

Linux Plumbers 2023

Beginner Linux Kernel Maintainer's Toolbox

Krzysztof Kozlowski, [Linaro](https://linaro.org/)
krzysztof.kozlowski@linaro.org
@krzk@social.kernel.org



Linaro
Developer Services

Introduction

- Maintainer in this context is someone who applies received patches and sends them further to the upstream maintainer
 - Upstream maintainer could be Linus Torvalds or someone between you and Linus

Introduction

- Maintainer in this context is someone who applies received patches and sends them further to the upstream maintainer
 - Upstream maintainer could be Linus Torvalds or someone between you and Linus
- The talk is focusing on the basics of maintainer workflow
 - There are a few, not always known expectations, from the maintainer
 - Linux-next
 - Workflow improvements
 - Tips for git.kernel.org
 - PGP keys
 - Dump mailing lists

Me

- Krzysztof Kozlowski
- I work for Linaro in Qualcomm Landing Team
- I maintain few Linux kernel pieces
 - Devicetree bindings, Samsung ARM SoC, NFC and more
 - I usually send pull requests to the SoC tree

Me

- Krzysztof Kozlowski
- I work for Linaro in Qualcomm Landing Team
- I maintain few Linux kernel pieces
 - Devicetree bindings, Samsung ARM SoC, NFC and more
 - I usually send pull requests to the SoC tree
- Why am I talking?
 - I don't handle that many patches, but my upstream maintainers seem happy
 - I develop quite a lot of patches for many different subsystems, so I started noticing things

Integration Tree and Robots

Schrödinger's Patch

- Sent patch for something, rather small, maybe cleanup or bugfix



Schrödinger's Patch

- Sent patch for something, rather small, maybe cleanup or bugfix
- Received one of:
 - *Thanks, applied*
 - *Reviewed-by: Werner Heisenberg <w@hberg.com>*
 - ... or received nothing. Silence.



Schrödinger's Patch

- Sent patch for something, rather small, maybe cleanup or bugfix
- Received one of:
 - *Thanks, applied*
 - *Reviewed-by: Werner Heisenberg <w@hberg.com>*
 - ... or received nothing. Silence.
- Month passed and Schrödinger checks linux-next to see if the patch is there
 - Nope, it's not in linux-next
 - Where is it then?



Schrödinger's Patch

- Sent patch for something, rather small, maybe cleanup or bugfix
- Received one of:
 - *Thanks, applied*
 - *Reviewed-by: Werner Heisenberg <w@hberg.com>*
 - ... or received nothing. Silence.
- Month passed and Schrödinger checks linux-next to see if the patch is there
 - Nope, it's not in linux-next
 - Where is it then?
 - There are only ~1000 kernel.org Linux repos
 - ~500 unique Git repos in the MAINTAINERS file
 - Plus repos on Github, Gitlab, Freedesktop.org
 - Not feasible



Schrödinger's Patch

- Sent patch for something, rather small, maybe cleanup or bugfix
- Received one of:
 - *Thanks, applied*
 - *Reviewed-by: Werner Heisenberg <w@hberg.com>*
 - ... or received nothing. Silence.
- Month passed and Schrödinger checks linux-next to see if the patch is there
 - Nope, it's not in linux-next
 - Where is it then?
 - There are only ~1000 kernel.org Linux repos
 - ~500 unique Git repos in the MAINTAINERS file
 - Plus repos on Github, Gitlab, Freedesktop.org
 - Not feasible
- Or maybe only some patches from a branch were applied, so Schrödinger rebases the branch on top of linux-next and resends what's left



Schrödinger's Patch Dilemma

- I received “*Thanks, applied*”, but patch is nowhere. It got lost.
 - Obviously need to ask what happened or resend the patch?

Schrödinger's Patch Dilemma

- I received “*Thanks, applied*”, but patch is nowhere. It got lost.
 - Obviously need to ask what happened or resend the patch?
- I received “*Reviewed-by*”, patch is not in linux-next
 - Obviously need to send a ping or resend the patch?

Schrödinger's Patch Dilemma

- I received “*Thanks, applied*”, but patch is nowhere. It got lost.
 - Obviously need to ask what happened or resend the patch?
- I received “*Reviewed-by*”, patch is not in linux-next
 - Obviously need to send a ping or resend the patch?
- I received nothing/silence
 - Obviously need to send a ping or resend the patch?

Schrödinger's Patch Dilemma

- I received “*Thanks, applied*”, but patch is nowhere. It got lost.
 - Obviously need to ask what happened or resend the patch?
- I received “*Reviewed-by*”, patch is not in linux-next
 - Obviously need to send a ping or resend the patch?
- I received nothing/silence
 - Obviously need to send a ping or resend the patch?
- Then maintainer responds:
 - - *I have already applied this, why are you ping?*
 - - *I am ping because I am confused! What is happening with my patch?*

Why do I need to waste mine and your time to resend the patch?



Why the Patch Was Not in the linux-next?

- When we talk about a patch being applied, it means it will be sent to upstream maintainer for the next merge-window
 - Eventually to Linus during the next merge window

Why the Patch Was Not in the linux-next?

- When we talk about a patch being applied, it means it will be sent to upstream maintainer for the next merge-window
 - Eventually to Linus during the next merge window
- Till it happens there is no presence in linux-next
 - Reduced visibility
 - Reduced testing
 - Confusion for submitter (“My patch was lost!”)



Why the Patch Was Not in the linux-next?

- When we talk about a patch being applied, it means it will be sent to upstream maintainer for the next merge-window
 - Eventually to Linus during the next merge window
- Till it happens there is no presence in linux-next
 - Reduced visibility
 - Reduced testing
 - Confusion for submitter (“My patch was lost!”)
 - Linus expects patches being in the linux-next before merge window starts [\[1\]](#)

Linux-next

- If you collect patches and send them further to upstream maintainer, add yourself to the linux-next

Linux-next

- If you collect patches and send them further to upstream maintainer, add yourself to the linux-next
- It's free!
 - As in "free beer" (so far no invoices from Stephen)



Linux-next

- If you collect patches and send them further to upstream maintainer, add yourself to the linux-next
- It's free!
 - As in “free beer” (so far no invoices from Stephen)
 - Not free from effort, so don't add your repo to linux-next if you are not a maintainer



Linux-next

- If you collect patches and send them further to upstream maintainer, add yourself to the linux-next
- It's free!
 - As in “free beer” (so far no invoices from Stephen)
 - Not free from effort, so don't add your repo to linux-next if you are not a maintainer
- How to do it?
 - Email to:
 - Stephen Rothwell <sfr@canb.auug.org.au>
 - linux-next@vger.kernel.org
 - With the names of branches:
 - For the next release (next/master)
 - For current RC fixes (next/pending-fixes)



Linux-next - Rules

- Common expectations are listed by Stephen in his response to you for adding new repository to the linux-next



Linux-next - Rules

- Common expectations are listed by Stephen in his response to you for adding new repository to the linux-next
- Rebasing branches is allowed (as far as linux-next is concerned)

Linux-next - Rules

- Common expectations are listed by Stephen in his response to you for adding new repository to the linux-next
- Rebasing branches is allowed (as far as linux-next is concerned)
- If you send patches, not pull requests, to your upstream maintainer consider dropping the patches once they get applied
 - Otherwise Stephen might send you emails about duplicated patches in linux-next



Linux-next - Rules

- Common expectations are listed by Stephen in his response to you for adding new repository to the linux-next
- Rebasing branches is allowed (as far as linux-next is concerned)
- If you send patches, not pull requests, to your upstream maintainer consider dropping the patches once they get applied
 - Otherwise Stephen might send you emails about duplicated patches in linux-next
- Do not add new material to the for-next branch during the merge window
 - It is OK to add fixes to the pending-fixes branch
 - But stuff not for current merge window should wait

Testing by Bots

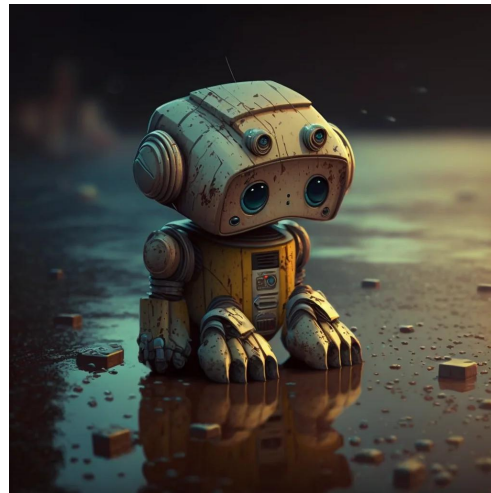
- Most known bot is Intel's 0-Day / LKP / Kernel Test Robot
 - There are others like Kernel CI or Linaro LKFT, but they don't focus on building every possible patch

Testing by Bots

- Most known bot is Intel's 0-Day / LKP / Kernel Test Robot
 - There are others like Kernel CI or Linaro LKFT, but they don't focus on building every possible patch
- If you apply patches to:
 - Repository on kernel.org -> probably you are already covered
 - Other place and your repo is in linux-next -> you will be covered with ~one day delay

Testing by Bots

- Most known bot is Intel's 0-Day / LKP / Kernel Test Robot
 - There are others like Kernel CI or Linaro LKFT, but they don't focus on building every possible patch
- If you apply patches to:
 - Repository on kernel.org -> probably you are already covered
 - Other place and your repo is in linux-next -> you will be covered with ~one day delay
- Other cases: probably no test coverage by bots



https://www.reddit.com/r/midjourney/comments/10fyutr/sad_robot/

Add Yourself to LKP

- Add yourself to LKP
 - Even if your repository is in kernel.org

Add Yourself to LKP

- Add yourself to LKP
 - Even if your repository is in kernel.org
 - Your branches will be built before they reach linux-next
 - You can get notifications on every build of your branch



Add Yourself to LKP

- Add yourself to LKP
 - Even if your repository is in kernel.org
 - Your branches will be built before they reach linux-next
 - You can get notifications on every build of your branch
 - You can also add development (non-maintainer) repositories, e.g. from Github
 - To avoid reports from “Kernel Test Robot” with warnings and build issues once you post your patchset

Add Yourself to LKP

- Add yourself to LKP
 - Even if your repository is in kernel.org
 - Your branches will be built before they reach linux-next
 - You can get notifications on every build of your branch
 - You can also add development (non-maintainer) repositories, e.g. from Github
 - To avoid reports from “Kernel Test Robot” with warnings and build issues once you post your patchset
- Add or update existing “repo” configuration file:
 - <https://github.com/intel/lkp-tests/tree/master/repo/linux>
 - repo/linux/foo-bar
- Send a pull request
 - Just follow some examples
 - <https://github.com/intel/lkp-tests/pull/271>

Add Yourself to LKP

- Add yourself to LKP
 - Even if your repository is in kernel.org
 - Your branches will be built before they reach linux-next
 - You can get notifications on every build of your branch
 - You can also add development (non-maintainer) repositories, e.g. from Github
 - To avoid reports from “Kernel Test Robot” with warnings and build issues once you post your patchset
- Add or update existing “repo” configuration file:
 - <https://github.com/intel/lkp-tests/tree/master/repo/linux>
 - repo/linux/foo-bar
- Send a pull request
 - Just follow some examples
 - <https://github.com/intel/lkp-tests/pull/271>
- See also LKP Wiki:
 - Repo-spec: <https://github.com/intel/lkp-tests/wiki/Repo-Spec>
 - FAQ: <https://github.com/intel/lkp-tests/blob/master/doc/faq.md>

Applying Patches



Applying Patches



Using one
simple tool
to apply the
patch from email

1. Look for Reviewed tags manually
2. Use custom mutt macro which dumps mail to mbox and pipes it to ``git am --signoff``
3. ``git commit --amend`` to manually add the tags
4. Respond to the email with "Thank you, applied"

Applying Patches

- Whatever your method was (Patchwork client, mutt, Save As & `git am`) since some time there is a better one: b4

Applying Patches

- Whatever your method was (Patchwork client, mutt, Save As & `git am`) since some time there is a better one: b4
- Why?
 - Nice Thank-you emails

Applying Patches

- Whatever your method was (Patchwork client, mutt, Save As & `git am`) since some time there is a better one: b4
- Why?
 - Nice Thank-you emails
 - Patchwork integration



Applying Patches

- Whatever your method was (Patchwork client, mutt, Save As & `git am`) since some time there is a better one: b4
- Why?
 - Nice Thank-you emails
 - Patchwork integration
 - Applying entire series or individual patches



Applying Patches

- Whatever your method was (Patchwork client, mutt, Save As & `git am`) since some time there is a better one: b4
- Why?
 - Nice Thank-you emails
 - Patchwork integration
 - Applying entire series or individual patches
 - Collecting Reviewed/Acked/Tested tags
 - Adding “Link:” tags



Applying Patches

- Whatever your method was (Patchwork client, mutt, Save As & `git am`) since some time there is a better one: b4
- Why?
 - Nice Thank-you emails
 - Patchwork integration
 - Applying entire series or individual patches
 - Collecting Reviewed/Acked/Tested tags
 - Adding “Link:” tags
 - Detecting newer versions
 - And comparing between versions (`b4 diff`)
 - And more...

Applying Patches

- Whatever your method was (Patchwork client, mutt, Save As & `git am`) since some time there is a better one: b4
- Why?
 - Nice Thank-you emails
 - Patchwork integration
 - Applying entire series or individual patches
 - Collecting Reviewed/Acked/Tested tags
 - Adding “Link:” tags
 - Detecting newer versions
 - And comparing between versions (`b4 diff`)
 - And more...
- For full guide see Konstantin Ryabitsev talk: “Doing more with lore and b4”
<https://lpc.events/event/11/contributions/983/>

B4 Configuration

- *linux-maintainer-repo/.git/config*:

[b4]

```
# For thank you emails:
thanks-commit-url-mask = https://git.kernel.org/MR_F00/linux/c/%s

# If you want Patchwork integration:
pw-key = FIXME: API token from https://patchwork.kernel.org/user/
pw-url = https://patchwork.kernel.org
pw-project = FIXME: Project name (e.g. linux-samsung-soc)
pw-review-state = under-review
pw-accept-state = accepted
pw-discard-state = deferred
```

Apply Entire Series

```
$ b4 shazam --apply-cover-trailers --add-link -s MESSAGE_ID
```

Apply Entire Series

```
$ b4 shazam --apply-cover-trailers --add-link -s MESSAGE_ID
```

```
Grabbing thread from lore.kernel.org/all/...
```

```
Checking for newer revisions
```

```
Grabbing search results from lore.kernel.org
```

```
Analyzing 7 messages in the thread
```

```
Checking attestation on all messages, may take a moment...
```

```
---
```

```
...
```

```
Total patches: 2
```

```
---
```

```
Applying: ARM: dts: samsung: exynos4412-midas: fix key-ok event code
```

```
Applying: ARM: dts: samsung: exynos4412-midas: use Linux event codes for input keys
```

```
---
```

```
Patchwork: setting state on patchwork.kernel.org/linux-samsung-soc
```

```
-> under-review : [v2,1/2] ARM: dts: samsung: exynos4412-midas: fix key-ok event code
```

```
-> under-review : [v2,2/2] ARM: dts: samsung: exynos4412-midas: use Linux event codes
```

Thank You Emails - Purpose

- To say “thank you”



Thank You Emails - Purpose

- To say “thank you”
- To confirm that patch was applied



Thank You Emails - Purpose

- To say “thank you”
- To confirm that patch was applied
- To share any other useful information, e.g. name of the branch or tree



Send a “Thank You”

```
git push origin for-next  
b4 ty -l
```



Send a “Thank You”

```
git push origin for-next
```

```
b4 ty -l
```

```
b4 ty -a -S
```



Send a “Thank You”

```
git push origin for-next  
b4 ty -l  
b4 ty -a -S
```

- Above thank-you letters will be quite simple with customized Git URL to the commits
 - Check `b4.thanks-am-template` git-config setting
 - <https://git.kernel.org/pub/scm/utils/b4/b4.git/tree/thanks-am-template.example>
- ``b4 ty`` has some other useful options
 - ``man b4`` for more details
 - <https://b4.docs.kernel.org/en/latest/config.html#thank-you-ty-settings>



Commit Hooks - Signed-off-by

- Committer Signed-off-by
 - Even maintainers get it wrong sometimes...
 - *Shouldn't Acked-by tag from me (maintainer of the ... tree) enough? I mean it does imply signed-off-by, right?*
 - No, it doesn't.



Commit Hooks - Signed-off-by

- Committer Signed-off-by
 - Even maintainers get it wrong sometimes...
 - *Shouldn't Acked-by tag from me (maintainer of the ... tree) enough? I mean it does imply signed-off-by, right?*
 - No, it doesn't.
- `verify_signedoff.sh`
 - Greg KH's script:
 - https://github.com/gregkh/gregkh-linux/blob/master/work/scripts/verify_signedoff.sh
 - The script verifies that last committer provided proper Signed-off-by



Commit Hooks - Fixes tag

- Fixes: b00bccb3f0bb ("drm/i915/pmu: Handle PCI unbind")

Commit Hooks - Fixes tag

- Fixes: b00bccb3f0bb ("drm/i915/pmu: Handle PCI unbind")
- Common problem:
 - Fixes tag points to the commit in linux-next, e.g. to one in maintainer's for-next branch

Commit Hooks - Fixes tag

- Fixes: b00bccb3f0bb ("drm/i915/pmu: Handle PCI unbind")
- Common problem:
 - Fixes tag points to the commit in linux-next, e.g. to one in maintainer's for-next branch
 - Maintainer rebases the branch

Commit Hooks - Fixes tag

- Fixes: b00bccb3f0bb ("drm/i915/pmu: Handle PCI unbind")
- Common problem:
 - Fixes tag points to the commit in linux-next, e.g. to one in maintainer's for-next branch
 - Maintainer rebases the branch
 - Fixes tag points to wrong commit
 - It's a real issue. Happens ~few commits per month:
 - <https://lore.kernel.org/all/?q=f%3ARothwell+%22Fixes+tag+needs+some+work+in%22>

Commit Hooks - Fixes tag

- Fixes: b00bccb3f0bb ("drm/i915/pmu: Handle PCI unbind")
- Common problem:
 - Fixes tag points to the commit in linux-next, e.g. to one in maintainer's for-next branch
 - Maintainer rebases the branch
 - Fixes tag points to wrong commit
 - It's a real issue. Happens ~few commits per month:
 - <https://lore.kernel.org/all/?q=f%3ARothwell+%22Fixes+tag+needs+some+work+in%22>
- verify_fixes.sh
 - https://github.com/krzk/tools/blob/master/linux/verify_fixes.sh
 - I extended original script from Greg KH, who probably borrowed it from Stephen Rothwell

Commit Hooks - Fixes tag

- Fixes: b00bccb3f0bb ("drm/i915/pmu: Handle PCI unbind")
- Common problem:
 - Fixes tag points to the commit in linux-next, e.g. to one in maintainer's for-next branch
 - Maintainer rebases the branch
 - Fixes tag points to wrong commit
 - It's a real issue. Happens ~few commits per month:
 - <https://lore.kernel.org/all/?q=f%3ARothwell+%22Fixes+tag+needs+some+work+in%22>
- verify_fixes.sh
 - https://github.com/krzk/tools/blob/master/linux/verify_fixes.sh
 - I extended original script from Greg KH, who probably borrowed it from Stephen Rothwell
 - The script verifies that "Fixes" tag is correct
 - Proper format
 - Fixes real SHA
 - Fixes ancestor commit
 - Maintainer rarely needs to take fixes for commits not in their trees or for some different branches

Commit Hooks - the Hook

```
#!/bin/bash
# SPDX-License-Identifier: GPL-2.0

TOOLS_DIR="$HOME/dev/tools/linux/"
LINUS_MASTER_REF="linus/master"

echo "[Signed-off-by] "
${TOOLS_DIR}verify_signedoff.sh HEAD^..HEAD
test $? -eq 0 && echo "OK"
echo

echo "[Fixes] "
${TOOLS_DIR}verify_fixes.sh "$LINUS_MASTER_REF" HEAD^..HEAD
test $? -eq 0 && echo "OK"
echo
```



Commit Hooks - the Hook (2)

- Idea borrowed from Lee Jones

- Full hook source:

<https://github.com/krzk/tools/blob/master/linux/git-hooks-post-commit>



Commit Hooks - the Hook (2)

- Idea borrowed from Lee Jones

- Full hook source:

<https://github.com/krzk/tools/blob/master/linux/git-hooks-post-commit>

- Adjust the paths to verify_xxx.sh scripts
- Copy to .git/hooks/post-applypatch and .git/hooks/post-commit



Commit Hooks - the Hook (2)

- Idea borrowed from Lee Jones
- Full hook source:
<https://github.com/krzk/tools/blob/master/linux/git-hooks-post-commit>
 - Adjust the paths to verify_xxx.sh scripts
 - Copy to .git/hooks/post-applypatch and .git/hooks/post-commit
 - Hook will be run on every patch apply (git am, so also b4 shazam) and tree rebase



Commit Hooks - the Hook (2)

- Idea borrowed from Lee Jones
- Full hook source:
<https://github.com/krzk/tools/blob/master/linux/git-hooks-post-commit>
 - Adjust the paths to verify_xxx.sh scripts
 - Copy to .git/hooks/post-applypatch and .git/hooks/post-commit
 - Hook will be run on every patch apply (git am, so also b4 shazam) and tree rebase
 - Hook does not fail the process, so one can still ignore its feedback



Commit Hooks - the Hook - Example Run

```
$ git ci --amend
```

```
[Checking commit] 959a6122c368 ARM: dts: samsung: exynos4412-midas: use Linux event codes
```

Commit Hooks - the Hook - Example Run

```
$ git ci --amend
```

```
[Checking commit] 959a6122c368 ARM: dts: samsung: exynos4412-midas: use Linux event codes
```

```
[Checkpatch]
```

```
OK
```

Commit Hooks - the Hook - Example Run

```
$ git ci --amend
```

```
[Checking commit] 959a6122c368 ARM: dts: samsung: exynos4412-midas: use Linux event codes
```

```
[Checkpatch]
```

```
OK
```

```
[Signed-off-by]
```

```
Commit 959a6122c368 ("ARM: dts: samsung: exynos4412-midas: use Linux event codes for input")
```

```
    committer Signed-off-by missing
```

```
    author email:    raymondhackley@protonmail.com
```

```
    committer email: krzysztof.kozlowski@linaro.org
```

```
    Signed-off-by: Raymond Hackley <raymondhackley@protonmail.com>
```

```
Errors in tree with Signed-off-by, please fix!
```

Commit Hooks - the Hook - Example Run

```
$ git ci --amend
```

```
[Checking commit] 959a6122c368 ARM: dts: samsung: exynos4412-midas: use Linux event codes
```

```
[Checkpatch]
```

```
OK
```

```
[Signed-off-by]
```

```
Commit 959a6122c368 ("ARM: dts: samsung: exynos4412-midas: use Linux event codes for input")
```

```
    committer Signed-off-by missing
```

```
    author email:    raymondhackley@protonmail.com
```

```
    committer email: krzysztof.kozlowski@linaro.org
```

```
    Signed-off-by: Raymond Hackley <raymondhackley@protonmail.com>
```

```
Errors in tree with Signed-off-by, please fix!
```


```
[Fixes]
```

```
OK
```

Tricks with kernel.org Git Repo

About Page

- You can customize the About page



index : kernel/git/krzk/linux.git

Linux Samsung SoC tree

about summary refs log tree commit diff stats homepage

Samsung SoC (Exynos, S3C) Linux kernel tree
=====

Pulled up by arm-soc folks.

Branches:

1. for-next: for linux-next, rebased sometimes
2. fixes: for current RC
3. next/defconfig, next/defconfig64, next/drivers, next/dt, next/dt64, next/soc, next/soc64: for next release
4. for-vX.Y/name: topic branches for vX.Y release (usually next release)

About Page (2)

```
git symbolic-ref HEAD refs/meta/cgit
git reset --hard
vi -p cgitr README
# git add, commit
git push origin HEAD:refs/meta/cgit
git checkout master
```

- Full guide: <https://korg.docs.kernel.org/cgit-meta-data.html>



Transparency Log

- Transparency log records all git-receive operations
- Full description:
<https://korg.docs.kernel.org/gitolite/transparency-log.html>



Transparency Log

- Transparency log records all git-receive operations
- Full description:
<https://korg.docs.kernel.org/gitolite/transparency-log.html>
- Transparency log is in public-inbox format, thus browsing is a bit of pain
 - You can see the diffs on Gitweb:
<https://git.kernel.org/pub/scm/infra/transparency-logs/gitolite/git/1.git/log/>
 - But easier browsing requires setting up public-inbox mirror

Transparency Log - Example

← → ↻ 0.0.0.0:8080/kernel.org-transparency-log/

public inbox for devnull@kernel.org

search [help](#) / [color](#) / [mirror](#) / [Atom feed](#)

[post-receive: pub/scm/linux/kernel/git/thierry.reding/linux-pwm](#)
2023-11-03 11:01 UTC

[post-receive: pub/scm/linux/kernel/git/gregkh/char-misc](#)
2023-11-03 11:00 UTC

[post-receive: pub/scm/linux/kernel/git/jlayton/linux](#)
2023-11-03 10:16 UTC

[post-receive: pub/scm/git/git](#)
2023-11-03 9:53 UTC

[post-receive: pub/scm/linux/kernel/git/netdev/net](#)
2023-11-03 9:16 UTC

[post-receive: pub/scm/linux/kernel/git/netdev/net](#)
2023-11-03 9:15 UTC

[post-receive: pub/scm/linux/kernel/git/netdev/net](#)
2023-11-03 9:14 UTC

[post-receive: pub/scm/linux/kernel/git/netdev/net](#)
2023-11-03 9:13 UTC

Transparency Log - Plain Push

* post-receive: pub/scm/linux/kernel/git/netdev/net

@ 2023-11-03 9:16 Gitolite

0 siblings, 0 replies; 6683+ messages in thread

From: Gitolite @ 2023-11-03 9:16 UTC ([permalink](#) / [raw](#))

service: git-receive-pack

repo: pub/scm/linux/kernel/git/netdev/net

user: davem

changes:

- ref: refs/heads/main

old: 63e201916b27260218e528a2f8758be47f99bbf4

new: cdbab6236605dc11780779d9af689aea7d58cab1

log: |

cdbab6236605dc11780779d9af689aea7d58cab1 tcp: fix fastopen code vs usec TS

Transparency Log - Trust

- kernel.org docs:
 - “Since several members of the Linux Foundation IT team have direct backend access to the gitolite server, any one of them (or anyone in possession of their compromised account) can fake a push record.”
 - “If you would like to help hedge against this risk, you are invited to sign your pushes.”

Transparency Log - Trust

- kernel.org docs:
 - “Since several members of the Linux Foundation IT team have direct backend access to the gitolite server, any one of them (or anyone in possession of their compromised account) can fake a push record.”
 - “If you would like to help hedge against this risk, you are invited to sign your pushes.”
- In your repo:

```
git config --local push.gpgSign if-asked
```

Transparency Log - Trust

- kernel.org docs:
 - “Since several members of the Linux Foundation IT team have direct backend access to the gitolite server, any one of them (or anyone in possession of their compromised account) can fake a push record.”
 - “If you would like to help hedge against this risk, you are invited to sign your pushes.”
- In your repo:

```
git config --local push.gpgSign if-asked
```

- Assuming you have your GnuPG key on a SmartCard (bold assumption!)...
 - You already sign your tags for pull requests, right?
- ... this will ask you to unlock SmartCard on the first push during given session
 - That's it!

Transparency Log - Example

← → ↻ 0.0.0.0:8080/kernel.org-transparency-log/

public inbox for devnull@kernel.org

search [help](#) / [color](#) / [mirror](#) / [Atom feed](#)

[post-receive: pub/scm/linux/kernel/git/thierry.reding/linux-pwm](#)
2023-11-03 11:01 UTC

[post-receive: pub/scm/linux/kernel/git/gregkh/char-misc](#)
2023-11-03 11:00 UTC

[post-receive: pub/scm/linux/kernel/git/jlayton/linux](#)
2023-11-03 10:16 UTC

[post-receive: pub/scm/git/git](#)
2023-11-03 9:53 UTC

[post-receive: pub/scm/linux/kernel/git/netdev/net](#)
2023-11-03 9:16 UTC

[post-receive: pub/scm/linux/kernel/git/netdev/net](#)
2023-11-03 9:15 UTC

[post-receive: pub/scm/linux/kernel/git/netdev/net](#)
2023-11-03 9:14 UTC

[post-receive: pub/scm/linux/kernel/git/netdev/net](#)
2023-11-03 9:13 UTC

Transparency Log - Signed Push

```
* post-receive: pub/scm/linux/kernel/git/gregkh/char-misc
@ 2023-11-03 11:00 Gitolite
  0 siblings, 0 replies, 5383+ messages in thread
From: Gitolite @ 2023-11-03 11:00 UTC (permalink / raw)
```

```
[-- Attachment #1: Type: text/plain, Size: 261 bytes --]
```

```
---
```

```
service: git-receive-pack
repo: pub/scm/linux/kernel/git/gregkh/char-misc
user: gregkh
git_push_cert_status: G
changes:
```

```
- ref: refs/tags/char-misc-6.7-rc1
  old: 0000000000000000000000000000000000000000000000000000000000000000
  new: 2eaaf2f210959da1ef59c5939094e494f499cc5e
```

```
[-- Attachment #2: git-push-certificate.txt --]
```

```
[-- Type: text/plain, Size: 1209 bytes --]
```

```
certificate version 0.1
pusher Greg Kroah-Hartman <gregkh@linuxfoundation.org> 1699009239 +0100
pushee gitolite.kernel.org:/pub/scm/linux/kernel/git/gregkh/char-misc.git
nonce 1699009238-6123e86e02cd42981cb57a9fa116d4eecc6b230e
```

```
0000000000000000000000000000000000000000000000000000000000000000 2eaaf2f210959da1ef59c5939094e494f499cc5e refs/tags/char-misc-6.7-rc1
-----BEGIN PGP SIGNATURE-----
```

PGP Keys



Why Do You Need a PGP Key?

- Why do you need a PGP key?

Why Do You Need a PGP Key?

- Why do you need a PGP key?
 - If you send pull requests to your upstream maintainer, hopefully you use PGP-signed git tags (`git tag -a -s`)



Why Do You Need a PGP Key?

- Why do you need a PGP key?
 - If you send pull requests to your upstream maintainer, hopefully you use PGP-signed git tags (`git tag -a -s``)
 - Pull requests should not use branches
 - Linus expects signed tags for any pull not using kernel.org infrastructure

Why Do You Need a PGP Key?

- Why do you need a PGP key?
 - If you send pull requests to your upstream maintainer, hopefully you use PGP-signed git tags (`git tag -a -s``)
 - Pull requests should not use branches
 - Linus expects signed tags for any pull not using kernel.org infrastructure
 - If you turned on signed pushes for git.kernel.org transparency log

Why Do You Need a PGP Key?

- Why do you need a PGP key?
 - If you send pull requests to your upstream maintainer, hopefully you use PGP-signed git tags (`git tag -a -s`)
 - Pull requests should not use branches
 - Linus expects signed tags for any pull not using kernel.org infrastructure
 - If you turned on signed pushes for git.kernel.org transparency log
- Your key does not need to be signed by others to be useful

Why Do You Need a PGP Key?

- Why do you need a PGP key?
 - If you send pull requests to your upstream maintainer, hopefully you use PGP-signed git tags (`git tag -a -s`)
 - Pull requests should not use branches
 - Linus expects signed tags for any pull not using kernel.org infrastructure
 - If you turned on signed pushes for git.kernel.org transparency log
- Your key does not need to be signed by others to be useful
- However eventually it should be signed, to participate in the Web of Trust

Why Do You Need a PGP Key?

- Why do you need a PGP key?
 - If you send pull requests to your upstream maintainer, hopefully you use PGP-signed git tags (`git tag -a -s`)
 - Pull requests should not use branches
 - Linus expects signed tags for any pull not using kernel.org infrastructure
 - If you turned on signed pushes for git.kernel.org transparency log
- Your key does not need to be signed by others to be useful
- However eventually it should be signed, to participate in the Web of Trust
- If you ever want to apply for kernel.org account, you better start collecting signatures of your key

Why Do You Need a PGP Key?

- Why do you need a PGP key?
 - If you send pull requests to your upstream maintainer, hopefully you use PGP-signed git tags (`git tag -a -s`)
 - Pull requests should not use branches
 - Linus expects signed tags for any pull not using kernel.org infrastructure
 - If you turned on signed pushes for git.kernel.org transparency log
- Your key does not need to be signed by others to be useful
- However eventually it should be signed, to participate in the Web of Trust
- If you ever want to apply for kernel.org account, you better start collecting signatures of your key
- Comprehensive guide on setting PGP/GnuPG keys (including smartcard):
<https://www.kernel.org/doc/html/latest/process/maintainer-pgp-guide.html>



Signing Keys

- In-person signing

Signing Keys

- In-person signing
 - Where? On a conference or let's meet for a beverage of your choice. You can find us:
 - <https://www.kernel.org/doc/ksmap/>
 - <https://korg.docs.kernel.org/ksmap.html>

Signing Keys

- In-person signing
 - Where? On a conference or let's meet for a beverage of your choice. You can find us:
 - <https://www.kernel.org/doc/ksmap/>
 - <https://korg.docs.kernel.org/ksmap.html>
 - How?
 - Add your key to <https://keys.openpgp.org> and confirm email identities
 - Or to the kernel.org keyring, more on this on next slide
 - Print paper slips with your PGP key: UIDs, Key ID and full HEX fingerprint
 - \$ gpg --fingerprint
 - Give printed paper slips and show your government ID
 - Wait for signed keys sent by email and apply the signatures to your key

Signing Keys

- In-person signing
 - Where? On a conference or let's meet for a beverage of your choice. You can find us:
 - <https://www.kernel.org/doc/ksmap/>
 - <https://korg.docs.kernel.org/ksmap.html>
 - How?
 - Add your key to <https://keys.openpgp.org> and confirm email identities
 - Or to the kernel.org keyring, more on this on next slide
 - Print paper slips with your PGP key: UIDs, Key ID and full HEX fingerprint
 - \$ gpg --fingerprint
 - Give printed paper slips and show your government ID
 - Wait for signed keys sent by email and apply the signatures to your key
- Video conference call signing
 - <https://korg.docs.kernel.org/accounts.html#keysigning-via-video-conferencing>

Sharing Signatures of Maintainers

- The signatures are so far only local so how anyone would see your key was signed?

Sharing Signatures of Maintainers

- The signatures are so far only local so how anyone would see your key was signed?
- Share your updated PGP key (with signatures)

Sharing Signatures of Maintainers

- The signatures are so far only local so how anyone would see your key was signed?
- Share your updated PGP key (with signatures)
 - However not through a keyserver
 - They either strip signatures or should not be used (due to possible flood of bogus signatures)

Sharing Signatures of Maintainers

- The signatures are so far only local so how anyone would see your key was signed?
- Share your updated PGP key (with signatures)
 - However not through a keyserver
 - They either strip signatures or should not be used (due to possible flood of bogus signatures)
 - Use Kernel developer PGP keyring

Sharing Signatures of Maintainers

- The signatures are so far only local so how anyone would see your key was signed?
- Share your updated PGP key (with signatures)
 - However not through a keyserver
 - They either strip signatures or should not be used (due to possible flood of bogus signatures)
 - Use Kernel developer PGP keyring
 - Caveat #1: If you **regularly contribute code to the Linux kernel**

Sharing Signatures of Maintainers

- The signatures are so far only local so how anyone would see your key was signed?
- Share your updated PGP key (with signatures)
 - However not through a keyserver
 - They either strip signatures or should not be used (due to possible flood of bogus signatures)
 - Use Kernel developer PGP keyring
 - Caveat #1: If you **regularly contribute code to the Linux kernel**
 - Please read the full guide first:
<https://korg.docs.kernel.org/pgpkeys.html>

Sharing Signatures of Maintainers

- The signatures are so far only local so how anyone would see your key was signed?
- Share your updated PGP key (with signatures)
 - However not through a keyserver
 - They either strip signatures or should not be used (due to possible flood of bogus signatures)
 - Use Kernel developer PGP keyring
 - Caveat #1: If you **regularly contribute code to the Linux kernel**
 - Please read the full guide first:
<https://korg.docs.kernel.org/pgpkeys.html>
 - TLDR:
 - Export your signed public PGP key and send it to: keys@linux.kernel.org
 - Caveat #2: minimum two signatures from keys already in the kernel.org keyring, not too far away from Linus

Dump the Mailing Lists (also called: lei+lore)



Public-inbox

- public-inbox
 - *Sharing of an email inbox via git to complement or replace traditional mailing lists. Readers may read via NNTP, IMAP, POP3, Atom feeds or HTML archives.*
 - <https://public-inbox.org/README.html>
 - lei
 - The command line tool from public-inbox project
 - lore.kernel.org
 - The server exposing mailing lists / public-inbox



Public-inbox

- public-inbox
 - Sharing of an email inbox via git to complement or replace traditional mailing lists. Readers may read via NNTP, IMAP, POP3, Atom feeds or HTML archives.
 - <https://public-inbox.org/README.html>
 - lei
 - The command line tool from public-inbox project
 - lore.kernel.org
 - The server exposing mailing lists / public-inbox
- Why?
 - Because the mail never stops! It just keeps coming and coming and coming. There's never a letup, it's relentless.



Public-inbox

- public-inbox
 - *Sharing of an email inbox via git to complement or replace traditional mailing lists. Readers may read via NNTP, IMAP, POP3, Atom feeds or HTML archives.*
 - <https://public-inbox.org/README.html>
 - lei
 - The command line tool from public-inbox project
 - lore.kernel.org
 - The server exposing mailing lists / public-inbox
- Why?
 - You can follow mailing lists without actually subscribing to them
 - You can also follow specific topics, like changes to specific files

Public-inbox

- public-inbox
 - Sharing of an email inbox via git to complement or replace traditional mailing lists. Readers may read via NNTP, IMAP, POP3, Atom feeds or HTML archives.
 - <https://public-inbox.org/README.html>
 - lei
 - The command line tool from public-inbox project
 - lore.kernel.org
 - The server exposing mailing lists / public-inbox
- Why?
 - You can follow mailing lists without actually subscribing to them
 - You can also follow specific topics, like changes to specific files
- There is a great talk explaining all this:
 - “Doing more with lore and b4”, <https://lpc.events/event/11/contributions/983/>

Public-inbox - lei

- lei will fetch emails from mailing lists, according to your search query, and store them:
 - On the IMAP server
 - You can directly sync to your Gmail account!
 - Authentication data obtained from git-credentials (/usr/share/doc/git/contrib/credential/libsecret/ as credential.helper)

Public-inbox - lei

- lei will fetch emails from mailing lists, according to your search query, and store them:
 - On the IMAP server
 - You can directly sync to your Gmail account!
 - Authentication data obtained from git-credentials (/usr/share/doc/git/contrib/credential/libsecret/ as credential.helper)
 - In local Maildir



Public-inbox - lei

- lei will fetch emails from mailing lists, according to your search query, and store them:
 - On the IMAP server
 - You can directly sync to your Gmail account!
 - Authentication data obtained from git-credentials (/usr/share/doc/git/contrib/credential/libsecret/ as credential.helper)
 - In local Maildir
- Search query following a Devicetree list and syncing to the IMAP folder:

```
lei q -I https://lore.kernel.org/all/ --threads --dedupe=mid -jobs=,2 \  
-o imaps://imap.gmail.com/LKML/dt \  
'l:devicetree.vger.kernel.org AND rt:2.month.ago..'
```

Public-inbox - lei

- lei will fetch emails from mailing lists, according to your search query, and store them:
 - On the IMAP server
 - You can directly sync to your Gmail account!
 - Authentication data obtained from git-credentials (/usr/share/doc/git/contrib/credential/libsecret/ as credential.helper)
 - In local Maildir
- Search query following a Devicetree list and syncing to the IMAP folder:

```
lei q -I https://lore.kernel.org/all/ --threads --dedupe=mid -jobs=,2 \  
-o imaps://imap.gmail.com/LKML/dt \  
'l:devicetree.vger.kernel.org AND rt:2.month.ago..'\  
  
lei up --all
```

Public-inbox - lei with diff-filename

- Looking for changes to specific files?

```
lei q -I https://lore.kernel.org/all/ --threads --dedupe=mid -jobs=,2 \  
-o imaps://imap.gmail.com/LKML/test/my-lovely-driver \  
'dfn:ufs-qcom AND rt:2.month.ago..'
```



Public-inbox - lei with diff-filename

- Looking for changes to specific files?

```
lei q -I https://lore.kernel.org/all/ --threads --dedupe=mid -jobs=,2 \  
-o imaps://imap.gmail.com/LKML/test/my-lovely-driver \  
'dfn:ufs-qcom AND rt:2.month.ago..'
```

```
lei q -I https://lore.kernel.org/all/ --threads --dedupe=mid -jobs=,2 \  
-o imaps://imap.gmail.com/LKML/test/regulators \  
'dfn:drivers/regulator/ AND rt:2.month.ago..'
```


Public-inbox - lei with diff-filename

- Looking for changes to specific files?

```
lei q -I https://lore.kernel.org/all/ --threads --dedupe=mid -jobs=,2 \  
-o imaps://imap.gmail.com/LKML/test/my-lovely-driver \  
'dfn:ufs-qcom AND rt:2.month.ago..'
```

```
lei q -I https://lore.kernel.org/all/ --threads --dedupe=mid -jobs=,2 \  
-o imaps://imap.gmail.com/LKML/test/regulators \  
'dfn:drivers/regulator/ AND rt:2.month.ago..'
```

- More on lei and lore:
 - <https://lwn.net/Articles/878205/>
 - https://lore.kernel.org/all/_text/help/

References



References (1)

- 0-day / LKP
 - <https://github.com/intel/lkp-tests/blob/master/doc/faq.md>
 - <https://github.com/intel/lkp-tests/wiki/Repo-Spec>
- *Doing more with lore and b4*, Konstantin Ryabitsev
 - <https://lpc.events/event/11/contributions/983/>
- b4
 - ``pip install b4``
 - <https://b4.docs.kernel.org/en/latest/installing.html>
 - <https://git.kernel.org/pub/scm/utils/b4/b4.git/tree/thanks-am-template.example>
- git commit hooks
 - https://github.com/krzk/tools/blob/master/linux/verify_fixes.sh
 - https://github.com/krzk/tools/blob/master/linux/verify_signedoff.sh
 - <https://github.com/krzk/tools/blob/master/linux/git-hooks-post-commit>
- git.kernel.org repo appearance
 - <https://korg.docs.kernel.org/cgit-meta-data.html>
- git.kernel.org transparency log
 - <https://korg.docs.kernel.org/gitolite/transparency-log.html>



References (2)

- PGP keys guide
 - <https://www.kernel.org/doc/html/latest/process/maintainer-pgp-guide.html>
- Kernel developer PGP keyring
 - <https://korg.docs.kernel.org/pgpkeys.html>
- Signing PGP keys
 - <https://korg.docs.kernel.org/accounts.html#keysigning-via-video-conferencing>
- Kernel key signing map
 - <https://www.kernel.org/doc/ksmap/>
 - <https://korg.docs.kernel.org/ksmap.html>
- lore
 - <https://lore.kernel.org/>
- lore and lei search syntax
 - https://lore.kernel.org/all/_/text/help/
 - <https://lwn.net/Articles/878205/>
- public-inbox
 - <https://public-inbox.org/README.html>





Thank you



Linaro
Developer Services

Introducing Linaro

Linaro collaborates with businesses and open source communities to:

- Consolidate the Arm code base & develop common, low-level functionality
- Create open source reference implementations & standards
- Upstream products and platforms on Arm

Why do we do this?

- To make it easier for businesses to build and deploy high quality and secure Arm-based products
- To make it easier for engineers to develop on Arm

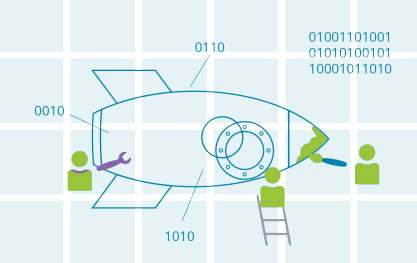
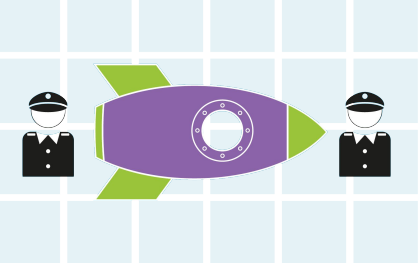
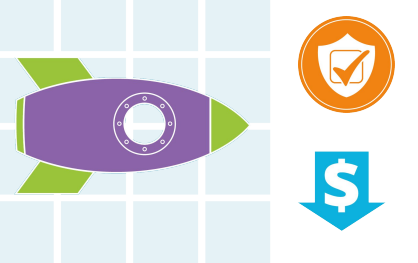

Two ways to collaborate with Linaro:

- 1 Join as a member and work with Linaro and collaborate with other industry leaders
- 2 Work with Linaro Developer Services on a one-to-one basis on a project

For more information go to: www.linaro.org

Linaro Developer Services

Linaro Developer Services helps companies build, deploy and maintain products on Arm

Arm Software expertise	Specialists in TEE on Arm	Continuous Integration through LAVA	Build, Test and deploy faster
			
As part of Linaro, Developer Services has some of the world's leading Arm Software experts .	We specialize in security and Trusted Execution Environment (TEE) on Arm.	We offer continuous integration (CI) and automated validation through LAVA (Linaro's Automation & Validation Architecture)	We support every aspect of product delivery, from building secure board support packages (BSPs), product validation and long-term maintenance.

For more information go to: <https://www.linaro.org/services/>

Linaro membership collaboration

