

# NSD SECURITY DAY06

1. [案例1：实现Zabbix报警功能](#)
2. [案例2：Zabbix自动发现](#)
3. [案例3：Zabbix主动监控](#)
4. [案例4：拓扑图与聚合图形](#)
5. [案例5：自定义监控案例](#)

## 1 案例1：实现Zabbix报警功能

### 1.1 问题

沿用第5天Zabbix练习，使用Zabbix实现报警功能，实现以下目标：

1. 监控Linux服务器系统账户
2. 创建Media，设置邮件服务器及收件人邮箱
3. 当系统账户数量超过26人时发送报警邮件

### 1.2 方案

自定义的监控项默认不会自动报警，首页也不会提示错误，需要配置触发器与报警动作才可以自定义报警。

什么是触发器（trigger）？

表达式，如内存不足300M，用户超过30个等

当触发条件发生后，会导致一个触发事件

触发事件会执行某个动作

什么是动作（action）？

动作是触发器的条件被触发后所执行的行为

可以是发送邮件、也可以是重启某个服务等

参考如下操作步骤：

1. 创建触发器并设置标记
2. 设置邮箱
3. 创建Action动作

### 1.3 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：创建触发器规则

##### 1) 创建触发器

创建触发器时强烈建议使用英文的语言环境，通过Configuration--> Templates，找到我们之前创建的count.line.passwd模板，点击模板后面的triggers，如图-1所示。

[Top](#)

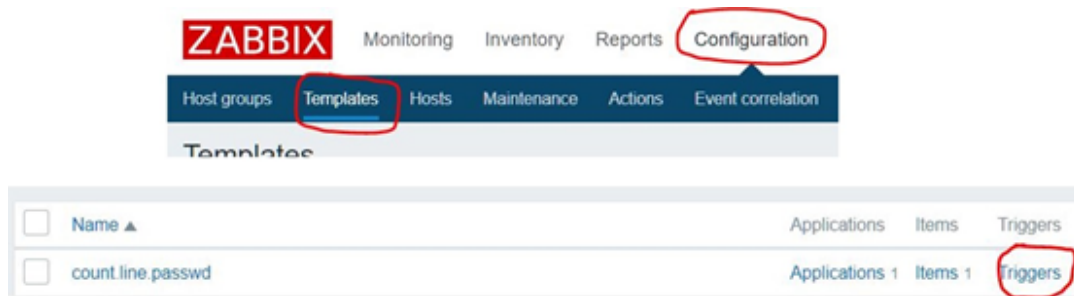


图-1

## 2) 触发器表达式

创建触发器时需要定义表达式，触发器表达式 (Expression) 是触发异常的条件，触发器表达式格式如下：

{<server>:<key>.<function>(<parameter>)}<operator><constant>

{主机: key.函数(参数)}<表达式>常数

在如图-2所示的蓝色方框中编写触发器表达式，可以直接手写，也可以通过add选择表达式模板。

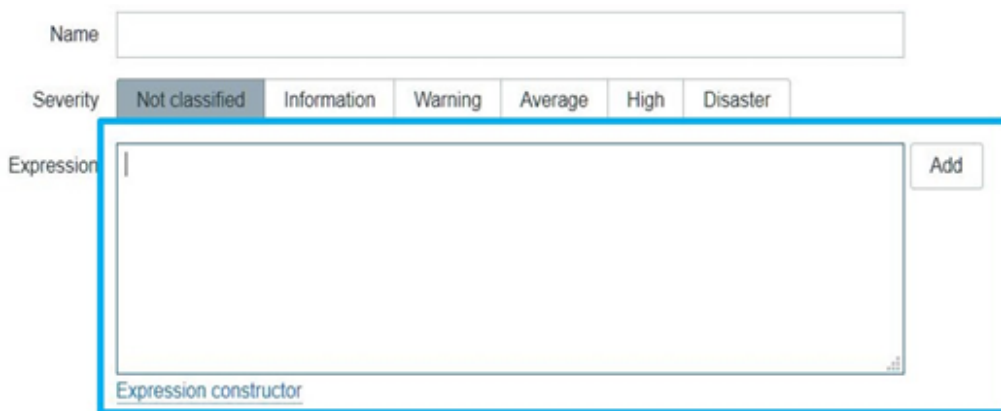


图-2

下面，我们看几个表达式的案例：

{web1:system.cpu.load[all,avg1].last(0)}>5 //0为最新数据

如果web1主机最新的CPU平均负载值大于5，则触发器状态Problem

{vfs.fs.size[/,free].max(5m)}<10G //5m为最近5分钟

根分区，最近5分钟的最大容量小于10G，则状态进入Problem

{vfs.file.cksum[/etc/passwd].diff(0)}>0 //0为最新数据

最新一次校验/etc/passwd如果与上一次有变化，则状态进入Problem

大多数函数使用秒作为参数，可以使用#来表示其他含义（具体参考表-1）。

avg, count, last, min and max 等函数支持额外的第二个参数time\_shift（时间偏移量），这个参数允许从过去一段时间内引用数据。

## 3) 配置触发器

设置触发器名称，如图-3所示，点击add添加表达式，填写表达式：监控项为账户数量，最近300秒账户数量大于26（根据系统账户数量实际填写），效果如图-4所示。

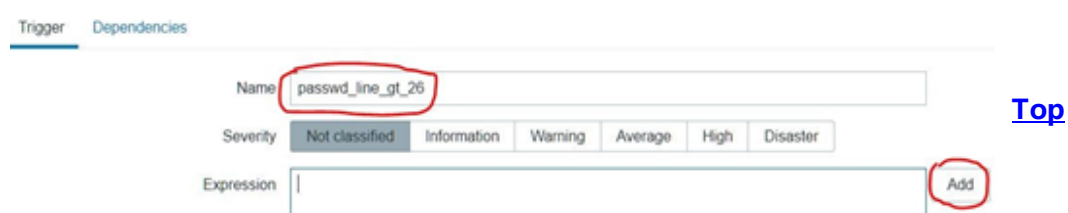


图-3

Item

count.line.passwd: count\_line\_passwd\_item

Select

Function

Last (most recent) T value is > N

Last of (T)

Time

▼

Time shift

300

Time

N

26

Insert

Cancel

图-4

选择触发器报警级别，如图-5所示，Add创建该触发器，如图-6所示。

All templates / count.line.passwd

Applications 1

Items 1

Triggers

Graphs 1

Screens

Discovery rules

W

Trigger

Dependencies

Name

passwd\_line\_gt\_26

Severity

Not classified

Information

Warning

Average

High

Disaster

Expression

{count line passwd count line passwd last(300)}>26

图-5

Enabled

☒

Add

Cancel

图-6

步骤二：设置邮件

1 ) 创建Media

通过Administration ( 管理 ) -->Media Type ( 报警媒体类型 ) -->选择Email ( 邮件 ) ，如图-7所示。

ZABBIX

监测中

资产记录

报表

配置

管理

一般

agent代理程序

认证

用户群组

用户

报警媒介类型

脚本

报警媒介类型

☐

名称 ▲

类型

☐

Email

电子邮件

☐

Jabber

Jabber

☐

SMS

短信

图-7

[Top](#)

设置邮件服务器信息，设置邮件服务器及邮件账户信息，如图-8所示。

报警媒介类型 选项

名称 Email

类型 电子邮件

SMTP服务器 localhost

SMTP服务器端口 25

SMTP HELO company.com

SMTP电邮 root@localhost

安全链接 无 STARTTLS(纯文本通信协议扩展)

认证 无 Username and password

已启用 ☒

更新 克隆 删除 取消

图-8

2)为用户添加Media

在Administration ( 管理 ) -->Users ( 用户 ) 中找到选择admin账户，如图-9所示。

ZABBIX 监测中 资产记录 报表 配置 管理

一般 agent代理程序 认证 用户群组 用户 报警媒介类型 脚本

用户

☐ 别名 ▲ 用户名第一部分

☒ Admin Zabbix

☐ guest

图-9

点击Admin账户后，在弹出的界面中选择Media ( 报警媒介 ) 菜单-->点击Add(添加)报警媒介，如图-10所示。

用户 报警媒介 权限

报警媒介

| 类型    | 收件人            | 当启用时            |
|-------|----------------|-----------------|
| Email | root@localhost | 1-7,00:00-24:00 |

添加

更新 删除 取消

图-10

点击Add ( 添加 ) 后，在Meida Type中填写报警类型，收件人，时间等信息，如图-11所示。

[Top](#)



The screenshot shows the '报警媒介' (Alarm Media) configuration page in Zabbix. The '类型' (Type) is set to 'Email'. The '收件人' (Recipient) is 'root@localhost'. The '当启用时' (When enabled) is '1-7,00:00-24:00'. Under '如果存在严重性则使用' (If severity exists, use), the following severities are checked: '未分类' (Unclassified), '信息' (Information), '警告' (Warning), '一般严重' (General severity), '严重' (Severity), and '灾难' (Disaster). The '已启用' (Enabled) checkbox is checked. There are '添加' (Add) and '取消' (Cancel) buttons at the bottom.

图-11

### 步骤三：创建Action动作

#### 1) Action动作

Action (动作) 是定义当触发器被触发的时候, 执行什么行为。

通过Configuration (配置) --> Actions (动作) --> Create action (创建动作), 如图-12所示。



图-12

#### 2) 配置Action动作的触发条件

填写Action动作的名称, 配置什么触发器被触发时会执行本Action动作 (账户数量大于26), 如图-13所示。



The screenshot shows the 'Create Action' form in Zabbix. The '名称' (Name) is 'Report Problem'. The '计算方式' (Calculation method) is '与/或' (And/Or). The '条件' (Conditions) table shows two conditions: 'A 维护状态 非在 维护' and 'B 触发器 = count.line.passwd.passwd\_line\_gt\_26'. The '新的触发条件' (New trigger condition) section shows a dropdown for '触发器' (Trigger) set to 'count.line.passwd.passwd\_line\_gt\_26'. The '已启用' (Enabled) checkbox is checked. There are '更新' (Update), '克隆' (Clone), '删除' (Delete), and '取消' (Cancel) buttons at the bottom.

图-13

#### 3) 配置Action动作的具体行为

配置动作的具体操作行为 (发送信息或执行远程命令), 无限次数发送邮件, 60秒1次, 发送给Admin用户, 如图-14和图-15所示。

[Top](#)



图-14



图-15

4 ) 测试效果

在被监控主机创建账户（让账户数量大于26），然后登录监控端Web页面，在仪表盘中查看问题报警（需要等待一段时间），如图-16所示。



图-16

[Top](#)

查看报警邮件，在监控服务器上使用mail命令查收报警邮件，如图-17所示。

```

>N 35 root@localhost.local Sat Feb 17 10:15 20/846 "Problem: passwd line gt 26"
N 36 root@localhost.local Sat Feb 17 10:15 21/923 "Problem: /etc/passwd has be
& 35
Message 35:
From root@localhost.localdomain Sat Feb 17 10:15:41 2018
Return-Path: <root@localhost.localdomain>
X-Original-To: root@localhost
Delivered-To: root@localhost.localdomain
From: <root@localhost.localdomain>
To: <root@localhost.localdomain>
Date: Sat, 17 Feb 2018 10:15:41 -0500
Subject: Problem: passwd line gt 26
Content-Type: text/plain; charset="UTF-8"
Status: R

Problem started at 10:13:39 on 2018.02.17
Problem name: passwd_line_gt_26
Host: zabbix_client_web
Severity: Warning

```

图-17

## 2 案例2 : Zabbix自动发现

### 2.1 问题

沿用前面的练习，配置Zabbix的自动发现机制，实现以下目标：

1. 创建自动发现规则
2. 创建自动发现后的动作，添加主机、为主机链接模板

### 2.2 方案

什么是自动发现 ( Discovery ) ？

当Zabbix需要监控的设备越来越多，手动添加监控设备越来越有挑战，此时，可以考虑使用自动发现功能，自动添加被监控主机，实现自动批量添加一组监控主机功能。

自动发现可以实现：

- 自动发现、添加主机，自动添加主机到组；
- 自动连接模板到主机，自动创建监控项目与图形等。

自动发现 ( Discovery ) 流程：

- 创建自动发现规则
- 创建Action动作，说明发现主机后自动执行什么动作
- 通过动作，执行添加主机，链接模板到主机等操作

### 2.3 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：自动发现规则

##### 1 ) 创建自动发现规则

通过Configuration ( 配置 ) --> Discovery ( 自动发现 ) --> Create discovery rule ( 创建发现规则 )，如图-18所示。

[Top](#)





图-18

## 2) 填写规则

填写自动发现的IP范围（逗号隔开可以写多个），多久做一次自动发现（默认为1小时，仅实验修改为1m），如图-19所示。配置检查的方式：Ping、HTTP、FTP、Agent的自定义key等检查，如图-20所示。



图-19



图-20

## 步骤二：创建动作

### 1) 创建Action动作

通过Configuration (配置) --> Actions Event source(事件源)：自动发现(Discovery)-->Create action (创建动作)，如图-21所示。

[Top](#)





图-21

## 2) 配置Action动作具体行为

配置动作，添加动作名称，添加触发动作的条件，如图-22所示。



图-22

点击操作（触发动作后要执行的操作指令），操作细节：添加主机到组，与模板链接（HTTP模板），如图-23所示。

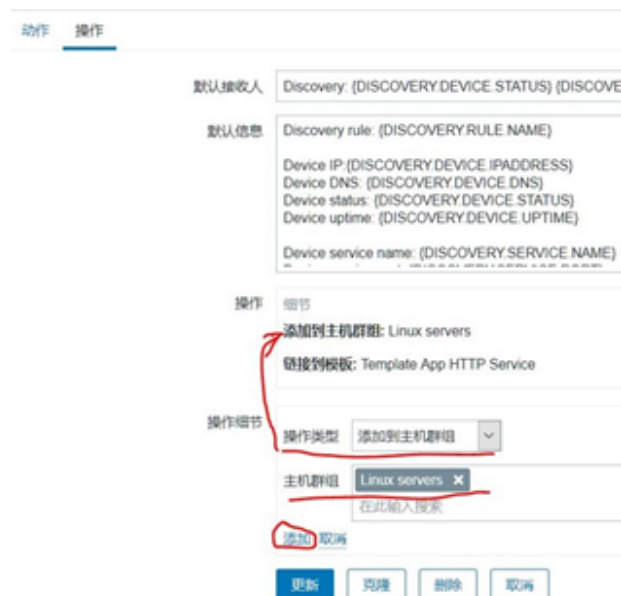


图-23

## 步骤二：添加新的虚拟机

### 1) 创建新的虚拟机（启动HTTP服务器）

[Top](#)

创建一台新的主机，验证zabbix是否可以自动发现该主机，可以重新部署一台新的虚拟机（注意前面的课程，我们已经创建了虚拟机zabbixclient\_web2，并且已经安装部署了Zabbix agent，如果

没有该虚拟机或没有安装Agent，则需要前在zabbixclient\_web2部署Agent)，也可以将旧虚拟机的IP地址，临时修改为其他IP。

## 2) 验证结果

登陆Zabbix服务器的Web页面，查看主机列表，确认新添加的主机是否被自动加入监控主机列表，是否自动绑定了监控模板。

# 3 案例3：Zabbix主动监控

## 3.1 问题

沿用前面的练习，配置Zabbix主动监控，实现以下目标：

1. 修改被监控主机agent为主动监控模式
2. 克隆模板，修改模板为主动监控模板
3. 添加监控主机，并链接主动监控模板

## 3.2 方案

默认zabbix采用的是被动监控，主动和被动都是对被监控端主机而言的！

被动监控：Server向Agent发起连接，发送监控key，Agent接受请求，响应监控数据。

主动监控：Agent向Server发起连接，Agent请求需要检测的监控项目列表，Server响应Agent发送一个items列表，Agent确认收到监控列表，TCP连接完成，会话关闭，Agent开始周期性地收集数据。

区别：Server不用每次需要数据都连接Agent，Agent会自己收集数据并处理数据，Server仅需要保存数据即可。

当监控主机达到一定量级后，Zabbix服务器会越来越慢，此时，可以考虑使用主动监控，释放服务器的压力。

另外，Zabbix也支持分布式监控，也是可以考虑的方案。

## 3.3 步骤

实现此案例需要按照如下步骤进行。

### 步骤一：添加被监控主机

#### 1) 为被监控主机安装部署zabbix agent

注意：前面的实验，我们已经在zabbixclient\_web2主机安装部署了zabbix agent，如果已经完成，则如下操作可以忽略。

```
01. [ root@zabbixclient_web2 ~] # yum -y install gcc pcre-devel
02. [ root@zabbixclient_web2 ~] # tar -xf zabbix-3.4.4.tar.gz
03. [ root@zabbixclient_web2 ~] # cd zabbix-3.4.4/
04. [ root@zabbixclient_web2 ~] # ./configure --enable-agent
05. [ root@zabbixclient_web2 ~] # make && make install
```

#### 2) 修改agent配置文件

将agent监控模式修改为主动模式。

[Top](#)

```

01. [ root@zabbixclient_web2 ~] # vim /usr/local/etc/zabbix_agentd.conf
02. #Server=127.0.0.1,192.168.2.5
03. //注释该行，允许谁监控本机
04. StartAgents=0
05. //被动监控时启动多个进程
06. //设置为0，则禁止被动监控，不启动zabbix_agentd服务
07. ServerActive=192.168.2.5
08. //允许哪些主机监控本机（主动模式），一定要取消127.0.0.1
09. Hostname=zabbixclient_web2
10. //告诉监控服务器，是谁发的数据信息
11. //一定要和zabbix服务器配置的监控主机名称一致（后面设置）
12. RefreshActiveChecks=120
13. //默认120秒检测一次
14. UnsafeUserParameters=1
15. //允许自定义key
16. Include=/usr/local/etc/zabbix_agentd.conf.d/
17. [ root@zabbixclient_web2 ~] # killall zabbix_agentd //关闭服务
18. [ root@zabbixclient_web2 ~] # zabbix_agentd //启动服务

```

## 步骤二：创建主动监控的监控模板

### 1) 克隆Zabbix自动的监控模板

为了方便，克隆系统自带模板（在此基础上就该更方便）。

通过Configuration（配置）-->Templates（模板）-->选择Template OS Linux  
-->全克隆，克隆该模板，新建一个新的模板。如图-24所示。

新模板名称为：Template OS Linux ServerActive。



图-24

### 2) 修改模板中的监控项目的监控模式

将模板中的所有监控项目全部修改为主动监控模式，通过Configuration（配置）-->Templates（模板）-->选择新克隆的模板，点击后面的Items（监控项）-->点击全选，选择所有监控项目，点击批量更新，将类型修改为：Zabbix Agent（Active主动模式），如图-25所示。

[Top](#)



图-25

### 3) 禁用部分监控项目

批量修改监控项的监控模式后，并非所有监控项目都支持主动模式，批量修改后，会发现有几个没有修改主动模式成功，说明，这些监控项目不支持主动模式，关闭即可。

可以点击类型排序，方便操作，点击状态即可关闭。如图-26所示。

| 触发器   | 键值             | 间隔 | 历史记录 | 趋势   | 类型 ▲           | 应用集          | 状态  |
|-------|----------------|----|------|------|----------------|--------------|-----|
| 触发器 1 | agent.version  | 1h | 1w   |      | Zabbix 客户端     | Zabbix agent | 停用的 |
| 触发器 1 | agent.hostname | 1h | 1w   |      | Zabbix 客户端     | Zabbix agent | 停用的 |
| 触发器 1 | agent.ping     | 1m | 1w   | 365d | Zabbix 客户端     | Zabbix agent | 停用的 |
| 触发器 1 | kernel.maxproc | 1h | 1w   | 365d | Zabbix客户端(主动式) | OS           | 已启用 |

图-26

## 步骤三：添加监控主机

### 1) 手动添加监控主机（主动模式监控）

在Zabbix监控服务器，添加被监控的主机（主动模式），设置主机名称：zabbixclient\_web2（必须与被监控端的配置文件Hostname一致），将主机添加到Linux servers组，IP地址修改为0.0.0.0，端口设置为0，如图-27和图-28所示。



图-27

[Top](#)

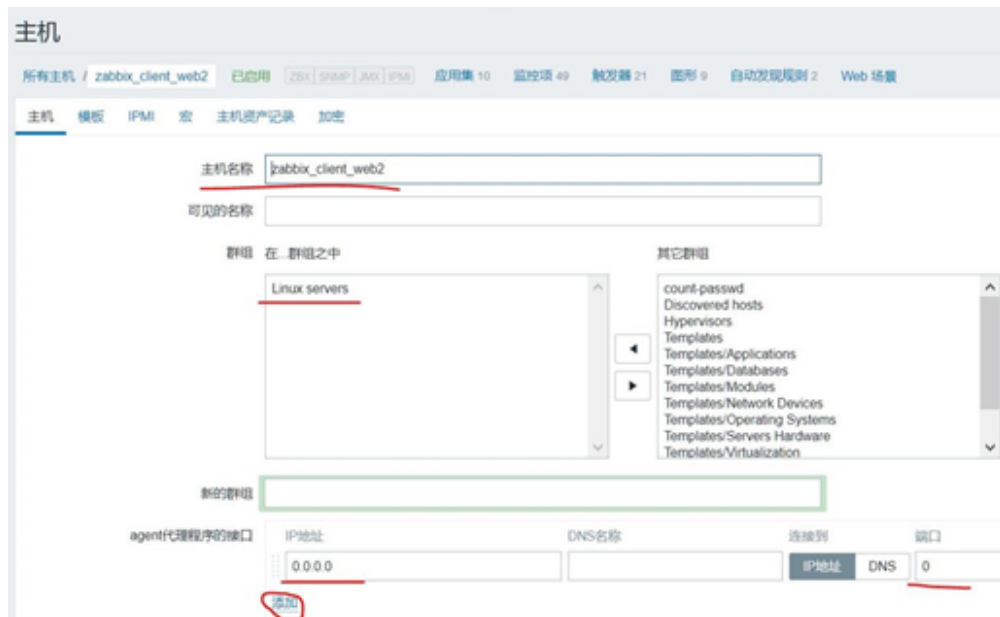


图-28

为主机添加监控模板，选择刚刚创建的模板（主动模式），添加链接模板到主机，如图-29所示。

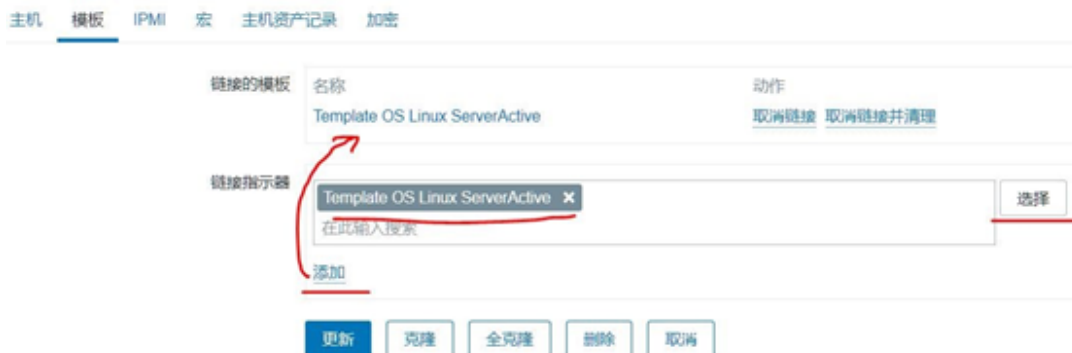


图-29

## 2) 验证监控效果

查看数据图表，通过Monitoring-->Graphs菜单，选择需要查看的主机组、主机以及图形，查看效果，如图-30所示。

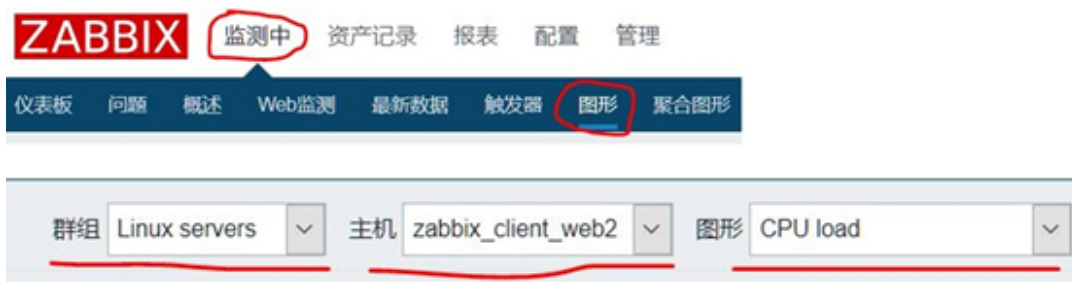


图-30

CPU、内存等其他数据可用正常获取，但是，查看分区图表时并无数据，因为分区数据采用的是自动发现监控，与普通监控项一样，修改为主动模式即可，选择Template OS Linux ServerActive模板，修改Discovery自动发现为主动模式。如图-31所示。

[Top](#)



图-31

## 4 案例4：拓扑图与聚合图形

### 4.1 问题

沿用前面的练习，熟悉zabbix拓扑图与聚合图形，实现以下目标：

1. 创建修改拓扑图
2. 创建聚合图形

### 4.2 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：创建拓扑图

##### 1) 创建拓扑

绘制拓扑图可以快速了解服务器架构，通过Monitoring（监控中）-->Maps（拓扑图），选择默认的Local network拓扑图，编辑即可（也可以新建一个拓扑图），如图-32所示。



图-32

##### 2) 拓扑图图表说明

- Icon（图标），添加新的设备后可以点击图标修改属性
- Shape（形状）
- Link（连线），先选择两个图标，再选择连线
- 完成后，点击Update（更新）

创建完拓扑图，效果如图-33所示。

[Top](#)



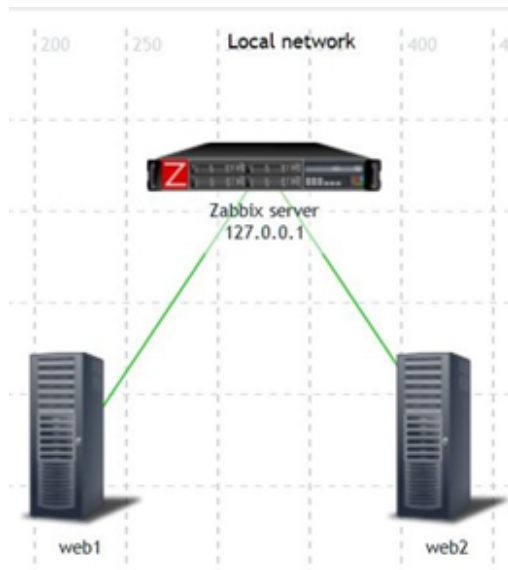


图-33

## 步骤二：创建聚合图形

### 1) 创建聚合图形

聚合图形可以在一个页面显示多个数据图表，方便了解多组数据。

通过Monitoring (监控中) --> Screens (聚合图形) --> Create screen(创建聚合图形)即可创建聚合图形，如图-34所示。



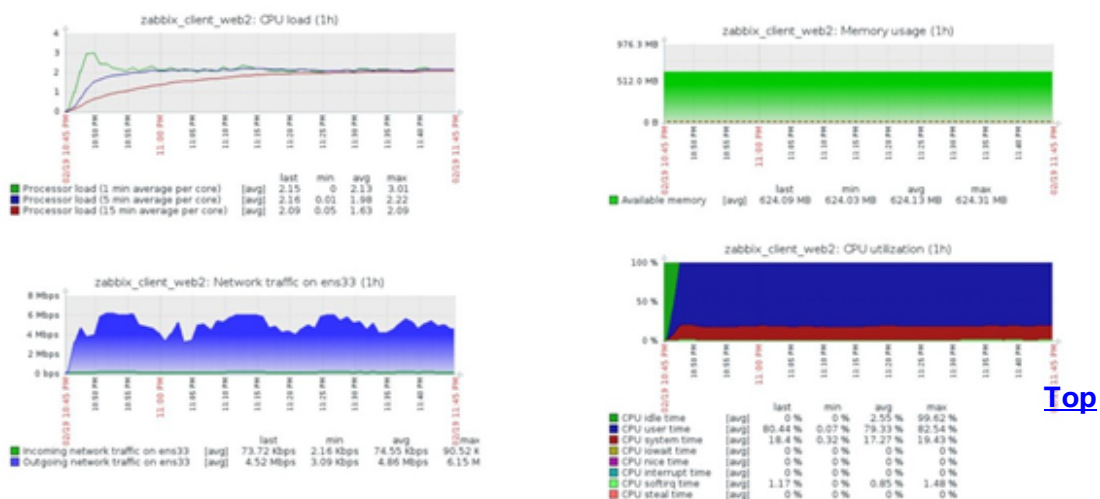
图-34

修改聚合图形参数如下：

- Owner：使用默认的Admin用户
- Name：名称设置为zabbixclient\_web2\_host
- Columns：列数设置为2列
- Rows：行数设置为4行

### 2) 为聚合图形中添加监控图形

选择刚刚创建的聚合图形 (zabbixclient\_web2\_host)，点击后面的构造函数 (constructor)，点击Change(更改)，设置每行每列需要显示的数据图表，如图-35所示。



Top



## 5 案例5：自定义监控案例

### 5.1 问题

沿用前面的练习，使用自定义key监控常用监控项目，实现以下目标：

1. 监控Nginx状态
2. 监控网络连接状态

### 5.2 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：监控Nginx服务状态

1) 准备环境，部署nginx软件

安装nginx软件，开启status模块

```
01. [ root@zabbixclient_web1 nginx-1.12.2 ] # ./configure \
02. > --with-http_stub_status_module
03. [ root@zabbixclient_web1 nginx-1.12.2 ] # make && make install
04. [ root@zabbixclient_web1 ~ ] # cat /usr/local/nginx/conf/nginx.conf
05. ... ..
06.     location /status {
07.         stub_status on;
08.     }
09. ... ..
10. [ root@zabbixclient_web1 ~ ] # curl http://192.168.2.100/status
11. Active connections: 1
12. server accepts handled requests
13. 10 10 3
14. Reading: 0 Writing: 1 Waiting: 0
```

2) 自定义监控key

语法格式：

UserParameter=key,command

UserParameter=key[\*],<command>

key里的所有参数，都会传递给后面命令的位置变量

如：

UserParameter=ping[\*],echo \$1

ping[0]，返回的结果都是0

ping[aaa]，返回的结果都是aaa

注意：被监控端修改配置文件，注意要允许自定义key并设置Include！

[Top](#)

## 创建自定义key

01. [ root@zabbixclient\_web1 ~] # vim /usr/local/etc/zabbix\_agentd.conf.d/nginx.status
02. UserParameter=nginx.status[\*],/usr/local/bin/nginx\_status.sh \$1
03. [ root@zabbixclient\_web1 ~] # killall zabbix\_agentd
04. [ root@zabbixclient\_web1 ~] # zabbix\_agentd

## 自定义监控脚本（仅供参考，未检测完整状态）

01. [ root@zabbixclient\_web1 ~] # vim /usr/local/bin/nginx\_status.sh
02. #!/bin/bash
03. case \$1 in
04. active)
05. curl -s http://192.168.2.100/status | awk '/Active/{ print \$NF} ';;
06. waiting)
07. curl -s http://192.168.2.100/status | awk '/Waiting/{ print \$NF} ';;
08. accepts)
09. curl -s http://192.168.2.100/status | awk 'NR==3{ print \$2} ';;
10. esac
11. [ root@zabbixclient\_web1 ~] # chmod +x /usr/local/bin/nginx\_status.sh

## 测试效果：

01. [ root@zabbixclient\_web1 ~] # zabbix\_get -s 127.0.0.1 \
02. -k 'nginx.status[accepts]'

登陆Zabbix监控Web，创建监控项目item，点击Configuration（配置）-->Hosts(主机)，点击主机后面的items（项目），点击Create item（创建项目）。修改项目参数如图-36所示。

Name: nginx\_status

Type: Zabbix agent

Key: nginx.status[accepts]

Host interface: 192.168.2.100: 10050

Type of information: Numeric (unsigned)

Units:

[Top](#)

图-36

## 步骤二：监控网络连接状态

### 1) 了解TCP协议

熟悉TCP三次握手，参考图-37。

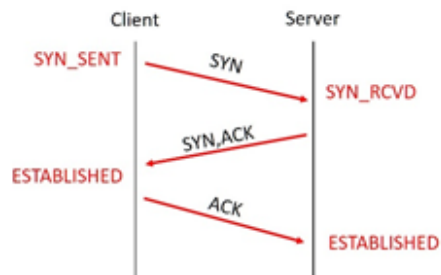


图-37

熟悉TCP连接的四次断开，参考图-38。

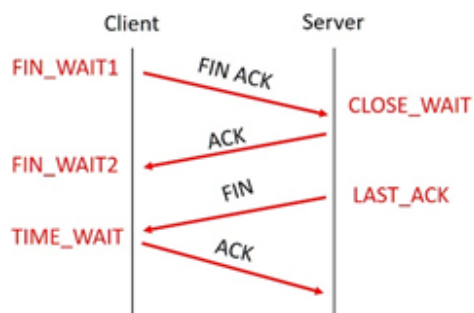


图-38

### 2) 查看网络连接状态

模拟多人并发连接

```
01. [ root@zabbixclient_web1 ~] # ab -c 1000 -n 100000 http://192.168.2.100/
```

查看网络连接状态，仔细观察、分析第二列的数据

```

01. [ root@zabbixclient_web1 ~] # ss - antup
02. //- a显示所有
03. //- t显示TCP连接状态
04. //- u显示UDP连接状态
05. //- n以数字形式显示端口号和IP地址
06. //- p显示连接对应的进程名称
  
```

### 3) 创建自定义key

注意：被监控端修改配置文件，注意要允许自定义key并设置Include。

[Top](#)

```

01. [ root@zabbixclient_web1 ~] # vim /usr/local/etc/zabbix_agentd.conf.d/net.status
02. UserParameter=net.status[*],/usr/local/bin/net_status.sh $1
03.
04. [ root@zabbixclient_web1 ~] # killall zabbix_agentd
05. [ root@zabbixclient_web1 ~] # zabbix_agentd

```

自定义监控脚本（仅供参考，未检测完整状态）

```

01. [ root@zabbixclient_web1 ~] # vim /usr/local/bin/net_status.sh
02. #!/bin/bash
03. case $1 in
04.     estab)
05.         ss - antp | awk '/^ESTAB/{ x++ } END{ print x }';;
06.     close_wait)
07.         ss - antp | awk '/^CLOSE-WAIT/{ x++ } END{ print x }';;
08.     time_wait)
09.         ss - antp | awk '/^TIME-WAIT/{ x++ } END{ print x }';;
10.     esac
11. [ root@zabbixclient_web1 ~] # chmod +x /usr/local/bin/net_status.sh

```

测试效果：

```

01. [ root@zabbixclient_web1 ~] # zabbix_get -s 127.0.0.1 \
02. -k 'net.status[time_wait]'

```

#### 4) 监控netstatus

在监控服务器，添加监控项目item，Configuration-->Hosts点击主机后面的items  
点击Create item，如图-39所示。



图-39

[Top](#)

[Top](#)