**Paas监控部署手册**

|  |  |  |  |
| --- | --- | --- | --- |
| 修订日期 | 修订内容 | 版本号 | 修改人 |
|  |  |  |  |
|  |  |  |  |

# 功能列表

Prometheus系统主要的三部分：prometheus、alertmanager、\*\_exporter（多个）

相关资料可参考：https://github.com/yunlzheng/prometheus-book

<https://www.lijiaocn.com/%E9%A1%B9%E7%9B%AE/2018/08/03/prometheus-usage.html>

|  |  |
| --- | --- |
| **序号** | **功能概要** |
| 1 | Prometheus环境 |
| 2 | 监控插件部署 |
| 3 | grafana环境部署 |
| 4 | 告警项目部署 |

# 1 Prometheus环境部署

prometheus是最主要的组件，负责采集数据，将告警发送到alertmanager，alertmanager再将告警以各种形式送出

## 1.1部署准备

1. 操作系统：centos7.2 x86\_64版本及以上

2）部署安装包 prometheus-2.2.1.linux-amd64.tar.gz。下载地址：<https://github.com/prometheus/prometheus/releases>

也可以通过下载程序：<https://prometheus.io/download/>

3）从部署的服务器中选取一台机器作为源机（保证源机和其他机器网络互通）。将从版本发布中心下载的部署安装包prometheus-x.x.xlinux-amd64.tar.gz（部署安装包可从版本发布中心下载）上传到源机的用户目录/{monitoruser}下的指定目录（即~目录的指定目录），并解压到该目录下

## 1.2 Prometheus部署

1）下载的prometheus-2.2.1.linux-amd64.tar.gz是二进制包，包解压后，进入~/prometheus-2.2.1.linux-amd64.tar.gz 修改配置文件，添加被检测服务器



配置文件配置项：

* 采集频率
* JOB

2）在目录~/ prometheus-2.2.1.linux-amd64.tar.gz下



启动命令 ./prometheus --web.read-timeout=5m --query.timeout=5m --web.max-connections=512 --web.listen-address=0.0.0.0:9090 --log.level=error --web.enable-lifecycle --storage.tsdb.retention=100d --config.file=prometheus.yml & (启动端口根据实际情况变动)

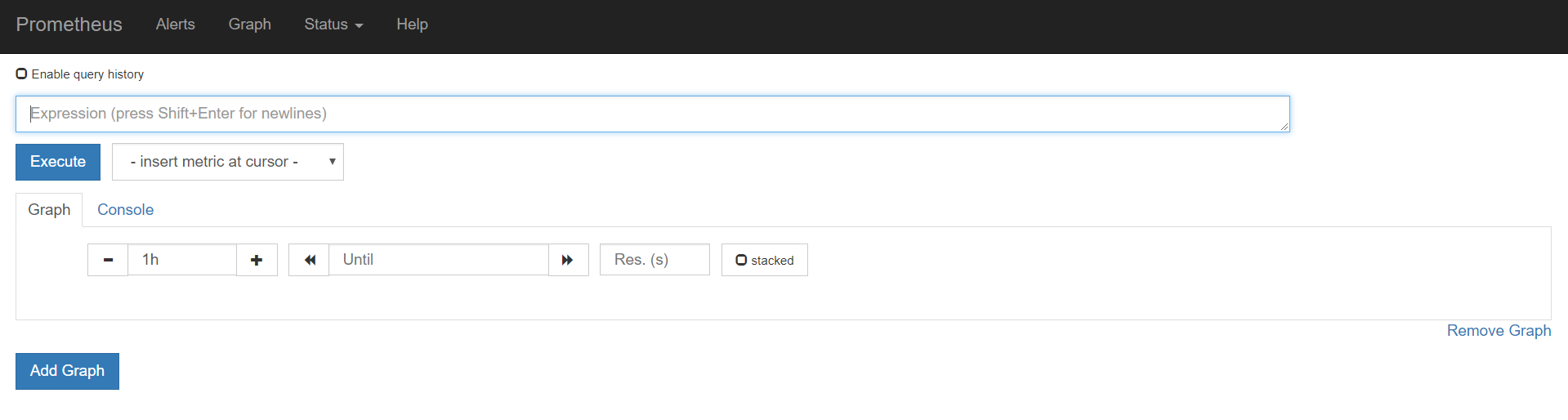
 将脚本放入 /etc/init.d，

chkconfig --add prometheus.service

chkconfig prometheus.service on

3）使用Prometheus自带的表达式浏览器检测服务是否启动。若未启动，无法打开页面

打开<http://ip:port/graph>



## 1.3 prometheus联邦部署

**略…**

1）安装介质同1.2。

打开prometheus.yml文件具体配置如下所示：



1. 更改配置后，启动prometheus联邦。

在目录~/ prometheus-2.2.1.linux-amd64.tar.gz下

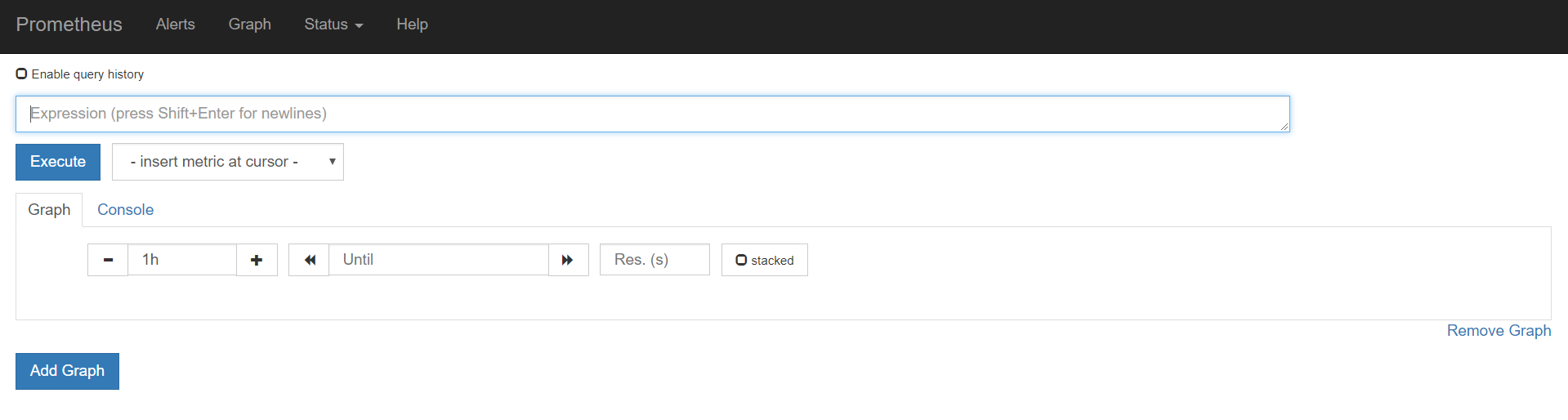


启动命令：./prometheus --web.read-timeout=120s --query.timeout=120s --web..max-connections=512 --web.listen-address=0.0.0.0:9001 --log.level=error --web.enable-lifecycle --storage.tsdb.retention=60d --config.file=prometheus.yml &

（启动端口根据实际情况变动）

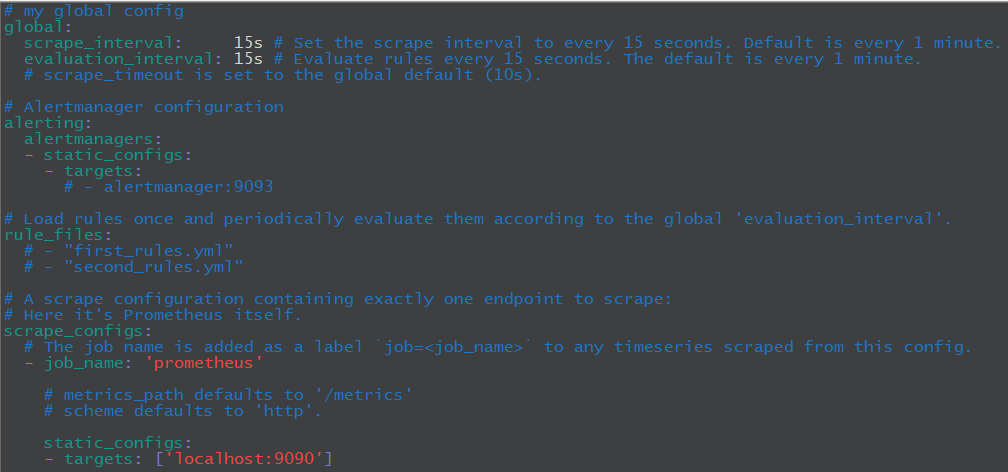
3）使用Prometheus自带的表达式浏览器检测服务是否启动。若未启动，无法打开页面

打开<http://ip:port/graph>



## 1.4 Prometheus.yml配置文件说明

配置文件是yaml格式，默认如下：



global:是常规配置，意思是，没15秒采集一次数据，每15秒做一次告警检测

alerting：告警模块alertmanager的配置

rule\_files: 指定加载的告警规则文件

scrape\_configs：指定prometheus要监控的目标

scrape\_config中每个监控目标是一个job，但job的类型有很多种，例如:

静态地址指定每一个目录：

- job\_name: 'node\_exporter'

static\_configs:

- targets: ['192.168.10.12:9100','192.168.10.13:9100','192.168.10.14:9100']

- job\_name: 'blackbox\_exporter'

static\_configs:

- targets: ['192.168.10.13:9115']

服务发现的方式，即动态发现目标：

服务发现是prometheus最强大的功能之一，这个功能配合[relabel\_config](https://prometheus.io/docs/prometheus/latest/configuration/configuration/#%3Crelabel_config%3E)、[\*\_exporter](https://prometheus.io/docs/instrumenting/exporters/)可以做成很多事情

### 1.4.1 动态生成配置文件

当targets的数量达到几百上千之后，配置文件看起来就会特别冗余, 一个大而冗余的主配置文件，切分成一个个小的配置文件。Prometheus这里用的方法就是基于文件的服务发现–“file\_sd\_config”。

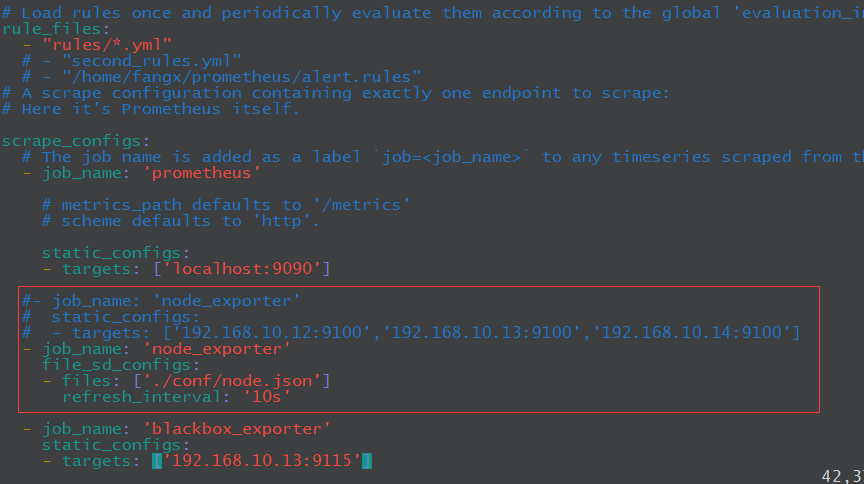
**1）修改prometheus.yml**

- job\_name: 'node\_exporter'

file\_sd\_configs:

- files: ['./conf/node.json'] #可以使用\*号,如\*.json

refresh\_interval: '10s'



**2)添加json文件**

cat conf/node.json

[

{

"targets":["192.168.10.12:9100"],

"lables":{

"hostname":"fan12"

}

},

{

"targets":["192.168.10.13:9100"],

"lables":{

"hostname":"fan13"

}

},

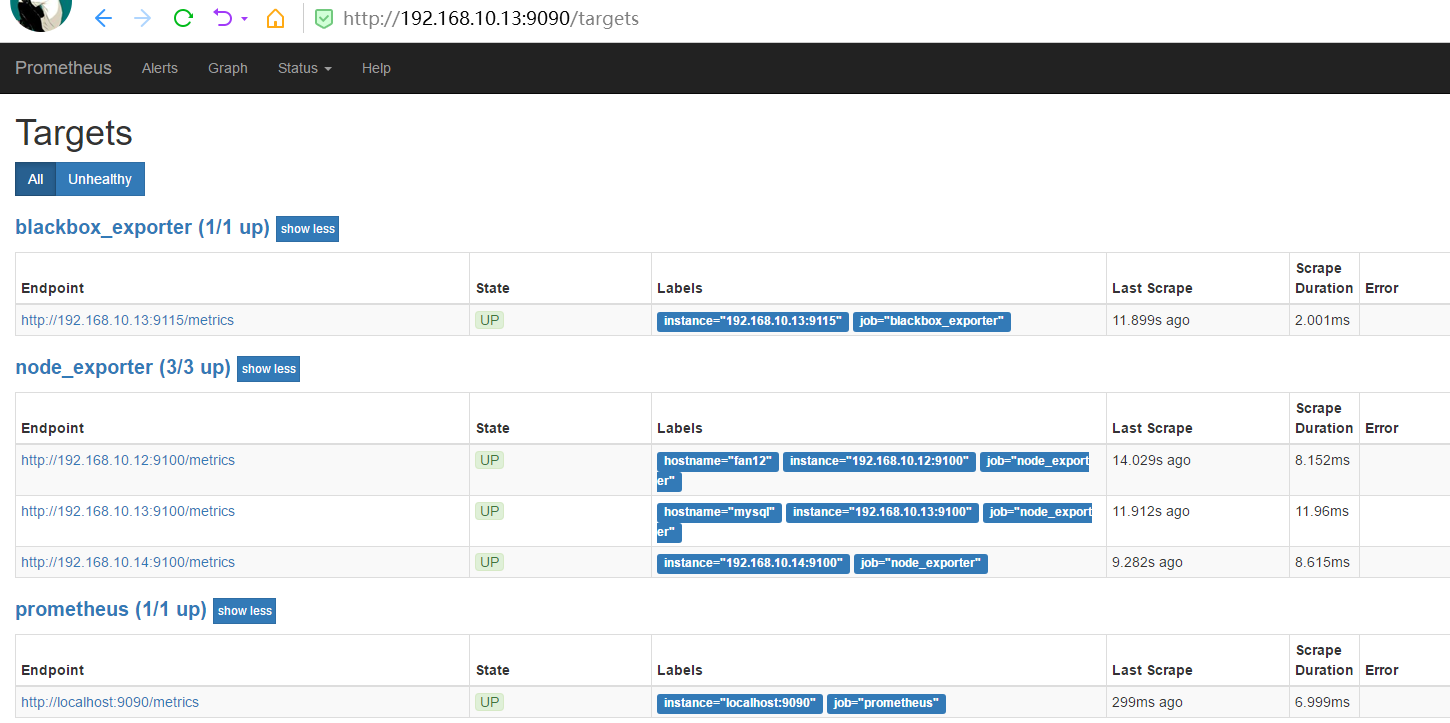
{

"targets":["192.168.10.14:9100"]

}

]

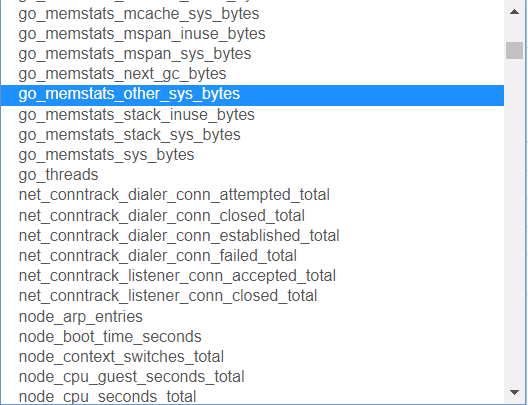
3) 登陆prometheus，10秒后就会发现node\_exporter节点，以后配置文件不需要配置prometheus.yml了，只需要配置files指定目录下的json文件就可以了。配置后不需要重启prometheus。



## 1.5 prometheus的查询语句

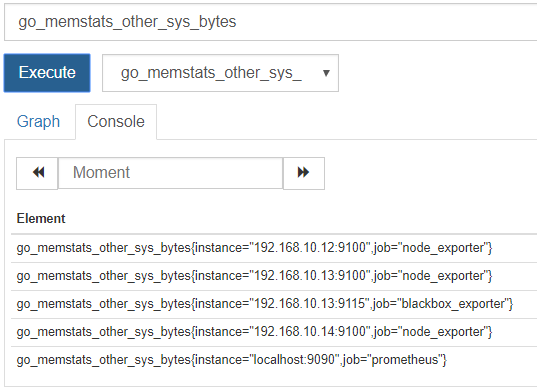
prometheus的查询语句非常重要，除了用来查询数据，还用于配置告警规则

查询语句就是指标的名称：



可以通过标签筛选：指标名称后面跟上大括号{ } 如下：

标签有4个操作符 =(等于) !=(不等于) =~(正则等于) !~(正则不等于)



# 2 监控插件部署

在Prometheus的架构设计中，Prometheus Server主要负责数据的收集，存储并且对外提供数据查询支持。而实际的监控样本数据的收集则是由Exporter完成。Exporter可以是一个独立运行的进程，对外暴露一个用于获取监控数据的HTTP服务。 Prometheus Server只需要定时从这些Exporter暴露的HTTP服务获取监控数据即可。

## 2.1 node\_exporter安装

### 2.1.1 部署准备

1. 安装包：node\_exporter-0.15.1.linux-amd64.tar.gz

安装包下载地址：<https://github.com/prometheus/node_exporter/releases>

作用：对主机的关键度量指标状态监控(采集到主机的运行指标如CPU, 内存，磁盘等信息),部署在被检测机器上。

yum install -y lm\_sensors

### 2.1.2 部署安装步骤

1）将安装包上传到被监测服务器的用户目录/{monitoruser}下的指定目录（即~目录的指定目录），并解压至该目录下

2）启动node\_exporter ,

在node\_exporter目录下执行启动命令：



命令：nohup ./node\_exporter & 默认端口9100



3）修改prometheus的配置文件，如1.3配置文件所示。**略…后续讲解**

# 采集node exporter监控数据

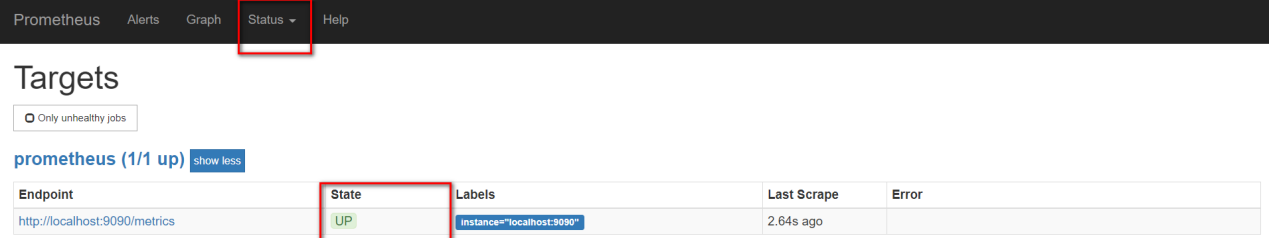
- job\_name: 'system\_13'

static\_configs:

- targets: ['192.168.10.13:9100']

4）重新启动prometheus.，登陆prometheus自带表达式浏览器，如1.4所示

在Status—>Targets页面上。可以看到监控的Target。状态时up，说明node\_exporter数据已被prometheus获取（显示会有一段时间间隔，需多刷新几次）。



5)grafana图形化

  或搜索模板1860 8919

## 2.2 blackbox\_exporter 安装

### 2.2.1 部署准备

1）安装包：blackbox\_exporter-0.12.0.linux-amd64.tar.gz

下载地址：<https://github.com/prometheus/blackbox_exporter/releases/tag/v0.12.0>

作用：HTTP, HTTPS, DNS, TCP ICMP等协议的网络状态探测的采取器,部署在被检测机器上

### 2.2.2 安装步骤

1）将安装包上传到被监测服务器的用户目录/{monitoruser}下的指定目录（即~目录的指定目录），并解压至该目录下

2）用root权限用户执行进行赋权

setcap cap\_net\_raw+ep /{monitoruser}/blackbox\_exporter-0.12.0.linux-amd64/blackbox\_exporter

1. 启动blackbox\_exporter ,

在blackbox\_exporter目录下执行启动命令： 

命令：nohup ./blackbox\_exporter & 默认端口9115

4）修改prometheus的配置文件，如1.3配置文件所示。**略**

5）重新启动prometheus，如2.1.2第四步所示

## 2.3 mysqld\_exporter安装

mysqld\_Pexporter下载地址：<https://github.com/prometheus/mysqld_exporter/releases>

本文档使用版本：mysqld\_exporter-0.11.0.linux-amd64.tar.gz

1）上传到对应目录，解压即可用。

2）mysqld\_exporter需要连接Mysql

首先为它创建用户并赋予所需要的权限

GRANT REPLICATION CLIENT ON \*.\* TO 'fangx'@'localhost' identified by '123456';

GRANT SELECT, PROCESS ON \*.\* TO 'fangx'@'localhost';

3) 创建my.cnf文件

[client]

user=exporter

password=123456

4）运行mysqld\_exporter

./mysqld\_exporter --config.my-cnf="./my.cnf" & #默认端口9104

5) prometheus配置,在后面加入job标签

- job\_name: 'mysql'

static\_configs:

- targets: ['192.168.10.12:9104']

labels:

instance: master

6) grafana图形化



7)告警配置参考



## 2.4 SNMP exproter 安装

主要监控网络设备

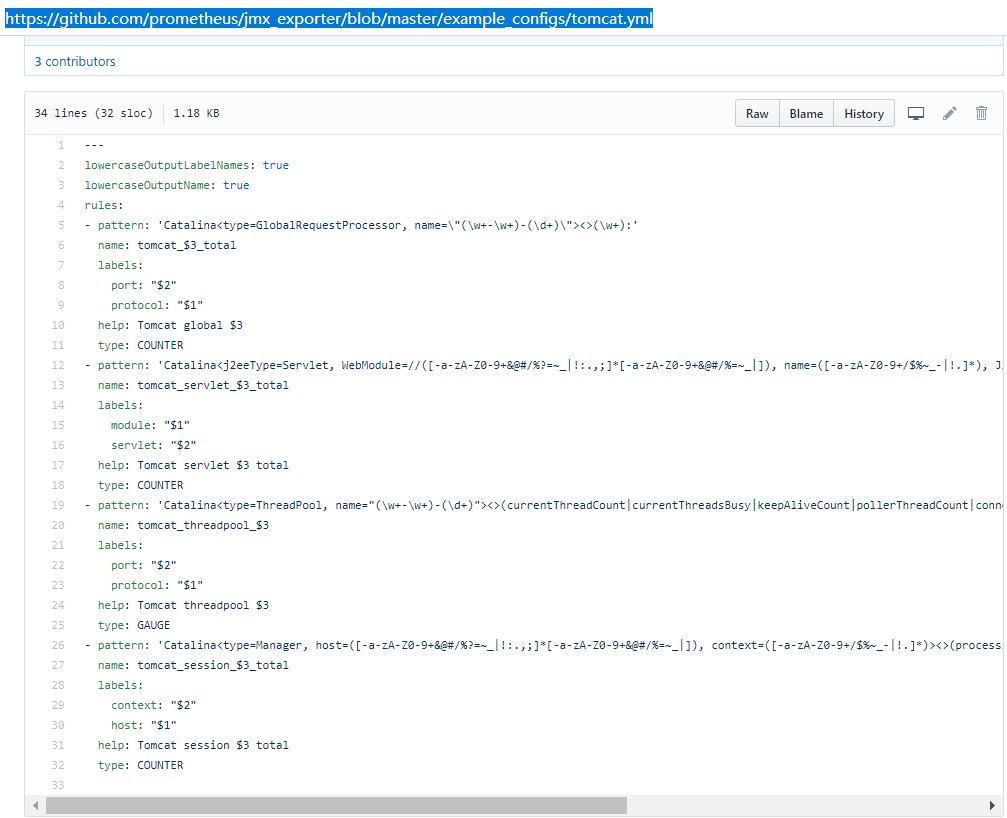
## 2.5 JMX\_exporter安装

参考：https://www.jianshu.com/p/adada9c1f7dd

<https://www.jianshu.com/p/271bf7301b83>

1）本文档使用版本jmx\_prometheus\_javaagent-0.9.jar下载地址：https://repo1.maven.org/maven2/io/prometheus/jmx/jmx\_prometheus\_javaagent

2）配置文件下载地址：https://github.com/prometheus/jmx\_exporter/blob/master/example\_configs/tomcat.yml



3) 部署JMX\_exporter

创建JMX\_exporter目录：mkdir JMX\_exporter, 把jmx\_prometheus\_javaagent-0.9.jar和tomcat.yml 放入此目录

4) 配置tomcat的catalina.sh文件或者创建setenv.sh脚本，让jmx\_exporter和tomcat一起启动，在里面添加如下内容

JAVA\_OPTS="-javaagent:/home/jmx\_exporter/jmx\_prometheus\_javaagent-0.9.jar=30013:/home/jmx\_exporter/tomcat.yml"

指定运行端口30013，如果tomcat有多实例要分配不同的端口

5）配置prometheus配置文件

- job\_name: 'jmx\_tomcat'

static\_configs:

- targets: ['192.168.10.14:30013']

6）grafana图形化

参考模板： 或者3457

## 2.6 监控redis

1）文档中使用版本是：redis\_exporter-v0.26.0.linux-amd64.tar.gz

下载地址：<https://github.com/oliver006/redis_exporter/releases/>

2）启动redis, 默认端口9121

## 无密码./redis\_exporter redis//192.168.1.120:6379 &

## 有密码./redis\_exporter -redis.addr 192.168.1.120:6379 -redis.password 123456 &

3）配置prometheus

- job\_name: 'redis'

static\_configs:

- targets: ['192.168.10.12:9121']

4）图形化

5）告警规则参考



## 2.7 监控nginx

1）官方没有提供nginx直接可用的exporter，这里使用第三方，主要使用安装包：

nginx-module-vts（Nginx的监控模块，能够提供JSON格式的数据产出）下载地址：<https://github.com/vozlt/nginx-module-vts>或用git clone git://github.com/vozlt/nginx-module-vts.git下载

nginx-vts-exporter要用于收集Nginx的监控数据，并给Prometheus提供监控接口，默认端口号9913。下载地址：<https://github.com/hnlq715/nginx-vts-exporter/releases>，本文档使用版本：nginx-vts-exporter-0.10.3.linux-amd64.tar.gz

2）编译安装nginx时需要加入nginx-module-vts模块,然后配置nginx.conf加入如下

location /status {

vhost\_traffic\_status\_display;

vhost\_traffic\_status\_display\_format html;

}

3）启动nginx-vts-exporter

.\nginx-vts-exporter -nginx.scrape\_uri=http://ip/status/format/json &

4)prometheus配置相应监控job

5）grafana图形化

使用官方模板 2949 或者 1623

## 2.8 监控zookeeper

参考：https://blog.csdn.net/qq\_25934401/article/details/84345905

## 2.9 监控kafka

参考：https://blog.csdn.net/qq\_25934401/article/details/84840740

## 2.10 监控 ELK

参考：<https://blog.csdn.net/qq_25934401/article/details/84977549>

https://blog.csdn.net/qq\_25934401/article/details/85049925

## 2.11 process\_exporter进程监控

1)文档中使用版本：process-exporter-0.4.0.linux-amd64.tar.gz下载地址：<https://github.com/ncabatoff/process-exporter/releases/>

2）新建监控进程文件

vim config.yml

3）启动process\_exporter

nohup ./process-exporter -config.path config.yml &

4）配置prometheus配置文件

定义一个标签，参考上个示例

5）图形化



**mongodb**: <https://github.com/dcu/mongodb_exporter/releases>



更多插件参考：<https://prometheus.io/docs/instrumenting/exporters/>

## 2.12 Pushgateway

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。

下载地址：<https://github.com/prometheus/pushgateway/releases/>

# 3 grafana环境部署

## 3.1 部署准备

1）安装包 grafana-5.1.3-1.x86\_64.rpm

下载地址 <https://grafana.com/get>

## 3.2 部署安装步骤

1）将grafana上传到用户目录/{monitoruser}下的指定目录（即~目录的指定目录）

2）安装：yum install urw-fonts -y

rpm -ivh grafana-5.4.3-1.x86\_64.rpm

3）启动grafana，执行命令：service grafana-server start

4）登陆 [http://ip:3000 /](http://ip:3000%20/)

账号密码：admin/admin

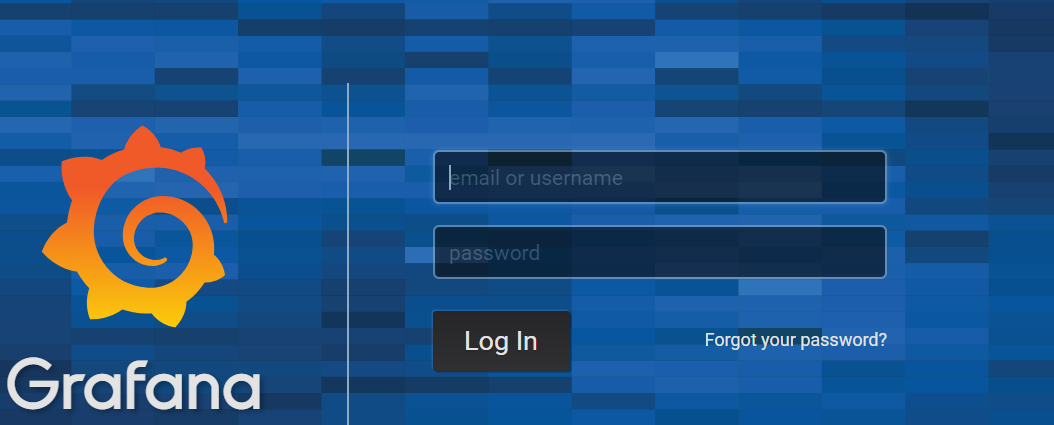
环境文件：/etc/sysconfig/grafana-server

日志文件：/var/log/grafana

数据库：/var/lib/grafana/grafana.db

配置文件：/etc/grafana/grafana.ini

更多介绍参考：https://blog.csdn.net/wangshuminjava/article/details/82290108

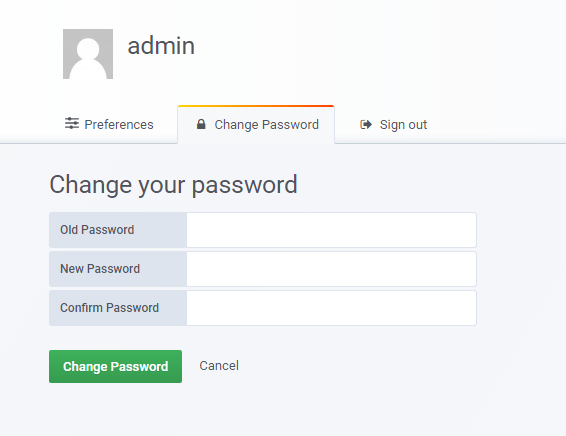
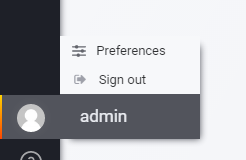


1. 登录

用户名:admin

密码：admin

1. 修改改密码，点击Preferences

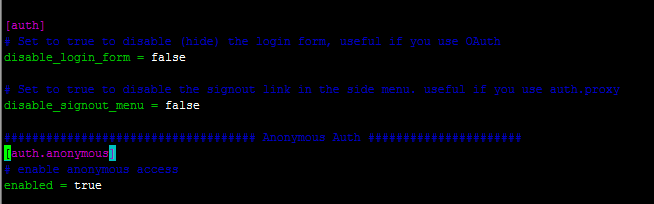


1. 免密看指标图：

打开文件

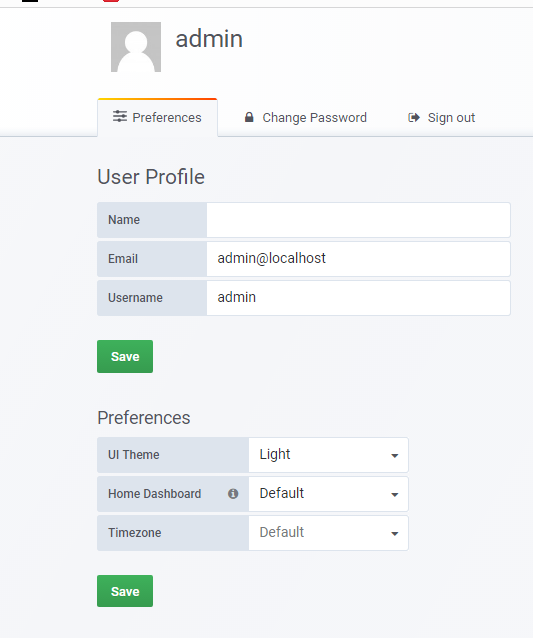


修改enabled为true



1. 修改底色：

UI Theme 有Light和Dark两个选项

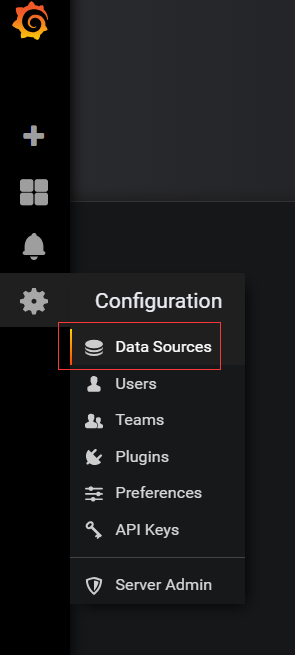
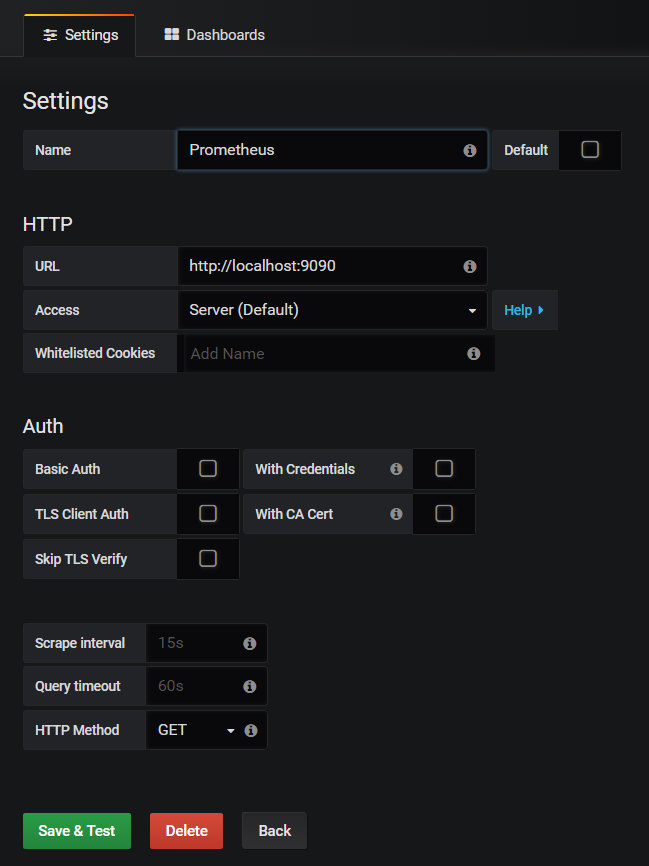


1. 监控图sqlite数据配置和迁移

迁移数据：grafana.db文件



## 3.3为grafana添加Prometheus数据源

在data sources处点击右边绿色标志（add data source）出现图2配置

# 4 报警

## 4.1 部署配置

安装包：[alertmanager-0.16.0-beta.0.linux-amd64.tar.gz](https://github.com/prometheus/alertmanager/releases/download/v0.16.0-beta.0/alertmanager-0.16.0-beta.0.linux-amd64.tar.gz)

下载地址：<https://prometheus.io/download/>

alertmanager是用来接收prometheus发出的告警，然后按照配置文件的要求，将告警用对应的方式发送出去

1）将安装包上传到被监测服务器的用户目录/{monitoruser}下的指定目录（即~目录的指定目录），并解压至该目录下

2）启动命令：nohup ./alertmanager & 默认端口9093

## 4.2 邮箱告警

1）配置alertmanager.yml 定义发送人，事件类型，接收人，配置完成后启动方式：nohup ./alertmanager --config.file alertmanager.yml &



global:

smtp\_smarthost: 'smtp.qq.com:25'

smtp\_from: '243662875@qq.com'

smtp\_auth\_username: '243662875@qq.com'

smtp\_auth\_password: '[pnqsdtlgfsmccaeg]'

route:

group\_by: ['cluster','alertname','service']

receiver: 'email'

group\_wait: 10s #组报警等待时间

group\_interval: 60s #组报警间隔时间

repeat\_interval: 30m #重复报警间隔时间

receivers:

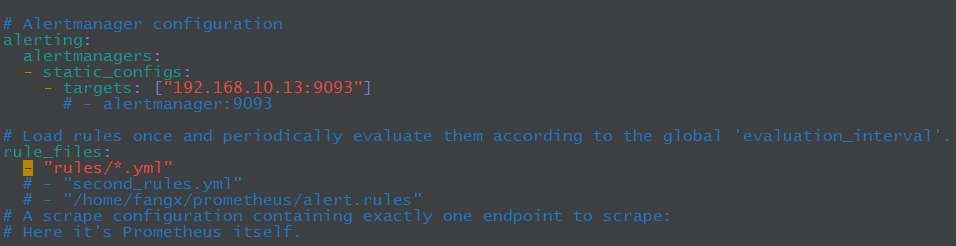
- name: 'email'

email\_configs:

- to: '243662875@qq.com' #收件人

- to: '17602199364@163.com' #收件人

2）配置prometheus.yml



alerting:

alertmanagers:

- static\_configs:

- targets: ["192.168.10.13:9093"] #alertmanager机器部署位置

rule\_files:

- "rules/\*.yml" #告警规则存放位置

3）配置告警规则，参考

groups:

- name: ServiceStatus #规则组名称

rules:

- alert: ServiceStatusAlert #单个规则的名称

expr: up == 0 #匹配规则, up==0

for: 10s #持续时间

labels: #标签

project: APP #自定义lables

annotations: #告警正文

summary: "Instance {{ $labels.instance }} 程序出现问题"

description: "{{ $labels.instance }} of job {{ $labels.job }} 已关闭超过1分钟"

以上配置完成后，就可以收到邮件告警

## 4.3 钉钉告警

参考：<https://theo.im/blog/2017/10/16/release-prometheus-alertmanager-webhook-for-dingtalk/>

和 http://www.cnblogs.com/pyuh/articles/9544121.html

1）配置alertmanager.yml

alertmanager的webhook集成了钉钉报警，所以他不是本来就有的。钉钉对格式要求很严格，还需要使用插件进行格式转换。

global:

resolve\_timeout: 5m

route:

receiver: webhook

group\_wait: 30s

group\_interval: 5m

repeat\_interval: 5m

group\_by: ['cluster','alertname','service']

routes:

- receiver: webhook

group\_wait: '10s'

match:

team: node

receivers:

- name: webhook

webhook\_configs:

- url: 'http://192.168.10.13:8060/dingtalk/ops\_dingding/send'

send\_resolved: true

特别说明：url的连接dingtalk/ops\_dingding/send中的ops\_dingding是钉钉插件启动时指定的名字。

2）配置prometheus.yml 参考邮件告警配置

3）将钉钉接入webhook

下载插件地址：<https://github.com/timonwong/prometheus-webhook-dingtalk>

主要事项：必须要安装go语言，如果没有安装yum install go -y 安装。

go语言默认安装位置 /usr/lib/golang，进入此目录创建目录

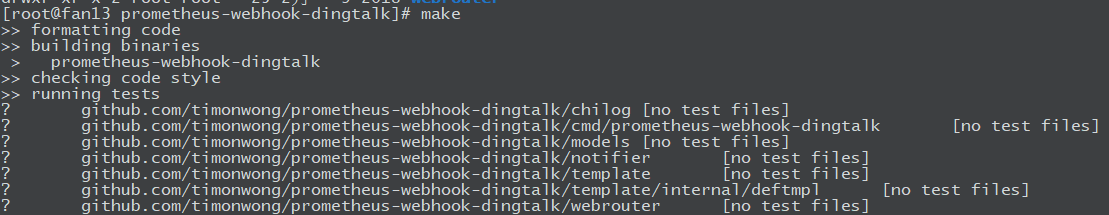
mkdir -p src/github.com/timonwong/

cd /usr/lib/golang/src/github.com/timonwong/ #插件必须放在此目录

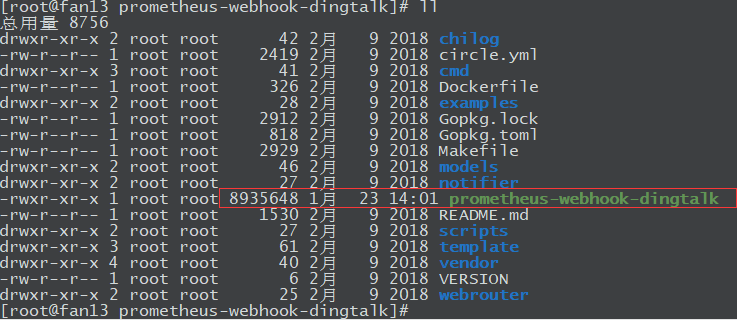
git clone <https://github.com/timonwong/prometheus-webhook-dingtalk.git>

cd prometheus-webhook-dingtalk

make #编译



make后出现下面红框内容表示正常



启动命令：nohup ./prometheus-webhook-dingtalk --ding.profile="ops\_dingding=https://oapi.dingtalk.com/robot/send?access\_token=680394819a326e89ac37d2902b2624f2bb4a2e9da2f974924891e58b097485df" 2>&1 1>dingding.log &

特别说明：ops\_dingding 后面的连接是钉钉机器人webhook连接，以上钉钉告警就配置完成。

## 4.4 告警规则介绍

# 5 prometheus查询语法

## 5.1函数列表

参考：<https://blog.csdn.net/qq_25934401/article/details/84561512>

## 5.2 操作符

参考：<https://blog.csdn.net/qq_25934401/article/details/84561687>