

Release G: Automation Composition for O-DU Slice Assurance use case

This page describes how to create and run the control loops for the "Hello World" O-DU Slice Assurance usecase. This is done in kubernetes environment using the complete ONAP installation done via OOM. This use case is created using Kubernetes participant available in ONAP

The use case implementations are located in the "nonrtic/rapp/ransliceassurance" repo.

- [Configuring SMO and Automation Compositions](#)
 - [VES collector configuration](#)
 - [Policy clamp runtime and Kubernetes participant version](#)
 - [Creating service required by SDNC](#)
 - [Configuring stream and job to collect the Events](#)
- [Configuring NONRTRIC and Slice Assurance](#)
 - [Configuring dmaap adapter producer](#)
 - [Configuring simulator name in Slice Assurance](#)
 - [Configuration with http port of SDNC](#)
- [Automation Composition Deployment](#)
 - [Build and Upload Helm charts for Slice Assurance](#)
 - [Commission/Instantiate Automation Composition via GUI](#)
 - [Commission/Instantiate Automation Composition via commands](#)

Configuring SMO and Automation Compositions

This section describes the configuration required in SMO setup for this use case.

Instructions for the SMO installation can be found [here](#). After the installation onap, nonrtic and network namespaces in Kubernetes should be running with the respective containers.

VES collector configuration

Ves collector should be configured with the stream details to receive events. The following should be done to configure,

```
kubectl -n onap edit cm onap-dcae-ves-collector-application-config-configmap
```

Add the configuration as shown below in application_config.yaml

```
collector.dmaap.streamid: .....|o-ran-sc-du-hello-world-pm-streaming-oas3=ves-oran-pm-stream
....
streams_publishes:
  .....
  ves-oran-pm-stream:
    dmaap_info:
      topic_url: http://message-router:3904/events/unauthenticated.
VES_O_RAN_SC_HELLO_WORLD_PM_STREAMING_OUTPUT
      type: message_router
```

Then restart the deployment with the following command

```
kubectl rollout restart deployment onap-dcae-ves-collector
```

After the restart ves collector should be up and running.

Policy clamp runtime and Kubernetes participant version

Policy clamp runtime and k8sparticipant had several issues fixed recently and it is expected to have this version at the time of writing this document

```
Kubernetes Participant: nexus3.onap.org:10001/onap/policy-clamp-ac-k8s-ppnt:6.3.0
Clamp runtime: nexus3.onap.org:10001/onap/policy-clamp-runtime-acm:6.3.1
```

Creating service required by SDNC

SDNC requires a service named onap-dmaap to be running, It has been renamed as message-router, but being in place under some SDNC configuration.

There is no configuration/environment variables using which this can be configured, In order to solve this issue, A service with the name onap-dmaap should be created as shown below,

onap-dmaap-service.yaml

```
apiVersion: v1
kind: Service
metadata:
  annotations:
    meta.helm.sh/release-name: onap-dmaap
    meta.helm.sh/release-namespace: onap
    msb.onap.org/service-info: '[ { "serviceName": "onap-dmaap", "version": "v1",
      "url": "/", "protocol": "REST", "enable_ssl": false, "port": "3904", "visualRange": "1"
    } ]'
  creationTimestamp: "2022-10-20T13:50:35Z"
  labels:
    app.kubernetes.io/instance: onap
    app.kubernetes.io/managed-by: Helm
    app.kubernetes.io/name: onap-dmaap
    helm.sh/chart: message-router-10.0.0
  name: onap-dmaap
  namespace: onap
  resourceVersion: "9186544"
  selfLink: /api/v1/namespaces/onap/services/onap-dmaap
  uid: 4be149fd-af76-49a3-94e2-466623d2d0be
spec:
  internalTrafficPolicy: Cluster
  ipFamilies:
    - IPv4
  ipFamilyPolicy: SingleStack
  ports:
    - name: https-api
      port: 3905
      protocol: TCP
      targetPort: api
    - name: http-api
      port: 3904
      protocol: TCP
      targetPort: api-plain
  selector:
    app.kubernetes.io/instance: onap
    app.kubernetes.io/name: message-router
  sessionAffinity: None
  type: ClusterIP
status:
  loadBalancer: {}
```

Then run the following command

```
kubectl apply -f onap-dmaap-service.yaml
```

Configuring stream and job to collect the Events

Stream and job needs to be configured for the VES collector to get the required events. This can be configured using the commands below,

Configuring Event stream

```
curl -k -v -X PUT -H "Content-Type: application/yang-data+json" -H "Accept: application/yang-data+json" -d '{ "subscription-streams": { "id": "stream-1", "administrative-state": "unlocked", "user-label": "stream1", "ves-endpoint-protocol": "https", "ves-endpoint-auth-method": "basic-auth", "ves-endpoint-ip": "dcae-ves-collector.onap", "ves-endpoint-port": "8443", "ves-endpoint-username": "sample1", "ves-endpoint-password": "sample1" } }' -u admin:Kp8bJ4SXszM0WXlhak3eHlcse2gAw84vaoGGmJvUy2U https://<SDNC_HOST>:<SDNC_PORT>/rests/data/network-topology:network-topology/topology=topology-netconf/node=o-du-1122/yang-ext:mount/o-ran-sc-du-hello-world:network-function/subscription-streams=stream-1
```

Payload required for configuring the job

payload.json

```
{
  "performance-measurement-jobs": {
    "id": "pm-1",
    "administrative-state": "unlocked",
    "user-label": "pm",
    "job-tag": "my-job-tag",
    "performance-metrics": [
      "/o-ran-sc-du-hello-world:network-function/o-ran-sc-du-hello-world:distributed-unit-functions[o-ran-sc-du-hello-world:id='o-du-1122']/o-ran-sc-du-hello-world:cell[o-ran-sc-du-hello-world:id='cell-1']/o-ran-sc-du-hello-world:supported-measurements[o-ran-sc-du-hello-world:performance-measurement-type='user-equipment-average-throughput-uplink']/o-ran-sc-du-hello-world:supported-snssai-subcounter-instances[o-ran-sc-du-hello-world:slice-differentiator='1']/o-ran-sc-du-hello-world:slice-service-type='1']"
    ],
    "granularity-period": 30,
    "stream-target": "stream-1"
  }
}
```

Configuring event collection job

```
curl -k -v -X PUT -H "Content-Type: application/yang-data+json" -H "Accept: application/yang-data+json" -d @payload.json -u admin:Kp8bJ4SXszM0WXlhak3eHlcse2gAw84vaoGGmJvUy2U https://<SDNC_HOST>:<SDNC_PORT>/rests/data/network-topology:network-topology/topology=topology-netconf/node=o-du-1122/yang-ext:mount/o-ran-sc-du-hello-world:network-function/performance-measurement-jobs=pm-1
```

Configuring NONRTRIC and Slice Assurance

Couple of NONRTRIC components should be configured as shown below

Configuring dmaap adapter producer

Dmaapadapter should be configured with new producer type using the steps below

```
kubect1 -n nonrtric edit cm dmaapadapterservice-configmap-data
```

Add the configuration as shown below in application_configuration.json

application_configuration.json

```
{
  "types": [
    ...
    {
      "id": "Performance_Measurement_Streaming",
      "dmaapTopicUrl": "/events/unauthenticated.VES_O_RAN_SC_HELLO_WORLD_PM_STREAMING_OUTPUT/myG/C1",
      "useHttpProxy": false
    }
    ...
  ]
}
```

Then restart the statefulset with the following command

```
kubect1 rollout restart statefulset dmaapadapterservice
```

Configuring simulator name in Slice Assurance

Stub version uses the node name as "O-DU-1122". It is different from the network simulators run by OOM. So the node name should be configured in helm values as shown below

Rapp simulator node name configuration

```
...
simulator:
  node: o-du-1122
...
```

Configuration with http port of SDNC

Slice assurance doesn't supports the https calls to SDNC as of now.

OOM installation doesn't have the http port enabled for SDNC.

So It can be configured with pod's ClusterIP/Nodeport using the helm values as shown below.

Rapp sdnc configuration

```
...
sdnc:
  address: http://10.1.49.120:8181
...
```

Automation Composition Deployment

Tosca template used for commission is shown below,

commission.yaml

```
# Copyright (C) 2022 Nordix Foundation. All rights reserved.
# =====
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
```

```

# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
# =====LICENSE_END=====
#
tosca_definitions_version: tosca_simple_yaml_1_3
data_types:
  onap.datatypes.ToscaConceptIdentifier:
    derived_from: tosca.datatypes.Root
    properties:
      name:
        type: string
        required: true
      version:
        type: string
        required: true
node_types:
  org.onap.policy.clamp.acm.Participant:
    version: 1.0.1
    derived_from: tosca.nodetypes.Root
    properties:
      provider:
        type: string
        required: false
  org.onap.policy.clamp.acm.AutomationCompositionElement:
    version: 1.0.1
    derived_from: tosca.nodetypes.Root
    properties:
      provider:
        type: string
        required: false
      metadata:
        common: true
      description: Specifies the organization that provides the automation composition element
      participant_id:
        type: onap.datatypes.ToscaConceptIdentifier
        required: true
      metadata:
        common: true
      participantType:
        type: onap.datatypes.ToscaConceptIdentifier
        required: true
      metadata:
        common: true
      description: The identity of the participant type that hosts this type of Automation Composition Element
      startPhase:
        type: integer
        required: false
      constraints:
        - greater_or_equal: 0
      metadata:
        common: true
      description: A value indicating the start phase in which this automation composition element will be
started, the
      first start phase is zero. Automation Composition Elements are started in their start_phase order and
stopped
      in reverse start phase order. Automation Composition Elements with the same start phase are started
and
      stopped simultaneously
      uninitializedToPassiveTimeout:
        type: integer
        required: false
      constraints:
        - greater_or_equal: 0
      default: 60
      metadata:
        common: true
      description: The maximum time in seconds to wait for a state chage from uninitialized to passive
      passiveToRunningTimeout:

```

```

    type: integer
    required: false
    constraints:
      - greater_or_equal: 0
    default: 60
    metadata:
      common: true
    description: The maximum time in seconds to wait for a state chage from passive to running
  runningToPassiveTimeout:
    type: integer
    required: false
    constraints:
      - greater_or_equal: 0
    default: 60
    metadata:
      common: true
    description: The maximum time in seconds to wait for a state chage from running to passive
  passiveToUninitializedTimeout:
    type: integer
    required: false
    constraints:
      - greater_or_equal: 0
    default: 60
    metadata:
      common: true
    description: The maximum time in seconds to wait for a state chage from passive to uninitialized
org.onap.policy.clamp.acm.AutomationComposition:
  version: 1.0.1
  derived_from: tosca.nodetypes.Root
  properties:
    provider:
      type: string
      required: false
      metadata:
        common: true
      description: Specifies the organization that provides the automation composition element
  elements:
    type: list
    required: true
    metadata:
      common: true
    entry_schema:
      type: onap.datatypes.ToscaConceptIdentifier
      description: Specifies a list of automation composition element definitions that make up this
automation composition definition
org.onap.policy.clamp.acm.PolicyAutomationCompositionElement:
  version: 1.0.1
  derived_from: org.onap.policy.clamp.acm.AutomationCompositionElement
  properties:
    policy_type_id:
      type: onap.datatypes.ToscaConceptIdentifier
      required: true
    policy_id:
      type: onap.datatypes.ToscaConceptIdentifier
      required: false
org.onap.policy.clamp.acm.CDSAutomationCompositionElement:
  version: 1.0.1
  derived_from: org.onap.policy.clamp.acm.AutomationCompositionElement
  properties:
    cds_blueprint_id:
      type: onap.datatypes.ToscaConceptIdentifier
      required: true
org.onap.policy.clamp.acm.K8SMicroserviceAutomationCompositionElement:
  version: 1.0.1
  derived_from: org.onap.policy.clamp.acm.AutomationCompositionElement
  properties:
    chart:
      type: string
      required: true
    configs:
      type: list

```

```

    required: false
requirements:
  type: string
  required: false
templates:
  type: list
  required: false
  entry_schema:
values:
  type: string
  required: true
topology_template:
  node_templates:
    org.onap.policy.clamp.acm.KubernetesParticipant:
      version: 2.3.4
      type: org.onap.policy.clamp.acm.Participant
      type_version: 1.0.1
      description: Participant for K8S
      properties:
        provider: ONAP
    org.onap.domain.sample.SliceAssurance_K8SMicroserviceAutomationCompositionElement:
      version: 1.2.3
      type: org.onap.policy.clamp.acm.K8SMicroserviceAutomationCompositionElement
      type_version: 1.0.1
      description: Automation composition element for the K8S microservice for O-DU Slice Assurance
      properties:
        provider: ONAP
        participant_id:
          name: K8sParticipant0
          version: 1.0.0
        participantType:
          name: org.onap.policy.clamp.acm.KubernetesParticipant
          version: 2.3.4
        uninitializedToPassiveTimeout: 180
        podStatusCheckInterval: 10
        chart:
          chartId:
            name: odu-app
            version: 1.0.0
            namespace: nonrtric
            releaseName: odu-app
            repository:
              repoName: chartmuseum
              address: http://10.0.1.1:18080
    org.onap.domain.sample.SliceAssuranceIcsVersion_K8SMicroserviceAutomationCompositionElement:
      version: 1.2.3
      type: org.onap.policy.clamp.acm.K8SMicroserviceAutomationCompositionElement
      type_version: 1.0.1
      description: Automation composition element for the K8S microservice for O-DU Slice Assurance Ics Version
      properties:
        provider: ONAP
        participant_id:
          name: K8sParticipant0
          version: 1.0.0
        participantType:
          name: org.onap.policy.clamp.acm.KubernetesParticipant
          version: 2.3.4
        uninitializedToPassiveTimeout: 180
        podStatusCheckInterval: 10
        chart:
          chartId:
            name: odu-app-ics-version
            version: 1.0.0
            namespace: nonrtric
            releaseName: odu-app-ics-version
            repository:
              repoName: chartmuseum
              address: http://10.0.1.1:18080
    org.onap.domain.sample.GenericK8s_AutomationCompositionDefinition:
      version: 1.2.3
      type: org.onap.policy.clamp.acm.AutomationComposition

```

```
type_version: 1.0.1
description: Automation composition for O-DU Slice Assurance
properties:
  provider: ONAP
  elements:
    - name: org.onap.domain.sample.SliceAssurance_K8SMicroserviceAutomationCompositionElement
      version: 1.2.3
    - name: org.onap.domain.sample.SliceAssuranceIcsVersion_K8SMicroserviceAutomationCompositionElement
      version: 1.2.3
```

Build and Upload Helm charts for Slice Assurance

Helm chart for Sliceassurance smo and ics version needs to be built and uploaded into the chartmuseum server running as part of OOM installation using the commands below,

Helm Build and Upload

```
cd ransliceassurance/smoversion/helm
helm package .
curl --data-binary "@odu-app-1.0.0.tgz" http://<NodeIP>:18080/api/charts

cd ransliceassurance/icsversion/helm
helm package .
curl --data-binary "@odu-app-ics-version-1.0.0.tgz" http://<NodeIP>:18080/api/charts
```

Commission/Instantiate Automation Composition via GUI

NodePort of policy-gui can be done by using the command "kubectl -n onap get svc | grep policy-gui".

Then, open a web browser and navigate to the url:

<https://<NodeIP>:<NodePort-policy-gui>/clamp/>

Use below credentials for the GUI:

username: demo@people.osaaf.org. password: demo123456!

ONAP CLAMP

POLICY Framework ▾ CLAMP Options ▾ LOOP Instance ▾ LOOP Operations ▾ TOSCA Automation Composition ▾ Help ▾

Signed in as: demo@people.osaaf.org

Loop Viewer - Empty (NO loop loaded yet) - ()

No LOOP (SVG)

Loop Status:

Component Name	Component State	Description
----------------	-----------------	-------------

Go to **Tosca Automation Composition** pane, and select **Upload Tosca to Commissioning** in order to upload the tosca template (provided later in the relevant sub-section).

ONAP CLAMP POLICY Framework ▾ CLAMP Options ▾ LOOP Instance ▾ LOOP Operations ▾ TOSCA Automation Composition ▾ Help ▾ Signed in as: demo@people.osaaf.org

Loop Viewer - Empty (NO loop loaded yet) - ()

No LOOP (SVG)

Instantiation Management

Commissioning

Manage Commissioned Automation Composition Template

Upload Automation Composition to Commissioning

Edit Automation Composition Properties

Loop Status:

Component Name	Component State	Description
----------------	-----------------	-------------

Loop Logs

Date	Type	Component	Log
------	------	-----------	-----

ONAP CLAMP POLICY Framework ▾ CLAMP Options ▾ LOOP Instance ▾ LOOP Operations ▾ TOSCA Automation Composition ▾ Help ▾ Signed in as: demo@people.osaaf.org

Loop Viewer - Empty (NO loop loaded yet) - ()

No LOOP (SVG)

Loop Status:

Component Name	Component State	Description
----------------	-----------------	-------------

Loop Logs

Date	Type	Component	Log
------	------	-----------	-----

Upload Tosca to Commissioning API

commission.yaml Browse

Only .yaml, .yml and .json files are supported

Upload Tosca Service Template

Upload Success

Tosca Service Template from commission.yaml was Successfully Uploaded

Type: application/x-yaml

Size: 77.77Kb

Close

After commissioning the tosca template, the next step is to instantiate the control loop. Go to **Tosca Automation Composition** pane, and select **Instantiation Management** and then press the **Create Instance** button. If no changes need to be made in the instance properties, press the **Save** button and it should show a message depicting that the instantiation operation was successful.

ONAP CLAMP POLICY Framework ▾ CLAMP Options ▾ LOOP Instance ▾ LOOP Operations ▾ TOSCA Automation Composition ▾ Help ▾ Signed in as: demo@people.osaaf.org

Loop Viewer - Empty (NO loop loaded yet) - ()

No LOOP (SVG)

Instantiation Management

Commissioning

Manage Commissioned Automation Composition Template

Upload Automation Composition to Commissioning

Edit Automation Composition Properties

Loop Status:

Component Name	Component State	Description
----------------	-----------------	-------------

Loop Logs

ONAP
OPEN NETWORK AUTOMATION PLATFORM

ed in as: demo@people.osaaf.org

Loop Viewer - Empty (NO loop loaded)

No LOOP (SVG)

Manage Instances

Create InstanceMonitor Instantiations

#	Instantiation Name	Edit Instantiation	Delete Instantiation	Change Order State	Instantiation Order State	Instantiation Current State
---	--------------------	--------------------	----------------------	--------------------	---------------------------	-----------------------------

Clear ErrorMessageClose

Loop Status:

Component Name	Component State	Description
----------------	-----------------	-------------

Loop Logs

Date	Type	Component	Log
------	------	-----------	-----

Create Tosca Instance Properties

Instance Name:

InstancePropertiesJSONproperties

org.onap.policy.clamp.acm.KubernetesParticipantJSONproperties

org.onap.domain.sample.SliceAssuranceIcvsVersion_K8SMicroserviceAutomationCompositionElementJSONproperties

org.onap.domain.sample.SliceAssurance_K8SMicroserviceAutomationCompositionElementJSONproperties

Instantiation Properties Success
Instance Properties was successfully saved

SaveClose

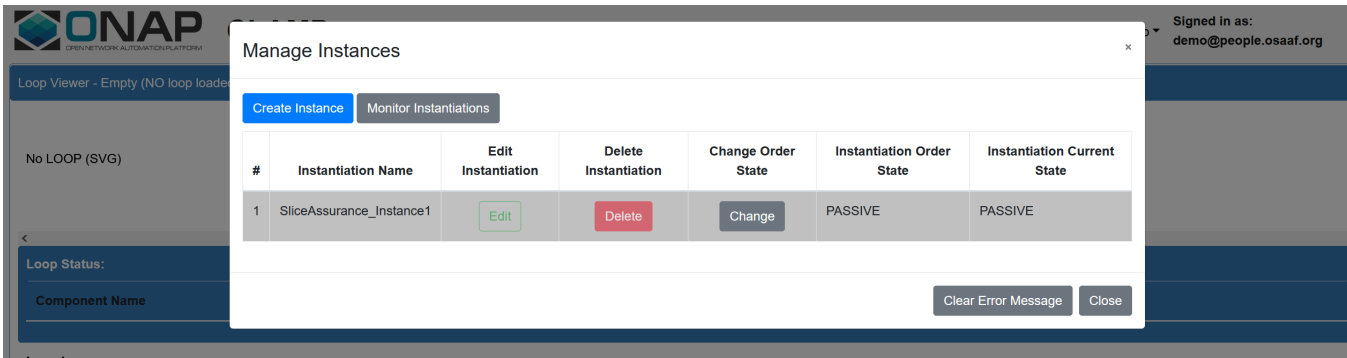
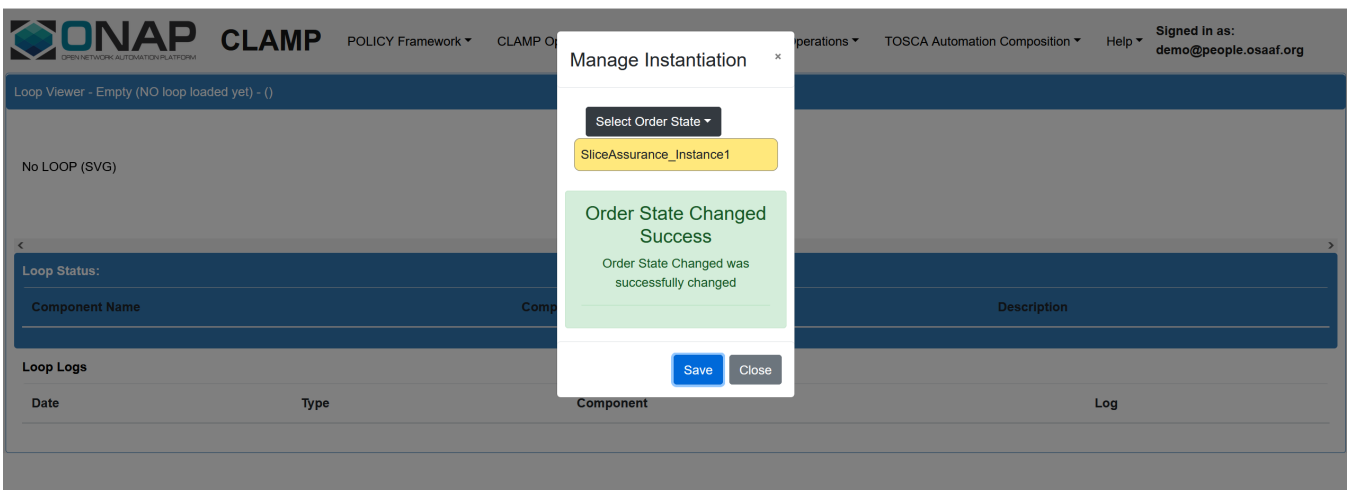
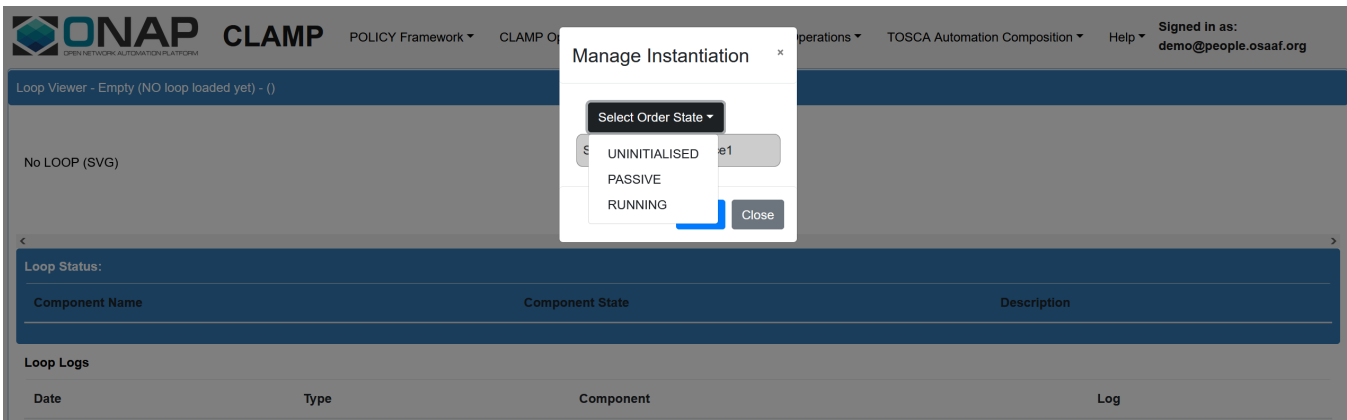
Manage Instances

Create InstanceMonitor Instantiations

#	Instantiation Name	Edit Instantiation	Delete Instantiation	Change Order State	Instantiation Order State	Instantiation Current State
1	SliceAssurance_Instance1	Edit	Delete	Change	UNINITIALISED	UNINITIALISED

Clear ErrorMessageClose

Press the **Change** button under **Change Order State**. Then, press the **Select Order State** drop-down menu, and select PASSIVE. Finally, press the **Save** button to change the control loop to PASSIVE state.



Once the control loop gets into the PASSIVE state, the corresponding version of the use case should be up and running (State of the Instance can be updated to RUNNING via the GUI).

```
root@kubeflow-poc-vm-1:/home/ubuntu# helm ls -n nonrtic
```

NAME	NAMESPACE	REVISION	UPDATED	STATUS	CHART	APP VERSIO
odu-app	nonrtic	1	2022-11-03 12:06:57.973580887 +0000 UTC	deployed	odu-app-1.0.0	1.0.0
odu-app-ics-version	nonrtic	1	2022-11-03 12:07:31.375085508 +0000 UTC	deployed	odu-app-ics-version-1.0.0	1.0.0

Commission/Instantiate Automation Composition via commands

Automation composition can be commissioned and initiated using commands as below,

Uploading the commission as shown below,

Upload commission Yaml

```
curl -v -X POST -k -u 'runtimeUser:ZiboTipdZeyt9]' -H Content-Type:application/yaml
https://<POLICY_RUNTIME_HOST>:6969/onap/policy/clamp/acm/v2/commission/ --data-binary @commission.yaml
```

It should give the below response,

Upload commission response

```
{
  "errorDetails": null,
  "affectedAutomationCompositionDefinitions": [
    {
      "name": "org.onap.domain.sample.GenericK8s_AutomationCompositionDefinition",
      "version": "1.2.3"
    },
    {
      "name": "org.onap.domain.sample.SliceAssuranceIcsVersion_K8SMicroserviceAutomationCompositionElement",
      "version": "1.2.3"
    },
    {
      "name": "org.onap.domain.sample.SliceAssurance_K8SMicroserviceAutomationCompositionElement",
      "version": "1.2.3"
    },
    {
      "name": "org.onap.policy.clamp.acm.KubernetesParticipant",
      "version": "2.3.4"
    }
  ]
}
```

Create an instance of automation composition

Create automation composition instance

```
curl -v -X POST -k -u 'runtimeUser:ZiboTipdZeyt9]' -H Content-Type:application/json
https://<POLICY_RUNTIME_HOST>:6969/onap/policy/clamp/acm/v2/instantiation/ --data-binary @instantiation.json
```

Payload file as follows,

instantiation.json

```
{
  "automationCompositionList": [
    {
      "name": "SliceAssuranceInstance",
      "version": "1.0.1",
      "definition": {
        "name": "org.onap.domain.sample.GenericK8s_AutomationCompositionDefinition",
        "version": "1.2.3"
      },
      "state": "UNINITIALISED",
      "orderedState": "UNINITIALISED",
      "description": "Slice Assurance SMO version",
      "elements": [
        "709c62b3-8918-41b9-a747-d21eb79c6c12": {
          "id": "709c62b3-8918-41b9-a747-d21eb79c6c12",
          "definition": {
            "name": "org.onap.domain.sample.
SliceAssurance_K8SMicroserviceAutomationCompositionElement",
            "version": "1.2.3"
          },
          "participantType": {
            "name": "org.onap.policy.clamp.acm.KubernetesParticipant",
            "version": "2.3.4"
          },
          "participantId": {
            "name": "K8sParticipant0",
            "version": "1.0.0"
          },
          "state": "UNINITIALISED",
          "orderedState": "UNINITIALISED",
          "description": "Odu App smo version k8s Control Loop Element"
        },
        "709c62b3-8918-41b9-a747-d21eb79c6c13": {
          "id": "709c62b3-8918-41b9-a747-d21eb79c6c13",
          "definition": {
            "name": "org.onap.domain.sample.
SliceAssuranceIcsVersion_K8SMicroserviceAutomationCompositionElement",
            "version": "1.2.3"
          },
          "participantType": {
            "name": "org.onap.policy.clamp.acm.KubernetesParticipant",
            "version": "2.3.4"
          },
          "participantId": {
            "name": "K8sParticipant0",
            "version": "1.0.0"
          },
          "state": "UNINITIALISED",
          "orderedState": "UNINITIALISED",
          "description": "Odu App ics version k8s Control Loop Element"
        }
      ]
    }
  ]
}
```

Instantiation should give the below response,

Instance creation response

```
{
  "errorDetails": null,
  "affectedAutomationCompositions": [
    {
      "name": "SliceAssuranceInstance",
      "version": "1.0.1"
    }
  ]
}
```

Update instance status to PASSIVE using the below command

Instance status update Request

```
curl -v -X POST -k -u 'runtimeUser:ZiboTiptdZeyt9]' -H Content-Type:application/json
https://<POLICY_RUNTIME_HOST>:6969/onap/policy/clamp/acm/v2/instantiation/command/ --data-binary @instantiation-
command.json
```

Payload file as follows,

instantiation-command.json

```
{
  "orderedState": "PASSIVE",
  "automationCompositionIdentifierList": [
    {
      "name": "SliceAssuranceInstance",
      "version": "1.0.1"
    }
  ]
}
```

Instance update status is as follows,

Response of instance status update

```
{
  "errorDetails": null,
  "affectedAutomationCompositions": [
    {
      "name": "SliceAssuranceInstance",
      "version": "1.0.1"
    }
  ]
}
```

Status can be updated to RUNNING using the above request with the updated payload.

The container should be up and running after this state and it can be verified using the below command.

```
kubectl -n nonrtric get pod
```