《基于 zabbix 自动化线上实战》

基础篇 第 5 讲

基于 zabbix 内置 key 的应用

一、内置 key 说明：

Zabbix 内置了很多丰富的 key，使得咱们再添加 linux os 模板的时候，已经帮我们把 key 给定义好，这样我们就能够直接链接模板就可以使用了。

我们这边的话列举一些内置 key，然后进行一些简单的说明:当我们内置 key 可以采集到数据的时候我们最好是不用去写自定义 key 再去采集的：(我见过一篇 51 CTO 的写监控用户登录数，还用 w 去监控，没有直接取调用内置 key)：

二、 详情可以查看官方文档：

https://www.zabbix.com/documentation/3.0/manual/config/items/itemtypes/zabbix_agent#supported_item_keys

三、内置监控项 key 列表：

agent.hostname

返回被监控端名称(字符串)

使用方式列举：后面使用的方式是一样的：

```
[root@BJ-monitor-h-01 bin]# ./zabbix_get -s 192.168.10.100 -k agent.hostname
Zabbix server
```

agent.ping

检测被监控端是否存活(1:运行中 其他:未运行)-使用函数 nodata()检测客户端是否正在运行

agent.version

zabbix agent 版本字符串

kernel.maxfiles

系统支持最大的 open files 整数

kernel.maxproc

系统支持最大的进程数量整数

log[file,<regexp>,<encoding>,<maxlines>,<mode>,<output>]

监控日志文件

file - 文件详细路径

regexp - 正则

encoding - 编码

maxlines - zabbix agent 向 server 或者 proxy 发送最大的行数。

这个参数覆盖配置文件 zabbxi_agentd.conf 中的’MaxLinesPerSecond’

mode - 可选值:all (默认), skip (跳过处理老数据).mode 参数从 2.0 版本开始支持

output - 可选项，输出格式模板.

示例: log[/var/log/syslog] log[/var/log/syslog,error] log[/home/zabbix/logs/logfile,,,100]

logrt[file_pattern,<regexp>,<encoding>,<maxlines>,<mode>,<output>]

Monitoring of log file with log rotation support.

file_pattern - 文件绝对路径

net.if.discovery

列出网卡.通常用于低级别的 discovery.JSON 对象

net.if.in[if,<mode>]

网卡入口流量整数.

if - 网卡名称

mode - 可用值: bytes - 字节数 (默认)

packets - 包数量

errors - 错误数量

dropped - 丢包数量

示例 keys: net.if.in[eth0,errors] net.if.in[eth0]

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

网卡出口流量（参数参见 net.if.in）

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

网卡进/出流量的总和（参数参见 net.if.in）

net.tcp.listen[port]

检测端口是否开启 0 - （not listen） 1 - in LISTEN stateport

示例: net.tcp.listen[80]

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

是否可以连接到指定的 TCP 端口 0 - cannot connect 1 - can connect

ip - IP 地址(默认是 127.0.0.1)

port - 端口

范例: net.tcp.port[,80] 检测 web 服务器端口是否运行中

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

检测服务是否开启，并且端口可用 0 - 服务挂了 1 - 服务运行中

service - 如下:ssh, ntp, ldap, smtp, ftp, http, pop, nntp,imap, tcp, https, telnet

ip - IP 地址 (默认 127.0.0.1)

port - 端口 (默认情况为标准端口号)

示例 key: net.tcp.service[ftp,,45]

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

检测服务器性能 0 - 服务挂了; seconds - 链接到服务器端口消耗的时间

service - 如下:ssh, ntp, ldap, smtp, ftp, http, pop, nntp,imap, tcp, https, telnet

ip - IP 地址 (默认 127.0.0.1)

port - 端口（默认情况为标准端口号）

示例 key: net.tcp.service.perf[ssh]

proc.mem[<name>,<user>,<mode>,<cmdline>]

用户进程消耗的内存内存使用量（字节单位).

name - 进程名（默认值 “all processes”）

user - 用户名（默认值“all users”）

mode - 可选值: avg, max, min, sum（默认）

cmdline - 命令行过滤(正则表达时)

示例 keys: proc.mem[,root] - root 的进程消耗了多少内存

proc.mem[zabbix_server,zabbix] - zabbix 用户运行的 zabbix_server 使用了多少内存

proc.mem[,oracle,max,oracleZABBIX]

proc.num[<name>,<user>,<state>,<cmdline>]

某用户某些状态的进程的数量进程数量

name - 进程名称（默认“all processes”）

user - 用户名（默认 “all users”）

state - 可用值: all（默认), run,sleep, zomb

cmdline - 命令行过滤(正则表达时)

示例 keys: proc.num[,mysql] - MySQL 用户运行的进程数量

proc.num[apache2,www-data] - www-data 运行了多少个 apache2 进程

proc.num[,oracle,sleep,oracleZABBIX]

备注：Windows 系统只支持 name 和 user 两个参数

system.boottime

系统启动的时间戳整数.unix 时间戳

system.cpu.intr

设备中断整数

system.cpu.load[<cpu>,<mode>]

CPU 负载浮点数

cpu - 可用值: all（默认), percpu（所有在线 cpu 的负载）

mode - 可用值:avg1（1 分钟 默认值), avg5(5 分钟平均), avg15（15 分钟平均值）

范例 key: system.cpu.load[,avg5]

system.cpu.num[<type>]

CPU 数量处理器个数 type - 可用值: online（默认值), max 范例: system.cpu.num

system.cpu.switches

上下文交换交换次数老命名方式: system[switches]

system.cpu.util[<cpu>,<type>,<mode>]

CPU 利用率百分比

cpu - cpu 数量（默认是所有 cpu）

type - 可用值：idle，nice，user（默认），system（windows 系统默认值），iowait，interrupt，softirq,steal

mode - 可用值：avg1（一分钟平均，默认值），avg5(5 分钟平均，avg15（15 分钟平均值）

范例 key：system.cpu.util[0,user,avg5]

system.hostname[<type>]

返回主机名字符串

type（仅用于 windows 系统） - 可用值：netbios(默认) or host

system.hw.cpu[<cpu>,<info>]

返回 CPU 信息字符/数字

cpu - cpu 数量或者 all（默认）

info - full（默认），curfreq，maxfreq，model 或者 vendor

例如：system.hw.cpu[0,vendor] AuthenticAMD 从 /proc/cpuinfo 、/sys/devices/system/cpu/[cpunum]/cpufreq/cpuinfo_max_freq 获取信息. 如果指定了 CPU 数量和 curfreq 或者 maxfreq，将会返回数值(Hz).

system.hw.devices[<type>]

列出 PCI 或者 USB 文本值

type - pci（默认）or usb

范例：system.hw.devices[pci] 00:00.0 Host bridge: Advanced Micro Devices [AMD] RS780 Host Bridge [..] 返回 lspci 或者 lsusb（不带参数）

system.hw.macaddr[<interface>,<format>]

列出 MAC 地址字符串

interface - all（默认）或者正则表达式

format - full（默认）、short

范例：system.hw.macaddr["eth0$",full] [eth0] 00:11:22:33:44:55 列出指定接口 mac 地址 如果 format 指定为 short，MAC 地址相同的将会被忽略掉

system.localtime[<type>]

系统时间.数字或者字符串

system.run[command,<mode>]

在制定的主机上运行命令文本

command - 命令

mode - wait（默认值，执行超时时间），nowait（不等待)最大可用返回 512KB 数据，包含空白数据。

命令输出数据必须是文本

例如：system.run[“ls -l /”] - 列出/的文件和目录.

Note：启用这个方法，agent 配置文件必须配置 EnableRemoteCommands=1 选项

system.sw.arch

返回软件信息字符串

范例: system.sw.arch

system.sw.os[<info>]

返回系统信息字符串

info - full (default), short ,name

范例: system.sw.os[short] Ubuntu 2.6.35-28.50-generic 2.6.35.11

信息来自如下文件：

/proc/version [short]

/proc/version_signature [name]

/etc/issue.net

system.sw.packages[<package>,<manager>,<format>]

已安装软件列表文本值

package - all (默认)或者正则表达式

manager - all (默认) or a package manager

format - full (默认) , short

范例: system.sw.packages[http]

system.swap.in[<device>,<type>]

交换分区 IN（磁盘交换到内存）数字

device - 交换分区设备 (默认 all)

type - 可选值: count (swapins 数量), sectors(sectors swapped in), pages (pages swapped in).

示例 key: system.swap.in[,pages]

数据采集自: Linux 2.4: /proc/swaps, /proc/partitions, /proc/stat

Linux 2.6: /proc/swaps, /proc/diskstats, /proc/vmstat

system.swap.out[<device>,<type>]

Swap out (f 内存到磁盘) .数字

device - swap 设备 (默认 all)

type - count (number of swapouts), sectors(sectors swapped out), pages (pages swapped out). 示

例 key: system.swap.out[,pages]

数据采集自: Linux 2.4: /proc/swaps, /proc/partitions, /proc/stat

Linux 2.6: /proc/swaps, /proc/diskstats, /proc/vmstat

system.swap.size[<device>,<type>]

交换分区大小字节或者百分比

device - 交换分区 (默认值 all)

type - free (free swap space, default), pfree (free swap space, in percent), pused (used swap space, in percent), total (total swap space), used (used swap space)

示例 system.swap.size[,pfree] - 空闲 swap 百分比

system.uname

返回主机相信信息.字符串

system.uptime

系统运行时长(秒)多少秒使用 s/uptime 来获取

system.users.num

登陆用户数量多少用户 agent 使用 who 命令获取

vfs.dev.read[<device>,<type>,<mode>]

磁盘读取状态整数，浮点数（如果 type 为如下）

device - 磁盘设备 (默认值 “all”)

type - 可选值:sectors, operations, bytes, sps, ops, bps(必须指定, 不同操作系统下不同).

sps, ops, bps stand for: sectors, operations, bytes per second, respectively

mode - 可选值: avg1, avg5, avg15.

备注: 只有 type 为 sps, ops, bps 的时候，第三个参数才被支持。

不同操作系统的 TYPE 参数： FreeBSD - bps Linux - sps OpenBSD - operations Solaris - bytes

示例 key: vfs.dev.read[,operations]

vfs.dev.write[<device>,<type>,<mode>]

磁盘写入状态整数，

device - 磁盘设备 (默认 all)

type - sectors, operations, bytes, sps, ops, bps

mode - one of avg1 (default),avg5 , avg15.

example: vfs.dev.write[,operations] Old naming: io

vfs.file.cksum[file]

计算文件校验 UNIX cksum.

file - 文件完整路径

vfs.file.contents[file,<encoding>]

获取文本内容若为空，只返回 LF/CR characters.

file - 文件完整路径

例如: vfs.file.contents[/etc/passwd] 文件不可以超过 64KB.

vfs.file.exists[file]

检测文件是否存在 1 - 存在 0 - 不存在

file - 文件完整路径

vfs.file.md5sum[file]

文件 MD5 校验码文件 MD5 哈希值

file - 完整路径

vfs.file.regexp[file,regexp,<encoding>,<start line>,<end line>,<output>]

文件中搜索字符串包含字符串的行，或者为空

file - 文件完整路径

regexp - GNU 正则表达式

encoding - 编码

start line - 从哪一行开始，默认第一行

end line - 从哪一行结束，默认最后一行

如: vfs.file.regexp[/etc/passwd,zabbix]

vfs.file.regexp[/path/to/some/file,"([0-9]+)$",,3,5,\1]

vfs.file.regexp[/etc/passwd,^zabbix:.:([0-9]+),,,,\1]

vfs.file.regmatch[file,regexp,<encoding>,<start line>,<end line>]

文件中搜索字符串 0 - 未找到 1 - 找到

file - 文件完整路径

regexp - GNU 正则表达式

encoding - 编码

start line - 哪行开始，默认第一行

end line - 哪行结束，默认最后一行

例如: vfs.file.regmatch[/var/log/app.log,error]

vfs.file.size[file]

文件大小字节 fzabbix 必须有可读此文件的权限

vfs.file.time[file,<mode>]

文件时间信息 Unix 时间戳.

mode - modify (默认, 修改时间), access - 最后访问时间, change - 最后改变时间

例如: vfs.file.time[/etc/passwd,modify] 备注：文件大小有限制

vfs.fs.discovery

列出挂载的文件系统 用于 lld.JSON 对象

vfs.fs.inode[fs,<mode>]

inodes 数量数字

fs - 文件系统

mode - total (默认), free, used, pfree (空闲百分比), pused (使用百分比)

例如: vfs.fs.inode[/,pfree]

vfs.fs.size[fs,<mode>]

磁盘空间，返回本地文件系统的使用量字节

fs - 文件系统

mode - total (默认), free, used, pfree (空闲百分比), pused (使用百分比).

例如: vfs.fs.size[/tmp,free]

vm.memory.size[<mode>]

内存大小字节或百分比

mode - total (默认), active, anon, buffers, cached, exec, file, free, inactive, pinned, shared, wired, used, pused, available

监控项 vm.memory.size[] 允许三种类型的参数：

第一类：包含 total - 总内存

第二类： 系统指定内存类型:active, anon, buffers, cached, exec, file, free, inactive,pinned, shared, wired. 第三类：用户级别，一共使用了多少内存，还有多少内存可用： used, pused, available,pavailable.

web.page.get[host,<path>,<port>]

获取网页内容网页源代码

host - 主机名/域名

path - 文件地址，默认/

port - 端口，默认 80 返回空字符串表示失败. 例如: web.page.get[

web.page.perf[host,<path>,<port>]

获取完全加载网页消耗的时长秒，返回 0 表示失败

host - 主机名/域名

path - html 地址，默认是/

port - 端口,默认 80

[root@BJ-monitor-h-01 bin]# ./zabbix_get -s 192.168.10.100 -k web.page.perf[www.baidu.com]

web.page.regexp[host,<path>,<port>,<regexp>,<length>,<output>]

在网页中搜索字符串 失败则返回空字符 (不匹配).

host - 主机名

path - html 文件路径 (默认值 /)

port - 端口 (默认 80)

regexp - GNU 正则表达式

length - 返回的最大的字符串数量

output - 输出格式模板可选项.

更多课程信息，请关注 龙果学院 官方网站 http://www.roncoo.com/
或关注 龙果 微信公众号 RonCoo_com