

监控篇

监控概念和基础知识

监控的概念

监控是指通过收集、处理和分析数据，对系统、应用程序、网络、服务器等进行实时或定期的监测和管理的过程。监控可以通过收集各种指标数据，如CPU占用率、内存使用率、网络流量、应用程序响应时间等，以便及时发现和解决问题，提高系统的可靠性和性能，同时也可以提高工作效率和用户满意度。

监控的分类

监控可以按照监控对象、监控方式和监控指标进行分类。

按照监控对象

- 系统监控：监控操作系统的资源使用情况，如CPU、内存、磁盘等。
- 应用程序监控：监控应用程序的性能、错误、日志等信息。
- 网络监控：监控网络设备、链路和流量等信息。
- 主机监控：监控主机的硬件和软件资源使用情况，如CPU、内存、磁盘等。
- 数据库监控：监控数据库的性能、连接数、死锁等信息。

按照监控方式

- 主动监控：定期向被监控对象发送请求，获取监控数据。
- 被动监控：被监控对象主动将监控数据发送给监控系统。

按照监控指标

- 性能监控：监控系统或应用程序的性能指标，如响应时间、吞吐量、并发数等。
- 错误监控：监控系统或应用程序的错误和异常，如程序崩溃、连接超时等。
- 日志监控：监控系统或应用程序的日志信息，如访问日志、错误日志等。

监控的重要性

监控对于维护系统稳定性、提高效率、优化用户体验等方面都具有重要作用。通过监控，我们可以及时发现系统性能问题、异常情况和潜在的故障，以便及时采取措施解决问题。同时，监控也可以帮助我们掌握系统资源使用情况，优化系统配置和调整资源分配，提高系统的可靠性和性能。

监控的实现方式

监控的实现方式可以分为基于代理的监控、无代理的监控和云监控。

基于代理的监控

基于代理的监控是通过在被监控对象所在的主机上安装代理程序，代理程序负责采集监控数据并将数据传递给监控服务器。基于代理的监控可以提高数据的采集效率和可靠性，但是需要在被监控主机上安装代理程序，可能会带来一定的安全风险。

无代理的监控

无代理的监控是在被监控对象上直接安装监控软件，以便直接采集监控数据并将数据传递给监控服务器。无代理的监控可以避免安装代理程序带来的安全风险，但是需要在被监控对象上安装监控软件，可能会影响被监控对象的性能。

云监控

云监控是将监控系统部署在云平台上，通过云平台提供的监控服务对系统、应用程序、网络、服务器等进行监控。云监控可以避免在被监控对象上安装监控软件、代理程序等，同时也可以提供更高效、更安全、更可靠的监控服务，但是需要考虑云平台监控服务的稳定性和安全性等问题。

监控工具

Grafana

Grafana 用途

Grafana 安装

本地部署

- grafana版本采用 6.7.x

```
1 yum localinstall
  https://mirrors.tuna.tsinghua.edu.cn/grafana/yum/rpm/Packages/grafana-6.7.2-
  1.x86_64.rpm -y
```

- 启动grafana并设为开机自启

```
1 systemctl start grafana-server.service && systemctl enable grafana-
  server.service
```

- 查看端口是否启用

```
1 netstat -tunlp | grep 3000
```

访问 dashBoard 默认密码admin admin

```
1 http://localhost:3000
```

汉化界面

前往 <https://github.com/grafana/grafana/tree/v6.7.x> 下载源码

提前安装好 `node` 和 `yarn` 进入源码目录编译

```
1 yarn install
2 yarn build
```

首次同步前，先备份 `/usr/share/grafana/public` 目录

```
1 mv /usr/share/grafana/public /usr/share/grafana/public.source
2 #将当前编译目录下的public目录拷贝到/usr/share/grafana/
3 #重启
4 service grafana-server restart
```

docker 部署

```
1 # create a folder for your data
2 mkdir data
3
4 # start grafana with your user id and using the data folder
5 docker run -d -p 3000:3000 --name=grafana \
6   --user "$(id -u)" \
7   --volume "$PWD/data:/var/lib/grafana" \
8   grafana/grafana-enterprise
```

k8s 部署

- 配置挂着目录

```
1 # k8s-master-1 Master节点配置持久化存储,如果有其他方式提供存储在Grafana-Deployment中
  配置即可, 或者使用Statefulset配置StorageClass来实现持久化存储
2 mkdir -p /grafana
3 chown 472:472 /grafana -R
```

- 编辑 grafana.yaml

```
1 ---
2 apiVersion: apps/v1
3 kind: Deployment
4 metadata:
5   labels:
6     app: grafana
7   name: grafana
8 spec:
9   selector:
10    matchLabels:
11      app: grafana
12   template:
13     metadata:
14       labels:
15         app: grafana
16     spec:
17       securityContext:
18         fsGroup: 472
19       supplementalGroups:
20         - 0
21     containers:
22       - name: grafana
23         image: grafana/grafana:8.5.1
24         imagePullPolicy: IfNotPresent
25         ports:
26           - containerPort: 3000
27           protocol: TCP
```

```

28     volumeMounts:
29       - name: grafana-data
30         mountPath: /var/lib/grafana
31     volumes:
32       - name: grafana-data
33         hostPath:
34           path: /grafana
35           type: Directory
36 ---
37 apiVersion: v1
38 kind: Service
39 metadata:
40   name: grafana
41 spec:
42   selector:
43     app: grafana
44   type: NodePort                                # 将其暴露出来，便于外部访问
45   ports:
46     - port: 3000
47       targetPort: 3000
48       nodePort: 3000                            # 固定端口，便于访问

```

- 运行以下命令：`kubectl apply -f grafana.yaml`
- 通过运行以下命令检查它是否正常工作：`kubectl port-forward service/grafana 3000:3000`
- 在浏览器中导航到。应该会看到一个 Grafana 登录页面。`localhost:3000`
- 用于登录的用户名和密码。`admin`

Grafana 使用

对接 zabbix

[安装zabbix插件](#)

```

1  # grafana下载boom theme插件
2  grafana-cli plugins install yesoreyeram-boomtheme-panel
3  # 世界ping
4  grafana-cli plugins install raintank-worldping-app
5  # Zabbix报警
6  grafana-cli plugins install alexanderzobnin-zabbix-app
7  # 世界地图面板
8  grafana-cli plugins install grafana-worldmap-panel
9
10 # 时钟
11 grafana-cli plugins install grafana-clock-panel
12 # 饼图
13 grafana-cli plugins install grafana-piechart-panel
14 grafana-cli plugins install macropower-analytics-panel
15 grafana-cli plugins install digiapulssi-breadcrumb-panel
16 grafana-cli plugins install andig-darksky-datasource
17 grafana-cli plugins install citilogics-geoloop-panel
18 grafana-cli plugins install bessler-pictureit-panel
19 grafana-cli plugins install nate1-plotly-panel
20 grafana-cli plugins install snuids-radar-panel
21 grafana-cli plugins install blackmirror1-statusbygroup-panel
22 grafana-cli plugins install snuids-trafficlights-panel

```

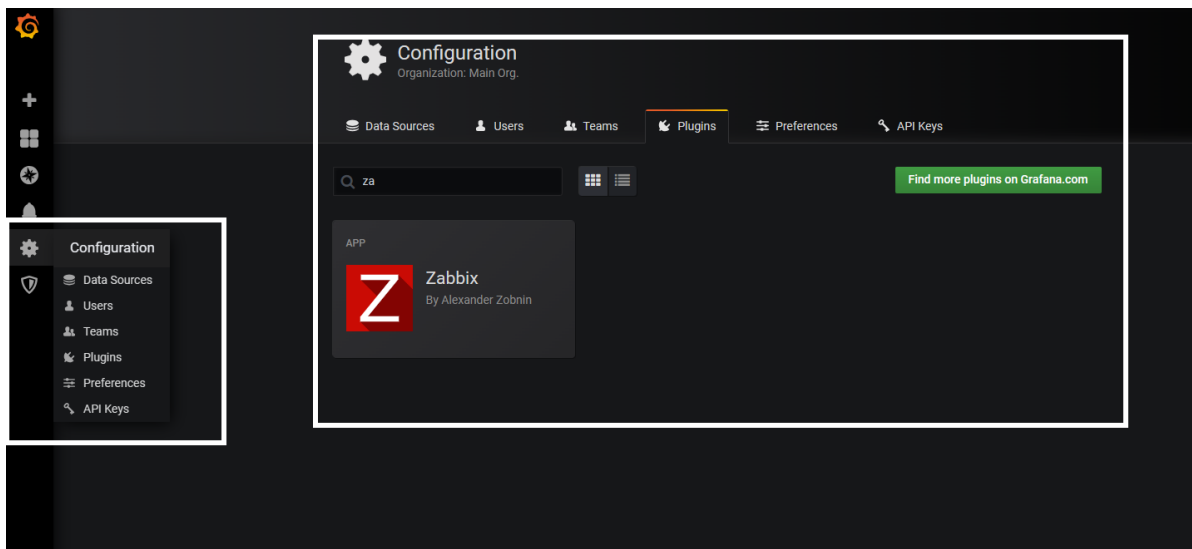
```
23 grafana-cli plugins install smartmakers-trafficlight-panel
24 grafana-cli plugins install btplc-trend-box-panel
25 grafana-cli plugins install fatcloud-windrose-panel
26 # 气泡图
27 grafana-cli plugins install digrich-bubblechart-panel
28 # json数据
29 grafana-cli plugins install grafana-simple-json-datasource
30 # k8s监控应用
31 grafana-cli plugins install grafana-kubernetes-app
32 # windRoseby 极坐标图
33 grafana-cli plugins install fatcloud-windrose-panel
34 # 雷达图
35 grafana-cli plugins install snuids-radar-panel
36 # 世界地图热力图
37 grafana-cli plugins install ovh-warp10-datasource
38 # 选点监控
39 grafana-cli plugins install natel-usgs-datasource
40 # es数据监控
41 grafana-cli plugins install stagemonitor-elasticsearch-app
42 # Plotly直接坐标系散点图
43 grafana-cli plugins install natel-plotly-panel
44 # 组织结构图
45 grafana-cli plugins install digiapulssi-organisations-panel
46 # ajax请求更新数据
47 grafana-cli plugins install ryantxu-ajax-panel
48 # Ps: 更新插件示例:
49 grafana-cli plugins update alexanderzobnin-zabbix-app 3.12.1
```

重启服务

```
1 | systemctl restart grafana-server.service
```

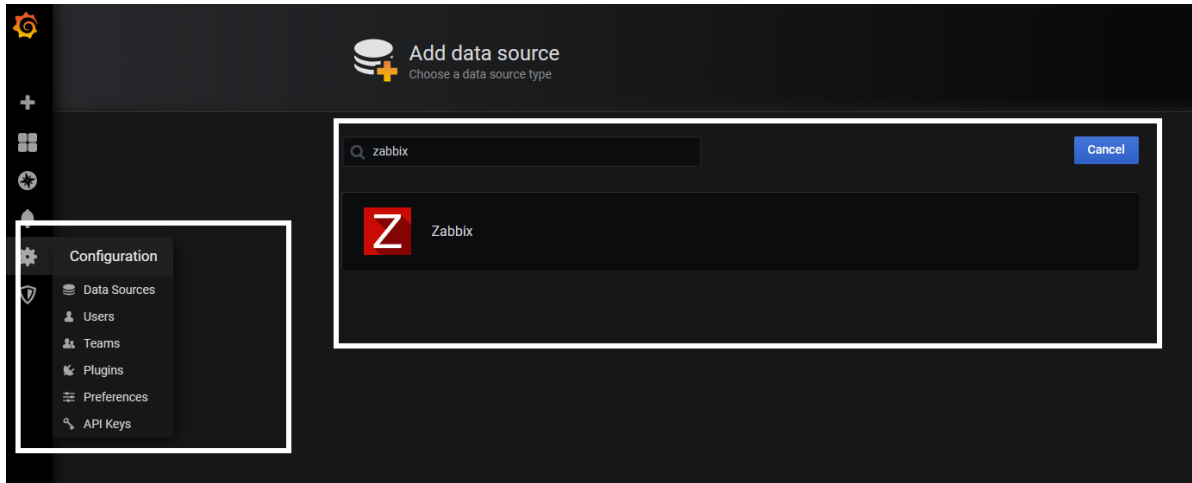
启用插件

【configuration】 - 【plugins】 - 【zabbix】

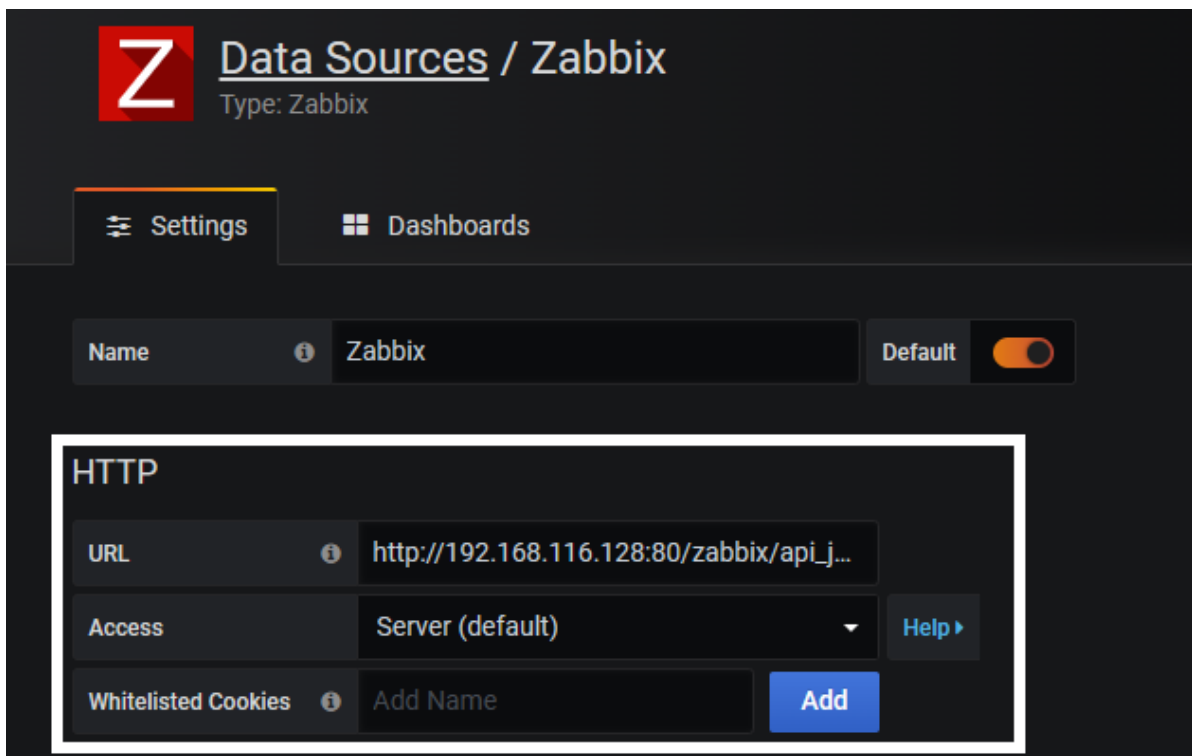


添加数据源

【configuration】 - 【datasources】 - 【zabbix】



点击进去后，编辑【HTTP/URL】



```
1 [root@localhost zabbix]# curl -s -X POST -H 'Content-Type:application/json' -  
d '{"jsonrpc": "2.0","method": "user.login","params": {"user":  
"Admin","password": "zabbix"},"id": 1}'  
http://192.168.116.128:80/zabbix/api_jsonrpc.php  
2 {"jsonrpc": "2.0", "result": "232f7f5a250d95b3647180dec2d64748", "id": 1}
```

编辑【zabbix api details】

Zabbix API details

Username	Admin		
Password		
Trends	<input checked="" type="checkbox"/>		
After	<i>i</i> 7d	Range	<i>i</i> 4d
Cache TTL	<i>i</i> 1h		

Direct DB Connection

Enable ☐

编辑【other】，并保存

Alerting

Enable alerting ☐


Other

Disable acknowledges for read-only users ☒

✓ Zabbix API version: 5.0.36

[Save & Test](#) [Delete](#) [Back](#)

导入仪表板并查看监控效果





Data Sources / Zabbix


Type: Zabbix


[Settings](#)


[Dashboards](#)


 Zabbix System Status

 Zabbix Template Linux Server

 Zabbix Server Dashboard

Re-import 

Re-import 

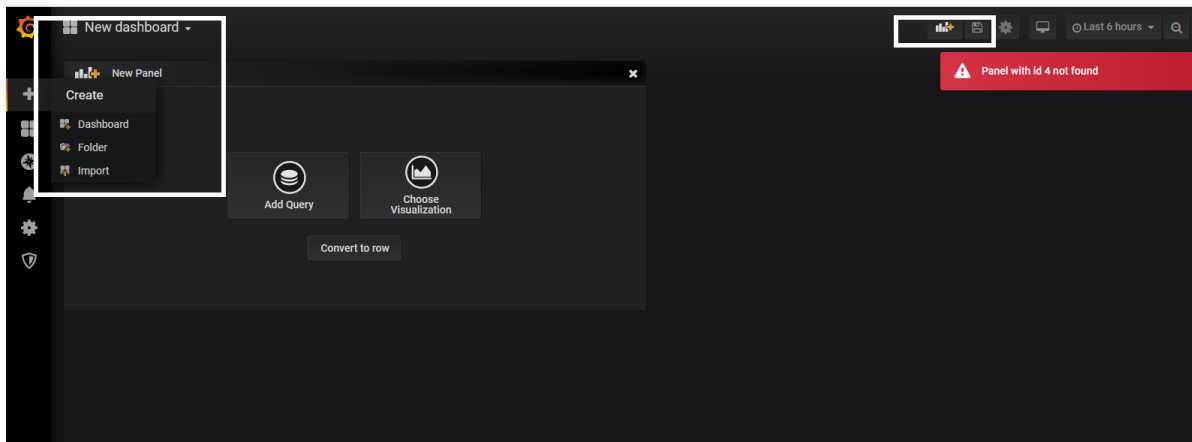
Re-import 



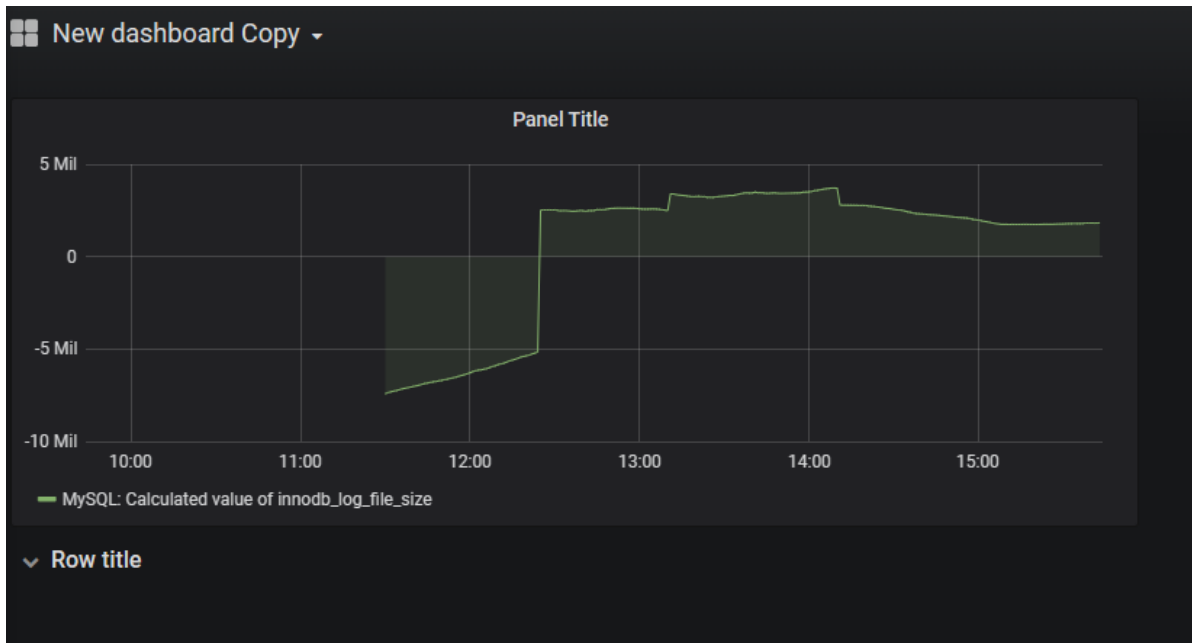
连接zabbix数据库

【选择mysql】-在之前的zabbix选择db【启用并选择上面创建好的 Zabbix DB Data Source】。

新建仪表盘



之后，点击保存，即可看到其他的数据展示。



prometheus

prometheus用途

prometheus安装

本地部署 服务器

- 前提配置

```
1 yum install -y wget ntp curl vim net-tools
2 tar -zcvf /etc/yum.repos.d/yumRepo.bak /etc/yum.repos.d/*.repo
3 rm -rf /etc/yum.repos.d/*.repo
4 wget -O /etc/yum.repos.d/CentOS-Base.repo
  http://mirrors.aliyun.com/repo/Centos-7.repo
5 yum install epel-release -y
6 yum clean all && yum makecache
7 timedatectl set-timezone Asia/Shanghai
8 timedatectl set-local-rtc 0
9 sed -i 's/server 0.centos.pool.ntp.org iburst/server ntp1.aliyun.com
  iburst/g' /etc/ntp.conf
10 sed -i 's/server 1.centos.pool.ntp.org iburst/server ntp2.aliyun.com
  iburst/g' /etc/ntp.conf
11 sed -i 's/server 2.centos.pool.ntp.org iburst/server 1.centos.pool.ntp.org
  iburst/g' /etc/ntp.conf
12 sed -i 's/server 3.centos.pool.ntp.org iburst/server 2.centos.pool.ntp.org
  iburst/g' /etc/ntp.conf
13 systemctl restart ntpd
14 systemctl enable ntpd
15 systemctl stop firewalld
16 systemctl disable firewalld
17 setenforce 0
18 sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config
```

- 安装 prometheus

下载

```
1 wget
  https://github.com/prometheus/prometheus/releases/download/v2.46.0/prometheus-2.46.0.freebsd-amd64.tar.gz
```

解压

```
1 tar -zxvf prometheus-2.17.1.linux-amd64.tar.gz -C /usr/local
```

将Prometheus做成软连接的形式

```
1 ln -s /usr/local/prometheus-2.17.1.linux-amd64 /usr/local/prometheus
```

创建用于运行Prometheus的组和用户

```
1 groupadd prometheus
2 useradd -g prometheus -s /sbin/nologin prometheus
```

创建Prometheus数据存储目录

```
1 mkdir -p /var/lib/prometheus
2 chown -R prometheus /var/lib/prometheus
```

给Prometheus主目录赋用户Prometheus权限

```
1 chown -R prometheus:prometheus /usr/local/prometheus/
```

将Prometheus加入到系统管理程序中

```
1 cat >/etc/systemd/system/prometheus.service <<EOF
2
3 [Unit]
4 Description=Prometheus
5 Documentation=https://prometheus.io/
6 After=network.target
7
8 [Service]
9 Type=simple
10 User=prometheus
11 # --storage.tsdb.path是可选项，默认数据目录在运行目录的./data目录中
12 ExecStart=/usr/local/prometheus/prometheus --
13   config.file=/usr/local/prometheus/prometheus.yml --
14   storage.tsdb.path=/var/lib/prometheus
15 Restart=on-failure
16
17 [Install]
18 WantedBy=multi-user.target
19 EOF
```

- 可选) 将客户端加入到Prometheus监控中

注意: 将配置文件中的ip地址改成你的被监控客户端的ip, (node_exporter的)端口号默认是9100

```

1 cp /usr/local/prometheus/prometheus.yml{,.bak}
2 vi /usr/local/prometheus/prometheus.yml
3
4 - job_name: 'xnode1-mycat33'
5   scrape_interval: 10s
6   static_configs:
7     - targets: ['192.168.116.128:9100']
8     labels:
9       instance: xnode1-instance

```

为了方便测试，进去后，将localhost改成本机ip

- 启动Prometheus并设置其开机自启

```

1 systemctl start prometheus.service
2 systemctl enable prometheus.service

```

- 验证prometheus的Web页面，prometheus默认的端口号是9090，浏览器输入

```
1 192.168.116.128: 9090
```

- 在prometheus的web页面上查看主机监控状态【Status---Targets】

Prometheus
Alerts
Graph
Status ▾
Help

Targets

All
Unhealthy

prometheus (0/1 up) [show less](#)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://192.168.116.128:9090/metrics	UNKNOWN	instance="192.168.116.128:9090" job="prometheus"	Never	0s	

- 使用prometheus的web方式查看主机的监控值

```

# HELP go_gc_duration_seconds A summary of the pause duration of garbage collection cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 1.1128e-05
go_gc_duration_seconds{quantile="0.25"} 7.6757e-05
go_gc_duration_seconds{quantile="0.5"} 0.000816215
go_gc_duration_seconds{quantile="0.75"} 0.000906684
go_gc_duration_seconds{quantile="1"} 0.001689115
go_gc_duration_seconds_sum 0.003611712
go_gc_duration_seconds_count 6
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 40
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.13.9"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 1.7710824e+07
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated, even if freed.
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 2.8682544e+07
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the profiling bucket hash table.
# TYPE go_memstats_buck_hash_sys_bytes gauge
go_memstats_buck_hash_sys_bytes 1.454001e+06
# HELP go_memstats_frees_total Total number of frees.
# TYPE go_memstats_frees_total counter

```

客户端安装部署node_exporter

- 下载node_exporter

```
1 | wget -P /usr/local/src  
https://github.com/prometheus/node_exporter/releases/download/v1.0.0-rc.0/node_exporter-1.0.0-rc.0.linux-amd64.tar.gz
```

- 解压node_exporter

```
1 | tar -zxvf /usr/local/src/node_exporter-1.0.0-rc.0.linux-amd64.tar.gz -C /usr/local
```

- 可选) 创建软连接 (如果你对linux不熟悉, 建议跟着继续做)

```
1 | ln -s /usr/local/node_exporter-1.0.0-rc.0.linux-amd64/  
/usr/local/node_exporter
```

- 创建用于运行node_exporter的用户

```
1 | groupadd prometheus  
2 | useradd -g prometheus -s /sbin/nologin prometheus
```

- 给node_exporter主目录赋权限

```
1 | chown -R prometheus:prometheus /usr/local/node_exporter/
```

- 将node_exporter加入到系统服务当中

```
1 | cat >/usr/lib/systemd/system/node_exporter.service <<EOF  
2 | [Unit]  
3 | Description=node_exporter  
4 | Documentation=https://prometheus.io/  
5 | After=network.target  
6 |  
7 | [Service]  
8 | Type=simple  
9 | User=prometheus  
10 | ExecStart=/usr/local/node_exporter/node_exporter  
11 | Restart=on-failure  
12 |  
13 | [Install]  
14 | WantedBy=multi-user.target  
15 | EOF
```

- 启动node_exporter并将其设置开机自启

```
1 | systemctl start node_exporter  
2 | systemctl enable node_exporter
```

- 检查node_exporter是否已启动, node_exporter默认的端口是9100

```
1 systemctl status node_exporter
2 ss -ntl |grep 9100
```

- (啰嗦) 将9100在防火墙中放行, 或者直接关闭防火墙

```
1 systemctl stop firewalld
2 systemctl disable firewalld
```

- 在prometheus的web上检查是否监控到了本机

在服务器中已经修改过配置文件, 将本客户端加入到了prometheus服务器的配置文件中了。

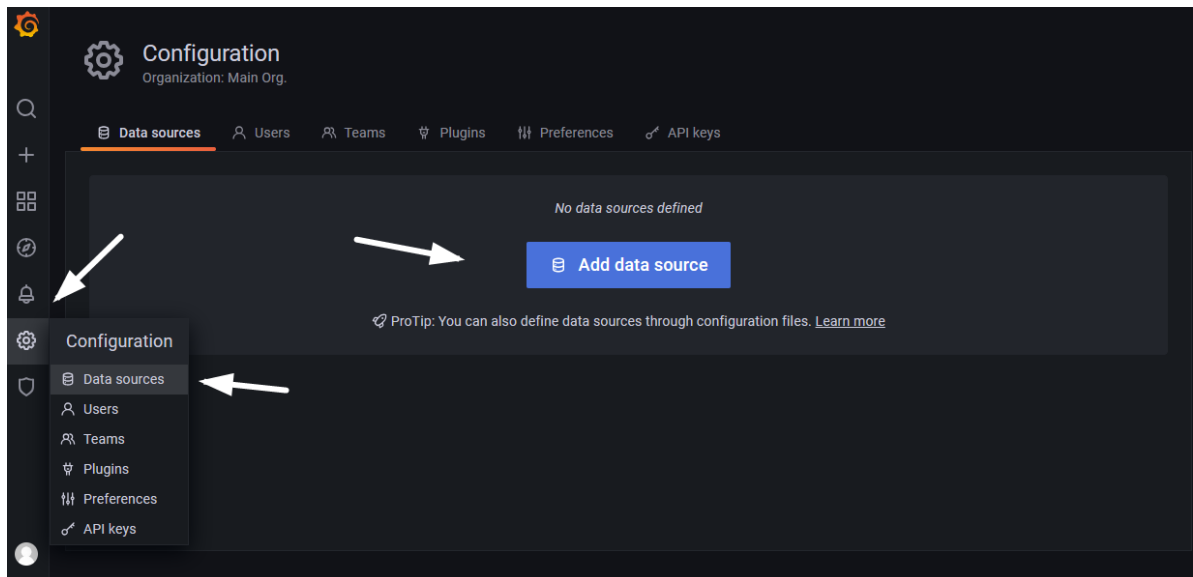
docker 部署

k8s 部署

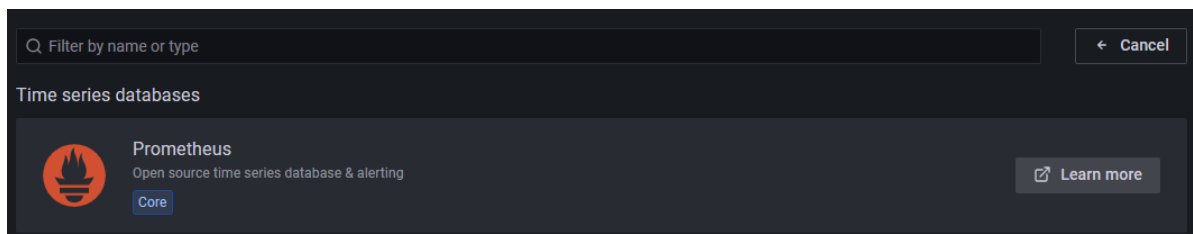
在Grafana上展示prometheus

- 浏览器打开Grafana主页 <http://192.168.112.128:3000>, 默认用户名和密码都是admin。
- 添加数据源

依次点开左侧的齿轮状图标【Configuration】---【Data Source】, 再新页面中点【Add data source】



- 点一下数据源类型Prometheus



- 为Grafana数据源prometheus添加数据源参数

Name: 随便输入

Default: 设置为开启状态

URL: <http://192.168.116.128:9090/>, 写你的prometheus主页地址。

Access: Server(default)

Scrape interval: 15s , 因为我们这是测试环境, 尽量把刷新数据的时间写小点。

Name: Prometheus-1

Default: ☒

HTTP

URL: http://192.168.116.128:9090/

Access: Server (default)

Allowed cookies: New tag (enter key to add)

Timeout: Timeout in seconds

Auth

Basic auth: ☐ With Credentials: ☐

TLS Client Auth: ☐ With CA Cert: ☐

Skip TLS Verify: ☐

Forward OAuth Identity: ☐

Custom HTTP Headers

+ Add header

Alerting

Manage alerts via Alerting UI: ☒

Alertmanager data source: Choose

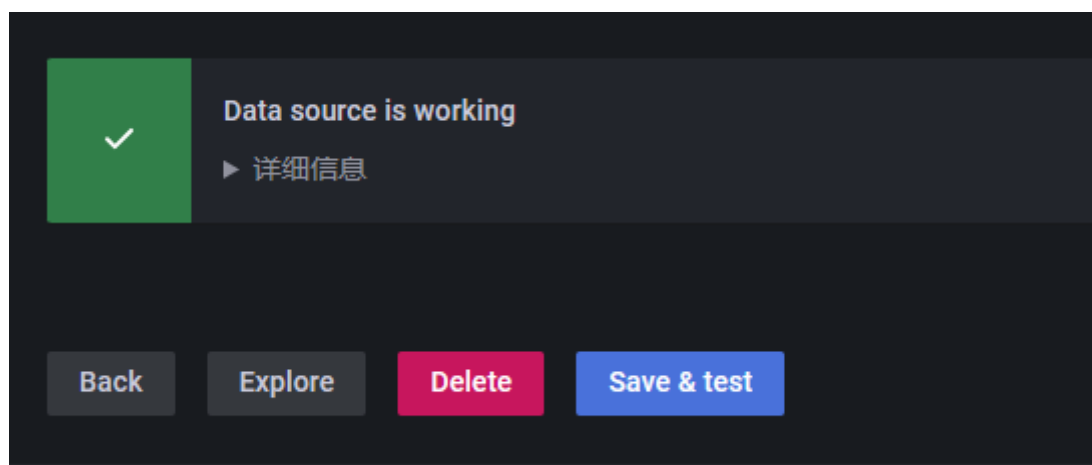
Scrape interval: 15s

Query timeout: 60s

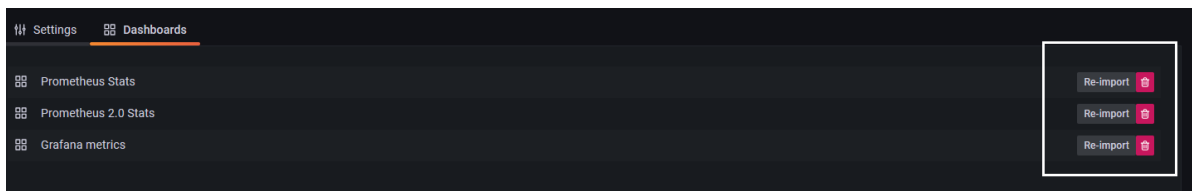
HTTP Method: POST

Save & Test

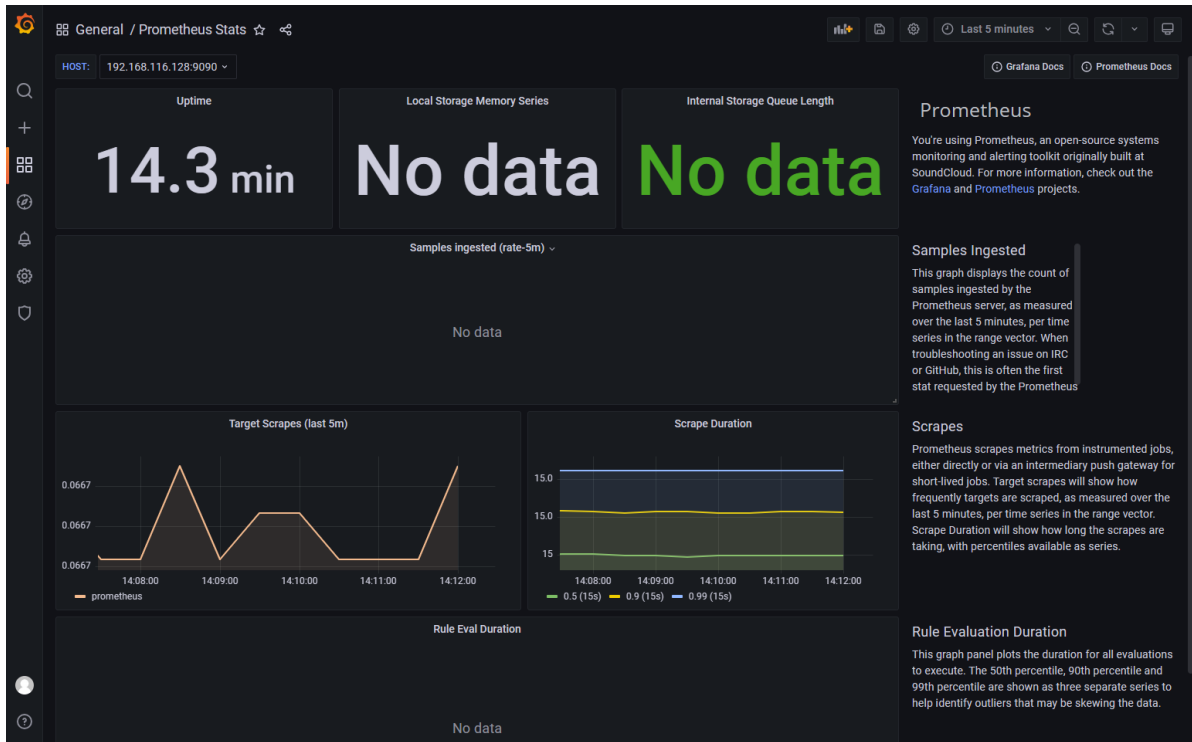
- 点【Save & Test】后, 能弹出绿色的【Data source is working】就说明我们的prometheus数据源添加成功了。



- 在Grafana上查看默认的prometheus仪表盘。



然后点一下【Prometheus Stats】就能看到默认的仪表盘了。



Zabbix

Zabbix 用途

Zabbix 安装

zabbix 服务端

创建 zabbix 用户组和用户

```
1 groupadd zabbix
2 useradd -r -g zabbix zabbix
```

安装配置zabbix专用yum源

```
1 rpm -Uvh https://repo.zabbix.com/zabbix/5.0/rhel/7/x86_64/zabbix-release-5.0-1.el7.noarch.rpm
2 yum clean all
3 yum makecache fast
```

安装zabbix服务端组件

```
1 yum -y install zabbix-server-mysql zabbix-web-mysql zabbix-get
```

如果安装过程出现版本冲突，然后重新运行上述

```
1 | cd /etc/yum.repos.d/
2 | mv epel.repo epel-bak
```

安装zabbix前端组件

```
1 | yum -y install centos-release-scl
```

编辑配置文件 `/etc/yum.repos.d/zabbix.repo`，将enabled设置为1，如下：

```
1 | vi /etc/yum.repos.d/zabbix.repo
2 |
3 | [zabbix]
4 | name=Zabbix Official Repository - $basearch
5 | baseurl=http://repo.zabbix.com/zabbix/5.0/rhel/7/$basearch/
6 | enabled=1
7 | gpgcheck=1
8 | gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-ZABBIX-A14FE591
9 |
10 | [zabbix-frontend]
11 | name=Zabbix Official Repository frontend - $basearch
12 | baseurl=http://repo.zabbix.com/zabbix/5.0/rhel/7/$basearch/frontend
13 | enabled=1
14 | gpgcheck=1
15 | gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-ZABBIX-A14FE591
16 |
17 | [zabbix-debuginfo]
18 | name=Zabbix Official Repository debuginfo - $basearch
19 | baseurl=http://repo.zabbix.com/zabbix/5.0/rhel/7/$basearch/debuginfo/
20 | enabled=0
21 | gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-ZABBIX-A14FE591
22 | gpgcheck=1
23 |
24 | [zabbix-non-supported]
25 | name=Zabbix Official Repository non-supported - $basearch
26 | baseurl=http://repo.zabbix.com/non-supported/rhel/7/$basearch/
27 | enabled=1
28 | gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-ZABBIX
29 | gpgcheck=1
```

安装zabbix前端页面、初始数据库、PHP及httpd组件

```
1 | yum -y install zabbix-web-mysql-scl zabbix-apache-conf-scl
```

安装 MySQL

- 下载 rpm

```
1 | wget https://dev.mysql.com/get/mysql80-community-release-el7-3.noarch.rpm
2 | yum -y localinstall mysql80-community-release-el7-3.noarch.rpm
3 | yum -y install mysql-devel #安装mysql的开发包
4 | yum install -y yum-utils
5 | # 查看可用的 mysql
6 | yum repolist enabled | grep "mysql.*-community.*"
```


可以看到目前启用(enable)的是 mysql80,如果需要安装 mysql 5.7,则需要先禁用(disable)mysql8.0,然后启用(enable)mysql5.7

```
1 yum-config-manager --disable mysql80-community
2 yum-config-manager --enable mysql57-community
```

- 校验当前启用(enable)的 mysql 版本是否为 5.7

```
1 yum repolist enabled | grep mysql
```

- 修改配置跳过校验

```
1 vi /etc/yum.repos.d/mysql-community.repo
```

修改对应安装版本的 `gpgcheck=0` 即可, 默认值为1。

- 开始安装

```
1 yum install -y mysql-community-server
```

- 启动并创建自启动

```
1 sudo service mysqld start
2 # systemctl start mysqld.service
3 systemctl enable mysqld.service
```

- 查看生成的初始密码

```
1 sudo grep 'temporary password' /var/log/mysqld.log
```

- 使用初始密码进行登录

```
1 mysql -uroot -p
```

- 登录成功之后设置新密码

```
1 # 5.7
2 set global validate_password_policy=LOW;
3 set global validate_password_length=6;
4 ALTER USER 'root'@'localhost' IDENTIFIED BY '123456';
5
6 # 8.0
7 alter user 'root'@'localhost' identified with mysql_native_password by '新密码';
8
9 # 刷新MySQL的系统权限命令
10 FLUSH PRIVILEGES;
```

- 授权其他机器远程登录

```
1 GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY '123456' WITH GRANT OPTION;
```

执行部署 zabbix 服务

```
1 cd /usr/local/data/zabbix
2
3 ./configure --prefix=/usr/local/data/zabbix --enable-server --enable-agent --
  with-mysql=/usr/bin/mysql_config --with-net-snmp --with-libcurl --with-
  libxml2
```

Enable agent 2: no

Enable Java gateway: no

LDAP support: no

IPv6 support: no

```
*****
*                Now run 'make install'                *
*                                                        *
*                Thank you for using Zabbix!             *
*                <http://www.zabbix.com>                 *
*****
```

[root@localhost zabbix-5.0.30]#

```
1 make && make install
```

然后添加环境变量

```
1 vim /etc/profile
```

```
1 # Zabbix Environment
2 export PATH=$PATH:/usr/local/data/zabbix/sbin:/usr/local/data/zabbix/bin/
```

```
1 source /etc/profile
```

创建用户导入数据库

```
1 create database zabbix character set utf8 collate utf8_bin;
2
3 set global validate_password_length=6;
4
5 set global validate_password_policy=0;
6
7 grant all privileges on zabbix.* to zabbix@'localhost' identified by
  '123456';
8
9 flush privileges;
10
```

```
11 set names utf8;
12
13 use zabbix;
14 source /usr/local/zabbix-5.0.30/database/mysql/schema.sql
15 source /usr/local/zabbix-5.0.30/database/mysql/data.sql
16 source /usr/local/zabbix-5.0.30/database/mysql/images.sql
```

编辑 zabbix_server 的配置文件

- 查找下配置文件位置

```
1 [root@localhost ~]# find / -name zabbix_server.conf
2 /etc/zabbix/zabbix_server.conf
```

- 编辑配置文件

```
1 vi /etc/zabbix/zabbix_server.conf
```

```
1 LogFile=/var/log/zabbix/zabbix_server.log
2 LogFileSize=0
3 PidFile=/var/run/zabbix/zabbix_server.pid
4 SocketDir=/var/run/zabbix
5 DBHost=localhost
6 DBName=zabbix
7 DBUser=zabbix
8 DBPassword=123456
9 DBPort=3306
10 SNMPTrapperFile=/var/log/snmptrap/snmptrap.log
11 Timeout=4
12 AlertScriptsPath=/usr/lib/zabbix/alertscripts
13 ExternalScripts=/usr/lib/zabbix/externalscripts
14 LogSlowQueries=3000
15 StatsAllowedIP=127.0.0.1
```

- 授权建立相关文件夹

```
1 mkdir /usr/local/data/zabbix/logs
2 mkdir /usr/local/data/zabbix/pid
3 chown -R zabbix:zabbix /usr/local/data/zabbix/
```

配置 PHP

- 修改时区信息

```
1 vi /etc/opt/rh/rh-php72/php-fpm.d/zabbix.conf
```

```
1 php_value[date.timezone] = Asia/Shanghai # 去掉前面分号
```

配置字体，避免前台监控图形中文乱码

```
1 yum -y install wqy-microhei-fonts
2 mv /usr/share/fonts/dejavu/DejaVuSans.ttf
  /usr/share/fonts/dejavu/DejaVuSans.ttf.bak
3 cp -f /usr/share/fonts/wqy-microhei/wqy-microhei.ttf
  /usr/share/fonts/dejavu/DejaVuSans.ttf
```

启动zabbix相关服务并设置开机自启动

```
1 systemctl restart zabbix-server httpd rh-php72-php-fpm
2 systemctl enable zabbix-server httpd rh-php72-php-fpm
```

浏览器上访问前端，进行初始化安装

- 访问<http://IP>，然后【默认点击下一步】

这一步结束后，初始化完成，并进入登录界面，默认用户名是Admin，密码是zabbix，登录后请及时修改密码

出现报错

```
1 PHP xmlwriter extension missing zabbix
2 PHP xmlreader extension missing zabbix
```

解决办法：

```
1 yum install php-xmlwriter -y
```

然后重新启动。

最新数据

主机群组 在此输入搜索 选择 名称 在此输入搜索 选择 查看无资料项目 ☒ 查看细节 ☐

主机 在此输入搜索 选择 应用 重设

<input type="checkbox"/> 主机	名称	最近检查记录	最新数据	更改
<input checked="" type="checkbox"/> Zabbix server	CPU (17 监控项)			
<input type="checkbox"/>	Context switches per second			图形
<input type="checkbox"/>	CPU guest nice time			图形
<input type="checkbox"/>	CPU guest time			图形
<input type="checkbox"/>	CPU idle time			图形
<input type="checkbox"/>	CPU interrupt time			图形
<input type="checkbox"/>	CPU iowait time			图形
<input type="checkbox"/>	CPU nice time			图形
<input type="checkbox"/>	CPU softirq time			图形
<input type="checkbox"/>	CPU steal time			图形
<input type="checkbox"/>	CPU system time			图形
<input type="checkbox"/>	CPU user time			图形
<input type="checkbox"/>	CPU utilization			图形
<input type="checkbox"/>	Interrupts per second			图形
<input type="checkbox"/>	Load average (1m avg)			图形
<input type="checkbox"/>	Load average (5m avg)			图形
<input type="checkbox"/>	Load average (15m avg)			图形
<input type="checkbox"/>	Number of CPUs			图形
<input checked="" type="checkbox"/> Zabbix server	General (9 监控项)			
<input type="checkbox"/>	Maximum number of open file descriptors			图形
<input type="checkbox"/>	Maximum number of processes			图形
<input type="checkbox"/>	Number of loaded in users			图形

Zabbix 客户端

客户端采用 agent2。

- 服务端和客户端都配置时间同步

```
1 yum install -y ntpdate
2 ntpdate -u ntp.aliyun.com
```

- 客户端配置时区，与服务器保持一致

```
1 mv /etc/localtime{,.bak}
2 ln -s /usr/share/zoneinfo/Asia/Shanghai
3 date
```

- 设置zabbix的下载源，安装zabbix-agent2

```
1 rpm -ivh https://mirrors.aliyun.com/zabbix/zabbix/5.0/rhel/7/x86_64/zabbix-
  release-5.0-1.e17.noarch.rpm
2
3 cd /etc/yum.repos.d/
4
5 sed -i 's#http://repo.zabbix.com#https://mirrors.aliyun.com/zabbix#'
  /etc/yum.repos.d/zabbix.repo
6
7 yum install -y zabbix-agent2
```

- 修改 agent2 配置文件

```
1 vim /etc/zabbix/zabbix_agent2.conf
```

```
1 Server=192.168.116.128           #80行, 指定 zabbix 服务端的 IP 地址
2 ServerActive=192.168.116.128     #123行, 指定 zabbix 服务端的 IP 地址
3 Hostname=localhost               #134行, 指定当前 zabbix 客户端的主机名
4
5 # ListenPort 监听端口
```

- 启动zabbix-agent2

```
1 systemctl start zabbix-agent2.service
2 systemctl enable zabbix-agent2.service
```

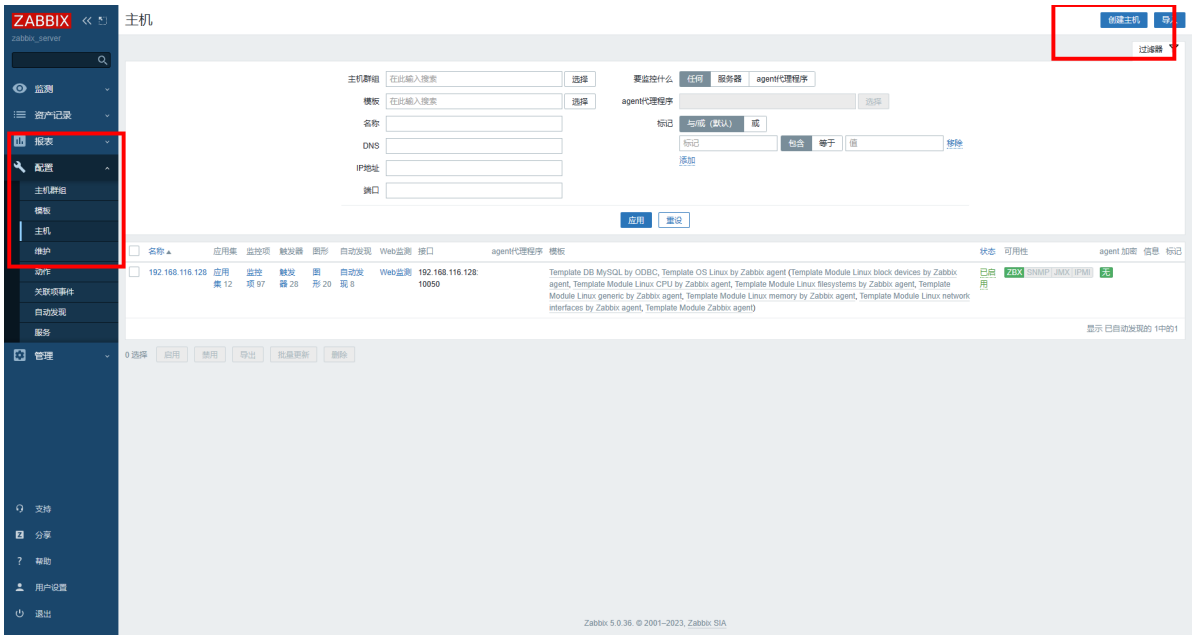
- 在服务端验证zabbix-agent2的连通性

```
1 yum install -y zabbix-get        #安装 zabbix 主动获取数据的命令
2
3 zabbix_get -s '192.168.116.128' -p 10050 -k 'agent.ping'
4 # 返回1为连接成功。
5
6 zabbix_get -s '192.168.116.128' -p 10050 -k 'system.hostname' #客户端的主机名
```

Zabbix 使用

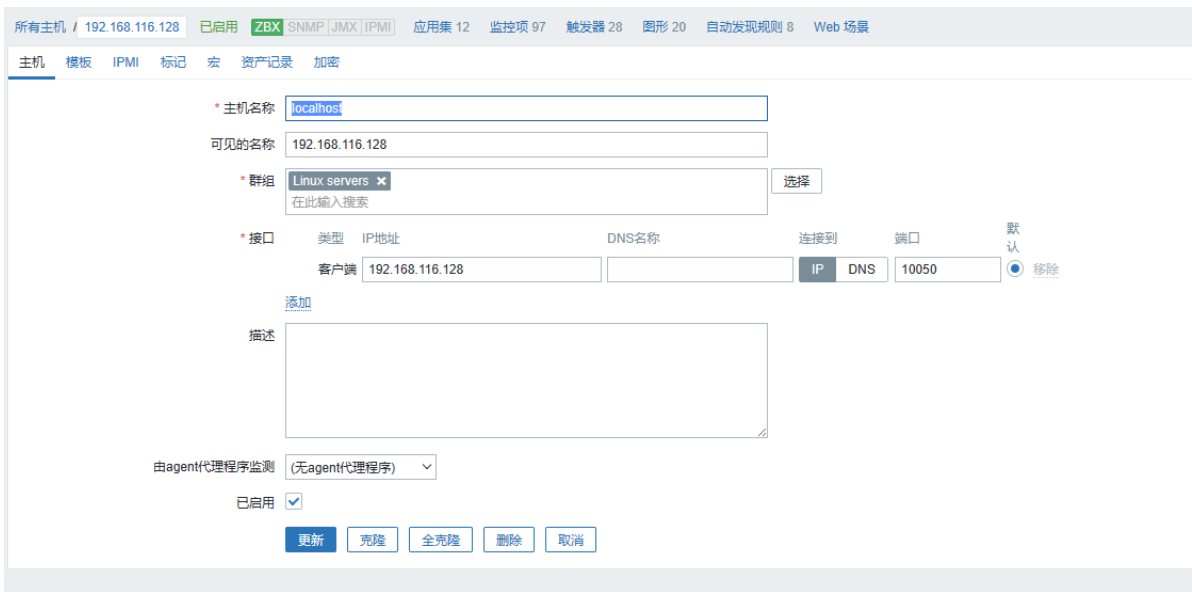
在Web页面中添加agent主机

点击【配置】-【主机】-【创建主机】



创建主机，填写信息如下：

- 主机名称和之前的【Hostname】一致
- 客户端是IP就是客户端主机的IP地址



监控MySQL

【主机】-【选择相应客户端主机】-【添加模板】-【修改模板的宏数据】

所有模板 / Template DB MySQL by Zabbix ag...

应用集 2 监控项 48 触发器 11 图形 6 聚合图形 1 自动发现规则 3 Web 场景

模板 链接的模板 标记 宏

模板 宏 继承模板的宏

宏	值	描述
{MYSQL.ABORTED_CONN.MAX.WARN}	3	Number of failed attempts to connect to the MySQL s erver for trigger expression. 移除
{MYSQL.BUFF_UTIL.MIN.WARN}	50	The minimum buffer pool utilization percentage for trig ger expression. 移除
{MYSQL.CREATED_TMP_DISK.TABLES. MAX.WARN}	10	The maximum number of created tmp tables on a dis k per second for trigger expressions. 移除
{MYSQL.CREATED_TMP_FILES.MAX.WA RN}	10	The maximum number of created tmp files on a disk p er second for trigger expressions. 移除
{MYSQL.CREATED_TMP_TABLES.MAX. WARN}	30	The maximum number of created tmp tables in memo ry per second for trigger expressions. 移除
{MYSQL.DSN}	tcp://192.168.116.128:3306	System data source name such as <tcp://host:port or unix:/path/to/socket/>. 移除
{MYSQL.INNODB_LOG_FILES}	2	Number of physical files in the InnoDB redo log for cal culating innodb_log_file_size. 移除
{MYSQL.PASSWORD}	123456	MySQL user password. 移除
{MYSQL.REPL_LAG.MAX.WARN}	30m	The lag of slave from master for trigger expression. 移除
{MYSQL.SLOW_QUERIES.MAX.WARN}	3	The number of slow queries for trigger expression. 移除
{MYSQL.USER}	root	MySQL user name. 移除

添加

更新 克隆 全克隆 删除 删除并清除 取消

lld 自动发现

创建发现规则

在Template OS Linux by Zabbix agent模板中测试。

所有模板 / Template OS Linux by Zabbix agent 应用集 11 监控项 42 触发器 14 图形 8 聚合图形 1 自动发现规则 3 Web 场景

主机群组 在此输入搜索 选择 类型 所有 状态 所有

主机 Template OS Linux by Zabbix agent 选择 更新间隔 资源周期不足 状态 所有

名称 键值

应用 重设

<input type="checkbox"/>	主机	名称 ▲	监控项	触发器	图形	主机	键值	间隔	类型	状态	信息
<input type="checkbox"/>	Template OS Linux by Zabbix agent	Template Module Linux block devices by Zabbix agent: Block devices discovery	监控项原型 9	触发器类型 1	图形原型 3	主机模板	vfs dev discovery	1h	Zabbix 客户端	已启用	
<input type="checkbox"/>	Template OS Linux by Zabbix agent	Template Module Linux filesystems by Zabbix agent: Mounted filesystem discovery	监控项原型 4	触发器类型 4	图形原型 1	主机模板	vfs fs discovery	1h	Zabbix 客户端	已启用	
<input type="checkbox"/>	Template OS Linux by Zabbix agent	Template Module Linux network interfaces by Zabbix agent: Network interface discovery	监控项原型 9	触发器类型 4	图形原型 1	主机模板	net.if discovery	1h	Zabbix 客户端	已启用	

显示 已自动发现的 3 中的 3

点击【自动发现规则】-【创建发现规则】

自动发现规则

所有模板 / Template OS Linux by Zabbix agent 自动发现清单 / LLD-自动发现端口 监控项原型 1 触发器类型 图形原型 1 主机模板

自动发现规则 预处理 LLD 宏 过滤器 覆盖

* 名称 LLD-自动发现端口

类型 Zabbix客户端(主动式)

* 键值 tcp.port

* 更新间隔 1m

自定义时间间隔

类型 灵活 调度 50s 1-7,00:00-24:00 移除

添加

* 资源周期不足 30d

描述

已启用 ☒

更新 克隆 测试 删除 取消

设置对应agent端自定义脚本

- python脚本

```
1  #!/usr/bin/python
2
3  import os
4  import json
5
6  data = {}
7  tcp_list = []
8  port_list = []
9
10 command = os.popen("ss -tln | awk -F'[ :]+' 'NR>=2{print $5}'")
11
12 for port in command:
13     port_list.append(port.strip())
14
15 for port in port_list:
16     port_dict = {}
17     port_dict["#{TCP_PORT}"] = port
18     tcp_list.append(port_dict)
19
20 data["data"] = tcp_list
21
22 content = json.dumps(data, sort_keys=True, indent=4)
23
24 print content
```

- 脚本测试

```
1  [root@localhost zabbix_agentd.d]# python /tmp/test.py
2  {
3      "data": [
4          {
5              "#{TCP_PORT}": "22"
6          },
7          {
8              "#{TCP_PORT}": "10051"
9          },
10         {
11             "#{TCP_PORT}": "9000"
12         },
13         {
14             "#{TCP_PORT}": "80"
15         }
16     ]
17 }
```


设置监控原型

点击刚才创建的发现规则【LLD-自动发现端口】 - 【监控项原型】 - 【创建监控项原型】

所有模板 / Template OS Linux by Zabbix agent 自动发现清单 / LLD-自动发现端口 监控项原型 触发器类型 图形原型 主机模板

监控项原型 预处理

* 名称

listen{#TCP_PORT}

类型

Zabbix 客户端

* 键值

net.tcp.listen[{#TCP_PORT}]

选择

信息类型

数字 (无正负)

单位

* 更新间隔

1m

自定义时间间隔

类型	间隔	期间	动作
灵活	调度	50s	1-7,00:00-24:00

添加

移除

* 历史数据保留时长

不保留历史数据 储存期 90d

* 趋势存储时间

不保留趋势数据 储存期 365d

设置图形原型

点击【图形原型】 - 【创建图形原型】

图形原型

所有模板 / Template OS Linux by Zabbix agent 自动发现清单 / LLD-自动发现端口 监控项原型 1 触发器类型 图形原型 主机模板

图形原型 预览

* 名称

graph{#TCP_PORT}

* 宽

900

* 高

200

图形类别

正常

查看图例

☒

查看工作时间

☒

查看触发器

☒

百分比线(左)

☐

百分比线(右)

☐

纵轴Y最小值MIN

可计算的

纵轴最大值

可计算的

* 监控项

名称	功能	绘图风格	纵轴Y侧	颜色	动作
1: Template OS Linux by Zabbix agent: listen{#TCP_PORT}	平均	线	左侧	1A7C11	移除

添加 添加原型

发现

☒

添加

取消

修改参数

编辑 `zabbix_agent2.conf`。注意，因为客户端使用的是agent2，根据自己使用的找对应的配置文件进行修改。

```
1 | vi /etc/zabbix/zabbix_agent2.conf
```

将 `UnsafeUserParameters` 修改如下:

```
1 UnsafeUserParameters=1
```

然后重新启动

```
1 systemctl restart zabbix-agent2.service
```

测试

```
1 [root@localhost zabbix]# zabbix_get -s 192.168.116.128 -p 10050 -k  
  "net.tcp.listen[22]"  
2 1  
3 [root@localhost zabbix]# zabbix_get -s 192.168.116.128 -p 10050 -k  
  "tcp.port"  
4 {  
5     "data": [  
6         {  
7             "#TCP_PORT": "22"  
8         },  
9         {  
10            "#TCP_PORT": "10051"  
11        },  
12        {  
13            "#TCP_PORT": "9000"  
14        },  
15        {  
16            "#TCP_PORT": "80"  
17        }  
18    ]  
19 }
```

查看 server web

<input type="checkbox"/> 主机	名称 ▲	间隔	历史记录	趋势	类型	最近检查记录	最新数据	更改	信息
▼ 192.168.116.128	LISTEN-PORT (4 监控项)								
<input type="checkbox"/>	listen[22] net.tcp.listen[22]	1m	90d	365d	Zabbix 客户端	2023-07-30 12:49:04	1		图形
<input type="checkbox"/>	listen[80] net.tcp.listen[80]	1m	90d	365d	Zabbix 客户端	2023-07-30 12:48:07	1		图形
<input type="checkbox"/>	listen[9000] net.tcp.listen[9000]	1m	90d	365d	Zabbix 客户端	2023-07-30 12:48:06	1		图形
<input type="checkbox"/>	listen[10051] net.tcp.listen[10051]	1m	90d	365d	Zabbix 客户端	2023-07-30 12:48:05	1		图形
显示 已自动发现的 4 中的 4									
0 选择 显示详细数据图 显示数据图									

使用zabbix sender

所有模板 / Template OS Linux by Zabbix agent 自动发现清单 / LLD-自动采集数据 监控项原型 触发器类型 图形原型 主机模板

自动发现规则 预处理 LLD 宏 过滤器 覆盖

* 名称

LLD-自动采集数据

类型

Zabbix采集器

* 键值

bind.parameter

* 资源周期不足

30d

允许的主机

描述

已启用

☒

更新

克隆

测试

删除

取消

```
1 - bind.parameter {"data": [{"#BIND_PERF": "BIND INCOMING QUERY"}, {"#BIND_PERF": "BIND INCOMING NOTIFY"}, {"#BIND_PERF": "BIND INCOMING A"}, {"#BIND_PERF": "BIND INCOMING SOA"}, {"#BIND_PERF": "BIND INCOMING PTR"}, {"#BIND_PERF": "BIND INCOMING AAAA"}, {"#BIND_PERF": "BIND INCOMING IXFR"}, {"#BIND_PERF": "BIND INCOMING AXFR"}, {"#BIND_PERF": "BIND INCOMING ANY"}, {"#BIND_PERF": "BIND OUTGOING A"}, {"#BIND_PERF": "BIND OUTGOING NS"}, {"#BIND_PERF": "BIND OUTGOING PTR"}, {"#BIND_PERF": "BIND OUTGOING AAAA"}, {"#BIND_PERF": "BIND OUTGOING DNSKEY"}]}
```

```
1 zabbix_sender -z 192.168.116.128 -p 10051 -vv -c /etc/zabbix/zabbix_agent2.conf -i /etc/zabbix/test.txt
```

但是没有生效，尝试创建监控项（不是自动发现规则），有数据了。