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

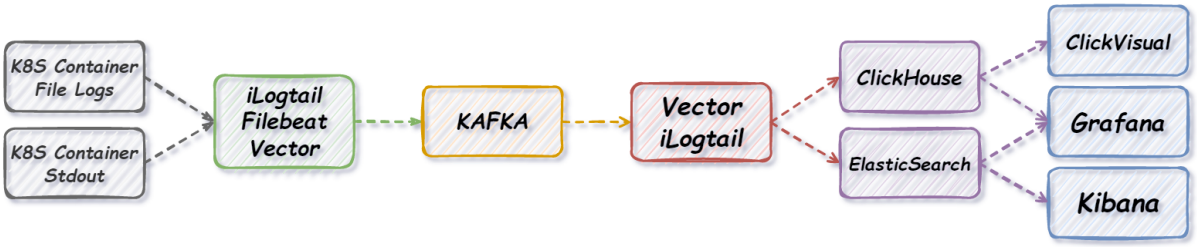
- 基于Vector + ClickHouse的请求日志采集与Grafana日志分析看板实战
- 基于ClickHouse的请求日志，URI级别的5XX与耗时异常监控告警实践
- 基于K8S的iLogtail/Filebeat + KAFKA + Vector + ClickHouse/ES大规模日志采集方案
  - 基于K8S的KRaft模式KAFKA集群部署方案
  - 基于K8S的ClickHouse集群部署方案
  - 基于K8S的ES集群部署方案
- **VIP微信群：提供运维交流答疑，原创运维技术文章优先本集合内发布，合集内文章疑难解答与远程协助。**

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

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

```yaml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: ilogtail-ds
  namespace: ops-monit
  labels:
    k8s-app: logtail-ds
spec:
  selector:
    matchLabels:
      k8s-app: logtail-ds
  template:
    metadata:
      labels:
        k8s-app: logtail-ds
    spec:
      tolerations:
        - operator: Exists
      containers:
        - name: logtail
          command:
          - /usr/local/ilogtail/ilogtail_control.sh
          args:
          - "start_and_block"
          - "-enable_containerd_upper_dir_detect=true"
          - "-dirfile_check_interval_ms=5000"
```

```
          - "-logtail_checkpoint_check_gc_interval_sec=120"
          env:
            - name: _node_name_
              valueFrom:
                fieldRef:
                  apiVersion: v1
                  fieldPath: spec.nodeName
            - name: _node_ip_
              valueFrom:
                fieldRef:
                  apiVersion: v1
                  fieldPath: status.hostIP
            - name: cpu_usage_limit
              value: "1"
            - name: mem_usage_limit
              value: "512"
          image: >-
            sls-opensource-registry.cn-shanghai.cr.aliyuncs.com/ilogtail-
community-edition/ilogtail:latest
          imagePullPolicy: IfNotPresent
          resources:
            limits:
              cpu: 1000m
              memory: 1Gi
            requests:
              cpu: 40m
              memory: 38Mi
          volumeMounts:
            - mountPath: /var/run
              name: run
            - mountPath: /logtail_host
              mountPropagation: HostToContainer
              name: root
              readOnly: true
            - mountPath: /usr/local/ilogtail/checkpoint
              name: checkpoint
            - mountPath: /usr/local/ilogtail/config/local
              name: user-config
              readOnly: true
      dnsPolicy: ClusterFirstWithHostNet
      hostNetwork: true
      volumes:
        - hostPath:
            path: /var/run
            type: Directory
          name: run
        - hostPath:
            path: /
            type: Directory
          name: root
        - hostPath:
            path: /etc/ilogtail-ilogtail-ds/checkpoint
            type: DirectoryOrCreate
          name: checkpoint
        - configMap:
            defaultMode: 420
```

```
            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
kind: ConfigMap
metadata:
  name: ilogtail-user-cm
  namespace: ops-monit
data:
  nginx_stdout.yaml: |
    enable: true
    inputs:
      - Type: service_docker_stdout  # 采集K8S容器标准输出
        Stdout: true
        Stderr: false
        K8sNamespaceRegex: "^istio-system$"  # 指定命名空间
        IncludeK8sLabel:
          app: istio-ingressgateway  # 指定需要匹配的标签
    flushers:
      - Type: flusher_kafka_v2
        Brokers:
          - ops-log1.kafka.casstime.com:9092
          - ops-log2.kafka.casstime.com:9092
          - ops-log3.kafka.casstime.com:9092
        Topic: prod-istio-std
        MaxMessageBytes: 10000000
```

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

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

```
        Fields:
          logtype: java-file
          env: cassmall
    flushers:
      - Type: flusher_kafka_v2
        Brokers:
          - cassops-log1.kafka.casstime.com:9092
          - cassops-log2.kafka.casstime.com:9092
          - cassops-log3.kafka.casstime.com:9092
        Topic:  prod-java-file-ilogtail
        MaxMessageBytes: 10000000
        Convert:
          TagFieldsRename:  # 对K8S元数据的字段名重命名
            k8s.pod.name: 'pod_name'
            k8s.node.name: 'node_name'
            k8s.namespace.name: 'namespace'
            container.image.name: 'deployimage'
            log.file.path: 'path'
```

Filebeat：DaemonSet部署

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: filebeat
  namespace: ops-monit
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  annotations:
  labels:
    k8s-app: filebeat
  name: filebeat
rules:
- apiGroups:
  - ""
  resources:
  - namespaces
  - pods
  verbs:
  - get
  - watch
  - list
apiVersion: rbac.authorization.k8s.io/v1
---
kind: ClusterRoleBinding
metadata:
  annotations:
  name: filebeat
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: filebeat
subjects:
```

```yaml
- kind: ServiceAccount
  name: filebeat
  namespace: ops-monit
---
apiVersion: apps/v1
kind: DaemonSet
metadata:
  labels:
    k8s-app: filebeat
  name: filebeat
  namespace: ops-monit
spec:
  selector:
    matchLabels:
      k8s-app: filebeat
  template:
    metadata:
      labels:
        k8s-app: filebeat
    spec:
      containers:
      - args:
        - -c
        - /etc/filebeat.yml
        - -e
        env:
        - name: NODE_NAME
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: spec.nodeName
        image: elastic/filebeat:7.17.24
        #image: elastic/filebeat:7.6.2
        imagePullPolicy: IfNotPresent
        name: filebeat
        resources:
          limits:
            cpu: "1"
            memory: 600Mi
          requests:
            cpu: 100m
            memory: 120Mi
        securityContext:
          runAsUser: 0
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
        volumeMounts:
        - mountPath: /etc/filebeat.yml
          name: config
          readOnly: true
          subPath: filebeat.yml
        - mountPath: /usr/share/filebeat/data
          name: data
        - mountPath: /var/lib/docker/containers
          name: varlibdockercontainers
          readOnly: true
```

```
        - mountPath: /var/log
          name: varlog
          readOnly: true
        - mountPath: /var/lib/kubelet/pods
          name: pod-test
          readOnly: true
      dnsConfig:
        options:
        - name: single-request-reopen
      dnsPolicy: ClusterFirstWithHostNet
      restartPolicy: Always
      securityContext: {}
      serviceAccount: filebeat
      serviceAccountName: filebeat
      tolerations:
      - operator: Exists
      volumes:
      - configMap:
          defaultMode: 416
          name: filebeat-config
        name: config
      - hostPath:
          path: /var/lib/docker/containers
          type: ""
        name: varlibdockercontainers
      - hostPath:
          path: /var/log
          type: ""
        name: varlog
      - hostPath:
          path: /var/lib/filebeat-data
          type: DirectoryOrCreate
        name: data
      - hostPath:
          path: /mnt/paas/kubernetes/kubelet/pods
          type: DirectoryOrCreate
        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
kind: ConfigMap
metadata:
  name: filebeat-config
  namespace: ops-monit
  labels:
    k8s-app: filebeat
data:
  filebeat.yml: |-
    filebeat.inputs:
    - type: log  # 文件日志
```

```
      paths:
        - /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
      multiline.type: pattern  # 多行日志匹配
      multiline.pattern: ^\d{4}-\d{2}-\d{2}\s\d{2}:\d{2}:\d{2}
      multiline.negate: true
      multiline.match: after
      fields: # 增加字段
        logtype: java-file
        env: f2b-gamma
      processors:
      - add_kubernetes_metadata: # 增加K8S元数据
          indexers:
            - pod_uid:
          matchers:
            - logs_path:
                logs_path: '/var/lib/kubelet/pods/'
                resource_type: 'pod'

    - type: container # 采集容器标准输出
      paths:
        - /var/log/containers/*_f2b_*node*.log # 默认标准输出在节点主机的路径
      fields: # 增加字段
        logtype: json-std
        env: f2b-gamma
      processors:
      - add_kubernetes_metadata: # 增加K8S元数据
          matchers:
            - logs_path:
                logs_path: /var/log/containers

    output.kafka: # 输出到KAFKA
      hosts: ["10.1.230.152:9092", "10.1.227.35:9092", "10.1.247.50:9092"]
      topic: 'topic-gamma-logs'
      max_message_bytes: 10000000
      partition.round_robin:
        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
```

```yaml
kind: ClusterRole
metadata:
  name: vector
  labels:
    app.kubernetes.io/name: vector
rules:
  - apiGroups:
      - ""
    resources:
      - namespaces
      - nodes
      - pods
    verbs:
      - list
      - watch
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: vector
  labels:
    app.kubernetes.io/name: vector
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: vector
subjects:
  - kind: ServiceAccount
    name: vector
    namespace: ops-monit
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: vector
  namespace: ops-monit
  labels:
    app.kubernetes.io/name: vector
spec:
  replicas: 1
  selector:
    matchLabels:
      app.kubernetes.io/name: vector
  template:
    metadata:
      annotations: {}
      labels:
        app.kubernetes.io/name: vector
        vector.dev/exclude: "true"
    spec:
      serviceAccountName: vector
      dnsPolicy: ClusterFirst
      containers:
        - name: vector
          image: "registry.cn-shenzhen.aliyuncs.com/starsl/vector:0.42.0-alpine"
```

```yaml
          imagePullPolicy: IfNotPresent
          args:
            - --config-dir
            - /etc/vector/
          env:
            - name: TZ
              value: Asia/Shanghai
          ports:
            - name: api
              containerPort: 8686
              protocol: TCP
          resources:
            limits:
              cpu: '2'
              memory: 2Gi
            requests:
              cpu: 500m
              memory: 100Mi
          volumeMounts:
            - name: data
              mountPath: "/vector-data-dir"
            - name: config
              mountPath: "/etc/vector/"
              readOnly: true
      terminationGracePeriodSeconds: 60
      affinity:
        podAntiAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            - labelSelector:
                matchLabels:
                  app.kubernetes.io/name: vector
              topologyKey: kubernetes.io/hostname
      volumes:
        - name: data
          emptyDir: {}
        - name: config
          projected:
            sources:
              - configMap:
                  name: vector
---
apiVersion: v1
kind: Service
metadata:
  namespace: ops-monit
  name: vector
  labels:
    app.kubernetes.io/name: vector
spec:
  ports:
    - name: api
      port: 8686
      protocol: TCP
  selector:
    app.kubernetes.io/name: vector
  type: NodePort
```

Vector：日志处理与入库配置

消费KAFKA处理日志并入库配置

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: vector
  namespace: ops-monit
data:
  global.toml: |
    data_dir = "/vector-data-dir"
    timezone = "Asia/Shanghai"
    [api]
      address = "0.0.0.0:8686"
      enabled = true

  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]
      type = "kafka"
      bootstrap_servers = "10.0.100.1:9092,10.0.100.2:9092,10.0.100.3:9092"
      group_id = "prod-java-file-ilogtail"
      topics = [ "prod-java-file-ilogtail" ]
      decoding.codec = "json"
      commit_interval_ms = 1000

    [transforms.2parse_java_all_ilogtail]
    inputs = ["1kafka_java_ilogtail"]
    type = "remap"
    source = '''
      .namespace = .tags.namespace
      .podname = .tags.pod_name
      .nodename = .tags.node_name
      .appname = .tags.appname
      .deployimage = .tags.deployimage
      # 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.
      . |= parse_regex!(.fields.content, r'^(?P<time>\d+-\d+-\d+
\d+:\d+:\d+,\d+)(\s|:)+(?P<level>\S+)(\S|\s)*$')
    '''

    [transforms.3delfields_java_ilogtail]
    inputs = ["2parse_java_all_ilogtail"]
    type = "remap"
    source = '''
    .unix_time, err = to_int(format_timestamp!(.timestamp, format: "%s"))
    .utc8_time = .unix_time + 28800
    .index_day = format_timestamp!(to_timestamp!(.utc8_time), "%Y.%m.%d")
    .path = .tags.path
    del(.unix_time)
    del(.utc8_time)
    del(.tags)
    .message = del(.fields.content)
    del(.topic)
    del(.offset)
```

```
    del(.partition)
    del(.message_key)
    del(.source_type)
    del(.fields.__file_offset__)
    .@timestamp = parse_timestamp(.time, format: "%F %T,%3f") ?? now()
    .captime = del(.timestamp)
    del(.time)
    ns, err = string(.namespace)
    if err != null {
      .namespace = "null"
    }
    '''

    [sinks.9es_java_ilogtail]
      type = "elasticsearch"
      inputs = ["3delfields_java_ilogtail"]
      endpoint = "http://10.1.72.33:19200"
      auth.strategy = "basic"
      auth.user = "elastic"
      auth.password = "M1k2wl2Ig4A"
      mode = "bulk"
      buffer.max_events = 50000
      batch.max_events = 50000
      batch.timeout_secs = 5
      bulk.index = "hwprod-java-file-{{ namespace }}-{{ index_day }}"

  istio.yaml: |-  # istio请求日志入库ClickHouse(yaml格式配置文件写法)
    sources:
      01_kafka_hwprod_istio:
        type: "kafka"
        bootstrap_servers: "kafka1:9092,kafka2:9092,kafka3:9092"
        group_id: "hwprod_istio"
        topics: [ "prod-istio-std" ]
        decoding:
          codec: "json"
        commit_interval_ms: 1000

    transforms:
      02_parse_hwprod_istio:
        drop_on_error: true
        reroute_dropped: true
        type: remap
        inputs:
          - 01_kafka_hwprod_istio
        source: |
          # json日志直接解析json,即可获取所有字段.
          . = parse_json!(.contents.content)
          .start_time = to_unix_timestamp(parse_timestamp!(.start_time,
format: "%+"),unit: "milliseconds")
          .upstream_service_time = to_int!(.upstream_service_time)

    sinks:
      03_ck_hwprod_istio:
        type: clickhouse
        inputs:
          - 02_parse_hwprod_istio
```

```
        database: istiologs
        endpoint: http://10.7.0.226:28123
        table: hwprod_istio_local
        compression: gzip
      04_out_istio_dropped:
        type: blackhole
        inputs:
          - 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

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: ilogtail-user-cm
  namespace: ops-monit
data:
  java_std_es8.yaml: |
    enable: true
    inputs:
      - Type: service_docker_stdout
        Stderr: true
        Stdout: true
        BeginLineCheckLength: 10
        BeginLineRegex: "\\d+-\\d+-\\d+.*"  # JAVA多行日志开头匹配正则
        K8sNamespaceRegex: "^(saas-prod)$"
    processors:
      - Type: processor_string_replace  # 如果日志的标准输出有颜色,可以使用此正则替换掉
        SourceKey: content
        Method: regex
        Match: \[\d*\;*\d*m|\\u001b
        ReplaceString: ''
      - 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。
        SourceKey: content
        Regex: (\d{4}-\d{2}-\d{2} \d{2}:\d{2}:\d{2},\d{3}) .*?[0-9a-z]*([A-Z]+)
        Keys:
          - log_time
          - level
        KeepSource: true
      - Type: processor_gotime  # 日志中的时间字段转成ES中@timestamp标准日期时间格式。
        SourceKey: "log_time"
        SourceFormat: "2006-01-02 15:04:05,000"
        SourceLocation: 8
        DestKey: "@timestamp"
        DestFormat: "2006-01-02T15:04:05.000Z"
        DestLocation: 0
        SetTime: false
        KeepSource: false
      - Type: processor_rename  # 字段重命名
```

```
        SourceKeys:
          - _time_
          - _source_
        DestKeys:
          - cap_time
          - log_src
    flushers:
      #- Type: flusher_stdout
      #  OnlyStdout: true
      - Type: flusher_elasticsearch
        Addresses:
          - http://10.26.6.152:9200
          - http://10.26.6.153:9200
          - http://10.26.6.154:9200
        Convert:
          Protocol: custom_single_flatten  # 数据字段做一级打平
          Encoding: json
        Index: java-logs-%{tag.k8s.namespace.name}_%{+yyyy.MM.dd}
        Authentication:
          PlainText:
            Username: ilogtail1
            Password: fSdgDh4jfg
        HTTPConfig:
            MaxIdleConnsPerHost: 10
            ResponseHeaderTimeout: Second
```

💎【数据库】ClickHouse与ES部署