IBM Netcool Operations Insight Event Integrations Operator 1.2.0

Reference Guide
December 11, 2020

SC28-3144-02

Note

Before using this information and the product it supports, read the information in Appendix A, “Notices and Trademarks,” on page 57.

Edition notice

This edition (SC28-3144-02) applies to version 1.2.0 of IBM Netcool Operations Insight Event Integrations Operator and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC28-3144-01.

Contents

About this guide

The following sections contain important information about using this guide.

Document control page

Use this information to track changes between versions of this guide.

The IBM Netcool Operations Insight Event Integrations Operator documentation is provided in softcopy format only. To obtain the most recent version, visit the IBM® Tivoli® Knowledge Center:

https://www.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/operators/all_operators/ops_intro.html

Table 1. Document modification history

Document version	Publication date	Comments
SC28-3144-00	June 26, 2020	First IBM publication.
SC28-3144-01	October 30, 2020	Updated for version 1.1.0. The following topics updated: • "Installing the operator" on page 6 • "Using the CLI with CASE" on page 20 • "Using the CLI (native)" on page 20 The following topics added: • "Installing the operator (air-gap method)" on page 9 • "Operator Upgrade and Rollback process" on page 13 • "Rolling back process" on page 16 • "Upgrading with the CLI" on page 14 • "Upgrading with the OLM UI" on page 13 • "Installing integration resources using the CLI" on page 17
SC28-3144-02	December 11, 2020	Updated for version 1.2.0. The following topics updated: • "Troubleshooting" on page 21 The following topics added: • "Deploying the Probe for Kafka Integration" on page 44 • "Integrating with IBM Event Streams for IBM Cloud" on page 50 • "Customizing probe rules in ConfigMap" on page 54

Chapter 1. IBM Netcool Operations Insight Event Integrations Operator

The IBM Netcool Operations Insight (NOI) Event Integrations Operator can deploy multiple integrations, for example: it can run instances of the Prometheus Probe, CEM Probe and CEM Gateway.

The NOI Event Integrations Operator currently supports the following integrations:

- IBM Netcool/OMNIbus Probe for IBM Cloud Event Management

  Creates the webhook endpoints to receive notification in the form of HTTP POST requests from IBM Cloud Event Management (CEM).
- IBM Netcool/OMNIbus Probe for Prometheus

  Creates the webhook endpoints to receive notification in the form of HTTP POST requests from Prometheus Alert Manager.
- IBM Netcool/OMNIbus Gateway for IBM Cloud Event Management

  Processes events and alerts from IBM Netcool/OMNIbus ObjectServer and forwards them to IBM Cloud Event Management (CEM) dashboard.

Each of the NOI integrations is defined as a separate resource kind.

What's new

This section describes what's new in IBM Netcool Operations Insight Event Integration CASE.

Version 1.2.0

Release date: 11 December 2020 Part number: CC8YGML

- Red Hat Openshift Container Platform (OCP) 4.6 support. This operator version is verified running on OCP 4.5 and 4.6
- New `KafkaProbe` kind which deploys the Message Bus probe as a Kafka consumer. You can configure this probe to integrate with Kafka systems such as IBM Event Streams on IBM Cloud.
- Several changes in the Operator cluster service version file to add links to specific documentation sections in the description for easier navigation and updated provided API name and description include a license file link in each CRD tile description.
- Add `app.kubernetes.io/part-of:` label into operator deployment resource.
- License file updated with NOI 1.6.3 license (L/N: L-TKAI-BTYDF9).

IBM Netcool Operations Insight v1.6.3 release notice: https://www.ibm.com/support/pages/node/6377824

CVE patches

This version resolves the following CVEs detected in our operator and operand docker image:

- `CVE-2020-26160`
- `CVE-2020-13956`
- `CVE-2012-5783`
- `CVE-2015-5262`

Version 1.1.0

Release date: 30 October 2020 Part number: CC8A7ML

- Supported on Red Hat Openshift Container Platform (OCP) 4.4 and 4.5.
- New CASE launcher script to assist mirroring image in an internal registry for offline (air-gap) installation or in a restricted network.
- Operator version 1.1.0 includes the following changes:
  1. [FIX] Missing parameter when using OLM UI to create operand instances.
  2. Prometheus Probe and CEM Probe version 4.5.0 support. This version includes an updated probe and utility images which resolves CVEs listed below.
  3. CEM Gateway version 1.3.0 support. This version includes an updated probe and utility images which resolves CVEs listed below.

IBM Netcool Operations Insight v1.6.2 release notice: https://www.ibm.com/support/pages/node/6243794

CVE patches

This version resolves the following CVEs detected in our operator and operand docker image:

- `CVE-2020-13777`
- `CVE-2020-12049`
- `CVE-2020-11080`

Version 1.0.1

Release date: 07 August 2020 Part number: CC7LBML

- Supported on Red Hat Openshift Container Platform (OCP) 4.3 and 4.4.
- [FIX] OLM unable to retrieve bundle information from operator bundle image.

IBM Netcool Operations Insight v1.6.1 release notice: https://www.ibm.com/support/pages/node/6221238

Version 1.0.0

Release date: 30 June 2020 Part number: CC714ML

Initial release.

- Supported on Red Hat Openshift Container Platform (OCP) 4.3 and 4.4.
- You can deploy Probe for IBM Cloud Event Management or Prometheus Alert Manager and Gateway for IBM Cloud Event Management on OCP by creating a `cemprobe`, `prometheusprobe` or `cemgateway` custom resource.

IBM Netcool Operations Insight v1.6.1 release notice: https://www.ibm.com/support/pages/node/6221238

Prerequisites

This section describes the prerequisites to installing and deploying the IBM Netcool Operations Insight Event Integrations Operator.

The IBM Netcool Operations Insight Event Integrations Operator requires IBM Tivoli Netcool/OMNIbus ObjectServer to be created and running on Red Hat OpenShift Container Platform (OCP) 4.6. IBM Netcool Operations Insight 1.6.3 is required to create IBM Tivoli Netcool/OMNIbus ObjectServer on OCP 4.6. Refer to the following installation instructions in IBM Knowledge Center: https://www.ibm.com/support/knowledgecenter/en/SSTPTP_1.6.3/com.ibm.netcool_ops.doc/soc/integration/task/int_installing-on-rhocp.html

If you are deploying the Probe for IBM Cloud Event Management or the Probe for Prometheus, refer to the following prerequisites page: https://www.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/helms/cloud_monitoring/wip/reference/cemh_prereq.html

If you are deploying Gateway for IBM Cloud Event Management (CEM), refer to following prerequisites page: https://www.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/helms/cloud_event_management/wip/reference/cemgwh_prereq.html

Resources required

Each probe or gateway requires the following resources per pod:

- CPU Requested : 100m (100 millicpu)
- Memory Requested : 128Mi (~ 134 MB)

You can increase or decrease the resource request and limit by modifying the resources settings (`resources.limits` and `resources.requests`) in the customer resource YAML.

Cluster configuration

The minimum cluster configuration required is 1 master node and 3 worker nodes. The recommended cluster configuration is 3 master nodes and 5 worker nodes. This operator requires worker nodes that support `amd64` architecture.

Security context constraints

On Red Hat OpenShift Container Platform (OCP) this operator requires a SecurityContextConstraints resource to be bound to the target namespace prior to installation.

The predefined SecurityContextConstraints name: `ibm-restricted-scc` has been verified for this operator, if your target namespace is bound to this SecurityContextConstraints resource you can proceed to install the operator.

Alternatively, for this operator you can define a custom SecurityContextConstraints resource which can be used to finely control the permissions and/or capabilities needed to deploy the operator. You can enable this custom SecurityContextConstraints resource using the command line tool.

- Custom `SecurityContextConstraints` definition:

```
apiVersion: security.openshift.io/v1
kind: SecurityContextConstraints
metadata:
  annotations:
    kubernetes.io/description: "This policy is the most restrictive,
      requiring pods to run with a non-root UID, and preventing pods from accessing the host.
      The UID and GID will be bound by ranges specified at the Namespace level."
    cloudpak.ibm.com/version: "1.1.0"
  name: ibm-netcool-integrations-scc
allowHostDirVolumePlugin: false
allowHostIPC: false
allowHostNetwork: false
allowHostPID: false
allowHostPorts: false
allowPrivilegedContainer: false
allowPrivilegeEscalation: false
allowedCapabilities: null
allowedFlexVolumes: null
allowedUnsafeSysctls: null
defaultAddCapabilities: null
defaultAllowPrivilegeEscalation: false
forbiddenSysctls:
  - "*"
fsGroup:
  type: MustRunAs
  ranges:
  - max: 65535
    min: 1
readOnlyRootFilesystem: false
```

```
requiredDropCapabilities:
- ALL
runAsUser:
  type: MustRunAsNonRoot
seccompProfiles:
- docker/default
seLinuxContext:
  type: RunAsAny
supplementalGroups:
  type: MustRunAs
  ranges:
  - max: 65535
    min: 1
volumes:
- configMap
- downwardAPI
- emptyDir
- persistentVolumeClaim
- projected
- secret
```

Gathering information for IBM Support

This topic describes how to gather information for IBM Support.

To help IBM Support to troubleshoot any issues observed in your IBM Netcool Operations Insight Event Integration instance, use the following procedure to collect logs and information from your cluster:

1. Log in to your Red Hat OpenShift Container Platform by using the `oc` CLI. You must login as a namespace administrator or a cluster administrator who has permissions to read resources in the target namespace.
2. Set the `namespace` variable to the namespace where the instance is deployed:

   ```
   export namespace=my-namespace
   ```

3. Get the instance description in your namespace:

   a. To get a specific instance resource description, run the following command:

   ```
   kubectl get cemprobe,prometheusprobe,cemgateway,kafkaprobe --namespace $namespace
    kubectl describe <instance kind>/<instance name> --namespace $namespace
   ```

   Replace `<instance kind>` and `<instance name>` in the second command with the resource name from the output from the first command.

   b. Or run the following command to get all instance descriptions in the namespace and capture the command output:

   ```
   for i in $(kubectl get cemprobe,prometheusprobe,cemgateway,kafkaprobe --no-headers --
   namespace
   $namespace | awk '{print $1}'); do kubectl describe $i ; done
   ```

4. Get the pod names of the instance:

   ```
   kubectl get pods -l app.kubernetes.io/instance=example-cemprobe --namespace $namespace
   ```

5. Using the pod names from the previous command, run the following command to get the instance pod logs and capture the command output:

   ```
   kubectl logs <pod name>
   ```

   If there is more than one pod, run the same command for each pod name.

6. Capture the operator `deployment` resource description:

   ```
   kubectl describe deployment netcool-integrations-operator --namespace $namespace
   ```

7. Capture the operator pod logs:

```
kubectl get pods -l app.kubernetes.io/instance=netcool-integrations-operator  --namespace
$namespace
kubectl logs <operator-pod> --namespace $namespace
```

Replace <operator-pod> in the command with the name of the operator pod from the `kubectl get pods` output.

8. Get the list of ConfigMap used by your instance and its data:

```
kubectl get configmap --namespace $namespace
kubectl kubectl describe <configmap> --namespace $namespace
```

Replace <configmap> with the ConfigMap name from the `kubectl get configmap` output. You should provide each ConfigMap data for troubleshooting.

To help IBM Support to troubleshoot any issues observed in your IBM Netcool Operations Insight Event Integration instance, follow the procedure below to collect logs and information from your cluster.

a. Log in to your Red Hat OpenShift Container Platform by using the oc CLI. You must login as a namespace administrator or cluster administrator who has permissions to read resources in the target namespace.

b. Set the `namespace` variable with the namespace where the instance is deployed.

```
export namespace=my-namespace
```

Role-based access control

If you are deploying the Probe for IBM Cloud Event Management or the Probe for Prometheus, refer to the following role-based control page to view the required RBAC settings for the probe: https://www.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/helms/all_helms/wip/reference/hlm_role_based_access.html

If you are deploying Gateway for IBM Cloud Event Management (CEM), refer to the following role-based control page to view the required RBAC settings for the gateway: https://www.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/helms/all_helms/wip/reference/hlm_role_based_access_cem_gw.html

Operator scope

This operator is a namespaced-scoped operator that watches a single namespace where the operator is deployed. To deploy probes or gateways on more than one namespace, it is recommended to deploy a separate operator to watch the namespace.

Storage

If you are deploying the Gateway for IBM Cloud Event Management (CEM), see the following storage page for the configuration options: https://www.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/helms/cloud_event_management/wip/reference/cemgwh_storage.html.

Pre-installation tasks

This section describes the tasks you need to perform before installing the IBM Netcool Operations Insight Event Integrations Operator.

Create a secret for your entitlement key

Check your entitlement and get an entitlement key from the My IBM Container Library website: https://myibm.ibm.com/products-services/containerlibrary. You need to store the entitlement key on your cluster

by creating an Image Pull Secret. Using a Kubernetes secret allows you to securely store the entitlement key on your cluster and assign it to the operator and the operand deployments to pull images from entitled registry.

Run the following command to create the secret and add it to your OpenShift namespace:

```
kubectl create secret docker-registry cp.icr.io --docker-server=cp.icr.io --docker-username=cp
--docker-password=<KEY> --docker-email=<EMAIL> --namespace <NAMESPACE>
```

Where:

<KEY> is your entitlement key.

<EMAIL> is your email address.

<NAMESPACE> is the name of your target OpenShift Container Platform namespace where the operator will be deployed.

The name of the secret that you are creating is `cp.icr.io` with `cp` as the username and `cp.icr.io` as the docker server. The secret name is used by the operator to deploy the offering in later steps. If you change the name of any of secrets that you create, you need to change the corresponding name in later steps.

Installing the operator

This section describes how to install the operator.

There are three methods for installing the operator:

1. Using the web console
2. Using the Command Line Interface
3. Using the Command Line Interface natively

Installing the operator using the web console

To install the IBM Netcool Operations Insight Event Integrations Operator using the web console, use the following steps:

1. On the OpenShift Web console, select the menu navigation **Administration -> Cluster Settings**.
2. Under the `Global Configuration` tab select `OperatorHub` configuration resource.
3. Under the `Sources` tab, click the `Create Catalog Source` button.
4. Provide a catalog source name and the image URL:

   `docker.io/ibmcom/ibm-netcool-integrations-operator-catalog:<latest tag>`

   If a specific version is required, change the image tag with the specific version or use the image digest.

   Select the **Create** button.
5. The catalog source appears, after a few minutes refresh the screen to see the number of operators count become 1.
6. Select the menu navigation **OperatorHub** and search for the `IBM Netcool Operations Insight Event Integrations Operator`. Select the **Install** button.
7. Select a namespace to install the operator. Do not use namespaces that are kubernetes or openshift owned like kube-system or default. If you do not already have a project, create a project under the navigation menu **Home -> Projects**.
8. Select the **Subscribe** button.
9. Under the navigation menu **Operators -> Installed Operators**, view the `IBM Netcool Operations Insight Event Integrations Operator`. It may take a few minutes to install. View the status

at the bottom of the installed `IBM Netcool Operations Insight Event Integrations Operator` page which should reflect `Succeeded`.

Creating integration instance

10. Select **Create instance** to create the probe or gateway.

    For details about populating the required configuration into YAML form, see "Configuring custom resources" on page 19.

    Note : If you observe that the Operator installation via the OCP Web console hangs, see "Operator installation hangs when using the OCP Web console" on page 23.

Note : More than one instance of each probe and gateway can be deployed in the same namespace, but the CR instance name must be unique to avoid conflicts.

Installing the operator using the Command Line Interface

To install the operator using the Command Line Interface, you can use the launcher script provided in the CASE. The launcher script is executed using the `cloudctl` tool and will perform necessary prerequisite checks for each inventory action that is performed.

Prerequisites:

- CASE archive downloaded and unpacked into your local file system.
- The required `SecurityContextConstraint` is applied to the target namespace.

1. Download the CASE archive using the command below or from IBM Passport Advantage:

   ```
   cloudctl case save \
    --case https://github.com/IBM/cloud-pak/raw/master/repo/case/ibm-netcool-integrations-
   <version>.tgz \
    --outputdir <destination dir>
   ```

   Where `<version>` is the CASE version to download, and `<destination dir>` is the target destination directory.

2. (Optional) Set the CASEPATH environment variable to the downloaded:

   ```
   export CASEPATH=<destination dir>/ibm-netcool-integrations-<version>.tgz
   ```

3. Determine if there is an existing `CatalogSource` resource for the IBM Netcool Operations Insight Event Integrations operator in the target namespace:

   ```
   kubectl get catalogsource --namespace $NAMESPACE
   ```

4. If there is an existing `CatalogSource` resource exists, edit the resource and udpate the `spec.image` attribute with the latest catalog index image:

   ```
   spec:
     image: docker.io/ibmcom/ibm-netcool-integrations-operator-catalog@<digest>
   ```

   Where `<digest>` with the latest image digest or tag.

5. If there is no `CatalogSource` resource, create one using the following command:

   ```
     cloudctl case launch  \
     --case $CASEPATH \
     --namespace $NAMESPACE \
     --inventory netcoolIntegrationsOperatorSetup \
     --action install-catalog \
     --tolerance 1
   ```

   Where `<destination dir>` is the destination directory of the CASE from the previous step.

6. Install the Operator by invoking the `install-operator` action:

   ```
   cloudctl case launch --case $CASEPATH \
   --namespace $NAMESPACE \
   --inventory netcoolIntegrationsOperatorSetup \
   ```

```
--action install-operator  \
--tolerance 1
```

7. List the pods and verify that the operator pod is running:

```
kubectl get pods --namespace $NAMESPACE
```

8. Follow the procedure in “Installing integration resources using the CLI” on page 17 to install probe or gateway instances.

Installing the operator using Command Line Interface natively

To install the operator using the CLI, you can use the launch script provided in the CASE. This method will install the operator without OLM. The launch script is executed using the `cloudctl` tool and will perform necessary prerequisite checks for each inventory action that is performed.

Note The launch scripts are provided the CASE from version 1.1.0.

Prerequisites:

- The required `SecurityContextConstraint` is applied to the target namespace.

1. Download the CASE archive using the command below or from IBM Passport Advantage:

```
cloudctl case save \
 --case https://github.com/IBM/cloud-pak/raw/master/repo/case/ibm-netcool-integrations-
<version>.tgz \
 --outputdir <destination dir>
```

   Where <version> is the CASE version to download, and <destination dir> is the target destination directory.

2. (Optional) Set the CASEPATH environment variable to the downloaded

```
export CASEPATH=<destination dir>/ibm-netcool-integrations-<version>.tgz
```

3. Install the Operator by invoking the `install-operator-native` action and pass in the `--operatorImage` argument to set the operator controller image to use.

```
cloudctl case launch --case $CASEPATH \
--namespace $NAMESPACE \
--inventory netcoolIntegrationsOperatorSetup \
--action install-operator-native \
--args "--operatorImage docker.io/ibmcom/ibm-netcool-integrations-operator@<image digest> --
secret <image pull secret name>" \
--tolerance 1
```

   where: <image digest> is the "ibm-netcool-integrations-operator" image digest, <image pull secret name> is an existing secret name to use as the image pull secret.

   The sample command above uses the image digest so that worker node is able to pull the image from an internal registry if configured with a `ImageContentSourcePolicy` custom resource to mirror the image. To get the image digest, you can use `skopeo` tool to inspect the image and or pull the image into your local file system and then inspect it.

   If the cluster is configured to pull images from a mirror registry, the

```
<image pull
        secret name>
```

   should contain the credentials of the internal mirror registry.

4. Follow the procedure in “Installing integration resources using the CLI” on page 17 to install probe or gateway instances.

Installing the operator (air-gap method)

This section describes how to install an operator if you have a restricted network or you are not connected to the Internet.

About the procedure

If your cluster is in a restricted network or not connected to the Internet, the OCP cluster will not be able to pull container images directly from public registries like Docker.io or IBM Entitled Registry. You will need to fetch the images into a Bastion server, load the images into an internal mirror registry, and configure image mirroring in your OCP cluster so that it will pull the container images from the internal mirror registry.

The following diagram provides a high-level view of how the setup works:

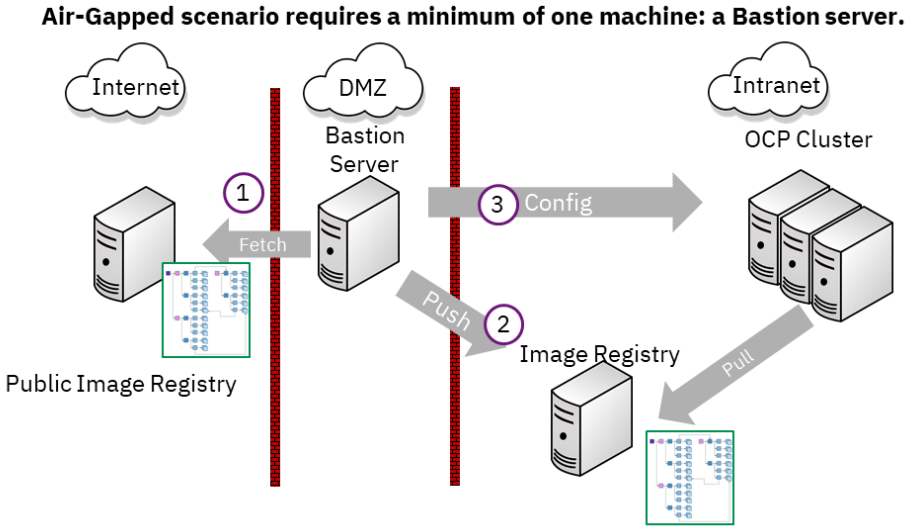

Figure 1. OLM Offline (air-gap) scenario

Prerequisites

The steps that follow are designed to run on a Bastion server that has access to the Internet and the internal network, and have the following requirements:

- A container registry that supports Docker Image Manifest V2 format. The Bastion server as well as the OpenShift cluster must be able to access it. For internal testing you can stand up your own container registry for your Fyre cluster. The OCP internal registry is not supported for air-gap use case because of several limitations.
- Red Hat OpenShift 4.5 or 4.5 cluster.
- `docker` or `podman` must be installed on the Bastion server. If docker is installed, it must be configured to allow access to the container registry.
- `oc` Red Hat OpenShift Container Platform CLI tool must be installed on the Bastion server.
- `cloudctl` IBM CloudPak CLI tool must be installed on the Bastion server.

Installing using the air-gap method

To install the operator using the air-gap method, use the following steps:

1. Install the command line tools
2. Get the CASE
3. Create environment variables for the mirror registry

4. Start a mirror registry
5. Prepare credentials to mirror the images
6. Mirror the images
7. Configure the cluster
8. Validate image mirroring
9. Add a catalog source
10. Install the operator
11. Installing an instance

Install the command line tools

The `cloudctl` tool is required to perform the installation.

To download the `[cloudctl]` tool for your platform, use the following steps:

1. Download the binary file:

```
wget https://github.com/IBM/cloud-pak-cli/releases/download/v<version-number>/<binary-file-name>
```

2. Extract the binary:

```
tar -xf <binary-file-name>
```

3. Run the following commands to modify and move the file:

```
chmod 755 <file-name>
mv <file-name> /usr/local/bin/cloudctl
```

Get the CASE

Download and unpack the Netcool Operations Insight Integrations CASE archive from Passport Advantage into a directory, for example:

`~/offline/case`

Create environment variables for the mirror registry

Use the following command:

```
export REGISTRY=<registry>
export REGISTRY_USERNAME=<username>
export REGISTRY_PASSWORD=<password>
```

For example:

```
export REGISTRY=api.ocp449-dev2.cp.fyre.ibm.com
export REGISTRY_PORT=5000
export REGISTRY_USERNAME=admin
export REGISTRY_PASSWORD=adminPAssw0rd
```

Start a mirror registry

To start a mirror registry, use the following steps:

1. Initialize the registry:

```
cloudctl case launch --case ~/offline/case/ibm-netcool-integrations-*.tgz --inventory airgap
--action init-registry --args "--registry $REGISTRY --user $REGISTRY_USERNAME --pass
$REGISTRY_PASSWORD" --tolerance 1
```

This creates a registry with the specified name, credentials and creates a self-signed TLS certificate.

2. Start the registry:

```
cloudctl case launch --case ~/offline/case/ibm-netcool-integrations-*.tgz --inventory airgap
--action start-registry --args "--engine podman" --tolerance 1
```

3. Create a certificate directory and copy the registry generated TLS certificates:

```
mkdir -p /etc/docker/certs.d/$REGISTRY/
cp /tmp/docker-registry/certs/ca.crt /etc/docker/certs.d/$REGISTRY/
cp /tmp/docker-registry/certs/server.crt /etc/docker/certs.d/$REGISTRY/
```

Prepare credentials to mirror the images

To mirror the images, first set the required credentials for the source registry and target registry. In the Bastion server use case, the source registry is the IBM Entitled Registry and Docker Hub, and the target registry is the internal mirror registry created in previous step:

```
export SOURCE_REGISTRY=cp.icr.io
export SOURCE_REGISTRY_USER=cp
export SOURCE_REGISTRY_PASS=<key>
export NAMESPACE=noi-integrations


cloudctl case launch  \
  --case ~/offline/case/ibm-netcool-integrations-*.tgz    \
  --namespace $NAMESPACE     \
  --inventory airgap   \
  --action configure-creds-airgap  \
  --args "--registry $SOURCE_REGISTRY --user $SOURCE_REGISTRY_USER --pass
$SOURCE_REGISTRY_PASS" \
  --tolerance 1

cloudctl case launch  \
  --case ~/offline/case/ibm-netcool-integrations-*.tgz    \
  --namespace $NAMESPACE     \
  --inventory airgap   \
  --action configure-creds-airgap  \
  --args "--registry $REGISTRY:$REGISTRY_PORT --user $REGISTRY_USERNAME --pass
$REGISTRY_PASSWORD" \
  --tolerance 1
```

Mirror the images

The following command will fetch the images from the public registries and mirror them in the mirror registry:

```
cd ~/offline
cloudctl case launch \
  --case case/ibm-netcool-integrations-*.tgz \
  --inventory airgap \
  --action mirror-images \
  --args "--registry $REGISTRY:$REGISTRY_PORT --inputDir $PWD/case" \
  --tolerance 1
```

Configure the cluster

To configure the cluster to try to pull the images from the mirror registry, an `ImageContentSourcePolicy` custom resource is required. This automatically redirects image pull requests from a specified registry location to an alternative location.

You also need to create a global image pull secret so that the images can be pulled from the internal registry. Both of these actions are handled by the following command in the launch script:

```
cloudctl case launch  \
  --case case/ibm-netcool-integrations-*.tgz    \
  --namespace $NAMESPACE     \
  --inventory airgap   \
  --action configure-cluster-airgap  \
```

```
  --args "--registry $REGISTRY:$REGISTRY_PORT --inputDir $PWD/case --dryRun" \
  --tolerance 1
```

Note: Remove the `--dryRun` option to apply the changes to the cluster. This will cause all nodes to be restarted to apply the configuration. Wait until all nodes are in `Ready` state before continuing.

Validate image mirroring

After `ImageContentSourcePolicy` resources are deployed and the cluster has finished updating each node, you can validate that mirroring works using the following steps:

1. Start a debug session with a node (get a list of nodes using `oc get node`):

   ```
   oc debug node/<node name>
   ```

2. Once the session starts, type in the `chroot /host` command as prompted.
3. Run the `podman pull` command on one of the images that should have been mirrored using a digest. For example:

   ```
   podman pull --log-level=debug --authfile /var/lib/kubelet/config.json docker.io/ibmcom/ibm-
   netcool-integrations-
   operator@sha256:77f49f66cb0c9917abb0a696ca662f6bc767cb4295bc01bfae15266f87ecfbed
   ```

   Check the output to see if the request to `docker.io` registry got redirected to the mirrored registry.

Add a catalog source

To add the operator into the OLM operator catalog, add a new `CatalogSource` custom resource by running the following command:

```
cloudctl case launch  \
  --case case/ibm-netcool-integrations-*.tgz    \
  --namespace $NAMESPACE     \
  --inventory netcoolIntegrationsOperatorSetup   \
  --action install-catalog  \
  --args "--registry $REGISTRY:$REGISTRY_PORT --inputDir $PWD/case" \
  --tolerance 1
```

Install the operator

To add the operator, use the following command:

```
cloudctl case launch  \
  --case case/ibm-netcool-integrations-*.tgz    \
  --namespace $NAMESPACE     \
  --inventory netcoolIntegrationsOperatorSetup   \
  --action install-operator  \
  --args "--registry $REGISTRY:$REGISTRY_PORT --inputDir $PWD/case --user $REGISTRY_USERNAME --
pass $REGISTRY_PASSWORD --secret noi-int-airgap" \
  --tolerance 1
```

Installing an instance

To install an instance, use the following steps:

1. The CASE includes the custom resource template files that you should use to install an instance. Extract the template CR files using the following command:

   ```
   tar -xvf ibm-netcool-integrations-<version>.tgz ibm-netcool-integrations/inventory/
   netcoolIntegrationsOperator/files/deploy/crs
   ```

   Where `<version>` is the CASE version.

2. Edit the custom resource files and save it with a different filename. Optionally, set the file path as $CRFILE environment variable.

```
cloudctl case launch  \
  --case case/ibm-netcool-integrations-*.tgz    \
  --namespace $NAMESPACE     \
  --inventory netcoolIntegrationsOperator   \
  --action apply-custom-resources  \
  --args "--inputDir $PWD/case --instance test-probe --crFile $CRFILE" \
  --tolerance 1
```

Or using the filepath in the `--crFile` argument:

```
cloudctl case launch  \
  --case case/ibm-netcool-integrations-*.tgz    \
  --namespace $NAMESPACE     \
  --inventory netcoolIntegrationsOperator   \
  --action apply-custom-resources  \
  --args "--inputDir $PWD/case --crFile $PWD/case/ibm-netcool-integrations/inventory/
netcoolIntegrationsOperator/files/deploy/crs/test-
probes.integrations.noi.ibm.com_v1beta1_prometheusprobe_cr.yaml" \
  --tolerance 1
```

Operator Upgrade and Rollback process

You can upgrade using the Operator Lifecycle Manager (OLM) UI, or with the command line interface.

Upgrading with the OLM UI

Use these instructions to upgrade an existing IBM Netcool® Operations Insight® Event Integrations Operator using the Red Hat® OpenShift® Operator Lifecycle Manager (OLM) user interface (UI).

Upgrade the Catalog Source

1. From the Red Hat OpenShift OLM UI, navigate to **Administration > Cluster Settings**, and then select the **OperatorHub** configuration resource under the Global Configurations tab.
2. Under the **Sources** tab, click the existing IBM Netcool Operations Insight Event Integrations catalog source.
3. Edit the catalog source YAML and upgrade image (`spec.image`) with the IBM Netcool Operations Insight Event Integrations catalog source image with image tag, for example

```
docker.io/ibmcom/ibm-netcool-integrations-operator-catalog:<latest version>
```

or with the image with digest, where `<latest version>` is the latest catalog source image tag. Click the **Save** button.

Upgrade the IBM Netcool Operations Insight Event Integrations operator

You can skip this section if the operator was installed in `Automatic` approval strategy. The operator should be updated automatically after the catalog source is upgraded.

1. Navigate to **Operators > Installed Operators**, select **Project** and then search for the IBM Netcool Operations Insight Event Integrations operator.
2. If the Status shows `UpgradePending`, it is likely that the operator was installed with `Manual` approval strategy. Click on the operator name and check if the upgrade is pending an approval. To approve the upgrade, under subscription details, click the **requires approval** button > Preview Install Plan > click **Approve**.
3. Navigate to **Operators > Installed Operators** and view the IBM Netcool Operations Insight Event Integrations operator. It takes several minutes for the operator to upgrade. Ensure that the operator status changes from `Upgrading` to `Succeeded`.

Upgrade the Probe or Gateway instances

1. Navigate to **Operators > Installed Operators**, select **Project**, and search for and select the IBM Netcool Operations Insight Event Integrations operator.

2. Edit the probe or gateway instance YAML under the **All instances** tab. It is recommended that you take a copy of the instance YAML before changing it, in case you later decide to rollback. For more information about configurable properties for each integration, see Deploying Probe for Prometheus Integration, Deploying the Probe for IBM Cloud Event Management Integration, or Deploying the Gateway for IBM Cloud Event Management Integration.
   - For Probe for Prometheus Integration or Probe for IBM CEM Integration:
     a. Update `spec.version: 4.5.0` to `spec.version: 4.6.0`.

```
spec:
  version: '4.6.0'
```

     b. Update the image parameters to upgrade to the latest container images.

```
spec:
  helmValues:
    image:
      repository: 'cp.icr.io/cp/noi-int/netcool-probe-messagebus'
      digest: 'sha256:db93b631dde5f724e0234842f56235893f1697e351e3f7e8b1e044905cbdcd07'
      tag: '13.1.0-amd64'
      testRepository: 'cp.icr.io/cp/noi-int/netcool-integration-util'
      testImageDigest:
'sha256:af9b0cf8ade76f2c32f48c54ce6ad8ad9e10a76f8af3940b5a18ff24b419f28c'
      testImageTag: '3.3.0-amd64'
```

   - For Gateway for IBM CEM Integration:
     a. Update `spec.version: 1.3.0` to `spec.version: 1.4.0`.

```
spec:
  version: '1.4.0'
```

     b. Update the image parameters to upgrade to the latest container images.

```
spec:
  helmValues:
    image:
      repository: 'cp.icr.io/cp/noi-int/netcool-gateway-cem'
      digest: 'sha256:d9612d1865d7bf13ef9c6083453e51a5f9bf4b45da196285dca14b0497f3d4b2'
      tag: '2.0.0-amd64'
```

3. Click the **Save** button.
4. Navigate to **Operators > Installed Operators**, and select **Project**. Search for and select the IBM Netcool Operations Insight Event Integrations operator.
5. Under the **All Instances** tab, view the status of each of the updates on the installation. When the instance's status shows **OK**, then the upgrade is successful. Optionally, verify that all pods are running to ensure that the upgrade has completed successfully.

Upgrading with the CLI

Use these instructions to upgrade an existing IBM Netcool® Operations Insight® Event Integrations Operator using the command line interface (CLI).

Upgrade the Catalog Source

1. Login to the cluster using the `oc` client.
2. In the target namespace, search for an existing `CatalogSource` resource for the IBM Netcool Operations Insight Event Integrations operator:

```
kubectl get catalogsource --namespace $namespace
```

   If it is not in the target namespace, run the same command in the `openshift-marketplace` namespace.

3. Edit the catalog source YAML and upgrade image (`spec.image`) with the IBM Netcool Operations Insight Event Integrations catalog source image with image tag, for example:

```
docker.io/ibmcom/ibm-netcool-integrations-operator-catalog:<latest version>
```

   or with the image with digest, where `<latest version>` is the latest catalog source image tag. Click the **Save** button.

4. Give a few minutes for the operator catalog to update and verify that the operator pod is restarted and it is in `Running` state.

Upgrade the IBM Netcool Operations Insight Event Integrations operator

You can skip this section if the opertor was installed in `Automatic` approval strategy. The operator should be updated automatically after the catalog source is upgraded.

1. Describe the existing `InstallPlan` resource for the operator:

```
kubectl get installplan --namespace $namespace
kubectl describe <install-plan resource>
```

   Replace `<install-plan resource>` with the name of the resource.

2. If the `Approved` field shows `RequiresApproval`, edit the `InstallPlan` resource and change the value value of `spec.approved` to `true`.
3. Run the following command to get the `.status.phase` field of the `InstallPlan` resource and ensure that the value of changes to `Complete`:

```
kubectl get installplan <install-plan resource> --namespace $namespace -o
jsonpath='{.status.phase}'
```

Upgrade the Probe or Gateway instances

1. Run the command below to obtain the existing instance resources in the target namespece:

```
kubectl get cemprobe,prometheusprobe,cemgateway --namespace $namespace
```

2. Edit the probe or gateway instance to update its specification using the

```
kubectl edit <resource type> <resource name>
```

   command. It is recommended that you take a copy of the instance YAML before changing it, in case you later decide to rollback. For more information about configurable properties for each integration, see Deploying Probe for Prometheus Integration, Deploying the Probe for IBM Cloud Event Management Integration, or Deploying the Gateway for IBM Cloud Event Management Integration.

   - For Probe for Prometheus Integration or Probe for IBM CEM Integration:

     a. Update `spec.version: 4.5.0` to `spec.version: 4.6.0`.

```
spec:
  version: '4.6.0'
```

     b. Update the image parameters to upgrade to the latest container images.

```
spec:
  helmValues:
    image:
      repository: 'cp.icr.io/cp/noi-int/netcool-probe-messagebus'
      digest: 'sha256:db93b631dde5f724e0234842f56235893f1697e351e3f7e8b1e044905cbdcd07'
      tag: '13.1.0-amd64'
      testRepository: 'cp.icr.io/cp/noi-int/netcool-integration-util'
      testImageDigest:
'sha256:af9b0cf8ade76f2c32f48c54ce6ad8ad9e10a76f8af3940b5a18ff24b419f28c'
      testImageTag: '3.3.0-amd64'
```

   - For Gateway for IBM CEM Integration:

a. Update `spec.version: 1.3.0` to `spec.version: 1.4.0`.

```
spec:
  version: '1.4.0'
```

b. Update the image parameters to upgrade to the latest container images.

```
spec:
  helmValues:
    image:
      repository: 'cp.icr.io/cp/noi-int/netcool-gateway-cem'
      digest: 'sha256:d9612d1865d7bf13ef9c6083453e51a5f9bf4b45da196285dca14b0497f3d4b2'
      tag: '3.0.0-amd64'
```

Rolling back process

Use these instructions to roll back an existing IBM Netcool® Operations Insight® Event Integrations Operator using the command line interface (CLI).

Before rolling back an operator, all instances that are running must be rolled back to the supported version of the previous operator.

Rolling back probe or gateway instances

1. Get the instance name:

```
kubectl get cemprobe,prometheusprobe,cemgateway --namespace $namespace
```

2. For each probe or gateway instance that was created with the current operator, revert the `spec.version` property.
   - For Probe for Prometheus Integration or Probe for IBM CEM Integration:
     a. Update `spec.version: 4.6.0` to `spec.version: 4.5.0`.

```
spec:
  version: '4.5.0'
```

   - For Gateway for IBM CEM Integration:
     a. Update `spec.version: 1.4.0` to `spec.version: 1.3.0`.

```
spec:
  version: '1.3.0'
```

3. Give a moment for the operator to process the update.

Rolling back the operator

To rollback the operator, you will need to uninstall the operator, revert the catalog source to the previous version and re-install the operator.

Note You should only roll back the operator if there is a critical error in the operator and it is unable to recover after restarting the pod. Errors in the operator could be caused due to an invalid configuration in the instance custom resource or version mismatch. Review your custom resource specification to investigate the issue.

Caution Uninstalling the operator while there are probe or gateway instances running may cause issues on the managed resources. It is recommended to uninstall or delete all instances before uninstalling the operator.

1. From the Red Hat OpenShift OLM UI, navigate to **Operators > Installed Operators**, select **Project**, and search for and select the IBM Netcool Operations Insight Event Integrations Operator. Under the **All Instances** tab, delete the instances. To delete instances via the command line, refer to the “Deleting an integration instance” on page 19 for more info.
2. Navigate to **Operators > Installed Operators**, select **Project**, search for and select the IBM Netcool Operations Insight Event Integrations Operator and uninstall the operator.

3. Navigate to **Administration > Cluster Settings**, and then select the **OperatorHub** configuration resource under the Global Configurations tab.
4. Under the **Sources** tab, click the existing IBM Netcool Operations Insight Event Integrations catalog source.
5. Edit the catalog source YAML and revert image (`spec.image`) with the IBM Netcool Operations Insight Event Integrations catalog source image with to the previous catalog source image. Click the **Save** button.
6. The operator catalog should refresh after a few minutes.
7. Select the menu navigation **OperatorHub** and search for the **IBM Netcool Operations Insight Event Integrations** Operator. Click the **Install** button.
8. Select a namespace to install the operator. Do not use namespaces that are kubernetes or openshift owned like kube-system or default. If you don't already have a project, create a project under the navigation menu **Home -> Projects**.
9. Select the **Subscribe** button.

Installing integration resources using the CLI

This method installs a probe or gateway using the CASE launch scripts. The launch script is executed using the `cloudctl` tool and will perform the necessary prerequisite checks for each inventory action that is performed.

Note: The launch scripts are provided the CASE from version 1.1.0.

Prerequisites:

- The IBM Netcool Operations Insight Event Integration Operator is installed in the target namespace.
- CASE archive downloaded and unpacked into your local file system.

1. Extract the custom resource template YAML files from the CASE archive.

```
tar -xvf $CASEPATH ibm-netcool-integrations/inventory/netcoolIntegrationsOperator/files/
deploy/crs
```

2. Configure the integrations by updating the custom resource template YAML file.
   a. To deploy the Probe for IBM Cloud Event Management (CEM), update `ibm-netcool-integrations/inventory/netcoolIntegrationsOperator/files/deploy/crs/probes.integrations.noi.ibm.com_v1beta1_cemprobe_cr.yaml`.
   b. To deploy the Probe for Prometheus, update `ibm-netcool-integrations/inventory/netcoolIntegrationsOperator/files/deploy/crs/probes.integrations.noi.ibm.com_v1beta1_prometheus_cr.yaml`.
   c. To deploy the Probe for Kafka Integration, update `ibm-netcool-integrations/inventory/netcoolIntegrationsOperator/files/deploy/crs/probes.integrations.noi.ibm.com_v1_kafkaprobe_cr.yaml`.
   d. To deploy the Gateway for IBM Cloud Event Management (CEM), update `ibm-netcool-integrations/inventory/netcoolIntegrationsOperator/files/deploy/crs/gateways.integrations.noi.ibm.com_v1beta1_cemgateway_cr.yaml`.
3. Run the `apply-custom-resources` action in the launch script and set the updated custom resource YAML file path using the `--crFile` for each custom resource to install. More than one instance of probe and gateway can be deployed in the same namespace. **Note:** The CR instance name must be unique to avoid conflicts.

```
cloudctl case launch --case $CASEPATH \
 --namespace $NAMESPACE     \
 --inventory netcoolIntegrationsOperator   \
 --action apply-custom-resources  \
 --args "--crFile $CRFILE" \
 --tolerance 1
```

Viewing the health of an operator

This section describes how to view the health of an operator.

To display the health status of the operator, run the following commands:

```
kubectl describe deployment netcool-integrations-operator --namespace <NAMESPACE>
kubectl describe pod -l app.kubernetes.io/instance==netcool-integrations-operator --namespace
<NAMESPACE>
```

To view the Operator pod logs to check if there are any errors, run the following command:

```
kubectl logs -l app.kubernetes.io/instance==netcool-integrations-operator --namespace
<NAMESPACE>
```

Pulling images from IBM Entitled Registry

To pull images from IBM Entitled Registry, use the following steps:

1. Log in to MyIBM Container Software Library with the IBMid and password that are associated with the entitled software.
2. In the **Entitlement keys** section, click **Copy key** to copy the entitlement key to the clipboard.
3. Save the APIkey.
4. Set the entitled registry information. Run export commands that set `ENTITLED_REGISTRY` to `cp.icr.io`, set `ENTITLED_REGISTRY_USER` to `cp`, and set `ENTITLED_REGISTRY_KEY` to the entitlement key that you got from the previous step.

   ```
   export ENTITLED_REGISTRY=cp.icr.io
   export ENTITLED_REGISTRY_USER=cp
   export ENTITLED_REGISTRY_KEY=<apikey>
   ```

5. Log in to the entitled registry with the following docker login command:

   ```
   docker login "$ENTITLED_REGISTRY" -u "$ENTITLED_REGISTRY_USER" -p "$ENTITLED_REGISTRY_KEY"
   ```

6. You can now pull images to your local file system.

Listing images in IBM Entitled Registry

Prerequisite: IBM Cloud CLI tool.

To install the `ibmcloud` CLI tool, see: https://cloud.ibm.com/docs/cli?topic=cli-install-ibmcloud-cli

To list images on the IBM Entitled Registry, use the following steps:

1. Install the Container Registry plug-in:

   ```
   ibmcloud plugin install container-registry -r 'IBM Cloud'
   ```

2. Log in to your IBM Cloud account:

   ```
   ibmcloud login -a https://cloud.ibm.com
   ```

3. Set the region as global:

   ```
   ibmcloud cr region-set global
   ```

4. List the available images by using the following command. This lists all Netcool Operations Insight Integrations images in the `noi-int` namespace:

   ```
   ibmcloud cr image-list --include-ibm | grep -I noi-int
   ```

Configuring custom resources

This section describes how to configure a custom resource (probe or gateway) instance.

To install a probe or gateway instance, you need to deploy a custom resource YAML in the namespace which is being watched by the Operator. See "Installing the operator" on page 6.

Refer to the deploying page of the custom resource for details about how to update the YAML for your environment.

Updating an integration instance

This section describes how to update an integration instance.

To update a probe or gateway instance, you can edit the specific custom resource deployed in the namespace. For example, to update a `PrometheusProbe` instance, use the following steps:

1. Get the instance resource YAML:

```
kubectl get prometheusprobe example-prometheusprobe --namespace <NAMESPACE> -o yaml > /tmp/
example-prometheusprobe-cr.yaml
```

2. Update the `/tmp/example-prometheusprobe-cr.yaml` configuration YAML and then deploy the updated YAML file:

```
kubectl apply -f /tmp/example-prometheusprobe-cr.yaml
```

Some probe and gateway configuration are exposed in `Configmap` resources such as probe rules files, probe and gateway properties file, and gateway table mapping and replication definitions. These configurations can be changed by updating the `Configmap` resource directly. The Operator should then perform the necessary updates to the pods. If required, you can scale down the number of replicas to zero and then scale up to the desired replica count to ensure that the probe and gateway pods reload the updated configurations from the `Configmap`.

Deleting an integration instance

This section describes how to delete an integration instance.

You can delete an integration instance by deleting the custom resource in the namespace. For example, to delete a `PrometheusProbe` resource, run the following command:

```
kubectl delete prometheusprobe example-prometheusprobe  --namespace <NAMESPACE>
```

Uninstalling the operator

You can uninstall the operator using the Operator Lifecycle Manager (OLM) UI, or with the command line interface.

Before deleting the Operator, ensure that you have deleted all instances by deleting the custom resources in the namespace. This is to ensure that all resources created by the existing instances are deleted properly. You may need to allow some time for the Operator to delete the resources before deleting the Operator deployment.

Using the OLM UI

This section describes how to delete the operator using the OLM UI.

1. Go to **Operator > Installed Operators**.
2. Select the project where you installed Netcool Operations Insight. Click **IBM Netcool Operations Insight Event Integrations Operator > All Instances** and delete the probe and gateway instances.

3. Delete the Netcool Operations Insight operator. Go to **Operator > Installed Operators**. Select the options menu for the IBM Netcool Operations Insight Event Integrations operator entry, and select **Uninstall Operator**.
4. Remove the catalog source entry. Go to **Administration > Cluster Settings > Global Configuration > OperatorHub > Sources**. Select the catalog source entry serving the operator and select **Delete CatalogSource**.
5. Delete the Custom Resource Definitions (CRDs). Go to **Administration > Custom Resource Definitions**, search for **integrations.noi** to select the CRDs under this group that were created by the operator installation and delete all the CRDs.

Using the CLI with CASE

This section describes how to uninstall the operator using the CASE launch script for offline airgap deployments, as well as online deployments.

1. Delete probe and gateway instances:

```
cloudctl case launch  \
  --case $CASEPATH    \
  --namespace $NAMESPACE     \
  --inventory netcoolIntegrationsOperator   \
  --action delete-custom-resources \
  --args "--crFile $CRFILE" \
  --tolerance 1
```

   Where $CRFILE variable is the path to the custom resource file that was use to create the instance and $CASEPATH is the path to the CASE archive in your local file system.

2. Delete the operator:

```
cloudctl case launch  \
  --case $CASEPATH    \
  --namespace $NAMESPACE     \
  --inventory netcoolIntegrationsOperatorSetup   \
  --action uninstall-operator  \
  --args "--inputDir $PWD/case" \
  --tolerance 1
```

   Optionally, you can also set the `--deleteOperatorGroup true` option to also remove the OperatorGroup resource when deleting the Operator. For example:

```
cloudctl case launch  \
  --case $CASEPATH    \
  --namespace $NAMESPACE     \
  --inventory netcoolIntegrationsOperatorSetup   \
  --action uninstall-operator  \
  --args "--inputDir $PWD/case --deleteOperatorGroup true" \
  --tolerance 1
```

3. To uninstall the catalog source, run the following command:

```
 cloudctl case launch  \
  --case $CASEPATH  \
  --namespace $NAMESPACE     \
  --inventory netcoolIntegrationsOperatorSetup   \
  --action uninstall-catalog  \
  --args "--inputDir $PWD/case" \
  --tolerance 1
```

Using the CLI (native)

This section describes how to delete the operator using the CLI (native).

You can remove the operator deployment using the custom resource file using the following command:
`kubectl delete -f deploy/operator.yaml`

Alternatively, you can remove the operator deployment using the CLI by specifying the deployment name:
`kubectl delete deployment netcool-integrations-operator --namespace <NAMESPACE>`

Limitations

This section describes various limitations that currently exist.

The IBM Netcool Operations Insight Event Integrations Operator has the following limitations:

- Only the AMD64 / x86_64 architecture is supported.
- CEM and Prometheus integrations are verified on Red Hat OpenShift Container Platform 4.5 and 4.6.
- There are several known limitations when enabling secure connection between probe clients and server:
  - The required files in the secret must be created using the `nc_gskcmd` utility.
  - If your ObjectServer is configured with FIPS 140-2, the password for the key database file (`omni.kdb`) must meet the requirements stated in the the following Knowledge Center page: https://www.ibm.com/support/knowledgecenter/SSSHTQ_8.1.0/com.ibm.netcool_OMNIbus.doc_8.1.0/omnibus/wip/install/task/omn_con_ssl_creatingkeydatabasefips.html.
  - When encrypting a string value using the encryption config key file (`encryption.keyfile`), you must use the `AES_FIPS` as the cipher algorithm. AES algorithm is not supported.
  - When connecting to an ObjectServer in the same cluster, you can connect the gateway to the secure connection proxy which is deployed with the IBM Netcool Operations Insight ObjectServer to encrypt the communication using TLS but the TLS termination is done at the proxy. It is recommended to enable Mutual Transport Layer Security (mTLS) in Service Mesh to encrypt cluster data network traffic. For more information, refer to the ServiceMeshControlPlane parameters section in the Service Mesh installation, usage, and release notes.
- Only one active CEM Gateway pod can use the Store-and-Forward (SAF) directory which is stored in the Persistent Volume. It is advised to scale down the StatefulSet to 0 and then back to 1 in order to properly start the CEM Gateway during a node maintenance and it should be able to resume processing the last known SAF file.
- There is no upgrade path from probe or gateway Helm charts to this Operator. Upgrades and rollback from Helm charts to Operator is not supported. Users must migrate the Helm release configuration to the respective probe or gateway custom resource YAML, deploy the custom resource YAML to install the probe or gateway, verify the integration is successful and working as expected before deleting the old Helm release.
- Please ensure that your cluster has internet access for the operator to pull the container images from entitled registry and public registry.

Troubleshooting

The following table describes how to troubleshoot issues when deploying the operator and how to resolve them.

Table 2. Problems

Problem	Cause	Resolution
Probe logs show an error when loading or reading rules files. Failed during field verification check. Fields `SiteName` and `ScopeID` not found	The OMNIbus ObjectServer event grouping automation is not installed, hence the required fields are missing.	Install the event grouping automation in your ObjectServer and redeploy the probe or gateway.
The following error occurred when deploying the probe with PodDisruptionBudget enabled: `poddisruptionbudgets.policy is forbidden`	The user deploying the probe does not have the correct role to deploy the probe with **`PodDisruptionBudget`** enabled.	Administrator or Cluster Administrator role is required to deploy the probe with **`PodDisruptionBudget`** enabled.

Table 2. Problems (continued)

Problem	Cause	Resolution
The probe deployment failed to mount the ConfigMaps or some object name appeared to be somewhat random.	If a long instance name is used, the operator will generate a random suffix for objects that exceeds the character limit. This might cause mapping issues between the Kubernetes objects.	Use a shorter instance name, namely less than 20 characters.
Pod failed to mount the ConfigMaps or some object name appear to be somewhat random.	If a long name is used, the operator will generate a random suffix for objects that exceeds the character limit. This might cause mapping issues between the Kubernetes objects.	Use a shorter name, namely less than 20 characters.
In an offline or airgap installation mode, an error may occur when trying to mirror container images in an internal registry if there is already an existing repository for any of the images. If the following error is shown when pushing the container images to your existing internal registry, it is likely due to the recent changes made in the CASE inventory resources.yaml file. `api.sqaocp4611.cp.fyre.ibm .com:5000 cp/noi-int/ibm- netcool-integrations- operator blobs=0 mounts=0 manifests=1 shared=0 info: Planning completed in 1.51s sha256:9f53d5e340fc0486a2f 0cd3130bd72733f0420bbcc054 b19400c7fcbd3a6763a api.sqaocp4611.cp.fyre.ibm .com:5000/cp/noi-int/ibm- netcool-integrations- operator-bundle:1.2.0- amd64 error: unable to push manifest to api.sqaocp4611.cp.fyre.ibm .com:5000/cp/noi-int/ netcool-integration- util:3.3.0-amd64: errors: manifest blob unknown: blob unknown to registry manifest blob unknown: blob unknown to registry manifest blob unknown: blob unknown to registry manifest blob unknown: blob unknown to registry manifest blob unknown: blob unknown to registry`	There is an update to the CASE inventory `resources.yaml` files where the media type of all container images has been updated to `application/ vnd.docker.distribution.m anifest.v2` from `application/ vnd.oci.image.index.v1` for consistency with other IBM container images. The media type affects which digest the image registry returns which may cause an error if you try to push the new images to an existing registry with the old media type.	Delete the existing images in the internal registry or recreate the registry by running the CASE launch action (mirror-images) to initialize a new registry and then push the images into it.

Known issues

This section describes the known issues that currently exist.

Operator installation hangs when using the OCP Web console

If you observe that the Operator installation via the OCP Web console hangs either in Upgrading or Installing state indefinitely, this could be due to a problem in the Operator Catalog Source in resolving one or more images required by the Operator Lifecycle Manager (OLM).

Workaround:

Download the PPA archive from Passport Advantage and follow the steps to install the operator using the CLI.

Chapter 2. Deploying integrations

For details of the integrations that you can deploy using the NOI Operator, see the following sections:

- “Deploying the Gateway for IBM Cloud Event Management Integration” on page 25
- “Deploying the Probe for IBM Cloud Event Management Integration” on page 33
- “Deploying the Probe for Prometheus Integration” on page 39

Deploying the Gateway for IBM Cloud Event Management Integration

Note: The NOI Event Integrations Operator must be deployed before deploying this integration. See “Installing the operator” on page 6. See also “Pre-installation tasks” on page 29

To deploy a new instance of the gateway, review the custom resource YAML template supplied with the operator package, update the parameters if required, and save the configurable parameters in a file, for example `example-cemgateway-cr.yaml`.

Then deploy the custom resource using the updated file:

```
kubectl apply -f gateways.integrations.noi.ibm.com_v1beta1_cemgateway_cr.yaml --namespace
<NAMESPACE>
```

The following table describes the configurable parameters for this gateway and lists their default values.

Configurable parameters

Parameter name	Description
spec.license.accept	The license state of the image being deployed. Enter `true` to install and use the image. The default value is `false`.
spec.version	Gateway application version. Accepted versions are `1.2.0`, `1.3.0` and `1.4.0`. Default value is `1.4.0`.
spec.helmValues.image.repository	Gateway image repository. Update this repository name to pull from a private image repository. The default value is `cp.icr.io/cp/noi-int/netcool-gateway-cem`, and the image name must not be changed.
spec.helmValues.image.tag	The specific gateway image tag to use. A tag is a label applied to a image in a repository. Tags are how various images in a repository are distinguished from each other. The default value is `3.0.0-amd64`.
spec.helmValues.image.digest	The digest to identify the gateway image. If unspecified, the image tag is used. The default value is `sha256:d9612d1865d7bf13ef9c6083453e51a5f9bf4b45da196285dca14b0497f3d4b2`.

Parameter name	Description
spec.helmValues.image.pullPolicy	The image pull policy. The default value is `Always`.
spec.helmValues.global.image.secretName	The name of the secret containing the docker config to pull the image from a private repository. Leave this parameter blank if the gateway image already exists in the local image repository or the Service Account has a been assigned with an Image Pull Secret. The default value is `nil`.
spec.helmValues.global.image.useTag	Configures pods to pull images using tags or digests. Set this property to `false` to use digest to identify the container images. Set this property to `true` to use tags instead of digests to identify the container images. Default value is `false`.
spec.helmValues.global.serviceAccountName	Name of the service account to be used by the pods. If the Cluster Administrator has already created a service account in the namespace, specify the name of the service account here. If left blank, the gateway will automatically create a new service account in the namespace when it is deployed. This new service account will be removed from the namespace when the gateway is removed. The default is `nil`.
spec.helmValues.global.persistence.useDynamicProvisioning	Use Storage Class to dynamically create Persistent Volume and Persistent Volume Claim. The default is `true`.
spec.helmValues.global.persistence.storageClassName	Storage Class for dynamic provisioning. The default is `nil`.
spec.helmValues.global.persistence.selector.label	The Persistent Volume Claim Selector label key to refine the binding process when dynamic provisioning is not used. The default is `nil`.
spec.helmValues.global.persistence.value	The Persistent Volume Claim Selector label value related to the Persistent Volume Claim Selector label key. The default is `nil`.
spec.helmValues.global.persistence.storageSize	Storage size to store CEM Gateway Store and Forward Files. The default is `3Gi`.
spec.helmValues.global.persistence.supplementalGroups	Provide the GID of the volumes as list (required for NFS). The default is `[]`.
spec.helmValues.netcool.connectionMode	The connection mode to use when connecting to the Netcool/OMNIbus ObjectServer. Refer to Securing probe and ObjectServer communications for more details. **Note:** Refer to the limitations section for more details on available connection modes for your environment. The default is `default`.

Parameter name	Description
spec.helmValues.netcool.primaryServer	The primary Netcool/OMNIbus server to connect to. This is usually set to NCOMS or AGG_P. The default value is AGG_P.
spec.helmValues.netcool.primaryHost	The host of the primary Netcool/OMNIbus server. Specify the ObjectServer hostname or IP address. The default value is nil
spec.helmValues.netcool.primaryIP	The primary Netcool/OMNIbus ObjectServer IP address. If specified along with primaryHost, a host alias entry will be added. The default is nil.
spec.helmValues.netcool.primaryPort	The port of the primary Netcool/OMNIbus ObjectServer. The default value is 4100.
spec.helmValues.netcool.primaryIDUCHost	The primary Netcool/OMNIbus ObjectServer IDUC Host or Service name. Should be set if the primary IDUC host is different from the primary ObjectServer hostname. When connecting to NOI on OCP, this should be set to the primary ObjectServer NodePort service name. The default is nil.
spec.helmValues.netcool.backupServer	The backup Netcool/OMNIbus server to connect to. If the **backupServer**, **backupHost** and **backupPort** parameters are defined in addition to the **primaryServer**, **primaryHost**, and **primaryPort** parameters, the gateway will be configured to connect to a virtual ObjectServer pair called AGG_V. If no backup ObjectServer is configured, only the primary server parameters will be used. The default value is nil.
spec.helmValues.netcool.backupHost	The host of the backup Netcool/OMNIbus ObjectServer. Specify the ObjectServer hostname. The default value is nil
spec.helmValues.netcool.backupIP	The backup Netcool/OMNIbus ObjectServer IP address. If specified along with primaryHost, a host alias entry will be added. The default is nil.
spec.helmValues.netcool.backupPort	The port of the backup Netcool/OMNIbus ObjectServer. The default value is none.
spec.helmValues.netcool.backupIDUCHost	The backup Netcool/OMNIbus ObjectServer IDUC Host or Service name. Should be set if the primary IDUC host is different from the primary ObjectServer hostname. When connecting to NOI on OCP, this should be set to the primary ObjectServer NodePort service name. The default is nil.

Parameter name	Description
spec.helmValues.netcool.secretName	This is a pre-created secret for `AuthOnly`, `SSLOnly`, or `SSLAndAuth` connection mode. Certain fields are required depending on the connection modes. The default is `nil`.
spec.helmValues.cemgateway.messageLevel	The gateway log message level. The default value is `warn`.
spec.helmValues.cemgateway.connections	The number of simultaneous connections the gateway makes with the Cloud Event Management. The default value is 3
spec.helmValues.cemgateway.connectionTimeout	The interval (in seconds) that the gateway allows for HTTP connections and responses to HTTP requests. The default value is 15
spec.helmValues.cemgateway.retryLimit	The maximum number of retries the gateway should make on an operation (for example, forwarding an event to the Event Source instance) that failed. The default value of 0 (zero) means there is no limit to the number of retries that the gateway makes on a failed operation. The default value is `0`.
spec.helmValues.cemgateway.retryWait	The number of seconds the gateway should wait before retrying an operation (for example, forwarding an event to the Event Source instance) that failed. The default value is 7.
spec.helmValues.cemgateway.reconnectTimeout	The time (in seconds) between each reconnection poll attempt that the gateway makes if the connection to the ObjectServer is lost. The default value is 30.
spec.helmValues.cemgateway.setUIDandGID	When set to true, the operator will specify the UID and GID values for the netcool user else the netcool user will not be assigned any UID or GID by the operator. The default value is `true`.
spec.helmValues.cemgateway.locale	Environment locale setting. Used as the LC_ALL environment variable. The default is `en_US.utf8`.
spec.helmValues.cemgateway.cemTlsSecretName	A pre-created secret name to store CEM certificate. In the secret, the key must be tls.crt which holds the certificate file in PEM format. The default value is `nil`.

Parameter name	Description
spec.helmValues.cemgateway.cemSecretName	A pre-created secret to store CEMWebhookURL, NewKeystorePassword and HttpAuthenticationPassword. CEMWebhookURL is the CEM webhook URL to send notification from gateway to CEM and the required key in the secret. NewKeystorePassword is the new password to access cacerts in gateway and the optional key in the secret. HttpAuthenticationPassword is the HTTP basic authentication password string which is required by the test pod and only used by an internal API to check the liveness of the CEM Gateway. HttpAuthenticationPassword is the optional key in the secret. The default value is `true`.
spec.helmValues.resources.limits.cpu	The container CPU limit. The default value is `500m`.
spec.helmValues.resources.limits.memory	The container memory limit. The default value is `512Mi`.
spec.helmValues.resources.requests.cpu	The container CPU requested. The default value is `100m`.
spec.helmValues.resources.requests.memory	The container Memory requested. The default value is `128Mi`.
spec.helmValues.arch	The worker node architecture. This is set to `amd64`, and cannot be changed.

Pre-installation tasks

Before installing the gateway, you must perform the following tasks:

1. Gathering ObjectServer details
2. Preparing ObjectServer communication secret
3. Preparing the Cloud Event Management (CEM) incoming integration
4. Preparing the Cloud Event Management (CEM) gateway secret
5. Preparing the Cloud Event Management (CEM) TLS secret
6. Pre-creating a Persistent Volume (PV)

1. Gathering ObjectServer details

For the gateway to successfully connect to ObjectServer, the gateway must be configured with:

1. ObjectServer host or service details
2. ObjectServer TLS certificate if SSL protected communication is enabled.
3. Credentials to authenticate with the ObjectServer.

Note : In production environments, it is recommended to use TLS/SSL enabled communications with the ObjectServer.

Follow these steps to obtain the required details for the ObjectServer:

1. Contact your administrator to find out the ObjectServer that the CEM Gateway should connect to. Note the NOI release name and namespace as `NOI_RELEASE_NAME` and `NOI_NAMESPACE` respectively. This information will be used when preparing the ObjectServer communication Kubernetes secret later.
2. Determine the ObjectServer NodePort service name and port number. Refer to Connecting with the ObjectServer NodePort for more info on identifying the ObjectServer NodePort and note down the service names and the NodePort number as `NOI_OBJECT_SERVER_PRIMARY_SERVICE` and `NOI_OBJECT_SERVER_PRIMARY_PORT` respectively. You may also note the backup ObjectServer service if you wish to connect to the backup ObjectServer too as `NOI_OBJECT_SERVER_BACKUP_SERVICE` and `NOI_OBJECT_SERVER_BACKUP_PORT` respectively . This information will be used when configuring the gateway.
3. Determine the TLS Proxy listening port by following the steps in Identifying the proxy listening port page and note down the TLS proxy Common Name (CN) and port number as `NOI_TLS_PROXY_CN` and `NOI_TLS_PROXY_PORT` respectively. This information will be used when configuring the gateway.
4. Determine the TLS Proxy certificate Kubernetes secret that should be used by the CEM Gateway. This is usually set in `{{ Release.Name}}-proxy-tls-secret` secret, where `{{ Release.Name }}` is the NOI release name. Note down the secret name and namespace. This information will be used when preparing the ObjectServer communication Kubernetes secret later.
5. Get the ObjectServer user password which is required by the Gateway when creating and Insert, Delete, Update, or Control (IDUC) communication protocol connection. The credential is provided in the `{{ Release.Name }}-omni-secret`, where `{{ Release.Name }}` is the NOI release name.
6. Obtain the cluster proxy IP address.

The "Gathered Facts" table below lists the details that is gathered from the above steps. These items will be referenced in the following sections.

Item	**Description and sample value**
CLUSTER_MASTER_IP	The cluster master node IP address. This should be set as the `netcool.primaryIP` and optionally `netcool.backupIP` to also connect to the backup ObjectServer.
CLUSTER_MASTER_HOST	The cluster master node hostname. This should be set as the `netcool.primaryHost` and optionally `netcool.backupHost` to also connect to the backup ObjectServer.
NOI_RELEASE_NAME	The NOI release name. For example, `noi-m76`
NOI_NAMESPACE	Namespace where NOI is installed. For example, `default`
NOI_OBJECT_SERVER_PRIMARY_SERVICE	The primary ObjectServer Nodeport service. For example, `noi-m76-objserv-agg-primary-nodeport`. This should be set as the `netcool.primaryIDUCHost` parameter.
NOI_OBJECT_SERVER_PRIMARY_PORT	The primary ObjectServer Nodeport number. This should be set as the `netcool.primaryPort` parameter.
NOI_OBJECT_SERVER_BACKUP_SERVICE	(Optional) The backup ObjectServer Nodeport service. For example `noi-m76-objserv-agg-backup-nodeport`. This should be set as the `netcool.backupIDUCHost` parameter.

Item	Description and sample value
NOI_OBJECT_SERVER_BACKUP_PORT	(Optional) The backup ObjectServer Nodeport number
NOI_TLS_PROXY_CN	The NOI TLS certificate subject Common Name. For example `proxy.noi-m76.mycluster.icp`. This should be set as the `netcool.primaryHost` (and optionally `netcool.backupHost`). For more details on how to obtain the Subject Common Name, see Configuring TLS encryption with Red Hat OpenShift or Configuring TLS encryption with a custom certificate.
NOI_TLS_PROXY_PORT_1	The NOI TLS Proxy service port number (first port).
NOI_TLS_PROXY_PORT_2	(Optional) The NOI TLS Proxy service port number (second port), required to connect to backup ObjectServer.
NOI_TLS_SECRET_NAME	Secret name containing the TLS certificate of the TLS Proxy. For example `noi-m76-proxy-tls-secret`
NOI_OMNI_USER	Username for IDUC connection.
NOI_OMNI_PASSWORD	Password for IDUC connection.

2. Preparing ObjectServer communication secret

The secret can be created using the utility script ("create-noi-secret.sh") provided in the `ibm-netcool-integrations/inventory/ibmNetcoolGatewayCEMSetup/files directory in the downloaded archive` directory. Before running this script, several pieces of information must be gathered and then configured in the script's configuration file ("create-noi-secret.config") which should be obtained following the steps in "Gathering ObjectServer Details" section above.

For this task, you will need the CEM Gateway image in your local file system because the script requires several utility commands such as `nco_gskcmd` and `nco_aes_crypt` to add the ObjectServer certificate into the key database.

Note : If you are using a root CA signer and want to import it into the Key Database file as a trusted CA signer, see the following Technote for instructions: https://www.ibm.com/support/pages/node/1274896

1. Follow the steps in “Pulling images from IBM Entitled Registry” on page 18 to pull the CEM Gateway image `netcool-gateway-cem` into your local file system. The following steps uses `netcool-gateway-cem:latest` as the image name and image tag for simplicity. You can list the images on the registry by following the steps in “Listing images in IBM Entitled Registry” on page 18 and select the latest `netcool-gateway-cem` image to pull.
2. From the command line, review and update the create-noi-secret.sh utility script configuration file (`create-noi-secret.config`) file provided in the `ibm-netcool-integrations/inventory/ibmNetcoolGatewayCEMSetup/files directory in the downloaded archive` directory. Several items from the "Gathered Facts" table above is required when configuring the script.
3. Run the "create-noi-secret.sh" to create the Gateway-ObjectServer communication secret. The script should be run as an administrator or a user with read permissions to the NOI TLS secret so that the script can retrieve the TLS certificate file.
4. Optionally, verify that the secret is successfully created using the `kubectl describe secret <secret name> --namespace <namespace>` command.

```
kubectl describe secret cem-gw-noi-secret --namespace cemgw-ns
Name:         cem-gw-noi-secret
Namespace:    cemgw-ns
Labels:       <none>
```

```
Annotations:  <none>

Type:  Opaque

Data
====
omni.sth:            193 bytes
AuthPassword:        70 bytes
AuthUserName:        4 bytes
encryption.keyfile:  36 bytes
omni.kdb:            10088 bytes
```

3. Preparing the Cloud Event Management (CEM) incoming integration

To forward events from IBM Netcool Operations Insight (NOI) to IBM CEM, the `ibm-cem` gateway can be installed in the same cluster as NOI or in another cluster. NOI and the CEM Gateway need to be installed in the same cluster. For detailed steps on installing and configuring IBM CEM, see Installing with Netcool Operations Insight.

The CEM Gateway requires the CEM webhook URL from the Netcool/OMNIbus Incoming Integration in CEM. Follow the steps below to create the incoming integration.

1. Login to the CEM User Interface as an administrator.
2. Click **Integrations** on the CEM **Administration** page.
3. Click **New integration**.
4. Go to the **Netcool/OMNIbus** tile and click **Configure**.
5. Enter a name for the integration and click **Copy** to add the generated webhook URL to the clipboard. Ensure you save the generated webhook to a file. The CEM webhook URL is required when creating CEM Gateway secret.
6. Enable this integration.
7. Click the **Save** button.

4. Preparing the Cloud Event Management (CEM) gateway secret

The gateway requires CEM Gateway Secret to send the events to CEM. This secret contains sensitive information such as `CEMWebhookURL`, `NewKeystorePassword` and `HttpAuthenticationPassword`. `CEMWebhookURL` is the CEM webhook URL and it is the required key in the secret. `NewKeystorePassword` is the new password to access cacerts in gateway and it is an optional key in the secret. `HttpAuthenticationPassword` is the HTTP basic authentication password string which is required by the test pod (`gateway test`). This is only used by an internal API to check the liveness of the CEM Gateway and it is an optional key in the secret.

You can refer to the example below to create the secret. In the example, `cem-gateway-secret` is the secret name. `cem-gateway-namespace` is the namespace where the CEM gateway will be installed. `cem-webhook-url` is CEM webhook URL. `http-basic-authentication-password` is HTTP basic authentication password. `new-keystore-password` is the new password to access cacerts in gateway. The secret and gateway must reside in the same namespace.

```
kubectl create secret generic cem-gateway-secret --namespace cemgw-ns \
--from-literal=CEMWebhookURL=cem-webhook-url \
--from-literal=NewKeystorePassword=new-keystore-password \
--from-literal=HttpAuthenticationPassword=http-basic-authentication-password
```

Optionally, verify that the secret is successfully created using the `kubectl describe secret cem-gateway-secret --namespace cem-gateway-namespace` command.

```
kubectl describe secret cem-gateway-secret --namespace cemgw-ns
Name:         cem-gateway-secret
Namespace:    cemgw-ns
Labels:       <none>
Annotations:  <none>

Type:  Opaque
```

```
Data
====
CEMWebhookURL:               60 bytes
NewKeystorePassword:         10 bytes
HttpAuthenticationPassword:  10 bytes
```

5. Preparing the Cloud Event Management (CEM) TLS secret

To allow the CEM Gateway to send events to CEM on OCP, a Transport Layer Security (TLS) certificate for Fully Qualified Domain Name (FQDN) of the CEM must be obtained to establish a secure trusted connection between the CEM Gateway and CEM. You must create the secret, if the TLS certificate is not signed by a well known certificate authority. Follow these steps to obtain the CEM TLS certificate from the CEM Ingress TLS secret.

1. Login to the cluster where CEM installed.
2. Obtain the CEM TLS certificate from the CEM Ingress TLS secret. The sample command below can be used to obtain the CEM TLS certificate from the CEM Ingress TLS secret. In the sample command below, `cem-tls-secret-name` is the CEM Ingress TLS secret name, `cem-tls-secret-namespace` is the namespace where the secret created, and `tls.crt` is the file contains the command output.

   ```
   kubectl get secret cem-tls-secret-name \
   --namespace cem-tls-secret-namespace \
   -o json | grep tls.crt | cut -d : -f2 | cut -d '"' -f2 | base64 --decode > tls.crt
   ```

3. Login to the cluster where the CEM Gateway will be installed and create the CEM TLS secret. The sample command below can be used to create CEM TLS secret. In the sample command that follows, `cem-tls-secret` is the secret name, `cemgw-ns` is the namespace where the CEM gateway will be installed and `tls.crt` is a CEM TLS certificate file. The CEM TLS certificate filename must be `tls.crt`.

   ```
   kubectl create secret generic cem-tls-secret \
   --namespace cemgw-ns \
   --from-file=tls.crt
   ```

6. Pre-creating a Persistent Volume (PV)

The gateway requires a Persistent Volume (PV) to store Store and Forward (SAF) files. You can opt to use dynamic provisioning and skip this step, if your cluster supports dynamic provisioning. Otherwise, to pre-create a PV or if you want the gateway Persistent Volume Claim (PVC) to bind to a pre-created PV. You should only perform one of the following steps to pre-create a PV.

- Refer to Understanding persistent storage to pre-create a PV of your choice.
- If your cluster supports Network File System (NFS) PV. Refer to the sample YAML file of creating NFS PV in `ibm-netcool-integrations/inventory/ibmNetcoolGatewayCEMSetup/files directory in the downloaded archive/clusterAdministration/ibm-netcool-gateway-cem-prod-nfs-pv.yaml` to pre-create a NFS PV. Refer to comments in `ibm-netcool-gateway-cem-prod-nfs-pv.yaml` and update the YAML file accordingly. Then, run this command `kubectl create -f ibm-netcool-gateway-cem-prod-nfs-pv.yaml` as a cluster administrator to provision a NFS PV. The details of creating NFS PV are available in Persistent storage using NFS.

Deploying the Probe for IBM Cloud Event Management Integration

Note: The NOI Event Integrations Operator must be deployed before deploying this integration. See “Installing the operator” on page 6.

To deploy a new instance of the probe, review the custom resource YAML template supplied with the operator package, update the parameters if required, and save the configurable parameters in a file, for example `example-cemprobe-cr.yaml`.

Then deploy the custom resource using the updated file:

```
kubectl apply -f probes.integrations.noi.ibm.com_v1beta1_cemprobe_cr.yaml --namespace
<NAMESPACE>
```

The following table describes the configurable parameters for this probe and lists their default values.

Configurable parameters

Parameter name	Description
spec.license.accept	The license state of the image being deployed. Enter `true` to install and use the image. The default value is `false`.
spec.version	Probe application version. Accepted versions are `4.4.0`, `4.5.0` and `4.6.0`. The default value is `4.6.0`.
spec.helmValues.image.repository	Probe image repository. Update this repository name to pull from a private image repository. The default value is `cp.icr.io/cp/noi-int/netcool-probe-messagebus`, and the image name must not be changed.
spec.helmValues.image.tag	The probe image tag. The default value is `13.2.0-amd64`.
spec.helmValues.image.digest	The digest to identify the probe image. If unspecified, the image tag is used. The default value is `sha256:db93b631dde5f724e0234842f56235893f1697e351e3f7e8b1e044905cbdcd07`.
spec.helmValues.image.testRepository	Utility image repository. Update this repository name to pull from a private image repository. The default is `cp.icr.io/cp/noi-int/netcool-integration-util`, and the image name must not be changed.
spec.helmValues.image.testImageDigest	Digest to identify the probe image. If unspecified, the image tag is used. The default value is `sha256:af9b0cf8ade76f2c32f48c54ce6ad8ad9e10a76f8af3940b5a18ff24b419f28c`.
spec.helmValues.image.testImageTag	Utility image tag. The default is `3.4.0-amd64`.
spec.helmValues.image.pullPolicy	The image pull policy. The default value is `Always`.
spec.helmValues.global.image.secretName	The name of the secret containing the docker config to pull the image from a private repository. Leave this parameter blank if the probe image already exists in the local image repository or the Service Account has a been assigned with an Image Pull Secret. The default value is `nil`.

Parameter name	Description
spec.helmValues.global.image.useTag	Configures pods to pull images using tags or digests. Set this property to `false` to use digest to identify the container images. Set this property to `true` to use tags instead of digests to identify the container images. Default value is `false`.
spec.helmValues.global.serviceAccountName	Name of the service account to be used by the pods. If the Cluster Administrator has already created a service account in the namespace, specify the name of the service account here. If left blank, the probe will automatically create a new service account in the namespace when it is deployed. This new service account will be removed from the namespace when the probe is removed. The default is `nil`.
spec.helmValues.netcool.connectionMode	The connection mode to use when connecting to the Netcool/OMNIbus ObjectServer. Refer to Securing probe and ObjectServer communications for more details. **Note:** Refer to the limitations section for more details on available connection modes for your environment. The default is `default`.
spec.helmValues.netcool.primaryServer	The primary Netcool/OMNIbus server to connect to. This is usually set to `NCOMS` or `AGG_P`. The default value is `nil`.
spec.helmValues.netcool.primaryHost	The host of the primary Netcool/OMNIbus server. Specify the ObjectServer hostname or IP address. The default value is `nil`
spec.helmValues.netcool.primaryPort	The port of the primary Netcool/OMNIbus server. The default value is `4100`.
spec.helmValues.netcool.backupServer	The backup Netcool/OMNIbus server to connect to. If the **backupServer**, **backupHost** and **backupPort** parameters are defined in addition to the **primaryServer**, **primaryHost**, and **primaryPort** parameters, the probe will be configured to connect to a virtual object server pair called `` `AGG_V` ``. If no backup ObjectServer is configured, only the primary server parameters will be used. The default value is `nil`.
spec.helmValues.netcool.backupHost	The host of the backup Netcool/OMNIbus server. Specify the ObjectServer hostname or IP address. The default value is `nil`
spec.helmValues.netcool.backupPort	The port of the backup Netcool/OMNIbus server. The default value is `nil`.
spec.helmValues.netcool.secretName	This is a pre-created secret for `AuthOnly`, `SSLOnly` or `SSLAndAuth` connection mode. Certain fields are required depending on the connection modes. The default is `nil`.

Parameter name	Description
spec.helmValues.probe.messageLevel	The probe log message level. The default value is `warn`.
spec.helmValues.probe.setUIDandGID	When set to true, the operator will specify the UID and GID values for the netcool user else the netcool user will not be assigned any UID or GID by the operator. Default value is `true`.
spec.helmValues.probe.sslServerCommonName	Set this parameter to a comma-separated list of acceptable SSL common names for when connecting to the ObjectServer using SSL. This should be set when the `CommonName` field of the received certificate does not match the name specified by the **`primaryServer`** property. When a **`backupServer`** is specified, the probe creates an ObjectServer pair with the name `AGG_V`. Set this parameter if the `CommonName` of the certificate does not match `AGG_V`. The default is `nil`.
spec.helmValues.probe.locale	The container environment locale settings.
spec.helmValues.cemProbe.enabled	Set this parameter to `true` to enable the probe for CEM. The default value is `true`.
spec.helmValues.cemProbe.tlsEnabled	Enables the CEM TLS Webhook for secure connectivity between the probe and CEM. Default is `false`.
spec.helmValues.cemProbe.replicaCount	The number of deployment replicas of the CEM Probe. This will be ignored if `cemProbe.autoscaling.enabled=true` and will use the **minReplicas** as the **replicaCount**. The default value is `1`.
spec.helmValues.cemProbe.ingress.enabled	Set this parameter to `true` to enable Ingress. The default value is `false`.
spec.helmValues.cemProbe.ingress.host	Sets the virtual host name for the same IP address. The value must have the Ingress controller default domain as a suffix. Default value is `nil`.
spec.helmValues.cemProbe.autoscaling.enabled	Set this parameter to `false` to disable auto-scaling. The default value is `true`.
spec.helmValues.cemProbe.autoscaling.minReplicas	The minimum number of probe replicas. The default value is `1`.
spec.helmValues.cemProbe.autoscaling.maxReplicas	The maximum number of probe replicas. The default value is 3.
spec.helmValues.cemProbe.autoscaling.cpuUtil	The target percentage CPU utilization. For example, enter `60` for 60% target utilization. The default value is `60`.

Parameter name	Description
spec.helmValues.cemProbe.poddisruptionbudget.enabled	Set this parameter to `true` to enable Pod Disruption Budget to maintain high availability during a node maintenance. Administrator role or higher is required to enable Pod Disruption Budget on clusters with Role Based Access Control. The default value is `false`.
spec.helmValues.cemProbe.poddisruptionbudget.minAvailable	The minimum number of available pods during node drain. This can be set to a number or a percentage, for example: 1 or 10%. **Caution:** Setting this parameter to 100%, or to the number of replicas, may block node drains entirely. The default value is 1.
spec.helmValues.resources.limits.cpu	The container CPU limit. The default value is `500m`.
spec.helmValues.resources.limits.memory	The container memory limit. The default value is `512Mi`.
spec.helmValues.resources.requests.cpu	The container CPU requested. The default value is `100m`.
spec.helmValues.resources.requests.memory	The container memory requested. The default value is `128Mi`.
spec.helmValues.arch	The worker node architecture. This is set to `amd64`, and cannot be changed.

Post-installation steps for the CEM Probe

This section describes the post-installation steps required for the CEM Probe

After deploying a `CEMProbe` instance, follow the procedure below to create an outgoing integration, register the probe's webhook endpoint, and create an event forwarding policy in IBM CEM.

1. The CEM Probe webhook can be reached via its service from within the cluster. For example, if the `CEMProbe` instance name is `example-cemprobe`, then the webhook URL is `http://example-cemprobe-ibm-netcool-probe-cemprobe.<NAMESPACE>/probe/webhook/cem` , where `<NAMESPACE>` is the namespace where the probe is deployed. If ingress or route is enabled, then the probe webhook is exposed externally via a Route resource and the hostname should be the value set in the custom resource YAML file.
2. As a CEM Administrator, go to **Menu -> Workload -> Brokered Services ->** and click **Launch** to launch and log in to the CEM UI.
3. Go to the Administration page and click **Integrations**.
4. Click the **Configuring an integration** button. Select an **Outgoing** integration for Netcool/OMNIbus. Click Configure and use the following steps to configure the outgoing integration.
   a. Provide a name for this outgoing integration. You can use the probe release name for easy reference.
   b. You can skip Step 2. Download and decompress a package of configuration files because this operator has the required configuration and rules files preconfigured.
   c. Enter the probe webhook endpoint obtained from the probe release in the previous step.
   d. Skip Step 4. Enter your credentials because this step is not applicable when integrating with a probe on OCP.
   e. Turn on the integration.
   f. Click Save and verify that the outgoing integration is created successfully.

5. To create an event forwarding policy, go to the Administration page and click Policies, then click Create event policy to create a new event policy and use the following steps to configure the event policy. You may add a forwarding rule in an existing policy if necessary.
   a. Give a name and description for this policy.
   b. Select **All Events** to forward all CEM events to Netcool/OMNIbus. Optionally, you may configure the policy to only forward selected events.
   c. Check the Forwarding events option, click the Add integrations button and then select the outgoing integration created in the previous step.
   d. Enable this event policy.
   e. Click Save and verify that the event policy is successfully installed. It should be listed in the Policies page.
6. It may take a moment before CEM starts to forward events to Netcool/OMNIbus. Verify that CEM events appear in the Netcool/OMNIbus Event List.

Configuring the secure gateway between CEM Cloud and the CEM Probe

Secure gateway is a IBM Cloud service where you can create a secure connection.

You can use the secure gateway to create either a non SSL or an SSL connection between CEM Cloud with a CEM Probe on OCP Cluster.

Using a non SSL connection

To create a non SSL connection between CEM Cloud with a CEM Probe on an OCP cluster, use the following the steps:

1. Create a secure gateway service on IBM Cloud.
2. Click **Add gateway**,

   Leave the parameters unchanged to use the default settings, or change them if required.
3. Click **Connect client** and download the secure gateway client for your probe server's operating system.
4. Install the secure gateway client based using the following instructions: https://cloud.ibm.com/docs/SecureGateway?topic=SecureGateway-client-install
5. After the installation is complete, start up the client on probe server using the following command:

   ```
   cd /opt/ibm/securegateway/client node lib/secgwclient.js -t eg: node lib/secgwclient.js
   <gateway_id> -t <security_token> -l --noUI
   ```
6. When you are interacting with the client session, set ACL to allow traffic into the probe.

   Note: `acl allow` will open all access. For details about Access Control List commands, see https://cloud.ibm.com/docs/SecureGateway?topic=SecureGateway-acl
7. When the client has started up, you should able to see from your IBM Cloud UI that the client is connected.
8. Click on the destination tab and add a destination. Use the following settings:
   a. Select the resource is located **on-premise**.
   b. `hostname: yjcem-cemprobe-yjoperator.apps.sqanetcool43.os.fyre.ibm.com` (kubectl get route in cluster which probe was installed)
   c. Specify `port : 80`
   d. Next window, choose **HTTP** and **None**.
   e. Next window, skip **IP Tables Rules**.
   f. Provide a name for the destination, and **Save**.

9. On the destination tab, click on the settings of the destination and copy the host (proxy) details. for example:

   `http://caplonsgstg-2.securegateway.test.appdomain.cloud:15094/`

   Go to CEM Cloud: outgoing url (example), you need to add the proxy host with the path for example:

   `http://caplonsgstg-2.securegateway.test.appdomain.cloud:15094/probe/webhook/cem`

Using an SSL connection

To create an SSL connection between CEM Cloud with a CEM Probe on an OCP cluster, use the following the steps:

1. Follow Steps 1 to 7 from "Using a non SSL connection" on page 38.

   If you already have a secure gateway set up, you can skip these steps and simply recreate your destination for SSL .

2. Click on the destination tab and add a destination. Use the following settings:

   a. Select the resource is located **on-premise**.

   b. `hostname: yjcem-cemprobe-yjoperator.apps.sqanetcool43.os.fyre.ibm.com` (kubectl get route in cluster which probe was installed)

   c. Specify `port : 443`

   d. Next window, choose **HTTPS ServerSide**.

   - Get the certificate using `openssl` from the cluster where probe was installed.

     ```
     openssl s_client -connect yjcem-cemprobe-
     yjoperator.apps.sqanetcool43.os.fyre.ibm.com:443 -showcerts > cemcert43.pem
     ```

   - Upload this certificate.
   - Server side indicator:

     ```
     yjcem-cemprobe-yjoperator.apps.sqanetcool43.os.fyre.ibm.com (route)
     ```

   e. Next window, skip **IP Tables Rules**.

   f. Provide a name for the destination, and **Save**.

3. At the destination tab, click on the settings of the destination and copy the host (proxy) details. For example:

   `http://caplonsgstg-2.securegateway.test.appdomain.cloud:15094`

   Go to CEM CLoud: outgoing url(example), you need to add the proxy host with the path. Add `https` for example:

   `https://caplonsgstg-2.securegateway.test.appdomain.cloud:15094/probe/webhook/cem`

Deploying the Probe for Prometheus Integration

Note: The NOI Event Integrations Operator must be deployed before deploying this integration. See "Installing the operator" on page 6.

To deploy a new instance of the probe, review the custom resource YAML template supplied with the operator package, update the parameters if required, and save the configurable parameters in a file, for example `example-prometheusprobe-cr.yaml`.

Then deploy the custom resource using the updated file:

```
kubectl apply -f example-prometheusprobe-cr.yaml --namespace <NAMESPACE>
```

The following table describes the configurable parameters for this probe and lists their default values.

Configurable parameters

Parameter name	Description
spec.license.accept	The license state of the image being deployed. Enter `true` to install and use the image. The default value is `false`.
spec.version	Probe application version. Accepted versions are `4.4.0`, `4.5.0` and `4.6.0`. The default value is `4.6.0`.
spec.helmValues.image.repository	Probe image repository. Update this repository name to pull from a private image repository. The default value is `cp.icr.io/cp/noi-int/netcool-probe-messagebus`, and the image name must not be changed.
spec.helmValues.image.tag	The probe image tag. The default value is `13.2.0-amd64`.
spec.helmValues.image.digest	The digest to identify the probe image. If unspecified, the image tag is used. The default value is `sha256:db93b631dde5f724e0234842f56235893f1697e351e3f7e8b1e044905cbdcd07`.
spec.helmValues.image.testRepository	Utility image repository. Update this repository name to pull from a private image repository. The default is `cp.icr.io/cp/noi-int/netcool-integration-util`, and the image name must not be changed.
spec.helmValues.image.testImageDigest	Digest to identify the probe image. If unspecified, the image tag is used. The default value is `sha256:af9b0cf8ade76f2c32f48c54ce6ad8ad9e10a76f8af3940b5a18ff24b419f28c`.
spec.helmValues.image.testImageTag	Utility image tag. The default is `3.4.0-amd64`.
spec.helmValues.image.pullPolicy	The image pull policy. The default value is `Always`.
spec.helmValues.global.image.secretName	The name of the secret containing the docker config to pull the image from a private repository. Leave this parameter blank if the probe image already exists in the local image repository or the Service Account has a been assigned with an Image Pull Secret. The default value is `nil`.
spec.helmValues.global.image.useTag	Configures pods to pull images using tags or digests. Set this property to `false` to use digest to identify the container images. Set this property to `true` to use tags instead of digests to identify the container images. Default value is `false`.

Parameter name	Description
spec.helmValues.global.serviceAccountName	Name of the service account to be used by the pods. If the Cluster Administrator has already created a service account in the namespace, specify the name of the service account here. If left blank, the probe will automatically create a new service account in the namespace when it is deployed. This new service account will be removed from the namespace when the probe is removed. The default is `nil`.
spec.helmValues.netcool.connectionMode	The connection mode to use when connecting to the Netcool/OMNIbus ObjectServer. Refer to Securing probe and ObjectServer communications for more details. **Note:** Refer to the limitations section for more details on available connection modes for your environment. The default is `default`.
spec.helmValues.netcool.primaryServer	The primary Netcool/OMNIbus server to connect to. This is usually set to `NCOMS` or `AGG_P`. The default value is `nil`.
spec.helmValues.netcool.primaryHost	The host of the primary Netcool/OMNIbus server. Specify the ObjectServer hostname or IP address. The default value is `nil`
spec.helmValues.netcool.primaryPort	The port of the primary Netcool/OMNIbus server. The default value is `4100`.
spec.helmValues.netcool.backupServer	The backup Netcool/OMNIbus server to connect to. If the **backupServer**, **backupHost** and **backupPort** parameters are defined in addition to the **primaryServer**, **primaryHost**, and **primaryPort** parameters, the probe will be configured to connect to a virtual object server pair called `` `AGG_V` ``. If no backup ObjectServer is configured, only the primary server parameters will be used. The default value is `nil`.
spec.helmValues.netcool.backupHost	The host of the backup Netcool/OMNIbus server. Specify the ObjectServer hostname or IP address. The default value is `nil`
spec.helmValues.netcool.backupPort	The port of the backup Netcool/OMNIbus server. The default value is `nil`.
spec.helmValues.netcool.secretName	This is a pre-created secret for `AuthOnly`, `SSLOnly` or `SSLAndAuth` connection mode. Certain fields are required depending on the connection modes. The default is `nil`.
spec.helmValues.probe.messageLevel	The probe log message level. The default value is `warn`.

Parameter name	Description
spec.helmValues.probe.setUIDandGID	When set to true, the operator will specify the UID and GID values for the netcool user else the netcool user will not be assigned any UID or GID by the operator. Default value is `true`.
spec.helmValues.probe.sslServerCommonName	Set this parameter to a comma-separated list of acceptable SSL common names for when connecting to the ObjectServer using SSL. This should be set when the `CommonName` field of the received certificate does not match the name specified by the **`primaryServer`** property. When a **`backupServer`** is specified, the probe creates an ObjectServer pair with the name `AGG_V`. Set this parameter if the `CommonName` of the certificate does not match `AGG_V`. The default is `nil`.
spec.helmValues.probe.locale	The container environment locale settings.
spec.helmValues.prometheusProbe.enabled	Set this parameter to `true` to enable the probe for Prometheus. The default value is `true`.
spec.helmValues.prometheusProbe.replicaCount	The number of deployment replicas of the Prometheus Probe. This will be ignored if `prometheusProbe.autoscaling.enabled=true` and will use the **minReplicas** as the **replicaCount**. The default value is `1`.
spec.helmValues.prometheusProbe.ingress.enabled	Set this parameter to `true` to enable Ingress. The default value is `false`.
spec.helmValues.prometheusProbe.ingress.host	Sets the virtual host name for the same IP address. The value must have the Ingress controller default domain as a suffix. Default value is `nil`.
spec.helmValues.prometheusProbe.ingress.tlsEnabled	Enables the TLS Webhook for secure connectivity on the ingress route. Default value is `false`.
spec.helmValues.prometheusProbe.autoscaling.enabled	Set this parameter to `false` to disable auto-scaling. The default value is `true`.
spec.helmValues.prometheusProbe.autoscaling.minReplicas	The minimum number of probe replicas. The default value is `1`.
spec.helmValues.prometheusProbe.autoscaling.maxReplicas	The maximum number of probe replicas. The default value is `3`.
spec.helmValues.prometheusProbe.autoscaling.cpuUtil	The target percentage CPU utilization. For example, enter `60` for `60%` target utilization. The default value is `60`.
spec.helmValues.prometheusProbe.poddisruptionbudget.enabled	Set this parameter to `true` to enable Pod Disruption Budget to maintain high availability during a node maintenance. Administrator role or higher is required to enable Pod Disruption Budget on clusters with Role Based Access Control. The default value is `false`.

Parameter name	Description
spec.helmValues.prometheusProbe.poddisruptionbudget.minAvailable	The minimum number of available pods during node drain. This can be set to a number or a percentage, for example: 1 or 10%. **Caution:** Setting this parameter to 100%, or to the number of replicas, may block node drains entirely. The default value is 1.
spec.helmValues.resources.limits.cpu	The container CPU limit. The default value is `500m`.
spec.helmValues.resources.limits.memory	The container memory limit. The default value is `512Mi`.
spec.helmValues.resources.requests.cpu	The container CPU requested. The default value is `100m`.
spec.helmValues.resources.requests.memory	The container memory requested. The default value is `128Mi`.
spec.helmValues.arch	The worker node architecture. This is set to `amd64`, and cannot be changed.

Post-installation steps for the Prometheus Probe

This section describes the post-installation steps required for the Prometheus Probe

Use the following steps to obtain the probe webhook URL and configure the Prometheus Alertmanager to add the probe webhook as a receiver.

1. The Prometheus Probe webhook can be reached via its service from within the cluster. For example, if the `PrometheusProbe` instance name is `example-prometheusprobe`, then the webhook URL is `http://example-prometheusprobe-ibm-netcool-probe-prometheusprobe.<NAMESPACE>/probe/webhook/prometheus` , where <NAMESPACE> is the namespace where the probe is deployed. If ingress or route is enabled, then the probe webhook is exposed externally via a Route resource and the hostname should be the value set in the custom resource YAML file.
2. Use the steps in the following guide to apply a new configuration:

   Applying custom Alertmanager configuration

   The following sample configuration shows the probe webhook as a receiver:

```
global:
  resolve_timeout: '5m'
receivers:
- name: 'null'
- name: 'netcool_probeclus'
  webhook_configs:
  - url: 'http://yjprome-ibm-netcool-probe-prometheusprobe.yjoperator:80/probe/webhook/
prometheus'
    send_resolved: true

route:
  group_by:
  - alertname
  group_interval: 5m
  group_wait: 30s
  receiver: netcool_probeclus
  repeat_interval: 5s
  routes:
  - receiver: netcool_probeclus
    match:
      alertname: Watchdog
```

Deploying the Probe for Kafka Integration

Creates a Kafka consumer to subscribe messages from a Kafka topic.

Note: The NOI Event Integrations Operator must be deployed before deploying this integration. See "Installing the operator" on page 6.

To deploy a new instance of the probe, review the custom resource YAML template supplied with the operator package, update the parameters if required, and save the configurable parameters in a file, for example `examplekafkaprobe-cr.yaml`.

Then deploy the custom resource using the updated file:

```
kubectl apply -f deploy/crds/probes.integrations.noi.ibm.com_v1_kafkaprobe_cr.yaml --namespace
<NAMESPACE>
```

The following table describes the configurable parameters for this probe and lists their default values.

Configurable parameters

Parameter name	Description
spec.license.accept	Use this parameter to specify the license state of the image being deployed. Enter `true` to install and use the image. The default value is `false`.
spec.version	Use this parameter to specify the probe application version. The accepted version is `3.0.0`. The default value is `3.0.0`.
spec.helmValues.global.image.secretName	Use this parameter to specify the name of the secret containing the Docker Config to pull the image from a private repository. Leave blank if the probe image already exists in the local image repository or the Service Account has a been assigned with an Image Pull Secret. The default value is `nil`
spec.helmValues.global.image.useTag	Use this parameter to configure pods to pull images using tags or digests. Set to false to use digest to identify the container images. Set to true to use tags instead of digests to identify the container images. The default is `false`.
spec.helmValues.global.serviceAccountName	Use this parameter to specify the name of the service account to be used by the helm chart. If the Cluster Administrator has already created a service account in the namespace, specify the name of the service account here. If left blank, the chart will automatically create a new service account in the namespace when it is deployed. This new service account will be removed from the namespace when the chart is removed. The default is `nil`.

Parameter name	Description
spec.helmValues.image.repository	Use this parameter to specify the probe image repository. Update this repository name to pull from a private image repository. The image name should be set to `netcool-probe-messagebus` The default value is `netcool-probe-messagebus`
spec.helmValues.image.tag	Use this parameter to specify the `netcool-probe-messagebus` image tag. The default value is `13.2.0-amd64`
spec.helmValues.image.digest	Use this parameter to specify the digest to identify the probe image. If unspecified, the image tag is used. The default is `sha256:db93b631dde5f724e0234842f56235893f1697e351e3f7e8b1e044905cbdcd07`.
spec.helmValues.image.testRepository	The utility image repository. Update this repository name to pull from a private image repository. The default value is `netcool-integration-util`
spec.helmValues.image.testImageTag	Use this parameter to specify the utility image tag. The default value is `3.4.0-amd64`
spec.helmValues.image.testImageDigest	Use this parameter to specify the digest to identify the utility image. If unspecified, the image tag is used. The default is `sha256:af9b0cf8ade76f2c32f48c54ce6ad8ad9e10a76f8af3940b5a18ff24b419f28c`.
image.pullPolicy	Use this parameter to specify the image pull policy. The default value is `Always`
spec.helmValues.netcool.connectionMode	Use this parameter to specify the connection mode to use when connecting to the Netcool/OMNIbus Object Server. See .Securing Probe and Object Server Communications for more details. **Note :** Refer to the limitations section for more details on available connection modes for your environment. The default is `default`.
spec.helmValues.netcool.primaryServer	Use this parameter to specify the primary Netcool/OMNIbus server that the probe should connect to (required). This is usually set to NCOMS or AGG_P. The default value is `nil`

Parameter name	Description
spec.helmValues.netcool.primaryHost	Use this parameter to specify the host of the primary Netcool/OMNIbus server (required). Specify the ObjectServer hostname or IP address. The default value is `nil`
spec.helmValues.netcool.primaryPort	Use this parameter to specify the port number of the primary Netcool/OMNIbus server (required). The default value is `4100`
spec.helmValues.netcool.backupServer	Use this parameter to specify the backup Netcool/OMNIbus server to connect to. If the **backupServer**, **backupHost** and **backupPort** parameters are defined, the probe will be configured to connect to a virtual object server pair called `AGG_V`. The default value is `nil`
spec.helmValues.netcool.backupHost	Use this parameter to specify the host of the backup Netcool/OMNIbus server. Specify the ObjectServer hostname or IP address. The default value is `nil`
spec.helmValues.netcool.backupPort	Use this parameter to specify the port of the backup Netcool/OMNIbus server. The default value is `0`
spec.helmValues.probe.messageLevel	Use this parameter to specify the probe log message level. The default value is `warn`
spec.helmValues.probe.setUIDandGID	If true, the helm chart specifies the UID and GID values to be assigned to the `netcool` user in the container. Otherwise, when `false` the `netcool` user will not be assigned any UID or GID by the helm chart. Refer to the deployed PSP or SCC in the namespace. The default is `true`
spec.helmValues.probe.heartbeatInterval	Use this parameter to specify the frequency (in seconds) with which the probe checks the status of the host server. The default value is `10`
spec.helmValues.probe.secretName	Use this parameter to specify the Secret for authentication credentials for the HTTP and Kafka Transport. The default value is `nil`

Parameter name	Description
spec.helmValues.probe.rulesConfigmap	Use this parameter to specify a customized ConfigMap to be used for the rules files. This field may be left blank if not required. The default value is `nil`
spec.helmValues.probe.jsonParserConfig .messagePayload	Use this parameter to specify the JSON tree to be identified as message payload from the notification (kafka consumer) channel. See example for more details on how to configure the Probe's JSON parser. The default value is `json`
spec.helmValues.probe.jsonParserConfig . messageHeader	Use this parameter to specify a JSON tree to be identified as message header from the notification (kafka consumer) channel. Attributes from the headers will be added to the generated event. The default value is `nil`
spec.helmValues.probe.jsonParserConfig . jsonNestedPayload	Use this parameter to specify a JSON tree within a nested JSON or JSON string to be identified as message payload from the notification (kafka consumer) channel. **messagebus.probe.jsonParserConfig.messagePayload** must be set to point to the attribute containing the JSON String. The default value is `nil`
spec.helmValues.probe.jsonParserConfig . jsonNestedHeader	Use this parameter to specify a JSON tree within a nested JSON or JSON string to be identified as message header from the notification (kafka consumer) channel. **messagebus.probe.jsonParserConfig.messageHeader** must be set to point to the attribute containing the JSON String. The default value is `nil`
spec.helmValues.probe.jsonParserConfig .messageDepth	Use this parameter to specify the number of JSON child node levels in the message to traverse during parsing. The default value is 3
spec.helmValues.probe.kafka.connection . zookeeperClient.target	Use this parameter to specify the ZooKeeper endpoint. The default value is `nil`
spec.helmValues.probe.kafka.connection . zookeeperClient.topicWatch	Use this parameter to enable the ZooKeeper topic watch service. The default value is `false`

Parameter name	Description
spec.helmValues.probe.kafka.connection. zookeeperClient.brokerWatch	Use this parameter to enable the ZooKeeper broker watch service. The default value is `false`
spec.helmValues.probe.kafka.connection .brokers	Use this parameter to specify the broker endpoints in a comma-separated list, for example: `PLAINTEXT:// kafkaserver1:9092,PLAINTEXT:// kafkaserver2:9092` The default value is `nil`
spec.helmValues.probe.kafka.connection .topics	Use this parameter to specify the topics in a comma-separated list, for example `topicABC,topicXYZ`. The default value is `nil`
spec.helmValues.probe.kafka.client.sec urityProtocol	Use this parameter to specify the security protocol. The default value is `nil`
spec.helmValues.probe.kafka.client.ssl . trustStoreSecretName	Use this parameter to specify the truststore Secret name and its password. The default value is `nil`
spec.helmValues.probe.kafka.client.ssl . keyStoreSecretName	Use this parameter to specify the keystore Secret name and its password. The default value is `nil`
spec.helmValues.probe.kafka.client. saslPlainMechanism	Use this parameter to enable the PLAIN mechanism for Simple Authentication and Security Layer connections. The default value is `false`
spec.helmValues.probe.kafka.client.con sumer.groupId	Use this parameter to specify the group identifier. The default value is `nil`
spec.helmValues.probe.kafka.client.con sumer.configs	Use this parameter to specify a multi-line string parameter to configure additional Kafka Consumer Configurations. Each entry must not contain space. The default is `key.deserializer=org.apache.kafka.comm on.serialization.StringDeserializer value.deserializer=org.apache.kafka.co mmon.serialization.StringDeserializer`.
spec.helmValues.probe.replicaCount	Use this parameter to specify the number of deployment replicas. Omitted when **autoscaling.enabled** set to `true`. The default is 1.

Parameter name	Description
spec.helmValues.probe.autoscaling.enabled	Use this parameter to enable or disable auto-scaling. The default value is `true`
spec.helmValues.probe.autoscaling.minReplicas	Use this parameter to specify the minimum number of probe replicas. The default value is `1`
spec.helmValues.probe.autoscaling.maxReplicas	Use this parameter to specify the maximum number of probe replicas. The default value is `5`
spec.helmValues.probe.autoscaling.cpuUtil	Use this parameter to specify the target CPU utilization. For example, enter `60` for `60%` target utilization. The default value is `60`
spec.helmValues.probe.poddisruptionbudget.enabled	Use this parameter to enable or disable Pod Disruption Budget to maintain high availability during a node maintenance. The default value is `false`
spec.helmValues.probe.poddisruptionbudget.minAvailable	Use this parameter to specify the number of minimum available number of pods during node drain. Can be set to a number or percentage, for example, `1` or `10%`. **CAUTION :** Setting to `100%` or equal to the number of replicas may block node drains entirely. The default value is `1`
spec.helmValues.resources.limits.cpu	Use this parameter to specify the CPU resource limits. The default value is `500m`
spec.helmValues.resources.limits.memory	Use this parameter to specify the memory resource limits. The default value is `512Mi`
spec.helmValues.resources.requests.cpu	Use this parameter to specify the CPU resource requests. The default value is `250m`
spec.helmValues.resources.requests.memory	Use this parameter to specify the memory resource requests. The default value is `256Mi`
spec.helmValues.arch	Use this parameter to specify the worker node architecture. This is fixed to `amd64`.

Integrating with IBM Event Streams for IBM Cloud

The following sections describe how to integrate the Probe for Kafka with IBM Event Streams for IBM Cloud:

IBM Event Streams for IBM Cloud Setup

1. To start creating IBM Event Streams for IBM Cloud, see https://cloud.ibm.com/docs/EventStreams?topic=EventStreams-getting_started .

   Note : IBM Event Streams for IBM Cloud only supports SASL_SSL. It does not support PLAINTEXT.

2. When you download, the sample producer/consumer should be the standard configuration as shown below. Check for any changes in the configuration files downloaded from the producer/consumer. The details of the broker, username and token can be retrieved from the UI:

   ```
   topic : kafka-java-console-sample-topic
   consumer.groupid : test-consumer-group
   ```

3. The secret shown below is created for the connection of the probe to the IBM Event Streams for IBM Cloud. The details of the username/token were retrieved in Step 2.

   ```
   kubectl create secret generic -n noi-integrations cloudes --from-
   literal=http_username='token' --from-literal=http_password='apikey' --from-
   literal=kafka_username='token' --from-literal=kafka_password='apikey'
   ```

4. When you start the sample producer, the producer will start sending a sample event. In order to change the events so that probe will receive events that contain more information, make the following change:

   a. Edit `event-streams-samples/kafka-java-console-sample/src/main/java/com/eventstreams/samples/ProducerRunnable.java`

   b. Add the following string below at `String Message`, comment the existing `String Message` and save the file.

   ```
   String message = "{\"response\":{\"Manager\":\"Kafka\",\"Node\":\"Kafka SASL_SSL Test
   \",\"Agent\":\"cloudkafka\",\"AlertGroup\":\"pod status\",\"AlertKey\":\"111\",\"Severity
   \":\"3\",\"Summary\":\"for on perm testing actual dove 4\",\"status\":0,\"startRow
   \":0,\"endRow\":0,\"totalRows\":1,\"data\":1535118829745,\"errors\":null}}";
   ```

   c. Run `gradle clean && gradle build` again for the changes to take effect.

   d. Start the producer and the consumer should start receiving the new events.

Installing the Kafka instance using OLM UI

Note : For the steps in this section, the sample yaml file is for the connection with IBM Event Streams for IBM Cloud.

1. Install Catalog:

   a. Go to **OCP Console**.

   b. Go to **Administration > Cluster Settings > Global Configuration > OperatorHub > Sources > Create Catalog Source**.

   A sample Catalog Source resource is shown below. If you would like to specify the image using its digest, the digest can be retrieved from the `images.csv` file from CASE bundle.

Once Catalog is installed successfully, the number of Operators will be changed to 1.

```
apiVersion: operators.coreos.com/v1alpha1
kind: CatalogSource
metadata:
  name: ibm-netcool-integrations-operator-catalog
  namespace: REPLACE_NAMESPACE
spec:
  displayName: IBM Netcool Operations Insight Event Integrations Catalog
  publisher: IBM
  sourceType: grpc
  image: docker.io/ibmcom/ibm-netcool-integrations-operator-catalog:latest
  updateStrategy:
    registryPoll:
      interval: 45m
```

2. Install Operator:

   a. Go to **Operators > OperatorHub**. Search for `IBM Netcool Operations Insight Event Integrations Operator`. Click **Install**. Choose the namespace. Choose **Auto** or **Manual**.

   b. Go to **Operators > Installed Operators**. If the installation of operator is successful, the **Status** will be shown as `Succeeded`. Alternatively you can check using command line on the status of the pods:

   ```
   oc get pods -n <namespace>
   ```

   If Operator is not starting up, check the logs and solve the issue:

   ```
   oc logs po/<podname> -n namespace
   ```

3. Install the probe instance:

   a. Go to Operators > Installed Operator. Click on "IBM Netcool Operations Insight Event Integrations Operator", Create Instance at KafkaProbe.

   b. Update the yaml files properties as below:

   ```
   probe.kafka.client.saslPlainMechanism: true
   probe.kafka.client.securityProtocol: SASL_SSL
   messagePayload: json.response
   ```

   For these three properties, information can be retrieved from IBM Event Streams for IBM Cloud sample producers/consumers downloaded.

   ```
   probe.kafka.client.consumer.groupid
   probe.kafka.connection.brokers
   probe.kafka.connection.topic
   ```

   Retrieve the `username` and `apikey` from IBM Event Streams for IBM Cloud following the procedures in https://cloud.ibm.com/docs/EventStreams.

   Then create a secret for the probe, for example:

   ```
       kubectl create secret generic cloudes \
       --namespace noi-integrations \
       --from-literal=http_username='<username>' \
       --from-literal=http_password='<apikey>'  \
       --from-literal=kafka_username='<username>' \
       --from-literal=kafka_password='<apikey>'
   ```

   Where <username> and <apikey> are the username and API token retrieved from IBM Event Streams for IBM Cloud.

   ```
       probe:
         secretName: cloudes
   ```

Below is the sample yaml file:

```
apiVersion: probes.integrations.noi.ibm.com/v1
kind: KafkaProbe
metadata:
 name: example-kafkaprobe
 labels:
   app.kubernetes.io/instance: example-kafkaprobe
   app.kubernetes.io/managed-by: netcool-integrations-operator
   app.kubernetes.io/name: kafkaprobe
 namespace: yjoperator
spec:
 helmValues:
   arch: amd64
   global:
     image:
       secretName: ''
       useTag: false
     serviceAccountName: ''
   image:
     digest: 'sha256:db93b631dde5f724e0234842f56235893f1697e351e3f7e8b1e044905cbdcd07'
     pullPolicy: Always
     repository: cp.icr.io/cp/noi-int/netcool-probe-messagebus
     tag: 13.2.0-amd64
     testImageDigest:
'sha256:af9b0cf8ade76f2c32f48c54ce6ad8ad9e10a76f8af3940b5a18ff24b419f28c'
     testImageTag: 3.4.0-amd64
     testRepository: cp.icr.io/cp/noi-int/netcool-integration-util
   netcool:
     backupHost: ''
     backupPort: 0
     backupServer: ''
     connectionMode: SSLAndAuth
     primaryHost: '10.17.15.114'
     primaryPort: 32732
     primaryServer: 'AGG_P'
     secretName: 'noi254secret'
   probe:
     heartbeatInterval: 10
     rulesConfigmap: ''
     sslServerCommonName: 'noi254-proxy.pink.svc'
     kafka:
       client:
         consumer:
           groupId: 'test-consumer-group'
         saslPlainMechanism: true
         securityProtocol: 'SASL_SSL'
         ssl:
           keyStoreSecretName: ''
           trustStoreSecretName: ''
       connection:
         brokers: 'broker-5-77s66rz443glj9c8.kafka.svc04.us-
south.eventstreams.cloud.ibm.com:9093,broker-3-77s66rz443glj9c8.kafka.svc04.us-
south.eventstreams.cloud.ibm.com:9093,broker-1-77s66rz443glj9c8.kafka.svc04.us-
south.eventstreams.cloud.ibm.com:9093,broker-0-77s66rz443glj9c8.kafka.svc04.us-
south.eventstreams.cloud.ibm.com:9093,broker-4-77s66rz443glj9c8.kafka.svc04.us-
south.eventstreams.cloud.ibm.com:9093,broker-2-77s66rz443glj9c8.kafka.svc04.us-
south.eventstreams.cloud.ibm.com:9093'
         topics: 'kafka-java-console-sample-topic'
         zookeeperClient:
           brokerWatch: false
           target: ''
           topicWatch: false
     jsonParserConfig:
       jsonNestedHeader: ''
       jsonNestedPayload: ''
       messageDepth: 3
       messageHeader: ''
       messagePayload: json
     replicaCount: 1
     autoscaling:
       cpuUtil: 60
       enabled: true
       maxReplicas: 5
       minReplicas: 1
     setUIDandGID: true
     locale: en_US.utf8
     secretName: 'cloudes'
     poddisruptionbudget:
       enabled: false
       minAvailable: 1
```

```
      messageLevel: debug
    resources:
      limits:
        cpu: 500m
        memory: 512Mi
      requests:
        cpu: 250m
        memory: 256Mi
  license:
    accept: true
  version: 3.0.0
```

Note : Do not replace this yaml file with your configuration. Just take the values for the properties needed.

4. Install the probe. Check `oc get pods` from the command line to see if all pods are up and running.
5. To uninstall the probe:
   a. Go to **Operators > Installed Operator > All Instances**.
   b. Choose the instance to uninstall.

Installing the Kafka instance using cloudctl case commands

1. Unzip case bundle downloaded. These files can be seen:

```
    ibm-netcool-integrations-1.2.0-images.csv
    ibm-netcool-integrations-1.2.0.tgz
    ibm-noi-int-v1.2.0.tar
```

`ibm-netcool-integrations-1.2.0.tgz` will be used in the `--case` parameter during installation and uninstallation of probes.

2. Unzip `ibm-netcool-integrations-1.2.0.tgz`, unzipped files here will be used at the `--args` parameter during installation and uninstallation of probes.
3. Install the probe using the **`cloudctl`** command:

```
cloudctl case launch  \
  --case case/ibm-netcool-integrations-*.tgz    \
  --namespace $NAMESPACE     \
  --instance <instance name> \
  --inventory netcoolIntegrationsOperator   \
  --action apply-custom-resources  \
  --args "--inputDir $PWD/case --crFile $PWD/case/ibm-netcool-integrations/inventory/
netcoolIntegrationsOperator/files/deploy/crs/$CRFILE.yaml" \
  --tolerance 1
```

4. Check `oc get pods` from the command line to see if all pods are up and running.
5. To uninstall the probe, use the following command:

```
cloudctl case launch  \
  --case case/ibm-netcool-integrations-*.tgz    \
  --namespace $NAMESPACE     \
  --instance <instance name> \
  --inventory netcoolIntegrationsOperator   \
  --action delete-custom-resources  \
  --args "--inputDir $PWD/case --crFile $PWD/case/ibm-netcool-integrations/inventory/
netcoolIntegrationsOperator/files/deploy/crs/$CRFILE.yaml" \
  --tolerance 1
```

Testing with IBM Event Streams for IBM Cloud

1. Start the producer using the following command (your command might vary as this is just an example). Make sure you have changed the sample event.

```
java -jar ./build/libs/kafka-java-console-sample-2.0.jar
"broker-5-77s66rz443glj9c8.kafka.svc04.us-
south.eventstreams.cloud.ibm.com:9093,broker-3-77s66rz443glj9c8.kafka.svc04.us-
```

```
south.eventstreams.cloud.ibm.com:9093,broker-1-77s66rz443glj9c8.kafka.svc04.us-
south.eventstreams.cloud.ibm.com:9093,broker-0-77s66rz443glj9c8.kafka.svc04.us-
south.eventstreams.cloud.ibm.com:9093,broker-4-77s66rz443glj9c8.kafka.svc04.us-
south.eventstreams.cloud.ibm.com:9093,broker-2-77s66rz443glj9c8.kafka.svc04.us-
south.eventstreams.cloud.ibm.com:9093" qPh7UVCSAWJrgr5wmiKjj-knQ_tx2fEiS0b0Z4wK7sVp -producer
```

2. Your probe instance should now be up and running.
   a. Check `oc logs po/podname -f` to see the logs.
   b. Verify that your probe has started receiving events from IBM Event Streams for IBM Cloud.
   c. Verify that events sent to NOI Event Lists properly.
3. Check for any errors in the probe logs.

Customizing probe rules in ConfigMap

This section shows how to modify the default rules file in the probe `ConfigMap` using the command line.

Note: The `ConfigMap` will be deleted when the probe instance is deleted. You should keep a copy of any custom rules file in case you need to use them in the future.

- "Getting the ConfigMap name and rules filename" on page 54
- "Extracting and Customizing the rules file" on page 55
- "Patching the Configmap" on page 55
- "Restarting the Probe Pods" on page 55

Getting the ConfigMap name and rules filename

You need to determine the probe instance name and get the `ConfigMap` name that contains the probe rules files. The `ConfigMap` with the `-rules` suffix should contain the rules files.

The following command uses `example-kafkaprobe` as the probe instance name to query the `ConfigMap` using the `app.kubernetes.io/instance` label. The `--namespace` option can also be specified if required:

```
$ kubectl get configmap -l app.kubernetes.io/instance=example-kafkaprobe
NAME                                    DATA   AGE
example-kafkaprobe-probe-mb-kfk-config   8      12d
example-kafkaprobe-probe-mb-kfk-rules    5      12d
```

Note: The rules file(s) name from the `Data` field. You need to extract the files that you want to customize. This example customizes the `message_bus.rules`.

```
$ kubectl describe configmap example-kafkaprobe-probe-mb-kfk-rules
Name:         example-kafkaprobe-probe-mb-kfk-rules
Namespace:    default
Labels:       app.kubernetes.io/component=mb
              app.kubernetes.io/instance=example-kafkaprobe
              app.kubernetes.io/managed-by=Tiller
              app.kubernetes.io/name=probe-mb-kfk
              hdm.ibm.com/chartversion=3.0.0
              hdm.ibm.com/lastreconciled=
              hdm.ibm.com/resourceowner=
              helm.sh/chart=ibm-netcool-probe-messagebus-kafka-prod
              release=example-kafkaprobe
Annotations:  hdm.ibm.com/lastknownstate: 67c4b74cd6d3b5c44718b38e3cd4517b

Data
====
message_bus.rules:
----
<message_bus.rules file content omitted>

message_bus_cbe.rules:
----
<message_bus_cbe.rules file content omitted>

message_bus_netcool.rules:
```

```
----
<message_bus_netcool.rules content omitted>

message_bus_wbe.rules:
----
<message_bus_wbe.rules content omitted>

message_bus_wef.rules:
----
<message_bus_wef.rules content omitted>

Events:  <none>
```

Extracting and Customizing the rules file

To customize the `message_bus.rules` file, first extract the file into you local host:

```
kubectl get configmap example-kafkaprobe-probe-mb-kfk-rules -o jsonpath='{.data.message_bus
\.rules}' > message_bus.rules
```

Then edit the file to make any customization. Ensure that the rules file's syntax is correct.

Patching the Configmap

To patch the `ConfigMap`, first create a YAML file of the "new" `ConfigMap` using the command below. Note that the name of the `ConfigMap` must be the same as the original `ConfigMap`. The output of this command is saved into the file: `example-kafkaprobe-probe-mb-kfk-rules-custom.yaml`.

```
kubectl create configmap --dry-run -o yaml \
--from-file=message_bus.rules \
example-kafkaprobe-probe-mb-kfk-rules.yaml > example-kafkaprobe-probe-mb-kfk-rules-custom.yaml
```

Use the `kubectl patch` command to patch the existing `ConfigMap` with the new YAML file (`example-kafkaprobe-probe-mb-kfk-rules-custom.yaml`)

```
kubectl patch configmap example-kafkaprobe-probe-mb-kfk-rules \
--patch "$(cat example-kafkaprobe-probe-mb-kfk-rules-custom.yaml)"
```

Verify that the `ConfigMap` is correctly patched.

```
kubectl describe configmap example-kafkaprobe-probe-mb-kfk-rules-custom
```

Restarting the Probe Pods

Kubernetes pods do not detect any changes to the `ConfigMap` and the probe pods must be restarted to reload the updated `ConfigMap`.

To restart the pod, scale down and scale up the deployment using the `kubectl scale deployment` command as shown below:

```
kubectl scale deployment example-kafkaprobe-probe-mb-kfk --replicas=0
kubectl scale deployment example-kafkaprobe-probe-mb-kfk --replicas=1
```

Verify that the container is restarting and running:

```
$ kubectl get pods -l app.kubernetes.io/instance=example-kafkaprobe
NAME                                              READY   STATUS                RESTARTS   AGE
example-kafkaprobe-probe-mb-kfk-8684b6b947-jk76q   1/1     Running               0          50s
```

Appendix A. Notices and Trademarks

This appendix contains the following sections:

- Notices
- Trademarks

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing 2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator, Department 49XA

3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, ibm.com, AIX, Tivoli, zSeries, and Netcool are trademarks of International Business Machines Corporation in the United States, other countries, or both.

Adobe, Acrobat, Portable Document Format (PDF), PostScript, and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Intel, Intel Inside (logos), MMX, and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

SC28-3144-02