

1 Zabbix agent

Overview

These checks use the communication with Zabbix agent for data gathering.

There are [passive](#) and [active](#) agent checks. When configuring an item, you can select the required type:

- *Zabbix agent* - for passive checks
- *Zabbix agent (active)* - for active checks

Supported item keys

The table provides details on the item keys that you can use with Zabbix agent items.

See also:

- [Items supported by platform](#)
- [Item keys specific for Windows agent](#)

Mandatory and optional parameters

Parameters without angle brackets are mandatory. Parameters marked with angle brackets < > are optional.

Key			
Description	Return value	Parameters	Comments
agent.hostname			
Agent host name.	String		Returns the actual value of the agent hostname from a configuration file.
agent.ping			
Agent availability check.	Nothing - unavailable 1 - available		Use the nodata() trigger function to check for host unavailability.
agent.version			
Version of Zabbix agent.	String		Example of returned value: 1.8.2
kernel.maxfiles			
Maximum number of opened files supported by OS.	Integer		
kernel.maxproc			
Maximum number of processes supported by OS.	Integer		
log[file,<regex>,<encoding>,<maxlines>,<mode>,<output>,<maxdelay>]			

Key			
Description	Return value	Parameters	Comments
Log file monitoring.	Log	<p>file - full path and name of log file</p> <p>regexp - regular expression describing the required pattern</p> <p>encoding - code page identifier</p> <p>maxlines - maximum number of new lines per second the agent will send to Zabbix server or proxy. This parameter overrides the value of 'MaxLinesPerSecond' in zabbix_agentd.conf</p> <p>mode - possible values: <i>all</i> (default), <i>skip</i> - skip processing of older data (affects only newly created items).</p> <p>output - an optional output formatting template. The <code>\0</code> escape sequence is replaced with the matched text while an <code>\N</code> (where <code>N=1..9</code>) escape sequence is replaced with Nth matched group (or an empty string if the N exceeds the number of captured groups).</p> <p>maxdelay - maximum delay in seconds. Type: float. Values: 0 - (default) never ignore log file lines; > 0.0 - ignore older lines in order to get the most recent lines analyzed within "maxdelay" seconds. Read the maxdelay notes before using it!</p>	<p>The item must be configured as an active check. If file is missing or permissions do not allow access, item turns unsupported.</p> <p>If output is left empty - the whole line containing the matched text is returned. Note that all global regular expression types except 'Result is TRUE' always return the whole matched line and the output parameter is ignored.</p> <p>Content extraction using the output parameter takes place on the agent.</p> <p>Examples: ⇒ <code>log[/var/log/syslog]</code> ⇒ <code>log[/var/log/syslog,error]</code> ⇒ <code>log[/home/zabbix/logs/logfile,,,100]</code></p> <p>The mode parameter is supported since Zabbix 2.0. The output parameter is supported since Zabbix 2.2. The maxdelay parameter is supported since Zabbix 3.2.</p> <p>See also additional information on log monitoring.</p>
log.count[file,<regexp>,<encoding>,<maxproclines>,<mode>,<maxdelay>]			
Count of matched lines in log file monitoring.	Integer	<p>file - full path and name of log file</p> <p>regexp - regular expression describing the required pattern</p> <p>encoding - code page identifier</p> <p>maxproclines - maximum number of new lines per second the agent will analyze. Default value is 4*'MaxLinesPerSecond' in zabbix_agentd.conf.</p> <p>mode - possible values: <i>all</i> (default), <i>skip</i> - skip processing of older data (affects only newly created items).</p> <p>maxdelay - maximum delay in seconds. Type: float. Values: 0 - (default) never ignore log file lines; > 0.0 - ignore older lines in order to get the most recent lines analyzed within "maxdelay" seconds. Read the maxdelay notes before using it!</p>	<p>The item must be configured as an active check. If file is missing or permissions do not allow access, item turns unsupported.</p> <p>See also additional information on log monitoring.</p> <p>Supported since Zabbix 3.2.0.</p>
logrt[file_regexp,<regexp>,<encoding>,<maxlines>,<mode>,<output>,<maxdelay>]			

Key			
Description	Return value	Parameters	Comments
Log file monitoring with log rotation support.	Log	<p>file_regexp - absolute path to file and regexp describing the file name pattern</p> <p>regexp - regular expression describing the required content pattern</p> <p>encoding - code page identifier</p> <p>maxlines - maximum number of new lines per second the agent will send to Zabbix server or proxy. This parameter overrides the value of 'MaxLinesPerSecond' in zabbix_agentd.conf</p> <p>mode - possible values: <i>all</i> (default), <i>skip</i> - skip processing of older data (affects only newly created items).</p> <p>output - an optional output formatting template. The <code>\0</code> escape sequence is replaced with the matched text while an <code>\N</code> (where <code>N=1...9</code>) escape sequence is replaced with Nth matched group (or an empty string if the <code>N</code> exceeds the number of captured groups).</p> <p>maxdelay - maximum delay in seconds. Type: float. Values: 0 - (default) never ignore log file lines; > 0.0 - ignore older lines in order to get the most recent lines analyzed within "maxdelay" seconds. Read the maxdelay notes before using it!</p>	<p>The item must be configured as an active check. Log rotation is based on the last modification time of files.</p> <p>If output is left empty - the whole line containing the matched text is returned. Note that all global regular expression types except 'Result is TRUE' always return the whole matched line and the output parameter is ignored.</p> <p>Content extraction using the output parameter takes place on the agent.</p> <p>Examples: ⇒ <code>logrt["/home/zabbix/logs/^logfile[0-9]{1,3}\$",,,100]</code> → will match a file like "logfile1" (will not match ".logfile1") ⇒ <code>logrt["/home/user/^logfile._*[0-9]{1,3}\$","pattern_to_match","UTF-8",100]</code> → will collect data from files such "logfile_abc_1" or "logfile_001".</p> <p>The mode parameter is supported since Zabbix 2.0. The output parameter is supported since Zabbix 2.2. The maxdelay parameter is supported since Zabbix 3.2.</p> <p>See also additional information on log monitoring.</p>
logrt.count[file_regexp,<regexp>,<encoding>,<maxproclines>,<mode>,<maxdelay>]			
Count of matched lines in log file monitoring with log rotation support.	Integer	<p>file_regexp - absolute path to file and regexp describing the file name pattern</p> <p>regexp - regular expression describing the required content pattern</p> <p>encoding - code page identifier</p> <p>maxproclines - maximum number of new lines per second the agent will analyze. Default value is 4*'MaxLinesPerSecond' in zabbix_agentd.conf.</p> <p>mode - possible values: <i>all</i> (default), <i>skip</i> - skip processing of older data (affects only newly created items).</p> <p>maxdelay - maximum delay in seconds. Type: float. Values: 0 - (default) never ignore log file lines; > 0.0 - ignore older lines in order to get the most recent lines analyzed within "maxdelay" seconds. Read the maxdelay notes before using it!</p>	<p>The item must be configured as an active check. Log rotation is based on the last modification time of files.</p> <p>See also additional information on log monitoring.</p> <p>Supported since Zabbix 3.2.0.</p>
net.dns[<ip>,<name>,<type>,<timeout>,<count>,<protocol>]			
Checks if DNS service is up.	<p>0 - DNS is down (server did not respond or DNS resolution failed)</p> <p>1 - DNS is up</p>	<p>ip - IP address of DNS server (leave empty for the default DNS server, ignored on Windows)</p> <p>name - DNS name to query</p> <p>type - record type to be queried (default is <i>SOA</i>)</p> <p>timeout (ignored on Windows) - timeout for the request in seconds (default is 1 second)</p> <p>count (ignored on Windows) - number of tries for the request (default is 2)</p> <p>protocol - the protocol used to perform DNS queries: <i>udp</i> (default) or <i>tcp</i></p>	<p>Example: ⇒ <code>net.dns[8.8.8.8,zabbix.com,MX,2,1]</code></p> <p>The possible values for type are: <i>ANY, A, NS, CNAME, MB, MG, MR, PTR, MD, MF, MX, SOA, NULL, WKS</i> (except for Windows), <i>HINFO, MINFO, TXT, SRV</i></p> <p>Internationalized domain names are not supported, please use IDNA encoded names instead.</p> <p>The protocol parameter is supported since Zabbix 3.0. SRV record type is supported since Zabbix agent versions 1.8.6 (Unix) and 2.0.0 (Windows).</p> <p>Naming before Zabbix 2.0 (still supported): <code>net.tcp.dns</code></p>
net.dns.record[<ip>,<name>,<type>,<timeout>,<count>,<protocol>]			

Key			
Description	Return value	Parameters	Comments
Performs a DNS query.	Character string with the required type of information	ip - IP address of DNS server (leave empty for the default DNS server, ignored on Windows) name - DNS name to query type - record type to be queried (default is SOA) timeout (ignored on Windows) - timeout for the request in seconds (default is 1 second) count (ignored on Windows) - number of tries for the request (default is 2) protocol - the protocol used to perform DNS queries: <i>udp</i> (default) or <i>tcp</i>	Example: ⇒ <code>net.dns.record[8.8.8.8,zabbix.com,MX,2,1]</code> The possible values for type are: <i>ANY, A, NS, CNAME, MB, MG, MR, PTR, MD, MF, MX, SOA, NULL, WKS</i> (except for Windows), <i>HINFO, MINFO, TXT, SRV</i> Internationalized domain names are not supported, please use IDNA encoded names instead. The protocol parameter is supported since Zabbix 3.0. SRV record type is supported since Zabbix agent versions 1.8.6 (Unix) and 2.0.0 (Windows). Naming before Zabbix 2.0 (still supported): <code>net.tcp.dns.query</code>
net.if.collisions[if]			
Number of out-of-window collisions.	Integer	if - network interface name	
net.if.discovery			
List of network interfaces. Used for low-level discovery.	JSON object		Supported since Zabbix agent version 2.0. On FreeBSD, OpenBSD and NetBSD supported since Zabbix agent version 2.2. Some Windows versions (for example, Server 2008) might require the latest updates installed to support non-ASCII characters in interface names.
net.if.in[if,<mode>]			
Incoming traffic statistics on network interface.	Integer	if - network interface name (Unix); network interface full description or IPv4 address (Windows) mode - possible values: <i>bytes</i> - number of bytes (default) <i>packets</i> - number of packets <i>errors</i> - number of errors <i>dropped</i> - number of dropped packets	On Windows, the item gets values from 64-bit counters if available. 64-bit interface statistic counters were introduced in Windows Vista and Windows Server 2008. If 64-bit counters are not available, the agent uses 32-bit counters. Multi-byte interface names on Windows are supported since Zabbix agent version 1.8.6. Examples: ⇒ <code>net.if.in[eth0,errors]</code> ⇒ <code>net.if.in[eth0]</code> You may obtain network interface descriptions on Windows with <code>net.if.discovery</code> or <code>net.if.list</code> items. You may use this key with a <i>Delta (speed per second)</i> store value in order to get bytes per second statistics.
net.if.out[if,<mode>]			
Outgoing traffic statistics on network interface.	Integer	if - network interface name (Unix); network interface full description or IPv4 address (Windows) mode - possible values: <i>bytes</i> - number of bytes (default) <i>packets</i> - number of packets <i>errors</i> - number of errors <i>dropped</i> - number of dropped packets	On Windows, the item gets values from 64-bit counters if available. 64-bit interface statistic counters were introduced in Windows Vista and Windows Server 2008. If 64-bit counters are not available, the agent uses 32-bit counters. Multi-byte interface names on Windows are supported since Zabbix agent 1.8.6 version. Examples: ⇒ <code>net.if.out[eth0,errors]</code> ⇒ <code>net.if.out[eth0]</code> You may obtain network interface descriptions on Windows with <code>net.if.discovery</code> or <code>net.if.list</code> items. You may use this key with a <i>Delta (speed per second)</i> store value in order to get bytes per second statistics.
net.if.total[if,<mode>]			

Key			
Description	Return value	Parameters	Comments
Sum of incoming and outgoing traffic statistics on network interface.	Integer	if - network interface name (Unix); network interface full description or IPv4 address (Windows) mode - possible values: <i>bytes</i> - number of bytes (default) <i>packets</i> - number of packets <i>errors</i> - number of errors <i>dropped</i> - number of dropped packets	On Windows, the item gets values from 64-bit counters if available. 64-bit interface statistic counters were introduced in Windows Vista and Windows Server 2008. If 64-bit counters are not available, the agent uses 32-bit counters. Examples: ⇒ 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. You may use this key with a <i>Delta (speed per second)</i> store value in order to get bytes per second statistics. Note that dropped packets are supported only if both net.if.in and net.if.out work for dropped packets on your platform.
net.tcp.listen[port]			
Checks if this TCP port is in LISTEN state.	0 - it is not in LISTEN state 1 - it is in LISTEN state	port - TCP port number	Example: ⇒ net.tcp.listen[80] On Linux supported since Zabbix agent version 1.8.4 Since Zabbix 3.0.0, on Linux kernels 2.6.14 and above, information about listening TCP sockets is obtained from the kernel's NETLINK interface, if possible. Otherwise, the information is retrieved from /proc/net/tcp and /proc/net/tcp6 files.
net.tcp.port[<ip>,<port>]			
Checks if it is possible to make TCP connection to specified port.	0 - cannot connect 1 - can connect	ip - IP address (default is 127.0.0.1) port - port number	Example: ⇒ net.tcp.port[,80] → can be used to test availability of web server running on port 80. For simple TCP performance testing use net.tcp.service.perf[tcp,<ip>,<port>] Note that these checks may result in additional messages in system daemon logfiles (SMTP and SSH sessions being logged usually). Old naming: <i>check_port[*]</i>
net.tcp.service[service,<ip>,<port>]			
Checks if service is running and accepting TCP connections.	0 - service is down 1 - service is running	service - either of: <i>ssh, ldap, smtp, ftp, http, pop, nntp, imap, tcp, https, telnet</i> (see details) ip - IP address (default is 127.0.0.1) port - port number (by default standard service port number is used)	Example: ⇒ net.tcp.service[ftp,,45] → can be used to test the availability of FTP server on TCP port 45. Note that these checks may result in additional messages in system daemon logfiles (SMTP and SSH sessions being logged usually). Checking of encrypted protocols (like IMAP on port 993 or POP on port 995) is currently not supported. As a workaround, please use net.tcp.port for checks like these. Checking of LDAP and HTTPS by Windows agent is currently not supported. Note that the telnet check looks for a login prompt (':' at the end). See also known issues of checking HTTPS service. <i>https</i> and <i>telnet</i> services are supported since Zabbix 2.0. Old naming: <i>check_service[*]</i>
net.tcp.service.perf[service,<ip>,<port>]			

Key			
Description	Return value	Parameters	Comments
Checks performance of TCP service.	0 - service is down seconds - the number of seconds spent while connecting to the service	service - either of: <i>ssh, ldap, smtp, ftp, http, pop, nntp, imap, tcp, https, telnet</i> (see details) ip - IP address (default is 127.0.0.1) port - port number (by default standard service port number is used)	Example: ⇒ <code>net.tcp.service.perf[ssh]</code> → can be used to test the speed of initial response from SSH server. Checking of encrypted protocols (like IMAP on port 993 or POP on port 995) is currently not supported. As a workaround, please use <code>net.tcp.service.perf[tcp,<ip>,<port>]</code> for checks like these. Checking of LDAP and HTTPS by Windows agent is currently not supported. Note that the telnet check looks for a login prompt (':' at the end). See also known issues of checking HTTPS service. <i>https</i> and <i>telnet</i> services are supported since Zabbix 2.0. Old naming: <code>check_service_perf[*]</code>
net.udp.listen[port]			
Checks if this UDP port is in LISTEN state.	0 - it is not in LISTEN state 1 - it is in LISTEN state	port - UDP port number	Example: ⇒ <code>net.udp.listen[68]</code> On Linux supported since Zabbix agent version 1.8.4
net.udp.service[service,<ip>,<port>]			
Checks if service is running and responding to UDP requests.	0 - service is down 1 - service is running	service - <i>ntp</i> (see details) ip - IP address (default is 127.0.0.1) port - port number (by default standard service port number is used)	Example: ⇒ <code>net.udp.service[ntp,,45]</code> → can be used to test the availability of NTP service on UDP port 45. This item is supported since Zabbix 3.0.0, but <i>ntp</i> service was available for <code>net.tcp.service[]</code> item in prior versions.
net.udp.service.perf[service,<ip>,<port>]			
Checks performance of UDP service.	0 - service is down seconds - the number of seconds spent waiting for response from the service	service - <i>ntp</i> (see details) ip - IP address (default is 127.0.0.1) port - port number (by default standard service port number is used)	Example: ⇒ <code>net.udp.service.perf[ntp]</code> → can be used to test response time from NTP service. This item is supported since Zabbix 3.0.0, but <i>ntp</i> service was available for <code>net.tcp.service[]</code> item in prior versions.
proc.cpu.util[<name>,<user>,<type>,<cmdline>,<mode>,<zone>]			
Process CPU utilisation percentage.	Float	name - process name (default is <i>all processes</i>) user - user name (default is <i>all users</i>) type - CPU utilisation type: <i>total</i> (default), <i>user</i> , <i>system</i> cmdline - filter by command line (it is a regular expression) mode - data gathering mode: <i>avg1</i> (default), <i>avg5</i> , <i>avg15</i> zone - target zone: <i>current</i> (default), <i>all</i> . This parameter is supported only on Solaris platform. Since Zabbix 3.0.3 if agent has been compiled on Solaris without zone support but is running on a newer Solaris where zones are supported and <code><zone></code> parameter is default or <i>current</i> then the agent will return NOTSUPPORTED (the agent cannot limit results to only current zone). However, <code><zone></code> parameter value <i>all</i> is supported in this case.	Examples: ⇒ <code>proc.cpu.util[,root]</code> → CPU utilisation of all processes running under the "root" user ⇒ <code>proc.cpu.util[zabbix_server,zabbix]</code> → CPU utilisation of all <i>zabbix_server</i> processes running under the <i>zabbix</i> user The returned value is based on single CPU core utilisation percentage. For example CPU utilisation of a process fully using two cores is 200%. The process CPU utilisation data is gathered by a collector which supports the maximum of 1024 unique (by name, user and command line) queries. Queries not accessed during the last 24 hours are removed from the collector. This key is supported since Zabbix 3.0.0 and is available on several platforms (see Items supported by platform).

Key			
Description	Return value	Parameters	Comments
proc.mem[<name>,<user>,<mode>,<cmdline>,<memtype>]			
Memory used by process in bytes.	Integer - with mode as <i>max</i> , <i>min</i> , <i>sum</i> Float - with mode as <i>avg</i>	name - process name (default is <i>all processes</i>) user - user name (default is <i>all users</i>) mode - possible values: <i>avg</i> , <i>max</i> , <i>min</i> , <i>sum</i> (default) cmdline - filter by command line (it is a regular expression) memtype - <i>type of memory</i> used by process	Examples: ⇒ <code>proc.mem[,root]</code> → memory used by all processes running under the "root" user ⇒ <code>proc.mem[zabbix_server,zabbix]</code> → memory used by all zabbix_server processes running under the zabbix user ⇒ <code>proc.mem[,oracle,max,oracleZABBIX]</code> → memory used by the most memory-hungry process running under oracle having oracleZABBIX in its command line <i>Note:</i> When several processes use shared memory, the sum of memory used by processes may result in large, unrealistic values. See notes on selecting processes with name and <code>cmdline</code> parameters (Linux-specific). The <code>memtype</code> parameter is supported on several platforms since Zabbix 3.0.0.
proc.num[<name>,<user>,<state>,<cmdline>]			
The number of processes.	Integer	name - process name (default is <i>all processes</i>) user - user name (default is <i>all users</i>) state - possible values: <i>all</i> (default), <i>run</i> , <i>sleep</i> , <i>zomb</i> cmdline - filter by command line (it is a regular expression)	Examples: ⇒ <code>proc.num[,mysql]</code> → number of processes running under the mysql user ⇒ <code>proc.num[apache2,www-data]</code> → number of apache2 processes running under the www-data user ⇒ <code>proc.num[,oracle,sleep,oracleZABBIX]</code> → number of processes in sleep state running under oracle having oracleZABBIX in its command line See notes on selecting processes with name and <code>cmdline</code> parameters (Linux-specific). On Windows, only the name and user parameters are supported.
sensor[device,sensor,<mode>]			
Hardware sensor reading.	Float	device - device name sensor - sensor name mode - possible values: <i>avg</i> , <i>max</i> , <i>min</i> (if this parameter is omitted, device and sensor are treated verbatim).	Reads <code>/proc/sys/dev/sensors</code> on Linux 2.4. Example: ⇒ <code>sensor[w83781d-i2c-0-2d,temp1]</code> Prior to Zabbix 1.8.4, the <code>sensor[temp1]</code> format was used. Reads <code>/sys/class/hwmon</code> on Linux 2.6+. See a more detailed description of sensor item on Linux. Reads the <code>hw.sensors</code> MIB on OpenBSD. Examples: ⇒ <code>sensor[cpu0,temp0]</code> → temperature of one CPU ⇒ <code>sensor["cpu[0-2]\$",temp,avg]</code> → average temperature of the first three CPU's Supported on OpenBSD since Zabbix 1.8.4.
system.boottime			
System boot time.	Integer (Unix timestamp)		
system.cpu.discovery			
List of detected CPUs/CPU cores. Used for low-level discovery.	JSON object		Supported on all platforms since 2.4.0.
system.cpu.intr			
Device interrupts.	Integer		
system.cpu.load[<cpu>,<mode>]			
CPU load.	Float	cpu - possible values: <i>all</i> (default), <i>percpu</i> (total load divided by online CPU count) mode - possible values: <i>avg1</i> (one-minute average, default), <i>avg5</i> , <i>avg15</i>	Example: ⇒ <code>system.cpu.load[,avg5]</code> <i>percpu</i> is supported since Zabbix 2.0.0. Old naming: <code>system.cpu.loadX</code>
system.cpu.num[<type>]			

Key			
Description	Return value	Parameters	Comments
Number of CPUs.	Integer	type - possible values: <i>online</i> (default), <i>max</i>	Example: ⇒ system.cpu.num
system.cpu.switches			
Count of context switches.	Integer		Old naming: <i>system[switches]</i>
system.cpu.util[<cpu>,<type>,<mode>]			
CPU utilisation percentage.	Float	cpu - <CPU number> or <i>all</i> (default) type - possible values: <i>idle</i> , <i>nice</i> , <i>user</i> (default), <i>system</i> (default for Windows), <i>iowait</i> , <i>interrupt</i> , <i>softirq</i> , <i>steal</i> , <i>guest</i> (on Linux kernels 2.6.24 and above), <i>guest_nice</i> (on Linux kernels 2.6.33 and above) mode - possible values: <i>avg1</i> (one-minute average, default), <i>avg5</i> , <i>avg15</i>	Example: ⇒ system.cpu.util[0,user,avg5] Old naming: <i>system.cpu.idleX</i> , <i>system.cpu.niceX</i> , <i>system.cpu.systemX</i> , <i>system.cpu.userX</i>
system.hostname[<type>]			
System host name.	String	type (Windows only, must not be used on other systems) - possible values: <i>netbios</i> (default) or <i>host</i>	The value is acquired by either <code>GetComputerName()</code> (for netbios) or <code>gethostname()</code> (for host) functions on Windows and by "hostname" command on other systems. Examples of returned values: <i>on Linux:</i> ⇒ system.hostname → linux-w7x1 ⇒ system.hostname → www.zabbix.com <i>on Windows:</i> ⇒ system.hostname → WIN-SERV2008-I6 ⇒ system.hostname[host] → Win-Serv2008-I6LonG The type parameter for this item is supported since Zabbix 1.8.6 . See also a more detailed description .
system.hw.chassis[<info>]			
Chassis information.	String	info - one of <i>full</i> (default), <i>model</i> , <i>serial</i> , <i>type</i> or <i>vendor</i>	Example: system.hw.chassis[full] Hewlett-Packard HP Pro 3010 Small Form Factor PC CZXXXXXXXX Desktop] This key depends on the availability of the SMBIOS table. Will try to read the DMI table from sysfs, if sysfs access fails then try reading directly from memory. Root permissions are required because the value is acquired by reading from sysfs or memory. Supported since Zabbix agent version 2.0.
system.hw.cpu[<cpu>,<info>]			
CPU information.	String or integer	cpu - <CPU number> or <i>all</i> (default) info - possible values: <i>full</i> (default), <i>curfreq</i> , <i>maxfreq</i> , <i>model</i> or <i>vendor</i>	Example: ⇒ system.hw.cpu[0,vendor] → AuthenticAMD Gathers info from <code>/proc/cpuinfo</code> and <code>/sys/devices/system/cpu/[cpunum]/cpufreq/cpuinfo_max_freq</code> . If a CPU number and <i>curfreq</i> or <i>maxfreq</i> is specified, a numeric value is returned (Hz). Supported since Zabbix agent version 2.0.
system.hw.devices[<type>]			
Listing of PCI or USB devices.	Text	type - <i>pci</i> (default) or <i>usb</i>	Example: ⇒ system.hw.devices[pci] → 00:00.0 Host bridge: Advanced Micro Devices [AMD] RS780 Host Bridge [..] Returns the output of either <code>lspci</code> or <code>lsusb</code> utility (executed without any parameters) Supported since Zabbix agent version 2.0.
system.hw.macaddr[<interface>,<format>]			

Key			
Description	Return value	Parameters	Comments
Listing of MAC addresses.	String	interface - <i>all</i> (default) or a regular expression format - <i>full</i> (default) or <i>short</i>	Lists MAC addresses of the interfaces whose name matches the given interface regexp (<i>all</i> lists for all interfaces). Example: ⇒ system.hw.macaddr["eth0\$",full] → [eth0] 00:11:22:33:44:55 If format is specified as <i>short</i> , interface names and identical MAC addresses are not listed. Supported since Zabbix agent version 2.0.
system.localtime[<type>]			
System time.	Integer - with type as <i>utc</i> String - with type as <i>local</i>	type - possible values: <i>utc</i> - (default) the time since the Epoch (00:00:00 UTC, January 1, 1970), measured in seconds. <i>local</i> - the time in the 'yyyy-mm-dd,hh:mm:ss.nnn,+hh:mm' format	Parameters for this item are supported since Zabbix agent version 2.0. Example: ⇒ system.localtime[local] → create an item using this key and then use it to display host time in the <i>Clock</i> screen element.
system.run[command,<mode>]			
Run specified command on the host.	Text result of the command 1 - with mode as <i>nowait</i> (regardless of command result)	command - command for execution mode - possible values: <i>wait</i> - wait end of execution (default), <i>nowait</i> - do not wait	Up to 512KB of data can be returned, including trailing whitespace that is truncated. To be processed correctly, the output of the command must be text. Example: ⇒ system.run[ls -l /] → detailed file list of root directory. <i>Note:</i> To enable this functionality, agent configuration file must contain <code>EnableRemoteCommands=1</code> option. <i>Note:</i> The return value of the item is standard output together with standard error produced by command. <i>Note:</i> Empty result is allowed starting with Zabbix 2.4.0. See also: Command execution .
system.stat[resource,<type>]			
System statistics.	Integer or float	ent - number of processor units this partition is entitled to receive (float) kthr,<type> - information about kernel thread states: <i>r</i> - average number of runnable kernel threads (float) <i>b</i> - average number of kernel threads placed in the Virtual Memory Manager wait queue (float) memory,<type> - information about the usage of virtual and real memory: <i>avm</i> - active virtual pages (integer) <i>fre</i> - size of the free list (integer) page,<type> - information about page faults and paging activity: <i>fi</i> - file page-ins per second (float) <i>fo</i> - file page-outs per second (float) <i>pi</i> - pages paged in from paging space (float) <i>po</i> - pages paged out to paging space (float) <i>fr</i> - pages freed (page replacement) (float) <i>sr</i> - pages scanned by page-replacement algorithm (float) faults,<type> - trap and interrupt rate: <i>in</i> - device interrupts (float) <i>sy</i> - system calls (float) <i>cs</i> - kernel thread context switches (float) cpu,<type> - breakdown of percentage usage of processor time: <i>us</i> - user time (float) <i>sy</i> - system time (float) <i>id</i> - idle time (float) <i>wa</i> - idle time during which the system had outstanding disk/NFS I/O request(s) (float) <i>pc</i> - number of physical processors consumed (float) <i>ec</i> - the percentage of entitled capacity consumed (float) <i>lbusy</i> - indicates the percentage of logical processor(s) utilization that occurred while executing at the user and system level (float) <i>app</i> - indicates the available physical processors in the shared pool (float) disk,<type> - disk statistics: <i>bps</i> - indicates the amount of data transferred (read or written) to the drive in bytes per second (integer) <i>tps</i> - indicates the number of transfers per second that were issued to the physical disk/tape (float) This item is supported since Zabbix 1.8.1 .	
system.sw.arch			

Key			
Description	Return value	Parameters	Comments
Software architecture information.	String		<p>Example: ⇒ system.sw.arch → i686</p> <p>Info is acquired from uname() function.</p> <p>Supported since Zabbix agent version 2.0.</p>
system.sw.os[<info>]			
Operating system information.	String	info - possible values: <i>full</i> (default), <i>short</i> or <i>name</i>	<p>Example: ⇒ system.sw.os[short]→ Ubuntu 2.6.35-28.50-generic 2.6.35.11</p> <p>Info is acquired from (note that not all files and options are present in all distributions): /proc/version (<i>full</i>) /proc/version_signature (<i>short</i>) PRETTY_NAME parameter from /etc/os-release on systems supporting it, or /etc/issue.net (<i>name</i>)</p> <p>Supported since Zabbix agent version 2.0.</p>
system.sw.packages[<package>,<manager>,<format>]			
Listing of installed packages.	Text	<p>package - <i>all</i> (default) or a regular expression</p> <p>manager - <i>all</i> (default) or a package manager</p> <p>format - <i>full</i> (default) or <i>short</i></p>	<p>Lists (alphabetically) installed packages whose name matches the given package regexp (<i>all</i> lists them all).</p> <p>Example: ⇒ system.sw.packages[mini,dpkg,short] → python-minimal, python2.6-minimal, ubuntu-minimal</p> <p>Supported package managers (executed command): dpkg (dpkg --get-selections) pkgtool (ls /var/log/packages) rpm (rpm -qa) pacman (pacman -Q)</p> <p>If format is specified as <i>full</i>, packages are grouped by package managers (each manager on a separate line beginning with its name in square brackets). If format is specified as <i>short</i>, packages are not grouped and are listed on a single line.</p> <p>Supported since Zabbix agent version 2.0.</p>
system.swap.in[<device>,<type>]			
Swap in (from device into memory) statistics.	Integer	<p>device - device used for swapping (default is <i>all</i>)</p> <p>type - possible values: <i>count</i> (number of swapins), <i>sectors</i> (sectors swapped in), <i>pages</i> (pages swapped in). See supported by platform for details on defaults.</p>	<p>Example: ⇒ system.swap.in[,pages]</p> <p>The source of this information is: /proc/swaps, /proc/partitions, /proc/stat (Linux 2.4) /proc/swaps, /proc/diskstats, /proc/vmstat (Linux 2.6)</p>
system.swap.out[<device>,<type>]			
Swap out (from memory onto device) statistics.	Integer	<p>device - device used for swapping (default is <i>all</i>)</p> <p>type - possible values: <i>count</i> (number of swapouts), <i>sectors</i> (sectors swapped out), <i>pages</i> (pages swapped out). See supported by platform for details on defaults.</p>	<p>Example: ⇒ system.swap.out[,pages]</p> <p>The source of this information is: /proc/swaps, /proc/partitions, /proc/stat (Linux 2.4) /proc/swaps, /proc/diskstats, /proc/vmstat (Linux 2.6)</p>
system.swap.size[<device>,<type>]			

Key			
Description	Return value	Parameters	Comments
Swap space size in bytes or in percentage from total.	Integer - for bytes Float - for percentage	device - device used for swapping (default is <i>all</i>) type - possible values: <i>free</i> (free swap space, default), <i>pfree</i> (free swap space, in percent), <i>used</i> (used swap space, in percent), <i>total</i> (total swap space), <i>used</i> (used swap space)	Example: ⇒ <code>system.swap.size[,pfree]</code> → free swap space percentage If <i>device</i> is not specified Zabbix agent will only take into account swap devices (files), physical memory will be ignored. For example, on Solaris systems <code>swap -s</code> command includes a portion of physical memory and swap devices (unlike <code>swap -l</code>). Note that this key might report incorrect swap space size/percentage on virtualized (VMware ESXi, VirtualBox) Windows platforms. In this case you may use the <code>perf_counter[\700(_Total)\702]</code> key to obtain correct swap space percentage. Old naming: <code>system.swap.free</code> , <code>system.swap.total</code>
system.uptime			
Identification of the system.	String		Example of returned value (Unix): FreeBSD localhost 4.2-RELEASE FreeBSD 4.2-RELEASE #0: Mon Nov i386 Example of returned value (Windows): Windows ZABBIX-WIN 6.0.6001 Microsoft® Windows Server® 2008 Standard Service Pack 1 x86 On Unix since Zabbix 2.2.0 the value for this item is obtained with <code>uname()</code> system call. Previously it was obtained by invoking “ <code>uname -a</code> ”. The value of this item might differ from the output of “ <code>uname -a</code> ” and does not include additional information that “ <code>uname -a</code> ” prints based on other sources. On Windows since Zabbix 3.0 the value for this item is obtained from <code>Win32_OperatingSystem</code> and <code>Win32_Processor</code> WMI classes. Previously it was obtained from volatile Windows APIs and undocumented registry keys. The OS name (including edition) might be translated to the user's display language. On some versions of Windows it contains trademark symbols and extra spaces. Note that on Windows the item returns OS architecture, whereas on Unix it returns CPU architecture.
system.uptime			
System uptime in seconds.	Integer		In item configuration , use s or uptime units to get readable values.
system.users.num			
Number of users logged in.	Integer		who command is used on the agent side to obtain the value.
vfs.dev.read[<device>,<type>,<mode>]			

Key			
Description	Return value	Parameters	Comments
Disk read statistics.	Integer - with type in <i>sectors, operations, bytes</i> Float - with type in <i>sps, ops, bps</i>	device - disk device (default is <i>all</i>) type - possible values: <i>sectors, operations, bytes, sps, ops, bps</i> This parameter must be specified, since defaults differ under various OSes. <i>sps, ops, bps</i> stand for: sectors, operations, bytes per second, respectively. mode - possible values: <i>avg1</i> (one-minute average, default), <i>avg5</i> , <i>avg15</i> . This parameter is supported only with type in: <i>sps, ops, bps</i> .	Default values of 'type' parameter for different OSes: AIX - operations FreeBSD - bps Linux - sps OpenBSD - operations Solaris - bytes Example: ⇒ <code>vfs.dev.read[,operations]</code> <i>sps, ops</i> and <i>bps</i> on supported platforms used to be limited to 8 devices (7 individual and one <i>all</i>). Since Zabbix 2.0.1 this limit is 1024 devices (1023 individual and one for <i>all</i>). If default <i>all</i> is used for the first parameter then the key will return summary statistics, including all block devices like <i>sda, sdb</i> and their partitions (<i>sda1, sda2, sdb3...</i>) and multiple devices (MD raid) based on those block devices/partitions and logical volumes (LVM) based on those block devices/partitions. In such cases returned values should be considered only as relative value (dynamic in time) but not as absolute values. Supports LVM since Zabbix 1.8.6. Only relative device names could be used (for example, sda) until Zabbix 1.8.6. Since then, an optional /dev/ prefix may be used (for example, /dev/sda). Old naming: <i>io[*]</i>
vfs.dev.write[<device>,<type>,<mode>]			
Disk write statistics.	Integer - with type in <i>sectors, operations, bytes</i> Float - with type in <i>sps, ops, bps</i>	device - disk device (default is <i>all</i>) type - possible values: <i>sectors, operations, bytes, sps, ops, bps</i> This parameter must be specified, since defaults differ under various OSes. <i>sps, ops, bps</i> stand for: sectors, operations, bytes per second, respectively. mode - possible values: <i>avg1</i> (one-minute average, default), <i>avg5</i> , <i>avg15</i> . This parameter is supported only with type in: <i>sps, ops, bps</i> .	Default values of 'type' parameter for different OSes: AIX - operations FreeBSD - bps Linux - sps OpenBSD - operations Solaris - bytes Example: ⇒ <code>vfs.dev.write[,operations]</code> <i>sps, ops</i> and <i>bps</i> on supported platforms used to be limited to 8 devices (7 individual and one <i>all</i>). Since Zabbix 2.0.1 this limit is 1024 (1023 individual and one for <i>all</i>). If default <i>all</i> is used for the first parameter then the key will return summary statistics, including all block devices like <i>sda, sdb</i> and their partitions (<i>sda1, sda2, sdb3...</i>) and multiple devices (MD raid) based on those block devices/partitions and logical volumes (LVM) based on those block devices/partitions. In such cases returned values should be considered only as relative value (dynamic in time) but not as absolute values. Supports LVM since Zabbix 1.8.6. Only relative device names could be used (for example, sda) until Zabbix 1.8.6. Since then, an optional /dev/ prefix may be used (for example, /dev/sda). Old naming: <i>io[*]</i>
vfs.file.cksum[file]			
File checksum, calculated by the UNIX cksum algorithm.	Integer	file - full path to file	Example: ⇒ <code>vfs.file.cksum[/etc/passwd]</code> Example of returned value: 1938292000 Old naming: <i>cksum</i> The file size limit depends on large file support .
vfs.file.contents[file,<encoding>]			

Key			
Description	Return value	Parameters	Comments
Retrieving contents of a file.	Text	file - full path to file encoding - code page identifier	Returns an empty string if the file is empty or contains LF/CR characters only. Example: ⇒ <code>vfs.file.contents[/etc/passwd]</code> This item is limited to files no larger than 64 Kbytes. Supported since Zabbix agent version 2.0.
vfs.file.exists[file]			
Checks if file exists.	0 - not found 1 - regular file or a link (symbolic or hard) to regular file exists	file - full path to file	Example: ⇒ <code>vfs.file.exists[/tmp/application.pid]</code> The return value depends on what <code>S_ISREG</code> POSIX macro returns. The file size limit depends on large file support .
vfs.file.md5sum[file]			
MD5 checksum of file.	Character string (MD5 hash of the file)	file - full path to file	Example: ⇒ <code>vfs.file.md5sum[/usr/local/etc/zabbix_agentd.conf]</code> Example of returned value: <code>b5052decb577e0fffd622d6ddc017e82</code> The file size limit (64 MB) for this item was removed in version 1.8.6. The file size limit depends on large file support .
vfs.file.regexp[file,regexp,<encoding>,<start line>,<end line>,<output>]			
Find string in a file.	The line containing the matched string, or as specified by the optional output parameter	file - full path to file regexp - GNU regular expression encoding - code page identifier start line - the number of first line to search (first line of file by default). end line - the number of last line to search (last line of file by default). output - an optional output formatting template. The <code>\0</code> escape sequence is replaced with the matched text while an <code>\N</code> (where <code>N=1...9</code>) escape sequence is replaced with Nth matched group (or an empty string if the <code>N</code> exceeds the number of captured groups).	Only the first matching line is returned. An empty string is returned if no line matched the expression. Content extraction using the output parameter takes place on the agent. The start line, end line and output parameters are supported from version 2.2. Examples: ⇒ <code>vfs.file.regexp[/etc/passwd,zabbix]</code> ⇒ <code>vfs.file.regexp[/path/to/some/file,"([0-9]+)\$",,3,5,\1]</code> ⇒ <code>vfs.file.regexp[/etc/passwd,"^zabbix::([0-9]+)",,,\1]</code> → getting the ID of user <code>zabbix</code>
vfs.file.regmatch[file,regexp,<encoding>,<start line>,<end line>]			
Find string in a file.	0 - match not found 1 - found	file - full path to file regexp - GNU regular expression encoding - code page identifier start line - the number of first line to search (first line of file by default). end line - the number of last line to search (last line of file by default).	The start line and end line parameters are supported from version 2.2. Example: ⇒ <code>vfs.file.regmatch[/var/log/app.log,error]</code>
vfs.file.size[file]			
File size (in bytes).	Integer	file - full path to file	The file must have read permissions for user <code>zabbix</code> . Example: ⇒ <code>vfs.file.size[/var/log/syslog]</code> The file size limit depends on large file support .
vfs.file.time[file,<mode>]			
File time information.	Integer (Unix timestamp)	file - full path to the file mode - possible values: <code>modify</code> (default) - modification time, <code>access</code> - last access time, <code>change</code> - last change time	Example: ⇒ <code>vfs.file.time[/etc/passwd,modify]</code> The file size limit depends on large file support .
vfs.fs.discovery			

Key			
Description	Return value	Parameters	Comments
List of mounted filesystems. Used for low-level discovery.	JSON object		Supported since Zabbix agent version 2.0. {#FSDRIVETYPE} macro is supported on Windows since Zabbix agent version 3.0.
vfs.fs.inode[fs,<mode>]			
Number or percentage of inodes.	Integer - for number Float - for percentage	fs - filesystem mode - possible values: <i>total</i> (default), <i>free</i> , <i>used</i> , <i>pfree</i> (free, percentage), <i>pusd</i> (used, percentage)	Example: ⇒ <code>vfs.fs.inode[/,pfree]</code> Old naming: <code>vfs.fs.inode.free[*]</code> , <code>vfs.fs.inode.pfree[*]</code> , <code>vfs.fs.inode.total[*]</code>
vfs.fs.size[fs,<mode>]			
Disk space in bytes or in percentage from total.	Integer - for bytes Float - for percentage	fs - filesystem mode - possible values: <i>total</i> (default), <i>free</i> , <i>used</i> , <i>pfree</i> (free, percentage), <i>pusd</i> (used, percentage)	In case of a mounted volume, disk space for local file system is returned. Example: ⇒ <code>vfs.fs.size[/tmp,free]</code> Reserved space of a file system is taken into account and not included when using the <i>free</i> mode. Old naming: <code>vfs.fs.free[*]</code> , <code>vfs.fs.total[*]</code> , <code>vfs.fs.used[*]</code> , <code>vfs.fs.pfree[*]</code> , <code>vfs.fs.pused[*]</code>
vm.memory.size[<mode>]			
Memory size in bytes or in percentage from total.	Integer - for bytes Float - for percentage	mode - possible values: <i>total</i> (default), <i>active</i> , <i>anon</i> , <i>buffers</i> , <i>cached</i> , <i>exec</i> , <i>file</i> , <i>free</i> , <i>inactive</i> , <i>pinned</i> , <i>shared</i> , <i>wired</i> , <i>used</i> , <i>pusd</i> (used, percentage), <i>available</i> , <i>pavailable</i> (available, percentage)	This item accepts three categories of parameters: 1) <i>total</i> - total amount of memory; 2) platform-specific memory types: <i>active</i> , <i>anon</i> , <i>buffers</i> , <i>cached</i> , <i>exec</i> , <i>file</i> , <i>free</i> , <i>inactive</i> , <i>pinned</i> , <i>shared</i> , <i>wired</i> ; 3) user-level estimates on how much memory is used and available: <i>used</i> , <i>pusd</i> , <i>available</i> , <i>pavailable</i> . See a more detailed description of <code>vm.memory.size</code> parameters . Old naming: <code>vm.memory.buffers</code> , <code>vm.memory.cached</code> , <code>vm.memory.free</code> , <code>vm.memory.shared</code> , <code>vm.memory.total</code>
web.page.get[host,<path>,<port>]			
Get content of web page.	Web page source as text (including headers)	host - hostname path - path to HTML document (default is /) port - port number (default is 80)	Returns an empty string on fail. Example: ⇒ <code>web.page.get[www.zabbix.com,index.php,80]</code>
web.page.perf[host,<path>,<port>]			
Loading time of full web page (in seconds).	Float	host - hostname path - path to HTML document (default is /) port - port number (default is 80)	Returns 0 on fail. Example: ⇒ <code>web.page.perf[www.zabbix.com,index.php,80]</code>
web.page.regex[host,<path>,<port>,<regexp>,<length>,<output>]			
Find string on a web page.	The matched string, or as specified by the optional output parameter	host - hostname path - path to HTML document (default is /) port - port number (default is 80) regexp - GNU regular expression length - maximum number of characters to return output - an optional output formatting template. The <code>\0</code> escape sequence is replaced with the matched text while an <code>\N</code> (where N=1...9) escape sequence is replaced with Nth matched group (or an empty string if the N exceeds the number of captured groups).	Returns an empty string if no match was found or on fail. Content extraction using the output parameter takes place on the agent. The output parameter is supported from version 2.2. Example: ⇒ <code>web.page.regex[www.zabbix.com,index.php,80,OK,2]</code>

A Linux-specific note. Zabbix agent must have read-only access to filesystem `/proc`. Kernel patches from www.grsecurity.org limit access rights of non-privileged users.

Available encodings

The encoding parameter is used to specify encoding for processing corresponding item checks, so that data acquired will not be corrupted. For a list of supported encodings (code page identifiers), please consult respective documentation, such as documentation for [libiconv](#) (GNU Project) or Microsoft Windows SDK documentation for “Code Page Identifiers”.

If empty encoding is passed, then UTF-8 (default locale for newer Unix/Linux distributions, see your system's settings) or ANSI with system-specific extension (Windows) is used by default.

Troubleshooting agent items

1. If used with passive agent, *Timeout* value in server configuration may need to be higher than *Timeout* in the agent configuration file. Otherwise the item may not get any value because the server request to agent timed out first.

From:

<https://www.zabbix.com/documentation/3.2/> - **Zabbix Documentation 3.2**

Permanent link:

https://www.zabbix.com/documentation/3.2/manual/config/items/itemtypes/zabbix_agent

Last update: **2019/08/29 12:44**

