Status of Media CI and Multi-committers

Hans Verkuil
Cisco Systems Norway

Terminology

- Developer: you post patches.
- Reviewer: you also review patches from others.
- Driver Maintainer: you are also responsible for one or more drivers (typically because you are the author), and review patches for those drivers. You are listed in the MAINTAINERS file.
- Media Maintainer: you are responsible for a group of drivers, you keep patchwork up to date, and create Pull Requests whenever the patches are considered ready for merging. Experience with drivers from multiple vendors is important, and you must be able to act independently from companies.
- Media Core Maintainer: you are also responsible for a large area of the subsystem, and the corresponding core frameworks, and create Pull Requests for changes to those core frameworks.
- Media Subsystem Maintainer: you are responsible for the whole subsystem, you process posted Pull Requests and merge them, and make Pull Requests for Linus.
- Linus: God

Media Maintainer Duties

1) Keep patchwork up to date, primarily regarding the patches within your domain. But it is very much appreciated if you also check if any undelegated patches can be marked Superseded or Not Applicable if you have some time.

2) When reviewing: be nice[tm]. No need to go overboard, but there is no need to be rude either. If you reject something, explain why. If someone repeatedly ignores your comments, then it is OK to point that out, but politely.

3) Allow for wildly different language skills: avoid using potentially obscure idioms yourself that might be misunderstood, and if the author seems rude, assume it is due to lack of English language skills.

4) Check code quality, check that the code does the right thing based on your domain expertise. If you think review from other experts is required, reach out to them. Changes to the core framework always need additional review. It's your responsibility to pull in the right devs.

5) Don't wait too long with reviews. If you know it will take a long time, then let the author know, or perhaps ask someone else to help out.

Media Maintainer Duties

6) When you are happy with it run it through Media CI. Note that Media-CI doesn't enforce checkpatch.pl, so you should check the results to see if there are things that really need to be addressed.

7) Someone else needs to review the patches you authored. Reach out to one or more devs (either irc or email) to ask for a review on the mailing list.

8) Keep track of the mainline release schedule: no major changes after rc6 (fixes are always allowed at any time). For fixes after -rc6: notify the subsystem maintainers (Mauro & Hans) as we need to create upstream PRs for that.

9) At least once per kernel cycle go through all the 'random patches' that people post, often from static analysis tools. Always review these carefully: 40-60% of these patches are bad.

Media Maintainer Duties

10) Keep patchwork up to date. Really!

Media CI

- Problem: the Media Subsystem Maintainers have to deal with all PRs, and any patches that fall through the cracks. Too much work for too few people.
- We want to move to a similar multi-committer model as the DRM subsystem uses: selected Media Maintainers can merge code directly into the media committers tree without having to post a PR.
- More committers requires that code goes through a standard set of tests: Media CI. This way nobody forgets to do all the checks.
- It took longer than expected, but is it now in good shape. We're still finding corner-cases where it fails patches when it shouldn't, but Ricardo is fantastic at fixing those very quickly.
- Media CI is no longer a blocker, it is now ready to deal with more than two committers.

Media CI: Two Issues

- When to use a Cc to stable if a Fixes tag is present remains problematic: CI tried to find the SHA referred to by the Fixes tag in linux-stable and, if not found, assumes no Cc is needed (i.e. it is a fix for something not yet in mainline). But linux-stable seems to take non-mainline patches as well, so the SHA is found even if it is not in mainline.

  1) Look in the mainline kernel instead,

  2) Just warn if there is a Fixes tag but no Cc to stable,

  3) Always require a Cc to stable if there is a Fixes tag.

- static-mchehab job: using Mauro's version of smatch: often out of date compared to the upstream version, and for me it never gave information that the upstream smatch didn't.

  1) Remove this job completely,

  2) Ignore failures (just like the checkpatch job)

Demo

Multi-Committers: Status

- Documentation.