ISOVALENT

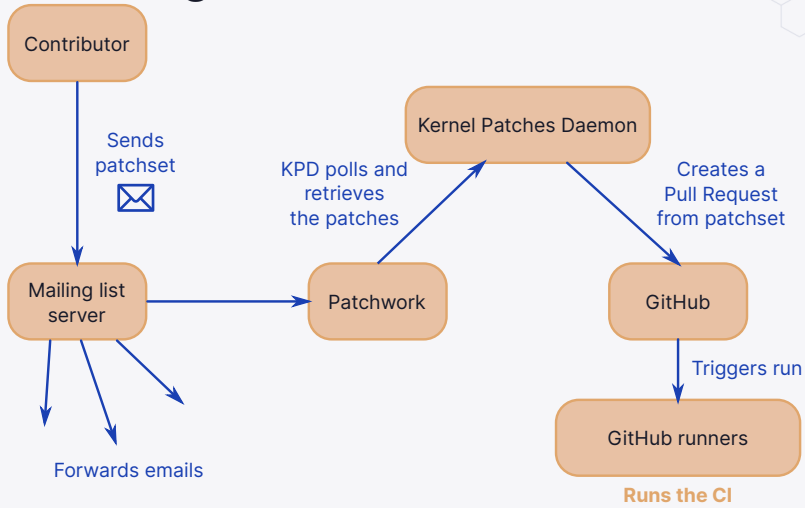
Reusing the BPF CI

Quentin Monnet
<quentin@isovalent.com>

Linux Plumbers Conference
14th September 2022

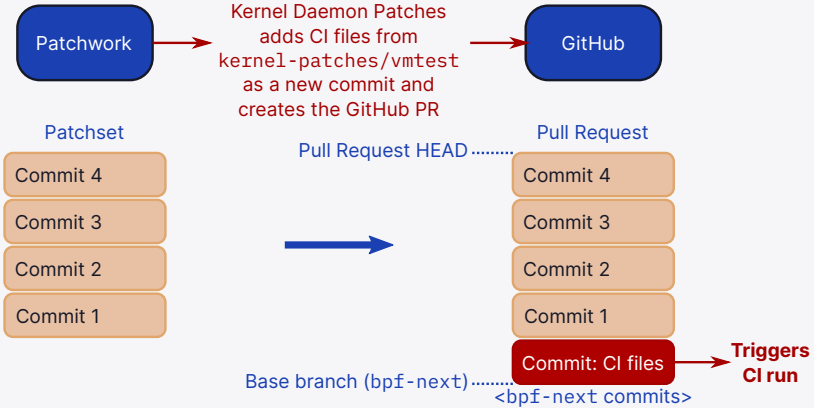
The BPF CI

Continuous Integration for BPF



- Runs since 2020, based after libbpf's, drgn's
- BPF subsystem (bpf@vger.kernel.org)

A Pull Request for running the CI



File `.github/workflows/test.yml` from `kernel-patches/vmtest` tells GitHub to run the workflow on each PR creation/update

The CI in a GitHub workflow

On PR creation/update, the workflow unfolds:

1. Download Linux kernel sources, patch if necessary
2. Install LLVM (APT repositories), pahole (build locally)
3. Build kernel, selftests, samples
4. Prepare rootfs
   - Download base image
   - Install kernel in image
   - Copy selftests and `kernel-patches/vmtest/` to image
5. Run tests in QEMU (mostly)
   - bpftool consistency checks
   - `test_progs`
   - `test_progs-no_alu32`
   - `test_maps`
   - `test_verifier`
   - (No `bpf_testmod.ko`)

More details on BPF CI

- bpfconf 2022: Mykola Lysenko,
  BPF CI current state and next steps
- Kernel Recipes 2022: David Vernet,
  Checking your work: validating the kernel by building and testing in CI

Reusing the CI

Running the BPF CI basically comes down to **adding files** to a patchset before **creating a Pull Request** on GitHub.

I can **do the same**, with **my own Linux fork** on GitHub.

Motivations for reusing the CI on another fork

- Run the CI prior to submitting
  - Raise mistakes early, don’t waste reviewers’ time
- Setup? Leave it to the CI
  - No need to install setup (LLVM/clang, pahole, etc.)
  - Run with a “blessed” kernel config
  - Generally speaking, no setup issue (`vmtest.sh` helps a lot, but not perfect)
- Run remotely on GitHub’s runners
  - No need to access a development machine (when traveling)
  - Save resources on own machine

[CI run] bpf: Fix a few typos in BPF helpers documentation #15

Open **qmonnet** wants to merge 3 commits into `bpf-next` from `ci-test/pr/helpers-doc-typos`

Conversation 0 · Commits 3 · Checks 12 · Files changed 8

qmonnet commented 16 days ago — Owner

No description provided.

qmonnet added 3 commits 16 days ago

- `bpf: Fix a few typos in BPF helpers documentation` — `04129ad`
- `add CI files` — `052d7a2`
- `disable self-hosted runners` — `c31485f`

Add more commits by pushing to the **`ci-test/pr/helpers-doc-typos`** branch on **qmonnet/linux**.

All checks have passed
12 successful checks — Hide all checks

- **bpf-ci / llvm-toolchain (pull_request)** Successful in 2s — Details
- **bpf-ci / set-matrix (pull_request)** Successful in 1s — Details
- **bpf-ci / build for x86_64 with gcc (pull_request)** Successful in 27m — Details
- **bpf-ci / build for x86_64 with llvm-16 (pull_request)** Successful in 30m — Details
- **bpf-ci / test_progs on x86_64 with gcc (pull_request)** Successful in 21m — Details
- **bpf-ci / test_progs_no_alu32 on x86_64 with gcc (pull_request)** Successful in 15m — Details

This branch has no conflicts with the base branch
Merging can be performed automatically.

A “reuse-friendly” CI?

… Suddenly, everything breaks

Preparing the CI run can be automated (*that's the point!*)

- Cooked myself a script, qmonnet/bpf-ci-check
- August 2022: Preparing blog post
- Runs CI on a patchset to take a screenshot
- … CI fails to launch!

… Suddenly, everything breaks

Preparing the CI run can be automated (*that's the point!*)

- Cooked myself a script, qmonnet/bpf-ci-check
- August 2022: Preparing blog post
- Runs CI on a patchset to take a screenshot
- … CI fails to launch!

July 2022: Change in workflow files, CI on self-hosted runners

- "*Let's make sure to only use self-hosted runners*"
- Motivation: More powerful runners, more architectures: s390x (big-endian)
- Easy to "fix", but I have to edit the workflow files
- (Later lifted when running under accounts/orgs other than `kernel-patches`—thanks!)

Reusing the CI: status

Discussion with Daniel Müller, takeaways:

- CI actively being worked on, changes/breakages are to be expected
- So far, not designed with external reuse in mind
- Ideally: have scripts available for reuse instead of hooking into the entire infrastructure

Open questions

So we can reuse the BPF CI on a Linux fork on GitHub.

- How to make the CI easier to reuse?
  - *Disclaimer: I don't lead the project!*
  - Keep consolidating between libbpf, vmtest repositories
  - Move workflow steps to reusable, versioned Actions?
  - Define a set of "guaranteed workflows"?
  - GitHub an issue for some contributors, but unlikely to change
- Who else might be reusing the CI in the future?
  - Other kernel subsystems (likely under `kernel-patches`)
  - cilium/linux: thinking of reusing it for development
  - bpftool: looking at possibilities for CI
  - Others?
- See also more generic Kernel CI project

Credits for the BPF CI

Main contributors:

- Andrii Nakryiko
- Daniel Müller
- Mykola Lysenko
- Sergey Iudin
- Yucong Sun
- ...

For the CI, the presentations, the discussions: Thank you!

ISOVALENT

Thank you!

GitHub repositories involved in CI