Star   Fork   Watch   Follow @collabnix

Monitoring in Kubernetes

Kubernetes - Beginners | Intermediate | Advanced

View on GitHub   Join Slack

Monitoring in Kubernetes

Understanding Kubernetes monitoring pipeline(s) is essential to help you diagnose run-time problems and to manage the scale of your pods, and cluster. Monitoring is one of these areas that are evolving very rapidly inside Kubernetes. It has a lot of pieces that are still in the influx and hence some confusion.

Kubernetes has two monitoring pipelines:

- The core metrics pipeline, which is an integral part of Kubernetes and alw
- The services monitoring (non-core) pipeline, which is a separate pipeline,

Core Monitoring Pipeline

Sometimes is referred to as the resource metrics pipeline. The core monitoring pipeline is installed with every distribution. It provides enough details to other components inside the Kubernetes cluster to run as expected, such as the scheduler to allocate pods and

containers, HPA and VPA to take proper decisions scaling pods.

Core Monitoring

The way it works is relatively simple:

```
• CAdvisor collects metrics about containers and nodes that on which it is i
• Kubelet exposes these metrics (default is one-minute resolution) through K
• Metrics Server discovers all available nodes and calls Kubelet API to get
• Metrics Server exposes these metrics through Kubernetes aggregation API.
```

Note:-

```
• Kubelet cannot run without CAdvisor. If you try to uninstall it or stop it
• Even though Heapster "soon to be deprecated" is currently dependent on CAd
```

Services Monitoring Pipeline

Services pipeline in abstract terms is relatively simple. Confusion usually comes from the plethora of services, agents that you can mix and match to get your pipeline up and running. Also, you can blame Heapster for that.

Services Monitoring Pipeline consists of three main components

```
• Collection agent
• Metrics Server
• Dashboards
```

Below is the typical workflow, including most common components

```
• Monitoring agent collects node metrics. cAdvisor collects containers and p
• Monitoring Aggregation service collects data from its own agent and cAdvis
```

```
    • Data is stored in the monitoring system's storage.
    • Monitoring aggregation service exposes metrics through APIs and dashboards
```

Service Monitoring

Some Monitoring Solutions:-

```
  • Prometheus – It is the official monitoring server sponsored and incubated
  • Kubernetes Dashboard – https://github.com/kubernetes/dashboard
  • Jaeger – https://github.com/jaegertracing/jaeger
  • Kube Watch – https://github.com/bitnami-labs/kubewatch
  • Weave Scope – https://github.com/weaveworks/scope
  • EFK Stack – Fluentd, Elastic Search and Kibana
```

Notes:-

```
  • Almost all monitoring systems piggyback on Kubernetes scheduling and orche
  • Most monitoring agents depend on Kubelet to collect container relevant met
  • Most monitoring aggregation services depend on agents pushing metrics to t
```

What should you consider in Kubernetes Services Pipeline?

Ideal Services pipeline depends on two main factors: • collection of relevant metrics • Awareness of continuous changes inside kubernetes cluster.

A good pipeline should focus on collecting relevant metrics. There are plenty of agents that can collect OS and process-level metrics. But you will find very few out there that can collect details about containers running at a given node, such as the number of running containers, container state, docker engine metrics, etc. cAdvisor is the best agent IMO for this job so far.

Awareness of continuous changes means that the monitoring pipeline is aware of different pods, containers instances and can relate them to their parent entities, i.e. Deployment, Statefulsets, Namespace, etc. It also means that the metrics server is aware of system-wide metrics that should be visible to users, such as the number of pending

pods, nodes status, etc.

Metrics Visualization

You can visualize metrics in many different ways. The most common open source tool that easily integrates with Prometheus is Grafana. The challenges you will face though is building proper dashboards to monitor the right metrics. That said, you should have dashboards monitoring the following:

```
• Cluster level capacity utilization, this shows how much CPU memory being a
• Kubernetes Orchestration Metrics, which tracks the status of your pods and
• Kubernetes Core Services, which visualizes the status of critical services
• Application Specific Metrics, which tracks the status of your apps. They s
```

Note

Grafana is not best suited for alerting. I see a lot of teams depend on it to create alerting rules. However, it is not as reliable and comprehensive as Prometheus alerting manager.

Changes To Watch For

Heapster is Going Away

Heapster is currently causing some confusion given that it is used to show both core pipeline metrics and services metrics. In reality, you can remove Heapster and nothing bad will happen to the core Kubernetes scheduling and orchestration scenarios. It was the default monitoring pipeline and I guess it still is the default in a lot of distributions. But you don't have to use it at all.

So, the Kubernetes community wanted to make the separation clearer between core and services monitoring pipelines. Hence, Heapster will be deprecated and replaced by the Metrics Server (MS) as the main source of aggregated core metrics. Think of the MS as a trimmed down version of Heapster.

Major immediate changes are: • No historical data or queries • Eliminating a lot of

container-specific metrics, pod focus metrics only. Metrics Server is meant to provide core metrics that are needed for core Kubernetes scenarios, such as autoscaling, scheduling, etc..

Metrics Server Will Get More Cool Features

- Infrastore will store Metric Server historical data with a support of simp
- It will support initially metrics collected by the Metrics Server. Maybe K

Join KubeDaily

8 Members Online

Support

MEMBERS ONLINE

- h3ll_boy
- MEE6
- Nitinkashyap   Apex Legends
- ojaswa
- Parmeshwar
- prasad
- Shady
- vikas027

Free voice chat from Discord | Connect |

Tweets by collabnix

kubelabs **is maintained by** **collabnix**.

This page was generated by GitHub Pages.