

Kubernetes

Enterprise plugin

This plugin allows you to obtain data from a Kubernetes environment, generating agents for each of its elements and monitoring statistics.

The information is obtained through the web, by means of the Kubernetes API, so it is not necessary to install any additional software for its operation. Although, in order to obtain the CPU and memory usage data of the containers and nodes, it will be necessary to install the "metrics-server" addon in the Kubernetes environment.

- [Introduction](#)
- [Compatibility Matrix](#)
- [Prerequisites](#)
- [Permit assignement for the api](#)
- [Metrics-server installation](#)
- [Settings](#)
- [Generated agent settings](#)
- [XML file transfer settings](#)
- [Additional settings](#)
- [Manual execution](#)
- [Modules generated by the plugin](#)

Introduction

This plugin allows you to obtain data from a Kubernetes environment, generating agents for each of its elements and monitoring statistics.

The information is obtained through the web, by means of the Kubernetes API, so it is not necessary to install any additional software for its operation. Although, in order to obtain the CPU and memory usage data of the containers and nodes, it will be necessary to install the "metrics-server" addon in the Kubernetes environment.

Compatibility Matrix

Developed for:

- Kubernetes v1.14.3.
- Metrics-server 1.8+.

Prerequisites

Connection with the Tentacle service associated with your Pandora FMS server is required.

Access credentials or an authentication token to the Kubernetes API of a user with enough permissions to query the API will be required.

Optionally, if you want to obtain the CPU and memory usage data of containers and nodes, install the "metrics-server" addon in the Kubernetes environment.

Permit assignement for the api

The following describes the steps to be followed to create a user with enough permissions to obtain the monitoring data.

1. Create a read "*Cluster role*" called "*api-read-only*": Create a role that grants the "*get*", "*list*" and "*watch*" permissions of all Kubernetes resources.

```
cat <<EOF | kubectl apply -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  annotations:
    rbac.authorization.kubernetes.io/autoupdate: "true"
  labels:
    name: api-read-only
rules:
- apiGroups:
  - '*'
resources:
- '*'
verbs:
- get
- list
- watch
- nonResourceURLs:
  - '*'
verbs:
- get
- list
- watch
EOF
```

2. Create a "*Cluster role binding*" called "*bind-api-read-only*": The previously created role will be

binded to an existing "*service account*."

```
kubectrl create clusterrolebinding bind-api-read-only \  
--clusterrole=api-read-only \  
--serviceaccount=namespace:user
```

Metrics-server installation

To install the "*metrics-server*" addon, download it in the Kubernetes environment. It can be obtained from its github project:

<https://github.com/kubernetes-incubator/metrics-server>

Once downloaded, deploy it through the "*kubectl*" command:

```
kubectl apply -f metrics-server/deploy/1.8+/
```

You can verify that the deployment was successful if, upon launching the following command, the status of its API is "*true*":

```
$ kubectl get apiservices | grep metrics-server  
  
v1beta1.metrics.k8s.io kube-system/metrics-server True 1m
```

If the deployment was successful, the "*metrics-server*" API should stay accessible:

```
$ kubectl get --raw "/apis/metrics.k8s.io/v1beta1/"  
  
{ "kind": "APIResourceList", "apiVersion": "v1", "groupVersion": "metrics.k8s.io/v1beta1", "resources": [ { "name": "nodes", "singularName": "", "namespaced": false, "kind": "NodeMetrics", "verbs": [ "get", "list" ] }, { "name": "pods", "singularName": "", "namespaced": true, "kind": "PodMetrics", "verbs": [ "get", "list" ] } ] }
```

In addition, after a few seconds, it should be possible to obtain CPU and memory usage data of containers and nodes:

```
$ kubectl top node  
  
NAME CPU(cores) CPU% MEMORY(bytes) MEMORY%  
  
kube 93m 4% 968Mi 56%  
  
$ kubectl top pod  
  
NAME CPU(cores) MEMORY(bytes)
```

pod1 0m 3Mi

pod2 0m 3Mi

pod3 0m 2Mi

pod4 0m 3Mi

pod5 0m 1Mi

Settings

The plugin configuration file "*pandora_kubernetes*" is divided into sections (all configuration parameters will be indicated without quotation marks even if they have blank spaces):

API access settings

API connection parameters

api_ip=192.168.80.145

api_port=8443

ssl=1

HTTP bearer authentication parameters

auth_method=bearer

auth_token=auth-token

HTTP basic authentication parameters

#auth_method=basic

#auth_user=user

#auth_pass=pass

api_ip

IP address to the Kubernetes API service.

port

Port listening to the Kubernetes API service.

auth_token

If "*auth_method=bearer*" indicates the authentication token of the user with access to the API. This token can be obtained by following these steps:

1. Obtaining the user's "*secret name*":

```
$ kubectl get serviceaccounts user -o yaml

apiVersion: v1

kind: ServiceAccount

metadata:

  creationTimestamp: "2019-06-19T10:11:48Z"

  name: user

  namespace: namespace

  resourceVersion: "327"

  selfLink: /api/v1/namespaces/namespace/serviceaccounts/user

  uid: a60f3652-927a-11e9-b423-ae1361a56794

  secrets:

    - name: secret_name
```

2. Obtaining the "*token*" of the "*secret name*":

```
$ kubectl describe secret secret_name

Name: secret-name

Namespace: namespace

Labels: <none>

Annotations: kubernetes.io/service-account.name: default
```

kubernetes.io/service-account.uid: a60f3652-927a-11e9-b423-ae1361a56794 Type: kubernetes.io/service-account-token

Data

====

token: auth-token

ca.crt: 1066 bytes

namespace: 7 bytes

auth_user

If "*auth_method=basic*" points out the user with access to the API.

ssl

Indicates whether the connection to the API is HTTPS or not (1 → HTTPS, 0 → HTTP).

auth_method

Authentication mechanism to be used. The values are either "*bearer*" or "*basic*".

auth_pass

If "*auth_method=basic*" indicates the user password with access to the API.

Generated agent settings

```
## Agents parameters
```

```
interval=300
```

```
group=Servers
```

```
#prefix=KUBE
```

interval

Monitoring interval of the generated agents in seconds. It will allow to define when modules will go into unknown status (by default, 2 times the defined interval without receiving any data).

group

Group to which the generated agents will be assigned. This group will be taken into account only if the "*autocreate_group*" parameter of the Pandora FMS server has not been correctly defined.

prefix

It allows to include a text string before the names of the generated agents, in order to identify the execution they come from (in case of having several plugin executions configured).

Agent mode

```
## Agent mode for deployments and pods
```

```
## 1 = learning mode
```

```
## 0 = normal mode
```

```
## 2 = autodisable mode (default value)
```

```
agent_mode=2
```

normal mode

To not disable the "deployments" and "pods" elements

autodisable mode

To disable "deployments" and "pods" elements (default)

XML file transfer settings

```
## XML local transfer parameters

tmp=/tmp

transfer_mode=local

local_folder=/var/spool/pandora/data_in

## XML remote transfer parameters

#tmp=/tmp

#transfer_mode=tentacle

#tentacle_ip=127.0.0.1

#tentacle_port=41121
```

tmp

Temporary directory in which the XML files of the generated agents will be created before being transferred to the Pandora FMS server.

transfer_mode

File transfer method to be used. If it is not set as "*tentacle*", the transfer method considered will be "*local*" (copying the XML files from the temporary directory to a defined one).

local_folder

Directory to which the XML files will be copied if the transfer method is not established as "*tentacle*".

tentacle_ip

IP address to which the XML files will be sent if the transfer method is established as "*tentacle*".

tentacle_port

Port to connect to the Tentacle server indicated in the "*tentacle_ip*" parameter.

tentacle_opts

Additional options for transferring files to the indicated Tentacle server.

Additional settings

```
## Disable monitoring

get_healthz=1

get_namespaces=1

get_services=1

get_components=1

get_pods = 1

get_nodes=1

get_metrics=1

## Additional Kubernetes metrics

# Example:

# Apiserver request latencies

bucket=apiserver_request_latencies_bucket{component="apiserver",group="scheduling.k8s.
io",resource="priorityclasses",scope="cluster",subresource="",verb="WATCH",version="v1 ",le="125000"}

## Extra parameters

#debug=0
```

get_healthz

If it is set to "0", the *"/healthz"* API monitoring data will no longer be obtained. Its default value if not specified is "1" (enabled).

get_namespaces

If it is set to "0", the `/api/v1/namespaces` API monitoring data will no longer be obtained. If not specified, its default value is "1" (enabled).

get_services

If it is set to "0", the `/api/v1/services` API monitoring data will no longer be obtained. If not specified, its default value is "1" (enabled).

get_components

If it is set to "0", the `/api/v1/componentstatuses` API monitoring data will no longer be obtained. If not specified, its default value is "1" (enabled).

get_pods

If set to '0' `/api/v1/pods` and `/apis/metrics.k8s.io/v1beta1/pods` API monitoring data (if the `metrics-server` addon has been installed) will no longer be obtained. If not specified, its default value is "1" (enabled).

get_nodes

If set to '0' `/api/v1/nodes` and `/apis/metrics.k8s.io/v1beta1/nodes` API monitoring data (if the `metrics-server` addon has been installed) will no longer be obtained. If not specified, its default value is "1" (enabled).

get_deployments

If set to '0' `/apis/extensions/v1beta1/deployments` API monitoring data will no longer be obtained. If not specified, its default value is "1" (enabled).

discard_agents

List of agents (separated by commas) to be discarded from plugin monitoring. They must be the names of the agents without the prefix indicated.

get_metrics

If it is set to "0", the `/metrics` API monitoring data will no longer be obtained. If not specified, its default value is "1" (enabled).

These metrics must be specified in the configuration file with the following format: **module name =metric**

For example, if it is specified in the configuration file:

Apiserver request latencies

```
bucket=apiserver_request_latencies_bucket{component="apiserver",group="scheduling.k8s.io",resource="priorityclasses",scope="cluster",subresource="",verb="WATCH",version="v1",le="125000"}
```

A module will be generated, which is called:

Apiserver request latencies bucket

with this metric value:

```
apiserver_request_latencies_bucket{component="apiserver",group="scheduling.k8s.io",resource="priorityclasses",scope="cluster",subresource="",verb="WATCH",version="v1",le="125000"}
```

debug

If it is set to "1", it will show detailed information about the actions that are carried out during the execution of the plugin. If not specified, its default value is "0" (disabled).

Manual execution

To run the plugin, configure the "*pandora_kubernetes.conf*" configuration file according to the preceding instructions.

Plugin execution:

```
./pandora_kubernetes.64 pandora_kubernetes.conf
```

Modules generated by the plugin

The execution of this plugin will generate the following agents and modules:

- **An agent named "Kubernetes":** It will contain monitoring data that is not related to containers and nodes. It will contain these modules: ○ **API status:** Indicates whether the API is accessible by the plugin.
- **Healthz:** Indicates whether the API is accessible by the plugin.
- **Healthz ping:** Indicates whether the API is accessible by the plugin.
- **Healthz log:** Indicates whether the API is accessible by the plugin.
- **Healthz etcd:** Indicates whether the API is accessible by the plugin.
- **Healthz poststarthook crd informer synced:** Indicates whether the API is accessible by the plugin.
- **Healthz poststarthook generic apiserver start informers:** Indicates whether the API is accessible by the plugin.
- **Healthz poststarthook start apiextensions controllers:** Indicates whether the API is accessible by the plugin.
- **Healthz poststarthook start apiextensions informers:** Indicates whether the API is accessible by the plugin.
- **Namespaces:** Amount of "*namespaces*" in the environment.
- **Services:** Amount of "*services*" in the environment.
- **Deployments:** Amount of "*deployments*" in the environment.
- **Components:** Amount of "*components*" in the environment.
- **Component <component>:** It will generate a module for each "*component*" indicating its status (healthy).

○ **<metric module name>**: It will generate a module for each metric indicated in the configuration file (with the indicated names).

- **An agent for each node**: Its parent agent will be "*Kubernetes*". They will contain these modules:

- **Pods**: Amount of "*pods*" in the node.
- **Pods (%)**: Occupancy percentage of "*pods*" in the node.
- **CPU (cores)**: CPU usage of the node.
- **CPU (%)**: CPU usage percentage of the node.
- **Memory (bytes)**: Node memory usage.
- **Memory (%)**: Node memory usage percentage.
- **<condition>**: It will generate a module for each "*condition*" in the node indicating its status.

- **An agent for each "pod"**: Its parent agent will be the node where it is located. They will contain the modules:

- **Pod status**: Indicates the status of "*pod*". Possible states:

■ 0 → Failed

■ 1 → Running

■ 2 → Succeeded

■ 3 → Pending

■ 4 → Unknown

- **Containers**: Number of containers in the "*pod*".

- **Container <container> CPU (cores)**: It will generate a module for each "*container*" in the "*pod*" indicating the CPU usage in the node.

- **Container <container> CPU (%):** It will generate a module for each "container" in the "pod" indicating the CPU usage in the node.
- **Container <container> memory (bytes):** It will generate a module for each "container" in the "pod" indicating the CPU usage in the node.
- **Container <container> memory (%):** It will generate a module for each "container" in the "pod" indicating the CPU usage in the node.
- **<condition>:** It will generate a module for each "condition" in the "pod" indicating its status.

- **An agent for each deployment:** Its parent agent will be "Kubernetes". They will contain these modules:

- **Replicas:** Total number of non-terminated pods targeted by this deployment (their labels match the selector).
- **Updated replicas:** Total number of non-terminated pods targeted by this deployment that have the desired template spec.
- **Ready replicas:** Total number of ready pods targeted by this deployment.
- **Available replicas:** Total number of available pods (ready for at least minReadySeconds) targeted by this deployment.
- **Unavailable replicas:** Total number of unavailable pods targeted by this deployment. This is the total number of pods that are still required for the deployment to have 100% available capacity. They may either be pods that are running but not yet available or pods that still have not been created.
- **Available:** Indicates whether the deployment is available.
- **Progressing:** Indicates whether the deployment is rolling out a new replica set.