LF NETWORKING

LFN Developer & Testing Forum

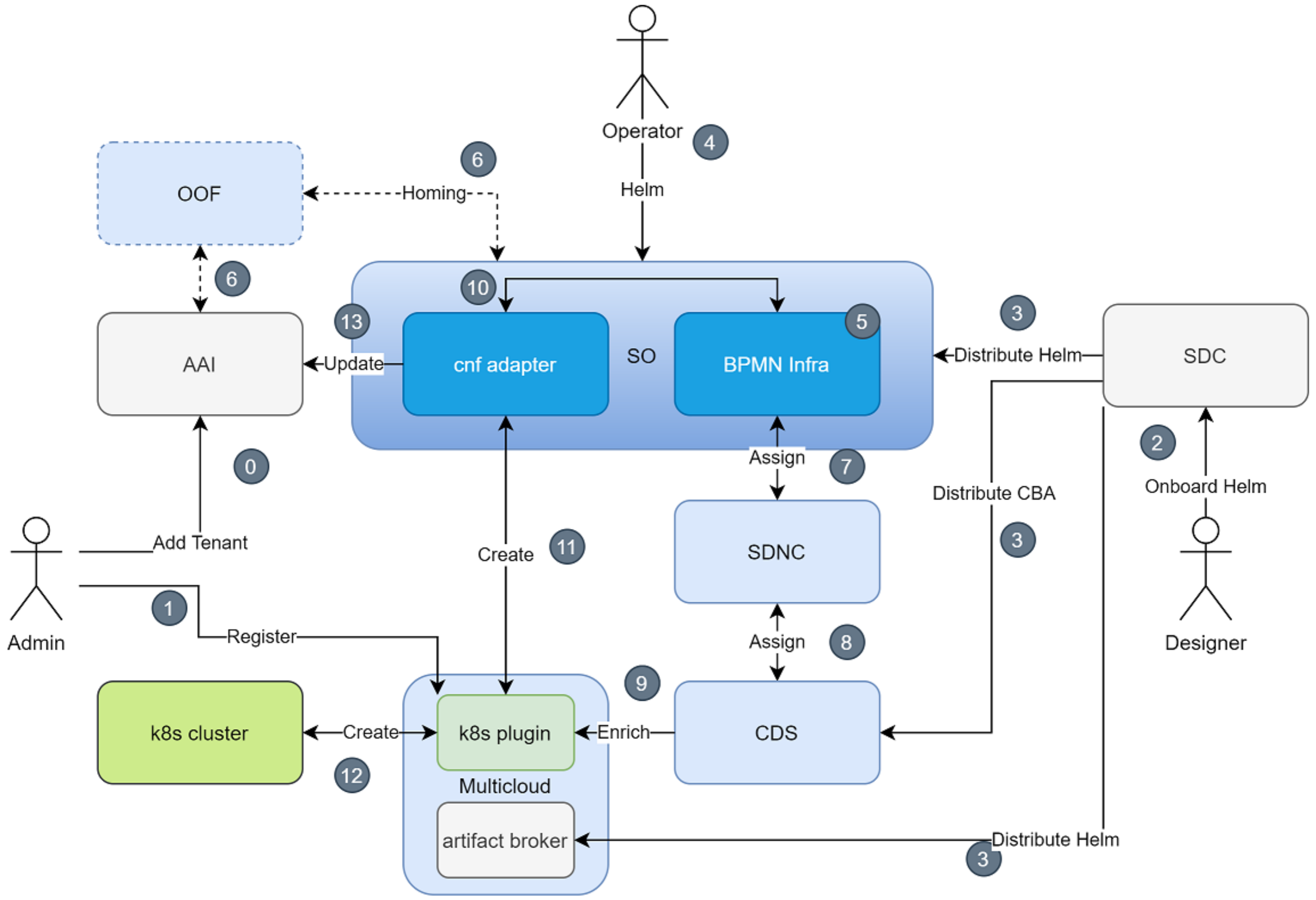
Native CNF Orchestration with Day2 support

vFW CNF Use Case

Lukasz Rajewski (Orange)
Konrad Banka (Samsung)
Seshu Kumar (Huawei)

08.06.2021

Native CNF Orchestration Path



REQ-341
Guilin

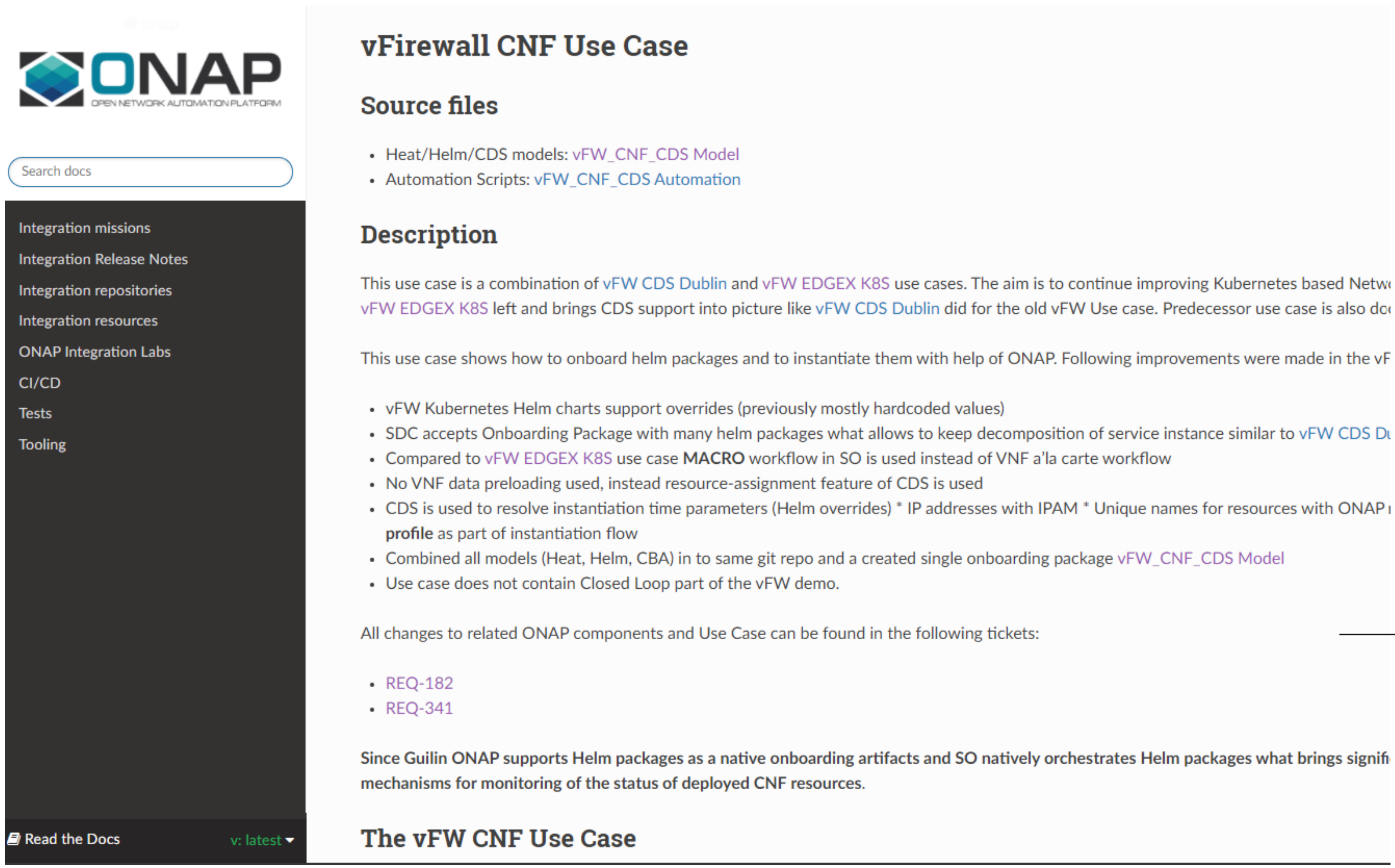
REQ-458
Honolul

REQ-627
Istanbul

Day 0/1/2 for Helm CNF

vFW CNF Use Case

LF NETWORKING
LFN Developer & Testing Forum



vFirewall CNF Use Case

Source files

- Heat/Helm/CDS models: vFW_CNF_CDS Model
- Automation Scripts: vFW_CNF_CDS Automation

Description

This use case is a combination of vFW CDS Dublin and vFW EDGEX K8S use cases. The aim is to continue improving Kubernetes based Netw
vFW EDGEX K8S left and brings CDS support into picture like vFW CDS Dublin did for the old vFW Use case. Predecessor use case is also do

This use case shows how to onboard helm packages and to instantiate them with help of ONAP. Following improvements were made in the vF

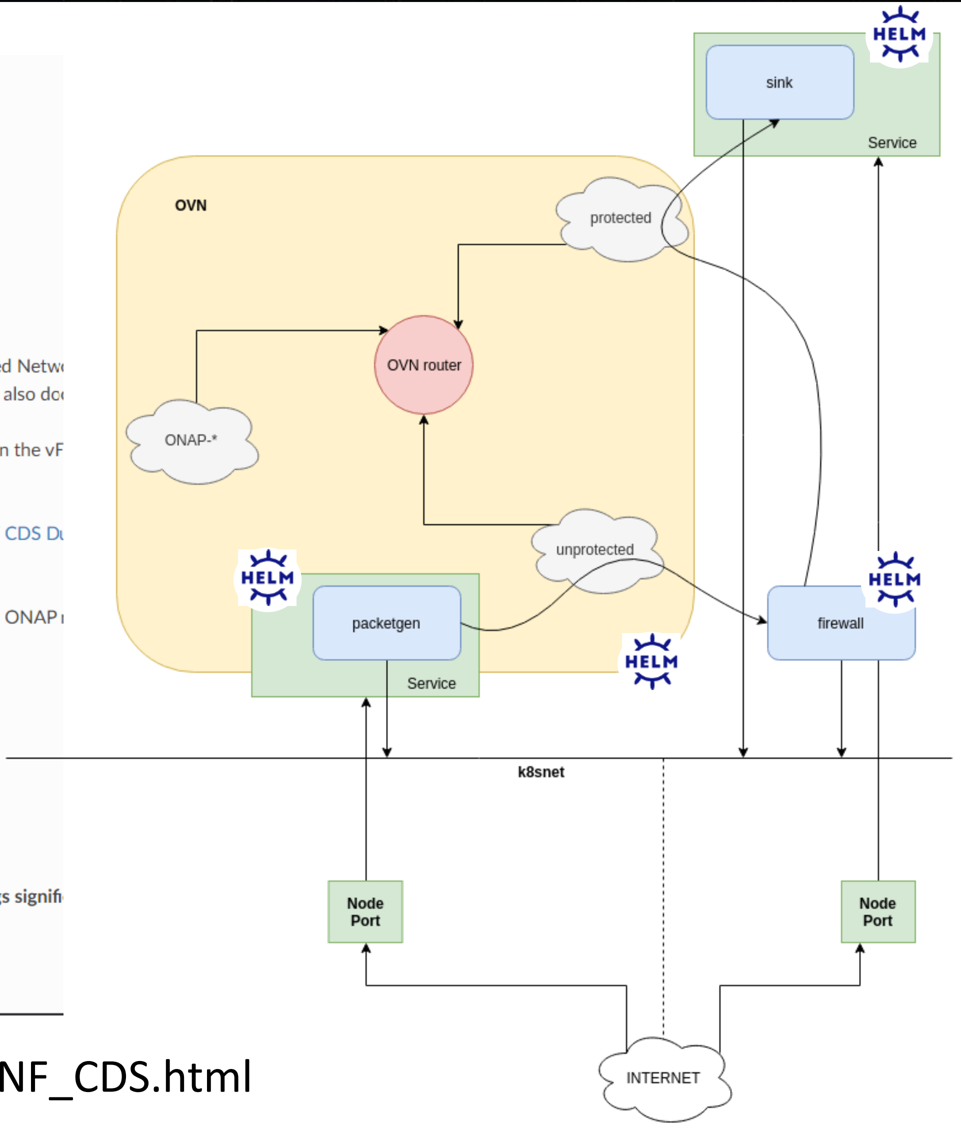
- vFW Kubernetes Helm charts support overrides (previously mostly hardcoded values)
- SDC accepts Onboarding Package with many helm packages what allows to keep decomposition of service instance similar to vFW CDS Du
- Compared to vFW EDGEX K8S use case **MACRO** workflow in SO is used instead of VNF a'la carte workflow
- No VNF data preloading used, instead resource-assignment feature of CDS is used
- CDS is used to resolve instantiation time parameters (Helm overrides) * IP addresses with IPAM * Unique names for resources with ONAP **profile** as part of instantiation flow
- Combined all models (Heat, Helm, CBA) in to same git repo and a created single onboarding package vFW_CNF_CDS Model
- Use case does not contain Closed Loop part of the vFW demo.

All changes to related ONAP components and Use Case can be found in the following tickets:

- REQ-182
- REQ-341

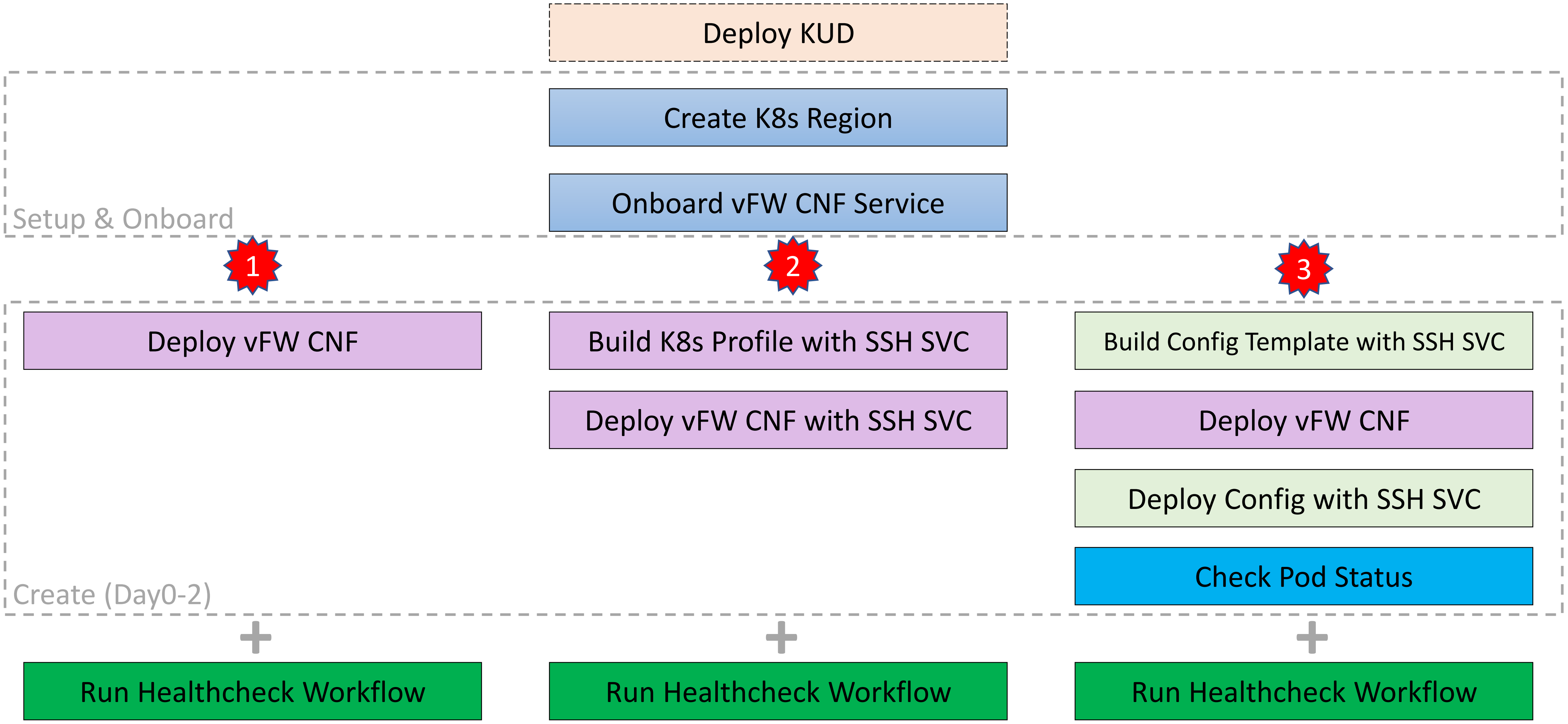
Since Guilin ONAP supports Helm packages as a native onboarding artifacts and SO natively orchestrates Helm packages what brings signifi mechanisms for monitoring of the status of deployed CNF resources.

The vFW CNF Use Case



https://docs.onap.org/projects/onap-integration/en/honolulu/docs_vFW_CNF_CDS.html

Scenarios


Deploy KUD
Create K8s Region
Onboard vFW CNF Service
Setup & Onboard
1
2
3
Deploy vFW CNF
Build K8s Profile with SSH SVC
Deploy vFW CNF with SSH SVC
Build Config Template with SSH SVC
Deploy vFW CNF
Deploy Config with SSH SVC
Check Pod Status
Create (Day0-2)
+
+
+
Run Healthcheck Workflow
Run Healthcheck Workflow
Run Healthcheck Workflow

LF NETWORKING

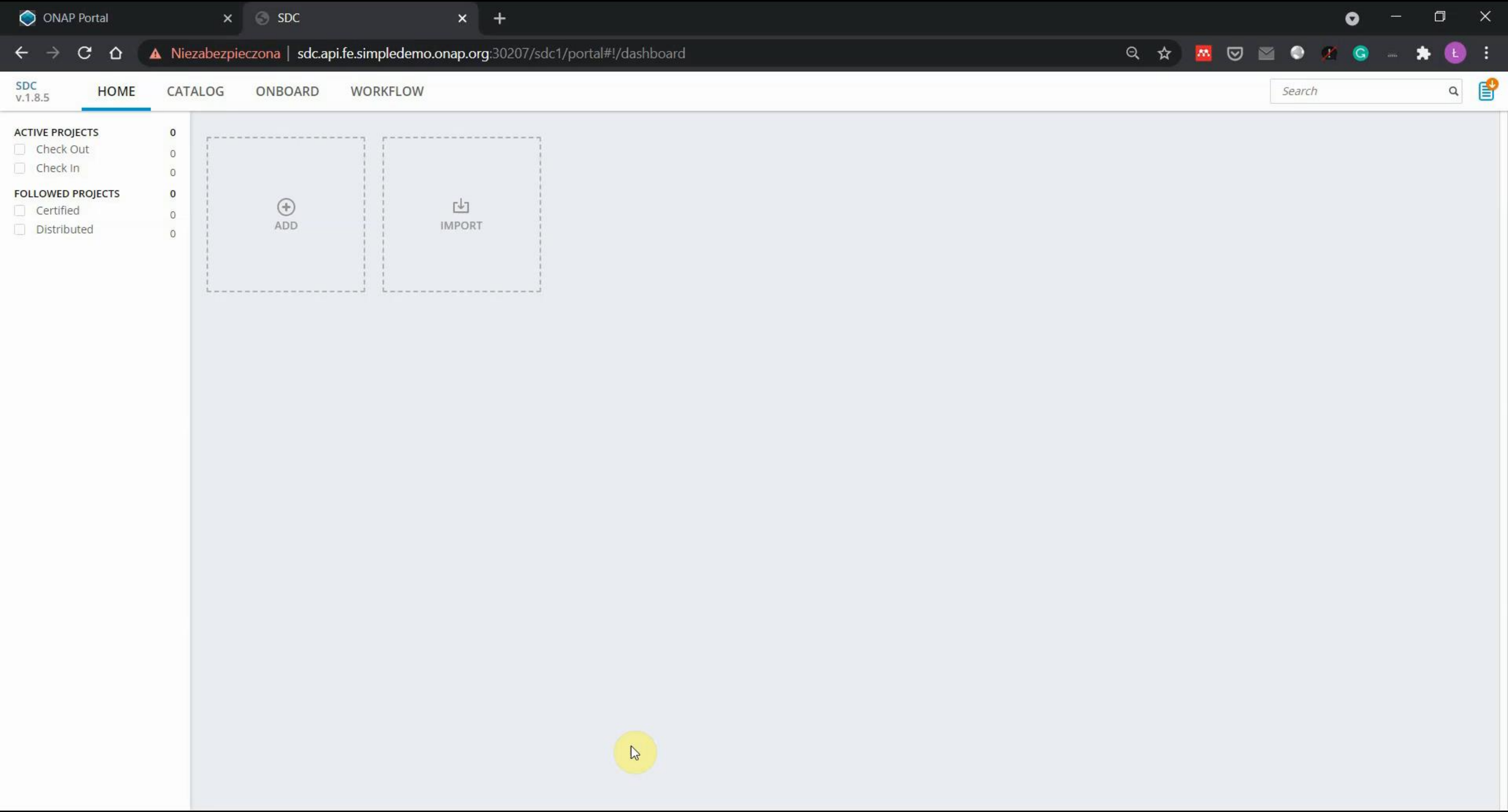
LFN Developer & Testing Forum

Setup

LF NETWORKING

LFN Developer & Testing Forum

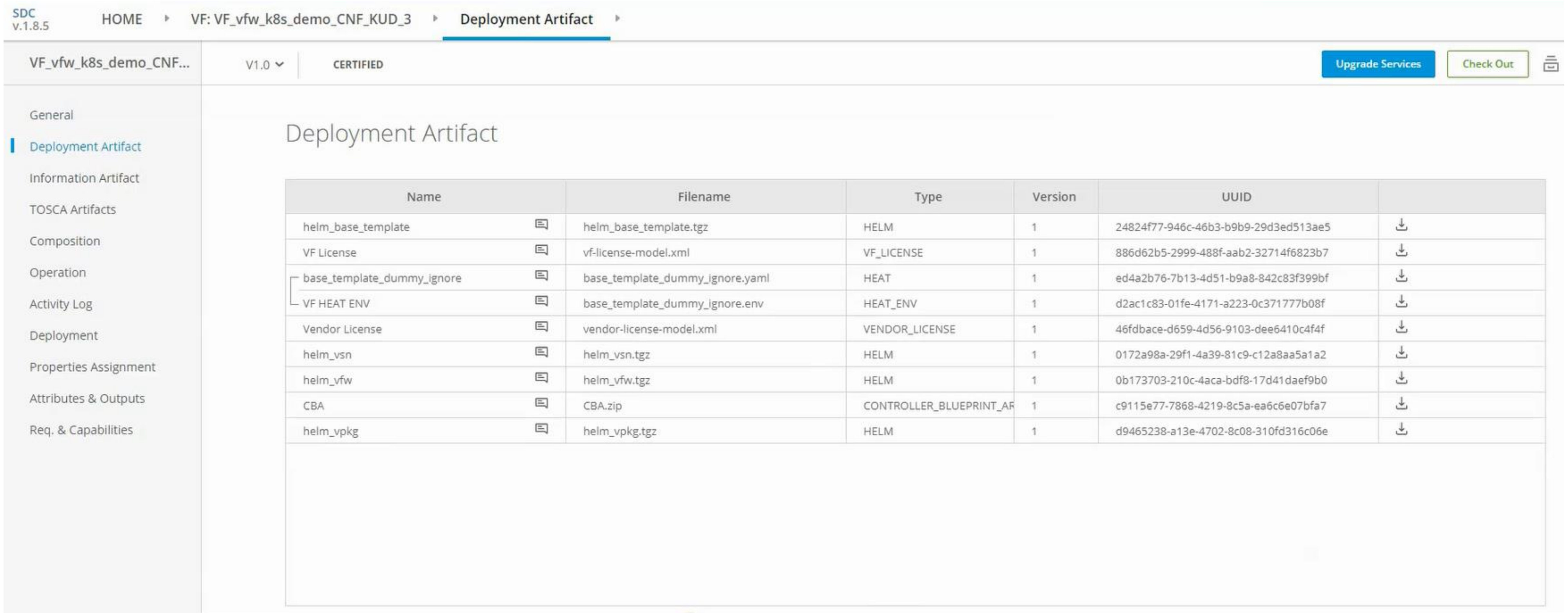
Onboard



ACTIVE PROJECTS	0
Check Out	0
Check In	0
FOLLOWED PROJECTS	0
Certified	0
Distributed	0

Onboarding Result

LF NETWORKING
LFN Developer & Testing Forum



Deployment Artifact

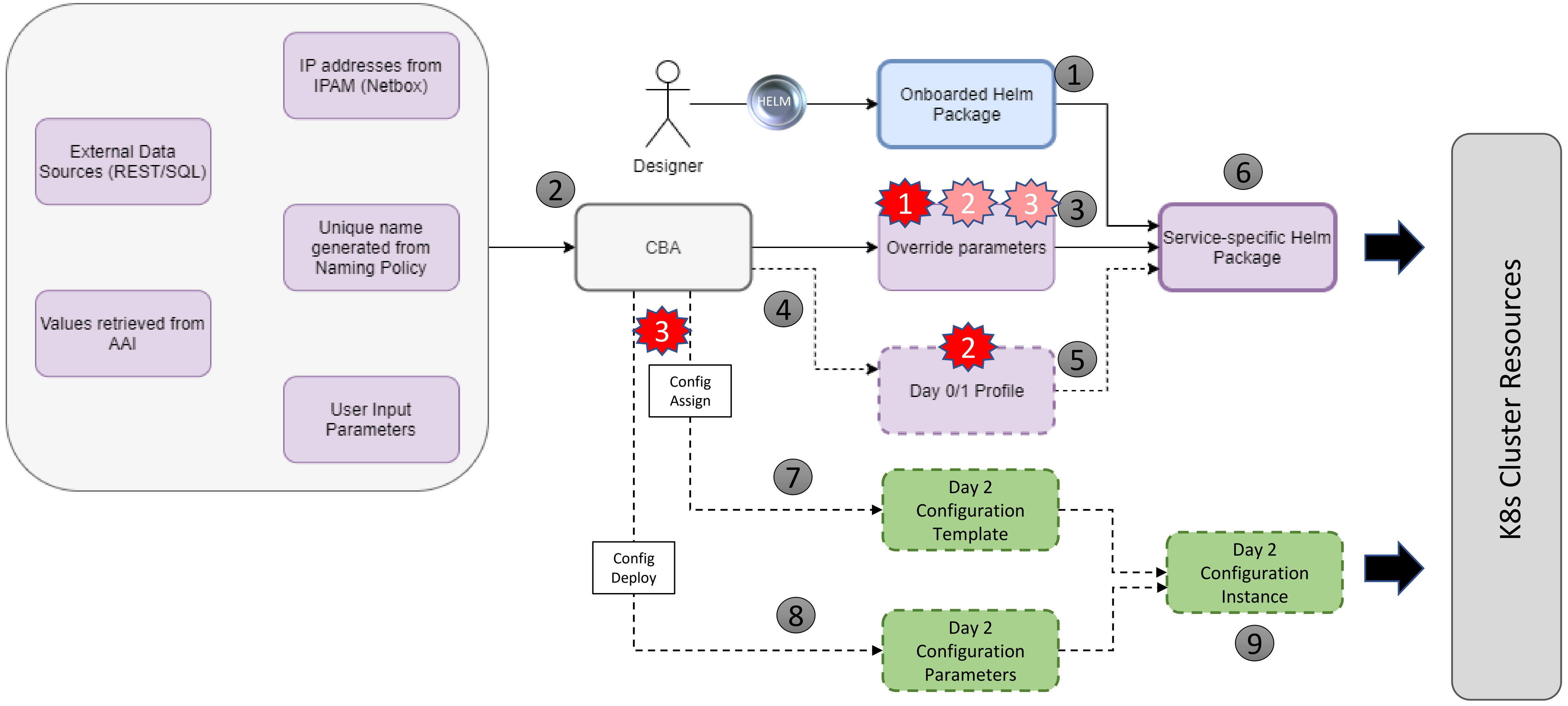
Name	Filename	Type	Version	UUID
helm_base_template	helm_base_template.tgz	HELM	1	24824f77-946c-46b3-b9b9-29d3ed513ae5
VF License	vf-license-model.xml	VF_LICENSE	1	886d62b5-2999-488f-aab2-32714f6823b7
base_template_dummy_ignore	base_template_dummy_ignore.yaml	HEAT	1	ed4a2b76-7b13-4d51-b9a8-842c83f399bf
VF HEAT ENV	base_template_dummy_ignore.env	HEAT_ENV	1	d2ac1c83-01fe-4171-a223-0c371777b08f
Vendor License	vendor-license-model.xml	VENDOR_LICENSE	1	46fdbace-d659-4d56-9103-dee6410c4f4f
helm_vsn	helm_vsn.tgz	HELM	1	0172a98a-29f1-4a39-81c9-c12a8aa5a1a2
helm_vfw	helm_vfw.tgz	HELM	1	0b173703-210c-4aca-bdf8-17d41daef9b0
CBA	CBA.zip	CONTROLLER_BLUEPRINT_AR	1	c9115e77-7868-4219-8c5a-ea6c6e07bfa7
helm_vpkg	helm_vpkg.tgz	HELM	1	d9465238-a13e-4702-8c08-310fd316c06e

LF NETWORKING

LFN Developer & Testing Forum

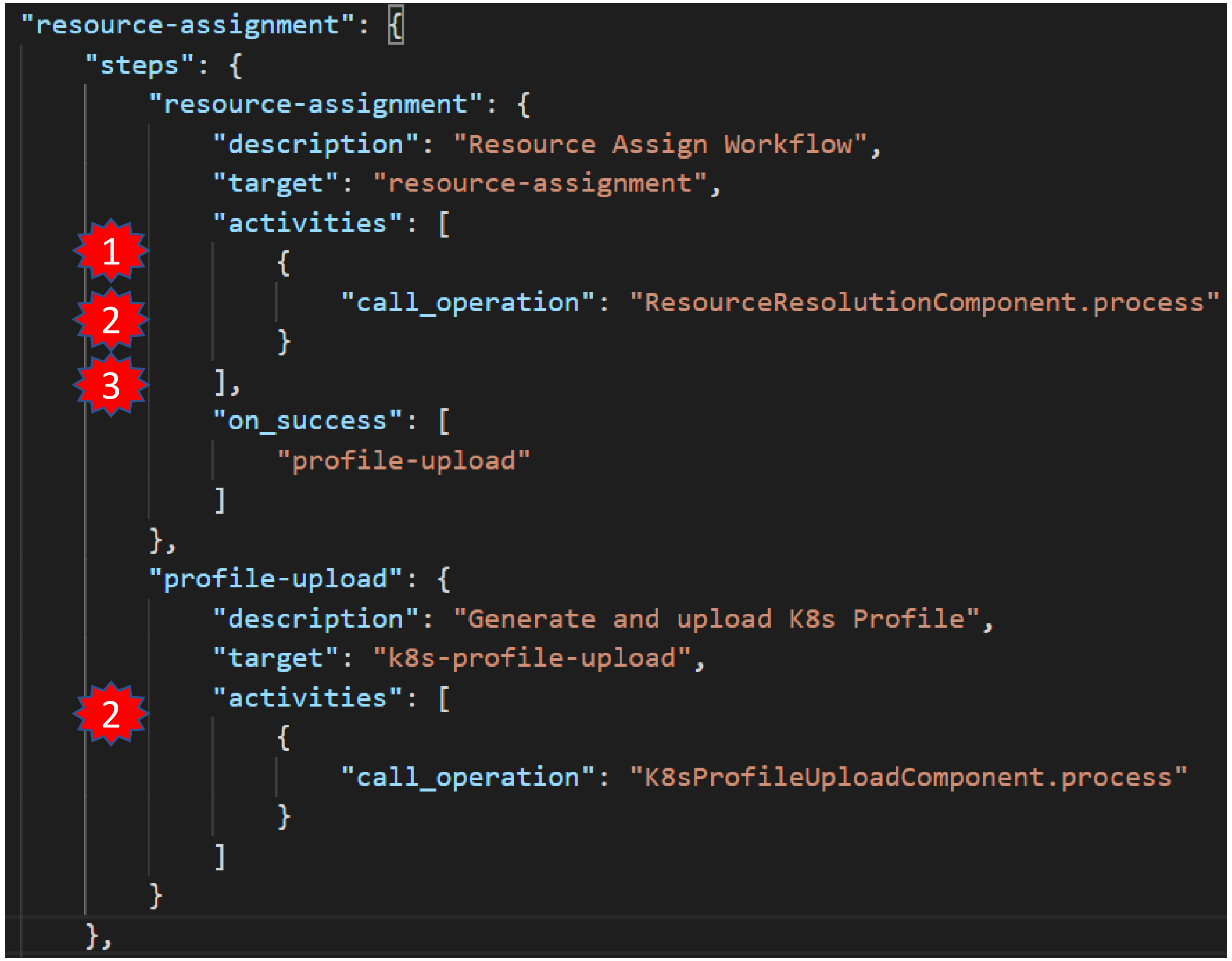
Create

Helm Package Day 0/1 + Day2


IP addresses from IPAM (Netbox)
External Data Sources (REST/SQL)
Unique name generated from Naming Policy
Values retrieved from AAI
User Input Parameters
Designer
HELM
Onboarded Helm Package
1
2
CBA
3
Config Assign
Config Deploy
1 2 3
Override parameters
3
4
2
Day 0/1 Profile
5
6
Service-specific Helm Package
7
Day 2 Configuration Template
8
Day 2 Configuration Parameters
Day 2 Configuration Instance
9
K8s Cluster Resources

CNF Day 0 – Helm Enrichment

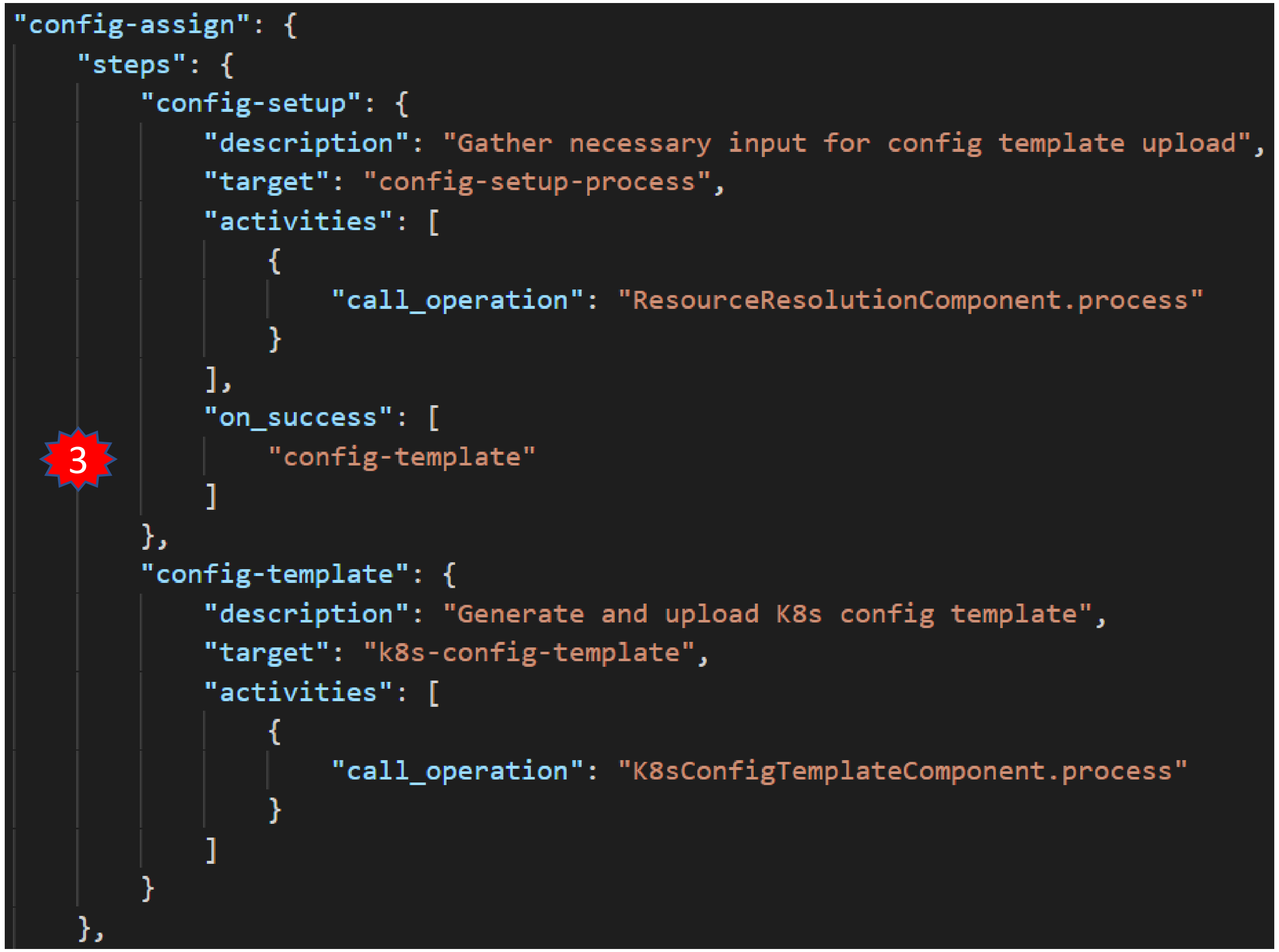
LF NETWORKING
LFN Developer & Testing Forum



```
"resource-assignment": {
    "steps": {
        "resource-assignment": {
            "description": "Resource Assign Workflow",
            "target": "resource-assignment",
            "activities": [
                {
                    "call_operation": "ResourceResolutionComponent.process"
                }
            ],
            "on_success": [
                "profile-upload"
            ]
        },
        "profile-upload": {
            "description": "Generate and upload K8s Profile",
            "target": "k8s-profile-upload",
            "activities": [
                {
                    "call_operation": "K8sProfileUploadComponent.process"
                }
            ]
        }
    },
```

- CNF instance based
- Modifies Helm package from VSP
- Part of Resource Assignment in CDS
- Native mechanisms in CDS
  - Customizable by CBA
- Modification of Helm values
  - Main
  - Nested
- Modification of Helm templates in the package from VSP
- Provisioning of new Helm templates in the package from VSP

CNF Day 2 – Config Preparation

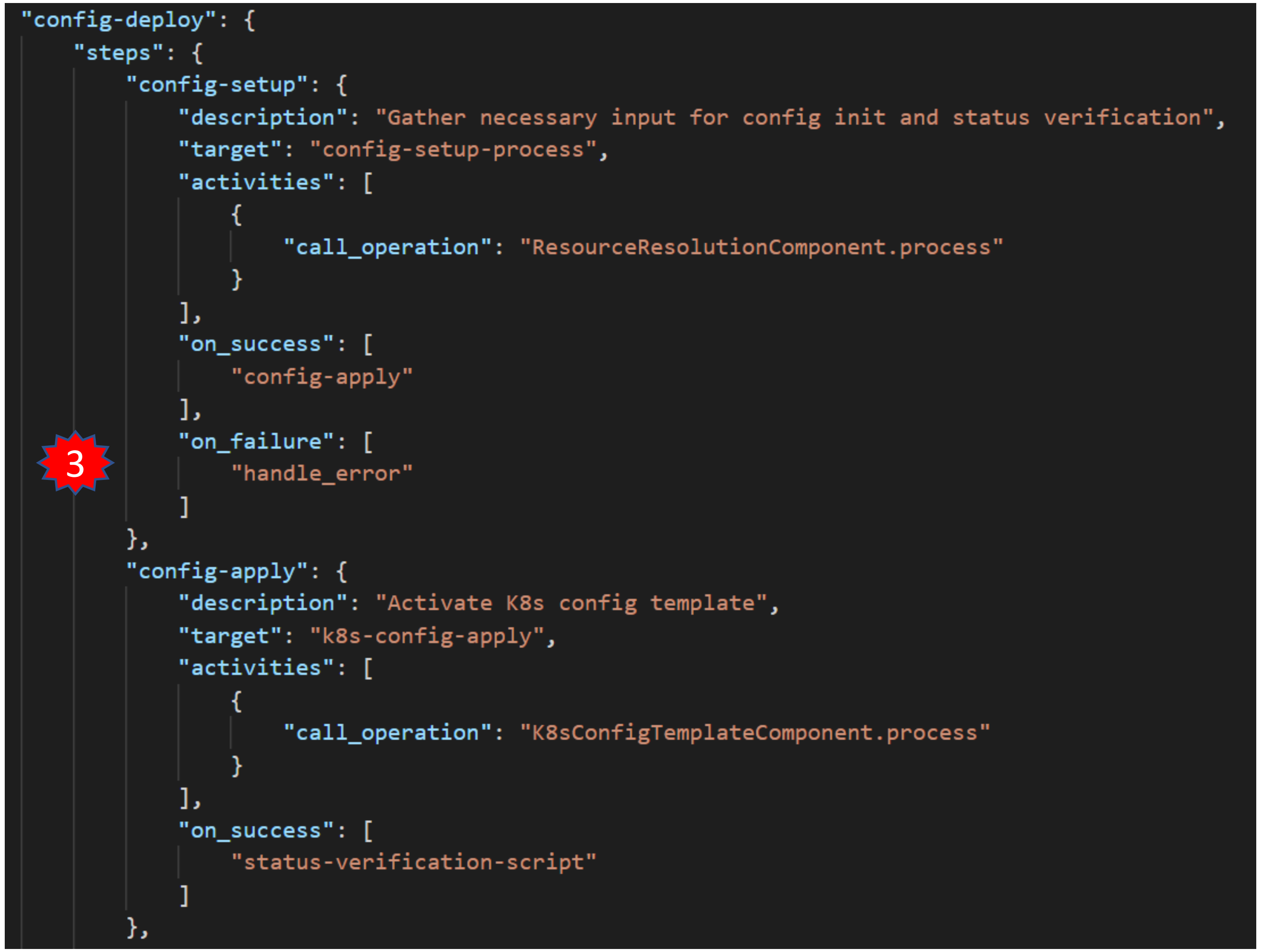


```
"config-assign": {
    "steps": {
        "config-setup": {
            "description": "Gather necessary input for config template upload",
            "target": "config-setup-process",
            "activities": [
                {
                    "call_operation": "ResourceResolutionComponent.process"
                }
            ],
            "on_success": [
                "config-template"
            ]
        },
        "config-template": {
            "description": "Generate and upload K8s config template",
            "target": "k8s-config-template",
            "activities": [
                {
                    "call_operation": "K8sConfigTemplateComponent.process"
                }
            ]
        }
    },
```

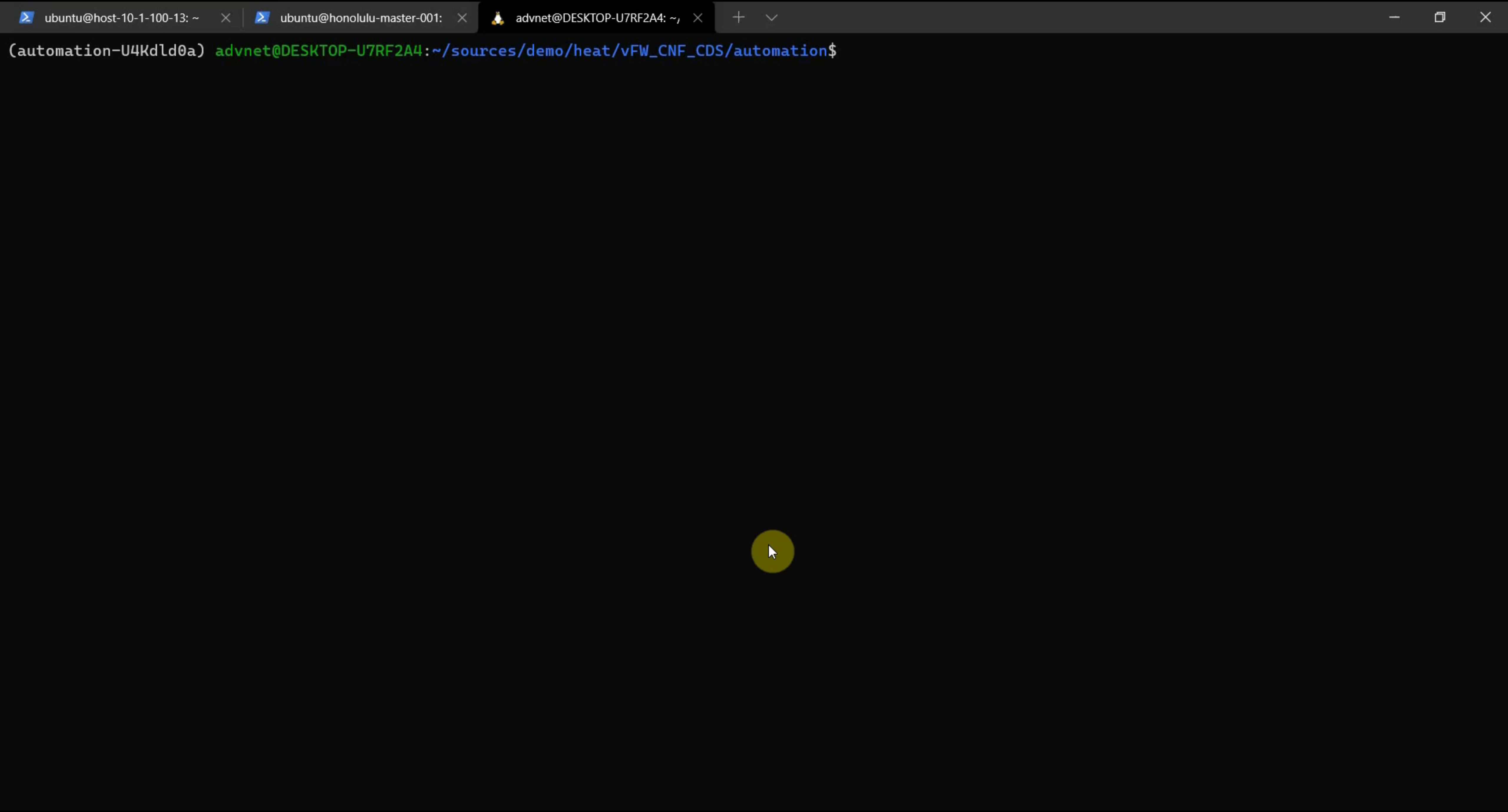
- CNF instance based
- Config Template (CfT)
  - Helm package
  - Build or modified by CDS
  - Part of CBA
- CfT preparation is part of Config-Assign in CDS
- Native mechanisms in CDS
  - Customizable by CBA
- Config Setup merges data
  - CBA
  - AAI i.e. vf-modules info
  - MDSAL – i.e. resolved Day 0
  - K8s – i.e. k8s resource status info
  - Kotlin, Python, REST
  - Complex JSON

CNF Day 2 – Config Creation

LF NETWORKING
LFN Developer & Testing Forum


```
"config-deploy": {
    "steps": {
        "config-setup": {
            "description": "Gather necessary input for config init and status verification",
            "target": "config-setup-process",
            "activities": [
                {
                    "call_operation": "ResourceResolutionComponent.process"
                }
            ],
            "on_success": [
                "config-apply"
            ],
            "on_failure": [
                "handle_error"
            ]
        },
        "config-apply": {
            "description": "Activate K8s config template",
            "target": "k8s-config-apply",
            "activities": [
                {
                    "call_operation": "K8sConfigTemplateComponent.process"
                }
            ],
            "on_success": [
                "status-verification-script"
            ]
        },
```
3

- CNF instance based
- Config Instance (CfI)
  - Instantiates CfT
  - Provides overrides for CfT
- CfI creation is part of Config-Deploy in CDS
  - Creates new k8s resources
  - Modifies k8s resources of existing CNF instance
- Native mechanisms in CDS
  - Customizable by CBA
- In vFW CNF Use Case followed by simple Status Check
  - Checks Pod Status until „Running”
  - Fails after 30 retries



ubuntu@host-10-1-100-13: ~ ubuntu@honolulu-master-001: advnet@DESKTOP-U7RF2A4: ~/

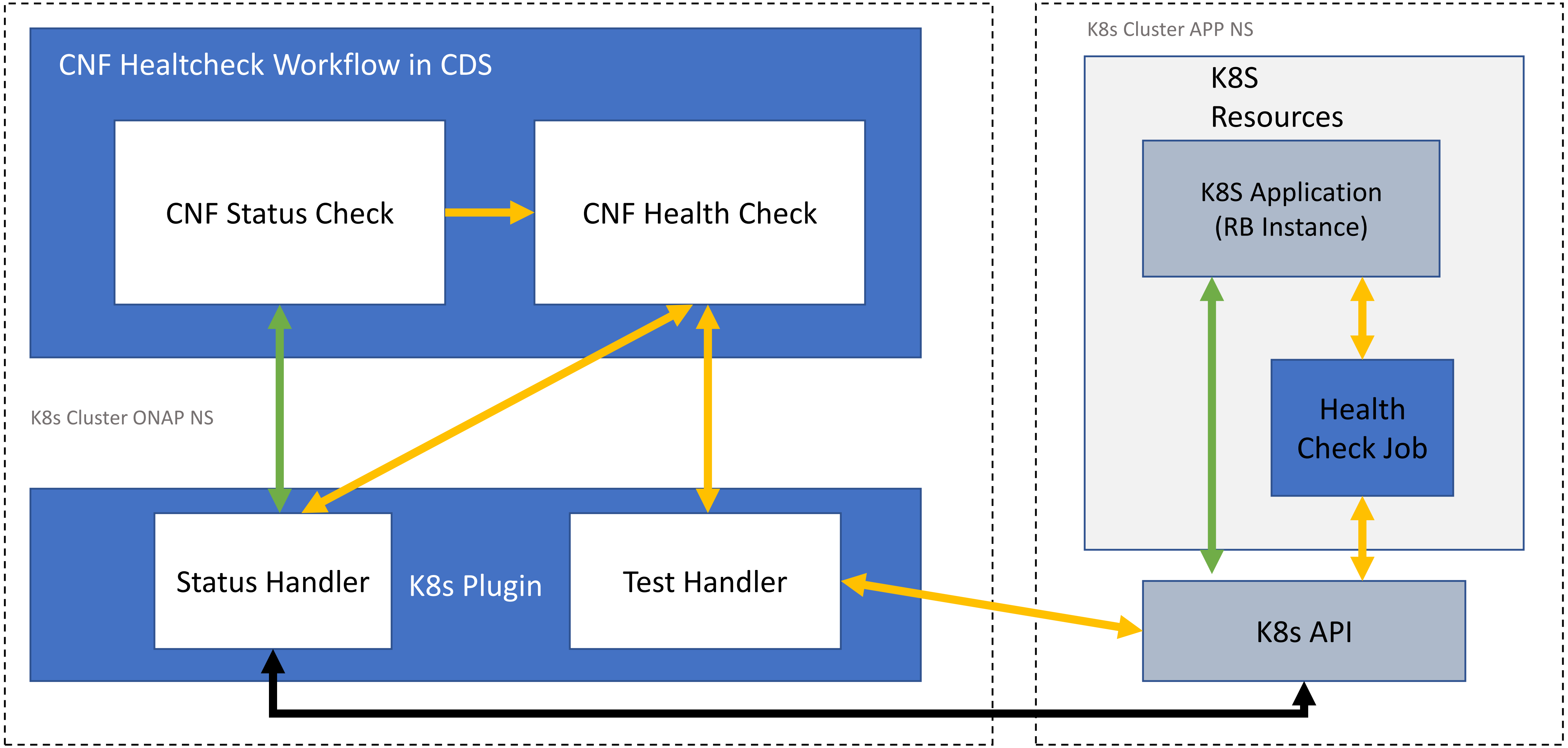
```
(automation-U4Kdld0a) advnet@DESKTOP-U7RF2A4:~/sources/demo/heat/vFW_CNF_CDS/automation$
```

LF NETWORKING

LFN Developer & Testing Forum

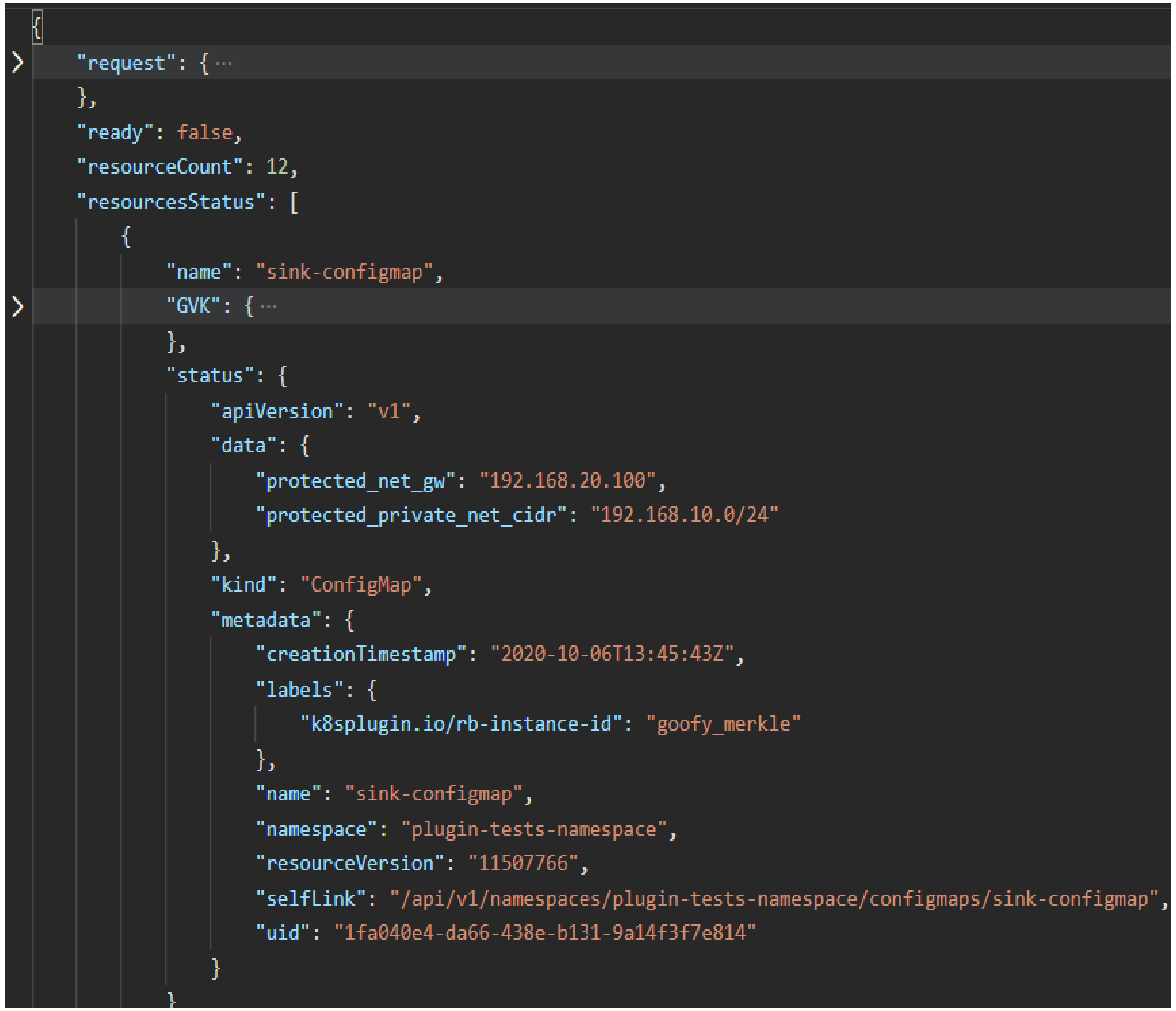
Healthcheck

CNF Health Check



CNF Health Check – Status Handler

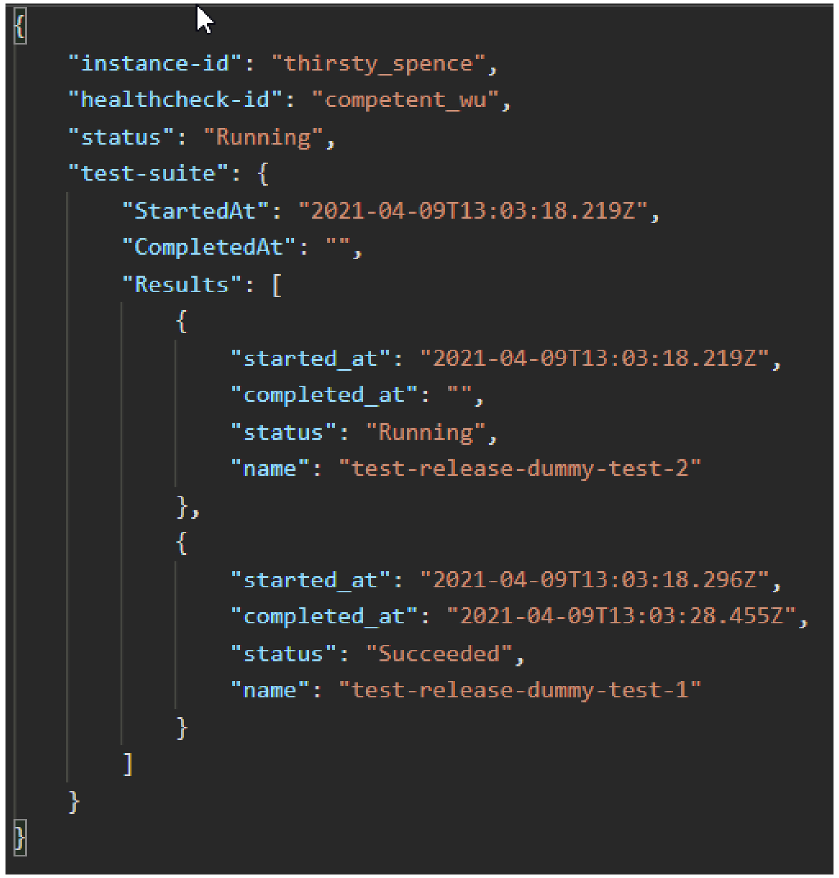
- Retrieves Helm Chart’s Instance (vf-module) resources’ status from managed k8s and exposes it via HTTP API
- Allows user to check any kind of data that k8s is aware of for the resources:
  - Pod’s State,
  - Deployment’s Replicas number,
  - Service’s NodePort
- For Health-check use, Status Handler client can parse the result and look for specific fields to ensure expected values:
  - Replica No > 3,
  - Service LoadBalancer Ready,
  - Pod allocated on Node different than X



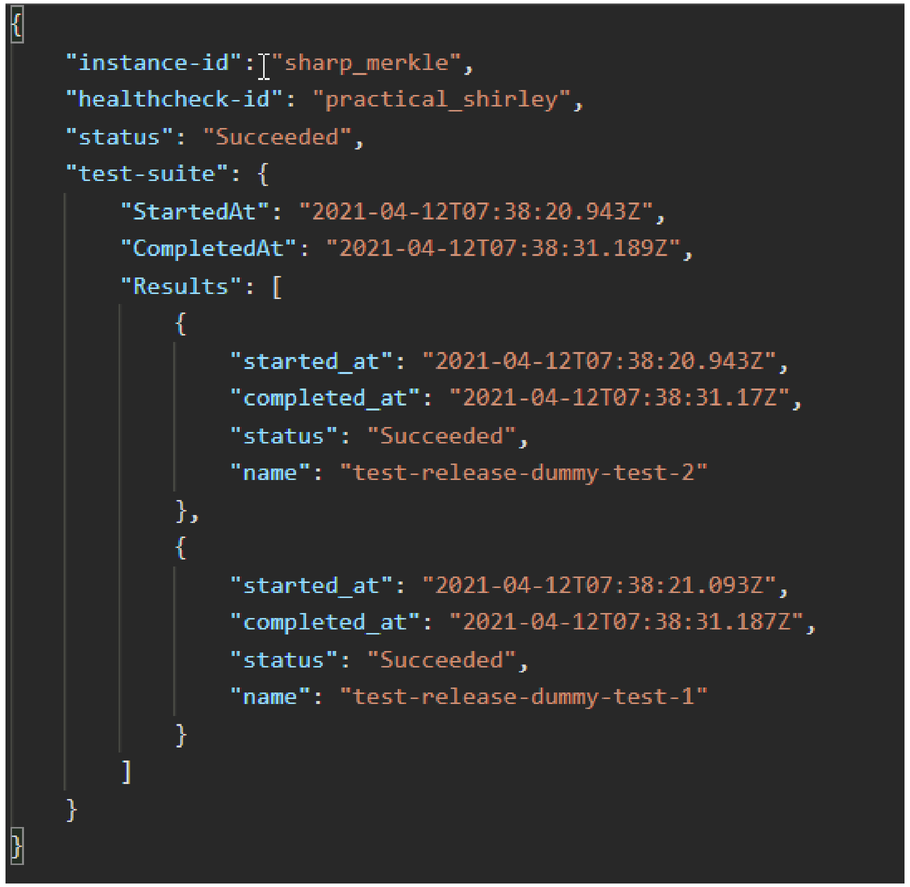
```
{
    "request": {⋯
    },
    "ready": false,
    "resourceCount": 12,
    "resourcesStatus": [
        {
            "name": "sink-configmap",
            "GVK": {⋯
            },
            "status": {
                "apiVersion": "v1",
                "data": {
                    "protected_net_gw": "192.168.20.100",
                    "protected_private_net_cidr": "192.168.10.0/24"
                },
                "kind": "ConfigMap",
                "metadata": {
                    "creationTimestamp": "2020-10-06T13:45:43Z",
                    "labels": {
                        "k8splugin.io/rb-instance-id": "goofy_merkle"
                    },
                    "name": "sink-configmap",
                    "namespace": "plugin-tests-namespace",
                    "resourceVersion": "11507766",
                    "selfLink": "/api/v1/namespaces/plugin-tests-namespace/configmaps/sink-configmap",
                    "uid": "1fa040e4-da66-438e-b131-9a14f3f7e814"
                }
            }
```

CNF Health Check – Test Handler

- Executes Tests provided within CNF's Helm Package (see https://helm.sh/docs/topics/chart_tests/)
- Test is executed asynchronously and allows investigating result of every hook run
- Aggregated Test result is computed by K8splugin once every hook finishes. Tests execution time is not limited.



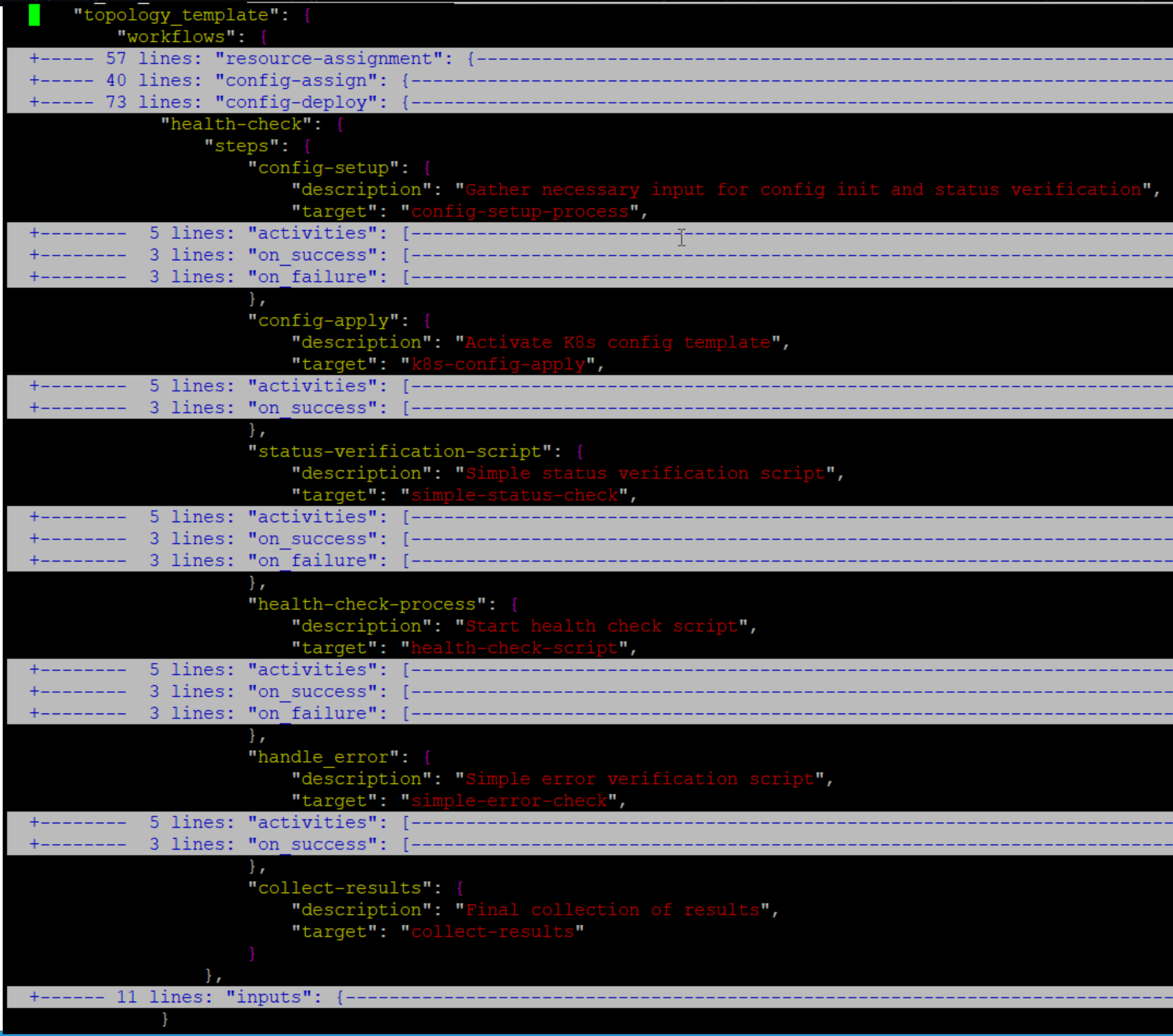
```
{
    "instance-id": "thirsty_spence",
    "healthcheck-id": "competent_wu",
    "status": "Running",
    "test-suite": {
        "StartedAt": "2021-04-09T13:03:18.219Z",
        "CompletedAt": "",
        "Results": [
            {
                "started_at": "2021-04-09T13:03:18.219Z",
                "completed_at": "",
                "status": "Running",
                "name": "test-release-dummy-test-2"
            },
            {
                "started_at": "2021-04-09T13:03:18.296Z",
                "completed_at": "2021-04-09T13:03:28.455Z",
                "status": "Succeeded",
                "name": "test-release-dummy-test-1"
            }
        ]
    }
}
```



```
{
    "instance-id": "sharp_merkle",
    "healthcheck-id": "practical_shirley",
    "status": "Succeeded",
    "test-suite": {
        "StartedAt": "2021-04-12T07:38:20.943Z",
        "CompletedAt": "2021-04-12T07:38:31.189Z",
        "Results": [
            {
                "started_at": "2021-04-12T07:38:20.943Z",
                "completed_at": "2021-04-12T07:38:31.17Z",
                "status": "Succeeded",
                "name": "test-release-dummy-test-2"
            },
            {
                "started_at": "2021-04-12T07:38:21.093Z",
                "completed_at": "2021-04-12T07:38:31.187Z",
                "status": "Succeeded",
                "name": "test-release-dummy-test-1"
            }
        ]
    }
}
```

CNF Health Check – vFW CBA PoC

- Healthcheck workflow defined in CBA verifies CNF healthiness by running several steps, among others:
  - `**config-setup**` and `**config-apply**` - resolve necessary IDs and names based on user-provided inputs
  - `**status-verification-script**` - 1st step of verification based solely on k8splugin's Status API
  - `**health-check-process**` - 2nd step testing CNF state using k8splugin's Healthcheck API



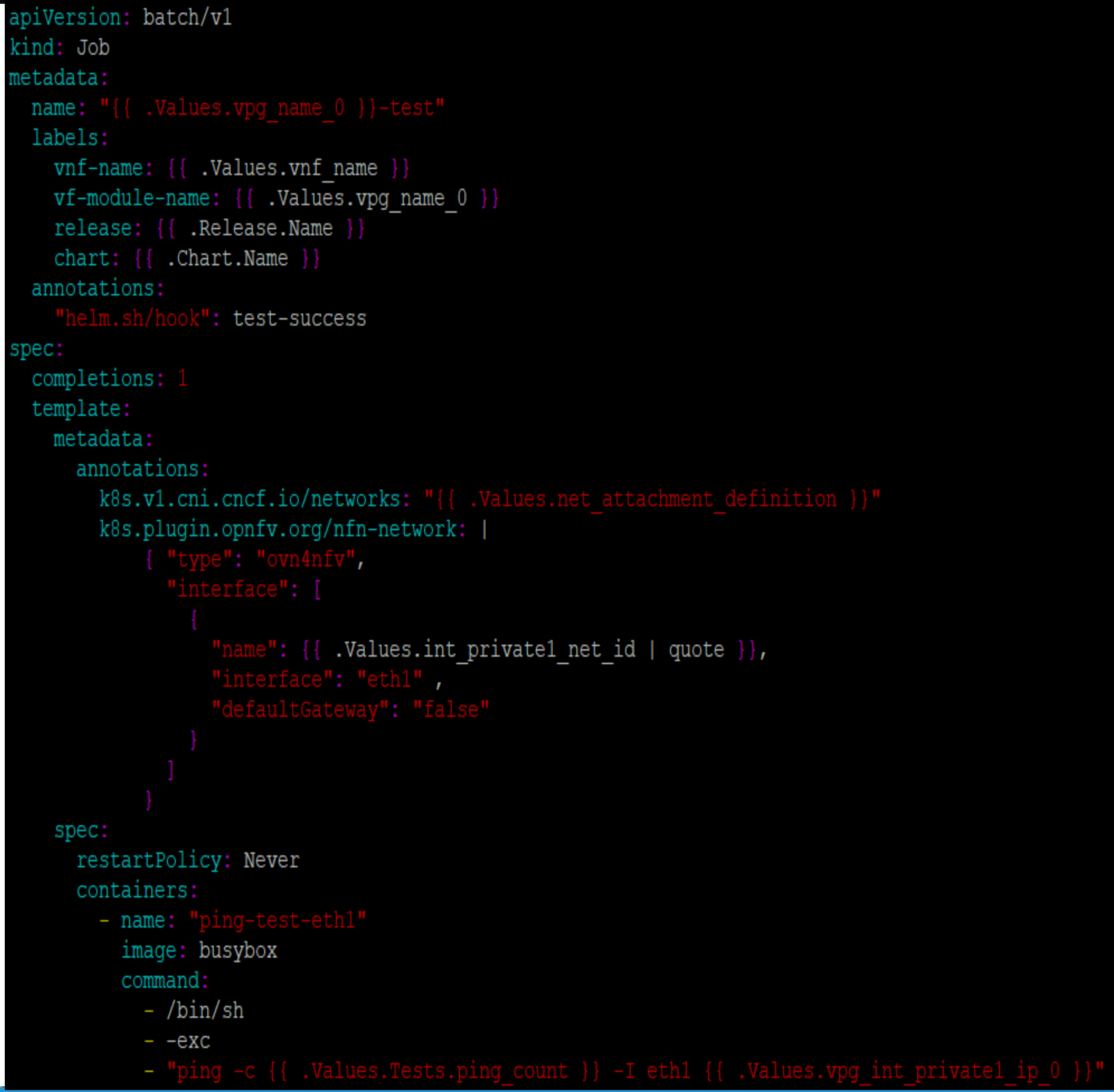
```
    "topology_template": {
        "workflows": {
+------ 57 lines: "resource-assignment": {----------------------------------------------------------------
+------ 40 lines: "config-assign": {----------------------------------------------------------------------
+------ 73 lines: "config-deploy": {----------------------------------------------------------------------
            "health-check": {
                "steps": {
                    "config-setup": {
                        "description": "Gather necessary input for config init and status verification",
                        "target": "config-setup-process",
+--------  5 lines: "activities": [-------------------------------------------------------------------
+--------  3 lines: "on_success": [-------------------------------------------------------------------
+--------  3 lines: "on_failure": [-------------------------------------------------------------------
                    },
                    "config-apply": {
                        "description": "Activate K8s config template",
                        "target": "k8s-config-apply",
+--------  5 lines: "activities": [-------------------------------------------------------------------
+--------  3 lines: "on_success": [-------------------------------------------------------------------
                    },
                    "status-verification-script": {
                        "description": "Simple status verification script",
                        "target": "simple-status-check",
+--------  5 lines: "activities": [-------------------------------------------------------------------
+--------  3 lines: "on_success": [-------------------------------------------------------------------
+--------  3 lines: "on_failure": [-------------------------------------------------------------------
                    },
                    "health-check-process": {
                        "description": "Start health check script",
                        "target": "health-check-script",
+--------  5 lines: "activities": [-------------------------------------------------------------------
+--------  3 lines: "on_success": [-------------------------------------------------------------------
+--------  3 lines: "on_failure": [-------------------------------------------------------------------
                    },
                    "handle_error": {
                        "description": "Simple error verification script",
                        "target": "simple-error-check",
+--------  5 lines: "activities": [-------------------------------------------------------------------
+--------  3 lines: "on_success": [-------------------------------------------------------------------
                    },
                    "collect-results": {
                        "description": "Final collection of results",
                        "target": "collect-results"
                    }
                },
+------ 11 lines: "inputs": {-------------------------------------------------------------------------
            }
```

CNF Health Check – vFW CBA PoC

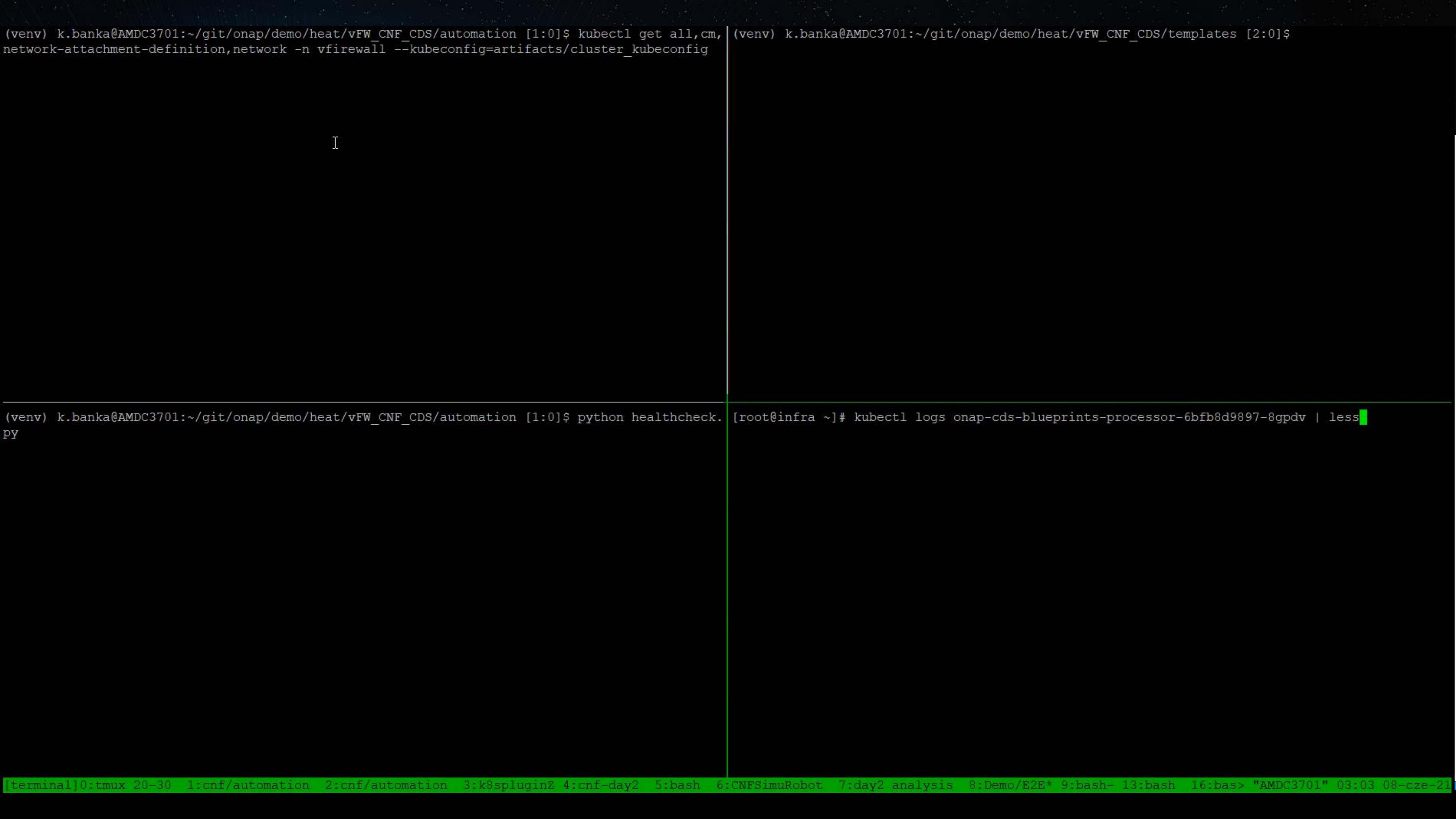
LFN Developer & Testing Forum

Example Test executed via Healthcheck API is a simple Job definition being part of orchestrated Helm Chart.

Defined Job attaches to custom networks used by vFW Pods and tests network interfaces reachability using `ping`.



```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: "{{ .Values.vpg_name_0 }}-test"
  labels:
    vnf-name: {{ .Values.vnf_name }}
    vf-module-name: {{ .Values.vpg_name_0 }}
    release: {{ .Release.Name }}
    chart: {{ .Chart.Name }}
  annotations:
    "helm.sh/hook": test-success
spec:
  completions: 1
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: "{{ .Values.net_attachment_definition }}"
        k8s.plugin.opnfv.org/nfn-network: |
            { "type": "ovn4nfv",
              "interface": [
                {
                  "name": {{ .Values.int_private1_net_id | quote }},
                  "interface": "eth1" ,
                  "defaultGateway": "false"
                }
              ]
            }
    spec:
      restartPolicy: Never
      containers:
        - name: "ping-test-eth1"
          image: busybox
          command:
            - /bin/sh
            - -exc
            - "ping -c {{ .Values.Tests.ping_count }} -I eth1 {{ .Values.vpg_int_private1_ip_0 }}"
```



```
(venv) k.banka@AMDC3701:~/git/onap/demo/heat/vFW_CNF_CDS/automation [1:0]$ kubectl get all,cm,
network-attachment-definition,network -n vfirewall --kubeconfig=artifacts/cluster_kubeconfig
```

```
(venv) k.banka@AMDC3701:~/git/onap/demo/heat/vFW_CNF_CDS/templates [2:0]$
```

```
(venv) k.banka@AMDC3701:~/git/onap/demo/heat/vFW_CNF_CDS/automation [1:0]$ python healthcheck.
py
```

```
[root@infra ~]# kubectl logs onap-cds-blueprints-processor-6bfb8d9897-8gpdv | less
```

```
[terminal]0:tmux 20-30  1:cnf/automation  2:cnf/automation  3:k8spluginZ 4:cnf-day2  5:bash  6:CNFSimuRobot  7:day2 analysis  8:Demo/E2E* 9:bash- 13:bash  16:bas> "AMDC3701" 03:03 08-cze-21
```

LF NETWORKING

LFN Developer & Testing Forum