An engineer's guide to Linux Kernel upgrades

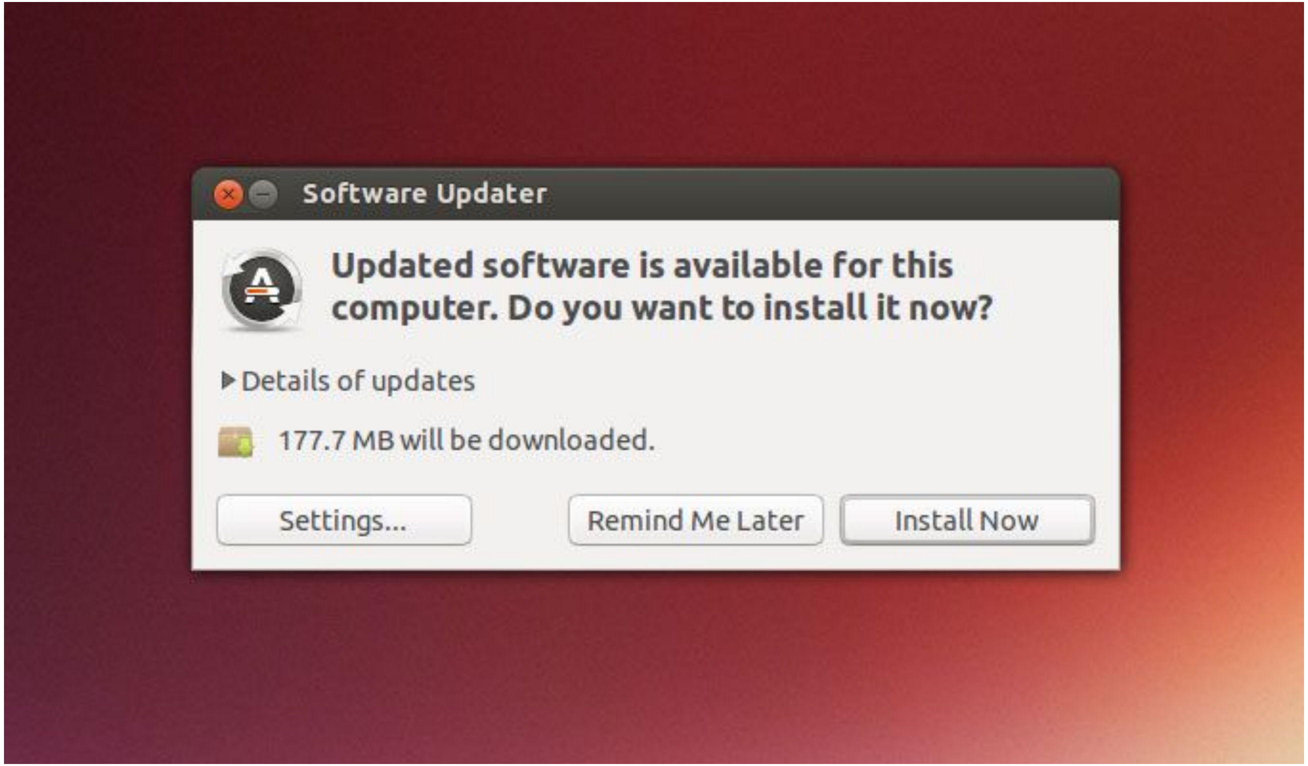
Ignat Korchagin
@ignatkn

$ whoami

- Linux team at Cloudflare
- Systems security and performance
- Low-level programming
- Linux Kernel maintainer for asymmetric keys

What do you do in this case?

Updates available!



Updated software is available for this computer. Do you want to install it now?

177.7 MB will be downloaded.

Updates available for production systems!

remind me later
disable updates
imgflip.com
JAKE-CLARK.TUMBLR

How do we perceive software updates?

Software updates perception

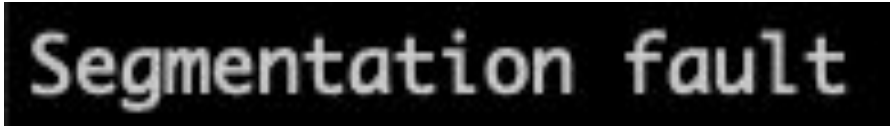
Regular software upgrades

Software updates perception

Regular software upgrades

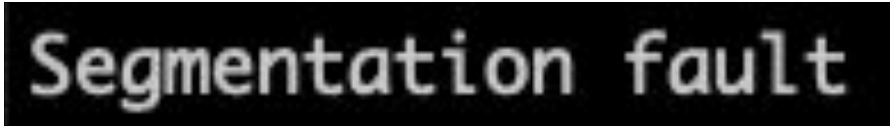
Linux Kernel upgrades

Regular software updates



```
Segmentation fault
```

Regular software updates

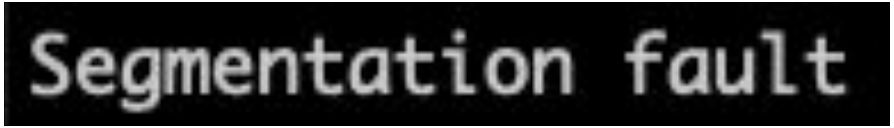


```
Segmentation fault
```

systemd service unit file

```
…
[Service]
Restart=always
…
```

Regular software updates

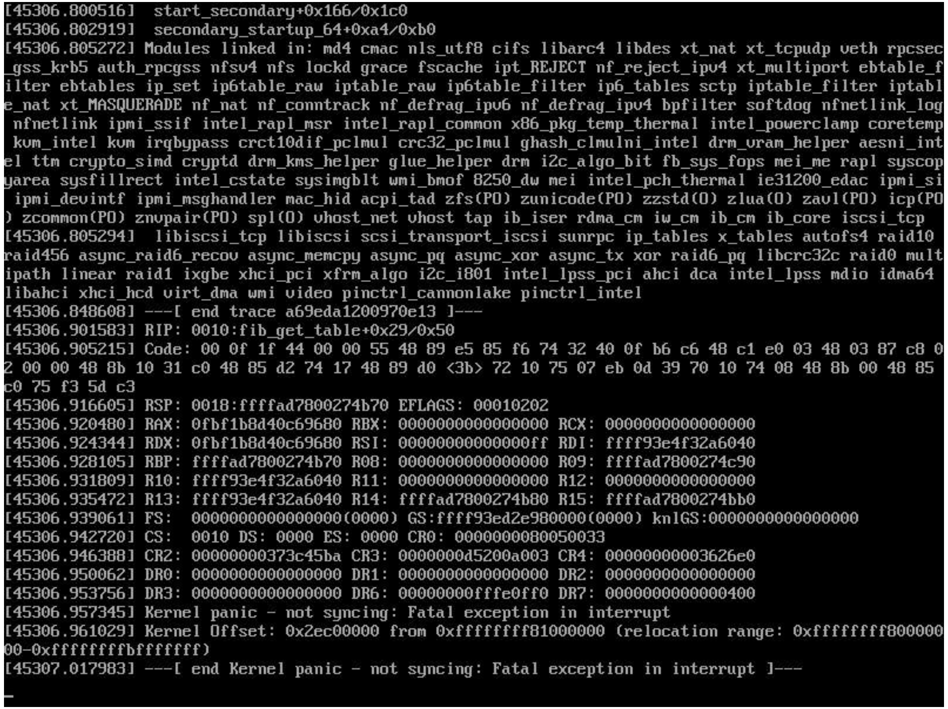


```
Segmentation fault
```

systemd service unit file

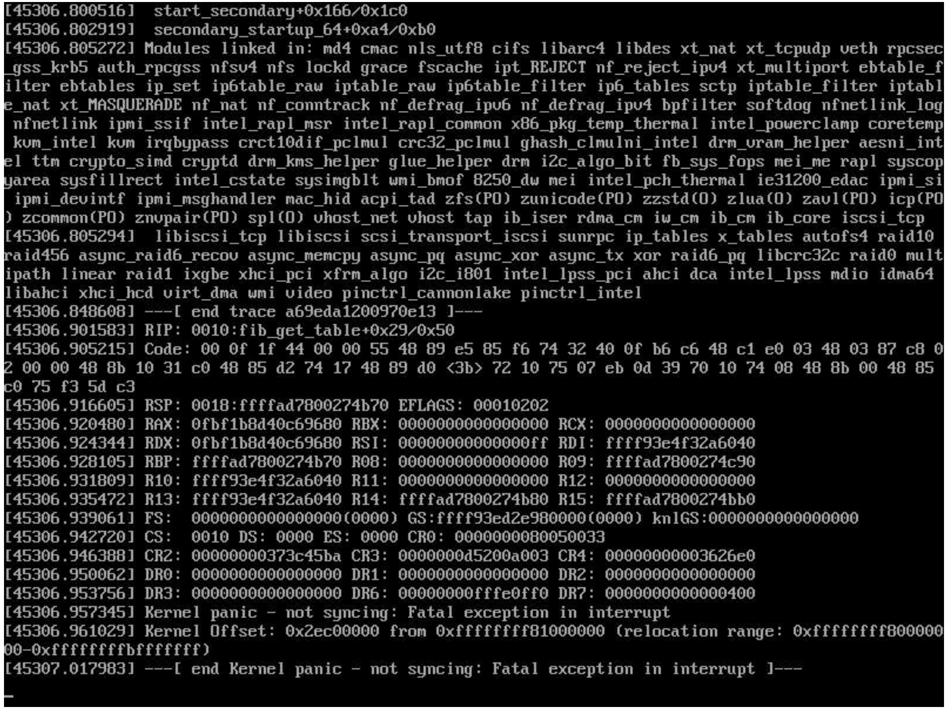
```
…
[Service]
Restart=always
…
```

Linux Kernel updates



```
[45306.800516]  start_secondary+0x166/0x1c0
[45306.802919]  secondary_startup_64+0xa4/0xb0
[45306.805272] Modules linked in: md4 cmac nls_utf8 cifs libarc4 libdes xt_nat xt_tcpudp veth rpcsec
_gss_krb5 auth_rpcgss nfsv4 nfs lockd grace fscache ipt_REJECT nf_reject_ipv4 xt_multiport ebtable_f
ilter ebtables ip_set ip6table_raw iptable_raw ip6table_filter ip6_tables sctp iptable_filter iptabl
e_nat xt_MASQUERADE nf_nat nf_conntrack nf_defrag_ipv6 nf_defrag_ipv4 bpfilter softdog nfnetlink_log
 nfnetlink ipmi_ssif intel_rapl_msr intel_rapl_common x86_pkg_temp_thermal intel_powerclamp coretemp
 kvm_intel kvm irqbypass crct10dif_pclmul crc32_pclmul ghash_clmulni_intel drm_vram_helper aesni_int
el ttm crypto_simd cryptd drm_kms_helper glue_helper drm i2c_algo_bit fb_sys_fops mei_me rapl syscop
yarea sysfillrect intel_cstate sysimgblt wmi_bmof 8250_dw mei intel_pch_thermal ie31200_edac ipmi_si
 ipmi_devintf ipmi_msghandler mac_hid acpi_tad zfs(PO) zunicode(PO) zzstd(O) zlua(O) zavl(PO) icp(PO
) zcommon(PO) znvpair(PO) spl(O) vhost_net vhost tap ib_iser rdma_cm iw_cm ib_cm ib_core iscsi_tcp
[45306.805294]  libiscsi_tcp libiscsi scsi_transport_iscsi sunrpc ip_tables x_tables autofs4 raid10
raid456 async_raid6_recov async_memcpy async_pq async_xor async_tx xor raid6_pq libcrc32c raid0 mult
ipath linear raid1 ixgbe xhci_pci xfrm_algo i2c_i801 intel_lpss_pci ahci dca intel_lpss mdio idma64
libahci xhci_hcd virt_dma wmi video pinctrl_cannonlake pinctrl_intel
[45306.848608] ---[ end trace a69eda1200970e13 ]---
[45306.901583] RIP: 0010:fib_get_table+0x29/0x50
[45306.905215] Code: 00 0f 1f 44 00 00 55 48 89 e5 85 f6 74 32 40 0f b6 c6 48 c1 e0 03 48 03 87 c8 0
2 00 00 48 8b 10 31 c0 48 85 d2 74 17 48 89 d0 <3b> 72 10 75 07 eb 0d 39 70 10 74 08 48 8b 00 48 85
c0 75 f3 5d c3
[45306.916605] RSP: 0018:ffffad7800274b70 EFLAGS: 00010202
[45306.920480] RAX: 0fbf1b8d40c69680 RBX: 0000000000000000 RCX: 0000000000000000
[45306.924344] RDX: 0fbf1b8d40c69680 RSI: 00000000000000ff RDI: ffff93e4f32a6040
[45306.928105] RBP: ffffad7800274b70 R08: 0000000000000000 R09: ffffad7800274c90
[45306.931809] R10: ffff93e4f32a6040 R11: 0000000000000000 R12: 0000000000000000
[45306.935472] R13: ffff93e4f32a6040 R14: ffffad7800274b80 R15: ffffad7800274bb0
[45306.939061] FS:  0000000000000000(0000) GS:ffff93ed2e980000(0000) knlGS:0000000000000000
[45306.942720] CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[45306.946388] CR2: 00000000373c45ba CR3: 0000000d5200a003 CR4: 00000000003626e0
[45306.950062] DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000
[45306.953756] DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400
[45306.957345] Kernel panic - not syncing: Fatal exception in interrupt
[45306.961029] Kernel Offset: 0x2ec00000 from 0xffffffff81000000 (relocation range: 0xffffffff800000
00-0xffffffffbfffffff)
[45307.017983] ---[ end Kernel panic - not syncing: Fatal exception in interrupt ]---
_
```

Linux Kernel updates



```
[45306.800516]  start_secondary+0x166/0x1c0
[45306.802919]  secondary_startup_64+0xa4/0xb0
[45306.805272] Modules linked in: md4 cmac nls_utf8 cifs libarc4 libdes xt_nat xt_tcpudp veth rpcsec
_gss_krb5 auth_rpcgss nfsv4 nfs lockd grace fscache ipt_REJECT nf_reject_ipv4 xt_multiport ebtable_f
ilter ebtables ip_set ip6table_raw iptable_raw ip6table_filter ip6_tables sctp iptable_filter iptabl
e_nat xt_MASQUERADE nf_nat nf_conntrack nf_defrag_ipv6 nf_defrag_ipv4 bpfilter softdog nfnetlink_log
 nfnetlink ipmi_ssif intel_rapl_msr intel_rapl_common x86_pkg_temp_thermal intel_powerclamp coretemp
 kvm_intel kvm irqbypass crct10dif_pclmul crc32_pclmul ghash_clmulni_intel drm_vram_helper aesni_int
el ttm crypto_simd cryptd drm_kms_helper glue_helper drm i2c_algo_bit fb_sys_fops mei_me rapl syscop
yarea sysfillrect intel_cstate sysimgblt wmi_bmof 8250_dw mei intel_pch_thermal ie31200_edac ipmi_si
 ipmi_devintf ipmi_msghandler mac_hid acpi_tad zfs(PO) zunicode(PO) zzstd(O) zlua(O) zavl(PO) icp(PO
) zcommon(PO) znvpair(PO) spl(O) vhost_net vhost tap ib_iser rdma_cm iw_cm ib_cm ib_core iscsi_tcp
[45306.805294]  libiscsi_tcp libiscsi scsi_transport_iscsi sunrpc ip_tables x_tables autofs4 raid10
raid456 async_raid6_recov async_memcpy async_pq async_xor async_tx xor raid6_pq libcrc32c raid0 mult
ipath linear raid1 ixgbe xhci_pci xfrm_algo i2c_i801 intel_lpss_pci ahci dca intel_lpss mdio idma64
libahci xhci_hcd virt_dma wmi video pinctrl_cannonlake pinctrl_intel
[45306.848608] ---[ end trace a69eda1200970e13 ]---
[45306.901583] RIP: 0010:fib_get_table+0x29/0x50
[45306.905215] Code: 00 0f 1f 44 00 00 55 48 89 e5 85 f6 74 32 40 0f b6 c6 48 c1 e0 03 48 03 87 c8 0
2 00 00 48 8b 10 31 c0 48 85 d2 74 17 48 89 d0 <3b> 72 10 75 07 eb 0d 39 70 10 74 08 48 8b 00 48 85
c0 75 f3 5d c3
[45306.916605] RSP: 0018:ffffad7800274b70 EFLAGS: 00010202
[45306.920480] RAX: 0fbf1b8d40c69680 RBX: 0000000000000000 RCX: 0000000000000000
[45306.924344] RDX: 0fbf1b8d40c69680 RSI: 00000000000000ff RDI: ffff93e4f32a6040
[45306.928105] RBP: ffffad7800274b70 R08: 0000000000000000 R09: ffffad7800274c90
[45306.931809] R10: ffff93e4f32a6040 R11: 0000000000000000 R12: 0000000000000000
[45306.935472] R13: ffff93e4f32a6040 R14: ffffad7800274b80 R15: ffffad7800274bb0
[45306.939061] FS:  0000000000000000(0000) GS:ffff93ed2e980000(0000) knlGS:0000000000000000
[45306.942720] CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[45306.946388] CR2: 00000000373c45ba CR3: 0000000d5200a003 CR4: 00000000003626e0
[45306.950062] DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000
[45306.953756] DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400
[45306.957345] Kernel panic - not syncing: Fatal exception in interrupt
[45306.961029] Kernel Offset: 0x2ec00000 from 0xffffffff81000000 (relocation range: 0xffffffff800000
00-0xffffffffbfffffff)
[45307.017983] ---[ end Kernel panic - not syncing: Fatal exception in interrupt ]---
_
```

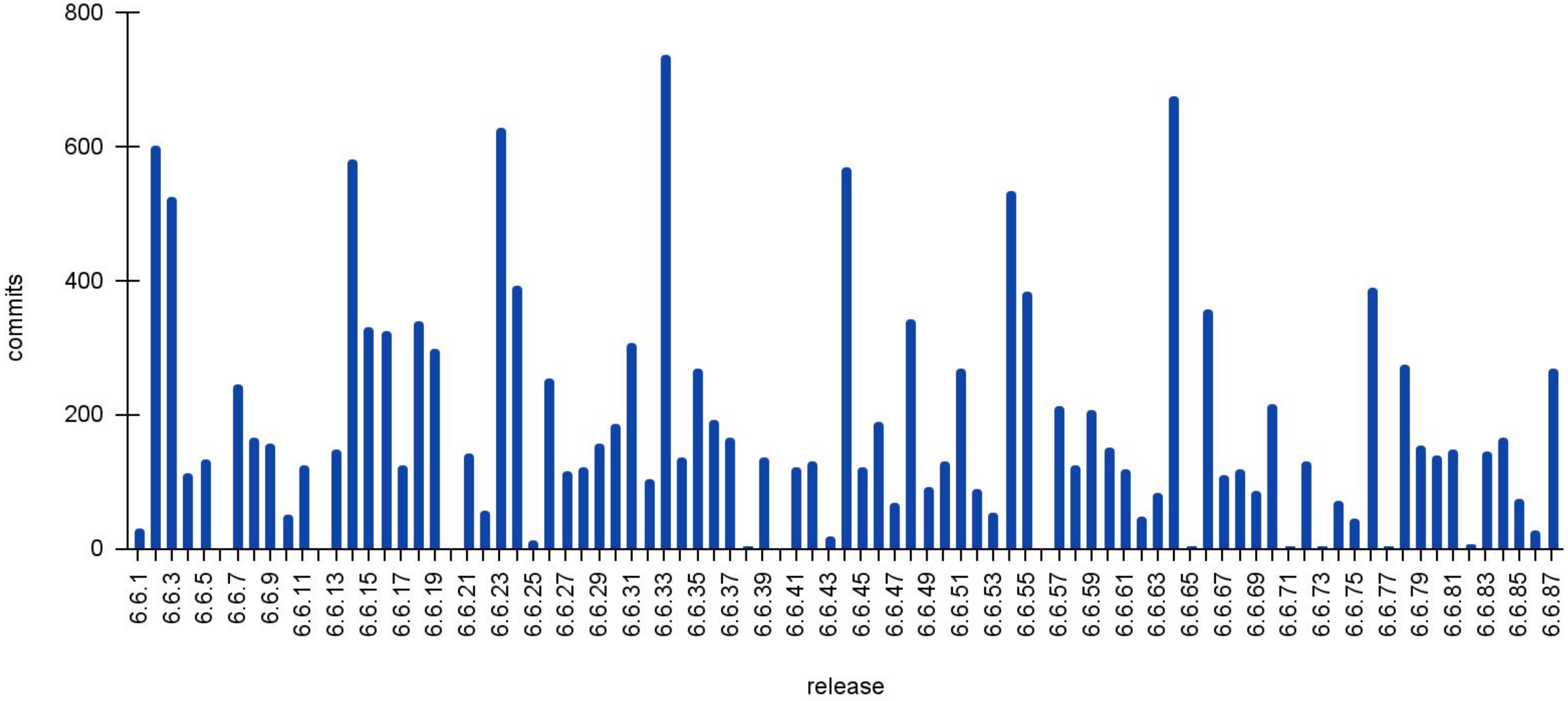
Common risks of not applying software updates

And Linux Kernel in particular

Bugs are not getting fixed

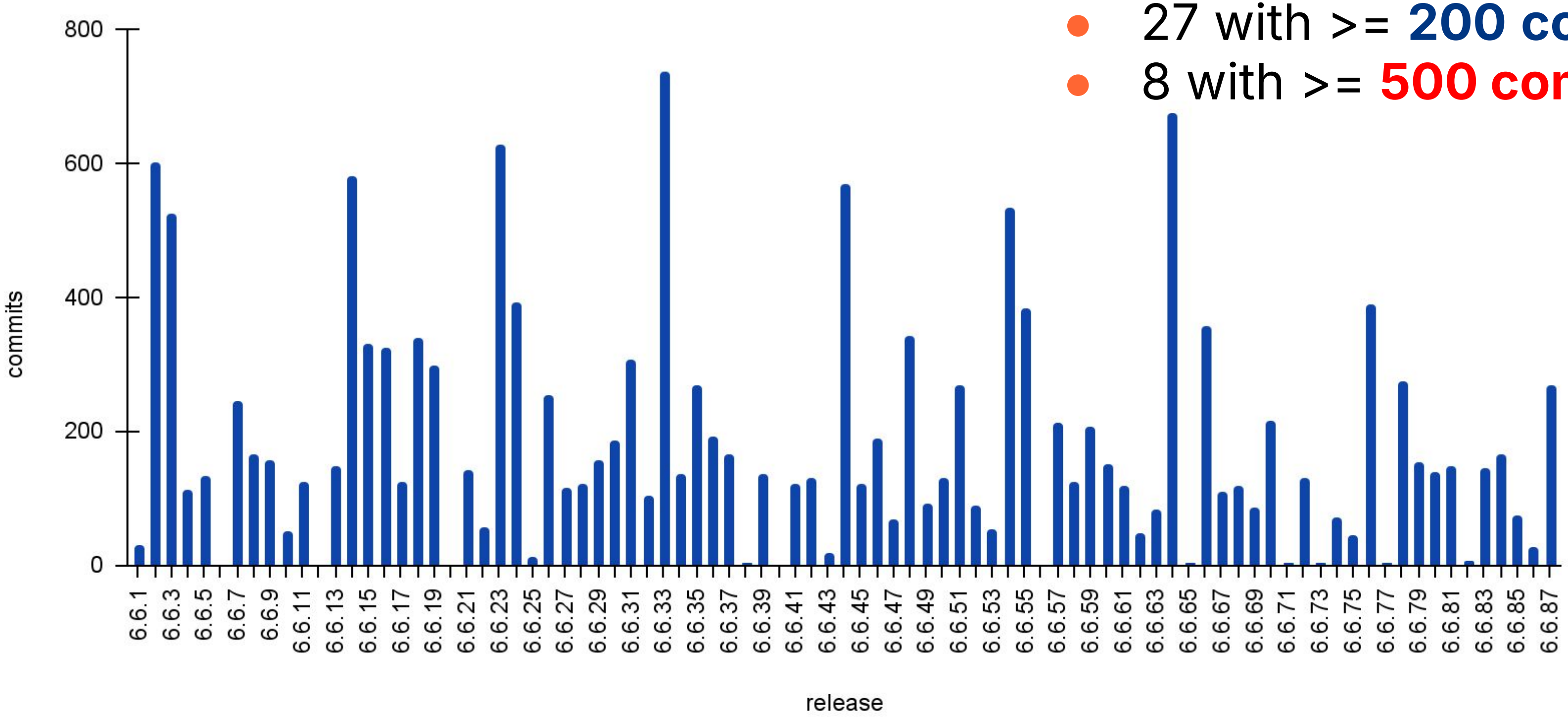
Bugs are not getting fixed

Commits per release for 6.6.x branch



Bugs are not getting fixed

Commits per release for 6.6.x branch



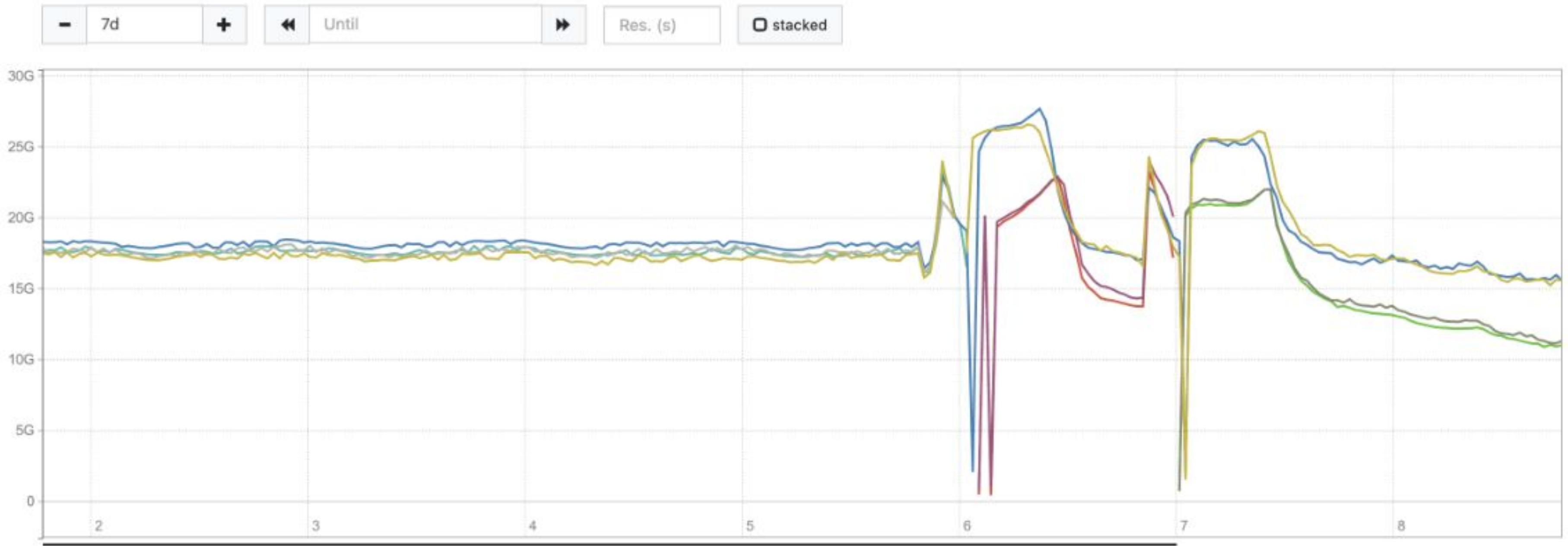
Out of 87 releases:

- 60 with >= **100 commits**
- 27 with >= **200 commits**
- 8 with >= **500 commits**

Missing out on performance improvements

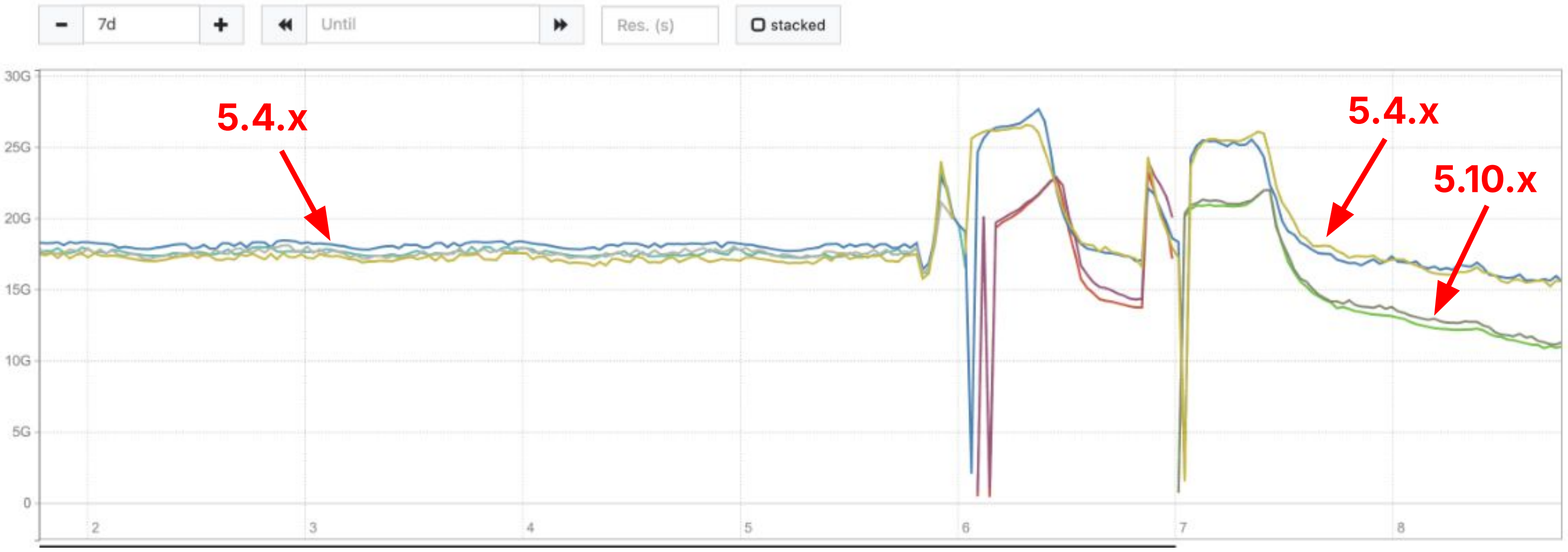
Missing out on performance improvements

Linux 5.4 to 5.10 migration



Missing out on performance improvements

Linux 5.4 to 5.10 migration



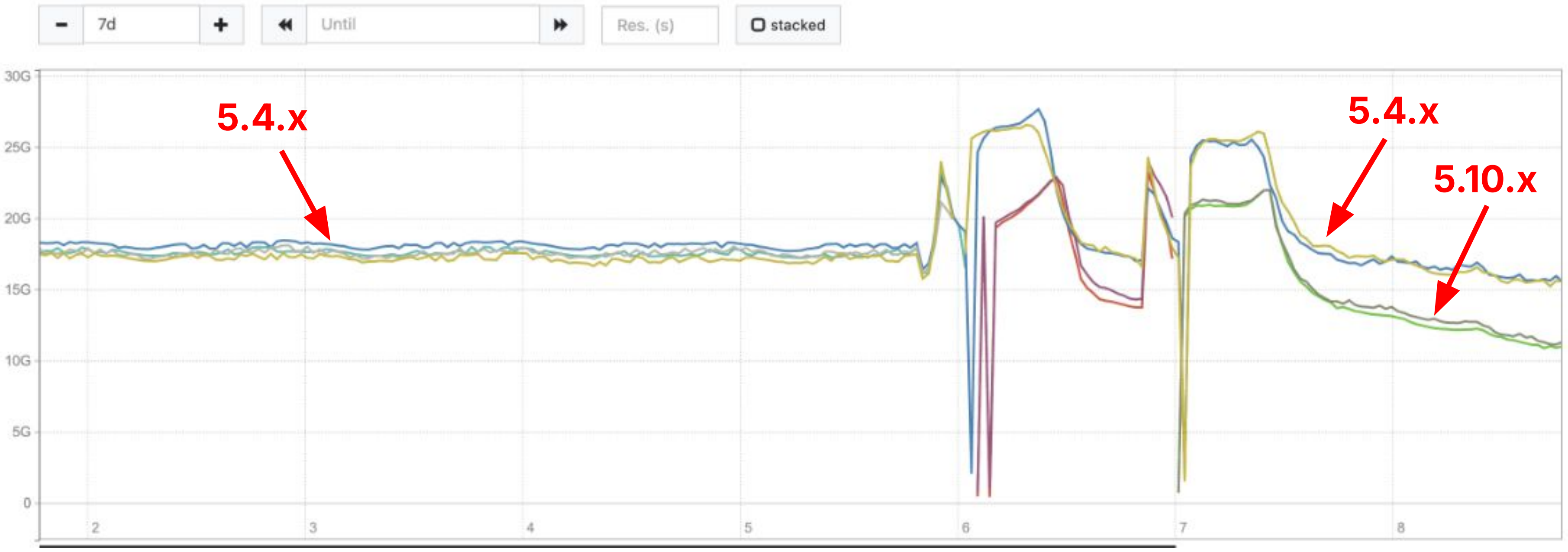
5.4.x

5.4.x

5.10.x

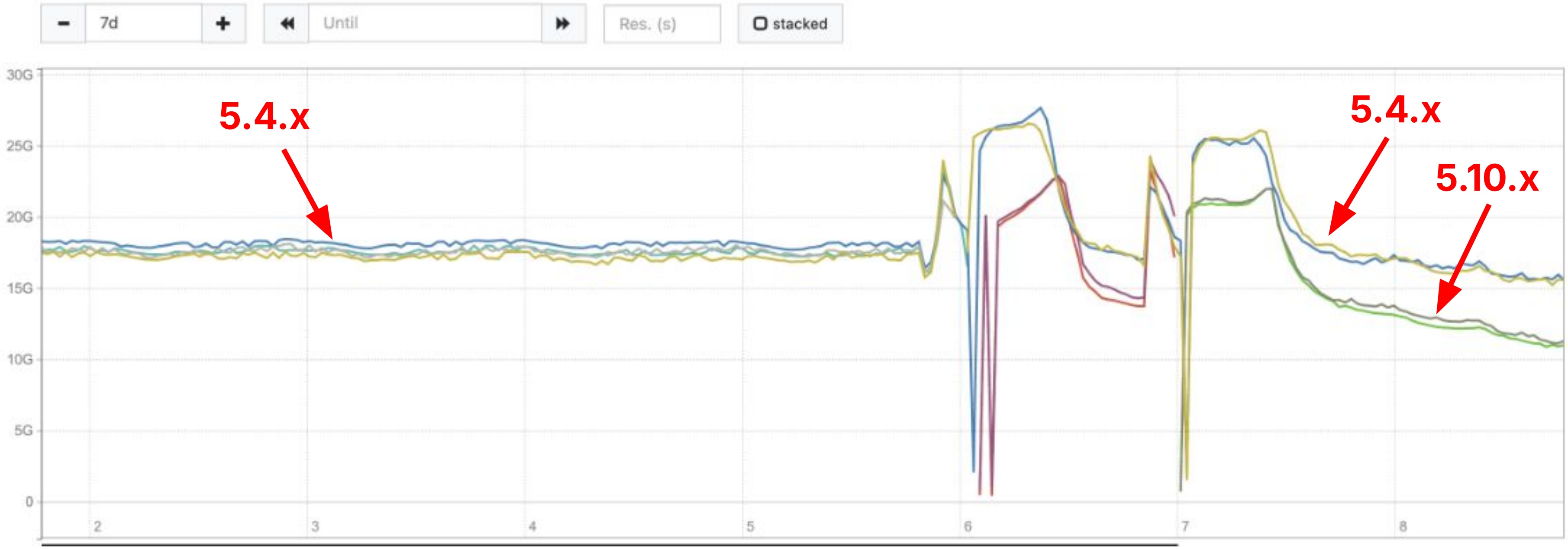
Missing out on performance improvements

Linux 5.4 to 5.10 migration: **saved ~4.5 GiB of RAM per server**



Missing out on performance improvements

Linux 5.4 to 5.10 migration: **saved ~4.5 GiB of RAM per server**



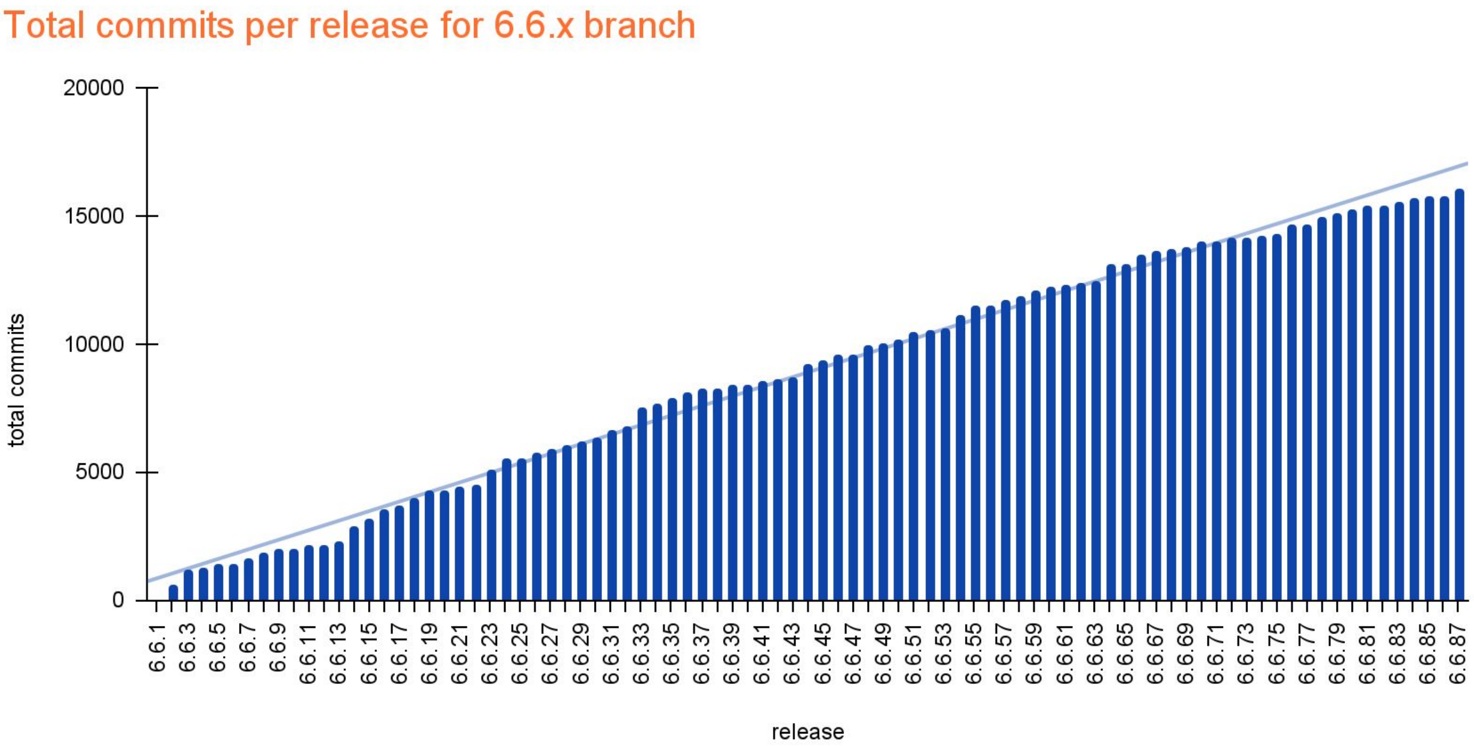
5.4.x

5.4.x

5.10.x

https://patchwork.kernel.org/project/linux-mm/cover/20191018002820.307763-1-guro@fb.com/

Accumulating change delta

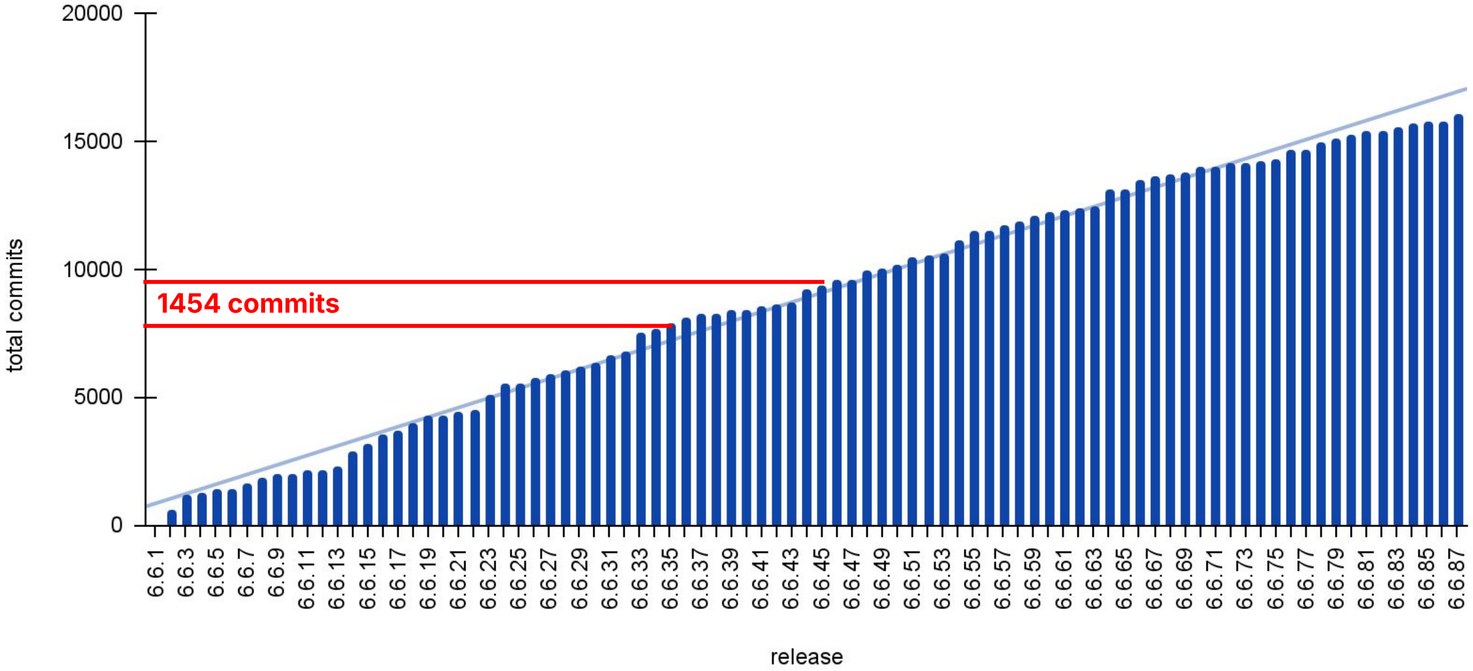


Accumulating change delta

Change delta (risk):

- 6.6.35 vs 6.6.45: 1454

Total commits per release for 6.6.x branch

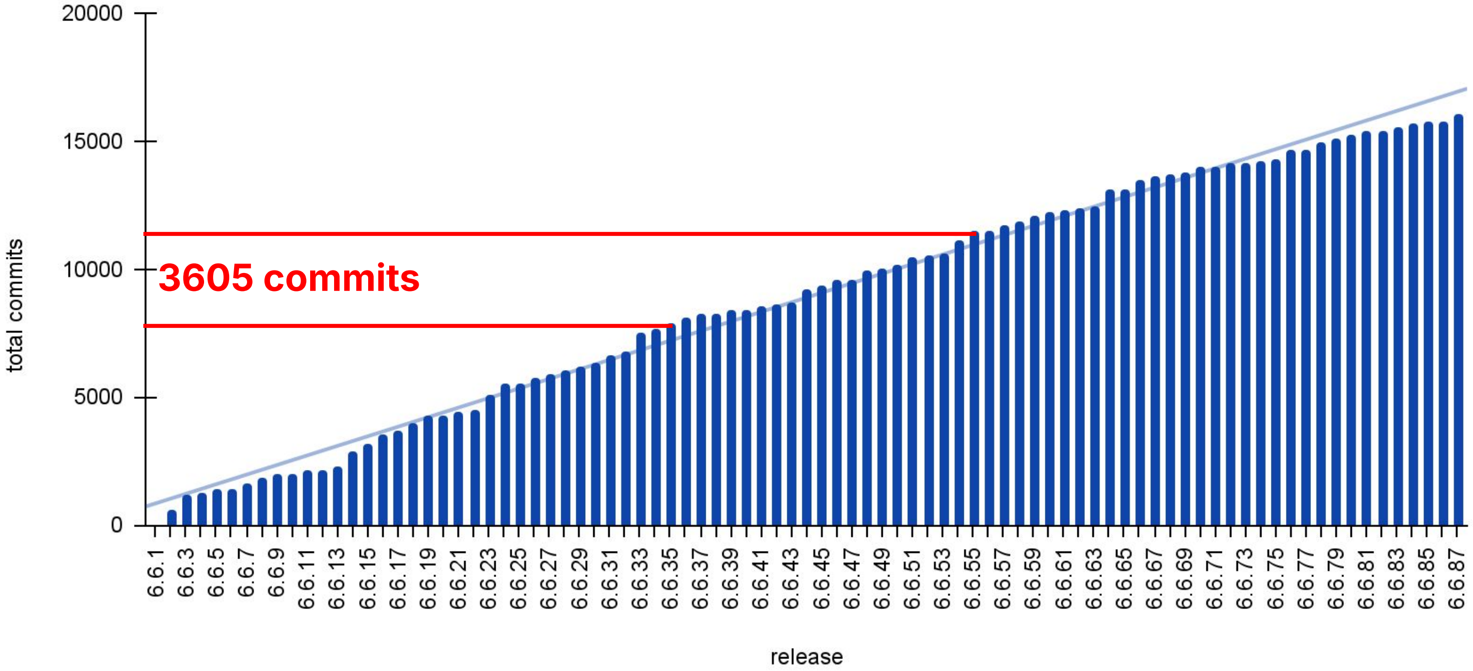


Accumulating change delta

Change delta (risk):

- 6.6.35 vs 6.6.45: 1454
- 6.6.35 vs 6.6.55: 3605

Total commits per release for 6.6.x branch

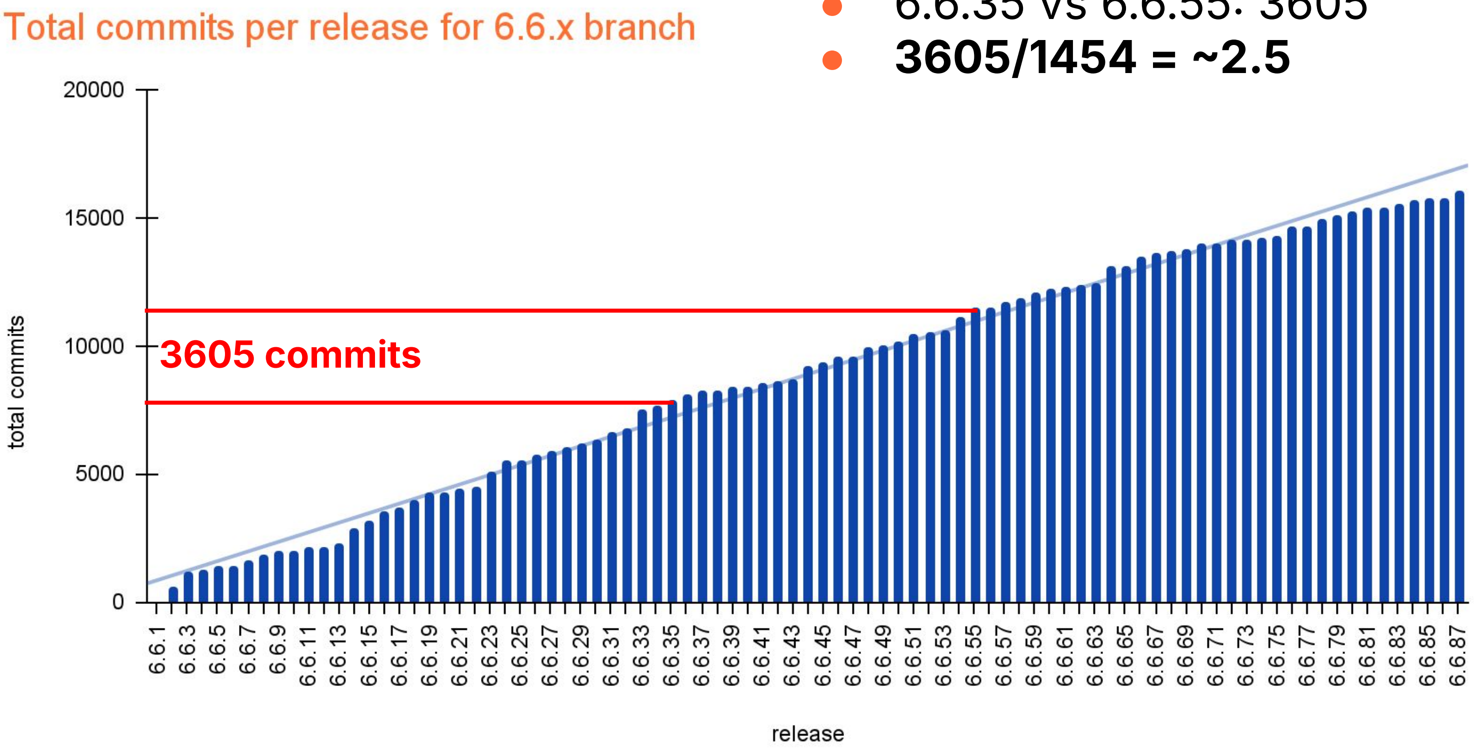


3605 commits

Accumulating change delta

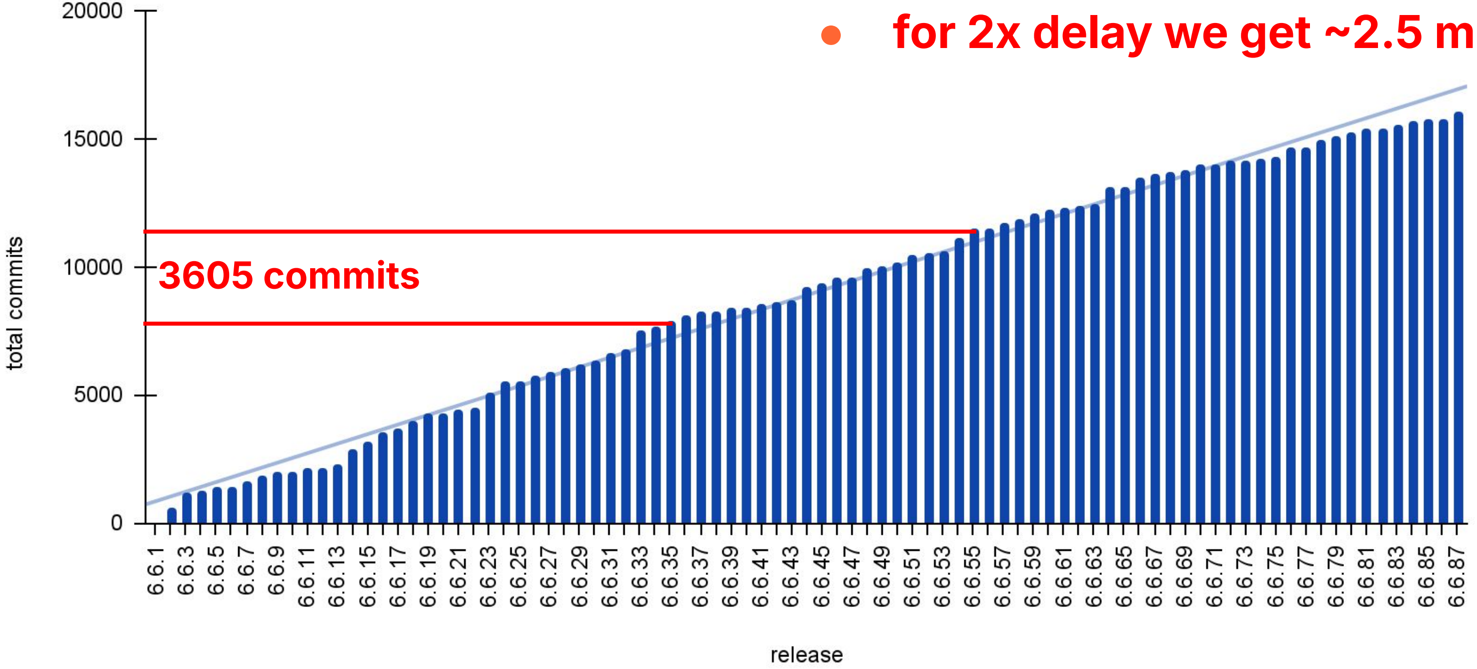
Change delta (risk):

- 6.6.35 vs 6.6.45: 1454
- 6.6.35 vs 6.6.55: 3605
- **3605/1454 = ~2.5**



Accumulating change delta

Total commits per release for 6.6.x branch



3605 commits

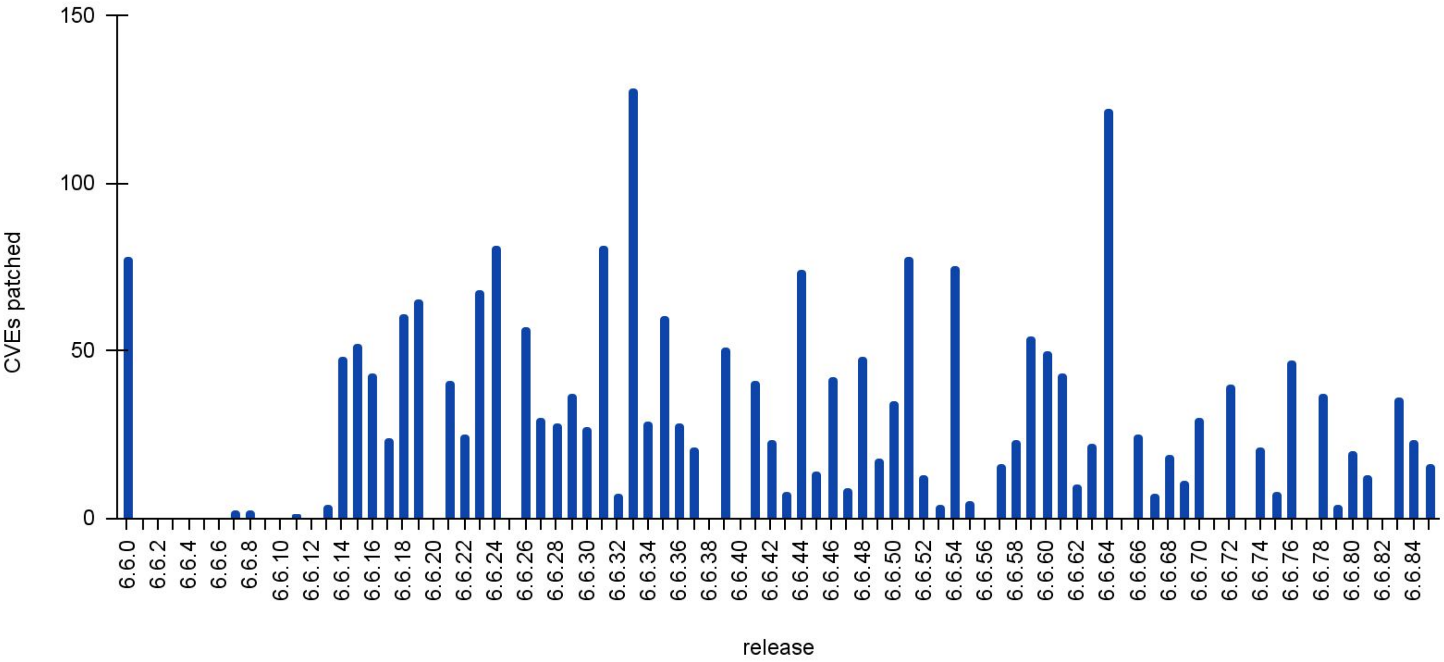
Change delta (risk):

- 6.6.35 vs 6.6.45: 1454
- 6.6.35 vs 6.6.55: 3605
- **3605/1454 = ~2.5**
- **for 2x delay we get ~2.5 more risk!**

Security vulnerabilities are not getting fixed

Security vulnerabilities are not getting fixed

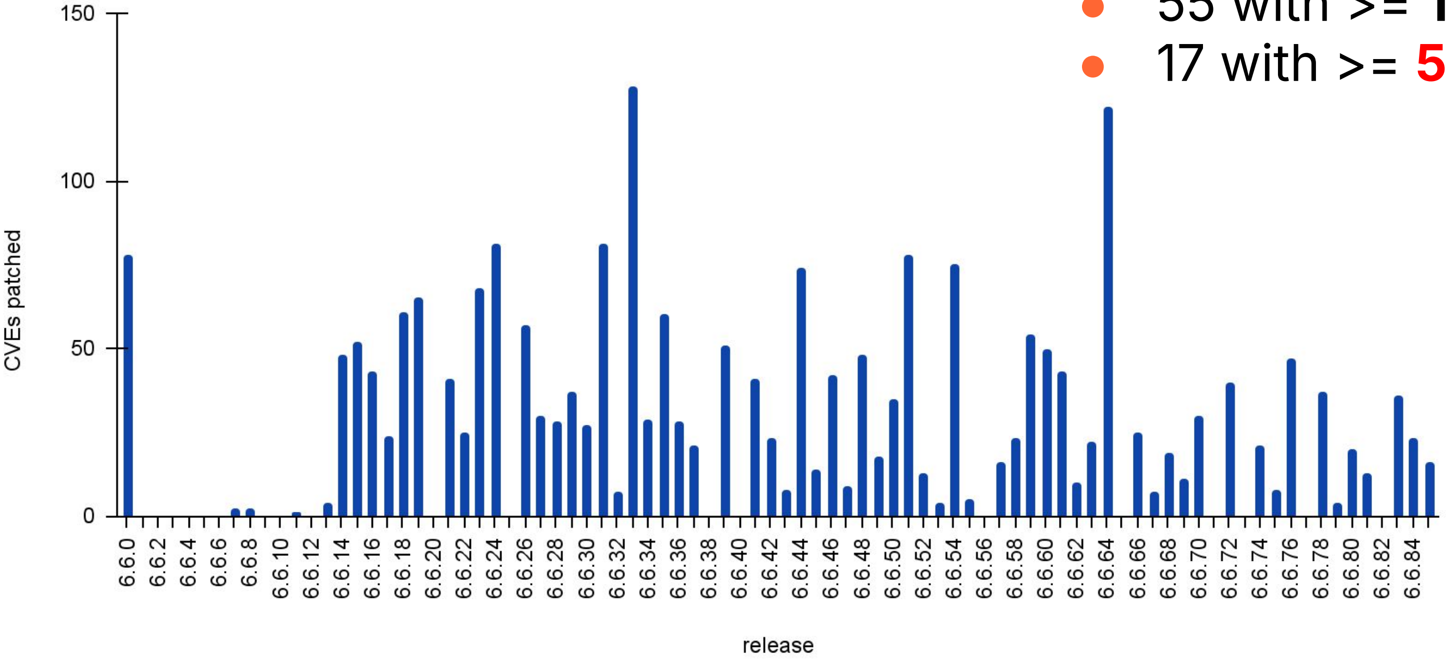
CVEs fixed per release for 6.6.x branch



source: https://lists.openwall.net/linux-cve-announce/

Security vulnerabilities are not getting fixed

CVEs fixed per release for 6.6.x branch



Out of 85 releases:

- 55 with >= **10 CVE patched**
- 17 with >= **50 CVEs patched**

source: https://lists.openwall.net/linux-cve-announce/

Compliance risks

Compliance risks

PCI DSS v4.0

6.3.3 All system components are protected from known vulnerabilities by installing applicable security patches/updates as follows:

- Critical or high-security patches/updates (identified according to the risk ranking process at Requirement 6.3.1) are installed within **one month of release**.
- All other applicable security patches/updates are installed within an appropriate time frame as determined by the entity (for example, within three months of release).

Compliance risks

Remember?

(Not so)fun fact:

if your uptime >= 30 days, you’re system is likely **vulnerable!**

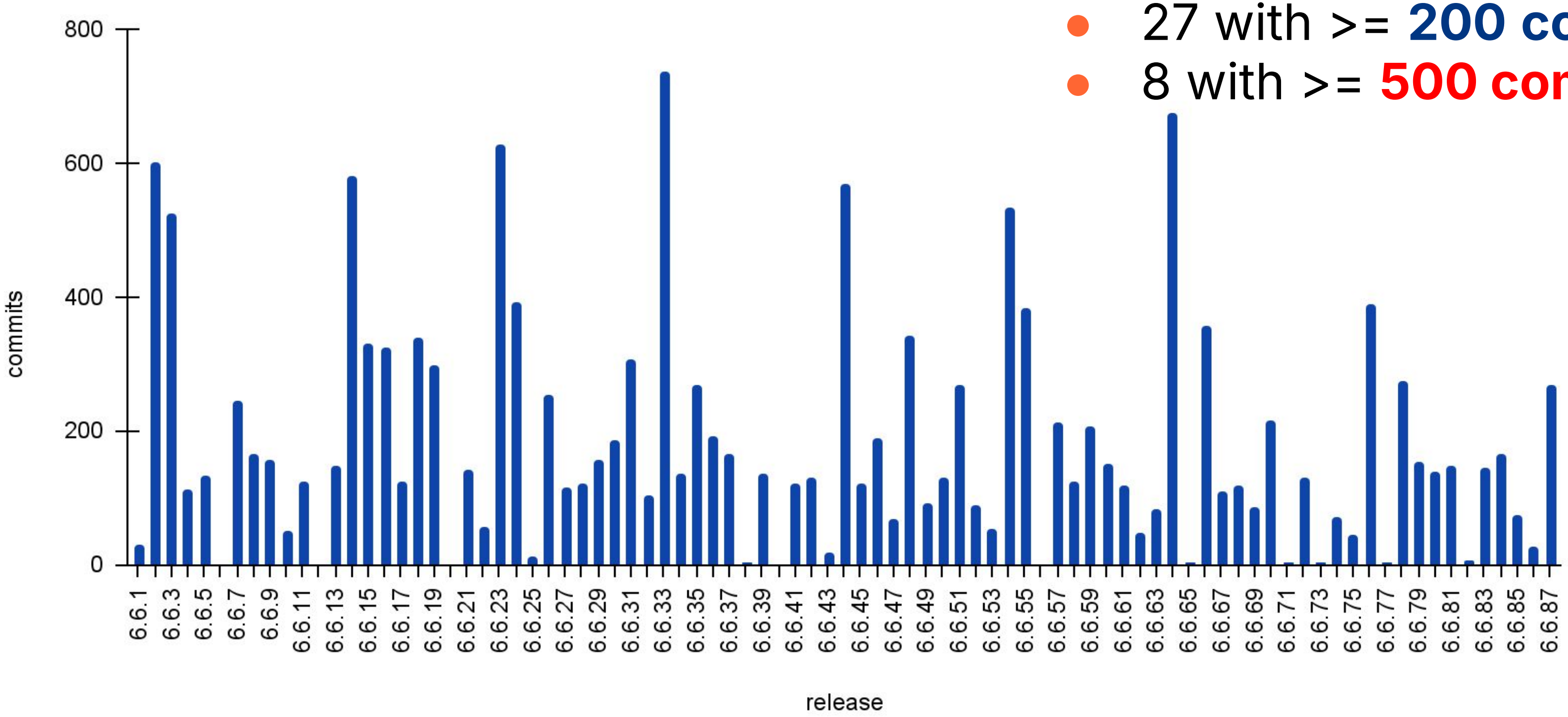
Common anti patterns for Linux Kernel releases

Let's justify the upgrade

Which things from the changelog are applicable to us?

Let's justify the upgrade

Commits per release for 6.6.x branch



Out of 87 releases:

- 60 with >= **100 commits**
- 27 with >= **200 commits**
- 8 with >= **500 commits**

Let’s justify the upgrade

Let’s justify the upgrade

Is this security vulnerability actually exploitable on our systems?

Is this vulnerability applicable to us?

The attacker

- Highly motivated to break into the system
- Spends exclusively almost 24/7 to design and implement a successful exploit

Is this vulnerability applicable to us?

The attacker

- Highly motivated to break into the system
- Spends exclusively almost 24/7 to design and implement a successful exploit

Is this vulnerability applicable to us?

The attacker

- Highly motivated to break into the system
- Spends exclusively almost 24/7 to design and implement a successful exploit

Security patch reviewer

- Highly motivated to go home on time
- Needs to review several patches a day
- Has other competing priorities

Is this vulnerability applicable to us?

The attacker

- Highly motivated to break into the system
- Spends exclusively almost 24/7 to design and implement a successful exploit

Security patch reviewer

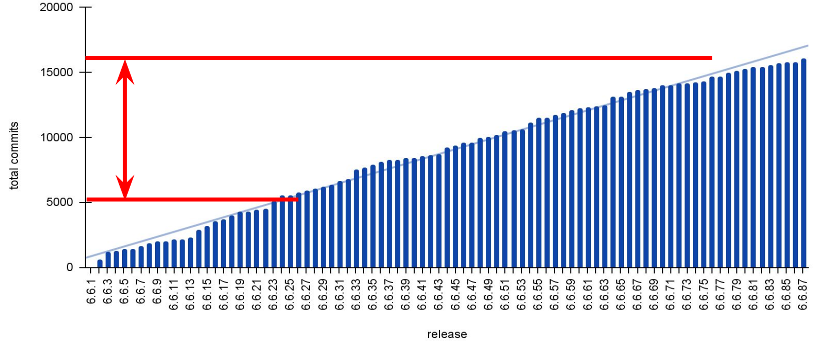
- Highly motivated to go home on time
- Needs to review several patches a day
- Has other competing priorities

Let it soak

Let’s soak it for 1 month in canary to ensure it is stable

Let it soak

Total commits per release for 6.6.x branch

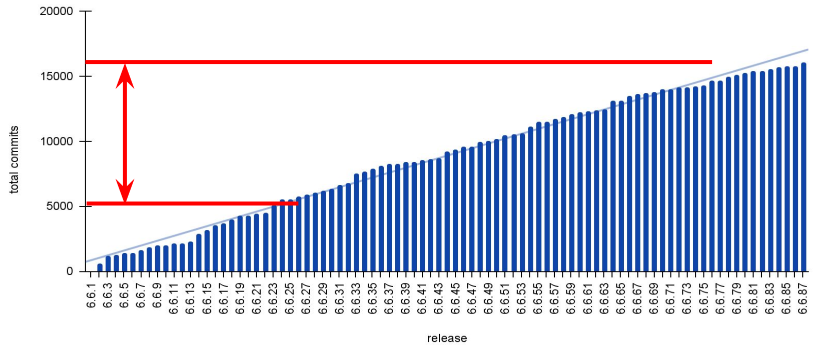


Change delta (risk):

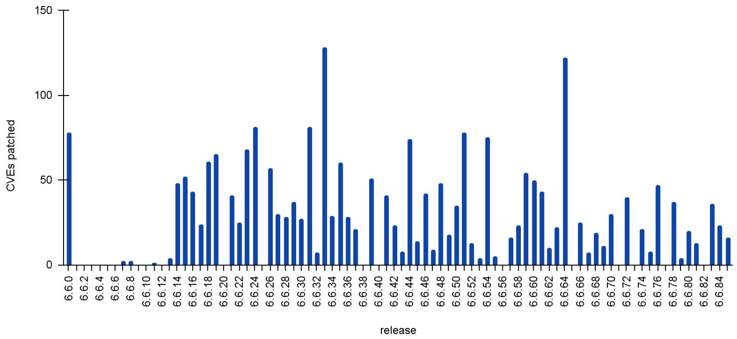
- 6.6.35 vs 6.6.45: 1454
- 6.6.35 vs 6.6.55: 3605
- **3605/1454 = ~2.5**
- **for 2x delay we get ~2.5 more risk!**

Let it soak

Total commits per release for 6.6.x branch



CVEs fixed per release for 6.6.x branch



Change delta (risk):

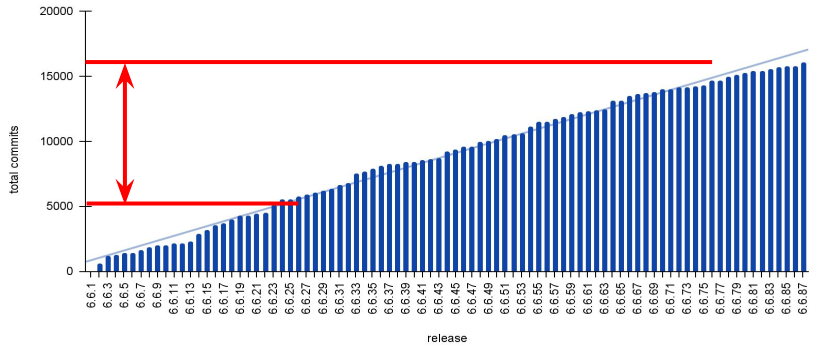
- 6.6.35 vs 6.6.45: 1454
- 6.6.35 vs 6.6.55: 3605
- **3605/1454 = ~2.5**
- **for 2x delay we get ~2.5 more risk!**

Out of 85 releases:

- 55 with >= **10 CVE patched**
- 17 with >= **50 CVEs patched**

Let it soak

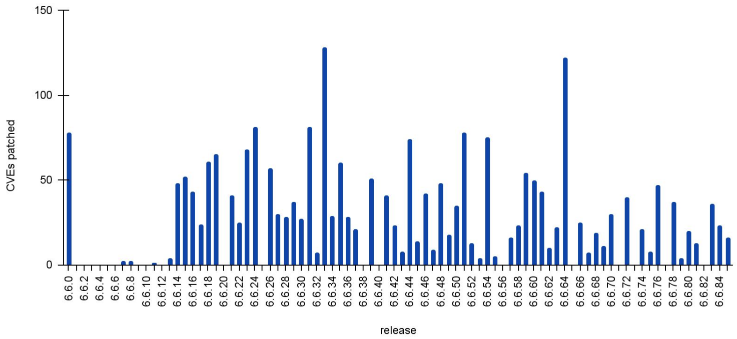
Total commits per release for 6.6.x branch



Change delta (risk):

- 6.6.35 vs 6.6.45: 1454
- 6.6.35 vs 6.6.55: 3605
- **3605/1454 = ~2.5**
- **for 2x delay we get ~2.5 more risk!**

CVEs fixed per release for 6.6.x branch



Out of 85 releases:

- 55 with >= **10 CVE patched**
- 17 with >= **50 CVEs patched**

Let it soak

High “soak” times probably means

- We don’t know what we are looking for
    - Lack of metrics/observability

Let it soak

High “soak” times probably means

- We don’t know what we are looking for
  - Lack of metrics/observability
- We don’t know our workload
  - What kernel features/subsystems are important to us

Let it soak

High “soak” times probably means

- We don’t know what we are looking for
  - Lack of metrics/observability
- We don’t know our workload
  - What kernel features/subsystems are important to us
- Lack of sufficient pre-production kernel testing
  - Unit tests
  - Integration tests
  - Performance tests

Too risky!

The Kernel is too critical! Let’s have more approvals before the deploy!

Too risky!

The Kernel is too critical! Let’s have more approvals before the deploy!

Too risky!

WHAT IF I TOLD YOU
KERNEL DEPLOYS ARE INHERENTLY SAFER THAN OTHER SOFTWARE
imgflip.com

Automated software deploys

Automated software deploys

Regular software

- Upgrade software package

Automated software deploys

Regular software

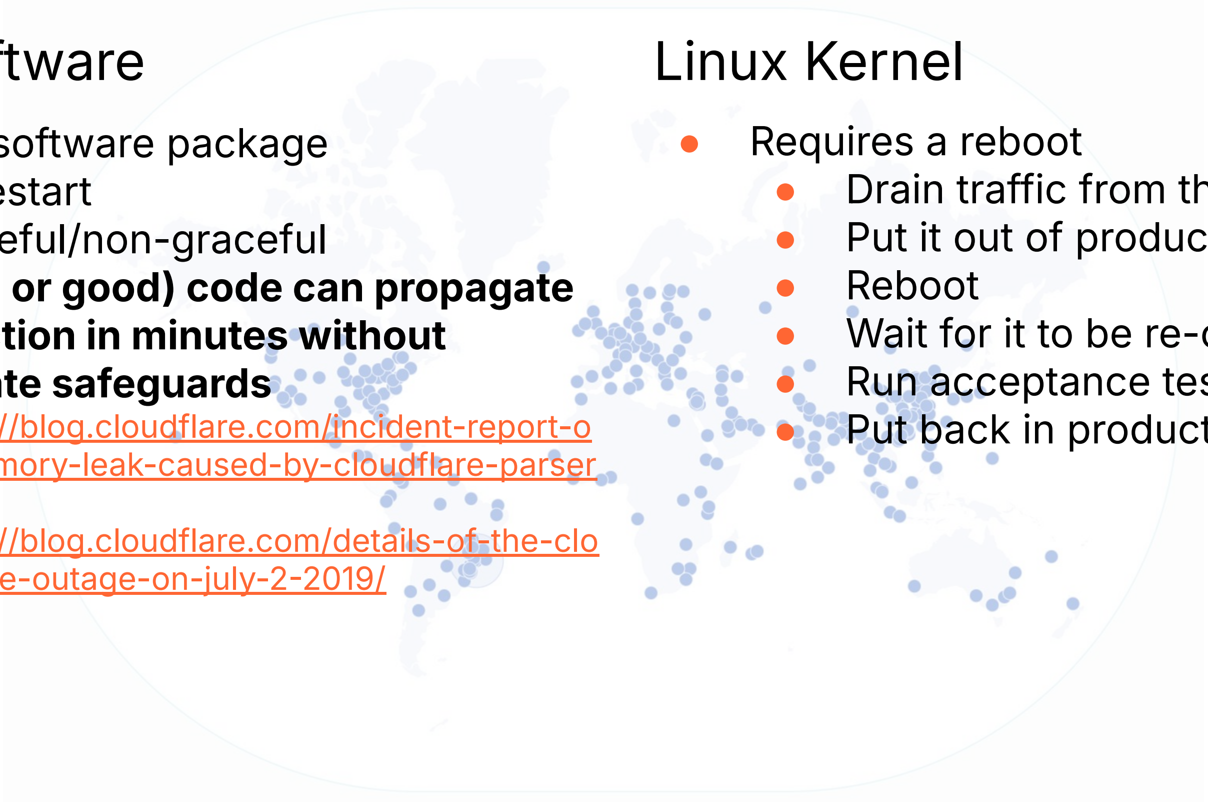
- Upgrade software package
- Service restart
  - graceful/non-graceful

Automated software deploys

Regular software

- Upgrade software package
- Service restart
  - graceful/non-graceful
- **New (bad or good) code can propagate to production in minutes without appropriate safeguards**
  - https://blog.cloudflare.com/incident-report-on-memory-leak-caused-by-cloudflare-parser-bug/
  - https://blog.cloudflare.com/details-of-the-cloudflare-outage-on-july-2-2019/

Automated software deploys



Regular software

- Upgrade software package
- Service restart
  - graceful/non-graceful
- **New (bad or good) code can propagate to production in minutes without appropriate safeguards**
  - https://blog.cloudflare.com/incident-report-on-memory-leak-caused-by-cloudflare-parser-bug/
  - https://blog.cloudflare.com/details-of-the-cloudflare-outage-on-july-2-2019/

Linux Kernel

- Requires a reboot
  - Drain traffic from the server
  - Put it out of production
  - Reboot
  - Wait for it to be re-configured
  - Run acceptance tests
  - Put back in production

Automated software deploys

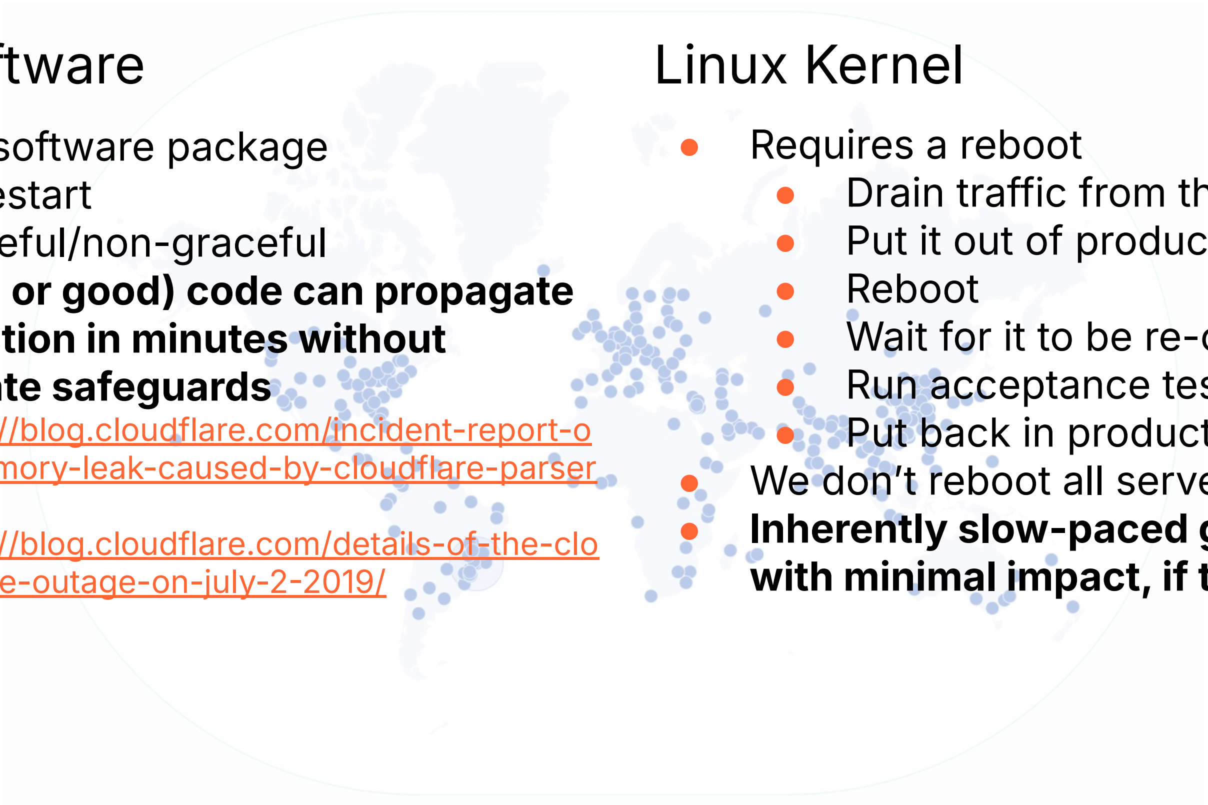
Regular software

- Upgrade software package
- Service restart
  - graceful/non-graceful
- **New (bad or good) code can propagate to production in minutes without appropriate safeguards**
  - https://blog.cloudflare.com/incident-report-on-memory-leak-caused-by-cloudflare-parser-bug/
  - https://blog.cloudflare.com/details-of-the-cloudflare-outage-on-july-2-2019/

Linux Kernel

- Requires a reboot
  - Drain traffic from the server
  - Put it out of production
  - Reboot
  - Wait for it to be re-configured
  - Run acceptance tests
  - Put back in production
- We don’t reboot all servers at once

Automated software deploys



Regular software

- Upgrade software package
- Service restart
  - graceful/non-graceful
- **New (bad or good) code can propagate to production in minutes without appropriate safeguards**
  - https://blog.cloudflare.com/incident-report-on-memory-leak-caused-by-cloudflare-parser-bug/
  - https://blog.cloudflare.com/details-of-the-cloudflare-outage-on-july-2-2019/

Linux Kernel

- Requires a reboot
  - Drain traffic from the server
  - Put it out of production
  - Reboot
  - Wait for it to be re-configured
  - Run acceptance tests
  - Put back in production
- We don’t reboot all servers at once
- **Inherently slow-paced gradual rollout with minimal impact, if things go wrong**

Linux Kernel releases explained

Not every kernel release is created equal

Kernel release numbers

X.XX.XX

Kernel release numbers

X.XX.XX

(ex 6.6.32)

Kernel release numbers

X.XX.XX

(ex 6.6.32)

https://semver.org/

Kernel release numbers

X.XX.XX

(ex 6.6.32)

But it is **NOT** a semver!

https://semver.org/

Kernel release numbers

X.XX.XX

Kernel release numbers

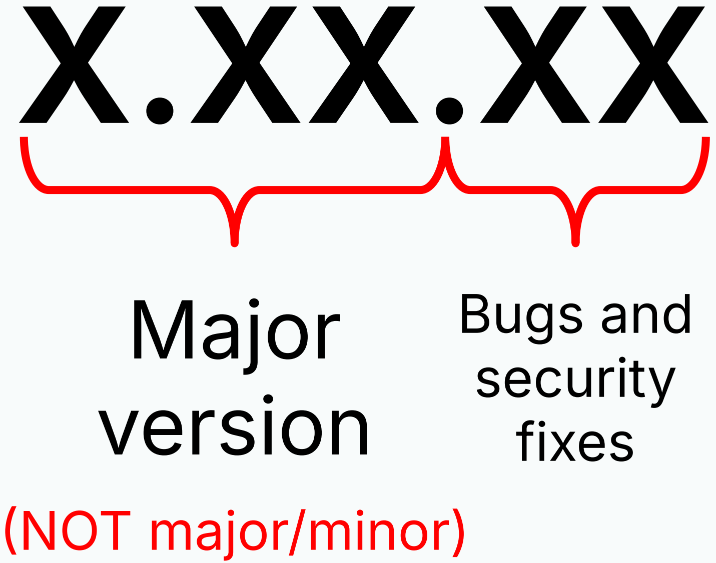
$$\underbrace{\text{X.XX}}_{\text{Major version}}\text{.XX}$$

Kernel release numbers

$$\underbrace{\text{X.XX}}_{\text{Major version}}\text{.XX}$$

(NOT major/minor)

Kernel release numbers



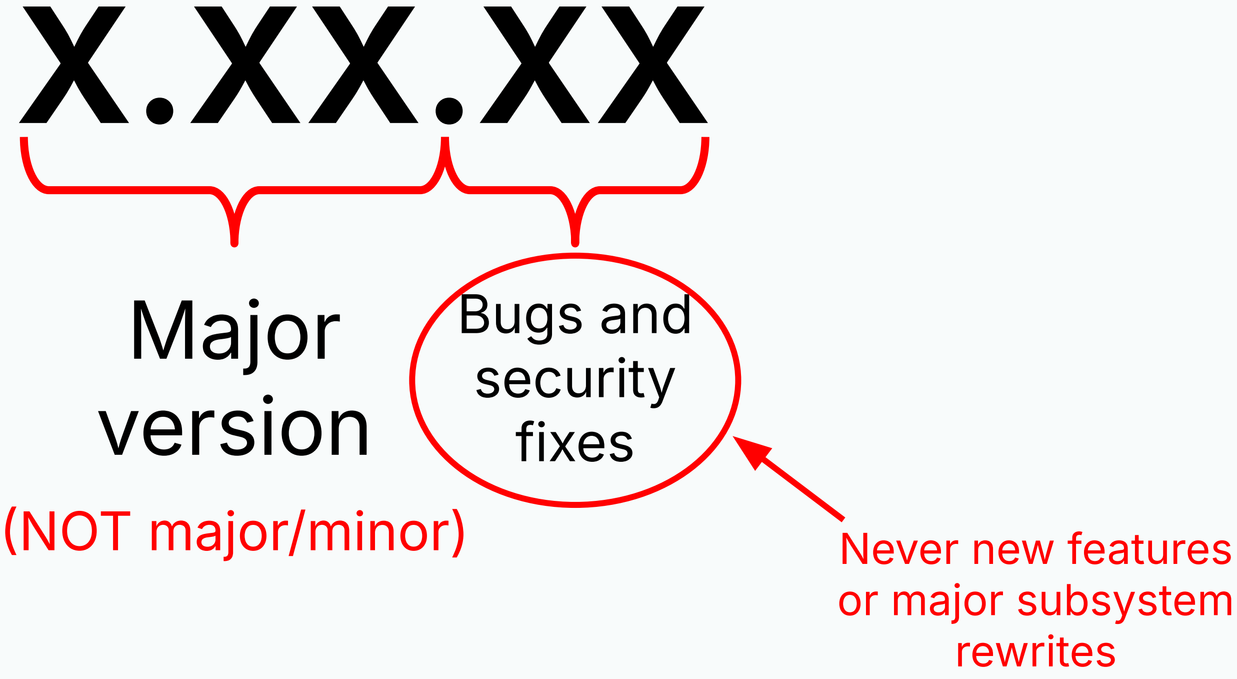
X.XX.XX

Major version

(NOT major/minor)

Bugs and security fixes

Kernel release numbers



X.XX.XX

Major version

(NOT major/minor)

Bugs and security fixes

Never new features or major subsystem rewrites

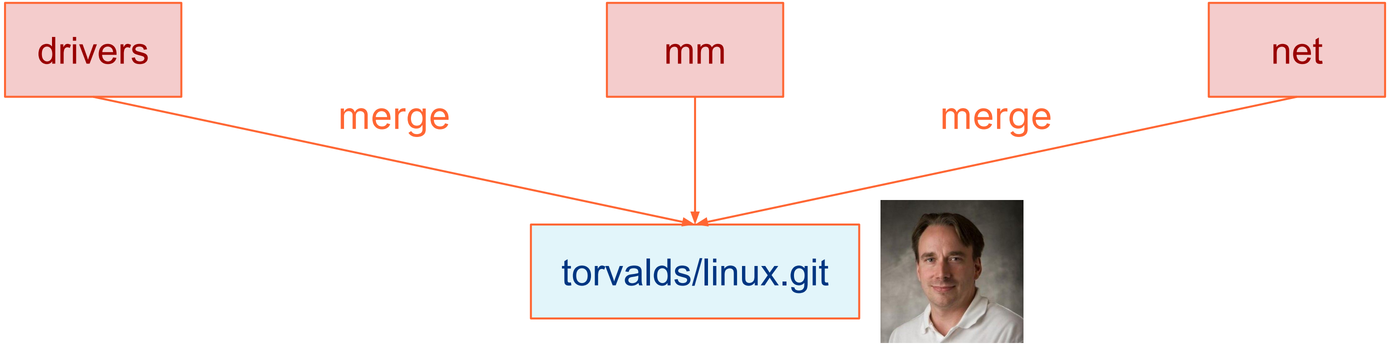
Kernel release flow

torvalds/linux.git

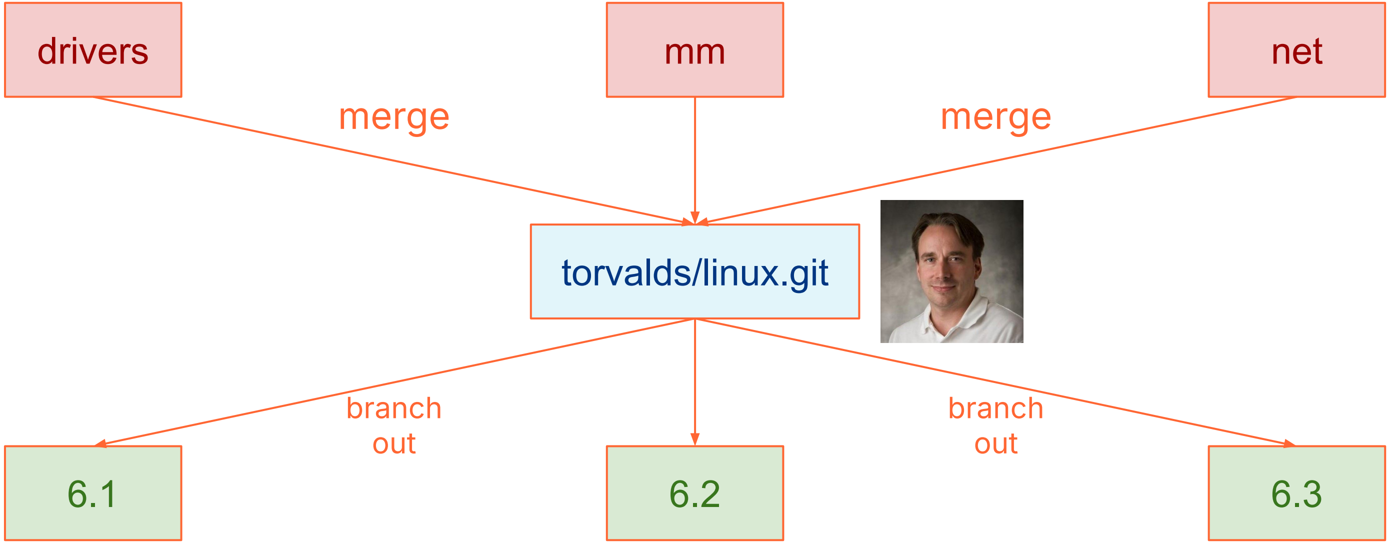
Kernel release flow

torvalds/linux.git

Kernel release flow



Kernel release flow



drivers

mm

net

merge

merge

torvalds/linux.git

branch out

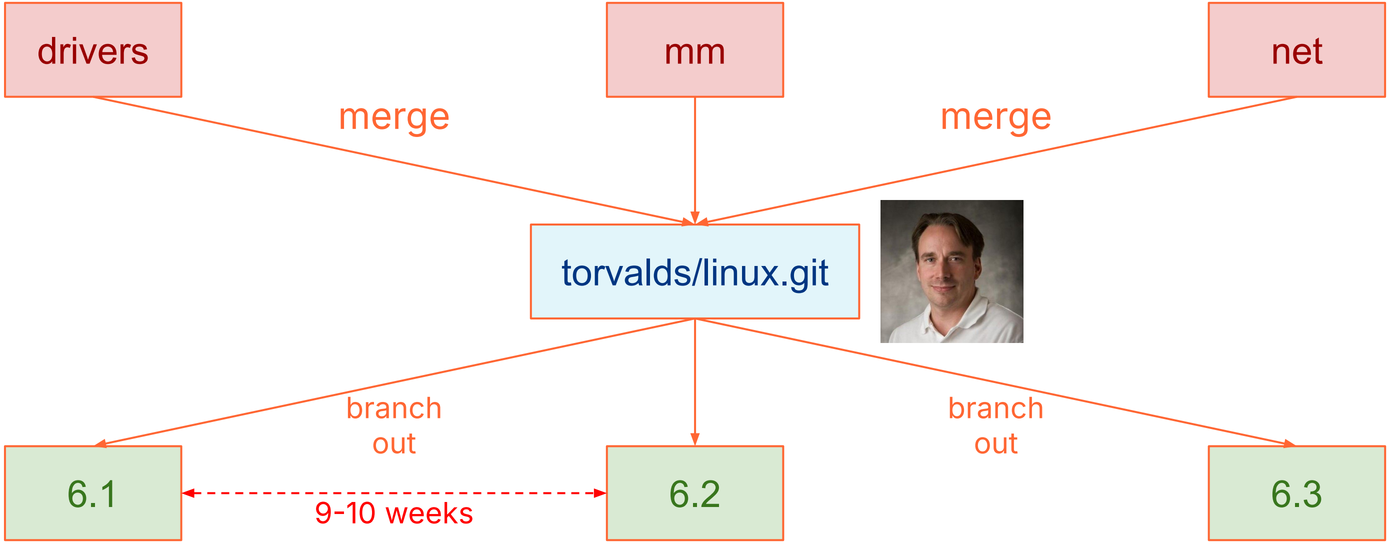
branch out

6.1

6.2

6.3

Kernel release flow



drivers

mm

net

merge

merge

torvalds/linux.git

branch out

branch out

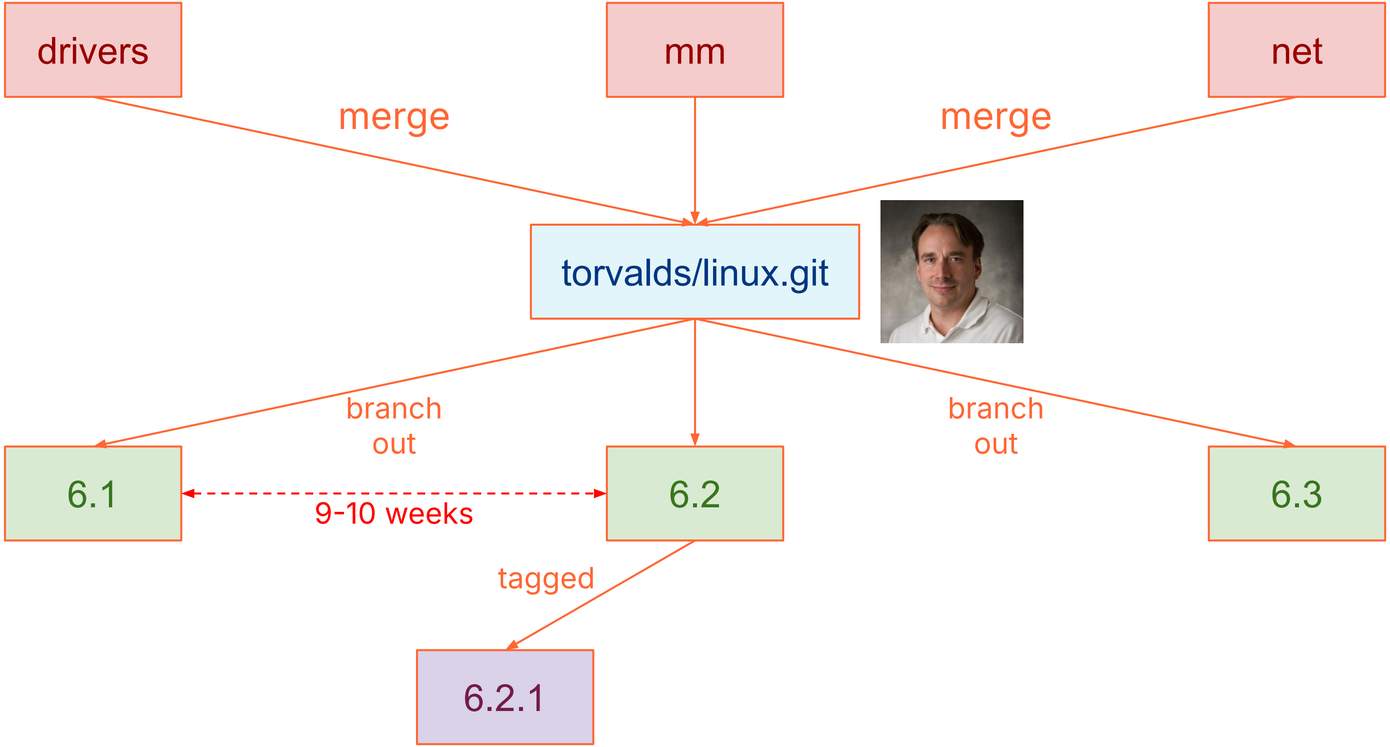
6.1

6.2

6.3

9-10 weeks

Kernel release flow



drivers

mm

net

merge

merge

torvalds/linux.git

branch out

branch out

6.1

6.2

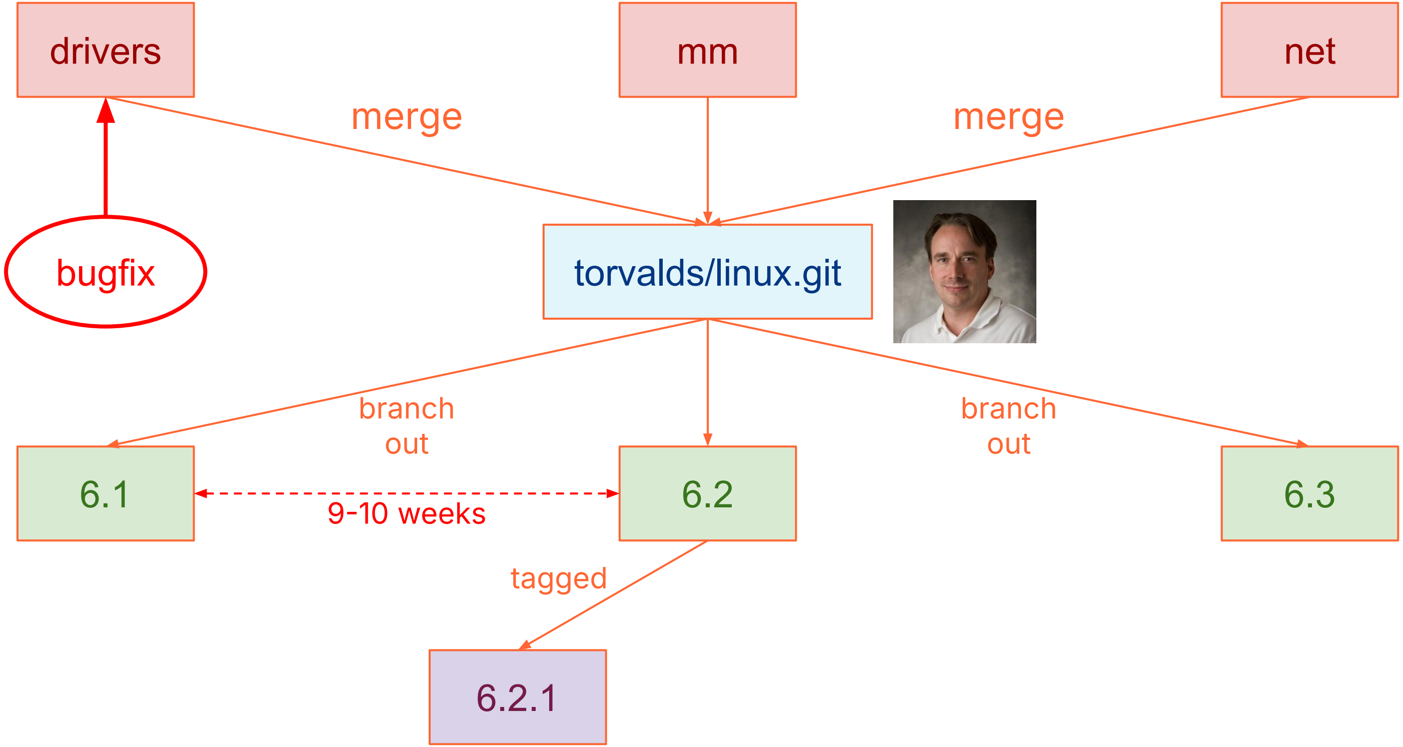
6.3

9-10 weeks

tagged

6.2.1

Kernel release flow



drivers

mm

net

merge

merge

bugfix

torvalds/linux.git

branch out

branch out

6.1

6.2

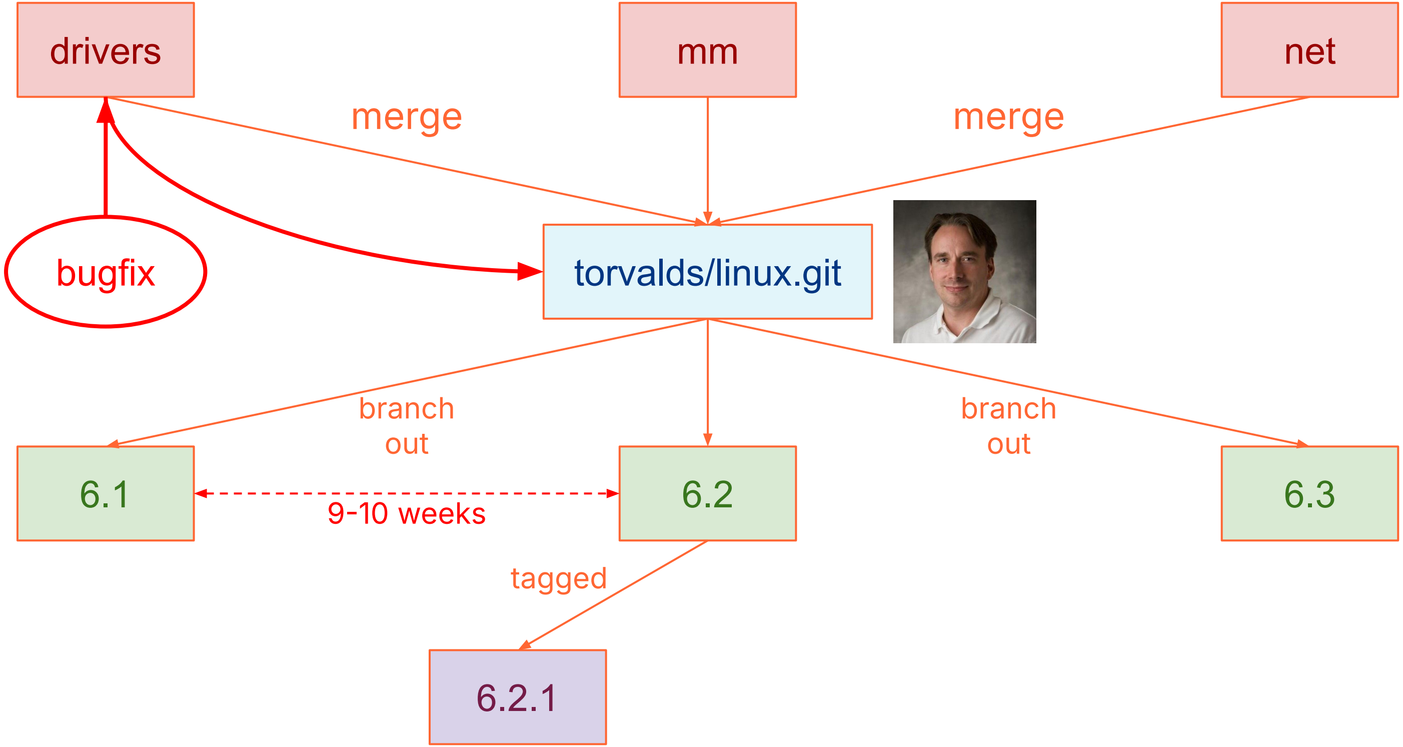
6.3

9-10 weeks

tagged

6.2.1

Kernel release flow



drivers

mm

net

merge

merge

bugfix

torvalds/linux.git

branch out

branch out

6.1

6.2

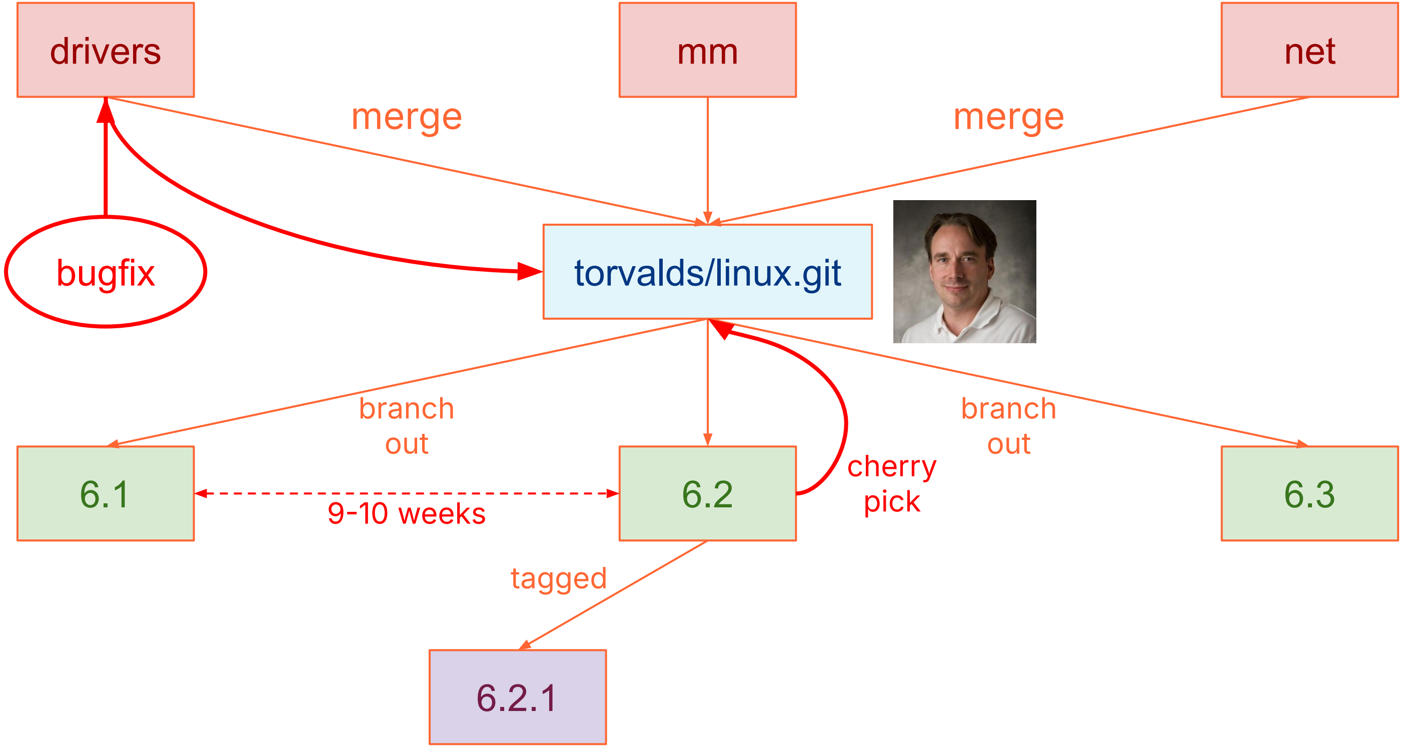
6.3

9-10 weeks

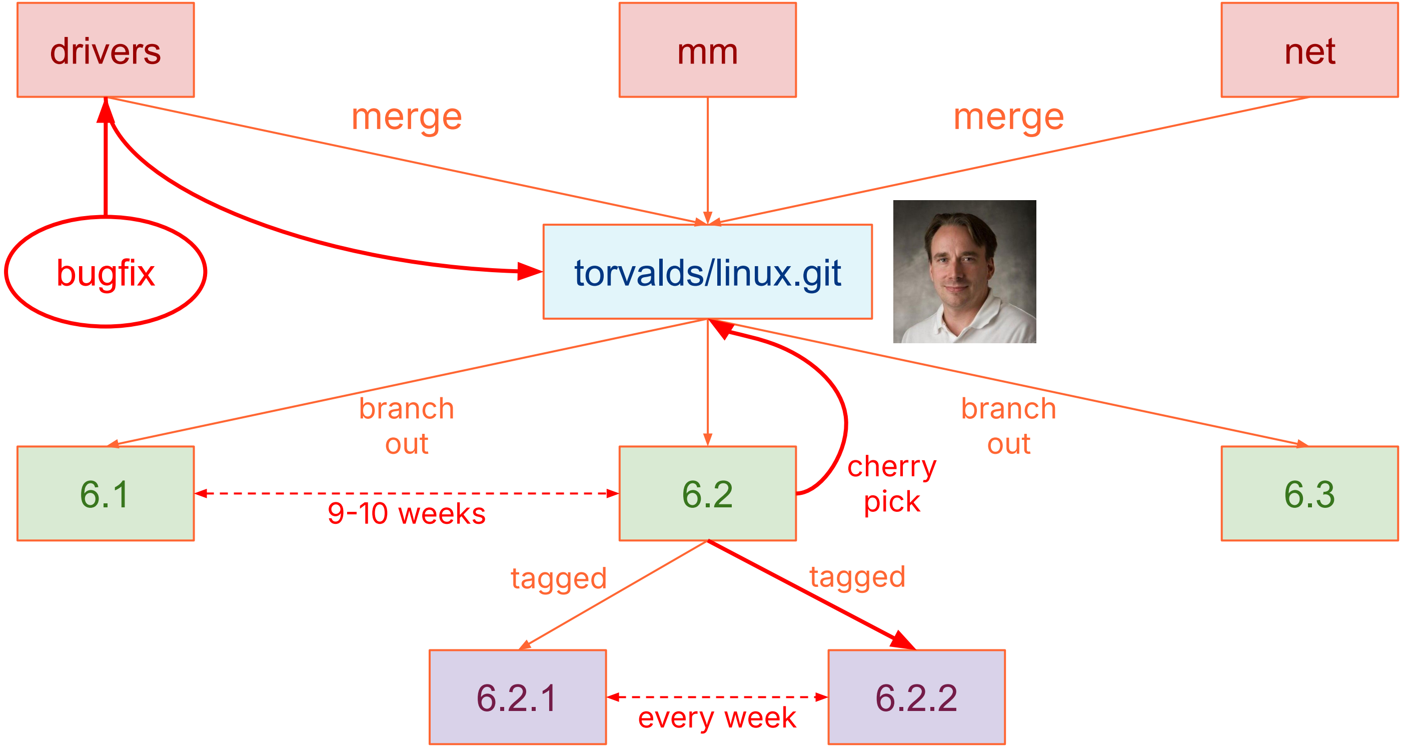
tagged

6.2.1

Kernel release flow



Kernel release flow



Linux Kernel releases

- A new major (stable) kernel version is released every 9-10 weeks
  - 2 weeks for development/7 weeks for bugfixing

Linux Kernel releases

- A new major (stable) kernel version is released every 9-10 weeks
  - 2 weeks for development/7 weeks for bugfixing
- Leftmost version number **means nothing**
  - 4.19.x → 4.20.x upgrade can contain more features/breaking changes than 4.20.x → 5.0.x

Linux Kernel releases

- A new major (stable) kernel version is released every 9-10 weeks
  - 2 weeks for development/7 weeks for bugfixing
- Leftmost version number **means nothing**
  - 4.19.x → 4.20.x upgrade can contain more features/breaking changes than 4.20.x → 5.0.x
- Bugfix/patch releases are released around once a week
  - Denoted by rightmost version number
  - Usually cherry-picked from the main Linux branch
  - No new features, therefore regressions are quite rare
  - May contain critical security patches
  - You **almost always** want to apply them

Longterm releases

- Usually a stable release branch is active around 2-3 months
  - After that it is EOL and no bugfixes are backported (including critical security vulnerabilities)
  - A new major stable version should be available at this point

Longterm releases

- Usually a stable release branch is active around 2-3 months
  - After that it is EOL and no bugfixes are backported (including critical security vulnerabilities)
  - A new major stable version should be available at this point
- But there are “longterm” stable releases
  - Bug and security fixes are backported for at least 2 years
  - Usually the last stable release of the year
    - Therefore, released once a year
  - Provides enough time for more rigid evaluation of the next “longterm” release

Longterm releases

- Usually a stable release branch is active around 2-3 months
  - After that it is EOL and no bugfixes are backported (including critical security vulnerabilities)
  - A new major stable version should be available at this point
- But there are “longterm” stable releases
  - Bug and security fixes are backported for at least 2 years
  - Usually the last stable release of the year
    - Therefore, released once a year
  - Provides enough time for more rigid evaluation of the next “longterm” release

https://www.kernel.org/category/releases.html

Safe and easy production kernel upgrades

Safe and easy production kernel upgrades

Don’t create a dedicated deploy process for the Linux Kernel

Safe and easy production kernel upgrades

Don’t create a dedicated deploy process for the Linux Kernel

- Kernel upgrades are usually less risky than other software
- A simple staged rollout is usually enough
- Kernel upgrades are naturally slow paced, because they require a reboot
  - A lot of headroom to abort the deploy if things look wrong

Safe and easy production kernel upgrades

Avoid justifying a bugfix kernel upgrades

Safe and easy production kernel upgrades

Avoid justifying a bugfix kernel upgrades

- Should be released with “no questions asked”
- Contain only bug fixes and security patches
    - And most likely some are always applicable
- Regressions are quite uncommon
- Minimise canary “soak” times
    - Use metrics-driven approach instead

Safe and easy production kernel upgrades

Stay on the “longterm” branch, if validating a major version is costly

Safe and easy production kernel upgrades

Stay on the “longterm” branch, if validating a major version is costly

- At least two years of bugfixes and security patches
- But start evaluating the next “longterm” release early in ~1 year
  - More features
  - Better performance and resource utilisation
- Accumulating less change delta

Safe and easy production kernel upgrades

Implement/improve pre-production testing for major version validation

Safe and easy production kernel upgrades

Implement/improve pre-production testing for major version validation

- Understand your workload
- Write tests, which exercise various kernel subsystems required by your workload
  - Can help when communicating issues to the kernel community
- Make metrics-driven decisions
  - Not time-based decisions (minise “soak” times)

Metrics, monitoring and deploy automation can help with human risk perception

Safe and easy production kernel upgrades

Metrics, monitoring and deploy automation can help with human risk perception

- Data-driven decision if the deploy looks good
- Provides quick early signals about regressions
- Can save the engineering team a debugging cycle
- Automation encourages regular upgrades
  - Removes the need for an operator to perform a “potentially risky” release

Conclusions

- Linux Kernel upgrades are not more risky than any other software
- You need to patch early and patch often
- Bugfix kernel releases should be applied with “no questions asked”
- Understanding your workload, metrics, monitoring and automation allow your systems to stay patched and secure

Thank you!

Questions?