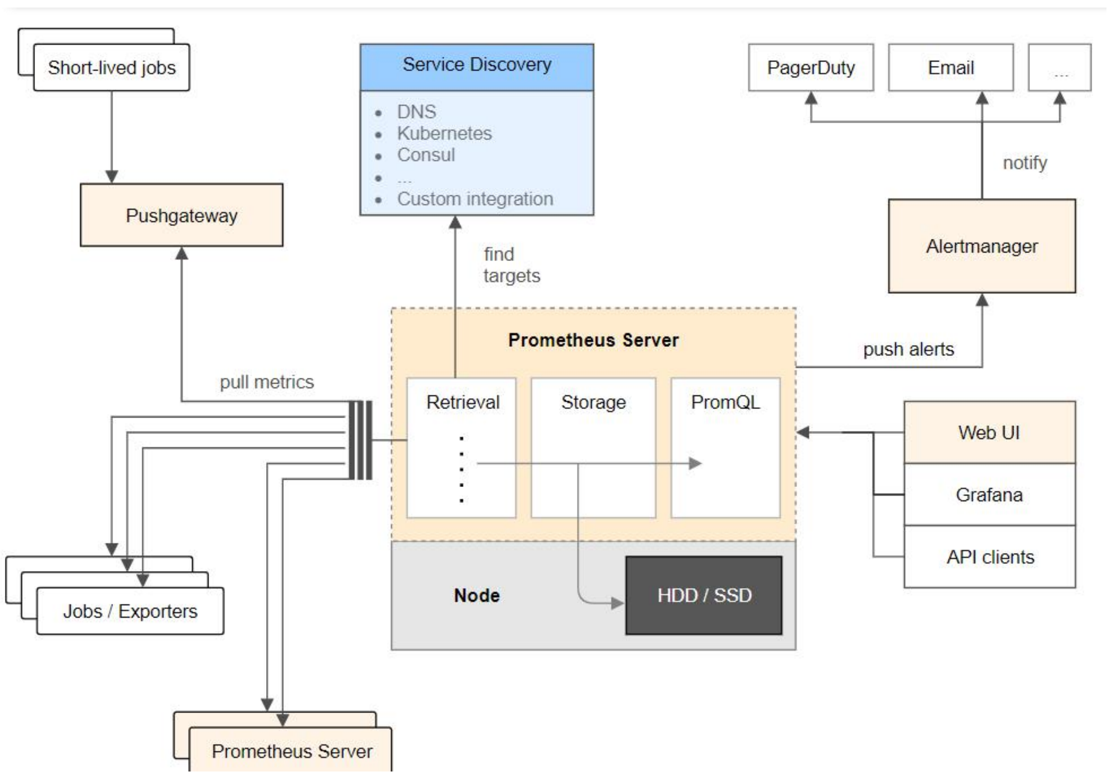
Prometheus 与 Grafana 企业级监控

1、prometheus 架构介绍



1.1　组件说明

prometheus server 是 Prometheus 组件中的核心部分，负责实现对监控数据的获取，存储以及查询。

exporter 简单说是采集端，通过 http 服务的形式保留一个 url 地址，prometheus server 通过访问该 exporter 提供的 endpoint 端点，即可获取到需要采集的监控数据。

AlertManager
在 prometheus 中，支持基于 PromQL 创建告警规则，如果满足定义的规则，则会产生一条告警信息，进入 AlertManager 进行处理。可以集成邮件，微信或者通过 webhook 自定义报警。

Pushgateway
由于 Prometheus 数据采集采用 pull 方式进行设置的， 内置必须保证 prometheus server 和对应的 exporter 必须通信，当网络情况无法直接满足时，可以使用 pushgateway 来进行中转，可以通过 pushgateway 将内部网络数据主动 push 到 gateway 里面去，而 prometheus 采用 pull

方式拉取 pushgateway 中数据。

1.2 总结：

prometheus 负责从 pushgateway 和 job 中采集数据， 存储到后端 Storatge 中，可以通过 PromQL 进行查询， 推送 alerts 信息到 AlertManager。 AlertManager 根据不同的路由规则进行报警通知

2、prometheus 部署

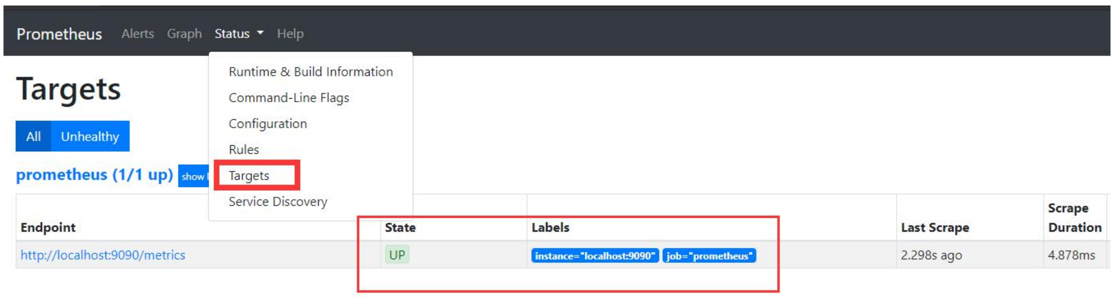
```
[root@jumpserver x]# tar xf prometheus-2.13.1.linux-amd64.tar.gz
[root@docker-3 src]# mv prometheus-2.13.1.linux-amd64 /usr/local/prometheus-2.13.1
[root@docker-3 src]# ln -s /usr/local/prometheus-2.13.1/ /usr/local/prometheus
[root@docker-3 src]#mkdir /usr/local/prometheus/data
```

添加到系统服务

```
[root@jumpserver x]# cat /usr/lib/systemd/system/prometheus.service
[Unit]
Description=https://prometheus.io

[Service]
Restart=on-failure
ExecStart=/usr/local/prometheus/prometheus
--storage.tsdb.path=/usr/local/prometheus/data
--config.file=/usr/local/prometheus/prometheus.yml

[Install]
WantedBy=multi-user.target

[root@docker-3 prometheus]# cp prometheus.yml prometheus.yml.bak

[root@jumpserver x]# systemctl start prometheus    #启动
```

http://ip:9090

访问测试



Targets

prometheus (1/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	2.298s ago	4.878ms

3.Prometheus 配置文件介绍

global： 此片段指定的是 prometheus 的全局配置， 比如采集间隔，抓取超时时间等。
rule_files： 此片段指定报警规则文件， prometheus 根据这些规则信息，会推送报警信息到 alertmanager 中。
scrape_configs: 此片段指定抓取配置，prometheus 的数据采集通过此片段配置。
alerting: 此片段指定报警配置， 这里主要是指定 prometheus 将报警规则推送到指定的 alertmanager 实例地址。
remote_write: 指定后端的存储的写入 api 地址。
remote_read: 指定后端的存储的读取 api 地址。

Global 配置参数

```
  # How frequently to scrape targets by default.
  [ scrape_interval: <duration> | default = 1m ]          # 抓取间隔

  # How long until a scrape request times out.
  [ scrape_timeout: <duration> | default = 10s ]         # 抓取超时时间

  # How frequently to evaluate rules.
  [ evaluation_interval: <duration> | default = 1m ]     # 评估规则间隔
```

scrapy_config 片段主要参数
一个 scrape_config 片段指定一组目标和参数， 目标就是实例，指定采集的端点， 参数描述如何采集这些实例， 主要参数如下

scrape_interval: 抓取间隔,默认继承 global 值。
scrape_timeout: 抓取超时时间,默认继承 global 值。
metric_path: 抓取路径， 默认是/metrics
*_sd_configs: 指定服务发现配置
static_configs: 静态指定服务 job。
relabel_config: relabel 设置。

4、PromQL 介绍

Prometheus 提供了一种名为 PromQL (Prometheus 查询语言)的函数式查询语言，允许用户实时选择和聚合时间序列数据。表达式的结果既可以显示为图形，也可以在 Prometheus 的表达式浏览器中作为表格数据查看，或者通过 HTTP API 由外部系统使用。

运算
乘：*
除：/
加：+

减：-

常用函数

sum() 函数：求出找到所有 value 的值

irate() 函数：统计平均速率

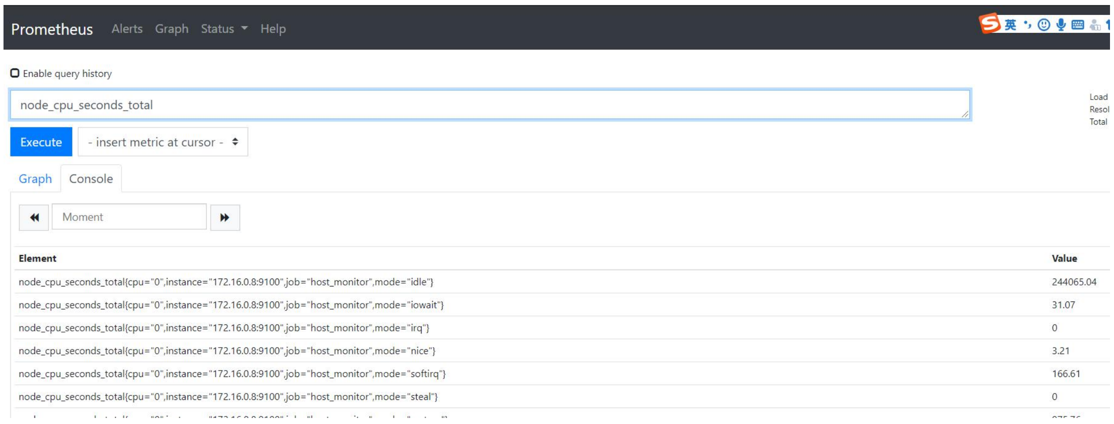
by (标签名)

范围匹配

# 5 分钟之内
[5m]

4.1 查询指定 mertic_name

node_cpu_seconds_total

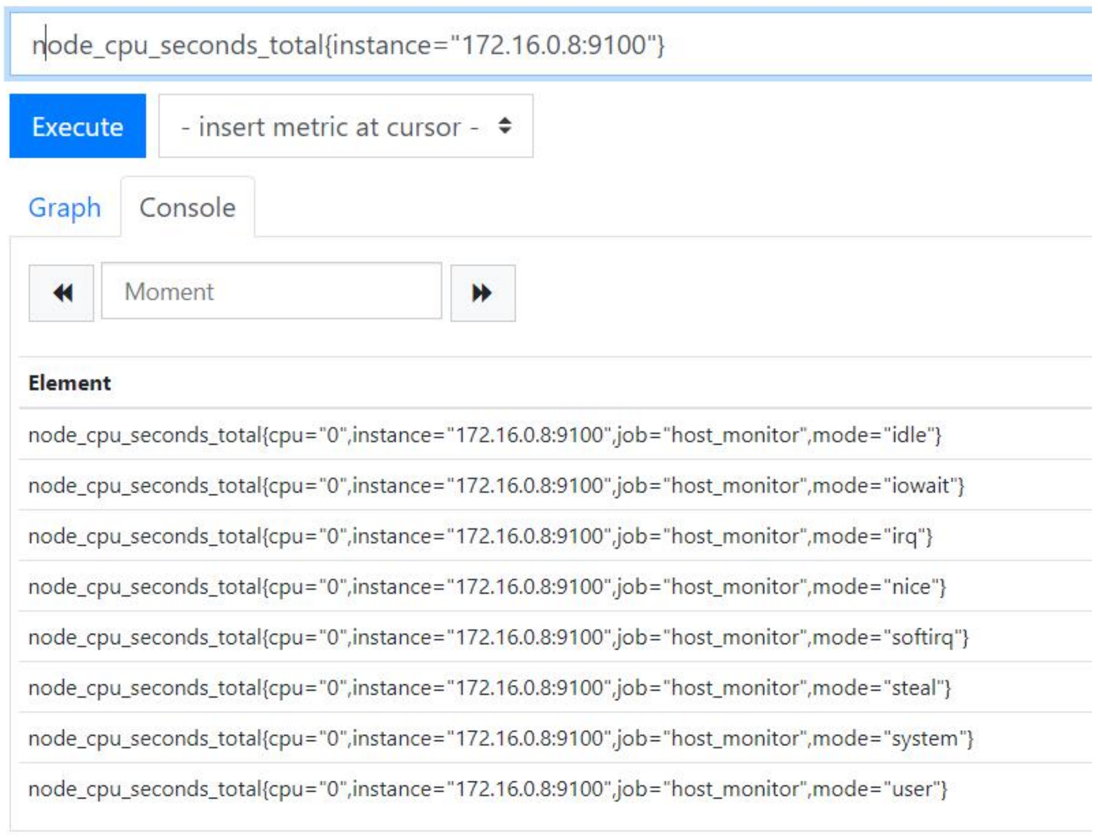


```
node_cpu_seconds_total
```

Element	Value
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="idle"}	244065.04
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="iowait"}	31.07
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="irq"}	0
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="nice"}	3.21
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="softirq"}	166.61
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="steal"}	0

4.2 带标签的查询

node_cpu_seconds_total{instance="172.16.0.8:9100"}



```
node_cpu_seconds_total{instance="172.16.0.8:9100"}
```

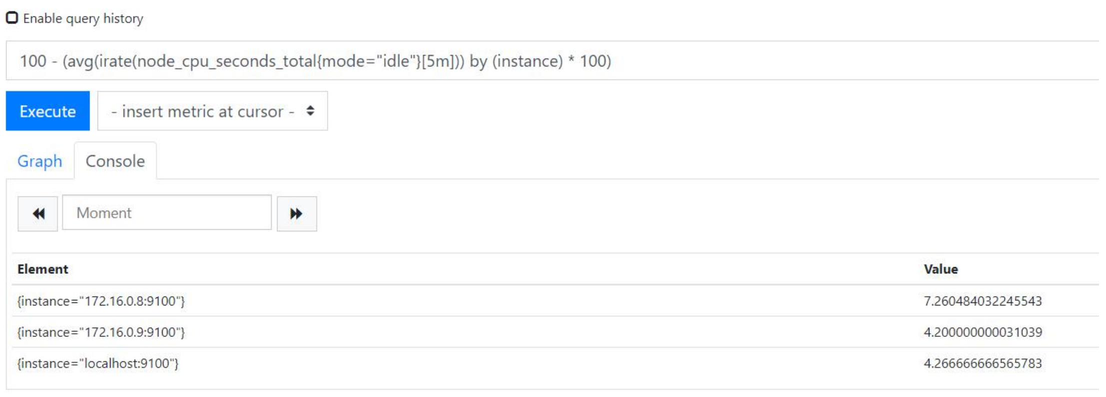
Element
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="idle"}
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="iowait"}
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="irq"}
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="nice"}
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="softirq"}
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="steal"}
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="system"}
node_cpu_seconds_total{cpu="0",instance="172.16.0.8:9100",job="host_monitor",mode="user"}

4.3 多标签查询

node_cpu_seconds_total{instance="172.16.0.8:9100",mode="system"}

4.4 计算 CPU 使用率

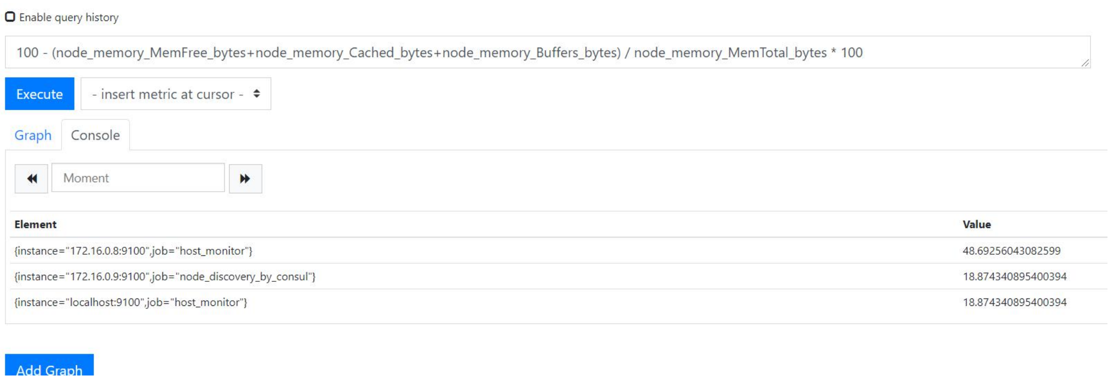
100 - (avg(irate(node_cpu_seconds_total{mode="idle"}[5m])) by (instance) * 100)



```
100 - (avg(irate(node_cpu_seconds_total{mode="idle"}[5m])) by (instance) * 100)
```

Element	Value
{instance="172.16.0.8:9100"}	7.260484032245543
{instance="172.16.0.9:9100"}	4.200000000031039
{instance="localhost:9100"}	4.266666666565783

4.5 计算内存使用率

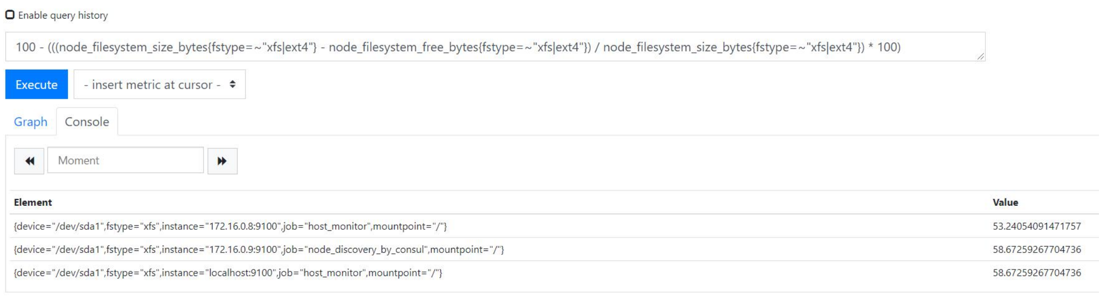


```
100 - (node_memory_MemFree_bytes+node_memory_Cached_bytes+node_memory_Buffers_bytes) / node_memory_MemTotal_bytes * 100
```

Element	Value
{instance="172.16.0.8:9100",job="host_monitor"}	48.69256043082599
{instance="172.16.0.9:9100",job="node_discovery_by_consul"}	18.874340895400394
{instance="localhost:9100",job="host_monitor"}	18.874340895400394

4.6 计算磁盘使用率

100 - (((node_filesystem_size_bytes{fstype=~"xfs|ext4"} - node_filesystem_free_bytes{fstype=~"xfs|ext4"}) / node_filesystem_size_bytes{fstype=~"xfs|ext4"}) * 100)



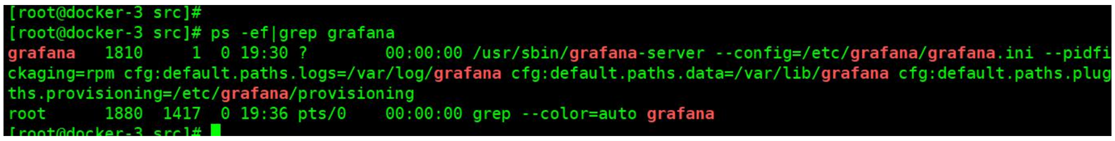
```
100 - (((node_filesystem_size_bytes{fstype=~"xfs|ext4"} - node_filesystem_free_bytes{fstype=~"xfs|ext4"}) / node_filesystem_size_bytes{fstype=~"xfs|ext4"}) * 100)
```

Element	Value
{device="/dev/sda1",fstype="xfs",instance="172.16.0.8:9100",job="host_monitor",mountpoint="/"}	53.24054091471757
{device="/dev/sda1",fstype="xfs",instance="172.16.0.9:9100",job="node_discovery_by_consul",mountpoint="/"}	58.67259267704736
{device="/dev/sda1",fstype="xfs",instance="localhost:9100",job="host_monitor",mountpoint="/"}	58.67259267704736

5、部署 grafana 及接入 prometheus

[root@jumpserver x]# yum localinstall grafana-6.4.3-1.x86_64.rpm -y

[root@jumpserver x]# systemctl start grafana-server



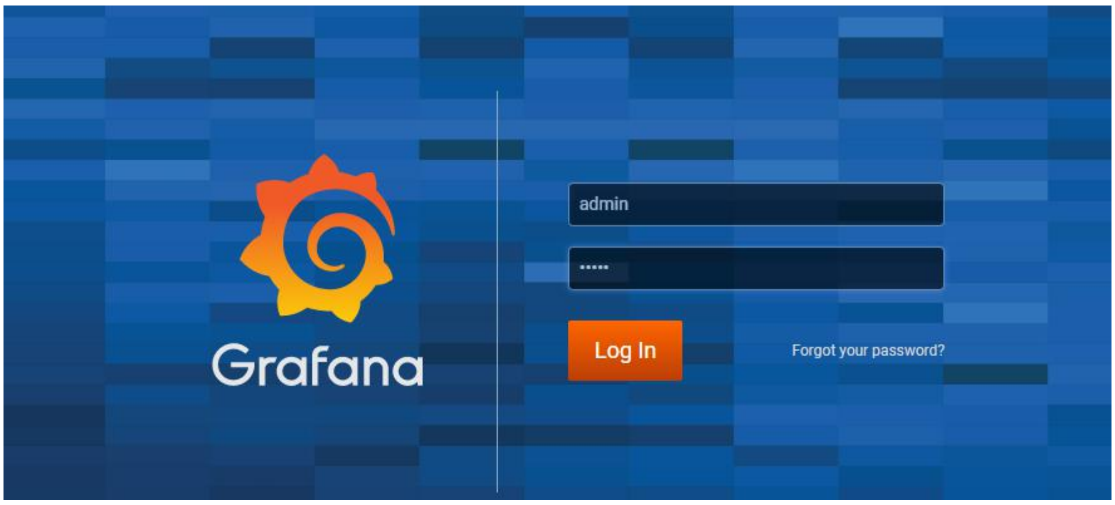
```
[root@docker-3 src]#
[root@docker-3 src]# ps -ef|grep grafana
grafana   1810     1  0 19:30 ?        00:00:00 /usr/sbin/grafana-server --config=/etc/grafana/grafana.ini --pidfi
ckaging=rpm cfg:default.paths.logs=/var/log/grafana cfg:default.paths.data=/var/lib/grafana cfg:default.paths.plug
ths.provisioning=/etc/grafana/provisioning
root      1880  1417  0 19:36 pts/0    00:00:00 grep --color=auto grafana
[root@docker-3 src]#
```

访问

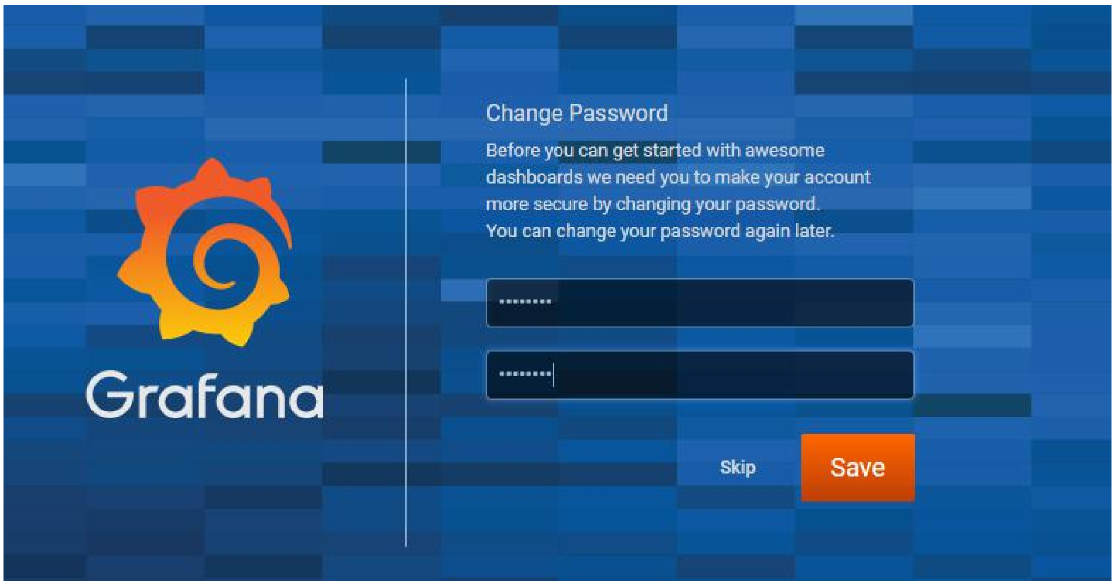
Http://ip:3000　默认用户名 密码　admin/admin

Grafana 接入 prometheus 数据源

第一步添加 prometheus 数据源

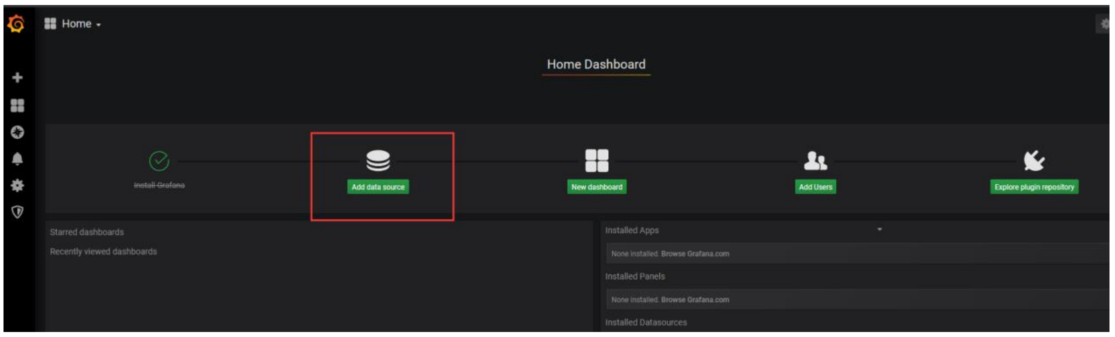


登录后提示需要修改密码



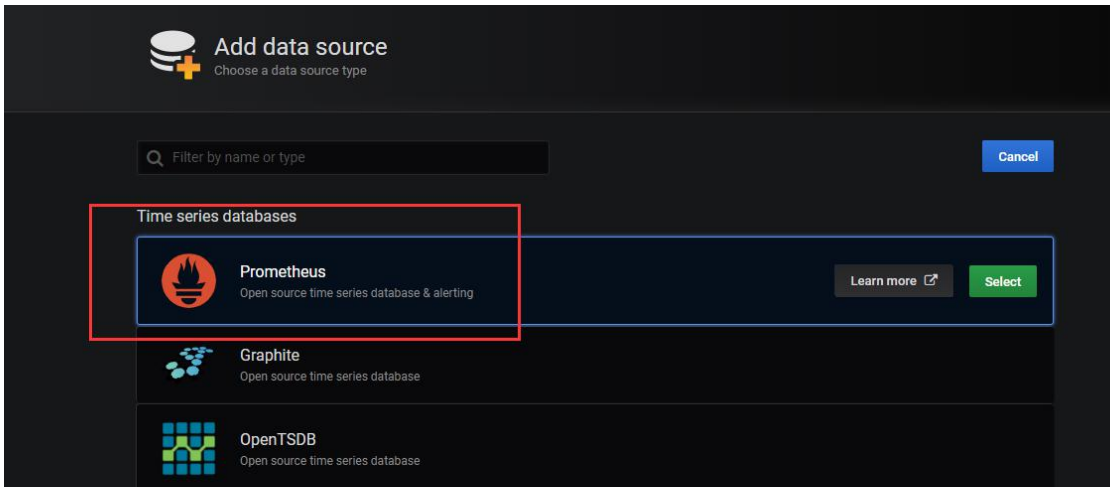
Change Password

Before you can get started with awesome dashboards we need you to make your account more secure by changing your password.
You can change your password again later.



Grafana 接入 prometheus 数据源

第一步添加 prometheus 数据源



Add data source
Choose a data source type

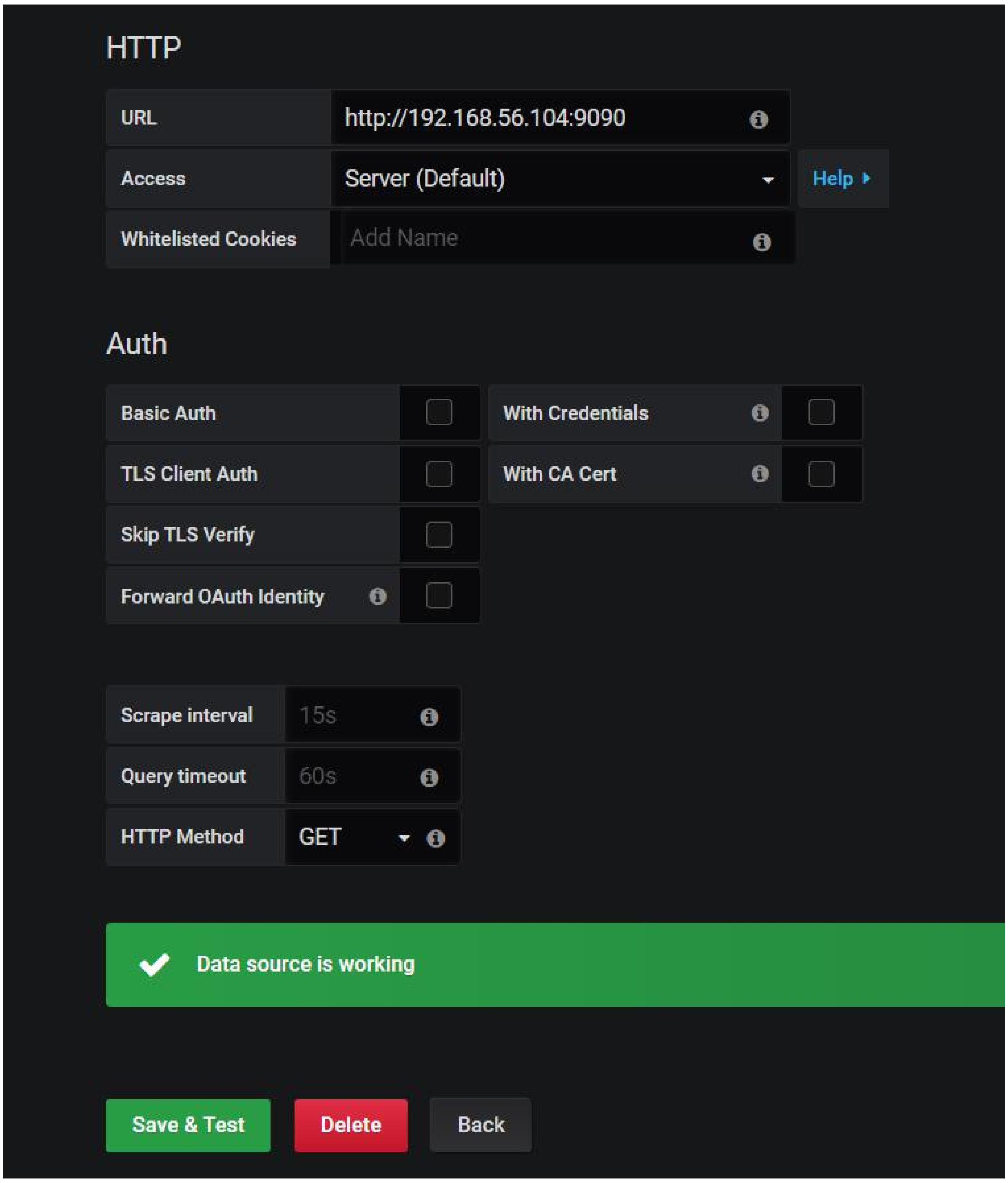
Filter by name or type
Cancel

Time series databases

Prometheus
Open source time series database & alerting
Learn more
Select

Graphite
Open source time series database

OpenTSDB
Open source time series database



HTTP

URL	http://192.168.56.104:9090
Access	Server (Default)
Whitelisted Cookies	Add Name

Auth

Basic Auth	☐	With Credentials	☐
TLS Client Auth	☐	With CA Cert	☐
Skip TLS Verify	☐		
Forward OAuth Identity	☐		

Scrape interval	15s
Query timeout	60s
HTTP Method	GET

Data source is working

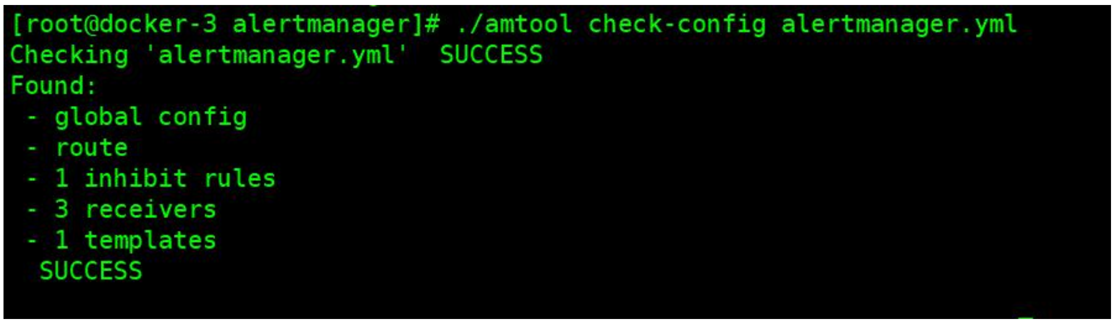
点击 save & test 这步必须通过

6、告警模块 alertermanager 部署

```
[root@docker-3 src]# tar xf alertmanager-0.20.0.linux-amd64.tar.gz
[root@docker-3 src]# mv alertmanager-0.20.0.linux-amd64 /usr/local/alertmanager-0.20
[root@docker-3 src]# ln -s /usr/local/alertmanager-0.20/ /usr/local/alertmanager

[root@docker-3 prometheus]# cat /usr/lib/systemd/system/alertmanager.service
[Unit]
Description=alertmanager System
Documentation=alertmanager System

[Service]
ExecStart=/usr/local/alertmanager/alertmanager \
  --config.file=/usr/local/alertmanager/alertmanager.yml

[Install]
WantedBy=multi-user.target
[root@docker-3 alertmanager]# cp alertmanager.yml alertmanager.yml.bak
```

检查语法

```
[root@docker-3 alertmanager]# ./amtool check-config alertmanager.yml
```



```
[root@docker-3 alertmanager]# ./amtool check-config alertmanager.yml
Checking 'alertmanager.yml'  SUCCESS
Found:
 - global config
 - route
 - 1 inhibit rules
 - 3 receivers
 - 1 templates
  SUCCESS
```

7、prometheus 告警实战

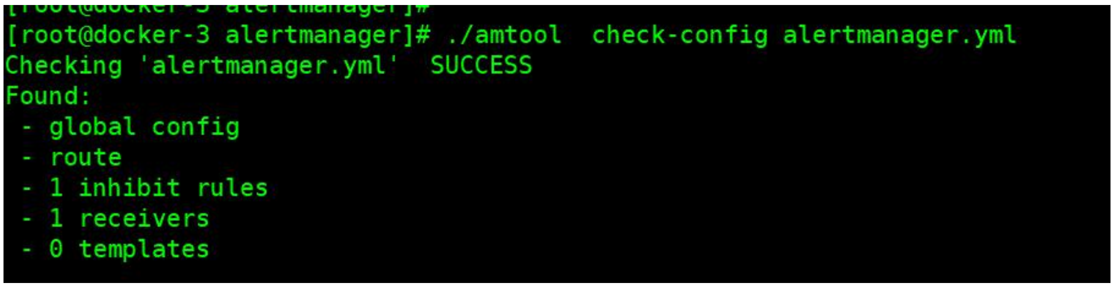
7.1 邮件告警

```
[root@docker-3 alertmanager]# cat alertmanager.yml
```

```
global:
  resolve_timeout: 5m
  smtp_smarthost: 'smtp.163.com:25'
  smtp_from: 'jumpservervip@163.com'
  smtp_auth_username: 'jumpservervip@163.com'
  smtp_auth_password: 'xxx'
  smtp_require_tls: false

route:
  group_by: ['alertname']
  group_wait: 10s
  group_interval: 10s
  repeat_interval: 1h
  receiver: 'email'
receivers:
- name: 'email'
  email_configs:
  - to: 'jumpservervip@126.com'
    send_resolved: true
inhibit_rules:
  - source_match:
      severity: 'critical'
    target_match:
      severity: 'warning'
    equal: ['alertname', 'dev', 'instance']
```

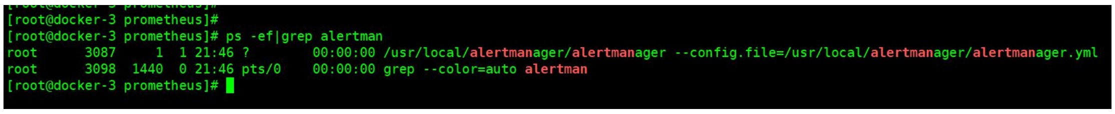
检查配置

```
[root@docker-3 alertmanager]# ./amtool   check-config alertmanager.yml
```



```
[root@docker-3 alertmanager]# ./amtool  check-config alertmanager.yml
Checking 'alertmanager.yml'  SUCCESS
Found:
 - global config
 - route
 - 1 inhibit rules
 - 1 receivers
 - 0 templates
```

```
[root@docker-3 alertmanager]# systemctl start alertmanager
```



```
[root@docker-3 prometheus]#
[root@docker-3 prometheus]# ps -ef|grep alertman
root      3087     1  1 21:46 ?        00:00:00 /usr/local/alertmanager/alertmanager --config.file=/usr/local/alertmanager/alertmanager.yml
root      3098  1440  0 21:46 pts/0    00:00:00 grep --color=auto alertman
[root@docker-3 prometheus]#
```

修改 prometheus 配置文件

```
[root@docker-3 alertmanager]# vim /usr/local/prometheus/prometheus.yml
```

1、修改 prometheus.yml 的 alerting 部分

```
# Alertmanager configuration
alerting:
  alertmanagers:
  - static_configs:
    - targets:
       - 172.16.0.9:9093
```

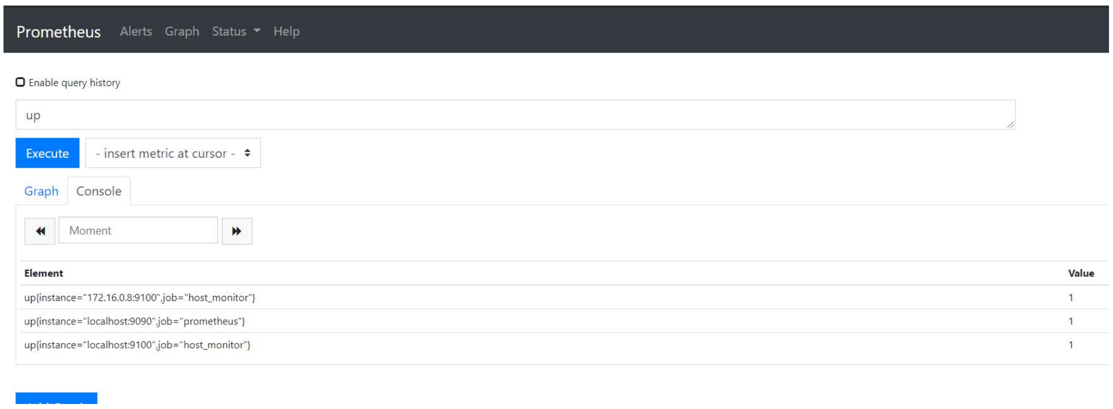
2、定义告警文件：

```
rule_files:
  - rules/*.yml
```

7.2 编写告警规则

```
[root@docker-3 alertmanager]# cd /usr/local/prometheus
[root@docker-3 alertmanager]# mkdir rules
[root@docker-3 alertmanager]# cd rules/
[root@docker-3 rules]# cat host_monitor.yml
groups:
- name: node-up
  rules:
  - alert: node-up
    expr: up == 0
    for: 15s
    labels:
      severity: 1
      team: node
    annotations:
      summary: "{{$labels.instance}}Instance has been down for more than 5 minutes"
```

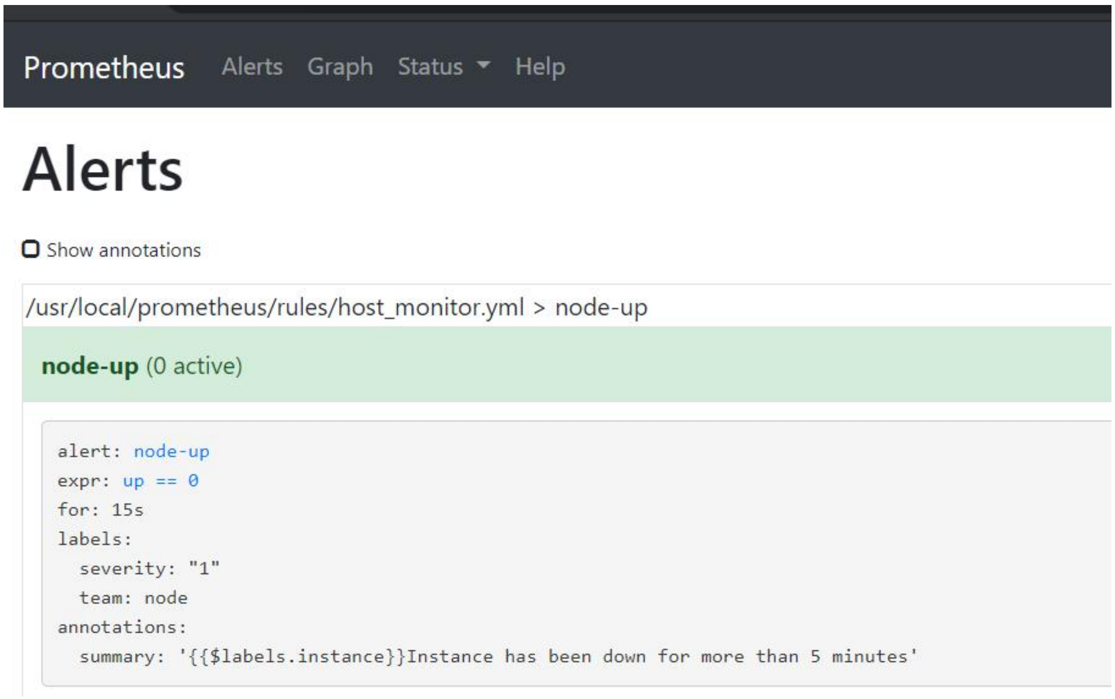
alert：告警规则的名称。
# expr：基于 PromQL 表达式告警触发条件，用于计算是否有时间序列满足该条件。
# for：评估等待时间，可选参数。用于表示只有当触发条件持续一段时间后才发送告警。在等待期间新产生告警的状态为 pending。
# labels：自定义标签，允许用户指定要附加到告警上的一组附加标签。
# annotations：用于指定一组附加信息，比如用于描述告警详细信息的文字等，annotations 的内容在告警产生时会一同作为参数发送到 Alertmanager。
# summary 描述告警的概要信息，description 用于描述告警的详细信息。
# 同时 Alertmanager 的 UI 也会根据这两个标签值，显示告警信息。



```
up
```

```
Element                                                  Value
up{instance="172.16.0.8:9100",job="host_monitor"}        1
up{instance="localhost:9090",job="prometheus"}           1
up{instance="localhost:9100",job="host_monitor"}         1
```

[root@docker-3 rules]# systemctl restart prometheus



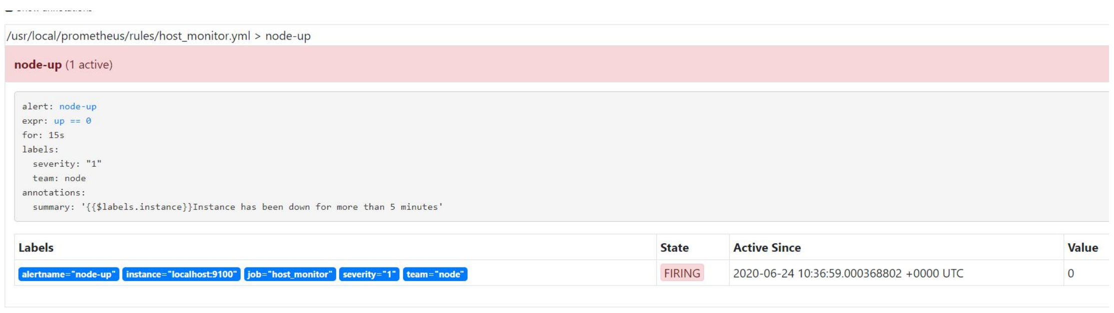
/usr/local/prometheus/rules/host_monitor.yml > node-up

node-up (0 active)

```
alert: node-up
expr: up == 0
for: 15s
labels:
  severity: "1"
  team: node
annotations:
  summary: '{{$labels.instance}}Instance has been down for more than 5 minutes'
```

状态说明 Prometheus Alert 告警状态有三种状态：Inactive、Pending、Firing。
1、Inactive：非活动状态，表示正在监控，但是还未有任何警报触发。
2、Pending：表示这个警报必须被触发。由于警报可以被分组、压抑/抑制或静默/静音，所以等待验证，一旦所有的验证都通过，则将转到 Firing 状态。
3、Firing：将警报发送到 AlertManager，它将按照配置将警报的发送给所有接收者。一旦警报解除，则将状态转到 Inactive，如此循环。

[root@docker-3 rules]# systemctl stop node_exporter ##停止观察

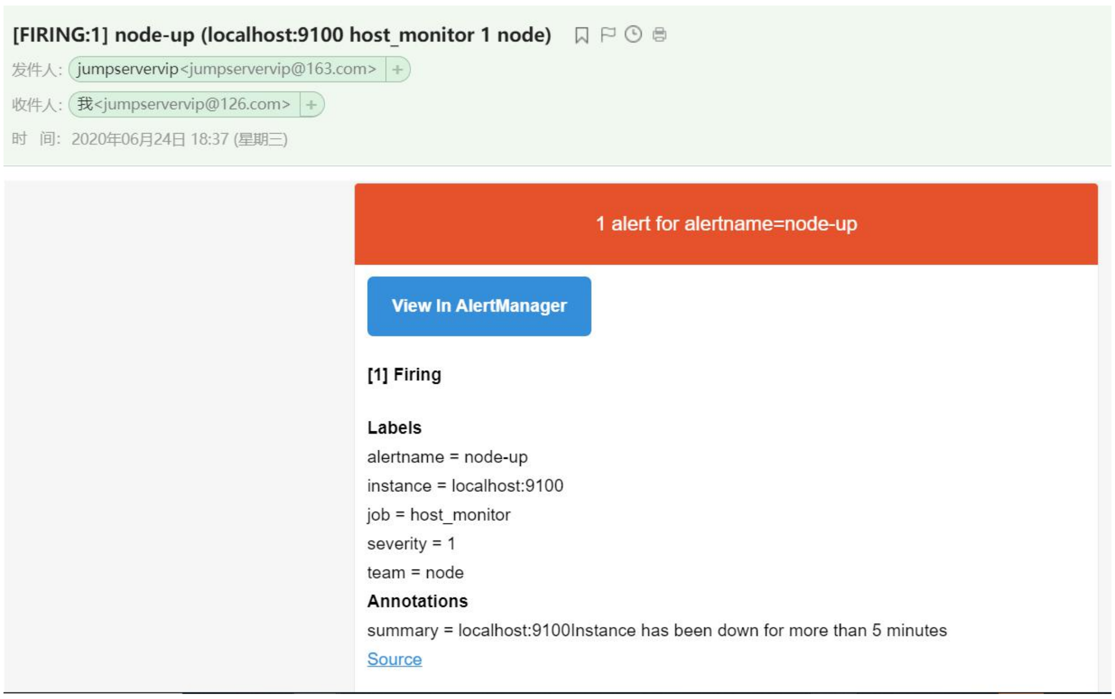


/usr/local/prometheus/rules/host_monitor.yml > node-up

node-up (1 active)

```
alert: node-up
expr: up == 0
for: 15s
labels:
  severity: "1"
  team: node
annotations:
  summary: '{{$labels.instance}}Instance has been down for more than 5 minutes'
```

Labels	State	Active Since	Value
alertname="node-up" instance="localhost:9100" job="host_monitor" severity="1" team="node"	FIRING	2020-06-24 10:36:59.000368802 +0000 UTC	0



[FIRING:1] node-up (localhost:9100 host_monitor 1 node)

发件人: jumpservervip<jumpservervip@163.com>

收件人: 我<jumpservervip@126.com>

时 间: 2020年06月24日 18:37 (星期三)

1 alert for alertname=node-up

View In AlertManager

[1] Firing

Labels

alertname = node-up

instance = localhost:9100

job = host_monitor

severity = 1

team = node

Annotations

summary = localhost:9100Instance has been down for more than 5 minutes

Source

7.3 优化告警模板

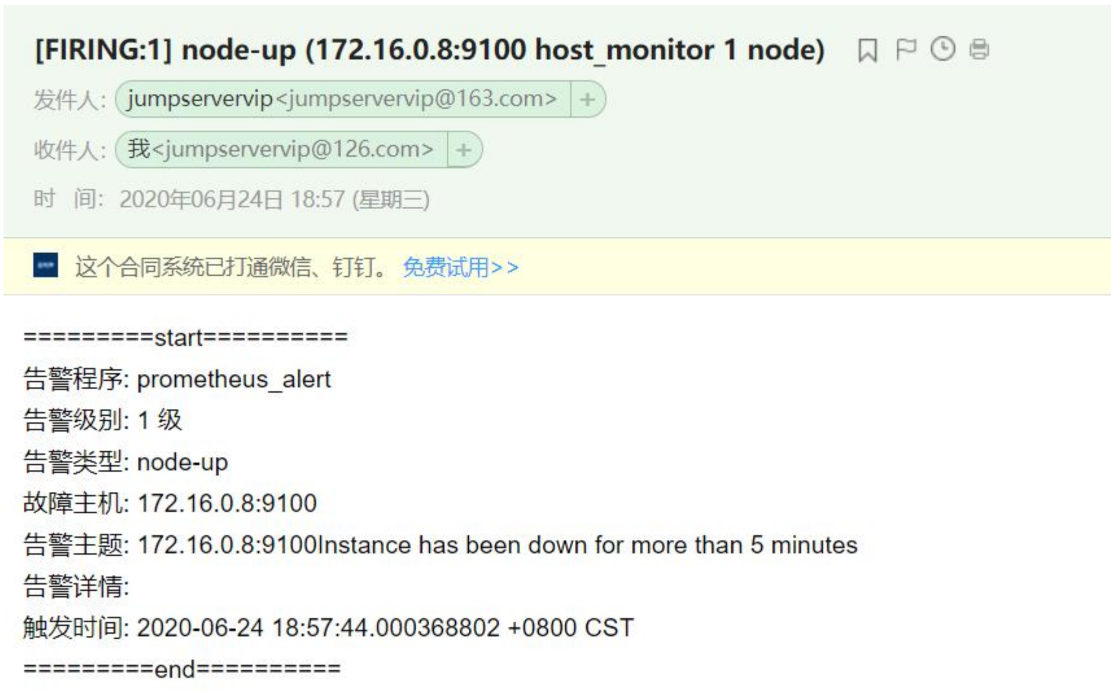
1、新建模板文件

```
[root@docker-3 rules]# cat /usr/local/alertmanager/email.tmpl
{{ define "email.to.html" }}
{{ range .Alerts }}
=========start==========<br>
告警程序: prometheus_alert <br>
告警级别: {{ .Labels.severity }} 级 <br>
告警类型: {{ .Labels.alertname }} <br>
故障主机: {{ .Labels.instance }} <br>
告警主题: {{ .Annotations.summary }} <br>
告警详情: {{ .Annotations.description }} <br>
触发时间: {{ .StartsAt }} <br>
```

```
=========end==========<br>
{{ end }}
{{ end }}
```

2、修改配置文件使用模板

```
[root@docker-3 rules]# cat /usr/local/alertmanager/alertmanager.yml
global:
  resolve_timeout: 5m
  smtp_smarthost: 'smtp.163.com:25'
  smtp_from: 'jumpservervip@163.com'
  smtp_auth_username: 'jumpservervip@163.com'
  smtp_auth_password: 'xxx'
  smtp_require_tls: false

route:
  group_by: ['alertname']
  group_wait: 10s
  group_interval: 10s
  repeat_interval: 1h
  receiver: 'email'
receivers:
- name: 'email'
  email_configs:
  - to: 'jumpservervip@126.com'
    html: '{{ template "email.to.html" . }}'    ##使用模板的方式发送
    send_resolved: true
inhibit_rules:
  - source_match:
      severity: 'critical'
    target_match:
      severity: 'warning'
    equal: ['alertname', 'dev', 'instance']
```


[FIRING:1] node-up (172.16.0.8:9100 host_monitor 1 node)

发件人: jumpservervip<jumpservervip@163.com>

收件人: 我<jumpservervip@126.com>

时 间: 2020年06月24日 18:57 (星期三)

这个合同系统已打通微信、钉钉。 免费试用>>

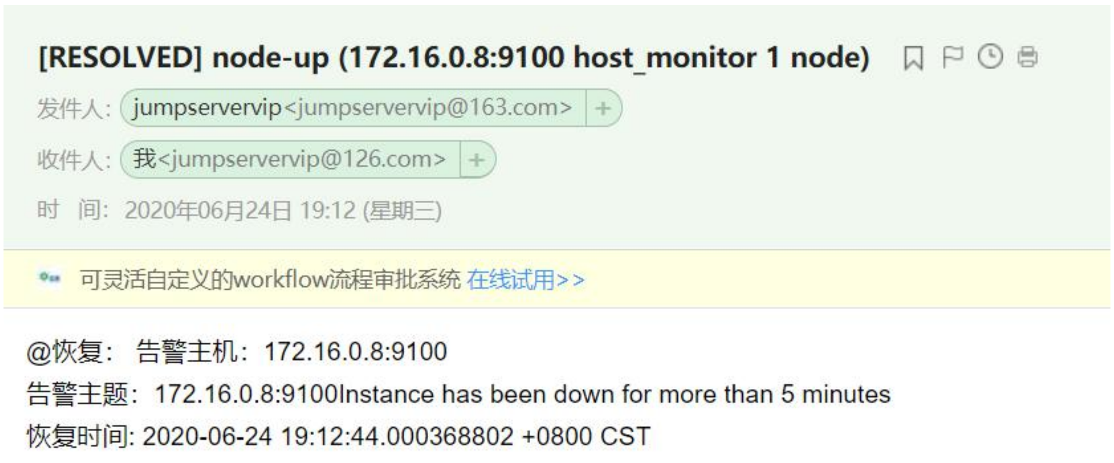
```
=========start==========
告警程序: prometheus_alert
告警级别: 1 级
告警类型: node-up
故障主机: 172.16.0.8:9100
告警主题: 172.16.0.8:9100Instance has been down for more than 5 minutes
告警详情:
触发时间: 2020-06-24 18:57:44.000368802 +0800 CST
=========end==========
```

告警恢复

在配置的时候，加上：send_resolved: true

1、修改模板添加恢复信息

```
[root@docker-3 rules]# cat /usr/local/alertmanager/email.tmpl
{{ define "email.to.html" }}
{{ if gt (len .Alerts.Firing) 0 }}{{ range .Alerts }}
@告警
告警程序: prometheus_alert <br>
告警级别: {{ .Labels.severity }}  级  <br>
告警类型: {{ .Labels.alertname }} <br>
故障主机: {{ .Labels.instance }} <br>
告警主题: {{ .Annotations.summary }} <br>
告警详情: {{ .Annotations.description }} <br>
触发时间: {{ .StartsAt    }} <br>
{{ end }}
{{ end }}
{{ if gt (len .Alerts.Resolved) 0 }}{{ range .Alerts }}
@恢复：
告警主机：{{ .Labels.instance }} <br>
告警主题：{{ .Annotations.summary }} <br>
恢复时间: {{ .EndsAt    }} <br>
{{ end }}
{{ end }}
{{ end }}
```


[RESOLVED] node-up (172.16.0.8:9100 host_monitor 1 node)

发件人: jumpservervip<jumpservervip@163.com>

收件人: 我<jumpservervip@126.com>

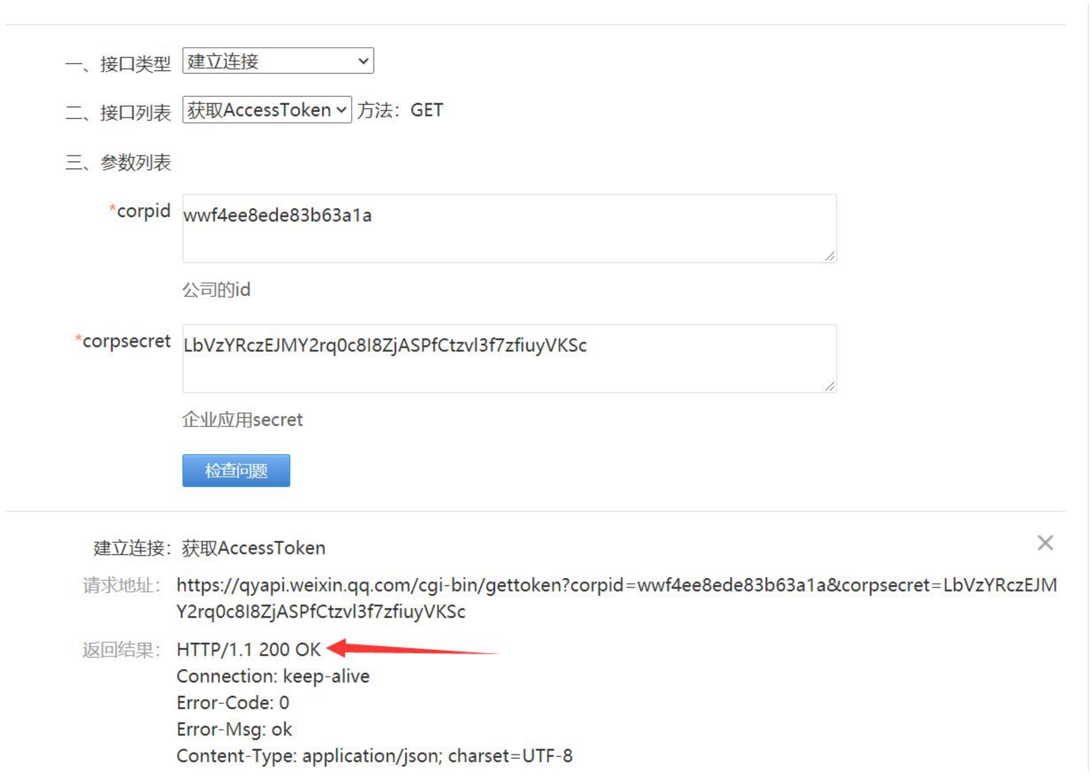
时 间: 2020年06月24日 19:12 (星期三)

可灵活自定义的workflow流程审批系统 在线试用>>

@恢复： 告警主机：172.16.0.8:9100
告警主题：172.16.0.8:9100Instance has been down for more than 5 minutes
恢复时间: 2020-06-24 19:12:44.000368802 +0800 CST

7.4 企业微信告警

测试账户可用性
https://work.weixin.qq.com/api/devtools/devtool.php


一、接口类型 建立连接

二、接口列表 获取AccessToken 方法：GET

三、参数列表

*corpid wwf4ee8ede83b63a1a

公司的id

*corpsecret LbVzYRczEJMY2rq0c8l8ZjASPfCtzvl3f7zfiuyVKSc

企业应用secret

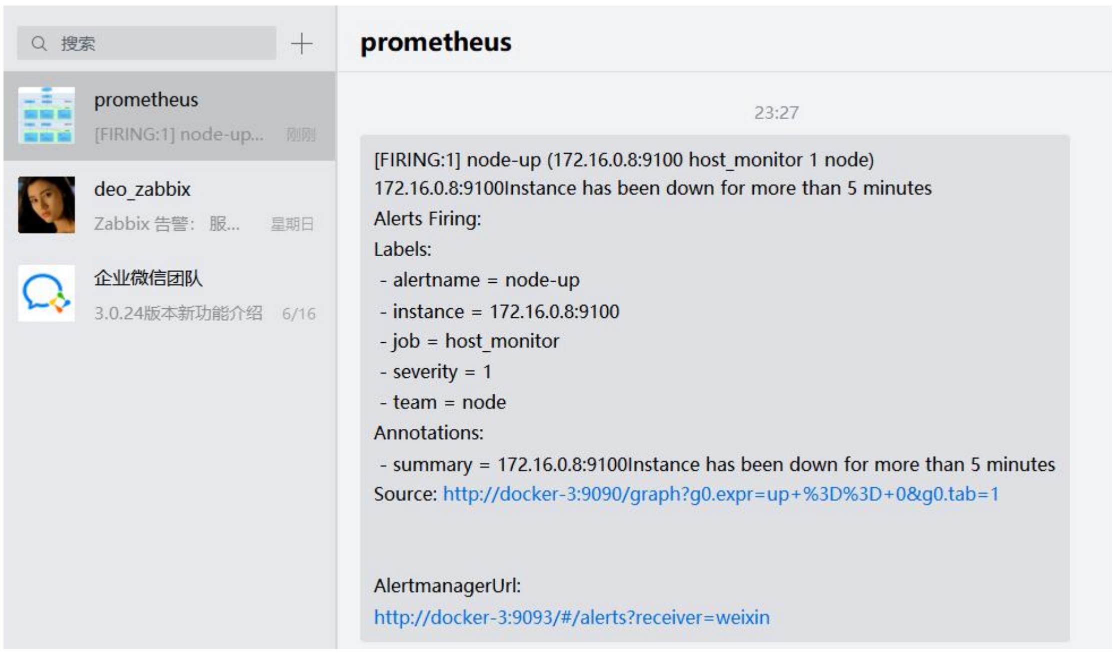
建立连接：获取AccessToken

请求地址：https://qyapi.weixin.qq.com/cgi-bin/gettoken?corpid=wwf4ee8ede83b63a1a&corpsecret=LbVzYRczEJMY2rq0c8l8ZjASPfCtzvl3f7zfiuyVKSc

返回结果：
```
HTTP/1.1 200 OK
Connection: keep-alive
Error-Code: 0
Error-Msg: ok
Content-Type: application/json; charset=UTF-8
```

corp_id: 企业微信账号唯一 ID， 可以在我的企业中查看。
to_party: 需要发送的组(部门)。
agent_id: 第三方企业应用的 ID

api_secret: 第三方企业应用的密钥



```
[FIRING:1] node-up (172.16.0.8:9100 host_monitor 1 node)
172.16.0.8:9100Instance has been down for more than 5 minutes
Alerts Firing:
Labels:
 - alertname = node-up
 - instance = 172.16.0.8:9100
 - job = host_monitor
 - severity = 1
 - team = node
Annotations:
 - summary = 172.16.0.8:9100Instance has been down for more than 5 minutes
Source: http://docker-3:9090/graph?g0.expr=up+%3D%3D+0&g0.tab=1


AlertmanagerUrl:
http://docker-3:9093/#/alerts?receiver=weixin
```

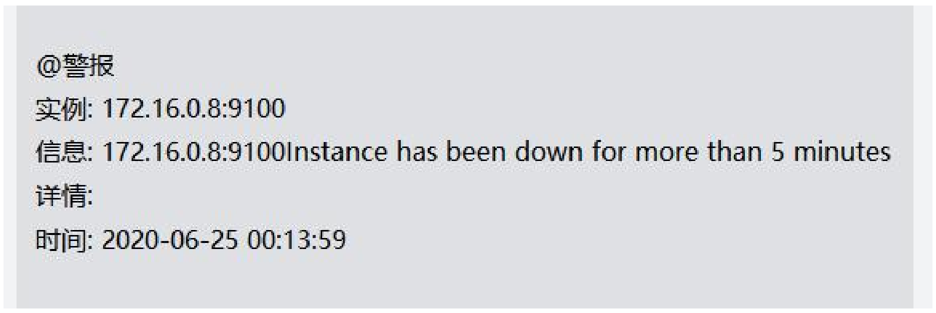
修改模板

```
[root@docker-3 alertmanager]# cat /usr/local/alertmanager/wechat.tmpl
{{ define "wechat.tmpl" }}
{{- if gt (len .Alerts.Firing) 0 -}}{{ range .Alerts }}
@警报
实例: {{ .Labels.instance }}
信息: {{ .Annotations.summary }}
详情: {{ .Annotations.description }}
时间: {{ .StartsAt.Format "2006-01-02 15:04:05" }}
{{ end }}{{ end -}}
{{- if gt (len .Alerts.Resolved) 0 -}}{{ range .Alerts }}
@恢复
实例: {{ .Labels.instance }}
信息: {{ .Annotations.summary }}
时间: {{ .StartsAt.Format "2006-01-02 15:04:05" }}
恢复: {{ .EndsAt.Format "2006-01-02 15:04:05" }}
{{ end }}{{ end -}}
{{- end }}
```

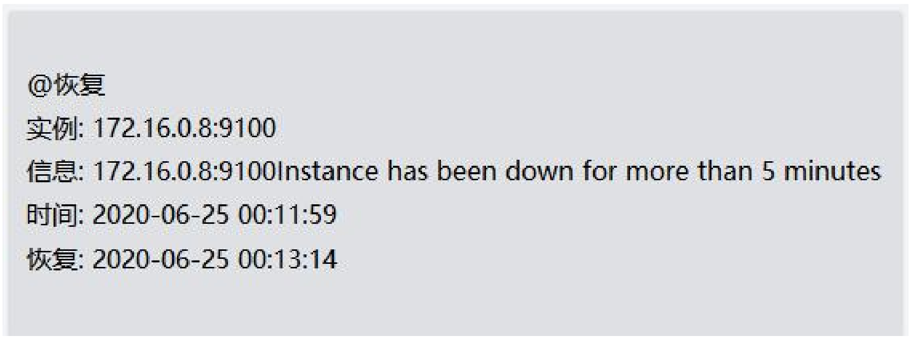
修改配置

```
[root@docker-3 alertmanager]# cat /usr/local/alertmanager/alertmanager.yml
global:
  resolve_timeout: 5m
```

```
templates:
  - '/usr/local/alertmanager/wechat.tmpl'

route:
  group_by: ['alertname']
  group_wait: 10s
  group_interval: 10s
  repeat_interval: 1h
  receiver: 'wechat'

receivers:

- name: 'wechat'
  wechat_configs:
  - corp_id: 'wwf4ee8ede83b63a1a'
    to_party: '1'
    agent_id: '1000003'
    api_secret: 'LbVzYRczEJMY2rq0c8I8ZjASPfCtzvl3f7zfiuyVKSc'
    send_resolved: true
    message:   '{{ template "wechat.tmpl" . }}'



inhibit_rules:
  - source_match:
      severity: 'critical'
    target_match:
      severity: 'warning'
    equal: ['alertname', 'dev', 'instance']
```



```
@警报
实例: 172.16.0.8:9100
信息: 172.16.0.8:9100Instance has been down for more than 5 minutes
详情:
时间: 2020-06-25 00:13:59
```

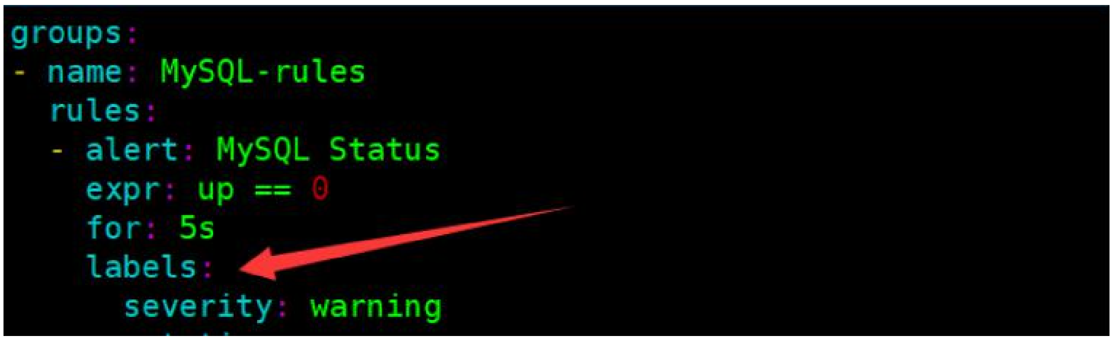


```
@恢复
实例: 172.16.0.8:9100
信息: 172.16.0.8:9100Instance has been down for more than 5 minutes
时间: 2020-06-25 00:11:59
恢复: 2020-06-25 00:13:14
```

7.5 告警的标签、路由、分组

标签：给每个监控项添加标签
/usr/local/prometheus/rules/mysql.yml
如下面的标签定义为

```
    labels:
      severity: warning
```



```
groups:
- name: MySQL-rules
  rules:
  - alert: MySQL Status
    expr: up == 0
    for: 5s
    labels:
      severity: warning
```

定义两个告警等级

```
  - source_match:
      severity: 'critical'     ###严重等级，发给 leader
    target_match:
      severity: 'warning'    ###一般告警，发给普通运维开发即可
```

路由

```
 routes:
  - match:
      severity: critical
    receiver: 'leader'
    continue: true
  - match_re:
      severity: ^(warning|critical)$
    receiver: 'devops'
    continue: true
```

定 义 路 由 匹 配 规 则 ， 匹 配 到 severity: critical ， 发 送 给 leader ， 匹 配 到 severity: ^(warning|critical)$
发给 devops

```
receivers:

- name: 'wechat'
  wechat_configs:
  - corp_id: 'wwf4ee8ede83b63a1a'
    to_party: '1'
    agent_id: '1000003'
    api_secret: 'LbVzYRczEJMY2rq0c8I8ZjASPfCtzvl3f7zfiuyVKSc'
    send_resolved: true
    message:    '{{ template "wechat.tmpl" . }}'
```

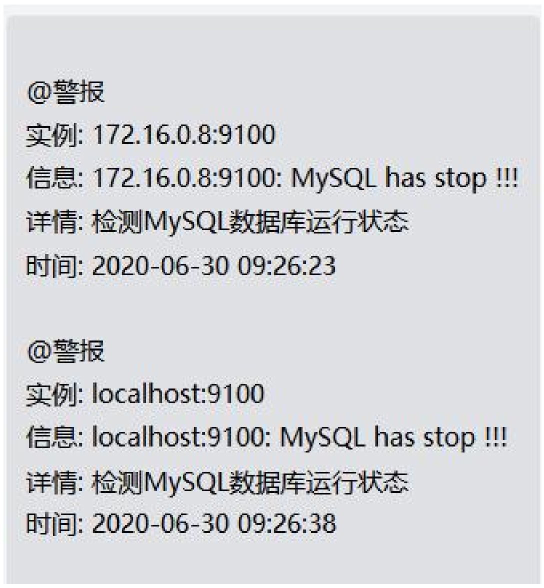
根据名字来匹配

告警分组

```
route:
  group_by: [severity]
```

```
[root@docker-3 alertmanager]# cat alertmanager.yml
global:
  resolve_timeout: 10s
  smtp_smarthost: 'smtp.163.com:25'
  smtp_from: 'jumpservervip@163.com'
  smtp_auth_username: 'jumpservervip@163.com'
  smtp_auth_password: 'xxx'
  smtp_require_tls: false

templates:
  - '/usr/local/alertmanager/*.tmpl'

route:
  group_by: [severity]
  group_wait: 10s
  group_interval: 3m
  repeat_interval: 3m
```

```
    receiver: 'email'
    routes:
    - match:
            severity: critical
        receiver: 'leader'
        continue: true
    - match_re:
            severity: ^(warning|critical)$
        receiver: 'devops'
        continue: true

receivers:
- name: 'email'
    email_configs:
    - to: 'jumpservervip@126.com'
        html: '{{ template "email.to.html" . }}'
        send_resolved: true

- name: 'leader'
    email_configs:
    - to: 'jumpservervip@163.com'
        html: '{{ template "email.to.html" . }}'
        send_resolved: true

- name: 'devops'
    wechat_configs:
    - corp_id: 'wwf4ee8ede83b63a1a'
        to_party: '1'
        agent_id: '1000003'
        api_secret: 'LbVzYRczEJMY2rq0c8I8ZjASPfCtzvl3f7zfiuyVKSc'
        send_resolved: true
        message:    '{{ template "wechat.tmpl" . }}'


inhibit_rules:
    - source_match:
            severity: 'critical'
        target_match:
            severity: 'warning'
        equal: ['alertname', 'instance']
```



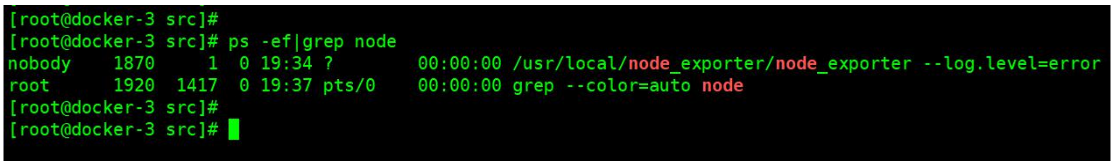
@警报
实例: 172.16.0.8:9100
信息: 172.16.0.8:9100: MySQL has stop !!!
详情: 检测MySQL数据库运行状态
时间: 2020-06-30 09:26:23

@警报
实例: localhost:9100
信息: localhost:9100: MySQL has stop !!!
详情: 检测MySQL数据库运行状态
时间: 2020-06-30 09:26:38

8. Prometheus 企业监控案例

8.1、主机监控

```
[root@docker-3 src]# tar xf node_exporter-0.18.1.linux-amd64.tar.gz
[root@docker-3 src]# mv node_exporter-0.18.1.linux-amd64 /usr/local/node_exporter-0.18.1
[root@docker-3 src]# ln -s /usr/local/node_exporter-0.18.1/ /usr/local/node_exporter

[root@jumpserver ~]# cat /usr/lib/systemd/system/node_exporter.service
[Unit]
Description=Prometheus node_exporter
[Service]
User=nobody
ExecStart=/usr/local/node_exporter/node_exporter --log.level=error
ExecStop=/usr/bin/killall node_exporter

[Install]
WantedBy=default.target

[root@jumpserver x]# systemctl start node_exporter
```



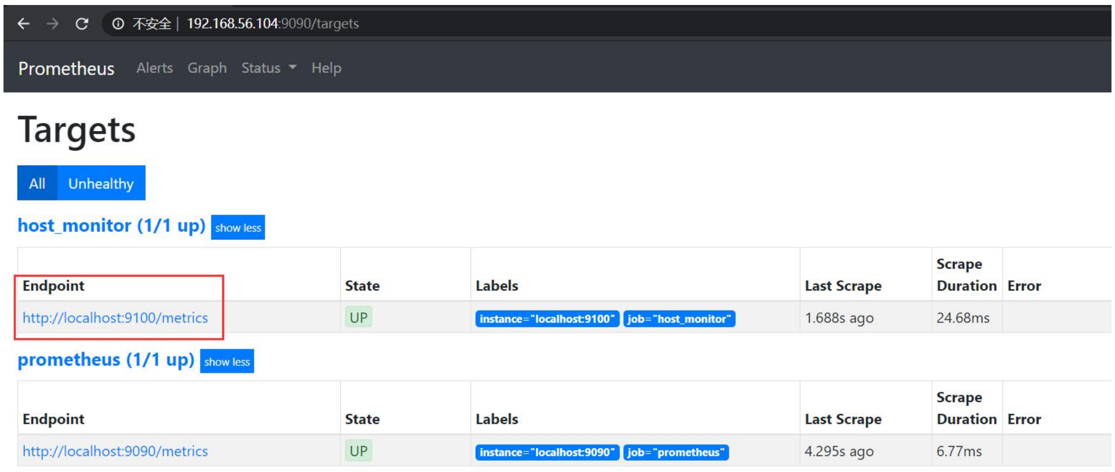
```
[root@docker-3 src]#
[root@docker-3 src]# ps -ef|grep node
nobody    1870     1  0 19:34 ?        00:00:00 /usr/local/node_exporter/node_exporter --log.level=error
root      1920  1417  0 19:37 pts/0    00:00:00 grep --color=auto node
[root@docker-3 src]#
[root@docker-3 src]#
```

```
[root@jumpserver x]# vim /usr/local/prometheus/prometheus.yml
  - job_name: 'host_monitor'
    static_configs:
```

```
    - targets: ['localhost:9100']      ##新增 9100 端口主机监控
```

检查语法

```
[root@docker-3 prometheus]# cd /usr/local/prometheus/
[root@docker-3 prometheus]# ./promtool check config prometheus.yml
Checking prometheus.yml
  SUCCESS: 0 rule files found

[root@jumpserver x]# systemctl restart prometheus
```



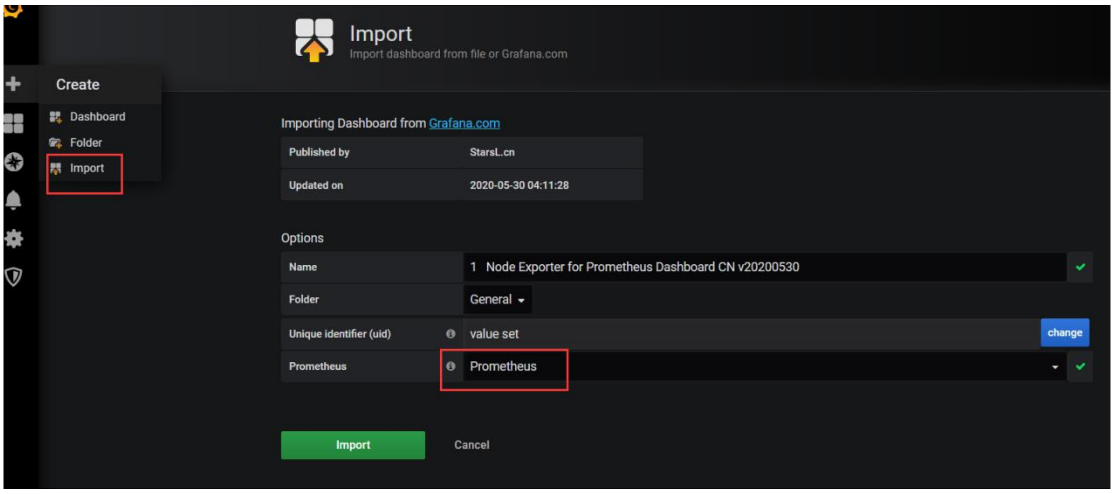
Targets

host_monitor (1/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9100/metrics	UP	instance="localhost:9100" job="host_monitor"	1.688s ago	24.68ms	

prometheus (1/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	4.295s ago	6.77ms	



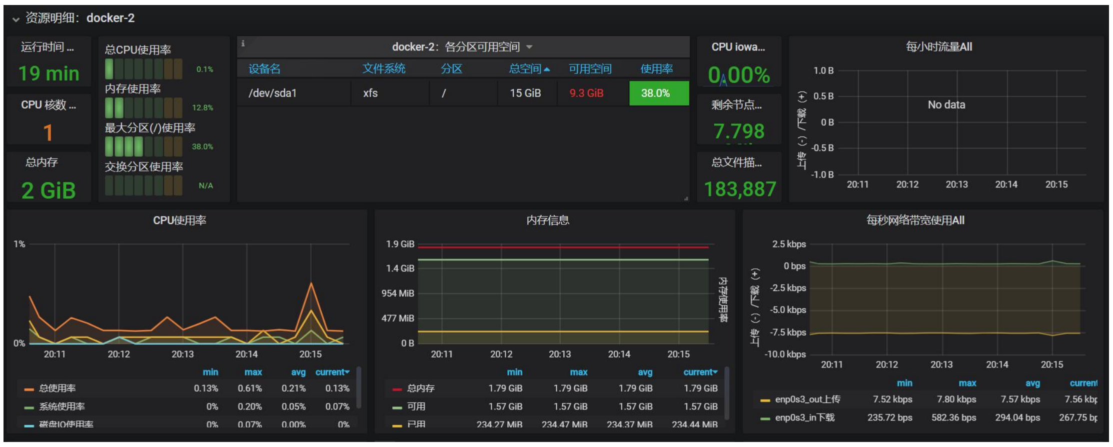
Importing Dashboard from Grafana.com

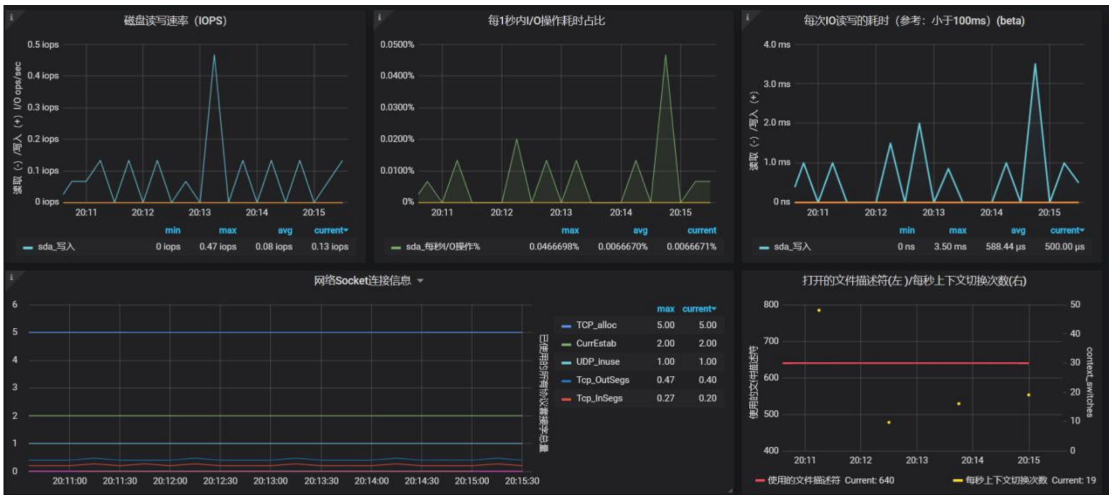
Published by	StarsL.cn
Updated on	2020-05-30 04:11:28

Options

Name	1 Node Exporter for Prometheus Dashboard CN v20200530
Folder	General
Unique identifier (uid)	value set
Prometheus	Prometheus

导入主机模板 8919




磁盘读写速率（IOPS）
每1秒内I/O操作耗时占比
每次IO读写的耗时（参考：小于100ms）(beta)
网络Socket连接信息
打开的文件描述符(左)/每秒上下文切换次数(右)

8.2、MySQL 单机监控

1、部署 mysql_exporter

```
[root@docker-3 src]# wget -c https://github.com/prometheus/mysqld_exporter/releases/download/v0.12.1/mysqld_exporter-0.12.1.linux-amd64.tar.gz

[root@docker-3 src]# tar xf mysqld_exporter-0.12.1.linux-amd64.tar.gz
[root@docker-3 src]# mv mysqld_exporter-0.12.1.linux-amd64 /usr/local/mysqld_exporter-0.12.1
[root@docker-3 src]# ln -s /usr/local/mysqld_exporter-0.12.1/ /usr/local/mysqld_exporter
```

通过 systemd 方式管理

```
[root@docker-2 ~]# cat /usr/lib/systemd/system/mysqld_exporter.service
[Unit]
Description=mysql Monitoring System
```

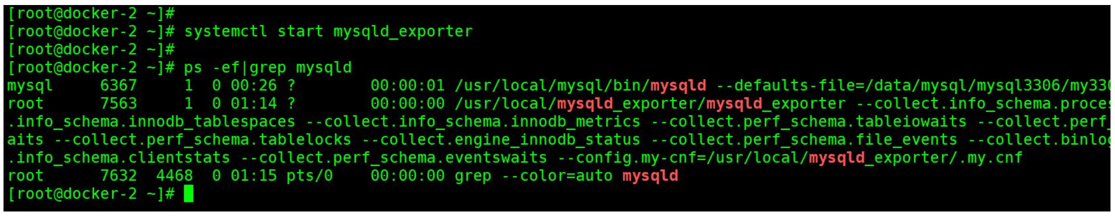
```
Documentation=mysql Monitoring System

[Service]
ExecStart=/usr/local/mysqld_exporter/mysqld_exporter \
            --collect.info_schema.processlist \
            --collect.info_schema.innodb_tablespaces \
            --collect.info_schema.innodb_metrics    \
            --collect.perf_schema.tableiowaits \
            --collect.perf_schema.indexiowaits \
            --collect.perf_schema.tablelocks \
            --collect.engine_innodb_status \
            --collect.perf_schema.file_events \
            --collect.binlog_size \
            --collect.info_schema.clientstats \
            --collect.perf_schema.eventswaits \
            --config.my-cnf=/usr/local/mysqld_exporter/.my.cnf

[Install]
WantedBy=multi-user.target
```

2、增加配置文件

```
[root@docker-3 src]# cat /usr/local/mysqld_exporter/.my.cnf
[client]
host=localhost
user=exporter
password=123456
socket=/tmp/mysql3306.sock
```

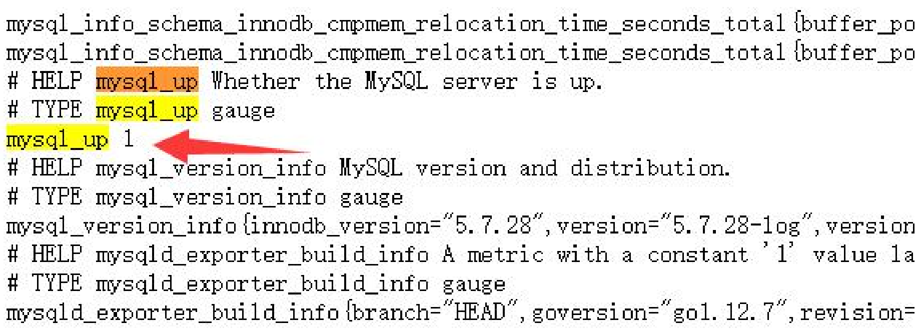
3、mysql 添加授权账户

```
db02 [(none)]>GRANT SELECT, PROCESS, SUPER, REPLICATION CLIENT, RELOAD ON *.* TO 'exporter'@'localhost' IDENTIFIED BY '123456';
Query OK, 0 rows affected, 1 warning (0.00 sec)

db02 [(none)]>flush privileges;

[root@docker-2 ~]# systemctl start mysqld_exporter
```



```
[root@docker-2 ~]#
[root@docker-2 ~]# systemctl start mysqld_exporter
[root@docker-2 ~]#
[root@docker-2 ~]# ps -ef|grep mysqld
mysql     6367     1  0 00:26 ?        00:00:01 /usr/local/mysql/bin/mysqld --defaults-file=/data/mysql/mysql3306/my330
root      7563     1  0 01:14 ?        00:00:00 /usr/local/mysqld_exporter/mysqld_exporter --collect.info_schema.proce
.info_schema.innodb_tablespaces --collect.info_schema.innodb_metrics --collect.perf_schema.tableiowaits --collect.perf_
aits --collect.perf_schema.tablelocks --collect.engine_innodb_status --collect.perf_schema.file_events --collect.binlo
.info_schema.clientstats --collect.perf_schema.eventswaits --config.my-cnf=/usr/local/mysqld_exporter/.my.cnf
root      7632  4468  0 01:15 pts/0    00:00:00 grep --color=auto mysqld
[root@docker-2 ~]#
```

http://ip:9104/metrics



```
mysql_info_schema_innodb_cmpmem_relocation_time_seconds_total{buffer_po
mysql_info_schema_innodb_cmpmem_relocation_time_seconds_total{buffer_po
# HELP mysql_up Whether the MySQL server is up.
# TYPE mysql_up gauge
mysql_up 1
# HELP mysql_version_info MySQL version and distribution.
# TYPE mysql_version_info gauge
mysql_version_info{innodb_version="5.7.28",version="5.7.28-log",version
# HELP mysqld_exporter_build_info A metric with a constant '1' value la
# TYPE mysqld_exporter_build_info gauge
mysqld_exporter_build_info{branch="HEAD",goversion="go1.12.7",revision=
```

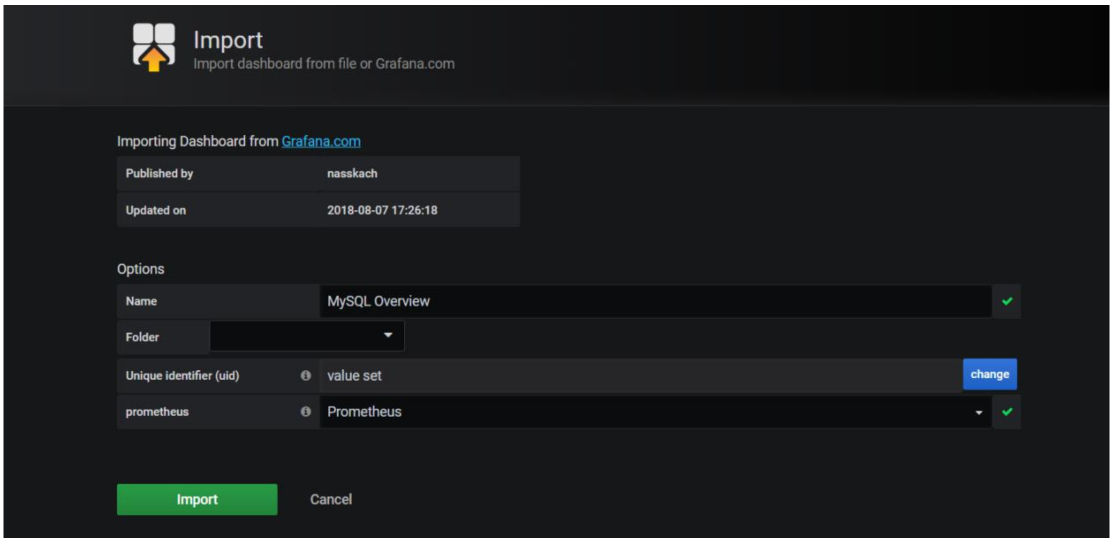
mysql_up 1 ##代表 mysql 被监控并且已经启动

4. 修改 prometheus 文件并重启

```
- job_name: 'mysql_monitor'
  static_configs:
  - targets: ['172.16.0.8:9104']
```

```
[root@docker-3 src]# systemctl restart prometheus
```

Mysql 转态监控模板 7362



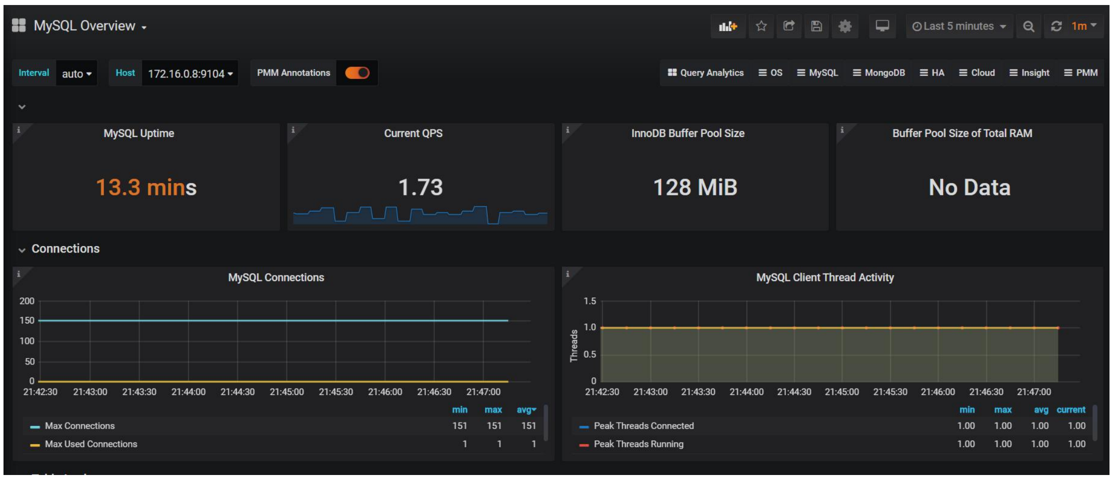
Import
Import dashboard from file or Grafana.com

Importing Dashboard from Grafana.com

Published by	nasskach
Updated on	2018-08-07 17:26:18

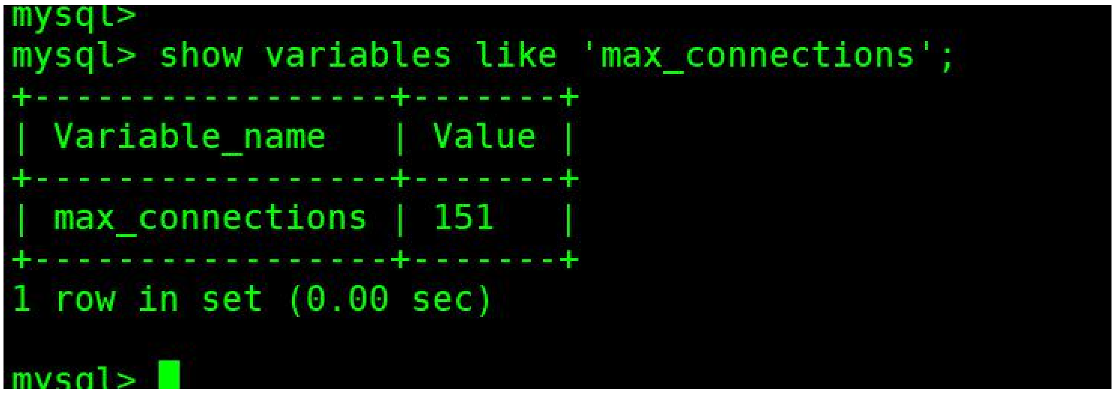
Options

Name	MySQL Overview
Folder	
Unique identifier (uid)	value set
prometheus	Prometheus



	min	max	avg
Max Connections	151	151	151
Max Used Connections	1	1	1

	min	max	avg	current
Peak Threads Connected	1.00	1.00	1.00	1.00
Peak Threads Running	1.00	1.00	1.00	1.00


```
mysql>
mysql> show variables like 'max_connections';
+-----------------+-------+
| Variable_name   | Value |
+-----------------+-------+
| max_connections | 151   |
+-----------------+-------+
1 row in set (0.00 sec)

mysql> 
```

8.3、MySQL 主从监控

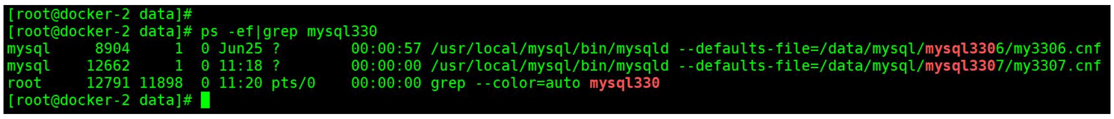
环境准备

```
[root@docker-3 src]# cat /data/mysql/mysql3306/my3306.cnf
[mysql]
prompt="\u@\h [\d]>"
[mysqld]
user = mysql
basedir = /usr/local/mysql
datadir = /data/mysql/mysql3306/data
log-error=/data/mysql/mysql3306/data/error_3306.log
server_id = 19
port = 3306
log_bin=/data/mysql/mysql3306/binlog/mysql-bin
binlog_format=row
gtid-mode=on
enforce-gtid-consistency=true
socket = /tmp/mysql3306.sock[root@docker-2 system]#
```

```
[root@docker-2 system]# cat /data/mysql/mysql3306/my3306.cnf
[mysqld]
user = mysql
basedir = /usr/local/mysql
datadir = /data/mysql/mysql3306/data
log_bin= /data/mysql/mysql3306/binlog/mysql-bin
server_id = 18
gtid-mode=on
enforce-gtid-consistency=true
```

port = 3306

socket = /tmp/mysql3306.sock



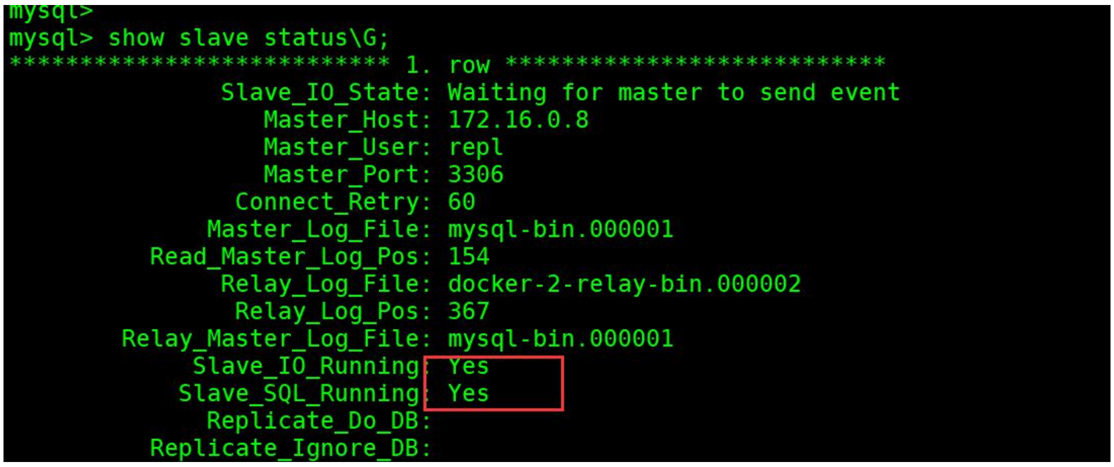
```
[root@docker-2 data]#
[root@docker-2 data]# ps -ef|grep mysql330
mysql     8904     1  0 Jun25 ?        00:00:57 /usr/local/mysql/bin/mysqld --defaults-file=/data/mysql/mysql3306/my3306.cnf
mysql    12662     1  0 11:18 ?        00:00:00 /usr/local/mysql/bin/mysqld --defaults-file=/data/mysql/mysql3307/my3307.cnf
root     12791 11898  0 11:20 pts/0    00:00:00 grep --color=auto mysql330
[root@docker-2 data]# 
```

主库

grant replication slave on *.* to repl@'172.16.0.%' identified by '123456';

从库

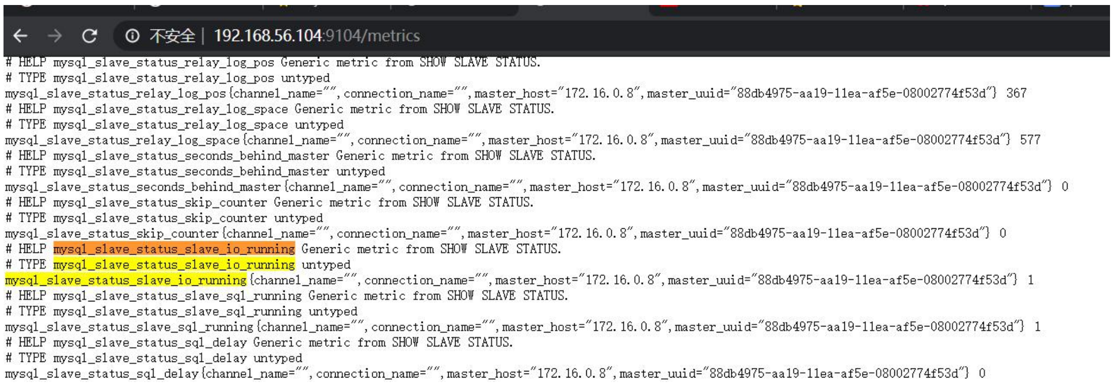
```
CHANGE MASTER TO MASTER_HOST='172.16.0.8',
MASTER_USER='repl',
MASTER_PASSWORD='123456',
MASTER_PORT=3306,
MASTER_AUTO_POSITION=1;

mysql> start slave;
mysql> show slave status\G;
```



```
mysql>
mysql> show slave status\G;
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 172.16.0.8
                  Master_User: repl
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000001
          Read_Master_Log_Pos: 154
               Relay_Log_File: docker-2-relay-bin.000002
                Relay_Log_Pos: 367
        Relay_Master_Log_File: mysql-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
```

从库增加 mysql_exporter 监控，过程和主从步骤一致

从库查看



```
# HELP mysql_slave_status_relay_log_pos Generic metric from SHOW SLAVE STATUS.
# TYPE mysql_slave_status_relay_log_pos untyped
mysql_slave_status_relay_log_pos{channel_name="",connection_name="",master_host="172.16.0.8",master_uuid="88db4975-aa19-11ea-af5e-08002774f53d"} 367
# HELP mysql_slave_status_relay_log_space Generic metric from SHOW SLAVE STATUS.
# TYPE mysql_slave_status_relay_log_space untyped
mysql_slave_status_relay_log_space{channel_name="",connection_name="",master_host="172.16.0.8",master_uuid="88db4975-aa19-11ea-af5e-08002774f53d"} 577
# HELP mysql_slave_status_seconds_behind_master Generic metric from SHOW SLAVE STATUS.
# TYPE mysql_slave_status_seconds_behind_master untyped
mysql_slave_status_seconds_behind_master{channel_name="",connection_name="",master_host="172.16.0.8",master_uuid="88db4975-aa19-11ea-af5e-08002774f53d"} 0
# HELP mysql_slave_status_skip_counter Generic metric from SHOW SLAVE STATUS.
# TYPE mysql_slave_status_skip_counter untyped
mysql_slave_status_skip_counter{channel_name="",connection_name="",master_host="172.16.0.8",master_uuid="88db4975-aa19-11ea-af5e-08002774f53d"} 0
# HELP mysql_slave_status_slave_io_running Generic metric from SHOW SLAVE STATUS.
# TYPE mysql_slave_status_slave_io_running untyped
mysql_slave_status_slave_io_running{channel_name="",connection_name="",master_host="172.16.0.8",master_uuid="88db4975-aa19-11ea-af5e-08002774f53d"} 1
# HELP mysql_slave_status_slave_sql_running Generic metric from SHOW SLAVE STATUS.
# TYPE mysql_slave_status_slave_sql_running untyped
mysql_slave_status_slave_sql_running{channel_name="",connection_name="",master_host="172.16.0.8",master_uuid="88db4975-aa19-11ea-af5e-08002774f53d"} 1
# HELP mysql_slave_status_sql_delay Generic metric from SHOW SLAVE STATUS.
# TYPE mysql_slave_status_sql_delay untyped
mysql_slave_status_sql_delay{channel_name="",connection_name="",master_host="172.16.0.8",master_uuid="88db4975-aa19-11ea-af5e-08002774f53d"} 0
```

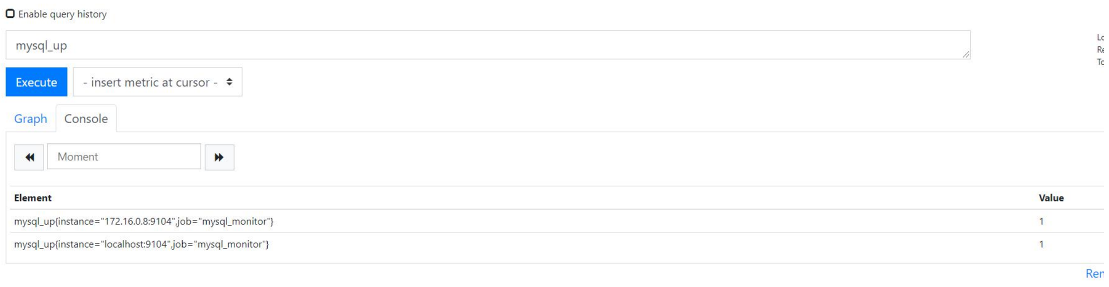
验证从库指标

mysql_slave_status_slave_io_running

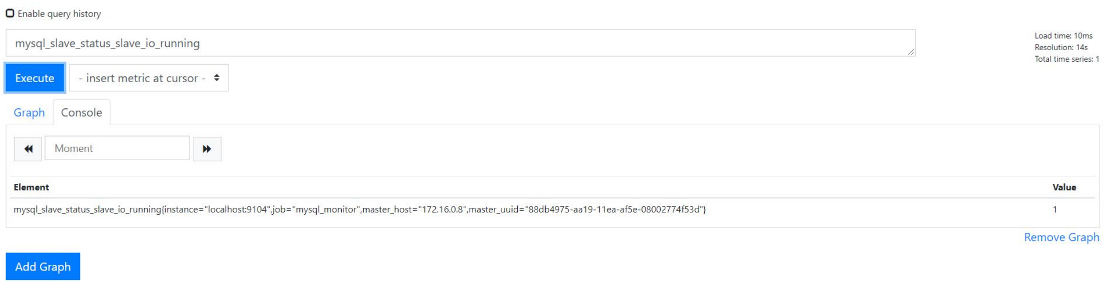
修改 prometheus 配置

```
  - job_name: 'mysql_monitor'
    static_configs:
    - targets: ['172.16.0.8:9104','localhost:9104']
```

```
[root@docker-3 src]# systemctl restart prometheus
```

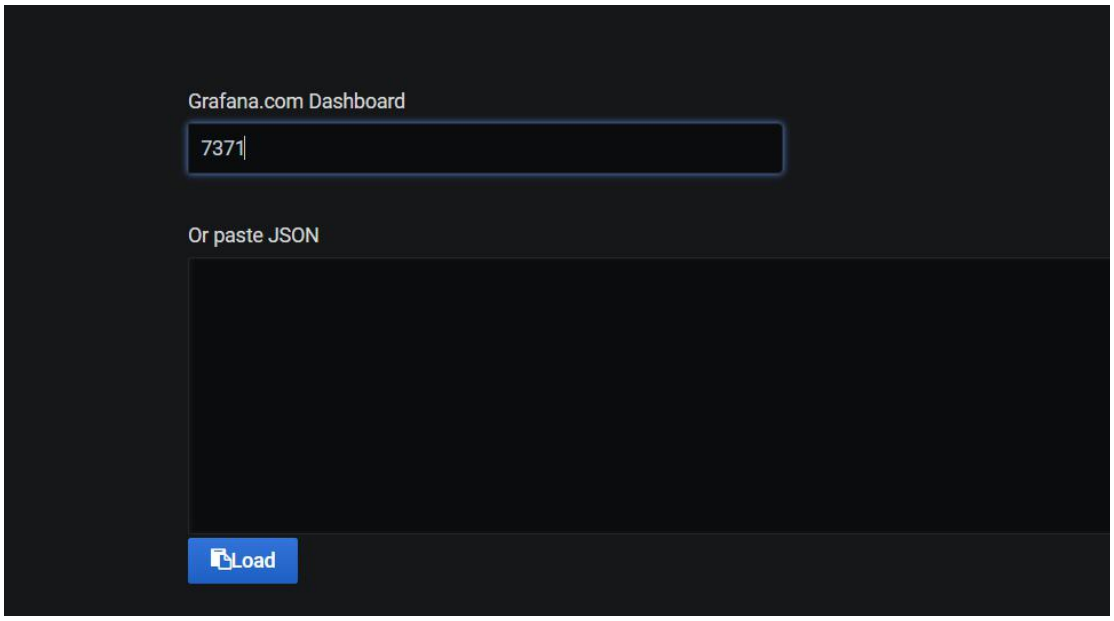


Element	Value
mysql_up{instance="172.16.0.8:9104",job="mysql_monitor"}	1
mysql_up{instance="localhost:9104",job="mysql_monitor"}	1



Element	Value
mysql_slave_status_slave_io_running{instance="localhost:9104",job="mysql_monitor",master_host="172.16.0.8",master_uuid="88db4975-aa19-11ea-af5e-08002774f53d"}	1

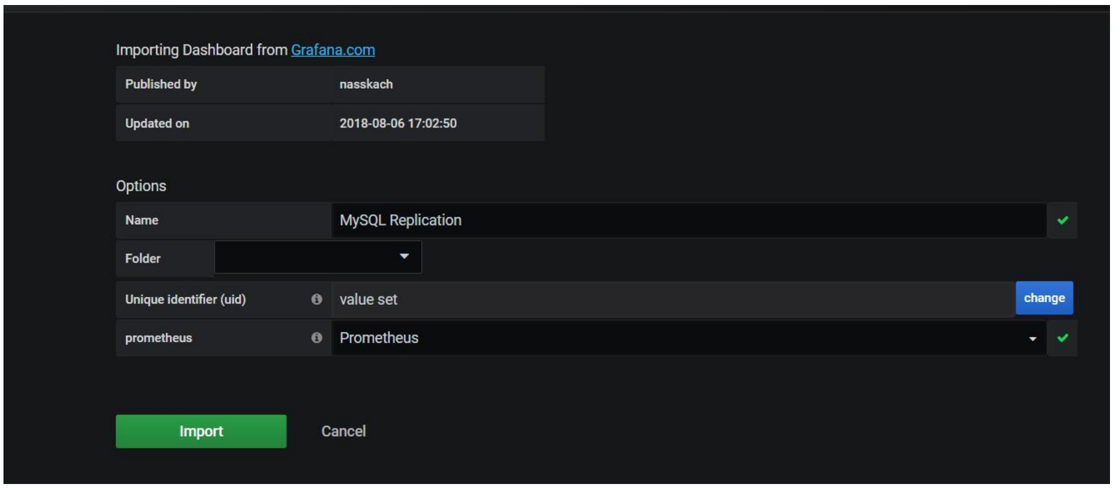
主从模板 7371



Grafana.com Dashboard

7371

Or paste JSON

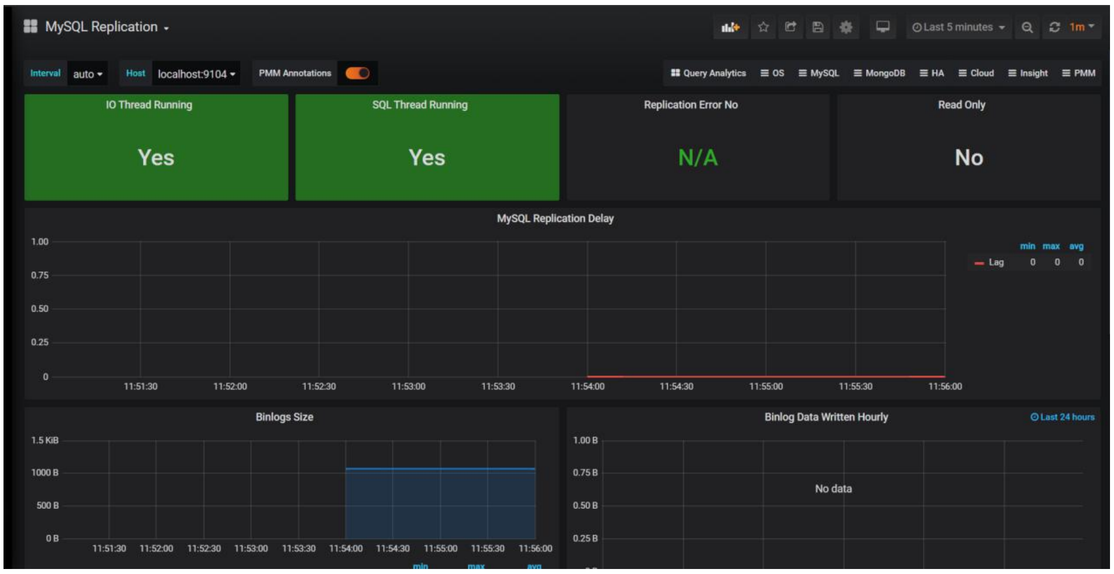


Importing Dashboard from Grafana.com

Published by	nasskach
Updated on	2018-08-06 17:02:50

Options

Name	MySQL Replication
Folder	
Unique identifier (uid)	value set
prometheus	Prometheus



MySQL Replication

Interval auto · Host localhost:9104 · PMM Annotations

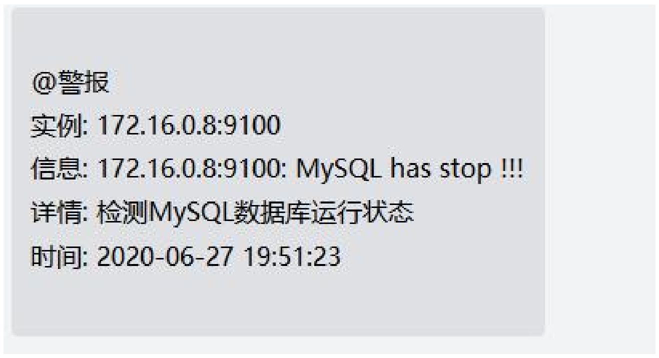
IO Thread Running	SQL Thread Running	Replication Error No	Read Only
Yes	Yes	N/A	No

8.4 添加 MySQL 告警规则

```
[root@docker-3 rules]# cat /usr/local/prometheus/rules/mysql.yml
groups:
- name: MySQL-rules
  rules:
  - alert: MySQL Status
    expr: up == 0
    for: 5s
    labels:
      severity: warning
    annotations:
      summary: "{{$labels.instance}}: MySQL has stop "
      description: "MySQL 数据库挂了,请检查"

  - alert: MySQL Slave IO Thread Status
    expr: mysql_slave_status_slave_io_running == 0
    for: 5s
    labels:
      severity: warning
    annotations:
      summary: "{{$labels.instance}}: MySQL Slave IO Thread has stop "
      description: "检测 MySQL 主从 IO 线程运行状态"

  - alert: MySQL Slave SQL Thread Status
    expr: mysql_slave_status_slave_sql_running == 0
    for: 5s
    labels:
      severity: warning
    annotations:
      summary: "{{$labels.instance}}: MySQL Slave SQL Thread has stop "
      description: "检测 MySQL 主从 SQL 线程运行状态"
```

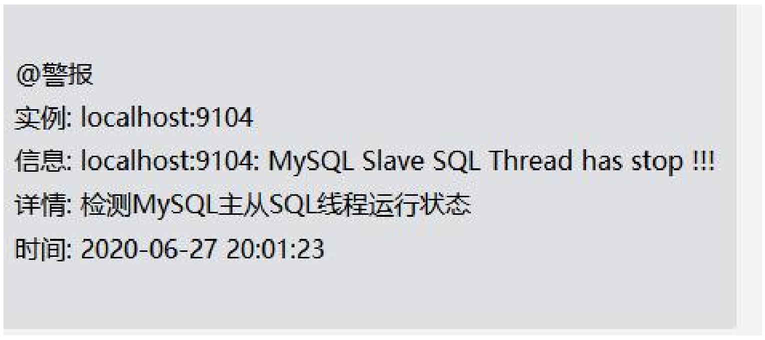
停止从库观察

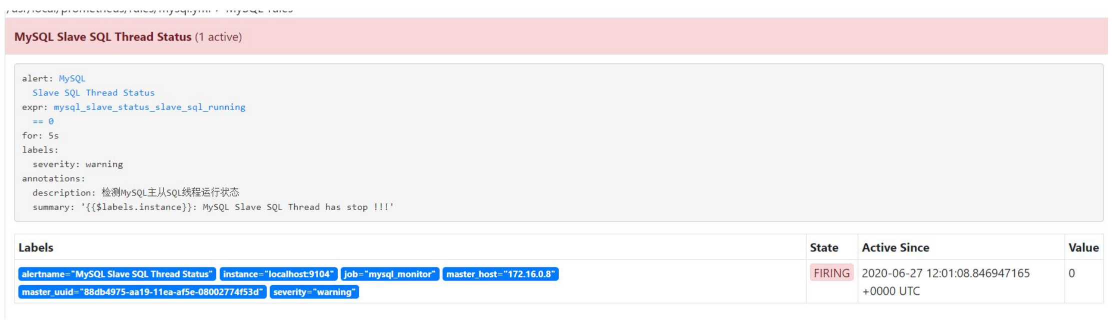
```
[root@docker-3 rules]# systemctl stop mysqld3306
```


@警报
实例: 172.16.0.8:9100
信息: 172.16.0.8:9100: MySQL has stop !!!
详情: 检测MySQL数据库运行状态
时间: 2020-06-27 19:51:23

停止从库 sql 线程观察

mysql> stop slave sql_thread;


@警报
实例: localhost:9104
信息: localhost:9104: MySQL Slave SQL Thread has stop !!!
详情: 检测MySQL主从SQL线程运行状态
时间: 2020-06-27 20:01:23

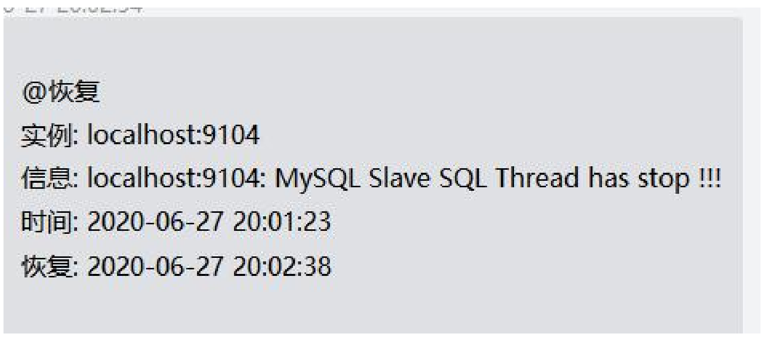

MySQL Slave SQL Thread Status (1 active)

```
alert: MySQL
  Slave SQL Thread Status
expr: mysql_slave_status_slave_sql_running
  == 0
for: 5s
labels:
  severity: warning
annotations:
  description: 检测MySQL主从SQL线程运行状态
  summary: '{{$labels.instance}}: MySQL Slave SQL Thread has stop !!!'
```

Labels	State	Active Since	Value
alertname="MySQL Slave SQL Thread Status" instance="localhost:9104" job="mysql_monitor" master_host="172.16.0.8" master_uuid="88db4975-aa19-11ea-af5e-08002774f53d" severity="warning"	FIRING	2020-06-27 12:01:08.846947165 +0000 UTC	0

恢复

mysql> start slave sql_thread;


@恢复
实例: localhost:9104
信息: localhost:9104: MySQL Slave SQL Thread has stop !!!
时间: 2020-06-27 20:01:23
恢复: 2020-06-27 20:02:38

8.5、Redis 监控

[root@docker-3 src]# wget https://github.com/oliver006/redis_exporter/releases/download/v0.30.0/redis_exporter-v0.30.0.linux-amd64.tar.gz

[root@docker-3 src]# mkdir /usr/local/redis_exporter
[root@docker-3 src]# tar xf redis_exporter-v0.30.0.linux-amd64.tar.gz
[root@docker-3 src]# mv redis_exporter /usr/local/redis_exporter/

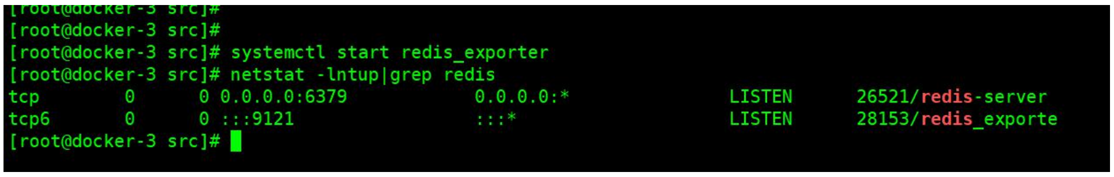
解压后只有一个二进制程序就叫 redis_exporter 通过 -h 可以获取到帮助信息，下面列出一些常用的选项：

-redis.addr：指明一个或多个 Redis 节点的地址，多个节点使用逗号分隔，默认为 redis://localhost:6379
-redis.password：验证 Redis 时使用的密码；
-redis.file：包含一个或多个 redis 节点的文件路径，每行一个节点，此选项与 -redis.addr 互斥。
-web.listen-address：监听的地址和端口，默认为 0.0.0.0:9121

运行 redis_exporter 服务
1，方式 A 直接启动。
无密码
./redis_exporter redis//172.16.0.9:6379 &
有密码
redis_exporter -redis.addr 172.16.0.9:6379 -redis.password 123456

Systemd 方式启动
vim /usr/lib/systemd/system/redis_exporter.service
[Unit]
Description=redis_exporter
Documentation=https://github.com/oliver006/redis_exporter
After=network.target
[Service]
Type=simple
User=prometheus
ExecStart=/usr/local/redis_exporter/redis_exporter -redis.addr 172.16.0.9:6379
Restart=on-failure
[Install]
WantedBy=multi-user.target

[root@docker-3 src]# useradd prometheus -s /sbin/nologin -M



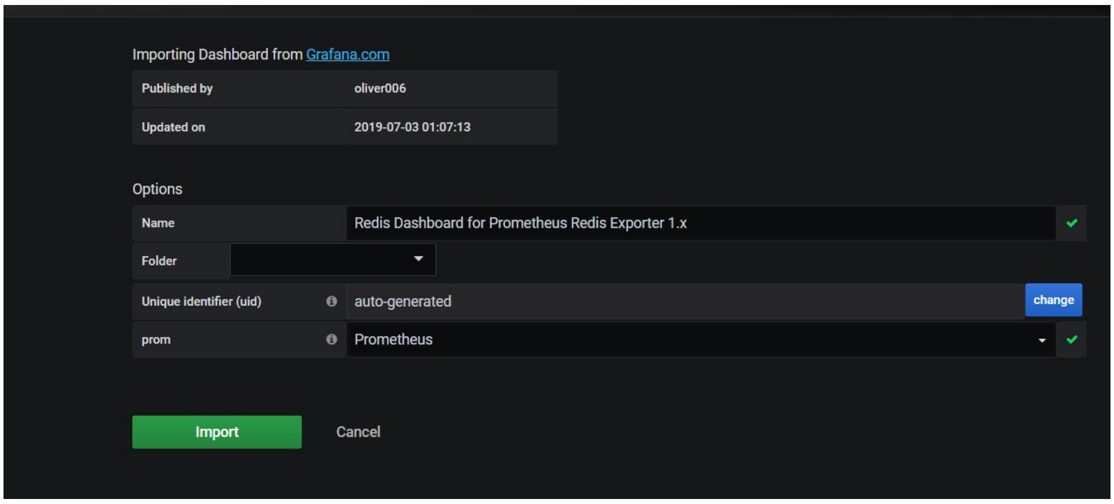
```
[root@docker-3 src]#
[root@docker-3 src]#
[root@docker-3 src]# systemctl start redis_exporter
[root@docker-3 src]# netstat -lntup|grep redis
tcp        0      0 0.0.0.0:6379            0.0.0.0:*               LISTEN      26521/redis-server
tcp6       0      0 :::9121                 :::*                    LISTEN      28153/redis_exporte
[root@docker-3 src]#
```

修改 prometheus 文件

```
  - job_name: 'redis_exporter'
    scrape_interval: 10s
    static_configs:
    - targets:   ['172.16.0.9:9121']
```

```
[root@docker-3 src]# /usr/local/prometheus/promtool check config /usr/local/prometheus/prometheus.yml
[root@docker-3 src]# systemctl restart prometheus
```

导入 redis 监控模板 763

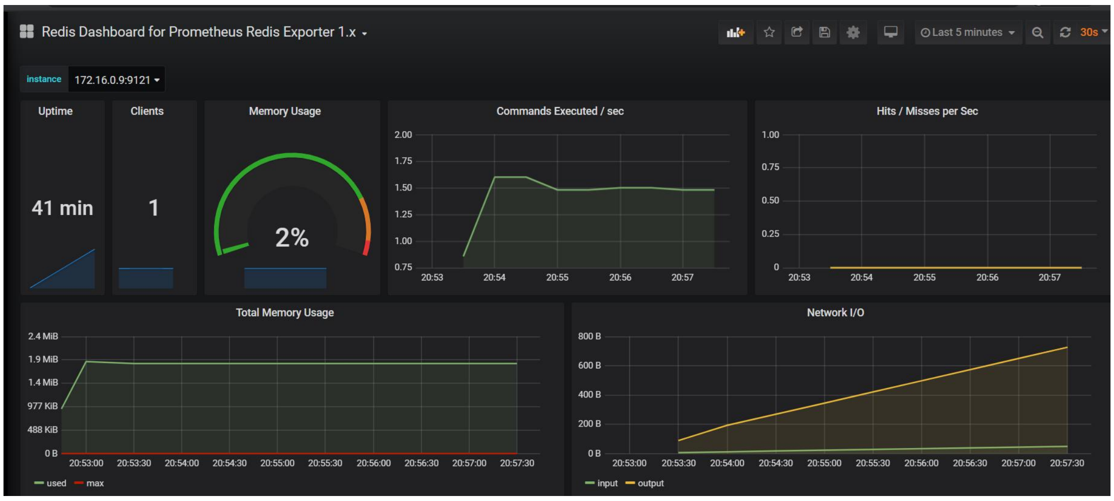


Importing Dashboard from Grafana.com

Published by	oliver006
Updated on	2019-07-03 01:07:13

Options

Name	Redis Dashboard for Prometheus Redis Exporter 1.x
Folder	
Unique identifier (uid)	auto-generated
prom	Prometheus

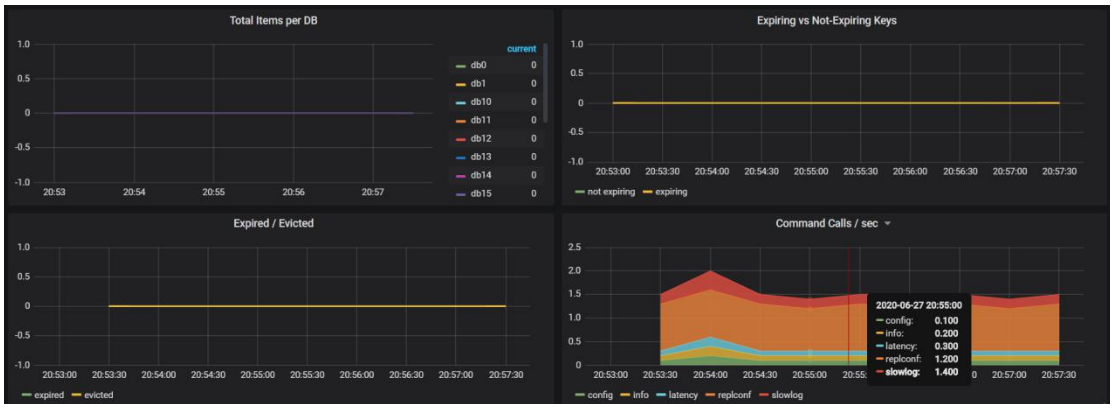


这里注意：如果 redis 没有配置内存 最大可用值

```
127.0.0.1:6379> CONFIG GET maxmemory
1) "maxmemory"
2) "0"
```

则该内存值在 grafana 界面显示是 0

配置参数如下
```
maxmemory 128m
```

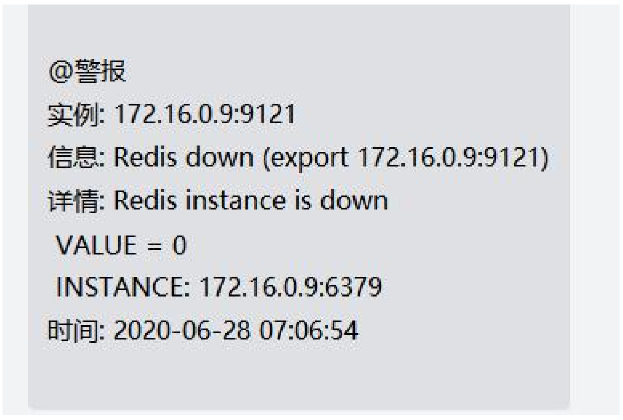


Redis 告警规则
```
[root@docker-3 rules]# cat redis.yml
groups:
- name: redis_instance
  rules:

#redis 实例宕机 危险等级: 5
  - alert: RedisInstanceDown
    expr: redis_up == 0
    for: 10s
    labels:
      severity: warning
    annotations:
      summary: "Redis down (export {{ $labels.instance }})"
      description: "Redis instance is down\n VALUE = {{ $value }}\n INSTANCE: {{ $labels.addr }} {{ $labels.alias }}"

#redis 内存占用过多 危险等级: 4
  - alert: RedisOutofMemory
    expr: redis_memory_used_bytes / redis_total_system_memory_bytes * 100 > 60
```

```
        for: 3m
        labels:
          severity: warning
        annotations:
          summary: "Out of memory (export {{ $labels.instance }})"
          description: "Redis is running out of memory > 80%\n VALUE= {{ $value }}\n INSTANCE: {{ $labels.addr }} {{ $labels.alias }}"

# redis 连接数过多 危险等级: 3
    - alert: RedisTooManyConnections
        expr: redis_connected_clients > 2000
        for: 3m
        labels:
          severity: warning
        annotations:
          summary: "Too many connections (export {{ $labels.instance}})"
          description: "Redis instance has too many connections\n value = {{$value}}\n INSTANCE: {{ $labels.addr }} {{ $labels.alias }}"
```

```
[root@docker-3 rules]# /usr/local/prometheus/promtool check config /usr/local/prometheus/prometheus.yml
[root@docker-3 rules]# systemctl restart prometheus
```

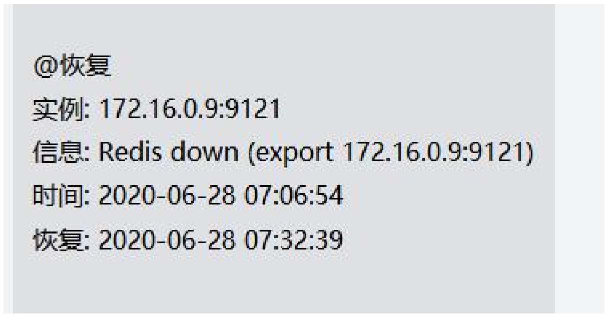
停掉 redis 观察



```
@警报
实例: 172.16.0.9:9121
信息: Redis down (export 172.16.0.9:9121)
详情: Redis instance is down
 VALUE = 0
 INSTANCE: 172.16.0.9:6379
时间: 2020-06-28 07:06:54
```

```
[root@docker-3 local]# redis-server /usr/local/redis/etc/redis.conf
```

恢复观察



@恢复
实例: 172.16.0.9:9121
信息: Redis down (export 172.16.0.9:9121)
时间: 2020-06-28 07:06:54
恢复: 2020-06-28 07:32:39

8.6 elasticsearch 集群监控

Es 集群环境准备
安装 java
```
yum install -y java-1.8.0-openjdk.x86_64
```
1.安装软件
```
rpm -ivh elasticsearch-6.6.0.rpm
```

2.修改配置文件
```
[root@db02 elasticsearch]# cat /etc/elasticsearch/elasticsearch.yml
cluster.name: Linux
node.name: node-2
path.data: /data/elasticsearch
path.logs: /var/log/elasticsearch
bootstrap.memory_lock: true
network.host: 172.16.0.7,127.0.0.1
http.port: 9200
discovery.zen.ping.unicast.hosts: ["172.16.0.7", "172.16.0.8"]
discovery.zen.minimum_master_nodes: 2
```
3.修改内存锁定
```
[root@db02 ~]# systemctl edit elasticsearch
[Service]
LimitMEMLOCK=infinity
```
4.创建数据目录并授权
```
mkidr /data/elasticsearch
chown =R elasticsearch:elasticsearch /data/elasticsearch
```
5.重启服务
```
systemctl daemon-reload
systemctl start elasticsearch
```
6.查看日志和端口
```
tail -f /var/log/elasticsearch/Linux.log
netstat -lntup:grep 9200
```

部署 es export

```
wget
https://github.com/justwatchcom/elasticsearch_exporter/releases/download/v1.1.0/elasticsearc
h_exporter-1.1.0.linux-amd64.tar.gz
tar -xvf elasticsearch_exporter-1.1.0.linux-amd64.tar.gz
mv elasticsearch_exporter-1.1.0.linux-amd64 /us r/local/elasticsearch_exporter-1.1.0
ln -s    /usr/local/elasticsearch_exporter-1.1.0 /usr/local/elasticsearch_exporter
```

进入目录下面启动

```
nohup ./elasticsearch_exporter --es.uri http://172.16.0.7:9200 &
```

--es.uri 默认 http://localhost:9200，连接到的 Elasticsearch 节点的地址（主机和端口）

Systemd 启动方式

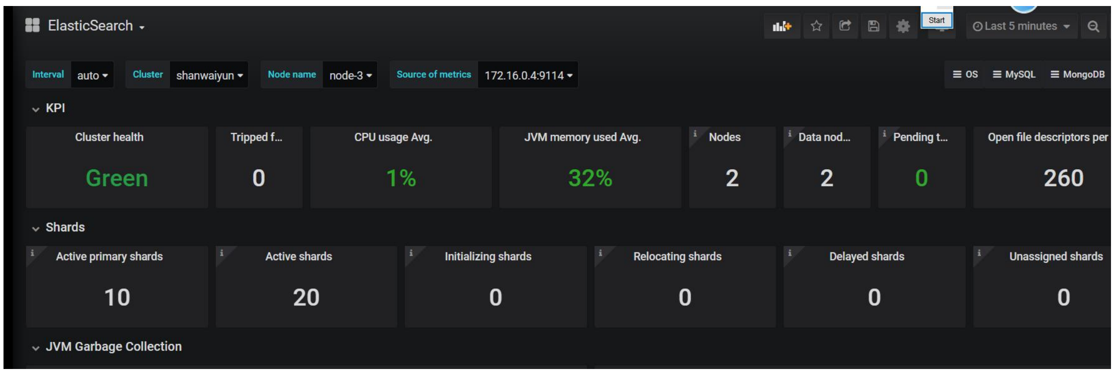
```
cat    /etc/systemd/system/elasticsearch_exporter.service
[Unit]
Description=Elasticsearch stats exporter for Prometheus
Documentation=Prometheus exporter for various metrics

[Service]
ExecStart=/usr/local/elasticsearch_exporter/elasticsearch_exporter --es.uri http://ip:9200

[Install]
WantedBy=multi-user.target
```

http://ip:9114/metrics/   查看采集到的信息

修改 prometheus 配置

```
  - job_name: 'elasticsearch_exporter'
      scrape_interval: 10s
      metrics_path: "/_prometheus/metrics"
      static_configs:
      - targets: ['172.16.0.5:9114','172.16.0.6:9114','172.16.0.7:9114',]
```

导入 granfana 模板 2322/266


ElasticSearch

Interval auto · Cluster shanwaiyun · Node name node-3 · Source of metrics 172.16.0.4:9114

KPI

Cluster health	Tripped f...	CPU usage Avg.	JVM memory used Avg.	Nodes	Data nod...	Pending t...	Open file descriptors per
Green	0	1%	32%	2	2	0	260

Shards

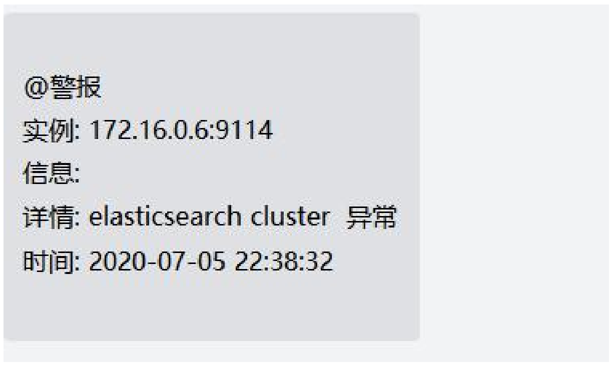
Active primary shards	Active shards	Initializing shards	Relocating shards	Delayed shards	Unassigned shards
10	20	0	0	0	0

JVM Garbage Collection

编写 es 告警规则
集群状态，green（ 所有的主分片和副本分片都正常运行）、yellow（所有的主分片都正常运行，但不是所有的副本分片都正常运行）red（有主分片没能正常运行）

```
groups:
- name: es
  rules:
  - alert: esclusterwrong
    expr: elasticsearch_cluster_health_status{color="green"}  != 1
    for: 10s
    labels:
      severity: critical
    annotations:
      description:  "elasticsearch cluster {{$labels.server}} 异常"

  - alert: esDown
    expr: elasticsearch_cluster_health_number_of_nodes  != 3
    for: 10s
    labels:
      severity: critical
    annotations:
      description:  "elasticsearch service {{$labels.instance}} down"
```

停止一台 es 观察

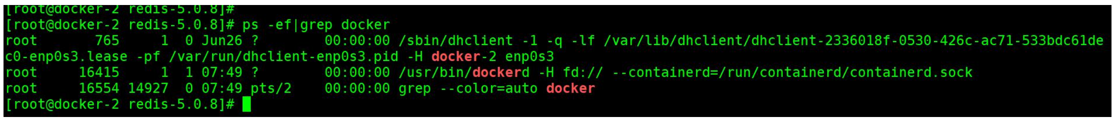

@警报
实例: 172.16.0.6:9114
信息:
详情: elasticsearch cluster 异常
时间: 2020-07-05 22:38:32

8.7、Docker 监控

cAdvisor 将容器统计信息公开为 Prometheus 指标。
默认情况下，这些指标在/metrics HTTP 端点下提供。
可以通过设置-prometheus_endpoint 命令行标志来自定义此端点。
要使用 Prometheus 监控 cAdvisor，只需在 Prometheus 中配置一个或多个作业，这些作业会在该指标端点处刮取相关的 cAdvisor 流程。

Docker 环境准备
CentOS 7（使用 yum 进行安装）

```
# step 1: 安装必要的一些系统工具
sudo yum install -y yum-utils device-mapper-persistent-data lvm2
# Step 2: 添加软件源信息
sudo yum-config-manager --add-repo https://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo
# Step 3: 更新并安装 Docker-CE
sudo yum makecache fast
sudo yum -y install docker-ce
# Step 4: 开启 Docker 服务
sudo service docker start
```



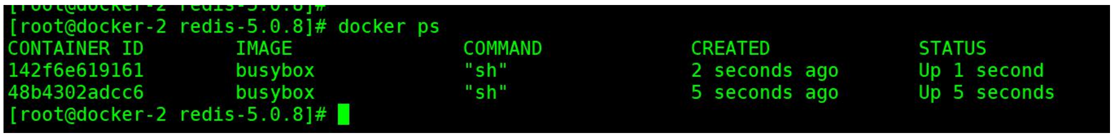
```
[root@docker-2 redis-5.0.8]#
[root@docker-2 redis-5.0.8]# ps -ef|grep docker
root       765     1  0 Jun26 ?        00:00:00 /sbin/dhclient -1 -q -lf /var/lib/dhclient/dhclient-2336018f-0530-426c-ac71-533bdc61de
c0-enp0s3.lease -pf /var/run/dhclient-enp0s3.pid -H docker-2 enp0s3
root     16415     1  1 07:49 ?        00:00:00 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
root     16554 14927  0 07:49 pts/2    00:00:00 grep --color=auto docker
[root@docker-2 redis-5.0.8]#
```

下载测试镜像

```
[root@docker-2 redis-5.0.8]# docker pull busybox
```

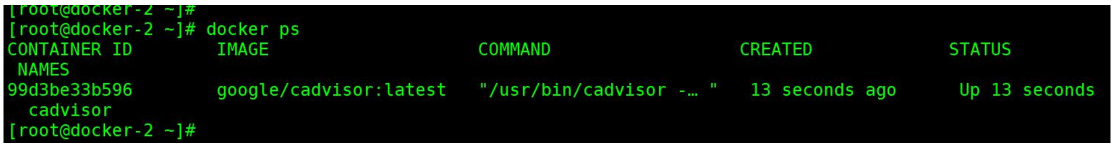
生成容器

```
[root@docker-2 redis-5.0.8]# docker run -itd --name bb1   busybox
[root@docker-2 redis-5.0.8]# docker run -itd --name bb2   busybox
```



```
[root@docker-2 redis-5.0.8]#
[root@docker-2 redis-5.0.8]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
142f6e619161        busybox             "sh"                2 seconds ago       Up 1 second
48b4302adcc6        busybox             "sh"                5 seconds ago       Up 5 seconds
[root@docker-2 redis-5.0.8]#
```

```
docker run \
  --volume=/:/rootfs:ro \
  --volume=/var/run:/var/run:ro \
  --volume=/sys:/sys:ro \
  --volume=/var/lib/docker/:/var/lib/docker:ro \
  --volume=/dev/disk/:/dev/disk:ro \
  --publish=8080:8080 \
  --detach=true \
  --name=cadvisor \
  google/cadvisor:latest
```



```
[root@docker-2 ~]#
[root@docker-2 ~]# docker ps
CONTAINER ID        IMAGE                    COMMAND                  CREATED             STATUS
 NAMES
99d3be33b596        google/cadvisor:latest   "/usr/bin/cadvisor -… "   13 seconds ago      Up 13 seconds
  cadvisor
[root@docker-2 ~]#
```

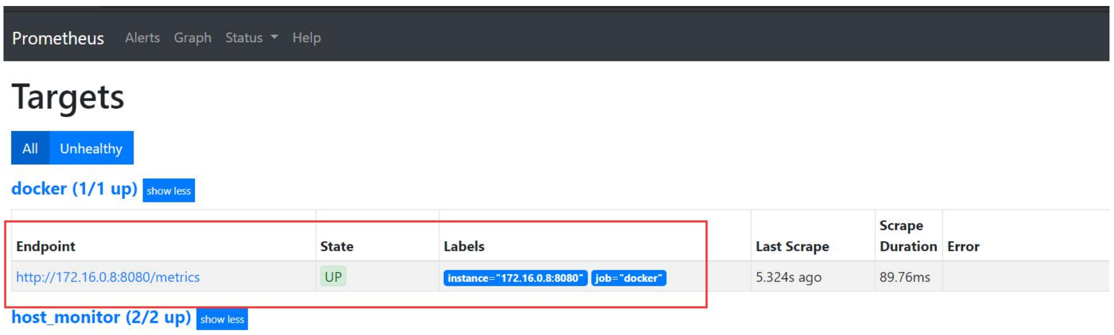
验证采集的数据

```
[root@docker-2 network-scripts]# curl http://172.16.0.8:8080/metrics
```

Prometheus 增加 docker 监控

```
  - job_name: 'docker'
    static_configs:
    - targets: ['172.16.0.8:8080']
```

```
[root@docker-3 rules]# /usr/local/prometheus/promtool check config /usr/local/prometheus/prometheus.yml^C
[root@docker-3 rules]# systemctl restart prometheus
```



Targets

docker (1/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://172.16.0.8:8080/metrics	UP	instance="172.16.0.8:8080" job="docker"	5.324s ago	89.76ms	

host_monitor (2/2 up)

容器 CPU 使用率:

```
sum(irate(container_cpu_usage_seconds_total{image!=""}[1m])) without (cpu)
```

查询容器内存使用量（单位：字节）:

```
container_memory_usage_bytes{image!=""}
```

查询容器网络接收量速率（单位：字节/秒）:

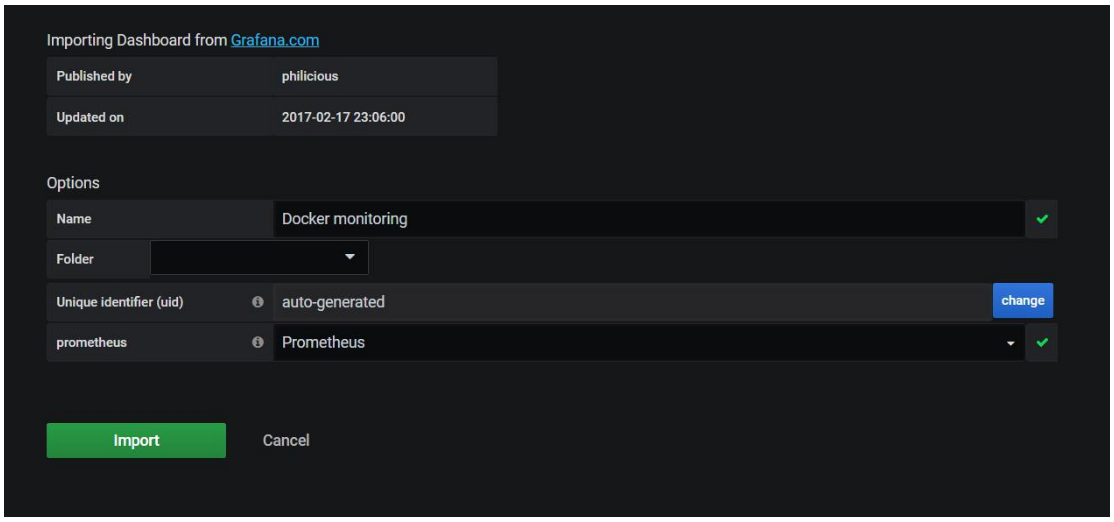
sum(rate(container_network_receive_bytes_total{image!=""}[1m])) without (interface)

查询容器网络传输量速率（单位：字节/秒）：
sum(rate(container_network_transmit_bytes_total{image!=""}[1m])) without (interface)

查询容器文件系统读取速率（单位：字节/秒）：
sum(rate(container_fs_reads_bytes_total{image!=""}[1m])) without (device)

查询容器文件系统写入速率（单位：字节/秒）：
sum(rate(container_fs_writes_bytes_total{image!=""}[1m])) without (device)

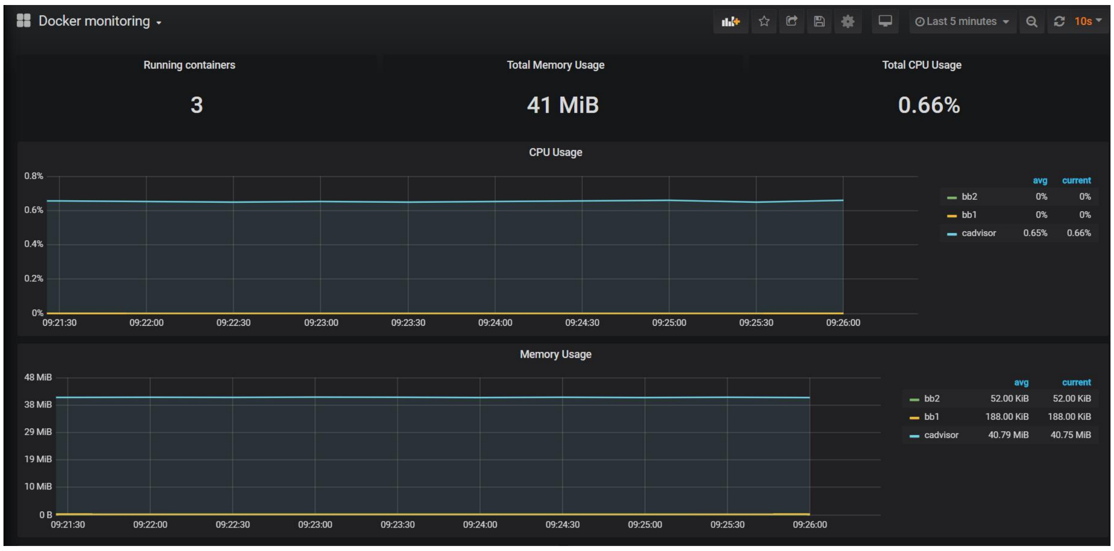
# grafana 模板：193 模板：


Importing Dashboard from Grafana.com

Published by	philicious
Updated on	2017-02-17 23:06:00

Options

Name	Docker monitoring
Folder	
Unique identifier (uid)	auto-generated
prometheus	Prometheus



	avg	current
bb2	0%	0%
bb1	0%	0%
cadvisor	0.65%	0.66%

	avg	current
bb2	52.00 KiB	52.00 KiB
bb1	188.00 KiB	188.00 KiB
cadvisor	40.79 MiB	40.75 MiB

9.Prometheus pushgaway 介绍

Pushgateway 是 Prometheus 生态中一个重要工具，使用它的原因主要是：

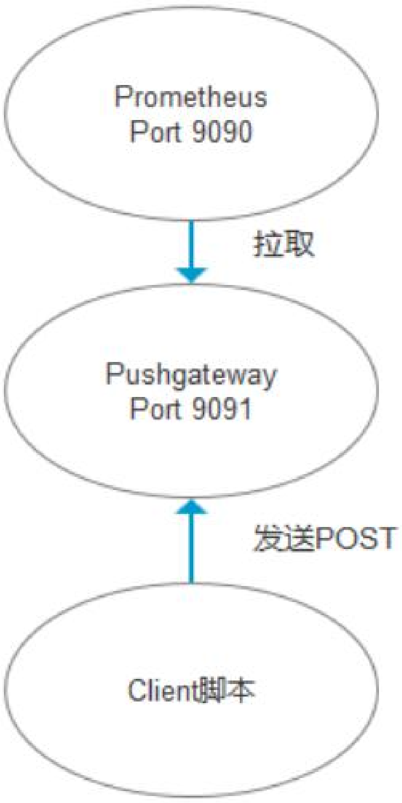
- Prometheus 采用 pull 模式，可能由于不在一个子网或者防火墙原因，导致 Prometheus 无法直接拉取各个 target 数据。
- 在监控业务数据的时候，需要将不同数据汇总，由 Prometheus 统一收集。

由于以上原因，不得不使用 pushgateway，但在使用之前，有必要了解一下它的一些弊端：

- 将多个节点数据汇总到 pushgateway，如果 pushgateway 挂了，受影响比多个 target 大。
- Prometheus 拉取状态 up 只针对 pushgateway，无法做到对每个节点有效。
- Pushgateway 可以持久化推送给它的所有监控数据。

因此，即使你的监控已经下线，prometheus 还会拉取到旧的监控数据，需要手动清理 pushgateway 不要的数据

数据流



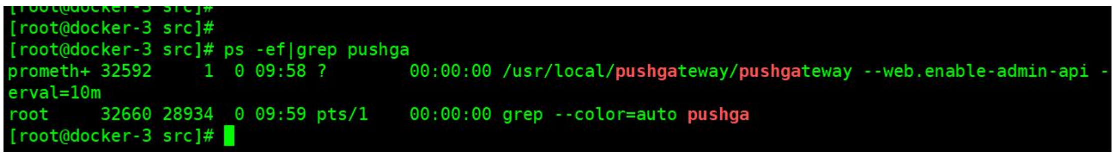
[root@docker-3 src]# wget -c

https://github.com/prometheus/pushgateway/releases/download/v1.2.0/pushgateway-1.2.0.linux-amd64.tar.gz

```
[root@docker-3 src]# tar xf pushgateway-1.2.0.linux-amd64.tar.gz
[root@docker-3 src]# mv pushgateway-1.2.0.linux-amd64 /usr/local/pushgateway-1.2.0
[root@docker-3 src]# ln -s /usr/local/pushgateway-1.2.0/ /usr/local/pushgateway
```

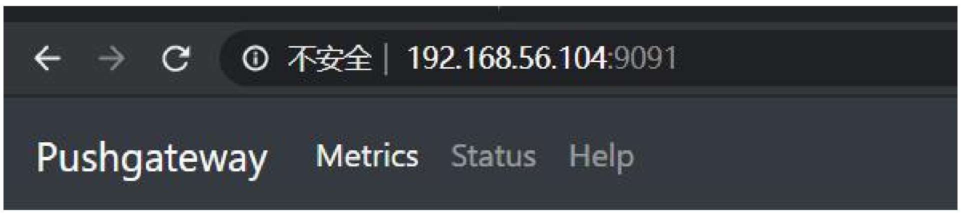
增加 systemd 启动方式

```
[root@docker-3 src]# cat /usr/lib/systemd/system/pushgateway.service
[Unit]
Description=prometheus
After=network.target

[Service]
User=prometheus
Group=prometheus
WorkingDirectory=/usr/local/pushgateway
ExecStart=/usr/local/pushgateway/pushgateway \
                      --web.enable-admin-api    \
                      --persistence.file="pushfile.txt" \
                      --persistence.interval=10m
[Install]
WantedBy=multi-user.target

[root@docker-3 src]# systemctl start pushgateway
```



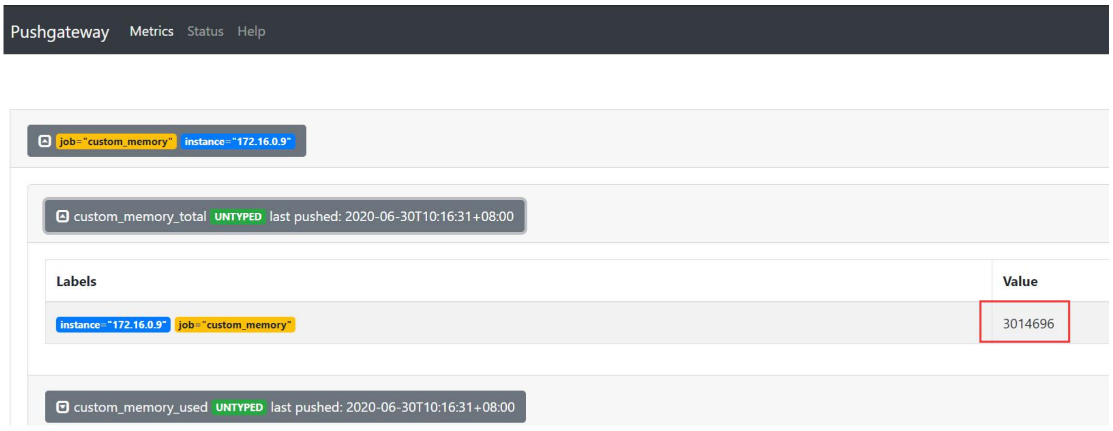
```
[root@docker-3 src]#
[root@docker-3 src]#
[root@docker-3 src]# ps -ef|grep pushga
prometh+ 32592     1  0 09:58 ?        00:00:00 /usr/local/pushgateway/pushgateway --web.enable-admin-api -
erval=10m
root     32660 28934  0 09:59 pts/1    00:00:00 grep --color=auto pushga
[root@docker-3 src]#
```

Web 访问

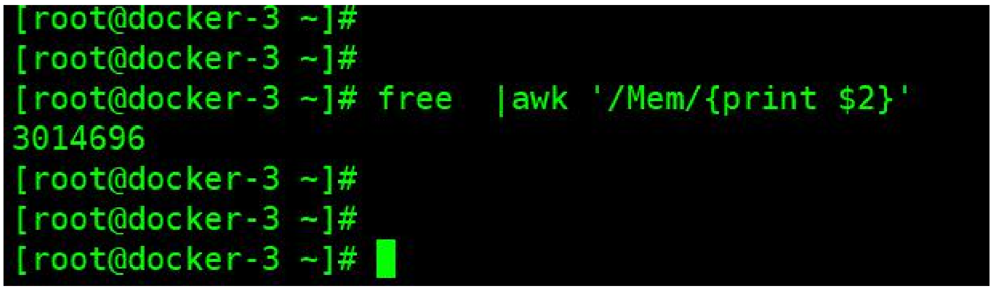


上报一个测试数据观察

```
[root@docker-3 ~]# cat push_memory.sh
#!/bin/bash
total_memory=$(free    |awk '/Mem/{print $2}')
used_memory=$(free    |awk '/Mem/{print $3}')

job_name="custom_memory"
instance_name="172.16.0.9"

cat <<EOF | curl --data-binary @- http://172.16.0.9:9091/metrics/job/$job_name/instance/$instance_name
#TYPE custom_memory_total    gauge
custom_memory_total $total_memory
#TYPE custom_memory_total    gauge
custom_memory_used $used_memory
EOF
[root@docker-3 ~]#
[root@docker-3 ~]# sh push_memory.sh
```



```
job="custom_memory" instance="172.16.0.9"
custom_memory_total UNTYPED last pushed: 2020-06-30T10:16:31+08:00
Labels                                              Value
instance="172.16.0.9" job="custom_memory"           3014696
custom_memory_used UNTYPED last pushed: 2020-06-30T10:16:31+08:00
```

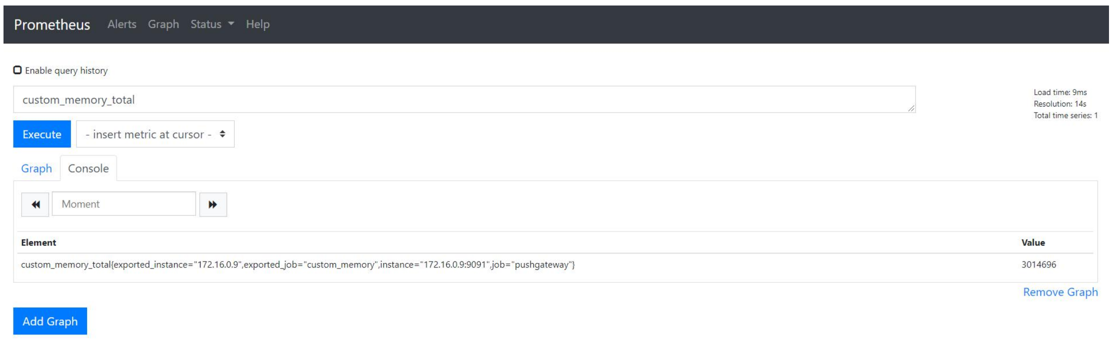


```
[root@docker-3 ~]#
[root@docker-3 ~]#
[root@docker-3 ~]# free   |awk '/Mem/{print $2}'
3014696
[root@docker-3 ~]#
[root@docker-3 ~]#
[root@docker-3 ~]#
```

Prometheus 增加 pushgateway 配置

```
  - job_name: 'pushgateway'
    static_configs:
      - targets: ['172.16.0.9:9091']
```

```
[root@docker-3 ~]# systemctl restart prometheus
```



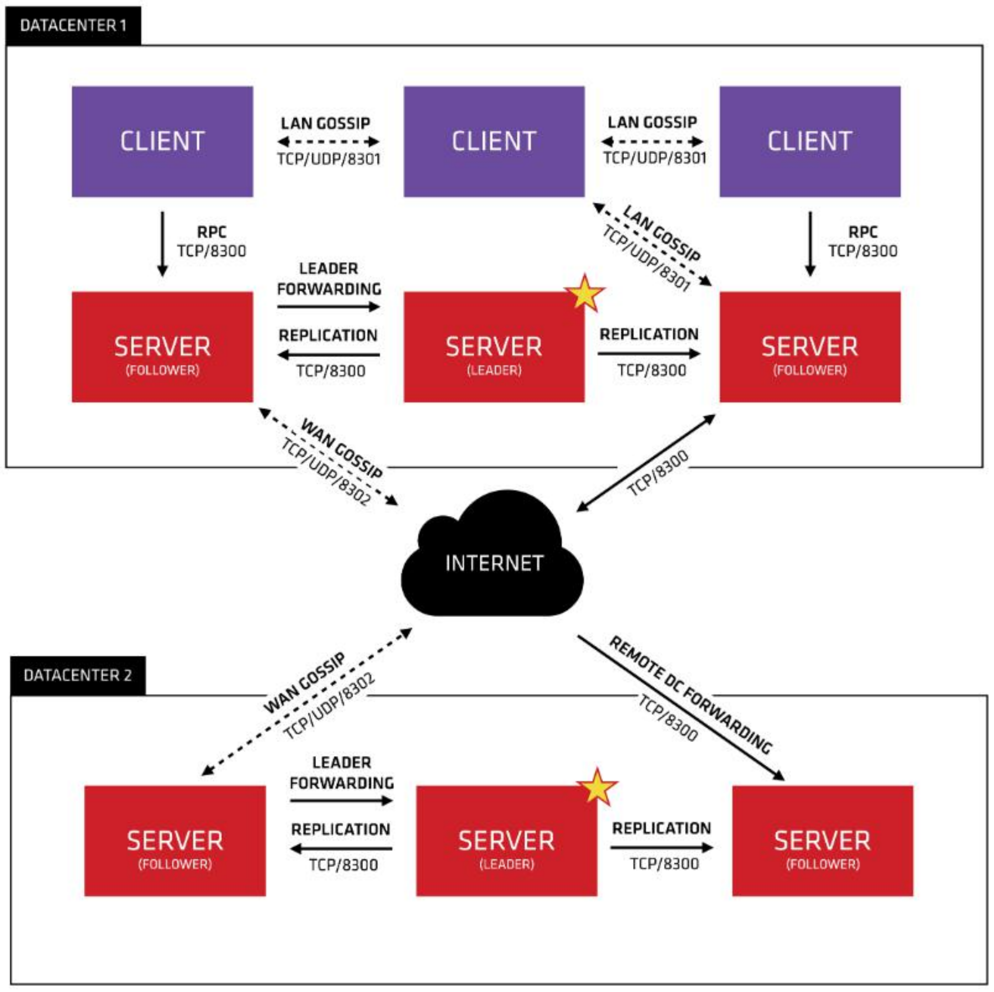
```
custom_memory_total

Load time: 9ms
Resolution: 14s
Total time series: 1

Element                                                                                                        Value
custom_memory_total{exported_instance="172.16.0.9",exported_job="custom_memory",instance="172.16.0.9:9091",job="pushgateway"}   3014696
```

10、Prometheus 自动化监控

10.1 Consul 分布式集群部署

1、Consul 介绍

Consul 是基于 GO 语言开发的开源工具，主要面向分布式，服务化的系统提供服务注册、服务发现和配置管理的功能。Consul 提供服务注册/发现、健康检查、Key/Value 存储、多数据中心和分布式一致性保证等功能。Prometheus 通过 Consul 可以很方便的实现服务自动发现和维护，同时 Consul 支持分布式集群部署，将大大提高了稳定性，通过 Prometheus 跟 Consul 集群二者结合起来，能够高效的进行数据维护同时保证系统稳定。



三个节点同时操作

```
[root@docker-1 src]# wget -c https://releases.hashicorp.com/consul/1.8.0/consul_1.8.0_linux_amd64.zip
[root@docker-1 src]# unzip consul_1.8.0_linux_amd64.zip
[root@docker-1 src]# mv consul /usr/local/bin/
[root@docker-1 src]# mkdir /data/
```

172.16.0.7

```
[root@docker-1 src]# nohup consul agent -server -bootstrap-expect=3 -data-dir=/data/consul -node=172.16.0.7 -bind=172.16.0.7 -client=0.0.0.0 -datacenter=shenzhen -ui &
```

172.16.0.8

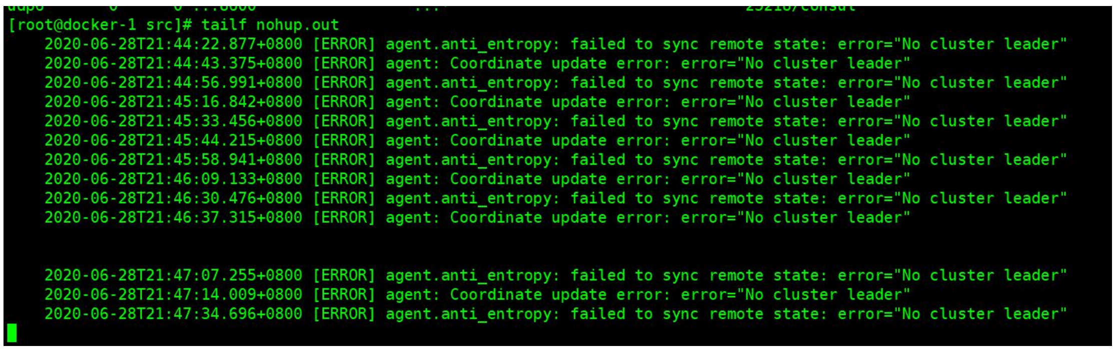
```
[root@docker-2 src]# nohup consul agent -server -bootstrap-expect=3 -data-dir=/data/consul -node=172.16.0.8 -bind=172.16.0.8 -client=0.0.0.0 -datacenter=shenzhen -ui &
```

172.16.0.9

```
[root@docker-3 src]# nohup consul agent -server -bootstrap-expect=3 -data-dir=/data/consul
```

-node=172.16.0.9 -bind=172.16.0.9 -client=0.0.0.0 -datacenter=shenzhen -ui &

此时



```
[root@docker-1 src]# tailf nohup.out
    2020-06-28T21:44:22.877+0800 [ERROR] agent.anti_entropy: failed to sync remote state: error="No cluster leader"
    2020-06-28T21:44:43.375+0800 [ERROR] agent: Coordinate update error: error="No cluster leader"
    2020-06-28T21:44:56.991+0800 [ERROR] agent.anti_entropy: failed to sync remote state: error="No cluster leader"
    2020-06-28T21:45:16.842+0800 [ERROR] agent: Coordinate update error: error="No cluster leader"
    2020-06-28T21:45:33.456+0800 [ERROR] agent.anti_entropy: failed to sync remote state: error="No cluster leader"
    2020-06-28T21:45:44.215+0800 [ERROR] agent: Coordinate update error: error="No cluster leader"
    2020-06-28T21:45:58.941+0800 [ERROR] agent.anti_entropy: failed to sync remote state: error="No cluster leader"
    2020-06-28T21:46:09.133+0800 [ERROR] agent: Coordinate update error: error="No cluster leader"
    2020-06-28T21:46:30.476+0800 [ERROR] agent.anti_entropy: failed to sync remote state: error="No cluster leader"
    2020-06-28T21:46:37.315+0800 [ERROR] agent: Coordinate update error: error="No cluster leader"


    2020-06-28T21:47:07.255+0800 [ERROR] agent.anti_entropy: failed to sync remote state: error="No cluster leader"
    2020-06-28T21:47:14.009+0800 [ERROR] agent: Coordinate update error: error="No cluster leader"
    2020-06-28T21:47:34.696+0800 [ERROR] agent.anti_entropy: failed to sync remote state: error="No cluster leader"
```

此时三台机器还未 join，不能算是一个集群，三台机器上的 consul 均不能正常工作，因为 leader 未选出。

集群节点加入
分别登录第 2 台和第 3 台虚拟机上执行如下命令，让 consul 加入集群：
172.16.0.8
[root@docker-2 src]# consul join 172.16.0.7
Successfully joined cluster by contacting 1 nodes.

172.16.0.9
[root@docker-3 src]# consul join 172.16.0.7
Successfully joined cluster by contacting 1 nodes.

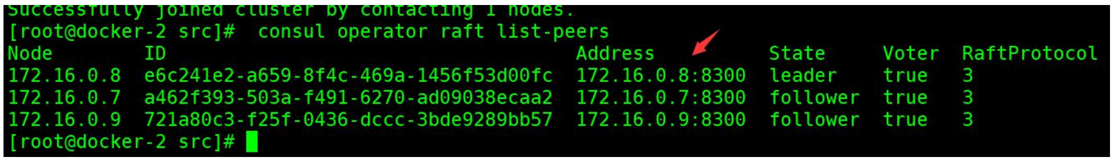
[root@docker-2 src]# tailf nohup.out 观察日志

2020-06-28T21:50:03.061+0800 [INFO] agent.server.raft: added peer, starting replication: peer=721a80c3-f25f-0436-dccc-3bde9289bb57

2020-06-28T21:50:03.062+0800 [INFO] agent.server: cluster leadership acquired

2020-06-28T21:50:03.062+0800 [INFO] agent.server: New leader elected: payload=172.16.0.8

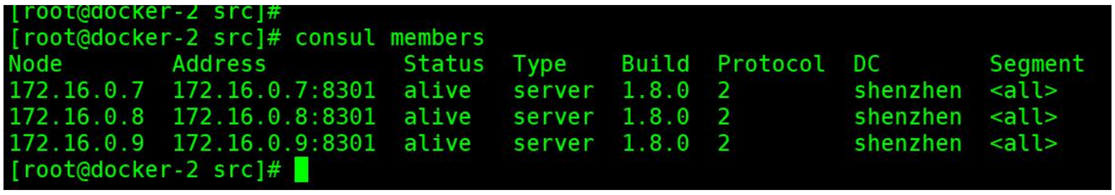
查看集群状态
[root@docker-2 src]# consul operator raft list-peers



```
Successfully joined cluster by contacting 1 nodes.
[root@docker-2 src]#  consul operator raft list-peers
Node        ID                                    Address         State     Voter  RaftProtocol
172.16.0.8  e6c241e2-a659-8f4c-469a-1456f53d00fc  172.16.0.8:8300  leader    true   3
172.16.0.7  a462f393-503a-f491-6270-ad09038ecaa2  172.16.0.7:8300  follower  true   3
172.16.0.9  721a80c3-f25f-0436-dccc-3bde9289bb57  172.16.0.9:8300  follower  true   3
[root@docker-2 src]#
```

查看成员状态

[root@docker-2 src]# consul members



```
[root@docker-2 src]#
[root@docker-2 src]# consul members
Node        Address          Status  Type    Build  Protocol  DC        Segment
172.16.0.7  172.16.0.7:8301  alive   server  1.8.0  2         shenzhen  <all>
172.16.0.8  172.16.0.8:8301  alive   server  1.8.0  2         shenzhen  <all>
172.16.0.9  172.16.0.9:8301  alive   server  1.8.0  2         shenzhen  <all>
[root@docker-2 src]#
```

集群测试

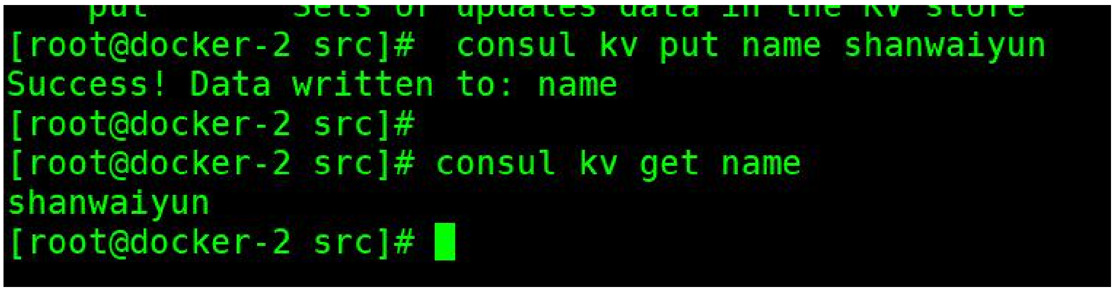
[root@docker-2 src]#   consul kv put name shanwaiyun

Success! Data written to: name

[root@docker-2 src]#

[root@docker-2 src]# consul kv get name

Shanwaiyun

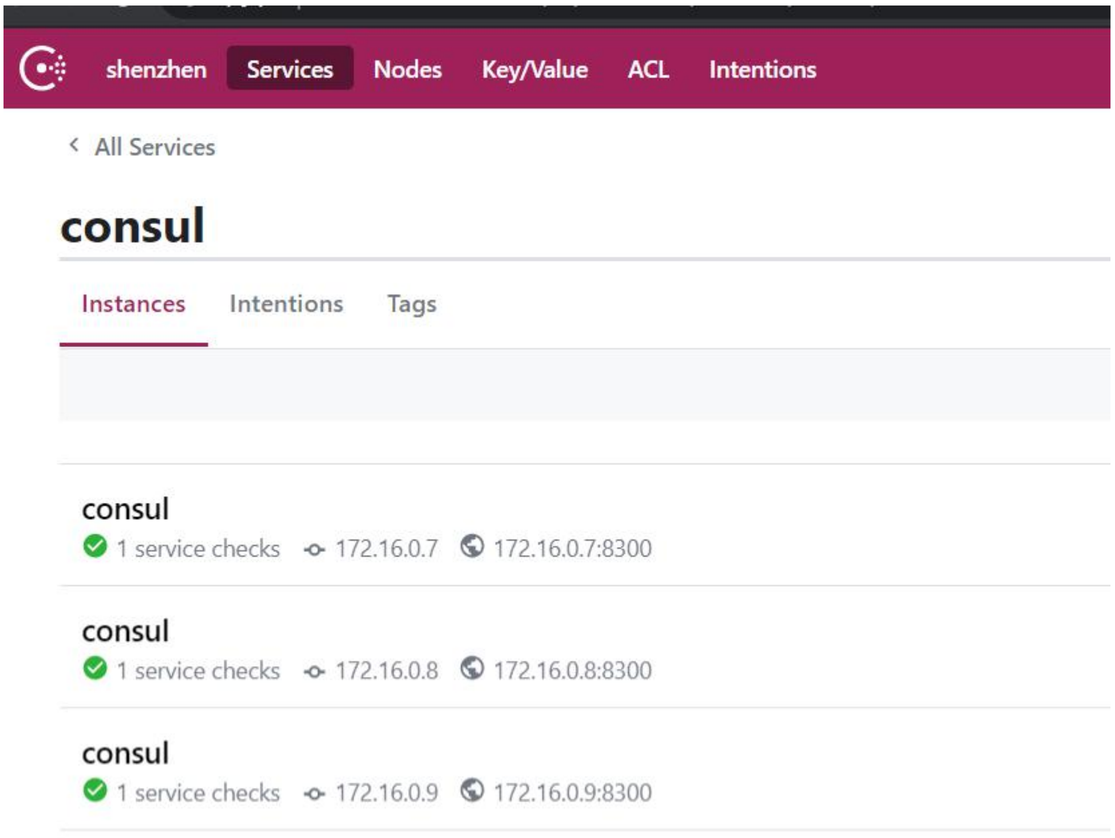


```
    put         Sets or updates data in the KV store
[root@docker-2 src]#  consul kv put name shanwaiyun
Success! Data written to: name
[root@docker-2 src]#
[root@docker-2 src]# consul kv get name
shanwaiyun
[root@docker-2 src]#
```

其他两台机器查看该 key 值   也是返回 shanwaiyun 这个   说明 key 值已经在集群中同步

Web 界面访问

http://192.168.56.104:8500/



consul

Instances Intentions Tags

consul
1 service checks 172.16.0.7 172.16.0.7:8300

consul
1 service checks 172.16.0.8 172.16.0.8:8300

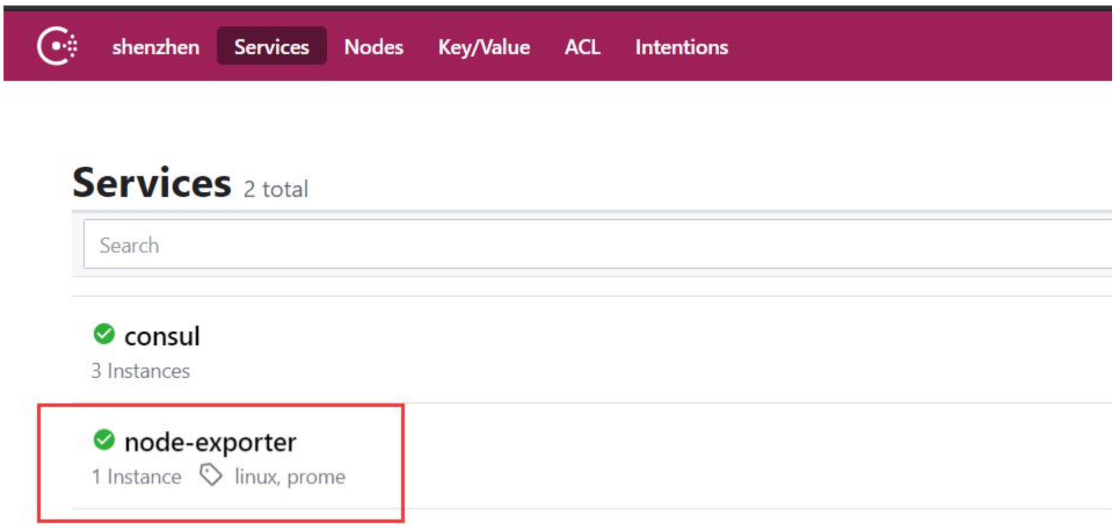
consul
1 service checks 172.16.0.9 172.16.0.9:8300

10.2 Prometheus 与 consul 整合

1、通过在 consul 注册服务或注销服务（监控 targets）
2、Prometheus 一直监视（watch）consul 服务，当发现 consul 中符合要求的服务有新变化是更新 Prometheus 监控对象

使用 API 把这里的启动的 node_exporter 服务注册到 consul

```
[root@docker-3 src]# curl -X PUT -d '{"id": "node-exporter","name": "node-exporter","address": "172.16.0.9","port": 9100,"tags": ["linux","prome"],"checks": [{"http": "http://172.16.0.9:9100/metrics", "interval": "5s"}]}' http://172.16.0.9:8500/v1/agent/service/register
```

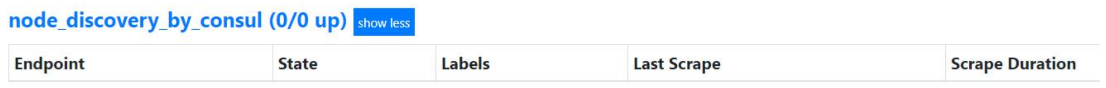


服务注销
如果想要注销这个服务，可以直接通过接口的方式删除 node-exporter 即可：
curl -X PUT http://172.16.0.9:8500/v1/agent/service/deregister/node-exporter

配置 prometheus 实现自动发现：

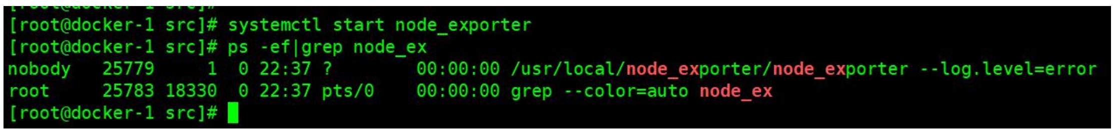
```
 - job_name: 'node_discovery_by_consul'
    metrics_path: /metrics
    scheme: http
    consul_sd_configs:
     - server: 172.16.0.9:8500
       services:
         - node-exporter
```

[root@docker-3 src]# /usr/local/prometheus/promtool check config /usr/local/prometheus/prometheus.yml
[root@docker-3 src]# systemctl restart prometheus



Endpoint	State	Labels	Last Scrape	Scrape Duration

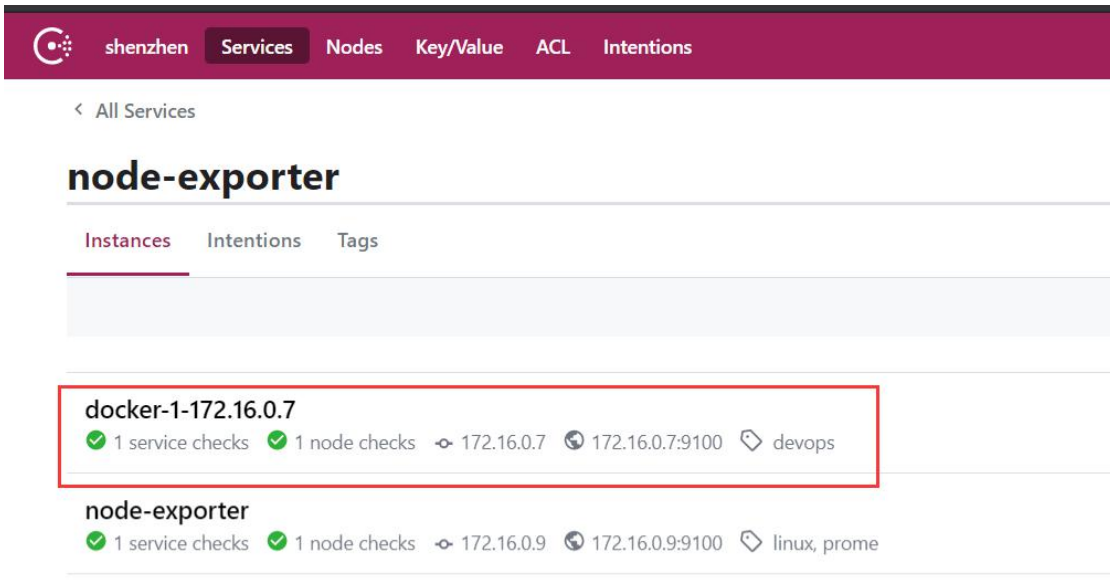
现在将 172.16.0.7 自动加入服务发现
先安装 node_exporter，过程略过



```
[root@docker-1 src]# systemctl start node_exporter
[root@docker-1 src]# ps -ef|grep node_ex
nobody   25779     1  0 22:37 ?        00:00:00 /usr/local/node_exporter/node_exporter --log.level=error
root     25783 18330  0 22:37 pts/0    00:00:00 grep --color=auto node_ex
[root@docker-1 src]# 
```

# 添加一个 node_exporter 的监控
[root@docker-1 src]# curl -X PUT -d '{"id": "docker-1-172.16.0.7","name":

"node-exporter","address": "172.16.0.7","port": 9100,"tags": ["devops"],"checks": [{"http": "http://172.16.0.7:9100/metrics", "interval": "5s"}]}' http://172.16.0.7:8500/v1/agent/service/register

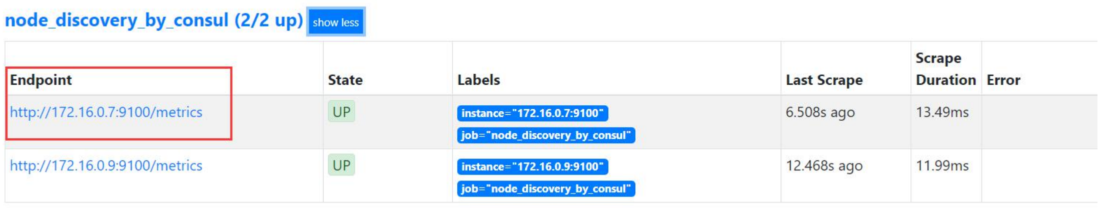


node-exporter

docker-1-172.16.0.7
1 service checks · 1 node checks · 172.16.0.7 · 172.16.0.7:9100 · devops

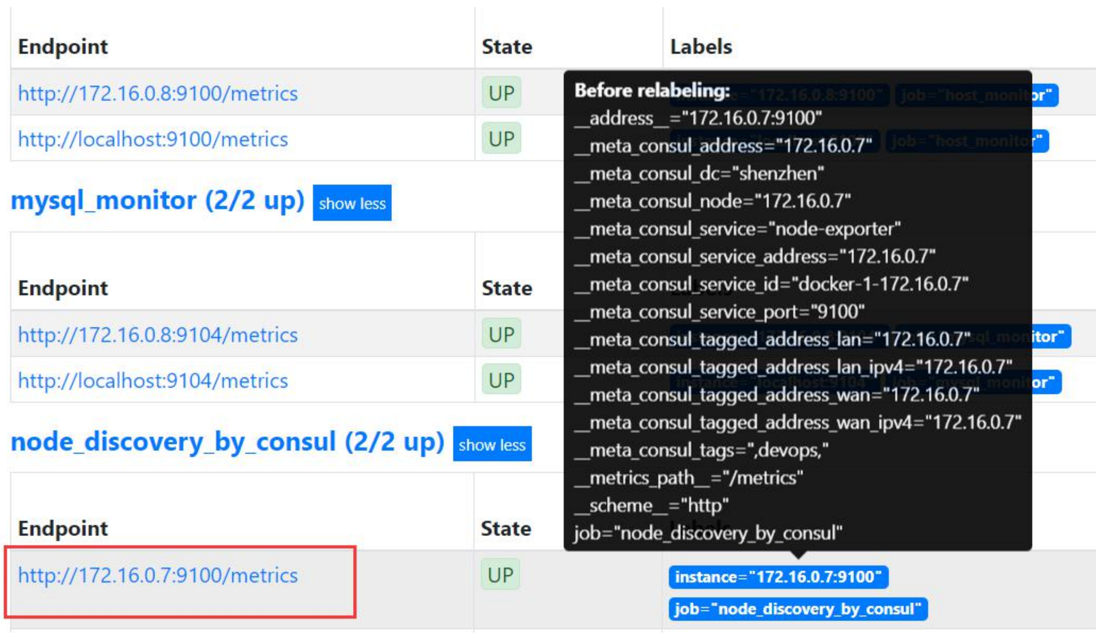
node-exporter
1 service checks · 1 node checks · 172.16.0.9 · 172.16.0.9:9100 · linux, prome

可以看到该节点被自动加入 prometheus 监控了



node_discovery_by_consul (2/2 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://172.16.0.7:9100/metrics	UP	instance="172.16.0.7:9100" job="node_discovery_by_consul"	6.508s ago	13.49ms	
http://172.16.0.9:9100/metrics	UP	instance="172.16.0.9:9100" job="node_discovery_by_consul"	12.468s ago	11.99ms	



Endpoint	State	Labels
http://172.16.0.8:9100/metrics	UP	
http://localhost:9100/metrics	UP	

mysql_monitor (2/2 up)

Endpoint	State
http://172.16.0.8:9104/metrics	UP
http://localhost:9104/metrics	UP

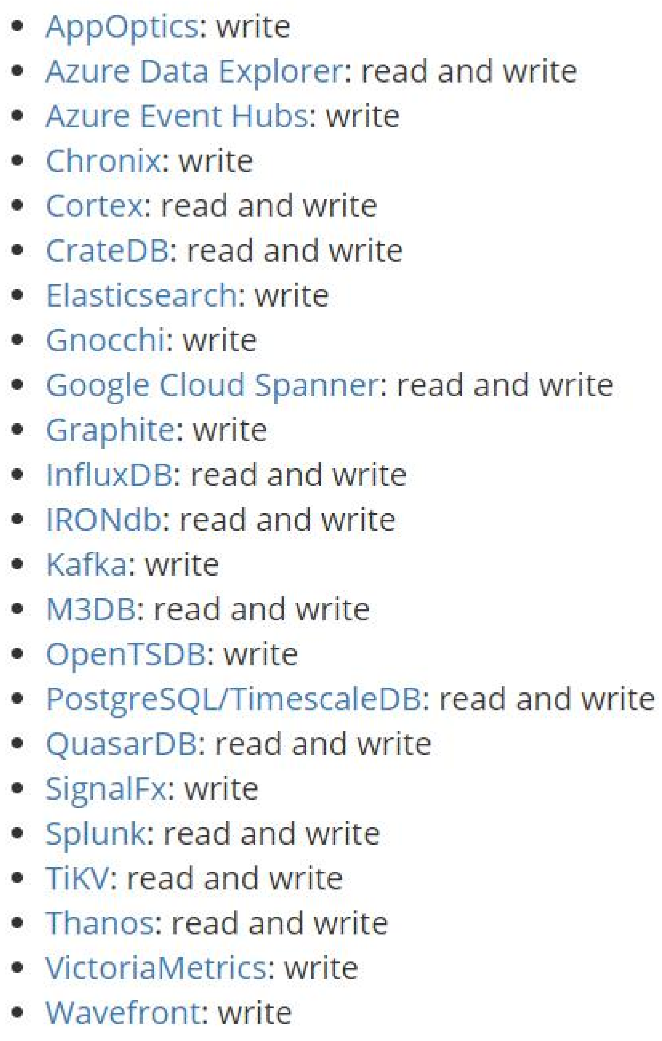
node_discovery_by_consul (2/2 up)

Endpoint	State	Labels
http://172.16.0.7:9100/metrics	UP	instance="172.16.0.7:9100" job="node_discovery_by_consul"

```
Before relabeling:
__address__="172.16.0.7:9100"
__meta_consul_address="172.16.0.7"
__meta_consul_dc="shenzhen"
__meta_consul_node="172.16.0.7"
__meta_consul_service="node-exporter"
__meta_consul_service_address="172.16.0.7"
__meta_consul_service_id="docker-1-172.16.0.7"
__meta_consul_service_port="9100"
__meta_consul_tagged_address_lan="172.16.0.7"
__meta_consul_tagged_address_lan_ipv4="172.16.0.7"
__meta_consul_tagged_address_wan="172.16.0.7"
__meta_consul_tagged_address_wan_ipv4="172.16.0.7"
__meta_consul_tags=",devops,"
__metrics_path__="/metrics"
__scheme__="http"
job="node_discovery_by_consul"
```

注销节点
[root@docker-3 src]# curl --request PUT "http://172.16.0.7:8500/v1/agent/service/deregister/docker-1-172.16.0.7"
##docker-1-172.16.0.7 代表 id

11、Prometheus 远端存储



- AppOptics: write
- Azure Data Explorer: read and write
- Azure Event Hubs: write
- Chronix: write
- Cortex: read and write
- CrateDB: read and write
- Elasticsearch: write
- Gnocchi: write
- Google Cloud Spanner: read and write
- Graphite: write
- InfluxDB: read and write
- IRONdb: read and write
- Kafka: write
- M3DB: read and write
- OpenTSDB: write
- PostgreSQL/TimescaleDB: read and write
- QuasarDB: read and write
- SignalFx: write
- Splunk: read and write
- TiKV: read and write
- Thanos: read and write
- VictoriaMetrics: write
- Wavefront: write

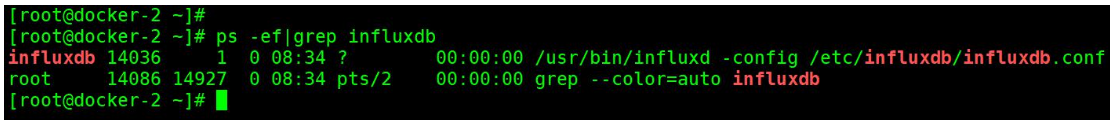
https://docs.influxdata.com/influxdb/v1.8/supported_protocols/prometheus

11.1 Influxdb 部署

```
cat <<EOF | sudo tee /etc/yum.repos.d/influxdb.repo
[influxdb]
name = InfluxDB Repository - RHEL \$releasever
baseurl                                                                      =
https://repos.influxdata.com/rhel/\$releasever/\$basearch/stable
enabled = 1
gpgcheck = 1
gpgkey = https://repos.influxdata.com/influxdb.key
EOF
```

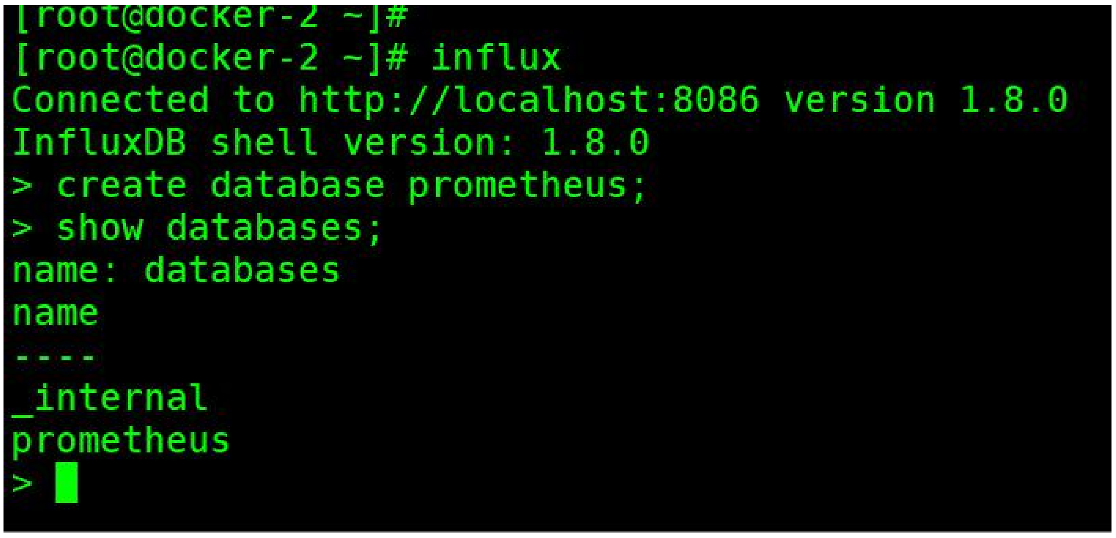
```
[root@docker-2 ~]#  yum install influxdb -y
```

```
[root@docker-2 ~]# systemctl start influxdb  ##启动
```



```
[root@docker-2 ~]#
[root@docker-2 ~]# ps -ef|grep influxdb
influxdb 14036     1  0 08:34 ?        00:00:00 /usr/bin/influxd -config /etc/influxdb/influxdb.conf
root     14086 14927  0 08:34 pts/2    00:00:00 grep --color=auto influxdb
[root@docker-2 ~]#
```

创建 prometheus 数据库



```
[root@docker-2 ~]#
[root@docker-2 ~]# influx
Connected to http://localhost:8086 version 1.8.0
InfluxDB shell version: 1.8.0
> create database prometheus;
> show databases;
name: databases
name
----
_internal
prometheus
>
```

修改服务脚本指定存储路径

--storage.tsdb.path=/usr/local/prometheus/data

```
[root@docker-3 prometheus]# cat /usr/lib/systemd/system/prometheus.service
[Unit]
Description=https://prometheus.io
```

```
[Service]
Restart=on-failure
ExecStart=/usr/local/prometheus/prometheus
--config.file=/usr/local/prometheus/prometheus.yml
--storage.tsdb.path=/usr/local/prometheus/data

[Install]
WantedBy=multi-user.target
```

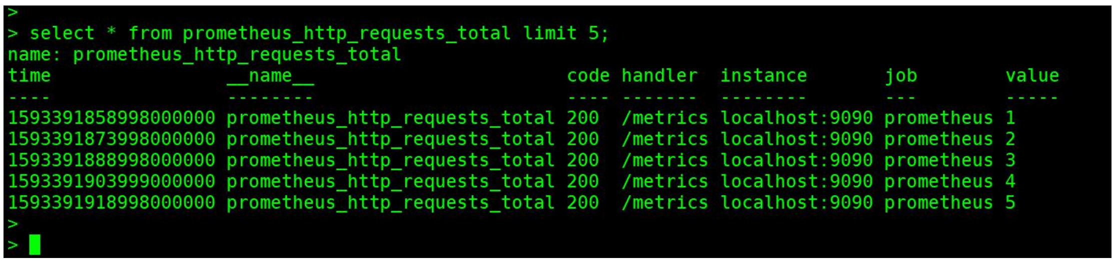
配置 prometheus 添加远程读写

```
remote_write:
  - url: "http://172.16.0.8:8086/api/v1/prom/write?db=prometheus"

remote_read:
  - url: "http://172.16.0.8:8086/api/v1/prom/read?db=prometheus"
```

```
[root@docker-3 prometheus]# systemctl restart prometheus
```

验证 influxdb 是否有数据写入

```
> use prometheus
> show measurements
> select * from prometheus_http_requests_total limit 5;
```



```
>
> select * from prometheus_http_requests_total limit 5;
name: prometheus_http_requests_total
time                __name__                       code handler  instance       job        value
----                --------                       ---- -------  --------       ---        -----
1593391858998000000 prometheus_http_requests_total 200  /metrics localhost:9090 prometheus 1
1593391873998000000 prometheus_http_requests_total 200  /metrics localhost:9090 prometheus 2
1593391888998000000 prometheus_http_requests_total 200  /metrics localhost:9090 prometheus 3
1593391903999000000 prometheus_http_requests_total 200  /metrics localhost:9090 prometheus 4
1593391918998000000 prometheus_http_requests_total 200  /metrics localhost:9090 prometheus 5
>
>
```

验证数据可靠性：

停止 Prometheus 服务。同时删除 Prometheus 的 data 目录,重启 Prometheus。打开 Prometheus UI 如果配置正常，Prometheus 可以正常查询到本地存储以删除的历史数据记录。

```
[root@docker-3 prometheus]# systemctl stop prometheus
[root@docker-3 prometheus]# mv data/ /tmp/
```