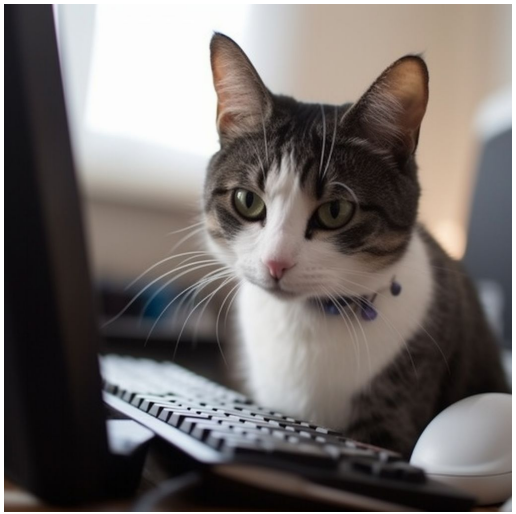
WIKIMEDIA FOUNDATION

The state of GitLab @wmf

Jelto Wodstrcil
Wikimedia Hackathon
Athens

Agenda

- Some history and facts about GitLab @WMF
- GitLab specifics
  - Types of CI Runners
  - Jobs scheduling
- New CI tooling
- Next steps
- Migration party



“cat sitting on a computer and using Wikipedia for the first time”

[1]: https://www.mediawiki.org/wiki/GitLab_consultation

GitLab and GitLab CI @ WMF

A short overview of what has happened and is happening around GitLab CI

- Community consultations revealed dissatisfaction with Gerrit and current CI stack[1]
- GitLab was evaluated as a industry standard code review system
  - lower friction to create new repositories
  - easier setup and self-service of Continuous Integration configuration
  - more familiarity with pull-request style workflows (like GitHub)

[1]: https://www.mediawiki.org/wiki/GitLab_consultation

GitLab and GitLab CI @ WMF

- GitLab is running for 1.5 years publicly[1]
- GitLab hosts 1100 projects and 700 users
- 4000 merge requests were created
- We are in the “Pipeline early adopters”[2] stage currently
  - Early adopters migration to GitLab
  - New projects explore GitLab
  - Add integration to existing tooling

[1]: https://gerrit.wikimedia.org/r/q/Icdc3a9d5920c2aec53336fa19023c5f57c4559ae
[2]: https://www.mediawiki.org/wiki/GitLab/Roadmap

GitLab and GitLab CI @ WMF

- GitLab is open! (mostly [1])
  - Write me, I can approve your account
- Login with wikitech at https://gitlab.wikimedia.org/

[1]: https://phabricator.wikimedia.org/T336711

GitLab CI

- “GitLab” is mostly referred to as the GitLab server (repos, projects, users, …)
- gitlab-runners are the individual CI job workers
- gitlab-runners can live in different environments
  - Local in docker
  - As a dedicated VM with docker or shell
  - In Kubernetes

“cat using a audio mixer”

GitLab CI

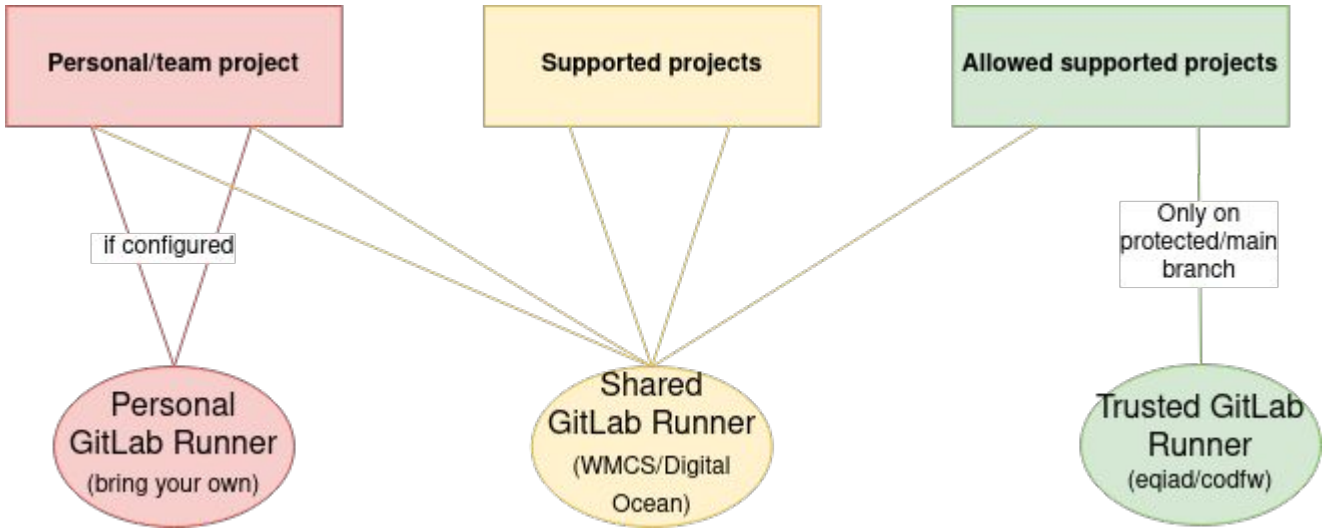
CI builds for open source project face a dilemma:

- We want secure and reliable builds and create trusted build artifacts
- Some builds are not safe to execute, because of malicious or unreliable code from untrusted sources
- CI should be easy to use for the community

Most of this problems are not new and there are “solutions”:

- Code review, trusted contributors
- multi-staged CI jobs, manual merges
- self-hosting

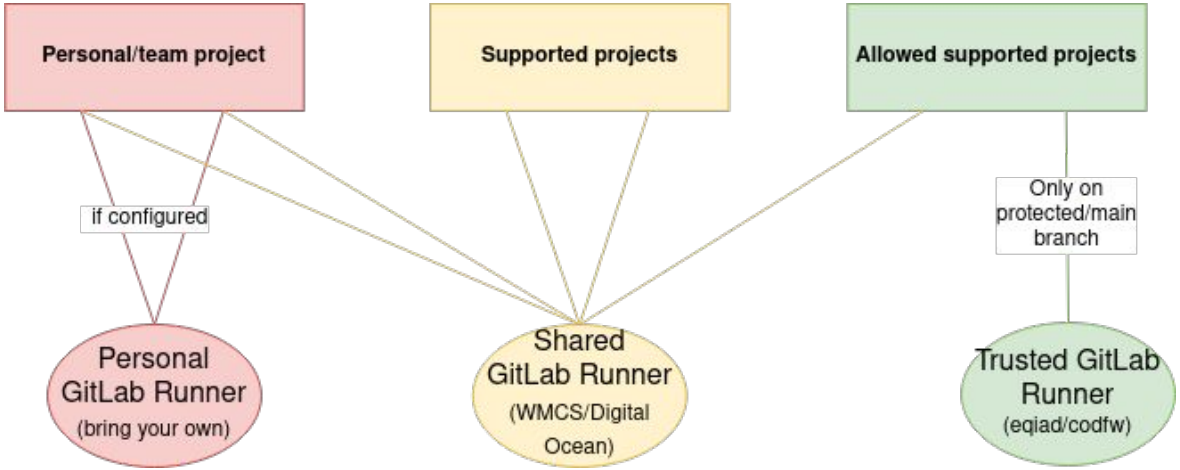
GitLab CI



https://wikitech.wikimedia.org/wiki/GitLab/Gitlab_Runner/Trusted_Runners#/media/File:GitLab_Runner_permission_model.png

GitLab CI

- Three-tier GitLab Runner setup
  - Self-managed Runners in your own project (optional)
  - Shared “Untrusted” Runners in WMCS and Digital Ocean
  - Trusted Runners in eqiad/codfw

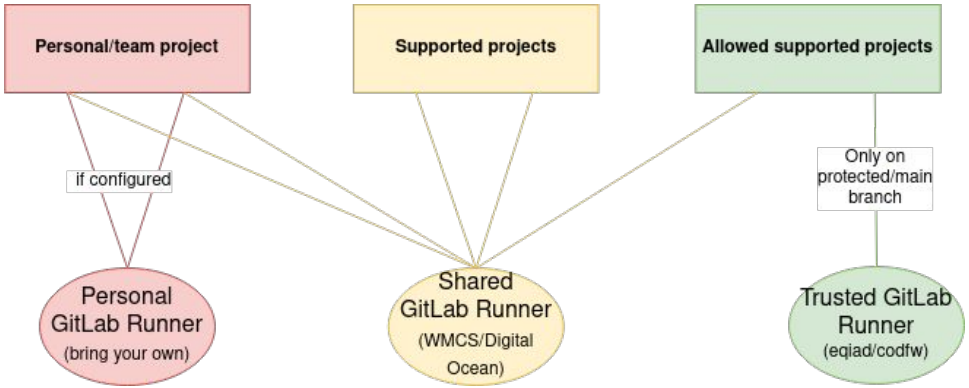


https://wikitech.wikimedia.org/wiki/GitLab/Gitlab_Runner/Trusted_Runners#/media/File:GitLab_Runner_permission_model.png

GitLab CI - Shared Runners

Shared Runners are available for most projects and execute any CI job

- Shared runners were restricted for all projects in /repos namespace, that restriction was lifted[1]
- Hosted on WMCS and Digital Ocean



[1]:https://phabricator.wikimedia.org/T297426

https://wikitech.wikimedia.org/wiki/GitLab/Gitlab_Runner/Trusted_Runners#/media/File:GitLab_Runner_permission_model.png

GitLab CI - Trusted Runners

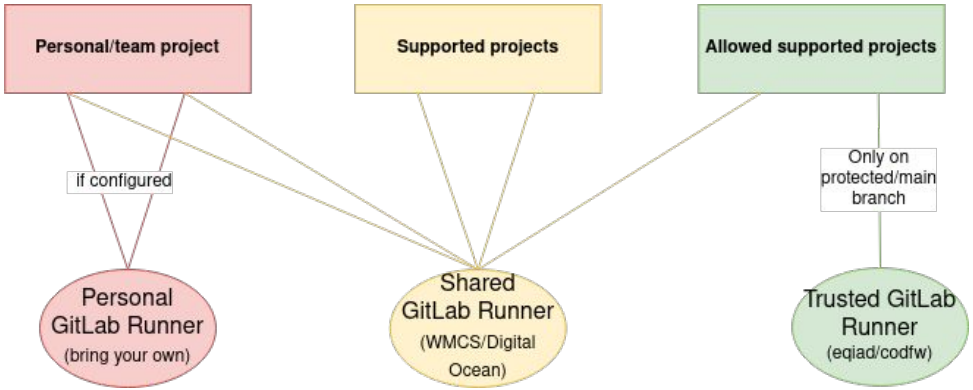
Trusted Runners accept jobs only from a explicitly allowed subset of sources (certain projects and branches). And it's harder to break out.

- Project has to be allowed by adding it to [gitlab-trusted-runner]
- merge and maintainer permission (Gerrit speak: "+2") "+2" required

Further resources:

https://wikitech.wikimedia.org/wiki/GitLab/Gitlab_Runner

https://gitlab.wikimedia.org/repos/releng/gitlab-trusted-runner



https://wikitech.wikimedia.org/wiki/GitLab/Gitlab_Runner/Trusted_Runners#/media/File:GitLab_Runner_permission_model.png

GitLab CI - Trusted Runners

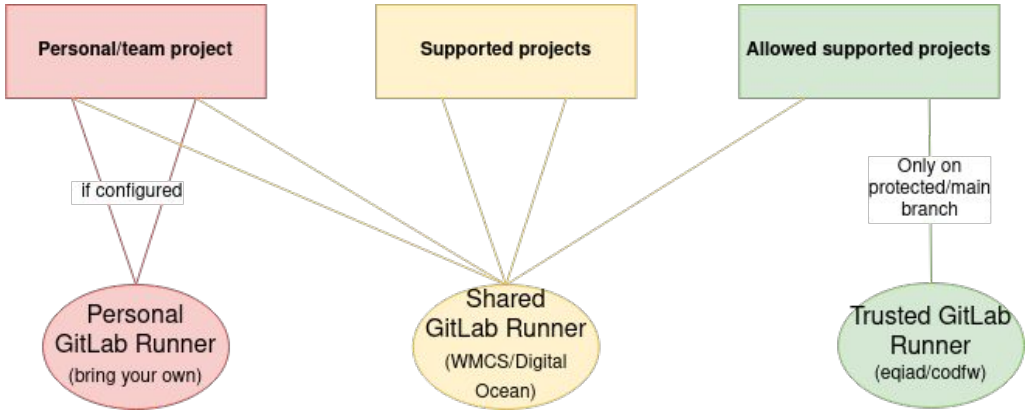
- Restricted access to Trusted Runners
- Hosted in trusted environment (production datacenters)
- Security hardening for Trusted Runners
  - CI jobs for protected branches only
  - Accept jobs with special, reviewed tags (job tags not git tags) only
  - Gitlab-runner is running as non-root
  - Only certain Docker images are allowed in CI jobs
  - Restrictive firewall settings
- External Security Review

“Cat as a senior system administrator”

GitLab CI - Self-managed Runners

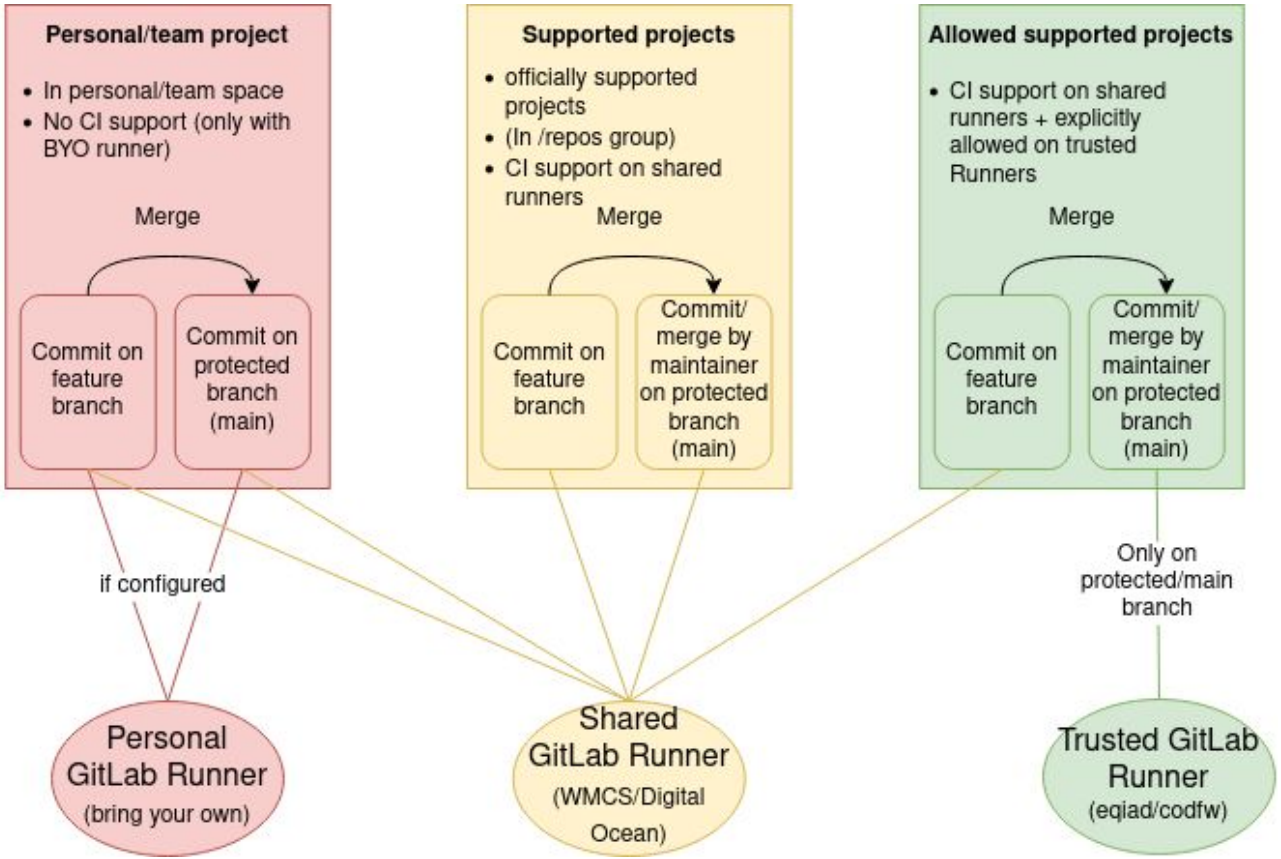
Project-specific self-maintained runners (for example with Docker on your dev machine)

- Useful if project has special requirements or wants to explore
- Runner has to be maintained by the individual team/project (maintenance overhead, security updates, reliability)
- If possible available runners should be used



https://wikitech.wikimedia.org/wiki/GitLab/Gitlab_Runner/Trusted_Runners#/media/File:GitLab_Runner_permission_model.png

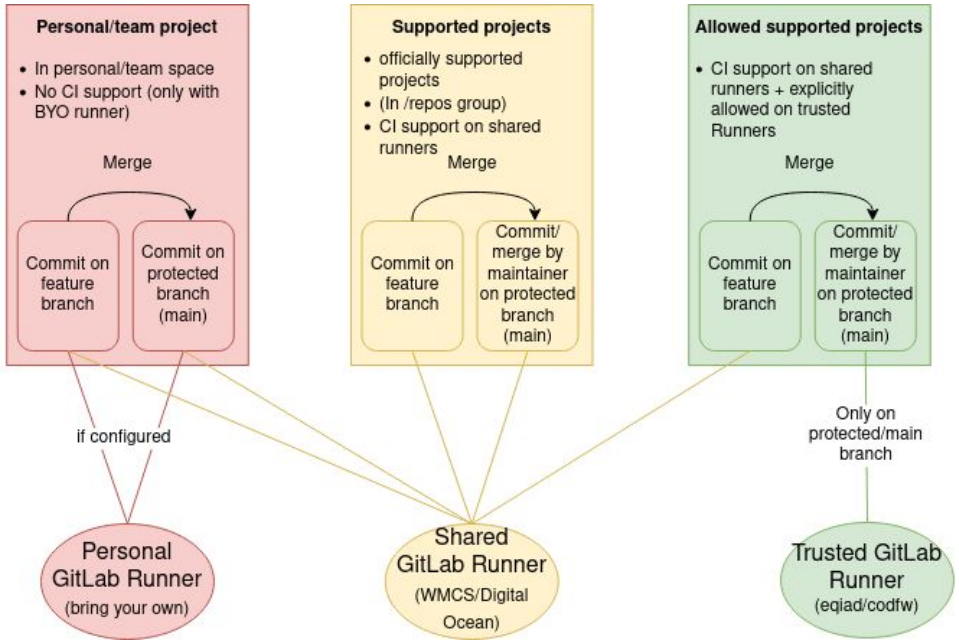
GitLab CI



https://wikitech.wikimedia.org/wiki/GitLab/Gitlab_Runner/Trusted_Runners#/media/File:GitLab_Runner_permission_model.png

GitLab CI

- Job scheduling depending on branch, project and tags
  - Feature branches/private repos run on Shared Runners
  - Merges on protected branches run on Trusted Runners (if needed)
  - CI-tags can be used to influence job scheduling
    - Git tag != gitlab tag



https://wikitech.wikimedia.org/wiki/GitLab/Gitlab_Runner/Trusted_Runners#/media/File:GitLab_Runner_permission_model.png

Existing and new CI tooling

- CI can be enhanced by including .gitlab-ci.yml other templates:
  - Central abstraction of CI templates: repos/releng/kokkuri
  - Currently supported: building and publishing container images from existing blubber files
- Additional security templates: repos/security/gitlab-ci-security-templates
- Pipeline converter: https://www.mediawiki.org/wiki/GitLab/pipeline_conversion
- Bot integration to Phabricator

Next steps

- Onboard more projects
- Create tooling and CI templates for other CI use cases
  - Debian package builds, language specific builds
- Open Shared Runners to all projects

WIKIMEDIA FOUNDATION

Feedback, Questions?

“Cat looking confused”

Thanks for your time!