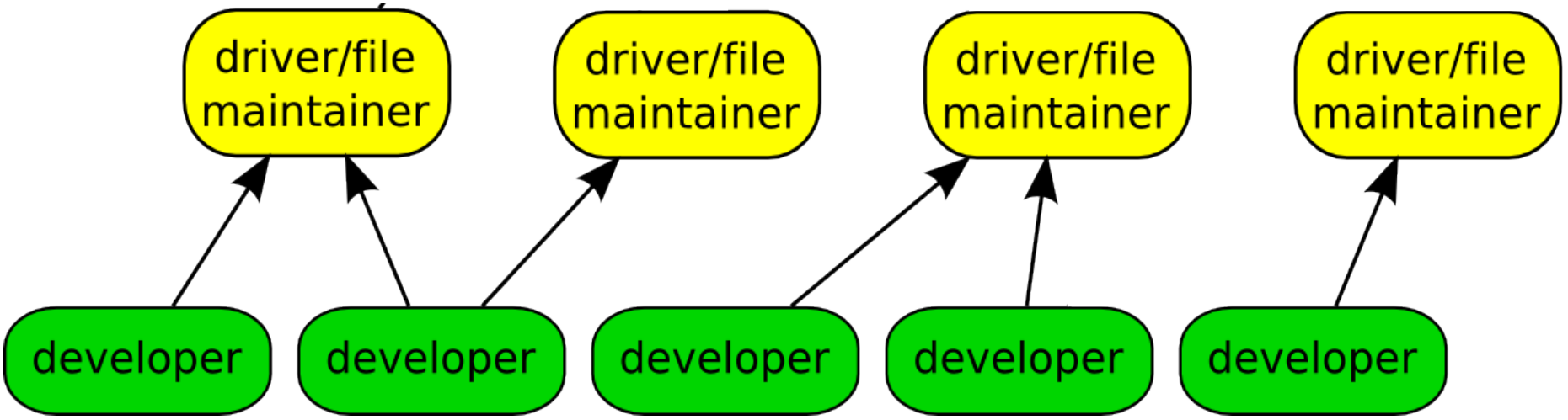

4.2.0
4.3.0-rc1
4.3.0-rc2
4.3.0-rc3
4.3.0-rc4
4.3.0-rc5
4.3.0-rc6
4.3.0-rc7
4.3.0
4.4.0-rc1
4.2.1
4.2.2
4.2.3
4.2.4
4.2.5
4.2.6
4.3.1

“Longterm kernels”

One picked per year
Maintained for two years

4.4 4.9 4.14


developer
developer
developer
developer
developer


driver/file maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
developer
developer
developer
developer
developer

```
commit ecf85e481a716cfe07406439fdc7ba9526bbfaeb
Author:     Robert Jarzmik <robert.jarzmik@free.fr>
AuthorDate: Tue Apr 21 20:33:10 2009 -0700
Commit:     Greg Kroah-Hartman <gregkh@suse.de>
CommitDate: Thu Apr 23 14:15:31 2009 -0700

    USB: otg: Fix bug on remove path without transceiver

    In the case where a gadget driver is removed while no
    transceiver was found at probe time, a bug in
    otg_put_transceiver() will trigger.

    Signed-off-by: Robert Jarzmik <robert.jarzmik@free.fr>
    Acked-by: David Brownell <dbrownell@users.sourceforge.net>
    Signed-off-by: Greg Kroah-Hartman <gregkh@suse.de>

--- a/drivers/usb/otg/otg.c
+++ b/drivers/usb/otg/otg.c
@@ -43,7 +43,8 @@ EXPORT_SYMBOL(otg_get_transceiver);
 void otg_put_transceiver(struct otg_transceiver *x)
 {
-       put_device(x->dev);
+       if (x)
+               put_device(x->dev);
 }
```

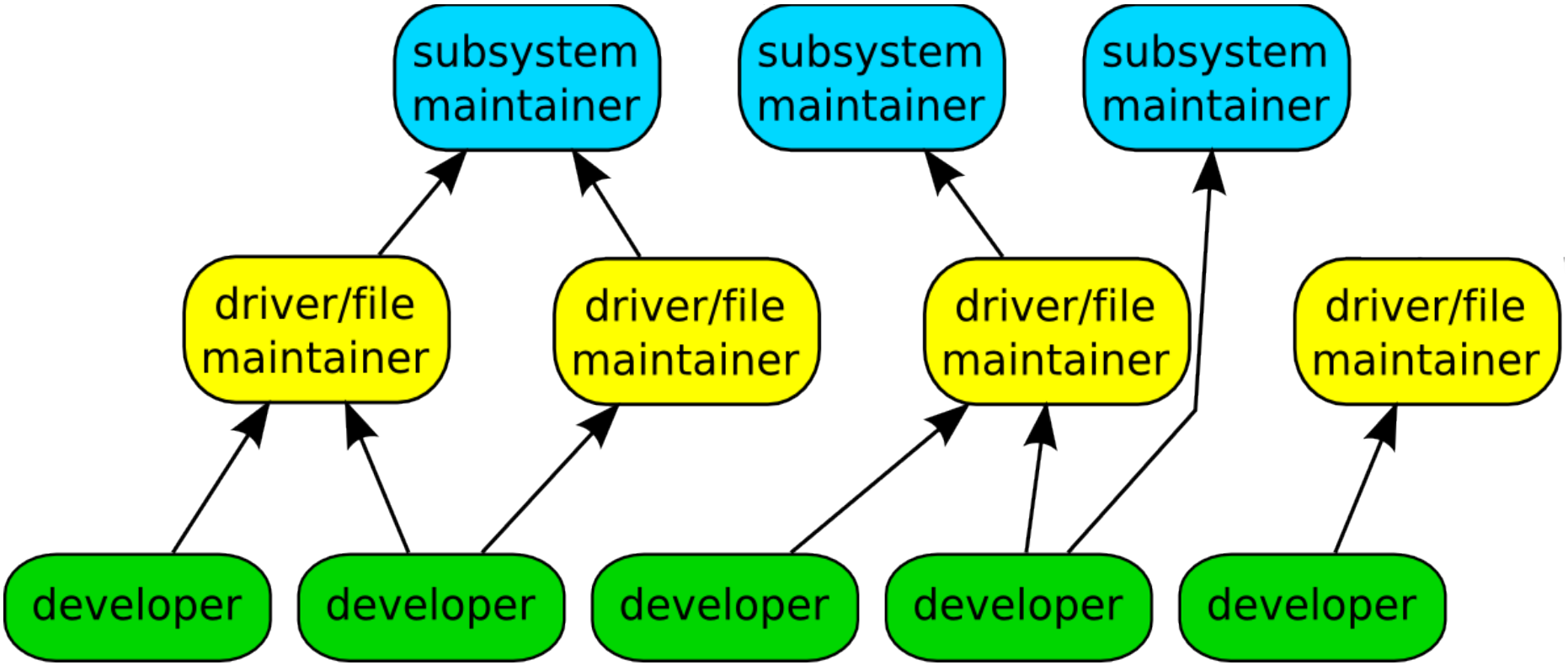
Developer's Certificate of Origin

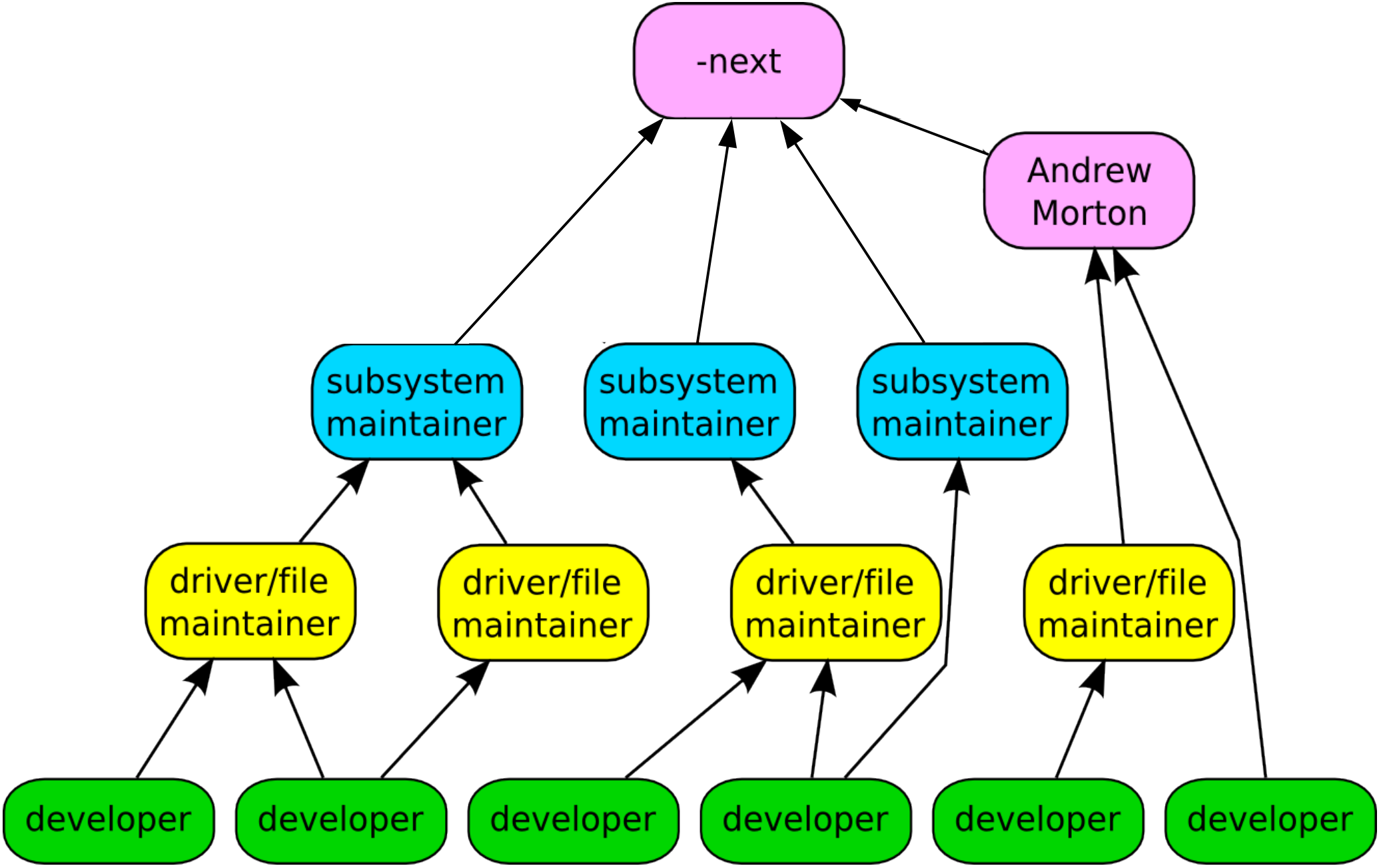
(a) I created this change; or

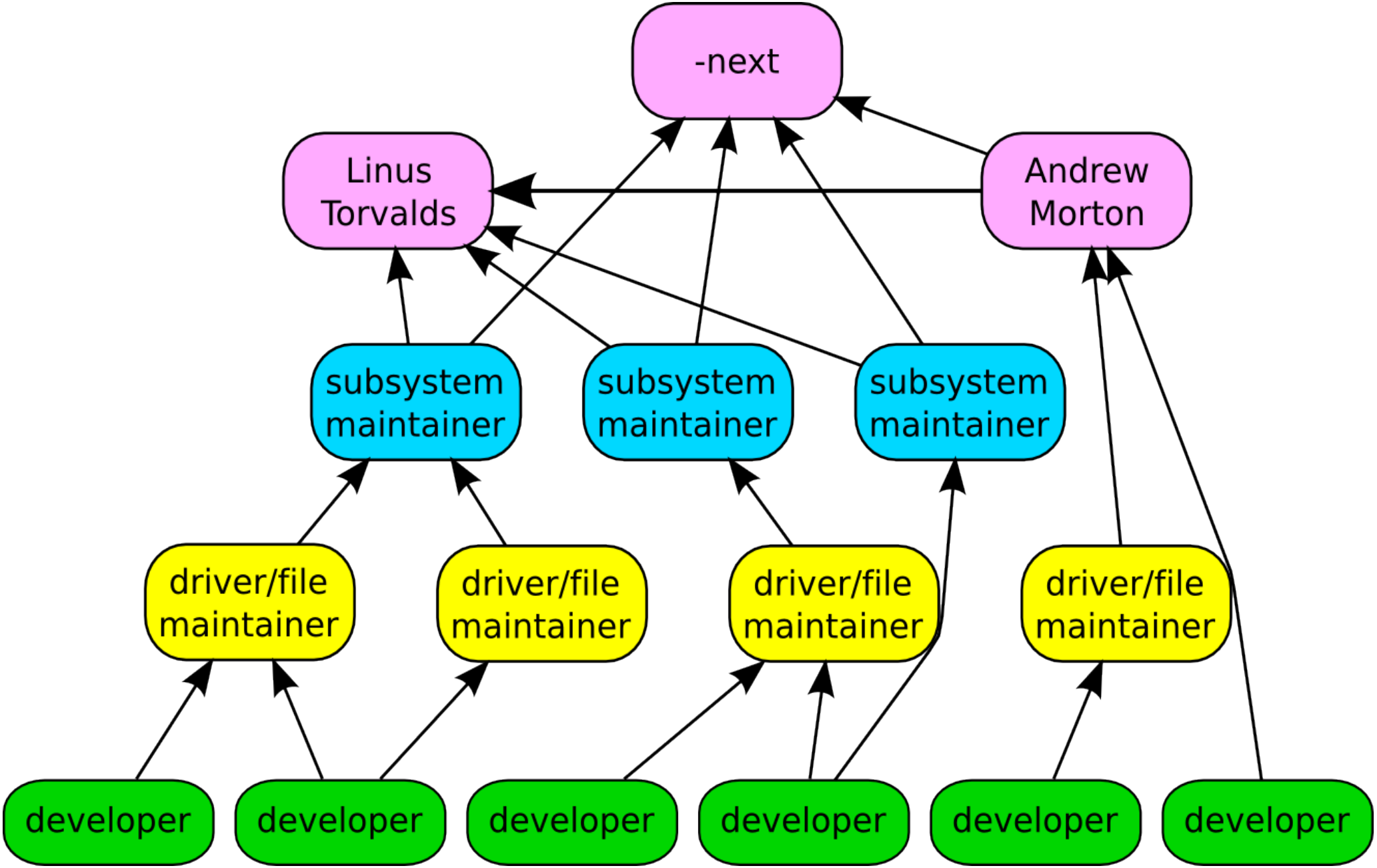
(b) Based this on a previous work with a compatible license; or

(c) Provided to me by (a), (b), or (c) and not modified

(d) This contribution is public.


subsystem maintainer
subsystem maintainer
subsystem maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
developer
developer
developer
developer
developer


-next
Andrew Morton
subsystem maintainer
subsystem maintainer
subsystem maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
developer
developer
developer
developer
developer
developer


-next
Linus Torvalds
Andrew Morton
subsystem maintainer
subsystem maintainer
subsystem maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
developer
developer
developer
developer
developer
developer

Top developers by quantity

Chris Wilson	1356
Johan Hovold	920
Mauro Carvalho Chehab	867
Arnd Bergmann	826
Christoph Hellwig	790
Arvind Yadav	710
Colin Ian King	641
Viresh Kumar	566
Greg Kroah-Hartman	507
Geert Uytterhoeven	492

Kernel releases 4.9.0 – 4.14.0

Top Signed-off-by:

David S. Miller	9430
Greg Kroah-Hartman	9342
Mauro Carvalho Chehab	3039
Mark Brown	2385
Linus Torvalds	2261
Alex Deucher	2095
Andrew Morton	2013
Ingo Molnar	1989
Martin Petersen	1681
Chris Wilson	1520

Kernel releases 4.9.0 – 4.14.0

Who is funding this work?

1. Intel	12.4%
2. “Amateurs”	8.5%
3. Red Hat	6.8%
4. Linaro	5.4%
5. IBM	3.9%
6. Google	3.4%
7. Consultants	3.3%
8. SUSE	3.1%
9. AMD	2.8%
10. Samsung	2.7%

Kernel releases 4.9.0 – 4.14.0

Who is funding this work?

11. Mellanox	2.2%
12. Renesas Electronics	2.1%
13. Oracle	1.7%
14. Broadcom	1.5%
15. ARM	1.5%
16. Huawei Technologies	1.5%
17. Texas Instruments	1.2%
18. Linux Foundation	1.1%
19. Free Electrons	1.1%
20. Canonical	1.1%

Kernel releases 4.9.0 – 4.14.0

“Working upstream saves time and money”

Dan Frye – VP Open Systems, IBM
Dirk Hohndel – Chief Technologist, Intel

How to change

How to change

Submit code early and often

How to change

Send small pieces

How to change

Ask community for feedback

How to change

Act on it

How to change

Remove legal hurdles

How to change

Force them to work in public

How to change

Allow them to be the community

Kernel Security

Kernel Security

Almost all bugs can be a “security” issue.

Kernel Security

Almost all bugs can be a “security” issue.

Fix them as soon as possible.

Kernel Security

Averaging 21 fixes per day

“If you are not using a stable / longterm kernel, your machine is insecure”

– me

“The kernel needs airbags”
– Konstantin Ryabitsev

slides.com/mricon/giant-bags-of-mostly-water#/

“We will always have bugs,
we must stop their exploitation”
– Kees Cook

outflux.net/slides/2015/ks/security.pdf

Kernel Hardening

kernsec.org/wiki/index.php/Kernel_Self_Protection_Project

Core Infrastructure Initiative

“Ceaseless change is the only constant thing in Nature.”

– John Candee Dean

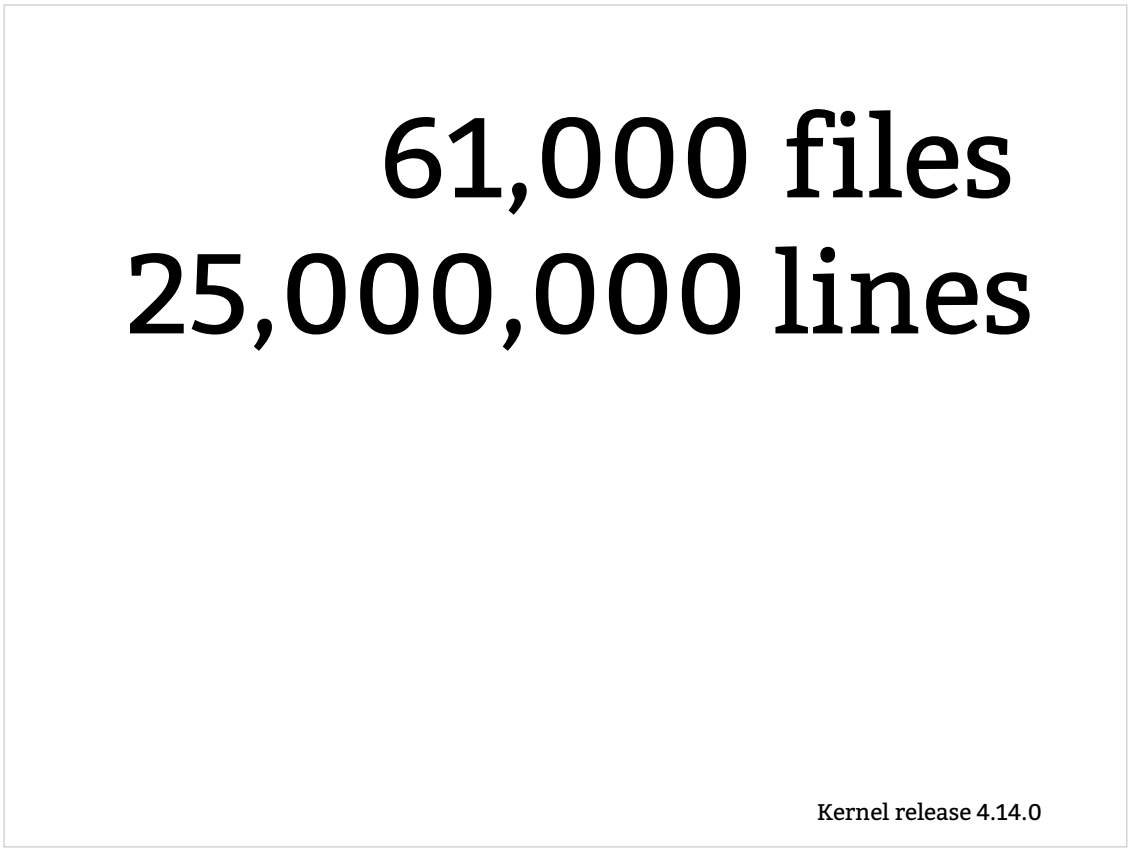
github.com/gregkh/kernel-development

Linux Kernel Development

Greg Kroah-Hartman
gregkh@linuxfoundation.org

github.com/gregkh/kernel-development

I'm going to discuss the how fast the kernel is moving, how we do it all, and how you can get involved.



This was for the 4.14 kernel release, which happened November 12, 2017.

4,394 developers
≈530 companies

Kernel releases 4.9.0 – 4.14.0
December 2016 – November 2017

This makes the Linux kernel the largest contributed body of software out there that we know of.

This is just the number of companies that we know about, there are more that we do not, and as the responses to our inquiries come in, this number will go up.

Have surpassed 400 companies for 4 years now.

10,000 lines added
2,700 lines removed
2,000 lines modified

Kernel releases 4.9.0 – 4.14.0
December 2016 – November 2017

10,000 lines added

2,700 lines removed

2,000 lines modified

Every day

Kernel releases 4.9.0 – 4.14.0
December 2016 – November 2017

8.5 changes per hour

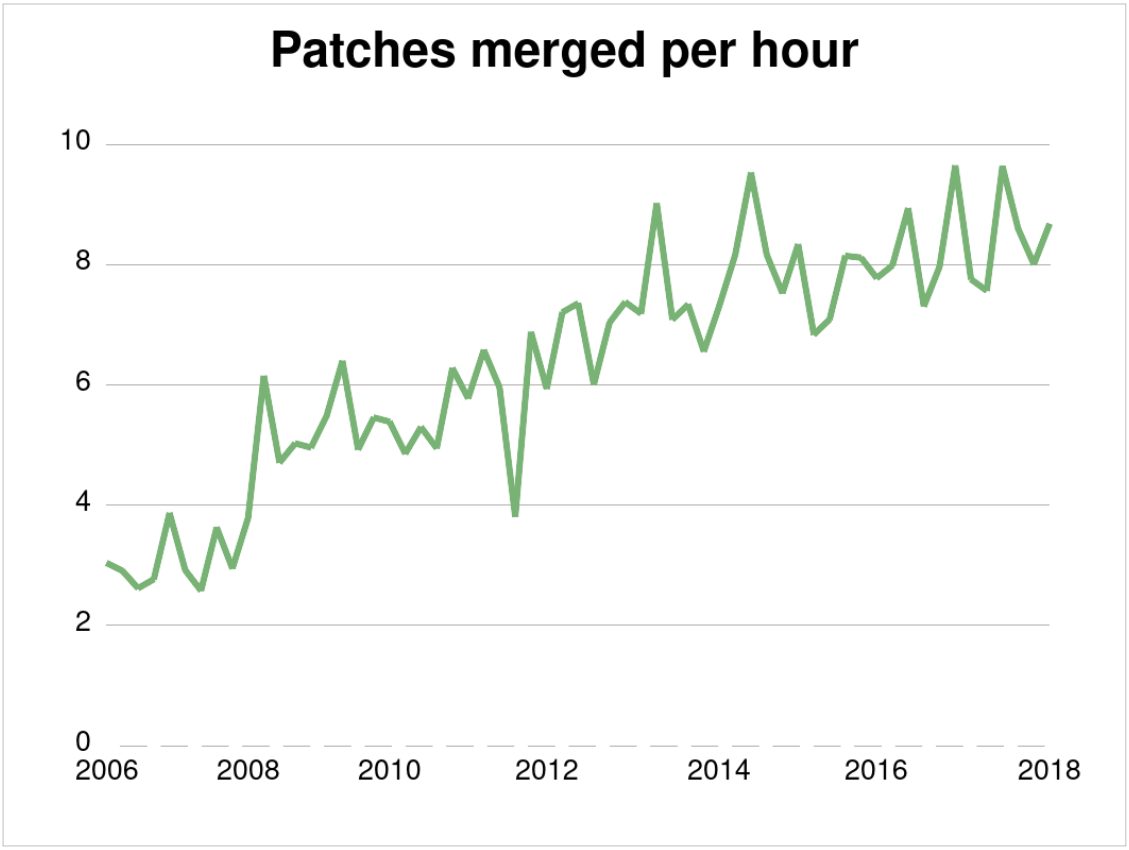
Kernel releases 4.9.0 – 4.14.0
December 2016 – November 2017

This is 24 hours a day, 7 days a week, for a full year.

We went this fast the year before this as well, this is an amazing rate of change.

Interesting note, all of these changes are all through the whole kernel.

For example, the core kernel is only 5% of the code, and 5% of the change was to the core kernel. Drivers are 55%, and 55% was done to them, it's completely proportional all across the whole kernel.



4.9 was the “largest” in number of changes that we have ever accepted, 16000. 4.12 was not far behind with 14500

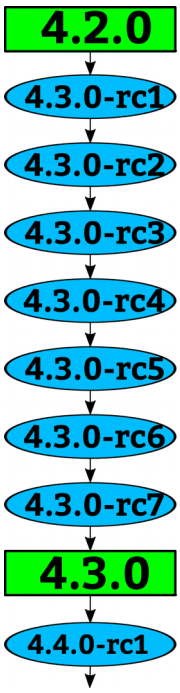
Now this is just the patches we accepted, not all of the patches that have been submitted, lots of patches are rejected, as anyone who has ever tried to submit a patch can attest to.

Over time, we are going faster, so if you stay out of the kernel tree, it takes you more work to just stay in sync with upstream

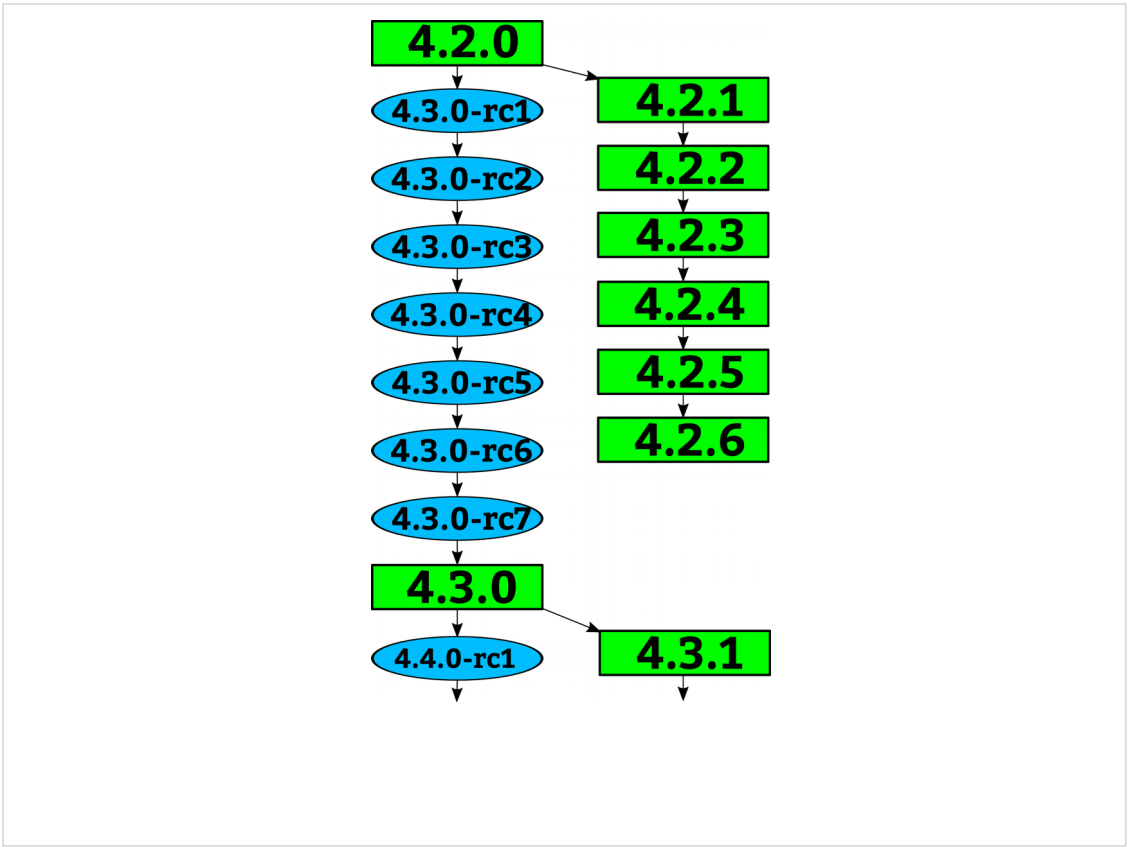
How we stay sane

Time based releases
Incremental changes

67 days to be exact, very regular experience.



How a kernel is developed.
Linus releases a stable kernel
- 2 week merge window from subsystem maintainers
- rc1 is released
- bugfixes only now
- 2 weeks later, rc2
- bugfixes and regressions
- 2 weeks later,rc3
And so on until all major bugfixes and regressions are resolved and then the cycle starts over again.



Greg takes the stable releases from Linus, and does stable releases with them, applying only fixes that are already in Linus's tree.

Requiring fixes to be in Linus's tree first ensures that there is no divergence in the development model.

After Linus releases a new stable release, the old stable series is dropped.

With the exception of “longterm” stable releases, those are special, the stick around for much longer...

“Longterm kernels”

One picked per year
Maintained for two years

4.4 4.9 4.14

I pick one kernel release per year to maintain for longer than one release cycle. This kernel I will maintain for at least 2 years.

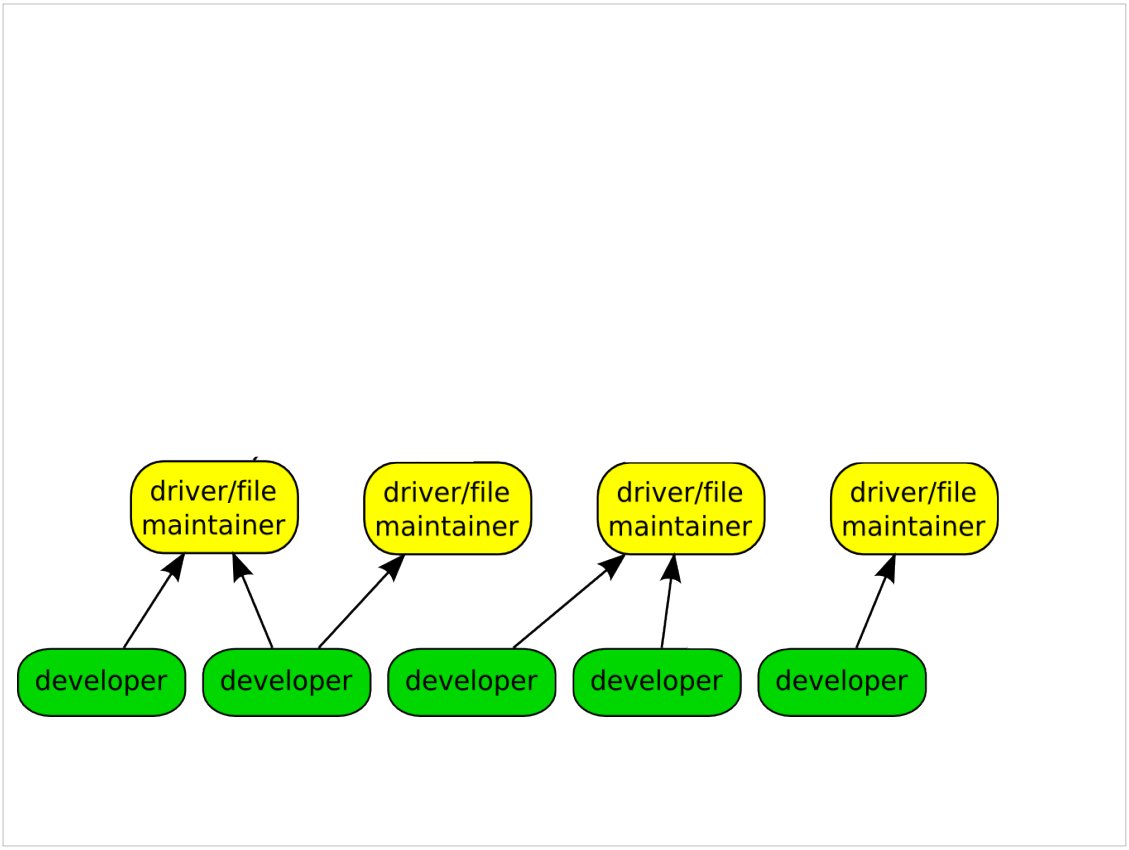
This means there are 2 longterm kernels being maintained at the same time.

4.4 and 4.9 and 4.14 are the longterm kernel releases I am currently maintaining

The LTSI project is based on the longterm kernels.

Like mentioned before, we have almost 3000 individual contributors. They all create a patch, a single change to the Linux kernel. This change could be something small, like a spelling correction, or something larger, like a whole new driver.

Every patch that is created only does one thing, and it can not break the build, complex changes to the kernel get broken up into smaller pieces.



The developers send their patch to the maintainer of the file(s) that they have modified.

We have about 700 different driver/file/subsystem maintainers

```
commit ecf85e481a716cfe07406439fdc7ba9526bbfaeb
Author:     Robert Jarzmik <robert.jarzmik@free.fr>
AuthorDate: Tue Apr 21 20:33:10 2009 -0700
Commit:     Greg Kroah-Hartman <gregkh@suse.de>
CommitDate: Thu Apr 23 14:15:31 2009 -0700

    USB: otg: Fix bug on remove path without transceiver

    In the case where a gadget driver is removed while no
    transceiver was found at probe time, a bug in
    otg_put_transceiver() will trigger.

    Signed-off-by: Robert Jarzmik <robert.jarzmik@free.fr>
    Acked-by: David Brownell <dbrownell@users.sourceforge.net>
    Signed-off-by: Greg Kroah-Hartman <gregkh@suse.de>

--- a/drivers/usb/otg/otg.c
+++ b/drivers/usb/otg/otg.c
@@ -43,7 +43,8 @@ EXPORT_SYMBOL(otg_get_transceiver);
 void otg_put_transceiver(struct otg_transceiver *x)
 {
-       put_device(x->dev);
+       if (x)
+               put_device(x->dev);
 }
```

This is an example of a patch.

It came from Robert, was acked by David, the maintainer at the time of the usb on-the-go subsystem, and then signed off by by me before it was commited to the kernel tree.

The change did one thing, it checked the value of the pointer before it was dereferenced, fixing a bug that would have crashed the kernel if it had been hit.

This is also a “blame” trail, showing who changed each line in the kernel, and who agreed with that change.

If a problem is found, these are the developers that you can ask about it.

Because of this, every line in the Linux kernel can be traced back to at least two developers who are responsible for it.

This is better than any other body of code.

Developer's Certificate of Origin

(a) I created this change; or

(b) Based this on a previous work with a compatible license; or

(c) Provided to me by (a), (b), or (c) and not modified

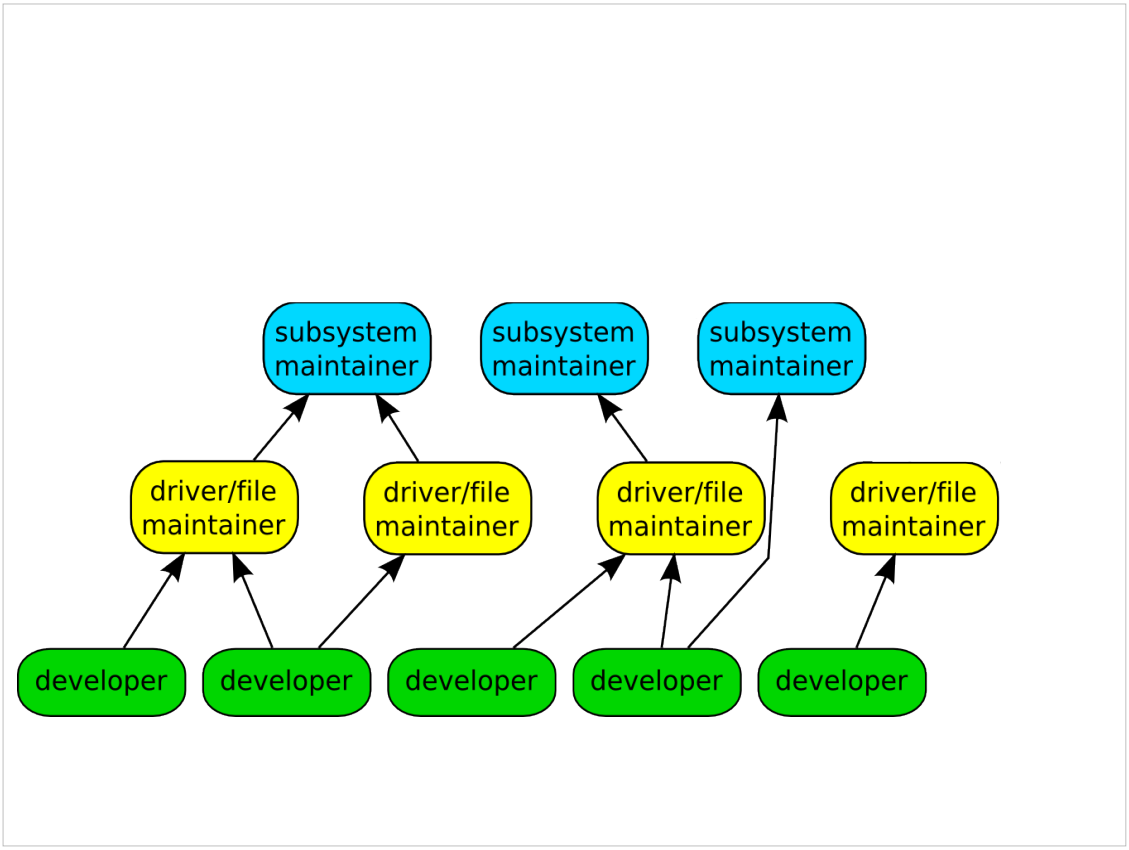
(d) This contribution is public.

This is what “Signed-off-by:” means.
All contributions to the Linux kernel have to agree to this, and every single patch has at least one signed-off-by line, usually all have at least two.

This is also a “blame” trail, showing who changed each line in the kernel, and who agreed with that change.

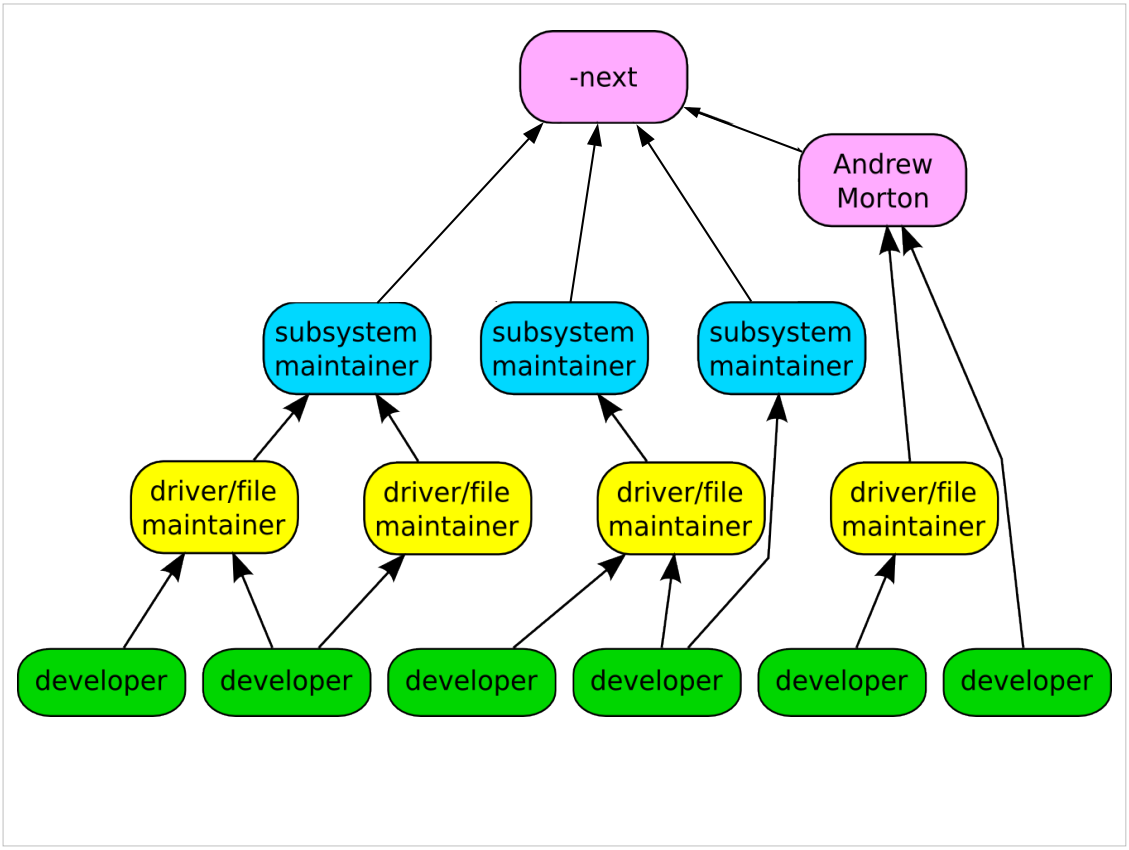
If a problem is found, this is the developers that you can ask about it.

Because of this, every line in the Linux kernel can be traced back to at least two developers who are responsible for it.
This is better than any other body of code.



After reviewing the code, and adding their own signed-off-by to the patch, the file/driver maintainer sends the patch to the subsystem maintainer responsible for that portion of the kernel.

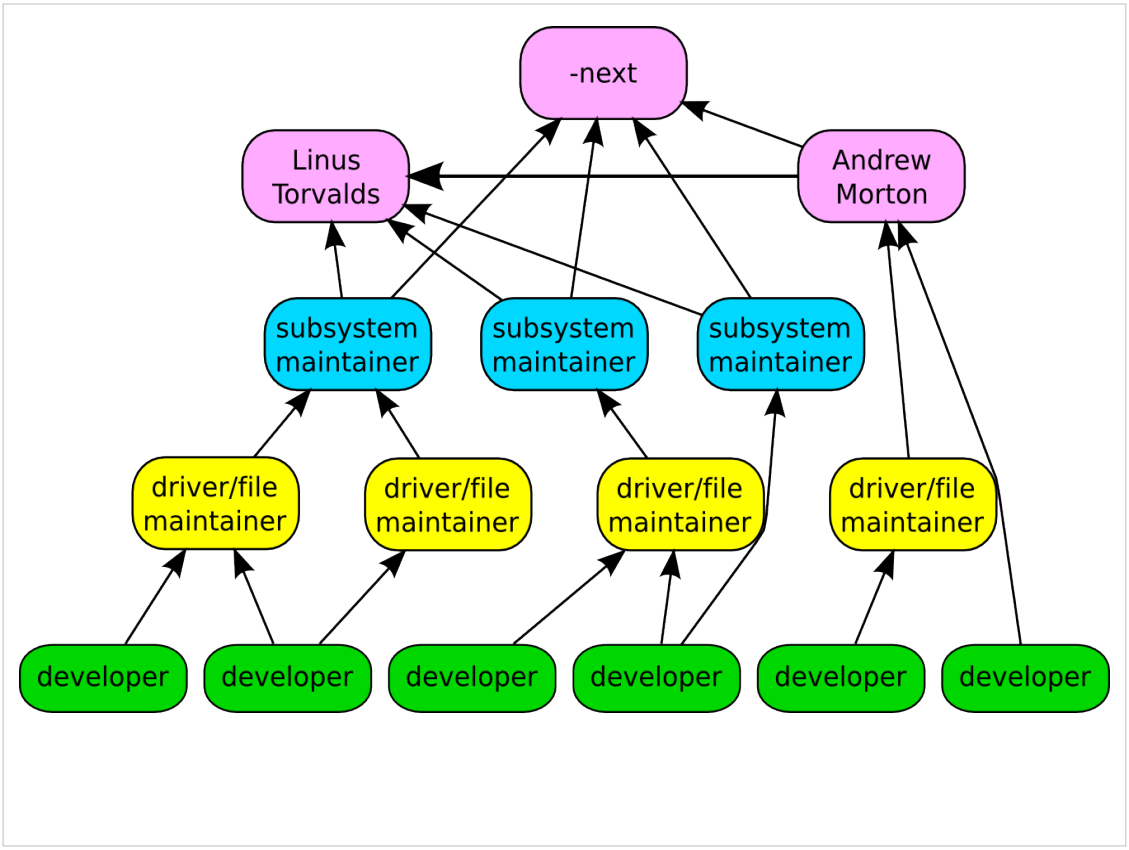
We have around 150 subsystem maintainers



Linux-next gets created every night from all of the different subsystem trees and build tested on a wide range of different platforms.

We have about 150 different trees in the linux-next release.

Andrew Morton picks up patches that cross subsystems, or are missed by others, and releases his -mm kernels every few weeks. This includes the linux-next release at that time.

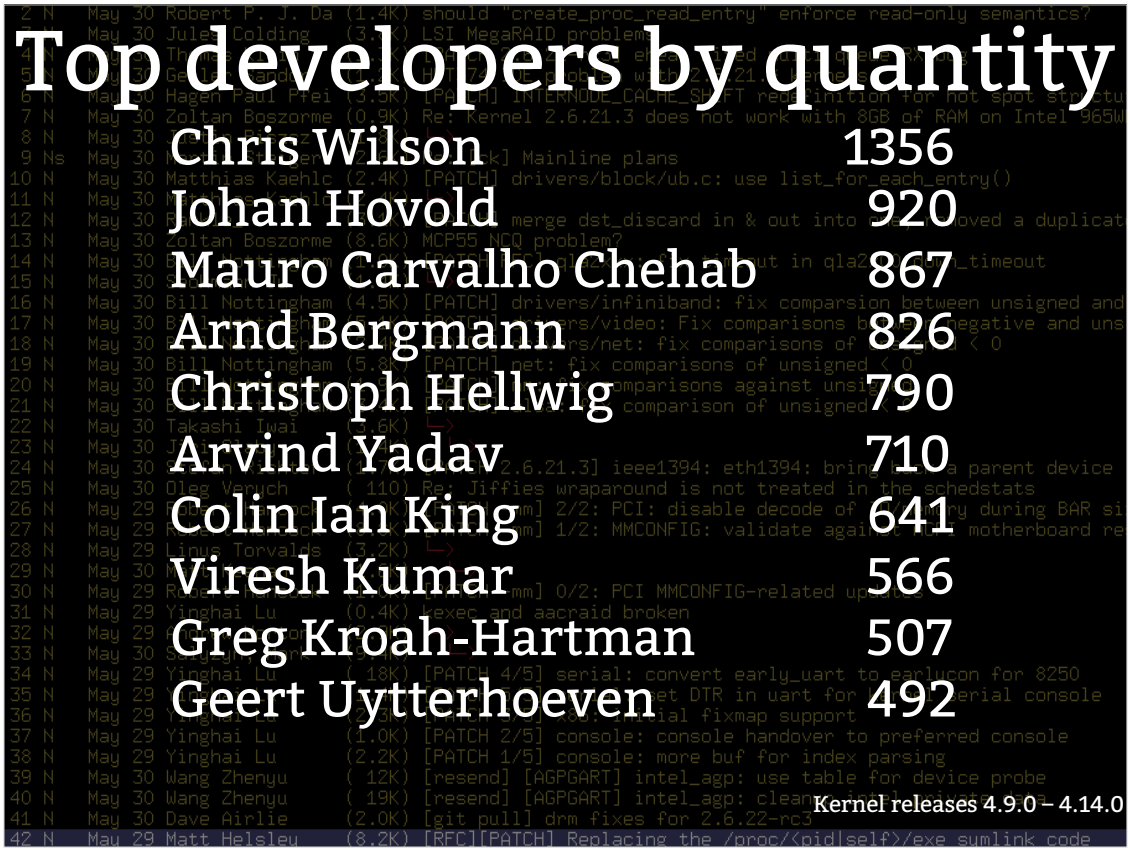


Every 3 months, when the merge window opens up, everything gets sent to Linus from the subsystem maintainers and Andrew Morton.

The merge window is 2 weeks long, and thousands of patches get merged in that short time.

All of the patches merged to Linus should have been in the linux-next release, but that isn't always the case for various reasons.

Linux-next can not just be sent to Linus as there are things in there that sometimes are not good enough to be merged just yet, it is up to the individual subsystem maintainer to decide what to merge.



Chris – intel graphics drivers
Johan – greybus, usb-serial, drivers
Mauro – Video 4 Linux (media drivers)
Arnd – janitorial cleanups and arch-generic
Viresh – greybus
Christoph – vfs, filesystems, xfs, everywhere
Colin - janitorial
Arvind - janitorial
Geert - janitorial
Greg – greybus, staging, lots of other messes

Top Signed-off-by:

David S. Miller	9430
Greg Kroah-Hartman	9342
Mauro Carvalho Chehab	3039
Mark Brown	2385
Linus Torvalds	2261
Alex Deucher	2095
Andrew Morton	2013
Ingo Molnar	1989
Martin Petersen	1681
Chris Wilson	1520

Kernel releases 4.9.0 – 4.14.0

David – networking, isa
Greg – driver core, usb, staging, greybus
Mauro – video 4 linux (media)
Mark – embedded sound
Linus – everything
Alex – radeon graphics
Andrew – everything
Ingo – x86
Martin – scsi / block
Chris – intel graphics
Kalle – wireless drivers

Who is funding this work?

1. Intel	12.4%
2. “Amateurs”	8.5%
3. Red Hat	6.8%
4. Linaro	5.4%
5. IBM	3.9%
6. Google	3.4%
7. Consultants	3.3%
8. SUSE	3.1%
9. AMD	2.8%
10. Samsung	2.7%

Kernel releases 4.9.0 – 4.14.0

So you can view this as either 12% is done by non-affiliated people, or 88% is done by companies.

Now to be fair, if you show any skill in kernel development you are instantly hired.

Why this all matters: If your company relies on Linux, and it depends on the future of Linux supporting your needs, then you either trust these other companies are developing Linux in ways that will benefit you, or you need to get involved to make sure Linux works properly for your workloads and needs.

Who is funding this work?

11. Mellanox	2.2%
12. Renesas Electronics	2.1%
13. Oracle	1.7%
14. Broadcom	1.5%
15. ARM	1.5%
16. Huawei Technologies	1.5%
17. Texas Instruments	1.2%
18. Linux Foundation	1.1%
19. Free Electrons	1.1%
20. Canonical	1.1%

Kernel releases 4.9.0 – 4.14.0

Intel – 10000 patches
Huawei - 1200 patches (almost half done by one developer!!!)
Canonical 896

Siemens – 51 patches, rank: 111

“Working upstream saves time and money”

Dan Frye – VP Open Systems, IBM
Dirk Hohndel – Chief Technologist, Intel

How to change

“Change or die”

How to change

Submit code early and often

No big code dumps.
They are hard to review, and even harder for you to modify and resend, slowing everything down and delaying any potential acceptance.

How to change

Send small pieces

How to change

Ask community for feedback

Some companies get Linux kernel community members together and discuss products and technologies directly with the senior engineers, no managers in the way.

Great feedback circle, the community gets to understand your products better, and no long explanation is needed when showing the code later, and the community gets to tell your engineers what they are doing wrong.

How to change

Act on it

Change products / roadmaps / features based on feedback. This makes Linux work better on your platforms which makes your platform better.

How to change

Remove legal hurdles

Let them contribute whatever they want and can.

Legal is to support the business, change the hurdles to be on the legal side, not the developers.

How to change

Force them to work in public

No internal mailing lists
All communication is done publicly
Let them argue in public

How to change

Allow them to be the community

Your developers will become the maintainers, driving the future of Linux forward, always keeping your products in mind as things evolve and change.

Kernel Security

Let’s talk about kernel security.

Kernel Security

Almost all bugs can be a “security” issue.

Anything that goes wrong in the kernel can usually be turned into a “security” problem.

Be it a DoS, or a reboot, or local root exploit, or worst case, a remote root exploit (very rare, thankfully.)

Kernel Security

Almost all bugs can be a “security” issue.

Fix them as soon as possible.

Because it’s really hard to determine if a bug is a “security” issue, our response is that we fix all bugs as soon as possible once we learn about them.

TTY bug in RH

Kernel Security

Averaging 21 fixes per day

If you look at the number of patches flowing into the stable tree, we are averaging 21 patches a day, every single day.

Now not all of them are “security” fixes. But some small percentage is.

This is for the latest kernel release, the 4.4 kernel is averaging 9 fixes a day, and 4.9 is still running at 13½ fixes a day!

> “If you are not using a stable / longterm kernel, your machine is insecure”
>
> – me

Your infrastructure HAS to support updating the kernel. If you can’t do that, you are insecure.

Even the “enterprise” kernels aren’t keeping up with this rate of change, the exception being Debian.

If you use these kernels, you HAVE to keep up to date.

Android example.

kernel.org sysadmin, in charge of the LF sysadmin team, Fedora infrastructure developer.

Great presentation on how you, as a sysadmin, can implement secure practices for your network. Full checklist and guide has been published.

But, even with those practices, we need low-level changes in order to save ourselves from the accidents that will happen.

We need “airbags” in the kernel, and elsewhere.

Things like SELinux, grsec, openwall we need them.

“We will always have bugs,
we must stop their exploitation”
– Kees Cook

outflux.net/slides/2015/ks/security.pdf

Kees Cook, kernel security developer, presentation at kernel summit last year.

We need to start doing things to make the kernel more “robust” from a security standpoint.

Even if it makes things harder for the developers.

Everyone agreed.

Kernel Hardening

kernsec.org/wiki/index.php/Kernel_Self_Protection_Project

Core Infrastructure Initiative

Kernel hardening project.

New security features are being added in each release, but if you don’t upgrade, you don’t get those features, and protection.

CII is helping to fund this, if you want to work on it, we need developers, and we will pay for it.

“Ceaseless change is the only constant thing in Nature.”

– John Candee Dean

1911 astronomer.

If your operating system isn’t constantly changing, then it is dead. The world doesn’t stop changing, learn to embrace the change in order to survive.

“static systems” die.

Obligatory Penguin Picture