Federator.ai 4.3 for OpenShift
Installation Guide
(for Red Hat Marketplace)
Overview

Federator.ai

ProphetStor Federator.ai is an AI-based solution that helps enterprise manage, optimize, auto-scale resources for any applications on OpenShift. Using advanced machine learning algorithms to predict application workload, Federator.ai scales the right amount of resources at the right time for optimized application performance.

- AI-based workload prediction for Kafka or other applications
- Resource recommendation based on workload prediction, application, OpenShift and other related metrics
- Automatic scaling of application containers through Datadog Watermark Pod Autoscaler (WPA)

Datadog Integration Workflows

The following diagram shows how applications metrics are used by Federator.ai to predict workload and to automatically scale applications for better performance. Specifically,

- Datadog Agent sends cluster/applications metrics to Datadog Services
- Federator.ai’s Data-adapter queries cluster/applications metrics from Datadog Services and forwards them to Federator.ai AI engine
- Data-adapter posts the prediction/recommendation/plan created by Federator.ai to Datadog Services
- Datadog Cluster Agent gets prediction/recommendation/plan from Datadog Services
- WPA applies plans and auto-scales applications
- Datadog Dashboard displays cluster/applications metrics and prediction/recommendation/plan by Federator.ai
Requirements and Recommended Resource Configuration

Platform
- OpenShift 4.4 and above

Federator.ai Resource Requirements
- Total Resource Requirements
  - 4 CPU cores
  - 4 GB Memory
  - StorageClass: 420GB (require ReadWriteMany access mode)
- Resource requirements for AI Engine
  - There must be at least one worker node with at least 2 CPU cores and 1 GB memory available
  - The 2 CPU cores and 1 GB memory are included in the total 4 CPU cores and 4 GB memory requirements

Federator.ai Version
- Version: Release 4.3
- 30 days trial license

Datadog Agent Version (reference)
- Datadog Agent version: v7.21.1
- Datadog Cluster Agent version: v1.7.0
- Datadog Watermark Pod Autoscaler version: v0.1.0
- kube-state-metrics: v1.5.0 (for OpenShift 3.11, Kubernetes 1.11 ~ 1.12)
  v1.9.6 (for OpenShift 4.3/4.4/4.5, Kubernetes 1.13 ~ 1.18.x)

Persistent Volumes
- The StorageClass that provides the persistent volumes must support RWX (read-write many) access mode.
- It is recommended to use persistent volumes instead of using ephemeral storage to store the data in the production environment.

Kafka
- For Federator.ai’s application-aware Kafka consumer resource/performance optimization feature, the following version of Kafka is supported:
  Kafka operator version: Strimzi/kafka:0.17.0-kafka-2.4.0
Federator.ai Installation and Configuration

Summary of Installation Steps

Step 0: Review pre-installation checklist items, make sure the environment and required information are ready.

Step 1: Collect information on Datadog Cloud Service account, API Key, Application Key. Instructions are provided below.

Step 2: Install and configure Datadog Agent/Cluster Agent if they have not been installed. Please follow Datadog documentation on how to install Datadog Agent and Cluster Agent.

Step 3: Install Federator.ai.

Step 4: Configure Federator.ai Data Adapter for Datadog.

Step 5: Optionally install Datadog WPA and apply WPA autoscaling CR if using Datadog WPA for autoscaling.

Step 6: Review installation result on Datadog Cloud Dashboard.

Pre-installation Check List

OpenShift:

<table>
<thead>
<tr>
<th>#</th>
<th>Check list Item</th>
<th>Requirement</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>What is the OpenShift version?</td>
<td>4.4 and above</td>
<td>Use the command below to get OpenShift version: <code>~# oc version Client Version: 4.4.5 Server Version: 4.4.5 Kubernetes Version: v1.17.1</code></td>
</tr>
<tr>
<td>2</td>
<td>StorageClass and Persistent Volumes requirement</td>
<td>StorageClass supports ReadWriteMany access mode. Available storage size is larger than 430GB. Minimum storage size for Federator.ai Release 4.3 is 430GB, including database, data, and logs.</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>OpenShift cluster CPU/memory requirement</td>
<td>Minimum CPU/mem/storage: - CPU: 4 Cores - Memory: 4 GB - Storage Class Capacity: 430GB At least one worker node with - CPU: 2 Cores - Memory: 1GB To be able to run AI Engine pod, there must be at least one worker node that has more than 2 CPU cores and 1 GB memory available. 2 CPU Cores and 1GB for AI Engine are included in the total 4 CPU Cores and 4GB memory requirements.</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Will there be a resource quota imposed for the namespace where Federator.ai is installed?</td>
<td>CPU/mem request quota should be more than minimum resource requirement - CPU: 4 Cores - Memory: 4 GB The CPU/memory required for Federator.ai depends on the number of clusters and applications being monitored/managed. Suggestion for initial namespace quota is - CPU 8 cores - Memory 12G The quota could be adjusted if number of managed clusters/applications increases. Use the command to get namespace resource quota <code>~# oc get resourcequota --all-namespaces</code></td>
<td></td>
</tr>
</tbody>
</table>

Datadog Agent:
<table>
<thead>
<tr>
<th>#</th>
<th>Check list Item</th>
<th>Requirement</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Is Datadog Agent installed?</td>
<td>Datadog Agent is mandatory</td>
<td>OpenShift resources and workload metrics are collected by Datadog Agent.</td>
</tr>
<tr>
<td>6</td>
<td>Is Datadog Cluster Agent installed?</td>
<td>Cluster Agent is mandatory for HPA autoscaling feature</td>
<td>Cluster Agent provides metrics to HPA Autoscaler for autoscaling.</td>
</tr>
<tr>
<td>7</td>
<td>Is Datadog WPA controller installed?</td>
<td>Datadog WPA is required if autoscaling is done by WPA</td>
<td>Datadog WPA is the HPA Autoscaler developed by Datadog. Users can use Datadog WPA or Kubernetes native HPA to do autoscaling.</td>
</tr>
<tr>
<td>8</td>
<td>Datadog Kafka Consumer integration is enabled?</td>
<td>Datadog Kafka Consumer integration is mandatory if user wants to use Kafka optimization feature</td>
<td>Use the command to confirm Kafka integration is enabled ~# oc exec &lt;datadog-agent-pod&gt; -n &lt;datadog-agent-namespace&gt; -- agent integration show datadog-kafka-consumer Refer to <a href="https://www.datadoghq.com/blog/monitor-kafka-with-datadog/">https://www.datadoghq.com/blog/monitor-kafka-with-datadog/</a> for Kafka Consumer integration installation</td>
</tr>
<tr>
<td>9</td>
<td>Datadog account API key</td>
<td>API key is mandatory for connecting Datadog Service</td>
<td>Follow the steps described in the “Before You Start” session to obtain the API key.</td>
</tr>
<tr>
<td>10</td>
<td>Datadog account Application key</td>
<td>Application key is mandatory for connecting Datadog Service</td>
<td>Follow the steps described in the “Before You Start” session to obtain the Application key.</td>
</tr>
<tr>
<td>11</td>
<td>DD_TAGS with value=&quot;kube_cluster:&lt;cluster_name&gt;&quot; is configured for Datadog Agent?</td>
<td>DD_TAGS with value=&quot;kube_cluster:&lt;cluster_name&gt;&quot; is required for Federator.ai to identify OpenShift cluster</td>
<td>Use the command below to confirm DD_TAGS with value=&quot;kube_cluster:&lt;cluster_name&gt;&quot; is configured ~# oc get daemonset &lt;datadog_agent_daemonset_name&gt; -n &lt;datadog_agnet_namespace&gt; --yaml ... - name: DD_TAGS value: kube_cluster:my-cluster</td>
</tr>
</tbody>
</table>

**Before You Start**

- The admin role for installing Federator.ai is "Cluster Admin."
- Datadog agent must be ready if Federator.ai runs in the same OpenShift cluster that is being monitored.
- Obtain Datadog account APIKey, APPKey.
  1. A Datadog account is required for connecting and using Datadog Cloud Service. If you don’t have an account, visit Datadog website and sign up for a free trial account. [https://www.datadoghq.com/](https://www.datadoghq.com/)
  2. Log in Datadog Cloud Service with your account and get an API key and Application key for using Datadog API
Copy the API Key and Application Key for Federator.ai Data-Adapter configuration

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. Go to Red Hat Marketplace and use keyword “federator” to search for products. Federator.ai Operator will be listed for installation.

4. Click “Purchase” or “Free Trial” to initiate the installation.
5. Click the “Install Operator” and select the namespace “federatorai” to start install the software. It’s recommended to keep all the default options.

You haven’t installed any Operators
You’re all ready to go, just click “Install Operator” to get started.

Install Operator

Update channel
Operators are organized into packages and streams of updates called “channels”. If an operator is available through multiple channels, you can choose which one you want to subscribe to. Learn more
• stable

Approval strategy
Automatic updates keep the operator and any instances on the cluster up to date. Manual updates require approval and are done via OpenShift console or CLI. Learn more
• Automatic
• Manual

Target clusters
Choose clusters where you want to install and manage this operator. Then select the Namespace scope for each cluster you are installing into. Learn more

<table>
<thead>
<tr>
<th>Name</th>
<th>Platform</th>
<th>Namespace Scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>json-test</td>
<td>Libvirt</td>
<td>federatorai</td>
</tr>
</tbody>
</table>

Install
6. It will take a few minutes to pull the software images and install Federator.ai in the “federatorai” project. Once the installation completes, Federator.ai will show up 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 (it is recommended) or leave the default ephemeral storage.

```yaml
apiVersion: federatorai.containers.ai/v1alpha1
kind: AlamedaService
metadata:
  name: my-alamedaservice
  namespace: federatorai
spec:
  keycode:
    codeNumber: D3JXM-LFTQK-QE23W-ZBNI-MA3WZ-A7HKQ
  enableExecution: false
  version: v4.3.956
  storages:
    - usage: log
      type: ephemeral
    - usage: data
      type: ephemeral
```
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 -> Route -> federatorai-dashboard-frontend”
10. Log in Federator.ai GUI with the default account id/password, “admin/admin”. The portal page of Federator.ai GUI shows the summary of Federator.ai and OpenShift cluster information.
Configuration
Federator.ai supports two types of applications, Kafka consumers and generic applications. The configuration procedure illustrated below uses one Kafka and one generic application (NGINX) as examples.

1. Prepare your Kafka configuration information if you will configure Federator.ai to manage Kafka consumers. This step is optional.

- Get Kafka connection string (e.g., “my-cluster-kafka-brokers.myproject:9092”)

```
~# oc get svc -n myproject
my-cluster-kafka-bootstrap  ClusterIP  10.107.237.39  <none>
9091/TCP,9092/TCP,9093/TCP,9404/TCP  15d
my-cluster-kafka-brokers   ClusterIP  None            <none>
9091/TCP,9092/TCP,9093/TCP  15d
my-cluster-kafka-exporter  ClusterIP  10.98.96.53    <none>    9404/TCP  15d
my-cluster-zookeeper-client ClusterIP  10.110.115.16  <none>  9404/TCP,2181/TCP  15d
my-cluster-zookeeper-nodes  ClusterIP  None            <none>  9404/TCP,2181/TCP,2888/TCP,3888/TCP  15d
```

- Find topic ID of interest (e.g., “topic001”)

```
~# oc get pod -n myproject
my-cluster-entity-operator-995df8959-vkwrn  3/3  Running  0  6d
my-cluster-kafka-0                           2/2  Running  0  3d5h
my-cluster-kafka-1                           2/2  Running  0  12h
my-cluster-kafka-2                           2/2  Running  0  4d3h
my-cluster-kafka-exporter-6b846888bd4dv2     1/1  Running  57  15d
my-cluster-zookeeper-0                       2/2  Running  0  6d
my-cluster-zookeeper-1                       2/2  Running  0  15d
my-cluster-zookeeper-2                       2/2  Running  0  15d
producer-topic001-8c8c4f5-xfdz7             1/1  Running  0  43h
strimzi-cluster-operator-77555d4b69-j4975    1/1  Running  1  6d
```

```
~# oc -n myproject exec my-cluster-kafka-0 -c kafka -- bin/kafka-topics.sh --bootstrap-server my-cluster-kafka-bootstrap:9092 --list
Open3DK 64-Bit Server VM warning: If the number of processors is expected to increase from one, then you should configure the number of parallel GC threads appropriately using -XX:ParallelGCThreads=N
- consumer_offsets
   topic0001
topic0002
```

- Find Consumer Group ID (e.g., “group0001”)

```
~# oc get pod -n myproject
my-cluster-entity-operator-995df8959-vkwrn  3/3  Running  0  6d
my-cluster-kafka-0                           2/2  Running  0  3d5h
my-cluster-kafka-1                           2/2  Running  0  12h
my-cluster-kafka-2                           2/2  Running  0  4d3h
...
```

```
~# oc -n myproject exec my-cluster-kafka-0 -c kafka -- bin/kafka-consumer-groups.sh --bootstrap-server my-cluster-kafka-bootstrap:9092 --list
Open3DK 64-Bit Server VM warning: If the number of processors is expected to increase from one, then you should configure the number of parallel GC threads appropriately using -XX:ParallelGCThreads=N
group0001
group0002
```
2. **Configure Federator.ai Data Adapter to connect to Datadog Service**

- Use the command to download configuration helper scripts from Github

  ```bash
  # curl https://raw.githubusercontent.com/containers-ai/federatorai-operator/master/deploy/federatorai-launcher.sh |bash
  ``

  Please input Federator.ai version tag (e.g., v4.2.755): v4.3.datadog

  - Downloading scripts ...
  - Downloading Federator.ai CR yamls ...
  - Downloading Federator.ai operator yamls ...
  - Done

  Do you want to use a private repository URL? [default: n]: n

  Do you want to launch the Federator.ai installation script? [default: y]: n

  Downloaded files are located under /tmp/federatorai-scripts/v4.3.datadog

- Use Data Adapter configuration helper script in “/tmp/federatorai-scripts/v4.3.datadog/scripts”

  ```bash
  # ls -l /tmp/federatorai-scripts/v4.3.datadog
  -rw-r--r-- 1 root root 11968 Aug 19 17:49 email-notifier-setup.sh
  -rw-r--r-- 1 root root 31708 Aug 19 17:49 federatorai-setup-for-datadog.sh
  -rw-r--r-- 1 root root 42722 Aug 19 17:49 install.sh
  -rw-r--r-- 1 root root 4949 Aug 19 17:49 node-label-assignor.sh
  -rw-r--r-- 1 root root 34819 Aug 19 17:49 planning-util.sh
  -rw-r--r-- 1 root root 49215 Aug 19 17:49 preloader-util.sh
  -rw-r--r-- 1 root root 1721 Aug 19 17:49 prepare-private-repository.sh
  -rw-r--r-- 1 root root 4433 Aug 19 17:49 uninstall.sh
  ```

- Change file permission to be executable

  ```bash
  # chmod +x federatorai-setup-for-datadog.sh
  ```

- Run the configuration helper script and follow the instructions to input configuration parameters

  ```bash
  # ./federatorai-setup-for-datadog.sh
  # cd /tmp/federatorai-scripts/v4.3.datadog/scripts
  # ./federatorai-setup-for-datadog.sh
  ```

  Checking environment version...
  
  ...Passed

  You are connecting to cluster: https://172.31.3.34:8443

  Do you want to reconfigure Datadog API & Application keys? [default: n]: n

  Do you want to configure alamedascaler for generic application? [default: y]: y

  Getting generic application info... No.1

  Input alamedascaler name []: nginx-sample

  Input cluster name []: k8s-4-205

  Getting controller info for nginx-sample alamedascaler... No.1

  Input target app kind (Deployment/DeploymentConfig/StatefulSet)[]: Deployment

  Input target app namespace []: nginx-sample

  Input Deployment name []: nginx-sample

  Do you want to enable HPA recommendation? [default: y]:

  Input minimum replicas number []: 1
Input maximum replicas number []: 5

Do you want to add another controller in nginx-sample alamedascaler? [default: n]: y

Getting controller info for nginx-sample alamedascaler... No.2
Input target app kind (Deployment/DeploymentConfig/StatefulSet)[]: Deployment
Input target app namespace []: nginx-sample
Input Deployment name []: nginx-sample-1
Do you want to enable HPA recommendation? [default: y]:
Input minimum replicas number []: 1
Input maximum replicas number []: 3

Do you want to add another controller in nginx-sample alamedascaler? [default: n]:
Do you want to add another generic application? [default: n]:

Do you want to configure alamedascaler for kafka? [default: y]:

Getting Kafka info... No.1
Input alamedascaler name []: kafka-consumer
Input cluster name []: k8s-4-205

Getting controller info for kafka-consumer alamedascaler... No.1
Input Kafka exporter namespace []: myproject
Input Kafka consumer group kind (Deployment/DeploymentConfig/StatefulSet) []: Deployment
Input Kafka consumer group kind name []: consumer1-topic0001-group-0001
Input Kafka consumer group namespace []: myproject
Input Kafka consumer topic name []: topic0001

You can use Kafka command-line tool 'kafka-consumer-group.sh' (download separately or enter into a broker pod, in /bin directory) to list consumer groups.
E.g.: "/bin/kafka-consumer-groups.sh --bootstrap-server <kafka-bootstrap-service>:9092 --describe --all-groups --members"
The first column of output is the 'kafkaConsumerGroupId'.
Input Kafka consumer group id []: group0001
Input Kafka consumer minimum replica number []: 1
Input Kafka consumer maximum replica number []: 3

Do you want to add another controller in kafka-consumer alamedascaler? [default: n]: y

Getting controller info for kafka-consumer alamedascaler... No.2
Input Kafka exporter namespace []: myproject
Input Kafka consumer group kind (Deployment/DeploymentConfig/StatefulSet) []: Deployment
Input Kafka consumer group kind name []: consumer2-topic0002-group-0002
Input Kafka consumer group namespace []: myproject
Input Kafka consumer topic name []: topic0002

You can use Kafka command-line tool 'kafka-consumer-group.sh' (download separately or enter into a broker pod, in /bin directory) to list consumer groups.
E.g.: "/bin/kafka-consumer-groups.sh --bootstrap-server <kafka-bootstrap-service>:9092 --describe --all-groups --members"
The first column of output is the 'kafkaConsumerGroupId'.
Input Kafka consumer group id []: group0002
Input Kafka consumer minimum replica number []: 1
Input Kafka consumer maximum replica number []: 5

Do you want to add another controller in kafka-consumer alamedascaler? [default: n]:
Do you want to add another Kafka set? [default: n]:

Updating Federator.ai data adapter configmap...
Warning: oc apply should be used on resource created by either oc create --save-config or oc apply configmap/federatorai-data-adapter-config configured
. . .Done.
Adding alamedascaler for generic applications...
alamedascaler.autoscaling.containers.ai/nginx-sample created

...Done.

Adding alamedascaler for Kafka...
alamedascaler.autoscaling.containers.ai/nginx-sample unchanged
alomadedascaler.autoscaling.containers.ai/kafka-consumer created

...Done.

Restarting Federator.ai data adapter...
pod "federatorai-data-adapter-b7d9db494-s9g6v" deleted

Checking pods...

All federatorai pods are ready.

...Done.

Setup Federator.ai for Datadog successfully
Yaml files generated are under ./config_result

#notes-1: input cluster name must match with the <cluster_name> configured in Datadog Agent
DD_TAGS (value="kube_cluster:<cluster_name>")

- Verify configuration result

```bash
~# ls -l config-result/
-rw-r--r-- 1 root root 35666 9月 16 12:05 adapter-configmap.yaml
-rw-r--r-- 1 root root 912 9月 16 12:05 kafka-consumer.yaml
-rw-r--r-- 1 root root 690 9月 16 12:05 nginx-sample.yaml
```

**kafka-consumer.yaml**

```yaml
---
apiVersion: autoscaling.containers.ai/v1alpha2
kind: AlamedaScaler
metadata:
  name: kafka-consumer
  namespace: federatorai
spec:
  clusterName: k8s-4-205
  controllers:
  - type: kafka
    enableExecution: false
    scaling: hpa
    kafka:
      exporterNamespace: myproject
      consumerGroup:
        namespace: myproject
        name: consumer1-topic0001-group-0001
        kind: Deployment
        topic: topic0001
        groupId: group0001
        hpaParameters:
          maxReplicas: 3
          minReplicas: 1
  - type: kafka
    enableExecution: false
    scaling: hpa
    kafka:
      exporterNamespace: myproject
      consumerGroup:
        namespace: myproject
        name: consumer2-topic0002-group-0002
```
3. (Optional) Install Datadog Watermark Pod Autoscaler Controller if you enable HPA autoscaling and would like to use WPA to do autoscaling

- Download Datadog WPA package

  ```
  # wget https://github.com/DataDog/watermarkpodautoscaler/archive/master.zip
  # unzip master.zip
  ```

- Install Watermark Pod Autoscaler controller

  WPA Helm Chart package requires using `helm` to install. If you don’t have `helm` installed, use the following command to install.

  ```
  ```

- Set up environment variables and then use `helm` command to install WPA

  ```
  # DD_NAMESPACE="default"
  # DD_NAMEWPA="wpacontroller"
  # helm install $DD_NAMEWPA -n $DD_NAMESPACE ./chart/watermarkpodautoscaler
  ```
```bash
~# pwd
/root/datadog_wpa/watermarkpodautoscaler
~# DD_NAMESPACE="default"
~# DD_NAMEWPA="wpacontroller"
~# helm install $DD_NAMEWPA -n $DD_NAMESPACE ./chart/watermarkpodautoscaler
~# oc get pods -n default

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>datadog-monitoring-61ckr</td>
<td>2/2</td>
<td>Running</td>
<td>0</td>
<td>2d19h</td>
</tr>
<tr>
<td>datadog-monitoring-cluster-agent-7d79559979-cnjhf</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>2d19h</td>
</tr>
<tr>
<td>datadog-monitoring-dwq7f</td>
<td>2/2</td>
<td>Running</td>
<td>0</td>
<td>2d19h</td>
</tr>
<tr>
<td>datadog-monitoring-hlm8x</td>
<td>2/2</td>
<td>Running</td>
<td>0</td>
<td>2d19h</td>
</tr>
<tr>
<td>datadog-monitoring-kube-state-metrics-7659787777d-b5dnq</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>6d3h</td>
</tr>
<tr>
<td>nfs-client-provisioner-7cd5f68cf7-cfqqb</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>6d3h</td>
</tr>
<tr>
<td>wpacontroller-watermarkpodautoscaler-6b4844f8dd4-zxm22</td>
<td>1/1</td>
<td>Running</td>
<td>18</td>
<td>6d3h</td>
</tr>
</tbody>
</table>

**Download WPA pod autoscaler CR yaml file**

```bash
~# wget https://github.com/DataDog/watermarkpodautoscaler/blob/master/deploy/crds/datadoghq.com_watermarkpodautoscalers_cr.yaml
```

**Edit datadoghq.com_watermarkpodautoscalers_cr.yaml**

Configure WPA to auto-scale Kafka consumer group and generic application (NGINX)

```yaml
~# mv datadoghq.com_watermarkpodautoscalers_cr.yaml wpa.yaml
~# vi wpa.yaml

apiVersion: datadoghq.com/v1alpha1
kind: WatermarkPodAutoscaler
metadata:
  name: consumer
  namespace: myproject
spec:
  # Add fields here
  # algorithm must be average
  algorithm: average
  maxReplicas: 10
  minReplicas: 1
  tolerance: 0.01
downscaleForbiddenWindowSeconds: 300
upscaleForbiddenWindowSeconds: 15
scaleUpLimitFactor: 90
scaleDownLimitFactor: 90
scaleTargetRef:
  kind: Deployment
  apiVersion: apps/v1
  name: consumer
readinessDelay: 10
metrics:
  # Resource or External type supported
  # Example usage of External type
  - type: External
    external:
      # do not edit highWatermakr, and lowWatermark
      # highWatermark and lowWatermark must be 1
      highWatermark: "1"
      lowWatermark: "1"
      metricName: federatorai.recommendation
      metricSelector:
        matchLabels:
          resource: replicas
          kube_cluster: k8s-4-205  # see below #notes-1 for more details
          kube_deployment: consumer
          kube_namespace: myproject
```

# Example usage of Resource type
#
- type: Resource
  
  resource:
    highWatermark: "50"
    lowWatermark: "10"
    name: cpu
    metricSelector:
      matchLabels:
        foo: bar

---

apiVersion: datadoghq.com/v1alpha1
kind: WatermarkPodAutoscaler
metadata:
  name: nginx-sample
  namespace: nginx-sample
spec:
  # Add fields here
  # algorithm must be average
  algorithm: average
  maxReplicas: 5
  minReplicas: 1
  tolerance: 0.01
  downscaleForbiddenWindowSeconds: 300
  upscaleForbiddenWindowSeconds: 15
  scaleUpLimitFactor: 90
  scaleDownLimitFactor: 90
  scaleTargetRef:
    kind: Deployment
    apiVersion: apps/v1
    name: nginx-sample
  readinessDelay: 10
  metrics:
  # Resource or External type supported
  # Example usage of External type
  - type: External
    external:
      # do not edit highWatermark, and lowWatermark
      # highWatermark and lowWatermark must be 1
      highWatermark: "1"
      lowWatermark: "1"
      metricName: federatorai.recommendation
      metricSelector:
        matchLabels:
          resource: replicas
          kube_cluster: k8s-4-205  
          kube_deployment: nginx-sample
          kube_namespace: nginx-sample

#notes-1: “kube_cluster” value must match with DD_TAGS (value=“kube_cluster:<cluster_name>”)
configured in Datadog Agent (datadog-values.yaml)

- Deploy WPA and confirm the status

~# oc apply -f wpa.yaml
Managing Federator.ai License Keycode

Federator.ai uses a keycode for license control. A 30-day trial keycode is installed by default. It is required to replace it with a valid keycode from ProphetStor in order 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.

Applying A New Keycode

1. Get the “AlamedaService” CR name
   ```bash
   ~# oc get alamedaservice --all-namespaces
   NAMESPACE     NAME                EXECUTION   VERSION    PROMETHEUS
   AGE
   federatorai my-alamedaservice false v4.3.958 https://prometheus-k8s.openshift-monitoring:9091 45d
   ```

2. Edit the “AlamedaService” CR
   ```bash
   ~# oc edit alamedaservice my-alamedaservice -n <namespace>
   ```

3. Go to “keycode:” section, replace the value of “codeNumber” with the new keycode and then save the change
   ```yaml
   apiVersion: federatorai.containers.ai/v1alpha1
   kind: AlamedaService
   metadata:
     name: my-alamedaservice
   .......
   spec:
     .......
     keycode:
       codeNumber: K4AM0C4TSDXXXXXXXXXXXXXXXXXXXXXX
   ```

Deleting An Existing Keycode

1. Get the “AlamedaService” CR name
   ```bash
   ~# oc get alamedaservice --all-namespaces
   NAMESPACE     NAME                EXECUTION   VERSION    PROMETHEUS
   AGE
   federatorai my-alamedaservice false v4.3.958 https://prometheus-k8s.openshift-monitoring:9091 45d
   ```

2. Edit the “AlamedaService” CR
   ```bash
   ~# oc edit alamedaservice my-alamedaservice
   ```
3. Go to “keycode:” section, delete the keycode from “codeNumber” and then save the change

```
apiVersion: federatorai.containers.ai/v1alpha1
kind: AlamedaService
metadata:
  name: my-alamedaservice
......
spec:
  ......
  keycode:
  codeNumber:
```

Activating A Keycode

1. Get the “AlamedaService” CR name

```
~# oc get alamedaservice --all-namespaces
NAMESPACE     NAME                EXECUTION   VERSION    PROMETHEUS
federatorai   my-alamedaservice   false       v4.3.958   https://prometheus-k8s.openshift-monitoring:9091   45d
```

2. Edit the “AlamedaService” CR

```
~# oc edit alamedaservice my-alamedaservice
```

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

```
apiVersion: federatorai.containers.ai/v1alpha1
kind: AlamedaService
metadata:
  name: my-alamedaservice
......
status:
    ......
  keycodeStatus:
    codeNumber: K4AM0C4TSDKXXXXXXXXXXXXXXXXXXXXXXXXQ
    lastErrorMessage: ""
  registrationData: H4sICAavJl8C/2ZlZGFpLXJLZ2RhdGEudGd6AQ3ad1DTZjABR0I0MoEKKmHzSakXyGh81wJB8kb4kRAgZ1wxa3oAIKUPYoSB0EY0uoKquADCGmgkiQCigqVoYgURCFI3tr3wegjWpwILNYA0UFy4hgsAVYrCGxiBFQExUNC05jUZoYTQmPYj2r+cIwazo/1vy76w+yWM/B2c3R3uElIEsagUrZnhh3s6lyZ/YfrFk5cVpI86XqYU4XqnoFLDYPsp01xeTw5iR365hoIoFkb8RD7VQP2prLF+eEPGkmGsIS7AxNoeI1cR2W6u76ek03Lp/TEBQxrKoUXEP5t1vF3RNgq6N2UoyPbrr+828Zi
e9613bzfzvHVs++z23vYvfu/f5/6UxdKPokbGQDo1kh6yOYgWjXX2mE8M9FX/mNgtBtg+/50/3j6P...
J3hu2gMPKh7XEl16h4Ojf5vPdF0Cvxb8zbTXkyjk1VgolSdVXGd1HDARd6sWvbcrbQbLP3M2bD9
```

---

Federator.ai 4.3 for OpenShift Installation Guide  21
4. Once ProphetStor received the activation request email and validated the “registrationData”, you will receive the license activation code, “signatureData”, via an email. Copy the “signatureData” from the email to the “keycode.signatureData” field and save the change. Your license keycode is now activated.

```yaml
apiVersion: federatorai.containers.ai/v1alpha1
kind: AlamedaService
metadata:
  name: my-alamedaservice

spec:
  keycode:
    codeNumber: K4AMOC4TSDDXXXXXXXXXXXXXXXXXXXXXX
    signatureData: F5nus478ertgnld430gvsef90gnYAt0UFY4hgsAYYrCGCxiBfQExUNC0S
                     KkMhZSiKSAygG pi86XqYU4E4a3oAiKUPYoSB0E0YooKquADCmkgkiQCigYrCGxergHwernREBo4E
                     wegjWpWWLNyat0UFY4hgsAYYrCGCxmUxdkP0kG95jUZoYTQmPYj2r+ciWazo/1vy76W+yW/B2c3R3
                     OYgjWx2mE8M9fh3s6lyZ/YfrFk5CVpidkP0kboMQDo1khRTNB0p1xeTw5iR365ho1ofk8dRD7VQP
                     2prLF+uEPkmGsIS7AxNoef1cR2W6u7GekO3Lp/TEBxrkOuXEP5M1vF3RNqd6N2UoyPbrr+8Z8i
                     e9613bfzzvHs+/2Z1zGFpLXJ1ZA03ad1DTXjA8R91oxdkP0kboMQDo1kh6yOYg8M9fX/RwtrBVerh
                     ...YBoerBTR445h4536g456UJdfsheryhryu6JwerJwerYjJKERS5ZQ6kZrFFhrk6sVwbcrbQLPregUh9
                     XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```
Appendix

Datadog Dashboards

The following Custom Datadog Dashboards are available after Federator.ai is installed.

- **ProphetStor Federator.ai Cluster Overview**

- **ProphetStor Federator.ai Application Overview**
• ProphetStor Federator.ai Kafka Overview

With integration of ProphetStor Federator.ai, users can easily track the Kafka message production/consumption rate, as well as the prediction of message production rate from Federator.ai dashboard. Based on the prediction of message production rate, Federator.ai automatically scales Kafka consumer replicas to handle the workload. This can be visualized from Federator.ai dashboard where the recommended consumer replicas and the current number of consumer replicas are shown. Additionally, overall consumer logs as well as the average latency in the queue before a message is received by a consumer are also shown on the dashboard for better performance monitoring.

• ProphetStor Federator.ai Cost Analysis Overview

<table>
<thead>
<tr>
<th>Multicloud Cost Analysis</th>
<th>Current Cluster Configuration</th>
<th>AWS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Current Cluster Cost</td>
<td>2078.52, toms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recommended Cluster</td>
<td>-AWS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Provision</td>
<td>Display Name</td>
<td>Size</td>
<td>Distance Type</td>
</tr>
<tr>
<td>aws</td>
<td>m4.large, hi2, 24v, m4.large</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>aws</td>
<td>m4.large, hi2, 24v, m4.large</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>Recommended Cluster</td>
<td>-Azure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Provision</td>
<td>Display Name</td>
<td>Size</td>
<td>Distance Type</td>
</tr>
<tr>
<td>azure</td>
<td>standard-d4.xlarge, 24v, m4.large</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>azure</td>
<td>standard-d4.xlarge, 24v, m4.large</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>Recommended Cluster</td>
<td>-GCP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Provision</td>
<td>Display Name</td>
<td>Size</td>
<td>Distance Type</td>
</tr>
<tr>
<td>gcp</td>
<td>2 standard-d4.xlarge, 24v, m4.large</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>gcp</td>
<td>2 standard-d4.xlarge, 24v, m4.large</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>

Federator.ai 4.3 for OpenShift Installation Guide  24