Gerrit Access

There are two types of Gerrit access described here, with different powers & responsibilities. Only ask for access if you're an active member of the community. New contributors should participate in the Gerrit code review process for some time before requesting access.

(For Github access, see https://go.dev/wiki/GithubAccess)

Running TryBots ("may-start-trybots")

TryBot access lets you kick off a test run of a CL in Gerrit prior to submission (pre-submit testing). TryBots run in a somewhat-secure and somewhat-isolated environment, but they're not perfectly security hardened. You must skim the CL for anything malicious before starting TryBots.

Voting Run-TryBot+1 asks Gopherbot to run the CL on the TryBots. When the TryBots finish, Gopherbot will reply with results, voting either TryBot-Result+1 (pass) or TryBot-Result-1 (fail).

Every TryBot run includes a default set of the most common builders. SlowBots provide additional testing controls.

All approvers (see next section) have TryBot access automatically. Others can request TryBot access, reference https://go-review.googlesource.com/#/admin/groups/1030,members in your bug. See Requesting Access below.

Approving CLs ("approvers")

Approvers can review and submit code changes (CLs), subject to the review rules described below. Being an approver comes with an expectation of responsibility: approvers are people who care about Go and want to help it succeed. An approver is not just someone who can make changes, but someone who has demonstrated their ability to collaborate with the team, get the most knowledgeable people to review code, contribute high-quality code, and follow through to fix issues (in code or tests).

To request approver access, reference https://go-review.googlesource.com/#/admin/groups/1005,members in your bug. See Requesting Access below.

Code Review Requirements

Every CL requires *both* a code review (Code-Review+2) from an approver and the involvement of two Google employees using Google-secured computers, either as code uploader or as a reviewer voting at least Code-Review+1. Requiring multiple people ensures that code cannot be submitted unilaterally from a single compromised account. The Google employee and hardware requirements further raise the bar: since CLs in many repos are essentially published by Google for download by users at commit time, the Google involvement is to approve this publication. Once a review has a Code-Review+2 and the necessary Google involvement, it can be submitted, by any approver. All these rules are enforced by the Gerrit server.

A Code-Review+2 vote means that you have read the change and are confident that it is correct and appropriate to submit. Typically, you should only Code-Review+2 code in directories or packages that you "own"; the exception is trivial and obviously correct changes. Note that all user-visible new features or changes—new API, new command-line flags, and so on—need to go through the proposal process. The CLs should reference the specific accepted proposal in the commit message ("For #NNN.").

When adding a Code-Review+2 vote, it is encouraged to also add Run-TryBots+1 and Auto-Submit+1: see the auto-submit section below for details.

A Code-Review+1 vote means that you have read the change and believe it seems reasonable but aren't making the definitive judgement that Code-Review+2 indicates. It also means you are confident the change does not introduce any sort of security vulnerability or other clearly inappropriate code change.

When a change has the appropriate reviews to be submitted, a Submit button appears in Gerrit (for approvers). You should only submit changes with a Code-Review+2 from the owner of that area (maybe you!).

To request approver access, reference https://go-review.googlesource.com/#/admin/groups/1005,members in your bug. See below.

Auto-Submit

If you are reviewing a CL and believe it can be approved and submitted as is, with no further changes, you can use the auto-submit functionality to run tests and submit the CL if the tests pass. To do this, vote Code-Review+2 as well as Auto-Submit+1 and Run-TryBot+1. When the tests pass, Gopherbot will submit it.

More precisely, Gopherbot watches for and automatically submits CLs that

- have Auto-Submit+1 and TryBot-Result+1 votes,
- have the necessary code reviews,
- have no unresolved comments,
- aren't marked #wait-release,
- and merge cleanly into the current branch head.

All approvers can add Auto-Submit+1 votes. An Auto-Submit+1 vote is not carried forward when a patch is reuploaded.

Requesting Access

To get request either of the access types above, file a bug (https://github.com/golang/go/issues/new?title=access:+&body=See+https://go.dev/wiki/GerritAccess) and list and state which access you want (its name and group URL), and state your Gerrit email address.

Decisions about granting access are made by the Go release team at Google. If your request is declined, it is almost always because you haven't been active enough for them to get a clear enough signal about your work, understanding of project conventions, and so on. Don't lose heart: it can take time to reach that level of familiarity.

Once you have access

Go help garden! See https://go.dev/wiki/Gardening.