Prometheus黑盒监控跟白盒监控

一、黑盒监控跟白盒监控是什么?

- 白盒监控：监控一些内部的数据，topic的监控数据，Redis key的大小。内部暴露的指标被称为白盒监控。**比较关注的是原因。**
- 黑盒监控：站在用户的角度看到的东西。网站不能打开，网站打开的比较慢。**比较关注现象，表示正在发生的问题，正在发生的告警**

二、黑盒监控步骤

- 特别说明：新版的Prometheus是直接安装了 **blackbox-exporter**的，黑盒监控也属于exporter
- 通过 kubectl get po -n monitoring -l app.kubernetes.io/name=blackbox-exporter 可以直接查看到

2.1、blackbox-exporter官网+配套grafana

```
https://github.com/prometheus/blackbox_exporter
https://github.com/prometheus/blackbox_exporter/blob/master/blackbox.yml
https://grafana.com/grafana/dashboards/5345
```

2.2、手动部署blackbox-exporter

- 创建ConfigMap，通过ConfigMap形式挂载进容器里

```
[root@k8s-master01 blackbox-exporter-黑盒监控]# cat  blackbox-cm.yaml
apiversion: v1
kind: ConfigMap
metadata:
  name: blackbox.conf
  namespace: monitoring
data:
  blackbox.yml: |-
    modules:
      http_2xx:
        prober: http
      http_post_2xx:
        prober: http
        http:
          method: POST
      tcp_connect:
        prober: tcp
      pop3s_banner:
        prober: tcp
        tcp:
          query_response:
          - expect: "^+OK"
          tls: true
          tls_config:
            insecure_skip_verify: false
      ssh_banner:
        prober: tcp
        tcp:
          query_response:
          - expect: "^SSH-2.0-"
      irc_banner:
        prober: tcp
        tcp:
          query_response:
          - send: "NICK prober"
          - send: "USER prober prober prober :prober"
          - expect: "PING :([^ ]+)"
            send: "PONG ${1}"
          - expect: "^:[^ ]+ 001"
      icmp:
        prober: icmp
```

- 通过deployment清单部署blackbox

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: blackbox-exporter
  name: blackbox-exporter
  namespace: monitoring
spec:
  replicas: 1
  selector:
    matchLabels:
      app: blackbox-exporter
  strategy:
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 0
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: blackbox-exporter
    spec:
      containers:
      - args:
        - --config.file=/mnt/blackbox.yml
        env:
        - name: TZ
          value: Asia/Shanghai
        - name: LANG
          value: C.UTF-8
        image: prom/blackbox-exporter:master
        imagePullPolicy: IfNotPresent
        lifecycle: {}
        name: blackbox-exporter
        ports:
        - containerPort: 9115
          name: web
          protocol: TCP
        resources:
          limits:
            cpu: 260m
            memory: 395Mi
          requests:
            cpu: 10m
            memory: 10Mi
        securityContext:
          allowPrivilegeEscalation: false
          capabilities: {}
          privileged: false
          procMount: Default
          readOnlyRootFilesystem: false
          runAsNonRoot: false
        volumeMounts:
        - mountPath: /usr/share/zoneinfo/Asia/Shanghai
          name: tz-config
        - mountPath: /etc/localtime
          name: tz-config
        - mountPath: /etc/timezone
          name: timezone
        - mountPath: /mnt
          name: config
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      securityContext: {}
      volumes:
      - hostPath:
          path: /usr/share/zoneinfo/Asia/Shanghai
          type: ""
        name: tz-config
      - hostPath:
          path: /etc/timezone
          type: ""
        name: timezone
      - configMap:
          name: blackbox.conf
        name: config
```

- 创建Service

```
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    app: blackbox-exporter
  name: blackbox-exporter
  namespace: monitoring
spec:
  ports:
  - name: container-1-web-1
    port: 9115
    protocol: TCP
    targetPort: 9115
  selector:
    app: blackbox-exporter
  sessionAffinity: None
  type: ClusterIP
```

- 查看创建的svc、pod
  - 如果是新版本的blackbox-exporter，会多一个19115的http协议的端口；9115是https的

```
[root@k8s-master01 ~]# kubectl get po -n monitoring -l app=blackbox-exporter
NAME                                 READY   STATUS    RESTARTS   AGE
blackbox-exporter-6fdf7796d6-cx7gv   1/1     Running   0          3m54s

#
[root@k8s-master01 ~]# kubectl get svc -n monitoring -l app=blackbox-exporter
NAME                TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
blackbox-exporter   ClusterIP   10.110.131.83   <none>        9115/TCP   2m39s
```

- 测试exporter是否正常
  - IP 换成SVC的IP
  - 有数据说明blackbox-exporter已经能够正常使用

```
curl "http://10.110.131.83:9115/probe?target=baidu.com&module=http_2xx"
```

三、黑盒监控之域名监控实战

- 创建配置文件【**Prometheus静态配置文件也是这步骤配置**】

```
# 这是监控的静态配置文件写法
[root@k8s-master01 blackbox-exporter-黑盒监控]# cat prometheus-additional.yaml
- job_name: "blackbox"
  metrics_path: /probe                        # metrics接口地址
  params:
    module: [http_2xx]                        # 使用http模块,还有其他的模块,具体查看官网
  static_configs:                             # Prometheus静态配置
    - targets:
      - http://www.baidu.com                  # 监控的域名,有多个可以写多个
      - https://prometheus.io
  relabel_configs:
    - source_labels: [__address__]
      target_label: __param_target
    - source_labels: [__param_target]
      target_label: instance
    - target_label: __address__
      replacement: blackbox-exporter:9115    # blackbox-exporter的svc name:port,同
一个ns直接svc name:port访问即可
```

- 创建secret

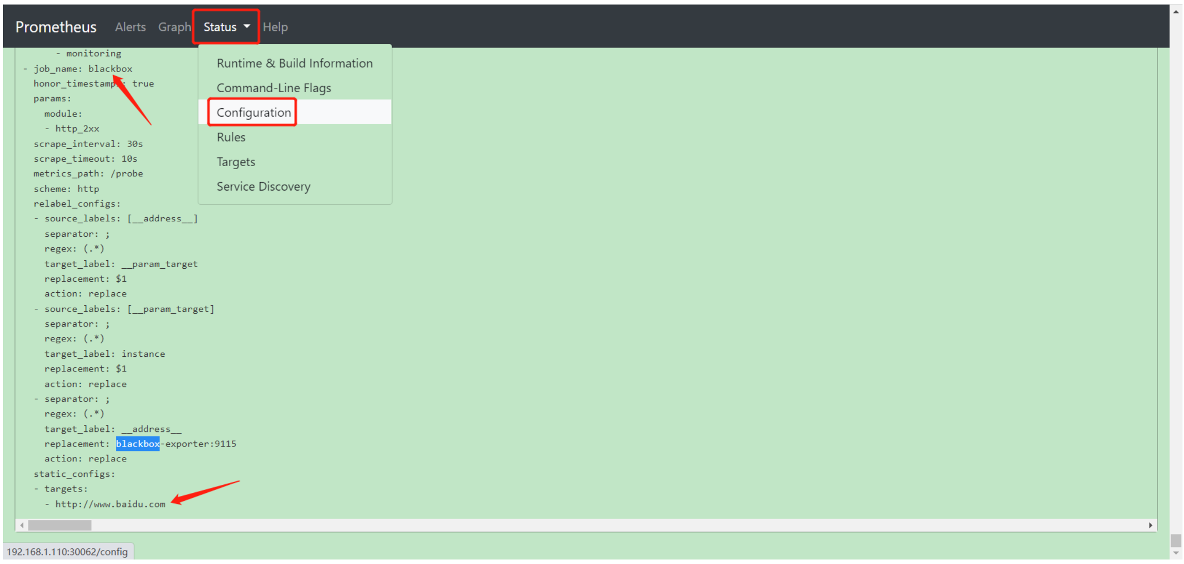
```
# 创建secret命令, 这里是创建到本地的文件中不是创建在k8s上
kubectl create secret generic additional-scrape-configs --from-file=prometheus-
additional.yaml --dry-run -oyaml > additional-scrape-configs.yaml


# 查看Secret
```

```
[root@k8s-master01 blackbox-exporter-黑盒监控]# cat additional-scrape-configs.yaml
apiVersion: v1
data:
  prometheus-additional.yaml:
LSBqb2JfbmFtZTogImJsYWNrYm94IgogIG1ldHJpY3NfcGF0aDogL3Byb2JlCiAgcGFyYW1zOgogICAg
bW9kdWxlOiBbaHR0cF8yeHhdICAjIExvb2sgZm9yIGEgSFRUUCAyMDAgcmVzcG9uc2UuCiAgc3RhdGlj
X2NvbmZpZ3M6CiAgICAtIHRhcmdldHM6CiAgICAgICAgICOgaHR0cDovL3d3dy5iYWlkdS5jb20gICAgCiAg
cmVsYWJlbF9jb25maWdzOgogICAgLSBzb3VyY2VfbGFiZWxzOiBbX19hZGRyZXNzX19dCiAgICAgIHRh
cmdldF9sYWJlbDogX19wYXJhbV90YXJnZXQKICAgIC0gc291cmNlX2xhYmVsczogW19fcGFyYW1fdGFy
Z2V0XQogICAgICB0YXJnZXRfbGFiZWw6IGluc3RhbmNlCiAgICAtIHRhcmdldF9sYWJlbDogX19hZGRy
ZXNzX18KICAgICAgcmVwbGFjZW1lbnQ6IGJsYWNrYm94LWV4cG9ydGVyOjkxMTUgICMgZXhwb3J0ZXLn
moRzdmMgbmFtZQo=
kind: Secret
metadata:
  creationTimestamp: null
  name: additional-scrape-configs

# 创建Secret到k8s中
[root@k8s-master01 blackbox-exporter-黑盒监控]# kubectl apply -f  additional-scrape-configs.yaml  -n monitoring
```

- 更改prometheus编排文件,增加静态配置路径【重要】

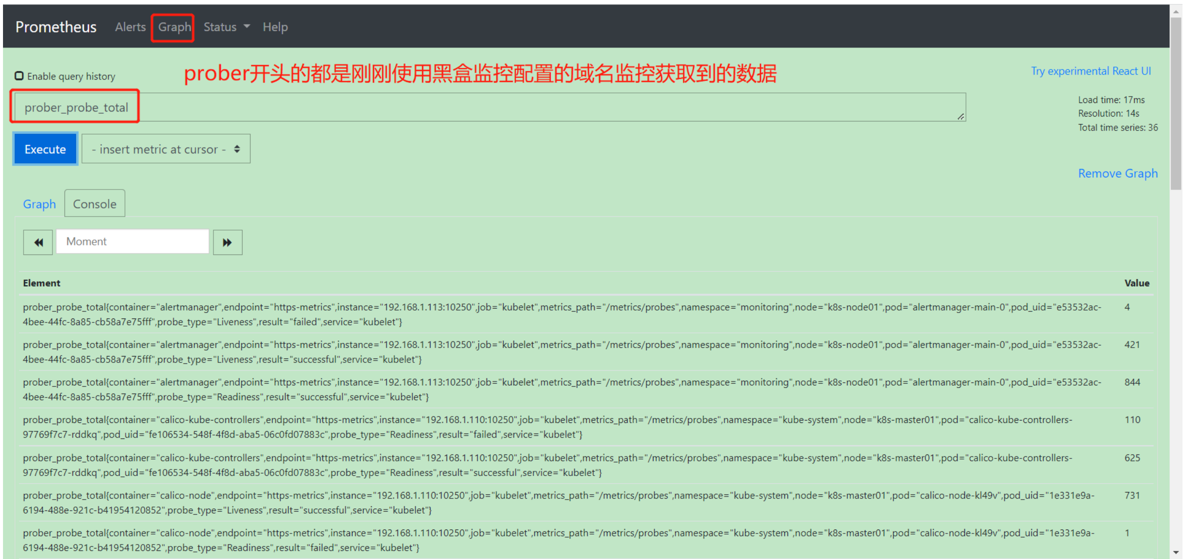
```
[root@k8s-master01 ~]# vim kube-prometheus/manifests/prometheus-prometheus.yaml
apiVersion: monitoring.coreos.com/v1
kind: Prometheus
metadata:
  name: prometheus
  labels:
    prometheus: prometheus
spec:
  replicas: 2
... 加上下面3行
  additionalScrapeConfigs:             # 固定参数
    name: additional-scrape-configs    # 这个是创建的secret的名字
    key: prometheus-additional.yaml    # 这个是静态规则的配置文件名字【全称,包括后缀】
...

# replace刚刚修改的文件
[root@k8s-master01 manifests]# kubectl replace -f  kube-prometheus/manifests/prometheus-prometheus.yaml  -n monitoring

# 手动删除prometheus的所有pod、使之重新构建
[root@k8s-master01 manifests]# kubectl delete po  prometheus-k8s-0 prometheus-k8s-1  -n monitoring
```

- prometheus-ui查看是否成功加载配置



```
      - monitoring
- job_name: blackbox
  honor_timestamps: true
  params:
    module:
    - http_2xx
  scrape_interval: 30s
  scrape_timeout: 10s
  metrics_path: /probe
  scheme: http
  relabel_configs:
  - source_labels: [__address__]
    separator: ;
    regex: (.*)
    target_label: __param_target
    replacement: $1
    action: replace
  - source_labels: [__param_target]
    separator: ;
    regex: (.*)
    target_label: instance
    replacement: $1
    action: replace
  - separator: ;
    regex: (.*)
    target_label: __address__
    replacement: blackbox-exporter:9115
    action: replace
  static_configs:
  - targets:
    - http://www.baidu.com
```

- prometheus-ui查看是否能拿到数据



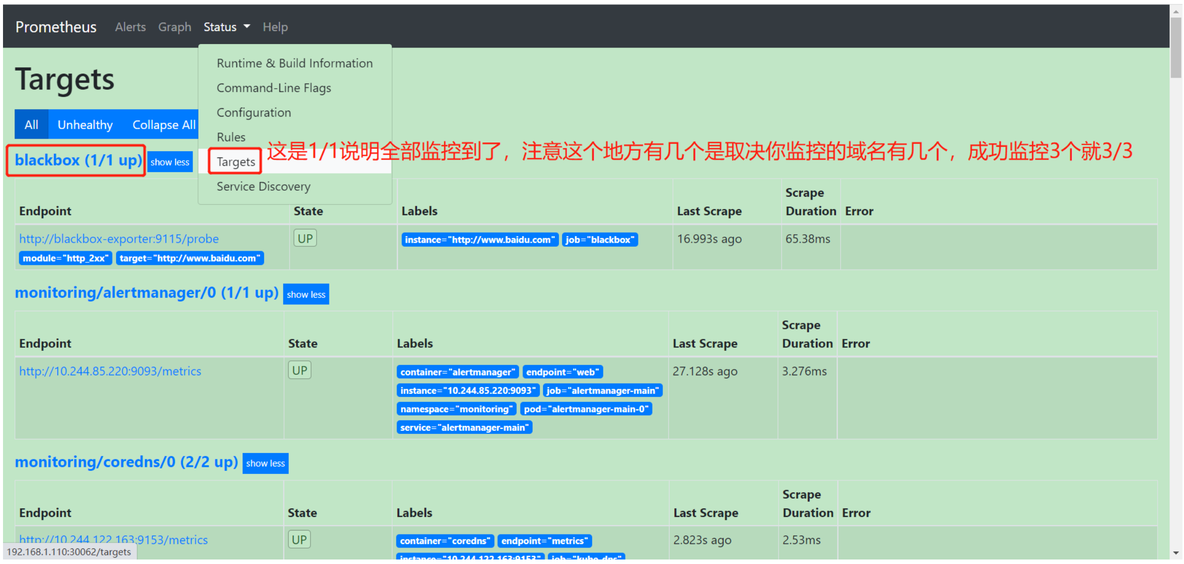
prober开头的都是刚刚使用黑盒监控配置的域名监控获取到的数据

prober_probe_total

Load time: 17ms
Resolution: 14s
Total time series: 36

Element	Value
prober_probe_total{container="alertmanager",endpoint="https-metrics",instance="192.168.1.113:10250",job="kubelet",metrics_path="/metrics/probes",namespace="monitoring",node="k8s-node01",pod="alertmanager-main-0",pod_uid="e53532ac-4bee-44fc-8a85-cb58a7e75fff",probe_type="Liveness",result="failed",service="kubelet"}	4
prober_probe_total{container="alertmanager",endpoint="https-metrics",instance="192.168.1.113:10250",job="kubelet",metrics_path="/metrics/probes",namespace="monitoring",node="k8s-node01",pod="alertmanager-main-0",pod_uid="e53532ac-4bee-44fc-8a85-cb58a7e75fff",probe_type="Liveness",result="successful",service="kubelet"}	421
prober_probe_total{container="alertmanager",endpoint="https-metrics",instance="192.168.1.113:10250",job="kubelet",metrics_path="/metrics/probes",namespace="monitoring",node="k8s-node01",pod="alertmanager-main-0",pod_uid="e53532ac-4bee-44fc-8a85-cb58a7e75fff",probe_type="Readiness",result="successful",service="kubelet"}	844
prober_probe_total{container="calico-kube-controllers",endpoint="https-metrics",instance="192.168.1.110:10250",job="kubelet",metrics_path="/metrics/probes",namespace="kube-system",node="k8s-master01",pod="calico-kube-controllers-97769f7c7-rddkq",pod_uid="fe106534-548f-4f8d-aba5-06c0fd07883c",probe_type="Readiness",result="failed",service="kubelet"}	110
prober_probe_total{container="calico-kube-controllers",endpoint="https-metrics",instance="192.168.1.110:10250",job="kubelet",metrics_path="/metrics/probes",namespace="kube-system",node="k8s-master01",pod="calico-kube-controllers-97769f7c7-rddkq",pod_uid="fe106534-548f-4f8d-aba5-06c0fd07883c",probe_type="Readiness",result="successful",service="kubelet"}	625
prober_probe_total{container="calico-node",endpoint="https-metrics",instance="192.168.1.110:10250",job="kubelet",metrics_path="/metrics/probes",namespace="kube-system",node="k8s-master01",pod="calico-node-kl49v",pod_uid="1e331e9a-6194-488e-921c-b41954120852",probe_type="Liveness",result="successful",service="kubelet"}	731
prober_probe_total{container="calico-node",endpoint="https-metrics",instance="192.168.1.110:10250",job="kubelet",metrics_path="/metrics/probes",namespace="kube-system",node="k8s-master01",pod="calico-node-kl49v",pod_uid="1e331e9a-6194-488e-921c-b41954120852",probe_type="Readiness",result="failed",service="kubelet"}	1

- prometheus-ui查看targets是否ok



Targets

这是1/1说明全部监控到了，注意这个地方有几个是取决你监控的域名有几个，成功监控3个就3/3

blackbox (1/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://blackbox-exporter:9115/probe module="http_2xx" target="http://www.baidu.com"	UP	instance="http://www.baidu.com" job="blackbox"	16.993s ago	65.38ms	

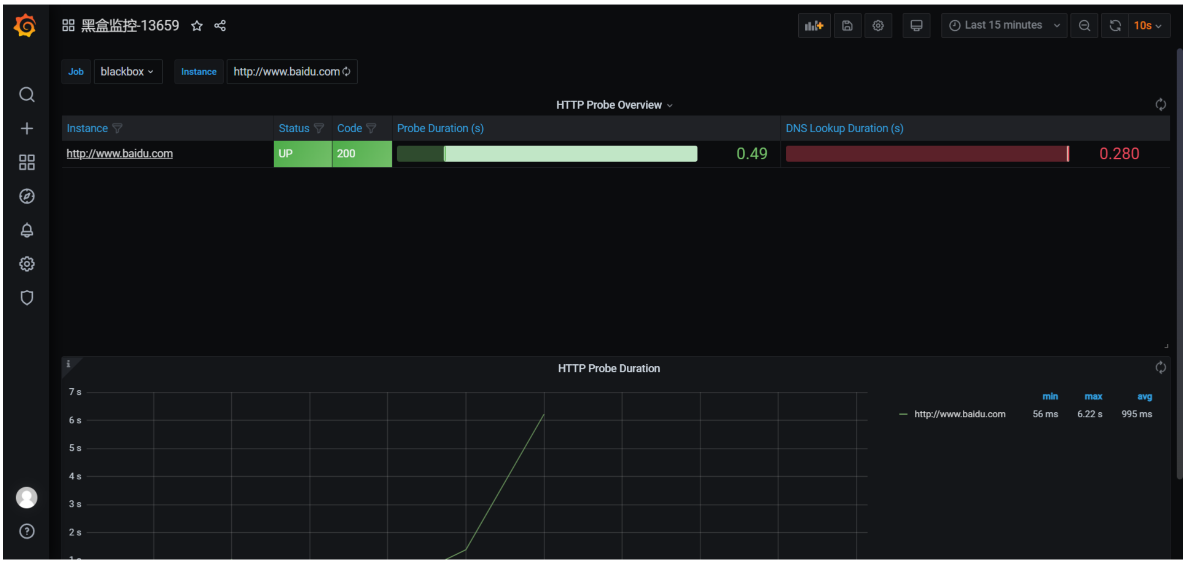
monitoring/alertmanager/0 (1/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://10.244.85.220:9093/metrics	UP	container="alertmanager" endpoint="web" instance="10.244.85.220:9093" job="alertmanager-main" namespace="monitoring" pod="alertmanager-main-0" service="alertmanager-main"	27.128s ago	3.276ms	

monitoring/coredns/0 (2/2 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://10.244.122.163:9153/metrics	UP	container="coredns" endpoint="metrics" instance="10.244.122.163:9153" job="kube-dns"	2.823s ago	2.53ms	

- grafana导入 【https://grafana.com/grafana/dashboards/5345】或者用13659；13659好看些



Job: blackbox Instance: http://www.baidu.com

HTTP Probe Overview

Instance	Status	Code	Probe Duration (s)	DNS Lookup Duration (s)
http://www.baidu.com	UP	200	0.49	0.280

四、域名访问延迟告警

假设需要对域名访问延迟进行监控，访问延迟大于1秒进行告警，此时可以创建一个PrometheusRule如下：

```
# cat blackbox.yaml
apiversion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
labels:
 app.kubernetes.io/component: exporter
 app.kubernetes.io/name: blackbox-exporter
 prometheus: k8s
 role: alert-rules
 name: blackbox
 namespace: monitoring
spec:
 groups:
 - name: blackbox-exporter
 rules:
 - alert: DomainAccessDelayExceeds1s
 annotations:
 description: 域名：{{ $labels.instance }} 探测延迟大于1秒，当前延迟为：{{ $value }}
 summary: 域名探测，访问延迟超过1秒
 expr: sum(probe_http_duration_seconds{job=~"blackbox"}) by (instance) > 1
 for: 10s
 labels:
 severity: warning
 type: blackbox
```