基于K8S的

iLogtail/Filebeat+KAFKA+Vector+ClickHouse/ES大规模日志采集最佳实践

本付费合集至少包括以下内容

- 基于Vector + ClickHouse的请求日志采集与Grafana日志分析看板实战
- 基于ClickHouse的请求日志,URI级别的5XX与耗时异常监控告警实践
- 基于K8S的iLogtail/Filebeat + KAFKA + Vector + ClickHouse/ES大规模日志采集方案
 - 基于K8S的KRaft模式KAFKA集群部署方案
 - 。 基于K8S的ClickHouse集群部署方案
 - 。 基于K8S的ES集群部署方案
- VIP微信群:提供运维交流答疑,原创运维技术文章优先本集合内发布,合集内文章疑难解答与远程协助。

为什么要创建付费合集

- 付费合集中包含的许多内容是独家提供的,只有付费用户才能享受到这些独特的资源和信息。
- 付费合集汇集了系统化的专题内容,读者可以更全面地掌握某一领域的知识,实现更深层次的学习和理解。
- 付费不仅是对内容的认可,也是对创作者辛勤付出的支持,帮助他们持续创作更多优质内容!

目录

基于K8S的iLogtail/Filebeat+KAFKA+Vector+ClickHouse/ES大规模日志采集最佳实践

```
本付费合集至少包括以下内容
为什么要创建付费合集
目录
整体架构
采集源
采集组件
  iLogtail
  Filebeat
  Vector
消息组件: KAFKA
  什么情况下需要使用KAFKA?
  使用KAFKA的好处
日志聚合处理与入库
  Vector
  iLogtail
存储数据库
  ElasticSearch
  ClickHouse
展示分析
   Kibana
   ClickVisual
   Grafana
```

▽【采集器】部署与日志的采集配置

iLogtail: DaemonSet部署

iLogtail: 采集配置

采集容器标准输出并写入KAFKA 采集容器内文件日志并写入KAFKA

Filebeat: DaemonSet部署

Filebeat: 采集配置

采集容器文件日志和标准输出并写入KAFKA

♥【消息组件】KAFKA部署与配置

Vector: Deployment部署 Vector: 日志处理与入库配置

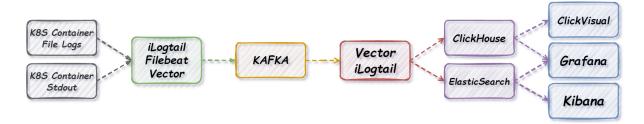
消费KAFKA处理日志并入库配置

iLogtail: 日志处理与入库配置

采集容器标准输出,处理JAVA多行日志并直接写入ES8

♥【数据库】ClickHouse与ES部署

整体架构



采集源

- K8S容器内文件日志
- K8S容器标准输出

采集组件

iLogtail

- 部署方式Daemonset
- 支持采集任意K8S容器内文件日志,提取K8S元信息
- 支持采集容器标准输出
- 不支持写入ES7.X
- 占用CPU比Filebeat大幅降低,占用内存与Filebeat相当

Filebeat

- 部署方式Daemonset
- 支持采集K8S容器内在 emptyDir 目录下的文件日志, 提取K8S元信息
- 支持采集容器标准输出

Vector

- 部署方式Daemonset
- 支持采集容器标准输出

消息组件: KAFKA

什么情况下需要使用KAFKA?

- 日志采集组件、日志处理组件出现使用CPU、内存异常升高
- 日志入库的延迟过长。
- 出现大量入库失败的情况。

使用KAFKA的好处

- 高吞吐量与低延迟: 支持高并发写入和读取, 实现实时数据处理和分析。
- 暂存机制:在数据入库失败时,可以配置系统从KAFKA重新消费这些数据。
- 采集和入库解耦:生产者只需将日志发送到KAFKA,不需要关心谁来消费或消费的速度。消费者可以独立地从KAFKA读取数据,处理和扩展变得更加灵活。
- 削峰填谷:即使在日志生成高峰期,KAFKA也能暂时存储大量数据,防止下游系统因处理能力不足 而崩溃。消费者可以以自己的节奏处理数据,KAFKA通过持久化存储和批量传输来平滑数据流,避 免系统在高负载时过载。

日志聚合处理与入库

Vector

- Rust
- 支持IP地址库解析, UserAgent解析
- 强大的数据处理函数、智能均衡KAFKA分区消费与入库, 官测正则处理13.2M/s
- 比logstash强悍太多的性能,声称比同类方案快10倍

iLogtail

- C++ & Golang
- 不支持IP地址库解析,不支持UserAgent解析
- 原生插件支持C++加速,宣称正则处理20M/s(原生插件仅可用于采集文本日志),其它类型可使用扩展插件处理,但性能会有一定影响。

存储数据库

ElasticSearch

全文检索引擎的文档数据库,对于**业务日志、异常日志、多行日志**这类,**非结构化**、半结构化的日志数据,经常需要做关键字查询,**模糊匹配**等操作,非常适合使用ES,使用倒排索引实现快速**全文搜索**。存储压缩率低,CPU/内存占用高。聚合分析性能没有ClickHouse好。

ClickHouse

一个列式存储数据库,尤其擅长处理**结构化**的大规模的SQL查询和聚合分析操作,所以针对NGINX这类结构化的**请求日志**,在处理**多维分析、聚合查询、分组统计**等操作速度极快,并且压缩比极高,存储成本比ES低10倍,CPU、内存的占用也有巨大优势。全文搜索性能没有ES好。

展示分析

Kibana

- 支持ES, 日志查询分析方便易用
- 查询页的分组统计默认只针对前500行(不能针对所有行)
- 可查询指定行的上下文

ClickVisual

- 支持ClickHouse, 类似Kibana界面, 没有Kibana好用
- 查询页的分组统计针对所有行
- 不可查询指定行的上下文

Grafana

• 支持ClickHouse与ES,数据分析展示最佳,日志查询界面没有Kibana好用。

▽【采集器】部署与日志的采集配置

iLogtail: DaemonSet部署

```
1 apiversion: apps/v1
 2
    kind: DaemonSet
 3 metadata:
      name: ilogtail-ds
 5
      namespace: ops-monit
 6
      labels:
 7
        k8s-app: logtail-ds
 8
   spec:
9
      selector:
10
        matchLabels:
11
          k8s-app: logtail-ds
12
      template:
13
        metadata:
          labels:
14
15
            k8s-app: logtail-ds
16
        spec:
17
         tolerations:
18
            - operator: Exists
19
          containers:
20
            - name: logtail
21
              command:
               - /usr/local/ilogtail/ilogtail_control.sh
22
23
              args:
24
              - "start_and_block"
25
               - "-enable_containerd_upper_dir_detect=true"
               - "-dirfile_check_interval_ms=5000"
```

```
- "-logtail_checkpoint_check_gc_interval_sec=120"
27
28
               env:
29
                 - name: _node_name_
                   valueFrom:
30
                     fieldRef:
31
32
                       apiversion: v1
33
                       fieldPath: spec.nodeName
34
                 - name: _node_ip_
                   valueFrom:
35
36
                     fieldRef:
                       apiversion: v1
37
                       fieldPath: status.hostIP
38
39
                 - name: cpu_usage_limit
40
                   value: "1"
                 - name: mem_usage_limit
41
                   value: "512"
42
43
               image: >-
44
                 sls-opensource-registry.cn-shanghai.cr.aliyuncs.com/ilogtail-
    community-edition/ilogtail:latest
45
               imagePullPolicy: IfNotPresent
46
               resources:
47
                 limits:
                   cpu: 1000m
48
49
                   memory: 1Gi
50
                 requests:
51
                   cpu: 40m
52
                   memory: 38Mi
53
               volumeMounts:
54
                 - mountPath: /var/run
55
                   name: run
                 - mountPath: /logtail_host
56
57
                   mountPropagation: HostToContainer
58
                   name: root
                   readOnly: true
59
60
                 - mountPath: /usr/local/ilogtail/checkpoint
61
                   name: checkpoint
62
                 - mountPath: /usr/local/ilogtail/config/local
63
                   name: user-config
                   readOnly: true
64
          dnsPolicy: ClusterFirstWithHostNet
65
66
          hostNetwork: true
67
          volumes:
             - hostPath:
68
                 path: /var/run
69
70
                 type: Directory
71
               name: run
             - hostPath:
72
73
                 path: /
74
                 type: Directory
75
               name: root
76
             - hostPath:
                 path: /etc/ilogtail-ilogtail-ds/checkpoint
77
78
                 type: DirectoryOrCreate
79
               name: checkpoint
             - configMap:
80
                 defaultMode: 420
81
```

```
name: ilogtail-user-cm
name: user-config
```

iLogtail: 采集配置

采集容器标准输出并写入KAFKA

根据微服务命名空间与标签来匹配微服务

```
apiversion: v1
 2
    kind: ConfigMap
 3
    metadata:
      name: ilogtail-user-cm
 4
 5
      namespace: ops-monit
 6
    data:
 7
      nginx_stdout.yaml: |
 8
        enable: true
 9
        inputs:
10
          - Type: service_docker_stdout # 采集K8S容器标准输出
11
            Stdout: true
            Stderr: false
12
            K8sNamespaceRegex: "^istio-system$" # 指定命名空间
13
14
            IncludeK8sLabel:
15
              app: istio-ingressgateway # 指定需要匹配的标签
16
        flushers:
17
          - Type: flusher_kafka_v2
18
            Brokers:
              - ops-log1.kafka.casstime.com:9092
19
              - ops-log2.kafka.casstime.com:9092
20
21
              ops-log3.kafka.casstime.com:9092
            Topic: prod-istio-std
22
            MaxMessageBytes: 10000000
23
```

采集容器内文件日志并写入KAFKA

根据文件在容器内的路径来匹配

```
1
   #接上面的配置,支持在一个configmap中有多个配置文件。
2
     java_file.yaml: |
3
       enable: true
4
       inputs:
 5
         - Type: input_file
 6
           FilePaths:
7
             - /opt/logs/*.log # 指定容器内的日志路径
8
           EnableContainerDiscovery: true #表示采集容器内的文件日志
9
           ExcludeFiles:
             - access.* # 需要排除的文件名
10
11
           ContainerFilters:
12
             ExcludeK8sLabel:
13
               app: pod-agentbuy-service # 可排除包含某些标签的容器
14
           Multiline:
15
             StartPattern: \d+-\d+-\d+.* # 匹配多行日志开头的正则
16
           AppendingLogPositionMeta: true # 添加采集文件的元信息
17
       processors:
18
         - Type: processor_add_fields # 增加字段
```

```
19
            Fields:
20
              logtype: java-file
21
              env: cassmall
22
        flushers:
          - Type: flusher_kafka_v2
23
24
            Brokers:
25
              - cassops-log1.kafka.casstime.com:9092
              - cassops-log2.kafka.casstime.com:9092
26
              - cassops-log3.kafka.casstime.com:9092
27
28
            Topic: prod-java-file-ilogtail
29
            MaxMessageBytes: 10000000
            Convert:
30
              TagFieldsRename: # 对K8S元数据的字段名重命名
31
32
                k8s.pod.name: 'pod_name'
33
                k8s.node.name: 'node_name'
                k8s.namespace.name: 'namespace'
34
                container.image.name: 'deployimage'
35
36
                log.file.path: 'path'
```

Filebeat: DaemonSet部署

```
apiversion: v1
 1
 2
    kind: ServiceAccount
 3
    metadata:
     name: filebeat
 4
 5
     namespace: ops-monit
 6
 7
    apiversion: rbac.authorization.k8s.io/v1
 8
    kind: ClusterRole
9
    metadata:
10
     annotations:
      labels:
11
12
        k8s-app: filebeat
13
     name: filebeat
14
   rules:
15
   - apiGroups:
     _ ""
16
17
     resources:
18
     - namespaces
19
      - pods
20
     verbs:
21
     - get
22
      - watch
23
      - list
24
    apiVersion: rbac.authorization.k8s.io/v1
25
26
   kind: ClusterRoleBinding
27
    metadata:
28
     annotations:
29
      name: filebeat
30
   roleRef:
31
      apiGroup: rbac.authorization.k8s.io
32
      kind: ClusterRole
      name: filebeat
33
34
   subjects:
```

```
35
    - kind: ServiceAccount
      name: filebeat
36
37
      namespace: ops-monit
38
    apiversion: apps/v1
39
40
    kind: DaemonSet
41
    metadata:
42
      labels:
        k8s-app: filebeat
43
44
      name: filebeat
45
      namespace: ops-monit
46
    spec:
      selector:
47
48
        matchLabels:
49
          k8s-app: filebeat
50
      template:
51
        metadata:
52
          labels:
53
             k8s-app: filebeat
54
        spec:
55
          containers:
56
          - args:
57
             - -c
58
             - /etc/filebeat.yml
59
60
             env:
61
             - name: NODE_NAME
62
               valueFrom:
63
                 fieldRef:
                   apiversion: v1
64
                   fieldPath: spec.nodeName
65
66
             image: elastic/filebeat:7.17.24
67
             #image: elastic/filebeat:7.6.2
             imagePullPolicy: IfNotPresent
68
69
             name: filebeat
70
             resources:
71
               limits:
                 cpu: "1"
72
73
                 memory: 600Mi
74
               requests:
75
                 cpu: 100m
76
                 memory: 120Mi
77
             securityContext:
78
               runAsUser: 0
79
             terminationMessagePath: /dev/termination-log
80
             terminationMessagePolicy: File
             volumeMounts:
81
             - mountPath: /etc/filebeat.yml
82
83
               name: config
84
               readOnly: true
85
               subPath: filebeat.yml
             - mountPath: /usr/share/filebeat/data
86
87
               name: data
88
             - mountPath: /var/lib/docker/containers
89
               name: varlibdockercontainers
90
               readOnly: true
```

```
91
              - mountPath: /var/log
 92
                name: varlog
 93
                readOnly: true
 94
              - mountPath: /var/lib/kubelet/pods
                name: pod-test
 95
 96
                readOnly: true
 97
           dnsConfig:
 98
              options:
              - name: single-request-reopen
99
100
           dnsPolicy: ClusterFirstWithHostNet
           restartPolicy: Always
101
           securityContext: {}
102
103
           serviceAccount: filebeat
104
           serviceAccountName: filebeat
           tolerations:
105
106
           - operator: Exists
107
           volumes:
108
           - configMap:
                defaultMode: 416
109
                name: filebeat-config
110
111
              name: config
           - hostPath:
112
113
                path: /var/lib/docker/containers
                type: ""
114
115
              name: varlibdockercontainers
            - hostPath:
116
117
                path: /var/log
                type: ""
118
119
              name: varlog
120
           - hostPath:
                path: /var/lib/filebeat-data
121
122
                type: DirectoryOrCreate
123
              name: data
124
            - hostPath:
125
                path: /mnt/paas/kubernetes/kubelet/pods
126
                type: DirectoryOrCreate
127
              name: pod-test
```

Filebeat: 采集配置

采集容器文件日志和标准输出并写入KAFKA

注意容器内的文件日志必须存放在 emptyDir 目录下

```
apiversion: v1
2
    kind: ConfigMap
3
   metadata:
 4
      name: filebeat-config
 5
      namespace: ops-monit
6
      labels:
7
        k8s-app: filebeat
8
    data:
9
      filebeat.yml: |-
        filebeat.inputs:
10
11
        - type: log # 文件日志
```

```
12
                              paths:
13
                                     - /var/lib/kubelet/pods/*/volumes/kubernetes.io~empty-dir/**/*.log #
             默认empty-dir在节点主机的路径
                              exclude_files: ['.*access.*'] # 可以排除某些文件名
14
15
                               multiline.type: pattern # 多行日志匹配
                               \label{eq:multiline.pattern: $$ \d{4}-\d{2}-\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d{2}:\d
16
                               multiline.negate: true
17
                              multiline.match: after
18
                               fields: # 增加字段
19
20
                                    logtype: java-file
                                    env: f2b-gamma
21
22
                              processors:
                               - add_kubernetes_metadata: # 增加K8S元数据
23
24
                                          indexers:
25
                                                 - pod_uid:
26
                                          matchers:
27
                                                 - logs_path:
                                                            logs_path: '/var/lib/kubelet/pods/'
28
29
                                                             resource_type: 'pod'
30
31
                         - type: container # 采集容器标准输出
32
                              paths:
                                     - /var/log/containers/*_f2b_*node*.log # 默认标准输出在节点主机的路径
33
34
                               fields: # 增加字段
35
                                    logtype: json-std
                                    env: f2b-gamma
36
37
                               processors:
38
                               - add_kubernetes_metadata: # 增加K8S元数据
39
                                          matchers:
40
                                                 - logs_path:
41
                                                            logs_path: /var/log/containers
42
43
                         output.kafka: # 输出到KAFKA
                               hosts: ["10.1.230.152:9092", "10.1.227.35:9092", "10.1.247.50:9092"]
44
45
                               topic: 'topic-gamma-logs'
46
                               max_message_bytes: 10000000
47
                               partition.round_robin:
48
                                    reachable_only: false
```

♥【消息组件】KAFKA部署与配置

▽【处理器】部署与日志处理配置

Vector: Deployment部署

```
apiVersion: v1
kind: ServiceAccount
metadata:
namespace: ops-monit
name: vector
automountServiceAccountToken: true
---
apiVersion: rbac.authorization.k8s.io/v1
```

```
9 kind: ClusterRole
10
    metadata:
11
      name: vector
12
      labels:
13
        app.kubernetes.io/name: vector
14
    rules:
15
     - apiGroups:
          _ ""
16
17
       resources:
18
          - namespaces
19
          - nodes
20
          - pods
21
        verbs:
22
          - list
23
           - watch
24
    apiversion: rbac.authorization.k8s.io/v1
25
26
    kind: ClusterRoleBinding
27
    metadata:
28
      name: vector
29
      labels:
30
        app.kubernetes.io/name: vector
31
    roleRef:
      apiGroup: rbac.authorization.k8s.io
32
33
      kind: ClusterRole
34
      name: vector
    subjects:
35
36

    kind: ServiceAccount

37
        name: vector
38
        namespace: ops-monit
39
40
    apiversion: apps/v1
41
    kind: Deployment
42
    metadata:
43
      name: vector
44
      namespace: ops-monit
45
      labels:
46
        app.kubernetes.io/name: vector
47
    spec:
48
      replicas: 1
49
      selector:
50
        matchLabels:
51
          app.kubernetes.io/name: vector
52
      template:
53
        metadata:
54
          annotations: {}
55
          labels:
56
            app.kubernetes.io/name: vector
57
            vector.dev/exclude: "true"
58
        spec:
59
          serviceAccountName: vector
          dnsPolicy: ClusterFirst
60
61
          containers:
62
            - name: vector
               image: "registry.cn-shenzhen.aliyuncs.com/starsl/vector:0.42.0-
63
    alpine"
```

```
64
                imagePullPolicy: IfNotPresent
 65
                args:
 66
                  - --config-dir
                  - /etc/vector/
 67
 68
                env:
 69
                  - name: TZ
 70
                    value: Asia/Shanghai
 71
                ports:
 72
                  - name: api
 73
                    containerPort: 8686
 74
                    protocol: TCP
 75
                resources:
                  limits:
 76
 77
                    cpu: '2'
 78
                    memory: 2Gi
 79
                  requests:
 80
                    cpu: 500m
 81
                    memory: 100Mi
                volumeMounts:
 82
 83
                  - name: data
                    mountPath: "/vector-data-dir"
 85
                  - name: config
                    mountPath: "/etc/vector/"
 86
                    readOnly: true
 87
 88
            terminationGracePeriodSeconds: 60
 89
            affinity:
              podAntiAffinity:
 90
 91
                requiredDuringSchedulingIgnoredDuringExecution:
 92
                  - labelSelector:
 93
                      matchLabels:
                        app.kubernetes.io/name: vector
 94
 95
                    topologyKey: kubernetes.io/hostname
 96
            volumes:
 97
              - name: data
 98
                emptyDir: {}
99
              - name: config
                projected:
100
101
                  sources:
102
                    - configMap:
103
                        name: vector
104
105
     apiversion: v1
106
     kind: Service
107
     metadata:
108
       namespace: ops-monit
109
       name: vector
110
       labels:
111
         app.kubernetes.io/name: vector
112
     spec:
113
       ports:
114
         - name: api
115
           port: 8686
116
            protocol: TCP
117
       selector:
118
         app.kubernetes.io/name: vector
119
       type: NodePort
```

Vector: 日志处理与入库配置

消费KAFKA处理日志并入库配置

```
1
    apiversion: v1
 2
    kind: ConfigMap
 3
    metadata:
 4
      name: vector
 5
      namespace: ops-monit
 6
 7
      global.toml: |
        data_dir = "/vector-data-dir"
 8
 9
        timezone = "Asia/Shanghai"
10
        [api]
          address = "0.0.0.0:8686"
11
12
          enabled = true
13
14
      prod-java-file-ilogtail.toml: |- # java业务日志入库ES(toml格式配置文件写法)
        [sources.1kafka_java_ilogtail]
15
          type = "kafka"
16
17
          bootstrap_servers = "10.0.100.1:9092,10.0.100.2:9092,10.0.100.3:9092"
18
          group_id = "prod-java-file-ilogtail"
          topics = [ "prod-java-file-ilogtail" ]
19
          decoding.codec = "json"
20
21
          commit_interval_ms = 1000
22
23
        [transforms.2parse_java_all_ilogtail]
24
        inputs = ["1kafka_java_ilogtail"]
25
        type = "remap"
        source = '''
26
27
          .namespace = .tags.namespace
28
          .podname = .tags.pod_name
29
          .nodename = .tags.node_name
30
          .appname = .tags.appname
31
          .deployimage = .tags.deployimage
32
          # java日志使用正则解析,获取时间和日志级别字段即可,并保留原始日志.
33
          . |= parse_regex!(.fields.content, r'^{?P<time>d+-d+}
    d+:d+:d+,d+)(s|:)+(?P<level>s+)(s|s)*$'
        1.1.1
34
35
36
        [transforms.3delfields_java_ilogtail]
37
        inputs = ["2parse_java_all_ilogtail"]
        type = "remap"
38
        source = '''
39
40
        .unix_time, err = to_int(format_timestamp!(.timestamp, format: "%s"))
        .utc8\_time = .unix\_time + 28800
41
        .index_day = format_timestamp!(to_timestamp!(.utc8_time), "%Y.%m.%d")
42
43
        .path = .tags.path
44
        del(.unix_time)
45
        del(.utc8_time)
46
        del(.tags)
47
        .message = del(.fields.content)
48
        del(.topic)
49
        del(.offset)
```

```
50
         del(.partition)
 51
         del(.message_key)
 52
         del(.source_type)
         del(.fields.__file_offset__)
 53
 54
         .@timestamp = parse_timestamp(.time, format: "%F %T,%3f") ?? now()
 55
         .captime = del(.timestamp)
 56
         del(.time)
 57
         ns, err = string(.namespace)
 58
         if err != null {
 59
           .namespace = "null"
         }
 60
         . . .
 61
 62
 63
         [sinks.9es_java_ilogtail]
           type = "elasticsearch"
 64
 65
           inputs = ["3delfields_java_ilogtail"]
           endpoint = "http://10.1.72.33:19200"
 66
           auth.strategy = "basic"
 67
           auth.user = "elastic"
 68
           auth.password = "M1k2w12Ig4A"
 69
 70
           mode = "bulk"
 71
           buffer.max_events = 50000
 72
           batch.max\_events = 50000
 73
           batch.timeout_secs = 5
 74
           bulk.index = "hwprod-java-file-{{ namespace }}-{{ index_day }}"
 75
       istio.yaml: |- # istio请求日志入库ClickHouse(yaml格式配置文件写法)
 76
 77
         sources:
 78
           01_kafka_hwprod_istio:
 79
             type: "kafka"
             bootstrap_servers: "kafka1:9092,kafka2:9092,kafka3:9092"
 80
 81
             group_id: "hwprod_istio"
 82
             topics: [ "prod-istio-std" ]
             decoding:
 83
               codec: "json"
 84
 85
             commit_interval_ms: 1000
 86
 87
         transforms:
 88
           02_parse_hwprod_istio:
 89
             drop_on_error: true
 90
             reroute_dropped: true
 91
             type: remap
 92
             inputs:
 93
               - 01_kafka_hwprod_istio
 94
             source: |
 95
               # json日志直接解析json,即可获取所有字段.
 96
               . = parse_json!(.contents.content)
 97
               .start_time = to_unix_timestamp(parse_timestamp!(.start_time,
     format: "%+"),unit: "milliseconds")
 98
               .upstream_service_time = to_int!(.upstream_service_time)
 99
100
         sinks:
101
           03_ck_hwprod_istio:
102
             type: clickhouse
103
             inputs:
104
               - 02_parse_hwprod_istio
```

```
105
             database: istiologs
106
             endpoint: http://10.7.0.226:28123
107
             table: hwprod_istio_local
             compression: gzip
108
109
           04_out_istio_dropped:
110
             type: blackhole
111
             inputs:
112
               - 02_parse_hwprod_istio.dropped
```

iLogtail: 日志处理与入库配置

采集容器标准输出,处理JAVA多行日志并直接写入ES8

```
1 apiversion: v1
2
   kind: ConfigMap
3
   metadata:
4
      name: ilogtail-user-cm
5
      namespace: ops-monit
6
   data:
7
      java_std_es8.yaml: |
8
        enable: true
9
        inputs:
10
          - Type: service_docker_stdout
11
            Stderr: true
12
            Stdout: true
            BeginLineCheckLength: 10
13
            BeginLineRegex: "\\d+-\\d+-\\d+.*" # JAVA多行日志开头匹配正则
14
            K8sNamespaceRegex: "^(saas-prod)$"
15
16
        processors:
          - Type: processor_string_replace # 如果日志的标准输出有颜色,可以使用此正则替
17
    换掉
18
            SourceKey: content
            Method: regex
19
            Match: [\d^*\; ^\d^m] \setminus u001b
20
            ReplaceString: ''
21
22
          - Type: processor_regex # JAVA日志处理: 只从日志中提取时间和级别字段,并保留
    源。
23
            SourceKey: content
24
            Regex: (d_4}-d_2)-d_2 \ d_2:\ d_2,\ d_3) \ .*?[0-9a-z]*([A-y-2]).
    Z]+)
25
            Keys:
26
              - log_time
27
              - level
28
            KeepSource: true
          - Type: processor_gotime # 日志中的时间字段转成ES中@timestamp标准日期时间格
29
    式。
30
            SourceKey: "log_time"
            SourceFormat: "2006-01-02 15:04:05,000"
31
32
            SourceLocation: 8
33
            DestKey: "@timestamp"
            DestFormat: "2006-01-02T15:04:05.000Z"
34
            DestLocation: 0
35
36
            SetTime: false
37
            KeepSource: false
38
          - Type: processor_rename # 字段重命名
```

```
39
            SourceKeys:
40
              - _time_
41
              - _source_
42
            DestKeys:
43
              - cap_time
              - log_src
44
45
        flushers:
46
          #- Type: flusher_stdout
          # OnlyStdout: true
47
48
          - Type: flusher_elasticsearch
49
            Addresses:
50
              - http://10.26.6.152:9200
              - http://10.26.6.153:9200
51
52
              - http://10.26.6.154:9200
53
            Convert:
54
              Protocol: custom_single_flatten # 数据字段做一级打平
55
              Encoding: json
56
            Index: java-logs-%{tag.k8s.namespace.name}_%{+yyyy.MM.dd}
57
            Authentication:
58
              PlainText:
59
                Username: ilogtail1
60
                Password: fSdgDh4jfg
61
            HTTPConfig:
                MaxIdleConnsPerHost: 10
62
63
                ResponseHeaderTimeout: Second
```

♥【数据库】ClickHouse与ES部署