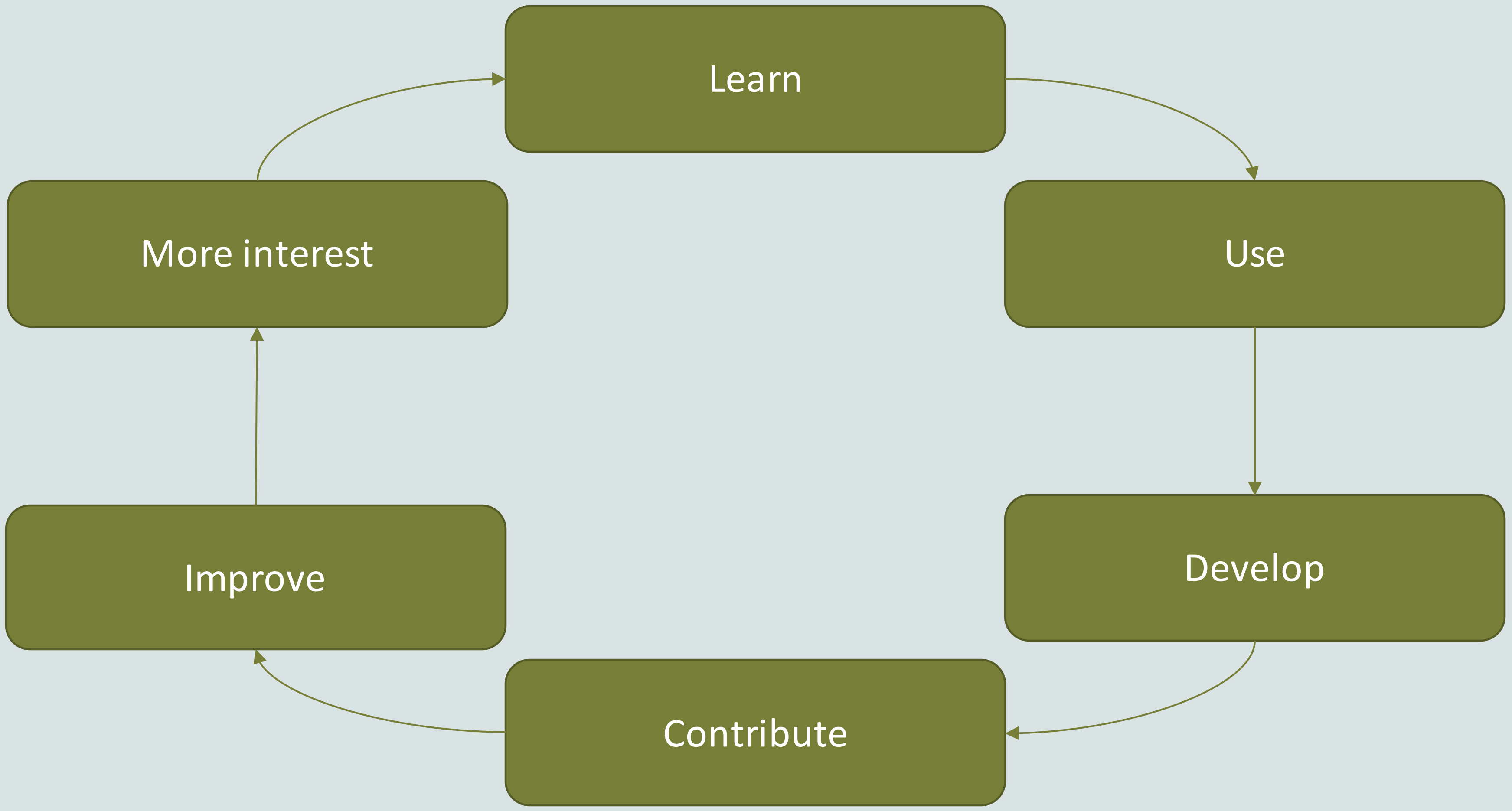
Contributing to gem5

A presentation by Bobby R. Bruce

With materials borrowed from Andreas Sandburg!

Our Strategy



Why should I contribute to gem5?

You're Nice!

- You've found a bug and have a fix.
- You've developed something useful and want t share it.

Fame!

- Get yourself known in the project.
- Good PR for your research to have it incorporated in gem5.

Fortune!

- Looks good on your CV.
- Companies contribute to gem5 all the time.

“I’m scared”

Understandable…

Very few patches get in straight away. Most patches are only accepted after requests for changes.

We try our best to keep feedback as constructive as possible (don’t take it personally!).

The purpose of this session is to make it less scary!

What can I contribute?

Your own changes (bug fixes are very welcome!)

Check the Issue Tracker:
https://gem5.atlassian.net

Some stuff we're always needing more of:

- Tests
- Incorporating Syscalls for SE mode
- Unimplemented ISA instructions/extensions
- Useful stdlib components
- Useful gem5 resources
- Updating documentation on the gem5 website
  - Even fixing typos is helpful!

Some useful resources

https://www.gem5.org/contributing

CONTRIBUTING.md in the gem5 directory

Sometimes using git is the biggest hurdle:

- https://git-scm.com/book/en/v2 : The git book
- https://dev.to/milu_franz/git-explained-the-basics-igc : I think this is a good tutorial but is very GitHub-centric (we don't use GitHub for gem5). Still, going through it would be beneficial.
- https://wiki.spheredev.org/index.php/Git_for_the_lazy : Does a quick run through of the basic Git commands. Can be good for reference.
- http://marklodato.github.io/visual-git-guide/index-en.html: A bit more complex but tries to introduce the git data structures involved in git
- https://towardsdatascience.com/git-help-all-2d0bb0c31483: Another resource outlining both the commands and explaining how git works.

Where do I make changes?

```
➢ git clone https://gem5.googlesource.com/public/gem5-website
```

```
> cd gem5-website
```

Where do I make changes?

```
> git switch –c my-change
```

You are working on top of the gem5-website stable branch.

This is permitted in the gem5-website repository. If your patch is accepted the website will be updated with that change ASAP.

You may work atop the “develop” branch if your change to the website should only be published upon the next major gem5 release (good for next release documentation updates).

What about the other gem5 repos?

gem5 Resources

https://gem5.googlesource.com/public/gem5-resources

Build atop "stable" to make changes for the current release.

Built atop "develop" to make changes for the upcoming release.

gem5

https://gem5.googlesource.com/public/gem5

Built atop "develop" to make changes. You cannot push to stable.

Making changes: CPP

Full style guide here: https://www.gem5.org/documentation/general_docs/development/coding_style/

High-level overview: https://www.gem5.org/contributing#making-modifications

Doxygen is highly recommended

http://doxygen.gem5.org

Making changes: Python

```
> pip install black
> black <python file>
```

For variable/method/etc. Naming conventions please follow the PEP 8 naming convention recommendations: https://peps.python.org/pep-0008/#naming-conventions

While we try to enforce naming conventions across the gem5 project, we are aware there are instances where they are not.

In such cases please **follow the convention of the code you are modifying**.

The biggest gotchas!

- Whitespace at the end of a line.
- Indentation not 4 space characters (please, no tabs)
- Lines too long (for CPP, no more than 79 characters!)

When in doubt, follow the style around you!

We have a style checker which should stop you committing if you've done something wrong, but it's not perfect and can be side-stepped.

Using git

```
> git add <files to add>
```

```
> git commit
```

Commit message rules

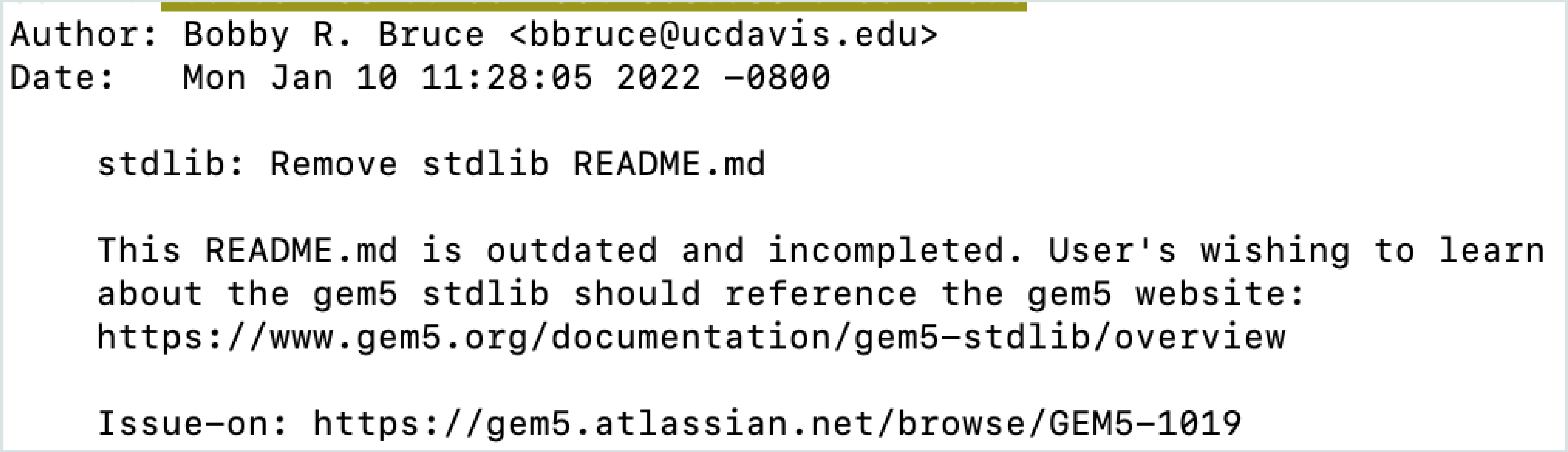
We have some unique rules for gem5:

1. The header must lead with tags (see MAINTAINERS.yaml for a list of tags).
2. Headers should be clear, short descriptions of what a patch will do.
3. Headers should be **no longer than 65 characters**
4. A blank line separates the header and the patch description.
5. Descriptions can span multiple paragraphs but lines **should not exceed 72 characters** (this is lax rule, it’s acceptable to exceed this if you’re quoting code, or including a URL).
6. If you're implementing a Jira request, cite the Jira URL.

View the Git Log

```
> git log
```

Example commit message



```
Author: Bobby R. Bruce <bbruce@ucdavis.edu>
Date:   Mon Jan 10 11:28:05 2022 -0800

    stdlib: Remove stdlib README.md

    This README.md is outdated and incompleted. User's wishing to learn
    about the gem5 stdlib should reference the gem5 website:
    https://www.gem5.org/documentation/gem5-stdlib/overview

    Issue-on: https://gem5.atlassian.net/browse/GEM5-1019
```

How do I push?

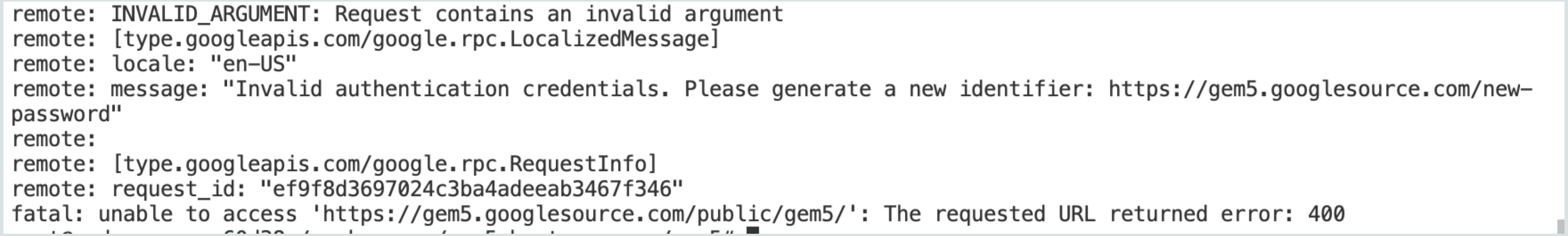
Pushing to Gerrit a little weird…

```
> git push origin HEAD:refs/for/stable%wip
```

“stable”, the branch you want to contribute to.

”%wip” means the patch is “Work In Progress”

You should all be getting this error



```
remote: INVALID_ARGUMENT: Request contains an invalid argument
remote: [type.googleapis.com/google.rpc.LocalizedMessage]
remote: locale: "en-US"
remote: message: "Invalid authentication credentials. Please generate a new identifier: https://gem5.googlesource.com/new-
password"
remote:
remote: [type.googleapis.com/google.rpc.RequestInfo]
remote: request_id: "ef9f8d3697024c3ba4adeeab3467f346"
fatal: unable to access 'https://gem5.googlesource.com/public/gem5/': The requested URL returned error: 400
```

Create your Gerrit account and authenticate

Probably not! You need to register with Gerrit:

1. Create an account at https://gem5-review.googlesource.com.

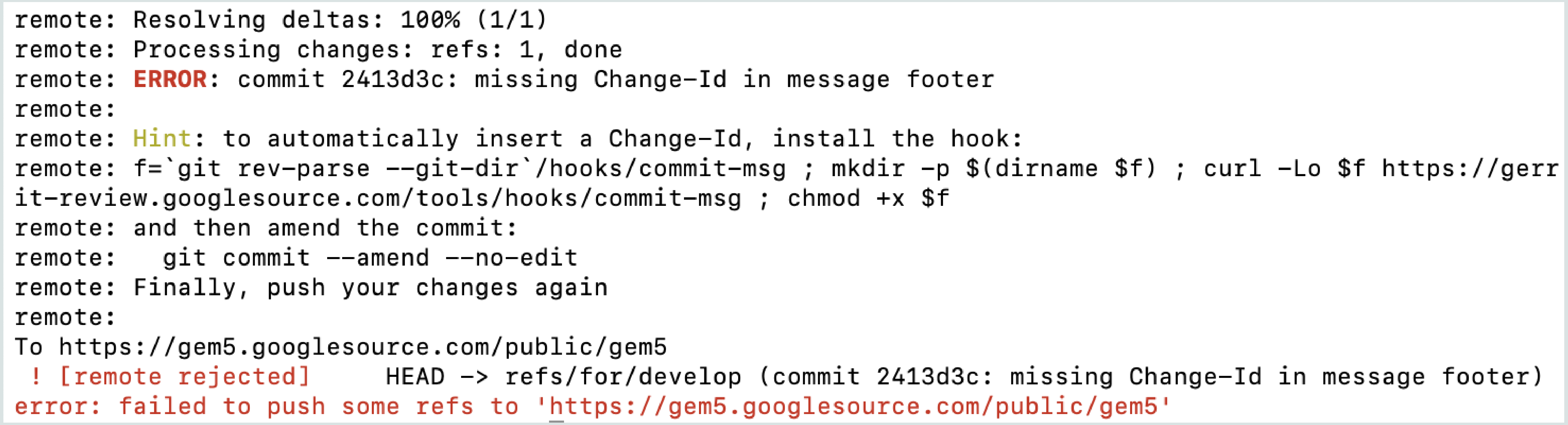
2. Go to User Settings.

3. Select Obtain password (under HTTP Credentials).
A new tab will open explaining how to authenticate your machine to make contributions to Gerrit. Follow these instructions and try pushing again.

Let’s try again

```
> git push origin HEAD:refs/for/stable%wip
```

You're probably getting this error



```
remote: Resolving deltas: 100% (1/1)
remote: Processing changes: refs: 1, done
remote: ERROR: commit 2413d3c: missing Change-Id in message footer
remote:
remote: Hint: to automatically insert a Change-Id, install the hook:
remote: f=`git rev-parse --git-dir`/hooks/commit-msg ; mkdir -p $(dirname $f) ; curl -Lo $f https://gerr
it-review.googlesource.com/tools/hooks/commit-msg ; chmod +x $f
remote: and then amend the commit:
remote:   git commit --amend --no-edit
remote: Finally, push your changes again
remote:
To https://gem5.googlesource.com/public/gem5
 ! [remote rejected]     HEAD -> refs/for/develop (commit 2413d3c: missing Change-Id in message footer)
error: failed to push some refs to 'https://gem5.googlesource.com/public/gem5'
```

Fix with:

```
> f=`git rev-parse --git-dir`/hooks/commit-msg ; mkdir -p $(dirname $f) ; curl -Lo $f
https://gerrit-review.googlesource.com/tools/hooks/commit-msg ; chmod +x $f
```

```
> git commit --amend --no-edit
```

```
> git log
```

Let’s try again

```
> git push origin HEAD:refs/for/stable%wip
```

Follow the link to your patch

Two Types of “Review”

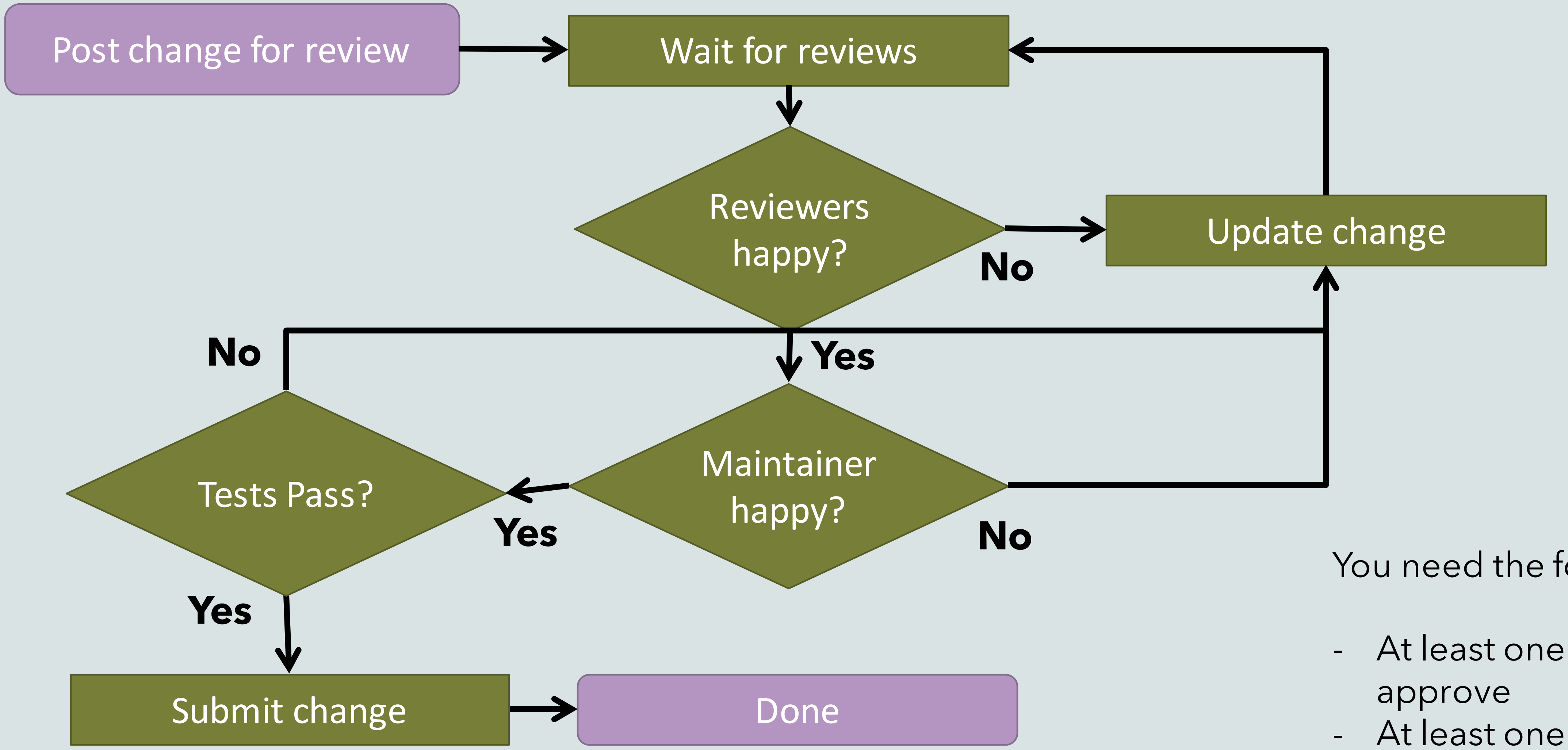
1. Ordinary reviewer: Literally anyone with a gem5 Gerrit account.
2. Maintainer: An exclusive club, see MAINTAINERS.yaml for the list.

You need sign off by a reviewer and a maintainer to get a patch into gem5.

Sometimes a maintainer will give votes for both (as a reviewer and as a maintainer) but we only recommend this for stuff the maintainer has high confidence in. Typicall we like two separate people to sign off on a patch

gem5

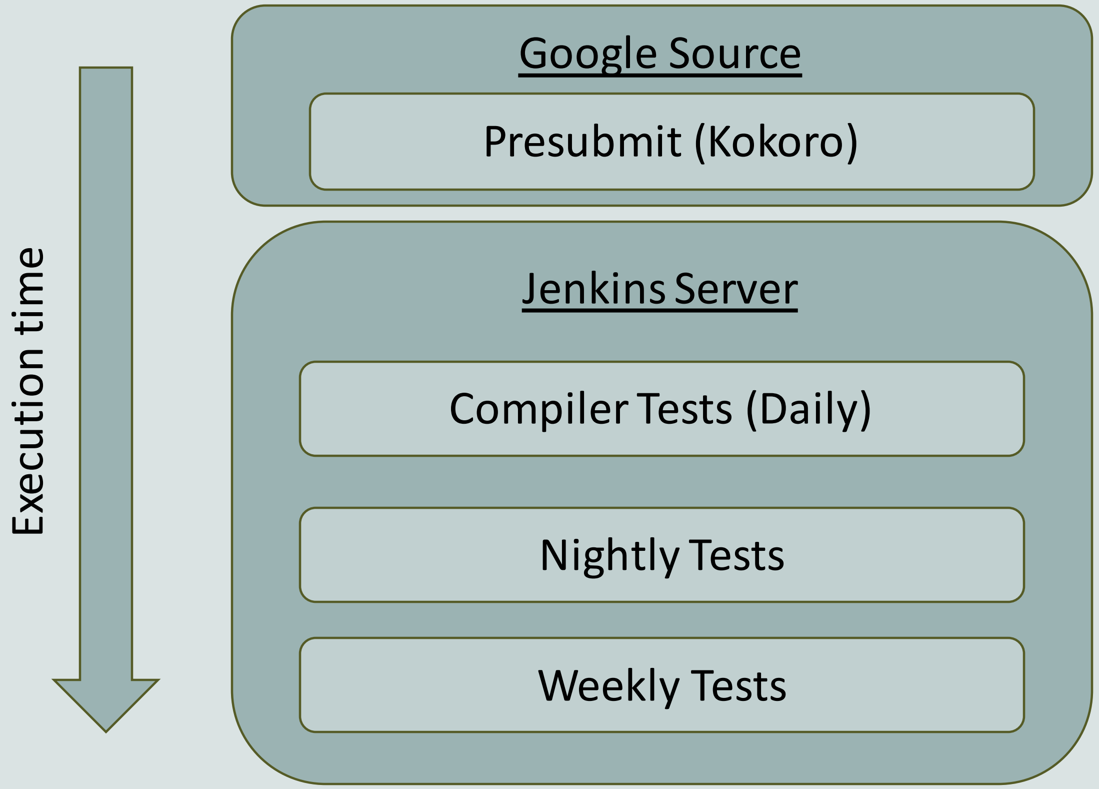
Gerrit Code Review Process



You need the following:

- At least one reviewer to approve
- At least one maintainer
- Our tests to pass

Testing overview



The most direct are the "Presubmit" tests, you cannot submit a to develop without these passing

The rest are run at various times on our Jenkins server: https://jenkins.gem5.org

gem5

Let’s do something!

```
> cd gem5-resources
```

```
> git pull
```

```
> git switch stable
```

Two routes here:

1. Create a toy change
2. Fix broken URLs: Some URLs and Docker files are using the wrong gem5 version! (v21-2 instead of v22-0).

```
> git push origin HEAD:refs/for/stable%wip
```