53Kubernetes 系列（五十）通过 Sidecar 容器使用 Logstash 和 FluentD 采集日志

Kubernetes 系列（五十）Kubernetes 练习 — 通过 Sidecar 容器使用 Logstash 和 FluentD 采集日志

我们将学习如何使用 Sidecar 容器模式在Kubernetes上安装Logstash和FluentD以进行日志聚合。

对于任何系统，日志聚合都非常重要。当您使用Kubernetes运行应用程序时，日志只属于一个Pod。如果该Pod被删除，日志也会丢失。

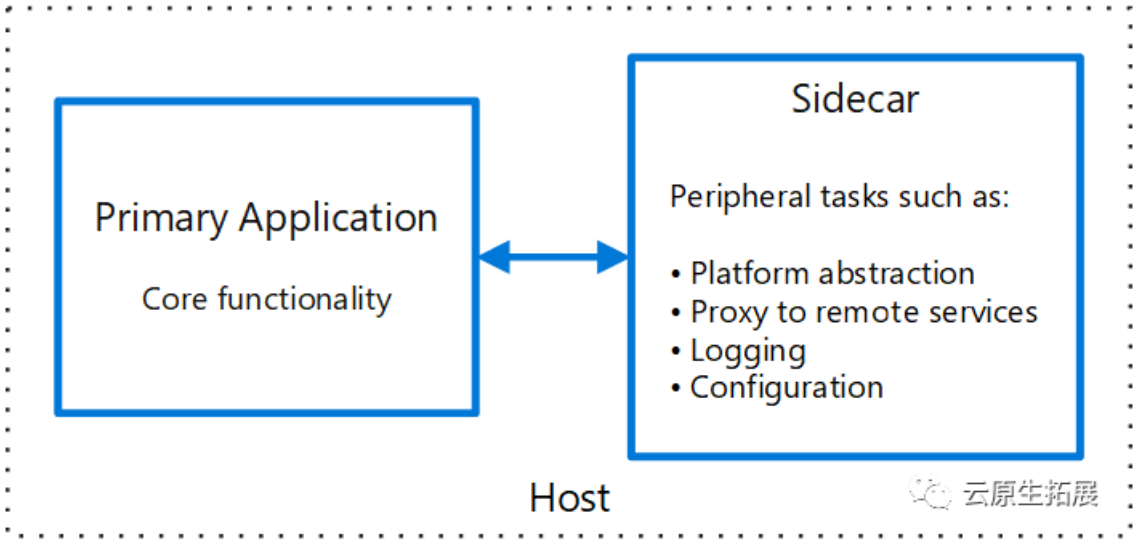
因此，如果我们想要跟踪系统故障，我们必须有一个 **日志聚合** 系统。现在，两个流行的日志栈是ELK (Elasticsearch Logstash Kibana)和EFK (Elasticsearch FluentD Kibana)。事实上还有另外一个 Loki 也不错，选择哪一个就看各人的判断了。

为了收集每个Pod上的日志，我们使用Sidecar Container。

Sidecar 容器

我们不需要在应用程序容器上实现日志收集过程，而是可以将该过程分离到另一个容器中，以避免影响应用程序容器的性能。这个容器叫做Sidecar container。

Sidecar 容器就是一个普通容器，只不过与主容器运行在同一个 Pod 中。这个 Sidecar 容器在某种程度上扩展和增强了应用程序容器。

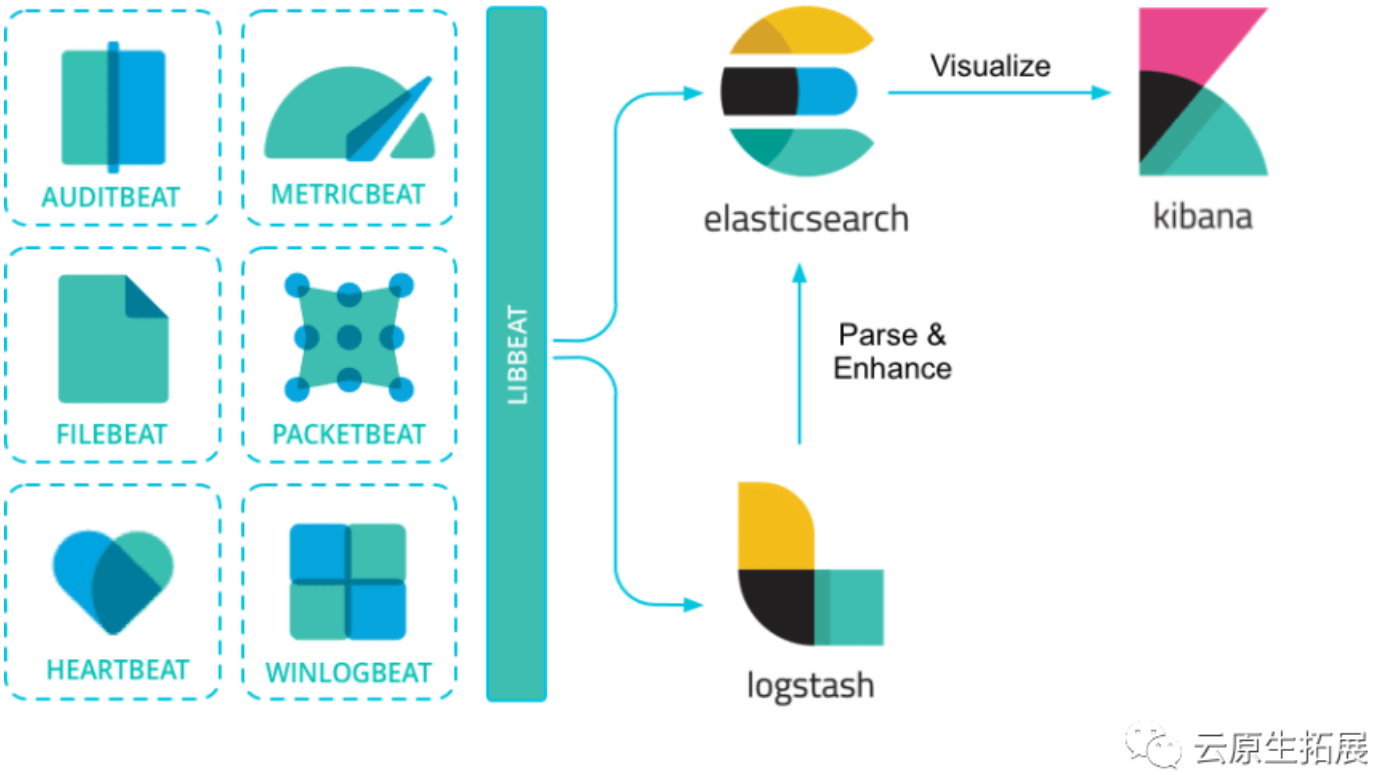


通过 Logstash 收集日志

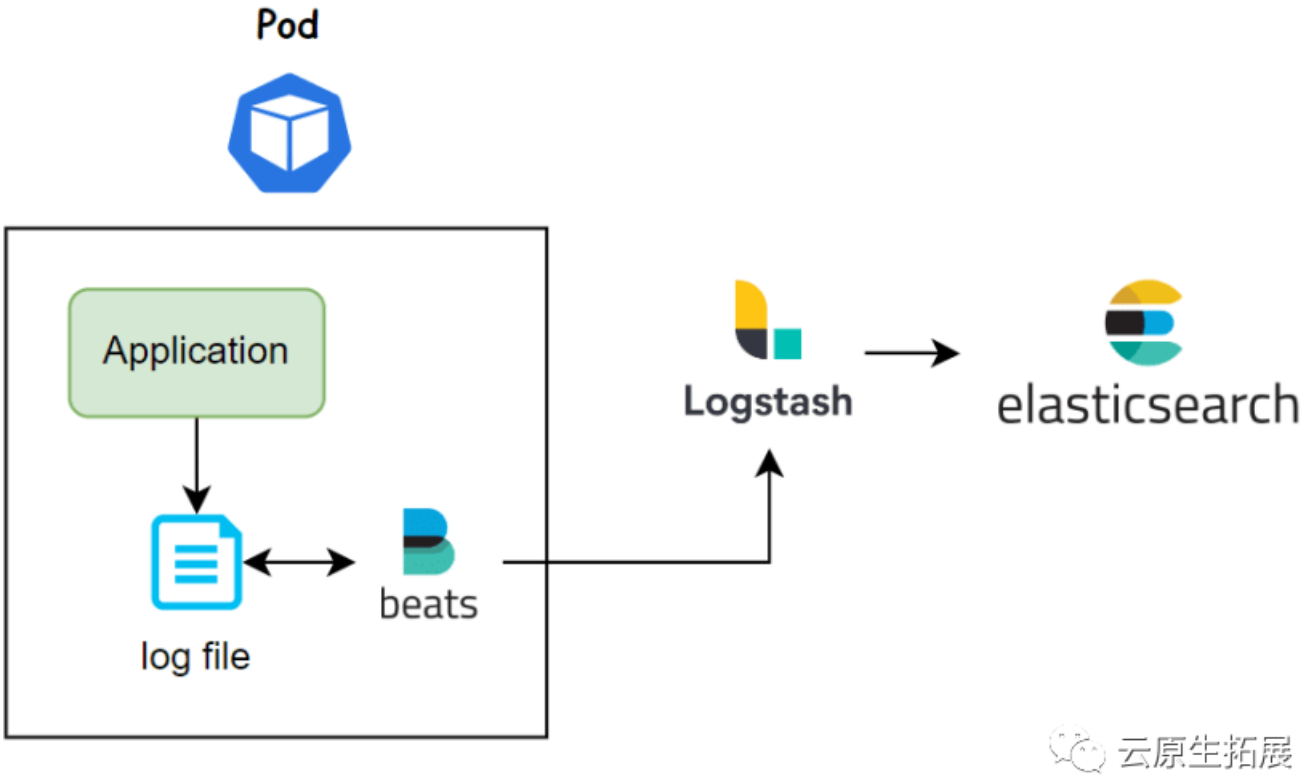
`Logstash` 最初的任务是监视日志并将它们转换为有意义的字段集，最终将输出流传输到定义的目的地。然而，它在性能方面存在问题。

因此，Elastic推出了Filebeat，用于监视日志并将输出流传输到定义的目的地。

Logstash充当的是一个聚合器，它从多种来源吸收数据，对其进行转换，然后将其发送到您最喜欢的“stash”。



理论讲完了，我们开始工作吧。首先，我们部署一个带应用程序容器的Pod，该容器将日志写入文件 `/var/log/access.log` ，然后在 Pod 中运行 Filebeat sidecar容器，以收集日志并将日志输出到Logstash。



创建文件 `filebeat.cm.yaml` 用于存储 Filebeat 配置信息.

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: filebeat-config
  labels:
    component: filebeat
data:
  conf.yaml: |
    filebeat.inputs:
    - type: log
      paths:
        - '/var/log/*.log'
    output:
      logstash:
        hosts: [ "logstash:5044" ]
```

我们配置了 filebeat 采集日志的路径：`/var/log/*.log`，然后输出到 Logstash.

创建 `application.yaml`.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: busybox
  labels:
    component: busybox
spec:
  strategy:
    type: Recreate
  selector:
    matchLabels:
      component: busybox
  template:
    metadata:
      labels:
        component: busybox
    spec:
      containers:
        - name: busybox
          image: busybox
          args:
            - sh
            - -c
            - >
              while true;
              do
                echo $(date) - filebeat log >> /var/log/access.log;
                sleep 10;
              done
          volumeMounts:
            - name: log
              mountPath: /var/log
        - name: filebeat
          image: elastic/filebeat:7.16.3
          args:
            - -c
            - /etc/filebeat/conf.yaml
            - -e
          volumeMounts:
            - name: filebeat-config
              mountPath: /etc/filebeat
            - name: log
              mountPath: /var/log
      volumes:
        - name: log
          emptyDir: {}
        - name: filebeat-config
          configMap:
            name: filebeat-config
```

在上面的Pod中，我们将Filebeat配置文件挂载到 `/etc/filebeat/conf.yaml` 文件，并使用args为Filebeat指定配置文件。

我们的应用程序容器每隔10秒向 `/var/log/access.log` 文件写入一个日志。我们使用emptyDir卷在两个容器之间共享存储。

下面,我们创建文件 `logstash.cm.yaml` 存储 Logstash 配置.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: logstash
  labels:
    component: logstash
data:
  access-log.conf: |
    input {
      beats {
        port => "5044"
      }
    }
    output {
      elasticsearch {
        hosts => [ "elasticsearch:9200" ]
      }
    }
```

创建 Logstash Deployment 文件 `logstash.yaml` .

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: logstash
  labels:
    component: logstash
spec:
  strategy:
    type: Recreate
  selector:
    matchLabels:
      component: logstash
  template:
    metadata:
      labels:
        component: logstash
    spec:
      containers:
        - name: logstash
          image: logstash:7.16.3
          ports:
            - containerPort: 5044
          volumeMounts:
            - name: logstash-config
              mountPath: /usr/share/logstash/pipeline
      volumes:
        - name: logstash-config
          configMap:
            name: logstash
---
apiVersion: v1
kind: Service
metadata:
  name: logstash
  labels:
    component: logstash
spec:
  ports:
  - port: 5044
  selector:
    component: logstash
```

我们将配置文件挂载到 `/usr/share/logstash/pipeline` 文件夹，Logstash将从这个文件夹加载配置文件。

创建 Elastichsearch (测试用).

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: elasticsearch
  labels:
    component: elasticsearch
spec:
  strategy:
    type: Recreate
  selector:
    matchLabels:
      component: elasticsearch
  template:
    metadata:
      labels:
        component: elasticsearch
    spec:
      containers:
        - name: elasticsearch
          image: elasticsearch:7.16.3
          ports:
            - containerPort: 9200
              name: client
            - containerPort: 9300
              name: nodes
          env:
            - name: JAVA_TOOL_OPTIONS
              value: -Xmx256m -Xms256m
            - name: discovery.type
              value: single-node
          resources:
            requests:
              memory: 500Mi
              cpu: 0.5
            limits:
              memory: 500Mi
              cpu: 0.5
---
apiVersion: v1
kind: Service
metadata:
  name: elasticsearch
  labels:
    component: elasticsearch
spec:
  ports:
  - port: 9200
    name: client
  - port: 9300
    name: nodes
  selector:
    component: elasticsearch
```

以及 Kibana.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: kibana
  labels:
    component: kibana
spec:
  strategy:
    type: Recreate
  selector:
    matchLabels:
      component: kibana
  template:
    metadata:
      labels:
        component: kibana
    spec:
      containers:
        - name: kibana
          image: kibana:7.16.3
          ports:
            - containerPort: 5601
---
apiVersion: v1
kind: Service
metadata:
  name: kibana
  labels:
    component: kibana
spec:
  ports:
  - port: 5601
  selector:
    component: kibana
```

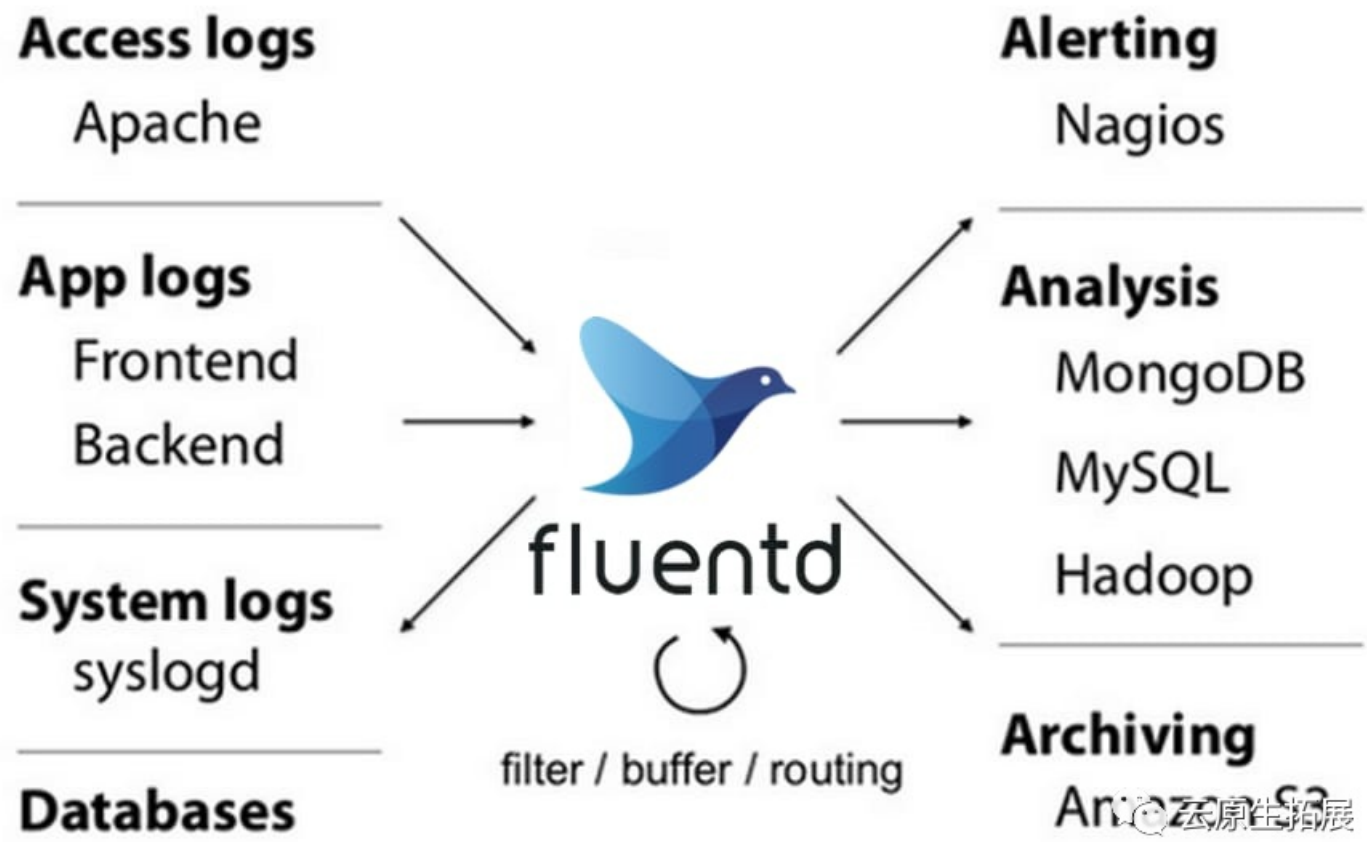
运行 apply 命令创建以上资源。

使用端口转发或者 ingress 来访问 Kibana Dashboard。

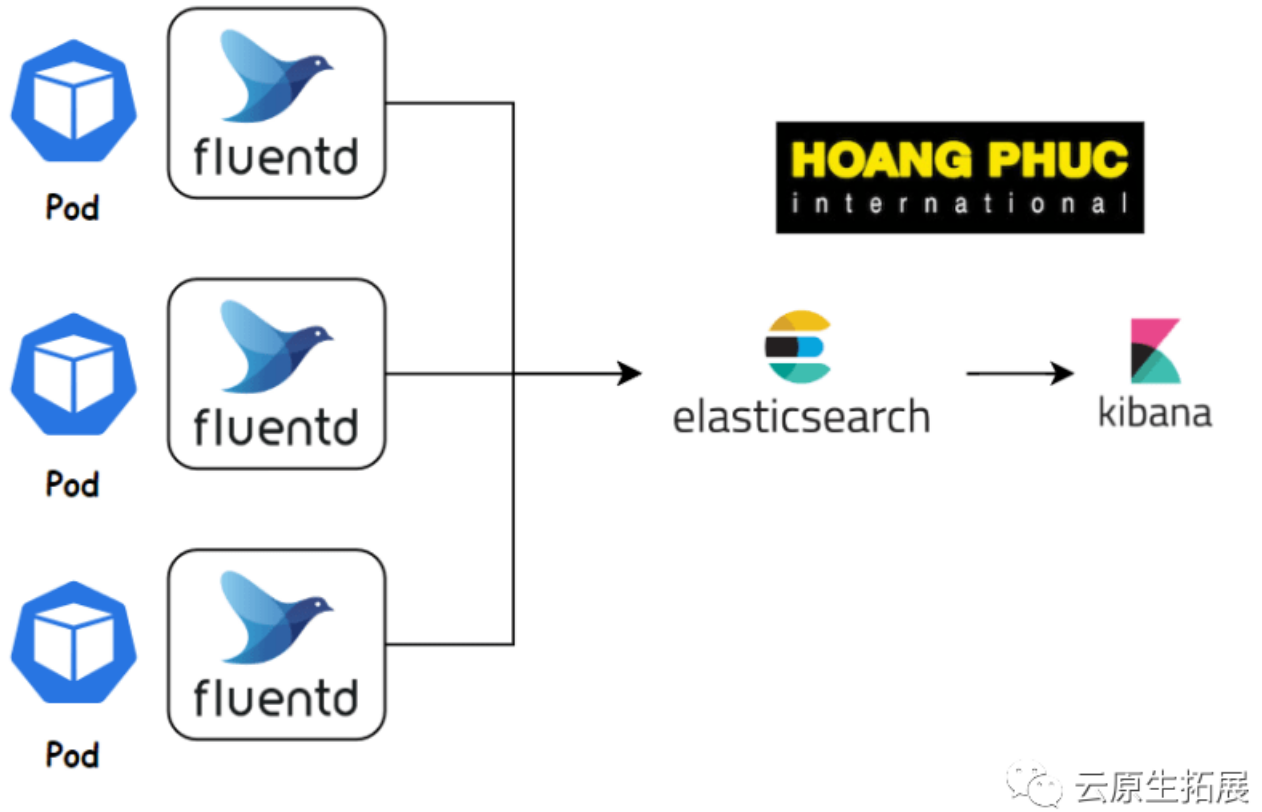
现在，转到菜单Stack Management >索引模式并创建一个索引模式，然后转到菜单Discover，您将看到我们从busybox容器收集的日志。

通过 FluentD 采集日志

FluentD也是一个日志收集工具，如Filebeat和Logstash。它是一个开源数据收集器，它允许您统一数据收集和消费，以便更好地使用和理解数据。



我们可以将它作为 Sidecar 容器从 pod 采集日志。



创建文件 `fluentd.cm.yaml` 存储 Filebeat 配置信息.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: fluentd-config
  labels:
    component: fluentd
data:
  fluent.conf: |
    <source>
      @type tail
      path /var/log/access.log
      pos_file /tmp/app.logs.pos
      tag app.logs
      <parse>
        @type none
      </parse>
    </source>
    <match app.logs>
      @type elasticsearch
      host elasticsearch
      port 9200
      logstash_format true
      logstash_prefix fluentd
      flush_interval 1s
    </match>
```

我们使用 `<source>` 标签来指定从何处采集日志，然后使用 `<match>` 标签将日志输出到 ES。

接下来，创建文件 `application.yaml` .

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: busybox
  labels:
    component: busybox
spec:
  strategy:
    type: Recreate
  selector:
    matchLabels:
      component: busybox
  template:
    metadata:
      labels:
        component: busybox
    spec:
      containers:
        - name: busybox
          image: busybox
          args:
            - sh
            - -c
            - >
              while true;
              do
                echo $(date) - filebeat log >> /var/log/access.log;
                sleep 10;
              done
          volumeMounts:
            - name: log
              mountPath: /var/log
        - name: fluentd
          image: govtechsg/fluentd-elasticsearch
          volumeMounts:
            - name: fluentd-config
              mountPath: /fluentd/etc
            - name: log
              mountPath: /var/log
      volumes:
        - name: log
          emptyDir: {}
        - name: fluentd-config
          configMap:
            name: fluentd-config
```

运行 apply 命令创建以上资源。

Fluentd 插件

需要注意的是，为了将日志输出到 Elasticsearch，我们必须使用Fluentd Elasticsearch Plugin。

正如您在上面看到的，我们使用的是 `govtechsg/fluentd-elasticsearch` 容器，这个容器已经有了elasticsearch 插件。

如果使用 `fluent/fluentd` 容器，它将给出一个 `@type elasticsearch` 无法找到的错误。

我们可以通过下面的 Dockerfile 来安装插件、构建新镜像：

```
FROM fluent/fluentd:v1.12.0-debian-1.0
USER root
RUN gem install fluent-plugin-elasticsearch --version 5.0.3
USER fluent
```

完整的 Fluentd 插件参考：https://www.fluentd.org/plugins/all

真实场景案例

也许你想知道为什么用Sidecar代替DaemonSet来处理日志?

这将取决于具体情况。在某些情况下，您无法在某些没有特权权限的工作节点上运行DaemonSet。例如，AWS EKS Fargate Pod。AWS Fargate是一种为容器提供按需、适当大小计算能力的技术。使用AWS Fargate，您不必自行调配、配置或扩展虚拟机组来运行容器。Amazon EKS通过使用AWS构建的控制器将Kubernetes与AWS Fargate集成。这些控制器作为AmazonEKS管理的Kubernetes控制平面的一部分运行，并负责将本地Kubernetes pod调度到Fargate上。 **您无法在Fargate上运行DaemonSet** 。这就是为什么这里应该使用Sidecar Container。

总结

至此，我们已经学习了如何使用 Sidecar Container 模式为 Pod 配置日志采集。ELK和EFK是两个非常流行的日志堆栈。

感谢阅读!

欢迎关注我的公众号“ **云原生拓展** ”，原创技术文章第一时间推送。