> 《Zabbix企业级分布式监控系统第2版》随书代码
>
> 代码仓库地址 https://github.com/zabbix-book/zabbix_v2
>
> 书籍购买地址 https://item.jd.com/12653708.html

394页

```
shell# vim /etc/zabbix/zabbix_proxy.conf
### Option: ProxyLocalBuffer
#       Proxy will keep data locally for N hours, even if the data have already
been synced with the server.
#       This parameter may be used if local data will be used by third party
applications.
#
# Mandatory: no
# Range: 0-720
# Default:
# ProxyLocalBuffer=0

### Option: ProxyOfflineBuffer
#       Proxy will keep data for N hours in case if no connectivity with Zabbix
Server.
#       Older data will be lost.
#
# Mandatory: no
# Range: 1-720
# Default:
# ProxyOfflineBuffer=1
```

395页

```
shell# rpm  -ivh  http://repo.zabbix.com/zabbix/4.0/rhel/7/x86_64/zabbix-
release-4.0-1.el7.noarch.rpm
shell# yum  install  -y  zabbix-proxy-mysql   mariadb-server
```

```
注意：如果安装的版本高于4.0，则请下载并安装最新版本的yum源地址，地址格式如下：
http://repo.zabbix.com/zabbix/4.0/rhel/7/x86_64/ #4.0代表当前的版本
在安装zabbix-release后，将会增加如下的yum源：

shell#  cat /etc/yum.repos.d/zabbix.repo
[zabbix]
name=Zabbix Official Repository - $basearch
baseurl=http://repo.zabbix.com/zabbix/4.0/rhel/7/$basearch/
```

```
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-ZABBIX-A14FE591

[zabbix-non-supported]
name=Zabbix Official Repository non-supported - $basearch
baseurl=http://repo.zabbix.com/non-supported/rhel/7/$basearch/
enabled=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-ZABBIX
gpgcheck=1
```

396页

```
shell# systemctl  start  mariadb    #启动数据库服务
shell# systemctl  enable mariadb    #配置开机自启动
shell# mysqladmin -uroot password admin   #设置root密码为admin
shell# mysql -uroot -padmin                  #用root登录MySQL
mysql> create  database  zabbix_proxy  character  set  utf8; #创建数据库
zabbix_proxy，并将其字符集设置为UTF-8
mysql> grant all privileges  on zabbix_proxy.*  to  zabbix_proxy@localhost
identified  by  'zabbix_proxy';  #将数据库zabbix_proxy授权给用户zabbix_proxy，授权
地址为localhost，访问密码为zabbix_proxy
mysql> flush  privileges;     #刷新权限
#由于默认安装的MySQL是空密码（5.5版本以下），5.6版本以上的数据库密码策略会有所不同，启动服务
时会自动生成一个复杂的root密码。可以用mysql_secure_installation命令进行安全配置
shell# mysql_secure_installation
```

```
shell# zcat /usr/share/doc/zabbix-proxy-mysql-4.0.0/schema.sql.gz | mysql -
uzabbix_proxy -pzabbix_proxy zabbix_proxy  #如果zcat命令执行失败，则可以尝试先解压缩
再导入MySQL中，如果导入成功，则以下步骤忽略
shell# cd /usr/share/doc/zabbix-proxy-mysql-4.0.0
shell# gzip –d schema.sql.gz #解压缩后的文件名称为schame.sql
shell# mysql -uzabbix_proxy -pzabbix_proxy zabbix_proxy
mysql> use zabbix_proxy
mysql> source /usr/share/doc/zabbix-proxy-mysql-4.0.0/schema.sql; #导入
schema.sql
```

397页

表9-2 zabbix_proxy.conf的重要参数

参　　数	参数配置	描　　述
ProxyMode	ProxyMode=0	默认参数值为0，即Zabbix-Proxy工作于主动模式下
ProxyMode=1	表示Zabbix-Proxy工作于被动模式下	
Server	Server=X.X.X.X	该参数工作于主动模式（ProxyMode=0）下，从X.X.X.X这个IP地址的Zabbix-Server中获取监控配置信息。在被动模式下此参数无效
ServerPort	ServerPort=10051	默认参数值为10051，工作于主动模式下，在被动模式下此参数无效
Hostname	Hostname=Zabbix proxy	Zabbix-Proxy的主机名字。需要注意，这个名字要具有唯一性，不能重复。在配置Proxy时，Web中的配置会用到此参数，即Administration-DM-Create proxy-Proxy name，这个是配置Zabbix-Proxy的重点
HostnameItem	HostnameItem=system.hostname	该参数在Hostname没有定义时才会生效
ListenPort	ListenPort=10051	Zabbix-Proxy的默认端口
SourceIP	SourceIP=X.X.X.X	用于多网卡环境中，指定Zabbix-Proxy连接的外网IP地址
DBHost	DBHost=localhost	Zabbix-Proxy的数据库IP地址
DBName	DBName=zabbix_proxy	Zabbix-Proxy的数据库名称
DBUser	DBUser= zabbix_proxy	Zabbix-Proxy的数据库用户名
DBPassword	DBPassword= zabbix_proxy	Zabbix-Proxy的数据库密码
DBSocket	DBSocket=/var/lib/mysql/mysql.sock	Zabbix-Proxy的mysql.sock文件

启动Zabbix-Proxy服务的命令如下：

```
shell# systemctl   start    zabbix-proxy  #启动服务
shell# systemctl   enable   zabbix-proxy  #添加到开机启动项
```

398页

9.1.5　查看Zabbix-Proxy日志

如果需要查看启动日志，则执行以下命令：

```
shell# tail -f /var/log/zabbix/zabbix_proxy.log
```

404页

9.4.2 配置过程

1. 配置Zabbix-Agent

在配置注册的Zabbix-Agent时，需要关注的配置参数如下：

```
shell# vim /etc/zabbix/zabbix_agentd.conf
ServerActive=10.0.0.1     #Zabbix-Server所在的IP地址，可以同时填写多个IP
Hostname=ec2-10-9-0-8     #主机名
HostnameItem=system.hostname  #获取主机名的key，当Hostname参数已设置值的时候，此参数失效。
HostMetadata=Linux linux_host  #主机元数据的标识，字符串的长度范围为0~255。可以满足于云环境主机的使用，因为在云环境中主机名通常没有规律可循，基本都是是随机生成的字符串，。
HostMetadataItem=system.uname  #①当HostMetadata参数已设置值的时候，此参数失效；②用于获取数据的key，此处的system.uname为获取系统内核名称的key；③对于其返回结果的字符串，字符的长度范围限制为0~255；④可以使用Zabbix-Agent原生内置的key，也可以使用用户自定义的key，还可以使用system.run[]这个key(前提是EnableRemoteCommands参数已经打开)。
```

407-408页

vfs.fs.discovery ，支持Zabbix-Agent。

```
{
    "data": [
        {
            "{#FSNAME}": "/",
            "{#FSTYPE}": "ext4"
        },
        {
            "{#FSNAME}": "/boot",
            "{#FSTYPE}": "ext4"
        }
    ]
}
```

net.if.discovery， 支持Zabbix-Agent。

```
{
    "data": [
        {
            "{#IFNAME}": "lo"
        },
        {
            "{#IFNAME}": "eth0"
        }
    ]
}
```

discovery[{#MACRO1},oid1,{#MACRO2},oid2,…,] ，支持SNMP v1/v2 /v3。

```
{"data":[
```

```
    {
        "{#SNMPINDEX}":"<idx>",
        "{#SNMPVALUE}":"<value>”
    },
    {
        "{#SNMPINDEX}":"<idx>",
        "{#SNMPVALUE}":"<value”>
    }]
}
system.cpu.discovery：

{
    "data": [
        {
            "{#CPU.NUMBER}": 0,
            "{#CPU.STATUS}": "online"
        },
        {
            "{#CPU.NUMBER}": 1,
            "{#CPU.STATUS}": "online"
        },
        {
            "{#CPU.NUMBER}": 2,
            "{#CPU.STATUS}": "online"
        }
    ]
}
db.odbc.discovery[<description>,<dsn>] ， 支持ODBC（Server）。

{
    "data": [
        {
            "{#CITY}": "BeiJing",
            "{#COUNT}": "13"
        },
        {
            "{#HCITY}": "ShangHai",
            "{#COUNT}": "7"
        }
    ]
}
```

409页

表9-3 内置的宏功能

宏	功能描述
{#SERVICE.NAME}	服务的名称
{#SERVICE.DISPLAYNAME}	服务的显示名称
{#SERVICE.DESCRIPTION}	服务的描述
{#SERVICE.STATE}	运行状态值0 Running1 Paused2 Start pending3 Pause pending4 Continue pending

宏	功能描述
	5 Stop pending6 Stopped7 Unknown
{#SERVICE.STATENAME}	服务状态（Running, Paused, Start pending, Pause pending, Continue pending, Stop pending, Stopped, Unknown ）
{#SERVICE.PATH}	服务的实际路径
{#SERVICE.USER}	服务的用户
{#SERVICE.STARTUP}	服务开机自启动类型0 Automatic1 Automatic delayed2 Manual3 Disabled4 Unknown
{#SERVICE.STARTUPNAME}	服务开机自启动类型（Automatic, Automatic delayed, Manual, Disabled, Unknown）
{#SERVICE.STARTUPTRIGGER}	启动时触发0 no startup triggers1 has startup triggers

410页

测试net.if.discovery，数据如下：

```
shell# zabbix_get -s 127.0.0.1 -k net.if.discovery
{
    "data": [
        {
            "{#IFNAME}": "lo"
        },
        {
            "{#IFNAME}": "eth0"
        }
    ]
}
```

测试vfs.fs.discovery，数据如下：

```
shell# zabbix_get -s 127.0.0.1 -k vfs.fs.discovery
{
```

```
    "data": [
        {
            "{#FSNAME}": "/",
            "{#FSTYPE}": "rootfs"
        },
       省略部分显示
        {
            "{#FSNAME}": "/",
            "{#FSTYPE}": "ext4"
        },
       省略部分显示
        {
            "{#FSNAME}": "/net",
            "{#FSTYPE}": "autofs"
        }
    ]
}
```

412页

9.5.5 配置Zabbix客户端
配置Zabbix客户端，命令如下：

```
shell# egrep -v "(#|^$)"  /etc/zabbix/zabbix_agentd.conf
LogFile=/var/log/zabbix/zabbix_agentd.log
EnableRemoteCommands=0                  #是否支持远程命令，0表示不支持
Server=127.0.0.1,192.168.0.240       #Zabbix-Server端的IP地址
StartAgents=8                           #Agent开启的进程个数
ServerActive=192.168.0.240:10051     #Zabbix-Server端的IP地址
Timeout=30                              #超时时间
Include=/etc/zabbix/zabbix_agentd.conf.d/ #子配置文件路径
UnsafeUserParameters=1                  #在自定义key中可以包含特殊字符
```

```
shell# cat /etc/zabbix/scripts/web_site_code_status
#!/bin/bash

# function:monitor web  status from zabbix
# License: GPL
# mail:itnihao@qq.com
# version:1.0 date:2012-12-09
source /etc/bashrc >/dev/null 2>&1
source /etc/profile  >/dev/null 2>&1

#/usr/bin/curl -o /dev/null -s -w %{http_code} http://$1/
Web_SITE_discovery () {
```

```
    Web_SITE=($(cat /etc/zabbix/scripts/Web.txt|grep -v "^#"))
    printf '{\n'
    printf '\t"data":[\n'
    for((i=0;i<${#Web_SITE[@]};++i))
    {
      num=$(echo $((${#Web_SITE[@]}-1)))
     if [ "$i" != ${num} ]; then
       printf "\t\t{ \n"
       printf "\t\t\t\"{#SITENAME}\":\"${Web_SITE[$i]}\"},\n"
     else
         printf  "\t\t{ \n"
        printf  "\t\t\t\"{#SITENAME}\":\"${Web_SITE[$num]}\"}]}\n"
     fi
    }
}
web_site_code () {
    /usr/bin/curl -o /dev/null -s -w %{http_code} http://$1
}
case "$1" in
web_site_discovery)
        Web_SITE_discovery
;;
web_site_code)
        web_site_code $2
;;
*)
echo "Usage:$0 {web_site_discovery|web_site_code [URL]}"
;;
esac
```

413页

```
shell# sh web_site_code_status  web_site_discovery
{
          "data":[
                    {
                          "{#SITENAME}":"www.itnihao.com"},
                    {
                          "{#SITENAME}":"www.baidu.com"},
                    {
                          "{#SITENAME}":"www.weibo.com"}]}
```

414页

此处如果采用Python来编写，则代码如下：

```
#!/bin/env python
# -*- coding: utf-8 -*-

# Last modified: 2017-08-12 14:47
# Author: itnihao
# Mail: itnihao@qq.com

import urllib2
import json
import sys
from optparse import OptionParser
import socket

def discovery():
    '''从Web.txt中发现URL，如果路径不同，请修改为实际路径'''
    try:
        f=open('Web.txt','r')
        url_list=f.read().split()
        f.close()
    except:
        return
    urls = []

    for k in  url_list:
        urls += [{'{#SITENAME}':k}]
    print json.dumps({'data':urls},sort_keys=True,indent=7,separators=
(',',':'))

def get_code(url):
    '''获取URL的HTTP code'''
    socket.setdefaulttimeout(5)

    try:
       response = urllib2.urlopen(url)
       code = response.getcode()
       response.close()
       print code
    except:
       print 0

if __name__ == '__main__':
    parser = OptionParser(
        usage="%prog [-l URL] [-d d]",
        version="%prog $Revision$",
        prog="url",
        description="""Zabbix Web Monitor Discovery""",
    )

    parser.add_option(
```

```
        "--discovery",
        action="store",
        type="string",
        dest="discovery",
        default="false",
        help="method is discovery",
    )

    parser.add_option(
        "--url",
        action="store",
        type="string",
        dest="url",
        default=None,
        help="url",
    )

    (opts, args) = parser.parse_args()
    m = opts.discovery
    url = opts.url
    if m == "true":
        discovery()
    elif url != None:
        get_code(url)
```

415页

输出格式如下：

```
shell# python webmonitor.py --discovery=true
{
          "data":[
                    {
                          "{#SITENAME}":"www.itnihao.com"
                    },
                    {
                          "{#SITENAME}":"www.baidu.com"
                    },
                    {
                          "{#SITENAME}":"www.weibo.com"}
            ]
}
shell# python webmonitor.py --url=http://www.baidu.com
200
```

9.5.7 自定义key配置文件
shell版本的脚本，自定义UserParameter配置文件如下：

```
shell# cat /etc/zabbix/zabbix_agentd.conf.d/web_site_discovery.conf
UserParameter=web.site.discovery,/etc/zabbix/scripts/web_site_code_status web_site_discovery
UserParameter=web.site.code[*],/etc/zabbix/scripts/web_site_code_status web_site_code $1
```

Python版本的脚本，自定义UserParameter配置文件如下：

```
shell# cat /etc/zabbix/zabbix_agentd.conf.d/web_site_discovery.conf
UserParameter=web.site.discovery,/etc/zabbix/scripts/web_site_code_status.py
discovery=true
UserParameter=web.site.code[*],/etc/zabbix/scripts/web_site_code_status web_site_code.py url=“$1“
```

416页

域名如下：

```
shell# cat /etc/zabbix/scripts/Web.txt
http://www.itnihao.com
http://www.baidu.com
http://www.weibo.com
```

测试命令如下：

```
shell# zabbix_get -s 127.0.0.1 -k web.site.discovery
{
          "data":[
                    {
                          "{#SITENAME}":"www.itnihao.com"},
                    {
                          "{#SITENAME}":"www.baidu.com"},
                    {
                          "{#SITENAME}":"www.weibo.com"}
]
}
shell# zabbix_get -s 127.0.0.1 -k web.site.code[www.itnihao.com]
200
shell# zabbix_get -s 127.0.0.1 -k web.site.code[www.baidu.com]
200
shell# zabbix_get -s 127.0.0.1 -k web.site.code[www.weibo.com]
200
```

以上自定义UserParameter的清单如下：

```
/etc/zabbix/zabbix_agentd.conf                          #Agent配置文件
/etc/zabbix/scripts/web_site_code_status            #权限755
/etc/zabbix/scripts/Web.txt                               #网站URL存放文件
```

```
/etc/zabbix/zabbix_agentd.conf.d/web_site_discovery.conf  #子配置文件
```

425页

对于Zabbix-Server的内部监控，提供了一个用于发现主机接口的key，如下所示。

`zabbix[host,discovery,interfaces]`

添加key后，在Zabbix-Server内部将会生成如下数据：

```
{"data":[{"{#IF.CONN}":"192.168.3.1","{#IF.IP}":"192.168.3.1","{#IF.DNS}":"","{#IF.PORT}":"10050","{#IF.TYPE}":"AGENT","{#IF.DEFAULT}":1}]}
```

其内置的一些宏功能如表9-4所示。

表9-4　内置的宏功能

宏（Macro）	功 能 描 述
{#IF.CONN}	接口的IP地址或者DNS名称
{#IF.IP}	接口的IP地址
{#IF.DNS}	接口的DNS名称
{#IF.PORT}	接口使用的端口
{#IF.TYPE}	接口类型（AGENT，SNMP，JMX，IPMI）
{#IF.DEFAULT}	默认的接口状态
0非	
1默认接口	
{#IF.SNMP.BULK}	是否启用SNMP批量处理
0不启用	
1启用	

428页

```
9.6.2  安装salt-master
安装salt-master，命令如下：
shell# rpm  -ivh  http://dl.fedoraproject.org/pub/epel/6/x86_64/epel- release-6-8.noarch.rpm
shell# yum  install  salt-master
```

启动服务，命令如下：

```
shell# service   salt-master   start
shell# chkconfig salt-master   on
```

429页

安装salt-minion，命令如下：

```
shell# yum  install salt-minion
```

配置salt-minion，命令如下：

```
shell# vim  /etc/salt/minion
master: salt-master.itnihao.com     #Master的IP地址或域名
id: zabbix-agent.itnihao.com        #Minion的标识
```

启动服务，命令如下：

```
shell# service   salt-minion   start
shell# chkconfig salt-minion   on
```

430-431页

在salt-master中配置命令如下：

```
shell# mkdir  /srv/salt/  #因为StatStack安装好后无/srv/salt目录，所以建立此文件夹
shell# vim /srv/salt/top.sls
base:
   '*.itnihao.com':
   - zabbix
shell# mkdir  /srv/salt/zabbix
shell# vim  /srv/salt/zabbix/init.sls
zabbix-agent:
    service:
        - running
        - watch:
            - file: zabbix_agentd.conf
            - file: zabbix_agentd.conf.d
            - pkg: zabbix-agentd
    require:
        - pkg: zabbix-agent
zabbix-agentd:
    pkg.installed:
        - name: zabbix-agent
        - version: '2.2.2-0.el6.zbx'
        - skip_verify: True
        - skip_suggestions: True
        - fromrepo: zabbix
```

```
        - refresh: True

zabbix_agentd.conf:
    file.managed:
        - name: /etc/zabbix/zabbix_agentd.conf
        - source: salt://zabbix/conf/zabbix_agentd.conf
        - mode: 644
        - user: zabbix
        - group: zabbix
        - template: jinja
zabbix_agentd.conf.d:
    file.recurse:
        - name: /etc/zabbix/zabbix_agentd.conf.d
        - source: salt://zabbix/conf/zabbix_agentd.conf.d
        - include_empty: True
        - user: zabbix
        - group: zabbix
        - dir_mode: 755
        - file_mode: 644
scripts:
    file.recurse:
        - name: /etc/zabbix/scripts
        - source: salt://zabbix/scripts
        - include_empty: True
        - user: zabbix
        - group: zabbix
        - dir_mode: 755
        - file_mode: 700
```

整个代码的目录结构如下：

```
/srv/salt/
├── top.sls
└── zabbix
    ├── conf
    |   ├── zabbix_agentd.conf        #Zabbix-Agent的配置文件
    |   └── zabbix_agentd.conf.d      #子配置文件
    |       ├── haproxy_host_status.conf
    |       ├── haproxy_main_status.conf
    |       ├── haproxy_status_discovery.conf
    ├── init.sls                          #StatStack[问题同上]的状态配置文件
    └── scripts
        ├── haproxy_host_status
        ├── haproxy_main_status
        └── haproxy_status_discovery
```

9.6.6 执行状态同步

执行状态同步，命令如下：

```
shell# salt  '*'  state.highstate
```