2026-04-18 LixCon meetup

- Date: 2026-04-18
- Participants: raito, rootile, piegames, Jade, lunaphied, qyriad, horrors

Agenda

- Code owners
- +2/-2 review semantics (somewhat adjacent to code owners)
- AI policy

Code owners

- piegames: I want to see some consensus and have written it down; what code ownership means to us? which permissions go with that? how do we distribute +1/+2 rights?
- jade: at the technical level, code ownership just sets recommended reviewers for a tree and also sets who is required to approve a change for a tree
  - +2 does not override codeowners
- piegames: so we can give +2 while keeping code owners coverage or we have +2 rights and code owners who are +2 rights for something?
- jade: those two looks the same to me because if you give a way +2 candy and you also give code owners of particularly directory like candy, this doesn't give particular broad merge capabilities; if you give +2 and code owners to F2 maintainers, they can merge tests
- [description of the system]
- piegames: it looks like the current system is quite complicated, is it worth keeping this level of flexibility?
- raito: i still think that we do not have the current right configuration for code owners, esp. wrt to security
- jade: if people self merge, that's bad and should be audited
- horrors: but the releng process has shown prior self merges
- jade: but this is automatically generated
- horrors: yes but that doesn't exempt it from code review
- [could not catch the rest of the conversation]
- piegames: now i understand how it works but i would like to think about how should it work; i would like a more traditional style
- jade: what we have at work is we have gh owners and it setups so it send pings if certain files changes, these people are not required to approve to merge things; you get someone from some other team and goes stamp a change to your code and that other engineer merges it within 15 minutes, this is obviously a culture failing but i also think this does happen in nixpkgs, code owners are not blocking
- piegames: ok, i think then code owners are doing too many things at the same time; assuming code owners are blocking code owners, does that allow them to merge only

code they own? should it?

- rootile: i.e. Should code-owners have scoped +2 permission on their ownership
  - jade note: (this is what happens when someone has "+2" on gerrit in general. we could eliminate +2 rights as a requirement altogether and just use codeowners. then have global approvers list?)
- Qyriad: if the answer to that question is no, either we can give +2 for that purpose or i would suggest having a model where say: if two different code owners of the same tree approve the change,
  - Raito: s/2/N where N is number of codeowners-trees changed
- piegames: What even are "committers" in this model?
- rootile: Keeping ownership dynamically up-to-date will require some maintenance and might lag behind
- Raito: what do you mean by "governance overhead"?
  - My vision is like: you're working on the subsystem a lot, and then you're absent for a while, right?
  - In a reasonable amount of time it'd be good to communicate that you're not going to be active and so you should update your OWNERS file
  - rootile: Who's responsible for poking people to check their OWNERS file?
  - Raito: could make an automated proposal after N inactivity
    - Yes it is bookkeeping but if the bookkeeping provides value then it's worth it
    - jade note: some trees don't move that fast and so idk if it should be per-tree but maybe across the entire repo
  - jade: see also: the psychological fear of, say, adding new owners to a tree
- horrors: any meaningful change should go through a short discussion
- Qyriad: any meaningful modification of the code owners should probably NOT go solely through Gerrit
- Raito: I think the initial vision behind, e.g., handing out these things easily, was to make onboarding people much more accessible (unlike, say, CppNix)
  - However we've been handing out access but that hasn't corresponded to people taking responsibility on these things
    - Qyriad: It doesn't help that people weren't clear on what codeowners meant
  - So now we have accumulated ownership but people may or may not really care
- Raito:
  - Re: docs on these things
- jade: agreement with those present that we can delegate ownership of things like tests to the maintainers of those areas. ie. functional2 owners can add more functional2 owners at a governance level.
- piegames wishlist:
  - List of all code owners -> where? global approvers?
    - OWNERS files in the Lix tree
      - -> jade note: backports, old branches? probably need an all-branches thing for global approvers at least
    - some form of merged summary?
  - List of all people with +2 rights
    - refs/meta/config in the Lix tree (requires some privileges to read)
  - Documentation of +2 and codeowners

    - Documentation of submit requirements
    - Automatically keep that documentation up to date
    - A definition of "committers" for the purpose of our governance document

Action items

```
- [piegames] Write documentation
- [raito] go through the permissions and review them
- [raito] make `refs/meta/config` public
```

+2/-2 review semantics

- raito: +1 is someone else should look at it, +2 is this can go, -1 is meh, needs more discussion, -2 is strong disagreement, might require escalation to core team vote
- Core question: is -2 disagreement with the change in its current form, or with the general idea of the change
- rootile: my interpretation is that -2 means "the concept is fundementally flawed and this should not be worked on again"
- jade: -2 is a blocker, the reviewer needs to re-review and aggree; -2 does not mean bad concept
- piegames: my interpretation is: -1 is a fixable issue, -2 is an unfixable/needs major rework issue
- piegames: From a contributor perspective, -2 might be a critical but easily fixable issue, or it might be a fundamental rejection of the change, and only comments will tell them apart. This is a bad experience
- jade: -2 is just the "request changes" thing on github
- qyriad: i want a distinction between desirable and undesirable
- rootile: Suggestion: Introduce -3 to distinguish those
    - Qyriad: when would we need -3?
        - rootile: say something was ported from CppNix that is just not something we want at all
            - Raito: we need to document that
- piegames: seems like blocking vs unlocking is kind of orthogonal
    - Raito: maybe we should make Unresolved comments blocking?
- Agreement on current state and change being necessary; though not on a solution
- needs re-discussion

Action items

- No general agreement found, will have to rediscuss

☐ Rename abandon to close (maybe fuck with the translation file)

☐ Change the label on -2 and -1 to e.g. "... see comment"

☐ Allow committers to abandon/close CLs by other people (document that doing this is generally rude and should be avoided)

Deferred

- Final semantics of -1/-2

AI Policy

Strongly shortened version:

- Anthropic has been working on things that would probably be good for Lix, but they have been unable to contribute to Lix
  - e.g., a system to compile code in a fully trusted and sandboxed and enclaved way; supply chain security stuff
- Lix is an AFNix project. as-of rn, lix is inheriting the AI policy of AFNix
- if lix wants to deviate, lix needs to discuss it with AFNix
- I'm strongly against allowing any kind of generated code within changes because once we allow some amount of AI-generated code, people will be tempted to find out the limits.
  - People will submit slop if we accept anything at all.
  - The act of dealing with bad changes is a lot of emotional/review capacity work.
- How do people become an expert? Easy contributions *should not be done with AI*, should be done by new contributors.
- If we do the easy work with AI as SMEs, we starve the newcomers of opportunities to do things.
- Having to switch to another infra than AFNix would be a lot of work! Their board is not interested in changing the AI policy.
- All vulns should be hand tested before submission.
- If your thing is good enough, it's not possible to know it's AI generated
- We might want to recommend against trying to use AI to understand the codebase because it might lie.
- **Everyone present agrees with the AFNix policy.**

Revision #1
Created 2026-05-13 20:08:36 UTC by piegames
Updated 2026-05-13 20:22:26 UTC by piegames