[Edit](http://maxiang.info/#/?provider=evernote&guid=151031e6-e207-4270-a571-6cfd9977457d&notebook=linux)

# zabbix

linux zabbix

* [zabbix](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#zabbix)
* [为什么需要监控，及监控内容](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#为什么需要监控及监控内容)
* [安装zabbix](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#安装zabbix)
* [数据库的备份](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#数据库的备份)
* [客户端的安装](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#客户端的安装)
* [增加主机](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#增加主机)
* [web监控](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#web监控)
* [自定义监控](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#自定义监控)
* [监控TCP状态](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#监控tcp状态)
* [Items](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#items)
* [Key](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#key)
* [使用脚本发邮件报警](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#使用脚本发邮件报警)
* [增加一个模块](https://app.yinxiang.com/note/151031e6-e207-4270-a571-6cfd9977457d?usn=6346#增加一个模块)

## 为什么需要监控，及监控内容

在IT环境中会存在各种各样的设备，硬件设备，软件设备。都是由很多模型构成（底层支持(网络环境，硬件)，系统架构(centos,kvm,xen,lvs,keepalived,haproxy)，上层应用(mysql,nginx,php,tomcat)

为了业务的正常运行，需要运维进行设计，架构维护和调优，在这个过程中，为了及时掌握基础环境和业务应用系统的可用性，需要获取各个组件的运行状态，这样便离不开监控系统

一般来说监控系统可以分为两部分：数据采集部分（客户端）和数据存储分析告警展示部分（服务端）

1. 硬件监控，
2. 运程控制卡
3. IPMI, DELL:IDRAC. HP ILO IBM IMM   
   ipmi <http://www.ibm.com/developerworks/cn/linux/l-ipmi/>   
   megacli 监控磁盘，做rede,然后监控redi的状态   
   硬件监控有：端口流量，端口速率；系统方面CPU使用率，内存使用率，连接数，请求数，系统在线时间；

邮件

1. 系统监控级别

* CPU: top,uptime 。w, cat /proc/cpuinfo,lscpu,cat /Nproc/loadavg, vimstat, mpstat 1,   
  load average: 0.01, 0.07, 0.03, 第一个平均值不超过CPU核数为佳
* 内存： free -m, 剩余1G报警
* 磁盘： df -h，剩余空间20%
* IO: iotop
* 网络 iftop
* 端口监控netstat

[root@buff ~]# ulimit -n

1024

[root@buff ~]# ulimit -a

core file size (blocks, -c) 0

data seg size (kbytes, -d) unlimited

scheduling priority (-e) 0

file size (blocks, -f) unlimited

pending signals (-i) 1781

max locked memory (kbytes, -l) 64

max memory size (kbytes, -m) unlimited

open files (-n) 1024

pipe size (512 bytes, -p) 8

POSIX message queues (bytes, -q) 819200

real-time priority (-r) 0

stack size (kbytes, -s) 10240

cpu time (seconds, -t) unlimited

max user processes (-u) 1781

virtual memory (kbytes, -v) unlimited

file locks (-x) unlimited

## 安装zabbix

zabbix-server 的数据库大小取决于NVPS(number of processed values per second)总体上NVPS反映了处理速度，与监控项的数目，监控的类型，取值间隔，history的保留时间和tends的保留时间有直接关系

假设有60000个监控项，刷新周期为60秒，那么每秒处理的数据为60000/60=1000,保存周期为90天，那么历史数据大小=(90\*24\*3600)\*1000\*50B 约为362G

其中50byetes为历史数据所占大小，

趋势数据为128b

事件数据为130

总占用大小=configuration + history + trends + envents

来点干货，直接开始安装zabbix

$ rpm -ivh <http://repo.zabbix.com/zabbix/>2.4/rhel/6/x86\_64/zabbix-release-2.4-1.el6.noarch.rpm

$ yum -y install zabbix-server-mysql zabbix-web-mysql mysql-server mysql wqy-microhei-fonts zabbix-get zabbix-agent

$ \cp /usr/share/mysql/my-medium.cnf /etc/my.cnf

$ sed -i "38a collation-server = utf8\_general\_ci\ninit\_connect = \'SET NAMES utf8\'\ncharacter-set-server = utf8" /etc/my.cnf

$ /etc/init.d/mysqld start

#在客户端只需要安装zabbix-agent

$ yum install -y zabbix zabbix-agent

修改vim /etc/my.cnf 在[mysqld]下增加

[mysqld]

collation-server = utf8\_general\_ci

character-set-server = utf8

init-connect = ‘SET NAMES utf8’

这主要是为了防止中文字符集乱码

#启动mysql

$ chkconfig mysqld on

$ service mysqld start

$ mysqladmin -uroot password sandow #修改密码

为zabbix创建数据库。

**create** **database** zabbix;

**grant** all **on** zabbix.\* **to** zabbix@'localhost' **identified** **by** 'zabbix';

**flush** **privileges**;

然后把数据导入到

$ **cd** /usr/share/doc/zabbix-server-mysql-2.4.7/create

$ mysql -uroot zabbix<schema.sql #导入数据结构

$ mysql -uroot zabbix<images.sql #导入图片

$ mysql -uroot zabbix<data.sql #导入数据，顺序不能变

#另一种方法

mysql> **source** /usr/share/doc/zabbix-server-mysql-2.4.7/create/schema.sql

mysql> **source** /usr/share/doc/zabbix-server-mysql-2.4.7/create/images.sql

mysql> **source** /usr/share/doc/zabbix-server-mysql-2.4.7/create/data.sql

#接下来配置zabbix

$ sed -i '107aDBPassword=zabbix' /etc/zabbix/zabbix\_server.conf

$ grep '^[0-Z]' /etc/zabbix/zabbix\_server.conf

LogFile=/var/**log**/zabbix/zabbix\_server.log

LogFileSize=0

PidFile=/var/run/zabbix/zabbix\_server.pid

DBName=zabbix

DBUser=zabbix

DBPassword=zabbix

DBSocket=/var/lib/mysql/mysql.sock

SNMPTrapperFile=/var/**log**/snmptt/snmptt.log

AlertScriptsPath=/usr/lib/zabbix/alertscripts

ExternalScripts=/usr/lib/zabbix/externalscripts

数据分离需要把socket注释掉然后把port开启

配置时区不然地安装的时候会报错的

$ vim /etc/httpd/conf.d/zabbix.conf

php\_value date.timezone Asia/Shanghai

#使用sed修改

sed -i '19c\\tphp\_value date.timezone Asia/Shanghai' /etc/httpd/conf.d/zabbix.conf

见证奇迹的时刻到了

$ /etc/init.d/httpd start

$ /etc/init.d/zabbix-server start

$ netstat -lntup|egrep '80|10051|3306'

tcp 0 0 0.0.0.0:10051 0.0.0.0:\* LISTEN 121786/zabbix\_serve

tcp 0 0 0.0.0.0:3306 0.0.0.0:\* LISTEN 85624/mysqld

tcp 0 0 :::10051 :::\* LISTEN 121786/zabbix\_serve

tcp 0 0 :::80 :::\* LISTEN 71356/httpd

#开机启动

$ chkconfig zabbix-server on

$ chkconfig httpd on

$ chkconfig zabbix-agent o

# 防火墙配置10050是agent端口，agent采用被动方式，server主动连接agent的10050端口，10051为server端口，agent采用主动或trapper方式连接10051端口，可以选择关闭防火墙

[root@Xmanager create]# cat /etc/sysconfig/iptables

# Firewall configuration written by system-config-firewall

# Manual customization of this file is not recommended.

\*filter

:INPUT ACCEPT [0:0]

:FORWARD ACCEPT [0:0]

:OUTPUT ACCEPT [0:0]

-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

-A INPUT -p icmp -j ACCEPT

-A INPUT -i lo -j ACCEPT

-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT

-A INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT

-A INPUT -m state --state NEW -m tcp -p tcp --dport 10051 -j ACCEPT

-A INPUT -m state --state NEW -m tcp -p tcp --dport 10050 -j ACCEPT

-A OUTPUT -m state --state NEW -m tcp -p tcp --dport 10050 -j ACCEPT

#-A FORWARD -j REJECT --reject-with icmp-host-prohibited

#-A INPUT -j REJECT --reject-with icmp-host-prohibited

COMMIT

# 修改/etc/php.ini

$ sed -i '946c\date.timezone = Asiz/Shanghai' /etc/php.ini

#可以仅改上面的，下面的如有需要再改其实在/etc/httpd/conf.d/zabbix.con里就有了，所以其实php.ini可以不修改

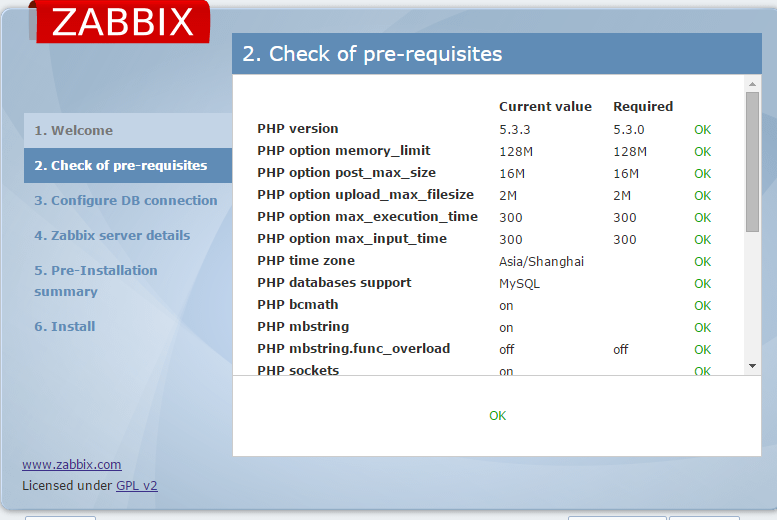
$ sed -i '440s/30/300/; 729s/8M/16M/; 132c\max\_input\_time=300; 1586c\mbstring.func\_overload = 2' /etc/php.ini

现在就可以在浏览器上输入网址进入zabbix服务端啦，

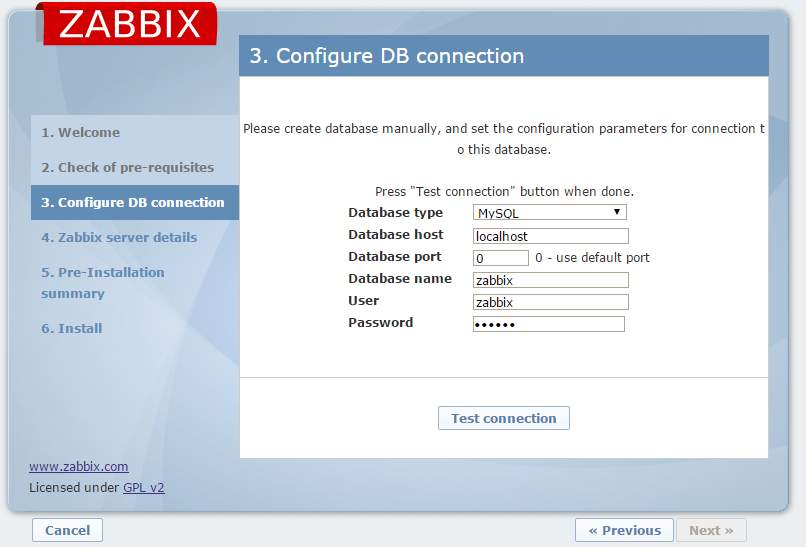
$ 10.0.0.151/zabbix #第一次会配置mysql不过这个也很简单的下一步下一步啦

#用户名Admin密码zabbix登陆页面。

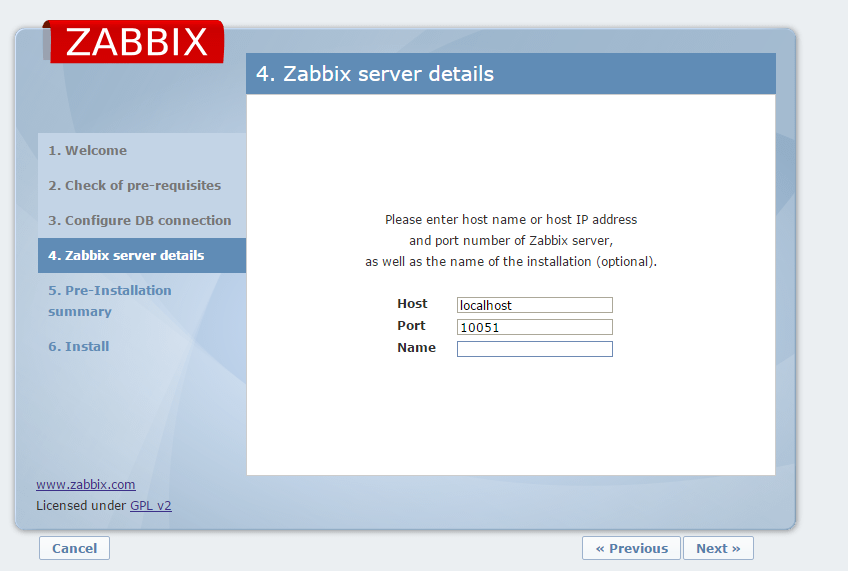
要确定在checkof pre-requistes里没有faile，然后下一步



选择数据库，这里选择mysql密码上面输上面授权的用户及密码



先test conneciton, OK后选下一步



这里的name可以随便起，然后在后面进放页面后会在右上角显示,然后下一步，下一步变OK。进入登录界面



配置后如果数据库有错可以修改/etc/zabbix/web/zabbix.conf.php在里面修改数据库的IP，database名，用户名，及密码。

若数据库找不到可以在/etc/zabbix/zabbix\_server.conf中设置DBSocket=/tmp/mysql.sock此为实际mysql的位置

$ cat /etc/zabbix/web/zabbix.conf.php

<?php

// Zabbix GUI configuration file.

global $DB;

$DB['TYPE'] = 'MYSQL';

$DB['SERVER'] = 'localhost';

$DB['PORT'] = '0';

$DB['DATABASE'] = 'zabbix';

$DB['USER'] = 'zabbix';

$DB['PASSWORD'] = 'zabbix';

// Schema name. Used **for** IBM DB2 and PostgreSQL.

$DB['SCHEMA'] = '';

$ZBX\_SERVER = 'localhost';

$ZBX\_SERVER\_PORT = '10051';

$ZBX\_SERVER\_NAME = 'sandow\_zabbix\_contral';

$IMAGE\_FORMAT\_DEFAULT = IMAGE\_FORMAT\_PNG;

?>

**configuration** 主要用于对zabbix需要监控的主机的管理

* **host groups** 增加组，用于对主机分组
* **Templates** 棋版，下面就用导入一个模版进去在右边search下面的import
* **Hosts** 主机，在这里可以添加需要监控的主机，这一项使用的次数非常多
* applications 对intems的一个分组
* Items 具体的监控项
* Triggers 阈值，达到一定状态就触发报警发邮件或者播放声音(需要在profile–messagin里开启声音)
* graphs 对items使用二维图显示，然后可以增加到Screen里
* **Screen** 把多个信息放在一起展示，

**Administartion** 主要主于对zabbix用户及用户组管理

* Users 创建用户组与用户
* media types 发送邮件配置

### 数据库的备份

如果忘记了zabbix登录密码，需要进入mysql更新

$ mysql -uroot -psandow

mysql> SELECT \* FROM zabbix.users WHERE **alias**='Admin'\G

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

userid: 1

**alias**: Admin

name: Zabbix

surname: Administrator

passwd: \*DEEF4D7D88CD046ECA02A80393B7780

url:

autologin: 1

autologout: 0

lang: en\_GB

refresh: 30

**type**: 3

theme: default

attempt\_failed: 0

attempt\_ip:

attempt\_clock: 0

rows\_per\_page: 50

1 row **in** **set** (0.00 sec)

mysql> UPDATE zabbix.users SET passwd=md5('zabbix') WHERE **alias**='Admin';

Query OK, 1 row affected (0.01 sec)

Rows matched: 1 Changed: 1 Warnings: 0

mysql> FLUSH PRIVILEGES;

Query OK, 0 rows affected (0.00 sec)

zabbix 数据库有些表是不需要备份的，仅需要备份一些特定的表就OK了，下面是脚本

#!/bin/bash

#author: itnihao

#mail: [itnihao@qq.com](mailto:itnihao@qq.com)

#[http://wwww.itnihao.com](http://wwww.itnihao.com/)

#<https://github.com/itnihao/zabbix-book/blob/master/03-chapter/Zabbix_MySQLdump_per_table.sh>

#chmod 700 ${PATH}/Zabbix\_MySQLdump\_per\_table\_v2.sh

#crontab -e (0 3 \* \* \* ${PATH}/Zabbix\_MySQLdump\_per\_table\_v2.sh)

red='\e[0;31m' # 红色

RED='\e[1;31m'

green='\e[0;32m' # 绿色

GREEN='\e[1;32m'

blue='\e[0;34m' # 蓝色

BLUE='\e[1;34m'

purple='\e[0;35m' # 紫色

PURPLE='\e[1;35m'

NC='\e[0m' # 没有颜色

**source** /etc/bashrc

**source** /etc/profile

MySQL\_USER=zabbix

MySQL\_PASSWORD=zabbix

MySQL\_HOST=localhost

MySQL\_PORT=3306

MySQL\_DUMP\_PATH=/mysql\_backup

MYSQL\_BIN\_PATH=/usr/bin/mysql

MYSQL\_DUMP\_BIN\_PATH=/usr/bin/mysqldump

MySQL\_DATABASE\_NAME=zabbix

DATE=$(date '+%Y-%m-%d')

**MySQLDUMP** () {

[ -d ${MySQL\_DUMP\_PATH} ] || mkdir ${MySQL\_DUMP\_PATH}

**cd** ${MySQL\_DUMP\_PATH}

[ -d logs ] || mkdir logs

[ -d ${DATE} ] || mkdir ${DATE}

**cd** ${DATE}

TABLE\_NAME\_ALL=$(${MYSQL\_BIN\_PATH} -u${MySQL\_USER} -p${MySQL\_PASSWORD} -h${MySQL\_HOST} ${MySQL\_DATABASE\_NAME} -e \

"show tables"|egrep -v "(Tables\_in\_zabbix|history\*|trends\*|acknowledges|alerts|auditlog|events|service\_alarms)")

**for** TABLE\_NAME **in** ${TABLE\_NAME\_ALL}

**do**

${MYSQL\_DUMP\_BIN\_PATH} --opt -u${MySQL\_USER} -p${MySQL\_PASSWORD} -P${MySQL\_PORT} -h${MySQL\_HOST} \

${MySQL\_DATABASE\_NAME} ${TABLE\_NAME} >${TABLE\_NAME}.sql

sleep 0.01

**done**

[ "$?" == 0 ] && **echo** "${DATE}: Backup zabbix succeed" >> ${MySQL\_DUMP\_PATH}/logs/ZabbixMysqlDump.log

[ "$?" != 0 ] && **echo** "${DATE}: Backup zabbix not succeed" >> ${MySQL\_DUMP\_PATH}/logs/ZabbixMysqlDump.log

**cd** ${MySQL\_DUMP\_PATH}/

[ "$?" == 0 ] && rm -rf $(date +%Y-%m-%d --date='5 days ago')

**exit** 0

}

**MySQLImport** () {

**cd** ${MySQL\_DUMP\_PATH}

DATE=$(ls ${MySQL\_DUMP\_PATH} |egrep "\b^[0-9]+-[0-9]+-[0-9]+$\b")

**echo** -e "${green}${DATE}"

**echo** -e "${blue}what DATE do you want to import,please input date:${NC}"

**read** SELECT\_DATE

**if** [ -d "${SELECT\_DATE}" ];**then**

**echo** -e "you select is ${green}${SELECT\_DATE}${NC}, do you want to contine,if,input ${red}(yes|y|Y)${NC},\

else then exit"

**read** Input

[[ 'yes|y|Y' =~ "${Input}" ]]

status="$?"

**if** [ "${status}" == "0" ];**then**

**echo** "now import SQL....... Please wait......."

**else**

**exit** 1

**fi**

**cd** ${SELECT\_DATE}

**for** PER\_TABEL\_SQL **in** $(ls \*.sql)

**do**

${MYSQL\_BIN\_PATH} -u${MySQL\_USER} -p${MySQL\_PASSWORD} -h${MySQL\_HOST} ${MySQL\_DATABASE\_NAME} < ${PER\_TABEL\_SQL}

**echo** -e "import ${PER\_TABEL\_SQL} ${PURPLE}........................${NC}"

**done**

**echo** "Finish import SQL,Please check Zabbix database"

**else**

**echo** "Don't exist ${SELECT\_DATE} DIR"

**fi**

}

**case** "$1" **in**

MySQLDUMP|mysqldump)

MySQLDUMP

;;

MySQLImport|mysqlimport)

MySQLImport

;;

\*)

**echo** "Usage: $0 {(MySQLDUMP|mysqldump) (MySQLImport|mysqlimport)}"

;;

**esac**

## 客户端的安装

$ yum -y install zabbix-agent

$ grep '^[0-Z]' /etc/zabbix/zabbix\_agentd.conf

PidFile=/var/run/zabbix/zabbix\_agentd.pid

LogFile=/var/**log**/zabbix/zabbix\_agentd.log

LogFileSize=0

Server=172.16.1.151 #被动模式下，允许哪台服务器连接agent，多IP用","隔开，server主机不需要修改

ServerActive=127.0.0.1 # 主动模式下，向那台服务器传送数据

Hostname=Zabbix server

Include=/etc/zabbix/zabbix\_agentd.d/

$ /etc/init.d/zabbix-agent start

$ netstat -tunlp|grep 10050

如果需要使用中文版的web-zabbix需要要修改下字体来解决中文乱码问题

yum -y install wqy-microhei-fonts

ll /etc/alternatives/zabbix-web-font

# 删除软链接

rm -f /etc/alternatives/zabbix-web-font

ln -s /usr/share/fonts/wqy-microhei/wqy-microhei.ttc /etc/alternatives/zabbix-web-font

### 增加主机

zabbix完整监控配置流程描述为 host groups >>> hosts >>> applications >>> items >>> triggers >>> event >>> actions >>> user groups >>> user >>> medias >>> audit

1.host groups 用于增加主机组， 分组的目的是将同一属性的主机归类，对于同一属性的主机或者模板，尽量归到分组，方便以后管理（分组的方式可以按：地理位置，业务，机器用途，系统版本，应用程序，等分）。在configuration大类下面，增加分组很简单，create host group 在group name里写入组名， hosts in 是选择那些设备属于这个新添加的组，save就OK。在mysql中可以使用命令SELECT \* FROM zabbix.groups;查看

2.templates 模板的添加，监控项，触发器，图形，web, discovery,等都是存在于主机之上的，由于多个主机都会用相同的监控配置，因此可以对这部分同类的数据进行归纳抽象，即将这些数据做成模板，当我们需要对其他监控数据进行配置的时候，只需要对相应的主机添加对应的模板即可。**cofiguration >>> templates >>> create template**,也可以导入模板，这个后面细讲

1. hosts 增加主机,添加主机的目的是对具体设备进行监控，

|  |  |
| --- | --- |
| **参数** | **描述** |
| 主机名(hostname) | 一个不重复的主机名,编辑该名称对应配置文件时，主机史必须与 此输入值相同 |
| 访问名(visible name) | 如果设置此名字，那么它将出现在主机列表，地图等地方 |
| 群组(groups in groups) | 选择所属群组，一个主机至少有一个群组 |
| 接口协议(agetn interfaces) | 一个主机的主机接口协议类型有agent, snmp, jmx,jpmi使用ADD增加对应协议的IP/DNS,连接项及端口 |
| 通过代理服务器进行监控(monitored by proxy) | 主机可能通过zabbix服务器或zabbix代理去监控客户端 |

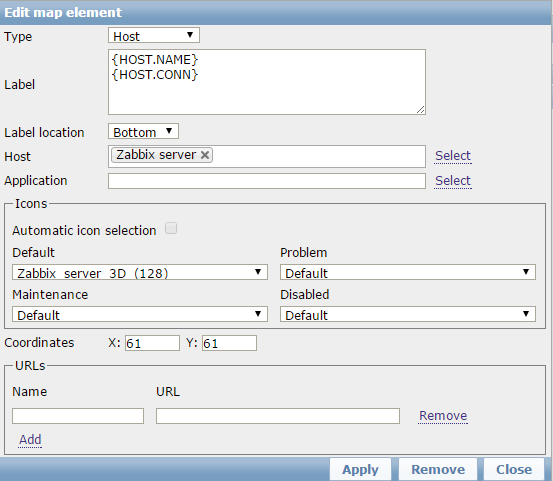
1. graphs 配置，graphs是将数据展示为图像，可以在主机中点击graphs来增加,

* graphs type 有normal常规图，stacked 叠图，pie 饼图，exploded，分解饼图.
* show legend 显示图表说明
* show working time 非工作时间用灰色前景显示，不用于饼图分解图
* show triggers 触发阈值便会红色线条显示
* Y axi min/max value Y轴最小/大值（calcuated-Y自动计算， Fixed-Y修正最大/小值，Item-items 最后一次获取的数值做为最大/小值

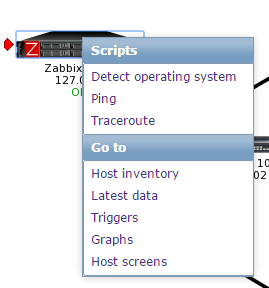


1. screens ，把多个图放在一起显示，可以展示多个host的多个信息，可以展示graphs, maps, server info. 配置好后可以在**Monitoring >>> screens**里显示点右上角加号可以在dashboard里看到，另外可以点击**slide shows**来幻灯片形式滚动，另外有一个图形化插件**Zatree**可以试着使用一下
2. Map, map的作用是将各种设备用网络拓扑图的方式展示，

* label 便是宏，可以参数官网
* 可以在administration-general里增加背景图片，然后在map中更改背景图片，在这里创建一个map然后在monitoring里查看

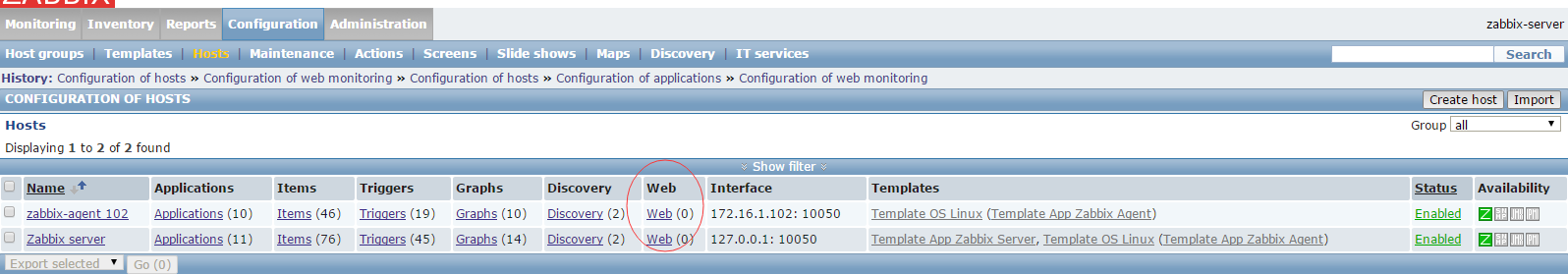


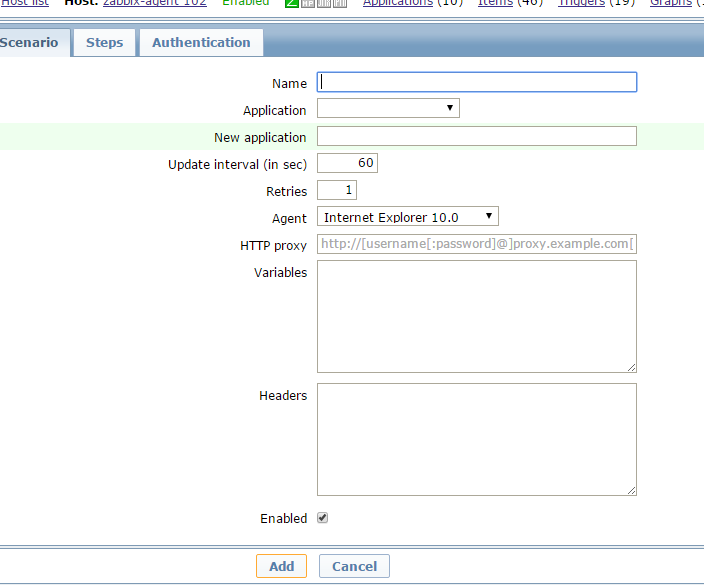
在monitoring - maps下可以看到刚才增加的主机，然后选中其中一台便可以执行下面操作



### web监控

web监控主要监控http 状态码（多针对40X，50X），响应速度（服务器性能，重复加载，网络因素，压缩），下载速度。在hosts 主机上点击 web >> create scenario





* host 配置属于host/templete的web 监控
* name web监控的名字具有唯一性（支持宏）
* application 选择web监控属于的组， web scenario items 配置属于组后可以在monitoring >>> latest data 分组中看到数据
* new application 输入名字即scenario创建新的组
* authentiction 认证选项
* none，无认证
* basic authentication 基本认证
* NTLM authentication NTLM (window nt lan manager)认证
* reties 重试机制，默认为1，最多支持10次重试
* agent 浏览器类型，支持自定义
* HTTP proxy 代理格式 默认端口为1080
* variables ：scenario级的变量(macros)可以在scenario steps(url, post variables)中配置
* {macros1}=value1 例 {username}=alex
* {macros2}=regex:<regular expression> 例 {hostid}=regex:hostid is ([0-9]+)

setps

* URL: 监控的 web页面（必须全路径）
* Post:传递给页面的参数
* variables: 设置变量
* timeout: 超时时间
* required: 页面中能匹配到字符
* status code 页面返回码

### 自定义监控

zabbix\_get 用于zabbix\_server到zabbix-agent的数据获取，通常可以用来检测验证agent的配置是否正确。其语法是

zabbix\_get [-hV] -s <host name or IP> [-p <port>] [-I <IP address>] -k <key>

* -s 远程zabbix-agent的IP地址或者是主机名
* -p 远程zabbix-agent的端口
* -I 本机出去的IP地址，用于一台机器中有多个网卡的情况
* -k 获取zabbix-agent数据所使用的key

下面举个自定义一个key用来测试zabbix\_get，

$ yum -y install zabbix-get

$ zabbix\_get -s 127.0.0.1 -p 10050 -k "system.cpu.load[all.avgl]"

0.0000

$ vim /etc/zabbix/zabbix\_agentd.conf

Server= 服务器端IP

UserParameter=login-user,who |wc -l #280行

$ /etc/init.d/zabbix-agent restart

# 服务端

$ zabbix\_get -s 172.16.1.72 -p 10050 -k "login-user"

2

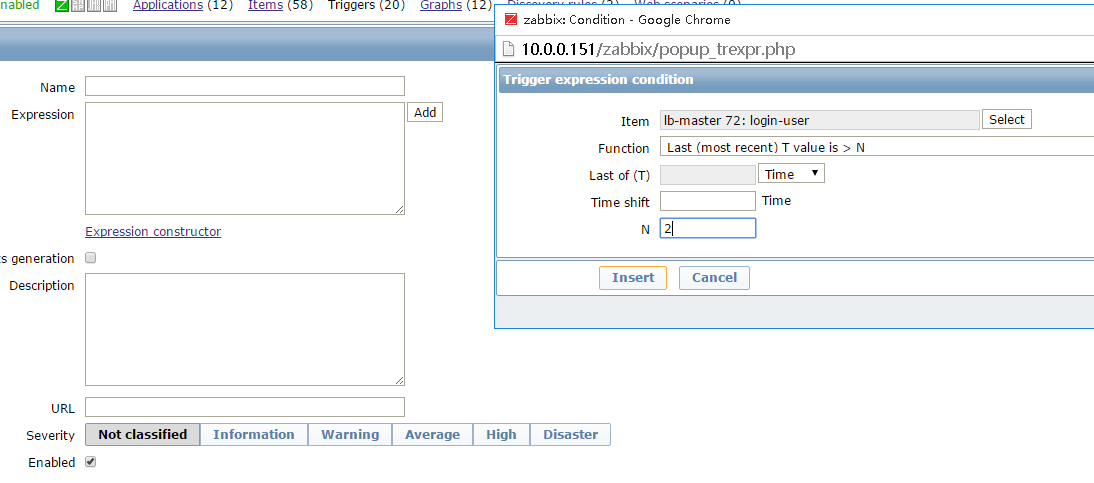
好的现在回到web-zabbix界面在 configuration –>> hosts –>> Items –>> create item

这里name最后与key一样(选择里的都是系统自带的key没有自定义的，所以这里手动输就好)

update interval 多久(秒)检测一次，还有最短最久保存时间

其它的都可以不用管

设置报警可以在**Triggers** 下创建，这里就不介绍啦



#### 监控TCP状态

$ ss **state** all|awk '{++S[$1]} END {**for** (a **in** S) {printf "%11-s %s\n", a,S[a]}}'

$ netstat -an|awk '/^tcp/ {++S[$NF]} END {**for**(a **in** S) {printf "%11-s %s\n", a, S[a]}}'

$ vim /etc/zabbix/zabbix\_agent.d/tcp\_connect.conf

UserParameter=tcp\_connect.established, ss **state** all|awk '($1=="ESTAB"){print $0}'|wc -l

UserParameter=tcp\_connect.synsent, ss **state** all|awk '($1=="SYN-SEND"){print $0}'|wc -l

UserParameter=tcp\_connect.synrecv, ss **state** all|awk '($1=="SYN-RECV"){print $0}'|wc -l

UserParameter=tcp\_connect.finwait1, ss **state** all|awk '($1=="FIN-WAIT-1"){print $0}'|wc -l

UserParameter=tcp\_connect.finwait2, ss **state** all|awk '($1=="FIN-WAIT-2"){print $0}'|wc -l

UserParameter=tcp\_connect.timewait, ss **state** all|awk '($1=="TIME-WAIT"){print $0}'|wc -l

UserParameter=tcp\_connect.close, ss **state** all|awk '($1=="CLOSE"){print $0}'|wc -l

UserParameter=tcp\_connect.closeing, ss **state** all|awk '($1=="CLOSING"){print $0}'|wc -l

UserParameter=tcp\_connect.closewait, ss **state** all|awk '($1=="CLOSE-WAIT"){print $0}'|wc -l

UserParameter=tcp\_connect.lastack, ss **state** all|awk '($1=="LAST-ACK"){print $0}'|wc -l

UserParameter=tcp\_connect.listen, ss **state** all|awk '($1=="LISTEN"){print $0}'|wc -l

然后导入模版，地址为E:\zabbix\_templates\tcp\_templates.xml 这里需要指出如果time\_wait过多时需要考虑调整内核，并且找出根本原因。

vi /etc/sysctl.conf

编辑文件，加入以下内容：

net.ipv4.tcp\_syncookies = 1

net.ipv4.tcp\_tw\_reuse = 1

net.ipv4.tcp\_tw\_recycle = 1

net.ipv4.tcp\_fin\_timeout = 30

然后执行 /sbin/sysctl -p 让参数生效。

* net.ipv4.tcp\_syncookies = 1 表示开启SYN Cookies。当出现SYN等待队列溢出时，启用cookies来处理，可防范少量SYN攻击，默认为0，表示关闭；
* net.ipv4.tcp\_tw\_reuse = 1 表示开启重用。允许将TIME-WAIT sockets重新用于新的TCP连接，默认为0，表示关闭；
* net.ipv4.tcp\_tw\_recycle = 1 表示开启TCP连接中TIME-WAIT sockets的快速回收，默认为0，表示关闭。
* net.ipv4.tcp\_fin\_timeout 修改系統默认的 TIMEOUT 时间

如果是mysql的time\_wait多，那么需要考虑代码中是否使用mysql.close()

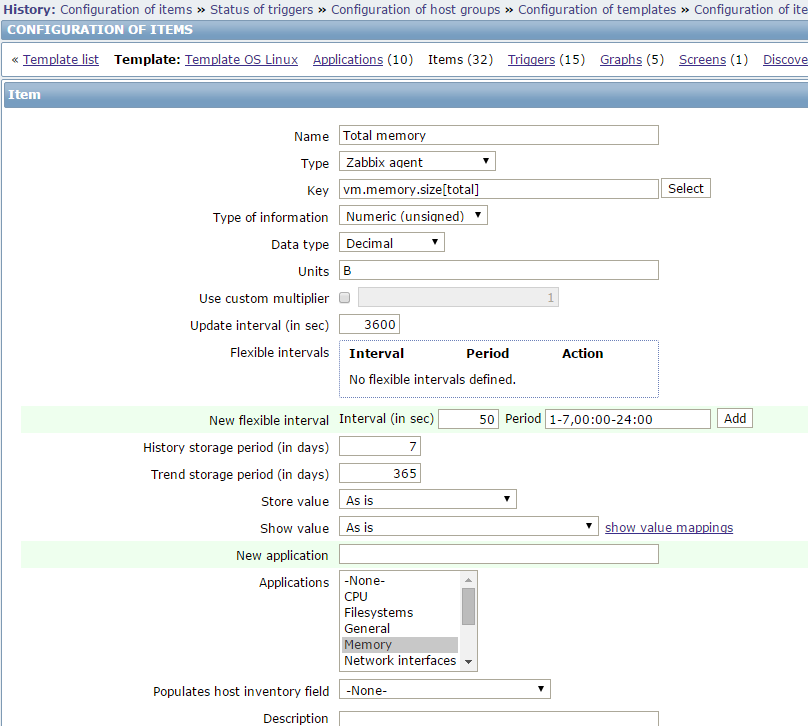
### Items

Items 就是监控项可以配置获取监控数据的方式，取值的数据类型，获取数值的间隔，历史数据保存时间，趋势数据保存时间，监控key的分组。Items可以存在于模板(template)中，也可以存在于主机中，模板中是可以复用，对需要重复配置的监控项归类。

可以在 Template >>> items 下增加items

$ zabbix\_get -s 127.0.0.1 -k vm.memory.size[total]

1952677888



Items 属性参数说明

* Host : 选择Host或者Template 也就是说，Items存在于Host或者Template中，不能单独存在
* Name : Item的名字，可以用宏变量$1,$2代表Item名称，第1,2…参数例   
  items key为 vfs.fs.size[/,free]   
  Free disk space on $1 便会显示为 “Free disk space on /”
* Type : Items默认类型包括Zabbix agent, Simple checks, SNMP, Zabbix internal, IPMI, JMX monitoring, Telnet, SSH 等多种监控方式
* Key ： Item的key通过选择菜单能查找出可以支持的item keys，但自定义的key不能通过选择菜单找出来，需要手动输入，
* Host interface :选择主机的接口，在配置了多种监控方式或者多个IP监控时，可以基于主机接口来选择
* Type of information ： 数据在进行类型转换之后存入数据库中（Numeric(unsiged):64位无符号的整数， Numeric(float) 浮点数类型，Character 字符类型， log 日志文件， Text 广西不限制大小
* Data type 数据类型用于存储Items 中key 所获取的数值，用于存储在不同的表中(history,history\_str)
* Boolean: 在数据存储的时候将原本的值替换为0或1， TRUE 为1，FALSE为0（区别大小写 其中TRUE有：true, yes,up,on, runing,enabled,avaible,t
* Octal 八进制
* Decimal 十进制
* Hexdecimal 十六进制数
* Units 如果设置了一个单位符号， Zabbix将会处理接收到的数据，并且把数值转换成为需要显示的单位，默认情况下，如果原始值超过1000，除以1000，并相应地显示
* 处理时间时有三个参数 Unixtime(yyyy,mm,dd hh:mm:ss). Uptime (hh:mm:ss) S : “yyy mmm dddhhh mmm sssms”
* Use custom multiplier 如果开启该选项，所接收到的数值将会被自乘以整数或者浮点数，用这个选项可以将KB，MB， 等转换为B，Bps, 另外，Zabbix无法正确设置prefixes(KMG等单位)
* Update interval(in sec) 间隔时间（秒）通过Item收集数据
* Flexible intervals 创建一个非常规的更新时间
* Keep history (in days) 数据库中保存历史数据的天数，这个可以在全局设置中设置
* Keep trends (in days) 在数据库中保存历史趋势（第小时的最小值，最大值，平均值，统计）超过时间设置的数据，将会被housekeeper进程清理掉
* Store value
* As is: 无预处理
* Delta(speed per second) 数值计算的方法(value - prev\_value)/(time - prev\_time) 当前值 减去上一次收到的值除以当前时间减去上一次接收到值的时间
* Delta(simple change) 数值计算的方法为(value - prev\_value)
* show value 对Item的值做一个视图显示，value mapping 不会改变接收到的值，仅仅改变显示。
* new application 输入一个名称将会把item加入一个新的application
* application 将item添加到一个或者多个已经存在的application
* Populate host inventory field 将item归属于那个资产管理组中

#### Key

key 可以带参数，该参数为一个数组列表，可以同时传递多个参数，

**vfs**.fs.size[/]

**vfs**.fs.size[/opt]

自定义key 在/etc/zabbix/zabbix\_agentd.conf 里设置 281行

UserParameter=key,command

如果需要使用特殊字符的处理需要设置一个参数 UnsafeUserParameters=1

因为需要设置很多key，所以会另选一个配置文件来设置key /etc/zabbix/zabbix\_agentd.d下来设置

key为wc可以传很多参数，后面跟命令，是需要得到的结果是

$ vim /etc/zabbix/zabbix\_agentd.conf

UserParameter=wc[\*],grep -c "$2" $1

$ zabbix\_get -s 127.0.0.1 -k wc[/etc/passwd,root]

2

网卡流量的监控 net.if.in[if, <mode>] ip为网卡接口， mode为想要取值的类型，(betes(默认),packets, errors,dropped)

还有几个key也可以了解一下

net.if.out[**if**, <mode>]

net.if.collisions[**if**, <mode>]

net.if.total[**if**, <mode>]

net.if.discovery

监控端口

**net**.tcp.listen[port]

**net**.tcp.port[<ip>,port]

**net**.tcp.service[service,<ip>,<port>]

**net**.ucp.listen[port]

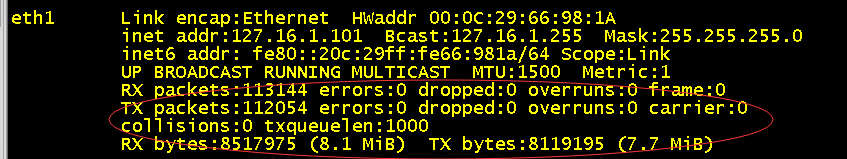
举例

[root@master zabbix\_agentd.d]# **zabbix\_get** **-s** 127.0.0.1 **-k** **net**.if.in[eth0]

50282864

[root@master zabbix\_agentd.d]# **zabbix\_get** **-s** 127.0.0.1 **-k** **net**.if.in[eth0,packets]

123687



可以看到这都是ifconfig里的数据

日志监控（agent必须工作在主动模式下在ServerActive为激活状态）

key : log[/path/to/file/file\_name,<regxp>,<encoding>,<maxlines>,<mode>,<output>] zabbix-agent 可以对日志内容通过与正则表达式过滤，需要注意的是日志，zabbix用户 必须对这个日志有读取权限否则会提示‘unsupported’

比如需要监控/var/log/secure就需要把这个文件属主改一下chown zabbix.root /var/log/secure

然后增加一个items,key 设置为**log[/var/log/secure,session]**,type设置为**zabbix agetn(active)** , type of information 设置为**log**, application 选择**security**

### 使用脚本发邮件报警

zabbix 传级脚本，3个参数，

* $1 邮件收件人
* $2 邮件主题目
* $3 邮件内容

grep Alert /etc/zabbix/zabbix\_server.conf

**cd** /usr/lib/zabbix/alertscripts

cat send\_email.sh

#!/bin/bash

email\_File=/tmp/email.log

**function** **main**(){

**echo** "**$3**" >**$email\_File**

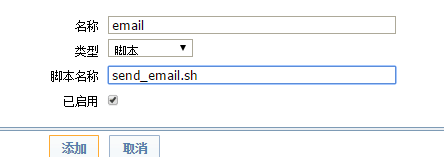
/usr/bin/dos2unix **$email\_File**

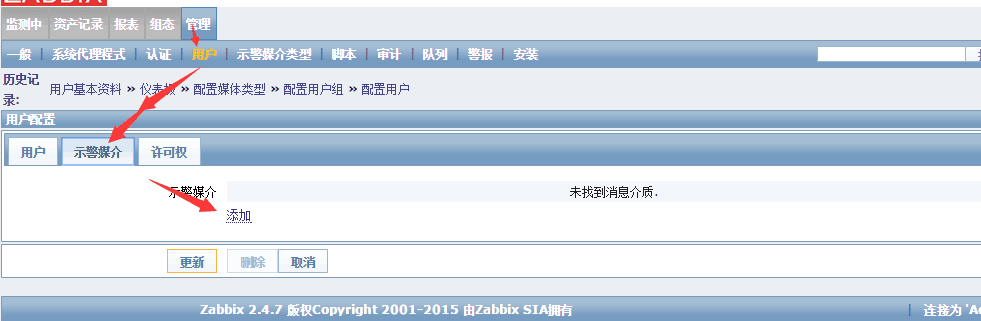
/bin/mail -s "**$2**" "**$1**" <**$email\_File**

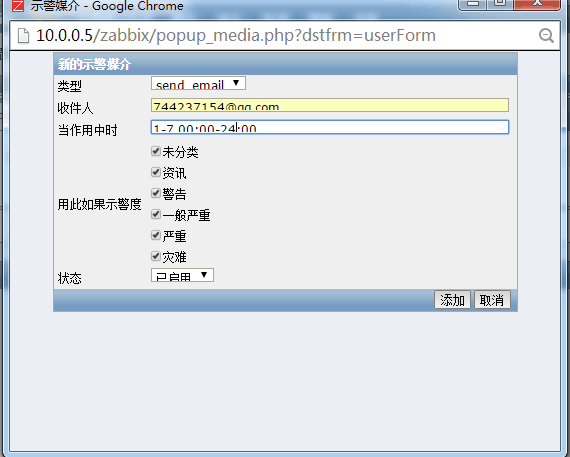
}

main "**$1**" "**$2**" "**$3**"

然后在web-zabbix端增加 administration >> media types >>> create media type







### 增加一个模块

在客户端

$ grep "Include" /etc/zabbix/zabbix\_agentd.conf #查看包括的文件夹

Include=/etc/zabbix/zabbix\_agentd.d/

$ **cd** /etc/zabbix/zabbix\_agentd.d/ #在面里去配置文件

$ cat zabbix-linux-plugin.conf

UserParameter=linux\_status[\*],/etc/zabbix/zabbix\_linux\_plugin.sh "$1" "$2" "$3"

$ cat /etc/zabbix/zabbix\_linux\_plugin.sh #配置执行的脚本，记得加执行权限

#!/bin/bash

**tcp\_status\_fun**(){

TCP\_STAT=$1

#netstat -n | awk '/^tcp/ {++state[$NF]} END {for(key in state) print key,state[key]}' > /tmp/netstat.tmp

ss -ant | awk 'NR>1 {++s[$1]} END {for(k in s) print k,s[k]}' > /tmp/netstat.tmp

TCP\_STAT\_VALUE=$(grep "$TCP\_STAT" /tmp/netstat.tmp | cut -d ' ' -f2)

**if** [ -z $TCP\_STAT\_VALUE ];**then**

TCP\_STAT\_VALUE=0

**fi**

**echo** $TCP\_STAT\_VALUE

}

**nginx\_status\_fun**(){

NGINX\_PORT=$1

NGINX\_COMMAND=$2

**nginx\_active**(){

/usr/bin/curl "http://127.0.0.1:"$NGINX\_PORT"/nginx\_status/" 2>/dev/null| grep 'Active' | awk '{print $NF}'

}

**nginx\_reading**(){

/usr/bin/curl "http://127.0.0.1:"$NGINX\_PORT"/nginx\_status/" 2>/dev/null| grep 'Reading' | awk '{print $2}'

}

**nginx\_writing**(){

/usr/bin/curl "http://127.0.0.1:"$NGINX\_PORT"/nginx\_status/" 2>/dev/null| grep 'Writing' | awk '{print $4}'

}

**nginx\_waiting**(){

/usr/bin/curl "http://127.0.0.1:"$NGINX\_PORT"/nginx\_status/" 2>/dev/null| grep 'Waiting' | awk '{print $6}'

}

**nginx\_accepts**(){

/usr/bin/curl "http://127.0.0.1:"$NGINX\_PORT"/nginx\_status/" 2>/dev/null| awk NR==3 | awk '{print $1}'

}

**nginx\_handled**(){

/usr/bin/curl "http://127.0.0.1:"$NGINX\_PORT"/nginx\_status/" 2>/dev/null| awk NR==3 | awk '{print $2}'

}

**nginx\_requests**(){

/usr/bin/curl "http://127.0.0.1:"$NGINX\_PORT"/nginx\_status/" 2>/dev/null| awk NR==3 | awk '{print $3}'

}

**case** $NGINX\_COMMAND **in**

active)

nginx\_active;

;;

reading)

nginx\_reading;

;;

writing)

nginx\_writing;

;;

waiting)

nginx\_waiting;

;;

accepts)

nginx\_accepts;

;;

handled)

nginx\_handled;

;;

requests)

nginx\_requests;

**esac**

}

**memcached\_status\_fun**(){

M\_PORT=$1

M\_COMMAND=$2

**echo** -e "stats\nquit" | nc 127.0.0.1 "$M\_PORT" | grep "STAT $M\_COMMAND " | awk '{print $3}'

}

**redis\_status\_fun**(){

R\_PORT=$1

R\_COMMAND=$2

(**echo** -en "INFO \r\n";sleep 1;) | nc 127.0.0.1 "$R\_PORT" > /tmp/redis\_"$R\_PORT".tmp

REDIS\_STAT\_VALUE=$(grep ""$R\_COMMAND":" /tmp/redis\_"$R\_PORT".tmp | cut -d ':' -f2)

**echo** $REDIS\_STAT\_VALUE

}

**main**(){

**case** $1 **in**

tcp\_status)

tcp\_status\_fun $2;

;;

nginx\_status)

nginx\_status\_fun $2 $3;

;;

memcached\_status)

memcached\_status\_fun $2 $3;

;;

redis\_status)

redis\_status\_fun $2 $3;

;;

\*)

**echo** $"Usage: $0 {tcp\_status key|memcached\_status key|redis\_status key|nginx\_status key}"

**esac**

}

main $1 $2 $3

$ /etc/init.d/zabbix-agent restart

##服务端

$ zabbix\_get -s 10.0.0.101 -p 10050 -k "linux\_status[tcp\_status,CLOSED]"

**Configuration**(主态) –>> **Templates** (棋版) –>> **Import** (汇入)在create template旁边 –>> 选择文件Linux\_TCP\_Status\_templates.xml

这时在template下就可以看到linux\_TCP\_templates模块啦