<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProphetStor Federator.ai Application Overview</td>
<td>26</td>
</tr>
<tr>
<td>ProphetStor Federator.ai Kafka Overview</td>
<td>27</td>
</tr>
<tr>
<td>ProphetStor Federator.ai Cost Analysis Overview</td>
<td>27</td>
</tr>
<tr>
<td>Sysdig Dashboard Overview</td>
<td>28</td>
</tr>
<tr>
<td>Federator.ai Cluster Overview</td>
<td>28</td>
</tr>
<tr>
<td>Federator.ai Application Overview</td>
<td>28</td>
</tr>
<tr>
<td>Federator.ai Application Overview</td>
<td>28</td>
</tr>
<tr>
<td>Federator.ai installation/uninstallation using Helm Chart</td>
<td>29</td>
</tr>
<tr>
<td>Federator.ai installation/uninstallation using Ansible</td>
<td>31</td>
</tr>
<tr>
<td>Prerequisite</td>
<td>31</td>
</tr>
<tr>
<td>Preparation (Ansible Control Node):</td>
<td>32</td>
</tr>
<tr>
<td>Installing Federator.ai</td>
<td>32</td>
</tr>
<tr>
<td>Troubleshooting</td>
<td>34</td>
</tr>
<tr>
<td>Downgrade from v4.7.0</td>
<td>34</td>
</tr>
<tr>
<td>Datadog Integration</td>
<td>34</td>
</tr>
</tbody>
</table>
Overview

Federator.ai

ProphetStor Federator.ai is an AI-based solution that helps enterprises manage and optimize resources for applications on Kubernetes and virtual machines (VMs) in VMware or AWS EC2 clusters. Using advanced machine learning algorithms to predict application workloads, Federator.ai offers:

- AI-based workload prediction for containerized applications in Kubernetes clusters and VMs in VMware or AWS EC2 clusters
- Resource recommendations based on workload prediction, application, Kubernetes, and other related metrics
- Automatic scaling of application containers
- Multicloud cost analysis and recommendations based on workload predictions for Kubernetes clusters and VM clusters
- Actual cost and potential savings based on recommendations for clusters, Kubernetes applications, VMs, and Kubernetes namespaces

Supported Metrics Data Sources

There are five different types of metrics data sources supported in released v4.7.0: Prometheus, Datadog, Sysdig, VMWare vCenter and AWS CloudWatch.

Prometheus (Kubernetes, Rancher, RedHat OpenShift)

Prometheus is a free and open-source event monitoring tool for containers or microservices. It uses the principle of scraping to collect numerical data based on time series. Metrics are collected in regular timestamps and stored locally. Federator.ai supports using Prometheus gathering Kubernetes cluster metrics, and leverage collected data for workload predictions, recommendations for resource planning, autoscaling containers/pods, and cost analysis for clusters deployed in a multicloud environment.

The following diagram shows how the metrics are collected from Prometheus by Federator.ai in a Kubernetes environment.
Datadog

Federator.ai has integrated with Datadog and utilizes the metrics collected by Datadog Agent for workload predictions. The following diagram shows how application 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 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

Sysdig

Federator.ai has integrated with Sysdig and utilizes the metrics collected by Sysdig Agent for workload predictions. The following diagram shows how application metrics are used by Federator.ai to predict workload and to automatically scale applications for better performance.
VMWare vCenter

VMware vCenter Server provides integrated management of all hosts and virtual machines in the data center from a single console, allowing IT administrators to improve control, simplify daily work, and reduce the complexity and cost of managing the IT environment. Federator.ai data adapter connects to VMware vCenter servers via VMware SDK to retrieve all of VMs workload metrics data for predictions, recommendations and cost analysis for VM clusters.

AWS CloudWatch

AWS CloudWatch is a monitoring service for AWS cloud resources and the applications running on AWS cloud. It provides visibility into resource utilization, operational performance, and overall demand patterns. The metrics collected by CloudWatch by default do not include memory usage of EC2 instances. CloudWatch agent is required for Federator.ai to collect memory usage metrics. Federator.ai supports two types of AWS VM clusters:
- Auto Scaling Group
- Individual VM
Requirements and Recommended Resource Configuration

Platforms

- OpenShift : 4.3~4.7
- Kubernetes : 1.11 ~ 1.22
- Rancher v2.4.8,v2.5.8,v2.5.9
- EKS/AKS/GKE

Data Source

- Datadog
- Sysdig
- Prometheus (Kubernetes, Rancher, OpenShift)
- VMWare vCenter 5.5/6.0/6.5/6.7/7.0
- AWS CloudWatch

Federator.ai Resource Requirements

- Total Resource Requirements
  - Request : 5.1 CPU cores (Limit :22 cores)
  - Request : 5.0 GB Memory (Limit : 42GB)
  - StorageClass: 168GB (require ReadWriteOnce access mode)

- Resource requirements for AI Engine
  - There must be at least one worker node with at least 2 CPU(Limit : 8 cores) cores and 1 GB of memory available
  - The 2 CPU cores and 1 GB memory are included in the total 5.1 CPU cores and 5.0 GB memory requirements

Federator.ai Version

- Version: Release 4.7
- Tag : v4.7.0-ga

Datadog Agent Version(reference)

- Datadog Agent helm chart version: v2.4.24, v2.13.0
- Datadog Agent version: v7.21.1, v7.27.0
- Datadog Cluster Agent version: v1.8.0, v1.12.0
- Datadog Watermark Pod Autoscaler version: v0.1.0

Prometheus Version(reference)

- OpenShift
  - Default installed Prometheus
- **Rancher**
  - Default bundled Prometheus
- **Kubernetes**
  - prometheus-operator-8.5.11
  - Rancher v2.4.8 kube-prometheus-stack-12.3.0
  - kube-prometheus-release-0.6
  - kube-prometheus-stack-12.5.0/15.4.6/17.0.3

**Sysdig Agent Version** *(reference)*
- Sysdig agent: 11.2.0/11.3.0

**Persistent Volumes**
- The StorageClass that provides the persistent volumes must support RWO (ReadWriteOnce) access mode.
- It is recommended to use persistent volumes instead of using ephemeral storage to store the data in the production environment. Any data on ephemeral storage will be lost after Federator.ai pods are restarted.

For Federator.ai’s application-aware resource/performance optimization feature, the following versions of applications are supported:

**Kafka**
- Kafka operator version *(Reference)*: Strimzi/kafka:0.17.0-kafka-2.4.0

**NGINX as Ingress**
- “nginx-ingress-controller” 0.23.0+ *(0.23.0 release on 2019 Feb 28)*
  - [https://github.com/kubernetes/ingress-nginx/blob/master/Changelog.md#0230](https://github.com/kubernetes/ingress-nginx/blob/master/Changelog.md#0230)
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: Preparation

- For Datadog, obtain API Key, Application Key of Datadog Cloud Service account. Instructions are provided below.
- For Prometheus, obtain Prometheus service URL (ex: http://<prometheus_svc_name>.<namespace>:9090)
- For Sysdig, obtain Sysdig API URL and Token.
- For VMware vCenter, obtain administrator login credential and vCenter IP or FQDN.
- For AWS CloudWatch, obtain Access Key ID and Secret Access Key of AWS account.

Step 2:

- For Datadog, install and configure Datadog Agent/Cluster Agent if they have not been installed. Please follow the Datadog documentation on how to install Datadog Agent and Cluster Agent.
- For Sysdig, install and configure Sysdig Agent. Please follow Sysdig documentation on how to install Sysdig Agent.

Step 3: Install Federator.ai.

Step 4: Configure Federator.ai Data Adapter for the external metrics data source via Federator.ai Initial Setup Wizard.

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

Step 6: Review installation result on Datadog/Sysdig Cloud Dashboard.

Pre-installation Check List

Kubernetes:

<table>
<thead>
<tr>
<th>#</th>
<th>Checklist Item</th>
<th>Requirement</th>
<th>Details</th>
</tr>
</thead>
</table>
| 1  | What is the Kubernetes version?                     | 1.11~1.22   | Use the command below to get the Kubernetes version: $ kubectl version ...
<p>| 2  | Does installation on this Kubernetes cluster require a private image repository? | If a private image repository is required, the following information is needed during installation | Input the URL and credential when the Federator.ai installation script asks for the information. |</p>
<table>
<thead>
<tr>
<th>StorageClass and Persistent Volumes requirement</th>
<th>StorageClass supports ReadWriteOnce access mode. Available storage size is larger than 168GB.</th>
<th>The minimum storage size for Federator.ai Release v4.7.0 is 168GB, including database, data, and logs.</th>
</tr>
</thead>
</table>
| Kubernetes cluster CPU/memory requirement     | Minimum CPU/mem/storage:  
- CPU: 5,100 (mcores)  
- Memory: 5.0 (GB)  
- Storage Class Capacity: 168GB  
At least one worker node with  
- CPU: 2 Cores  
- Memory: 1GB | To be able to run the AI Engine pod, there must be at least one worker node that has more than 2 CPU cores and 1 GB of memory available.  
2 CPU Cores and 1GB for AI Engine are included in the total 5.1 CPU Cores and 5 GB memory requirements. |
| Is this Kubernetes cluster allowed for NodePort configuration? | Federator.ai creates two NodePorts for GUI and REST API by default  
- REST API - https://<server>:31011  
- GUI - https://<server>:31012 | If NodePort is not allowed, answer ‘N’ when the installation script prompts for creating NodePorts.  
Users need to expose Federator.ai GUI and REST API service manually. |
| Will there be a resource quota imposed for the namespace where Federator.ai is installed? | CPU/mem request quota should be more than the minimum resource requirement  
- CPU: 5.1 Cores  
- Memory: 5.0 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 the number of managed clusters/applications increases.  
Use the command to get namespace resource quota  
$ kubectl get resourcequota --all-namespaces |
| Does this deployment require resource request/limit specified? | By default, Federator.ai deployments do not specify resource requests/limits. It can be done by setting up an environment variable before installation starts. | To turn on resource request/limit settings for all Federator.ai deployments, manually export environment variable before running ‘federatorai-launcher.sh’  
$ export ENABLE_RESOURCE_REQUIREMENT=y  
$ ./federatorai-launcher.sh |

### Prometheus:

<table>
<thead>
<tr>
<th>#</th>
<th>Checklist Item</th>
<th>Requirement</th>
<th>Details</th>
</tr>
</thead>
</table>
| 1   | What is the Prometheus version? (for Kubernetes) | Recommended version-Prometheus operator helm chart version: 8.5.11-Prometheus operator version: 0.34.0 -Prometheus server version: 2.13.1 | Use the command below to get Prometheus version:  
$ helm ls -A |grep -i prometheus  
prometheus-operator monitoring 1 2020-03-13 15:35:05.28963154 +0800 CST deployed prometheus-operator-2.1.3 v0.6.0  
prometheus-operator monitoring 1 2020-03-13 14:34:16.132479221 +0800 CST deployed prometheus-operator-8.12.1 0.37.0 |
$ kubectl get deployment -A -o custom-columns=IMAGE:.spec.template.spec.containers[0].image | grep -i prometheus

directman12/k8s-prometheus-adapter-amd64: v0.6.0
quay.io/coreos/prometheus-operator: v0.37.0

Datadog Agent:

<table>
<thead>
<tr>
<th>#</th>
<th>Checklist Item</th>
<th>Requirement</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Is Datadog Agent installed?</td>
<td>Datadog Agent is mandatory</td>
<td>Kubernetes resources and workload metrics are collected by Datadog Agent.</td>
</tr>
<tr>
<td>2</td>
<td>Is Datadog Cluster Agent installed?</td>
<td>Cluster Agent is mandatory</td>
<td>Cluster Agent provides metrics to HPA Autoscaler for autoscaling.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for the HPA autoscaling</td>
<td>feature</td>
</tr>
<tr>
<td>3</td>
<td>Is Datadog WPA controller installed?</td>
<td>Datadog WPA is required</td>
<td>Datadog WPA is the HPA Autoscaler developed by Datadog. Users can use</td>
</tr>
<tr>
<td></td>
<td></td>
<td>if auto-scaling is done by</td>
<td>Datadog WPA or Kubernetes native HPA to do autoscaling.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WPA</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Datadog Kafka Consumer integration is</td>
<td>Datadog Kafka Consumer</td>
<td>Use the command to confirm Kafka integration is enabled</td>
</tr>
<tr>
<td></td>
<td>enabled?</td>
<td>integration is mandatory</td>
<td>$ kubectl exec &lt;datadog-agent-pod&gt; -n &lt;datadog-agent-namespace&gt; --</td>
</tr>
<tr>
<td></td>
<td></td>
<td>if user wants to use</td>
<td>agent integration show datadog-kafka-consumer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Kafka optimization feature</td>
<td>Refer to <a href="https://www.datadoghq.com/blog/monitor-kafka-with-datadog/">https://www.datadoghq.com/blog/monitor-kafka-with-datadog/</a></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>for Kafka Consumer integration installation</td>
</tr>
<tr>
<td>5</td>
<td>Datadog account API key</td>
<td>An API key is mandatory for</td>
<td>Follow the steps described in the “Before You Start” session to obtain</td>
</tr>
<tr>
<td></td>
<td></td>
<td>connecting Datadog Service</td>
<td>the API key.</td>
</tr>
<tr>
<td>6</td>
<td>Datadog account Application key</td>
<td>An application key is</td>
<td>Follow the steps described in the “Before You Start” session to obtain</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mandatory for connecting</td>
<td>the Application key.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datadog Service</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Is one of cluster name is configured</td>
<td>&quot;kube_cluster&quot;,&quot;cluster_name&quot;</td>
<td>Case 1.&gt; New Datadog Agent installation: Install Datadog agent and</td>
</tr>
<tr>
<td></td>
<td>for the Datadog agent/cluster agent?</td>
<td>,&quot;kub e_cluster_name(DD_CLSUTER_NAME)&quot; one of them is required for</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.&gt;DD_TAGS with value</td>
<td>Federation.ai to identify</td>
<td>Federation.ai to identify Kubernetes cluster.</td>
</tr>
<tr>
<td></td>
<td>2 rallion name =”kube_cluster:&lt;cluster_name &gt;”in values.yaml</td>
<td>Kubernetes cluster.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>or</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.”cluster_name” in</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>values.yaml, or</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.”DD_CLUSTER_NAME” in</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datadog cluster agent</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>deployment</td>
<td></td>
</tr>
</tbody>
</table>
|    |                                        | "kube_cluster","cluster_name","kub  
|    |                                        | e_cluster_name(DD_CLSUTER_NAME)" one of them is required for           |
|    |                                        | Federation.ai to identify    |                                                                         |
|    |                                        | Kubernetes cluster.         |                                                                         |
|    |                                        |                                                                         |                                                                         |
- $helm upgrade ...
- Check “DD_Cluster_Name”
  $kubectl get daemonset
  <datadog_agent_daemonset_name> -n
  <datadog_agnet_namesapce> -o yaml

- name: DD_CLUSTER_NAME
  value: <cluster-name>

3. In Datadog Agent and Cluster Agent installed environment, with no cluster_name setting
Update Datadog Agent by "helm upgrade"
- assign a cluster name
datadog:
  clusterName: <cluster-name>
- $helm upgrade ...
- Check “DD_Cluster_Name”
  $kubectl get daemonset
  <datadog_agent_daemonset_name> -n
  <datadog_agnet_namesapce> -o yaml

As:
- name: DD_CLUSTER_NAME
  value: <cluster-name>

4. In Datadog Agent and Cluster Agent installed environment, with cluster_name setting
Use the command below to confirm DD_Cluster_Name
- $kubectl get daemonset
  <datadog_agent_daemonset_name> -n
  <datadog_agnet_namesapce> -o yaml

As:
- name: DD_CLUSTER_NAME
  value: <cluster-name>

Before You Start

Datadog
- The admin role for installing Federator.ai is "Cluster Admin."
- Datadog agent must be ready if Federator.ai runs in the same Kubernetes cluster that is being monitored.
- Obtain Datadog account API Key, Application Key.
  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/
  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 metrics data source configuration

Sysdig

- Different Sysdig API URL is needed for different regions:
  - For US East, Sysdig API URL is https://app.sysdigcloud.com
  - For US West, Sysdig API URL is https://us2.app.sysdig.com
  - For European Union, Sysdig API URL is https://eu1.app.sysdig.com
- Copy Sysdig Monitor API Token for Federator.ai metrics data source configuration
VMware vCenter

- You can define a VM cluster from any VMs under the same cluster path. See below for an example of cluster path on vCenter.

```
cluster_path:
"Datacenter/cluster1"
```

```
"my-vm-cluster1"
```

```
"my-vm-cluster2"
```

AWS CloudWatch

- Obtain CloudWatch Account Key ID and Secret Access Key.
1. Use your AWS account ID or account alias, your IAM username, and your password to sign in to the IAM console.

2. Go to “Access management > Users > Security credentials” to get Access Key ID and Secret Access Key

New Installation

(For upgrading from previous Federator.ai, please refer to the next section.)

1. Log into Kubernetes cluster

2. Install the Federator.ai for Kubernetes by using the following command

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

   ~# curl https://raw.githubusercontent.com/containers-ai/prophetstor/master/deploy/federatorai-launcher.sh | bash
   Downloading v4.7.0-ga tgz file ...
   Done
   Do you want to use a private repository URL? [default: n]:
   Do you want to launch the Federator.ai installation script? [default: y]:
   Executing install.sh ...
   Checking environment version...
   ...Passed
   Enter the namespace you want to install Federator.ai [default: federatorai]:

   `tag_number = v4.7.0-ga
   install_namespace = federatorai`

   Is the above information correct? [default: y]:

   Downloading v4.7.0-ga tgz file ...
   Done
   Applying Federator.ai operator yaml files...
   `namespace/federatorai created`
   `serviceaccount/federatorai-operator created`
   `customresourcedefinition.apixtensions.k8s.io/alamedaservices.federatorai.containers.ai created`
   `deployment.apps/federatorai-operator created`
   `clusterrole.rbac.authorization.k8s.io/federatorai-operator created`
   `clusterrole.rbac.authorization.k8s.io/alameda-gc created`
   `clusterrolebinding.rbac.authorization.k8s.io/federatorai-operator created`
   `role.rbac.authorization.k8s.io/federatorai-operator created`
   `role.rbac.authorization.k8s.io/alameda-gc created`
Checking pods...
Waiting for pod federatorai-operator-609566b7c-rmphp in namespace federatorai to be ready.
phase: [Running]
Waiting for pods in namespace federatorai to be ready...

All pods under namespace(federatorai) are ready.

Install Federator.ai operator v4.7.0-ga successfully

Downloading Federator.ai alamedaservice sample file ...
Done

Downloading Federator.ai alamedascaler sample files ...
Done

Which storage type you would like to use? ephemeral or persistent?
[default: persistent]:
Specify log storage size [e.g., 2 for 2GB, default: 2]:
Specify AI engine storage size [e.g., 10 for 10GB, default: 10]:
Specify InfluxDB storage size [e.g., 100 for 100GB, default: 100]:
Specify storage class name: managed-nfs-storage
Do you want to expose dashboard and REST API services for external access? [default: y]:

-------------------------------------------------------------------
install_namespace = federatorai
storage_type = persistent
log storage size = 2 GB
AI engine storage size = 10 GB
InfluxDB storage size = 100 GB
storage class name = managed-nfs-storage
expose service = y
-------------------------------------------------------------------
Is the above information correct [default: y]:
Processing...
Waiting for datahub(v4.7.0-ga) pod to appear ...

datahub pod is present.

Checking pods...
Waiting for pod alameda-ai-66f5c7b6b4-rx87j in namespace federatorai to be ready. phase: [Pending]
Waiting for pods in namespace federatorai to be ready...
Waiting for pod alameda-ai-66f5c7b6b4-rx87j in namespace federatorai to be ready. phase: [Pending]
Waiting for pods in namespace federatorai to be ready...
Waiting for pod alameda-ai-66f5c7b6b4-rx87j in namespace federatorai to be ready. phase: [Pending]
Waiting for pods in namespace federatorai to be ready...
Waiting for pod alameda-ai-66f5c7b6b4-rx87j in namespace federatorai to be ready. phase: [Running]
Waiting for pods in namespace federatorai to be ready...
Waiting for pod alameda-operator-7ff69f4bb5-v22ws in namespace federatorai to be ready. phase: [Running]
Waiting for pods in namespace federatorai to be ready...

All pods under namespace(federatorai) are ready.
The default alamedaorganization under namespace federatorai is ready.

You can now access GUI through https://<YOUR IP>:31012
The default login credential is admin/admin
Also, you can start to apply alamedascaler CR for the target you would like to monitor. Review the administration guide for further details.

You can now access Federatorai REST API through https://<YOUR IP>:31011
The default login credential is admin/admin
The REST API online document can be found in https://<YOUR IP>:31011/apis/v1/swagger/index.html

Install Federator.ai v4.7.0-ga successfully
Downloaded YAML files are located under /opt/federatorai/installation
Downloaded files are located under /opt/federatorai/repo/v4.7.0-ga

3. Verify Federator.ai pods are running properly

```
~# kubectl get pod -n federatorai
NAME                  READY STATUS   RESTARTS AGE
alameda-ai-66f5c7b6b4-rx87j 1/1  Running   0  16m
alameda-ai-dispatcher-78d8556bd5-c4h2l 1/1  Running   0  16m
alameda-analyzer-668566588-6cprq 1/1  Running   0  16m
alameda-datahub-79f85cd56-ff8j6 1/1  Running   1  16m
alameda-executor-7f9d899578-85jxz 1/1  Running   3  16m
alameda-influxdb-0 1/1  Running   0  16m
alameda-notifier-8bb7cc889-c6b8x 1/1  Running   2  16m
alameda-operator-7ff69f4bb5-v22ws 1/1  Running   5  16m
alameda-rabbitmq-d5868cd97-n62gt 1/1  Running   0  16m
fedemeter-api-5cb94ff9c9-dbnxw 1/1  Running   0  16m
fedemeter-influxdb-0 1/1  Running   0  16m
federatorai-agent-7cccf55f84-jjjb4 1/1  Running   0  16m
federatorai-dashboard-backend-5fd697fd86-kks5n 1/1  Running   0  16m
federatorai-dashboard-frontend-59595cc866-6578j 1/1  Running   0  16m
federatorai-data-adapter-76bc6ff8b7-xrdrk 1/1  Running   0  16m
federatorai-operator-669566b7c-rmpkh 1/1  Running   0  18m
federatorai-recommender-dispatcher-77b974bb6-cbvlt 1/1  Running   0  16m
federatorai-recommender-worker-554c7f8694-w9w5j 1/1  Running   0  16m
federatorai-rest-75bf7dfff54-smxh8 1/1  Running   1  16m
```
4. Log on Federator.ai GUI and finish installation through the Initial Setup Wizard. For more information on Initial Setup Wizard, please see Federator.ai 4.7.0 User Guide.

https://<master_node_IP>:31012
Login ID: admin
Password: admin
Upgrade from a previous version

Federator.ai v4.7.0 supports upgrade from previous versions. The Federator.ai installation script automatically detects previously installed Federator.ai. When the installation script prompts if a backup of the previous configuration is needed, just enter yes to save a copy of the configuration if roll back to the previous version is needed.

Prerequisite

1. Federator.ai version is 4.6.x
2. Federator.ai installed and running with Persistent Volume

Upgrade

1. Log into Kubernetes cluster
2. Install the Federator.ai for Kubernetes by using the following command

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

~# curl https://raw.githubusercontent.com/containers-ai/prophetstor/master/deploy/federatorai-launcher.sh | bash
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
Dload  Upload   Total   Spent    Left  Speed
100 16783  100 16783    0     0  25155      0  --:--:-- --:--:-- --:--:-- 25161
Please enter Federator.ai version tag [default: latest]: v4.7.0-ga
Please input Federator.ai files save path [default: /opt]:

Download v4.7.0-ga tgz file ...
Done
Do you want to use a private repository URL? [default: n]:
Do you want to launch the Federator.ai installation script? [default: y]:

Executing install.sh ...
Checking environment version...
...Passed
Previous build with tag v4.6.1-ga detected in namespace federatorai

----------------------------------------
Upgrade:
tag_number = v4.7.0-ga
install_namespace = federatorai
----------------------------------------
Is the above information correct? [default: y]:
Please input path for storing backup configuration: [default: /opt/federatorai/configuration_backup]
Backup configuration...
backup yamls saved to folder /opt/federatorai/configuration_backup/federatorai-backup-1616551135
Done.

Downloading v4.7.0-ga tgz file ...
Done
Updating InfluxDB owner...
Done

Applying Federator.ai operator yaml files...
deployment.apps "federatorai-operator" deleted
Applying 00-namespace.yaml...
namespace/federatorai unchanged
Applying 01-serviceaccount.yaml...
serviceaccount/federatorai-operator unchanged
Applying 02-alamedaservice.crd.yaml...
customresourcedefinition.apiextensions.k8s.io/alamedaservices.federatorai.containers.ai configured
Delay applying 03-federatorai-operator.deployment.yaml
Applying 04-clusterrole.yaml...
clusterrole.rbac.authorization.k8s.io/federatorai-operator configured
Applying 05-clusterrolebinding.yaml...
clusterrolebinding.rbac.authorization.k8s.io/federatorai-operator unchanged
Applying 06-role.yaml...
role.rbac.authorization.k8s.io/federatorai-operator configured
Applying 07-rolebinding.yaml...
rolebinding.rbac.authorization.k8s.io/federatorai-operator unchanged
Applying 08-service.yaml...
service/federatorai-operator-service created
Applying 09-secret.yaml...
secret/federatorai-operator-service-cert created
Applying 10-mutatingwebhook.yaml...
mutatingwebhookconfiguration.admissionregistration.k8s.io/federatorai-operator servicesmutation created
Applying 11-validatingwebhook.yaml...
validatingwebhookconfiguration.admissionregistration.k8s.io/federatorai-operator servicesvalidation created
Applying 03-federatorai-operator.deployment.yaml...
deployment.apps/federatorai-operator created
federatorai-operator pod is present.

Waiting for pod federatorai-operator-75bdc65496-tz6fx in namespace federatorai to be ready ... phase: [Pending]
Waiting for pod federatorai-operator-75bdc65496-tz6fx in namespace federatorai to be ready ... phase: [Pending]
Waiting for pod federatorai-operator-75bdc65496-tz6fx in namespace federatorai to be ready ... phase: [Pending]
Waiting for pod federatorai-operator-75bdc65496-tz6fx in namespace federatorai to be ready ... phase: [Running]

federatorai-operator pod is ready.

Downloading Federator.ai alamedaservice sample file ...
Done

Downloading Federator.ai alamedascaler sample files ...
Done

======================================================================================================
Update alamedaservice...
alamedaservice.federatorai.containers.ai/my-alamedaservice patched
Done.
alamedaservice.federatorai.containers.ai/my-alamedaservice patched
alamedaservice.federatorai.containers.ai/my-alamedaservice patched
Processing...
Waiting for datahub(v4.7.0-ga) pod to appear ...
datahub pod is present.

Checking pods...
Waiting for pod alameda-ai-667df48565-99qft in namespace federatorai to be ready. phase: [Running]
Waiting for pods in namespace federatorai to be ready...
Waiting for pod alameda-ai-667df48565-99qft in namespace federatorai to be ready. phase: [Running]
Waiting for pods in namespace federatorai to be ready...
Waiting for pod alameda-ai-dispatcher-6845456b68-8kjfr in namespace federatorai to be ready. phase: [Running]
Waiting for pods in namespace federatorai to be ready...

All pods under namespace(federatorai) are ready.
The default alamedaorganization under namespace federatorai is ready.

========================================================================
You can now access GUI through https://<YOUR IP>:31012
The default login credential is admin/admin
Also, you can start to apply alamedасасcaler CR for the target you would like to monitor.
Review the administration guide for further details.
========================================================================

========================================================================
You can now access Federatorai REST API through https://<YOUR IP>:31011
The default login credential is admin/admin
The REST API online document can be found in https://<YOUR IP>:31011/apis/v1/swagger/index.html
========================================================================

Install Federator.ai v4.7.0-ga successfully

Downloaded YAML files are located under /opt/federatorai/installation
Downloaded files are located under /opt/federatorai/repo/v4.7.0-ga

3. Verify Federator.ai pods are running properly

~# kubectl get pod -n federatorai
NAME                READY STATUS    RESTARTS AGE
alameda-ai-6b56d6db77-xzr9x  1/1   Running   0  19m
alameda-ai-dispatcher-7d46f46849-nd4z2  1/1   Running   3  19m
alameda-analyzer-56bd4d4f8d-jvwe6f  1/1   Running   0  19m
alameda-datahub-597fb6f964-pqmh6  1/1   Running   3  20m
alameda-executor-6b4bff9b47-d9fdw  1/1   Running   4  19m
alameda-influxdb-0  1/1   Running   0  19m
alameda-notifier-87cf6b94c-xn5pz  1/1   Running   1  19m
alameda-operator-cf64f6c9-c677m  1/1   Running   0  20m
alameda-rabbitmq-addcc8dd7-99jv2  1/1   Running   0  19m
# Installing Datadog Watermark Pod Autoscaler (WPA)

If you wish to enable HPA autoscaling via Datadog WPA for your application, please follow the instructions below to install Datadog WPA.

- **Download Datadog WPA package**

```bash
~# 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.*

  ```bash
  ```

- **Set up environment variables and then use ‘helm’ command to install WPA**

  ```bash
  $ DD_NAMESPACE="default"
  $ DD_NAMEWPA="wpacontroller"
  $ helm install $DD_NAMEWPA -n $DD_NAMESPACE ./chart/watermarkpodautoscaler
  ```

- **Download WPA pod autoscaler CR yaml file**

```bash
~# pwd
/root/datadog_wpa/watermarkpodautoscaler
~# DD_NAMESPACE="default"
~# DD_NAMEWPA="wpacontroller"
~# helm install $DD_NAMEWPA -n $DD_NAMESPACE ./chart/watermarkpodautoscaler
~# kubectl get pods -n default
NAME                            READY STATUS    RESTARTS AGE
datadog-monitoring-6lckr        2/2     Running   0       2d19h
datadog-monitoring-cluster-agent-7d79559979-cnjhj 1/1     Running   0       2d19h
datadog-monitoring-dwq7f         2/2     Running   0       2d19h
datadog-monitoring-hlm8x         2/2     Running   0       2d19h
datadog-monitoring-kube-state-metrics-765978777d-b5dnq 1/1     Running   0       6d3h
nfs-client-provisioner-7cd5f68cf7-cfqqb 1/1     Running   0       6d3h
wpacontroller-watermarkpodautoscaler-68484f8dd4-zxm22 1/1     Running   18      6d3h
```
~# 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)

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

```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
downscaledForbiddenWindowSeconds: 300
upscaledForbiddenWindowSeconds: 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
    # 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  # 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 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  
          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

~# kubectl apply -f wpa.yaml
Appendix

Datadog Dashboards Overview

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 and 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 replica 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>Cluster Configuration</th>
<th>AWS</th>
<th>Azure</th>
<th>GCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recommended Cluster</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Savings</td>
<td>839.22</td>
<td>786.82</td>
<td>570.15</td>
</tr>
<tr>
<td>CPU Capacity</td>
<td>1239.30</td>
<td>1291.70</td>
<td>1508.37</td>
</tr>
</tbody>
</table>

Current Cluster Cost: 2078.52
Sysdig Dashboard Overview

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

Federator.ai Cluster Overview

Federator.ai Application Overview
Federator.ai Application Overview

Federator.ai installation/uninstallation using Helm Chart

Prerequisites

- Kubernetes version 1.18 or later
- OpenShift version 4.x.x or later
- Helm version is 3.x.x or later

Add Helm chart repository

```bash
~# helm repo add prophetstor https://prophetstor-ai.github.io/federatorai-operator-helm/
```

Test the Helm chart repository

```bash
~# helm search repo federatorai
```

Installing with the release name my-name:

```bash
~# helm install `my-name` prophetstor/federatorai --namespace=federatorai --create-namespace
```

To uninstall/delete the my-name deployment:

```bash
~# helm ls --all-namespaces
helm delete `my-name` --namespace=federatorai
```
Configuration

The following table lists the configurable parameters of the chart and their default values are specified in values.yaml.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>image.pullPolicy</td>
<td>Container pull policy</td>
</tr>
<tr>
<td>image.repository</td>
<td>Image for Federator.ai operator</td>
</tr>
<tr>
<td>image.tag</td>
<td>Image Tag for Federator.ai operator</td>
</tr>
<tr>
<td>federatorai.imageLocation</td>
<td>Image Location for services containers</td>
</tr>
<tr>
<td>federatorai.version</td>
<td>Image Tag for services containers</td>
</tr>
<tr>
<td>federatorai.persistence.enabled</td>
<td>Enable persistent volumes</td>
</tr>
<tr>
<td>federatorai.persistence.storageClass</td>
<td>Storage Class Name of persistent volumes</td>
</tr>
<tr>
<td>federatorai.persistence.storages.logStorage.size</td>
<td>Log volume size</td>
</tr>
<tr>
<td>federatorai.persistence.aiCore.dataStorage.size</td>
<td>AlCore data volume size</td>
</tr>
<tr>
<td>federatorai.persistence.influxdb.dataStorage.size</td>
<td>Influxdb data volume size</td>
</tr>
<tr>
<td>federatorai.persistence.fedemeterInfluxdb.dataStorage.size</td>
<td>Fedemeter influxdb data volume size</td>
</tr>
<tr>
<td>services.dashboardFrontend.nodePort</td>
<td>Port of the Dashboard service</td>
</tr>
</tbody>
</table>

Specify each parameter using the --set key=value[,key=value] argument to helm install

Tip: You can use the default values.yaml

Sample: values.yaml

```yaml
## Default values for Federator.ai
## This is a YAML-formatted file.
## Declare variables to be passed into your templates.
##
image:
  pullPolicy: IfNotPresent
  repository: quay.io/prophetstor/federatorai-operator-ubi
  tag: v4.7.0-ga

## Set default values
##
federatorai:
  imageLocation: quay.io/prophetstor
  version: v4.7.0-ga
```
## If the persistence is enabled, a default StorageClass is needed in the k8s cluster to provision volumes.

```yaml
persistence:
  enabled: true
  storageClass: "standard"

storages:
  logStorage:
    size: 2Gi
  aiCore:
    dataStorage:
      size: 10Gi
  influxdb:
    dataStorage:
      size: 100Gi
  fedemeterInfluxdb:
    dataStorage:
      size: 10Gi

services:
  dashboardFrontend:
    ## Specify the nodePort value for the dashboard frontend
    ## Comment out the following line to disable nodePort service
    nodePort: 31012
  rest:
    ## Specify the nodePort value for the REST service
    ## Comment out the following line to disable nodePort service
    nodePort: 31011
```

### Alternative installation with configuration file

A YAML file that specifies the values for the parameters can be provided while installing the chart. For example

```bash
~# helm install my-name prophetstor/federatorai -f values.yaml --namespace=federatorai --create-namespace
```

### Federator.ai installation/uninstallation using Ansible

Only support Federator.ai since v4.4.0 or later

#### Prerequisite

**Ansible Control Node**

<table>
<thead>
<tr>
<th>Software</th>
<th>Version:</th>
<th>Query Command:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ansible</td>
<td>2.10.2 or later</td>
<td>ansible --version</td>
</tr>
<tr>
<td>Ansible Collection - community.kubernetes</td>
<td>1.1.1 or later</td>
<td>ansible-galaxy collection list or ansible-galaxy collection install community.kubernetes -vvv</td>
</tr>
<tr>
<td>Python</td>
<td>3.7 or later</td>
<td>python3 --version</td>
</tr>
</tbody>
</table>
Preparation (Ansible Control Node):

1. Install Ansible
   
   [Link: https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html]

2. Install collection “community.kubernetes”
   
   ```
   ~# ansible-galaxy collection install community.kubernetes
   ```

3. Install python & pip
   
   [Link: https://www.python.org/downloads/]

4. Install OpenShift python client if you are using OpenShift clusters
   
   ```
   ~# pip3 install openshift
   ```

5. Download Ansible playbook for Federator.ai

6. Modify user_variable.yaml file for customizing needed info.

Installing Federator.ai

**Variables for in user_variable.yml**

<table>
<thead>
<tr>
<th>Group</th>
<th>Variable Name</th>
<th>Sample value</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Federator.ai env</td>
<td>federatorai_version</td>
<td>v4.7.0-ga</td>
<td>Federator.ai version tag</td>
<td>Y</td>
</tr>
<tr>
<td>Storage for Federator.ai pods</td>
<td>storage_type</td>
<td>ephemeral or persistent</td>
<td>Using ephemeral persistent volume type</td>
<td>Y</td>
</tr>
<tr>
<td>Storage info</td>
<td>log_storage_size</td>
<td>10</td>
<td>Log size reserved for every pod. 10 means 10GB</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Private repo</td>
<td>enable_private_repo</td>
<td>y</td>
<td>Using private repo to pull the Federator.ai required docker images</td>
<td>N</td>
</tr>
<tr>
<td>Pod resource</td>
<td>enable_resource_requirement</td>
<td>y</td>
<td>Add pod resource requirement (limits &amp; requests) for every Federator.ai pod</td>
<td>N</td>
</tr>
</tbody>
</table>
Expose services (Only be used when openshift_env is “n”)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>expose_dashboard_and_rest_api_services</td>
<td>y</td>
<td>Expose the dashboard and API services in the Kubernetes cluster.</td>
<td>Y</td>
</tr>
</tbody>
</table>

Cluster type

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshift_env</td>
<td>n</td>
<td>Input “y” if installed cluster is OpenShift cluster</td>
<td>Y</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>installed_namespace</td>
<td>federatorai</td>
<td>namespace where Federator.ai will be installed</td>
<td>N</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>image_url_prefix</td>
<td>&quot;&quot;</td>
<td>Input the private repo URL</td>
<td>N</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>ai_engine_size</td>
<td>10</td>
<td>Storage size reserved for Alameda AI engine.</td>
<td>N</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>influxdb_storage_size</td>
<td>100</td>
<td>Data size reserved for InfluxDB pod.</td>
<td>N</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>storage_class_name</td>
<td>scname</td>
<td>To specifying storage class name for provisioning persistent volumes</td>
<td>Y</td>
</tr>
</tbody>
</table>

Steps:

1. Go to Ansible playbook folder

```
~ # cd ansible_for_federatorai
```

2. Modify user_variable.yaml (under uninstaller folder) file for customizing needed info.

3. Export K8S_AUTH_KUBECONFIG to specify kubeconfig file path for Ansible collection (community.kubernetes).

```
~ # export K8S_AUTH_KUBECONFIG=/root/.kube/config.135
```

4. Run Ansible playbook

```
~ # ansible-playbook federatorai_installation.yaml
```

Uninstalling Federator.ai

For Uninstallation, please use the file under ansible_for_federatorai/uninstaller directory.

Variables in user_variable.yml.
<table>
<thead>
<tr>
<th>Group</th>
<th>Variable Name</th>
<th>Sample value</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage for Federator.ai pods</td>
<td>storage_type</td>
<td>ephemeral or persistent</td>
<td>Specify current Federator.ai storage type (ephemeral or persistent)</td>
<td>Y</td>
</tr>
<tr>
<td>Preserve current persistent volume (Only be used when storage_type is persistent)</td>
<td>preserve_pv</td>
<td>Y</td>
<td>Specify whether to preserve Federator.ai PVs</td>
<td></td>
</tr>
</tbody>
</table>

Steps:

1. Get to Ansible playbook uninstallation folder

   ```
   ~# cd ansible_for_federatorai/uninstaller
   ```

2. Modify user_variable.yaml (under uninstaller folder) file for customizing needed info.

3. Export `K8S_AUTH_KUBECONFIG` to specify kubeconfig file path for Ansible collection (community.kubernetes)

   ```
   ~# export K8S_AUTH_KUBECONFIG=/root/.kube/config.135
   ```

4. Run Ansible playbook

   ```
   ~# ansible-playbook federatorai_uninstaller.yaml
   ```

Troubleshooting

Downgrade from v4.7.0

- V4.7.0->v4.6.x

  Following v4.6.x installation step to rollback to previous version v4.6.x directly. v4.6.x installation script automatically keeps existing configuration, metrics, and prediction data, which is stored on persistent volume.

Datadog Integration

1. Datadog WPA dumps errors during autoscaling

   - Error message in WPA Controller

     ```
     ~# kubectl get pod -n default
     NAME           READY STATUS    RESTARTS AGE
     datadog-agent-2m6kk 1/1  Running  2  2d
     datadog-agent-8kd54  1/1  Running  0  2d
     datadog-agent-94r16  1/1  Running  0  2d
     ```
<table>
<thead>
<tr>
<th>Service</th>
<th>Status</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>datadog-agent-mq4mv</td>
<td>1/1</td>
<td>- WPA is incompatible with Kubernetes 1.11</td>
</tr>
<tr>
<td>datadog-cluster-agent-74f44fdd4d-82tjp</td>
<td>1/1</td>
<td>- Install WPA on Kubernetes 1.11 dumps errors</td>
</tr>
<tr>
<td>docker-registry-1-wv59s</td>
<td>1/1</td>
<td>must only have &quot;properties&quot;, &quot;required&quot; or &quot;description&quot; at the root if the status subresource is enabled</td>
</tr>
<tr>
<td>prometheus-adapter-799b7dfc4f-rs7zj</td>
<td>1/1</td>
<td>Workaround</td>
</tr>
<tr>
<td>registry-console-2-jxfd1</td>
<td>1/1</td>
<td>- Comment out ‘subresources’ key in WatermarkPodAutoscaler CRD</td>
</tr>
<tr>
<td>router-1-sw71</td>
<td>1/1</td>
<td>~# cd</td>
</tr>
<tr>
<td>wpacontroller-watermarkpodautoscaler-7ffbb97f9d-hcbsg</td>
<td>1/1</td>
<td>datadog_wpa/watermarkpodautoscaler_for_k8s_1.11/chart/watermarkpodautoscaler/templates</td>
</tr>
<tr>
<td></td>
<td></td>
<td>datadoghq.com_watermarkpodautoscalers_crd.yaml</td>
</tr>
</tbody>
</table>

```
~# kubectl logs wpacontroller-watermarkpodautoscaler-7ffbb97f9d-hcbsg -n default
```

```

["level":"error","ts":1589633961.600972,"logger":"wpa_controller","msg":"Error during reconcileWPA","Request.Namespace":null,"Request.Name":null,"consumer1-topic0001-group-0001":null,"error":null,"the server could not find the requested resource (put watermarkpodautoscalers.datadoghq.com consumer1-topic0001-group-0001)","stacktrace":null]  
```

```sh
cd datadog_wpa/watermarkpodautoscaler_for_k8s_1.11/chart/watermarkpodautoscaler/templates
v datadoghq.com_watermarkpodautoscalers_crd.yaml
...
...
  shortNames:
    - wpa
  singular: watermarkpodautoscaler
  scope: Namespaced
  #subresources:  < comment out
  # status: {}  < comment out
  validation:
    openAPIV3Schema:
```
WatermarkPodAutoscaler is the Schema for the watermarkpodautoscalers API properties:

apiVersion:
  description: 'APIVersion defines the versioned schema of this representation...

Note: It can auto-scale monitored application, but dump some errors during update status

- Related Datadog WPA ticket
  

2. Data Adapter reports errors

- Error messages in Data Adapter logs

```
$ oc get pods | grep federatorai-data-adapter | grep Running | awk '{print $1}' | -- cat /var/log/telegraf.log
```

- Reason
  
  Datadog Agent does not work with 'kube-state-metrics' comes with OpenShift

- Solution
  
  Install another compatible ‘kube-state-metrics’

  If there is another kube-state-metrics running on openshift, rename all the clusterrole and clusterrolebinding name of kube-state-metrics to prevent kube-state-metrics clusterrole name collision

  restart datadog agent and make sure agent integrate with kube-state-metrics property.

  check all the node agent status by following command

  ```
  $ oc exec <datadog-agent-pod-name> agent status
  ```