printf("Hello, srcmgr\n");

BY MARK JOHNSTON

FreeBSD developers and devsummit attendees have likely heard of the newish srcmgr ("source manager") team at some point. But seeing as it's been a year or so since srcmgr started having regular calls, it seems time to introduce ourselves a bit more widely.

srcmgr is a team of src developers whose goal is to help organize FreeBSD src development. For context, FreeBSD's development model is somewhat unusual among OSS projects: rather than having an individual or small team of developers who direct the project and make high-stakes decisions, FreeBSD's src developers belong to a relatively flat hierarchy, subject, of course, to maintainership rules and conventions. Developers are collectively obliged to help push along src development, be it by finding or fixing bugs, writing documentation, reviewing code, adding features and tests, etc.

This development model works reasonably well — consider, for instance, that the FreeBSD project itself is older than some of our developers — but it has shortcomings too. Individual developers have few formal responsibilities; they are expected to contribute and work with each other, but there is little formal oversight. This model works well with small groups of developers who know and trust each other and can motivate one another; unsurprisingly, this characterized FreeBSD development during its early days. Over time, however, challenges arise. Long-time developers move on, the system grows more complex and harder to maintain, and the number of new developers increases, putting strain on experienced mentors, who typically have little free time.

FreeBSD's src developers belong to a relatively flat hierarchy

Historically, the FreeBSD Core Team provided fallback support: whenever there was a dispute or a problem with a neglected part of the tree, they would step in. This worked historically, as Core was for a long time mainly composed of src developers despite representing all of FreeBSD. But in the last couple of terms, the ports team has gained more representation within the Core team. Thus, recent Core teams have had fewer resources to devote to src-specific issues, and Core's attention should be focused on the project's long-term strategic direction rather than day-to-day matters.

Enter srcmgr. srcmgr was officially announced on the internal FreeBSD developer mailing list on October 8th, 2024, and currently consists of me, Ed Maste, Warner Losh, and John Baldwin. We also have five "lurkers," developers who attend srcmgr calls and participate without formally being srcmgr members; they aim to test the waters and decide whether they want to commit to becoming official members. At a glance, srcmgr plays the same role in the src tree as the portmgr and doceng teams do in the ports and doc trees, respectively: we try to provide oversight and help tackle challenges specific to our area.

Although the idea for a srcmgr team had been floated before, my first exposure came from conversations with John Baldwin and Ed Maste at BSDCan in 2023. It was recognized that Core tended to be overburdened and not have the capacity to do proactive work to

shepherd src development. Meanwhile, I was frustrated by our overall handling of bug reports, new contributors, and code reviews. As an individual developer, it was too much work to keep on top of everything while also doing regular paid work.

So, what does srcmgr do in practice? Well, our charter gives an outline. We meet every two weeks for roughly two hours; the first hour is spent reviewing agenda items, discussing them, and providing status updates. The second hour is typically spent triaging recent src bug reports and/or GitHub pull requests.

The primary responsibility of the group is to vote on src commit bits: when an src committer has been working with a contributor and believes that the contributor would make good use of src commit access themselves, they can send a proposal to srcmgr, who then votes on whether to grant the commit bit. This process tends to be uncontroversial and straightforward and consumes little time in practice.

We also spend time on "maintenance" tasks, such as disabling commit bits of inactive developers, pushing forward the deprecation of obsolete or unmaintained features that consume project resources, and updating developer policies and — with much help from FreeBSD's cluster admin team — bits of infrastructure such as git commit hooks.

The primary responsibility of the group is to vote on src commit bits

Most of our time is spent pushing along various initiatives, the principal aims being to 1) make experienced developers more productive, and 2) make it easier for newcomers to contribute. For the first goal, we have tried to push the creation of tools to make FreeBSD development easier. Today, it is far too difficult for new FreeBSD developers (interested contributors, GSoC students, employees at FreeBSD-using companies, etc.) to set up an environment where they can quickly and reliably test changes to the src tree. We have an extensive regression test suite, but actually running it requires a fair bit of setup and is tricky to automate. Setting up an efficient, interactive compile-edit-test loop is also tricky. Many developers have custom scripts and workflows to enable this, but that means that many developers end up reinventing the wheel; moreover, we do not have a good "canned" setup that newcomers can quickly adopt and customize. Solving this problem in general is a challenge, as FreeBSD is a large codebase with many components that require specialized development approaches. Still, there is a lot of room for improvement.

Following the theme of tooling, we have also been working on scripts to make MFCs easier and to programmatically catch certain classes of problems, such as missing backports on stable branches, particularly when commit B fixes a bug in commit A but is missed when merging commit A. Another area of focus is triaging incoming work for project members: bug reports, code review requests, and contributor patches. While individual FreeBSD developers spend a lot of time handling day-to-day requests of this nature, there is little oversight that ensures that high-priority issues don't fall through the cracks. srcmgr is composed of developers who are familiar with the src tree and can quickly identify who should be "tagged" on a particular issue to help move it forward.

Finally, one ongoing initiative is the hosting of "bug-busting" sessions, typically on Zoom or meet.freebsd.org. We announce these sessions in advance on the developer mailing list and invite folks to join us for roughly 3 hours of triaging and bug work. I typically lead these sessions by reviewing individual bug reports and looking for opportunities to make quick progress: assigning them to a subject matter expert, asking follow-up questions of the submitter, and discussing the problem with others on the call. People are free to participate as they see fit; some will quietly work on specific bugs in the background while keeping an ear on the chatter, and others will follow along or triage bugs in parallel.

These sessions (and srcmgr participation in general) do a lot to keep me motivated: real-time interaction with other developers helps keep up engagement for most of us who work remotely, and being able to make fast, tangible progress on bugs and pull requests keeps the backlog from feeling overwhelming. It's much easier to take pride in the project when we can stay on top of incoming issues and work requests, and this collaboration helps maintain our sense of shared responsibility.

There is a lot more srcmgr would like to do to help the project, and we have a lot of initiatives that we will be pushing along in the coming year, especially as the FreeBSD 15.0 release date edges closer and we get some time to recharge over the holidays.

If you have any feedback or ideas, please always feel free to email us at: srcmgr@FreeBSD.org.

MARK JOHNSTON is a FreeBSD developer living in Toronto, Ontario, Canada. When not sitting at a computer, he enjoys playing in a city dodgeball league with friends.