Grafana + Prometheus + Node\_exporter监控资源

目录

[1 Linux下进行安装 2](#_Toc141523660)

[1.1 下载地址 2](#_Toc141523661)

[2 安装/解压 2](#_Toc141523662)

[2.1 安装grafana 2](#_Toc141523663)

[2.2 安装prometheus 3](#_Toc141523664)

[2.3 安装node\_exporter 3](#_Toc141523665)

[3 修改配置 3](#_Toc141523666)

[3.1 grafana 3](#_Toc141523667)

[3.2 prometheus 3](#_Toc141523668)

[4 启动 4](#_Toc141523669)

[4.1 启动grafana 4](#_Toc141523670)

[4.2 prometheus 4](#_Toc141523671)

[4.3 node\_exporter 5](#_Toc141523672)

[5 访问 6](#_Toc141523673)

[6 添加自定义监控 6](#_Toc141523674)

[6.1 pushgateway 6](#_Toc141523675)

[6.1.1 下载地址 6](#_Toc141523676)

[6.1.2 安装 6](#_Toc141523677)

[6.1.3 启动 7](#_Toc141523678)

[6.1.4 编写shell脚本 7](#_Toc141523679)

[6.1.5 定义定时任务 8](#_Toc141523680)

[6.2 Python代码 9](#_Toc141523681)

[6.2.1 Python连接Linux执行命令监控数据 9](#_Toc141523682)

# 1 Linux下进行安装

## 1.1 下载地址

<https://grafana.com/grafana/download>

<https://prometheus.io/download/>

下载grafana：

yum install -y

<https://dl.grafana.com/enterprise/release/grafana-enterprise-9.2.6-1.x86_64.rpm>

下载prometheus：

wget [https://github.com/prometheus/prometheus/releases/download/v2.37.8/prometheus-2.37.8.linux-amd64.tar.gz](https://github.com/prometheus/prometheus/releases/download/v2.45.0/prometheus-2.45.0.linux-amd64.tar.gz)

下载node\_exporter：

wget [https://github.com/prometheus/node\_exporter/releases/download/v1.5.0/node\_exporter-1.5.0.linux-amd64.tar.gz](https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz)

# 2 安装/解压

## 2.1 安装grafana

yum install grafana-enterprise-9.2.6-1.x86\_64.rpm 或

rpm -Uvh (--nodeps) grafana-enterprise-9.2.6-1.x86\_64.rpm

## 2.2 安装prometheus

tar -xzvf prometheus-2.37.8.linux-amd64.tar.gz

## 2.3 安装node\_exporter

tar -xzvf node\_exporter-1.5.0.linux-amd64.tar.gz

# 3 修改配置

## 3.1 grafana

使用vim /etc/grafana/grafana.ini，

将[server]->;http\_addr= 的 ; 去掉

## 3.2 prometheus

使用vim prometheus/Prometheus.yml

在scrape\_configs->static\_configs中添加需要监控的ip:端口，例：192.168.3.5:9100，用逗号分隔

# 4 启动

## 4.1 启动grafana

手动启动：systemctl start grafana-server

设置开机自启动：systemctl enable grafana-server

## 4.2 prometheus

后台启动：在prometheus安装目录下执行

nohup ./prometheus &

添加启动服务

vim /usr/lib/systemd/system/prometheus.service

[Unit]

Description= Prometheus

After=network.target

[Service]

Type=simple

User=prometheus

#此处是Prometheus所在路径以及数据所在路径

ExecStart=/usr/local/Prometheus/prometheus-2.37.8.linux-amd64/prometheus --config.file=/usr/local/Prometheus/prometheus-2.37.8.linux-amd64/prometheus.yml --storage.tsdb.path=/data/prometheus/data

ExecReload=/bin/kill -HUP $MAINPID

Restart=on-failure

[Install]

WantedBy=multi-user.target

设置开机自启动

systemctl daemon-reload

systemctl enable prometheus.service

systemctl start prometheus.service

## 4.3 node\_exporter

后台启动：在node\_exporter安装目录下执行

nohup ./node\_exporter &

vim /etc/systemd/system/node\_exporter.service

[Unit]

Description=Prometheus Node Exporter

After=network.target

[Service]

User=node\_exporter

Group=node\_exporter

Type=simple

ExecStart=/usr/bin/node\_exporter

[Install]

WantedBy=multi-user.target

设置开机自启动

systemctl daemon-reload

systemctl enable node\_exporter

systemctl start node\_exporter

# 5 访问

访问grafana

输入Linux ip:3000，例：192.168.3.5:3000

访问prometheus

输入Linux ip:9090，例：192.168.3.5:9090

访问node\_exporter

输入Linux ip:9100，例：192.168.3.5:9100

# 6 添加自定义监控

## 6.1 pushgateway

### 6.1.1 下载地址

<https://prometheus.io/download/>

wget <https://github.com/prometheus/pushgateway/releases/download/v1.4.3/pushgateway-1.4.3.linux-amd64.tar.gz>

### 6.1.2 安装

tar -xzvf

<https://github.com/prometheus/pushgateway/releases/download/v1.4.3/pushgateway-1.4.3.linux-amd64.tar.gz>

### 6.1.3 启动

nohup ./pushgateway &

### 6.1.4 编写shell脚本

#!/usr/bin/bashinstance\_name=`hostname -f|cut -d'.' -f1` #截取主机名if [ ${instance\_name} == "localhost" ];then echo "Must FQDN hostname" #要求主机名不能是localhost，不要主机名区别不了 exit 1fi#定义keylabel\_wait="count\_netstat\_wait\_connections"#定义valuecount\_netstat\_wait\_connections=`netstat -an|grep -i wait|wc -l`echo "${label\_wait}:${count\_netstat\_wait\_connections}"#推送数据给pushgatewayecho "${label\_wait} ${count\_netstat\_wait\_connections}"|curl --data-binary @- http://192.168.3.35:9091/metrics/job/pushgateway/instance/${instance\_name}#定义keylabel\_wait="count\_coredump"#定义valuecount\_coredump=`ls -lrt /var/lib/systemd/coredump|grep "^-"|wc -l`echo "${label\_wait}:${count\_coredump}"#推送数据给pushgatewayecho "${label\_wait} ${count\_coredump}"|curl --data-binary @- [http://192.168.3.35:9091/metrics/job/pushgateway/instance/${instance\_name}](http://192.168.3.35:9091/metrics/job/pushgateway/instance/$%7binstance_name%7d)

### 6.1.5 定义定时任务

Crontab -e

每五秒执行一次脚本，输入

\*/1 \* \* \* \* sleep 5 && sh /root/shell\_scripts/pushgateway\_shell.sh

## 6.2 Python代码

### 6.2.1 Python连接Linux执行命令监控数据

# -\*- encoding: utf-8 -\*-# Author: Komorebi# Date: 2023/7/23 12:37# Describe: Prometheus monitor server portimport randomimport prometheus\_clientfrom prometheus\_client import Gaugefrom prometheus\_client.core import CollectorRegistryfrom flask import Response, Flaskfrom utils.connLinux import ConnLinux, connLinuxapp = Flask(\_\_name\_\_, static\_url\_path="/main")# 实例化 REGISTRYregistry = CollectorRegistry(auto\_describe=False)gauge = Gauge( name="Server\_port", documentation="monitor server port status.", labelnames=["sertype", "host", "port"], registry=registry)@app.route("/metrics")def requests\_count(): result = ConnLinux().exec\_command("ls -l /root/shell\_scripts|grep '^-'|wc -l") # 模拟多个值传入 rows = [ {"sertype": "zookeeper", "host": "192.168.1.22", "port": "2181", "status": result}, {"sertype": "zookeeper", "host": "192.168.1.33", "port": "2181", "status": random.randint(10, 30)}, {"sertype": "zookeeper", "host": "192.168.1.44", "port": "2181", "status": random.randint(15, 35)}, {"sertype": "mysql", "host": "192.168.1.88", "port": "3306", "status": random.randint(5, 25)}, {"sertype": "mysql", "host": "192.168.1.99", "port": "3306", "status": random.randint(20, 40)} ] for row in rows: sertype = "".join(row.get("sertype")) ip = "".join(row.get("host")) port = "".join(row.get("port")) status = row.get("status") gauge.labels(sertype, ip, port).set(status) return Response(prometheus\_client.generate\_latest(registry), mimetype="text/plain")if \_\_name\_\_ == "\_\_main\_\_": app.run(host="0.0.0.0", port=31672, debug=True)