Stop merging broken code with ZUUL

Jin Nguyen

Trinh Nguyen

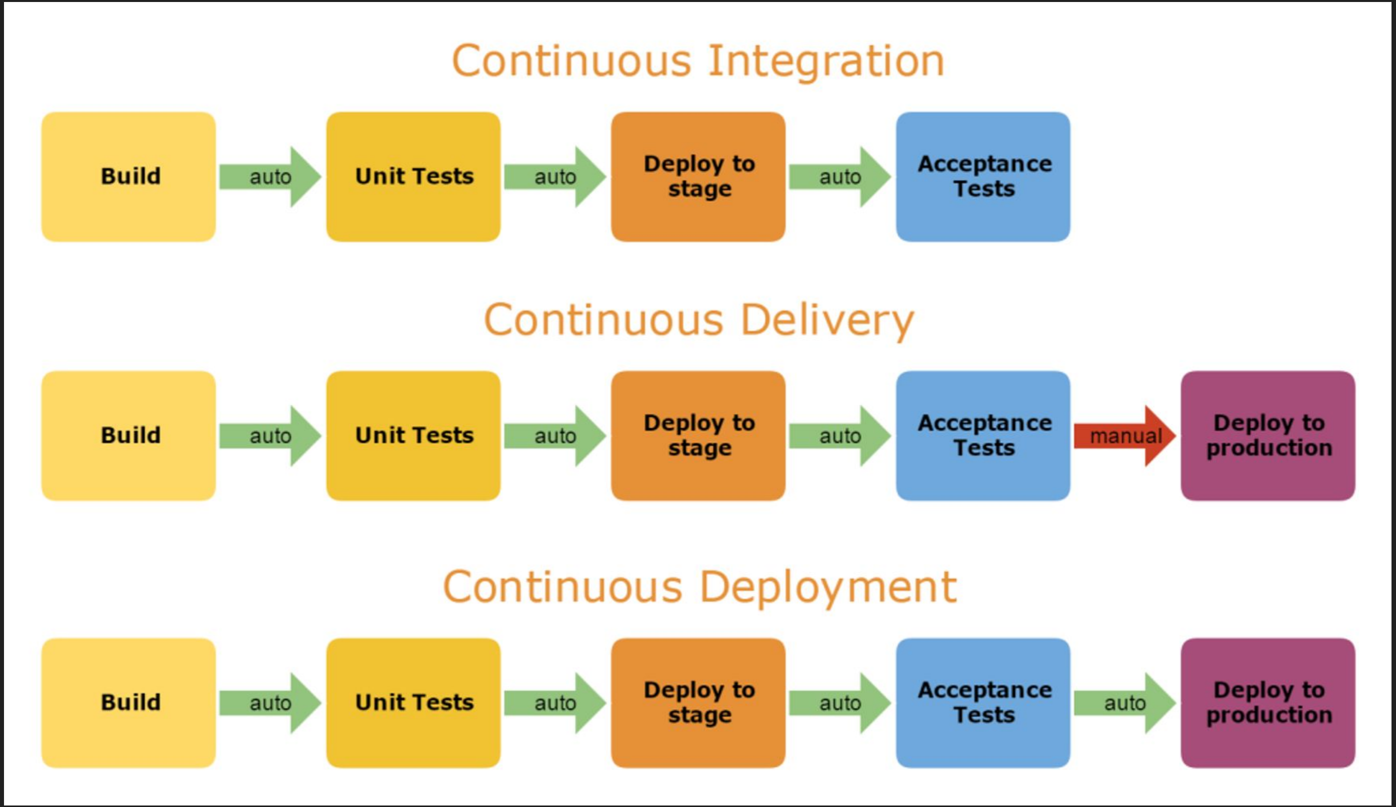
- Cloud Solution Architect & Product Manager, FPT Korea
- VietOpenInfra & VietOpenMEC Organizer
- Project Team Lead for OpenStack Searchlight (Stein~Victoria)
- My universe: dangtrinh.com
- Reach me at: dangtrinhnt@gmail.com / @dangtrinhnt

About me

Agenda

- CI/CD
- What is ZUUL?
- What we are doing
- Issues
- The future

CI/CD



Continuous Integration

Build → auto → Unit Tests → auto → Deploy to stage → auto → Acceptance Tests

Continuous Delivery

Build → auto → Unit Tests → auto → Deploy to stage → auto → Acceptance Tests → manual → Deploy to production

Continuous Deployment

Build → auto → Unit Tests → auto → Deploy to stage → auto → Acceptance Tests → auto → Deploy to production

https://www.linuxnix.com/what-is-continuous-integration-delivery-deployment-and-ci-cd-pipeline/cicd/

https://technologyconversations.com/2017/03/06/the-ten-commandments-of-continuous-delivery/

1. Be Agile
2. Refactor
3. Educate Everyone
4. Be Small
5. Practice TDD
6. Define Your CD Pipeline As Code
7. Have a Fast Pipeline
8. Consider Fixing a Failed Pipeline As Highest Priority
9. Run The CD Pipeline Locally
10. Commit Only To The Master Branch (or short-lived branches)

https://technologyconversations.com/2017/03/06/the-ten-commandments-of-continuous-delivery/

What is Zuul?

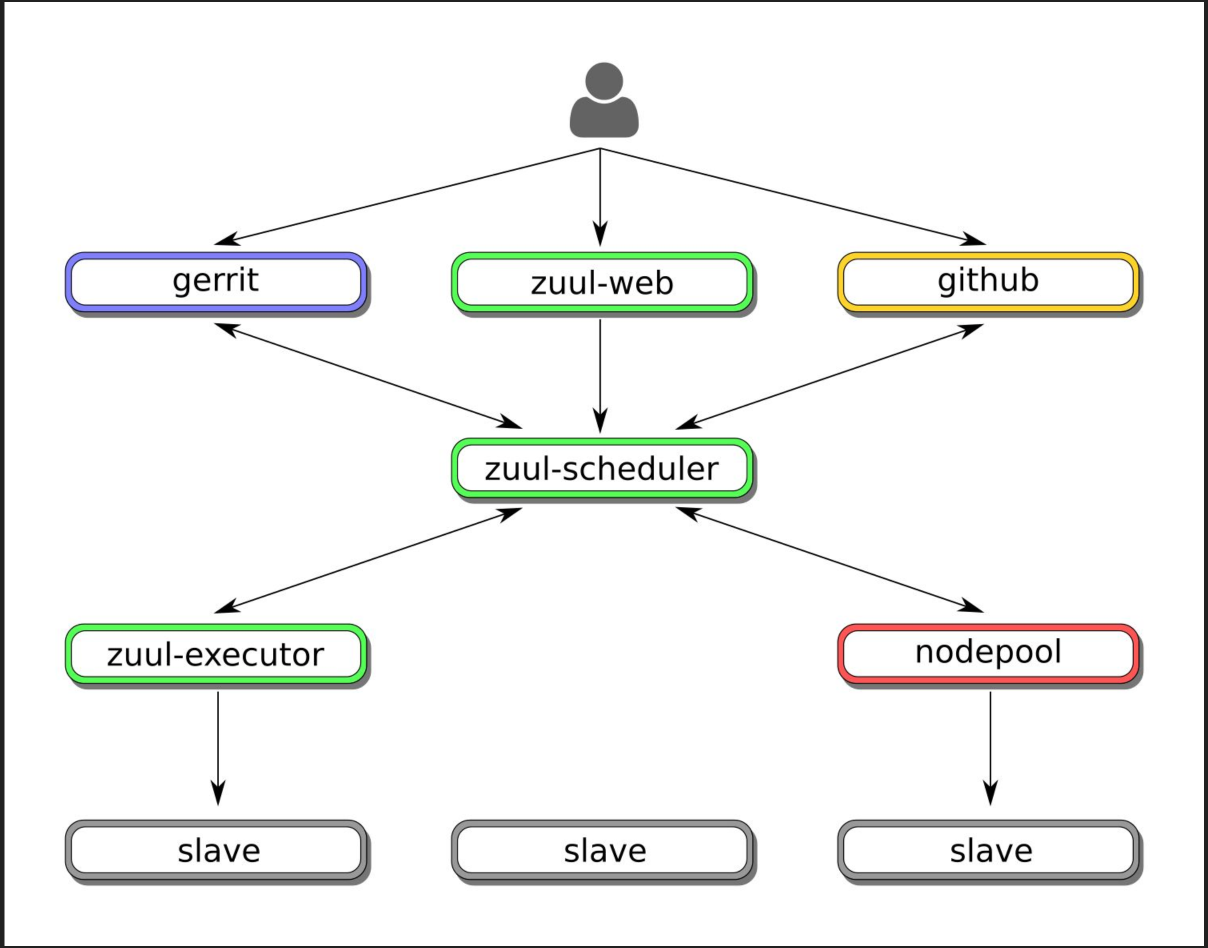
Zuul’s configuration is organized around the concept of a pipeline. In Zuul, a pipeline encompasses a workflow process which can be applied to one or more projects. For instance:

- A “**check**” pipeline might describe the actions which should cause newly proposed changes to projects to be tested.
- A “**gate**” pipeline might implement Project Gating to automate merging changes to projects only if their tests pass.
- A “**post**” pipeline might update published documentation for a project when changes land.

What is Zuul?

- **Gating on git branches**
- **Speculative testing**
- **Scaling**
- **Pipelines**
- **Multi-repository**
- **Parallel testing**
- **In-repository job configuration**
- **Pre-merge job loading**
- **Multi-node job support**
- **Ansible support**

What is Zuul?

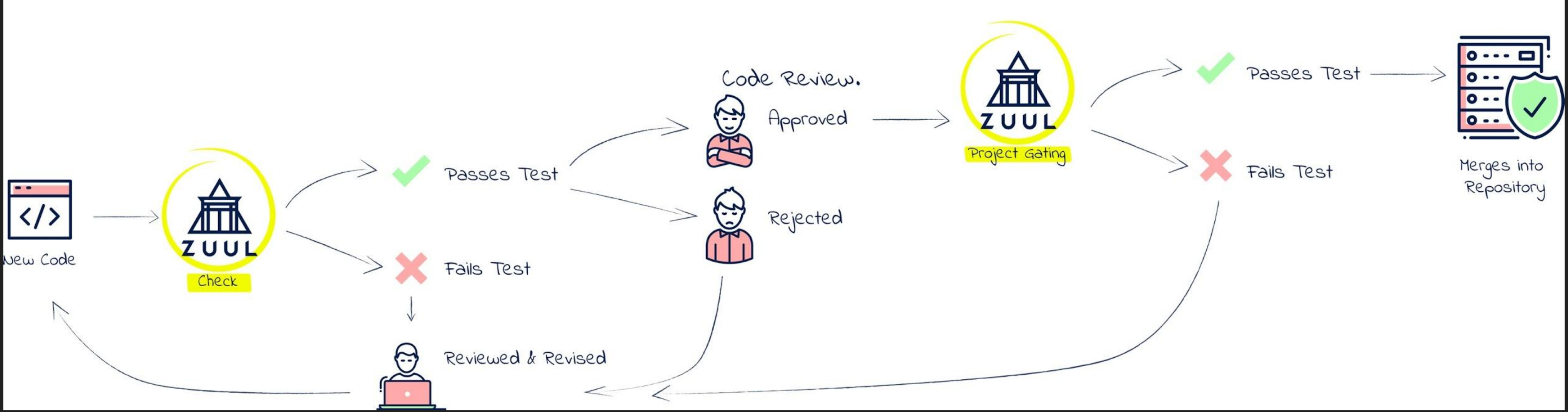


https://www.softwarefactory-project.io/zuul-hands-on-part-1-what-is-zuul.html

What is Zuul?

- **Code review (CR) system** hosts the changes to gate with Zuul. Zuul supports Gerrit (code reviews) and Github (Pull Requests).
- **zuul-scheduler** receives events from remote CR systems, and schedules the execution of jobs according to a project's job configuration; then reports job results to the CR system.
- **zuul-executor** uses Ansible to execute jobs remotely on test nodes provided by Nodepool.
- **Nodepool** launches, provisions and ultimately destroy nodes needed to run tests jobs (e.g., AWS EC2, OpenStack instances, etc.).
- **zuul-web** is Zuul's Web frontend and provides a REST API.

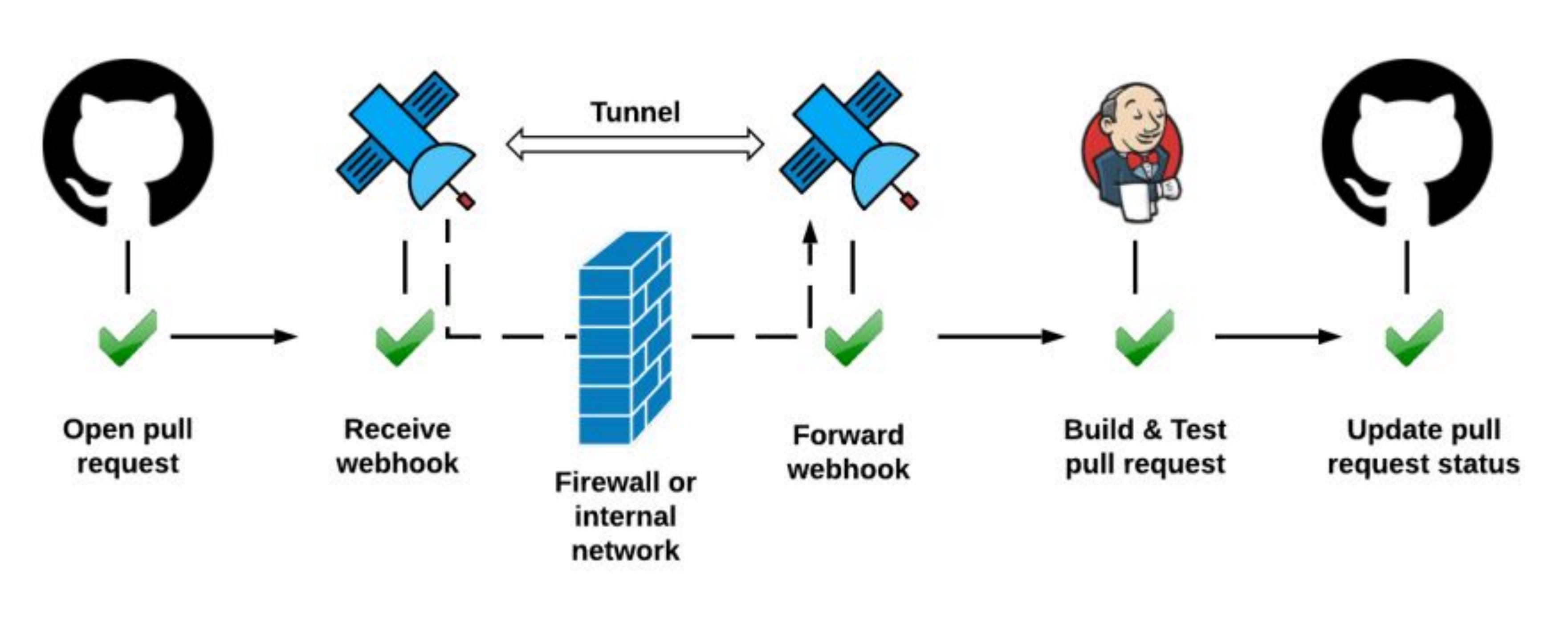
What is Zuul?



vexxhost

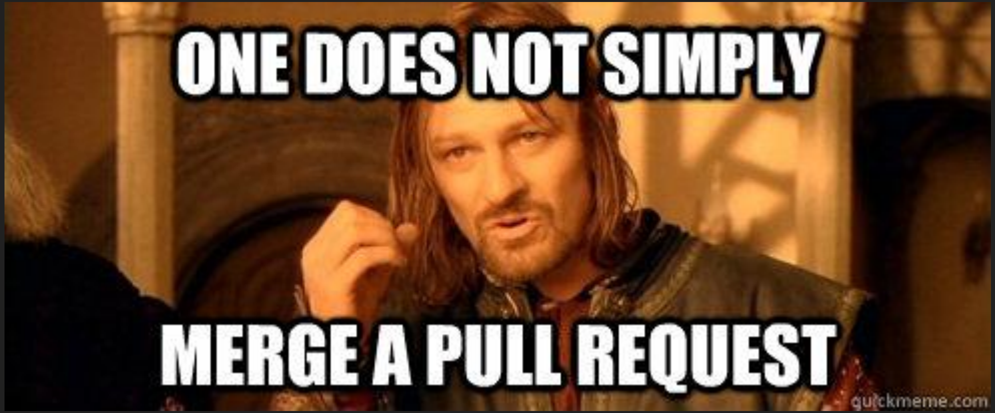
What we are doing

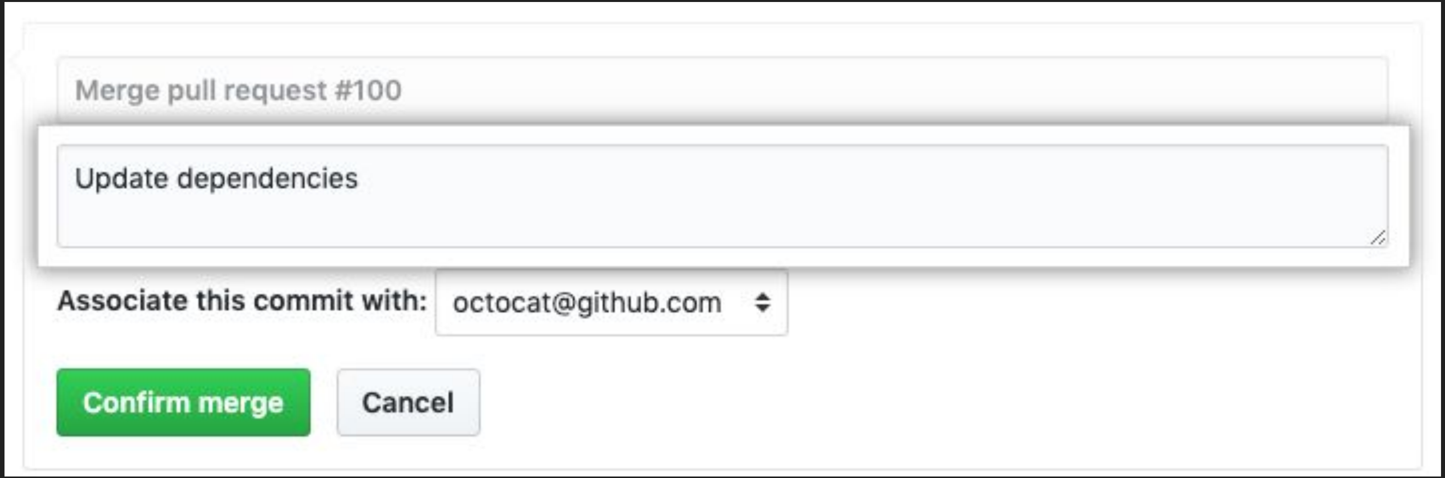
Trigger test (only 1) through Jenkins


Tunnel
Open pull request
Receive webhook
Firewall or internal network
Forward webhook
Build & Test pull request
Update pull request status

What we are doing

Manually merge PRs



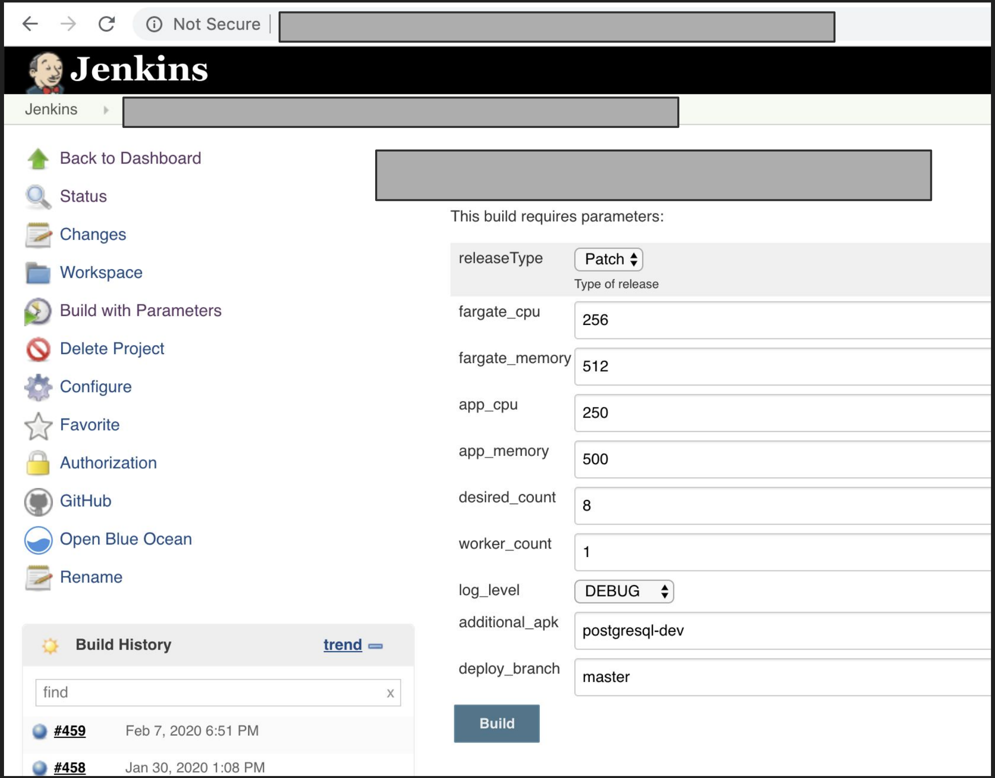

Merge pull request #100

Update dependencies

Associate this commit with: octocat@github.com

What we are doing

Manually deploy



[redacted]

[redacted]

[redacted]

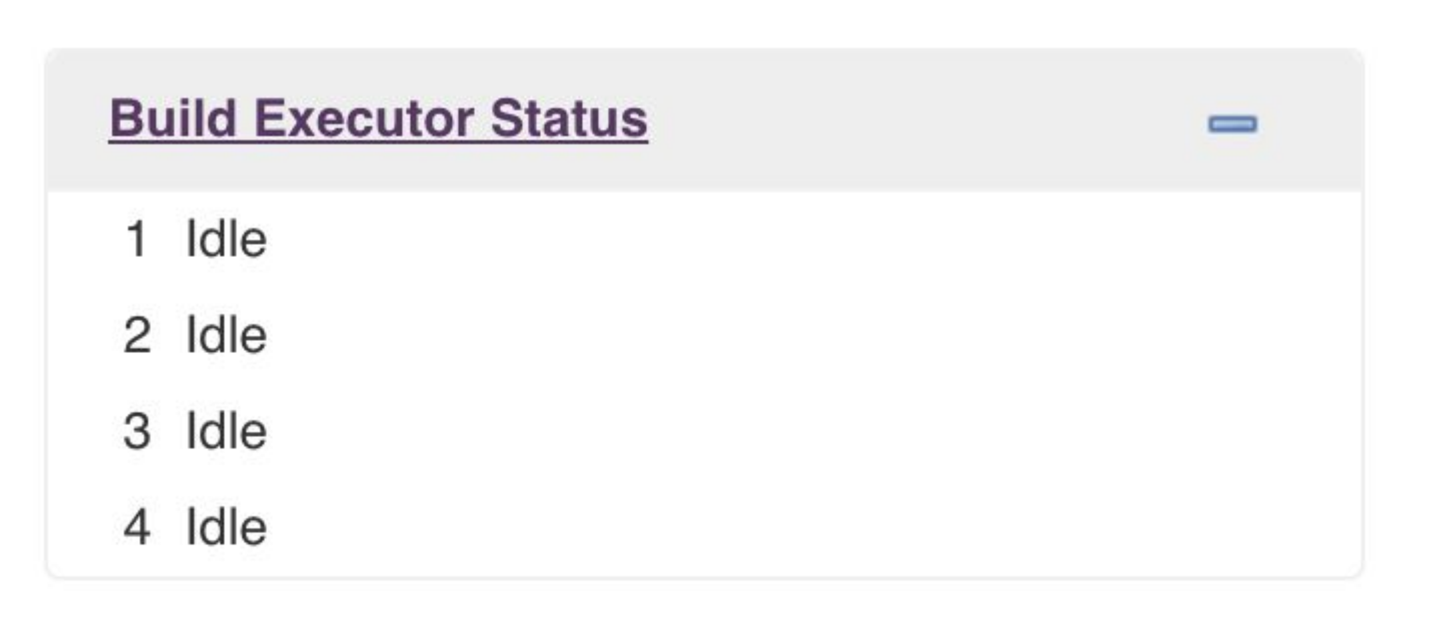
This build requires parameters:

releaseType	Patch Type of release
fargate_cpu	256
fargate_memory	512
app_cpu	250
app_memory	500
desired_count	8
worker_count	1
log_level	DEBUG
additional_apk	postgresql-dev
deploy_branch	master

#459	Feb 7, 2020 6:51 PM
#458	Jan 30, 2020 1:08 PM

More manual works == More errors

Issues



Build Executor Status

1	Idle
2	Idle
3	Idle
4	Idle

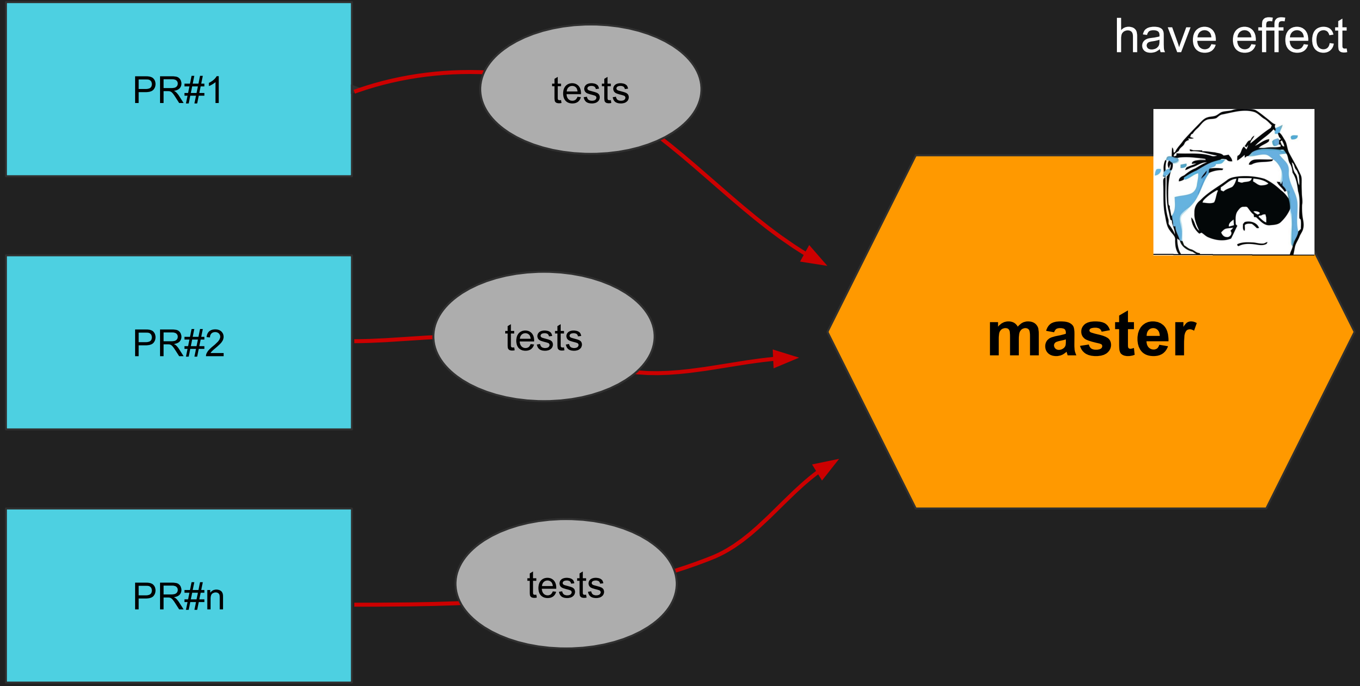
Issues

fb.com/cgrecord.net
One hour on this planet is 7 years on Earth
Good, we'll wait for our Jenkins build here!

Issues

Issues

master is not guaranteed for building or passing the tests because PRs may have effect on each others.



PR#1

tests

PR#2

tests

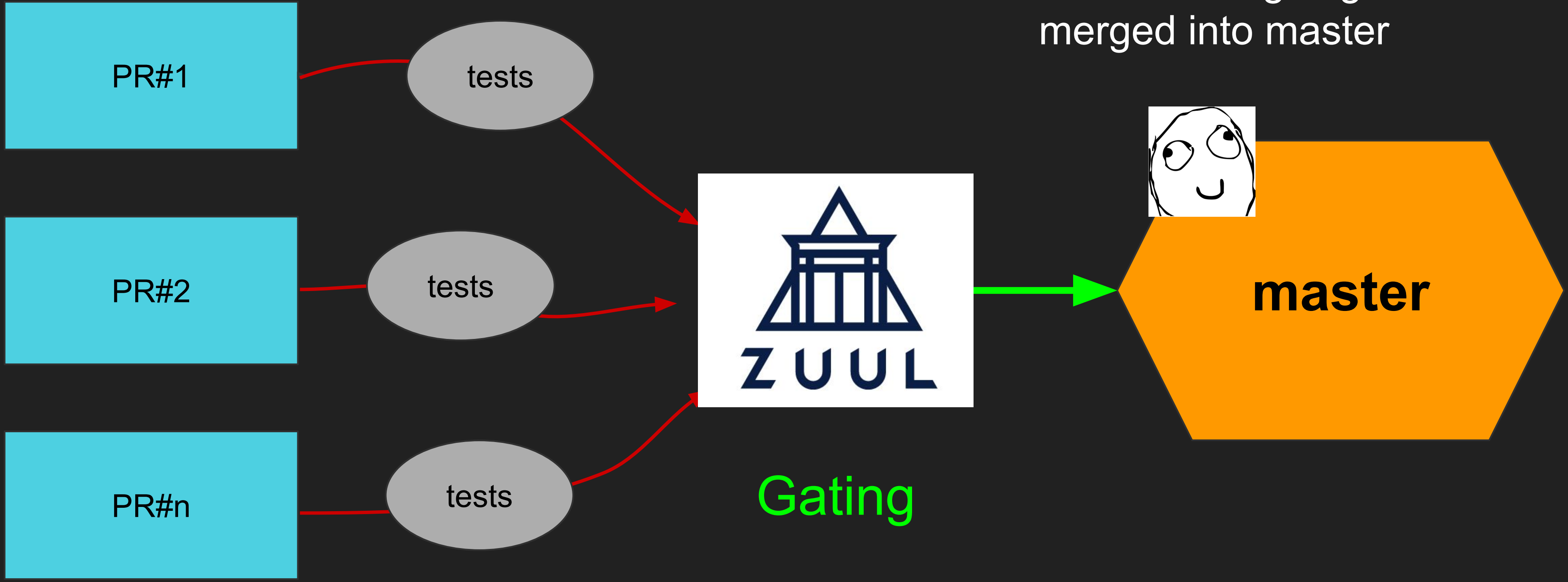
PR#n

tests

master

Solution

Zuul coordinates all the PRs and makes sure broken changes get merged into master



PR#1

PR#2

PR#n

tests

tests

tests

ZUUL

master

Gating

Project gating

What is project gating? And why Zuul?

- The process of gating attempts to prevent changes that introduce regressions from being merged
- Keeps the mainline of development open and working for all developers, and only when a change is confirmed to work without disruption is it merged
- Zuul can help automate this process

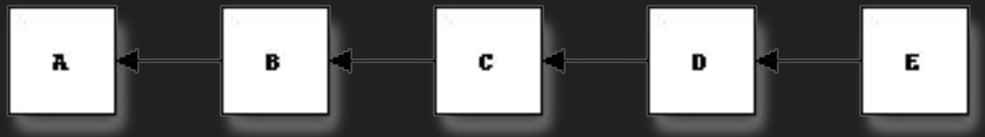
https://zuul-ci.org/docs/zuul/discussion/gating.html

Zuul’s gating - Testing in parallel

For example, if a reviewer approves five changes in rapid succession:

A, B, C, D, E

Zuul’s gating - Testing in parallel



Zuul’s gating - Testing in parallel

Jobs for A: merge change A, then test

Jobs for B: merge changes A and B, then test

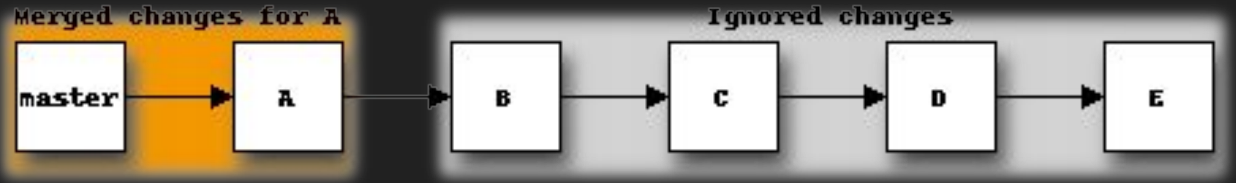
Jobs for C: merge changes A, B and C, then test

Jobs for D: merge changes A, B, C and D, then test

Jobs for E: merge changes A, B, C, D and E, then test

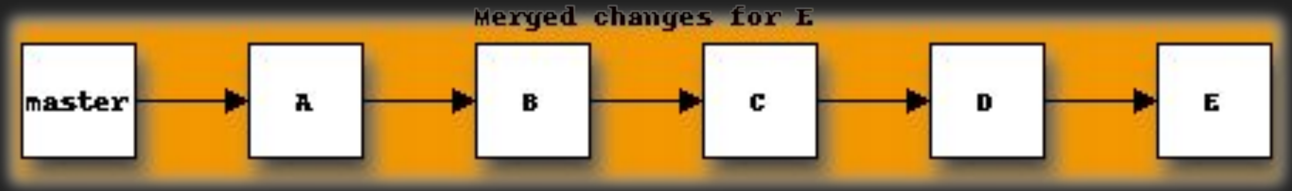
Zuul’s gating - Testing in parallel

Jobs triggered to tests A will only test A and ignore B, C, D



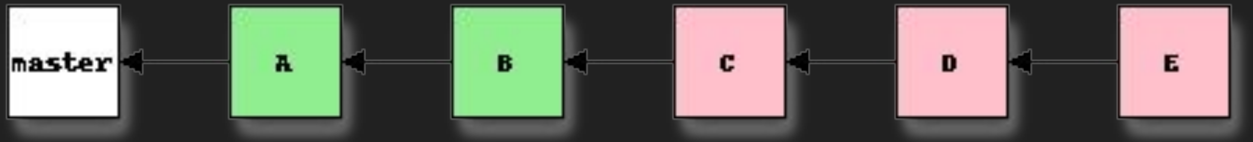
Zuul’s gating - Testing in parallel

The jobs for E would include the whole dependency chain: A, B, C, D, and E. E will be tested assuming A, B, C, and D passed:



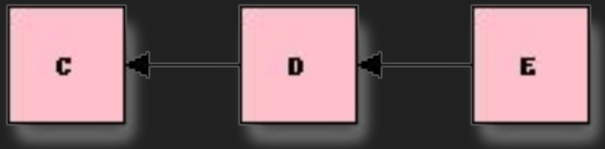
Zuul’s gating - Testing in parallel

If changes A and B pass tests (green), and C, D, and E fail (red):



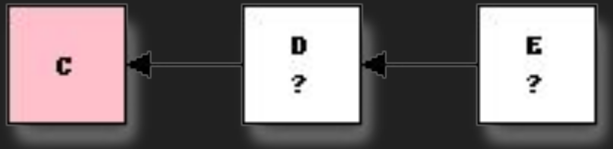
Zuul’s gating - Testing in parallel

Zuul will merge change A followed by change B, leaving this queue:



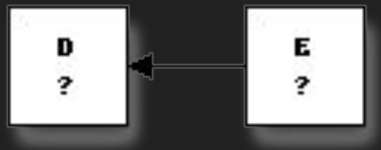
Zuul’s gating - Testing in parallel

Since D was dependent on C, it is not clear whether D’s failure is the result of a defect in D or C:



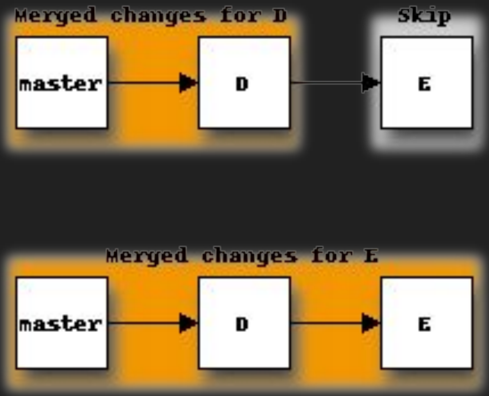
Zuul’s gating - Testing in parallel

Since C failed, Zuul will report its failure and drop C from the queue, keeping D and E:

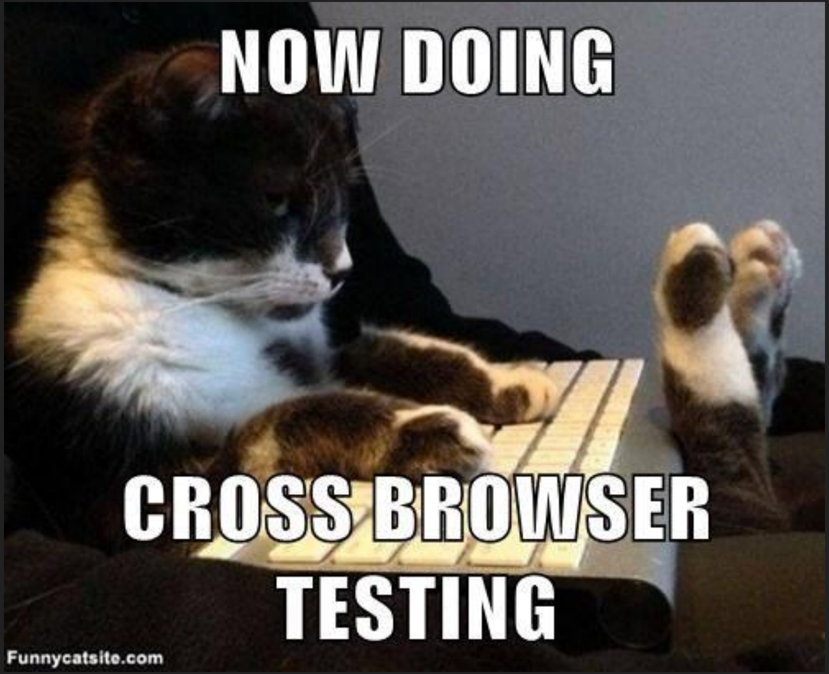


Zuul’s gating - Testing in parallel

This queue is the same as if two new changes had just arrived, so Zuul starts the process again testing D against the tip of the branch, and E against D:



Cross project testing



Cross project testing - Dependencies

Just put this in your commit message:

...

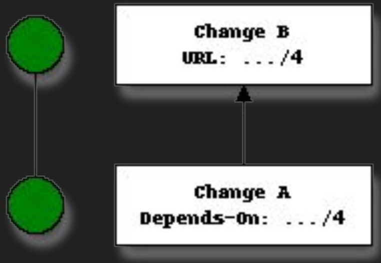
Depends-On: https://github.com/example/test/pull/4

...

Changes may depend on changes in any other project, even projects not on the same system (i.e., a Gerrit change may depend on a GitHub pull request).

Cross project testing - Dependent pipeline

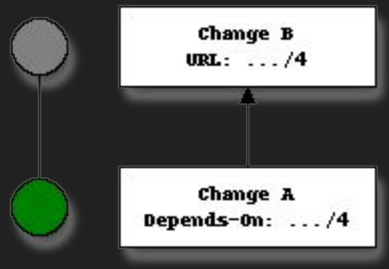
When Zuul sees changes with cross-project dependencies, it serializes them in the usual manner when enqueuing them into a pipeline. This means that if change **A depends on B**, then when they are added to a dependent pipeline, B will appear first and A will follow:



If tests for B fail, both B and A will be removed from the pipeline, and it will not be possible for A to merge until B does.

Cross project testing - Independent pipeline

Changes that land in the independent pipeline are tested independently from each other, meaning that the tests are not sharing a common workspace during testing.



This is to indicate that the grey changes are only there to establish dependencies.

About to catch feelings
And I'm over here like...

WHAT IF I TOLD YOU
THIS MEME IS OVER USED
imgflip.com

AUTOMATE
ALL THE THINGS!
imgflip.com

References

http://zuul.opendev.org/

http://zuul.opendev.org/t/openstack/status

https://zuul-ci.org/docs/zuul/

https://superuser.openstack.org/articles/zuul-case-study-the-openstack-foundation/

https://www.softwarefactory-project.io/tag/zuul-hands-on-series.html

https://www.youtube.com/watch?v=8LA7V8bc81k

https://julien.danjou.info/stop-merging-your-pull-request-manually/

https://zuul-ci.org/docs/zuul/discussion/gating.html

ZUUL

Thank You!