Federator.ai for OpenShift
Installation and Configuration Guide
Content

Deploy Federator.ai for OpenShift 4.x ................................................................. 2
  Requirements .................................................................................................................. 2
  Installation ...................................................................................................................... 2
Configure Federator.ai to Manage Application Pods .................................................. 8
  Configuration .................................................................................................................. 8
Manage Federator.ai License Keycode ......................................................................... 11
  Apply Keycode .............................................................................................................. 11
  Delete Keycode ............................................................................................................ 11
  Activate Keycode .......................................................................................................... 12
Deploy Federator.ai for OpenShift 4.x

Requirements

- **Platform**: OpenShift 4.x
- **User role**: Cluster admin

Installation

1. Log in OpenShift administration console as cluster admin

2. Go to “Projects” page and create a new project, for example, “federatorai” for Federator.ai
3. Click “Operators -> OperatorHub” on the main menu, Federator.ai is certified by OperatorHub, by searching keyword “federator”, Federator.ai Operator is listed for installation.

4. There are 2 versions of Federator.ai, Community and Official, they provide identical full functions. Confirm the left-top “Project” selector indicates the current working project is “federatorai”, then click the official version and install it.
5. After starting Federator.ai installation, OpenShift prompts users Operator Subscription options. It’s recommended to keep the default options.

6. It will take a few minutes to pull images and install Federator.ai in the “federatorai” project. Once the installation completes, Federator.ai will show on “Installed Operators” page.

7. Click “Federator.ai” to see the details of Federator.ai Operator. After Federator.ai Operator is installed, configure Federator.ai Operator to install the rest of Federator.ai components by creating an “AlamedaService” resource. You can configure with your persistent volumes or leave the default ephemeral storage.
8. Click “my-alamedaservice” resource to see the details, “Resources” view shows the status of the rest of Federator.ai components. When the status of all components are “Running”, Federator.ai installation is complete successfully.
9. The URL of Federator.ai GUI can be found at “Projects -> federatorai -> Workloads -> alameda-grafana -> Routes”

10. Log in Federator.ai GUI with the default account/password, “admin/admin”. The portal page of Federator.ai GUI shows the summary of Federator.ai and OpenShift cluster information.
11. Click top-left “Dashboard” menu will show the full list of Federator.ai dashboards
Configure Federator.ai to Manage Application Pods

To monitor and manage application pods, Federator.ai defines an “AlamedaScaler” CRD for users to specify applications to be managed. By creating an “AlamedaScaler” CR and configuring with the application label and namespace, Federator.ai will discover and start managing the pods of the application in the namespace.

In this example, we create an “AlamedaScaler” CR for Federator.ai to manage a NGINX+Node.js application, “demo-nginx-npm”, running in the namespace, “demo-nginx-npm”.

Currently Federator.ai supports the application which is a “Deployment”, “DeploymentConfig” or “Statefulset”.

Configuration

1. Get the label of the NGINX+Node.js application

There are two ways to get the label of an application

   1. By ‘oc’ command line command:

```
   $ oc get deploy -n demo-nginx-npm --show-labels
```

   (In this example, the application “NGINX+Node.js” is a Deployment. If your application is a DeploymentConfig, then the ‘oc’ command should change to:
   ‘oc get dc -n demo-nginx-npm --show-labels’

   where ‘dc’ denotes ‘DeploymentConfig’.)

   2. By OpenShift GUI:

2. Create “AlamedaScaler” CR and configure the application label and namespace

2.1. Edit the sample “alamedascaler.yaml” with the application label and namespace.
“alamedascaler.yaml” should be downloaded by “install.sh” and saved in “/tmp/install-op/” directory. It can also be downloaded at the URL:
https://github.com/containers-ai/alameda/raw/v4.2.301/example/samples/nginx/alamedascaler.yaml

$ vi /tmp/install-op/alamedascaler.yaml

“alamedascaler.yaml”:

```yaml
apiVersion: autoscaling.containers.ai/v1alpha1
kind: AlamedaScaler
metadata:
  name: npm
  namespace: demo-nginx-npm
spec:
  policy: stable
  enableExecution: false
  scalingTool:
    type: hpa
    selector:
      matchLabels:
        app: demo-nginx-npm
```

2.2. Apply alamedascaler.yaml to create the “AlamedaScaler” CR

$ oc apply -f /tmp/install-op/alamedascaler.yaml

Read the “AlamedaScaler” CR by ‘oc get’ command to confirm Federator.ai is successfully managing the “demo-nginx-npm” application.

$ oc get alamedascaler -n demo-nginx-npm -o yaml
3. Log on Federator.ai GUI to get the detailed information of the application

```bash
[root@h7-130 ~]# oc get alamedascaler -n demo-nginx-npm -o yaml
apiVersion: v1
items:
- apiVersion: autoscaling.containers.ai/v1alpha1
deployment
metadata:
  annotations:
status:
  alamedaController:
    deployments:
      demo-nginx-npm/demo-nginx-npm:
        name: demo-nginx-npm
        namespace: demo-nginx-npm
        pods:
          demo-nginx-npm/demo-nginx-npm-865bd55b87-dmkkx:
            containers:
              - name: demo-nginx-npm
                resources: {}
              name: demo-nginx-npm-865bd55b87-dmkkx
            namespace: demo-nginx-npm
            uid: 4c8643a4-e692-11e9-8e7c-005056bc2ae3
            demo-nginx-npm/demo-nginx-npm-865bd55b87-nh4bl:
              containers:
                - name: demo-nginx-npm
                  resources: {}
                name: demo-nginx-npm-865bd55b87-nh4bl
              namespace: demo-nginx-npm
              uid: 4c7f91a7-e692-11e9-8e7c-005056bc2ae3
              specReplicas: 2
              uid: 4c7ac9c1-e692-11e9-8e7c-005056bc2ae3
kind: List
metadata:
  resourceVersion: ""
  selfLink: ""
```
Manage Federator.ai License Keycode

Federator.ai uses a keycode to control the license. A 30-day trial keycode is installed by default. It requires replacing with a valid keycode from ProphetStor to continue using Federator.ai after the 30-day trial. The keycode operations are done by editing the “AlamedaService” CR which is created during Federator.ai installation.

Apply Keycode

1. Get “AlamedaService” CR name

   $ oc get alamedaservice --all-namespaces

   

   ![ocl output]

2. Edit the “AlamedaService” CR

   $oc edit alamedaservice my-alamedaservice

3. Go to “keycode:” section, replace the value of “codeNumber” with the new keycode and then save the change

   ![keycode change]

Delete Keycode

1. Get “AlamedaService” CR name

   $ oc get alamedaservice --all-namespaces

   

   ![ocl output]

2. Edit the “AlamedaService” CR

   $oc edit alamedaservice my-alamedaservice

3. Go to “keycode:” section, delete the keycode from “codeNumber” and then save the change

   ![keycode change]
Activate Keycode

1. Get “AlamedaService” CR name

```bash
$ oc get alamedaservice --all-namespaces
```

```
# oc get alamedaservice --all-namespaces
NAMESPACE NAME AGE
federatorai my-alamedaservice 7d
```

2. Edit the “AlamedaService” CR

```bash
$oc edit alamedaservice my-alamedaservice
```

3. Go to “status.keycodeStatus:” section, copy the value of “registrationData” and email to register@prophetstor.com

![Registration Data Image]

4. Once ProphetStor received the activation request email and validated the “registrationData”, it returns the activation code, “signatureData”, via an email. Copy the “signatureData” from the email, fill in the “keycode. signatureData” field and save the change.

![Signature Data Image]