### 3.8.6 监控ElasticSearch集群

上一小节讲解了如何一键安装Prometheus监控，实际使用中仅使用资源监控和异常报警是远远不够的，还需要监控各类基础组件，如ElasticSearch、RabbitMQ等，并根据实际使用情况，对资源监控进行分类并设置报警策略，比如各类组件集群的状态、将不同的业务报警发送给不同的技术人员等。

本节将演示对Kubernetes集群外部的ElasticSearch集群监控，对于Kubernetes集群内部的ElasticSearch监控配置相同。

使用Prometheus进行业务监控，一般可分为以下步骤：

（1）创建Endpoints用于连接至外部服务。

（2）创建Service用于与Endpoint进行匹配。

（3）创建Exporter采集工具并配置上面创建的Service用于连接被监控业务。

（4）创建Exporter的Service用于被Prometheus发现并注册。

（5）创建ServiceMonitor将被监控资源注册至Prometheus。

根据上述步骤，首先创建用于连接Kubernetes集群外部的ElasticSearch集群的Endpoints和Service（集群内部的ElasticSearch一般都已经创建了Service和Endpoints，可以直接使用，无须再次创建），修改external-es.yaml文件中的Endpoints的addresses（地址）和port（端口），然后创建即可：

kind: Endpoints

apiVersion: v1

metadata:

name: external-es

namespace: logging

labels:

k8s-app: elasticsearch-logging

subsets:

- addresses:

- ip: 192.168.100.193

- ip: 192.168.100.194

- ip: 192.168.100.195

ports:

- port: 9200

name: es

protocol: TCP

---

apiVersion: v1

kind: Service

metadata:

name: external-es

namespace: logging

labels:

k8s-app: elasticsearch-logging

spec:

type: ClusterIP

clusterIP: None

ports:

- port: 9200

protocol: TCP

targetPort: 9200

name: es

提 示

当Service的名字和Endpoint的名字相同时，会自动建立联系。

一般情况下，在Prometheus监控中，对于数据的采集均使用一个被名为Exporter的采集工具，然后通过Deployment或者DaemonSet将此Exporter工具部署到集群中，用于数据的采集和汇总。比如采集宿主机节点信息的exporter：node-exporter：

$ kubectl get po -n monitoring -l app=node-exporter

NAME READY STATUS RESTARTS AGE

node-exporter-bpqx2 2/2 Running 0 51d

node-exporter-hrvjg 2/2 Running 2 51d

node-exporter-jlwsr 2/2 Running 0 42d

node-exporter-jns4f 2/2 Running 2 51d

node-exporter-lrnnw 2/2 Running 0 51d

node-exporter-pkttq 2/2 Running 0 51d

node-exporter-s6scl 2/2 Running 0 51d

node-exporter-s7lkm 2/2 Running 2 51d

node-exporter-sdt6b 2/2 Running 0 51d

以同样的方式创建ElasticSearch的Exporter用于采集ElasticSearch集群的各类监控数据。ElasticSearch Exporter文件（es-exporter-deploy.yaml文件）如下（需要注意的是，-es.uri=http://external-es.logging.svc:9200参数中配置的外部ElasticSearch集群的Service地址，本例配置为上述创建的Service）：

apiVersion: extensions/v1beta1

kind: Deployment

metadata:

labels:

k8s-app: elasticsearch-exporter

release: es-exporter

name: es-exporter-elasticsearch-exporter

namespace: monitoring

spec:

progressDeadlineSeconds: 600

replicas: 1

revisionHistoryLimit: 10

selector:

matchLabels:

k8s-app: elasticsearch-exporter

release: es-exporter

strategy:

rollingUpdate:

maxSurge: 1

maxUnavailable: 0

type: RollingUpdate

template:

metadata:

creationTimestamp: null

labels:

k8s-app: elasticsearch-exporter

release: es-exporter

spec:

containers:

- command:

- elasticsearch\_exporter

- -es.uri=http://external-es.logging.svc:9200

- -es.all=true

- -es.indices=true

- -es.timeout=30s

- -web.listen-address=:9108

- -web.telemetry-path=/metrics

image: justwatch/elasticsearch\_exporter:1.0.2

imagePullPolicy: IfNotPresent

livenessProbe:

failureThreshold: 3

httpGet:

path: /health

port: http

scheme: HTTP

initialDelaySeconds: 30

periodSeconds: 10

successThreshold: 1

timeoutSeconds: 10

name: elasticsearch-exporter

ports:

- containerPort: 9108

name: http

protocol: TCP

readinessProbe:

failureThreshold: 3

httpGet:

path: /health

port: http

scheme: HTTP

initialDelaySeconds: 10

periodSeconds: 10

successThreshold: 1

timeoutSeconds: 10

resources: {}

securityContext:

capabilities:

drop:

- SETPCAP

- MKNOD

- AUDIT\_WRITE

- CHOWN

- NET\_RAW

- DAC\_OVERRIDE

- FOWNER

- FSETID

- KILL

- SETGID

- SETUID

- NET\_BIND\_SERVICE

- SYS\_CHROOT

- SETFCAP

procMount: Default

readOnlyRootFilesystem: true

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

dnsPolicy: ClusterFirst

restartPolicy: Always

schedulerName: default-scheduler

securityContext:

runAsNonRoot: true

runAsUser: 1000

terminationGracePeriodSeconds: 30

上述步骤创建了用于采集数据的Exporter，还需要创建一个Exporter的Service，用于将运行Exporter的Pod暴露给Prometheus，创建Service文件（es-exporter-svc.yaml）如下：

apiVersion: v1

kind: Service

metadata:

labels:

k8s-app: elasticsearch-exporter

release: es-exporter

name: es-exporter-elasticsearch-exporter

namespace: monitoring

spec:

ports:

- name: http

port: 9108

protocol: TCP

targetPort: 9108

selector:

k8s-app: elasticsearch-exporter

release: es-exporter

sessionAffinity: None

type: ClusterIP

上述各类资源创建完成以后，需要创建ServiceMonitor将监控注册到Prometheus中，如果配置了自动发现则无须此步骤。创建ServiceMonitor文件（es-servicemonitor.yaml）如下：

apiVersion: monitoring.coreos.com/v1

kind: ServiceMonitor

metadata:

labels:

k8s-app: elasticsearch-exporter

release: es-exporter

name: es-exporter-elasticsearch-exporter

namespace: monitoring

spec:

endpoints:

- honorLabels: true

interval: 10s

path: /metrics

port: http

scheme: http

jobLabel: es-exporter

namespaceSelector:

matchNames:

- monitoring

selector:

matchLabels:

k8s-app: elasticsearch-exporter

release: es-exporter

当ServiceMonitor创建完成以后，对ElasticSearch集群的监控就完成了，此时可以通过访问Prometheus的UI界面查看ElasticSearch是否成功注册到了Prometheus，访问上述部署的Prometheus（上述部署的Prometheus的端口为30900），查看Target，如图3-31所示。

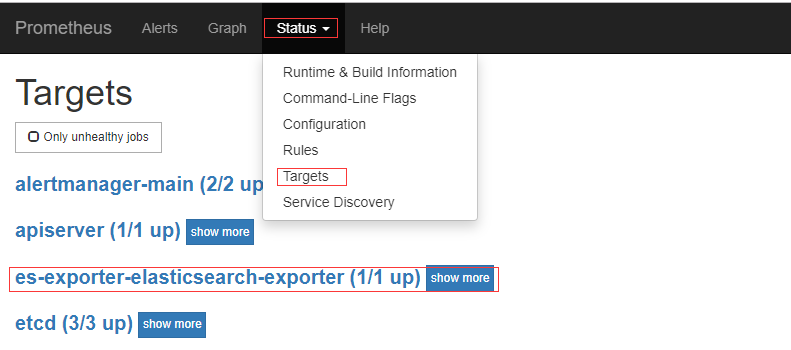


图3-31 Prometheus UI界面

在Prometheus的UI界面中能看到es-exporter-elasticsearch-exporter的target即表示监控成功。

添加完监控以后，需要在Grafana中添加监控ElasticSearch的Dashboard，用于展示ElasticSearch数据。将登录之前创建的Grafana导入Dashboard（ElasticSearch-Cluster-1555488325737.json），如图3-32所示。

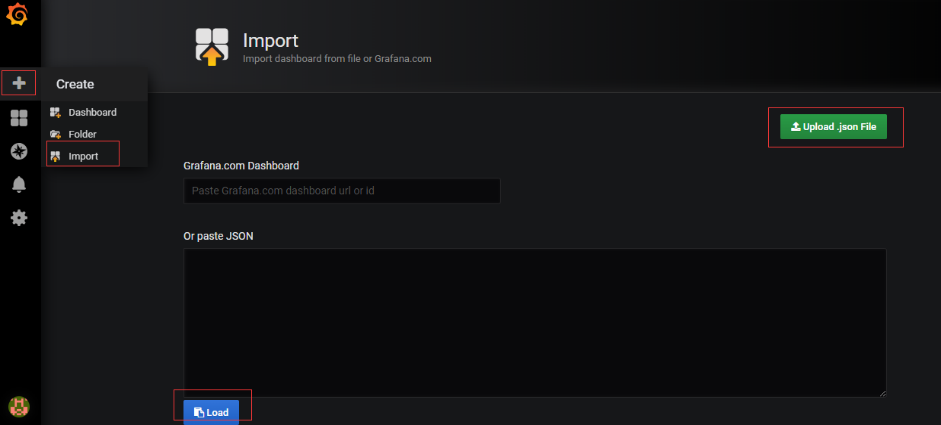


图3-32 导入Dashboard

对于ElasticSearch的监控包含了很多的监控项，比如：KPI、Shards、JVM信息等，可以通过上述添加的Dashboard查看对应的监控数据，如图3-33所示。

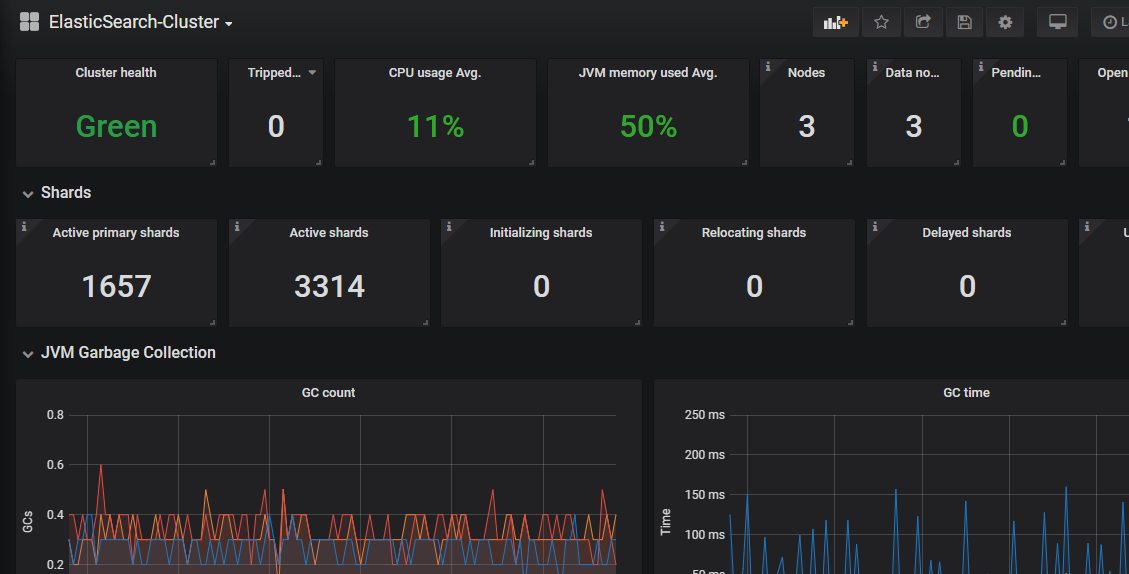


图3-33 查看监控数据

通过上述步骤即完成了对ElasticSearch集群的监控，其他应用的监控，比如：RabbitMQ、Redis、Zookeeper等配置均可参考上述步骤。有关采集数据的工具在开源社区GitHub上基本上都能找到对应的Exporter，同样参考上述步骤即可完成对其他服务的监控。

### 3.8.7 监控报警配置实战

上一节讲解了Prometheus如何添加监控并通过Grafana展示图表，本节将演示对于一个新的监控如何添加报警策略，这也是监控体系中至关重要的一步。虽然之前一键安装Prometheus后也创建了一些基本的监控报警策略，但是在生产环境中还是要根据业务需求进行自定义监控报警。

通过Prometheus添加监控以后，可以在Prometheus的Web 界面的Graph页面查看对应的业务数据，如查看上节监控的ElasticSearch的监控数据，如图3-34所示。

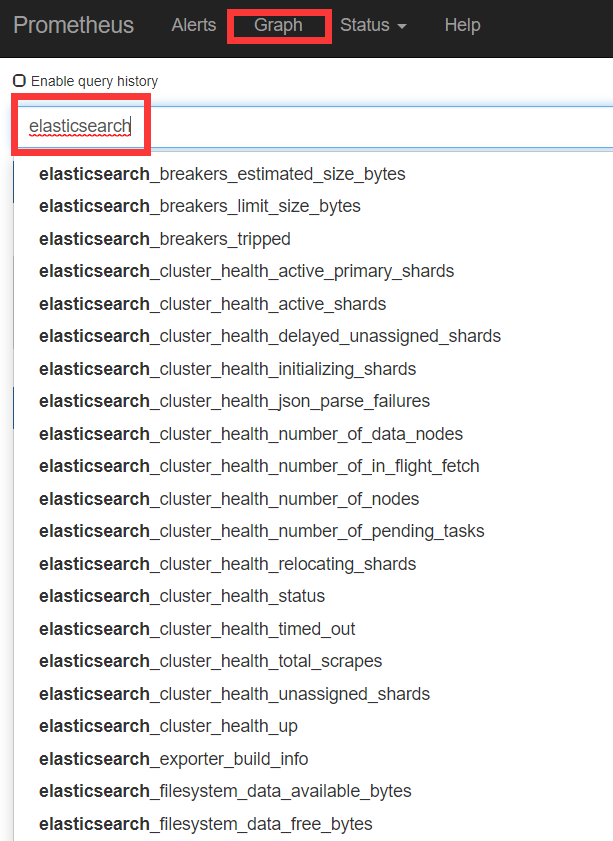


图4-34 ElasticSearch的监控数据

之后选择对应的监控指标，查看当前监控指标的数据，如查看当前ElasticSearch的集群状态，如图3-35所示。

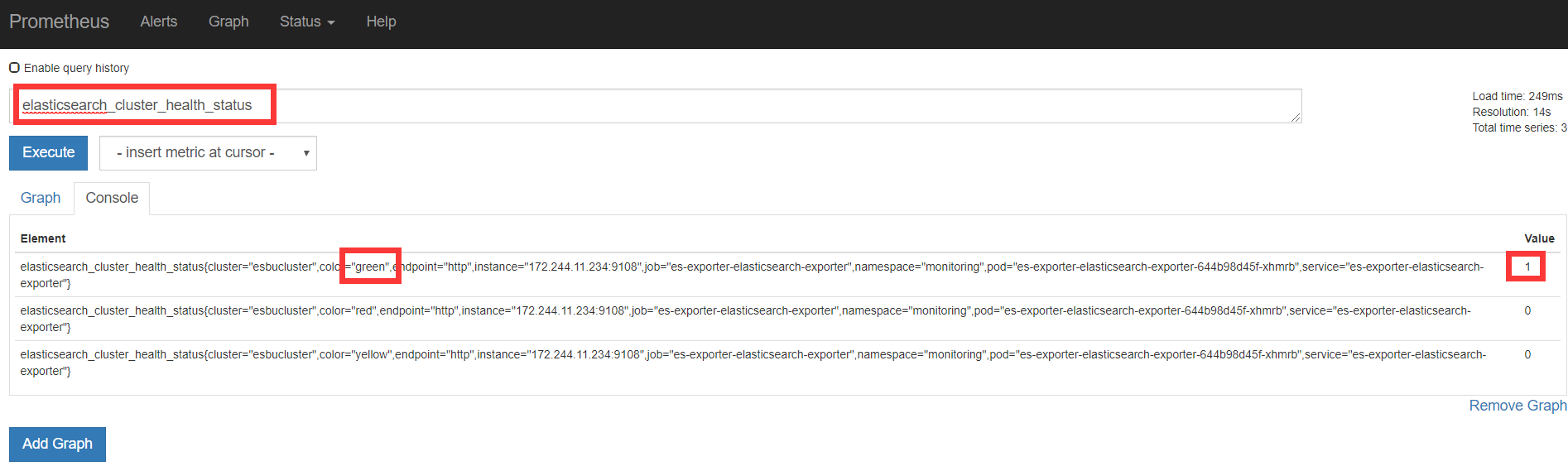


图3-35 监控数据查看

可以看到，此时集群状态green为1，red和yellow为0，说明此时集群状态为正常状态。通过这些监控指标可以获取到被监控业务的一些监控数据，然后通过这些数据，可以根据需求添加对应的报警策略，比如当集群状态的yellow值为1时触发报警，对应的报警策略如下：

- alert: Elastic\_Cluster\_Health\_Yellow

expr: elasticsearch\_cluster\_health\_status{color="yellow"}==1

for: 10m

labels:

severity: warning

value: '{{$value}}'

annotations:

description: "Instance {{ $labels.instance }}: not all primary and replica shards are allocated in elasticsearch cluster {{ $labels.cluster }}."

summary: "Instance {{ $labels.instance }}: not all primary and replica shards are allocated in elasticsearch cluster {{ $labels.cluster }}"

其中，需要注意的参数如下：

* alert：报警策略名称。
* expr：报警策略表达式。
* for：评估等待时间。
* annotations：报警信息，一般为报警内容。

上述为监控指标中的某一项指标的监控策略，可以根据上述方法添加其他监控项的报警策略，也可以使用已经定义好的监控策略文件，直接更新之前创建的报警策略：

$ kubectl replace -f prometheus-k8s-rules.yaml -n monitoring

该文件在末尾处添加了ElasticSearch集群关键的监控指标报警策略，当然读者也可以自行修改，然后更新之前创建的报警策略即可。更新报警策略配置以后，Prometheus会自动更新该配置，此时可以在Prometheus的Web 界面中看到对应的报警策略，如图3-36所示。

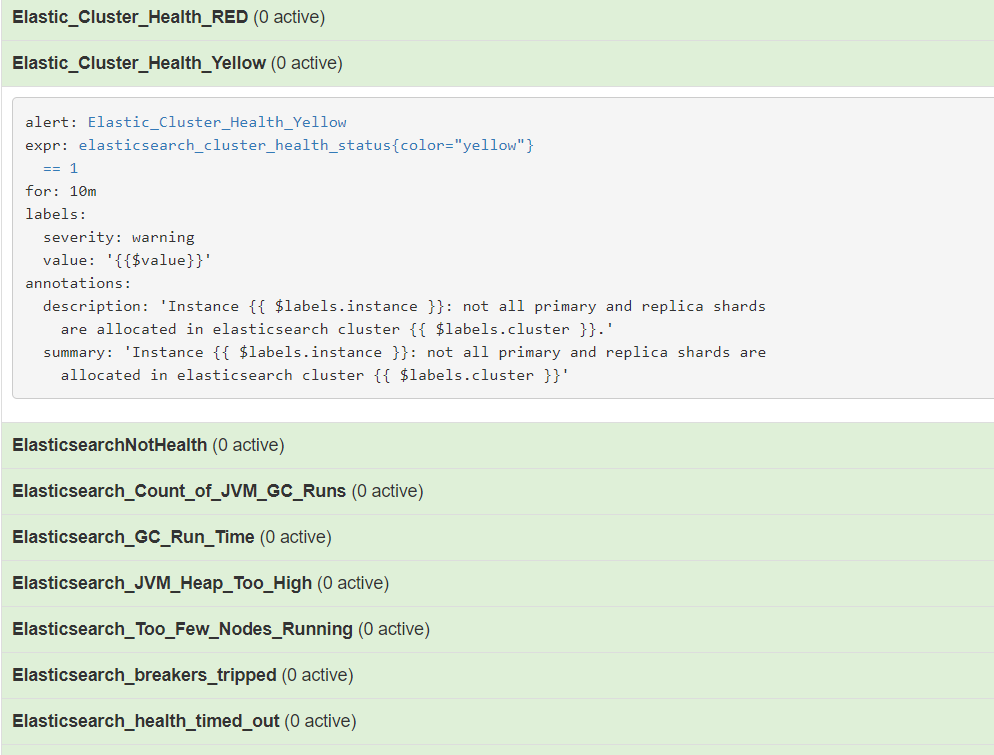


图3-36 报警策略

至此，完成了对ElasticSearch集群的监控和报警，对于其他应用的监控及报警配置基本相同，只需找到对应的Exporter，然后根据3.8.6节的步骤添加监控即可。Prometheus社区提供了丰富的Exporter，涵盖了基础设施、中间件、网络设备等各个方面的监控功能。常见的Exporter如表3-1所示。

表3-1 常见的Exporter

|  |  |
| --- | --- |
| 类型 | Exporter |
| 数据库 | MySQL Exporter, Redis Exporter, MongoDB Exporter, MSSQL Exporter |
| 硬件 | Apcupsd Exporter，IoT Edison Exporter， IPMI Exporter, Node Exporter |
| 消息队列 | Beanstalkd Exporter, Kafka Exporter, NSQ Exporter, RabbitMQ Exporter |
| 存储 | Ceph Exporter, Gluster Exporter, HDFS Exporter, ScaleIO Exporter |
| HTTP服务 | Apache Exporter, HAProxy Exporter, Nginx Exporter |
| API服务 | AWS ECS Exporter， Docker Cloud Exporter, Docker Hub Exporter, GitHub Exporter |
| 日志 | Fluentd Exporter, Grok Exporter |
| 监控系统 | Collectd Exporter, Graphite Exporter, InfluxDB Exporter, Nagios Exporter, SNMP Exporter |
| 其它 | Blockbox Exporter, JIRA Exporter, Jenkins Exporter， Confluence Exporter |