Amber Releasing Process

Locate the git commit point ready for release.

1. Find the commit ID of the last change to be included in the Amber release
   a. If this is the current head of master branch:
      i.
```
# git clone the repo, cd to the repo directory, then:
COMMITID=$(git log -1 |head -1 |cut -f2 -d' ')
echo $COMMITID
```
   b. Otherwise use "git log" to locate the ID of the commit

Committer to create **Amber** branch

1. Use Gerrit UI to create the branch:
   a. Login to Gerrit, locate the repo, click on repo name (left column);
   b. On the left side panel, select menu choice "Branches";
   c. Click on the "Click New" link at the top-right corner;
   d. On the pop-up, enter **Amber** in **branch name** box, and the commit ID recorded from step 1 in **Initial Revision**box. Click on "Create" link.

Releasing Binary Artifacts

Overview

At the time of releasing, binary artifacts built from O-RAN SC source code such as docker container images are "released" to public repositories (e.g. docker.io) with Linux Foundation signature. Depending on the type of a binary artifact, it may be released by a self-service process or a manual process.

The self-service process applies to docker images, Maven artifacts, and Pypi packages. The contributor making the request will need to edit an artifact specification file describing which artifact to release. Then a predefined Jenkins Job will be triggered to complete the releasing process.

Detailed documentation can be found at: https://docs.releng.linuxfoundation.org/projects/global-jjb/en/latest/jjb/lf-release-jobs.html

For artifact of a different type, the releasing is done manually via a Linux Foundation helpdesk ticket.

Preparation

Jenkins Release Jobs

.To check whether self-service release jobs have already been defined for a specific repo, go to http://jenkins.o-ran-sc.org and among the Jenkins jobs listed under the tab for the repo look for two jobs named {{ repo-name }}-release-verify and {{ repo-name }}-release-merge. For example **it-dep-release-verify** and **it-dep-release-merge** for the it/dep repo.

The first job (release-verify job) is to be triggered when an artifact release yaml file for the repo is submitted into Gerrit for review, similar to how regular verify jobs being triggered when code change submission happens. The release-verify job verifies all parameters specified in the artifact release yaml file are valid and the artifact to be released actually exits in the snapshot/staging repository.

The second job (release-merge job) is to be triggered when an artifact release yaml file for the repo is accepted by a repo committer and merged, similar to how regular merge jobs being triggered when code change submission being merged. The release-merge job would complete the pulling of the artifact from its source repo (snapshot or staging) then pushing onto its target (release) repo.

Locating the Artifact to be Released (and regenerate if necessary)

Because releasing an artifact generally involves fetching the artifact from its source repo (a snapshot or staging repo) and then publishing it onto a release repo, the artifact must already exist on the source repo. For example if a docker image is to be released, such an image must exist on O-RAN SC's staging Nexus3 repo.

Here are the steps for check for the existence of a docker image:

1. Browse to http://nexus3.o-ran-sc.org, do not login.
2. Select "**Browse**" from the left menu, then **docker.staging** from the repo list.
3. From all the images listed, find and expand the **o-ran-sc** group to look for the image to be release.
4. If such image exists, expand the image to make sure the tag to be released is also listed. Write down the image name and tag. We will need such info later.

If the to be released image (or tag) is not there, there can be several reasons:

1. If such an image existed before, the disappearance is likely due to the fact that the image has expired. The docker.staging repo only keeps images for up to 15 days. Older images will be automatically removed.
2. Because we are switching to using the o-ran-sc or o-ran-scp as dev org prefix recently, there has been no code changes after the switching, hence building a docker image under new name (with prefix) has never been triggered.
3. This image has never been generated successfully.

For the first two cases, we need to regenerate the image. This can usually be done in one of two ways.

1. If there is no newer changes merged into this repo after the code submission that triggered the building of the to-be-release docker container image, we need to use the Gerrit UI to find this already merged code change, and click on the "reply" button. In the message box, type the magic word "**remerge**". This key word will trigger the merge job and the merge job will rebuild the docker image.
    a. One way to locate the last Gerrit submission that triggered the merge build is to go to jenkins.o-ran-sc.org, locate the merge job that built the docker image among the list of Jenkins jobs for this repo on the repo tab, click on the job to bring out its build history on the left side of the page, then find the last build of this job from the left panel. The build's detail (by clicking on the build number) contains a link to the Gerrit submission that triggered the build. This is the Gerrit change that we want to reply to with the magic word.
    b. If the docker container image of interest is one of several images built from this repo, make sure looking for the code change that triggered the building of this specific docker container image. This needs to be done very carefully because some change may trigger multiple docker container image builds, and different changes may cover different sets of container image builds. One rather conservation way of remerging in this kind of repos is to not only "remerge" the last change that triggered the image build for the container of interest, but also "remerge" all changes after this change in the order that such changes are originally merged into Gerrit.
2. If there are newer code changes merged after the change that triggered the building of the to-be-release docker container image, "remerge" such an out-of-order change may cause unexpected results. In this case an alternative method can be used instead. That is, to make a "no-op" code change to a file that will trigger the rebuilding of the to-be-release docker container image. For example, adding or removing some comment text to the Dockerfile for the container.

If it is the 3rd case, then we need to investigate why the image has never been generated by finding the merge job that would generate the image and look into its logs.

Releasing Artifact

Self-service releasing

When the preparation for releasing an artifact is complete, we have the release jobs and the artifact exists, it is time to edit the release file. This is the file placed under the release or .releases directory repo root that describes the artifact to be released. For example, below is the release file for o-ran-sc/it-dep-secret container:

releases/container-release-it-dep-secret.yaml

```
1  ---
2  distribution_type: container
3  container_release_tag: 0.0.2
4  container_pull_registry: nexus3.o-ran-sc.org:10004
5  container_push_registry: nexus3.o-ran-sc.org:10002
6  project: it/dep
7  ref: b1cc45af5be31e34bd4cc04554a064ed0143b711
8  containers:
9      - name: it-dep-secret
10       version: 0.0.2
```

Note that values on lines 3, 6, 7, 9, and 10 need to be adapted for the repo and container to be released.

Line 3: the tag to be used for release container;

Line 6: which repo;

Line 7: the $COMMITID that we identified earlier;

Line 9: name of the to-be-released docker image on its source repo (pull registry) (no need to have the o-ran-sc/ prefix here);

Line 10: version tag of the to-be-released docker image on its source repo (pull registry).

Detailed documentation can be found at: https://docs.releng.linuxfoundation.org/projects/global-jjb/en/latest/jjb/lf-release-jobs.html. In particular, finding the section about "Release Files" for the type of artifact of interest.

Although it is okay to keep multiple of such release yaml files under **releases** or **.releases**, each commit of changes under this directory can ONLY CHANGE ONE FILE.

Release File for Pypi Package

To be added

Release File Maven package

To be added

Manual (Linux Foundation) releasing

For types of artifact that are not listed in the self-service releasing process, they can be released by making request to Linux Foundation Helpdesk by filing a ticket at: https://jira.linuxfoundation.org/servicedesk/customer/portal/2/create/107.