> 《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

322页

```
shell# egrep -v "(^#|^$)" /etc/zabbix/zabbix_agentd.conf
Server=127.0.0.1,10.0.2.50        #被动模式连接的Zabbix-Server的IP地址
ServerActive=127.0.0.1,10.0.2.50  #主动模式连接的Zabbix-Server的IP地址，开启此项参数，将会自动打开主动模式；将其注释，则会关闭主动模式。
Hostname=Host-001 #在主动模式中，Hostname作为Zabbix-Server处理数据的唯一依据，要求Hostname在Zabbix-Server中具有唯一性。当有多个Zabbix-Agent的主机名都配置相同时，会造成该主机名下的主动模式监控项数据存储错乱，因为在不同的时间周期内，其存储的是不同Zabbix-Agent的数据。[已修复]
StartAgents=3 #Agent的进程个数，用于被动模式，如果大于0，则会监听10050端口。如果只需要主动模式，则可以将其设置为0，将被动模式关闭
```

```
shell# tail -f /var/log/zabbix/zabbix_server.log #在Zabbix-Server查看日志
1463:20181031:102756.847 cannot send list of active checks to "10.1.0.15": host [Host-001] not found    #Zabbix-Agent向Zabbix-Server发起数据请求，查询主机名为“Host-001”的监控项列表，而Zabbix-Server经过查询后，发现不存在“Host-001”的主机名，因此将此错误记录到日志。
shell# tail -f /var/log/zabbix/zabbix_agentd.log #登录10.1.0.15主机查看日志
17085:20181031:102756.847 no active checks on server [10.0.2.50:10051]: host [Host-001] not found  #在Zabbix Web的主机列表中并不存在主机名为Host-001的主机。
```

323页

```
shell# vim  /etc/zabbix/zabbix_agentd.conf
Server=127.0.0.1,10.0.2.50  #多个IP之间用逗号隔开
shell# systemctl restart zabbix-agent
```

324页

```
shell# vim  /etc/zabbix/zabbix_agentd.conf
#       Example: ServerActive=127.0.0.1:20051,zabbix.domain,[::1]:30051,::1,
[12fc::1]
ServerActive=127.0.0.1:10051,10.10.10.1:10051 #多个IP用逗号隔开
Hostname=Host-001    #主机名
shell# systemctl restart zabbix-agent
```

328页

```
shell# zabbix_sender -z 10.0.2.14 -p 10051 -s "Trapper" -k trapperlog -o
"trapper work is ok" -vv
zabbix_sender [4429]: DEBUG: answer [{"response":"success","info": "processed:
1; failed: 0; total: 1; seconds spent: 0.000054"}]
info from server: "processed: 1; failed: 0; total: 1; seconds spent: 0.000054"
sent: 1; skipped: 0; total: 1
以上消息提示发送成功。如果发送失败，则会给出失败提示，需要检测上面的各参数设置是否正确。

shell# zabbix_sender -z 10.0.2.14 -p 10051 -s  "Trapper" -k  trapperLlogbad -o
"trapper work is ok" -vv
zabbix_sender [5036]: DEBUG: answer [{"response":"success","info": "processed:
0; failed: 1; total: 1; seconds spent: 0.000029"}]
info from server: "processed: 0; failed: 1; total: 1; seconds spent: 0.000029"
sent: 1; skipped: 0; total: 1
```

329-330页

```
主机名 key value timestamps
下面我们通过示例来演示写一个文件。

shell# vim /tmp/es_stats.tmp
- es.status yellow
- es.number_of_nodes 1
- es.unassigned_shards 25
- es.initializing_shards 0
- es.active_primary_shards 25
- es.relocating_shards 0
- es.active_shards 25
- es.number_of_data_nodes 1
- es.process.cpu.percent 3
- es.indices.search.fetch_total 238
- es.jvm.mem.non_heap_used_in_bytes 111875576
- es.process.cpu.total_in_millis 3264300
- es.process.open_file_descriptors 447
- es.indices.refresh.total_time_in_millis 575787
- es.indices.indexing.index_time_in_millis 78646
- es.jvm.mem.pools.old.peak_max_in_bytes 89522176
- es.thread_pool.index.queue 0
- es.jvm.mem.pools.survivor.max_in_bytes 4456448
- es.indices.indexing.delete_time_in_millis 0
- es.http.total_opened 71313
- es.jvm.threads.peak_count 49
- es.indices.merges.total_time_in_millis 2177650
- es.indices.flush.total_time_in_millis 5381
通过zabbix_sender读取文件数据发送给Zabbix-Server服务器，命令如下：

shell# /usr/bin/zabbix_sender -c /etc/zabbix/zabbix_agentd.conf -s Es-001 -i
/tmp/es_stats.tmp -vv
zabbix_sender [21713]: DEBUG: answer [{"response":"success","info": "processed:
64; failed: 0; total: 64; seconds spent: 0.000396"}]
info from server: "processed: 64; failed: 0; total: 64; seconds spent:
0.000396"
sent: 64; skipped: 0; total: 64
```

337页

```
shell ./configure --with-net-snmp
1. 配置被监控端的SNMP
以Linux系统为例，语句如下：

shell# yum -y install net-snmp
shell# mv /etc/snmpd/snmpd.conf  /etc/snmpd/snmpd.conf.bak
shell# vim /etc/snmpd/snmpd.conf
com2sec mynetwork 192.168.0.240 public_monitor
com2sec mynetwork 127.0.0.1 public
group MyROGroup v2c mynetwork
access MyROGroup "" any noauth prefix all none none
view all included .1 80
shell# chkconfig snmpd on
shell# service snmpd restart
如果是配置Windows的SNMP监控，则配置方法稍有不同。
如果不是服务器，而是路由器、交换机、防火墙等其他硬件设备，则需要通过命令行或者Web界面来配置
SNMP监控。
2. 使用snmpwalk获取SNMP数据
在Zabbix-Server上测试，语句如下：

#SNMPv2测试命令
shell# snmpwalk -v 2c -c public 127.0.0.1
shell# snmpwalk -v 2c -c public 127.0.0.1 SNMPv2-MIB::sysUpTime.0
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (166696) 0:27:46.96
#SNMPv3测试命令
```

```
shell# snmpwalk -v 3 -u snmp -a SHA -A SHA@PWD -x AES -X AES@PWD -l authPriv
172.18.3.4  SNMPv2-MIB::sysUpTime.0
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (166696) 0:27:46.96
如果没有snmpwalk命令，则需要安装net-snmp-utils（类RHEL系统）。语句如下：

shell# yum install net-snmp-utils
```

339页

```
4. 添加SNMPv3的Item
添加SNMPv3的Item，如图8-21所示。对应的snmpwalk命令如下：

shell# snmpwalk -v 3 -u snmp -a SHA -A SHA@PWD -x AES -X AES@PWD -l authPriv
172.18.3.4  1.3.6.1.2.1.1.5.0
```

340页

```
shell# snmpwalk -v 2c -c public 192.168.1.2 1.3.6.1.2.1.2.2.1.2
IF-MIB::ifDescr.1 = STRING: lo
IF-MIB::ifDescr.2 = STRING: eth0
IF-MIB::ifDescr.3 = STRING: eth1
```

341页

```
{
    "data": [
        {
            "{#SNMPINDEX}": "1",
            "{#IFDESCR}": "lo"
        },
        {
            "{#SNMPINDEX}": "2",
            "{#IFDESCR}": "eth0"
        },
        {
            "{#SNMPINDEX}": "3",
            "{#IFDESCR}": "eth1""
        }
    ]
}
```

```
shell# snmpwalk -v 2c -c public 192.168.1.2 1.3.6.1.2.1.31.1.1.1.6
IF-MIB::ifHCInOctets.1 = Counter64: 1306368016205
IF-MIB::ifHCInOctets.2 = Counter64: 343739706698
IF-MIB::ifHCInOctets.3 = Counter64: 0
```

343页

```
1. 安装snmptrapd软件包

shell# yum -y install net-snmp-perl perl-Digest-SHA1 perl-Sys-Syslog perl-List-
MoreUtils perl-IO-stringy net-snmp-utils perl perl-Module-Build perl-Time-HiRes
2. 安装snmptt软件包

shell# rpm -ivh https://dl.fedoraproject.org/pub/epel/7/x86_64/Packages/
e/epel-release-7-11.noarch.rpm
shell# yum -y install snmptt
# 在epel源中安装的软件包是snmptt-1.4-0.9.beta2.el7.noarch
```

344页

```
3. 检查snmptthandler是否存在

shell# ls /usr/share/snmptt/snmptthandler-embedded
如果输出显示不存在，则需要检查snmptt安装是否成功。
4. 修改snmptrapd.conf配置文件

shell# vim /etc/snmp/snmptrapd.conf
authCommunity   log,execute,net public
perl do "/usr/share/snmptt/snmptthandler-embedded"
5. 修改snmptt.ini配置文件
修改snmptt.ini配置文件，黑色字体字是需要修改的参数。

shell# vim /etc/snmp/snmptt.ini
[General]
......
mode = daemon
......
net_snmp_perl_enable = 1
......
translate_log_trap_oid = 2
......
mibs_environment = ALL
```

```
......
date_time_format = %H:%M:%S %Y/%m/%d
......
[Logging]
stdout_enable = 0
log_enable = 1
log_file = /var/log/snmptt/snmptt.log
......
syslog_enable = 0
......
[Debugging]
DEBUGGING = 1
DEBUGGING_FILE = /var/log/snmptt/snmptt.debug
DEBUGGING_FILE_HANDLER = /var/log/snmptt/snmptthandler.debug
6. 修改snmptt.conf配置文件
修改snmptt.conf配置文件，此文件用于存放匹配OID的规则。这里我们添加一个General事件，配置如下：

shell# vim /etc/snmp/snmptt.conf
EVENT general .* "General event" Normal
FORMAT ZBXTRAP $aA $ar $1
```

345页

```
7. 启动snmptrapd服务
修改snmptrapd的启动参数，并启动服务，命令如下：

shell# vim /etc/sysconfig/snmptrapd
OPTIONS="-m +ALL -Lsd -On -p /var/run/snmptrapd.pid"
shell# systemctl daemon-reload
shell# systemctl enable  snmptrapd
shell# systemctl restart snmptrapd
8. 启动snmptt服务

shell# systemctl enable  snmptt
shell# systemctl restart snmptt
9. 测试snmptrap服务
测试我们刚刚添加的General事件，命令如下：

shell# snmptrap -v 2c -c public 127.0.0.1 '' .1.3.6.1.4.1.8072.9999
.1.3.6.1.4.1.8072.9999 s 'I am a snmp trap message'
查看日志，命令如下：

shell# tail -f /var/log/snmptt/snmptt.log
12:55:17 2018/10/21 NET-SNMP-MIB::netSnmpExperimental Normal "General event"
127.0.0.1 - ZBXTRAP 127.0.0.1 127.0.0.1 I am a snmp trap message
10. 配置日志切割
snmptt软件包默认已经做了日志切割，此步骤可不用手动配置，参考如下：
```

```
shell# vi /etc/logrotate.d/zabbix_traps
/var/log/snmptt/snmptt.log {
    weekly
    size 10M
    compress
    compresscmd /usr/bin/bzip2
    compressoptions -9
    notifempty
    dateext
    dateformat -%Y%m%d
    missingok
    maxage 365
    rotate 10
}
11. 修改Zabbix-Server配置文件
修改Zabbix-Server配置文件，读取snmptt的日志文件，命令如下：

shell# vi /etc/zabbix/zabbix_server.conf
StartSNMPTrapper=100
SNMPTrapperFile=/var/log/snmptt/snmptt.log  #必须和snmptt.ini的日志路径一致
shell# systemctl restart zabbix-server
```

346页

```
#MIB文件下载地址为https://kb.vmware.com/selfservice/microsites/search.do?
language=en_US&cmd=displayKC&externalId=1013445
shell# unzip 1013445_VMware-mibs-6.0.0-2906283.zip #解压缩后目录是vmw
shell# mkdir -p snmpttconf/vmw
shell# for i in `ls vmw/*.mib`; do snmpttconvertmib --in=$i --out=snmpttconf/
$i.conf; done #使用命令将 MIB文件转换为 snmptt配置文件
shell# cat snmpttconf/vmw/*.conf > snmptt.vmw.conf
shell# sed -i 's/FORMAT/FORMAT ZBXTRAP $aA/g' snmptt.vmw.conf
shell# egrep -v "(converted|#)" snmptt.vmw.conf >> /etc/snmp/snmptt.conf
shell# systemctl restart snmptt
2. 测试SNMPTraps
查看Vmware的SNMP OID，如下所示。

shell# tail -n 100 /etc/snmp/snmptt.conf
EVENT authenticationFailure .1.3.6.1.6.3.1.1.5.5 "Status Events" Normal
FORMAT ZBXTRAP $aA An authenticationFailure trap signifies that the SNMP $*
SDESC
An authenticationFailure trap signifies that the SNMP
entity has received a protocol message that is not
properly authenticated.  While all implementations
of SNMP entities MAY be capable of generating this
```

```
trap, the snmpEnableAuthenTraps object indicates
whether this trap will be generated.
Variables:
EDES
```

347页

```
shell# snmptrap -v 2c -c public 127.0.0.1 '' .1.3.6.1.6.3.1.1.5.5
.1.3.6.1.6.3.1.1.5.5 s 'Vmware auth fail'
shell# tail -f /var/log/snmptt/snmptt.log
16:21:37 2018/10/21 SNMPv2-MIB::authenticationFailure Normal "Status Events"
127.0.0.1 - ZBXTRAP 127.0.0.1 An authenticationFailure trap signifies that the
SNMP Vmware auth fail
```

349页

```
shell# sed  -i  '/# StartIPMIPollers=0/a[这里有个a，对吗？][正确，sed命令的一种语
法]StartIPMIPollers=5' zabbix_server.conf
shell# service  zabbix-server  restart
在Zabbix中，对IPMI功能的支持是通过IPMI命令中sensor的参数获取数据。[不通顺]Zabbix-Server
在获取IPMI监控数据时，在zabbix_server.conf中开启DebugLevel=4，会添加Added sensor字符
的日志。程序代码在源码的src/zabbix_server/poller/checks_ipmi.c中，如下所示。

zabbix_log(LOG_LEVEL_DEBUG, "Added sensor: host:'%s:%d' id_type:%d id_sz:%d
id:'%s'"
   " reading_type:0x%x ('%s') type:0x%x ('%s') full_name:'%s'", h->ip, h->port,
   s->id_type, s->id_sz, sensor_id_to_str(id_str, sizeof(id_str), s->id, s-
>id_type, s->id_sz),
   s->reading_type, ipmi_sensor_get_event_reading_type_string(s->sensor), s-
>type,
   ipmi_sensor_get_sensor_type_string(s->sensor), full_name);
Zabbix中IPMI的设计文档地址如下：
https://www.zabbix.org/wiki/Docs/specs/ZBXNEXT-300
```

350页

```
1. 安装IPMI工具
这里以Linux系统CentOS 6为例，安装命令如下：

shell# yum install OpenIPMI ipmitool
2. 启动IPMI服务
安装完毕后，启动服务。使用如下命令在物理机上才能成功启动服务，虚拟机不支持。

shell# systemctl  start ipmi
starting ipmi drivers:                                [  ok  ]
shell# systemctl start ipmievd
starting ipmievd:                                     [  ok  ]
```

351页

```
3. 配置IPMI地址
如果你的物理服务器已经配置了IPMI，则不用通过操作系统中的ipmitool命令来配置IPMI地址了。

shell# ipmitool lan print 1   #显示lan 1的配置信息
shell# ipmitool lan set 1 ipaddr 10.10.10.10
shell# ipmitool lan set 1 netmask 255.255.255.0
shell# ipmitool lan set 1 defgw ipaddr 10.10.10.1
配置用户：

shell# ipmitool lan set 1 access on          #开启lan 1的用户访问
shell# ipmitool user list 1                  #列出lan 1的用户
shell# ipmitool user set name 10 sensor
shell# ipmitool user set password 10 sensor
shell# ipmitool user enable 10
shell# ipmitool user priv 10 2 1
shell# ipmitool user list 1
ID  Name           Callin  Link Auth  IPMI Msg   Channel Priv Limit
2   root             true    true       true       ADMINISTRATOR
10  sensor          true    false      true       USER

4. ipmitool常用命令

shell# ipmitool -I lan -H 服务器地址 -U root -P 密码 power off （硬关机，直接切断电源）
shell# ipmitool -I lan -H 服务器地址 -U root -P 密码 power soft （软关机，即如同轻按一下开机按钮）
shell# ipmitool -I lan -H 服务器地址 -U root -P 密码 power on （硬开机）
shell# ipmitool -I lan -H 服务器地址 -U root -P 密码 power reset （硬重启，这也许会常用）
shell# ipmitool -I lan -H 服务器地址 -U root -P 密码 power status （获取当前电源状态）

5. 查看IPMI支持的参数
```

```
shell# ipmitool -H 10.10.10.10  -Usensor -L USER sensor list
```

355页

```
shell# yum install java java-devel   zabbix-java-gateway
```

356页

```
8.8.4  配置Zabbix-Java-Gateway
在Zabbix-Java-Gateway服务器中，修改配置文件，如下所示。

shell# egrep '=' /etc/zabbix/zabbix_java_gateway.conf
LISTEN_IP="0.0.0.0"
LISTEN_PORT=10052      #Zabbix-Java-Gateway监听的端口
PID_FILE="/var/run/zabbix/zabbix_java.pid"
START_POLLERS=50       #Zabbix-Java-Gateway进程开启的数量
在Zabbix-Server服务器中，修改配置文件，如下所示。

shell# egrep  -v  "(^#|^$)"  /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
JavaGateway=X.X.X.X    #Java-Gateway服务器的IP地址，如果Java-Gateway和
                         #Zabbix-Server在一台机器中，就可以写为127.0.0.1
JavaGatewayPort=10052    #Zabbix-Java-Gateway连接的端口
StartJavaPollers=5          #Java轮询进程的个数，要小于START_POLLERS=50
ExternalScripts=/etc/zabbix/externalscripts
Zabbix-Server中的参数StartJavaPollers的数量需要小于[和下面代码不对应，代码中是小于或等于
号]等于Zabbix-Java-Gateway的参数START_POLLERS的数量。

StartJavaPollers <= START_POLLERS
```

357页

```
shell# tail -f /var/log/zabbix/zabbix_java_gateway.log
2018-08-03 22:50:39.003 [main] INFO  com.zabbix.gateway.JavaGateway - Zabbix
Java Gateway 4.0.0alpha9 (revision 82958) has started
2018-08-03 22:50:39.016 [main] INFO  com.zabbix.gateway.JavaGateway - listening
on 0.0.0.0/0.0.0.0:10052
```

```
比如Java应用程序为/usr/share/doc/openjdk-6-jre-
headless/demo/jfc/Notepad/Notepad.jar，若想监控该应用程序的运行情况，则可以用如下命令开
启JMX的支持。

shell# java \
-Dcom.sun.management.jmxremote \
-Dcom.sun.management.jmxremote.port=10053 \
-Dcom.sun.management.jmxremote.authenticate=false \
-Dcom.sun.management.jmxremote.ssl=false \
-jar  /usr/share/doc/openjdk-6-jre-headless/demo/jfc/Notepad/Notepad.jar
```

```
shell# java -jar  /usr/share/doc/openjdk-6-jre-headless/demo/jfc/Notepad/
Notepad.jar
```

```
shell# java \
-Djava.rmi.server.hostname=192.168.0.200\
-Dcom.sun.management.jmxremote \
-Dcom.sun.management.jmxremote.port=10053 \
-Dcom.sun.management.jmxremote.authenticate=true \
-Dcom.sun.management.jmxremote.password.file=/etc/java-6-
openjdk/management/jmxremote.password \
-Dcom.sun.management.jmxremote.access.file=/etc/java-6-
openjdk/management/jmxremote.access \
-Dcom.sun.management.jmxremote.ssl=true \
-Djavax.net.ssl.keyStore=$YOUR_KEY_STORE \
-Djavax.net.ssl.keyStorePassword=$YOUR_KEY_STORE_PASSWORD \
-Djavax.net.ssl.trustStore=$YOUR_TRUST_STORE \
-Djavax.net.ssl.trustStorePassword=$YOUR_TRUST_STORE_PASSWORD \
-Dcom.sun.management.jmxremote.ssl.need.client.auth=true \
-jar /usr/share/doc/openjdk-6-jre-headless/demo/jfc/Notepad/Notepad.jar
```

358页

```
假如Linux系统为CentOS 7，则通过以下命令安装Tomcat。

shell# yum install tomcat –y
```

```
2. 配置Tomcat的JMX

shell# vim /usr/libexec/tomcat/server
#源码安装修改catalina.sh, 放在开头即可
#!/bin/bash

省略部分代码......
-Djava.util.logging.config.file=${LOGGING_PROPERTIES} \
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager"
#添加JMX开启的参数
export CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote"
export CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.
port=10053"
export CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.
authenticate=false"
export CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.ssl= false"
export CATALINA_OPTS="$CATALINA_OPTS -Djava.rmi.server.hostname= 192.168.46.14"
   #此参数可防止主机名不正确而引起JMX无法连接的问题



if [ "$1" = "start" ] ; then
省略部分代码......
修改参数, 需要重启Tomcat, 命令如下:

shell# systemctl restart tomcat
查看10053端口是否开启, 命令如下:

shell# netstat -nlput|grep 10053
tcp6       0      0 :::10053        :::*    LISTEN      16728/java
```

```
对以二进制方式安装的Tomcat配置JMX, 命令如下:

shell# vim apache-tomcat-8.5.32/bin/catalina.sh
#   use nohup so that the Tomcat process will ignore any hangup
#   signals. Default is "false" unless running on HP-UX in which
#   case the default is "true"
# -----------------------------------------------------------------
export CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote"
export CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.
port=10053"
export CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.
authenticate=false"
export CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.ssl= false"
export CATALINA_OPTS="$CATALINA_OPTS -Djava.rmi.server.hostname= 192.168.46.14"
# OS specific support.  $var _must_ be set to either true or false
```

361页

```
shell# wget http://crawler.archive.org/cmdline-jmxclient/cmdline- jmxclient-
0.10.3.jar
#备用地址为https://github.com/zabbix-book/cmdline-jmxclient
shell# java -jar  cmdline-jmxclient-0.10.3.jar -  10.211.55.9:10053
java.lang:type=Memory HeapMemoryUsage
10/25/2013 18:07:49 +0800 org.archive.jmx.Client HeapMemoryUsage:
committed: 32178176
init: 24313856
max:  224395246
used: 15336752
```

```
jmx["java.lang:type=MemoryPool",HeapMemoryUsage.used]
```

362页

```
shell# vim  /usr/sbin/jmx_get
#!/usr/bin/env bash
# https://github.com/zabbix-book/jxm-get
# 脚本来自https://support.zabbix.com/browse/ZBXNEXT-3764

if [ $# != 5 ];then
echo "Usage: $0 <JAVA_GATEWAY_HOST> <JAVA_GATEWAY_PORT> <JMX_SERVER> <JMX_PORT>
<KEY>"
exit;
fi

# create connection
exec 3<>/dev/tcp/$1/$2

# compose message
MSG="{\"request\": \"java gateway jmx\",\"jmx_endpoint\":\"service:jmx:rmi:
///jndi/rmi://$3:$4/jmxrmi\",\"keys\": [\"$5\"]}"
# write message length as zero-padded 16-digit hexadecimal number
printf -v LEN '%016x' "${#MSG}"

# prepare message length in little endian representation
BYTES=""
for i in {0..14..2}
do
BYTES="\\x${LEN:$i:2}$BYTES"
done

# prepend protocol header and message length
printf "ZBXD\\1$BYTES%s" "$MSG" >&3
```

```
# output the result skipping 5 bytes of "ZBXD\\1" header and 8 bytes of message
length
tail -c+13 <&3
```

```
shell# chmod 755  /usr/sbin/jmx_get  #使脚本具有执行权限
shell# jmx_get 10.211.55.10 10052 10.211.55.9 10053
'jmx[\"java.lang:type=Memory\",HeapMemoryUsage.used]'
{"data":[{"value":"8701688"}],"response":"success"}
```

363页

```
JMX的MBean如下：

java.lang:type=GarbageCollector,name=Copy
java.lang:type=GarbageCollector,name=MarkSweepCompact
```

```
用JMX key表示如下：

jmx["java.lang:type=GarbageCollector,name=Copy",CollectionCount]
jmx["java.lang:type=GarbageCollector,name=Copy",CollectionTime]
jmx["java.lang:type=GarbageCollector,name=MarkSweepCompact",CollectionCount]
jmx["java.lang:type=GarbageCollector,name=MarkSweepCompact",CollectionTime]
```

```
jmx["java.lang:type=GarbageCollector,name=Copy",CollectionCount]
jmx["java.lang:type=GarbageCollector,name=Copy",CollectionTime]
jmx["java.lang:type=GarbageCollector,name=MarkSweepCompact", CollectionCount]
jmx["java.lang:type=GarbageCollector,name=MarkSweepCompact",CollectionTime]
jmx["java.lang:type=GarbageCollector,name=ConcurrentMarkSweep",
CollectionCount]
jmx["java.lang:type=GarbageCollector,name=ConcurrentMarkSweep", CollectionTime]
jmx["java.lang:type=GarbageCollector,name=PS MarkSweep",CollectionCount]
jmx["java.lang:type=GarbageCollector,name=PS MarkSweep",CollectionTime]
```

```
jmx.discovery  #获取所有JMX MBean属性
jmx.discovery[beans]  #获取所有JMX MBeans
jmx.discovery[attributes,"*:type=GarbageCollector,name=*"]  #获取所有的
GarbageCollector属性
jmx.discovery[beans,"*:type=GarbageCollector,name=*"]  #获取类型为
GarbageCollector的所有属性
```

```
shell# jmx_get 10.211.55.10 10052 10.211.55.9 10053 'jmx.discovery[beans,
\"java.lang:type=GarbageCollector,name=*\"]'
  {"data":[
    {"value":"{
      "data":[
        {
          "{#JMXDOMAIN}":"java.lang",
          "{#JMXTYPE}":"GarbageCollector",
          "{#JMXOBJ}":"java.lang:type=GarbageCollector,name=Copy",
          "{#JMXNAME}":"Copy"
        },
        {
          "{#JMXDOMAIN}":"java.lang",
          "{#JMXTYPE}":"GarbageCollector",
          "{#JMXOBJ}":"java.lang:type=GarbageCollector,name=MarkSweepCompact",
          "{#JMXNAME}":"MarkSweepCompact"}
      ]
    }"
  }
],
"response":"success"
}
```

365页

```
填写的主要数据如下：
Name：Discovery GarbageCollector
Key：jmx.discovery[beans,"java.lang:type=GarbageCollector,name=*"]
JMX endpoint：service:jmx:rmi:///jndi/rmi://{HOST.CONN}:{HOST.PORT}/jmxrmi
添加Item原型，如图8-45所示。
填写的主要数据如下：
Name：GarbageCollector {#JMXNAME} CollectionCount
Key：jmx[java.lang:type=GarbageCollector,name={#JMXNAME},CollectionCount]
JMX endpoint：service:jmx:rmi:///jndi/rmi://{HOST.CONN}:{HOST.PORT}/jmxrmi
```

369页

```
hell# curl "https://samples.openweathermap.org/data/2.5/weather?lat=35&lon=
139&appid=b6907d289e10d714a6e88b30761fae22"  #如以上链接失效，读者可以通过
https://openweathermap.org/current获取到
{
coord: {
    lon: 139.01,
    lat: 35.02
},
weather: [{
    id: 800,
    main: "Clear",
    description: "clear sky",
    icon: "01n"
}],
base: "stations",
main: {
    temp: 285.514,
    pressure: 1013.75,
    humidity: 100,
    temp_min: 285.514,
    temp_max: 285.514,
    sea_level: 1023.22,
    grnd_level: 1013.75
},
wind: {
    speed: 5.52,
    deg: 311
},
clouds: {
    all: 0
},
dt: 1485792967,
sys: {
    message: 0.0025,
    country: "JP",
    sunrise: 1485726240,
    sunset: 1485763863
},
id: 1907296,
name: "Tawarano",
cod: 200
}
```

380页

```
触发器的表达式如下:

{Zabbix server:web.test.rspcode[Zabbix  Web Login,Zabbix登录检测
10.211.55.9].last()}<>200 and {Zabbix server:web.test.rspcode[Zabbix Web Login,
Zabbix登录检测10.211.55.9 ].count(#3,"200",eq,10m)}
```

381页

```
shell# odbcinst -j
unixODBC 2.3.1
DRIVERS............: /etc/odbcinst.ini
SYSTEM DATA SOURCES: /etc/odbc.ini
FILE DATA SOURCES..: /etc/ODBCDataSources
USER DATA SOURCES..: /root/.odbc.ini
SQLULEN Size.......: 8
SQLLEN Size........: 8
SQLSETPOSIROW Size.: 8
```

382页

```
shell# cat /etc/odbcinst.ini
# Example driver definitions

# Driver from the postgresql-odbc package
# Setup from the unixODBC package
[PostgreSQL]
Description     = ODBC for PostgreSQL
Driver            = /usr/lib/psqlodbcw.so
Setup             = /usr/lib/libodbcpsqlS.so
Driver64          = /usr/lib64/psqlodbcw.so
Setup64           = /usr/lib64/libodbcpsqlS.so
FileUsage         = 1


# Driver from the mysql-connector-odbc package
# Setup from the unixODBC package
[MySQL]
Description       = ODBC for MySQL
Driver            = /usr/lib/libmyodbc5.so
Setup             = /usr/lib/libodbcmyS.so
Driver64          = /usr/lib64/libmyodbc5.so
Setup64           = /usr/lib64/libodbcmyS.so
FileUsage         = 1
```

```
shell# yum install mysql-connector-odbc
shell# vim /etc/odbc.ini
[mysql_127_0_0_1]           #数据源名称，即DSN
Description    = MySQL test database #数据源描述
Driver         = MySQL      #数据库驱动，使用/etc/odbcinst.ini的MySQL驱动名称
Server         = 127.0.0.1 #数据库服务器的域名或者IP地址
User           = zabbix     #数据库用户名
Password     = zabbix     #数据库密码
Port           = 3306       #数据库端口
Database       = zabbix     #数据库名称
```

383页

```
使用isql命令连接数据库，此处我们选择mysql_127_0_0_1这个数据源。

shell# isql mysql_127_0_0_1
+---------------------------------------+
| Connected!                            |
|                                       |
| sql-statement                         |
| help [tablename]                      |
| quit                                  |
|                                       |
+---------------------------------------+
SQL> show tables; #运行MySQL命令查看表
+-------------------------------------------------------------+
| Tables_in_zabbix                                            |
+-------------------------------------------------------------+
| acknowledges                                                |
| actions                                                     |
```

385-386页

```
shell# vi /etc/profile
export ODBCSYSINI=/etc
export ODBCINI=/etc/odbc.ini
```

```
安装Oracle的客户端软件包，命令如下（需要到Oracle官方网站下载软件包，见10.1.2.2节）：
shell# rpm -ivh oracle-instantclient11.2-basic-11.2.0.4.0-1.x86_64.rpm oracle-
instantclient11.2-devel-11.2.0.4.0-1.x86_64.rpm oracle-instantclient11.2-odbc-
11.2.0.4.0-1.x86_64.rpm oracle-instantclient-basic-10.2.0.5-1.x86_64.rpm
oracle-instantclient-devel-10.2.0.5-1.x86_64.rpm oracle-instantclient-odbc-
10.2.0.5-1.x86_64.rpm
```

```
配置Oracle的环境变量，命令如下：

shell# export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/lib/oracle/11.2/
client64/lib:/usr/lib/oracle/10.2.0.5/client64/lib
shell# ln -s /usr/lib64/libodbcinst.so.2.0.0 /usr/lib64/libodbcinst.so.1
配置Oracle OBDC驱动，命令如下：

shell# vim /etc/odbcinst.ini
[Oracle10g]
Description= Oracle ODBC driver for Oracle 10g
Driver     = /usr/lib/oracle/10.2.0.5/client/lib/libsqora.so.10.1

[Oracle11g]
Description  = Oracle ODBC driver for Oracle 11g
Driver       = /usr/lib/oracle/11.2/client64/lib/libsqora.so.11.1
配置连接信息，命令如下：

shell# vim /etc/odbc.ini
[ORCLTEST]
Driver = Oracle11g #Oracle11g驱动
ServerName = 172.18.30.145:1521/orcl
Port = 1521
UserID = zabbix     #监控账户
Password = zabbix   #监控密码

[oracle_172_18_30_146]
Driver = Oracle10g  #Oracle10g驱动
ServerName = 172.18.30.146:1521/orcl  #Oracle的连接信息
Port = 1521
UserID = zabbix     #监控账户
Password = zabbix   #监控密码
使用命令行连接测试，命令如下：

shell# isql -v ORCLTEST
# isql -v oracle_172_18_30_146
+---------------------------------------+
| Connected!                            |
|                                       |
| sql-statement                       |
| help [tablename]                    |
| quit                                |
|                                    |
+----------------------------------------+
SQL>
如果在使用isql命令遇到错误，例如给出如下提示：

# isql -v oracle_172_18_30_146 zabbix zabbix
[01000][unixODBC][Driver Manager]Can't open lib '/usr/lib/oracle/10.2.0.5/
client/lib/libsqora.so.10.1' : file not found
[ISQL]ERROR: Could not SQLConnect
则可能是Oracle环境变量设置有问题，或者缺少相关的依赖包，可用ldd命令查看缺少的库文件名称。命令
如下：

shell# ldd /usr/lib/oracle/10.2.0.5/client/lib/libsqora.so.10.1
ldd: warning: you do not have execution permission for `/usr/lib/oracle/
10.2.0.5/client/lib/libsqora.so.10.1'
省略部分输出内容......
libclntsh.so.10.1 => /usr/lib/oracle/10.2.0.5/client64/lib/libclntsh.so.10.1
(0x00007f0672f2d000)
libodbcinst.so.1 => not found
libc.so.6 => /lib64/libc.so.6 (0x00007f0672b60000)
/lib64/ld-linux-x86-64.so.2 (0x00007f0674e59000)
libnnz10.so => /usr/lib/oracle/10.2.0.5/client64/lib/libnnz10.so
```

387页

```
安装PostgresSQL OBDC驱动，命令如下：

shell# yum install -y postgresql-odbc

shell# vim /etc/odbcinst.ini
# Example driver definitions

# Driver from the postgresql-odbc package
# Setup from the unixODBC package
[PostgreSQL]
Description     = ODBC for PostgreSQL
Driver            = /usr/lib/psqlodbcw.so
Setup             = /usr/lib/libodbcpsqlS.so
Driver64          = /usr/lib64/psqlodbcw.so
Setup64           = /usr/lib64/libodbcpsqlS.so
FileUsage         = 1
```

```
增加数据库连接信息，可以使用如下参数：

shell# vim /etc/odbc.ini
```

```
[PostgreSQL_10]
Description = Postgres to test
Driver = PostgreSQL
Trace = Yes
TraceFile = sql.log
Database = <database-name-here>
Servername = <server-name-or-ip-here>
UserName = <username>
Password = <password>
Port = 5432
Protocol = 6.4
ReadOnly = No
RowVersioning = No
ShowSystemTables = No
ShowOidColumn = No
FakeOidIndex = No
ConnSettings =
使用命令行连接测试，命令如下：

shell# isql -v PostgreSQL_10
```

388页

```
SQL> SELECT h1.host, COUNT(h2.host) AS count FROM hosts h1 LEFT JOIN hosts h2
ON h1.hostid = h2.proxy_hostid WHERE h1.status IN (5, 6) GROUP BY h1.host;
+------------------------------+----------------------+
| host                         | count                |
+------------------------------+----------------------+
| BeiJing                      |    10                |
| IDC1                         |    20                |
| IDC2                         |    20                |
| ShangHai                     |    14                |
| Zabbix Proxy                 |   12                 |
+------------------------------+----------------------+
```

389页

```
#db.odbc.discovery[proxyies,mysql_127_0_0_1]会生成如下的JSON数据结构
{
    "data": [
        {
            "{#HOST}": "BeiJing",
            "{#COUNT}": "10"
        },
        {
            "{#HOST}": "IDC1",
            "{#COUNT}": "20"
        },
        {
            "{#HOST}": "IDC2",
            "{#COUNT}": "20"
        },
        {
            "{#HOST}": "ShangHai",
            "{#COUNT}": "14"
        },
        {
            "{#HOST}": "Zabbix Proxy",
            "{#COUNT}": "12"
        }
    ]
}
```

391页

```
shell# zabbix_get -s 127.0.0.1 -k system.run["ls /"]
bin
boot
dev
etc
home
lib
```