| **Key** | | | | |
| --- | --- | --- | --- | --- |
|  | **描述** | **返回值** | **参数** | **注释** |
|  | | | | |
| **agent.hostname** | | | | |
|  | Agent主机名. | String |  | 从配置文件返回Agent主机名的实际值。 |
|  | | | | |
| **agent.ping** | | | | |
|  | Agent可用性检查 | Nothing - 不可用  1 - 可用 |  | 使用**nodata()**触发器函数检查主机不可用性。 |
|  | | | | |
| **agent.version** | | | | |
|  | Zabbix Agent的版本 | 字符串 |  | 例如返回值: 1.8.2 |
|  | | | | |
| **kernel.maxfiles** | | | | |
|  | 系统支持的打开文件的最大数量 | 整数 |  |  |
|  | | | | |
| **kernel.maxproc** | | | | |
|  | 系统支持的最大进程数 | 整数 |  |  |
|  | | | | |
| **log[file,<regexp>,<encoding>,<maxlines>,<mode>,<output>,<maxdelay>]** | | | | |
|  | 日志文件监控。 | Log | **file** - 日志文件完整路径和名称 **regexp** - 描述所需模式的正则表达式 **encoding** - 编码 [标识符](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/zabbix_agent#available_encodings) **maxlines** - Agent将发送到Zabbix服务器或代理的每秒最大行数。此参数覆盖[zabbix\_agentd.conf](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/config/zabbix_agentd)中的“MaxLinesPerSecond”值 **mode** - 可能的值: all (默认值), skip - 跳过处理历史的数据（仅影响新创建的监控项）。 **output** - 可选项，输出格式模板。 \0转义序列替换为匹配的文本，而\N（其中N = 1 … 9）转义序列被替换为第N个匹配组（如果N超过捕获组的数量，则为空字符串）。 **maxdelay** - 最大延迟（秒）。 类型：float。 值：0-（默认）不忽略日志文件行; > 0.0-忽略旧行，以便在“maxdelay”秒内获取最近分析的行。使用前请阅读 [maxdelay](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/log_items#using_maxdelay_parameter)注释！ | 监控项必须定义为[主动检查](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/activepassive#active_checks). 如果文件丢失或权限不允许访问，则监控项不受支持。  如果output 为空 - 返回包含匹配文本的整行。请注意，除“Result为TRUE”之外的所有全局正则表达式类型始终返回整个匹配行，并忽略输出参数。  在Agent端使用输出参数提取内容。  示例: ⇒ log[/var/log/syslog] ⇒ log[/var/log/syslog,error] ⇒ log[/home/zabbix/logs/logfile,,,100]  mode 参数从Zabbix 2.0之后被支持。 output 从Zabbix 2.2之后被支持 maxdelay参数从Zabbix 3.2以后被支持。   更多信息请参考[日志文件监控](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/log_items). |
|  | | | | |
| **log.count[file,<regexp>,<encoding>,<maxproclines>,<mode>,<maxdelay>]** | | | | |
|  | 日志文件监控中匹配行的数量。 | 整数 | **file** - 日志文件完整的路径和名称 **regexp** - 正则表达式 **encoding** - 编码 [标识符](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/zabbix_agent#available_encodings) **maxproclines** - Agent将分析每秒最大行数。默认值为 4\*'MaxLinesPerSecond'在[zabbix\_agent配置文件](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/config/zabbix_agentd). **mode** - 可选的值: all (默认), skip - 跳过处理老数据（仅影响新创建的监控项）。 **maxdelay** - 最大延迟秒数。类型: float. 值: 0 - (默认) 从不忽略每行日志; > 0.0 - 忽略旧行，以便在“maxdelay”秒内获取最近分析的行。在使用前请阅读[maxdelay参数](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/log_items#using_maxdelay_parameter) 的注解！ | 该监控项必须配置为[主动检查](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/activepassive#active_checks). 如果文件丢失或权限不允许访问，则监控项不受支持。  查看更多信息在[日志文件监控](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/log_items).  从Zabbix 3.2.0开始支持 |
|  | | | | |
| **logrt[file\_regexp,<regexp>,<encoding>,<maxlines>,<mode>,<output>,<maxdelay>]** | | | | |
|  | 支持监控轮询的日志文件。 | Log | **file\_regexp** - 文件名以及正则表达式定义的文件名的绝对路径。 **regexp** - 描述匹配内容的正则表达式。 **encoding** - 编码 [标识符](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/zabbix_agent#available_encodings) **maxlines** - Agent发送到Zabbix服务器或者Proxy服务器的每秒最大生成行数。此参数将重写配置文件[zabbix\_agent配置文件](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/config/zabbix_agentd)的参数'MaxLinesPerSecond'的值 **mode** - 可选的值: all (默认), skip - 跳过处理旧数据（仅影响新创建的监控项）。 **output** - 一个可选的输出格式模板。 **\0**转义序列替换为匹配文本，而**\N**（其中N = 1 … 9）转义序列被替换为第N个匹配组（如果N超过捕获组的数量，则为空字符串）。 **maxdelay** - 最大延迟（秒）。 类型：float。 值：0-（默认）不忽略日志文件行; > 0.0-忽略旧行，以便在“maxdelay”秒内获取最近分析的行。使用前请阅读[maxdelay参数](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/log_items#using_maxdelay_parameter)注释！ | 监控项必须定义为[主动检查](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/activepassive#active_checks). 日志轮询是基于文件的最后修改时间。  如果output 为空 - 返回包含匹配文本的整行. 请注意，除“Result为TRUE”之外的所有全局正则表达式类型始终返回整个匹配行，并忽略输出参数。  在Agent端使用输出参数提取内容。  示例: ⇒ logrt["/home/zabbix/logs/^logfile[0-9]{1,3}$",,,100] → 将返回一个文件类似"logfile1" (不会匹配".logfile1") ⇒ logrt["/home/user/^logfile\_.\*\_[0-9]{1,3}$","pattern\_to\_match","UTF-8",100] → 将从文件收集信息例如 "logfile\_abc\_1" 或者 "logfile\_\_001".  mode 参数从Zabbix 2.0以后开始支持。 output参数从Zabbix 2.2开始支持。 maxdelay参数从Zabbix 3.2开始支持。  更多信息请参考[日志文件监控](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/log_items). |
|  | | | | |
| **logrt.count[file\_regexp,<regexp>,<encoding>,<maxproclines>,<mode>,<maxdelay>]** | | | | |
|  | 支持对循环日志文件监控中匹配的行数。 | 整型 | **file\_regexp** - 文件名以及正则表达式定义的文件名的绝对路径。 **regexp** - 描述匹配内容的正则表达式。 **encoding** - 编码 [标识符](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/zabbix_agent#available_encodings) **maxproclines** - Agent将分析每秒最大新生成行数。 默认值为 4\*'MaxLinesPerSecond' 定义在 [zabbix\_agent配置文件](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/config/zabbix_agentd). **mode** - 可能的值: all (默认), skip - 跳过处理旧数据（仅影响新创建的监控项）。 **maxdelay** - 最大延迟（秒）。 类型：float。 值：0-（默认）不忽略日志文件行; > 0.0-忽略旧行，以便在“maxdelay”秒内获取最近分析的行。使用前请阅读[maxdelay参数](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/log_items#using_maxdelay_parameter)注释！ | 监控项必须定义为[主动检查](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/activepassive#active_checks). 日志轮询是基于文件的最后修改时间。  更多信息请参考 [日志文件监控](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/log_items)。  从Zabbix 3.2.0后开始支持。 |
|  | | | | |
| **net.dns[<ip>,name,<type>,<timeout>,<count>,<protocol>]** | | | | |
|  | 检查DNS服务是否开启。 | 0 - DNS宕了 (服务器没有响应或DNS解析失败)  1 - DNS正在运行 | **ip** - DNS服务器的IP地址（默认DNS服务器为空，在Windows上被忽略） **name** - 要查询的DNS名称 **type** - 要查询的记录类型 (默认为 SOA) **timeout** (在windows上忽略) - 请求的超时秒数（默认为1秒） **count** (在windows上忽略) - 请求的尝试次数 (默认为2) **protocol** - 用于执行DNS查询的协议：udp (默认) 或者 tcp | 示例: ⇒ net.dns[8.8.8.8,zabbix.com,MX,2,1]   type 可选的值为: ANY, A, NS, CNAME, MB, MG, MR, PTR, MD, MF, MX, SOA, NULL, WKS (Windows系统除外), HINFO, MINFO, TXT, SRV  不支持国际化域名，请改用IDNA编码名称。  Zabbix 3.0支持协议参数。 Zabbix Agent从版本1.8.6（Unix）和2.0.0（Windows）开始支持SRV记录类型。  Zabbix 2.0之前命名（仍然支持）: net.tcp.dns |
|  | | | | |
| **net.dns.record[<ip>,name,<type>,<timeout>,<count>,<protocol>]** | | | | |
|  | 执行一个DNS查询 | 字符串与所需类型的信息 | **ip** - DNS服务器的IP地址（默认DNS服务器为空，在Windows上被忽略） **name** - 要查询的DNS名称 **type** - 要查询的记录类型 (默认为 SOA) **timeout** (在windows上忽略) - 请求的超时秒数（默认为1秒） **count** (在windows上忽略) - 请求的尝试次数 (默认为2) **protocol** - 用于执行DNS查询的协议：udp (默认) 或者 tcp | 示例: ⇒ net.dns[8.8.8.8,zabbix.com,MX,2,1]   type 可选的值为: ANY, A, NS, CNAME, MB, MG, MR, PTR, MD, MF, MX, SOA, NULL, WKS (Windows系统除外), HINFO, MINFO, TXT, SRV  不支持国际化域名，请改用IDNA编码名称。  Zabbix 3.0支持协议参数。 Zabbix Agent从版本1.8.6（Unix）和2.0.0（Windows）开始支持SRV记录类型。  Zabbix 2.0之前命名（仍然支持）: net.tcp.dns.query |
|  | | | | |
| **net.if.collisions[if]** | | | | |
|  | Number of out-of-window collisions. | 整型 | **if** - 网卡名称 |  |
|  | | | | |
| **net.if.discovery** | | | | |
|  | 网络接口列表 用于低级发现。 | JSON 对象 |  | Zabbix agent从2.0之后开始支持。  Zabbix agent在FreeBSD, OpenBSD 和 NetBSD系统从2.2开始支持。   某些Windows版本（例如Server 2008）可能需要安装最新的更新以支持网卡名称中的非ASCII字符。 |
|  | | | | |
| **net.if.in[if,<mode>]** | | | | |
|  | 网卡流入量统计。 | 整型 | **if** - 网卡名 (Unix); 网卡完整描述或IPv4地址（Windows） **mode** - 可用的值: bytes - 字节数(默认) packets - 包数量 errors - 错误数量 dropped - 丢包数量 | 在Windows上，该选项从64位计数器获取值（如果可用）。64位接口统计计数器在Windows Vista和Windows Server 2008中引入。如果64位计数器不可用，代理使用32位计数器。  从Zabbix Agent1.8.6版本起，支持Windows上的多字节接口名称。  示例: ⇒ net.if.in[eth0,errors] ⇒ net.if.in[eth0]  你可以使用net.if.discovery或net.if.list监控项在Windows上获取网卡说明。  你可以使用该键与Delta（每秒速度）存储值，以获得每秒字节的统计信息。 |
|  | | | | |
| **net.if.out[if,<mode>]** | | | | |
|  | 网卡流出量统计。 | 整型 | **if** - 网卡名称 (Unix); 网卡完整描述或IPv4地址（Windows） **mode** - 可用的值: bytes - 字节数(默认) packets - 包数量 errors - 错误数量 dropped - 丢包数量 | 在Windows上，该选项从64位计数器获取值（如果可用）。64位接口统计计数器在Windows Vista和Windows Server 2008中引入。如果64位计数器不可用，代理使用32位计数器。  从Zabbix Agent1.8.6版本起，支持Windows上的多字节接口名称。  示例: ⇒ net.if.out[eth0,errors] ⇒ net.if.out[eth0]  你可以使用net.if.discovery或net.if.list监控项在Windows上获取网卡说明。  你可以使用该键与Delta（每秒速度）存储值，以获得每秒字节的统计信息。 |
|  | | | | |
| **net.if.total[if,<mode>]** | | | | |
|  | 网卡的进出流量统计信息的总和。 | 整型 | **if** - 网卡名称(Unix); 网卡完整描述或IPv4地址（Windows） **mode** - 可用的值: bytes - 字节数(默认) packets - 包数量 errors - 错误数量 dropped - 丢包数量 | 在Windows上，该选项从64位计数器获取值（如果可用）。64位接口统计计数器在Windows Vista和Windows Server 2008中引入。如果64位计数器不可用，代理使用32位计数器。  示例: ⇒ net.if.total[eth0,errors] ⇒ net.if.total[eth0]  You may obtain network interface descriptions on Windows with net.if.discovery or net.if.list items.  你可以使用net.if.discovery或net.if.list监控项在Windows上获取网卡说明。  你可以使用该键与Delta（每秒速度）存储值，以获得每秒字节的统计信息。  请注意，只有当net.if.in和net.if.out都用于平台上丢弃的数据包时，丢弃的数据包才被支持。 |
|  | | | | |
| **net.tcp.listen[port]** | | | | |
|  | 检查此TCP端口是否处于监听状态。 | 0 - 未监听   1 - 处于监听状态 | **port** - TCP端口 | 示例: ⇒ net.tcp.listen[80]  在Zabbix代理版本1.8.4之后支持Linux。  从Zabbix 3.0.0之后，在Linux内核2.6.14及更高版本上从内核的NETLINK接口获取有关监听TCP套接字的信息。否则，将从/proc/net/tcp和/proc/net/tcp6文件中检索该信息。 |
|  | | | | |
| **net.tcp.port[<ip>,port]** | | | | |
|  | 检查是否可以将TCP连接到指定的端口。 | 0 - 不能连接  1 - 可以连接 | **ip** - IP地址 (默认是 127.0.0.1) **port** - 端口 | 示例: ⇒ net.tcp.port[,80] → 可用于测试在端口80上运行的Web服务器的可用性。  对于简单的TCP性能测试，使用 net.tcp.service.perf[tcp,<ip>,<port>]  请注意，这些检查可能会导致增加系统守护程序日志文件中的额外信息（通常会记录SMTP和SSH会话）。   旧的命名方式: check\_port[\*] |
|  | | | | |
| **net.tcp.service[service,<ip>,<port>]** | | | | |
|  | 检查服务是否正在运行并接受TCP连接。 | 0 - 服务down了  1 - 服务正在运行 | **service** - 如下任一服务: ssh, ldap, smtp, ftp, http, pop, nntp, .imap, tcp, https, telnet (查看[详细信息](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/service_check_details)) **ip** - IP地址 (默认是 127.0.0.1) **port** - 端口号 (默认为标准服务端口号) | 示例: ⇒ net.tcp.service[ftp,,45] → 可用于检测FTP服务器上TCP端口45的可用性。  请注意，这些检测可能会导致增加系统守护程序日志文件的信息（通常会记录SMTP和SSH会话）。  目前不支持检测加密协议（如端口993上的IMAP或端口995上的POP）。 一个解决方案是使用net.tcp.port来检测这些。  目前不支持Windows Agent检测LDAP和HTTPS。  请注意，telnet检测查找登录提示符（'：'在结尾）。  请参考HTTPS服务检测的 [已知问题](https://www.zabbix.com/documentation/3.4/zh/manual/installation/known_issues#https_checks)。  https 和 telnet 服务从Zabbix 2.0开始支持。  旧命名: check\_service[\*] |
|  | | | | |
| **net.tcp.service.perf[service,<ip>,<port>]** | | | | |
|  | 检测TCP服务性能 | 0 - 服务停止。  seconds - 连接到服务花费的时间（秒） | **service** - 如下任一服务: ssh, ldap, smtp, ftp, http, pop, nntp, imap, tcp, https, telnet (参考 [详细描述](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/service_check_details)) **ip** - IP 地址(默认为 127.0.0.1) **port** - 端口号 (默认为标准服务端口号) | 示例: ⇒ net.tcp.service.perf[ssh] → 可以用来检测SSH服务器的初始响应速度。  目前不支持检测加密协议 (如IMAP 上的端口993或者POP上的端口995) 。一个解决方案是使用net.tcp.service.perf[tcp,<ip>,<port>]来检测。  目前不支持Windows代理检查LDAP和HTTPS。  请注意，telnet检测查找登录提示符（'：'在结尾）。  请参考检测HTTPS服务的[已知问题](https://www.zabbix.com/documentation/3.4/zh/manual/installation/known_issues#https_checks)  https 和 telnet 服务从Zabbix 2.0开始支持。  旧名称: check\_service\_perf[\*] |
|  | | | | |
| **net.udp.listen[port]** | | | | |
|  | 检测UDP端口是否处于监听状态。 | 0 - 未监听。  1 - 处在监听状态。 | **port** - UDP端口 | 示例: ⇒ net.udp.listen[68]  在Linux平台从Zabbix agent version 1.8.4 开始支持。 |
|  | | | | |
| **net.udp.service[service,<ip>,<port>]** | | | | |
|  | 检查服务是否正在运行并能响应UDP请求。 | 0 - 服务停了。  1 - 服务正在运行 | **service** - ntp (参考 [详细信息](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/service_check_details)) **ip** - IP地址 (默认是127.0.0.1) **port** - 端口号 (默认使用标准服务端口号) | 示例: ⇒ net.udp.service[ntp,,45] → 可用于测试UDP端口45上NTP服务的可用性。  此选项从Zabbix 3.0.0起支持，但ntp服务可用于以前版本中的net.tcp.service []选项。 |
|  | | | | |
| **net.udp.service.perf[service,<ip>,<port>]** | | | | |
|  | 检测UDP服务的性能 | 0 - 服务停了  seconds - 等待服务响应的秒数 | **service** - ntp (参考[详细信息](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/service_check_details)) **ip** - IP地址 (默认为 127.0.0.1) **port** - 端口 (默认使用标准服务端口号) | 示例: ⇒ net.udp.service.perf[ntp] → 可用于测试NTP服务的响应时间。  此选项从Zabbix 3.0.0起支持，但ntp服务可用于以前版本中的net.tcp.service []选项。 |
|  | | | | |
| **proc.cpu.util[<name>,<user>,<type>,<cmdline>,<mode>,<zone>]** | | | | |
|  | 进程CPU利用率百分比。 | 浮点型 | **name** - 进程名 (默认为 all processes) **user** - 用户名 (默认为 all users) **type** - CPU利用率类型: total (默认), user, system **cmdline** - 可按命令行过滤（支持正则表达式） **mode** - 数据收集模式: avg1 (默认), avg5, avg15  **zone** - 目标区域: current (默认), all. 此参数仅在Solaris平台上受支持。从Zabbix 3.0.3开始，如果代理程序已在Solaris上编译且没有区域支持，而是在支持区域的较新Solaris上运行，并且<zone>参数为缺省值或当前值，则代理程序将返回NOTSUPPORTED（该代理程序不能将结果限制为仅当前区）。但是，在这种情况下，支持<zone>参数值all。 | 示例: ⇒ proc.cpu.util[,root] → 在“root”用户下运行的所有进程的CPU利用率。 ⇒ proc.cpu.util[zabbix\_server,zabbix] → 在zabbix用户下运行的所有zabbix\_server进程的CPU利用率。  返回值基于单CPU核的利用率。例如，使用两个内核的进程的CPU利用率为200％。  进程CPU利用率数据由收集器收集，该收集器最多支持1024个唯一（按名称，用户和命令行）查询。 过去24小时内未被访问的查询将从收集器中删除。  自Zabbix 3.0.0起支持此Key，并可在多个平台上使用 (请查看 [平台支持的监控项](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/supported_by_platform)). |
|  | | | | |
| **proc.mem[<name>,<user>,<mode>,<cmdline>,<memtype>]** | | | | |
|  | 用户进程使用的内存。 | 整型 | **name** - 进程名 (默认是全部进程) **user** - 用户名 (默认是 全部用户) **mode** - 可能的值: avg, max, min, sum (默认值) **cmdline** - 按命令行过滤（它是一个正则表达式） **memtype** - 进程使用的[内存类型](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/proc_mem_notes) | 示例: ⇒ proc.mem[,root] → “root”用户运行的所有进程使用的内存 ⇒ proc.mem[zabbix\_server,zabbix] → zabbix用户运行的所有zabbix\_server进程使用的内存 ⇒ proc.mem[,oracle,max,oracleZABBIX] → oracle用户下，包含有oracleZABBIX命令行运行的所有内存最多的内存  注意: 当多个进程使用共享内存时，进程使用的内存总和可能导致大到不切实际的值。  参考 [说明](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/proc_mem_num_notes) 关于选择进程name 和 cmdline 参数(指定为Linux)。  memtype 参数从Zabbix 3.0.0开始在多个 [平台](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/supported_by_platform) 支持。 |
|  | | | | |
| **proc.num[<name>,<user>,<state>,<cmdline>]** | | | | |
|  | 进程数量。 | 整型 | **name** - 进程名称 (默认是 all processes) **user** - 用户名 (默认是 all users) **state** - 可选的值: all (default), run, sleep, zomb **cmdline** - 按命令行过滤（它是一个正则表达式） | 示例: ⇒ proc.num[,mysql] → 在mysql用户下运行的进程数 ⇒ proc.num[apache2,www-data] → 在www-data用户下运行的apache2进程数 ⇒ proc.num[,oracle,sleep,oracleZABBIX] → 在oracleZABBIX命令行下的oracle用户运行的睡眠状态进程数。  参考 [说明](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/proc_mem_num_notes) 关于选择进程 name 和 cmdline 参数(适用于Linux).  在Windows上，只支持name和user参数。 |
|  | | | | |
| **sensor[device,sensor,<mode>]** | | | | |
|  | 硬件传感器读数。 | 浮点型 | **device** - 设备名称  **sensor** - 传感器名  **mode** - 可能的值: avg, max, min (如果省略此参数，则会对设备和传感器进行逐字处理). | 在Linux 2.4上读取 /proc/sys/dev/sensors   示例: ⇒ sensor[w83781d-i2c-0-2d,temp1]  在Zabbix 1.8.4之前，使用传感器[temp1]格式。 |
| 在Linux 2.6以后的版本上读取 /sys/class/hwmon   请参阅Linux上[sensor](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/sensor)项目的更详细说明。 |
| 在OpenBSD上读取hw.sensors MIB文件  示例: ⇒ sensor[cpu0,temp0] → CPU的温度 ⇒ sensor["cpu[0-2]$",temp,avg] → 前三个CPU温度的平均值  从Zabbix 1.8.4开始支持OpenBSD。 |
|  | | | | |
| **system.boottime** | | | | |
|  | 系统启动时间 | 整数 (Unix时间戳) |  |  |
|  | | | | |
| **system.cpu.discovery** | | | | |
|  | 检测到的CPU/CPU内核列表。用于低级发现。 | JSON对象 |  | 所有平台从2.4.0开始支持 |
|  | | | | |
| **system.cpu.intr** | | | | |
|  | 设备中断数 | 整数 |  |  |
|  | | | | |
| **system.cpu.load[<cpu>,<mode>]** | | | | |
|  | [CPU负载](http://en.wikipedia.org/wiki/Load_(computing)). | 浮点数 | **cpu** - 可能的值: all (default), percpu (总负载除以在线CPU数) **mode** - 可能的值: avg1 (一分钟平均值, 默认值), avg5, avg15 | 示例: ⇒ system.cpu.load[,avg5]  percpu 从Zabbix 2.0.0开始支持  旧名称: system.cpu.loadX |
|  | | | | |
| **system.cpu.num[<type>]** | | | | |
|  | CPU的数量 | 整数 | **type** - 可能的值: online (默认), max | 示例: ⇒ system.cpu.num |
|  | | | | |
| **system.cpu.switches** | | | | |
|  | 上下文交换的数量。 | 整数 |  | 旧名称: system[switches] |
|  | | | | |
| **system.cpu.util[<cpu>,<type>,<mode>]** | | | | |
|  | CPU利用率。 | 浮点型 | **cpu** - <CPU数量> 或者 all (默认值) **type** - 可能的值: idle, nice, user (默认值), system (Windows系统默认值), iowait, interrupt, softirq, steal, guest (在Linux kernels 2.6.24 以及以上支持), guest\_nice (在Linux kernels 2.6.33 以及以上支持) **mode** - 可能的值: avg1 (1分钟平均值, 默认值), avg5, avg15 | 示例: ⇒ system.cpu.util[0,user,avg5]  旧名称: system.cpu.idleX, system.cpu.niceX, system.cpu.systemX, system.cpu.userX |
|  | | | | |
| **system.hostname[<type>]** | | | | |
|  | 系统主机名。 | 字符串型 | **type** (仅Windows不得在其它系统上使用) - 可能的值: netbios (默认) 或者 host | 该值由Windows上的GetComputerName（）（对于netbios）或gethostname（）（用于主机）函数以及其它系统上的“hostname”命令获取。  返回值示例: Linux系统: ⇒ system.hostname → linux-w7x1 ⇒ system.hostname → www.zabbix.com Windows系统: ⇒ system.hostname → WIN-SERV2008-I6 ⇒ system.hostname[host] → Win-Serv2008-I6LonG  参数type Zabbix **1.8.6**开始支持。  请参考 [更详细的描述](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/install/windows_agent#configuration). |
|  | | | | |
| **system.hw.chassis[<info>]** | | | | |
|  | 机架信息。 | 字符串 | **info** - 完整的 (默认)、型号、序列、类型或供应商之一 | 示例: system.hw.chassis[full] Hewlett-Packard HP Pro 3010 Small Form Factor PC CZXXXXXXXX Desktop]  此key取决于[SMBIOS](http://en.wikipedia.org/wiki/System_Management_BIOS)表的可用性。 将尝试从sysfs读取DMI表，如果sysfs访问失败，尝试直接从内存中读取。  需要根权限，因为通过从sysfs或内存读取获取该值。  Zabbix agent从2.0开始支持。 |
|  | | | | |
| **system.hw.cpu[<cpu>,<info>]** | | | | |
|  | CPU信息 | 字符串或者整型 | **cpu** - <CPU数量> 或者 全部 (默认) **info** - 可能的值： full (默认), curfreq, maxfreq, model 或者vendor | 示例: ⇒ system.hw.cpu[0,vendor] → AuthenticAMD  从 /proc/cpuinfo 和 /sys/devices/system/cpu/[cpunum]/cpufreq/cpuinfo\_max\_freq 获取信息。  如果指定了CPU编号和curfreq或maxfreq，则返回数值（Hz）。  Zabbix agent从版本2.0开始支持 |
|  | | | | |
| **system.hw.devices[<type>]** | | | | |
|  | 列出PCI或者USB设备 | 文本型 | **type** - pci (默认) 或者 usb | 示例: ⇒ system.hw.devices[pci] → 00:00.0 Host bridge: Advanced Micro Devices [AMD] RS780 Host Bridge [..]  返回lspci或lsusb实用程序的输出（没有任何参数）  Zabbix agent 从版本2.0开始支持 |
|  | | | | |
| **system.hw.macaddr[<interface>,<format>]** | | | | |
|  | 列出MAC地址 | 字符串型 | **interface** - all (默认) 或者为一个正则表达式 **format** - full (默认) 或者 short | 列出与给定接口正则表达式名称匹配的网卡的MAC地址（所有网卡的所有列表）。  示例: ⇒ system.hw.macaddr["eth0$",full] → [eth0] 00:11:22:33:44:55  如果format被指定为short，则不会列出接口名称和相同的MAC地址。 \\Zabbix agent从版本2.0开始支持。 |
|  | | | | |
| **system.localtime[<type>]** | | | | |
|  | 系统时间 | 整数 - 类型 为 utc  字符串 - 类型 为 local | **type** - 可能的值: utc - (默认值) 从纪元以来的时间（1970年1月1日00:00:00 UTC），以秒为单位。\ local - 'yyyy-mm-dd，hh：mm：ss.nnn，+ hh：mm'格式的时间 | 此监控项参数从Zabbix agent 版本2.0开始支持。  示例: ⇒ system.localtime[local] → 使用该key创建一个监控项，然后使用它在“时钟”[screen element](https://www.zabbix.com/documentation/3.4/zh/manual/config/visualisation/screens/elements#clock)中显示主机时间。 |
|  | | | | |
| **system.run[command,<mode>]** | | | | |
|  | 在主机上运行指定的命令。 | 命令执行的文本结果  1 - 模式为nowait（不管命令结果如何） | **command** - 要执行的命令 **mode** - 可能的值: wait - 等待执行结束（默认）， nowait - 不等待 | 最多可以返回512KB的数据，包括截断的尾随空格。 要被正确的处理，命令的输出必须是文本。  示例: ⇒ system.run[ls -l /] → 根目录的详细文件列表。  注意: 要启用此功能，[Zabbix agent配置文件](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/config/zabbix_agentd) 必须包含EnableRemoteCommands=1 选项。 注意: 监控项的返回值是标准输出以及由命令产生的标准错误输出。 如果没有使用nowait标志，则会[检查](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/command_execution#exit_code_checking)执行结果。 注意: 从Zabbix 2.4.0开始，空结果是允许的。  同时参考: [执行指令](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/command_execution). |
|  | | | | |
| **system.stat[resource,<type>]** | | | | |
|  | 系统信息。 | 整型或者浮点型 | **ent** - 该分区有权接收的处理器单元数（float） **kthr,<type>** - 关于内核线程状态的信息: r - 平均可运行内核线程数（float） b - 虚拟内存管理器等待队列中的平均内核线程数（float） **memory,<type>** - 有关虚拟和真实内存使用情况的信息： avm - 活动虚拟页面（整数） fre - 自由列表的大小（整数） **page,<type>** - 关于页面错误和分页活动的信息： fi - 每秒文件页面输入（float） fo - 每秒文件页面输出（float） pi - 从调页空间（float）分页的页面 po - 页面分页到调页空间（float） fr - 页面被释放（页面替换）（浮点） sr - 通过页面替换算法扫描的页面 (float) **faults,<type>** - trap和中断率: in - 设备中断 (float) sy - 系统调用 (float) cs - 内核线程上下文切换 (float) **cpu,<type>** - 处理器时间使用百分比的细分: us - 用户时间 (float) sy - 系统时间 (float) id - 空闲时间 (float) wa - 系统具有未完成的磁盘/NFS I/O请求（float）的空闲请求时间(float) pc - 消耗的物理处理器数量(float) ec - 被授权的容量消耗的百分比(float) lbusy - 表示在用户和系统级执行时发生的逻辑处理器利用率的百分比(float) app - 表示共享池中的可用物理处理器(float) **disk,<type>** - 磁盘信息: bps - 表示以每秒字节为单位传输（读取或写入）驱动器的数据量 (integer) tps - 表示发送到物理磁盘/磁带的每秒传输次数(float) 此监控项从Zabbix **1.8.1.** 开始支持 | |
|  | | | | |
| **system.sw.arch** | | | | |
|  | 软件架构信息。 | 字符串型 |  | 示例: ⇒ system.sw.arch → i686  信息从uname（）函数中获取。  Zabbix agent 从版本2.0开始支持 |
|  | | | | |
| **system.sw.os[<info>]** | | | | |
|  | 操作系统信息 | 字符串 | **info** - 可能的值:  full (默认), short 或者 name | 示例: ⇒ system.sw.os[short]→ Ubuntu 2.6.35-28.50-generic 2.6.35.11  信息获取（注意，并非所有发行版中都存在所有文件和选项）： /proc/version (full) /proc/version\_signature (short) /etc/os-release中支持它的系统上的PRETTY\_NAME参数，或/etc/issue.net（name）  Zabbix agent从版本2.0开始支持。 |
|  | | | | |
| **system.sw.packages[<package>,<manager>,<format>]** | | | | |
|  | 列出已安装的软件包。 | 文本 | **package** - all (默认) 或者为正则表达式 **manager** - all (默认) 或者为包管理器 **format** - full (默认) 或者 short | 列表（按字母顺序）安装的包名称与给定的包regexp匹配的包（全部列出它们全部）。  示例: ⇒ system.sw.packages[mini,dpkg,short] → python-minimal, python2.6-minimal, ubuntu-minimal  支持包管理器(执行命令): dpkg (dpkg --get-selections) pkgtool (ls /var/log/packages) rpm (rpm -qa) pacman (pacman -Q)  如果format被指定为full，则软件包由包管理器分组（每个管理器在单独的行上以其方括号开头）。 如果format 被指定为short, 包管理器不分组，并列在一行里。  Zabbix agent从版本2.0开始支持 |
|  | | | | |
| **system.swap.in[<device>,<type>]** | | | | |
|  | 交换（从设备到内存）统计。 | 整型 | **device** - 用于交换的设备 (默认是all) **type** - 可能的值: count (swapins的数量), sectors (换入的区域), pages (换入的页). 有关默认的详细信息请参考 [支持的平台](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/supported_by_platform) | 示例: ⇒ system.swap.in[,pages]  这个信息的来源是: /proc/swaps, /proc/partitions, /proc/stat (Linux 2.4) /proc/swaps, /proc/diskstats, /proc/vmstat (Linux 2.6) |
|  | | | | |
| **system.swap.out[<device>,<type>]** | | | | |
|  | 交换（从内存到设备）统计。 | 整型 | **device** - 用于交换的设备 (默认是all) **type** - 可能的值: count (swapouts的数量), sectors (换出的区域), pages (换出的页). 有关默认的详细信息请参考[支持的平台](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/supported_by_platform) | 示例: ⇒ system.swap.out[,pages]  信息来源是: /proc/swaps, /proc/partitions, /proc/stat (Linux 2.4) /proc/swaps, /proc/diskstats, /proc/vmstat (Linux 2.6) |
|  | | | | |
| **system.swap.size[<device>,<type>]** | | | | |
|  | 交换空间大小（以字节为单位）或百分比（total）。 | Integer - 字节  Float - 百分比 | **device** - 用于交换的设备 (默认是 all) **type** - 可能的值: free (可用的交换空间, 默认值), pfree (空闲交换空间，百分比), pused (使用交换空间，百分比), total(总交换空间), used (使用交换空间) | 如果没有指定设备，Zabbix代理只会考虑交换设备（文件），物理内存将被忽略。 例如，在Solaris系统上，swap -s命令包含一部分物理内存和交换设备（与swap -l不同）。  请注意，此key可能会报告虚拟化（VMware ESXi，VirtualBox）Windows平台上的百分比不正确。在这种情况下，使用perf\_counter [\700（\_Total）\702]键来获取正确的交换使用数据。  旧名称: system.swap.free, system.swap.total |
|  | | | | |
| **system.uname** | | | | |
|  | 系统相关信息 | 字符串 |  | 返回值的示例(Unix): FreeBSD localhost 4.2-RELEASE FreeBSD 4.2-RELEASE #0: Mon Nov i386  返回值示例(Windows): Windows ZABBIX-WIN 6.0.6001 Microsoft® Windows Server® 2008 Standard Service Pack 1 x86  从Zabbix 2.2.0开始在Unix上，该监控项的值是通过uname（）系统调用获得的。以前它是通过调用“uname -a”获得的。 此监控项的值可能与“uname -a”的输出不同，并且不包含基于其它来源输出的“uname -a”的信息。  从Zabbix 3.0开始的Windows系统上，该监控项的值是从Win32\_OperatingSystem和Win32\_Processor WMI类获取信息。以前它是从不稳定的Windows API和未记录的注册表项获得的。操作系统名称（包括版本）可能会被翻译成用户的显示语言。 在某些版本的Windows上，它包含商标符号和额外的空格。  请注意，在Windows上，该项目返回操作系统架构，而在Unix上则返回CPU架构。 |
|  | | | | |
| **system.uptime** | | | | |
|  | 系统正常运行时间（以秒为单位） | 整数 |  | 在[监控项配置](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/item#configuration)中, 使用 **s** 或者 **uptime** 单位来获取可读取的值。 |
|  | | | | |
| **system.users.num** | | | | |
|  | 已登录用户数 | 整数 |  | **who** 命令用于代理端获取该值。 |
|  | | | | |
| **vfs.dev.read[<device>,<type>,<mode>]** | | | | |
|  | 磁盘读取统计信息。 | 整数 - 类型 为 sectors, operations, bytes  Float - 类型 为 sps, ops, bps | **device** - 磁盘设备 (默认为 all)  **type** - 可能的值: sectors, operations, bytes, sps, ops, bps 必须指定此参数，因为各种操作系统的默认值不同。 sps, ops, bps 表示: sectors, operations, bytes per second, respectively. **mode** - 可能的值: avg1 (1分钟平均值, 默认), avg5, avg15. 此参数仅支持的类型为: sps, ops, bps. | 不同操作系统的“类型”参数的默认值： AIX - operations FreeBSD - bps Linux - sps OpenBSD - operations Solaris - bytes  示例: ⇒ vfs.dev.read[,operations]  在支持的平台上的sps, ops 和 bps 曾被限制为 8 个设备(7个独立的和1个 all). 从Zabbix 2.0.1 开始，这个限制提高到1024个设备(1023个独立的和1个all).  如果默认为全部用于第一个参数，那么该key将返回摘要统计信息，包括所有块设备，如sda，sbd及其分区（sda1，sda2，sdb3 …）和基于这些块设备/分区的多个设备（MD raid）和基于这些设备/分区的逻辑卷（LVM）。在这种情况下，返回值只能作为相对值（动态时间）而不是绝对值。  LVM的支持从Zabbix 1.8.6开始。  直到Zabbix 1.8.6才能使用相关的设备名称（例如，sda）。 从那时起，可选的 /dev/前缀（例如/dev/sda）必须被使用。  旧名称: io[\*] |
|  | | | | |
| **vfs.dev.write[<device>,<type>,<mode>]** | | | | |
|  | 磁盘写入统计信息。 | 整数 - 类型 为sectors, operations, bytes  浮点型 - 类型 为 sps, ops, bps | **device** - 磁盘设备 (默认为all)  **type** - 可能的值: sectors, operations, bytes, sps, ops, bps 因为各种操作系统的默认值有所不同，所以这个参数必须被指定。 sps, ops, bps 代表: sectors, operations, bytes per second, respectively. **mode** - 可能的值: avg1 (1分钟平均值, 默认), avg5, avg15. 此参数仅支持这些类型: sps, ops, bps. | 不同操作系统的“类型”参数的默认值： AIX - operations FreeBSD - bps Linux - sps OpenBSD - operations Solaris - bytes  示例: ⇒ vfs.dev.write[,operations]  sps, ops and bps在支持的平台上的sps, ops 和 bps 曾被限制为 8 个设备(7个独立的和1个 all). 从Zabbix 2.0.1 开始，这个限制提高到1024个设备(1023个独立的和1个all)。  如果默认为全部用于第一个参数，那么该key将返回摘要统计信息，包括所有块设备，如sda，sbd及其分区（sda1，sda2，sdb3 …）和基于这些块设备/分区的多个设备（MD raid）和基于这些设备/分区的逻辑卷（LVM）。在这种情况下，返回值只能作为相对值（动态时间）而不是绝对值。  LVM的支持从Zabbix 1.8.6开始。  直到Zabbix 1.8.6才能使用相关的设备名称（例如，sda）。 从那时起，可选的 /dev/前缀（例如/dev/sda）必须被使用。  旧名字: io[\*] |
|  | | | | |
| **vfs.dir.size[dir,<regex\_incl>,<regex\_excl>,<mode>,<max\_depth>]** | | | | |
|  | 目录大小（以字节为单位）。 | 整数 | **dir** - 目录的绝对路径  **regex\_incl** - 正则表达式描述包含的文件名模式（如果为空则包括所有文件;空字符串是默认值） **regex\_excl** - 正则表达式描述用于排除的文件名模式（如果为空不排除任何文件;空字符串是默认值） **mode** - 可能的值: apparent (默认) - 获得明确的文件大小，而不是磁盘利用率(作为 du -sb dir), disk - 获取磁盘使用情况 (作为 du -s -B1 dir). 和du命令不同，vfs.dir.size 监控项在计算目录大小时会将隐藏的文件记录帐户 (作为 du -sb .[^.]\* \* 在 dir内).  **max\_depth** - 要遍历的子目录的最大深度。 **-1** (默认) - 无限, **0** - 不会遍历到子目录。 | 仅计算具有zabbix用户读取权限的目录。  示例:  ⇒ vfs.dir.size[/tmp,log] - 计算/tmp中包含“log”的所有文件的大小  ⇒ vfs.dir.size[/tmp,log,^.+\.old$] - 计算/tmp中包含“log”的所有文件的大小，不包括包含'.old'的文件   文件大小限制取决于[大文件支持](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/large_file_support)。 |
|  | | | | |
| **vfs.file.cksum[file]** | | | | |
|  | 文件checksum校验，由UNIX cksum算法计算实现。 | 整型 | **file** - 文件全路径 | 示例: ⇒ vfs.file.cksum[/etc/passwd]  返回值示例: 1938292000  旧名字: cksum  文件大小限制取决于[大文件支持](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/large_file_support)。 |
|  | | | | |
| **vfs.file.contents[file,<encoding>]** | | | | |
|  | 检索文件的内容。 | 文本 | **file** - 文件全路径  **encoding** - 编码页 [标识符](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/zabbix_agent#available_encodings) | 如果文件为空或仅包含LF/CR字符，则返回空字符串。  Example: ⇒ vfs.file.contents[/etc/passwd]  此选项对文件限制是不超过64KB的文件。  Zabbix agent从版本2.0开始支持。 |
|  | | | | |
| **vfs.file.exists[file]** | | | | |
|  | 检测文件是否存在。 | 0 - 不存在   1 - 常规文件或到常规存在文件的link（符号或硬） | **file** - 文件的全路径 | 示例: ⇒ vfs.file.exists[/tmp/application.pid]  返回值取决于S\_ISREG POSIX宏返回的值。  文件大小限制取决于 [大文件支持](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/large_file_support)。 |
|  | | | | |
| **vfs.file.md5sum[file]** | | | | |
|  | 文件的MD5 checksum。 | 字符串(文件的MD5哈希) | **file** - 文件的全路径 | 示例: ⇒ vfs.file.md5sum[/usr/local/etc/zabbix\_agentd.conf]  返回值示例: b5052decb577e0fffd622d6ddc017e82  此项目的文件大小限制（64 MB）在1.8.6版中已删除。  文件大小限制取决于 [大文件支持](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/large_file_support)。 |
|  | | | | |
| **vfs.file.regexp[file,regexp,<encoding>,<start line>,<end line>,<output>]** | | | | |
|  | 查找文件中的字符串。 | 包含匹配字符串的行，或由可选输出参数指定的行。 | **file** - 文件完整路径  **regexp** - GNU正则表达式 **encoding** - 编码页 [标识符](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/zabbix_agent#available_encodings) **start line** - 满足查询到的第一行的数量（默认为文件的第1行）。 **end line** - 要查询的最后一行的数量（默认为文件的最后一行）。 **output** - 一个可选的输出格式模板。 \0转义序列替换为匹配的文本，而\N（其中N = 1 … 9）转义序列被替换为第N个匹配组（如果N超过捕获组的数量，则为空字符串）。 | 只返回第一个匹配行。\\如果没有行与表达式匹配，则返回空字符串。  使用输出参数的提取过程发生在代理端。  start line, end line 和output 参数从版本2.2开始支持。  示例: ⇒ 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] → 获取用户Zabbix的ID |
|  | | | | |
| **vfs.file.regmatch[file,regexp,<encoding>,<start line>,<end line>]** | | | | |
|  | 查询文件中的字符串。 | 0 - 不匹配  1 - 匹配 | **file** - 文件全路径  **regexp** - GNU正则表达式 **encoding** - 编码页 [标识符](https://www.zabbix.com/documentation/3.4/zh/manual/config/items/itemtypes/zabbix_agent#available_encodings) **start line** - 满足查询到的第一行的数量（默认为文件的第1行）。 **end line** - 要查询的最后一行的数量（默认为文件的最后一行）。 | start line, end line 和output 参数从版本2.2开始支持。  示例: ⇒ vfs.file.regmatch[/var/log/app.log,error] |
|  | | | | |
| **vfs.file.size[file]** | | | | |
|  | 文件大小(按字节)。 | 整数 | **file** - 文件全路径 | 文件必须具有Zabbix用户读的权限。  示例: ⇒ vfs.file.size[/var/log/syslog]  文件大小限制取决于 [大文件支持](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/large_file_support)。 |
|  | | | | |
| **vfs.file.time[file,<mode>]** | | | | |
|  | 文件时间信息。 | 整数(Unix时间戳) | **file** - 文件全路径 **mode** - 可能的值: modify (默认) - 更新时间, access - 最后一次访问时间, change - 最后一次修改时间 | 示例: ⇒ vfs.file.time[/etc/passwd,modify]  文件大小限制取决于 [大文件支持](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/large_file_support)。 |
|  | | | | |
| **vfs.fs.discovery** | | | | |
|  | 挂载的文件系统列表。用于低级发现。 | JSON对象 |  | Zabbix agent从版本2.0开始支持。  Zabbix agent从3.0开始支持{#FSDRIVETYPE} 宏 |
|  | | | | |
| **vfs.fs.inode[fs,<mode>]** | | | | |
|  | inode的数量或百分比。 | 整型 - 针对数量  浮点值 - 针对百分比 | **fs** - 文件系统 **mode** - 可能的值:  total (默认), free, used, pfree (剩余, 百分比), pused(已用的, 百分比) | 示例: ⇒ vfs.fs.inode[/,pfree]   旧名字: vfs.fs.inode.free[\*], vfs.fs.inode.pfree[\*], vfs.fs.inode.total[\*] |
|  | | | | |
| **vfs.fs.size[fs,<mode>]** | | | | |
|  | 磁盘空间，以字节为单位，用百分比表示。 | 整数 - 针对字节  浮点- 针对百分比 | **fs** - 文件系统 **mode** - 可能的值:  total (默认), free, used, pfree (剩余, 百分比), pused(已用, 百分比) | 在安装卷的情况下，返回本地文件系统的磁盘空间。  示例: ⇒ vfs.fs.size[/tmp,free]  考虑到文件系统的保留空间，并且在使用自由模式时不包括。  旧名称: vfs.fs.free[\*], vfs.fs.total[\*], vfs.fs.used[\*], vfs.fs.pfree[\*], vfs.fs.pused[\*] |
|  | | | | |
| **vm.memory.size[<mode>]** | | | | |
|  | 内存大小，以字节为单位，以百分比表示。 | 整数 - 用于字节  浮点 - 用于百分比 | **mode** - 可能的值:  total (默认), active, anon, buffers, cached, exec, file, free, inactive, pinned, shared, wired, used, pused (已用, 百分比), available, pavailable (可用, 百分比) | 此监控项接受3类参数:  1) total - 总内存量; 2) platform-特定内存类型: active, anon, buffers, cached, exec, file, free, inactive, pinned, shared, wired; 3) user-用户级别估计使用和可用的内存量: used, pused, available, pavailable.  请参阅vm.memory.size的了解更详细的[参数](https://www.zabbix.com/documentation/3.4/zh/manual/appendix/items/vm.memory.size_params)描述.  旧名称: vm.memory.buffers, vm.memory.cached, vm.memory.free, vm.memory.shared, vm.memory.total |
|  | | | | |
| **web.page.get[host,<path>,<port>]** | | | | |
|  | 获取网页内容。 | 网页源码 | **host** - 主机名 **path** - HTML文档路径 (默认是/) **port** - 端口号 (默认是80) | 失败时返回一个空字符串。   示例: ⇒ web.page.get[www.zabbix.com,index.php,80] |
|  | | | | |
| **web.page.perf[host,<path>,<port>]** | | | | |
|  | 加载完整网页的时间（以秒为单位）。 | 浮点 | **host** - 主机名 **path** - HTML文档的路径 (默认是/) **port** - 端口号(默认是80) | 失败时返回一个空字符串。   示例: ⇒ web.page.perf[www.zabbix.com,index.php,80] |
|  | | | | |
| **web.page.regexp[host,<path>,<port>,<regexp>,<length>,<output>]** | | | | |
|  | 在网页上查找字符串。 | 匹配的字符串，或由可选的“输出”参数指定 | **host** - 主机名 **path** - HTML文档路径(默认是/) **port** - 端口号(默认是80) **regexp** - GNU正则表达式 **length** - 返回的最大字符数 **output** - 一个可选的输出格式模板。 \0转义序列替换为匹配的文本，而\N（其中N = 1 … 9）转义序列被替换为第N个匹配组（如果N超过捕获组的数量，则为空字符串）。 | 如果找不到匹配或失败，则返回一个空字符串。  使用output参数的内容在代理端上进行提取。  参数output从版本2.2开始支持   示例: ⇒ web.page.regexp[www.zabbix.com,index.php,80,OK,2] |