监控nginx-export案例

之前我们讲到了，监控k8s集群中自动获取的mecrics指标状态，如果我们要获取的指标是一些外部的特殊指标呢？比如nginx，kafka，rabbitmq，es，jenkins等等其他的

首先更改prometheus-clusterRole.yaml否则无法发现,没有权限

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: prometheus-k8s
rules:
- apiGroups:
  - ""
  resources:
  - nodes
  - services
  - endpoints
  - pods
  - nodes/proxy
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - ""
  resources:
  - configmaps
  - nodes/metrics
  verbs:
  - get
- nonResourceURLs:
  - /metrics
  verbs:
  - get
```

配置ingress-nginx-ServiceMonitor，(svc课前已做好名称为：nginx-metrics 在namespace: ingress-nginx下面)

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
    app.kubernetes.io/name: nginx
  name: kube-nginx
  namespace: monitoring
spec:
  endpoints:
  - interval: 30s
    port: metrics
  namespaceSelector:
    matchNames:
    - kube-system
  selector:
    matchLabels:
      app.kubernetes.io/name: nginx
```

然后我们就可以看到service-discovery 中看到他抓取的接口了

自动发现

ServiceMonitor虽然可以发现监控接口，但是如果数量庞大起来我们需要配置无数个ServiceMonitor为我们维护大大增加了成本，但是自动发现配置简化了这个流程。

我们来到命名空间ingress-nginx 查看nginx-ingress-controller可以看到他里面自带有个标签为下面图这里就是一个自动发现的配置

```
      annotations:
        prometheus.io/port: "10254" #metrics端口
        prometheus.io/scrape: "true" #自动配置
```

我们先删除之前的nginx-ServiceMonitor,配置下面的additional-cm.yaml

增加additional-cm.yaml配置文件

```
apiVersion: v1
data: {}
kind: Secret
```

```
metadata:
  name: additional-configs
  namespace: monitoring
type: Opaque
stringData:
  prometheus-additional.yaml: |-
    - job_name: 'kubernetes-endpoints'
      kubernetes_sd_configs:
      - role: endpoints
      relabel_configs:
      - source_labels:
[__meta_kubernetes_service_annotation_prometheus_io_scrape] #服务
对象注解
        action: keep
        regex: true
      - source_labels:
[__meta_kubernetes_service_annotation_prometheus_io_scheme] #指定
协议
        action: replace
        target_label: __scheme__
        regex: (https?)
      - source_labels:
[__meta_kubernetes_service_annotation_prometheus_io_path] #指定路径
默认是/metrics
        action: replace
        target_label: __metrics_path__
        regex: (.+)
      - source_labels: [__address__,
__meta_kubernetes_service_annotation_prometheus_io_port] #指定
metrics端口
        action: replace
        target_label: __address__
        regex: ([^:]+)(?::\d+)?;(\d+)
        replacement: $1:$2
      - action: labelmap
        regex: __meta_kubernetes_service_label_(.+)
      - source_labels: [__meta_kubernetes_namespace]
        action: replace
        target_label: namespace
      - source_labels: [__meta_kubernetes_service_name]
        action: replace
        target_label: svc
      - source_labels: [__meta_kubernetes_pod_name]
        action: replace
        target_label: pod
    - job_name: 'kubernetes-pods'
```

```
      honor_timestamps: true
      scrape_interval: 30s
      scrape_timeout: 10s
      metrics_path: /metrics
      scheme: http
      kubernetes_sd_configs:
      - role: pod
      relabel_configs:
      - source_labels:
[__meta_kubernetes_pod_annotation_prometheus_io_scrape]
        separator: ;
        regex: "true"
        replacement: $1
        action: keep
      - source_labels:
[__meta_kubernetes_pod_annotation_prometheus_io_scheme]
        separator: ;
        regex: (https?)
        target_label: __scheme__
        replacement: $1
        action: replace
      - source_labels:
[__meta_kubernetes_pod_annotation_prometheus_io_path]
        separator: ;
        regex: (.+)
        target_label: __metrics_path__
        replacement: $1
        action: replace
      - source_labels: [__address__,
__meta_kubernetes_pod_annotation_prometheus_io_port]
        separator: ;
        regex: ([^:]+)(?::\d+)?;(\d+)
        target_label: __address__
        replacement: $1:$2
        action: replace
      - separator: ;
        regex: __meta_kubernetes_service_label_(.+)
        replacement: $1
        action: labelmap
      - source_labels: [__meta_kubernetes_namespace]
        separator: ;
        regex: (.*)
        target_label: namespace
        replacement: $1
        action: replace
      - source_labels: [__meta_kubernetes_service_name]
```

```
        separator: ;
        regex: (.*)
        target_label: svc
        replacement: $1
        action: replace
      - source_labels: [__meta_kubernetes_pod_name]
        separator: ;
        regex: (.*)
        target_label: pod
        replacement: $1
        action: replace
```

然后配置prometheus-prometheus.yaml增加

```
  additionalScrapeConfigs:
    name: additional-configs
    key: prometheus-additional.yaml
#之后看看prometheus是否重启，没有重启更改副本数量重启下
```

然后我们可以在targets下看到配置了

有一个问题是为什么我配置了ingress-nginx在Service Discovery能看见为什么在Targets中看不到接口信息呢，因为Targets每个规则显示的是prometheus-Configuration中配置的，如果想要在Service Discovery和Targets中同时看到需要配置如下例子（只截取一小段）：

```
- job_name: monitoring/alertmanager/0 #这里就是Targets显示的配置信息
  honor_timestamps: true
  scrape_interval: 30s
  scrape_timeout: 10s
  metrics_path: /metrics
  scheme: http
  relabel_configs:
  - source_labels: [__meta_kubernetes_service_label_alertmanager]
    separator: ;
    regex: main
    replacement: $1
    action: keep
  - source_labels:
[__meta_kubernetes_service_label_app_kubernetes_io_component]
```

```
    separator: ;
    regex: alert-router
    replacement: $1
    action: keep
```