

Gerrit Code Review

First steps with Git and Code Review Workflow



Luca Milanesio
luca@gerritforge.com

10 git commands commonly used in Gerrit Workflow

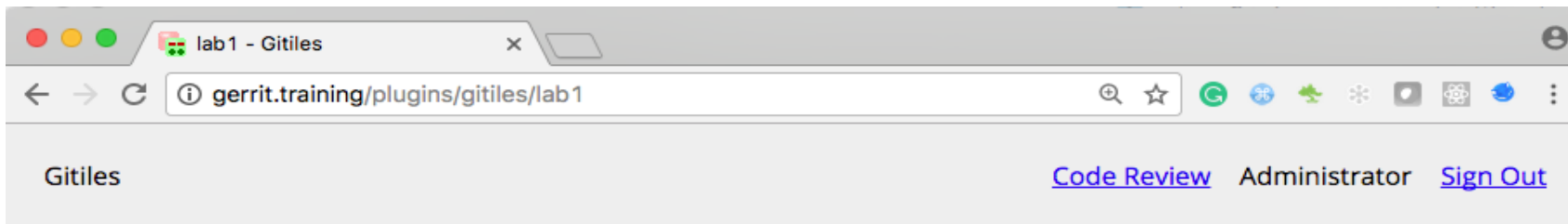


1. git clone
2. git fetch
3. git checkout
4. git add
5. git commit --amend
6. git reset
7. git rebase
8. git merge
9. git cherry-pick
10. git remote



Practicing advanced Git commands

Lab#1: Gerrit Repositories Browser: Gitiles



[gerrit.training](#) / **lab1**

Repository for practicing common Git commands

Clone this repo:

```
git clone http://gerrit.training/lab1
```

Branches

`83db6be` [Initial empty repository](#) by Administrator · 33 hours ago [master](#)

[master](#)

Lab#1: Clone the repository from Gerrit



```
$ git clone http://gerrit.training/lab1
Cloning into 'lab1'...
remote: Counting objects: 2, done
remote: Finding sources: 100% (2/2)
remote: Total 2 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (2/2), done.
Checking connectivity... done.

$ cd lab1
(master) $
```

Lab#1: Create a feature branch and add one commit



```
(master) $ git checkout -b feature/my-true-story
Switched to a new branch 'feature/my-true-story'

(feature/my-true-story) $ echo "This is me" > my-true-story.md

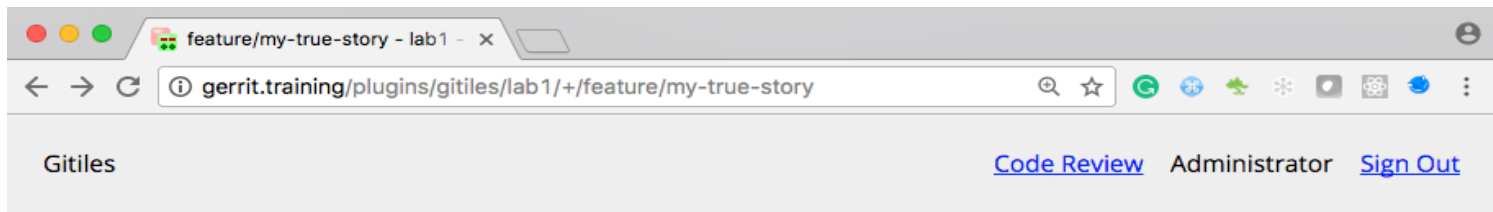
(feature/my-true-story) $ git add . && git commit -m "My true story"
[feature/my-true-story 0368861] My true story
1 file changed, 1 insertion(+)
create mode 100644 my-true-story.md
```

Lab#1: Push to Gerrit



```
(feature/my-true-story) $ git push --set-upstream origin feature/my-true-story
Username for 'http://gerrit.training': admin
Password for 'http://admin@gerrit.training': secret
Counting objects: 3, done.
Writing objects: 100% (3/3), 266 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote: Processing changes: refs: 1, done
To http://gerrit.training/lab1
* [new branch]    feature/my-true-story -> feature/my-true-story
Branch feature/my-true-story set up to track remote branch feature/my-true-story from origin.
```

Lab#1: See feature branches in Gitiles



[gerrit.training](#) / [lab1](#) / **feature/my-true-story**

```
commit 0368861622c4acec7487a224bd7134e6d1047ee3      [log] [tgz]
author Luca Milanesio <luca.milanesio@gmail.com>    Thu May 18 08:52:49 2017 +0100
committer Luca Milanesio <luca.milanesio@gmail.com> Thu May 18 08:52:49 2017 +0100
tree 8b247218eb07c518303bfc66909b049c69c3aced
parent 83db6be61abd7e4f05e9dc3e1b76edf8611fff0f [diff]
```

```
My true story
```

[my-true-story.md](#) [Added - diff]

1 file changed

```
tree: 8b247218eb07c518303bfc66909b049c69c3aced
```

 [my-true-story.md](#)

@gitterenterprise @gerritreview

Lab#1: Another developer pushes a feature branch



```
(master) $ git checkout -b feature/another-story
Switched to a new branch 'feature/another-story'

(feature/another-story) $ echo "This is you" > another-story.md

(feature/another-story) $ git add . && git commit -m "Another story"
[feature/my-true-story 036bc6d] Another story
1 file changed, 1 insertion(+)
create mode 100644 another-story.md

(feature/another-story) $ git push origin feature/another-story
```

Lab#1: Working in "detached HEAD" mode

```
$ git fetch origin feature/another-story
```

```
From http://gerrit.training/lab1
```

```
* branch          feature/another-story -> FETCH_HEAD
```

```
$ git checkout FETCH_HEAD
```

```
Note: checking out 'FETCH_HEAD'.
```

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may do so (now or later) by using `-b` with the checkout command again. Example:

```
git checkout -b <new-branch-name>
```

HEAD is now at 09385fc... Another story

Lab#1 – Amending a commit in "detached" HEAD



```
((09385fc...))$ echo "yet another word" >> another-story.md
```

```
((09385fc...))$ git commit -a --amend -m "Amended commit"
```

```
[detached HEAD bc4970b] Amended commit
```

```
Date: Thu May 18 23:35:12 2017 +0100
```

```
1 file changed, 1 insertion(+)
```

```
create mode 100644 another-story.md
```

```
((bc4970b...))$ git push -f origin HEAD:feature/another-story
```

```
Counting objects: 3, done.
```

```
Writing objects: 100% (3/3), 269 bytes | 0 bytes/s, done.
```

```
Total 3 (delta 0), reused 0 (delta 0)
```

```
remote: Processing changes: refs: 1, done
```

```
To http://gerrit.training/lab1
```

```
+ 09385fc...bc4970b HEAD -> feature/another-story (forced update)
```

@gitenterprise @gerritreview

Lab#1 – Fetching last HEAD of a branch



```
((09385fc...))$ git checkout feature/another-story
Previous HEAD position was c14001a... Amended commit
Switched to branch 'feature/another-story'
Your branch and 'origin/feature/another-story' have diverged,
and have 1 and 1 different commits each, respectively.
(use "git pull" to merge the remote branch into yours)

(feature/another-story)$ git fetch origin
From http://gerrit.training/lab1
* branch          feature/another-story -> feature/another-story (forced update)

(feature/another-story)$ git reset --hard origin/feature/another-story
HEAD is now at c14001a Amended commit
```

Lab#1 – Rebase a feature branch against master (1)



```
$ git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.

(master)$ echo foo > bar
(master)$ git add bar && git commit -m foo
[master 52c1df5] foo
1 file changed, 1 insertion(+)
create mode 100644 bar

(master)$ git push
Counting objects: 3, done.
Writing objects: 100% (3/3), 237 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote: Processing changes: refs: 1, done
```

To <http://gerrit.training/lab1>

83db6be..52c1df5 master -> master

@gitenterprise @gerritreview

Lab#1 – Rebase a feature branch against master (2)



```
$ git checkout feature/another-story
```

```
Switched to branch 'feature/another-story'
```

```
Your branch is up-to-date with 'origin/feature/another-story'.
```

```
(feature/another-story)$ git fetch && git rebase origin/master
```

```
First, rewinding head to replay your work on top of it...
```

```
Applying: Amended commit
```

```
(feature/another-story)$ git push -f origin feature/another-story
```

```
Counting objects: 3, done.
```

```
Delta compression using up to 4 threads.
```

```
Compressing objects: 100% (2/2), done.
```

```
Writing objects: 100% (3/3), 316 bytes | 0 bytes/s, done.
```

```
Total 3 (delta 0), reused 0 (delta 0)
```

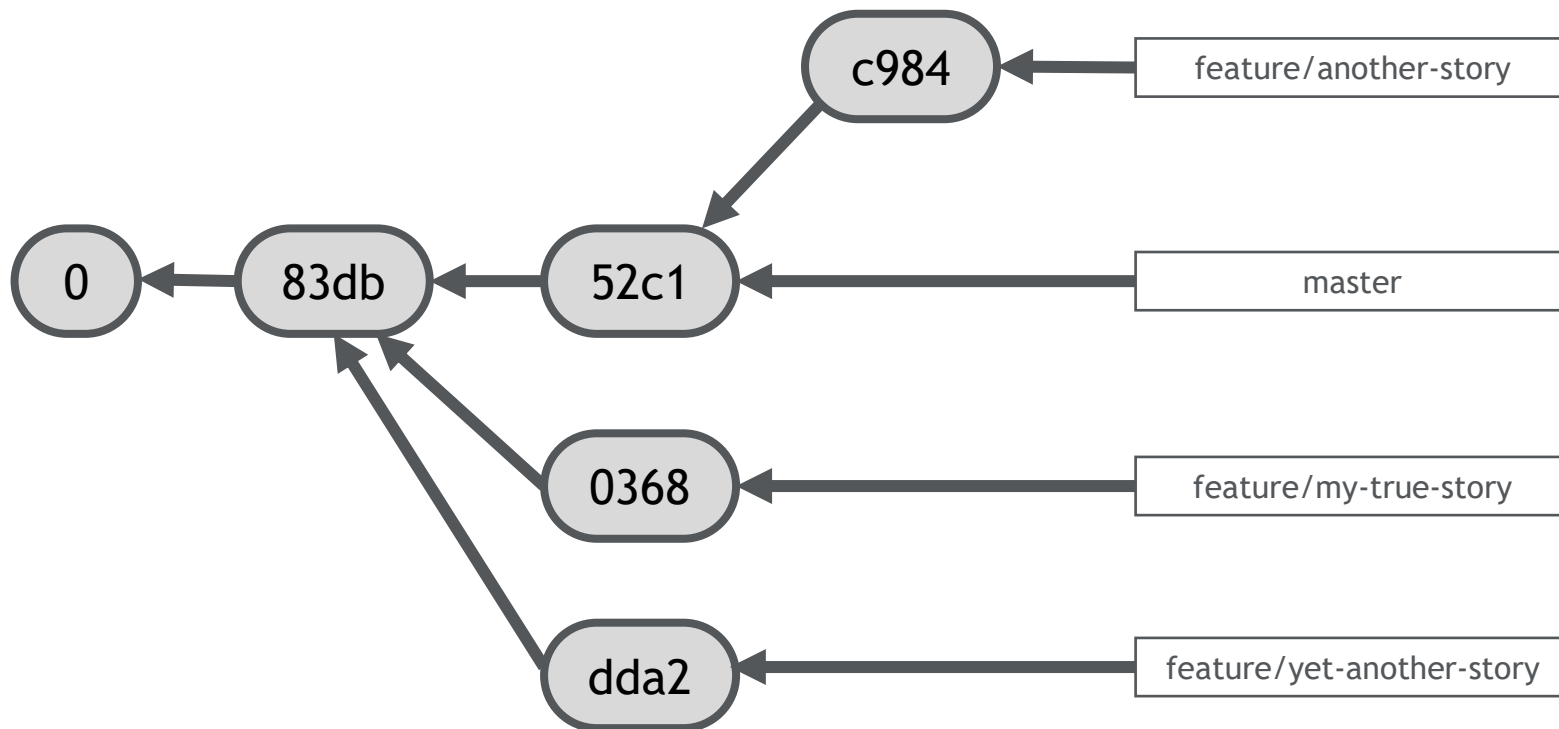
```
remote: Processing changes: refs: 1, done
```

```
To http://gerrit.training/lab1
```

```
+ c14001a...c9841bf feature/another-story -> feature/another-story (forced update)
```

@gitenterprise @gerritreview

Lab#1 – Cherry-pick commits between branches (1)



Lab#1 – Cherry-pick commits between branches (2)



```
(feature/another-story)$ git cherry-pick 0368
[feature/another-story 31911f1] My true story
Date: Thu May 18 08:52:49 2017 +0100
1 file changed, 1 insertion(+)
create mode 100644 my-true-story.md
```

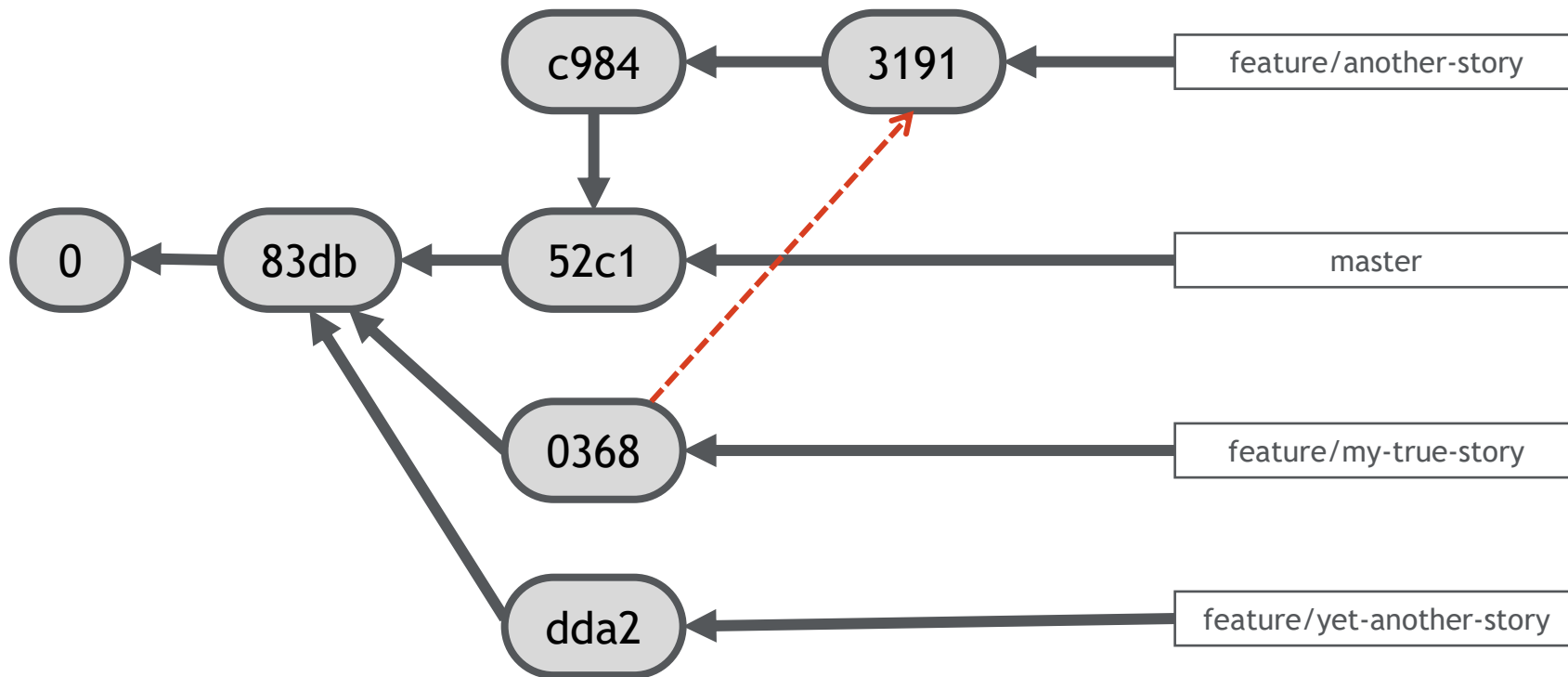
```
(feature/another-story)$ git push
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 332 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote: Processing changes: refs: 1, done
```

```
To http://gerrit.training/lab1
```

```
c9841bf..31911f1 feature/another-story -> feature/another-story
```

@gitenterprise @gerritreview

Lab#1 – Cherry-pick commits between branches (3)



Lab#1 – Working with remotes



```
$ git remote -v
origin      http://gerrit.training/lab1 (fetch)
origin      http://gerrit.training/lab1 (push)

$ git remote set-url --push origin ssh://admin@gerrit.training:29418/lab1
$ git remote -v
origin      http://gerrit.training/lab1 (fetch)
origin      ssh://admin@gerrit.training:29418/lab1 (push)

$ git remote add sshorigin ssh://admin@gerrit.training:29418/lab1
$ git remote -v
origin      http://gerrit.training/lab1 (fetch)
origin      ssh://admin@gerrit.training:29418/lab1 (push)
sshorigin   ssh://admin@gerrit.training:29418/lab1 (fetch)
sshorigin   ssh://admin@gerrit.training:29418/lab1 (push)
```

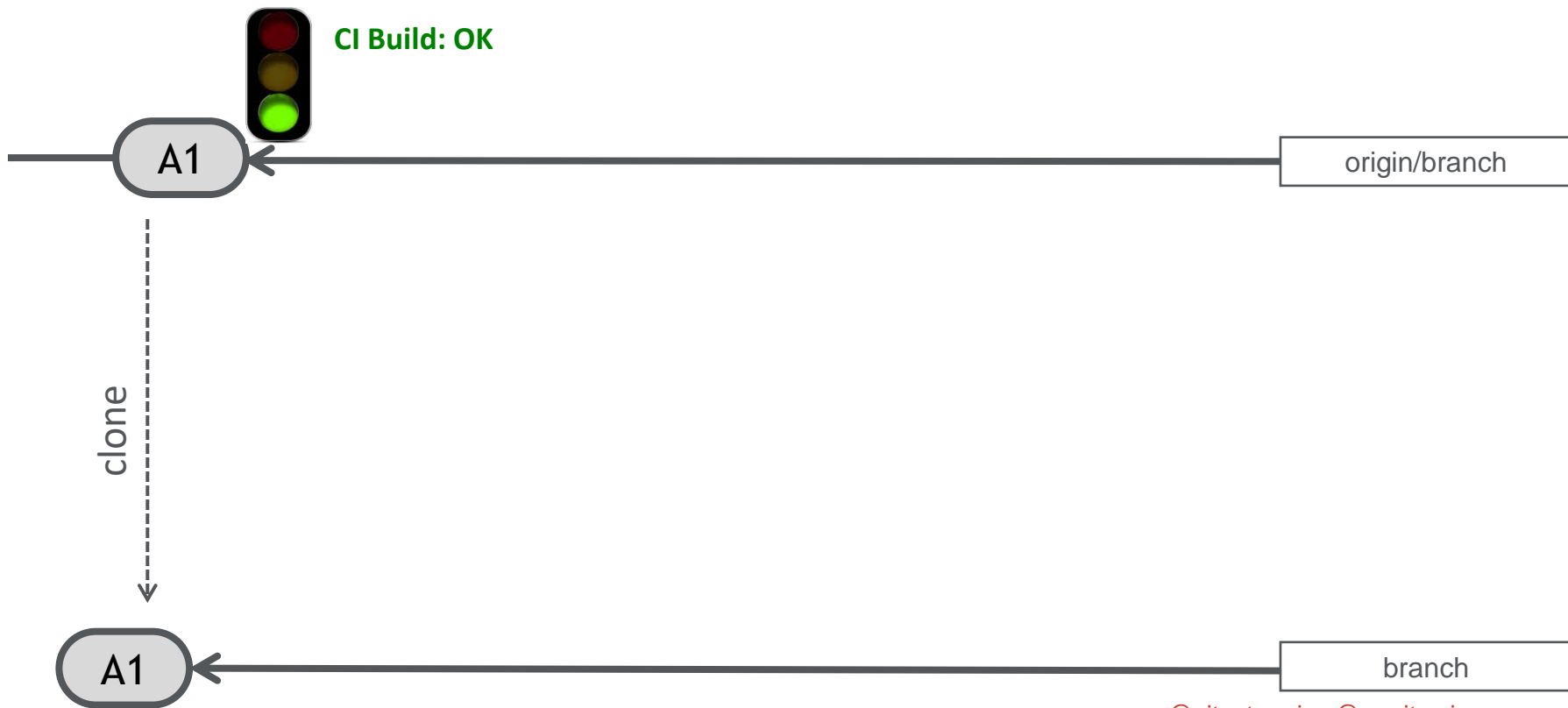


Gerrit Code Review Workflow

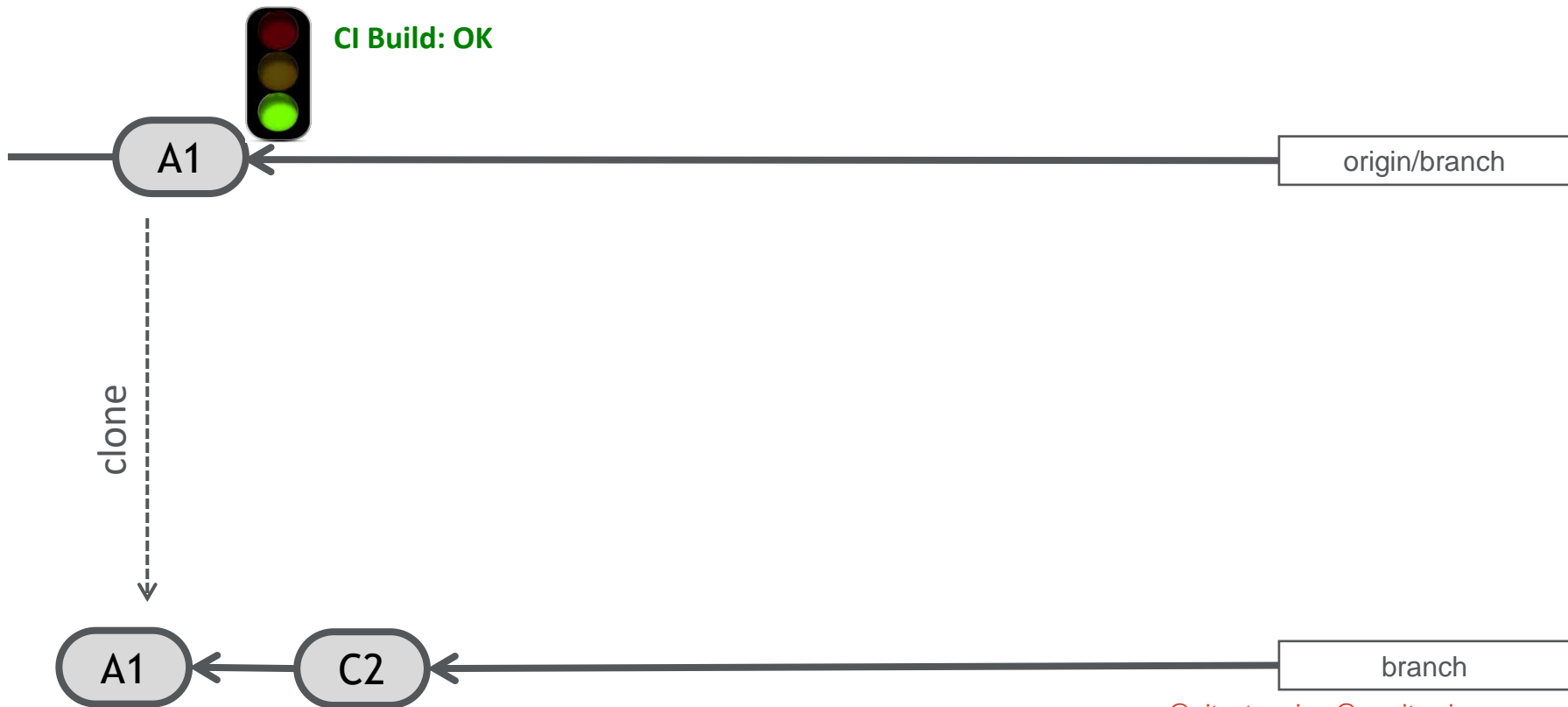
Code Review Workflow (0)



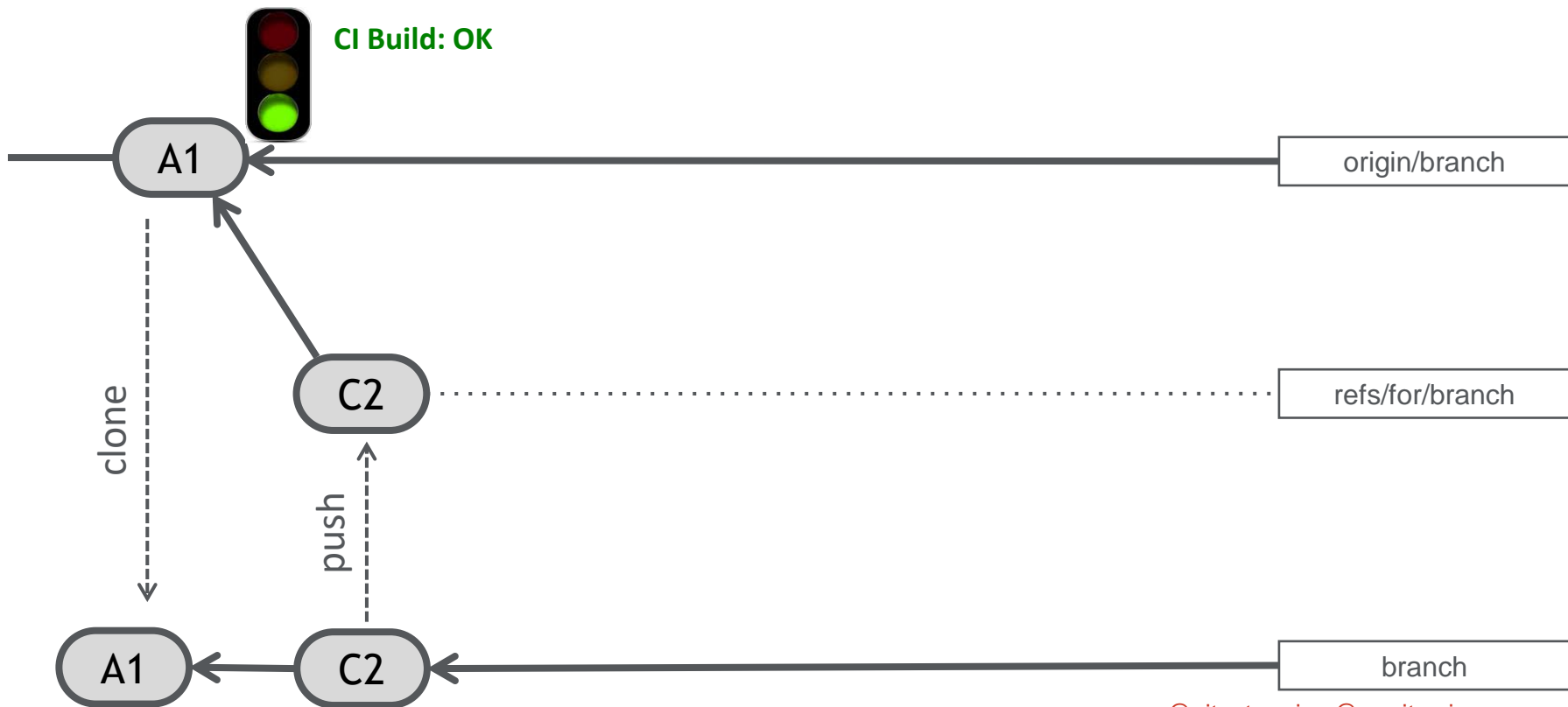
Code Review Workflow (1)



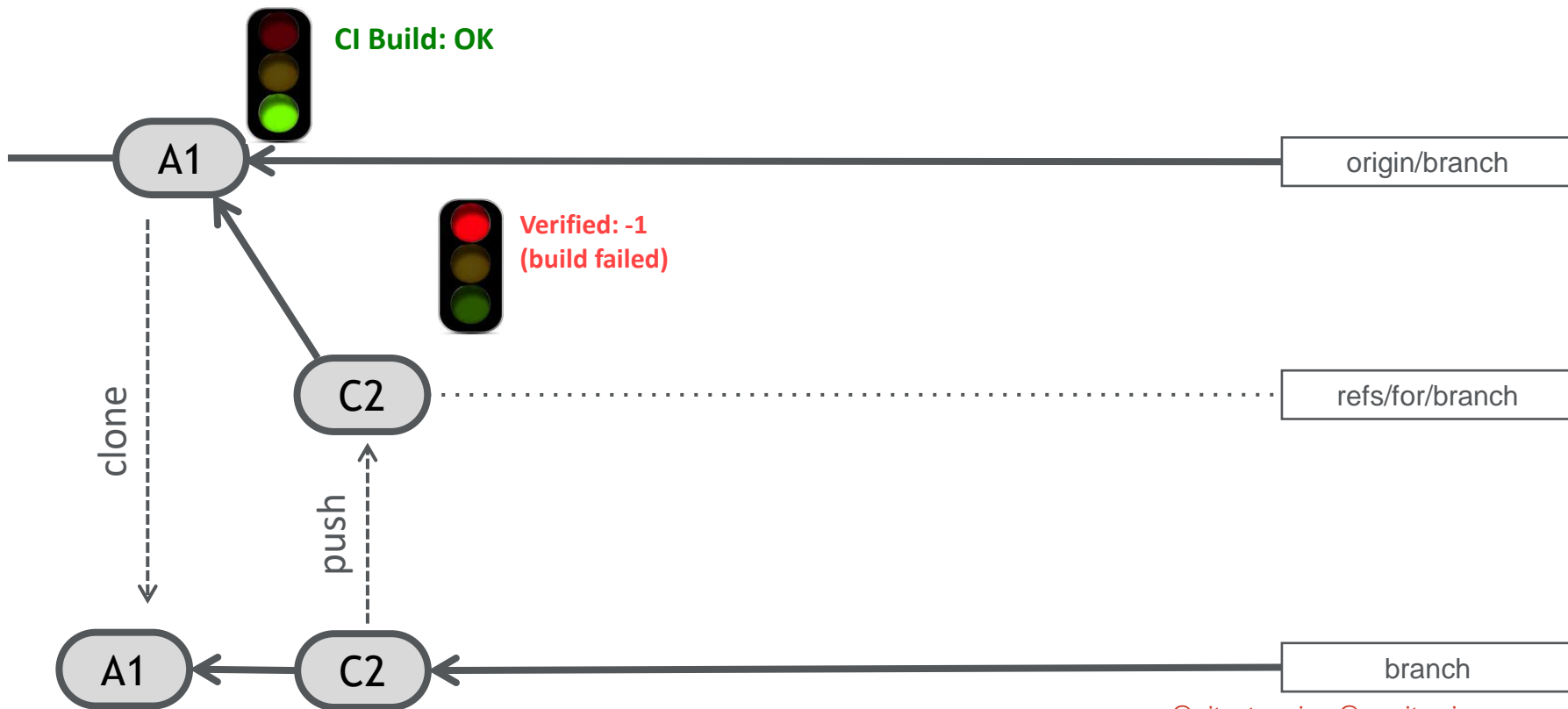
Code Review Workflow (2)



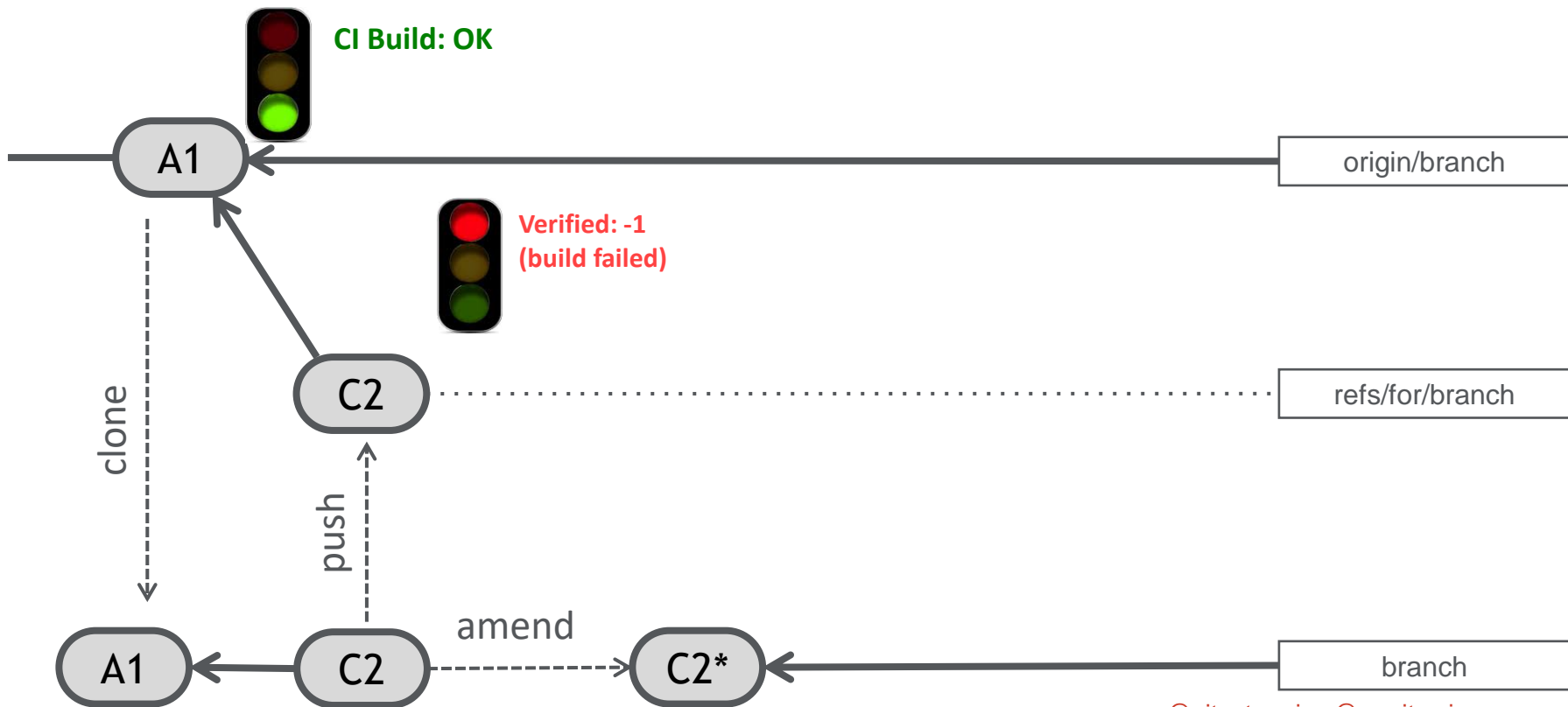
Code Review Workflow (3)



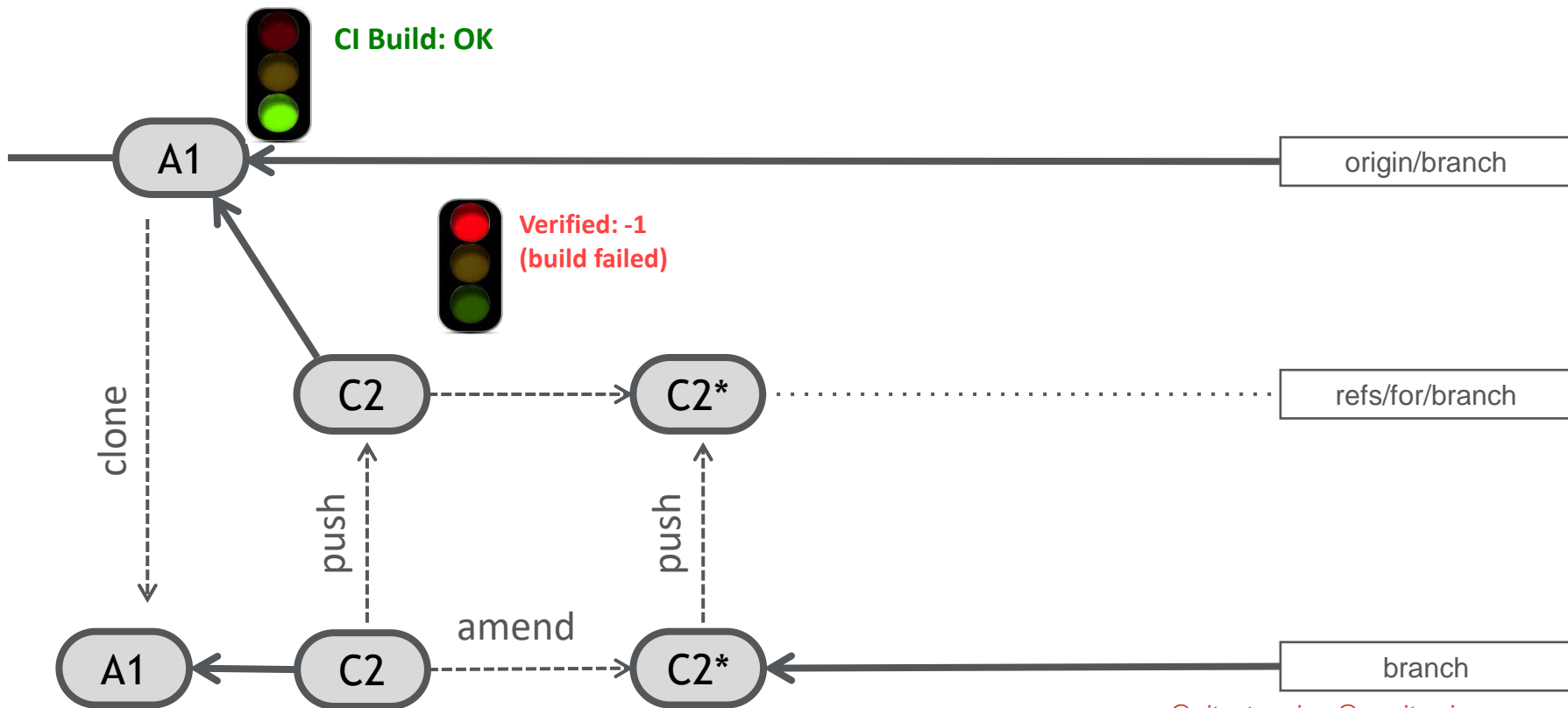
Code Review Workflow (4)



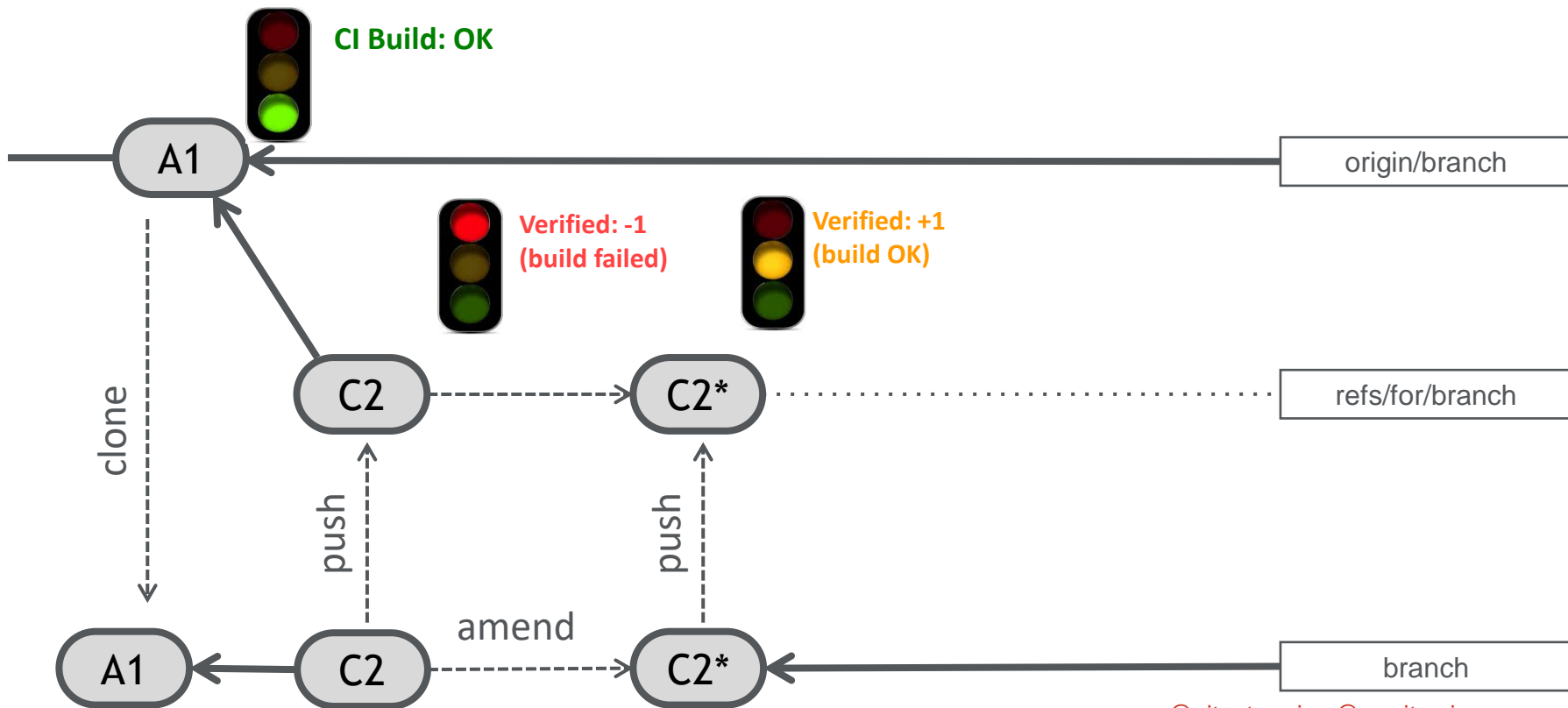
Code Review Workflow (5)



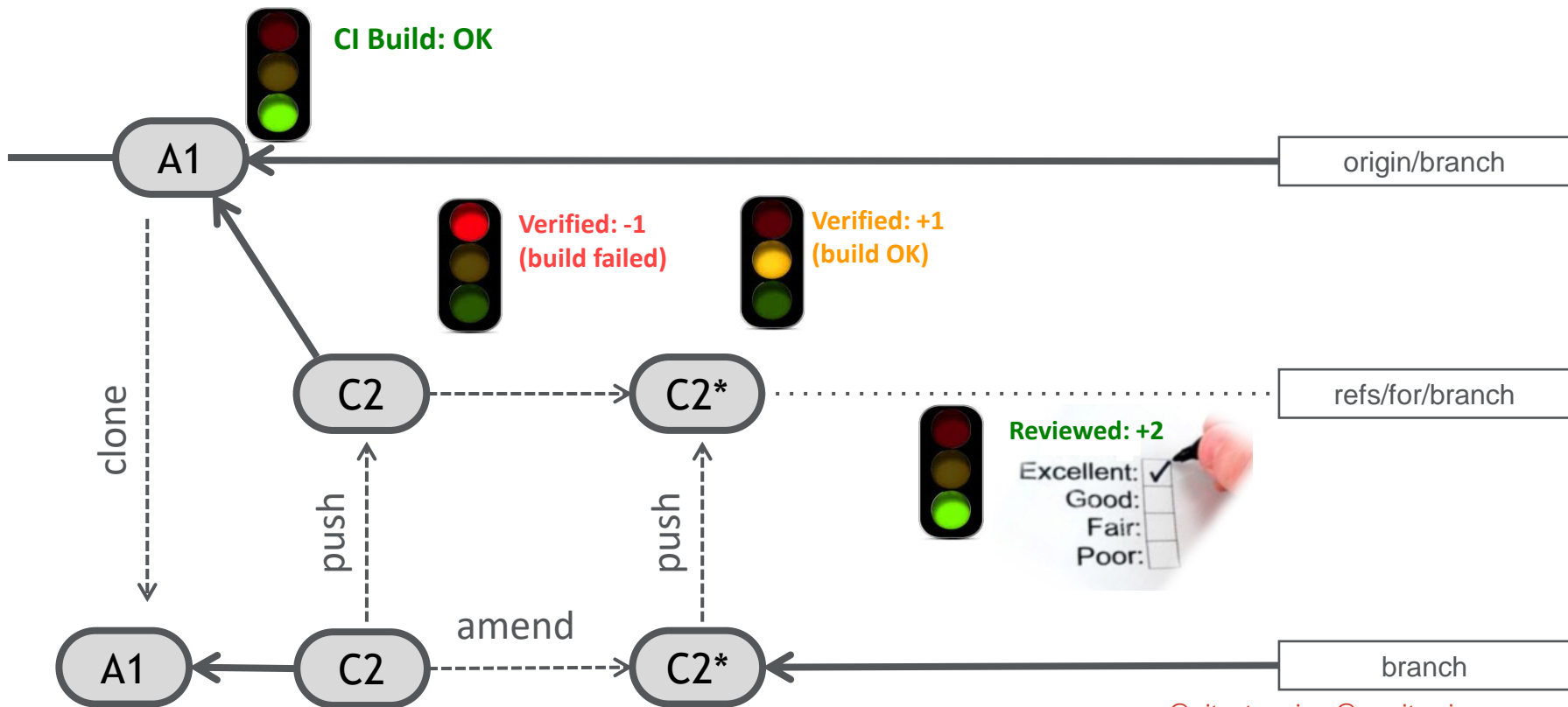
Code Review Workflow (6)



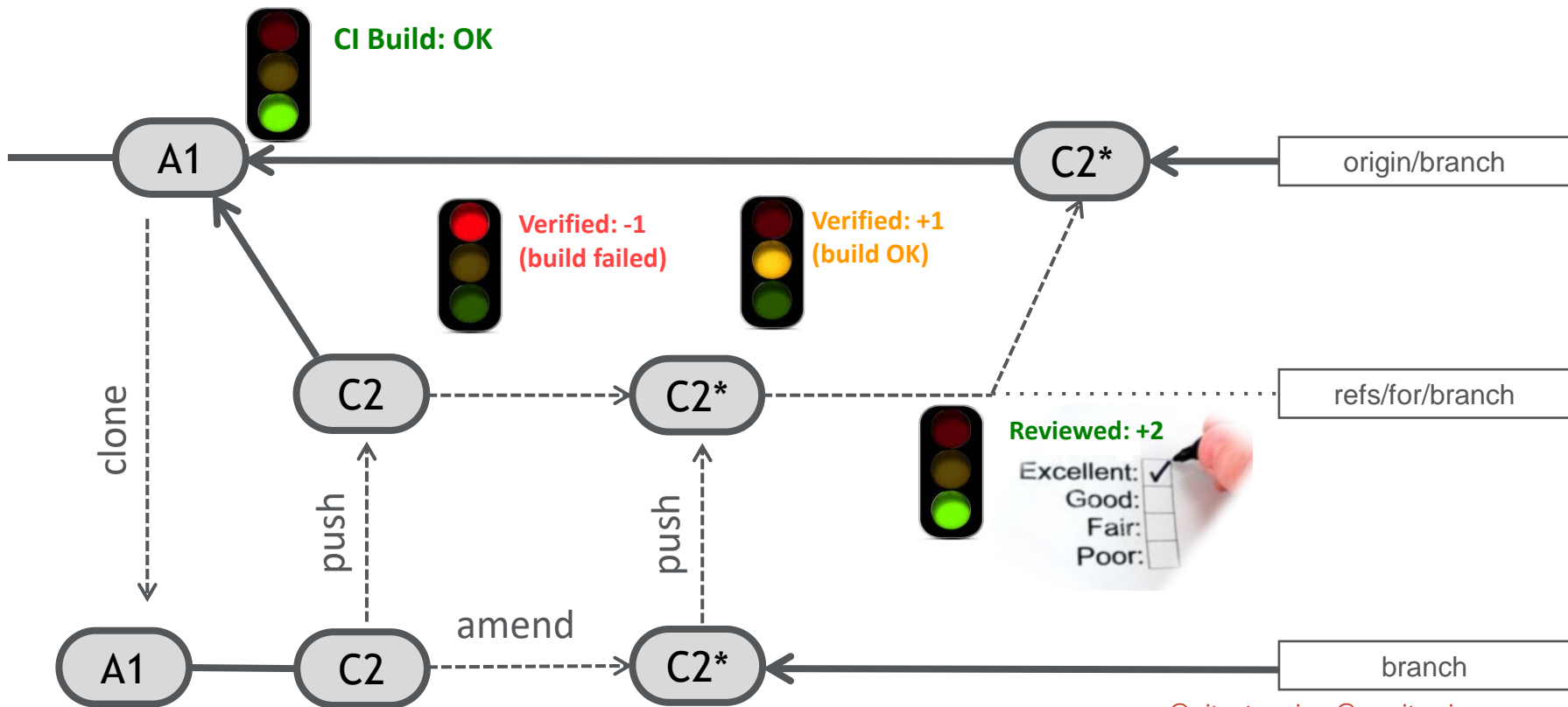
Code Review Workflow (7)



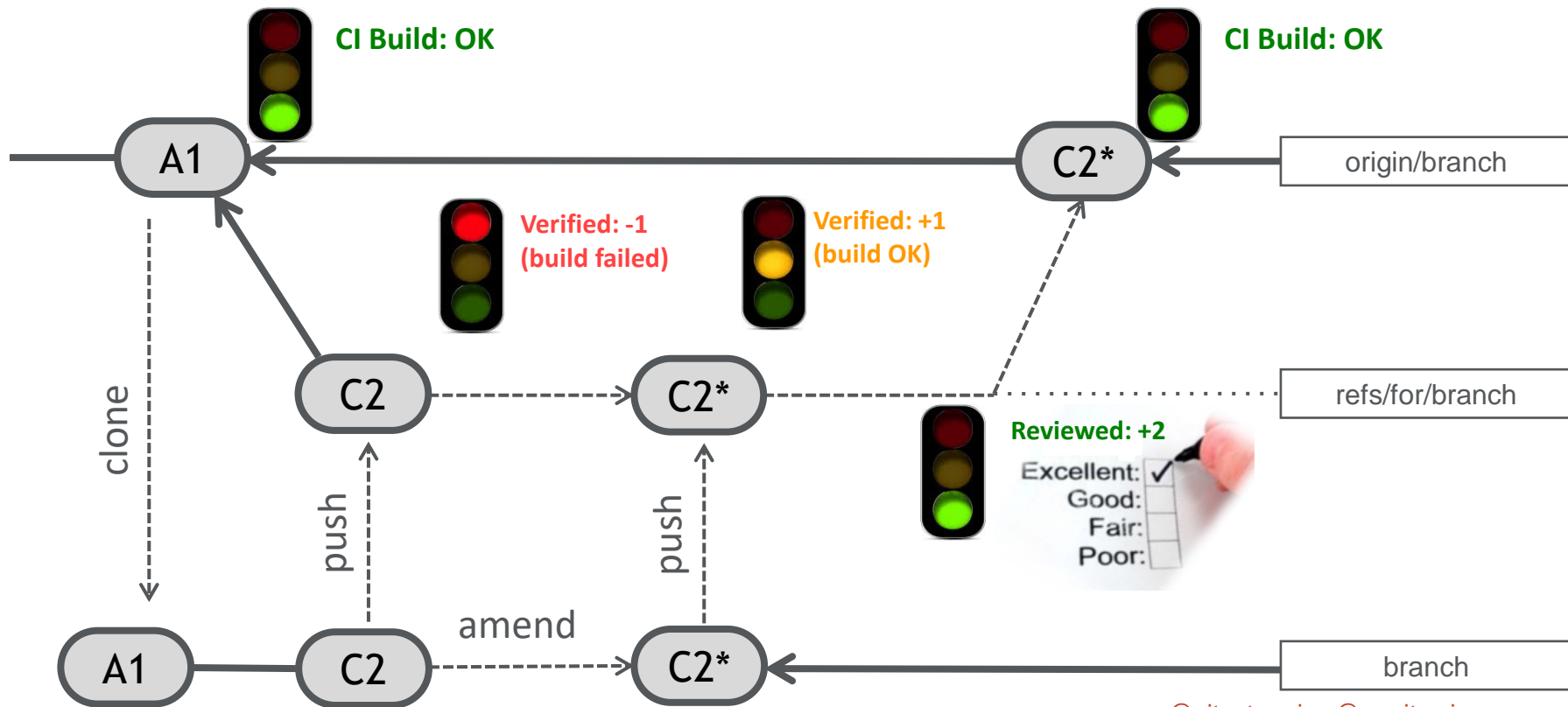
Code Review Workflow (8)



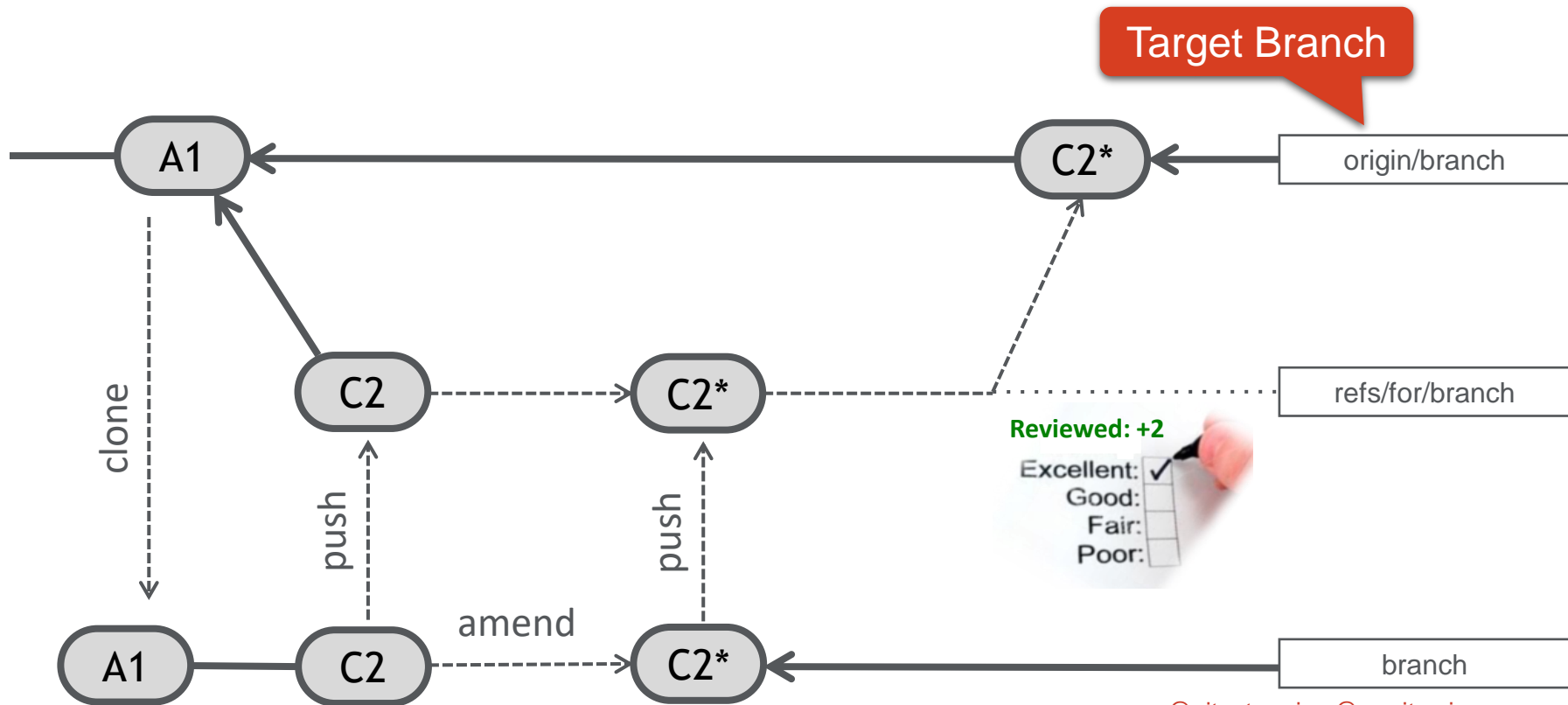
Code Review Workflow (9)



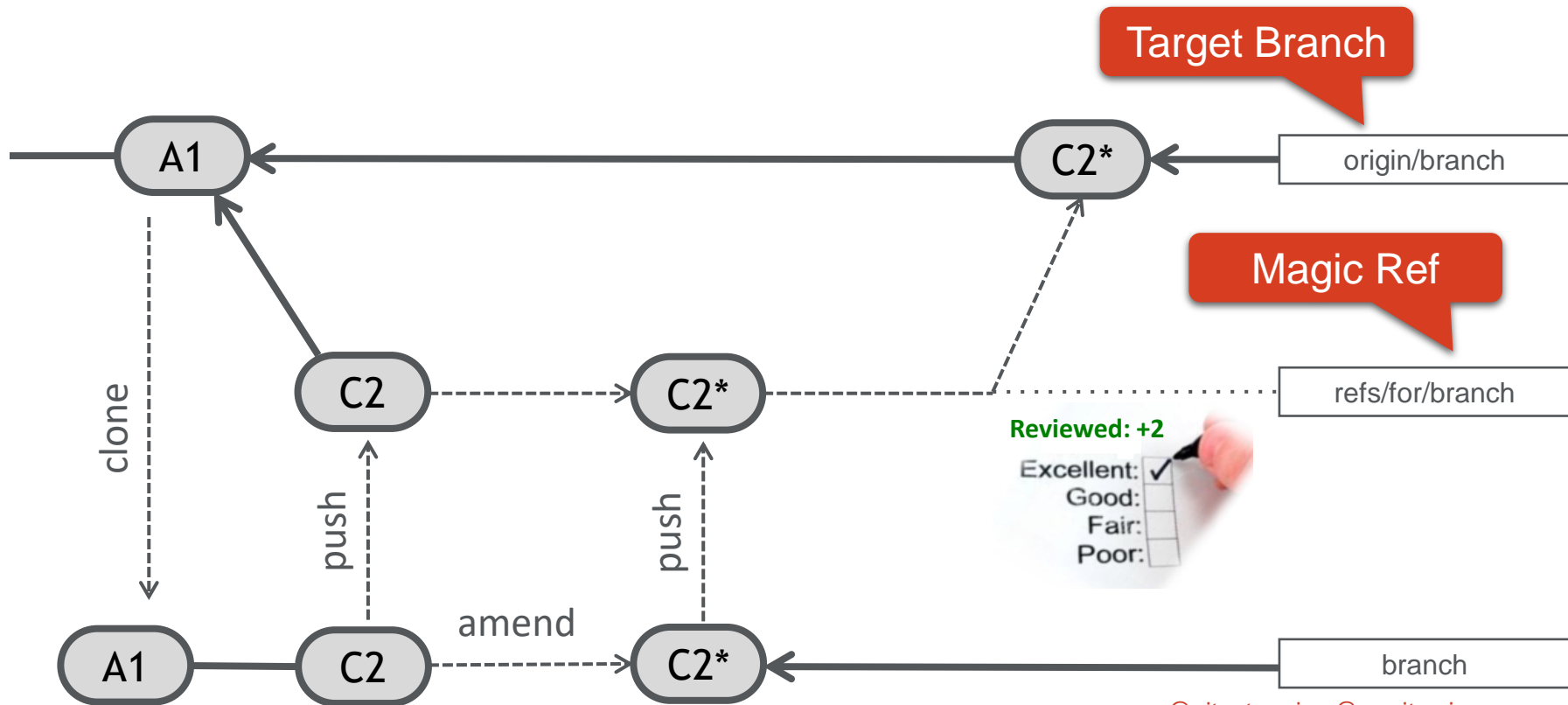
Code Review Workflow (10)



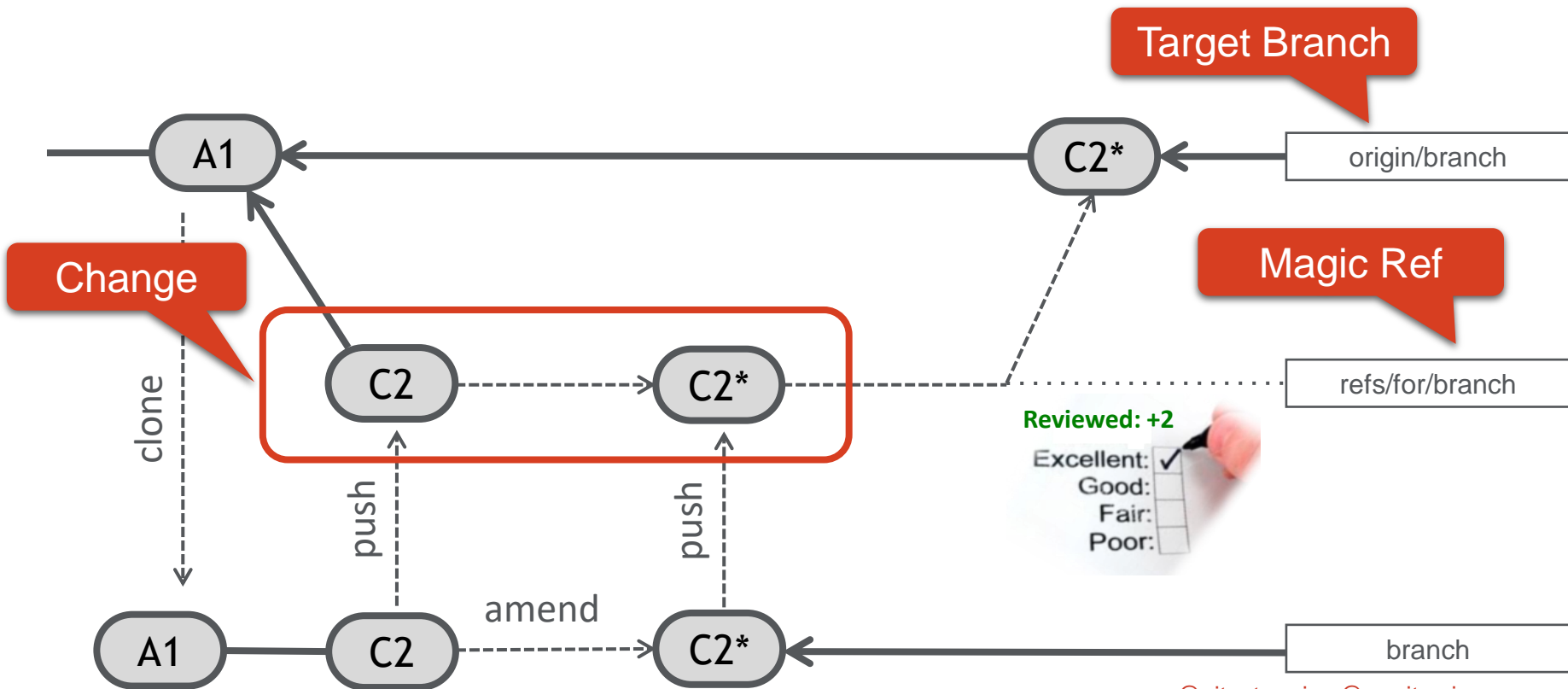
Gerrit concepts: target branch



Gerrit concepts: magic ref



Gerrit concepts: change (1)



Gerrit concepts: change (2)

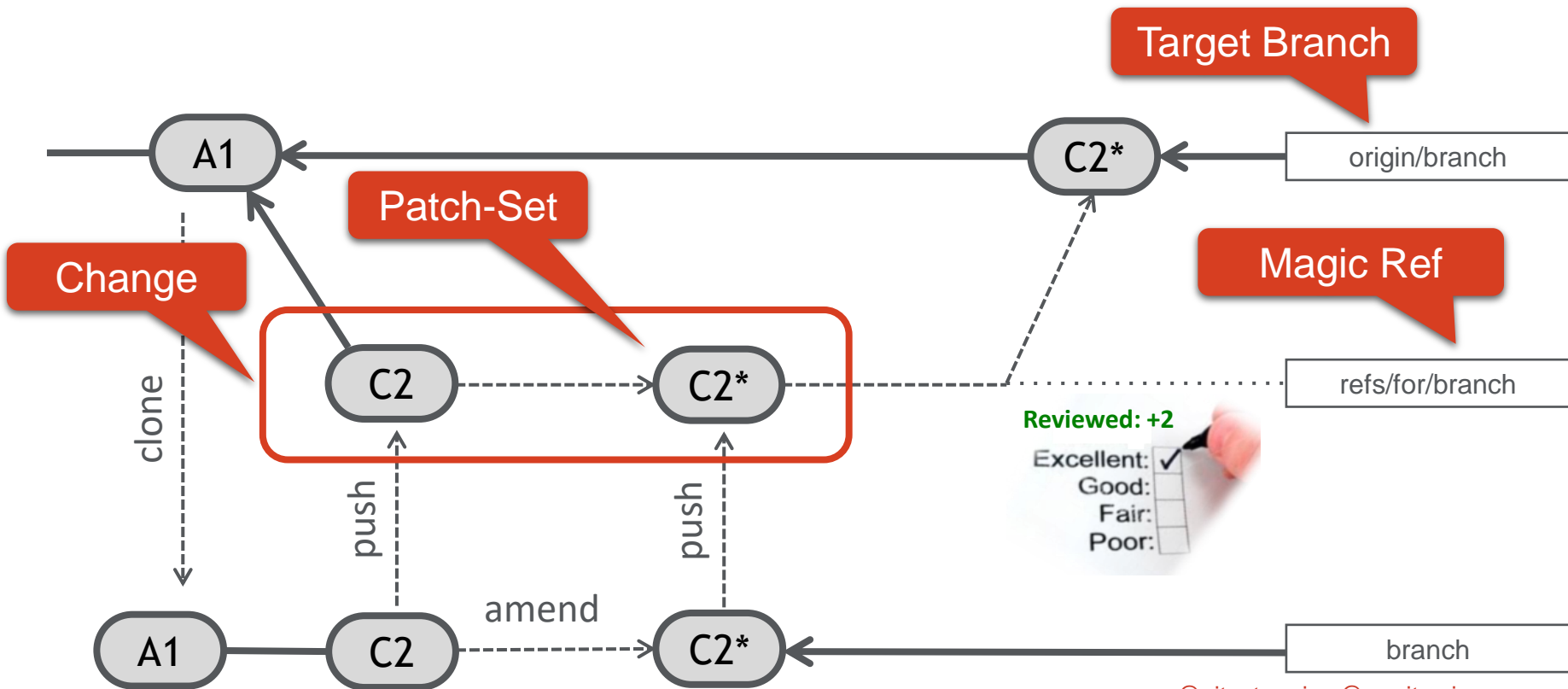
- Assigned by the Git client
- Generated by a Gerrit hook
- Appended to the Git commit footer meta-data
- Kept across multiple amendments to the commit

Commit headline for review

This is the Git commit message description
possibly multi line

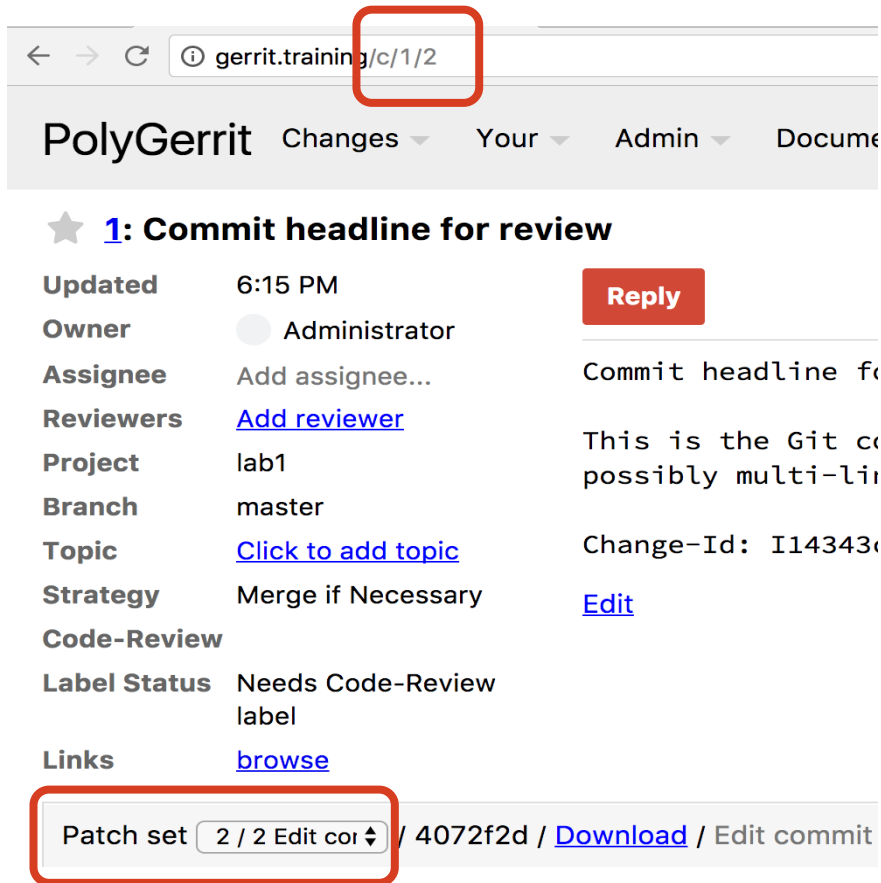
Change-Id: I14343c9ef7445558111ccfc345d5e9eb1d60529a

Gerrit concepts: patch-set (1)



Gerrit concepts: patch-set (2)

- Assigned by Gerrit Server
- Unique for every commit
- Identified by
 - Change Number (sequence)
 - Patch Number (sequence)



The screenshot shows the Gerrit web interface for a commit. The browser address bar shows 'gerrit.training/c/1/2', with '1/2' highlighted by a red box. The page title is 'PolyGerrit'. The main content area displays the commit details for '1: Commit headline for review'. The details include: Updated (6:15 PM), Owner (Administrator), Assignee (Add assignee...), Reviewers (Add reviewer), Project (lab1), Branch (master), Topic (Click to add topic), Strategy (Merge if Necessary), Code-Review (Needs Code-Review label), and Links (browse). A 'Reply' button is visible on the right. At the bottom, the 'Patch set' section shows '2 / 2 Edit cor' with a dropdown arrow, followed by the commit ID '4072f2d' and links for 'Download' and 'Edit commit'. The 'Patch set' section is also highlighted with a red box.

gerrit.training/c/1/2

PolyGerrit Changes Your Admin Document

★ **1: Commit headline for review**

Updated 6:15 PM [Reply](#)

Owner Administrator

Assignee Add assignee... Commit headline fo

Reviewers [Add reviewer](#)

Project lab1 This is the Git co
possibly multi-lin

Branch master

Topic [Click to add topic](#) Change-Id: I14343c

Strategy Merge if Necessary [Edit](#)

Code-Review

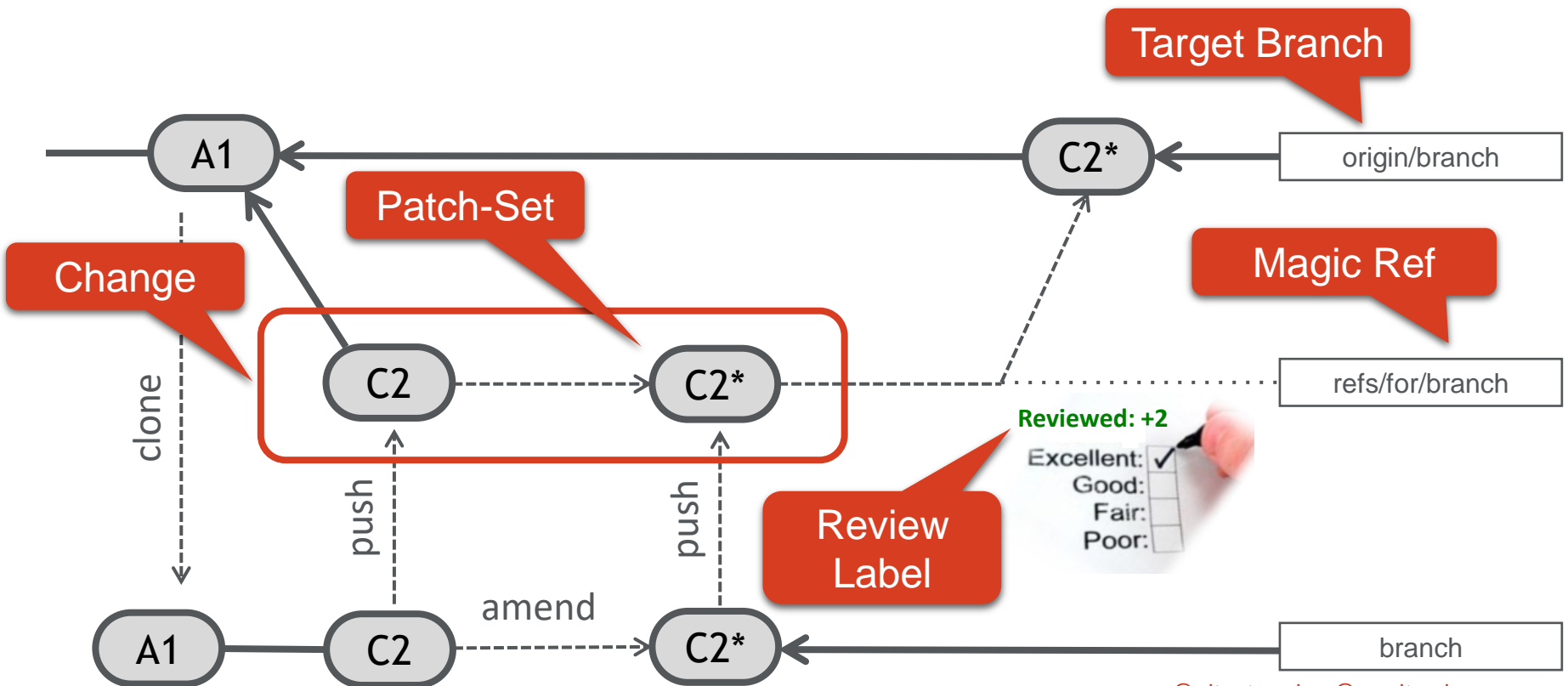
Label Status Needs Code-Review
label

Links [browse](#)

Patch set 2 / 2 Edit cor / 4072f2d / [Download](#) / Edit commit



Gerrit concepts: review label and score (1)



Gerrit concepts: review label and score (2)

- Label = category of review
- Default Labels:
 - Code-Review (human)
 - Verified (build)
- Score types
 - Neutral (comments)
 - Positive (good)
 - Negative (changes needed)
- Special Scores
 - Max = approved
 - Min = veto

PolyGerrit Changes ▾ Your ▾ Admin ▾ Docume

★ **1: Commit headline for review**

Updated	6:53 PM	Reply
Owner	Administrator	
Assignee	Add assignee...	Commit headline
Reviewers	Add reviewer	This is the Git possibly multi-1
Project	lab1	Change-Id: I1434
Branch	master	
Topic	Click to add topic	Edit
Strategy	Merge if Necessary	
Code-Review	+2 Administrator ✕	
Label Status	Ready to submit	
Links	browse	

Gerrit concept: Submit / Merge




- Last patch-set → target branch
- Enabled by rules on review
- Default Submit rules: Code-Review = +2 and Verified = +1

PolyGerrit Changes ▾ Your ▾ Admin ▾ Documentation ▾ Search

★ **1: Commit headline for review**

Updated
6:53 PM

[Reply](#) [More ▾](#) [Rebase](#) [Abandon](#) [Submit](#)

Owner
 Administrator

Commit headline for review

Assignee
Add assignee...

This is the Git commit message description
possibly multi-line

Reviewers
[Add reviewer](#)

Change-Id: I14343c9ef7445558111ccfc345d5e9eb1d60529a

[Edit](#)

Project

Gerrit Code Review building blocks:

1. Advanced Git Server

2. Review API

3. Extensible Workflow





First Code Review Workflow

Lab#2: Gerrit Project Page



← → ↻ gerrit.training/#/admin/projects/lab2

- All
 - My
 - Projects**
 - People
 - Plugins
 - Documentation
- List **General** Branches Tags Access Dashboards Create New Project

Project lab2

Clone | Clone with commit-msg hook | anonymous http | http

```
git clone http://admin@gerrit.training/a/lab2 && (cd lab2 && curl ..
```



Description

Copy to clipboard

Lab#2: Clone the repository with commit-msg hook

```
$ git clone http://admin@gerrit.training/a/lab2 && (cd lab2 && curl -kLo `git rev-parse --git-dir`/hooks/commit-msg  
http://admin@gerrit.training/tools/hooks/commit-msg; chmod +x `git rev-parse --git-dir`/hooks/commit-msg)
```

```
Cloning into 'lab2'...
```

```
remote: Counting objects: 2, done
```

```
remote: Finding sources: 100% (2/2)
```

```
Unpacking objects: 100% (2/2), done.
```

```
remote: Total 2 (delta 0), reused 0 (delta 0)
```

```
Checking connectivity... done.
```

% Total	% Received	% Xferd	Average Speed	Time	Time	Time	Current
			Dload Upload	Total	Spent	Left	Speed
100	4691	100	4691	0	0	527k	0
						--:--:--	--:--:--
						--:--:--	--:--:--
						--:--:--	572k

```
$ cd lab2
```

```
(master) $
```

Lab#2: Create a local commit and push for review



```
(master)$ echo "My first review" > first-review
(master)$ git add . && git commit -m "My first review"
[master 4ee8383] My first review
 1 file changed, 1 insertion(+)
 create mode 100644 first-review

(master)$ git push origin HEAD:refs/for/master
Counting objects: 3, done.
Writing objects: 100% (3/3), 309 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote: Processing changes: new: 1, refs: 1, done
remote:
remote: New Changes:
remote:  http://gerrit.training/2 My first review
remote:
To http://admin@gerrit.training/a/lab2
```

* [new branch] master -> refs/for/master

@gitenterprise @gerritreview

Lab#2: Push for review explained

```
(master)$ echo "My first review" > first-review
(master)$ git add . && git commit -m "My first review"
[master 4ee8383] My first review
1 file changed, 1 insertion(+)
create mode 100644 first-review

(master)$ git push origin HEAD:refs/for/master
Counting objects: 3, done.
Writing objects: 100% (3/3), 309 bytes/s, done.
Total 3 objects, 309 bytes compressed.
remote: Compressing objects: 100% (3/3), 309 bytes/s, done.
remote: Total 3 (delta 0), reused 0 (delta 0), compressed 3 to 3 bytes.
remote: New Changes:
remote: http://gerrit.training/2 My first review
remote:
To http://admin@gerrit.training/a/lab2
```

ref/for/master = magic
ref for reviews
targeting master

HEAD = latest commit
of current branch

* [new branch] master -> refs/for/master

@gitenterprise @gerritreview

Lab#2: List of open changes



gerrit.training/q/status:open

PolyGerrit Changes Your Admin Documentation status:open Search

Subject	Status	Owner	Project	Branch	Updated	Size	CR
★ My first review		Administrator	lab2	master	8:32 PM	+1, -0	

Commit headline

Commit author

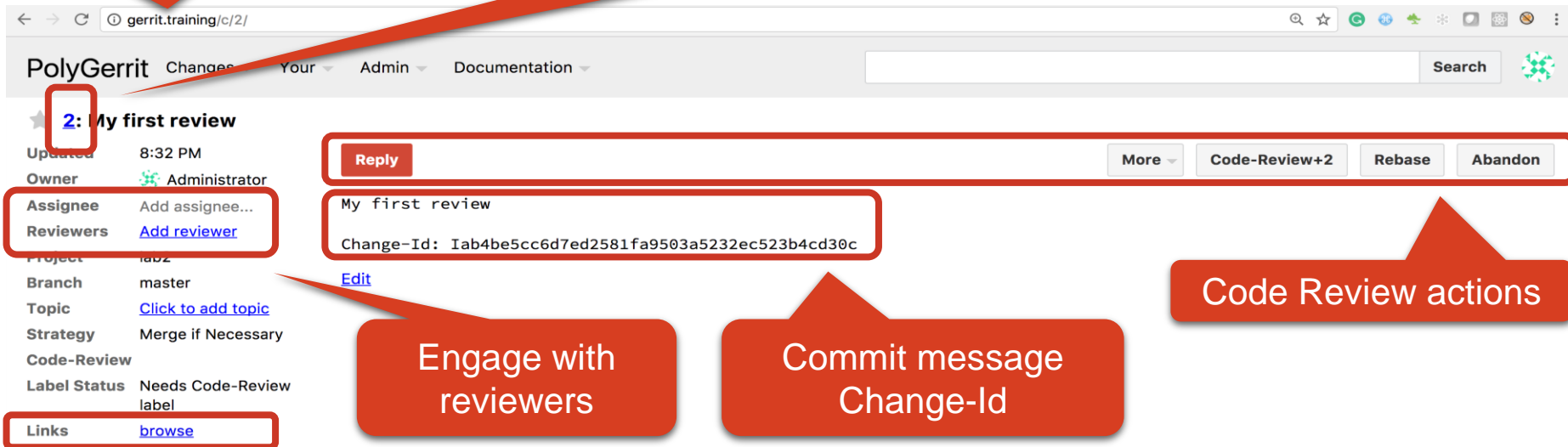
Gerrit project

Target branch

Lab#2: Change details (1)

Change URL

Change Number and Link



The screenshot shows a web browser displaying a GerritForge change page. The URL is `gerrit.training/c/2/`. The page title is "PolyGerrit" and the change title is "2: My first review". The change is owned by "Administrator" and is in the "master" branch. The change ID is `Iab4be5cc6d7ed2581fa9503a5232ec523b4cd30c`. The page includes a "Links" section with a "browse" link, a "Code-Review" section with a "Needs Code-Review label" status, and a "Code Review actions" section with buttons for "Reply", "More", "Code-Review+2", "Rebase", and "Abandon".

Annotations on the screenshot include:

- A red box around the change number "2" in the title, with a callout pointing to the "Change URL" text.
- A red box around the change ID, with a callout pointing to the "Change Number and Link" text.
- A red box around the "Assignee" and "Reviewers" fields, with a callout pointing to the "Engage with reviewers" text.
- A red box around the "Code-Review" section, with a callout pointing to the "Open review in Code Browser (Gitiles)" text.
- A red box around the "Code Review actions" section, with a callout pointing to the "Code Review actions" text.
- A red box around the "Change-Id" field, with a callout pointing to the "Commit message Change-Id" text.

Engage with reviewers

Commit message Change-Id

Code Review actions

Open review in Code Browser (Gitiles)



Lab#2: Change details (2)

Current Patch-Set

Git commands to fetch the Patch-Set

Diff view controls

Patch set 1/1 / 4ee8383 [Download](#) [Add a patch set description](#)

[Show diffs](#) / [Hide diffs](#) / [Side By Side](#) / Diff against [Base](#)

Files

Commit message

File

```
1 Parent: fb57fee7 (Initial empty repository)
2 Author: Luca Milanesio <luca.milanesio@gmail.com>
3 AuthorDate: 2017-05-21 19:31:36 +0100
4 Commit: Luca Milanesio <luca.milanesio@gmail.com>
5 CommitDate: 2017-05-21 19:31:36 +0100
6
7 My first review
8
9 Change-Id: Iab4be5cc6d7ed2581fa9503a5232ec523b4cd30c
```

A first-review +1 -0

File

```
1 My first review
```

+1 -0

Side-by-Side
Or
Unified diff-view

Change audit
including
review comments

Messages

Administrator Uploaded patch set 1.

[Expand all](#) / [Show comments only](#)

8:32 PM

Lab#2: Adding in-line comments



Patch set / 4ee8383 / [Download](#) / [Add a patch set description](#)

Files

[Commit message](#)

File	File
1	Parent: fb57fee7 (Initial empty repository)
2	Author: Luca Milanesio <luca.milanesio@gmail.co
3	AuthorDate: 2017-05-21 19:31:36 +0100
4	Commit: Luca Milanesio <luca.milanesio@gmail.co
5	CommitDate: 2017-05-21 19:31:36 +0100
6	
7	My first review
8	
9	Change-Id: Iab4be5cc6d7ed2581fa9503a5232ec523b4cd30

Press C to comment.

Patch set / 4ee8383 / [Download](#) / [Add a patch set description](#)

Files

[Commit message](#)

File	File
1	Parent: fb57fee7 (Initial empty repository)
2	Author: Luca Milanesio <luca.milanesio@gmail.
3	AuthorDate: 2017-05-21 19:31:36 +0100
4	Commit: Luca Milanesio <luca.milanesio@gmail.
5	CommitDate: 2017-05-21 19:31:36 +0100
6	
7	My first review
DRAFT ⓘ	
<input type="text" value="Are you sure?"/>	
<input type="button" value="Save"/> <input type="button" value="Cancel"/> <input type="button" value="Discard"/>	
8	
9	Change-Id: Iab4be5cc6d7ed2581fa9503a5232ec523b4cc

Lab#2: Draft comments review

Patch set / 4ee8383 / [Download](#) / [Add a patch set description](#)

Files

Commit message

File	File
1	Parent: fb57fee7 (In...
2	Author: Luca Milanesio <luca.milanesio@gmail.com>
3	AuthorDate: 2017-05-21 19:31:36 +0100
4	Commit: Luca Milanesio <luca.milanesio@gmail.com>
5	CommitDate: 2017-05-21 19:31:36 +0100
6	
7	My first review
8	
9	Change-Id: Iab4be5cc6d7ed2581fa9503a5232ec523b4cd30c

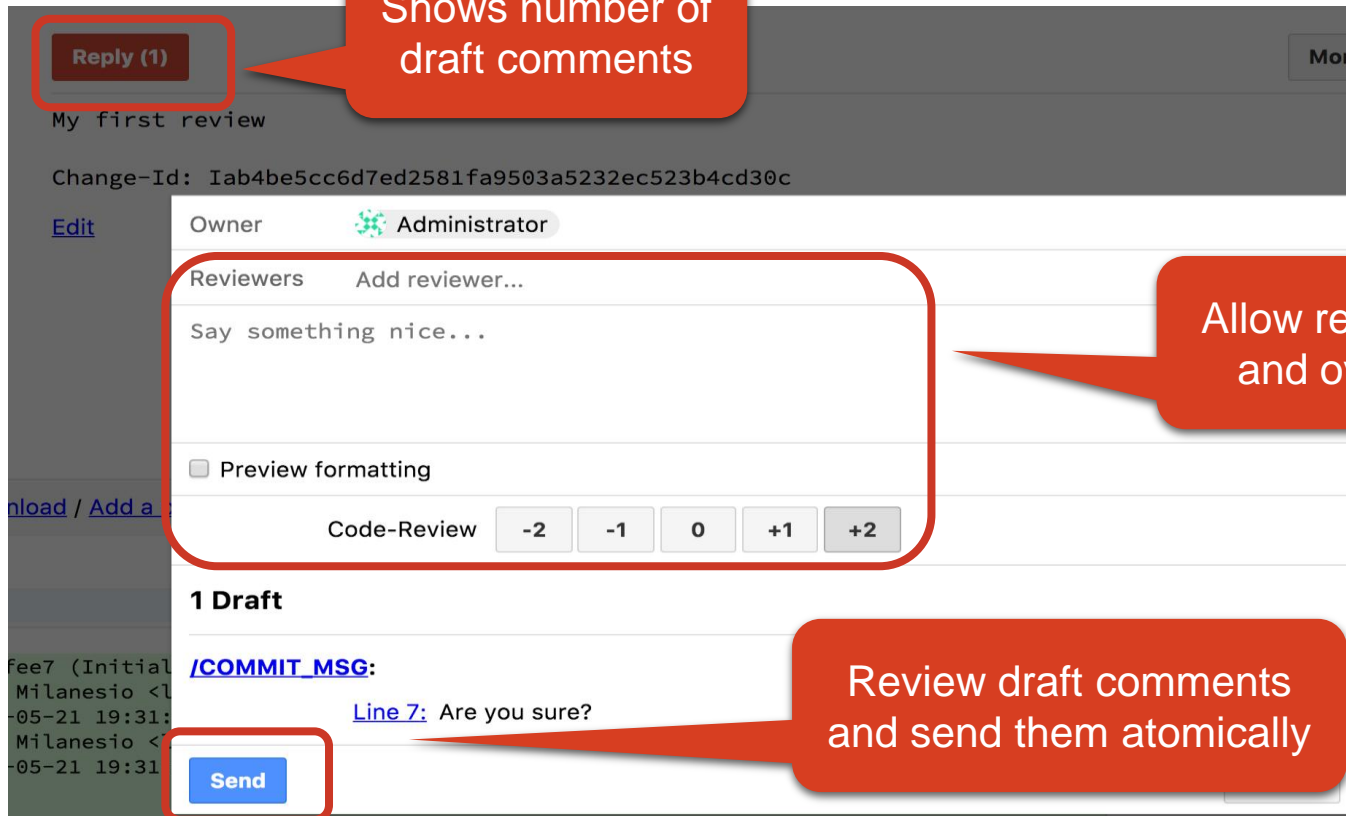
DRAFT ⓘ

Are you sure?

Comments can be associated to words, entire line or entire file

Comments can be added, edited, discarded and then sent atomically

Lab#2: Sending draft comments



The screenshot shows the GerritForge review interface. A red box highlights the 'Reply (1)' button. A red callout bubble points to it with the text 'Shows number of draft comments'. Another red box highlights the main review form area, including the 'Say something nice...' text area, the 'Preview formatting' checkbox, and the 'Code-Review' buttons (-2, -1, 0, +1, +2). A red callout bubble points to this area with the text 'Allow review label, score and overall message'. A third red box highlights the 'Send' button at the bottom. A red callout bubble points to it with the text 'Review draft comments and send them atomically'. The interface also shows the 'Owner' as 'Administrator', a 'Change-Id', and a '1 Draft' section with a commit message and a line of code.

Allow review label, score and overall message

Review draft comments and send them atomically

Lab#2: Review notification



- All reviewers receive notification via e-mail (see example)

David Pursehouse (Gerrit)

Change in gerrit[stable-2.14]: Bazel: Allow plugins to non-transitively depend on prolog rules

To: David Ostrovsky, Luca Milanesio,

Reply-To: change-107003@gerrit-review.googleusercontent.com

David Pursehouse **posted comments** on this change.

[View Change](#)

Patch set 2: **Code-Review +2**

(1 comment)

[File gerrit-plugin-api/BUILD:](#)

[Patch Set #2, Line 22:](#) `"//gerrit-server/src/main/prolog:common",`

Unrelated to this change, because it's like this in other places too, but why do we need to explicitly specify "src/main" here?

To view, visit [change 107003](#). To unsubscribe, visit [settings](#).

review

Lab#2: Reply to comments and keep the discussion thread



Messages

[Expand all](#) / [Show comments only](#)



Administrator Uploaded patch set 1.

8:32 PM



Administrator

Patch Set 1: Code-Review+2

10:43 PM

(1 comment)

[/COMMIT_MSG:](#)

[Line 7:](#) Are you sure?

Reply



Administrator

Patch Set 1:

10:43 PM

(1 comment)

[/COMMIT_MSG:](#)

[Line 7:](#) Done

Reply

Change
message
s
keep the
message
thread

Lab#2: Submit the change



PolyGerrit

Changes ▾

Your ▾

Admin ▾

Documentation ▾

Search



★ 2: My first review

Updated
10:43 PM


Reply

More ▾

Rebase

Abandon

Submit

Owner
 Administrator

My first review

Change-Id: Iab4be5cc6d7ed2581fa9503a5232ec523b4cd30c

Assignee
Add assignee...

[Edit](#)

Submit rules are satisfied.
Press the Submit button

@gitenterprise @gerritreview

Lab#2: See the change in code browser (Gitles)

← → ↻ gerrit.training/plugins/gitiles/lab2/+/master 🔍 ☆ 🌐 🌱 ⚙️ 📺 ⚙️ 🚫 ⋮

Gitiles [Code Review](#) Administrator [Sign Out](#)

[gerrit.training](#) / [lab2](#) / **master**

```
commit 4ee83830654bec8ba948299108bedf0a090e74fa [log] [tgz]
author Luca Milanesio <luca.milanesio@gmail.com> Sun May 21 19:31:36 2017 +0100
committer Luca Milanesio <luca.milanesio@gmail.com> Sun May 21 19:31:36 2017 +0100
tree e171d9b1bc328ce6625be9c4d45649e3f4b3345f
parent fb57fee7783d8be8cd3d16572599a0c61c66d640 [diff]
```

My first review

Change-Id: [Iab4be5cc6d7ed2581fa9503a5232ec523b4cd30c](#)

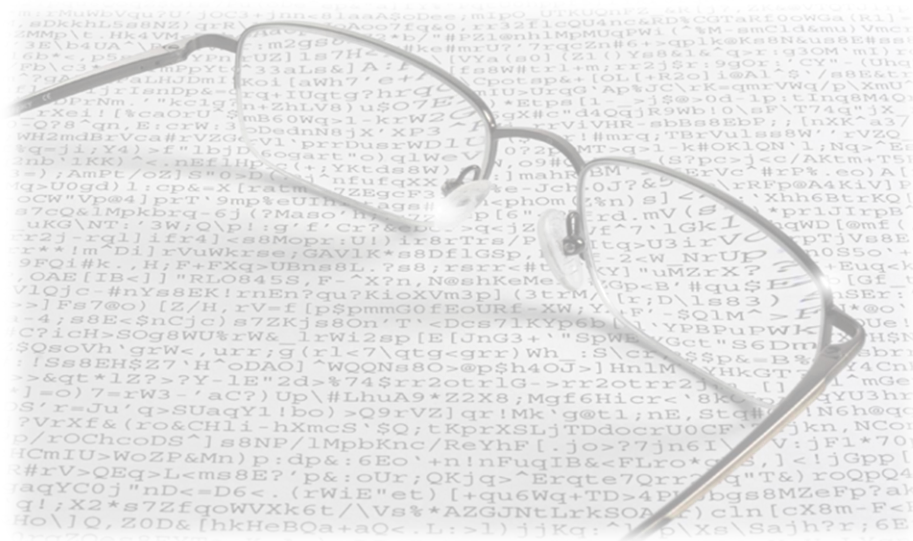
[first-review](#) [Added - diff]

1 file changed

```
tree: e171d9b1bc328ce6625be9c4d45649e3f4b3345f
```

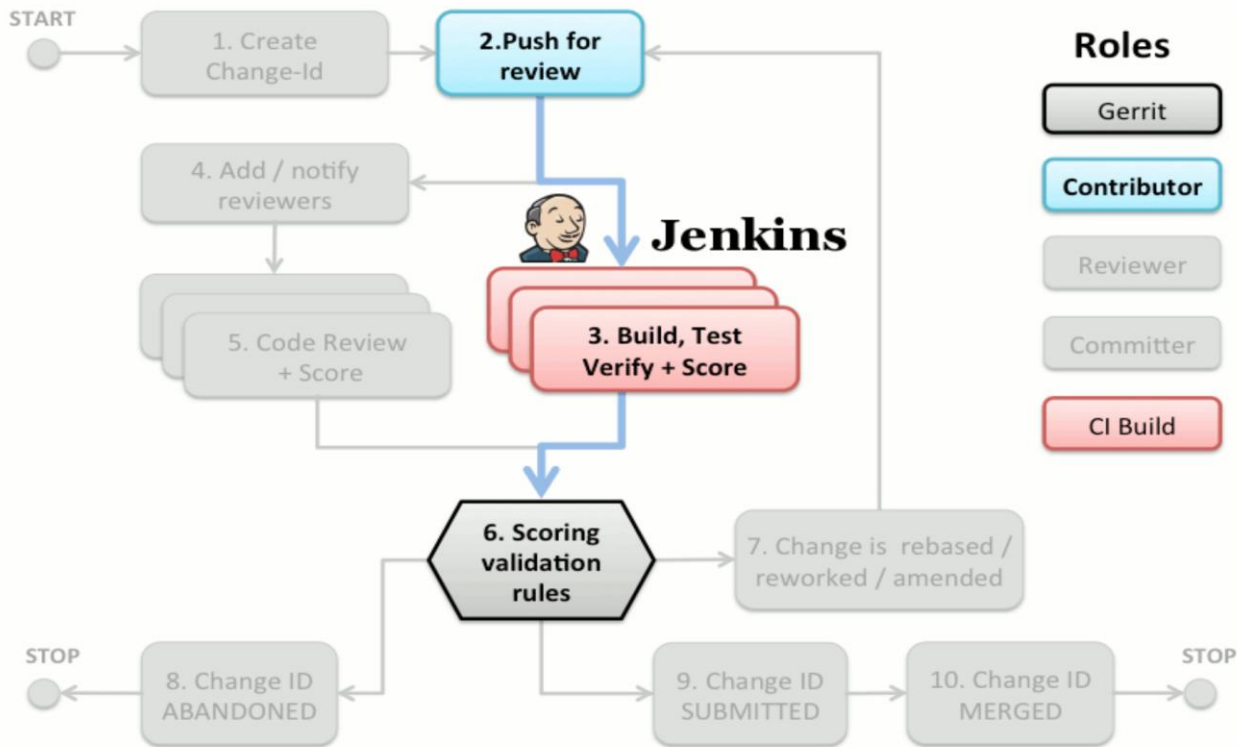
 [first-review](#)

Keeps
reference
to
Change-Id



Validating Commits Automatically

Jenkins Build Validation

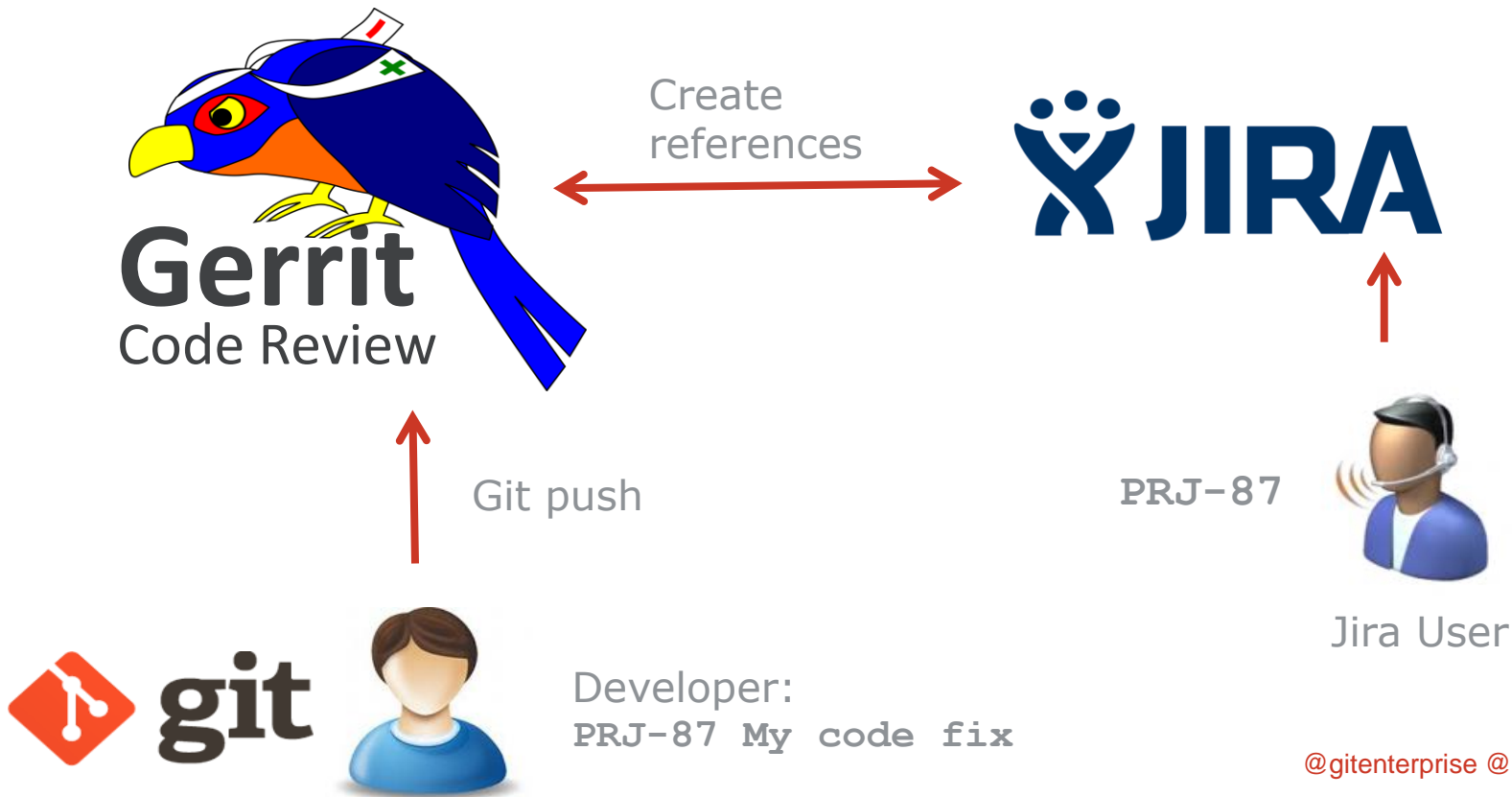


- ### Roles
- Gerrit
 - Contributor**
 - Reviewer
 - Committer
 - CI Build

Gives Verified +1 / -1 based on the Build and Tests

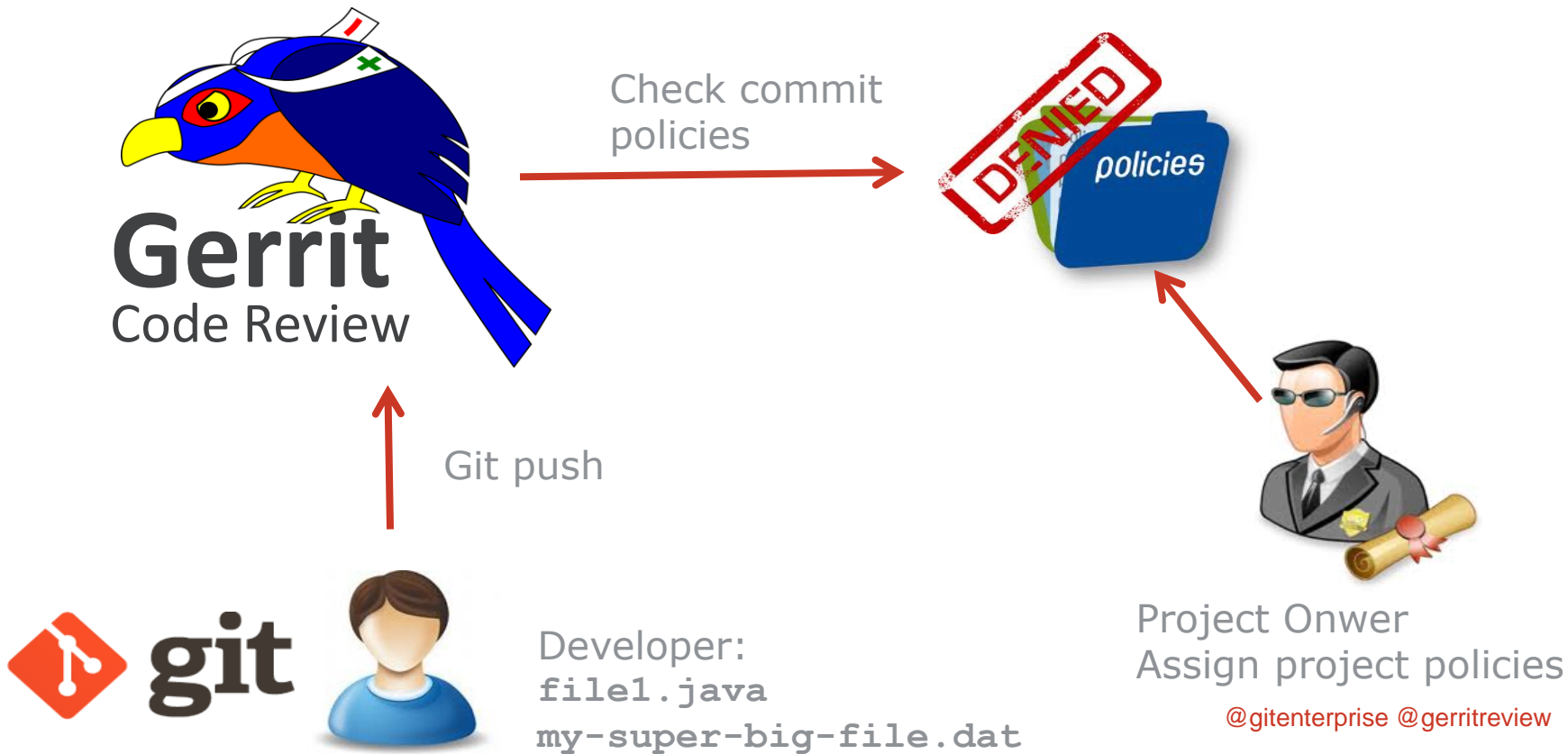
- Post Change / Patch-Set upload trigger
 - Triggers Jenkins build on new Changes
 - Triggers Jenkins build on new Patch-Sets
- Feedback as auto-review
 - Automatic scoring when build completed
 - Score → Submit rule evaluation

Atlassian Jira validation



- Validate commit during the push
 - Commit message restrictions (regex)
 - Existence of the Jira ticket
- Cross-reference Change to Jira ID
 - Change → Jira
 - Jira → Change
 - Change audit-trail in Jira
- Blocks creation of Change / Patch-Set / Commits
 - Push failed
 - Failure description

Content validation



Content Validation during push

- Project owner access
- Define blocks
- Require footer
- Block keywords

Binary Types 🗨️:

Blocked Content Type 🗨️:

Blocked File Extensions 🗨️:

Blocked Keyword Pattern 🗨️:

Invalid Filename Pattern 🗨️:

Max Path Length 🗨️:

Projects 🗨️:

Refs 🗨️:

Reject Duplicate Pathnames Locale 🗨️: en

Required Footers 🗨️:

Binary Types

+ [input]
Delete

Blocked Content Type

+ [input]
Delete

Blocked Content Type Whitelist 🗨️

Blocked File Extensions

+ [input]
Delete

Blocked Keyword Pattern

+ [input]
Delete

Invalid Filename Pattern

+ [input]
Delete

0

Projects

+ [input]
Delete

Refs

+ [input]
Delete

Reject Duplicate Pathnames 🗨️

Reject Submodules 🗨️

Reject Symbolic Links 🗨️

Reject Windows Line Endings 🗨️

Required Footers

+ [input]



Experiment Jenkins, Jira and Content Validations

Questions?



Q&A

@gitenterprise @gerritreview



Add additional patches to
existing Changes



Create dependent Changes



Server-side Change rebase and conflict resolution



- Each Change should add something **USABLE**
 - Change = complete feature
OR
 - Change = complete bug-fix
- Each Change should be **STABLE**

- Each change should focus on ONLY ONE THING
 - do not mix features, bug-fixes
- Why is this bad?
 - You need the bug-fix in another branch.
(git cherry-pick is not easy)
 - The feature broke something.
(git revert cannot be used)
 - It's more difficult to review.

- Push only Changes that are **READY**
 - Q: Who wants to merge unfinished changes?
 - Q: Who wants to review unfinished changes?
- It is unclear for reviewers
 - what is an issue and
 - what is simply not done yet

- Explain the WHY for the Change
 - what was changed can be seen from the diff
- Example: is this commit message good or bad?
"Disable category GET API"
 - Commit Message without motivation
 - No info about why this had to be disabled.
... git blame gets less useful ☹️

- Make BIG features as series of SMALL Changes
- Each Change adds something usable
- Mark the change series as a topic

Q&A