

106Kubernetes 系列（九十九）使用 Loki 和 k8s Event Exporter 可视化 Event 进行监控

在本主题中，我们将探讨收集 Kubernetes 事件并在 Grafana 仪表板上可视化它们的目标。为了实现这一目标，我们需要一个 Kubernetes 集群、部署在集群内的 Grafana 和 Helm。用于此目的的关键组件是 Kubernetes 事件导出器和 Grafana Loki。让我们深入了解如何将这些组件组合在一起以实现有效的事件监控和可视化。

前提条件

Kubernetes Event Exporter 是一个强大的工具，可以解决 Kubernetes 事件丢失的问题。它将这些事件导出到各种输出，提高可观察性并启用自定义警报。借助此导出器，将事件导出到其他工具变得毫不费力，从而增强事件可观察性、创建自定义警报并有效聚合事件数据。

在本教程中，我们将使用 Loki 作为日志聚合器，并使用 Grafana 进行日志可视化。Loki 高效地收集和存储日志数据，非常适合聚合 Kubernetes 事件。Grafana 提供了一个用户友好的平台，可以使用收集的日志数据创建富有洞察力的仪表板和可视化效果。通过将这些工具与 Kubernetes Event Exporter 集成，您可以实现 Kubernetes 集群的全面可观察性、监控和警报，从而实现数据驱动的决策以及对应用程序和基础设施的高效管理。

Loki Deployment

首先，我们将使用 Helm 将 Loki 部署为轻量级单一二进制文件。第一步，我们创建一个 values 文件并将其命名为 loki-values.yaml。此外，值得一提的是，我已禁用身份验证，因为 Grafana 无法在启用身份验证的情况下建立连接。此外，我在以单一二进制模式部署 Helm Chart 时遇到了问题。错误消息“loki-memberlist无法解析”提示我们添加属性“memberlist.service.publishNotReadyAddresses: true”来阻止它。

```
loki:
  commonConfig:
    replication_factor: 1
  storage:
    type: 'filesystem'
  auth_enabled: false
singleBinary:
  replicas: 1

persistence:
  enabled: true
  accessModes:
    - ReadWriteOnce
  size: 50Gi
  annotations: {}

memberlist:
  service:
    publishNotReadyAddresses: true
```

要部署 Helm chart，请将 Grafana 的 chart 存储库添加到 Helm：

```
helm repo add grafana https://grafana.github.io/helm-charts
```

之后，我们运行命令来部署 **Loki**，在我们的例子中，它将位于“**monitoring**”命名空间中。我用调试标志来确保所有配置都是完美的

```
helm install loki grafana/loki -n monitoring -f loki-values.yaml --debug
```

Kubernetes Event Exporter

部署 **Loki** 后，我们将继续部署 **Kubernetes** 事件导出器。就像 **Loki** 一样，我们将利用此存储库中的 **Helm chart** 以及名为 **value.yaml** 的自定义值文件。

```

replicaCount: 1

image:
  repository: ghcr.io/resmoio/kubernetes-event-exporter
  pullPolicy: IfNotPresent
  # Overrides the image tag whose default is the chart appVersion.
  tag: latest

imagePullSecrets: []
nameOverride: ""
fullnameOverride: ""

serviceAccount:
  # Specifies whether a service account should be created
  create: true
  # Annotations to add to the service account
  annotations: {}
  # The name of the service account to use.
  # If not set and create is true, a name is generated using the fullname template
  name: ""

service:
  type: ClusterIP
  port: 2112

## Override the deployment namespace
namespaceOverride: "monitoring"

rbac:
  # If true, create & use RBAC resources
  create: true

resources:
  # We usually recommend not to specify default resources and to leave this as a conscious
  # choice for the user. This also increases chances charts run on environments with little
  # resources, such as Minikube. If you do want to specify resources, uncomment the following
  # lines, adjust them as necessary, and remove the curly braces after 'resources:'.
  limits:
    cpu: 100m
    memory: 128Mi
  requests:
    cpu: 100m
    memory: 128Mi

nodeSelector:
  kubernetes.io/os: linux

affinity: {}

```

除了这个文件之外，您还需要包含“config”属性，我将在下面描述。在“config”属性中，您可以指定事件导出器数据的目标。在我们的例子中，我们将使用 Loki，但您可以集成文档中列出的许多其他应用程序。由于 Loki 没有直接集成，因此我们将

其用作简单的 Webhook，向 Loki API 发出请求。此外，我们将格式化我们的请求以显示全面的事件数据。接收器被列为“dump”，以便使用命令 `kubectl logs` 通过命令行轻松访问和读取日志。

```
config: |
  logLevel: debug
  logFormat: json
  route:
    routes:
      - match:
          - receiver: "dump"
          - receiver: "loki"
      - drop:
          - namespace: "monitoring"
  receivers:
    - name: "dump"
      stdout: {}
    - name: "loki"
      webhook:
        endpoint: "http://loki.monitoring.svc.cluster.local:3100/loki/api/v1/push"
        headers:
          Content-Type: application/json
          User-Agent: "kube-event-exporter"
  layout:
    streams:
      - stream:
          app: kube-api
          source: event-exporter
          severity: "{{ .Type }}"
          reason: "{{ .Reason }}"
          name: "{{ .InvolvedObject.Name }}"
          namespace: "{{ .InvolvedObject.Namespace }}"
          kind: "{{ .InvolvedObject.Kind }}"
          host: "{{ .Source.Host }}"
          component: "{{ .Source.Component }}"
        values:
          - - "{{ mul .GetTimestampMs 1000000 }}"
            - "severity={{ .Type }} namespace={{ .InvolvedObject.Namespace }} object={{ .InvolvedObject.Kind }}"
```

就像在 Loki 部署中一样，我们添加存储库：

```
helm repo add kubernetes-event-exporter https://resmoio.github.io/kubernetes-event-exporter
```

并在“monitoring”命名空间中使用我们的values.yaml文件进行部署：

```
helm install kubernetes-event-exporter -f values.yaml . -n monitoring --debug
```


使用 Grafana 进行数据可视化

最后，让我们确保一切正常运行。

```
> kubectl get pods -n monitoring
```

NAME	READY	STATUS	RESTARTS
kubernetes-event-exporter-cd9dc4bbb-mnf6d	1/1	Running	0
loki-0	1/1	Running	0
loki-canary-fqmtr	1/1	Running	0
loki-gateway-56b5df76-wthkr	1/1	Running	0
loki-grafana-agent-operator-5798474874-5crtt	1/1	Running	0
loki-logs-gj8dl	2/2	Running	0

现在，我们可以方便地在 Grafana 中可视化我们的日志。为了实现这一目标，我们将 Loki 添加为 Grafana 中的数据源。



Data Sources / Loki

Type: Loki

Settings

Alerting supported

Name ⓘ

Loki

Default

☐

HTTP

URL ⓘ

http://loki:3100

Allowed cookies ⓘ

New tag (enter key to add)

Add

Timeout ⓘ

Timeout in seconds

Auth

Basic auth

☐

With Credentials ⓘ

☐

TLS Client Auth

☐

With CA Cert ⓘ

☐

Skip TLS Verify

☐

Forward OAuth Identity ⓘ

☐


Custom HTTP Headers

+ Add header

Alerting

Manage alerts via Alerting UI

☒

 云原生拓展

使用查询 `{source="event-exporter"} |= ""` 探索日志，以从事件导出器检索所有日志。

Explore

Kick start your query

Explain ⓘ

Label browser

Options

Type: Range

+ Add query

Query history

Inspector

Logs volume

Logs

Time

Unique labels

Wrap lines

Prettify JSON

DeDup

None

Exact

Numbers

Signature

Display results

Newest first

Oldest first

Common labels: kube-apis, event-exporter, Line limit: 1000 (52 returned), Total bytes processed: 11.1 kB

> 2023-06-30 15:35:49 severity:Normal namespace= object:Application/ reason=ResourceUpdated message=updated health status: Progressing -> Healthy


> 2023-06-30 15:35:40 severity:Normal namespace= object:Application/ reason=ResourceUpdated message=updated health status: Healthy -> Progressing

> 2023-06-30 10:53:09 severity:Normal namespace= object:Application/ reason=ResourceUpdated message=updated health status: Progressing -> Healthy

> 2023-06-30 10:52:50 severity:Normal namespace= object:Application/ reason=ResourceUpdated message=updated health status: Healthy -> Progressing

Download logs

Start of range

 云原生拓展

总结

总之，本教程演示了收集 Kubernetes 事件并在 Grafana 仪表板上可视化它们的过程。通过利用Kubernetes Event Exporter 和Grafana Loki，实现了事件监控和可视化。本教程还概述了使用 Helm 部署 Loki 和 Kubernetes 事件导出器的步骤。通过这些工具的集成，Kubernetes集群可以实现全面的可观察、监控和报警。