# cnf-certification-one-shot-automation

This repo purpose is to automate the partners CNF certification that have been done the sanity checking for following steps: - Using Preflight script or manual to perform sanity checking and all these test cases that preflight is testing and it Must be passed! and its verdict passed also.  
**Main Test Case must pass as follow:**

======================================================  
Image Name Test Case Status   
------------------------------------------------------  
universal-smf-ava HasLicense PASSED   
universal-smf-ava HasUniqueTag PASSED   
universal-smf-ava LayerCountAcceptable PASSED   
universal-smf-ava HasNoProhibitedPackages PASSED   
universal-smf-ava HasRequiredLabel PASSED   
universal-smf-ava RunAsNonRoot PASSED   
universal-smf-ava HasModifiedFiles PASSED   
universal-smf-ava BasedOnUbi PASSED   
Verdict: PASSED

For details of sanity check script using preflight scan, it can be found [here](https://github.com/ansvu/quick_scan_container_images_online_offline)

Preflight All Test Cases

[Container Policy]: invoked on container images  
- HasLicense  
- HasUniqueTag  
- LayerCountAcceptable  
- HasNoProhibitedPackages  
- HasRequiredLabel  
- RunAsNonRoot  
- HasModifiedFiles  
- BasedOnUbi  
  
[Container Root Exception Policy]: automatically applied for container images if preflight determines a root exception flag has been added to your Red Hat Connect project  
- HasLicense  
- HasUniqueTag  
- LayerCountAcceptable  
- HasNoProhibitedPackages  
- HasRequiredLabel  
- HasModifiedFiles  
- BasedOnUbi  
  
[Container Scratch Exception Policy]: automatically applied for container checks if preflight determines a scratch exception flag has been added to your Red Hat Connect project  
- HasLicense  
- HasUniqueTag  
- LayerCountAcceptable  
- HasRequiredLabel  
- RunAsNonRoot

* Using chart-verifier either use dci to run or manually to generate report.yaml
* Chart Verifier Test Checking

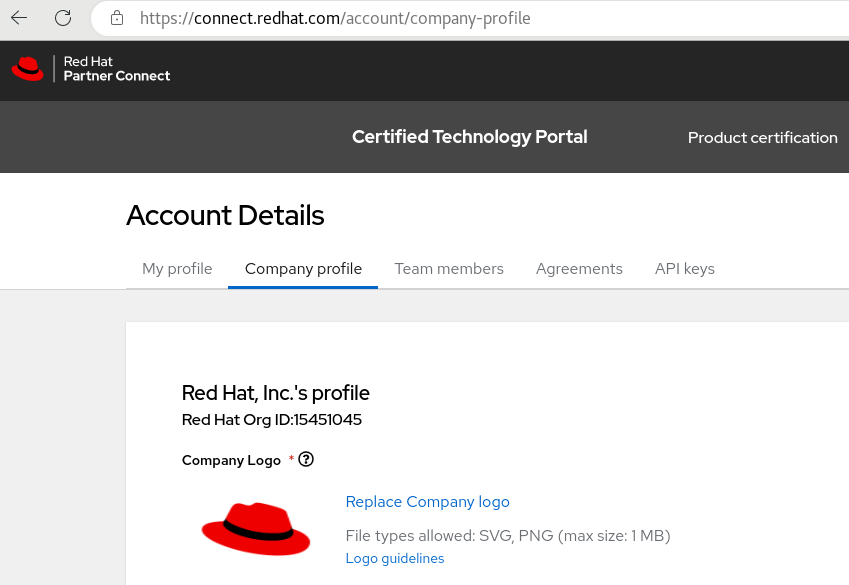
- check: v1.1/images-are-certified  
 type: Mandatory  
 outcome: PASS  
 reason: 'Image is Red Hat certified : registry.access.redhat.com/ubi8/nginx-118:1-42'  
 - check: v1.0/signature-is-valid  
 type: Mandatory  
 outcome: SKIPPED  
 reason: 'Chart is not signed : Signature verification not required'  
 - check: v1.0/contains-test  
 type: Mandatory  
 outcome: PASS  
 reason: Chart test files exist  
 - check: v1.1/has-kubeversion  
 type: Mandatory  
 outcome: PASS  
 reason: Kubernetes version specified  
 - check: v1.0/chart-testing  
 type: Mandatory  
 outcome: PASS  
 reason: Chart tests have passed  
 - check: v1.0/has-readme  
 type: Mandatory  
 outcome: PASS  
 reason: Chart has a README  
 - check: v1.0/contains-values-schema  
 type: Mandatory  
 outcome: PASS  
 reason: Values schema file exist  
 - check: v1.0/is-helm-v3  
 type: Mandatory  
 outcome: PASS  
 reason: API version is V2, used in Helm 3  
 - check: v1.0/required-annotations-present  
 type: Mandatory  
 outcome: PASS  
 reason: All required annotations present  
 - check: v1.0/contains-values  
 type: Mandatory  
 outcome: PASS  
 reason: Values file exist  
 - check: v1.0/helm-lint  
 type: Mandatory  
 outcome: PASS  
 reason: Helm lint successful  
 - check: v1.0/not-contain-csi-objects  
 type: Mandatory  
 outcome: PASS  
 reason: CSI objects do not exist  
 - check: v1.0/not-contains-crds  
 type: Mandatory  
 outcome: PASS  
 reason: Chart does not contain CRDs

For more details of chart-verifier test cases, please click [chart-verifier-test-cases-details](https://github.com/redhat-certification/chart-verifier/blob/main/docs/helm-chart-checks.md)

## Prerequisites

* [DCI](https://blog.distributed-ci.io/introduction-to-the-red-hat-distributed-ci.html) is Red Hat Distributed CI, written in Ansible.
* [Preflight](https://github.com/redhat-openshift-ecosystem/openshift-preflight) is a command-line interface for validating if OpenShift operator bundles and containers meet minimum requirements for Red Hat OpenShift Certification.
* Set up a jumphost with internet access, install the dci-openshift-appagent, detailed guide can be found in this link [dci-openshift-app-agent-install](https://doc.distributed-ci.io/dci-openshift-app-agent/)
* Install dci-pipeline  
  [DCI-PIPELINE](https://github.com/redhat-cip/dci-pipeline)
* It is recommended consistently check latest version of the DCI app agent package, and upgrade to latest version before using.

$ sudo dnf upgrade --refresh --repo dci -y

* DCI Control server credential [create remote-ci credentials](https://www.distributed-ci.io/remotecis) Click on credentials.yaml then save contents to a path where su - dci-openshift-app-agent user is or you can save any path you legit. This file will be used by the dci-pipeline parameter dci\_credentials: /etc/dci-pipeline/dci\_credentials.yml
* Again container images sanity check with preflight and CNF helmchart report.yaml must be green and passed all the tests before one-shot-cnf-certification-automation to work
* Create github token to be used for helmchart PR  
  github\_token\_path: "/opt/cache/dcicertbot-token.txt"  
  [how-to-create-github-token](https://github.com/redhatci/ansible-collection-redhatci-ocp/blob/main/roles/create_certification_project/README.md#github-token)
* Create Pyxis Apikey  
  A token to access specific partner Pyxis catalog data using REST API. [Create Pyxis API Key](https://connect.redhat.com/account/api-keys)
* Kubeconfig  
  A kubeconfig that can access the OCP cluster
* Product-listing ID  
  Before a container or helm chart/operator can be publicly listed into RedHat catalog, a Product-Listing must be created, it only need to create once according to CNF type. Follow this link to [Create-Product-Listing](https://connect.redhat.com/manage/products)
* Organization ID  
  Mandatory when using create\_container\_project. Company ID will be used for the verification of container certification project Organization-ID Company-Profile. 

## Prepare DCI Pipeline Container E2E Certification Setting

This DCI Pipeline Setting will include create container certification project, update mandatory parameters, scan container+submit the results and attach the product-listing to newly created project.

**dci-pipeline-settings-container-e2e-certification.yaml:**

---  
- name: Certification-Container-E2E  
 stage: certifycontainer  
 ansible\_playbook: /usr/share/dci-openshift-app-agent/dci-openshift-app-agent.yml  
 ansible\_cfg: /usr/share/dci-openshift-app-agent/ansible.cfg  
 ansible\_inventory: /etc/dci-openshift-app-agent/hosts.yml  
 dci\_credentials: /etc/dci-pipeline/dci\_credentials.yml  
 ansible\_extravars:  
 dci\_name: create,update,attach,scan container certification project using DCI Pipeline  
 dci\_configuration: Using DCI Pipeline to create container certification project   
 dci\_tags: ["debug", "certification-container"]  
 dci\_cache\_dir: /var/lib/dci-pipeline  
 dci\_config\_dirs: [/etc/dci-openshift-agent]  
 dci\_workarounds: []  
 partner\_creds: "/var/lib/dci-openshift-app-agent/demo-auth.json"  
 check\_for\_existing\_projects: true  
 organization\_id: 1111111  
 do\_must\_gather: false  
 preflight\_run\_health\_check: false  
 check\_workload\_api: false  
 page\_size: 400 #lab has a lot of archive projects so it needs to define page\_size higher than partner account  
 #preflight\_container\_wait: 0 # value in minute if waiting for backend to publish your images to catalog due to the delay.  
 pyxis\_apikey\_path: "/var/lib/dci-openshift-app-agent/demo-pyxis-apikey.txt"  
 preflight\_containers\_to\_certify:  
 - container\_image: "quay.io/avu0/auto-publish-ubi8-nginx-demo7:v120"  
 create\_container\_project: true  
 short\_description: "I am doing a full-automation e2e auto-publish for following image auto-publish-ubi8-nginx-demo7"  
   
 cert\_settings:  
 auto\_publish: false  
 build\_categories: "Standalone image"  
 registry\_override\_instruct: "<p>This is an instruction how to get the image link.</p>"  
 email\_address: "me@redhat.com"  
 application\_categories: "Networking"  
 os\_content\_type: "Red Hat Universal Base Image (UBI)"  
 privileged: false  
 release\_category: "Generally Available"  
 repository\_description: "This is a test for Demo how to automate to create project,SCAN and update settings"  
   
 cert\_listings:  
 published: false  
 type: "container stack"  
 pyxis\_product\_list\_identifier: "222222222222222222222222"  
 attach\_product\_listing: true  
  
 components: []  
 inputs:  
 kubeconfig: /var/lib/dci-openshift-app-agent/kubeconfig

## Prepare DCI Pipeline Helmchart E2E Certification Settings

This DCI Pipeline Setting will include create helmchart certification project, update mandatory parameters, and attach the product-listing to newly created project.

**dci-pipeline-settings-helmchart-e2e-cert-chartverifier-pr.yaml:**

---  
- name: Certification-Helmchart-PR  
 stage: helmchartcreation  
 prev\_stages: certifycontainer  
 ansible\_playbook: /usr/share/dci-openshift-app-agent/dci-openshift-app-agent.yml  
 ansible\_cfg: /usr/share/dci-openshift-app-agent/ansible.cfg  
 ansible\_inventory: /etc/dci-openshift-app-agent/hosts.yml  
 dci\_credentials: /etc/dci-pipeline/dci\_credentials.yml  
 topic: OCP-4.13  
 ansible\_extravars:  
 dci\_name: create,update,attach,helmchart certification project using DCI Pipeline  
 dci\_configuration: Using DCI Pipeline to create helmchart certification project and PR   
 dci\_tags: ["debug", "certification-helmcart"]  
 dci\_cache\_dir: /var/lib/dci-pipeline  
 dci\_config\_dirs: [/etc/dci-openshift-agent]  
 dci\_workarounds: []  
 partner\_creds: "/var/lib/dci-openshift-app-agent/demo-auth.json"  
 check\_for\_existing\_projects: true  
 organization\_id: 11111111  
 do\_must\_gather: false  
 check\_workload\_api: false  
 page\_size: 400  
 pyxis\_apikey\_path: "/var/lib/dci-openshift-app-agent/demo-pyxis-apikey.txt"  
 helmchart\_to\_certify:  
 - repository: "https://github.com/ansvu/finalchart"  
 short\_description: "This is a short description finalchart"  
 chart\_name: "finalchart"  
 create\_helmchart\_project: true  
  
 cert\_settings:  
 email\_address: "avu@redhat.com"  
 distribution\_method: "undistributed"  
 github\_usernames: "ansvu"  
 application\_categories: "Networking"  
 long\_description: "This is a long description about this final chart"  
 distribution\_instructions: "You must be present to get this helm-chart!"  
  
 cert\_listings:  
 attach\_product\_listing: true  
 published: false  
 type: "container stack"  
 pyxis\_product\_list\_identifier: "2222222222222222222222222"  
  
 components: []  
 inputs:  
 kubeconfig: "/var/lib/dci-openshift-app-agent/kubeconfig"  
...  
  
- name: Certification-Helmchart-PR  
 stage: helmchartpr  
 prev\_stages: helmchartcreation  
 ansible\_playbook: /usr/share/dci-openshift-app-agent/dci-openshift-app-agent.yml  
 ansible\_cfg: /usr/share/dci-openshift-app-agent/ansible.cfg  
 ansible\_inventory: /etc/dci-openshift-app-agent/hosts.yml  
 dci\_credentials: /etc/dci-pipeline/dci\_credentials.yml  
 topic: OCP-4.13  
 ansible\_extravars:  
 dci\_name: test helmchart PR on github using DCI Pipeline  
 dci\_configuration: Testing helmpchart PR with pipeline   
 dci\_tags: ["debug", "certification-helmchart-pr"]  
 dci\_cache\_dir: /var/lib/dci-pipeline  
 dci\_config\_dirs: [/etc/dci-openshift-agent]  
 dci\_workarounds: []  
 do\_must\_gather: false  
 check\_workload\_api: false  
 do\_chart\_verifier: true  
 partner\_name: "redhat-arkady-test"  
 partner\_email: "redhat-arkady-test@redhat.com"  
 github\_token\_path: "/var/lib/dci-openshift-app-agent/github-token.txt"  
 dci\_charts:  
 - name: finalchart  
 chart\_file: https://ansvu.github.io/finalchart/finalchart-0.1.4.tgz  
 deploy\_chart: true  
 create\_pr: true  
  
 components: []  
 inputs:  
 kubeconfig: /var/lib/dci-openshift-app-agent/kubeconfig  
...

## Prepare DCI Pipeline Openshift-cnf E2E Certification Setting

This DCI Pipeline Setting will include create Openshift-cnf certification project and attach the product-listing to newly created project. Once this vendor validated project is created, then the support-case also created automatically. To finalize the Vendor-valiated certification, it needs the Backend or someone who access to saleforce BE to approve it.

**dci-pipeline-settings-openshift-cnf-e2e-cert.yaml:**

---  
- name: Openshift-CNF-Project-Creation Using Pipeline  
 stage: openshiftcnf  
 prev\_stages: helmchartpr  
 ansible\_playbook: /usr/share/dci-openshift-app-agent/dci-openshift-app-agent.yml  
 ansible\_cfg: /usr/share/dci-openshift-app-agent/ansible.cfg  
 ansible\_inventory: /etc/dci-openshift-app-agent/hosts.yml  
 dci\_credentials: /etc/dci-pipeline/dci\_credentials.yml  
 topic: OCP-4.13  
 ansible\_extravars:  
 dci\_name: create openshift-cnf project using DCI Pipeline  
 dci\_configuration: Using DCI Pipeline to create openshift-cnf project   
 dci\_tags: ["debug", "openshift-cnf"]  
 dci\_cache\_dir: /var/lib/dci-pipeline  
 dci\_config\_dirs: [/etc/dci-openshift-agent]  
 dci\_workarounds: []  
 partner\_creds: "/var/lib/dci-openshift-app-agent/demo-auth.json"  
 check\_for\_existing\_projects: true  
 organization\_id: 11111111  
 do\_must\_gather: false  
 check\_workload\_api: false  
 page\_size: 400  
 pyxis\_apikey\_path: "/var/lib/dci-openshift-app-agent/demo-pyxis-apikey.txt"  
 cnf\_to\_certify:  
 - cnf\_name: "DemoCNF23.8 on OCP4.14"  
 create\_cnf\_project: true  
  
 cert\_listings:  
 email\_address: "me@redhat.com"  
 published: true  
 type: "container stack"  
 pyxis\_product\_list\_identifier: "22222222222222222222222"  
 attach\_product\_listing: true  
  
 components: []  
 inputs:  
 kubeconfig: /var/lib/dci-openshift-app-agent/kubeconfig

## How To Run One Shot CNF Certification Automation

$ dci-pipeline dci-pipeline-settings-helmchart-e2e-cert-chartverifier-pr.yaml dci-pipeline-settings-helmchart-e2e-cert-chartverifier-pr.yaml dci-pipeline-settings-openshift-cnf-e2e-cert.yaml