部署 prometheus operator

作者：fish2018
原文链接：https://ld246.com/article/1566379324671
来源网站：链滴

prometheus-operator创建了4个CRD:

- Prometheus

由 Operator 依据一个自定义资源 kind: Prometheus 类型中，所描述的内容而部署的 Prometheus erver 集群，可以将这个自定义资源看作是一种特别用来管理Prometheus Server的StatefulSets资。

- ServiceMonitor

一个Kubernetes自定义资源(和 kind: Prometheus 一样是CRD)，该资源描述了Prometheus Server Target列表，Operator 会监听这个资源的变化来动态的更新Prometheus Server的Scrape targets让prometheus server去reload配置(prometheus有对应reload的http接口 /-/reload)。而该资源要通过Selector来依据 Labels 选取对应的Service的endpoints，并让 Prometheus Server 通过 Servce 进行拉取（拉）指标资料(也就是metrics信息),metrics信息要在http的url输出符合metrics格式的息,ServiceMonitor也可以定义目标的metrics的url.

- Alertmanager

Prometheus Operator 不只是提供 Prometheus Server 管理与部署，也包含了 AlertManager，并一样通过一个 kind: Alertmanager 自定义资源来描述信息，再由 Operator 依据描述内容部署 Alertanager 集群。

- PrometheusRule

对于Prometheus而言，在原生的管理方式上，我们需要手动创建Prometheus的告警文件，并且通在Prometheus配置中声明式的加载。而在Prometheus Operator模式中，告警规则也编程一个通过ubernetes API 声明式创建的一个资源.告警规则创建成功后，通过在Prometheus中使用想servicemnitor那样用 ruleSelector 通过label匹配选择需要关联的PrometheusRule即可

部署

```
git clone https://github.com/coreos/kube-prometheus.git
```

修改kube-prometheus/manifests/grafana-service.yaml和prometheus-service.yaml，在spec下加 type: NodePort 用于暴露服务

```
kubectl apply -f kube-prometheus/manifests
```

首次执行会报错，过一会再执行一遍kubectl apply，应该是一些依赖的资源还没创建完，后面依赖的资源先创建了，所以过一会儿再执行一遍就可以了

```
[root@k8s03 kube-prometheus]# kubectl get crd |grep coreos
alertmanagers.monitoring.coreos.com          2019-07-31T05:50:57Z
podmonitors.monitoring.coreos.com            2019-07-31T05:50:57Z
prometheuses.monitoring.coreos.com           2019-07-31T05:50:58Z
prometheusrules.monitoring.coreos.com        2019-07-31T05:50:59Z
servicemonitors.monitoring.coreos.com        2019-07-31T05:51:00Z

[root@k8s03 kube-prometheus]# kubectl get pods -n monitoring
NAME                                  READY   STATUS             RESTARTS   AGE
alertmanager-main-0                   2/2     Running            0          10m
alertmanager-main-1                   2/2     Running            0          7m38s
alertmanager-main-2                   2/2     Running            0          4m1s
grafana-7dc5f8f9f6-ghtk5              1/1     Running            0          19m
kube-state-metrics-58b66579dc-cm7g8   3/4     ImagePullBackOff   0          19m
node-exporter-kb2j9                   2/2     Running            0          19m
```

```
node-exporter-lqs5d                  2/2     Running          0          19m
node-exporter-tf6f6                  2/2     Running          0          19m
prometheus-adapter-668748ddbd-z9lbv  1/1     Running          0          19m
prometheus-k8s-0                     3/3     Running          1          10m
prometheus-k8s-1                     3/3     Running          1          10m
prometheus-operator-7447bf4dcb-7d4t4 1/1     Running          0          19m

[root@k8s03 manifests]# kubectl get svc -n monitoring
NAME                    TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)             AGE
alertmanager-main       ClusterIP   10.100.176.121   <none>        9093/TCP            32m
alertmanager-operated   ClusterIP   None             <none>        9093/TCP,6783/TCP   23m
grafana                 NodePort    10.100.48.63     <none>        3000:31493/TCP      32m
kube-state-metrics      ClusterIP   None             <none>        8443/TCP,9443/TCP   32m
node-exporter           ClusterIP   None             <none>        9100/TCP            32m
prometheus-adapter      ClusterIP   10.102.205.39    <none>        443/TCP             32m
prometheus-k8s          NodePort    10.102.176.248   <none>        9090:31782/TCP      32m
prometheus-operated     ClusterIP   None             <none>        9090/TCP            23m
prometheus-operator     ClusterIP   None             <none>        8080/TCP            32m
```

打开prometheus的targets页面，可以看到有2个组件监控不到

```
monitoring/kube-controller-manager/0 (0/0 up)
monitoring/kube-scheduler/0 (0/0 up)
```

查看prometheus-serviceMonitorKubeScheduler.yaml和prometheus-serviceMonitorKubeControllerManager.yaml可以看到servicemonitor通过k8s-app=kube-scheduler和k8s-app: kube-controller-manager进行匹配的，

使用命令kubectl get svc -n kube-system可以看到并没有这两个service，所以需要手动创建这两个srvice,注意labels和selector部分的配置必须和ServiceMonitor对象中的保持一致。

10251是kube-scheduler组件 metrics 数据所在的端口，10252是kube-controller-manager组件监控数据所在端口。

cat prometheus-kubeSchedulerService.yaml

```
apiVersion: v1
kind: Service
metadata:
  namespace: kube-system
  name: kube-scheduler
  labels:
    k8s-app: kube-scheduler
spec:
  selector:
    component: kube-scheduler
  ports:
  - name: http-metrics
    port: 10251
    targetPort: 10251
    protocol: TCP
```

cat prometheus-kubeControllerManagerService.yaml

```
apiVersion: v1
```

```
kind: Service
metadata:
  namespace: kube-system
  name: kube-controller-manager
  labels:
    k8s-app: kube-controller-manager
spec:
  selector:
    component: kube-controller-manager
  ports:
  - name: http-metrics
    port: 10252
    targetPort: 10252
    protocol: TCP
```

```
kubectl apply -f prometheus-kubeSchedulerService.yaml
kubectl apply -f prometheus-kubeControllerManagerService.yaml
```

查看prometheus-operator 状态

```
kubectl get pod -n monitoring -o wide | grep prometheus-operator
kubectl get service -n monitoring | grep prometheus-operator
kubectl get ServiceMonitor -n monitoring | grep prometheus-operator
kubectl api-versions| grep monitoring
kubectl get --raw "/apis/monitoring.coreos.com/v1"|jq .
```