Testing and CI/CD

Introduction

All code changes are pushed to gerrit (https://gerrit.tungsten.io/). This is review system where 3 flags are used: **Code-Review** and **Approved** are set by users and they indicate that change is sufficient and approved; **Verified** flag is set by CI system.

For now Jenkins is used as a CI/CD system - https://tf-jenkins.progmaticlab.com/ (name will be changed soon). Jenkins has special plugin which listens events from gerrit and react.

Further name 'TF Jenkins' will be used for CI/CD and TF Jenkins is a common name of whole infra which serves:

- checking and gating for reviews from https://gerrit.tungsten.io
- running various checks for whole functionality (nightly)
- publishing artefacts to dockerhub

Whole infrastructure also has Nexus, plain mirrors, slaves for Jenkins, aquasec service, logs storage, grafana for monitoring.

Source code: https://github.com/tungstenfabric/tf-jenkins

Gerrit support

TF CI provides the ability to run jobs to check your patchsets sent for review.

Each project might be associated with several templates that are launched when a new patchset is pushed/approved/etc. You can see the list of projects and associated templates in the **config/main.yaml** (https://github.com/tungstenfabric/tf-jenkins/blob/master/config/main.yaml) file. Please find current and relevant description inside this file.

When new patchset is pushed to review then CI runs jobs from 'check' pipeline (pipeline is a definition for group of jobs which are run for different circumstances). If check failed due to unrelated reason then user can re-run by adding comment 'check' or 'recheck' to review.

When patchset has Verified +1 and approved then CI runs jobs from 'gate' pipeline. User can re-trigger it with comment 'gate'.

TF CI supports running of any defined job in review even if it's not declared in check/gate pipeline. To start checking a specific template, add a comment to your review with template name like **check template template-name**. For several templates at once comment is **check templates name1 name2**. Relevant template can also be found in configuration files.

It is possible to stop all jobs running for the current review. To do this, add a comment 'cancel' to your review.

TF CI supports '`Depends-On: I...`' parameter in commit message. It should contain Commit-Id of dependent review. In this case CI cherry-picks both change sets into sources tree and run jobs with merged content. It calls explicit dependency. This technique is applicable for any tree of dependent reviews but this tree must not have circular dependencies. Another option is implicit dependency - it's a dependency based on git tree. You can upload several commits for one repo at time - they will be shown in gerrit as a relation chain. And CI will take changes from parent commits by SHA for checking. If that review that current depends on gets new patchset then checks for current review will be cancelled. Due to some limitations cross-branched dependencies are not supported.

After checking TF CI posts a message with results along with timings, links to logs. Overall time is a time for whole checking. And stream time is a summarised time from all jobs in this stream. Sometimes stream time can be bigger that checks time due to parallel runs.

Infra details

Whole infra deployment is stored as ansible scripts (https://github.com/tungstenfabric/tf-jenkins/tree/master/setup).

Setup details can be found in README.md. In general user prepares several VM (details are below), prepares inventory and run ansible playbooks.

Jenkins master

Ubuntu 18.04 based VM, 4 CPU, 16 Gb RAM, 500Gb for root volume.

Jenkins master is deployed as a docker container. Most further configuration (user, plugins, ...) is applied inside Jenkins itself and via Configiration-As-A-Code plugin.

Architecture of CI code requires Jenkins slave with mirrors in the same private network as slave.

Jenkins master doesn't process any jobs except its configuration. All jobs are run

Jenkins slave

Ubuntu 18.04 based VM, 8 CPU, 32 Gb RAM, 500Gb for root volume.

To be able to create/remove workers/networks infra must have individual slave in each region.

For now we have only one slave in one region of vexxhost. But we checked that code can work with different slave on AWS.

Slave doesn't run any TF product related code - just infra code. Slave is a jumphost in a specific provider/region. It creates worker VM-s and run all code there.

Additional sources:

- Document explaining works on new CI resides here: https://docs.google.com/document/d/1CMwfU_fwgf8Mlae-1z13XvhWBi3p79PKOMSALf2MGf0/edit?usp=sharing
- Relevant TF Jenkins description - https://github.com/tungstenfabric/tf-jenkins#readme
- https://github.com/tungstenfabric/tf-dev-env This project is used for building of TF dockers' images.
- https://github.com/tungstenfabric/tf-devstack This project is used for various deployment scenarios.
- https://github.com/tungstenfabric/tf-dev-test This project is used for running concrete test suite - tf-test (also called as sanity) and tf-deployment-test (see below).
- https://github.com/tungstenfabric/tf-deployment-test This project contains various deployment test like ZIU, etc.