

# 第10章-容器管理与容器监控

学习目标：

- 能够说出Rancher软件的作用，能够在Rancher中部署微服务
- 能够说出influxDB的作用，能够创建数据库、用户、赋予权限
- 能够说出cAdvisor 的作用，能够创建容器与influxDB连接
- 能够说出Grafana 的作用，能够使用Grafana监控容器的内存数据，并配置警告

## 1 容器管理工具Rancher

### 1.1 什么是Rancher

Rancher是一个开源的企业级全栈化容器部署及管理平台。Rancher为容器提供一揽子基础架构服务：CNI兼容的网络服务、存储服务、主机管理、负载均衡、防护墙..... Rancher让上述服务跨越公有云、私有云、虚拟机、物理机环境运行，真正实现一键式应用部署和管理。

<https://www.cnrancher.com/>

### 1.2 Rancher安装

(1) 下载Rancher 镜像

```
docker pull rancher/server
```

(2) 创建Rancher容器

```
docker run -d --name=rancher --restart=always -p 9090:8080 rancher/server
```

restart为重启策略

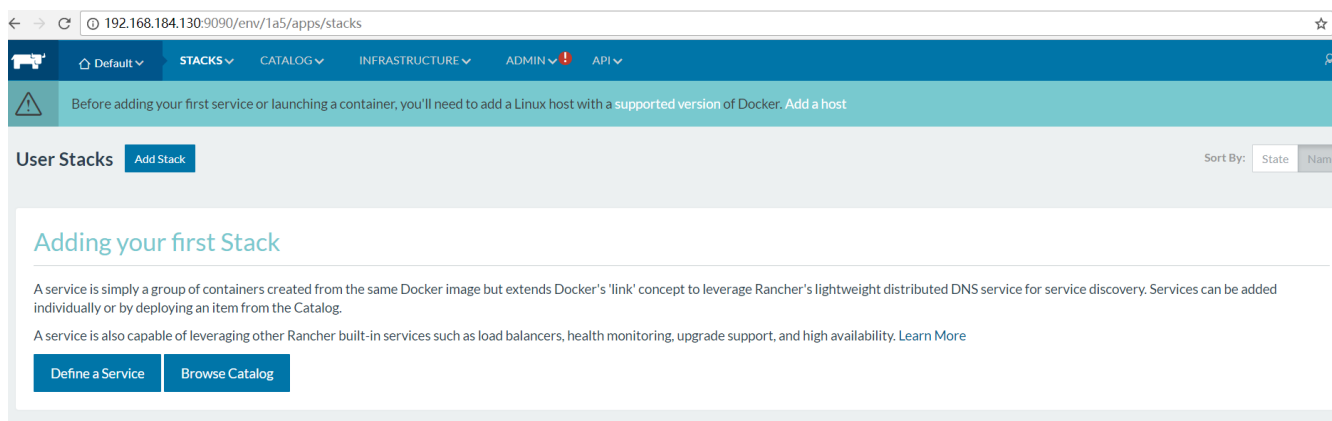
- no，默认策略，在容器退出时不重启容器
- on-failure，在容器非正常退出时（退出状态非0），才会重启容器

- on-failure:3, 在容器非正常退出时重启容器，最多重启3次
- always, 在容器退出时总是重启容器
- unless-stopped, 在容器退出时总是重启容器，但是不考虑在Docker守护进程启动时就已经停止了容器

(3) 在浏览器输入地址: <http://192.168.184.136:9090> 即可看到高端大气的欢迎页

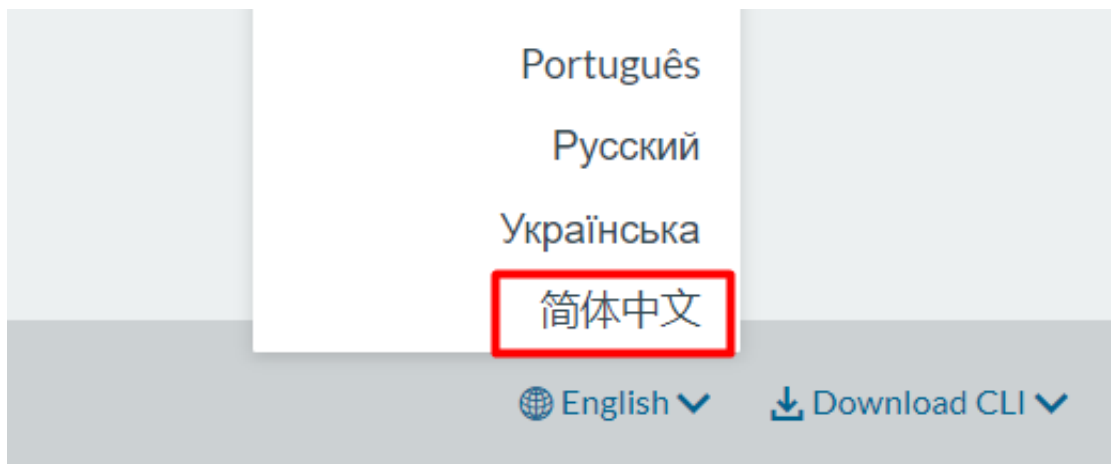


点击Got It 进入主界面



(4) 切换至中文界面

点击右下角的English 在弹出菜单中选择中文



切换后我们就可以看到亲切的中文界面啦~



## 1.3 Rancher初始化

### 1.3.1 添加环境

Rancher 支持将资源分组归属到多个环境。每个环境具有自己独立的基础架构资源及服务，并由一个或多个用户、团队或组织所管理。

例如，您可以创建独立的“开发”、“测试”及“生产”环境以确保环境之间的安全隔离，将“开发”环境的访问权限赋予全部人员，但限制“生产”环境的访问权限给一个小的团队。

(1) 选择“Default --> 环境管理” 菜单

在添加第一个服务或容器之前，必须至少添加一台安装了支持的Docker版本的Linux主机。添加主机

**环境** 添加环境

Rancher 支持将资源分组归属到多个**环境**。每个环境具有自己独立的基础架构资源及服务，并由一个或多个用户、团队或组织所管理。

例如，您可以创建独立的“开发”、“测试”及“生产”环境以确保环境之间的安全隔离，将“开发”环境的访问权限赋予全部人员，但限制“生产”环境的访问权限给一个小的团队。

状态	名称	描述	模板	编排
Unhealthy	Default	无描述	Cattle	Cattle

## (2) 填写名称，点击“创建”按钮

添加环境

名称: tensquare\_dev 描述: 十次方开发环境

环境模板

Cattle (选中) Kubernetes Mesos Swarm Windows

Orchestration: Cattle  
Framework: Network Services, Scheduler, Healthcheck Service  
Networking: Rancher IPsec

## (3) 按照上述步骤，添加十次方测试环境和生产环境

状态	名称	描述	模板	编排	默认
Unhealthy	Default	无描述	Cattle	Cattle	✓
Unhealthy	tensquare_dev	十次方开发环境	Cattle	Cattle	-
Active	tensquare_pro	十次方生产环境	Cattle	Cattle	-
Unhealthy	tensquare_test	十次方测试环境	Cattle	Cattle	-

## (4) 你可以通过点击logo右侧的菜单在各种环境下切换

tensquare\_dev 应用 应用商店 基础架构 系统管理 API

在添加第一个服务或容器之前，必须至少添加一台安装了支持的Docker版本的Linux主机。添加主机

**用户应用** 添加应用

### 创建第一个应用

服务是一组由相同docker镜像创建的容器，服务扩展了Docker的“link”概念以利用Rancher的轻量级分布式DNS服务用于服务发现。服务可以单独添加或通过应用商店部署。

服务也能够利用其他Rancher内置服务，如负载均衡、健康监控、升级支持以及高可用。了解更多

定义一个服务 浏览应用商店

## 1.3.2 添加镜像库

192.168.184.135

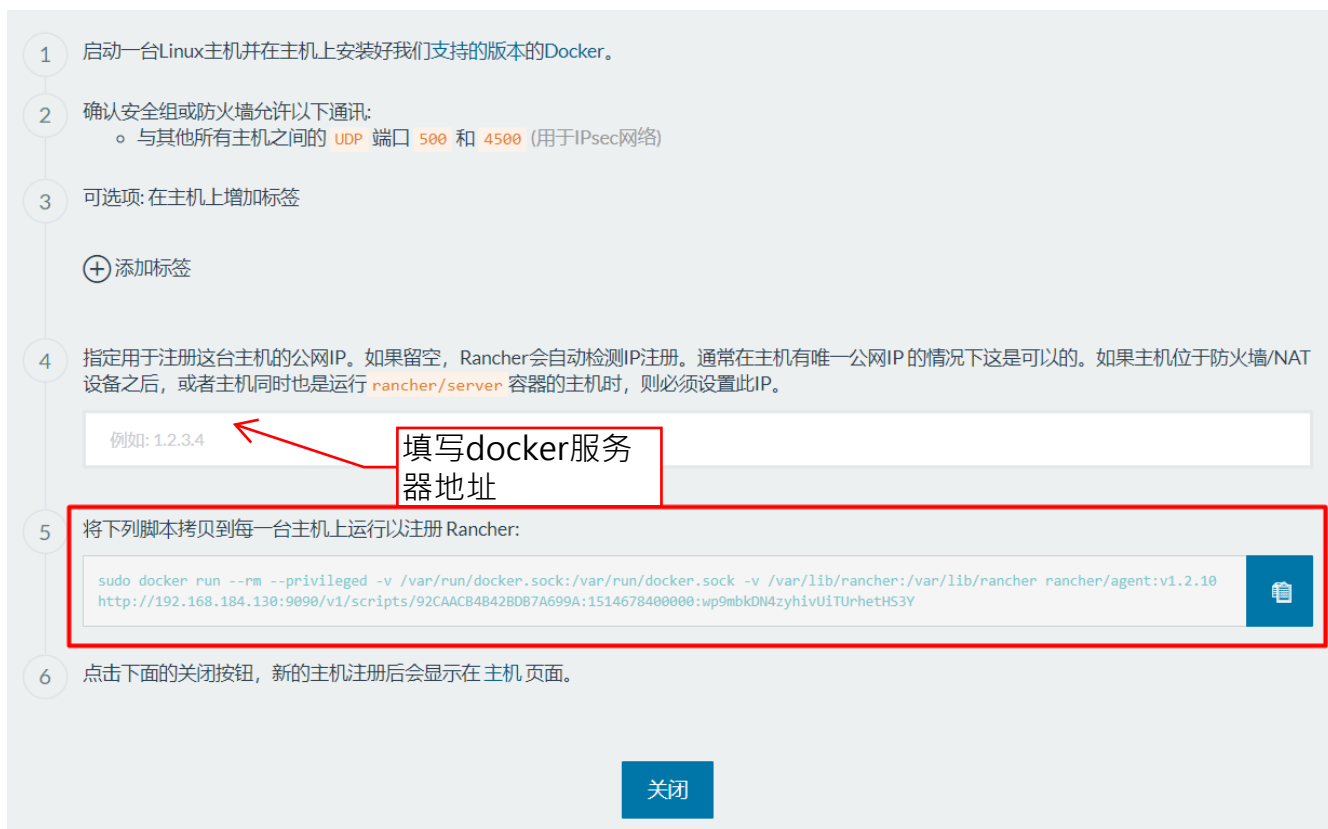
基础架构/镜像库/  
custom

## 1.3.3 添加主机

(1) 选择基础架构-->主机 菜单，点击添加主机



(2) 拷贝脚本



(3) 在服务器（虚拟机）上运行脚本



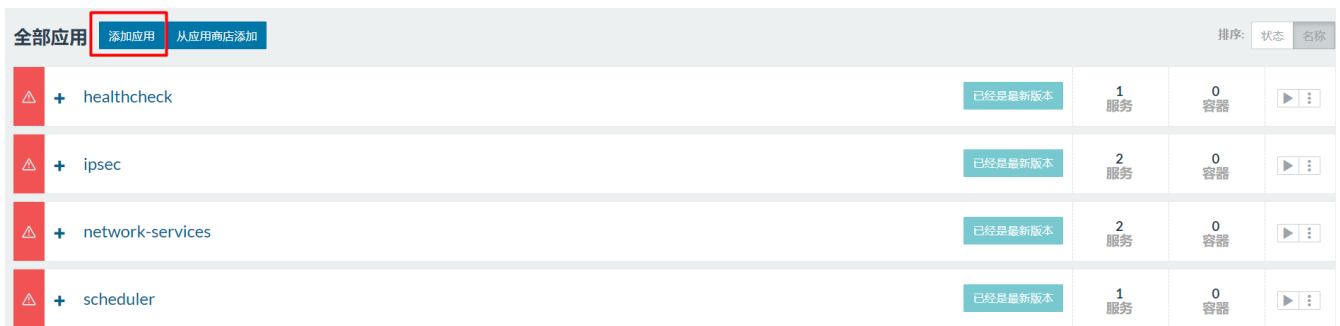
```
[root@pinyoyougou-docker ~]# sudo docker run --rm --privileged -v /var/run/docker.sock:/var/run/docker.sock -v /var/lib/rancher:  
:9090/v1/scripts/92CAACB4B42BDB7A699A:1514678400000:wp9mbkDN4zyhivUiTurhetHS3Y  
Unable to find image 'rancher/agent:v1.2.10' locally  
Trying to pull repository docker.io/rancher/agent ...  
v1.2.10: Pulling from docker.io/rancher/agent  
b3e1c725a85f: Pull complete  
6a710864a9fc: Pull complete  
d0ac3b234321: Pull complete  
87f567b5cf58: Pull complete  
063e24b217c4: Pull complete  
d0a3f58caef0: Pull complete  
16914729cfd3: Pull complete  
4956f3e025a2: Pull complete  
64292afc18d7: Pull complete  
Digest: sha256:5618f36cdb6c6fe25baa3e72977f3a226d43ffd1d74dda6a8860e32b73a1e171  
  
INFO: Running Agent Registration Process, CATTLE_URL=http://192.168.184.130:9090/v1  
INFO: Attempting to connect to: http://192.168.184.130:9090/v1  
INFO: http://192.168.184.130:9090/v1 is accessible  
INFO: Configured Host Registration URL info: CATTLE_URL=http://192.168.184.130:9090/v1 ENV_URL=http://192.168.184.130:9090/v1  
INFO: Inspecting host capabilities  
INFO: Boot2Docker: false  
INFO: Host writable: true  
INFO: Token: xxxxxxxx  
INFO: Running registration  
INFO: Printing Environment  
INFO: ENV: CATTLE_ACCESS_KEY=4A9796CCA881CF11FEDD  
INFO: ENV: CATTLE_HOME=/var/lib/cattle  
INFO: ENV: CATTLE_REGISTRATION_ACCESS_KEY=registrationToken  
INFO: ENV: CATTLE_REGISTRATION_SECRET_KEY=xxxxxxx  
INFO: ENV: CATTLE_SECRET_KEY=xxxxxxx  
INFO: ENV: CATTLE_URL=http://192.168.184.130:9090/v1  
INFO: ENV: DETECTED_CATTLE_AGENT_IP=172.17.0.1  
INFO: ENV: RANCHER_AGENT_IMAGE=rancher/agent:v1.2.10  
INFO: Launched Rancher Agent: 9cdd147e7f79d92a027064ce8eccba4290a1ca99db403929d97f66ee2dbc832d  
[root@pinyoyougou-docker ~]#
```

(4) 点击关闭按钮后，会看到界面中显示此主机。我们可以很方便地管理主机的每个容器的开启和关闭



## 1.3.4 添加应用

点击应用-->全部(或用户) ， 点击“添加应用”按钮



填写名称和描述

## 添加应用

名称

tensquare

描述

十次方

可选:导入COMPOSE

可选:docker-compose.yml

docker-compose.yml文件的内容

上传

可选:rancher-compose.yml

rancher-compose.yml文件的内容

上传

高级选项 ^

创建

取消

点击“创建”按钮，列表中增加了新增的应用

全部应用		添加应用	从应用商店添加	排序: 状态 名称			
+	healthcheck	已经是最新版本	1 服务	0 容器			
+	ipsec	已经是最新版本	2 服务	0 容器			
+	network-services	已经是最新版本	2 服务	0 容器			
+	scheduler	已经是最新版本	1 服务	0 容器			
+	tensquare 十次方	添加服务	0 服务	0 容器			

## 1.4 应用部署

### 1.4.1 MySQL部署

镜像: centos/mysql-57-centos7 增加数据库服务

名称

mysql

描述

mysql

选择镜像\*

centos/mysql-57-centos7

创建前总是拉取镜像

端口映射

公开主机端口

3306

>

私有容器端口

3306

/

协议

TCP

显示主机IP选项

注意：添加环境变量 MYSQL\_ROOT\_PASSWORD=123456

环境变量

添加环境变量

变量

MYSQL\_ROOT\_PASSWORD

=

值

123456

高级技巧: 在键(Key)输入栏中粘贴一行或多行的key=value键值对能够批量输入。



点击创建按钮，完成创建 上述操作相当于以下docker命令

```
docker run -di --name mysql -p 3306:3306 -e MYSQL_ROOT_PASSWORD=123456  
centos/mysql-57-centos7
```

Active	mysql ⓘ	镜像: centos/mysql-57-centos7 端口: 3306	服务·	1 容器	ⓘ ⋮
--------	---------	--------------------------------------	-----	------	-----

完成后服务列表中存在并且状态为激活 使用SQLyog测试链接，执行建表语句

## 1.4.2 RabbitMQ部署

镜像: rabbitmq:management 端口映射5671 5672 4369 15671 15672 25672

名称	描述
rabbitmq	rabbitmq

选择镜像\* ☐ 创建前总是拉取镜像

rabbitmq:management

+ 端口映射

公开主机端口	私有容器端口	协议
5671	5671	TCP
5672	5672	TCP
4369	4369	TCP
15671	15671	TCP
15672	15672	TCP
25672	25672	TCP

显示主机IP选项

浏览器访问 <http://192.168.184.136:15672/>

## 1.4.3 Redis部署（学员实现）

进入应用，点击添加服务 名称redis，镜像redis，端口映射6379

名称	描述
redis	redis

选择镜像\* ☐ 创建前总是拉取镜像

redis

+ 端口映射

公开主机端口	私有容器端口	协议
6379	6379	TCP

显示主机IP选项

创建后使用客户端测试链接

```
redis-cli -h 192.168.184.136
```

测试成功

## 1.4.4 MongoDB部署（学员实现）

名称mongo 镜像mongo 端口映射27017

## 1.4.5 ElasticSearch部署（学员实现）

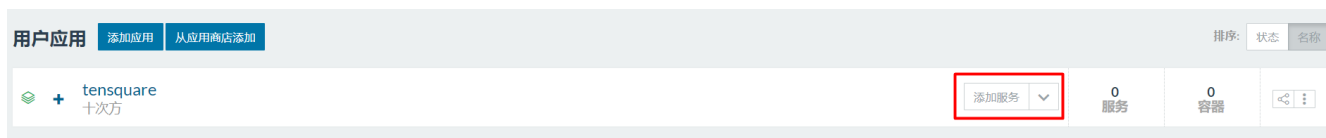
名称elasticsearch 镜像elasticsearch:5.6.8 端口映射9300 9200

添加后，浏览器测试：<http://192.168.184.136:9200/>

# 1.5 微服务容器部署

## 1.5.1 Eureka微服务容器化部署

（1）在用户应用界面中点击“添加服务”



（2）填写名称、描述、镜像和端口映射，点击创建按钮

名称eureka 镜像 192.168.184.135:5000/tensquare\_eureka:1.0-SNAPSHOT

## 添加服务

数量

☒ Run 1 个容器

☐ 总是在每台主机上运行一个此容器的实例

+ 添加从容器

---

名称  描述

选择镜像\*  ☒ 创建前总是拉取镜像

+ 端口映射

公开主机端口	私有容器端口	协议
<input type="text" value="6868"/>	<input type="text" value="6868"/>	TCP

### (3) 服务添加成功

应用:  添加服务

描述: 十次方

Active	eureka	镜像: 192.168.184.130:5000/tensquare_eureka:0.0.8-SNAPSHOT 端口: 6868	服务	1 容器
--------	--------	---	----	------

### (4) 我们现在访问以下我们的系统

<http://192.168.184.136:6868/> 可以正常访问

## 1.5.2 配置中心微服务部署

### 创建容器

添加服务config 镜像 192.168.184.135:5000/tensquare\_config:1.0-SNAPSHOT

映射端口: 12000

测试 浏览器输入 <http://192.168.184.135:12000/base-dev.yml> 可以查看到配置文件内容

## 1.5.3 基础微服务部署

(1) 添加服务base-service 镜像tensquare\_base:1.0-SNAPSHOT 端口映射9001

名称	描述	
base-service	基础信息微服务	
选择镜像* <input type="checkbox"/> 创建前总是拉取镜像		
192.168.184.130:5000/tensquare_base:0.0.1-SNAPSHOT		
+ 端口映射		
公开主机端口	私有容器端口	协议
9002	9002	TCP

(2) 测试微服务 浏览器打开网址 <http://192.168.184.136:9001/label> 看是否可以看到标签列表

## 1.6 扩容与缩容

### 1.6.1 扩容

(1) 在Rancher将创建的base-service（基础信息微服务）删除

(2) 重新创建base-service，不设置端口映射

名称	描述
base-service	基础信息微服务
选择镜像* <input type="checkbox"/> 创建前总是拉取镜像	
192.168.184.130:5000/tensquare_base:0.0.1-SNAPSHOT	
+ 端口映射	

(3) 在选择菜单API -->WebHooks，点击“添加接收器”按钮

 Default 应用 应用商店 基础架构 系统管理 API

 在添加第一个服务或容器之前，必须至少添加一台安装了支持的Docker版本的Linux主机。添加主机

 Experimental: More webhook features will be added in future releases, and the existing capabilities may change.

Receiver Hooks 添加接收器

Receiver hooks 提供一个URL，在访问该URL时能够触发Rancher内部相应的动作。

状态	名称	类型	详情
无receiver hooks.			

(4) 填写名称等信息，选择要扩容的服务，点击创建按钮



名称\*

scale-base

类型

扩缩容服务

操作

☒ 扩容 ☐ 缩容

目标服务\*

tensquare/base-service

步长

2

最小数量

1



最大数量

20

创建

取消

(5) 接收器列表中新增了一条记录，点击触发地址将地址复制到剪切板

Receiver Hooks					添加接收器
Receiver hooks 提供一个URL，在访问该URL时能够触发Rancher内部相应的动作。					
状态	名称	类型	详情	触发地址	
Active	scale-base	扩缩容服务	扩缩容动作 up tensquare/base-service 以步长 2	 	

(6) 使用postman测试:

POST

http://192.168.184.130:9090/v1-webhooks/endpoint?key=Hqmo1oZF3lFmSa5q4knUpDQJlfnHfcqUfo0ZPqHh&projectId=1a7

Params

Send

Authorization

Headers

Body

Pre-request Script

Tests

TYPE

Inherit auth from parent

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

This request is not inheriting any authorization helper at the moment. Save it in a collection to use the parent's authorization help

测试后，发现容器由原来的1个变为了3个

描述: 十次方					
Active	base-db ①	镜像: centos/mysql-57-centos7 端口: 33062	服务·	1 容器	⊕ ⋮
Active	base-service ①	镜像: 192.168.184.130:5000/tensquare_base:0.0.1-SNAPSHOT	服务·	3 容器	⊕ ⋮
Active	eureka ①	镜像: 192.168.184.130:5000/tensquare_eureka:0.0.8-SNAPSHOT 端口: 6868	服务·	1 容器	⊕ ⋮
Active	ha-base-service ①	到: base-service 端口: 9002/tcp	负载均衡	1 容器	⊕ ⋮
Active	redis ①	镜像: redis 端口: 6379	服务·	1 容器	⊕ ⋮

打开erueka，发现服务也有3个

Instances currently registered with Eureka			
Application	AMIs	Availability Zones	Status
TENSQUARE-BASE	n/a (3)	(3)	UP (3) - a1aaa18ebb08:tensquare-base:9002 , ba879bac3a8a:tensquare-base:9002 , b0f0495894ca:tensquare-base:9002

## 1.6.2 缩容

刚才我们实现了扩容，那么如何减少容器数量呢？我们来试试如何缩容

(1) 添加接收器 ,选择缩容，步长为1表示每次递减1个， 点击创建按钮

名称\*

dec-base

类型

扩缩容服务

操作

☐ 扩容 ☒ 缩容

目标服务\*

tensquare/base-service

步长

1

最小数量

1

最大数量

20

创建

取消

## (2) 创建成功后，复制触发地址

Receiver hooks 提供一个URL，在访问该URL时能够触发Rancher内部相应的动作。

状态	名称	类型	详情	触发地址
Active	dec-base	扩容服务	扩容动作 down tensquare/base-service 以步长 1	
Active	scale-base	扩容服务	扩容动作 up tensquare/base-service 以步长 2	

## (3) 使用postman测试

描述: 十次方

Active	base-db	镜像: centos/mysql-57-centos7 端口: 33062	服务	1 容器	
Active	base-service	镜像: 192.168.184.130:5000/tensquare_base:0.0.1-SNAPSHOT	服务	2 容器	
Active	eureka	镜像: 192.168.184.130:5000/tensquare_eureka:0.0.8-SNAPSHOT 端口: 6868	服务	1 容器	
Active	ha-base-service	到: base-service 端口: 9002/tcp	负载均衡	1 容器	
Active	redis	镜像: redis 端口: 6379	服务	1 容器	

# 2 influxDB

## 2.1 什么是influxDB

influxDB是一个分布式时间序列数据库。cAdvisor仅仅显示实时信息，但是不存储监视数据。因此，我们需要提供时序数据库用于存储cAdvisor组件所提供的监控信息，以便显示除实时信息之外的时序数据。

## 2.2 influxDB安装

### (1) 下载镜像

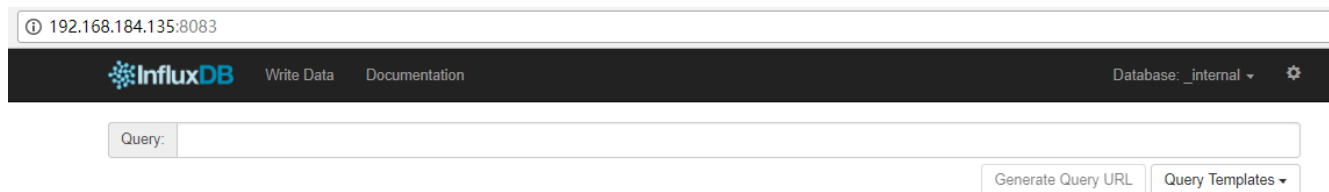
```
docker pull tutum/influxdb
```

### (2) 创建容器

```
docker run -di \
    -p 8083:8083 \
    -p 8086:8086 \
    --expose 8090 \
    --expose 8099 \
    --name influxsrv \
    tutum/influxdb
```

端口概述： 8083端口:web访问端口 8086:数据写入端口

打开浏览器 <http://192.168.184.135:8083/>



## 2.3 influxDB常用操作

### 2.3.1 创建数据库

```
CREATE DATABASE "cadvisor"
```

回车创建数据库

```
SHOW DATABASES
```

查看数据库

### 2.3.2 创建用户并授权

创建用户

```
CREATE USER "cadvisor" WITH PASSWORD 'cadvisor' WITH ALL PRIVILEGES
```

查看用户

```
SHOW USRES
```

用户授权

```
grant all privileges on cadvisor to cadvisor
grant WRITE on cadvisor to cadvisor
grant READ on cadvisor to cadvisor
```

### 2.3.3 查看采集的数据



切换到cadvisor数据库，使用以下命令查看采集的数据

SHOW MEASUREMENTS

现在我们还没有数据，如果想采集系统的数据，我们需要使用**Cadvisor**软件来实现

## 3 cAdvisor

### 3.1 什么是cAdvisor

Google开源的用于监控基础设施应用的工具，它是一个强大的监控工具，不需要任何配置就可以通过运行在Docker主机上的容器来监控Docker容器，而且可以监控Docker主机。更多详细操作和配置选项可以查看Github上的cAdvisor项目文档。

### 3.2 cAdvisor安装

#### (1) 下载镜像

```
docker pull google/cadvisor
```

#### (2) 创建容器

```
docker run --volume=:/rootfs:ro --volume=/var/run:/var/run:rw --  
volume=/sys:/sys:ro --volume=/var/lib/docker:/var/lib/docker:ro --  
publish=8080:8080 --detach=true --link influxsrv:influxsrv --name=cadvisor  
google/cadvisor -storage_driver=influxdb -storage_driver_db=cadvisor -  
storage_driver_host=influxsrv:8086
```

WEB前端访问地址

<http://192.168.184.135:8080/containers/>

性能指标含义参照如下地址

[https://blog.csdn.net/ZHANG\\_H\\_A/article/details/53097084](https://blog.csdn.net/ZHANG_H_A/article/details/53097084)

再次查看influxDB，发现已经有很多数据被采集进去了。

# 4 Grafana

## 4.1 什么是Grafana

Grafana是一个可视化面板（Dashboard），有着非常漂亮的图表和布局展示，功能齐全的度量仪表盘和图形编辑器。支持Graphite、zabbix、InfluxDB、Prometheus和OpenTSDB作为数据源。

Grafana主要特性：灵活丰富的图形化选项；可以混合多种风格；支持白天和夜间模式；多个数据源。

## 4.2 Grafana安装

### （1）下载镜像

```
docker pull grafana/grafana
```

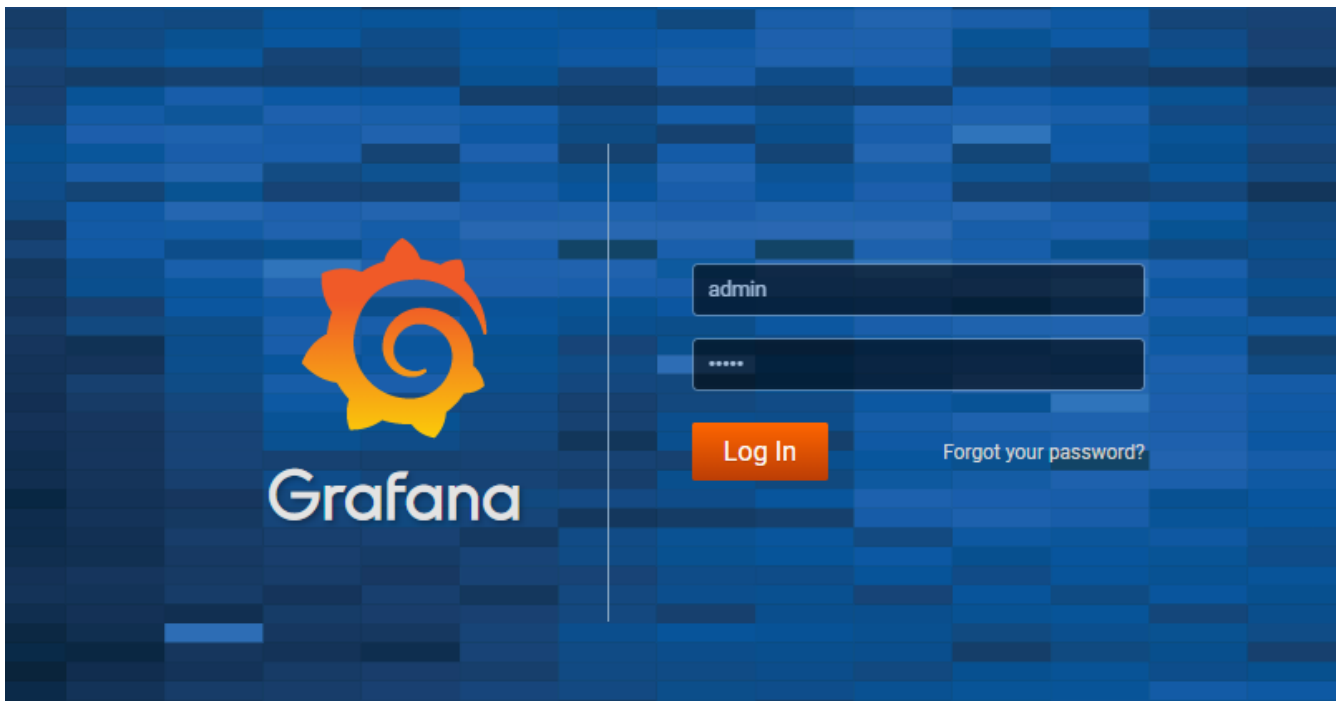
### （2）创建容器

```
docker run -d -p 3001:3000 -e INFLUXDB_HOST=influxsrv -e  
INFLUXDB_PORT=8086 -e INFLUXDB_NAME=cadvisor -e INFLUXDB_USER=cadvisor -e  
INFLUXDB_PASS=cadvisor --link influxsrv:influxsrv --name grafana  
grafana/grafana
```

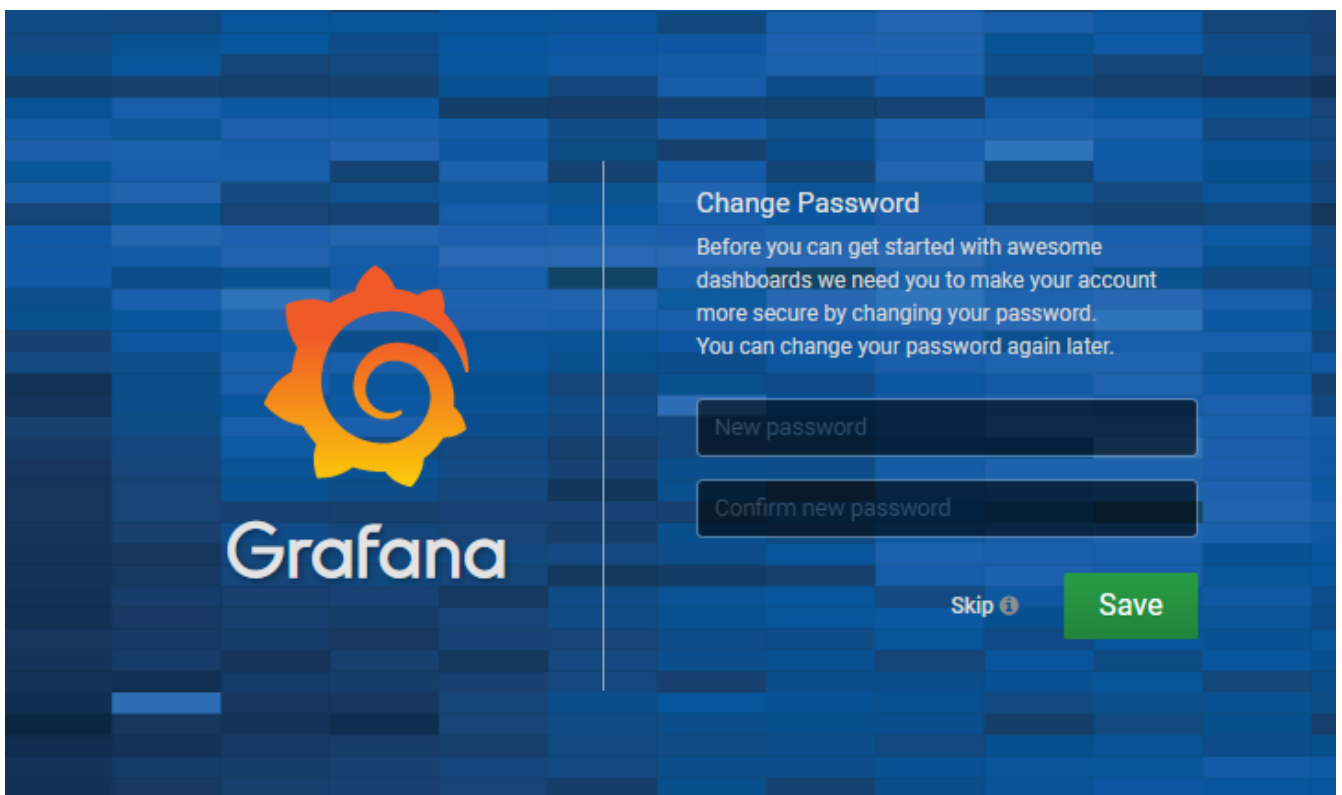
### （3）访问

```
http://192.168.184.135:3001
```

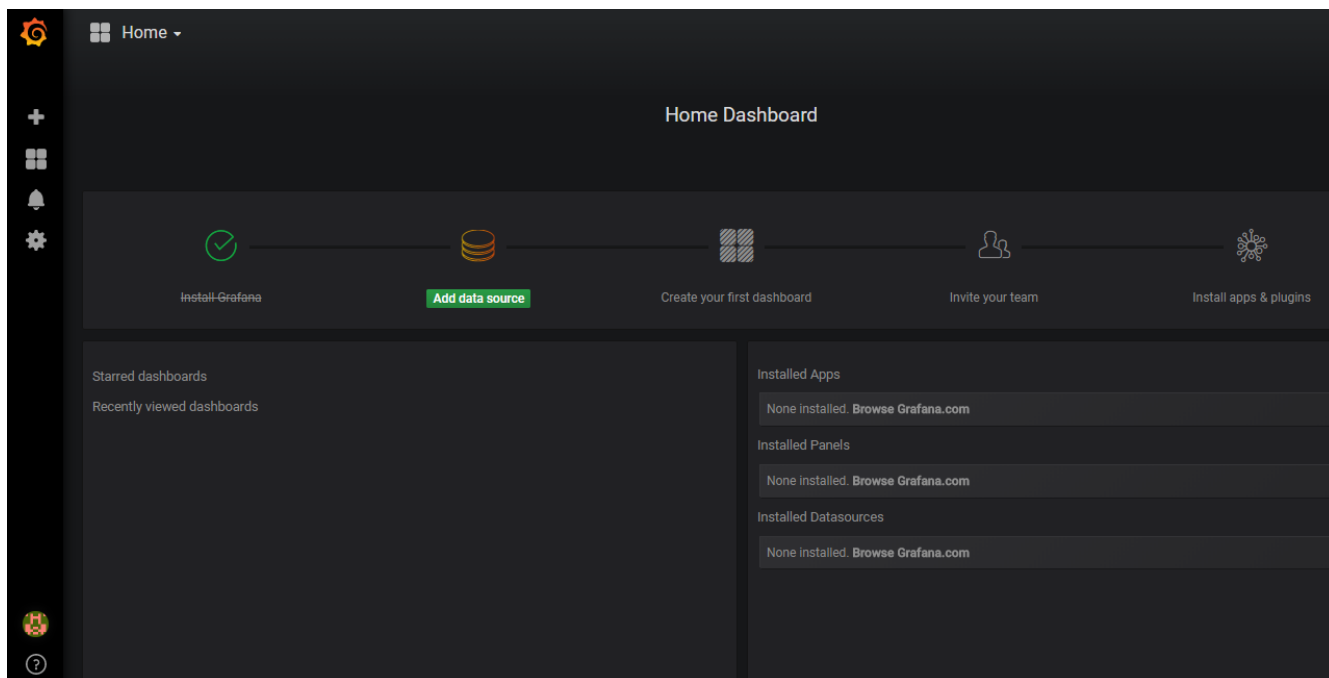
用户名密码均为admin



(4) 登录后提示你修改密码



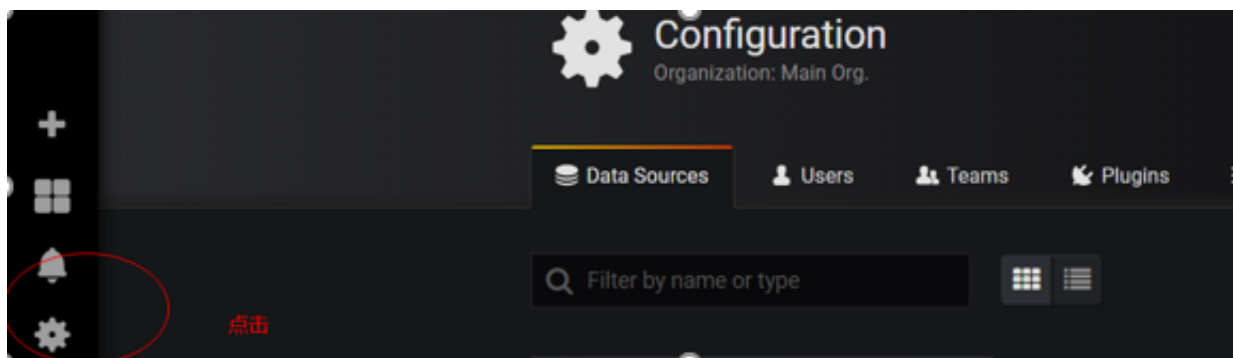
(5) 之后进入主页面



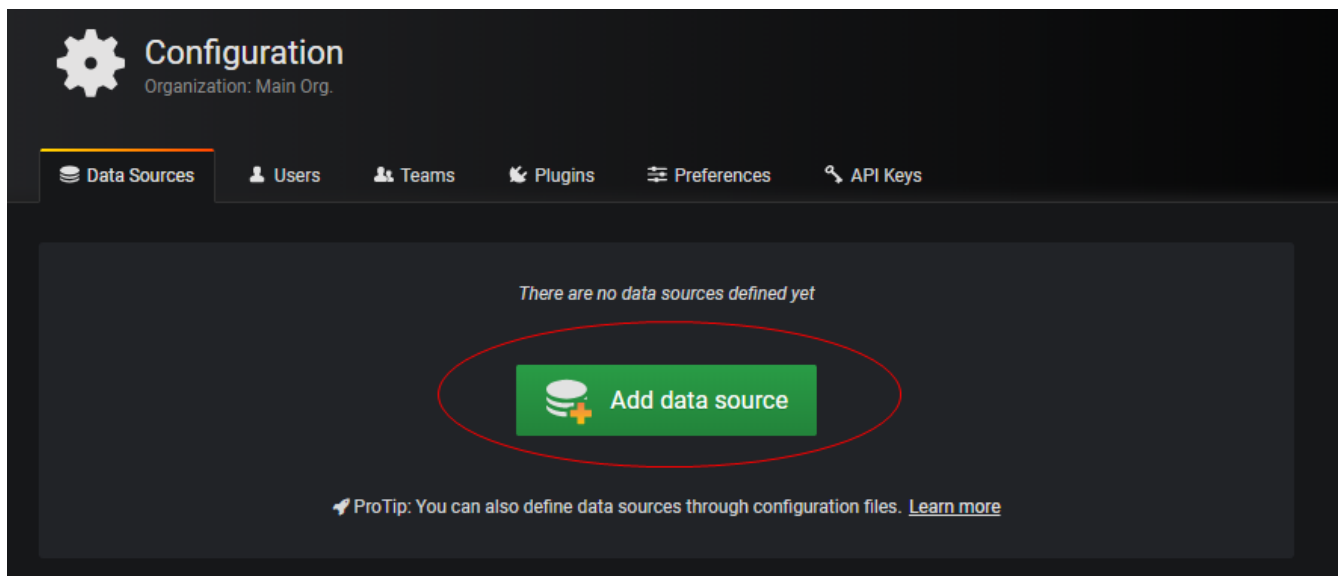
## 4.3 Grafana的使用

### 4.3.1 添加数据源

(1) 点击设置，DataSource



(2) 点击添加data source



(3) 为数据源起个名称，指定类型、地址、以及连接的数据库名、用户名和密码



Name

influxdb

i

Type

InfluxDB

HTTP

URL

http://192.168.184.135:8086

i

Access

Server (Default)

▼

Help ▶

Auth

Basic Auth

☐

With Credentials

i

☐

TLS Client Auth

☐

With CA Cert

i

☐

Skip TLS Verification (Insecure)

☐

Advanced HTTP Settings

Whitelisted Cookies

Add Name

i

InfluxDB Details

Database

cadvisor

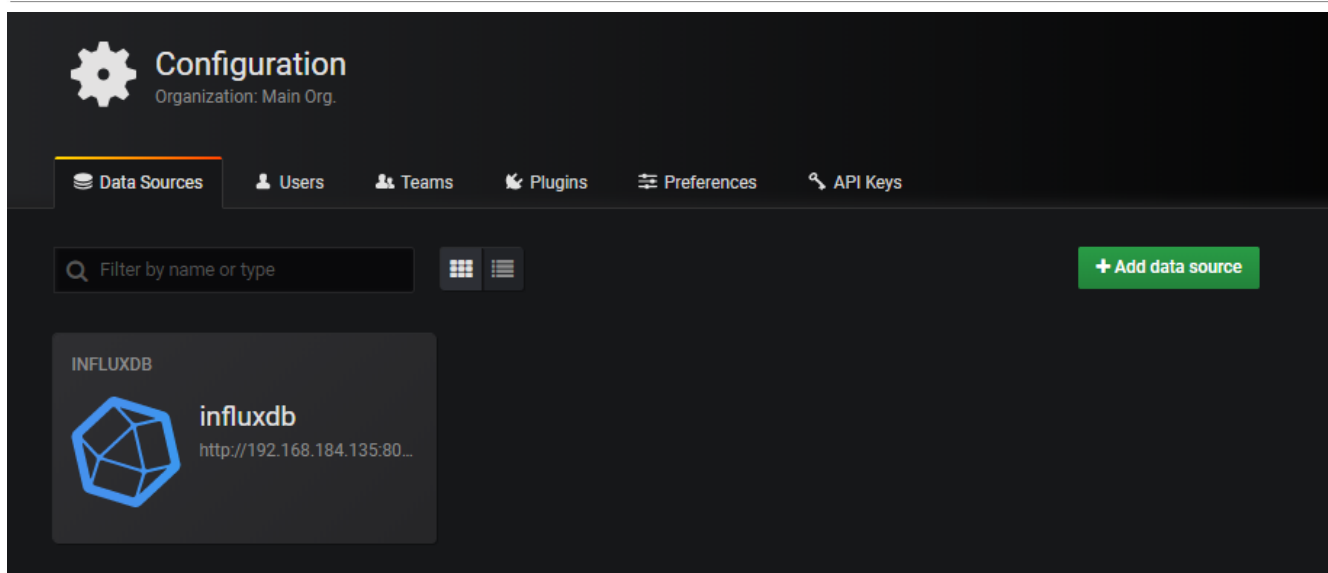
User

cadvisor

Password

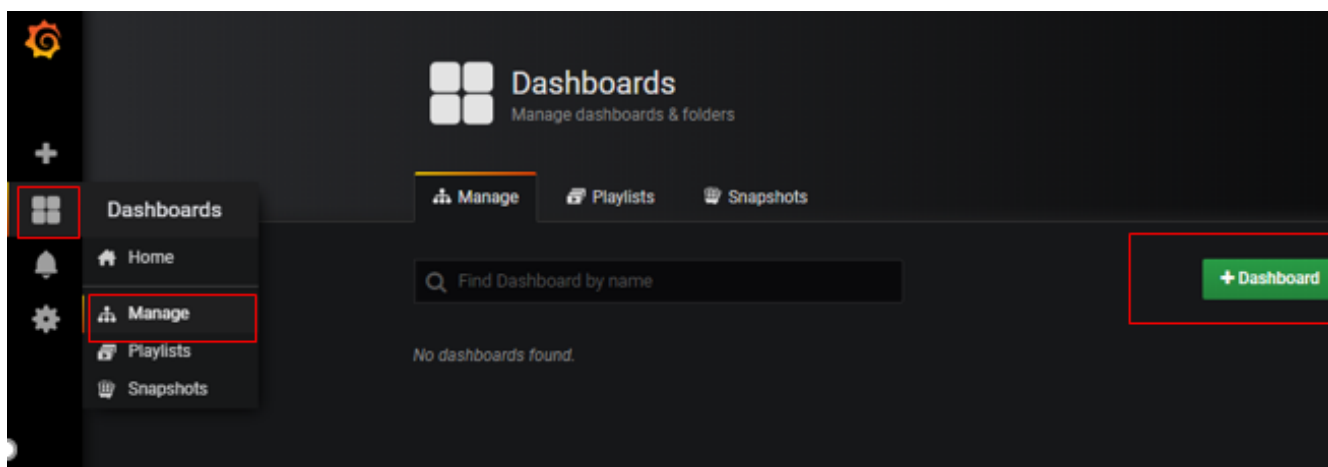
.....

点击保存。数据源建立成功



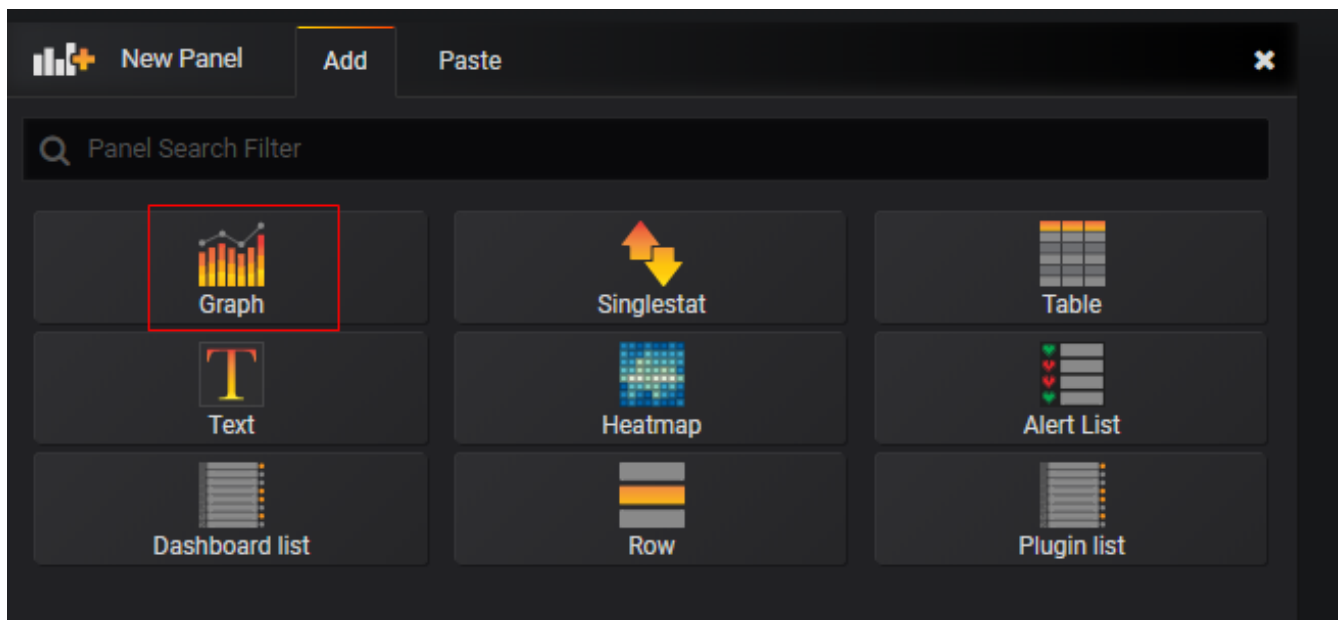
## 4.3.2 添加仪表盘

(1) 选择Dashboards --Manager

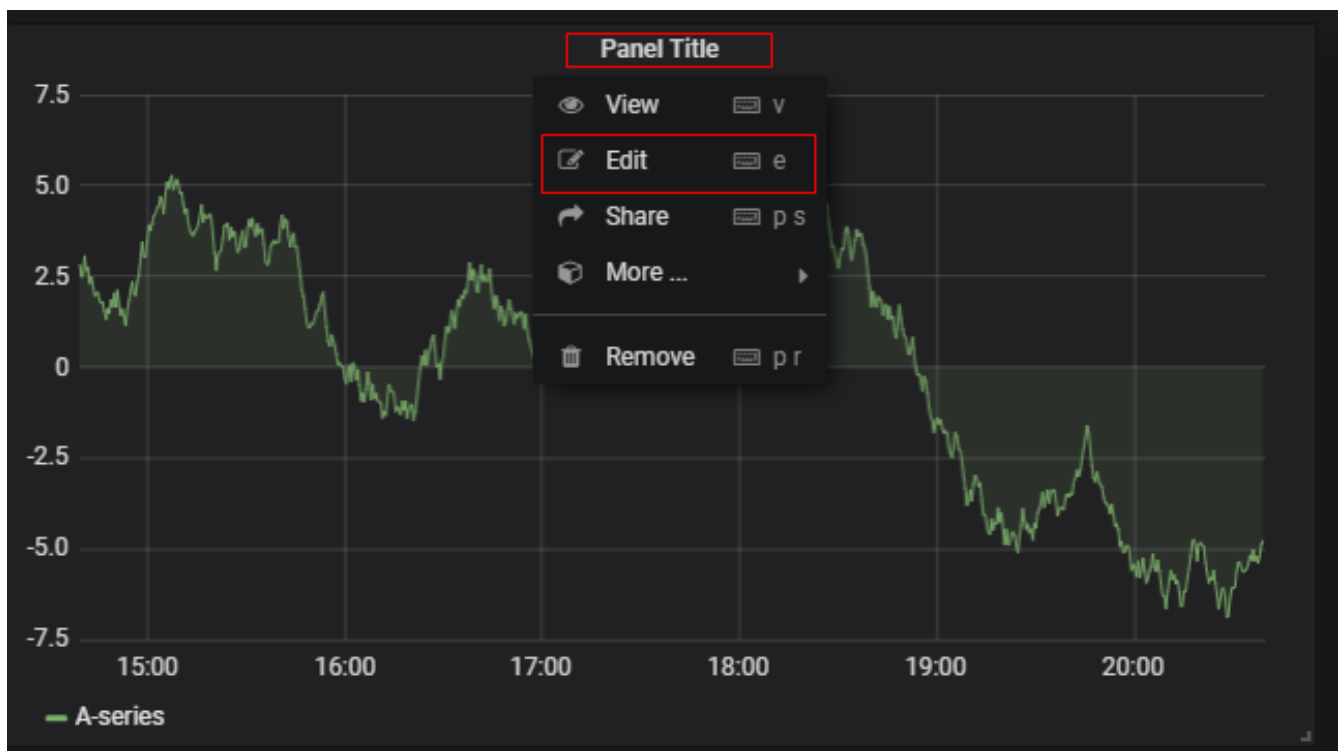


(2) 点击“添加”按钮

(3) 点击Graph 图标



(4) 出现下面图表的界面，点击Panel Title 选择Edit (编辑)



(5) 定义标题等基础信息



Graph

General Metrics Axes Legend Display Alert Time range

Info

Title 基础微服务-内存监控

Description Panel description, supports markdown & links

Transparent ☐

Repeat For each value of

(6) 设置查询的信息为内存，指定容器名称

基础微服务-内存监控

500 Mil

450 Mil

400 Mil

350 Mil

15:00 15:30 16:00 16:30 17:00 17:30 18:00 18:30 19:00 19:30 20:00 20:30

memory\_usage.mean

Graph

General Metrics Axes Legend Display Alert Time range

Data Source influxdb

Options Help Query Inspector

FROM default memory\_usage WHERE container\_name = r-tensquare-base-1-dea78f6a

SELECT field (value) mean ()

GROUP BY time (\$\_interval) fill (null)

FORMAT AS Time series

(7) 指定y轴的单位 为M

基础微服务-内存监控

1000 MiB

800 MiB

600 MiB

400 MiB

200 MiB

0 MiB

15:00 15:30 16:00 16:30 17:00 17:30 18:00 18:30 19:00 19:30 20:00 20:30

memory\_usage.mean

Graph

General Metrics Axes Legend Display Alert Time range

Left Y

Show ☒

Unit mebibytes

Scale linear

Y-Min 0 Y-Max 1000

Decimals auto

Label

Right Y

Show ☒

Unit short

Scale linear

Y-Min auto Y-Max auto

Decimals auto

Label

X-Axis

Show ☒

Mode Time

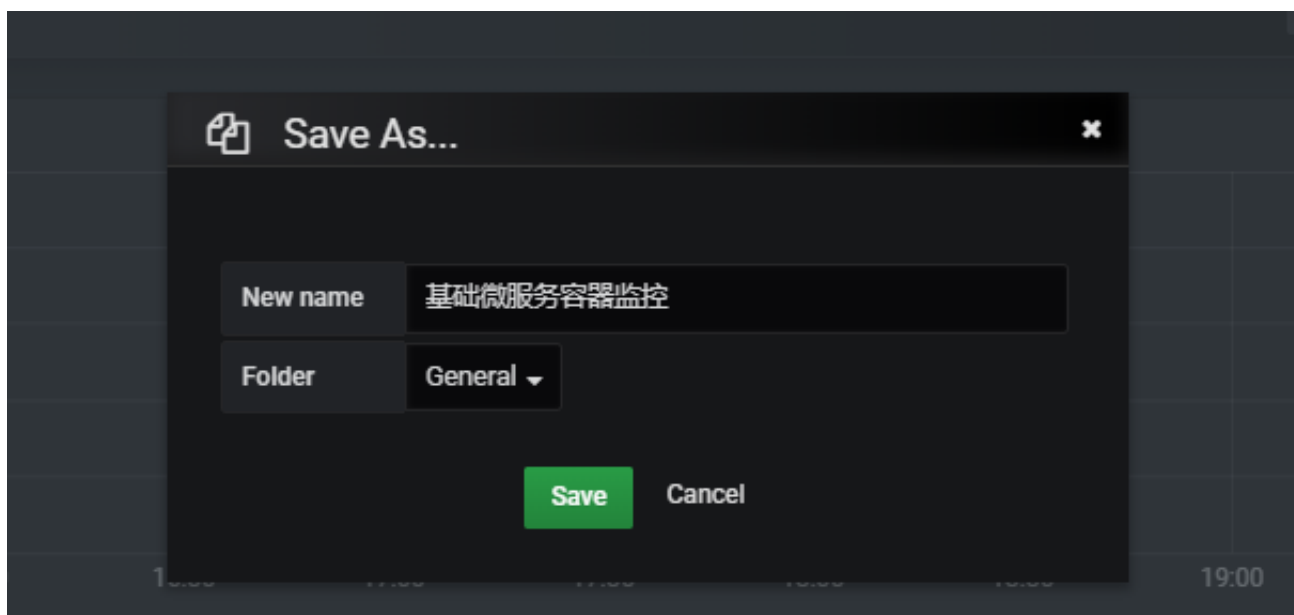
Y-Axes

Align ☐

(8) 保存



填写名称

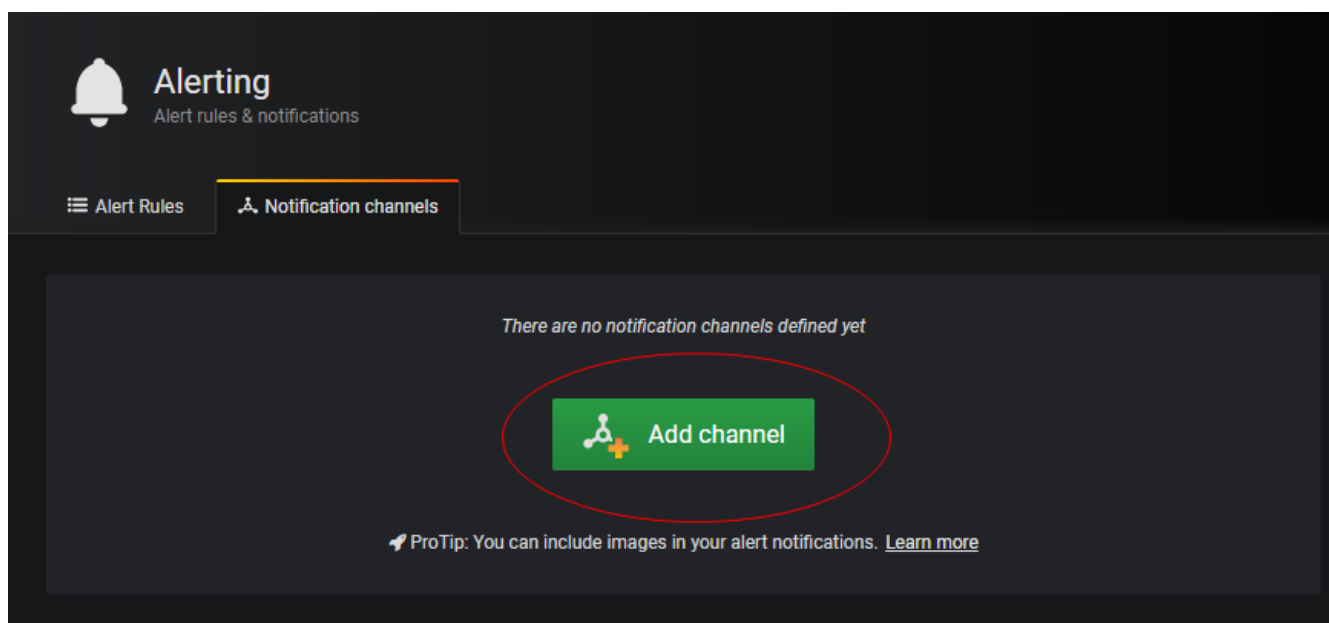


### 4.4.3 预警通知设置


(1) 选择菜单 alerting--> Notification channels



(2) 点击Add channel 按钮



(3) 填写名称，选择类型为webhook ,填写钩子地址



## Alerting

Alert rules & notifications

Alert Rules

Notification channels

### New Notification Channel

Name

scale-base

Type

webhook

Send on all alerts

☐

Include image

☒

### Webhook settings

Url

http://192.168.184.135:9090/v1-webhooks/endpoint...

Http Method

POST

Username

Password


Save

Send Test

Back

这个钩子地址是之前对base微服务扩容的地址

Receiver hooks 提供一个URL，在访问该URL时能够触发Rancher内部相应的动作。

状态	名称	类型	详情	触发地址
Active	scale-base	扩容服务	扩容动作 up? 以步长 2	

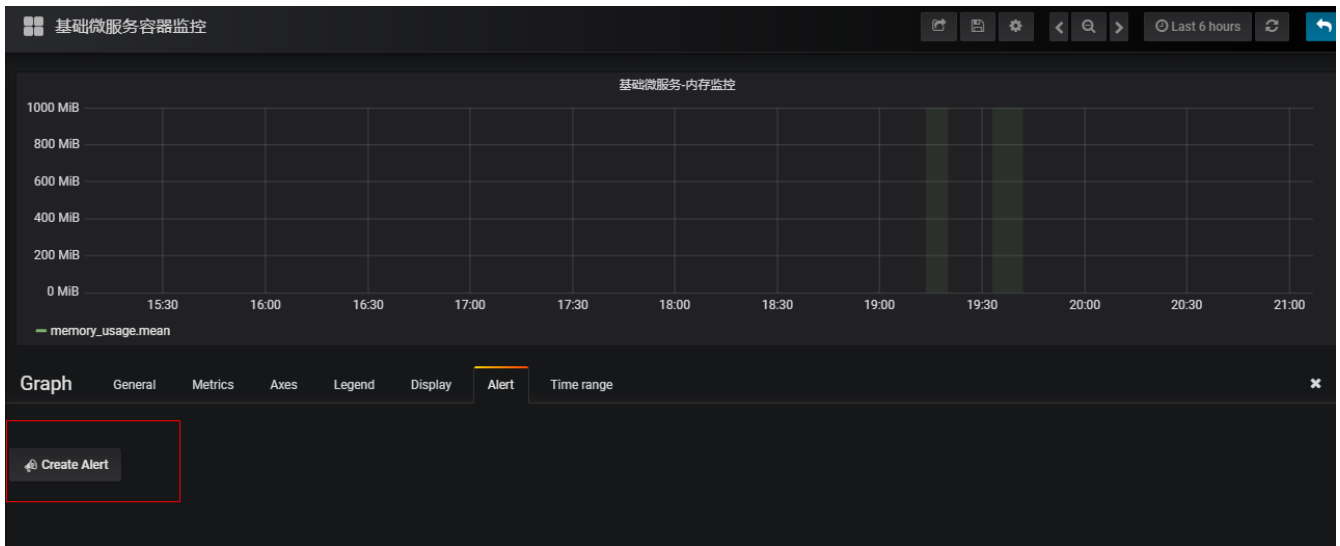
(4) 点击SendTest 测试 观察基础微服务是否增加容器

(5) 点击save保存

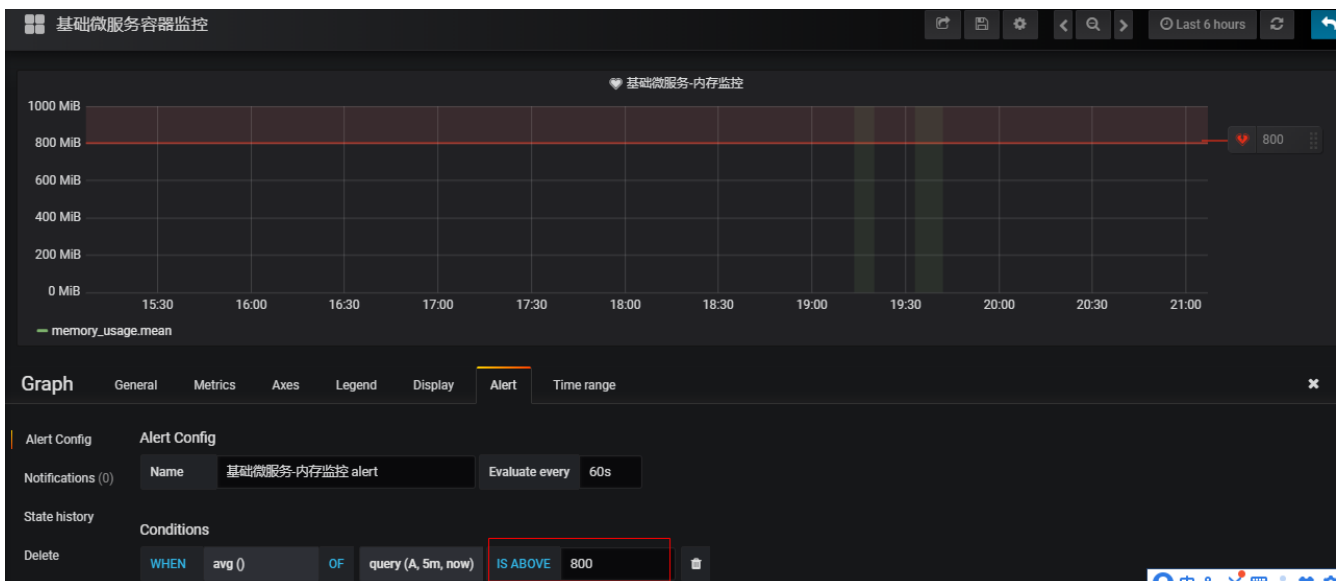
(6) 按照同样的方法添加缩容地址

## 4.4.4 仪表盘预警设置

## (1) 再次打开刚刚编辑的仪表盘

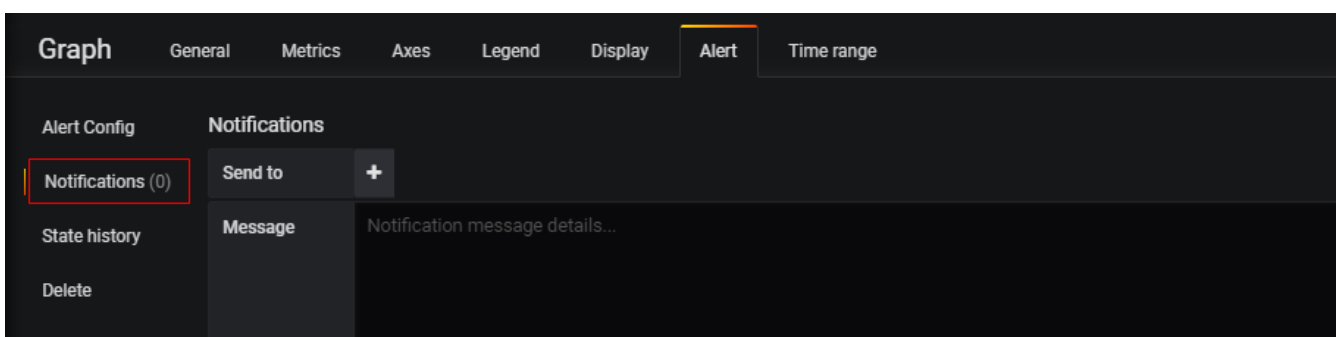


## (2) 点击 Create Alert



## 设置预警线

## (3) 选择通知



Graph

General

Metrics

Axes

Legend

Display

Alert

Time range

Alert Config

Notifications

Notifications (1)

State history

Delete

Send to

scale-base %

+

Message

内存超过800M

保存更改