DCAE MOD User Guide

Types of Users and Usage Instructions:

Short Video Series available at : https://www.youtube.com/playlist?list=PLj-oRfbkqkfnN_2vnfhivCesJ118SA_zG

Demo day demonstration recording and slides available at : https://wiki.onap.org/display/DW/2020-04-16+DCAE+Demo

Sr. No	User	Usage Instructions
1.	Developers who are looking to onboard their mS	· Access the Nifi Web UI url provided to you · Follow steps 2.b to 2.d · You should be able to see your microservices in the Nifi Web UI by clicking and dragging 'Processor' on the canvas, and searching for the name of the microservice/component/processor.
2.	Designers who are building the flows through UI and triggering distribution	· Access the Nifi Web UI url provided to you · Follow steps 3 to the end of the document
3.	Infrastructure/ Admins who want to stand up DCAE Mod and validate it	· Follow start to the end

1. Deployment of DCAE MOD components via Helm charts

The DCAE MOD components are deployed using the standard ONAP OOM deployment process. When deploying ONAP using the helm deploy command, DCAE MOD components are deployed when the dcaemod.enabled flag is set to true, either via a --set option on the command line or by an entry in an overrides file. In this respect, DCAE MOD is no different from any other ONAP subsystem.

The default DCAE MOD deployment relies on an nginx ingress controller being available in the Kubernetes cluster where DCAE MOD is being deployed. The Rancher RKE installation process sets up a suitable ingress controller. In order to enable the use of the ingress controller, it is necessary to override the OOM default global settings for ingress configuration. Specifically, the installation needs to set the following configuration in an override file:

Global Ingress configuration

```
#Global ingress configuration
ingress:
  enabled: true
  virtualhost:
    baseurl: "simpledemo.onap.org"
```

When DCAE MOD is deployed with an ingress controller, several endpoints are exposed outside the cluster at the ingress controller's external IP address and port. (In the case of a Rancher RKE installation, there is an ingress controller on every worker node, listening at the the standard HTTP port (80).) These exposed endpoints are needed by users using machines outside the Kubernetes cluster.

Endpoint	Routes to (cluster internal address)	Description

/nifi	http://dcaemod-designtool:8080/nifi	Design tool Web UI
/nifi-api	http://dcaemod-designtool:8080/nifi-api	Design tool API
/nifi-jars	http://dcaemod-nifi-registry:18080/nifi-jars	Flow registry listing of JAR files built from component specs
/onboarding	http://dcaemod-onboarding-api:8080/onboarding	Onboarding API
/distributor	http://dcaemod-distributor-api:8080/distributor	Distributor API

To access the design Web UI, for example, a user would use the URL : http://*ingress_controller_address:ingress_controller_port*/nifi. *ingress_controller_address* is the the IP address or DNS FQDN of the ingress controller and *ingress_controller_port* is the port on which the ingress controller is listening for HTTP requests. (If the port is 80, the HTTP default, then there is no need to specify a port.)

There are two additional *internal* endpoints that users need to know, in order to configure a registry client and a distribution target in the design tool's controller settings.

Configuration Item	Endpoint URL
Registry client	http://dcaemod-nifi-registry:18080
Distribution target	http://dcaemod-runtime-api:9090

As OOM/ingress template has been updated in Guilin release to enable virtual host, MOD API's and UI access via ingress should use **dcaemod.api.simpledemo.onap.org**

Add entry for **dcaemod.simpledemo.onap.org** in /etc/hosts with the correct IP (any of K8S node IP can be specified)

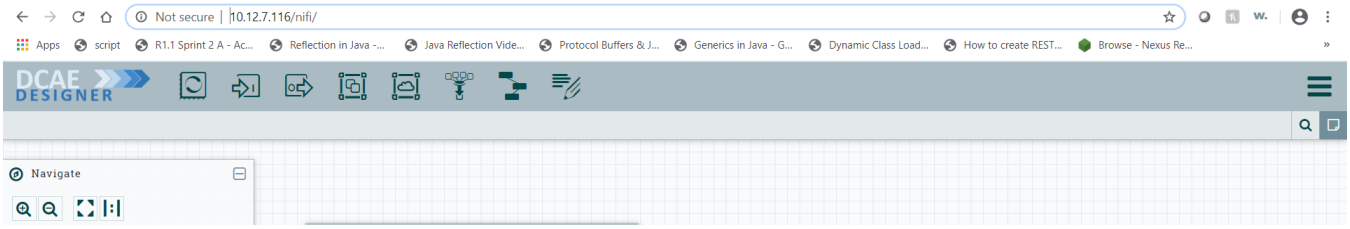
Using DCAE MOD without an Ingress Controller

Not currently supported.

2. Configuring DCAE mod

Our demo is hosted on 10.12.7.116. The IP Address for the purpose of this demo will hence be 10.12.7.116. In case of other deployments, we would have used the IP Address, or the DNS FQDN, if there is one, for one of the Kubernetes nodes.

Now let's access the Nifi (DCAE designer) UI - http://dcaemod.simpledemo.onap.org/nifi/



a) Get the artifacts to test and onboard.

Let's fetch the artifacts/ spec files

A sample Component DCAE-VES-Collector : https://git.onap.org/dcaegen2/collectors/ves/tree/dpo/spec/vescollector-componentspec.json

A sample Data Format : https://git.onap.org/dcaegen2/collectors/ves/tree/dpo/data-formats/VES-5.28.4-dataformat.json

For the purpose of onboarding, a Sample Request body should be of the type -

```
{ "owner": "<some value>", "spec": <some json object> }
```

where the json object inside the spec field can be a component spec json.

Request bodies of this type will be used in the onboarding requests you make using curl or the onboarding swagger interface.

The prepared Sample Request body for a component dcae-ves-collector looks like so


```
{
  "spec": {
    "self": {
      "version": "1.5.4",
      "name": "dcae-ves-collector",
      "description": "Collector for receiving VES events through restful interface",
      "component_type": "docker"
    },
    "streams": {
      "subscribes": [],
      "publishes": [
        {
          "format": "VES_specification",
          "version": "5.28.4",
          "type": "message router",
          "config_key": "ves-fault"
        },
        {
          "format": "VES_specification",
          "version": "5.28.4",
          "type": "message router",
          "config_key": "ves-measurement"
        },
        {
          "format": "VES_specification",
          "version": "5.28.4",
          "type": "message router",
          "config_key": "ves-syslog"
        },
        {
          "format": "VES_specification",
          "version": "5.28.4",
          "type": "message router",
          "config_key": "ves-heartbeat"
        },
        {
          "format": "VES_specification",
          "version": "5.28.4",
          "type": "message router",
          "config_key": "ves-other"
        },
        {
          "format": "VES_specification",
          "version": "5.28.4",
          "type": "message router",
          "config_key": "ves-mobileflow"
        },
        {
          "format": "VES_specification",
          "version": "5.28.4",
          "type": "message router",
          "config_key": "ves-statechange"
        },
        {
          "format": "VES_specification",
          "version": "5.28.4",
          "type": "message router",
          "config_key": "ves-thresholdCrossingAlert"
        },
        {
          "format": "VES_specification",
          "version": "5.28.4",
          "type": "message router",
          "config_key": "ves-voicequality"
        },
        {
          "format": "VES_specification",
          "version": "5.28.4",
          "type": "message router",
          "config_key": "ves-sipsignaling"
        },
        {
          "format": "VES_specification",
          "version": "7.30.1",
          "type": "message router",
          "config_key": "ves-pnfRegistration"
```

```
      },
      {
        "format": "VES_specification",
        "version": "7.30.1",
        "type": "message router",
        "config_key": "ves-notification"
      },
      {
        "format": "VES_specification",
        "version": "7.30.1",
        "type": "message router",
        "config_key": "ves-perf3gpp"
      }
    ]
  },
  "services": {
    "calls": [],
    "provides": [
      {
        "route": "/eventListener/v1",
        "verb": "POST",
        "request": {
          "format": "VES_specification",
          "version": "4.27.2"
        },
        "response": {
          "format": "ves.coll.response",
          "version": "1.0.0"
        }
      },
      {
        "route": "/eventListener/v2",
        "verb": "POST",
        "request": {
          "format": "VES_specification",
          "version": "4.27.2"
        },
        "response": {
          "format": "ves.coll.response",
          "version": "1.0.0"
        }
      },
      {
        "route": "/eventListener/v3",
        "verb": "POST",
        "request": {
          "format": "VES_specification",
          "version": "4.27.2"
        },
        "response": {
          "format": "ves.coll.response",
          "version": "1.0.0"
        }
      },
      {
        "route": "/eventListener/v4",
        "verb": "POST",
        "request": {
          "format": "VES_specification",
          "version": "4.27.2"
        },
        "response": {
          "format": "ves.coll.response",
          "version": "1.0.0"
        }
      },
      {
        "route": "/eventListener/v5",
        "verb": "POST",
        "request": {
          "format": "VES_specification",
          "version": "5.28.4"
        },
        "response": {
          "format": "ves.coll.response",
          "version": "1.0.0"
        }
      },
      {
        "route": "/eventListener/v7",
        "verb": "POST",
        "request": {
          "format": "VES_specification",
          "version": "7.30.1"
```

```
          },
          "response": {
            "format": "ves.coll.response",
            "version": "1.0.0"
          }
        }
      ]
    },
    "parameters": [
      {
        "name": "collector.service.port",
        "value": 8080,
        "description": "standard http port collector will open for listening;",
        "sourced_at_deployment": false,
        "policy_editable": false,
        "designer_editable": false
      },
      {
        "name": "collector.service.secure.port",
        "value": 8443,
        "description": "secure http port collector will open for listening ",
        "sourced_at_deployment": false,
        "policy_editable": false,
        "designer_editable": true
      },
      {
        "name": "collector.keystore.file.location",
        "value": "/opt/app/dcae-certificate/cert.jks",
        "description": "fs location of keystore file in vm",
        "sourced_at_deployment": false,
        "policy_editable": false,
        "designer_editable": false
      },
      {
        "name": "collector.keystore.passwordfile",
        "value": "/opt/app/dcae-certificate/jks.pass",
        "description": "location of keystore password file in vm",
        "sourced_at_deployment": false,
        "policy_editable": false,
        "designer_editable": false
      },
      {
        "name": "collector.truststore.file.location",
        "value": "/opt/app/dcae-certificate/trust.jks",
        "description": "fs location of truststore file in vm",
        "sourced_at_deployment": false,
        "policy_editable": false,
        "designer_editable": false
      },
      {
        "name": "collector.truststore.passwordfile",
        "value": "/opt/app/dcae-certificate/trust.pass",
        "description": "location of truststore password file in vm",
        "sourced_at_deployment": false,
        "policy_editable": false,
        "designer_editable": false
      },
      {
        "name": "collector.dmaap.streamid",
        "value": "fault\u003dves-fault|syslog\u003dves-syslog|heartbeat\u003dves-
heartbeat|measurementsForVfScaling\u003dves-measurement|measurement\u003dves-measurement|mobileFlow\u003dves-
mobileflow|other\u003dves-other|stateChange\u003dves-statechange|thresholdCrossingAlert\u003dves-
thresholdCrossingAlert|voiceQuality\u003dves-voicequality|sipSignaling\u003dves-
sipsignaling|notification\u003dves-notification|pnfRegistration\u003dves-pnfRegistration|perf3gpp\u003dves-
perf3gpp",
        "description": "domain-to-streamid mapping used by VESCollector to distributes events based on domain.
Both primary and secondary config_key are included for resilency (multiple streamid can be included commma
separated). The streamids MUST match to topic config_keys. For single site without resiliency deployment -
configkeys with -secondary suffix can be removed",
        "sourced_at_deployment": true,
        "policy_editable": false,
        "designer_editable": false
      },
      {
        "name": "auth.method",
        "value": "noAuth",
        "description": "Property to manage application mode, possible configurations: noAuth - default option -
no security (http) , certOnly - auth by certificate (https), basicAuth - auth by basic auth username and password
(https),certBasicAuth - auth by certificate and basic auth username / password (https),",
        "sourced_at_deployment": false,
        "policy_editable": false,
        "designer_editable": false
      },
      {
```

```
        "name": "header.authlist",
        "value": "sample1,$2a$10$pgjaxDzSuc6XVFEeqvxQ5u90DKJnM/u7TJTcinAlFJVaavXMWf
/Zi|userid1,$2a$10$61gNubgJJl9lh3nvQvY9X.x4e5ETWJJ7ao7ZhJEvmfJigov26Z6uq|userid2,$2a$10$G52y/3uhuhWAMy.
bx9Se8uzWinmbJa.dlm1LW6bYPdPkkywLDPLiy",
        "description": "List of id and base 64 encoded password.For each onboarding VNF - unique userid and
password should be assigned and communicated to VNF owner. Password value should be base64 encoded in config
here",
        "policy_editable": false,
        "sourced_at_deployment": true,
        "designer_editable": true
      },
      {
        "name": "collector.schema.checkflag",
        "value": 1,
        "description": "Schema check validation flag. When enabled, collector will validate input VES events
against VES Schema defined on collector.schema.file ",
        "sourced_at_deployment": false,
        "policy_editable": false,
        "designer_editable": false
      },
      {
        "name": "collector.schema.file",
        "value": "{\"v1\":\"./etc/CommonEventFormat_27.2.json\",\"v2\":\"./etc/CommonEventFormat_27.2.json\",\"
v3\":\"./etc/CommonEventFormat_27.2.json\",\"v4\":\"./etc/CommonEventFormat_27.2.json\",\"v5\":\"./etc
/CommonEventFormat_28.4.1.json\",\"v7\":\"./etc/CommonEventFormat_30.1.1.json\"}",
        "description": "VES schema file name per version used for validation",
        "designer_editable": true,
        "sourced_at_deployment": false,
        "policy_editable": false
      },
      {
        "name": "event.transform.flag",
        "value": 1,
        "description": "flag to enable tranformation rules defined under eventTransform.json; this is applicable
when event tranformation rules preset should be activated for transforming \u003cVES5.4 events to 5.4",
        "sourced_at_deployment": false,
        "policy_editable": false,
        "designer_editable": false
      },
      {
        "name": "tomcat.maxthreads",
        "value": "200",
        "description": "Tomcat control for concurrent request",
        "sourced_at_deployment": false,
        "policy_editable": false,
        "designer_editable": false
      }
    ],
    "auxilary": {
      "healthcheck": {
        "type": "http",
        "interval": "15s",
        "timeout": "1s",
        "endpoint": "/healthcheck"
      },
      "volumes": [
        {
          "container": {
            "bind": "/opt/app/dcae-certificate"
          },
          "host": {
            "path": "/opt/app/dcae-certificate"
          }
        },
        {
          "container": {
            "bind": "/opt/app/VESCollector/logs"
          },
          "host": {
            "path": "/opt/logs/DCAE/VESCollector/logs"
          }
        },
        {
          "container": {
            "bind": "/opt/app/VESCollector/etc"
          },
          "host": {
            "path": "/opt/logs/DCAE/VESCollector/etc"
          }
        }
      ],
      "ports": [
        "8080:0",
        "8443:0"
```

```
      ],
      "log_info": {
        "log_directory": "/opt/app/VESCollector/logs/"
      },
      "tls_info": {
        "cert_directory": "/opt/app/dcae-certificate/",
        "use_tls": true
      }
    },
    "artifacts": [
      {
        "type": "docker image",
        "uri": "nexus3.onap.org:10001/onap/org.onap.dcaegen2.collectors.ves.vescollector:latest"
      }
    ]
  },
  "owner": "onboarding_dev"
}
```


The prepared Sample request body for a sample data format looks like so -


```
{
  "spec": {
    "self": {
      "name": "VES_specification",
      "version": "4.27.2",
      "description": "VES spec from v4.1 and 27.2 spec"
    },
    "dataformatversion": "1.0.0",
    "jsonschema": {
      "$schema": "http://json-schema.org/draft-04/schema#",
      "definitions": {
        "schemaLicenseAndCopyrightNotice": {
          "description": "Copyright (c) 2017, AT\u0026T Intellectual Property.  All rights reserved",
          "type": "object",
          "properties": {
            "apacheLicense2.0": {
              "description": "Licensed under the Apache License, Version 2.0 (the \u0027License\u0027); you may
not use this file except in compliance with the License. You may obtain a copy of the License at:",
              "type": "string"
            },
            "licenseUrl": {
              "description": "http://www.apache.org/licenses/LICENSE-2.0",
              "type": "string"
            },
            "asIsClause": {
              "description": "Unless required by applicable law or agreed to in writing, software distributed
under the License is distributed on an \u0027AS IS\u0027 BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND,
either express or implied.",
              "type": "string"
            },
            "permissionsAndLimitations": {
              "description": "See the License for the specific language governing permissions and limitations
under the License.",
              "type": "string"
            }
          }
        },
        "codecsInUse": {
          "description": "number of times an identified codec was used over the measurementInterval",
          "type": "object",
          "properties": {
            "codecIdentifier": {
              "type": "string"
            },
            "numberInUse": {
              "type": "number"
            }
          },
          "required": [
            "codecIdentifier",
            "numberInUse"
          ]
        },
        "command": {
          "description": "command from an event collector toward an event source",
          "type": "object",
          "properties": {
            "commandType": {
              "type": "string",
```

```
              "enum": [
                "heartbeatIntervalChange",
                "measurementIntervalChange",
                "provideThrottlingState",
                "throttlingSpecification"
              ]
            },
            "eventDomainThrottleSpecification": {
              "$ref": "#/definitions/eventDomainThrottleSpecification"
            },
            "measurementInterval": {
              "type": "number"
            }
          },
          "required": [
            "commandType"
          ]
        },
        "commandList": {
          "description": "array of commands from an event collector toward an event source",
          "type": "array",
          "items": {
            "$ref": "#/definitions/commandListEntry"
          },
          "minItems": 0
        },
        "commandListEntry": {
          "description": "reference to a command object",
          "type": "object",
          "properties": {
            "command": {
              "$ref": "#/definitions/command"
            }
          },
          "required": [
            "command"
          ]
        },
        "commonEventHeader": {
          "description": "fields common to all events",
          "type": "object",
          "properties": {
            "domain": {
              "description": "the eventing domain associated with the event",
              "type": "string",
              "enum": [
                "fault",
                "heartbeat",
                "measurementsForVfScaling",
                "mobileFlow",
                "other",
                "stateChange",
                "syslog",
                "thresholdCrossingAlert"
              ]
            },
            "eventId": {
              "description": "event key that is unique to the event source",
              "type": "string"
            },
            "eventType": {
              "description": "unique event topic name",
              "type": "string"
            },
            "functionalRole": {
              "description": "function of the event source e.g., eNodeB, MME, PCRF",
              "type": "string"
            },
            "internalHeaderFields": {
              "$ref": "#/definitions/internalHeaderFields"
            },
            "lastEpochMicrosec": {
              "description": "the latest unix time aka epoch time associated with the event from any component--
as microseconds elapsed since 1 Jan 1970 not including leap seconds",
              "type": "number"
            },
            "priority": {
              "description": "processing priority",
              "type": "string",
              "enum": [
                "High",
                "Medium",
                "Normal",
                "Low"
```

```
              ]
            },
            "reportingEntityId": {
              "description": "UUID identifying the entity reporting the event, for example an OAM VM; must be
populated by the ATT enrichment process",
              "type": "string"
            },
            "reportingEntityName": {
              "description": "name of the entity reporting the event, for example, an OAM VM",
              "type": "string"
            },
            "sequence": {
              "description": "ordering of events communicated by an event source instance or 0 if not needed",
              "type": "integer"
            },
            "sourceId": {
              "description": "UUID identifying the entity experiencing the event issue; must be populated by the
ATT enrichment process",
              "type": "string"
            },
            "sourceName": {
              "description": "name of the entity experiencing the event issue",
              "type": "string"
            },
            "startEpochMicrosec": {
              "description": "the earliest unix time aka epoch time associated with the event from any component--
as microseconds elapsed since 1 Jan 1970 not including leap seconds",
              "type": "number"
            },
            "version": {
              "description": "version of the event header",
              "type": "number"
            }
          },
          "required": [
            "domain",
            "eventId",
            "functionalRole",
            "lastEpochMicrosec",
            "priority",
            "reportingEntityName",
            "sequence",
            "sourceName",
            "startEpochMicrosec"
          ]
        },
        "counter": {
          "description": "performance counter",
          "type": "object",
          "properties": {
            "criticality": {
              "type": "string",
              "enum": [
                "CRIT",
                "MAJ"
              ]
            },
            "name": {
              "type": "string"
            },
            "thresholdCrossed": {
              "type": "string"
            },
            "value": {
              "type": "string"
            }
          },
          "required": [
            "criticality",
            "name",
            "thresholdCrossed",
            "value"
          ]
        },
        "cpuUsage": {
          "description": "percent usage of an identified CPU",
          "type": "object",
          "properties": {
            "cpuIdentifier": {
              "type": "string"
            },
            "percentUsage": {
              "type": "number"
            }
```

```
          },
          "required": [
            "cpuIdentifier",
            "percentUsage"
          ]
        },
        "errors": {
          "description": "receive and transmit errors for the measurements domain",
          "type": "object",
          "properties": {
            "receiveDiscards": {
              "type": "number"
            },
            "receiveErrors": {
              "type": "number"
            },
            "transmitDiscards": {
              "type": "number"
            },
            "transmitErrors": {
              "type": "number"
            }
          },
          "required": [
            "receiveDiscards",
            "receiveErrors",
            "transmitDiscards",
            "transmitErrors"
          ]
        },
        "event": {
          "description": "the root level of the common event format",
          "type": "object",
          "properties": {
            "commonEventHeader": {
              "$ref": "#/definitions/commonEventHeader"
            },
            "faultFields": {
              "$ref": "#/definitions/faultFields"
            },
            "measurementsForVfScalingFields": {
              "$ref": "#/definitions/measurementsForVfScalingFields"
            },
            "mobileFlowFields": {
              "$ref": "#/definitions/mobileFlowFields"
            },
            "otherFields": {
              "$ref": "#/definitions/otherFields"
            },
            "stateChangeFields": {
              "$ref": "#/definitions/stateChangeFields"
            },
            "syslogFields": {
              "$ref": "#/definitions/syslogFields"
            },
            "thresholdCrossingAlertFields": {
              "$ref": "#/definitions/thresholdCrossingAlertFields"
            }
          },
          "required": [
            "commonEventHeader"
          ]
        },
        "eventDomainThrottleSpecification": {
          "description": "specification of what information to suppress within an event domain",
          "type": "object",
          "properties": {
            "eventDomain": {
              "description": "Event domain enum from the commonEventHeader domain field",
              "type": "string"
            },
            "suppressedFieldNames": {
              "description": "List of optional field names in the event block that should not be sent to the
Event Listener",
              "type": "array",
              "items": {
                "type": "string"
              }
            },
            "suppressedNvPairsList": {
              "description": "Optional list of specific NvPairsNames to suppress within a given Name-Value Field",
              "type": "array",
              "items": {
                "$ref": "#/definitions/suppressedNvPairs"
```

```
            }
          }
        },
        "required": [
          "eventDomain"
        ]
      },
      "eventDomainThrottleSpecificationList": {
        "description": "array of eventDomainThrottleSpecifications",
        "type": "array",
        "items": {
          "$ref": "#/definitions/eventDomainThrottleSpecification"
        },
        "minItems": 0
      },
      "eventList": {
        "description": "array of events",
        "type": "array",
        "items": {
          "$ref": "#/definitions/event"
        }
      },
      "eventThrottlingState": {
        "description": "reports the throttling in force at the event source",
        "type": "object",
        "properties": {
          "eventThrottlingMode": {
            "description": "Mode the event manager is in",
            "type": "string",
            "enum": [
              "normal",
              "throttled"
            ]
          },
          "eventDomainThrottleSpecificationList": {
            "$ref": "#/definitions/eventDomainThrottleSpecificationList"
          }
        },
        "required": [
          "eventThrottlingMode"
        ]
      },
      "faultFields": {
        "description": "fields specific to fault events",
        "type": "object",
        "properties": {
          "alarmAdditionalInformation": {
            "description": "additional alarm information",
            "type": "array",
            "items": {
              "$ref": "#/definitions/field"
            }
          },
          "alarmCondition": {
            "description": "alarm condition reported by the device",
            "type": "string"
          },
          "alarmInterfaceA": {
            "description": "card, port, channel or interface name of the device generating the alarm",
            "type": "string"
          },
          "eventSeverity": {
            "description": "event severity or priority",
            "type": "string",
            "enum": [
              "CRITICAL",
              "MAJOR",
              "MINOR",
              "WARNING",
              "NORMAL"
            ]
          },
          "eventSourceType": {
            "description": "type of event source; examples: other, router, switch, host, card, port,
slotThreshold, portThreshold, virtualMachine, virtualNetworkFunction",
            "type": "string"
          },
          "faultFieldsVersion": {
            "description": "version of the faultFields block",
            "type": "number"
          },
          "specificProblem": {
            "description": "short description of the alarm or problem",
            "type": "string"
```

```
      },
      "vfStatus": {
        "description": "virtual function status enumeration",
        "type": "string",
        "enum": [
          "Active",
          "Idle",
          "Preparing to terminate",
          "Ready to terminate",
          "Requesting termination"
        ]
      }
    },
    "required": [
      "alarmCondition",
      "eventSeverity",
      "eventSourceType",
      "specificProblem",
      "vfStatus"
    ]
  },
  "featuresInUse": {
    "description": "number of times an identified feature was used over the measurementInterval",
    "type": "object",
    "properties": {
      "featureIdentifier": {
        "type": "string"
      },
      "featureUtilization": {
        "type": "number"
      }
    },
    "required": [
      "featureIdentifier",
      "featureUtilization"
    ]
  },
  "field": {
    "description": "name value pair",
    "type": "object",
    "properties": {
      "name": {
        "type": "string"
      },
      "value": {
        "type": "string"
      }
    },
    "required": [
      "name",
      "value"
    ]
  },
  "filesystemUsage": {
    "description": "disk usage of an identified virtual machine in gigabytes and/or gigabytes per second",
    "type": "object",
    "properties": {
      "blockConfigured": {
        "type": "number"
      },
      "blockIops": {
        "type": "number"
      },
      "blockUsed": {
        "type": "number"
      },
      "ephemeralConfigured": {
        "type": "number"
      },
      "ephemeralIops": {
        "type": "number"
      },
      "ephemeralUsed": {
        "type": "number"
      },
      "filesystemName": {
        "type": "string"
      }
    },
    "required": [
      "blockConfigured",
      "blockIops",
      "blockUsed",
      "ephemeralConfigured",
```

```
            "ephemeralIops",
            "ephemeralUsed",
            "filesystemName"
          ]
        },
        "gtpPerFlowMetrics": {
          "description": "Mobility GTP Protocol per flow metrics",
          "type": "object",
          "properties": {
            "avgBitErrorRate": {
              "description": "average bit error rate",
              "type": "number"
            },
            "avgPacketDelayVariation": {
              "description": "Average packet delay variation or jitter in milliseconds for received packets:
Average difference between the packet timestamp and time received for all pairs of consecutive packets",
              "type": "number"
            },
            "avgPacketLatency": {
              "description": "average delivery latency",
              "type": "number"
            },
            "avgReceiveThroughput": {
              "description": "average receive throughput",
              "type": "number"
            },
            "avgTransmitThroughput": {
              "description": "average transmit throughput",
              "type": "number"
            },
            "durConnectionFailedStatus": {
              "description": "duration of failed state in milliseconds, computed as the cumulative time between a
failed echo request and the next following successful error request, over this reporting interval",
              "type": "number"
            },
            "durTunnelFailedStatus": {
              "description": "Duration of errored state, computed as the cumulative time between a tunnel error
indicator and the next following non-errored indicator, over this reporting interval",
              "type": "number"
            },
            "flowActivatedBy": {
              "description": "Endpoint activating the flow",
              "type": "string"
            },
            "flowActivationEpoch": {
              "description": "Time the connection is activated in the flow (connection) being reported on, or
transmission time of the first packet if activation time is not available",
              "type": "number"
            },
            "flowActivationMicrosec": {
              "description": "Integer microseconds for the start of the flow connection",
              "type": "number"
            },
            "flowActivationTime": {
              "description": "time the connection is activated in the flow being reported on, or transmission
time of the first packet if activation time is not available; with RFC 2822 compliant format: Sat, 13 Mar 2010 11:
29:05 -0800",
              "type": "string"
            },
            "flowDeactivatedBy": {
              "description": "Endpoint deactivating the flow",
              "type": "string"
            },
            "flowDeactivationEpoch": {
              "description": "Time for the start of the flow connection, in integer UTC epoch time aka UNIX time",
              "type": "number"
            },
            "flowDeactivationMicrosec": {
              "description": "Integer microseconds for the start of the flow connection",
              "type": "number"
            },
            "flowDeactivationTime": {
              "description": "Transmission time of the first packet in the flow connection being reported on;
with RFC 2822 compliant format: Sat, 13 Mar 2010 11:29:05 -0800",
              "type": "string"
            },
            "flowStatus": {
              "description": "connection status at reporting time as a working / inactive / failed indicator
value",
              "type": "string"
            },
            "gtpConnectionStatus": {
              "description": "Current connection state at reporting time",
              "type": "string"
```

```
            },
            "gtpTunnelStatus": {
              "description": "Current tunnel state  at reporting time",
              "type": "string"
            },
            "ipTosCountList": {
              "description": "array of key: value pairs where the keys are drawn from the IP Type-of-Service
identifiers which range from \u00270\u0027 to \u0027255\u0027, and the values are the count of packets that had
those ToS identifiers in the flow",
              "type": "array",
              "items": {
                "type": "array",
                "items": [
                  {
                    "type": "string"
                  },
                  {
                    "type": "number"
                  }
                ]
              }
            },
            "ipTosList": {
              "description": "Array of unique IP Type-of-Service values observed in the flow where values range
from \u00270\u0027 to \u0027255\u0027",
              "type": "array",
              "items": {
                "type": "string"
              }
            },
            "largePacketRtt": {
              "description": "large packet round trip time",
              "type": "number"
            },
            "largePacketThreshold": {
              "description": "large packet threshold being applied",
              "type": "number"
            },
            "maxPacketDelayVariation": {
              "description": "Maximum packet delay variation or jitter in milliseconds for received packets:
Maximum of the difference between the packet timestamp and time received for all pairs of consecutive packets",
              "type": "number"
            },
            "maxReceiveBitRate": {
              "description": "maximum receive bit rate",
              "type": "number"
            },
            "maxTransmitBitRate": {
              "description": "maximum transmit bit rate",
              "type": "number"
            },
            "mobileQciCosCountList": {
              "description": "array of key: value pairs where the keys are drawn from LTE QCI or UMTS class of
service strings, and the values are the count of packets that had those strings in the flow",
              "type": "array",
              "items": {
                "type": "array",
                "items": [
                  {
                    "type": "string"
                  },
                  {
                    "type": "number"
                  }
                ]
              }
            },
            "mobileQciCosList": {
              "description": "Array of unique LTE QCI or UMTS class-of-service values observed in the flow",
              "type": "array",
              "items": {
                "type": "string"
              }
            },
            "numActivationFailures": {
              "description": "Number of failed activation requests, as observed by the reporting node",
              "type": "number"
            },
            "numBitErrors": {
              "description": "number of errored bits",
              "type": "number"
            },
            "numBytesReceived": {
              "description": "number of bytes received, including retransmissions",
```

```
            "type": "number"
          },
          "numBytesTransmitted": {
            "description": "number of bytes transmitted, including retransmissions",
            "type": "number"
          },
          "numDroppedPackets": {
            "description": "number of received packets dropped due to errors per virtual interface",
            "type": "number"
          },
          "numGtpEchoFailures": {
            "description": "Number of Echo request path failures where failed paths are defined in 3GPP TS
29.281 sec 7.2.1 and 3GPP TS 29.060 sec. 11.2",
            "type": "number"
          },
          "numGtpTunnelErrors": {
            "description": "Number of tunnel error indications where errors are defined in 3GPP TS 29.281 sec
7.3.1 and 3GPP TS 29.060 sec. 11.1",
            "type": "number"
          },
          "numHttpErrors": {
            "description": "Http error count",
            "type": "number"
          },
          "numL7BytesReceived": {
            "description": "number of tunneled layer 7 bytes received, including retransmissions",
            "type": "number"
          },
          "numL7BytesTransmitted": {
            "description": "number of tunneled layer 7 bytes transmitted, excluding retransmissions",
            "type": "number"
          },
          "numLostPackets": {
            "description": "number of lost packets",
            "type": "number"
          },
          "numOutOfOrderPackets": {
            "description": "number of out-of-order packets",
            "type": "number"
          },
          "numPacketErrors": {
            "description": "number of errored packets",
            "type": "number"
          },
          "numPacketsReceivedExclRetrans": {
            "description": "number of packets received, excluding retransmission",
            "type": "number"
          },
          "numPacketsReceivedInclRetrans": {
            "description": "number of packets received, including retransmission",
            "type": "number"
          },
          "numPacketsTransmittedInclRetrans": {
            "description": "number of packets transmitted, including retransmissions",
            "type": "number"
          },
          "numRetries": {
            "description": "number of packet retries",
            "type": "number"
          },
          "numTimeouts": {
            "description": "number of packet timeouts",
            "type": "number"
          },
          "numTunneledL7BytesReceived": {
            "description": "number of tunneled layer 7 bytes received, excluding retransmissions",
            "type": "number"
          },
          "roundTripTime": {
            "description": "round trip time",
            "type": "number"
          },
          "tcpFlagCountList": {
            "description": "array of key: value pairs where the keys are drawn from TCP Flags and the values
are the count of packets that had that TCP Flag in the flow",
            "type": "array",
            "items": {
              "type": "array",
              "items": [
                {
                  "type": "string"
                },
                {
                  "type": "number"
```

```
                }
              ]
            }
          },
          "tcpFlagList": {
            "description": "Array of unique TCP Flags observed in the flow",
            "type": "array",
            "items": {
              "type": "string"
            }
          },
          "timeToFirstByte": {
            "description": "Time in milliseconds between the connection activation and first byte received",
            "type": "number"
          }
        },
        "required": [
          "avgBitErrorRate",
          "avgPacketDelayVariation",
          "avgPacketLatency",
          "avgReceiveThroughput",
          "avgTransmitThroughput",
          "flowActivationEpoch",
          "flowActivationMicrosec",
          "flowDeactivationEpoch",
          "flowDeactivationMicrosec",
          "flowDeactivationTime",
          "flowStatus",
          "maxPacketDelayVariation",
          "numActivationFailures",
          "numBitErrors",
          "numBytesReceived",
          "numBytesTransmitted",
          "numDroppedPackets",
          "numL7BytesReceived",
          "numL7BytesTransmitted",
          "numLostPackets",
          "numOutOfOrderPackets",
          "numPacketErrors",
          "numPacketsReceivedExclRetrans",
          "numPacketsReceivedInclRetrans",
          "numPacketsTransmittedInclRetrans",
          "numRetries",
          "numTimeouts",
          "numTunneledL7BytesReceived",
          "roundTripTime",
          "timeToFirstByte"
        ]
      },
      "internalHeaderFields": {
        "description": "enrichment fields for internal VES Event Listener service use only, not supplied by
event sources",
        "type": "object"
      },
      "latencyBucketMeasure": {
        "description": "number of counts falling within a defined latency bucket",
        "type": "object",
        "properties": {
          "countsInTheBucket": {
            "type": "number"
          },
          "highEndOfLatencyBucket": {
            "type": "number"
          },
          "lowEndOfLatencyBucket": {
            "type": "number"
          }
        },
        "required": [
          "countsInTheBucket"
        ]
      },
      "measurementGroup": {
        "description": "measurement group",
        "type": "object",
        "properties": {
          "name": {
            "type": "string"
          },
          "measurements": {
            "description": "array of name value pair measurements",
            "type": "array",
            "items": {
              "$ref": "#/definitions/field"
```

```
            }
          }
        },
        "required": [
          "name",
          "measurements"
        ]
      },
      "measurementsForVfScalingFields": {
        "description": "measurementsForVfScaling fields",
        "type": "object",
        "properties": {
          "additionalMeasurements": {
            "description": "additional measurement fields",
            "type": "array",
            "items": {
              "$ref": "#/definitions/measurementGroup"
            }
          },
          "aggregateCpuUsage": {
            "description": "aggregate CPU usage of the VM on which the VNFC reporting the event is running",
            "type": "number"
          },
          "codecUsageArray": {
            "description": "array of codecs in use",
            "type": "array",
            "items": {
              "$ref": "#/definitions/codecsInUse"
            }
          },
          "concurrentSessions": {
            "description": "peak concurrent sessions for the VM or VNF over the measurementInterval",
            "type": "number"
          },
          "configuredEntities": {
            "description": "over the measurementInterval, peak total number of: users, subscribers, devices,
adjacencies, etc., for the VM, or subscribers, devices, etc., for the VNF",
            "type": "number"
          },
          "cpuUsageArray": {
            "description": "usage of an array of CPUs",
            "type": "array",
            "items": {
              "$ref": "#/definitions/cpuUsage"
            }
          },
          "errors": {
            "$ref": "#/definitions/errors"
          },
          "featureUsageArray": {
            "description": "array of features in use",
            "type": "array",
            "items": {
              "$ref": "#/definitions/featuresInUse"
            }
          },
          "filesystemUsageArray": {
            "description": "filesystem usage of the VM on which the VNFC reporting the event is running",
            "type": "array",
            "items": {
              "$ref": "#/definitions/filesystemUsage"
            }
          },
          "latencyDistribution": {
            "description": "array of integers representing counts of requests whose latency in milliseconds
falls within per-VNF configured ranges",
            "type": "array",
            "items": {
              "$ref": "#/definitions/latencyBucketMeasure"
            }
          },
          "meanRequestLatency": {
            "description": "mean seconds required to respond to each request for the VM on which the VNFC
reporting the event is running",
            "type": "number"
          },
          "measurementInterval": {
            "description": "interval over which measurements are being reported in seconds",
            "type": "number"
          },
          "measurementsForVfScalingVersion": {
            "description": "version of the measurementsForVfScaling block",
            "type": "number"
          },
```

```
            "memoryConfigured": {
              "description": "memory in MB configured in the VM on which the VNFC reporting the event is running",
              "type": "number"
            },
            "memoryUsed": {
              "description": "memory usage in MB of the VM on which the VNFC reporting the event is running",
              "type": "number"
            },
            "numberOfMediaPortsInUse": {
              "description": "number of media ports in use",
              "type": "number"
            },
            "requestRate": {
              "description": "peak rate of service requests per second to the VNF over the measurementInterval",
              "type": "number"
            },
            "vnfcScalingMetric": {
              "description": "represents busy-ness of the VNF from 0 to 100 as reported by the VNFC",
              "type": "number"
            },
            "vNicUsageArray": {
              "description": "usage of an array of virtual network interface cards",
              "type": "array",
              "items": {
                "$ref": "#/definitions/vNicUsage"
              }
            }
          },
          "required": [
            "measurementInterval"
          ]
        },
        "mobileFlowFields": {
          "description": "mobileFlow fields",
          "type": "object",
          "properties": {
            "additionalFields": {
              "description": "additional mobileFlow fields if needed",
              "type": "array",
              "items": {
                "$ref": "#/definitions/field"
              }
            },
            "applicationType": {
              "description": "Application type inferred",
              "type": "string"
            },
            "appProtocolType": {
              "description": "application protocol",
              "type": "string"
            },
            "appProtocolVersion": {
              "description": "application protocol version",
              "type": "string"
            },
            "cid": {
              "description": "cell id",
              "type": "string"
            },
            "connectionType": {
              "description": "Abbreviation referencing a 3GPP reference point e.g., S1-U, S11, etc",
              "type": "string"
            },
            "ecgi": {
              "description": "Evolved Cell Global Id",
              "type": "string"
            },
            "flowDirection": {
              "description": "Flow direction, indicating if the reporting node is the source of the flow or
destination for the flow",
              "type": "string"
            },
            "gtpPerFlowMetrics": {
              "$ref": "#/definitions/gtpPerFlowMetrics"
            },
            "gtpProtocolType": {
              "description": "GTP protocol",
              "type": "string"
            },
            "gtpVersion": {
              "description": "GTP protocol version",
              "type": "string"
            },
            "httpHeader": {
```

```
              "description": "HTTP request header, if the flow connects to a node referenced by HTTP",
              "type": "string"
            },
            "imei": {
              "description": "IMEI for the subscriber UE used in this flow, if the flow connects to a mobile
device",
              "type": "string"
            },
            "imsi": {
              "description": "IMSI for the subscriber UE used in this flow, if the flow connects to a mobile
device",
              "type": "string"
            },
            "ipProtocolType": {
              "description": "IP protocol type e.g., TCP, UDP, RTP...",
              "type": "string"
            },
            "ipVersion": {
              "description": "IP protocol version e.g., IPv4, IPv6",
              "type": "string"
            },
            "lac": {
              "description": "location area code",
              "type": "string"
            },
            "mcc": {
              "description": "mobile country code",
              "type": "string"
            },
            "mnc": {
              "description": "mobile network code",
              "type": "string"
            },
            "mobileFlowFieldsVersion": {
              "description": "version of the mobileFlowFields block",
              "type": "number"
            },
            "msisdn": {
              "description": "MSISDN for the subscriber UE used in this flow, as an integer, if the flow connects
to a mobile device",
              "type": "string"
            },
            "otherEndpointIpAddress": {
              "description": "IP address for the other endpoint, as used for the flow being reported on",
              "type": "string"
            },
            "otherEndpointPort": {
              "description": "IP Port for the reporting entity, as used for the flow being reported on",
              "type": "number"
            },
            "otherFunctionalRole": {
              "description": "Functional role of the other endpoint for the flow being reported on e.g., MME, S-
GW, P-GW, PCRF...",
              "type": "string"
            },
            "rac": {
              "description": "routing area code",
              "type": "string"
            },
            "radioAccessTechnology": {
              "description": "Radio Access Technology e.g., 2G, 3G, LTE",
              "type": "string"
            },
            "reportingEndpointIpAddr": {
              "description": "IP address for the reporting entity, as used for the flow being reported on",
              "type": "string"
            },
            "reportingEndpointPort": {
              "description": "IP port for the reporting entity, as used for the flow being reported on",
              "type": "number"
            },
            "sac": {
              "description": "service area code",
              "type": "string"
            },
            "samplingAlgorithm": {
              "description": "Integer identifier for the sampling algorithm or rule being applied in calculating
the flow metrics if metrics are calculated based on a sample of packets, or 0 if no sampling is applied",
              "type": "number"
            },
            "tac": {
              "description": "transport area code",
              "type": "string"
            },
```

```
            "tunnelId": {
              "description": "tunnel identifier",
              "type": "string"
            },
            "vlanId": {
              "description": "VLAN identifier used by this flow",
              "type": "string"
            }
          },
          "required": [
            "flowDirection",
            "gtpPerFlowMetrics",
            "ipProtocolType",
            "ipVersion",
            "otherEndpointIpAddress",
            "otherEndpointPort",
            "reportingEndpointIpAddr",
            "reportingEndpointPort"
          ]
        },
        "otherFields": {
          "description": "additional fields not reported elsewhere",
          "type": "array",
          "items": {
            "$ref": "#/definitions/field"
          }
        },
        "requestError": {
          "description": "standard request error data structure",
          "type": "object",
          "properties": {
            "messageId": {
              "description": "Unique message identifier of the format ABCnnnn where ABC is either SVC for Service
Exceptions or POL for Policy Exception",
              "type": "string"
            },
            "text": {
              "description": "Message text, with replacement variables marked with %n, where n is an index into
the list of \u003cvariables\u003e elements, starting at 1",
              "type": "string"
            },
            "url": {
              "description": "Hyperlink to a detailed error resource e.g., an HTML page for browser user agents",
              "type": "string"
            },
            "variables": {
              "description": "List of zero or more strings that represent the contents of the variables used by
the message text",
              "type": "string"
            }
          },
          "required": [
            "messageId",
            "text"
          ]
        },
        "stateChangeFields": {
          "description": "stateChange fields",
          "type": "object",
          "properties": {
            "additionalFields": {
              "description": "additional stateChange fields if needed",
              "type": "array",
              "items": {
                "$ref": "#/definitions/field"
              }
            },
            "newState": {
              "description": "new state of the entity",
              "type": "string",
              "enum": [
                "inService",
                "maintenance",
                "outOfService"
              ]
            },
            "oldState": {
              "description": "previous state of the entity",
              "type": "string",
              "enum": [
                "inService",
                "maintenance",
                "outOfService"
              ]
```

```
            },
            "stateChangeFieldsVersion": {
              "description": "version of the stateChangeFields block",
              "type": "number"
            },
            "stateInterface": {
              "description": "card or port name of the entity that changed state",
              "type": "string"
            }
          },
          "required": [
            "newState",
            "oldState",
            "stateInterface"
          ]
        },
        "suppressedNvPairs": {
          "description": "List of specific NvPairsNames to suppress within a given Name-Value Field for event
Throttling",
          "type": "object",
          "properties": {
            "nvPairFieldName": {
              "description": "Name of the field within which are the nvpair names to suppress",
              "type": "string"
            },
            "suppressedNvPairNames": {
              "description": "Array of nvpair names to suppress within the nvpairFieldName",
              "type": "array",
              "items": {
                "type": "string"
              }
            }
          },
          "required": [
            "nvPairFieldName",
            "suppressedNvPairNames"
          ]
        },
        "syslogFields": {
          "description": "sysLog fields",
          "type": "object",
          "properties": {
            "additionalFields": {
              "description": "additional syslog fields if needed",
              "type": "array",
              "items": {
                "$ref": "#/definitions/field"
              }
            },
            "eventSourceHost": {
              "description": "hostname of the device",
              "type": "string"
            },
            "eventSourceType": {
              "description": "type of event source; examples: other, router, switch, host, card, port,
slotThreshold, portThreshold, virtualMachine, virtualNetworkFunction",
              "type": "string"
            },
            "syslogFacility": {
              "description": "numeric code from 0 to 23 for facility--see table in documentation",
              "type": "number"
            },
            "syslogFieldsVersion": {
              "description": "version of the syslogFields block",
              "type": "number"
            },
            "syslogMsg": {
              "description": "syslog message",
              "type": "string"
            },
            "syslogPri": {
              "description": "0-192 combined severity and facility",
              "type": "number"
            },
            "syslogProc": {
              "description": "identifies the application that originated the message",
              "type": "string"
            },
            "syslogProcId": {
              "description": "a change in the value of this field indicates a discontinuity in syslog reporting",
              "type": "number"
            },
            "syslogSData": {
              "description": "syslog structured data consisting of a structured data Id followed by a set of key
```

```
value pairs",
              "type": "string"
            },
            "syslogSdId": {
              "description": "0-32 char in format name@number for example ourSDID@32473",
              "type": "string"
            },
            "syslogSev": {
              "description": "numerical Code for  severity derived from syslogPri as remaider of syslogPri / 8",
              "type": "string"
            },
            "syslogTag": {
              "description": "msgId indicating the type of message such as TCPOUT or TCPIN; NILVALUE should be
used when no other value can be provided",
              "type": "string"
            },
            "syslogVer": {
              "description": "IANA assigned version of the syslog protocol specification - typically 1",
              "type": "number"
            }
          },
          "required": [
            "eventSourceType",
            "syslogMsg",
            "syslogTag"
          ]
        },
        "thresholdCrossingAlertFields": {
          "description": "fields specific to threshold crossing alert events",
          "type": "object",
          "properties": {
            "additionalFields": {
              "description": "additional threshold crossing alert fields if needed",
              "type": "array",
              "items": {
                "$ref": "#/definitions/field"
              }
            },
            "additionalParameters": {
              "description": "performance counters",
              "type": "array",
              "items": {
                "$ref": "#/definitions/counter"
              }
            },
            "alertAction": {
              "description": "Event action",
              "type": "string",
              "enum": [
                "CLEAR",
                "CONT",
                "SET"
              ]
            },
            "alertDescription": {
              "description": "Unique short alert description such as IF-SHUB-ERRDROP",
              "type": "string"
            },
            "alertType": {
              "description": "Event type",
              "type": "string",
              "enum": [
                "CARD-ANOMALY",
                "ELEMENT-ANOMALY",
                "INTERFACE-ANOMALY",
                "SERVICE-ANOMALY"
              ]
            },
            "alertValue": {
              "description": "Calculated API value (if applicable)",
              "type": "string"
            },
            "associatedAlertIdList": {
              "description": "List of eventIds associated with the event being reported",
              "type": "array",
              "items": {
                "type": "string"
              }
            },
            "collectionTimestamp": {
              "description": "Time when the performance collector picked up the data; with RFC 2822 compliant
format: Sat, 13 Mar 2010 11:29:05 -0800",
              "type": "string"
            },
```

```
            "dataCollector": {
              "description": "Specific performance collector instance used",
              "type": "string"
            },
            "elementType": {
              "description": "type of network element - internal ATT field",
              "type": "string"
            },
            "eventSeverity": {
              "description": "event severity or priority",
              "type": "string",
              "enum": [
                "CRITICAL",
                "MAJOR",
                "MINOR",
                "WARNING",
                "NORMAL"
              ]
            },
            "eventStartTimestamp": {
              "description": "Time closest to when the measurement was made; with RFC 2822 compliant format: Sat,
13 Mar 2010 11:29:05 -0800",
              "type": "string"
            },
            "interfaceName": {
              "description": "Physical or logical port or card (if applicable)",
              "type": "string"
            },
            "networkService": {
              "description": "network name - internal ATT field",
              "type": "string"
            },
            "possibleRootCause": {
              "description": "Reserved for future use",
              "type": "string"
            },
            "thresholdCrossingFieldsVersion": {
              "description": "version of the thresholdCrossingAlertFields block",
              "type": "number"
            }
          },
          "required": [
            "additionalParameters",
            "alertAction",
            "alertDescription",
            "alertType",
            "collectionTimestamp",
            "eventSeverity",
            "eventStartTimestamp"
          ]
        },
        "vNicUsage": {
          "description": "usage of identified virtual network interface card",
          "type": "object",
          "properties": {
            "broadcastPacketsIn": {
              "type": "number"
            },
            "broadcastPacketsOut": {
              "type": "number"
            },
            "bytesIn": {
              "type": "number"
            },
            "bytesOut": {
              "type": "number"
            },
            "multicastPacketsIn": {
              "type": "number"
            },
            "multicastPacketsOut": {
              "type": "number"
            },
            "packetsIn": {
              "type": "number"
            },
            "packetsOut": {
              "type": "number"
            },
            "unicastPacketsIn": {
              "type": "number"
            },
            "unicastPacketsOut": {
              "type": "number"
```

```
            },
            "vNicIdentifier": {
              "type": "string"
            }
          },
          "required": [
            "bytesIn",
            "bytesOut",
            "packetsIn",
            "packetsOut",
            "vNicIdentifier"
          ]
        }
      },
      "title": "Event Listener",
      "type": "object",
      "properties": {
        "event": {
          "$ref": "#/definitions/event"
        }
      },
      "eventList": {
        "$ref": "#/definitions/eventList"
      }
    }
  },
  "owner": "onboarding_dev"
}
```

b) To onboard a data format and a component

Each component has a description that tells what it does.

These requests would be of the type-

```
curl -X POST http://<onboardingapi host>/onboarding/dataformats     -H "Content-Type: application/json" -d
@<filepath to request>
curl -X POST http://<onboardingapi host>/onboarding/components     -H "Content-Type: application/json" -d
@<filepath to request>

In our case,
curl -X POST http://dcaemod.simpledemo.onap.org/onboarding/dataformats     -H "Content-Type: application/json" -d
@<filepath to request>
curl -X POST http://dcaemod.simpledemo.onap.org/onboarding/components     -H "Content-Type: application/json" -d
@<filepath to request>
```

You can download the Components and Data Formats for the demo from –

Components:

https://git.onap.org/dcaegen2/collectors/ves/tree/dpo/spec/vescollector-componentspec.json

https://git.onap.org/dcaegen2/analytics/tca-gen2/tree/dcae-analytics/dpo/tcagen2_spec.json

Corresponding Data Formats:

https://git.onap.org/dcaegen2/collectors/ves/tree/dpo/data-formats
https://git.onap.org/dcaegen2/analytics/tca-gen2/tree/dcae-analytics/dpo/

c) Verify the resources were created using

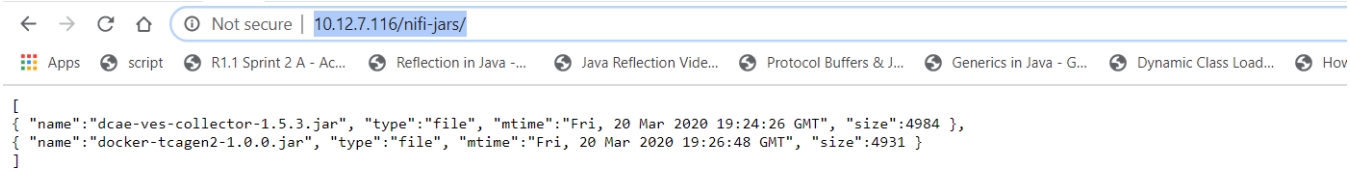
```
curl -X GET http://dcaemod.simpledemo.onap.org/onboarding/dataformats

curl -X GET http://dcaemod.simpledemo.onap.org/onboarding/components
```

d) Verify the genprocessor (which polls onboarding periodically to convert component specs to nifi processor), converted the component

Open http://dcaemod.simpledemo.onap.org/nifi-jars/ in a browser.

These jars should now be available for you to use in the nifi UI as processors.



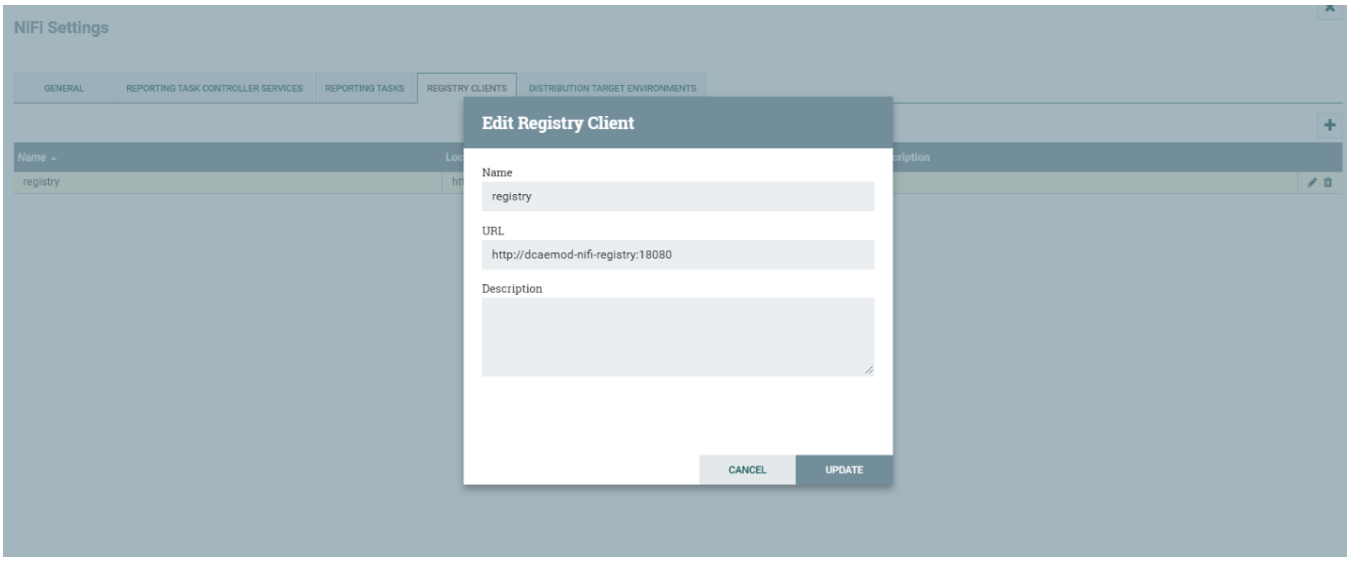

```
[
{ "name":"dcae-ves-collector-1.5.3.jar", "type":"file", "mtime":"Fri, 20 Mar 2020 19:24:26 GMT", "size":4984 },
{ "name":"docker-tcagen2-1.0.0.jar", "type":"file", "mtime":"Fri, 20 Mar 2020 19:26:48 GMT", "size":4931 }
]
```


3. Design & Distribution Flow

a) Configure Nifi Registry url

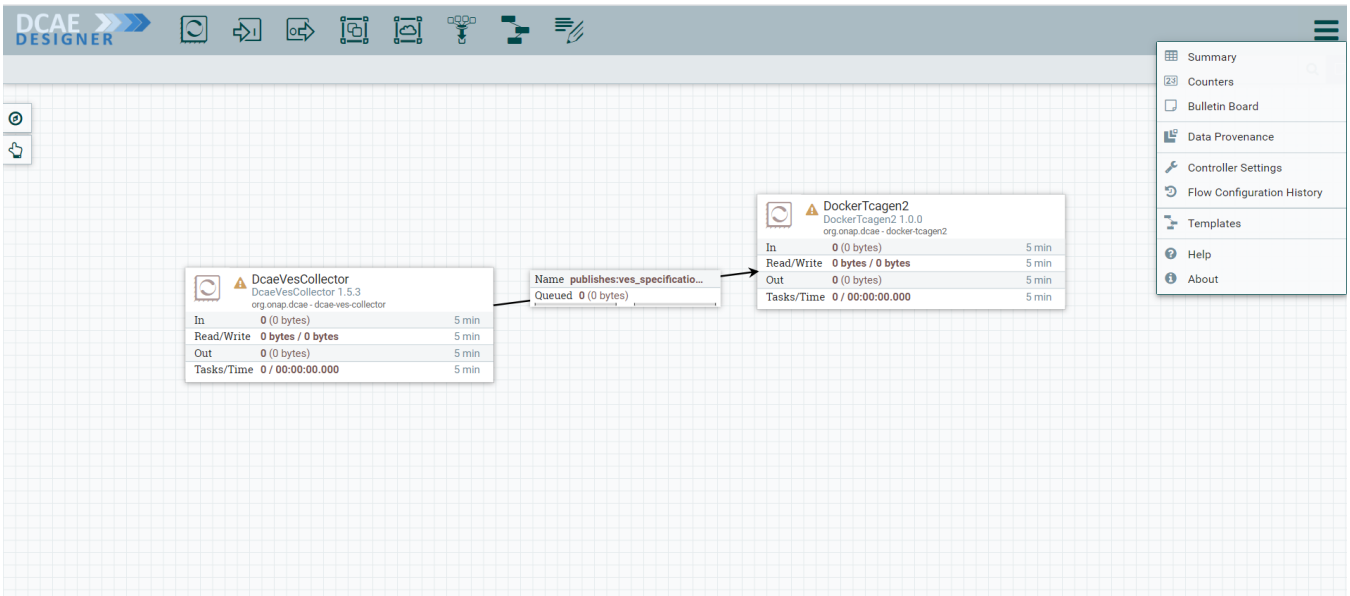
Next check Nifi settings by selecting the Hamburger button in the Nifi UI. It should lead you to the Nifi Settings screen

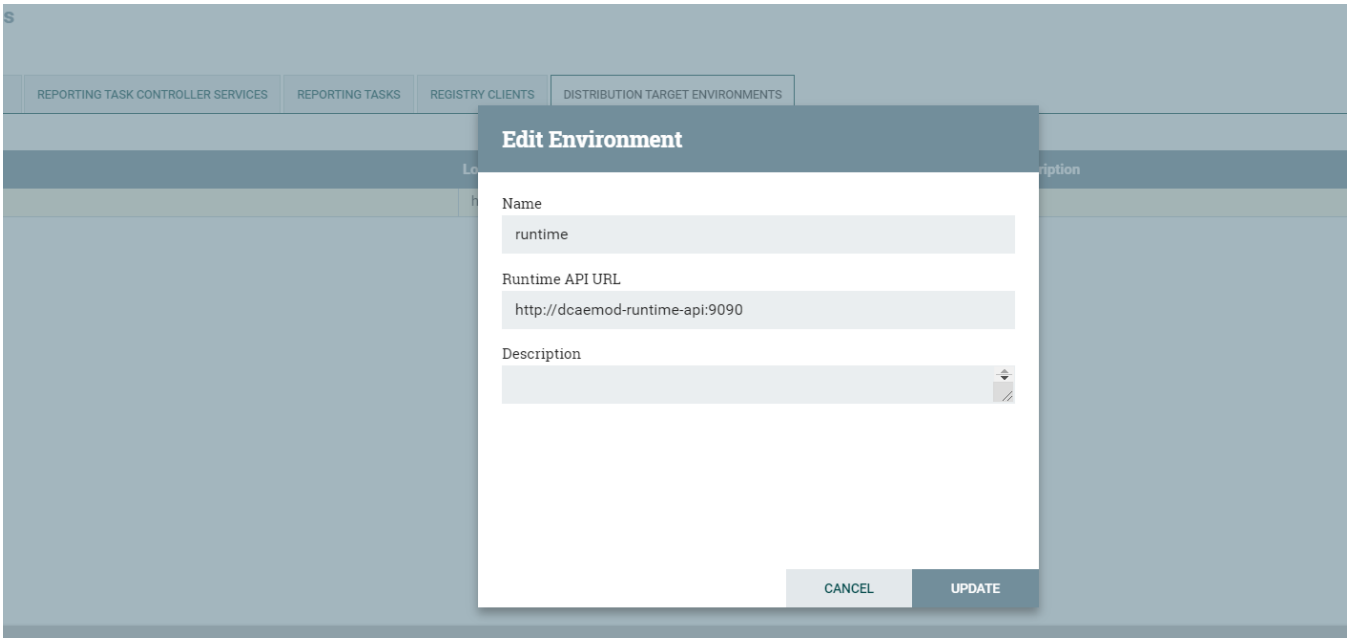
Add a registry client. The Registry client url will be http://dcaemod-nifi-registry:18080



b) Add distribution target which will be the runtime api url

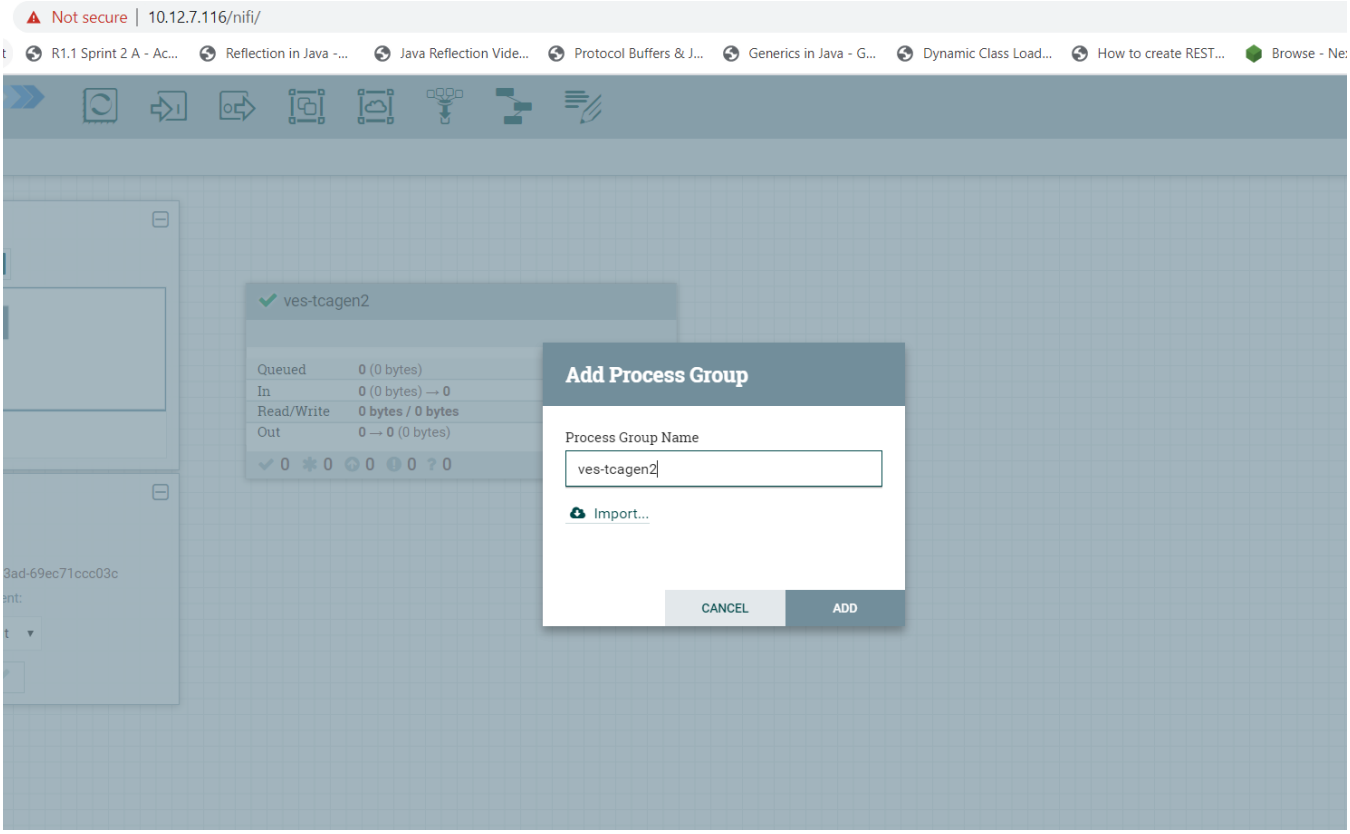
Set the distribution target in the controller settings





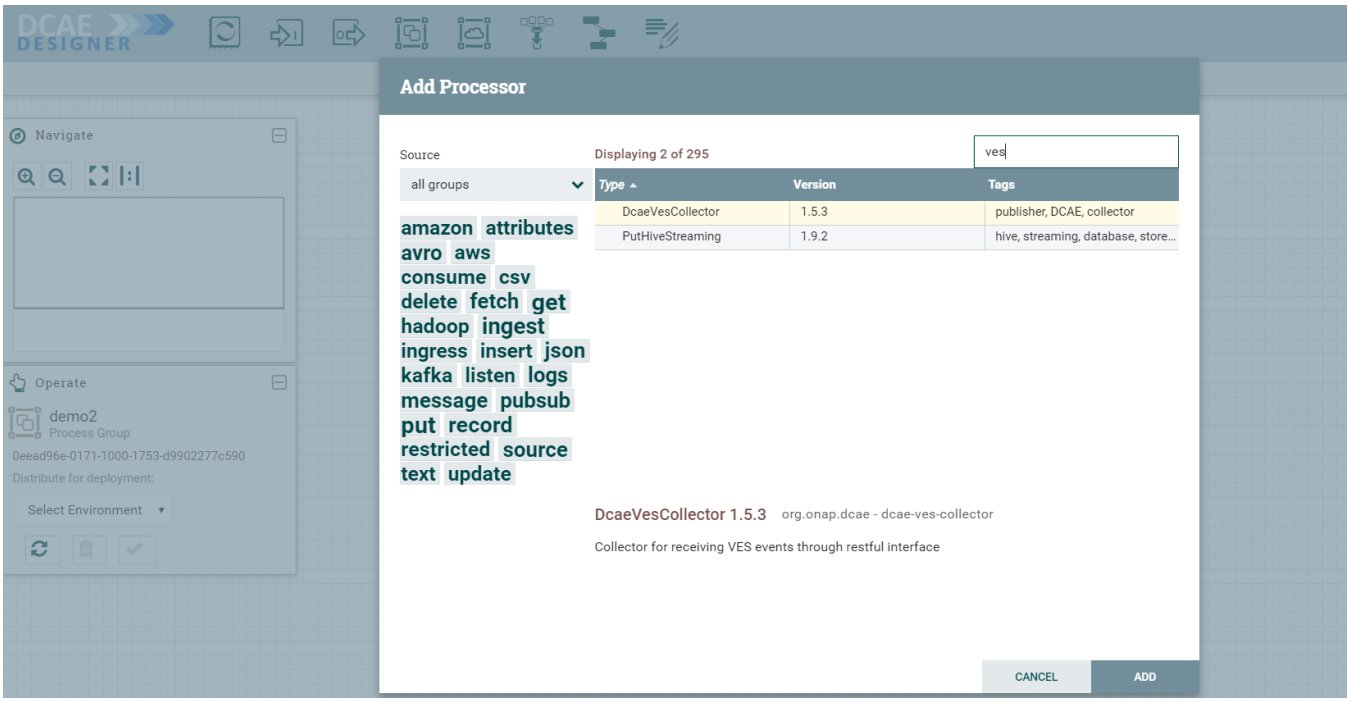
Distribution target URL will be http://dcaemod-runtime-api:9090

c) To start creating flows, we need to create a process group first. The name of the process group will be the name of the flow. Drag and Drop on the canvas, the 'Processor Group' icon from the DCAE Designer bar on the top.

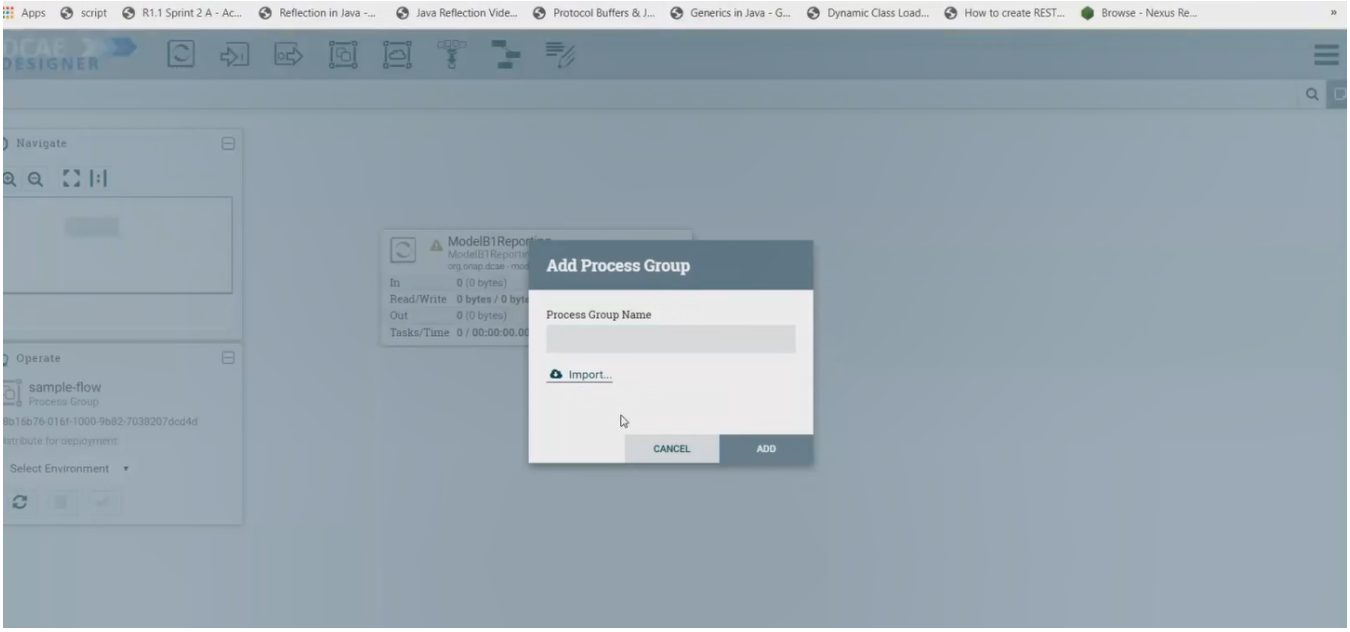


Now enter the process group by double clicking it,

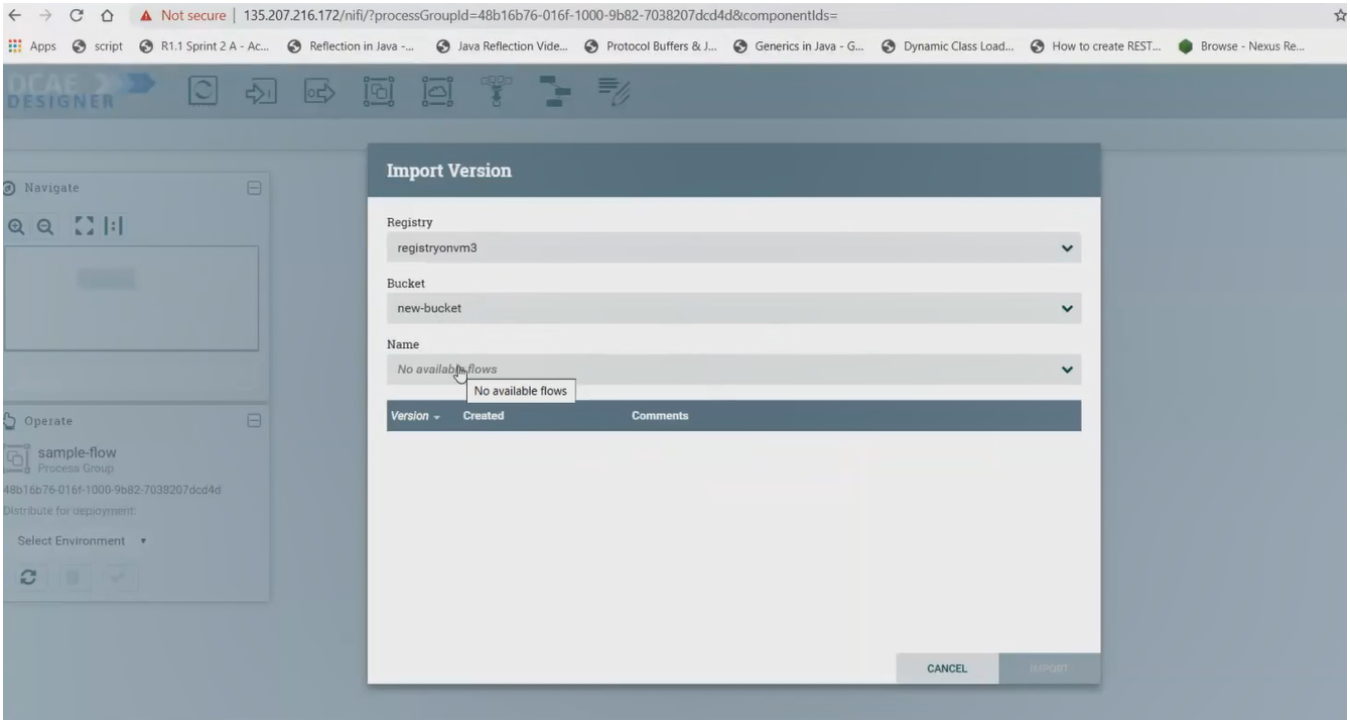
You can now drag and drop on the canvas ‘Processor’ icon from the top DCAE Designer tab. You can search for a particular component in the search box that appears when you attempt to drag the ‘Processor’ icon to the canvas.



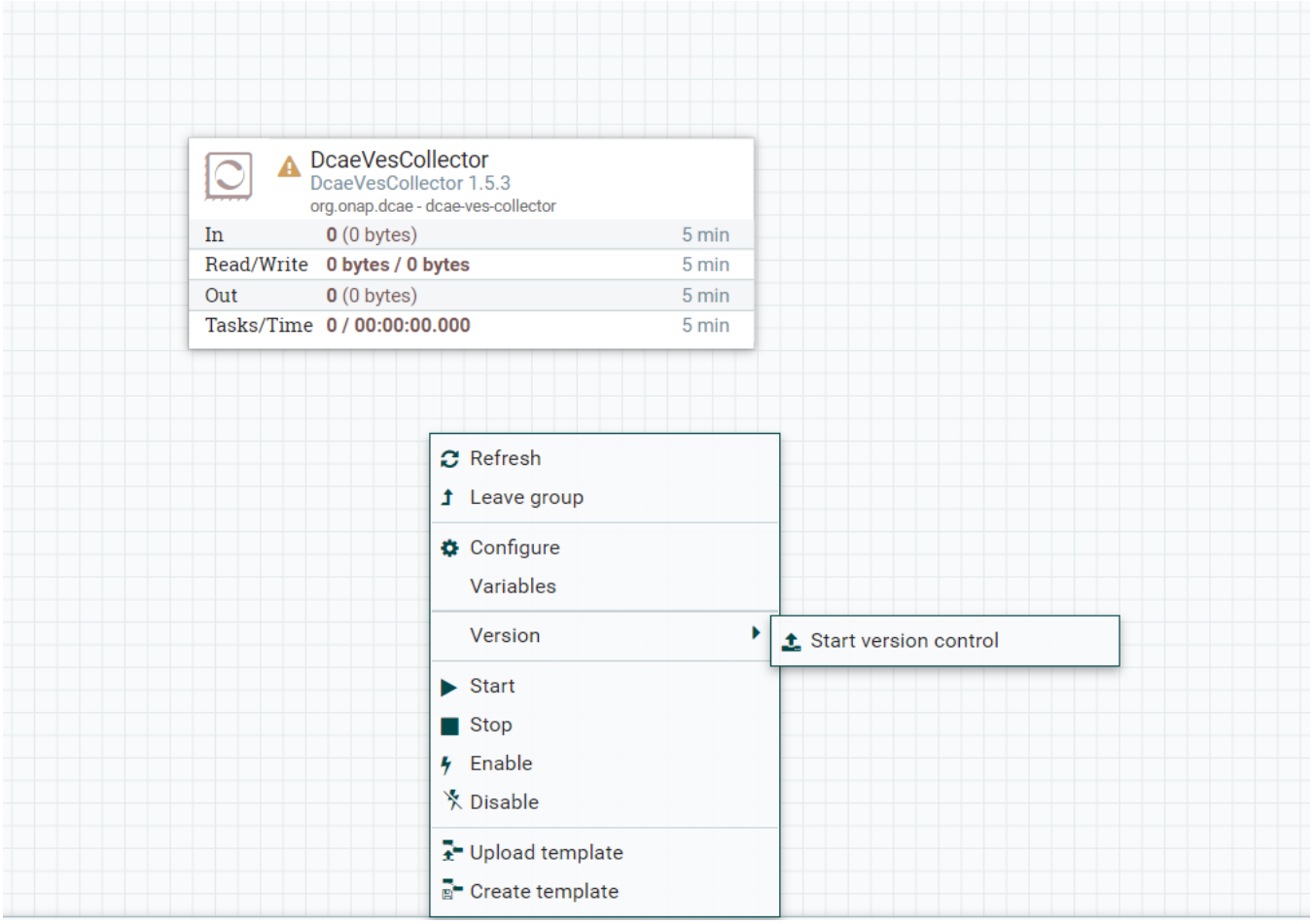
If the Nifi registry linking worked, you should see the “Import” button when you try to add a Processor or Process group to the Nifi canvas, like so-



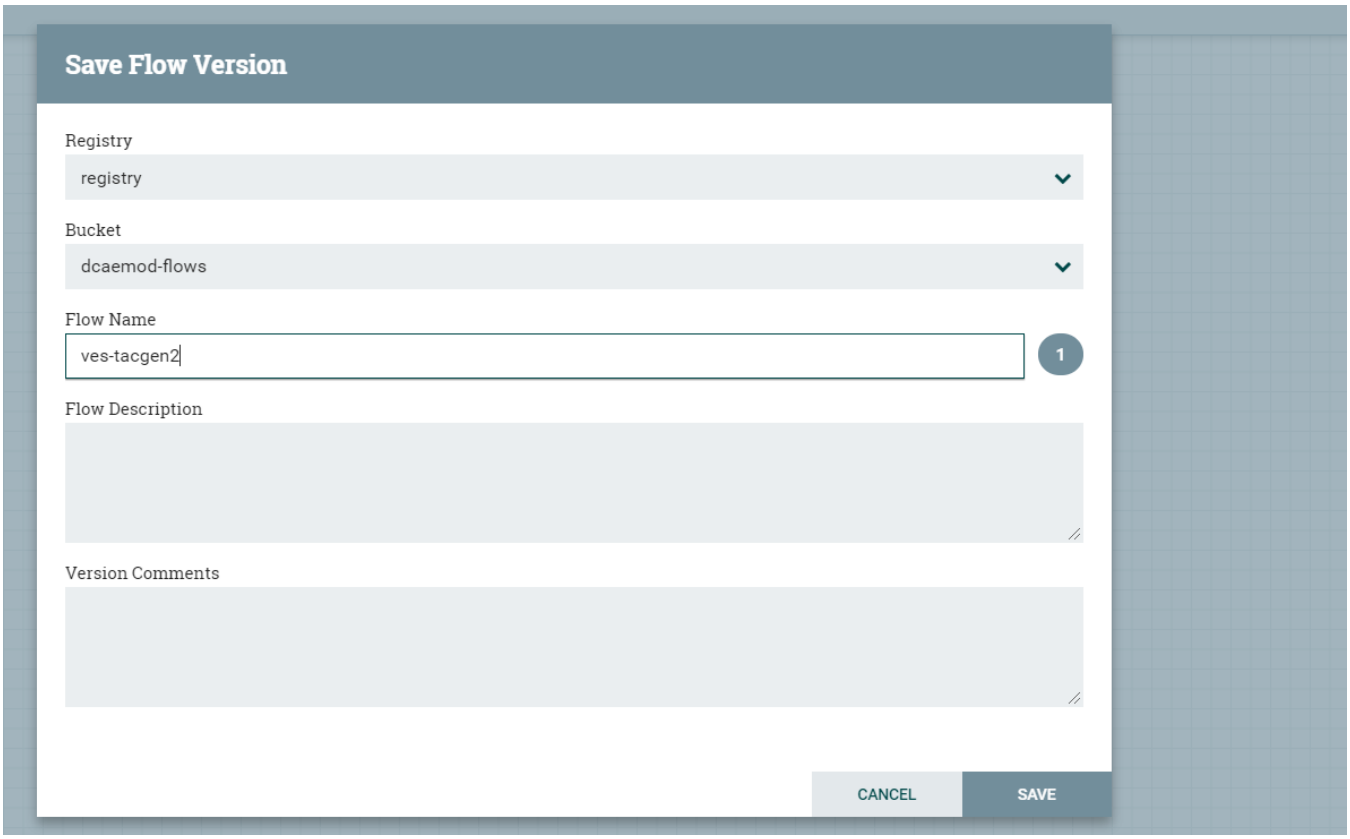
By clicking on the import button, we can import already created saved and version controlled flows from the Nifi registry, if they are present.



We can save created flows by version controlling them like so starting with a 'right click' anywhere on the canvas-



Ideally you would name the flow and process group the same, because functionally they are similar.



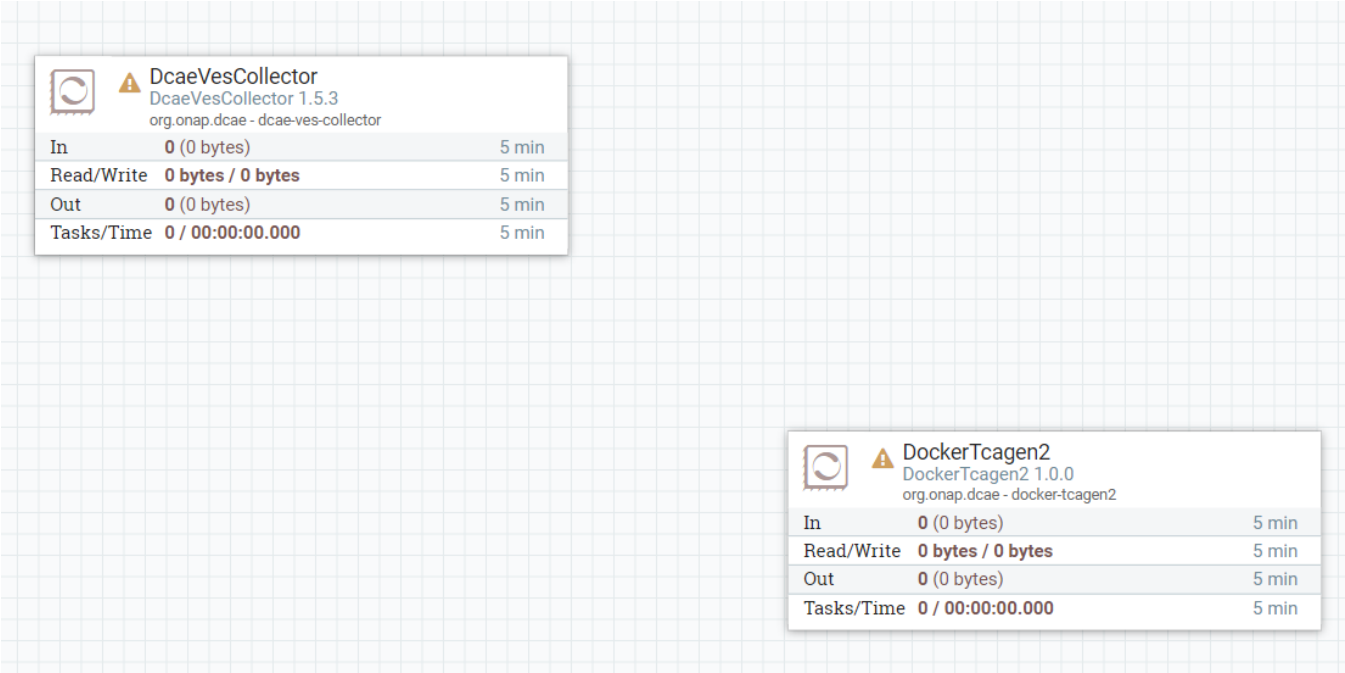
When the flow is checked in, the bar at the bottom shows a green checkmark

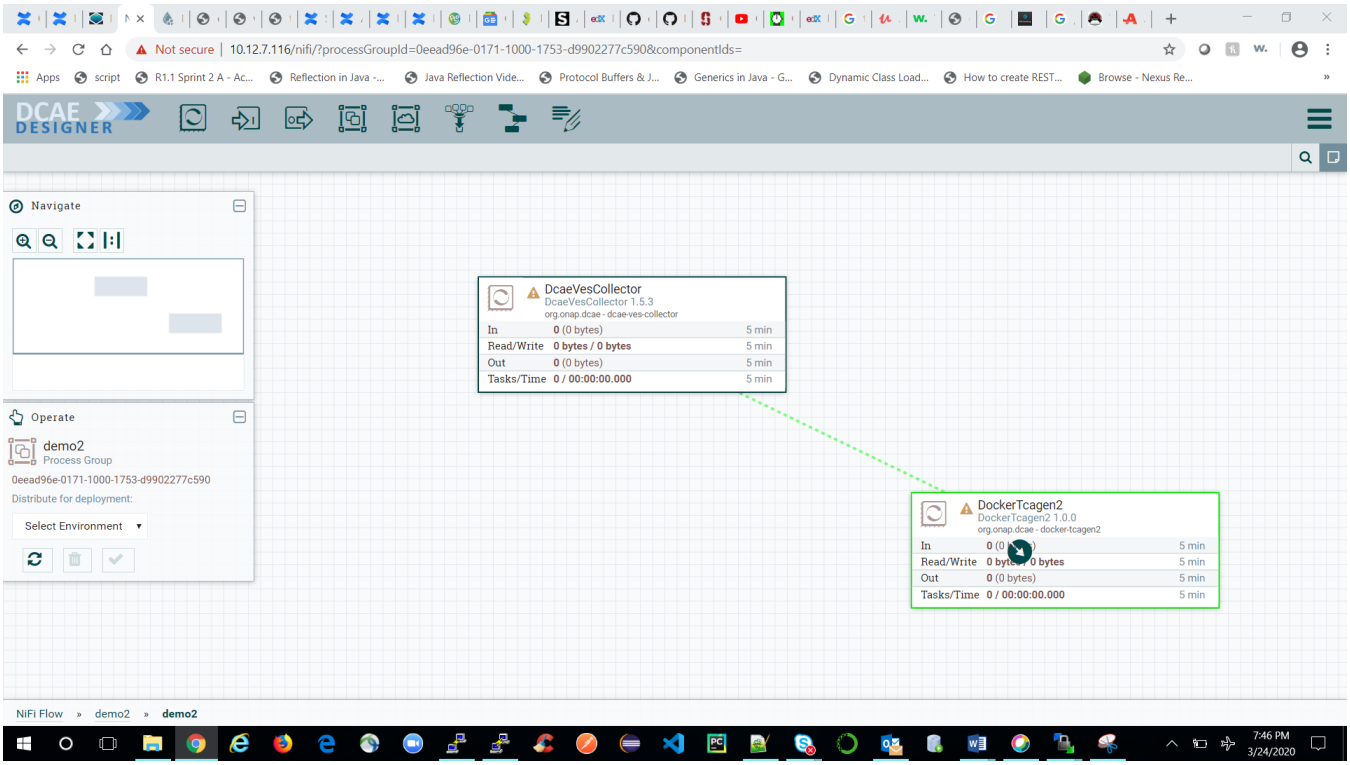
Note: Even if you move a component around on the canvas, and its position on the canvas changes, it is recognized as a change, and it will have to recommitted.

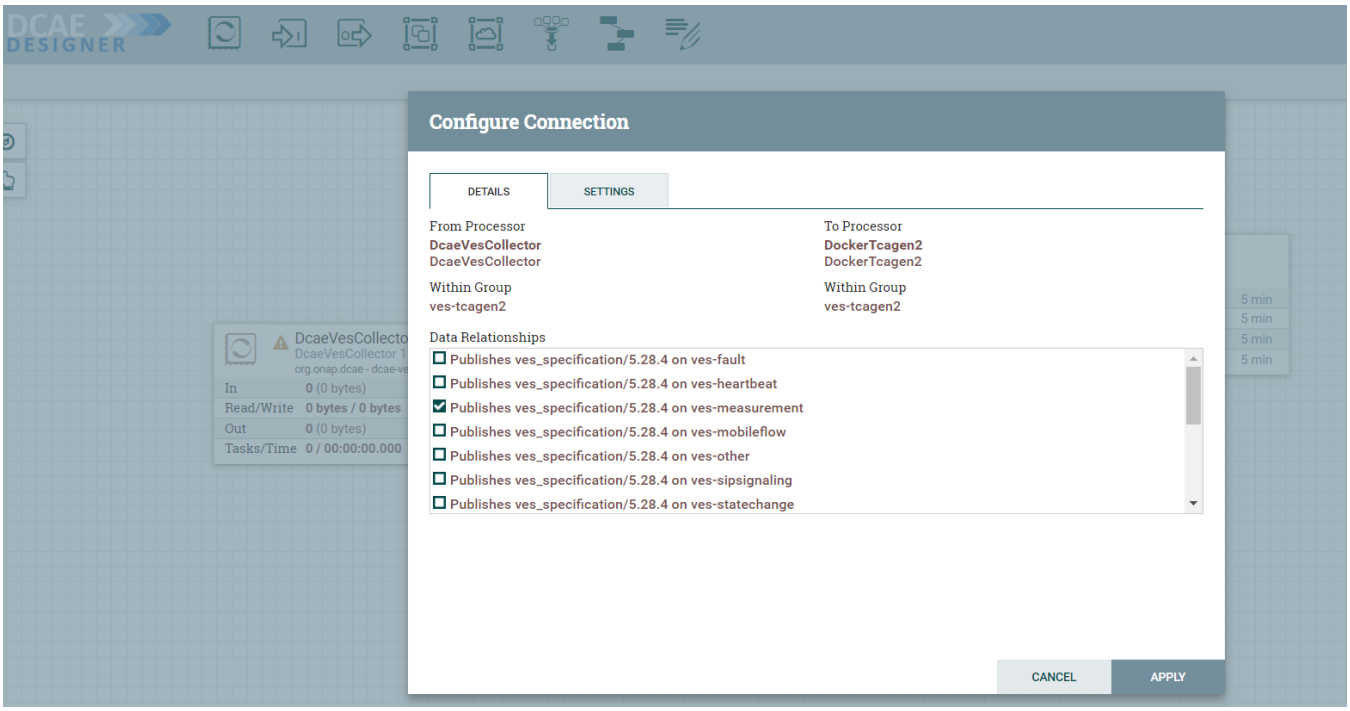
d) Adding components and building the flow

You can add additional components in your flow and connect them.

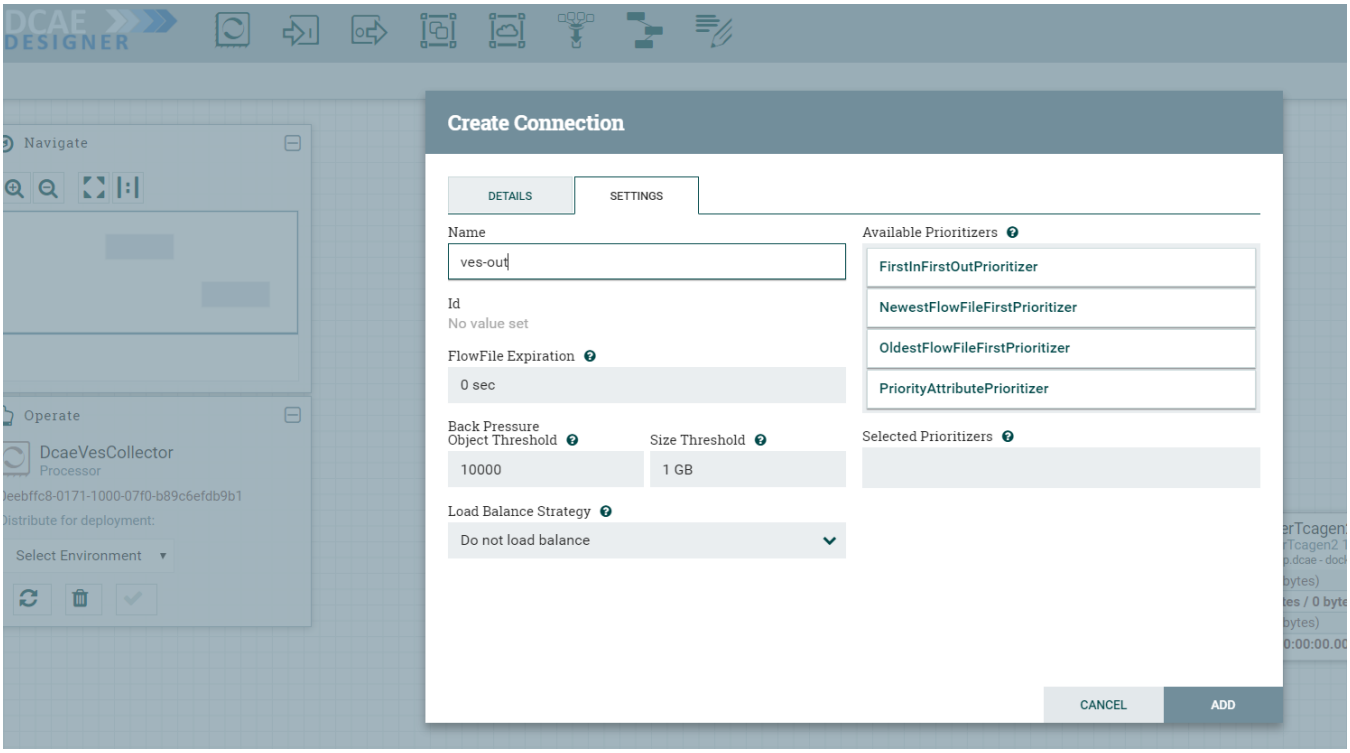
DcaeVesCollector connects to DockerTcagen2.


DcaeVesCollector
DcaeVesCollector 1.5.3
org.onap.dcae - dcae-ves-collector
In 0 (0 bytes) 5 min
Read/Write 0 bytes / 0 bytes 5 min
Out 0 (0 bytes) 5 min
Tasks/Time 0 / 00:00:00.000 5 min
DockerTcagen2
DockerTcagen2 1.0.0
org.onap.dcae - docker-tcagen2
In 0 (0 bytes) 5 min
Read/Write 0 bytes / 0 bytes 5 min
Out 0 (0 bytes) 5 min
Tasks/Time 0 / 00:00:00.000 5 min


Not secure | 10.12.7.116/nifi/?processGroupId=0eead96e-0171-1000-1753-d9902277c590&componentIds=
DCAE DESIGNER
Navigate
Operate
demo2
Process Group
0eead96e-0171-1000-1753-d9902277c590
Distribute for deployment:
Select Environment
DcaeVesCollector
DcaeVesCollector 1.5.3
org.onap.dcae - dcae-ves-collector
In 0 (0 bytes) 5 min
Read/Write 0 bytes / 0 bytes 5 min
Out 0 (0 bytes) 5 min
Tasks/Time 0 / 00:00:00.000 5 min
DockerTcagen2
DockerTcagen2 1.0.0
org.onap.dcae - docker-tcagen2
In 0 (0 bytes) 5 min
Read/Write 0 bytes / 0 bytes 5 min
Out 0 (0 bytes) 5 min
Tasks/Time 0 / 00:00:00.000 5 min
NiFi Flow » demo2 » demo2
7:46 PM
3/24/2020

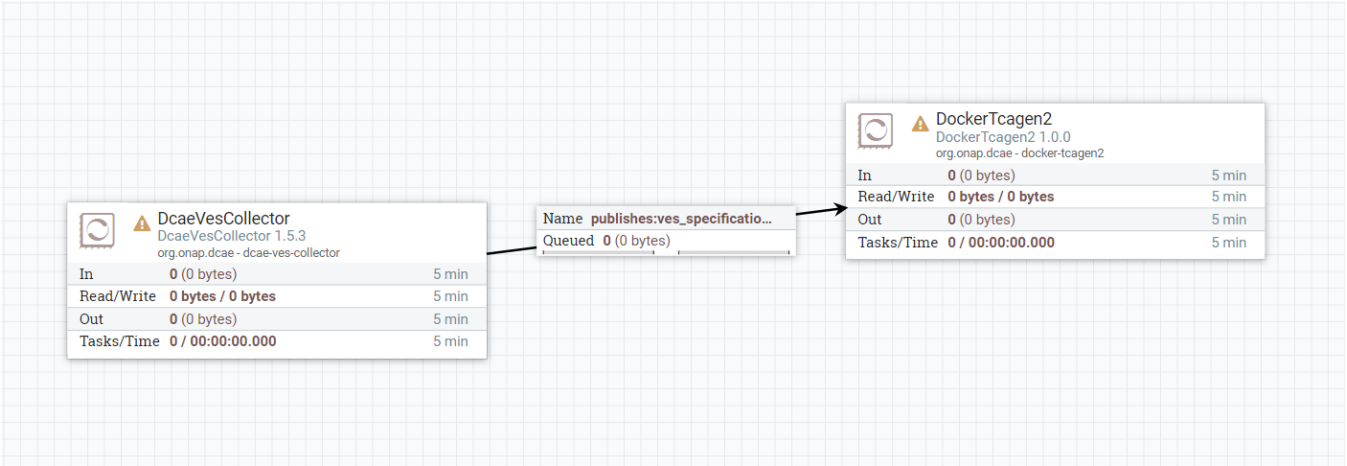


Along the way you need to also provide topic names in the settings section. These can be arbitrary names.



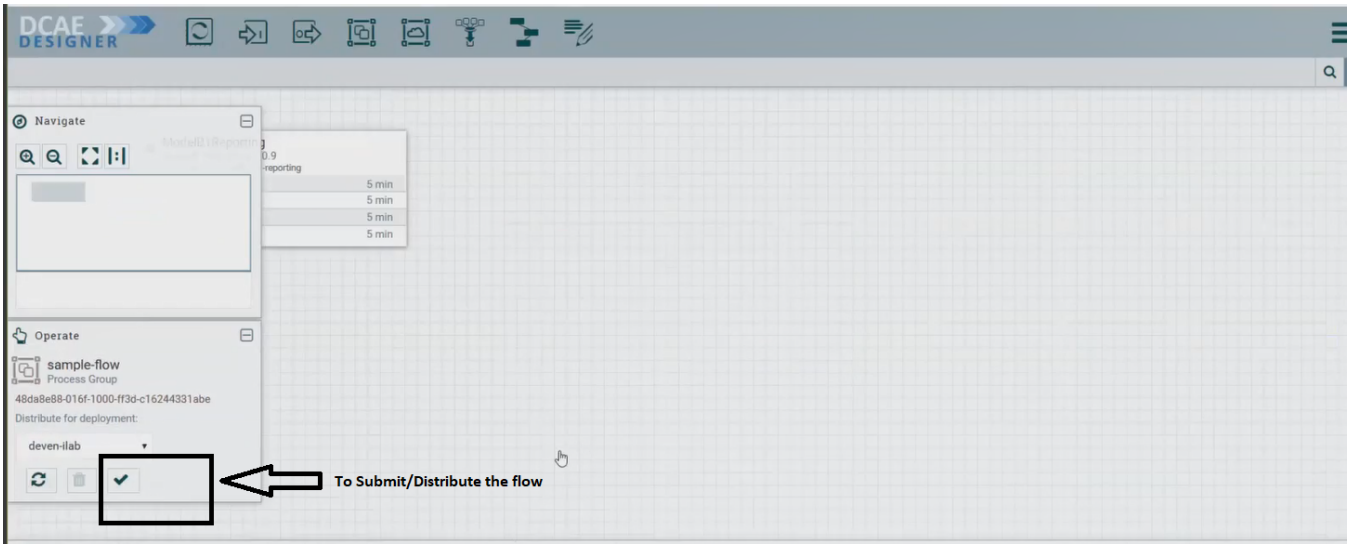
To recap, see how DcaeVesCollector connects to DockerTcagen2. Look at the connection relationships. Currently there is no way to validate these relationships. Notice how it is required to name the topics by going to Settings.

The complete flow after joining our components looks like so

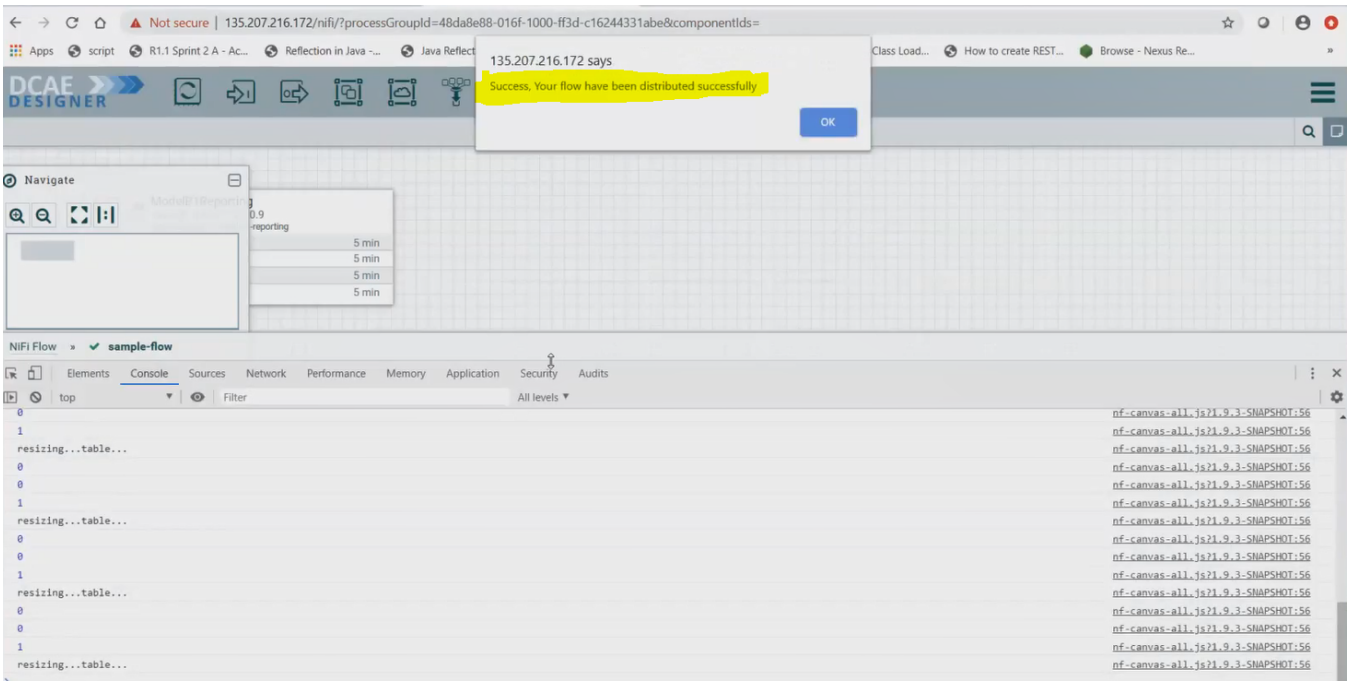


e) Submit/ Distribute the flow:

Once your flow is complete and saved in the Nifi registry, you can choose to submit it for distribution.



If the flow was submitted successfully to the runtime api, you should get a pop up a success message like so -

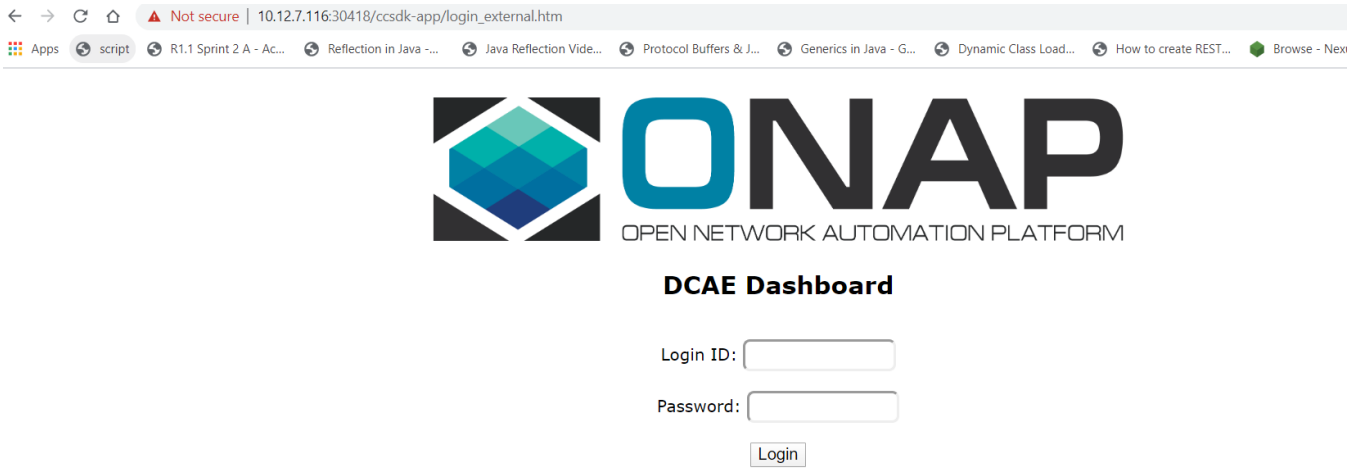


At this step, the design was packaged and sent to Runtime api.

The runtime is supposed to generate the blueprint out of the packaged design/flow and push it to the DCAE inventory and the DCAE Dasboard.

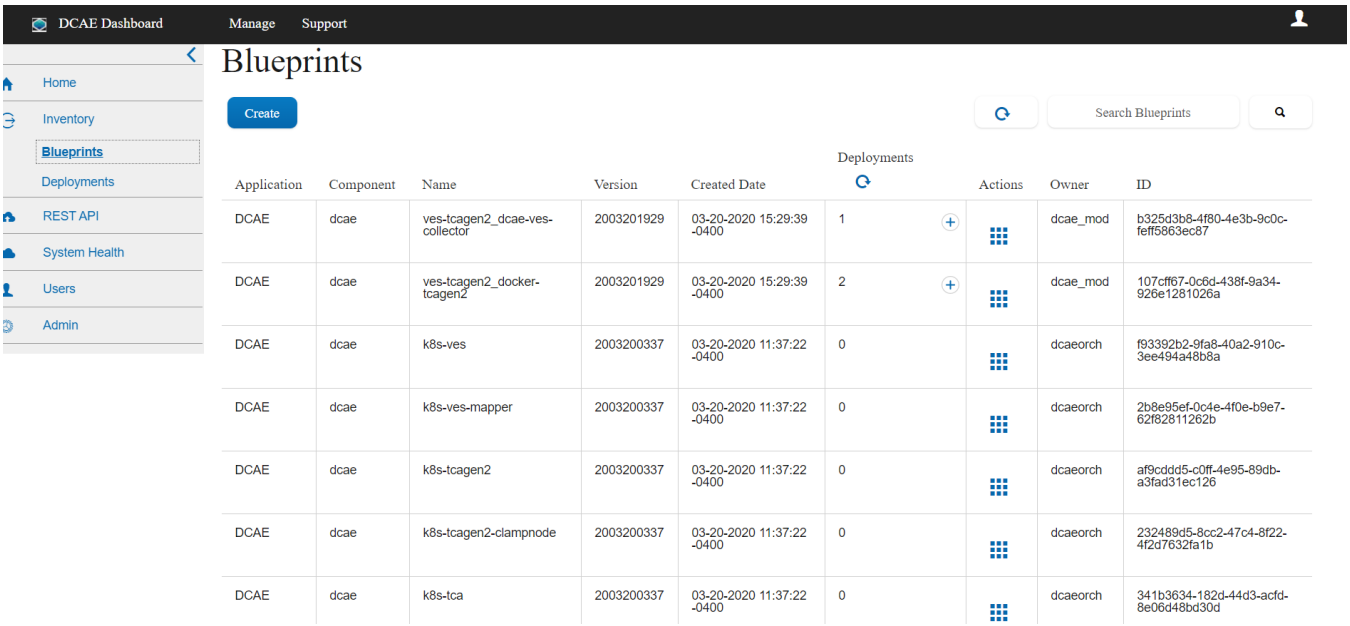
f) Checking the components in the DCAE Dashboard

You should see the generated artifact/ blueprint in the DCAE Dashboard dashboard at https://10.12.7.116:30418/ccsdk-app/login_external.htm in our deployment. The name for each component will be appended by the flow name followed by underscore followed by the component's name.

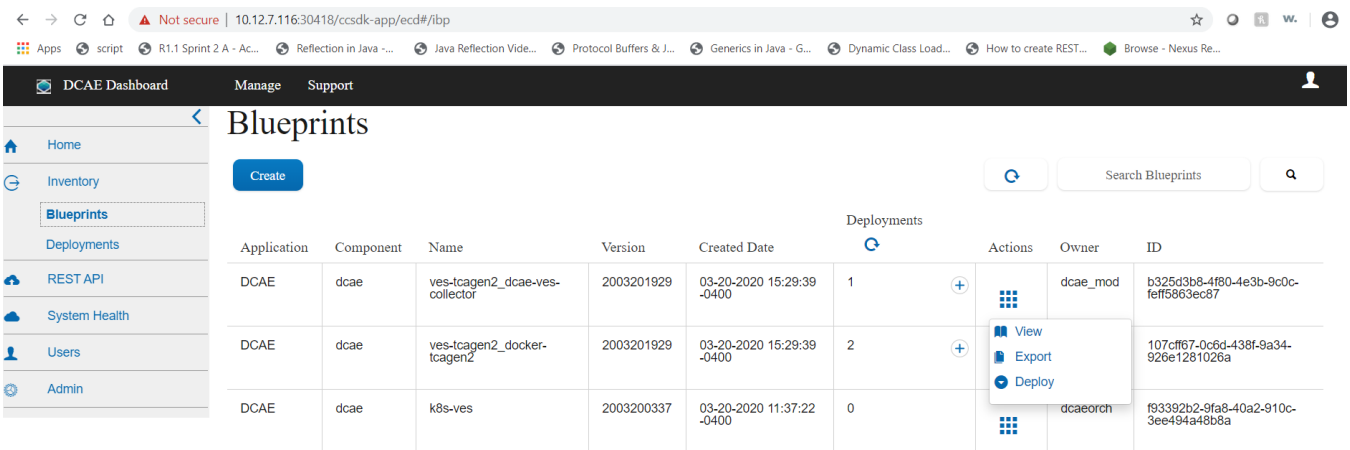


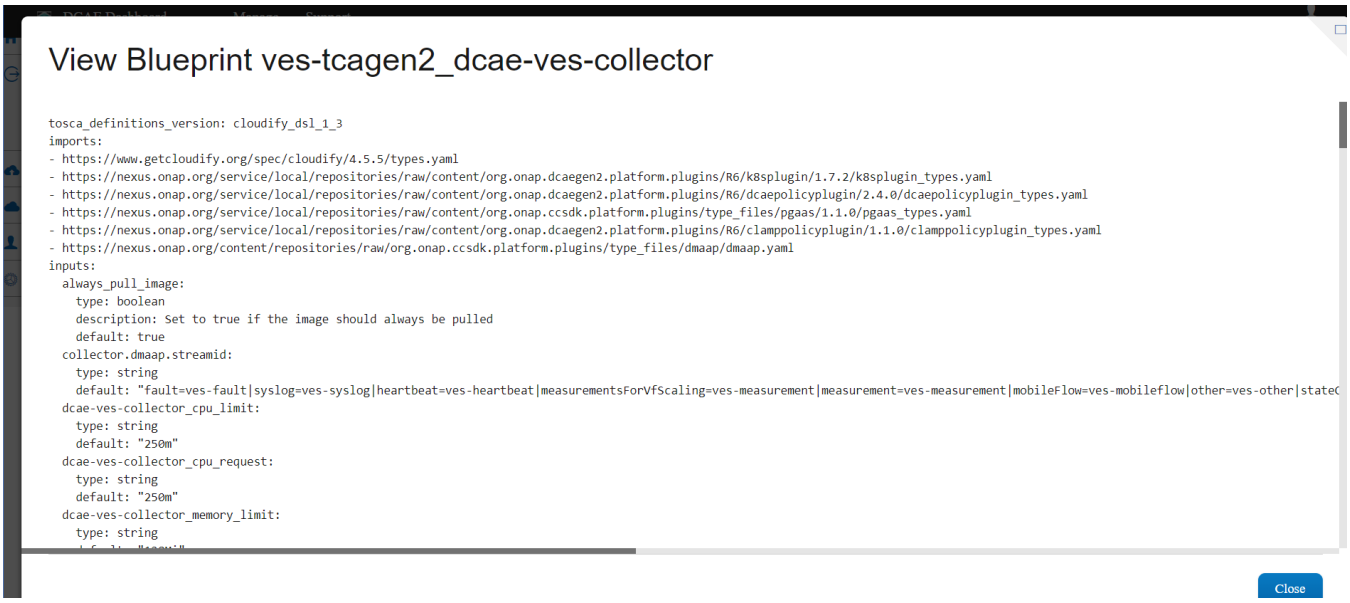
The credentials to access the DCAE Dashboard are

> *Login: su1234*
>
> *Password: fusion*

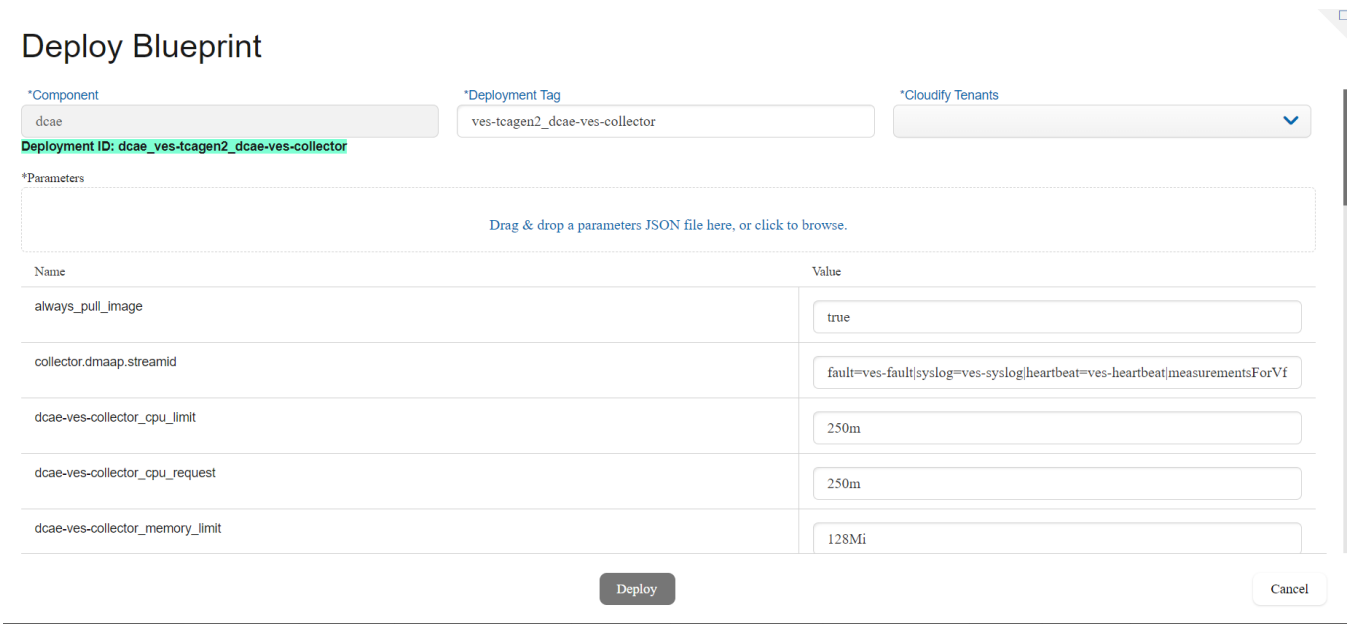


The generated Blueprint can be viewed.





Finally, the generated Blueprint can be deployed.



You can use/import the attached input configurations files to deploy. Drag and Drop these sample JSON files to fill in the configuration values.

NOTE 1: Increase memory limit to 512Mi

NOTE 2: Verify image URL

tca-deploy.input.json

ves-deploy.input.json

Deployment Executions

Auto-refresh

Last execution

Created Date	Workflow ID	ID	Status
2020-04-06T19:06:04.735Z	create_deployment_environment	2d0174ac-96af-4040-8ec6-b10201addd27	successful
2020-04-06T19:06:35.428Z	install	cd594059-d288-4f80-9e37-86e0ec894371	successful

Events/Logs
execution id: cd594059-d288-4f80-9e37-86e0ec894371

Include Logs

Reported Timestamp	Type	Event Type	Message
2020-04-06T19:08:05.859Z	cloudify_event	workflow_node_event	Node instance started
2020-04-06T19:08:05.640Z	cloudify_event	task_succeeded	Task succeeded 'k8splugin.create_and_start_container_for_platforms'
2020-04-06T19:08:05.158Z	cloudify_log		k8s deployment is ready for: dcae-ves-collector
2020-04-06T19:06:43.302Z	cloudify_log		Waiting up to 1800 secs for dcae-ves-collector to become ready