Zabbix Manual

Welcome to the user manual for Zabbix 2.0 software. These pages are created to help our users successfully manage their monitoring tasks with Zabbix, from the simple to the more complex.

2.0/manual.txt · Last modified: 2012/05/23 11:27 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

6. Configuration

2.0/manual/config.txt · Last modified: 2011/09/23 12:50 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Hosts and host groups

What is a "host"?

Typical Zabbix hosts are the devices you wish to monitor (servers, workstations, switches, etc).

Creating hosts is one of the first monitoring tasks in Zabbix. For example, if you want to monitor some parameters on a server “x”, you must first create a host called, say, “Server X” and then you can look to add monitoring items to it.

Hosts are organized into host groups.

Proceed to creating and configuring a host.

2.0/manual/config/hosts.txt · Last modified: 2011/09/28 09:07 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Configuring a host

Overview

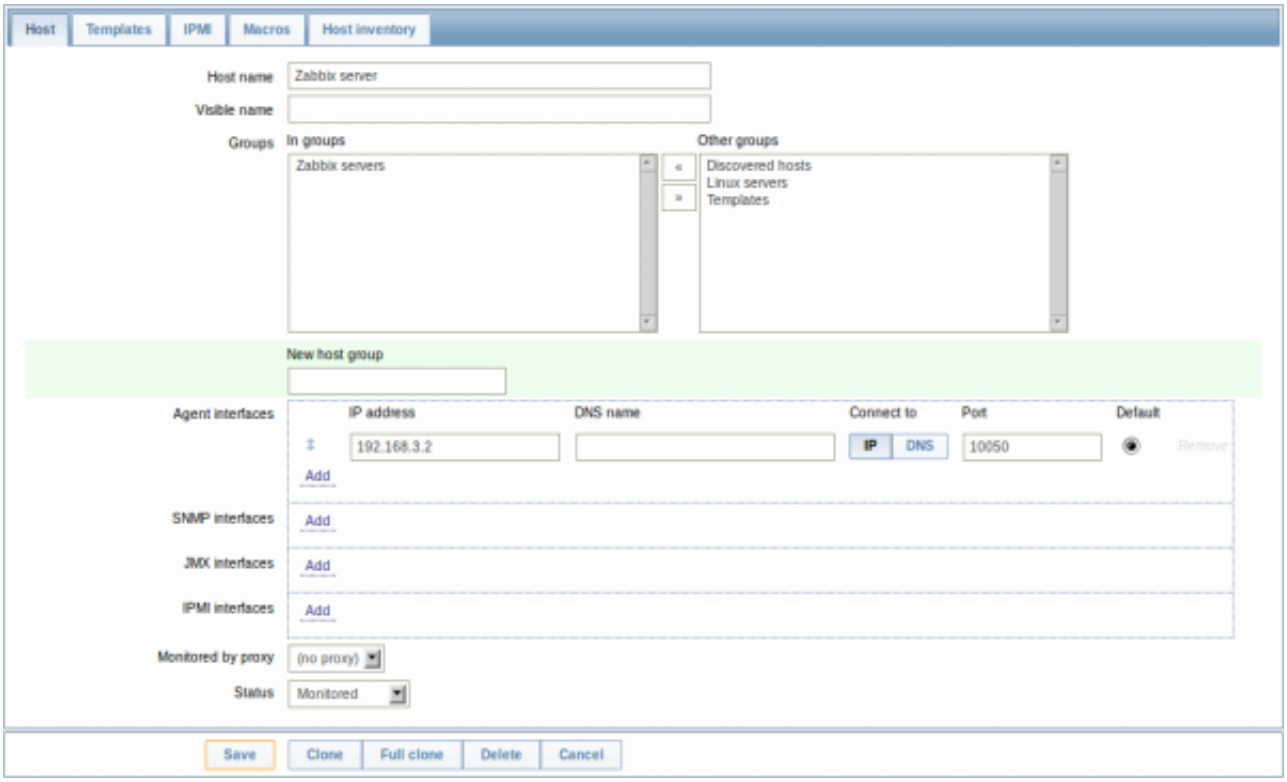
To configure a host in Zabbix frontend, do the following:

- Go to: *Configuration → Hosts*
- Click on *Create* to the right (or on the host name to edit an existing host)
- Enter parameters of the host in the form

You can also use the *Clone* and *Full clone* buttons in the form of an existing host to create a new host. Clicking on *Clone* will retain all host parameters and template linkage (keeping all entities from those templates). *Full clone* will additionally retain directly attached entities (items, triggers, graphs and applications).

Configuration

The **Host** tab contains general host attributes:



Parameter	Description
Host name	Enter a unique host name. Alpha-numericals, spaces, full stops and underscores are allowed. *Note:* With Zabbix agent running on the host you are configuring, the agent configuration file parameter *Hostname* must have the same value as the host name entered here. The name in the parameter is needed in the processing of active checks.
Visible name	If you set this name, it will be the one visible in lists, maps, etc. This attribute has UTF-8 support.
Groups	Select host groups the host belongs to. A host must belong to at least one host group.

New host group	A new group can be created and linked to the host. Ignored, if empty.
Interfaces	Several host interface types are supported for a host: *Agent*, *SNMP*, *JMX* and *IPMI*. To add a new interface, click on *Add* in the *Interfaces* block and enter *IP/DNS*, *Connect to* and *Port* info. *Note:* Interfaces that are used in any items cannot be removed and link *Remove* is greyed out for them.
IP address	Host IP address (optional).
DNS name	Host DNS name (optional).
Connect to	Clicking the respective button will tell Zabbix server what to use to retrieve data from agents: **IP** - Connect to the host IP address (recommended) **DNS** - Connect to the host DNS name
Port	TCP port number. Default value for Zabbix agent is 10050.
Default	Check the radio button to set the default interface.
Monitored by proxy	The host can be monitored either by Zabbix server or one of Zabbix proxies: **(no proxy)** - host is monitored by Zabbix server **Proxy name** - host is monitored by Zabbix proxy "Proxy name"
Status	Host status: **Monitored** - Host is active, ready to be monitored **Not monitored** - Host is not active, thus not monitored

The **Templates** tab allows you to link templates to the host. All entities (items, triggers, graphs and applications) will be inherited from the template.

To link a new template, click on *Add*. To unlink a template, use one of the two options:

- *Unlink* - unlink the template, but preserve its items, triggers and graphs
- *Unlink and clear* - unlink the template and remove all its items, triggers and graphs

The **IPMI** tab contains IPMI management attributes.

Parameter	Description
Authentication algorithm	Select the authentication algorithm.
Privilege level	Select the privilege level.
Username	User name for authentication.
Password	Password for authentication.

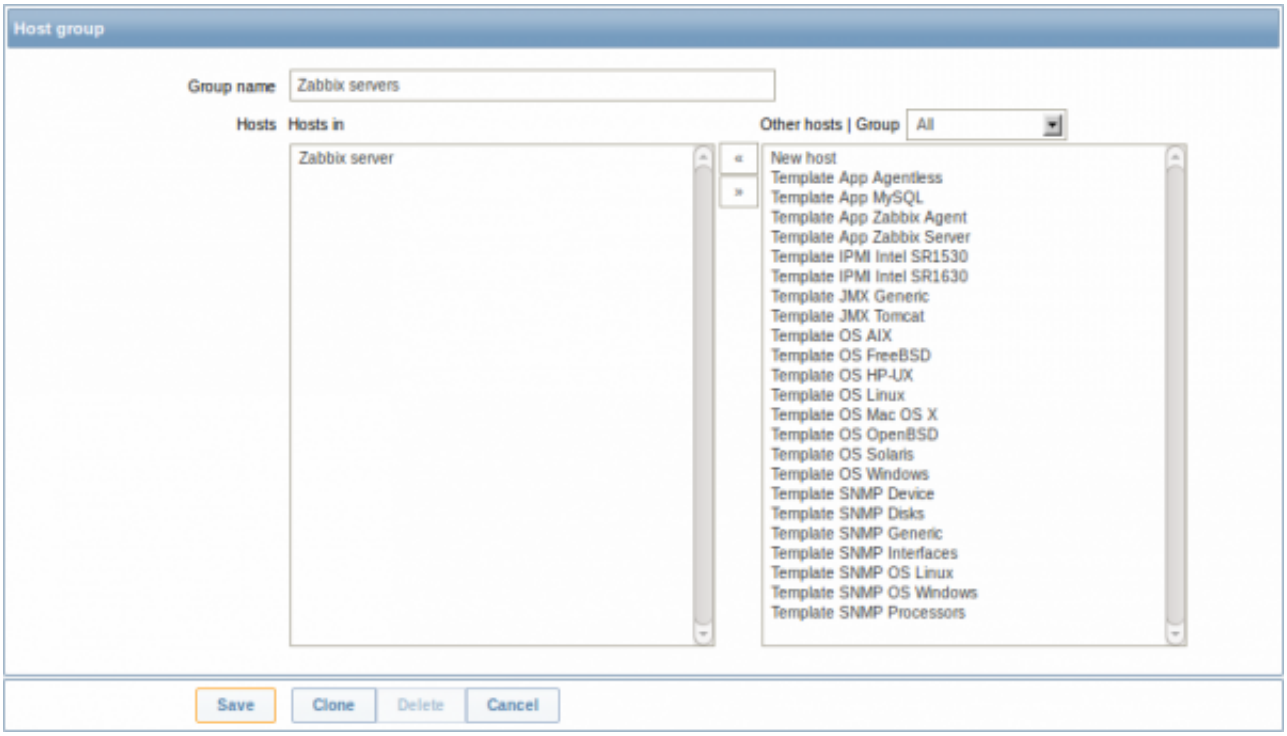
The **Macros** tab allows you to define host-level user macros.

The **Host inventory** tab allows you to manually enter inventory information for the host. You can also select to enable *Automatic* inventory population, or disable inventory population for this host.

Configuring a host group

To configure a host group in Zabbix frontend, do the following:

- Go to: *Configuration → Host groups*
- Click on *Create Group* in the upper right corner of the screen
- Enter parameters of the group in the form



Parameter	Description
Group name	Enter a unique host group name. The name must be unique within a Zabbix node.
Hosts	Select hosts, members of the group. A host group may have zero, one or more hosts.

2.0/manual/config/hosts/host.txt · Last modified: 2012/05/31 09:59 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Inventory

Overview

You can keep the inventory of networked devices in Zabbix.

There is a special *Inventory* menu in the Zabbix front-end. However, you will not see any data there initially and it is not where you enter data. Building inventory data is done manually when configuring a host or automatically by using some automatic population options.

Building inventory

Manual mode

When configuring a host, in the *Host inventory* tab you can enter such details as the type of device, serial number, location, responsible person, etc - data that will populate inventory information.

If a URL is included in host inventory information and it starts with 'http' or 'https', it will result in a clickable link in the *Inventory* section.

Automatic mode

Host inventory can also be populated automatically. For that to work, when configuring a host the inventory mode in the *Host inventory* tab must be set to *Automatic*.

Then you can configure host items to populate any host inventory field with their value, indicating the destination field with the respective attribute (called *Item will populate host inventory field*) in item configuration.

Items that are especially useful for automated inventory data collection:

- system.hw.chassis[full|type|vendor|model|serial] - default is [full], root permissions needed
- system.hw.cpu[all|cpunum,full|maxfreq|vendor|model|curfreq] - default is [all,full]
- system.hw.devices[pci|usb] - default is [pci]
- system.hw.macaddr[interface,short|full] - default is [all,full], interface is regex
- system.sw.arch
- system.sw.os[name|short|full] - default is [name]
- system.sw.packages[package,manager,short|full] - default is [all,all,full], package is regex

Inventory overview

The overview of all inventory data entered is available in the *Inventory* menu.

In *Inventory → Overview* you can group the display of available data by various fields of the inventory.

In *Inventory → Hosts* you can see all hosts that have inventory information. Clicking on the host name will reveal all the details in a form.

Inventory macros

There are host inventory macros {INVENTORY.*} available for use in notifications, for example:

“Server in {INVENTORY.LOCATION1} has a problem, responsible person is {INVENTORY.CONTACT1}, phone number {INVENTORY.POC.PRIMARY.PHONE.A1}.”

{PROFILE.*} macros from previous Zabbix versions are still supported but it's highly recommended to change those to {INVENTORY.*}

For more details, see the Macros supported by location page.

2.0/manual/config/hosts/inventory.txt · Last modified: 2012/08/22 16:24 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 Mass update

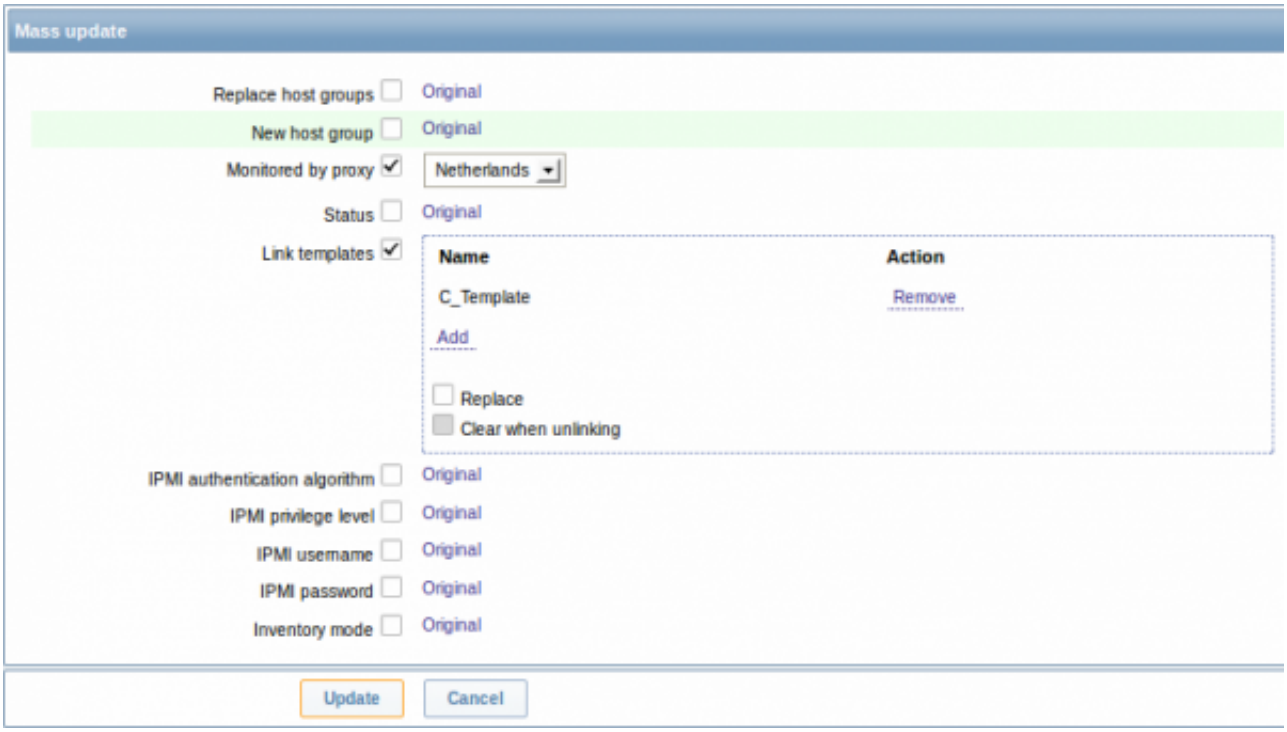
Overview

Sometimes you may want to change some attribute for a number of hosts at once. Instead of opening each individual host for editing, you may use the mass update function for that.

Using mass update

To mass-update some hosts, do the following:

- Mark the checkboxes of the hosts to update in the list
- Select *Mass update* from the dropdown below and click on *Go*
- Mark the checkboxes of the attributes to update
- Enter new values for the attributes and click on *Save*



2.0/manual/config/hosts/hostupdate.txt · Last modified: 2012/07/11 09:29 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Items

Overview

Items are the ones that gather data from a host.

Once you have configured a host, you need to add some monitoring items to start getting actual data.

An item is an individual metric. One way of quickly adding many items is to attach one of the predefined templates to a host. For optimized system performance though, you may need to fine-tune the templates to have only as many items and as frequent monitoring as is really necessary.

In an individual item you specify what sort of data will be gathered from the host.

For that purpose you use the item key. Thus an item with the key name **system.cpu.load** will gather data of the processor load, while an item with the key name **net.if.in** will gather incoming traffic information.

To specify further parameters with the key, you include those in square brackets after the key name. Thus, system.cpu.load**[avg5]** will return processor load average for the last 5 minutes, while net.if.in**[eth0]** will show incoming traffic in the interface eth0.

For all supported item types and item keys, see individual sections of item types.

Proceed to creating and configuring an item.

2.0/manual/config/items.txt · Last modified: 2013/08/14 13:14 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

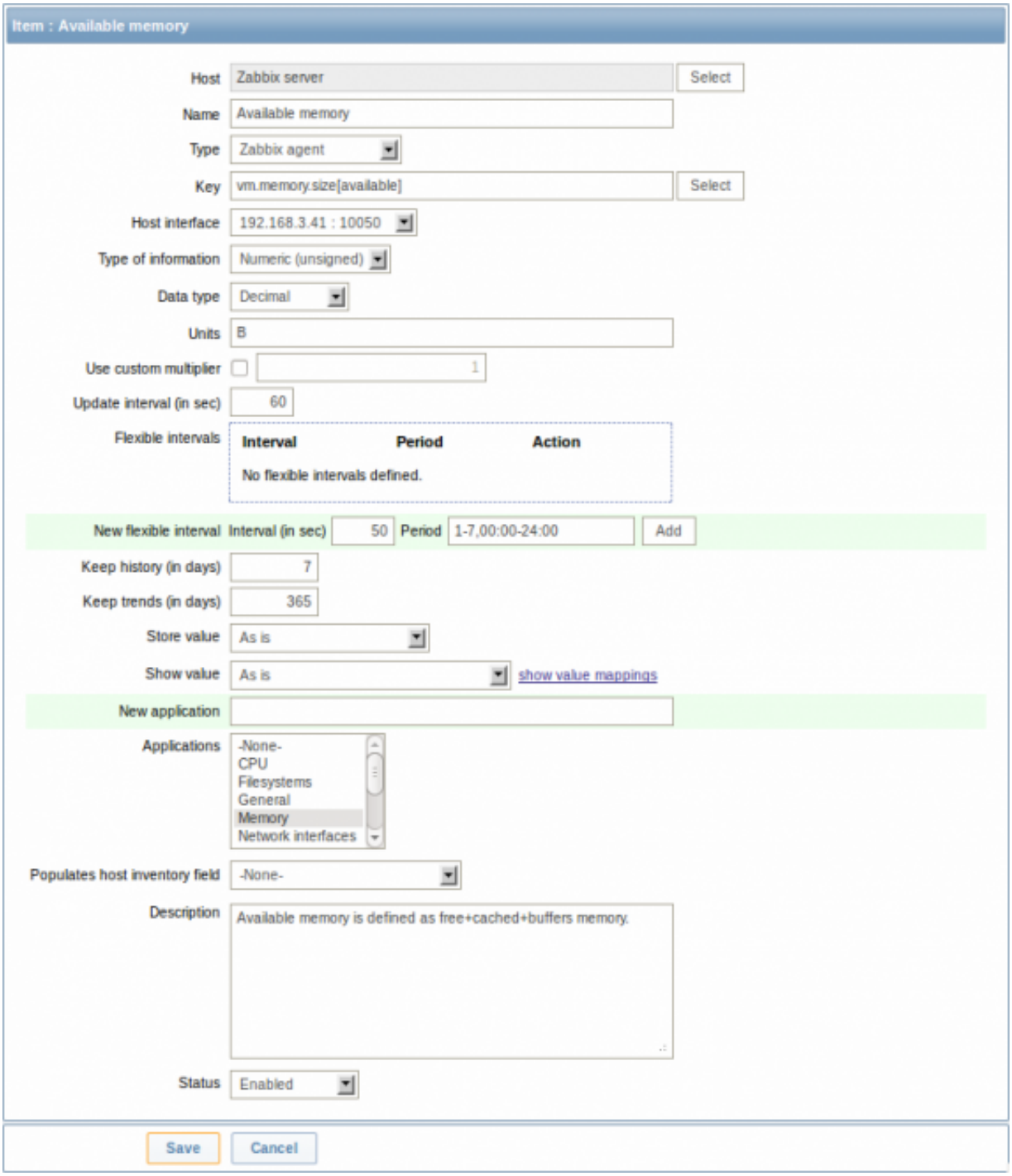
1 Creating an item

Overview

To create an item in Zabbix frontend, do the following:

- Go to: *Configuration* → *Hosts*
- Click on *Items* in the row of the host
- Click on *Create item* in the upper right corner of the screen
- Enter parameters of the item in the form

Configuration



Item attributes:

Parameter	Description
Host	Select the host or template.
Name	This is how the item will be named. The following macros can be used: **$1, $2...$9** – referring to the first, second... ninth parameter of the item key For example: Free disk space on $1 If the item key is "vfs.fs.size[/,free]", the description will automatically change to "Free disk space on /"
Type	Item type. See individual item type sections.

Key	Item key. The supported item keys can be found in individual item type sections. The key must be unique within a single host. If key type is 'Zabbix agent', 'Zabbix agent (active)', 'Simple check' or 'Zabbix aggregate', the key value must be supported by Zabbix agent or Zabbix server. See also: the correct key format.
Host interface	Select the host interface. This field is available when editing an item on the host level.
Type of information	Type of data as stored in the database after performing conversions, if any. **Numeric (unsigned)** - 64bit unsigned integer **Numeric (float)** - floating point number **Character** - character (string) data limited to 255 bytes **Log** - log file. Must be set for keys log[]. **Text** - text of unlimited size
Data type	Data type is used for integer items in order to specify the expected data type: **Boolean** - textual representation translated into either 0 or 1. Thus, 'TRUE' is stored as 1 and 'FALSE' is stored as 0. All values are matched in a case-insensitive way. Currently recognized values are, for: *TRUE* - true, t, yes, y, up, running, enabled, available *FALSE* - false, f, no, n, down, unused, disabled, unavailable Additionally, any non-zero numeric value is considered to be TRUE and zero is considered to be FALSE. **Octal** - data in octal format **Decimal** - data in decimal format **Hexadecimal** - data in hexadecimal format Zabbix will automatically perform conversion to numeric.
Units	If a unit symbol is set, Zabbix will add post processing to the received value and display it with the set unit postfix. By default, if the raw value exceeds 1000, it is divided by 1000 and displayed accordingly. For example, if you set *bps* and receive a value of 881764, it will be displayed as 881.76 Kbps. Special processing is used for **B** (byte), **Bps** (bytes per second) units, which are divided by 1024. Thus, if units are set to **B** or **Bps** Zabbix will display: 1 as 1B/1Bps 1024 as 1KB/1KBps 1536 as 1.5KB/1.5KBps Special processing is used if the following time-related units are used: **unixtime** - translated to "yyyy.mm.dd hh:mm:ss". To translate correctly, the received value must be a *Numeric (unsigned)* type of information. **uptime** - translated to "hh:mm:ss" or "N days, hh:mm:ss" For example, if you receive the value as 881764 (seconds), it will be displayed as "10 days, 04:56:04" **s** - translated to "yyy mmm ddd hhh mmm sss ms"; parameter is treated as number of seconds. For example, if you receive the value as 881764 (seconds), it will be displayed as "10d 4h 56m" Only 3 upper major units are shown, like "1m 15d 5h" or "2h 4m 46s". If there are no days to display, only two levels are displayed - "1m 5h" (no minutes, seconds or milliseconds are shown). Will be translated to "< 1 ms" if the value is less than 0.001. See also the unit blacklist.
Use custom multiplier	If you enable this option, all received values will be multiplied by the integer or floating-point value set in the value field. Use this option to convert values received in KB, MBps, etc into B, Bps. Otherwise Zabbix cannot correctly set prefixes (K, M, G etc).
Update interval (in sec)	Refresh this item every N seconds. *Note*: If set to '0', the item will not be polled. However, if a flexible interval also exists with a non-zero value, the item will be polled during the flexible interval duration.
Flexible intervals	You can create exceptions to *Update interval*. For example: Interval: **10**, Period: **1-5,09:00-18:00** - will set the refresh to every 10 seconds for working hours. Otherwise default update interval will be used. If multiple flexible intervals overlap, the smallest *Interval* value is used for the overlapping period. See the page about setting time periods for description of the *Period* format. *Note*: If set to '0', the item will not be polled during the flexible interval duration and will resume polling according to the *Update interval* once the flexible interval period is over. *Note*: Not available for Zabbix agent active items.
Keep history (in days)	Number of days to keep detailed history in the database. Older data will be removed by Housekeeper. It is recommended to keep the recorded values for the smallest possible number of days to reduce the size of value

	history in the database. Instead of keeping long history of values, you can keep longer data of trends.
Keep trends (in days)	Keep aggregated (hourly min, max, avg, count) detailed history for N days in the database. Older data will be removed by Housekeeper. *Note:* Keeping trends is not available for non-numeric data - character, log and text.
Store value	**As is** - no pre-processing **Delta (speed per second)** - evaluate value as (**value-prev_value**)/(**time-prev_time**), where *value* - current value *value_prev* - previously received value *time* - current timestamp *prev_time* - timestamp of previous value This setting is extremely useful to get speed per second for a constantly growing value. *Note*: If current value is smaller than the previous value, Zabbix discards that difference (stores nothing) and waits for another value. This helps to work correctly with, for instance, a wrapping (overflow) of 32-bit SNMP counters. **Delta (simple change)** - evaluate as (**value-prev_value**), where *value* - current value *value_prev* - previously received value
Show value	Apply value mapping to this item. Value mapping does not change received values, it is for displaying data only. It works with integer items only. For example, "Windows service states".
Log time format	Available for items of type **Log** only. Supported placeholders: * **y**: *Year (0001-9999)* * **M**: *Month (01-12)* * **d**: *Day (01-31)* * **h**: *Hour (00-23)* * **m**: *Minute (00-59)* * **s**: *Second (00-59)* If left blank the timestamp will not be parsed. For example, consider the following line from the Zabbix agent log file: " 23480:20100328:154718.045 Zabbix agent started. Zabbix 1.8.2 (revision 11211)." It begins with six character positions for PID, followed by date, time, and the rest of the line. Log time format for this line would be "pppppp:yyyyMMdd:hhmmss". Note that "p" and ":" chars are just placeholders and can be anything but "yMdhms".
New application	Enter the name of a new application for the item.
Applications	Link item to one or more existing applications.
Populates host inventory field	You can select a host inventory field that the value of item will populate. This will work if automatic inventory population is enabled for the host.
Description	Enter an item description.
Status	**Enabled** - the item will be processed. **Disabled** - the item will not be processed. **Not supported** - the item is not supported. This item will not be processed, however Zabbix may try to periodically set the status of such items to *Enabled* according to the interval set for refreshing unsupported items.

You can also create an item by opening an existing one, pressing the *Clone* button and then saving under a different name.

When editing an existing template level item on a host level, a number of fields are read-only. You can use the link in the form header and go to the template level and edit them there, keeping in mind that the changes on a template level will change the item for all hosts that the template is linked to.

Unit blacklist

By default, specifying a unit for an item will result in a multiplier prefix being added - for example, value 2048 with unit B would be displayed as 2KB. For a pre-defined, hardcoded list of units this is prevented:

- ms
- RPM
- rpm
- %

Note that both lowercase and uppercase **rpm** (*rpm* and *RPM*) strings are blacklisted.

Unsupported items

An item can become unsupported if its value cannot be retrieved for some reason. Such items are still rechecked at a fixed interval, configurable in Administration section.

2.0/manual/config/items/item.txt · Last modified: 2013/02/23 00:13 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Item key

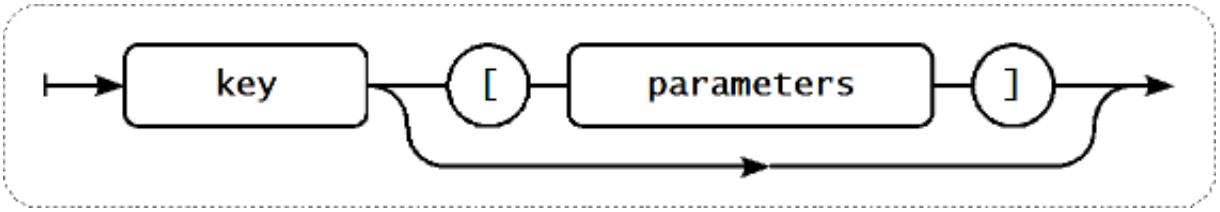
1.1 Flexible and non-flexible parameters

A flexible parameter is a parameter which accepts an argument. For example, in vfs.fs.size[*] the asterisk symbol '*' indicates a flexible parameter. '*' is any string that will be passed as an argument to the parameter. Correct definition examples:

- vfs.fs.size[/]
- vfs.fs.size[/opt]

1.2 Key format

Item key format, including key parameters, must follow syntax rules. The following illustrations depict the supported syntax. Allowed elements and characters at each point can be determined by following the arrows - if some block can be reached through the line, it is allowed, if not - it is not allowed.



To construct a valid item key, one starts with specifying the key name, then there's a choice to either have parameters or not - as depicted by the two lines that could be followed.

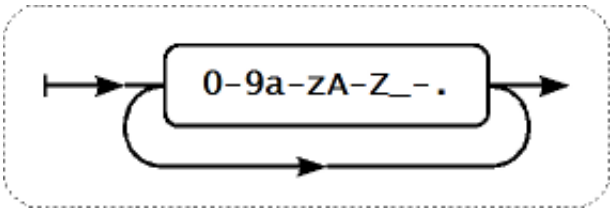
Key name

The key name itself has a limited range of allowed characters, which just follow each other. Allowed characters are:

```
0-9a-zA-Z_-.
```

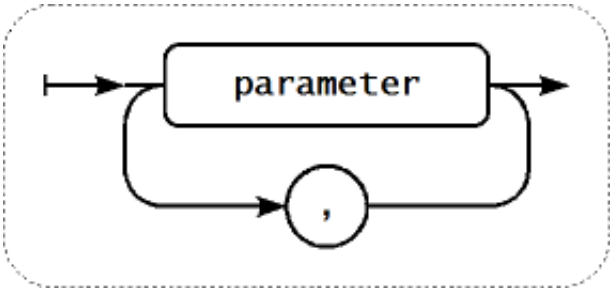
Which means:

- all numbers;
- all lowercase letters;
- all uppercase letters;
- underscore;
- dash;
- dot.

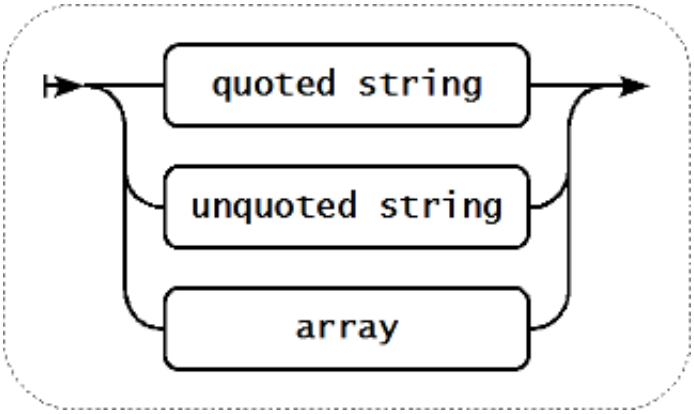


Key parameters

An item key can have multiple parameters that are comma separated.



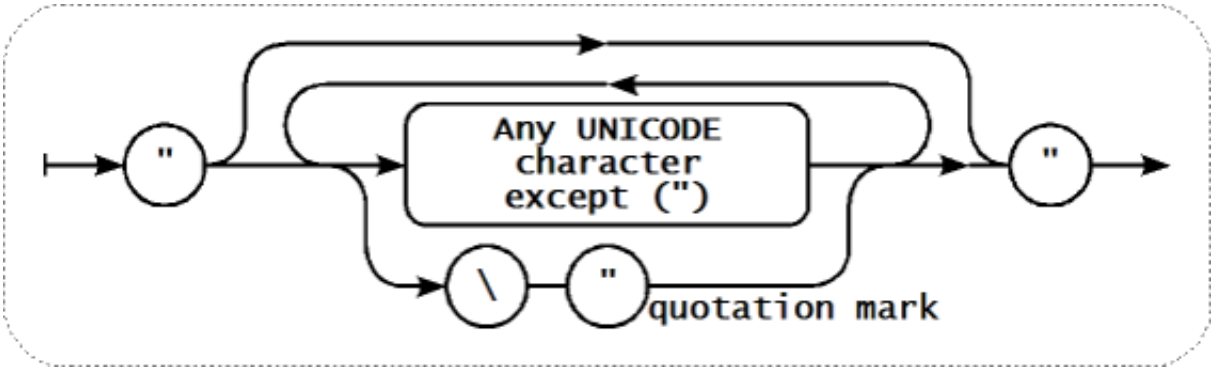
Each key parameter can be either a quoted string, an unquoted string or an array.



The parameter can also be left empty, thus using the default value. In that case, the appropriate number of commas must be added if any further parameters are specified. For example, item key **icmpping[„200„500]** would specify that the interval between individual pings is 200 milliseconds, timeout - 500 milliseconds, and all other parameters are left at their defaults.

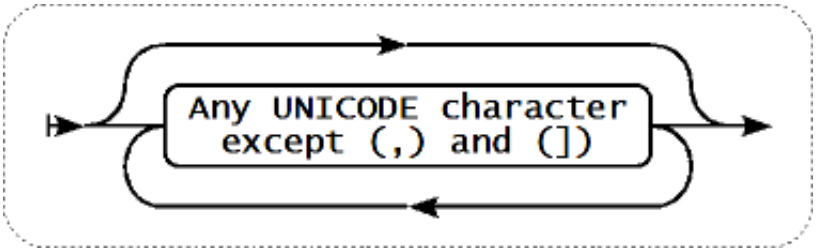
Parameter - quoted string

If the key parameter is a quoted string, any Unicode character is allowed, and included double quotes must be backslash escaped.



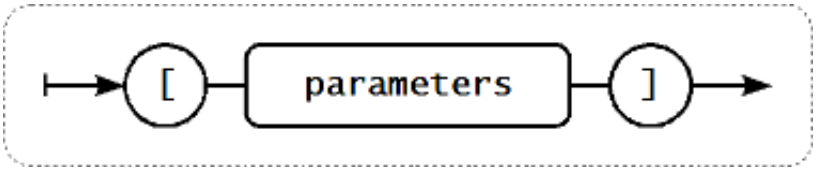
Parameter - unquoted string

If the key parameter is an unquoted string, any Unicode character is allowed except comma and right square bracket (]).



Parameter - array

If the key parameter is an array, it is again enclosed in square brackets, where individual parameters come in line with the rules and syntax of specifying multiple parameters.



1.3 Available encodings

The parameter “encoding” 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 [http://www.gnu.org/software/libiconv/] (GNU Project) or Microsoft Windows SDK documentation for “Code Page Identifiers”. If an empty “encoding” parameter is passed, then ANSI with system-specific extension (Windows) or UTF-8 (default locale for newer Unix/Linux distributions, see your system's settings) is used by default.

2.0/manual/config/items/item/key.txt · Last modified: 2011/10/11 11:48 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Item types

Overview

Item types are the variety of checks offered by Zabbix - *Zabbix agent, Simple checks, SNMP, Zabbix internal, IPMI, JMX monitoring* etc.

Some checks are performed by the Zabbix server alone (as agent-less monitoring) while others require the Zabbix agent or even Zabbix Java gateway (with JMX monitoring).

With each item type you must specify the required parameters and use the supported set of item keys.

The details of all item types are included in the subpages of this section.

1. Starting with Zabbix 2.0, multiple interfaces can be set in the host definition - Agent, SNMP, JMX, and IPMI. If a particular item type requires a particular interface (like an IPMI item needs an IPMI interface on the host), that interface must exist, in the host definition. Also, if an item can use more than one interface, it will search the available host interfaces (in the order: Agent→SNMP→JMX→IPMI) for the first appropriate one to be linked with.
2. All items that return text (character, log, text types of information) now can return whitespace only as well (where applicable), setting the return value to an empty string. (supported since 2.0)

2.0/manual/config/items/itemtypes.txt · Last modified: 2012/01/26 14:02 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

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 WIN32 agent

Key				
▲	**Description**	**Return value**	**Parameters**	**Comments**
agent.hostname				
	Returns agent host name.	String value	-	Returns the actual value of the agent hostname from a configuration file.
agent.ping				
	Check the agent availability.	Returns '1' if agent is available, nothing if unavailable.	-	Use function **nodata()** 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.	Number of files. Integer.		
kernel.maxproc				
	Maximum number of processes supported by OS.	Number of processes. Integer.		
log[file,<regexp>,<encoding>,<maxlines>,<mode>]				
	Monitoring of log file.	Log.	**file** – full path and name of log file **regexp** – regular expression describing the required pattern **encoding** – code page identifier **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 **mode** – possible values:	The item must be configured as an active check. *Example key*: log[/home/zabbix/logs/logfile,,,100] See a more detailed description.

			mode - possible values: *all* (default), *skip* (skip processing of older data). The **mode** parameter is supported from version 2.0.	
logrt[file_format,<regexp>,<encoding>,<maxlines>,<mode>]				
	Monitoring of log file with log rotation support.	Log.	**file_format** - absolute path to file and filename format given as a regexp **regexp** - regular expression describing the required pattern **encoding** - code page identifier **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 **mode** - possible values: *all* (default), *skip* (skip processing of older data). The **mode** parameter is supported from version 2.0.	The item must be configured as an active check. *Example key*: logrt["/home/zabbix/logs/^logfile[0-9]{1,3}$",,,100] Log rotation is based on the last modification time of files. See a more detailed description.
net.dns[<ip>,zone,<type>,<timeout>,<count>]				
	Checks if DNS service is up.	0 - DNS is down (server did not respond or DNS resolution failed) 1 - DNS is up	**ip** - IP address of DNS server (leave empty for the default DNS server, ignored on Windows) **zone** - zone to test the DNS **type** - record type to be queried (default is SOA) **timeout** (ignored on Windows) - timeout for the request (default is 1 second) **count** (ignored on Windows) - number of tries for the request (default is 2)	*Example key*: net.dns[8.8.8.8,zabbix.com,MX,2,1] The possible values for **type** are: ANY, A, NS, CNAME, MB, MG, MR, PTR, MD, MF, MX, SOA, NULL, WKS (except for Windows), HINFO, MINFO, TXT, SRV Internationalized domain names are not supported, please use IDNA encoded names instead. Naming before Zabbix 2.0 (still supported): *net.tcp.dns*
net.dns.record[<ip>,zone,<type>,<timeout>,<count>]				
	Performs a DNS query.	On success returns a character string with the required type of information.	**ip** - IP address of DNS server (leave empty for the default DNS server, ignored on Windows) **zone** - zone to test the DNS **type** - record type to be queried (default is SOA) **timeout** (ignored on Windows) - timeout for the request (default is 1 second) **count** (ignored on Windows) - number of tries for the request (default is 2)	*Example key*: net.dns.record[8.8.8.8,zabbix.com,MX,2,1] The possible values for **type** are: ANY, A, NS, CNAME, MB, MG, MR, PTR, MD, MF, MX, SOA, NULL, WKS (except for Windows), HINFO, MINFO, TXT, SRV SRV record type is supported since Zabbix agent version 1.8.6. Internationalized domain names are not supported, please use IDNA encoded names instead. Naming before Zabbix 2.0 (still supported): *net.tcp.dns.query*
net.if.collisions[if]				
	Out-of-window collision.	Number of collisions. Integer.	**if** - interface	
net.if.discovery				
	List of network interfaces. Used for low-level discovery.	JSON object		Supported since Zabbix agent version 2.0.

	discovery.			
net.if.in[if,<mode>]				
	Incoming traffic statistics on network interface.	Integer.	**if** - network interface name **mode** - possible values: *bytes* - number of bytes (default) *packets* - number of packets *errors* - number of errors *dropped* - number of dropped packets	Multi-byte interface names on Windows are supported since Zabbix agent version 1.8.6. *Example keys*: net.if.in[eth0,errors] net.if.in[eth0] You may use this key with a *Delta (speed per second)* 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 **mode** - possible values: *bytes* - number of bytes (default) *packets* - number of packets *errors* - number of errors *dropped* - number of dropped packets	Multi-byte interface names on Windows are supported since Zabbix agent 1.8.6 version. *Example keys*: net.if.out[eth0,errors] net.if.out[eth0] You may use this key with a *Delta (speed per second)* store value in order to get bytes per second statistics.
net.if.total[if,<mode>]				
	Sum of incoming and outgoing traffic statistics on network interface.	Integer.	**if** - network interface name **mode** - possible values: *bytes* - number of bytes (default) *packets* - number of packets *errors* - number of errors *dropped* - number of dropped packets	*Example keys*: net.if.total[eth0,errors] net.if.total[eth0] You may use this key with a *Delta (speed per second)* 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 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
net.tcp.port[<ip>,port]				
	Check, if it is possible to make TCP connection to port number 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. Old naming: check_port[*] 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).
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: *ssh*, *ntp*, *ldap*, *smtp*, *ftp*, *http*, *pop*, *nntp*, *imap*, *tcp*, *https*, *telnet* **ip** - IP address (default is 127.0.0.1) **port** - port number (by default standard service port number is used)	*Example key*: 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 prompt (':' at the end). Old naming: check_service[*] Note that before Zabbix 1.8.3 version **service.ntp** should be used instead of **ntp**. **https** and **telnet** services are supported since Zabbix 2.0.
net.tcp.service.perf[service,<ip>,<port>]				

	Checks performance of service.	0 - service is down; seconds - the number of seconds spent while connecting to the service	**service** - either of: *ssh*, *ntp*, *ldap*, *smtp*, *ftp*, *http*, *pop*, *nntp*, *imap*, *tcp*, *https*, *telnet* **ip** - IP address (default is 127.0.0.1) **port** - port number (by default standard service port number is used)	*Example key*: net.tcp.service.perf[ssh] - 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 net.tcp.service.perf[tcp,<ip>,<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 prompt (':' at the end). Old naming: check_service_perf[*] Note that before Zabbix 1.8.3 version **service.ntp** should be used instead of **ntp**. **https** and **telnet** services are supported since Zabbix 2.0.

net.udp.listen[port]			
Checks if this UDP port is in LISTEN state.	0 - it is not 1 - it is in LISTEN state	**port** - UDP port number	Example: net.udp.listen[68] On Linux supported since Zabbix agent version 1.8.4

proc.mem[<name>,<user>,<mode>,<cmdline>]			
Memory used by process running under some user.	Memory used by process (in bytes).	**name** - process name (default is "all processes") **user** - user name (default is "all users") **mode** - possible values: *avg*, *max*, *min*, *sum* (default) **cmdline** - filter by command line	*Example keys*: proc.mem[,root] - memory used by all processes running under the "root" user proc.mem[zabbix_server,zabbix] - memory used by all zabbix_server processes running under the zabbix user proc.mem[,oracle,max,oracleZABBIX] - memory used by the most memory-hungry process running under oracle having oracleZABBIX in its command line

proc.num[<name>,<user>,<state>,<cmdline>]			
The number of processes having certain state running under some user.	Number of processes.	**name** - process name (default is "all processes") **user** - user name (default is "all users") **state** - possible values: *all* (default), *run*, *sleep*, *zomb* **cmdline** - filter by command line	*Example keys*: proc.num[,mysql] - number of processes running under the mysql user proc.num[apache2,www-data] - number of apache2 processes running under the www-data user proc.num[,oracle,sleep,oracleZABBIX] - number of processes in sleep state running under oracle having oracleZABBIX in its command line On Windows, only the *name* and *user* parameters are supported.

sensor[device,sensor,<mode>]			
Hardware sensor reading.		**device** - device name (if <mode> is used, it is a regular expression) **sensor** - sensor name (if <mode> is used, it is a regular expression) **mode** - possible values: *avg*, *max*, *min* (if this parameter is omitted, device and sensor are treated verbatim).	On Linux 2.4, reads */proc/sys/dev/sensors*. Example key: sensor[w83781d-i2c-0-2d,temp1] Prior to Zabbix 1.8.4, the sensor[temp1] format was used. On OpenBSD, reads the *hw.sensors* MIB. Example keys: sensor[cpu0,temp0] - one temperature of one CPU sensor[cpu[0-2]$,temp,avg] - average temperature of the first three CPU's Supported on OpenBSD since Zabbix 1.8.4.

system.boottime			
Timestamp of system boot.	Integer.		A UNIX timestamp (date and time, down to a second) returned.

system.cpu.intr			
Device interrupts.	Integer.		

system.cpu.load[<cpu>,<mode>]			
CPU load [http://en.wikipedia.org/wiki/Load_(computing)].	Processor load. Float.	**cpu** - possible values: *all* (default), *percpu* (total load divided by online CPU count) **mode** - possible values: *avg1* (one-minute average, default), *avg5* (5-minute average), *avg15* (an average within 15 minutes)	*Example key*: system.cpu.load[,avg5] Old naming: system.cpu.loadX Parameter **percpu** is supported since Zabbix 2.0.0.

system.cpu.num[<type>]				
	Number of CPUs.	Number of available processors.	**type** - possible values: *online* (default), *max*	*Example key*: system.cpu.num
system.cpu.switches				
	Context switches.	Switches count.		Old naming: system[switches]
system.cpu.util[<cpu>,<type>,<mode>]				
	CPU(s) utilisation.	Processor utilisation in percent.	**cpu** - CPU number (default is all CPUs) **type** - possible values: *idle*, *nice*, *user* (default), *system* (default for Windows), *iowait*, *interrupt*, *softirq*, *steal* **mode** - possible values: *avg1* (one-minute average, default), *avg5* (5-minute average), *avg15* (an average within 15 minutes)	*Example key*: system.cpu.util[0,user,avg5] Old naming: system.cpu.idleX, system.cpu.niceX, system.cpu.systemX, system.cpu.userX
system.hostname[<type>]				
	Returns host name.	String value	**type** (only on Windows, ignored on other systems) - possible values: *netbios* (default) or *host*	The value is acquired by either GetComputerName() (for **netbios**) or gethostname() (for **host**) functions on Windows and by "hostname" command on other systems. **The *type* parameter for this item is supported since 1.8.6 version.** Examples of returned values: on Linux: *system.hostname* → linux-w7x1 *system.hostname* → www.zabbix.com on Windows: *system.hostname* → WIN-SERV2008-I6 *system.hostname[host]* → Win-Serv2008-I6LonG See also a more detailed description.
system.hw.chassis[<info>]				
	Returns chassis info	String value	**info** - one of full (default), model, serial, type or vendor	Example: system.hw.chassis[full] Hewlett-Packard HP Pro 3010 Small Form Factor PC CZXXXXXXXX Desktop] **Root permissions** are required because the value is acquired by reading from memory. Supported since Zabbix agent version 2.0.
system.hw.cpu[<cpu>,<info>]				
	Returns CPU info	String or numeric value	**cpu** - CPU number or all (default) **info** - one of full (default), curfreq, maxfreq, model or vendor	Example: system.hw.cpu[0,vendor] AuthenticAMD Gathers info from /proc/cpuinfo and /sys/devices/system/cpu/[cpunum]/cpufreq/cpuinfo_max_freq. If a CPU number and **curfreq** or **maxfreq** is specified, a numeric value is returned (Hz). Supported since Zabbix agent version 2.0.
system.hw.devices[<type>]				
	Lists PCI or USB devices	Text value	**type** - pci (default) or usb	Example: system.hw.devices[pci] 00:00.0 Host bridge: Advanced Micro Devices [AMD] RS780 Host Bridge [..] Returns the output of either lspci or lsusb utility (executed without any parameters) Supported since Zabbix agent version 2.0.

system.hw.macaddr[<interface>,<format>]				
	Lists MAC addresses	String value	**interface** – all (default) or a regular expression **format** – full (default) or short	Example: system.hw.macaddr["eth0$",full] [eth0] 00:11:22:33:44:55 Lists MAC adresses of the interfaces whose names match the given **interface** regex ("all" lists for all interfaces). If **format** is specified as **short**, interface names and identical MAC addresses are not listed. Supported since Zabbix agent version 2.0.
system.localtime[<type>]				
	System time.	Integer or string value.	**utc** – (default) the time since the Epoch (00:00:00 UTC, January 1, 1970), measured in seconds. **local** – the time in the 'yyyy-mm-dd,hh:mm:ss.nnn,+hh:mm' format Parameters for this item supported from version 2.0.	
system.run[command,<mode>]				
	Run specified command on the host.	Text result of the command.	**command** – command for execution **mode** – one of wait (default, wait end of execution), nowait (do not wait)	Up to 512KB of data can be returned (64KB before Zabbix 2.0.5), 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. *Note*: To enable this functionality, agent configuration file must have EnableRemoteCommands=1 option.
system.stat[resource,<type>]				
	Virtual memory statistics	Numeric value	**ent** – number of processor units this partition is entitled to receive (float) **kthr,<type>** – information about kernel thread states: **r** – average number of runnable kernel threads (float) **b** – 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: **avm** – active virtual pages (integer) **fre** – size of the free list (integer) **page,<type>** – information about page faults and paging activity: **fi** – file page-ins per second (float) **fo** – file page-outs per second (float) **pi** – pages paged in from paging space (float) **po** – pages paged out to paging space (float) **fr** – pages freed (page replacement) (float) **sr** – pages scanned by page-replacement algorithm (float) **faults,<type>** – trap and interrupt rate: **in** – device interrupts (float) **sy** – system calls (float) **cs** – kernel thread context switches (float) **cpu,<type>** – breakdown of percentage usage of processor time: **us** – user time (float) **sy** – system time (float) **id** – idle time (float) **wa** – idle time during which the system had outstanding disk/NFS I/O request(s) (float) **pc** – number of physical processors consumed (float) **ec** – the percentage of entitled capacity consumed (float) **lbusy** – indicates the percentage of logical processor(s) utilization that occurred while executing at the user and system level (float) **app** – indicates the available physical processors in the shared pool (float) **disk,<type>** – disk statistics: **bps** – indicates the amount of data transferred (read or written) to the drive in bytes per second (integer) **tps** – indicates the number of transfers per second that were issued to the physical disk/tape (float) **This item is supported starting from version 1.8.1.**	
system.sw.arch				
				Example: system.sw.arch i686

	Returns software architecture	String value		Info is acquired from uname() function. Supported since Zabbix agent version 2.0.
system.sw.os[<info>]				
	Returns OS info	String value	**info** - one of full (default), short or name	Example: system.sw.os[short] Ubuntu 2.6.35-28.50-generic 2.6.35.11 Info is acquired from (note that not all files are present in all distributions): [full] - /proc/version [short] - /proc/version_signature [name] - /etc/issue.net Supported since Zabbix agent version 2.0.
system.sw.packages[<package>,<manager>,<format>]				
	Lists installed packages	Text value	**package** - all (default) or a regular expression **manager** - all (default) or a package manager **format** - full (default) or short	Example: system.sw.packages[mini,dpkg,short] python-minimal, python2.6-minimal, ubuntu-minimal Lists (alphabetically) installed packages whose names match the given **package** regex ("all" lists them all). Supported packages managers: **manager** (executed command) dpkg (dpkg --get-selections) pkgtool (ls /var/log/packages) rpm (rpm -qa) pacman (pacman -Q) If **format** is specified as **full**, packages are grouped by package managers (each manager on a seperate line beginning with it's name in square brackets). If **format** is specified as **short**, packages are not grouped and are listed on a single line. Supported since Zabbix agent version 2.0.
system.swap.in[<device>,<type>]				
	Swap in (from device into memory) statistics.	Numeric value	**device** - device used for swapping (default is all) **type** - possible values: *count* (number of swapins), *sectors* (sectors swapped in), *pages* (pages swapped in). See supported by platform for details on defaults.	*Example key*: system.swap.in[,pages] *The source of this information is:* *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 (from memory onto device) statistics.	Numeric value	**device** - device used for swapping (default is all) **type** - possible values: *count* (number of swapouts), *sectors* (sectors swapped out), *pages* (pages swapped out). See supported by platform for details on defaults.	*Example key*: system.swap.out[,pages] *The source of this information is:* *Linux 2.4*: /proc/swaps, /proc/partitions, /proc/stat *Linux 2.6*: /proc/swaps, /proc/diskstats, /proc/vmstat
system.swap.size[<device>,<type>]				
	Swap space size.	Number of bytes or percentage.	**device** - device used for swapping (default is all) **type** - possible values: *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)	*Example key*: system.swap.size[,pfree] - free swap space percentage Old naming: system.swap.free, system.swap.total

system.uname				
	Returns detailed host information.	String value		Example of returned value: FreeBSD localhost 4.4-RELEASE FreeBSD 4.4-RELEASE #0: Tue Sep 18 11:57:08 PDT 2001 murray@builder.FreeBSD.org: /usr/src/sys/compile/GENERIC i386
system.uptime				
	System uptime in seconds.	Number of seconds.		In item configuration, use **s** or **uptime** units to get readable values.
system.users.num				
	Number of users logged in.	Number of users.		**who** command is used on the agent side to obtain the value.
vfs.dev.read[<device>,<type>,<mode>]				
	Disk read statistics.	Integer if **type** is in: *sectors*, *operations*, *bytes* Float if **type** is in: *sps*, *ops*, *bps*	**device** - disk device (default is "all"[1]) **type** - possible values: *sectors*, *operations*, *bytes*, *sps*, *ops*, *bps* (must be specified, since defaults differ under various OSes). *sps*, *ops*, *bps* stand for: sectors, operations, bytes per second, respectively **mode** - possible values: *avg1* (one-minute average, default), *avg5* (five-minute average), *avg15* (15-minute average). *Note*: The third parameter is supported only if the **type** is in: sps, ops, bps.	*Example key*: vfs.dev.read[,operations] Old naming: io[*] Usage of the **type** parameters ops, bps and sps on supported platforms is limited to 8 devices (7 individual devices and one "all"). Starting with Zabbix 2.0.1 this limit is increased to 1024 (1023 individual devices and one for "all"). Supports LVM since Zabbix 1.8.6. Until Zabbix 1.8.6, only relative device names may be used (for example, **sda**), since 1.8.6 an optional /**dev**/ prefix may be used (for example, /**dev**/**sda**)
vfs.dev.write[<device>,<type>,<mode>]				
	Disk write statistics.	Integer if **type** is in: *sectors*, *operations*, *bytes* Float if **type** is in: *sps*, *ops*, *bps*	**device** - disk device (default is "all"[1]) **type** - one of sectors, operations, bytes, sps, ops, bps (must specify exactly which parameter to use, since defaults are different under various OSes). sps, ops, bps means: sectors, operations, bytes per second respectively **mode** - one of avg1 (default),avg5 (average within 5 minutes), avg15. *Note*: The third parameter is supported only if the **type** is in: sps, ops, bps.	Example: vfs.dev.write[,operations] Old naming: io[*] The **type** parameters ops, bps and sps on supported platforms are limited to 8 devices (7 individual devices and one "all"). Starting with Zabbix 2.0.1 this limit is increased to 1024 (1023 individual devices and one for "all"). Supports LVM since Zabbix 1.8.6. Until Zabbix 1.8.6, only relative device names may be used (for example, **sda**), since 1.8.6 optional /**dev**/ prefix may be used (for example, /**dev**/**sda**)
vfs.file.cksum[file]				
	Calculate file checksum	File checksum, calculated by algorithm used by UNIX cksum.	**file** - full path to file	Example of returned value: 1938292000 Example: vfs.file.cksum[/etc/passwd] Old naming: cksum The file size limit depends on large file support.
vfs.file.contents[file,<encoding>]				
	Get file contents	Contents of a file or EOF if it is empty or it contains only LF/CR characters.	**file** - full path to file	Example: vfs.file.contents[/etc/passwd] This item is limited to files no larger than 64 Kbytes. Supported since Zabbix agent version 2.0.

vfs.file.exists[file]				
	Check if file exists	1 – regular file or a link (symbolic or hard) to regular file exists. 0 – otherwise	**file** – full path to file	Example: vfs.file.exists[/tmp/application.pid] The return value depends on what S_ISREG POSIX macro returns. The file size limit depends on large file support.
vfs.file.md5sum[file]				
	File's MD5 checksum	MD5 hash of the file.	**file** – full path to file	Example of returned value: b5052decb577e0fffd622d6ddc017e82 Example: vfs.file.md5sum[/usr/local/etc/zabbix_agentd.conf] 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>]				
	Find string in a file	The whole line from file containing the matched string or EOF if expression not found	**file** – full path to file **regexp** – GNU regular expression **encoding** – Code Page identifier	Only the first matching line is returned. Example: vfs.file.regexp[/etc/passwd,zabbix]
vfs.file.regmatch[file,regexp,<encoding>]				
	Find string in a file	0 – expression not found 1 – found	**file** – full path to file **regexp** – GNU regular expression **encoding** – Code Page identifier	Example: vfs.file.regmatch[/var/log/app.log,error]
vfs.file.size[file]				
	File size	Size in bytes.	**file** – full path to file	File must have read permissions for user zabbix Example: vfs.file.size[/var/log/syslog] The file size limit depends on large file support.
vfs.file.time[file,<mode>]				
	File time information.	Unix timestamp.	**file** – full path to the file **mode** – one of modify (default, modification time), access – last access time, change – last change time	Example: vfs.file.time[/etc/passwd,modify] The file size limit depends on large file support.
vfs.fs.discovery				
	List of mounted filesystems. Used for low-level discovery.	JSON object		Supported since Zabbix agent version 2.0.
vfs.fs.inode[fs,<mode>]				
	Number of inodes	Numeric value	**fs** – filesystem **mode** – one of total (default), free, used, pfree (free, percentage), pused (used, percentage)	Example: vfs.fs.inode[/,pfree] Old naming: vfs.fs.inode.free[*], vfs.fs.inode.pfree[*], vfs.fs.inode.total[*]
vfs.fs.size[fs,<mode>]				
			fs – filesystem **mode** – one of total	In case of a mounted volume, disk space for local file system is

	Disk space	Disk space in bytes	**mode** – one of total (default), free, used, pfree (free, percentage), pused (used, percentage)	returned. Example: vfs.fs.size[/tmp,free] Old naming: vfs.fs.free[*], vfs.fs.total[*], vfs.fs.used[*], vfs.fs.pfree[*], vfs.fs.pused[*]
vm.memory.size[<mode>]				
	Memory size	Memory size in bytes or in percentage from total	**mode** – one of total (default), active, anon, buffers, cached, exec, file, free, inactive, pinned, shared, wired, used, pused, available, pavailable	Old naming: vm.memory.buffers, vm.memory.cached, vm.memory.free, vm.memory.shared, vm.memory.total Item vm.memory.size[] accepts three categories of parameters. First category consists of **total** – total amount of memory. Second category contains platform-specific memory types: **active**, **anon**, **buffers**, **cached**, **exec**, **file**, **free**, **inactive**, **pinned**, **shared**, **wired**. Third category are user-level estimates on how much memory is used and available: **used**, **pused**, **available**, **pavailable**. See a more detailed description of vm.memory.size parameters.
web.page.get[host,<path>,<port>]				
	Get content of web page	Web page source as text	**host** – hostname **path** – path to HTML document (default is /) **port** – port number (default is 80)	Returns EOF on fail. Example: web.page.get[www.zabbix.com,index.php,80]
web.page.perf[host,<path>,<port>]				
	Get timing of loading full web page	Time in seconds	**host** – hostname **path** – path to HTML document (default is /) **port** – port number (default is 80)	Returns 0 on fail. Example: web.page.perf[www.zabbix.com,index.php,80]
web.page.regexp[host,<path>,<port>,<regexp>,<length>]				
	Get first occurrence of regexp in web page	Matched string	**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	Returns EOF on fail (no match). Example: web.page.regexp[www.zabbix.com,index.php,80,OK,2]

[1] If default "all" is used for the first parameter of **vfs.dev.*** keys then the keys will return summary statistics, including: all block devices like sda, sbd and their partitions sda1, sda2, sdb3 ... 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.

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

2.0/manual/config/items/itemtypes/zabbix_agent.txt · Last modified: 2013/10/16 19:27 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

WIN32-specific item keys

Item keys

The table provides details on the item keys that you can use with Zabbix WIN32 agent only.

On a 64-bit system, a 64-bit Zabbix agent version is required for all checks related to running 64-bit processes to work correctly.

Key			
▲	**Description**	**Return value**	**Comments**
eventlog[name,<regexp>,<severity>,<source>,<eventid>,<maxlines>,<mode>]			
	Monitoring of event logs.	Log. **name** - name of event log **regexp** - regular expression describing the required pattern **severity** - regular expression describing severity The parameter accepts the following values: "Information", "Warning", "Error", "Failure Audit", "Success Audit" **source** - source identifier **eventid** - regular expression describing the event identifier(s) **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.win.conf **mode** - possible values: *all* (default), *skip* (skip processing of older data). The **mode** parameter is supported from version 2.0.	The item must be configured as an active check. Examples: eventlog[Application] eventlog[Security,,"Failure Audit",,529|680] eventlog[System,,"Warning|Error"] eventlog[System,,,,^1$] eventlog[System,,,,@TWOSHORT] - here a custom regular expression named **TWOSHORT** is referenced (defined as a **Result is TRUE** type, the expression itself being **^1$|^70$**).
net.if.list			
	List of network interfaces: Type Status IPv4 Description	String.	Supported since Zabbix agent version 1.8.1. Multi-byte interface names supported since Zabbix agent version 1.8.6. Disabled interfaces are not listed. Note that enabling/disabling some components may change their ordering in the Windows interface name.
perf_counter[counter,<interval>]			
	Value of any Windows performance counter, where "counter" is the counter path, and "interval" is the time period for storing the average value. See also: Windows	Average value of the "counter" during last "interval" seconds. Default value, if not given, for "interval" is 1.	Performance Monitor can be used to obtain list of available counters. Until version 1.6 this parameter will return correct value only for counters that require just one sample (like \System\Threads). It will not work as expected for counters that require more that one sample - like CPU utilisation. Since 1.6 **interval** is used, so the check returns an average value for last "interval" seconds every time.

	performance counters.		

proc_info[<process>,<attribute>,<type>]			
	Different information about specific process(es).	<**process**> - process name (same as in proc_cnt[] parameter) <**attribute**> - requested process attribute. <**type**> - representation type (meaningful when more than one process with the same name exists)	The following attributes are currently supported: **vmsize** - Size of process virtual memory in Kbytes **wkset** - Size of process working set (amount of physical memory used by process) in Kbytes **pf** - Number of page faults **ktime** - Process kernel time in milliseconds **utime** - Process user time in milliseconds **io_read_b** - Number of bytes read by process during I/O operations **io_read_op** - Number of read operation performed by process **io_write_b** - Number of bytes written by process during I/O operations **io_write_op** - Number of write operation performed by process **io_other_b** - Number of bytes transferred by process during operations other than read and write operations **io_other_op** - Number of I/O operations performed by process, other than read and write operations **gdiobj** - Number of GDI objects used by process **userobj** - Number of USER objects used by process Valid types are: **min** - minimal value among all processes named <process> **max** - maximal value among all processes named <process> **avg** - average value for all processes named <process> **sum** - sum of values for all processes named <process> Examples: proc_info[iexplore.exe,wkset,sum] - to get the amount of physical memory taken by all Internet Explorer processes proc_info[iexplore.exe,pf,avg] - to get the average number of page faults for Internet Explorer processes Note: io_*, gdiobj and userobj attributes are available only on Windows 2000 and later versions of Windows, not on Windows NT 4.0.

service_state[*]			
	State of service. Parameter is service name.	0 - running 1 - paused 2 - start pending 3 - pause pending 4 - continue pending 5 - stop pending 6 - stopped 7 - unknown 255 - no such service	Parameter must be real service name as seen in service properties under “Name:” or name of EXE file.

services[<type>,<state>,<exclude>]			
	List of services, separated by a newline or 0, if list would be empty.	**type** - one of all (default), automatic, manual, disabled **state** - one of all (default), stopped, started, start_pending, stop_pending, running, continue_pending, pause_pending, paused **exclude** - list of services to exclude it from the result. Excluded services should be written in double quotes, separated by comma, without spaces.	Examples: services[,started] - list of started services services[automatic, stopped] - list of stopped services, that should be run services[automatic, stopped, "service1,service2,service3"] - list of stopped services, that should be run, excluding services with names service1, service2 and service3

		This parameter is supported starting from version 1.8.1.	

Monitoring Windows services

This tutorial provides step-by-step instructions for setting up the monitoring of Windows services. It is assumed that Zabbix server and agent are configured and operational.

To monitor the up/down status of a service you need to perform the following steps:

Step 1

Get the service name.

You can get that name by going to the services mmc and bringing up the properties of the service. In the General tab you should see a field called 'Service name'. The value that follows is the name you will use when setting up an item for monitoring.

For example, if you wanted to monitor the “workstation” service then your service might be: **lanmanworkstation**.

Step 2

Configure an item for monitoring the service, with:

- *Key*: service_state[lanmanworkstation]
- *Type of information*: Numeric (unsigned)
- *Show value*: select the *Windows service state* value mapping

2.0/manual/config/items/itemtypes/zabbix_agent/win_keys.txt · Last modified: 2013/10/16 14:28 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 SNMP agent

Overview

You may want to use SNMP monitoring on devices such as printers, network switches, routers or UPS that usually are SNMP-enabled and on which it would be impractical to attempt setting up complete operating systems and Zabbix agents.

To be able to retrieve data provided by SNMP agents on these devices, Zabbix server must be initially configured with SNMP support.

SNMP checks are performed over the UDP protocol only.

If monitoring SNMPv3 devices, make sure that msgAuthoritativeEngineID (also known as snmpEngineID or “Engine ID”) is never shared by two devices. It must be unique for each device.

For SNMPv3 privacy and authentication currently MD5 and DES protocols are supported.

Configuring SNMP monitoring

To start monitoring a device through SNMP, the following steps have to be performed:

Step 1

Create a host for the device with an SNMP interface.

Enter the IP address. Set the host status to NOT MONITORED. You can use one of the provided SNMP templates (*Template SNMP Device* and others) that will automatically add a set of items. However, the template may not be compatible with the host.

SNMP checks do not use *Agent port*, it is ignored.

Step 2

Find out the SNMP string (or OID) of the item you want to monitor.

To get a list of SNMP strings, use the **snmpwalk** command (part of ucd-snmp/net-snmp [http://www.net-snmp.org/] software which you should have installed as part of the Zabbix installation) or equivalent tool:

```
shell> snmpwalk -v 2c -c public <host IP> .
```

As '2c' here stands for SNMP version, you may also substitute it with '1', to indicate SNMP Version 1 on the device.

This should give you a list of SNMP strings and their last value. If it doesn't then it is possible that the SNMP 'community' is different from the standard 'public' in which case you will need to find out what it is.

You can then go through the list until you find the string you want to monitor, e.g. if you wanted to monitor the bytes coming in to your switch on port 3 you would use the `IF-MIB::ifInOctets.3` string from this line:

```
IF-MIB::ifInOctets.3 = Counter32: 3409739121
```

You may now use the **snmpget** command to find out the numeric OID for 'IF-MIB::ifInOctets.3':

```
shell> snmpget -v 2c -c public -On 10.62.1.22 IF-MIB::ifInOctets.3
```

Note that the last number in the string is the port number you are looking to monitor. See also: Dynamic indexes.

This should give you something like the following:

```
.1.3.6.1.2.1.2.2.1.10.3 = Counter32: 3472126941
```

Again, the last number in the OID is the port number.

3COM seem to use port numbers in the hundreds, e.g. port 1 = port 101, port 3 = port 103, but Cisco use regular numbers, e.g. port 3 = 3.

Some of the most used SNMP OIDs are translated automatically to a numeric representation by Zabbix.

Step 3

Create an item for monitoring.

So, now go back to Zabbix and click on Items, selecting the SNMP host you created earlier. Depending on whether you used a template or not when creating your host, you will have either a list of SNMP items associated with your host or just a new item box. We will work on the assumption that you are going to create the item yourself using the information you have just gathered using snmpwalk and snmpget, so enter a plain English description in the 'Description' field of the new item box. Make sure the 'Host' field has your switch/router in it and change the 'Type' field to “SNMPv* agent”. Enter the community (usually public) and enter the textual or numeric OID that you retrieved earlier into the 'SNMP OID' field, for example: .1.3.6.1.2.1.2.2.1.10.3

Enter the 'SNMP port' as 161 and the 'Key' as something meaningful, e.g. SNMP-InOctets-Bps. Choose a Multiplier if you want one and enter an 'update interval' and 'keep history' if you want it to be different from the default. Set the 'Status' to Monitored, the 'Type of information' to *Numeric (float)* and the 'Store value' to DELTA (important otherwise you will get cumulative values from the SNMP device instead of the latest change).

Now save the item and go back to the hosts area of Zabbix. From here change the SNMP device status to 'Monitored' and check in *Latest data* for your SNMP data!

Example 1

General example:

Parameter	Description
Community	public
OID	1.2.3.45.6.7.8.0 (or .1.2.3.45.6.7.8.0)
Key	<Unique string to be used as reference to triggers> For example, “my_param”.

Note that OID can be given in either numeric or string form. However, in some cases, string OID must be converted to numeric representation. Utility snmpget may be used for this purpose:

```
shell> snmpget -On localhost public enterprises.ucdavis.memory.memTotalSwap.0
```

Monitoring of SNMP parameters is possible if either --with-net-snmp or --with-ucd-snmp flag was specified while configuring Zabbix sources.

Example 2

Monitoring of uptime:

Parameter	Description
Community	public
Oid	MIB::sysUpTime.0
Key	router.uptime
Value type	Float
Units	uptime
Multiplier	0.01

2.0/manual/config/items/itemtypes/snmp.txt · Last modified: 2013/03/14 13:17 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Dynamic indexes

Overview

While you may find the required index number (for example, of a network interface) among the SNMP OIDs, sometimes you may not completely rely on the index number always staying the same.

Index numbers may be dynamic - they may change over time and your item may stop working as a consequence.

To avoid this scenario, it is possible to define an OID which takes into account the possibility of an index number changing.

For example, if you need to retrieve the index value to append to **ifInOctets** that corresponds to the **GigabitEthernet0/1** interface on a Cisco device, use the following OID:

```
ifInOctets["index","ifDescr","GigabitEthernet0/1"]
```

The syntax

A special syntax for OID is used:

<OID of data>["index","<base OID of index>","<string to search for>"]

Parameter	Description
OID of data	Main OID to use for data retrieval on the item.
index	Method of processing. Currently one method is supported: **index** - search for index and append it to the data OID
base OID of index	This OID will be looked up to get the index value corresponding to the string.
string to search for	The string to use for an exact match with a value when doing lookup. Case sensitive.

Example

Getting memory usage of apache process.

If using this OID syntax:

```
HOST-RESOURCES-MIB::hrSWRunPerfMem["index","HOST-RESOURCES-MIB::hrSWRunPath", "/usr/sbin/apache2"]
```

the index number will be looked up here:

```
...
HOST-RESOURCES-MIB::hrSWRunPath.5376 = STRING: "/sbin/getty"
HOST-RESOURCES-MIB::hrSWRunPath.5377 = STRING: "/sbin/getty"
HOST-RESOURCES-MIB::hrSWRunPath.5388 = STRING: "/usr/sbin/apache2"
HOST-RESOURCES-MIB::hrSWRunPath.5389 = STRING: "/sbin/sshd"
...
```

Now we have the index, 5388. The index will be appended to the data OID in order to receive the value we are interested in:

```
HOST-RESOURCES-MIB::hrSWRunPerfMem.5388 = INTEGER: 31468 KBytes
```

Dynamic indexes are cached since Zabbix version 1.6.3.

Using dynamic indexes leads to more SNMP queries in Zabbix versions up to 1.7. Dynamic index lookup and data retrieval is performed in single connection since Zabbix version 1.7.

2.0/manual/config/items/itemtypes/snmp/dynamicindex.txt · Last modified: 2012/09/04 15:03 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Special OIDs

Some of the most used SNMP OIDs are translated automatically to a numeric representation by Zabbix. For example, **ifIndex** is translated to **1.3.6.1.2.1.2.2.1.1**, **ifIndex.0** is translated to **1.3.6.1.2.1.2.2.1.1.0**.

The table contains list of the special OIDs.

Special OID	Identifier	Description
ifIndex	1.3.6.1.2.1.2.2.1.1	A unique value for each interface.
ifDescr	1.3.6.1.2.1.2.2.1.2	A textual string containing information about the interface.This string should include the name of the manufacturer, the product name and the version of the hardware interface.
ifType	1.3.6.1.2.1.2.2.1.3	The type of interface, distinguished according to the physical/link protocol(s) immediately 'below' the network layer in the protocol stack.
ifMtu	1.3.6.1.2.1.2.2.1.4	The size of the largest datagram which can be sent / received on the interface, specified in octets.
ifSpeed	1.3.6.1.2.1.2.2.1.5	An estimate of the interface's current bandwidth in bits per second.
ifPhysAddress	1.3.6.1.2.1.2.2.1.6	The interface's address at the protocol layer immediately `below' the network layer in the protocol stack.
ifAdminStatus	1.3.6.1.2.1.2.2.1.7	The current administrative state of the interface.
ifOperStatus	1.3.6.1.2.1.2.2.1.8	The current operational state of the interface.
ifInOctets	1.3.6.1.2.1.2.2.1.10	The total number of octets received on the interface, including framing characters.
ifInUcastPkts	1.3.6.1.2.1.2.2.1.11	The number of subnetwork-unicast packets delivered to a higher-layer protocol.
ifInNUcastPkts	1.3.6.1.2.1.2.2.1.12	The number of non-unicast (i.e., subnetwork- broadcast or subnetwork-multicast) packets delivered to a higher-layer protocol.
ifInDiscards	1.3.6.1.2.1.2.2.1.13	The number of inbound packets which were chosen to be discarded even though no errors had been detected to prevent their being deliverable to a higher-layer protocol. One possible reason for discarding such a packet could be to free up buffer space.
ifInErrors	1.3.6.1.2.1.2.2.1.14	The number of inbound packets that contained errors preventing them from being deliverable to a higher-layer protocol.
ifInUnknownProtos	1.3.6.1.2.1.2.2.1.15	The number of packets received via the interface which were discarded because of an unknown or unsupported protocol.
ifOutOctets	1.3.6.1.2.1.2.2.1.16	The total number of octets transmitted out of the interface, including framing characters.
ifOutUcastPkts	1.3.6.1.2.1.2.2.1.17	The total number of packets that higher-level protocols requested be transmitted, and which were not addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent.
ifOutNUcastPkts	1.3.6.1.2.1.2.2.1.18	The total number of packets that higher-level protocols requested be transmitted, and which were addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent.
ifOutDiscards	1.3.6.1.2.1.2.2.1.19	The number of outbound packets which were chosen to be discarded even though no errors had been detected to prevent their being transmitted. One possible reason for discarding such a packet could be to free up buffer space.
ifOutErrors	1.3.6.1.2.1.2.2.1.20	The number of outbound packets that could not be transmitted because of errors.
ifOutQLen	1.3.6.1.2.1.2.2.1.21	The length of the output packet queue (in packets).

2.0/manual/config/items/itemtypes/snmp/special_mibs.txt · Last modified: 2011/10/13 09:56 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 SNMP traps

Overview

Receiving SNMP traps is the opposite to querying SNMP-enabled devices.

In this case the information is sent from a SNMP-enabled device and is collected or "trapped" by Zabbix.

Usually traps are sent upon some condition change and the agent connects to the server on port 162 (as opposed to port 161 on the agent side that is used for queries). Using traps may detect some short problems that occur amidst the query interval and may be missed by the query data.

Receiving SNMP traps in Zabbix is designed to work with **snmptrapd** and one of the built-in mechanisms for passing the traps to Zabbix - either a perl script or SNMPTT.

The workflow of receiving a trap:

1. **snmptrapd** receives a trap
2. snmptrapd passes the trap to SNMPTT or calls Perl trap receiver
3. SNMPTT or Perl trap receiver parses, formats and writes the trap to a file
4. Zabbix SNMP trapper reads and parses the trap file
5. For each trap Zabbix finds all corresponding SNMP interfaces on hosts for the received IP or DNS address
6. For each found SNMP interface, the trap is compared to all regexes in "snmptrap[regex]" items. If found, the trap is set as the value of **all** matching items. If no match is found but there exists an "snmptrap.fallback" item, the trap is set as the value of that item.
7. If no match has been found for any of the corresponding SNMP interfaces, Zabbix by default logs the unmatched trap. (This is configured by "Log unmatched SNMP traps" in Administration → General → Other).

3.1 Configuring SNMP traps

Configuring the following fields in the frontend is specific for this item type:

- Your host must have an SNMP interface

In *Configuration → Hosts*, in the **Host interface** field set an SNMP interface with the correct IP or DNS address. The address from each received trap is compared to the IP and DNS addresses of all SNMP interfaces to find the corresponding hosts.

- Configure the item

In the **Key** field use one of the SNMP trap keys:

Key		
Description	**Return value**	**Comments**
snmptrap[regex]		
Catches all SNMP traps from a corresponding address that match **regex**	SNMP trap	This item can be set only for SNMP interfaces. **This item is supported starting from version 2.0.0.** *Note*: Starting with Zabbix 2.0.5, user macros and global regular expressions are supported in the parameter of this item key.
snmptrap.fallback		
Catches all SNMP traps from a corresponding address that were not caught by any of the snmptrap[] items for that interface	SNMP trap	This item can be set only for SNMP interfaces. **This item is supported starting from version 2.0.0.**

Multi-line regex matching is not supported at this time.

Set the **Type of information** to be 'Log' for the timestamps to be parsed. Note that other formats such as 'Numeric' are also acceptable but might require a custom trap handler.

For SNMP trap monitoring to work, it must first be correctly set up.

3.2 Setting up SNMP trap monitoring

Configuring Zabbix server/proxy

To read the traps, Zabbix server or proxy must be configured to start the SNMP trapper process and point to the trap file that is being written by SNMPTT or a perl trap receiver. To do that, edit the configuration file (zabbix_server.conf or zabbix_proxy.conf):

1. StartSNMPTrapper=1
2. SNMPTrapperFile=[TRAP FILE]

Configuring SNMPTT

At first, snmptrapd should be configured to use SNMPTT.

For the best performance, SNMPTT should be configured as a daemon using **snmptthandler-embedded** to pass the traps to it. See instructions for configuring SNMPTT in its homepage:

http://snmptt.sourceforge.net/docs/snmptt.shtml [http://snmptt.sourceforge.net/docs/snmptt.shtml]

When SNMPTT is configured to receive the traps, configure SNMPTT to log the traps:

1. log traps to the trap file which will be read by Zabbix:
   log_enable = 1
   log_file = [TRAP FILE]
2. set the date-time format:
   date_time_format = %H:%M:%S %Y/%m/%d = [DATE TIME FORMAT]

Now format the traps for Zabbix to recognise them (edit snmptt.conf):

1. Each FORMAT statement should start with “ZBXTRAP [address]”, where [address] will be compared to IP and DNS addresses of SNMP interfaces on Zabbix. E.g.:
   EVENT coldStart .1.3.6.1.6.3.1.1.5.1 “Status Events” Normal
   FORMAT ZBXTRAP $aA Device reinitialized (coldStart)
2. See more about SNMP trap format below.

Do not use unknown traps - Zabbix will not be able to recognise them. Unknown traps can be handled by defining a general event in snmptt.conf:
EVENT general .* “General event” Normal

Configuring Perl trap receiver

Requirements: Perl, Net-SNMP compiled with --enable-embedded-perl (done by default since Net-SNMP 5.4)

Perl trap receiver (look for misc/snmptrap/zabbix_trap_receiver.pl) can be used to pass traps to Zabbix server directly from snmptrapd. To configure it:

- add the perl script to snmptrapd configuration file (snmptrapd.conf), e.g.:
  perl do "[FULL PATH TO PERL RECEIVER SCRIPT]";
- configure the receiver, e.g:
  $SNMPTrapperFile = '[TRAP FILE]';
  $DateTimeFormat = '[DATE TIME FORMAT]';

If script name is not quoted, snmptrapd will refuse to start up with messages, similar to these:

```
Regexp modifiers "/l" and "/a" are mutually exclusive at (eval 2) line 1, at end of line
Regexp modifier "/l" may not appear twice at (eval 2) line 1, at end of line
```

SNMP trap format

All customised perl trap receivers and SNMPTT trap configuration must format the trap in the following way:

```
[timestamp] [the trap, part 1] ZBXTRAP [address] [the trap, part 2]
```

where

- [timestamp] - timestamp used for log items
- ZBXTRAP - header that indicates that a new trap starts in this line
- [address] - IP address used to find the host for this trap

Note that “ZBXTRAP” and ”[address]” will be cut out from the message during processing. If the trap is formatted otherwise, Zabbix might parse the traps unexpectedly.

Example trap line in the file:

```
11:30:15 2011/07/27 .1.3.6.1.6.3.1.1.5.3 Normal "Status Events" localhost - ZBXTRAP 192.168.1.1 Link down on interface 2.  Admin state: 1.  Operational state: 2
```

This will result in the following trap for SNMP interface with IP=192.168.1.1:

```
11:30:15 2011/07/27 .1.3.6.1.6.3.1.1.5.3 Normal "Status Events" localhost - Link down on interface 2.  Admin state: 1.  Operational state: 2
```

3.3 System requirements

Log rotation

Zabbix does not provide any log rotation system - that should be handled by the user. The log rotation should first rename the old file and only later delete it so that no traps are lost:

1. Zabbix opens the trap file at the last known location and goes to step 3
2. Zabbix checks if the currently opened file has been rotated by comparing the inode number to the define trap file's inode number. If there is no opened file, Zabbix resets the last location and goes to step 1.
3. Zabbix reads the data from the currently opened file and sets the new location.
4. The new data are parsed. If this was the rotated file, the file is closed and goes back to step 2.
5. If there was no new data, Zabbix sleeps for 1 second and goes back to step 2.

File system

Because of the trap file implementation, Zabbix needs the file system to support inodes to differentiate files (the information is acquired by a stat() call).

3.4 Setup example

This example uses snmptrapd + SNMPTT to pass traps to Zabbix server. Setup:

1. **zabbix_server.conf** - configure Zabbix to start SNMP trapper and set the trap file:
   StartSNMPTrapper=1
   SNMPTrapperFile=/tmp/my_zabbix_traps.tmp
2. **snmptrapd.conf** - add SNMPTT as the trap handler:
   traphandle default snmptt
3. **snmptt.ini** - configure output file and time format:
   log_file = /tmp/my_zabbix_traps.tmp
   date_time_format = %H:%M:%S %Y/%m/%d
4. **snmptt.conf** - define a default trap format:
   EVENT general .* "General event" Normal
   FORMAT ZBXTRAP $aA $ar
5. Create an SNMP item TEST:
   Host's SNMP interface IP: 127.0.0.1
   Key: snmptrap["General"]
   Log time format: hh:mm:ss yyyy/MM/dd

This results in:

1. Command used to send a trap:
   snmptrap -v 1 -c public 127.0.0.1 '.1.3.6.1.6.3.1.1.5.3' '0.0.0.0' 6 33 '55' .1.3.6.1.6.3.1.1.5.3 s “teststring000”
2. The received trap:
   15:48:18 2011/07/26 .1.3.6.1.6.3.1.1.5.3.0.33 Normal "General event" localhost - ZBXTRAP 127.0.0.1 127.0.0.1
3. Value for item TEST:
   15:48:18 2011/07/26 .1.3.6.1.6.3.1.1.5.3.0.33 Normal "General event" localhost - 127.0.0.1

This simple example uses SNMPTT as **traphandle**. For better performance on production systems, use embedded Perl to pass traps from snmptrapd to SNMPTT or directly to Zabbix.

2.0/manual/config/items/itemtypes/snmptrap.txt · Last modified: 2013/08/19 16:46 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported
[http://creativecommons.org/licenses/by-nc-sa/3.0/]

4 IPMI checks

Overview

You can monitor the health and availability of Intelligent Platform Management Interface (IPMI) devices in Zabbix.
To perform IPMI checks Zabbix server must be initially configured with IPMI support.

IPMI is a standardized interface for remote “lights-out” or “out-of-band” management of computer systems. It allows to monitor hardware status directly from the so-called “out-of-band” management cards, independently from the operating system or whether the machine is powered on at all.

Zabbix IPMI monitoring works only for devices having IPMI support (HP iLO, DELL DRAC, IBM RSA, Sun SSP, etc).

Configuration

Host configuration

A host must be configured to process IPMI checks. An IPMI interface must be added, with the respective IP and port numbers, and IPMI authentication parameters must be defined.

See the configuration of hosts for more details.

Server configuration

By default, the Zabbix server is not configured to start any IPMI pollers, thus any added IPMI items won't work. To change this, open the Zabbix server configuration file (zabbix_server.conf) as root and look for the following line:

```
# StartIPMIPollers=0
```

Uncomment it and set poller count to, say, 3, so that it reads:

```
StartIPMIPollers=3
```

Save the file and restart zabbix_server afterwards.

Item configuration

When configuring an item on a host level:

- For *Host interface* select the IPMI IP and port
- Select 'IPMI agent' as the *Type*
- Specify the *IPMI sensor* (for example 'FAN MOD 1A RPM' on Dell Poweredge)
- Enter an item key that is unique within the host (say, ipmi.fan.rpm)
- Select the respective type of information ('Numeric (float)' in this case), units (most likely 'rpm') and any other required item attributes

2.0/manual/config/items/itemtypes/ipmi.txt · Last modified: 2012/04/09 15:11 by dotneft

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

5 Simple checks

5.1 Overview

Simple checks are normally used for remote agent-less checks of services.

Note that Zabbix agent is not needed for simple checks. Zabbix server is responsible for the processing of simple checks (making external connections, etc).

Examples of using simple checks:

```
net.tcp.service[ftp,,155]
net.tcp.service[http]
net.tcp.service.perf[http,,8080]
```

5.2 Supported simple checks

List of supported simple checks:

Key				
▲	**Description**	**Return value**	**Parameters**	**Comments**
icmpping[<target>,<packets>,<interval>,<size>,<timeout>]				
	Checks if server is accessible by ICMP ping.	0 - ICMP ping fails 1 - ICMP ping successful	**target** - host IP or DNS name **packets** - number of packets **interval** - time between successive packets in milliseconds **size** - packet size in bytes **timeout** - timeout in milliseconds	Example: **icmpping[,4]** - if at least one packet of the four is returned, the item will return 1. See also details of processing ICMP pings.
icmppingloss[<target>,<packets>,<interval>,<size>,<timeout>]				
	Returns percentage of lost packets.	Loss of packets in percent	**target** - host IP or DNS name **packets** - number of packets **interval** - time between successive packets in milliseconds **size** - packet size in bytes **timeout** - timeout in milliseconds	See also details of processing ICMP pings.
icmppingsec[<target>,<packets>,<interval>,<size>,<timeout>,<mode>]				
	Returns ICMP ping response	Number of	**target** - host IP or DNS name **packets** - number of packets **interval** - time between successive packets in milliseconds	If host is not available (timeout reached), the item will

	time.	seconds	**size** - packet size in bytes **timeout** - timeout in milliseconds **mode** - one of min, max, avg (default)	return 0.
net.tcp.service[service,<ip>,<port>]				
	Check if service is running and accepting TCP connections.	0 - service is down 1 - service is running	**service** - one of ssh, ntp, ldap, smtp, ftp, http, pop, nntp, imap, tcp, https, telnet **ip** - IP address of the Zabbix host definition **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 with **tcp** service indicating the port is mandatory. 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.service[tcp,<ip>,port] for checks like these. Note that telnet check looks for a prompt (':' at the end). Services **https** and **telnet** supported since Zabbix 2.0.
net.tcp.service.perf[service,<ip>,<port>]				
	Check performance of service.	0 - service is down sec - number of seconds spent while connecting to the service	**service** - one of ssh, ntp, ldap, smtp, ftp, http, pop, nntp, imap, tcp, https, telnet **ip** - IP address of the Zabbix host definition **port** - port number (by default standard service port number is used).	Example: **net.tcp.service.perf[ssh]** can be used to test the speed of initial response from SSH server. Note that with **tcp** service indicating the port is mandatory. 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.service.perf[tcp,<ip>,port] for checks like these. Note that telnet check looks for a prompt (':' at the end). Services **https** and **telnet** supported since Zabbix 2.0.

Timeout processing

Zabbix will not process a simple check longer than the Timeout seconds defined in the Zabbix server configuration file.

5.3 ICMP pings

Zabbix uses external utility **fping** for processing of ICMP pings.

The utility is not part of Zabbix distribution and has to be additionally installed. If the utility is missing, has wrong permissions or its location does not match the location set in the Zabbix server configuration file ('FpingLocation' parameter), ICMP pings (**icmpping**, **icmppingloss**, **icmppingsec**) will not be processed.

fping must be executable by the user Zabbix daemons run as and setuid root. Run these commands as user **root** in order to set up correct permissions:

```
shell> chown root:zabbix /usr/sbin/fping
shell> chmod 4710 /usr/sbin/fping
```

Defaults, limits and description of values for ICMP check parameters:

Parameter	Unit	Description	Fping's flag	Defaults set by		Allowed limits by zabbix server	
				fping	server	min	max
packets	number	number of request packets to a target	-C		3	1	10000

interval	milliseconds	time to wait between successive packets	-p	1000		20	unlimited
size	bytes	packet size in bytes 56 bytes on x86, 68 bytes on x86_64	-b	56 or 68		24	65507
timeout	milliseconds	timeout to wait after last packet sent (affected by "-C" flag)	-t	500		50	unlimited

Warning: fping defaults can differ depending on platform and version - if in doubt, check fping documentation.

Zabbix writes addresses to be checked by any of three *icmpping** keys to a temporary file, which is then passed to **fping**. If items have different key parameters, only ones with identical key parameters are written to a single file.
All addresses written to the single file will be checked by fping in parallel mode, so Zabbix icmp pinger process will spend fixed amount of time disregarding count of addresses in the file.

2.0/manual/config/items/itemtypes/simple_checks.txt · Last modified: 2013/04/30 10:14 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

6 Log file monitoring

Overview

Zabbix can be used for centralized monitoring and analysis of log files with/without log rotation support.

Notifications can be used to warn users when a log file contains certain strings or string patterns.

To monitor a log file you must have:

- Zabbix agent running on the host
- log monitoring item set up

The size limit of a monitored log file depends on large file support.

Configuration

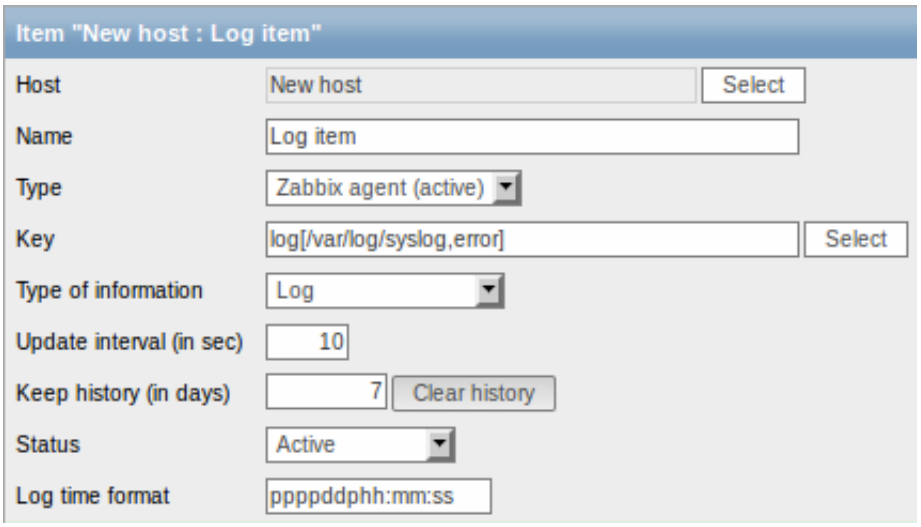
Verify agent parameters

Make sure that in the agent configuration file:

- 'Hostname' parameter matches the host name in the frontend
- Servers in the 'ServerActive' parameter are specified for the processing of active checks

Item configuration

Configure a log monitoring item:



Specifically for log monitoring items you must enter:

Type	Select **Zabbix agent (active)** here.
	Set either: **log**[file name with path to file,<regexp>,<encoding>,<maxlines>,<mode>] or **logrt**[filename format with path to file,<regexp>,<encoding>,<maxlines>,<mode>]

Key	For example: *log[/var/log/syslog]* *log[/var/log/syslog,error]* *logrt["/home/user/filelog_.*_[0-9]{1,3}","pattern_to_match","UTF-8",100]*. The last one will collect data from files such “filelog_abc_1” or “filelog__001”. For more details see **log** and **logrt** entries in the supported agent item keys section. Make sure that the file has read permissions for user 'zabbix' otherwise the item status will be set to 'unsupported'. Zabbix agent will filter entries of log file by the regexp if present.
Type of information	Select **Log** here.
Update interval (in sec)	The parameter defines how often Zabbix agent will check for any changes in the log file. Setting it to 1 second will make sure that you get new records as soon as possible.
Log time format	Supported placeholders: * **y**: *Year (0001-9999)* * **M**: *Month (01-12)* * **d**: *Day (01-31)* * **h**: *Hour (00-23)* * **m**: *Minute (00-59)* * **s**: *Second (00-59)* If left blank the timestamp will not be parsed. For example, consider the following line from the Zabbix agent log file: ” 23480:20100328:154718.045 Zabbix agent started. Zabbix 1.8.2 (revision 11211).” It begins with six character positions for PID, followed by date, time, and the rest of the line. Log time format for this line would be “ppppppp:yyyyMMdd:hhmmss”. Note that “p” and ”:” chars are just placeholders and can be anything but “yMdhms”.

Important notes

- The server and agent keep a trace of the monitored log's size and last modification time (for logrt) in two counters.
- The agent starts reading the log file from the point it stopped the previous time.
- The number of bytes already analyzed (the size counter) and last modification time (the time counter) are stored in the Zabbix database and are sent to the agent, to make sure it starts reading the log file from this point.
- Whenever the log file becomes smaller than the log size counter known by the agent, the counter is reset to zero and the agent starts reading the log file from the beginning taking the time counter into account.
- All files matching the filename format in the provided directory are analyzed every cycle the agent tries to get the next line from the log (for logrt).
- If there are several matching files with the same last modification time in the directory, then the agent will read lexicographically the smallest one.
- Zabbix agent processes new records of a log file once per *Update interval* seconds.
- Zabbix agent does not send more than **maxlines** of a log file per second. The limit prevents overloading of network and CPU resources and overrides the default value provided by **MaxLinesPerSecond** parameter in the agent configuration file.
- Additionally, log values are always limited to 50% of the agent send buffer size, even if there are no non-log values in it. So for the **maxlines** values to be sent in one connection (and not in several connections), the agent BufferSize parameter must be at least maxlines x 2.
- In the absence of log items all agent buffer size is used for non-log values. When log values come in they replace the older non-log values as needed, up to the designated 50%.

- Special note for "\" path separators: if file_format is "file\.log", then there should not be a "file" directory, since it is not possible to unambiguously define whether "." is escaped or is the first symbol of the file name.
- Regular expressions for **logrt** are supported in filename only, directory regular expression matching is not supported.

2.0/manual/config/items/itemtypes/log_items.txt · Last modified: 2013/05/02 16:29 by zalex_ua

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

7 Calculated items

7.1 Overview

With calculated items you can create calculations on the basis of other items.

Thus, calculated items are a way of creating virtual data sources. The values will be periodically calculated based on an arithmetical expression.

The resulting data will be stored in the Zabbix database as for any other item - this means storing both history and trend values for fast graph generation. Calculated items may be used in trigger expressions, referenced by macros or other entities same as any other item type.

To use calculated items, choose the item type **Calculated**.

7.2 Configurable fields

The **key** is a unique item identifier (per host). You can create any key name using supported symbols.

Calculation definition should be entered in the **Formula** field (named 'Expression' in 1.8.1 and 1.8.2). There is virtually no connection between the formula and the key. The key parameters are not used in formula in any way.

The correct syntax of a simple formula is:

```
func(<key>|<hostname:key>,<parameter1>,<parameter2>,...)
```

Where:

ARGUMENT	DEFINITION
func	One of the functions supported in trigger expressions: last, min, max, avg, count, etc
key	The key of another item whose data you want to use. It may be defined as **key** or **hostname:key**. *Note:* Putting the whole key in double quotes ("...") is strongly recommended to avoid incorrect parsing because of spaces or commas within the key. If there are also quoted parameters within the key, those double quotes must be escaped by using the backslash (\). See **Example 5** below.
parameter(s)	Any additional parameters that may be required.

All items that are referenced from the calculated item formula must exist and be collecting data. Also, if you change the item key of a referenced item, you have to manually update any formulas using that key.

User macros in the formula will be expanded if used to reference a parameter or a constant. User macros will NOT be expanded if used to reference a function, host name, item key or operator.

A more complex formula may use a combination of functions, operators and brackets. You can use all functions and operators supported in trigger expressions. Note that the syntax is slightly different, however logic and operator precedence are exactly the same.

Supported characters for a function:

```
a..zA..Z0..9_
```

Supported characters for a hostname:

```
a..zA..Z0..9 ._-
```

Supported characters for a key:

```
a..zA..Z0..9.,_
```

Unlike trigger expressions, Zabbix processes calculated items according to the item update interval, not upon receiving a new value.

A calculated item may become unsupported in several cases:

1. referenced item(s) not found
2. no data to calculate a function
3. division by zero
4. incorrect syntax used

Support for calculated items was introduced in Zabbix 1.8.1

7.3 Usage examples

Example 1

Calculating percentage of free disk space on '/'.

Use of function **last**:

```
100*last("vfs.fs.size[/,free]")/last("vfs.fs.size[/,total]")
```

Zabbix will take the latest values for free and total disk spaces and calculate percentage according to the given formula.

Example 2

Calculating a 10-minute average of the number of values processed by Zabbix.

Use of function **avg**:

```
avg("Zabbix Server:zabbix[wcache,values]",600)
```

Note that extensive use of calculated items with long time periods may affect performance of the Zabbix Server.

Example 3

Calculating total bandwidth on eth0.

Sum of two functions:

```
last("net.if.in[eth0,bytes]")+last("net.if.out[eth0,bytes]")
```

Example 4

Calculating percentage of incoming traffic.

More complex expression:

```
100*last("net.if.in[eth0,bytes]")/(last("net.if.in[eth0,bytes]")+last("net.if.out[eth0,bytes]"))
```

Example 5

Using aggregated items correctly within a calculated item.

Take note of how double quotes are escaped within the quoted key:

```
last("grpsum[\"video\",\"net.if.out[eth0,bytes]\",\"last\",\"0\"]") / last("grpsum[\"video\",\"nginx_stat.sh[active]\",\"last\",\"0\"]")
```

2.0/manual/config/items/itemtypes/calculated.txt · Last modified: 2012/09/07 13:33 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

8 Internal checks

8.1 Overview

Internal checks allow the monitoring of internals of Zabbix. To use this item, choose the item type **Zabbix internal**.

Internal checks are calculated by Zabbix server.

Internal checks are still processed by Zabbix pollers.

8.2 Supported checks

Key		
▲	**Description**	**Comments**
zabbix[boottime]		
	Startup time of Zabbix server process in seconds.	In seconds since the epoch.
zabbix[history]		
	Number of values stored in table HISTORY	Do not use if MySQL InnoDB, Oracle or PostgreSQL is used!
zabbix[history_log]		
	Number of values stored in table HISTORY_LOG	Do not use if MySQL InnoDB, Oracle or PostgreSQL is used! **This item is supported starting from version 1.8.3.**
zabbix[history_str]		
	Number of values stored in table HISTORY_STR	Do not use if MySQL InnoDB, Oracle or PostgreSQL is used!
zabbix[history_text]		
	Number of values stored in table HISTORY_TEXT	Do not use if MySQL InnoDB, Oracle or PostgreSQL is used! **This item is supported starting from version 1.8.3.**
zabbix[history_uint]		
	Number of values stored in table HISTORY_UINT	Do not use if MySQL InnoDB, Oracle or PostgreSQL is used! **This item is supported starting from version 1.8.3.**
zabbix[host,<type>,available]		
	Returns availability of a particular type of checks on the host. Value of this item corresponds to availability icons in the host list.	Valid types are: **agent**, **snmp**, **ipmi**, **jmx**. Return value: 0 (not available), 1 (available), 2 (unknown). **This item is supported starting from version 2.0.0.**

zabbix[items]		
	Number of items in Zabbix database	
zabbix[items_unsupported]		
	Number of unsupported items in Zabbix database	
zabbix[java,,<param>]		
	Returns information associated with Zabbix Java gateway.	If <param> is **ping**, "1" is returned. Can be used to check Java gateway availability using nodata() trigger function. If <param> is **version**, version of Java gateway is returned. Example: "2.0.0". Second parameter must be empty and is reserved for future use. **This item is supported starting from version 2.0.0.**
zabbix[process,<type>,<mode>,<state>]		
	Time a particular Zabbix process or a group of processes (identified by <type> and <mode>) spent in <state> in percentage. It is calculated for last minute only. If <mode> is Zabbix process number that is not running (for example, with 5 pollers running <mode> is specified to be 6), such an item will turn into unsupported state. Minimum and maximum refers to the usage percentage for a single process. So if in a group of 3 pollers usage percentages per process were 2, 18 and 66, min would return 2 and max would return 66.	The following process types are currently supported: **alerter** - process for sending notifications **configuration syncer** - process for managing in-memory cache of configuration data **db watchdog** - sender of a warning message in case DB is not available **discoverer** - process for discovery of devices **escalator** - process for escalation of actions **history syncer** - history DB writer **housekeeper** - process for removal of old historical data **http poller** - web monitoring poller **icmp pinger** - poller for icmpping checks **ipmi poller** - poller for IPMI checks **java poller** - poller for Java checks **node watcher** - process for sending historical data and configuration changes between nodes **poller** - normal poller for passive checks **proxy poller** - poller for passive proxies **self-monitoring** - process for collecting internal server statistics **timer** - process for evaluation of time-related trigger functions and maintenances **trapper** - trapper for active checks, traps, inter-node and -proxy communication **unreachable poller** - poller for unreachable devices Note: You can also see these

Processes report what they are doing in shared memory and the self-monitoring process summarizes that data each second. State changes (busy/idle) are registered upon change - thus a process that becomes busy registers as such and doesn't change or update the state until it becomes idle. This ensures that even fully hung processes will be correctly registered as 100% busy. Currently, "busy" means "not sleeping", but in the future additional states might be introduced - waiting for locks, performing database queries, etc. On Linux and most other systems, resolution is 1/100 of a second.	process types in a server log file. Valid modes are: **avg** - average value for all processes of a given type (default) **count** - returns number of forks for a given process type, <**state**> should not be specified **max** - maximum value **min** - minimum value <**process number**> - process number (between 1 and the number of pre-forked instances). For example, if 4 trappers are running, the value is between 1 and 4. Valid states are: **busy** - process is in busy state, for example, processing request (default). **idle** - process is in idle state doing nothing. Examples: zabbix[process,poller,avg,busy] - average time of poller processes spent doing something during the last minute zabbix[process,"icmp pinger",max,busy] - maximum time spent doing something by any ICMP pinger process during the last minute zabbix[process,trapper,count] - amount of currently running trapper processes **This item is supported starting from version 1.8.5.**
zabbix[proxy,<name>,<param>]	
Access to Proxy related information.	<name> - Proxy name List of supported parameters (<param>): lastaccess - timestamp of last heart beat message received from Proxy For example, zabbix[proxy,"Germany",lastaccess] Trigger function fuzzytime() can be used to check availability of proxies.
zabbix[queue,<from>,<to>]	
Number of server monitored items in the Queue which are delayed by <from> to <to> seconds, inclusive.	<from> - default: 6 seconds <to> - default: infinity Time-unit symbols (s,m,h,d,w) are supported for these parameters. **Parameters from and to are supported starting from version 1.8.3.**
zabbix[rcache,<cache>,<mode>]	
	Cache: **buffer** Mode:

	Availability statistics of Zabbix configuration cache.	**total** - total size of buffer **free** - size of free buffer **pfree** - percentage of free buffer **used** - size of used buffer
zabbix[requiredperformance]		
	Required performance of the Zabbix server, in new values per second expected.	Approximately correlates with "Required server performance, new values per second" in *Reports → Status of Zabbix*. Supported since Zabbix 1.6.2.
zabbix[trends]		
	Number of values stored in table TRENDS	Do not use if MySQL InnoDB, Oracle or PostgreSQL is used!
zabbix[trends_uint]		
	Number of values stored in table TRENDS_UINT	Do not use if MySQL InnoDB, Oracle or PostgreSQL is used! **This item is supported starting from version 1.8.3.**
zabbix[triggers]		
	Number of triggers in Zabbix database	
zabbix[uptime]		
	Uptime of Zabbix server process in seconds.	
zabbix[wcache,<cache>,<mode>]		
	Statistics and availability of Zabbix write cache.	

Cache	**Mode**		
values	all	Total number of values processed by Zabbix server, except unsupported items.	Counter.
	float	Number of processed float values.	Counter.
	uint	Number of processed unsigned integer values.	Counter.
	str	Number of processed character/string values.	Counter.
	log	Number of processed log items.	Counter.
	text	Number of processed text items.	Counter.
	not supported	Number of processed unsupported items.	Counter. *Not supported* mode is supported starting with Zabbix **1.8.6.**
history	pfree	Percentage of free history buffer.	A low number indicates performance problems on the database side.
	free	Size of free history buffer.	
	total	Total size of history buffer.	
	used	Size of used history buffer.	
trend	pfree	Percentage of free trend buffer.	
	free	Size of free trend buffer.	
	total	Total size of trend buffer.	
	used	Size of used trend buffer.	
	pfree	Percentage of free text history buffer.	
	free	Size of free text history buffer.	

	text			
		total	Total size of text history buffer.	
		used	Size of used text history buffer.	

2.0/manual/config/items/itemtypes/internal.txt · Last modified: 2013/11/05 07:18 by richlv

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

9 SSH checks

9.1 Overview

SSH checks are performed as agent-less monitoring. Zabbix agent is not needed for SSH checks.

To perform SSH checks Zabbix server must be initially configured with SSH2 support.

The minimum supported libssh2 library version is 1.0.0.

9.2 Configuration

9.2.1 Server configuration

By default, Zabbix server is not configured to perform SSH checks, thus any added SSH items won't work. To change this, open the Zabbix server configuration file (zabbix_server.conf) as `root` and look for the following line:

```
# SSHKeyLocation=
```

Uncomment it and set full path to a folder where public and private keys will be located:

```
SSHKeyLocation=/home/zabbix/.ssh
```

Save the file and restart zabbix_server afterwards.

/home/zabbix here is the home directory for the *zabbix* user account and *.ssh* is a directory where by default public and private keys will be generated by a ssh-keygen [http://en.wikipedia.org/wiki/Ssh-keygen] command inside the home directory.

Usually installation packages of zabbix-server from different OS distributions create the *zabbix* user account with a home directory in not very well-known places (as for system accounts). For example, for CentOS it's */var/lib/zabbix*, for Debian it's */var/run/zabbix*.

Before starting to generate the keys, an approach to reallocate the home directory to a better known place (intuitively expected) could be considered. This will correspond with the *SSHKeyLocation* Zabbix server configuration parameter mentioned above.

These steps can be skipped if *zabbix* account has been added manually according to the installation section because in this case most likely the home directory is already located at */home/zabbix*.

To change the setting for the *zabbix* user account all working processes which are using it have to be stopped:

```
# service zabbix-agent stop
# service zabbix-server stop
```

To change the home directory location with an attempt to move it (if it exists) a command should be executed:

```
# usermod -m -d /home/zabbix zabbix
```

It's absolutely possible that a home directory did not exist in the old place (in the CentOS for example), so it

should be created at the new place. A safe attempt to do that is:

```
# test -d /home/zabbix || mkdir /home/zabbix
```

To be sure that all is secure, additional commands could be executed to set permissions to the home directory:

```
# chown zabbix:zabbix /home/zabbix
# chmod 700 /home/zabbix
```

Previously stopped processes now can be started again:

```
# service zabbix-agent start
# service zabbix-server start
```

Now steps to generate public and private keys can be performed by a command:

```
# sudo -u zabbix ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/zabbix/.ssh/id_rsa):
Created directory '/home/zabbix/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/zabbix/.ssh/id_rsa.
Your public key has been saved in /home/zabbix/.ssh/id_rsa.pub.
The key fingerprint is:
90:af:e4:c7:e3:f0:2e:5a:8d:ab:48:a2:0c:92:30:b9 zabbix@it0
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|        .        |
|       o         |
| .      o        |
|+      . S       |
|.+    o =        |
|E .    * =       |
|=o . ..* .       |
|... oo.o+        |
+-----------------+
```

Note: public and private keys (*id_rsa.pub* and *id_rsa* respectively) have been generated by default in the */home/zabbix/.ssh* directory which corresponds to the Zabbix server *SSHKeyLocation* configuration parameter.

9.2.2 Shell configuration form

This step should be performed only once for every host that will be monitored by SSH checks.

By using the following command the **public** key file can be installed on a remote host *10.10.10.10* so that then SSH checks can be performed with a *root* account:

```
# sudo -u zabbix ssh-copy-id root@10.10.10.10
The authenticity of host '10.10.10.10 (10.10.10.10)' can't be established.
RSA key fingerprint is 38:ba:f2:a4:b5:d9:8f:52:00:09:f7:1f:75:cc:0b:46.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.10.10' (RSA) to the list of known hosts.
root@10.10.10.10's password:
Now try logging into the machine, with "ssh 'root@10.10.10.10'", and check in:
  .ssh/authorized_keys
```

```
to make sure we haven't added extra keys that you weren't expecting.
```

Now it's possible to check the SSH login using the default private key (*/home/zabbix/.ssh/id_rsa*) for *zabbix* user account:

```
# sudo -u zabbix ssh root@10.10.10.10
```

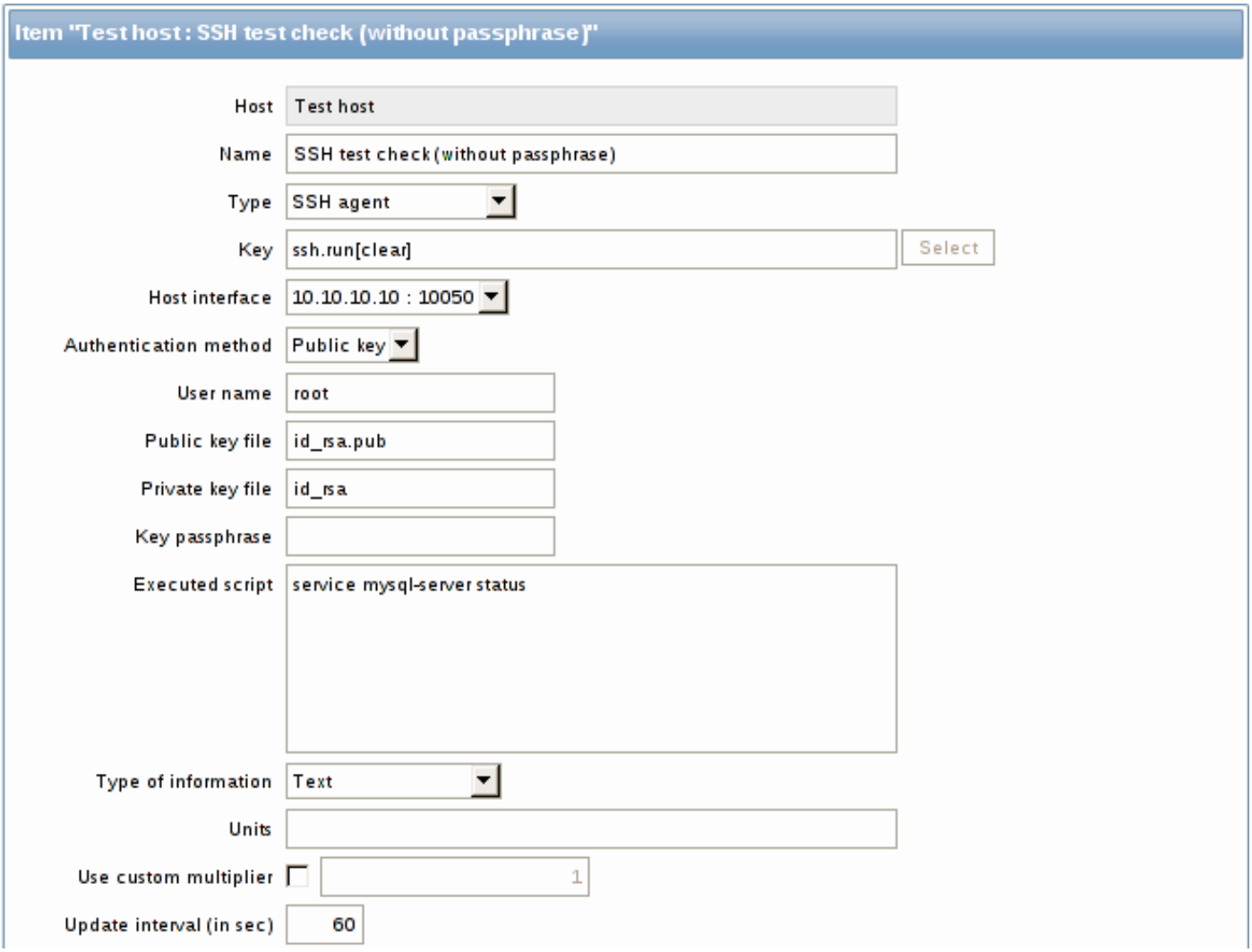
If the login is successful, then the configuration part in the shell is finished and remote SSH session can be closed.

9.2.3 Item configuration

Actual command(s) to be executed must be placed in the **Executed script** field in the item configuration. Multiple commands can be executed one after another by placing them on a new line. In this case returned values also will be formatted as multi lined.

Item parameter	Description	Comments
Key	Unique (per host) item key in format **ssh.run[<unique short description>,<ip>,<port>,<encoding>]**	<unique short description> is required and should be unique for all SSH items per host Default port is 22, not the port specified in the interface to which this item is assigned
Authentication method	One of the “Password” or “Public key”	
User name	User name to authenticate on remote host. Required	
Public key file	File name of public key if *Authentication method* is “Public key”. Required	Example: *id_rsa.pub* – default public key file name generated by a command ssh-keygen [http://en.wikipedia.org/wiki/Ssh-keygen]
Private key file	File name of private key if *Authentication method* is “Public key”. Required	Example: *id_rsa* – default private key file name
Password or **Key passphrase**	Password to authenticate or Passphrase **if** it was used for the private key	Leave the *Key passphrase* field empty if passphrase was not used
Executed script	Executed shell command(s) using SSH remote session	Examples: *date +%s* *service mysql-server status* *ps aux | grep httpd | wc -l*

The resulting item configuration should look like this:



Some Linux distributions like Debian, Ubuntu do not support encrypted private keys (with passphrase) if a libssh2 library installed from packages.
For more details see a report ZBX-4850 [https://support.zabbix.com/browse/ZBX-4850]

2.0/manual/config/items/itemtypes/ssh_checks.txt · Last modified: 2013/11/05 17:09 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

10 Telnet checks

10.1 Overview

Telnet checks are performed as agent-less monitoring. Zabbix agent is not needed for Telnet checks.

10.2 Configurable fields

Actual command(s) to be executed must be placed in the **Executed script** field in the item configuration. Multiple commands can be executed one after another by placing them on a new line. In this case returned value also will be formated as multi lined.

Supported characters that the prompt can end with:

- $
- #
- >
- %

A telnet prompt line which ended with one of these characters will be removed from the returned value, but only for the first command in the commands list, i.e. only at a start of the telnet session.

Key	Description	Comments
telnet.run[<unique short description>,<ip>,<port>,<encoding>]	Run a command on a remote device using telnet connection	

If a telnet check returns a value with non-ASCII characters and in non-UTF8 encoding then the *<encoding>* parameter of the key should be properly specified. See encoding of returned values page for more details.

2.0/manual/config/items/itemtypes/telnet_checks.txt · Last modified: 2012/04/12 16:13 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

11 External checks

11.1 Overview

External check is a check executed by Zabbix server by running a shell script or a binary.

External checks do not require any agent running on a host being monitored.

The syntax of the item key is:

```
script[<parameter1>,<parameter2>,...]
```

Where:

ARGUMENT	DEFINITION
script	Name of a shell script or a binary.
parameter(s)	Optional command line parameters.

If you don't want to pass any parameters to the script you may use:

```
script[] or
script
```

Zabbix server will look in the directory defined as the location for external scripts (parameter 'ExternalScripts' in Zabbix Server configuration file) and execute the command. The command will be executed as the user Zabbix server runs as, so any access permissions or environment variables should be handled in a wrapper script, if necessary, and permissions on the command should allow that user to execute it. Only commands in the specified directory are available for execution.

Zabbix uses the standard output of the script as the value (the full output with trimmed trailing whitespace is returned since Zabbix 2.0). Standard error and exit codes are discarded.

Do not overuse external checks! It can decrease performance of the Zabbix system a lot.

11.2 Usage example

Executing the script **check_oracle.sh** with parameters "-h <host IP address>".

```
check_oracle.sh["-h","{HOST.CONN}"]
```

Zabbix will execute:

```
check_oracle.sh "-h" "192.168.1.4"
```

2.0/manual/config/items/itemtypes/external.txt · Last modified: 2012/01/26 14:06 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

12 Aggregate checks

12.1 Overview

In aggregate checks Zabbix server collects aggregate information by doing direct database queries.

Aggregate checks do not require any agent running on the host being monitored.

The syntax of the aggregate item key is:

```
groupfunc["Host group","Item key",itemfunc,timeperiod]
```

Multiple host groups may be used since Zabbix 1.8.2 by inserting a comma-delimited array.

Supported group functions (groupfunc) are:

GROUP FUNCTION	DESCRIPTION
grpavg	Average value
grpmax	Maximum value
grpmin	Minimum value
grpsum	Sum of values

Supported item functions (itemfunc) are:

ITEM FUNCTION	DESCRIPTION
avg	Average value
count	Number of values
last	Last value
max	Maximum value
min	Minimum value
sum	Sum of values

The last **timeperiod** parameter specifies a time period of latest collected values. Supported unit symbols can be used in this parameter for convenience, for example '5m' (minutes) instead of '300' (seconds) or '1d' (day) instead of '86400' (seconds).

The `timeperiod` parameter is ignored by the server if the third parameter (item function) is *last*.

Amount of values (prefixed with #) is not supported.

Only active items on enabled hosts are included in the calculations.

12.2 Usage examples

Examples of keys for aggregate checks:

Example 1

Total disk space of host group 'MySQL Servers'.

```
grpsum["MySQL Servers","vfs.fs.size[/,total]",last,0]
```

Example 2

Average processor load of host group 'MySQL Servers'.

```
grpavg["MySQL Servers","system.cpu.load[,avg1]",last,0]
```

Example 3

5-minute average of the number of queries per second for host group 'MySQL Servers'.

```
grpavg["MySQL Servers",mysql.qps,avg,5m]
```

Example 4

Average CPU load on all hosts in multiple host groups.

```
grpavg[["Servers A","Servers B","Servers C"],system.cpu.load,last,0]
```

2.0/manual/config/items/itemtypes/aggregate.txt · Last modified: 2013/02/14 11:22 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

13 Trapper items

Overview

Trapper items accept incoming data instead of querying for it.

It is useful for any data you might want to "push" into Zabbix.

To use a trapper item you must:

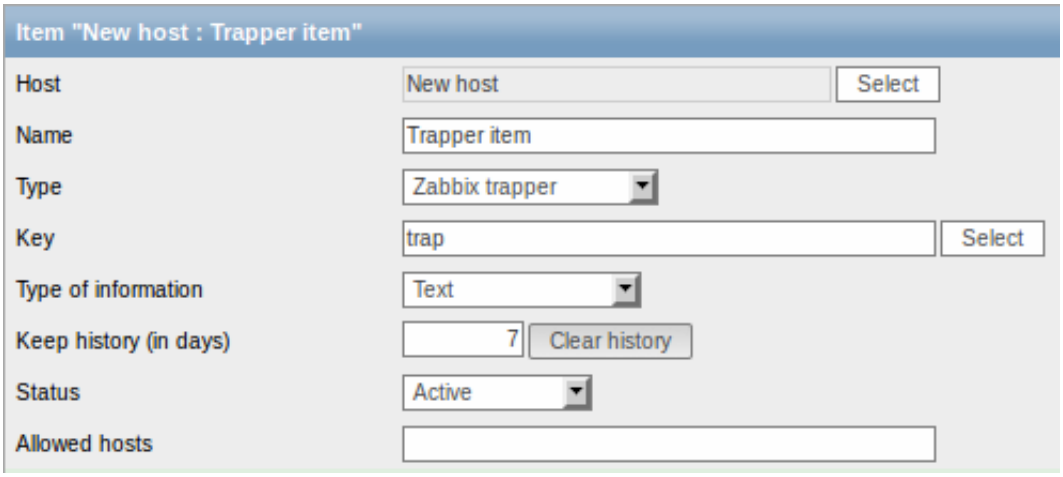
- have a trapper item set up in Zabbix
- send in the data into Zabbix

Configuration

Item configuration

To configure a trapper item:

- Go to: *Configuration* → *Hosts*
- Click on *Items* in the row of the host
- Click on *Create item*
- Enter parameters of the item in the form



The fields that require specific information for trapper items are:

Type	Select **Zabbix trapper** here.
Key	Enter a key that will be used to recognize the item when sending in data.
Type of information	Select the type of information that will correspond the format of data that will be sent in.
Allowed hosts	If specified, the trapper will accept incoming data only from this comma-delimited list of hosts. No spaces allowed.

You may have to wait up to 60 seconds after saving the item until the server picks up the changes from a configuration cache update, before you can send in values.

Sending in data

In the simplest of cases, we may use zabbix_sender utility to send in some 'test value':

```
zabbix_sender -z <server IP address> -p 10051 -s "New host" -k trap -o "test value"
```

To send in the value we use these keys:

-z - to specify Zabbix server IP address

-p - to specify Zabbix server port number (10051 by default)

-s - to specify the host (make sure to use the 'technical' host name here, instead of the 'visible' name)

-k - to specify the key of the item we just defined

-o - to specify the actual value to send

Display

This is the result in *Monitoring → Latest data*:

2.0/manual/config/items/itemtypes/trapper.txt · Last modified: 2013/08/22 13:11 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

14 JMX monitoring

14.1 Overview

JMX monitoring can be used to monitor JMX counters of a Java application.

In Zabbix 1.8, if you wanted to monitor JMX counters of a Java application, your best choice would have been the Zapcat JMX Zabbix Bridge [http://www.kjkoster.org/zapcat/]. You would either modify the source code of your application to reference the Zapcat JAR file and programmatically start a Zabbix agent, or you would install a ready-made Zapcat plugin for applications that support it (such as Jetty or Tomcat).

Zabbix 2.0 adds native support for JMX monitoring by introducing a new Zabbix daemon called “Zabbix Java gateway”.

When Zabbix server wants to know the value of a particular JMX counter on a host, it asks the Zabbix **Java gateway**, which in turn uses the JMX management API [http://java.sun.com/javase/technologies/core/mntr-mgmt/javamanagement/] to query the application of interest remotely.

For more details on Zabbix Java gateway, including where to get it and how to set it up see this section of the manual.

14.2 Enabling remote JMX monitoring for Java application

A Java application does not need any additional software installed, but it needs to be started with the command-line options specified below to have support for remote JMX monitoring.

As a bare minimum, if you just wish to get started by monitoring a simple Java application on a local host with no security enforced, start it with these options:

```
java \
-Dcom.sun.management.jmxremote \
-Dcom.sun.management.jmxremote.port=12345 \
-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
```

This makes Java listen for incoming JMX connections on port 12345 and tells it not to require authentication or SSL.

If you wish to be more stringent about security, there are many other Java options available to you. For instance, the next example starts the application with a more versatile set of options and opens it to a wider network, not just local host.

```
java \
-Djava.rmi.server.hostname=192.168.3.14 \
-Dcom.sun.management.jmxremote \
-Dcom.sun.management.jmxremote.port=12345 \
-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
```

Most (if not all) of these settings can be specified in /etc/java-6-openjdk/management/management.properties (or wherever that file is on your system).

Note that if you wish to use SSL, you have to modify startup.sh script by adding `-Djavax.net.ssl.*` options to Java gateway, so that it knows where to find key and trust stores.

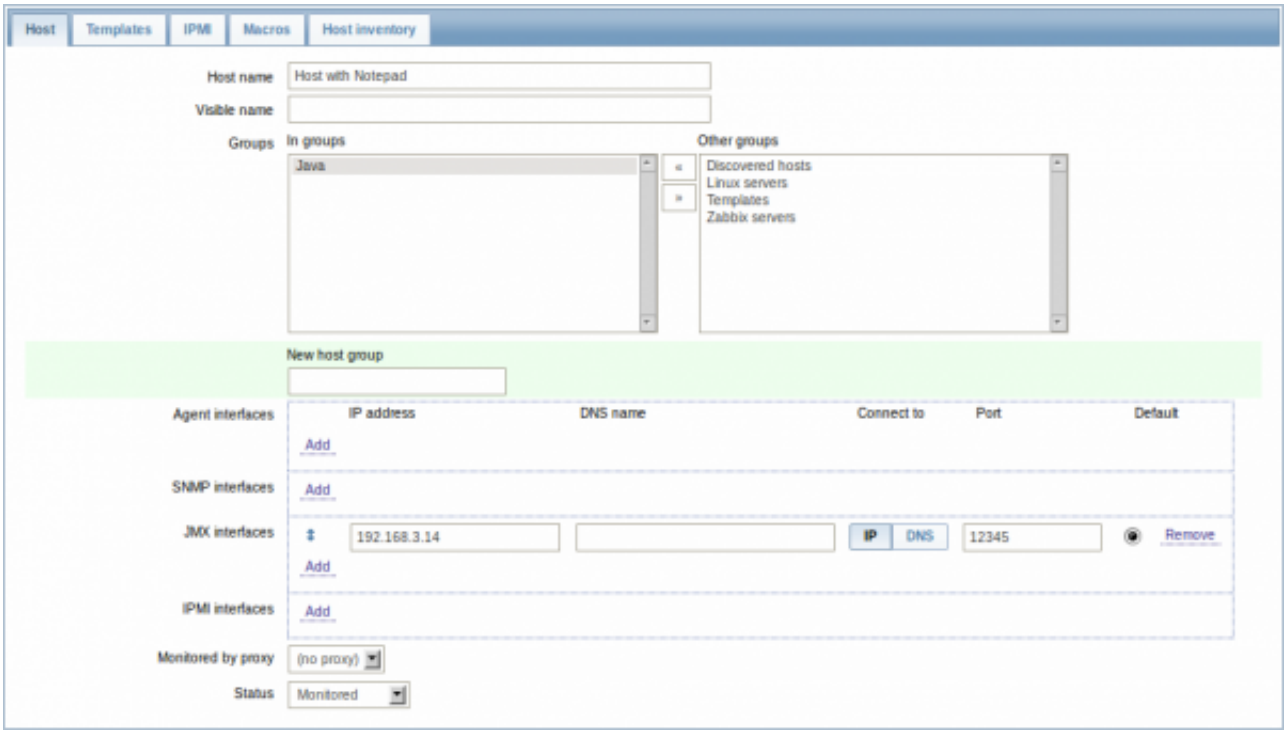
See Monitoring and Management Using JMX [http://download.oracle.com/javase/1.5.0/docs/guide/management/agent.html] for a detailed description.

14.3 Configuring JMX interfaces and items in Zabbix GUI

With Java gateway running, server knowing where to find it and a Java application started with support for remote JMX monitoring, it is time to configure the interfaces and items in Zabbix GUI.

Configuring JMX interface

You begin by creating a JMX-type interface on the host of interest:



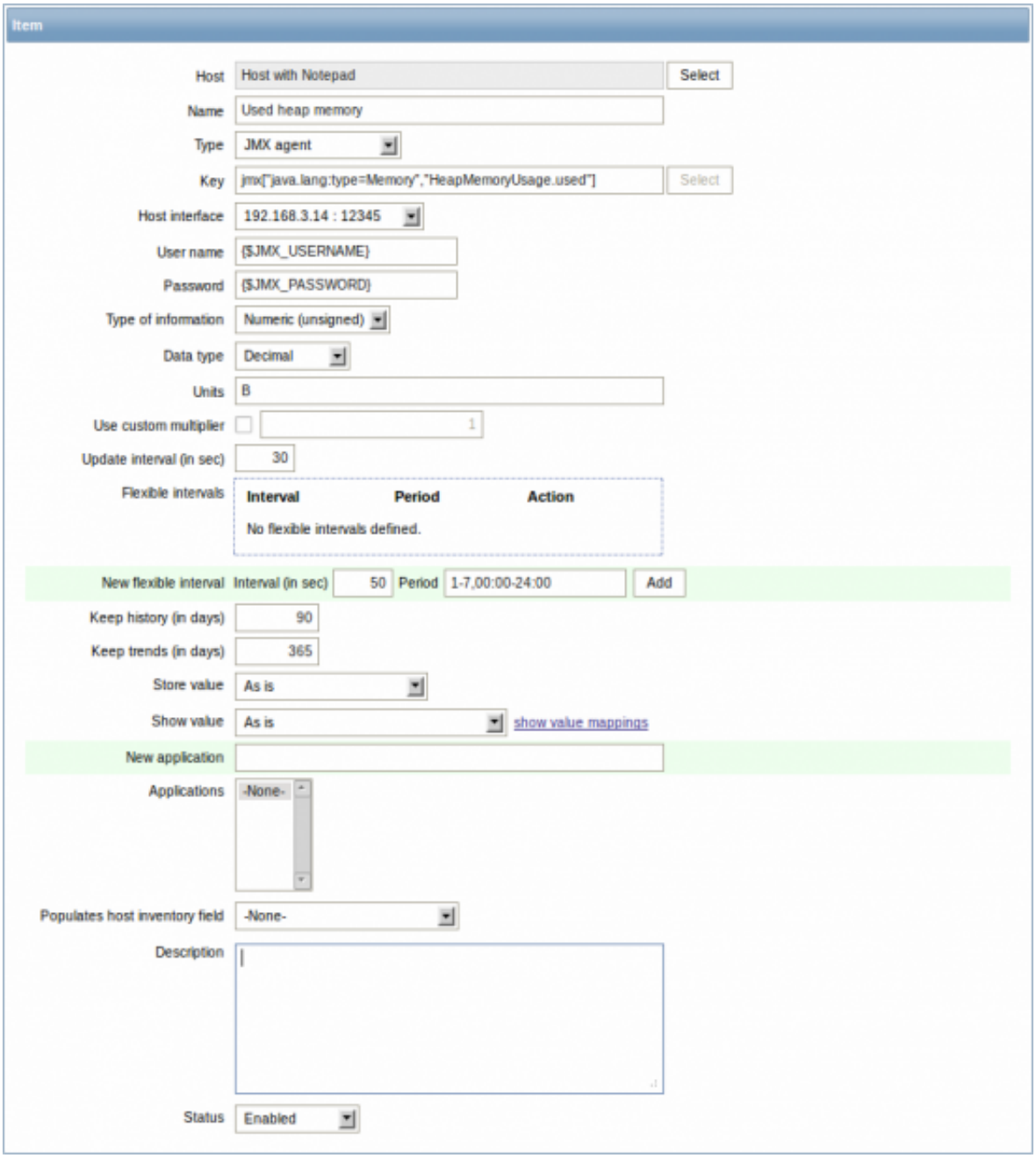
Adding JMX agent item

For each JMX counter you are interested in you add an item of type **JMX agent** attached to that interface. If you have configured authentication on your Java application, then you also specify username and password.

The key in the screenshot below says `jmx["java.lang:type=Memory","HeapMemoryUsage.used"]`. The JMX item key syntax is similar to Zapcat items, except that a comma is used for separating arguments instead of "][". The key consists of 2 parameters:

- object name - which represents the object name of an MBean
- attribute name - an MBean attribute name with optional composite data field names separated by dots

See below for more detail on JMX item keys.



If you wish to monitor a Boolean counter that is either “true” or “false”, then you specify type of information as “Numeric (unsigned)” and data type as “Boolean”. Server will store Boolean values as 1 or 0, respectively.

JMX item keys in more detail

Simple attributes

An MBean object name is nothing but a string which you define in your Java application. An attribute name, on the other hand, can be more complex. In case an attribute returns primitive data type (an integer, a string etc.)

there is nothing to worry about, the key will look like this:

```
jmx[com.example:Type=Hello,weight]
```

In this example an object name is “com.example:Type=Hello”, attribute name is “weight” and probably the returned value type should be “Numeric (float)”.

Attributes returning composite data

It becomes more complicated when your attribute returns composite data. For example: your attribute name is “apple” and it returns a hash representing its parameters, like “weight”, “color” etc. Your key may look like this:

```
jmx[com.example:Type=Hello,apple.weight]
```

This is how an attribute name and a hash key are separated, by using a dot symbol. Same way, if an attribute returns nested composite data the parts are separated by a dot:

```
jmx[com.example:Type=Hello,fruits.apple.weight]
```

Problem with dots

So far so good. But what if an attribute name or a hash key contains dot symbol? Here is an example:

```
jmx[com.example:Type=Hello,all.fruits.apple.weight]
```

That's a problem. How to tell Zabbix that attribute name is “all.fruits”, not just “all”? How to distinguish a dot that is part of the name from the dot that separates an attribute name and hash keys?

Before **2.0.4** Zabbix Java gateway was unable to handle such situations and users were left with UNSUPPORTED items. Since 2.0.4 this is possible, all you need to do is to escape the dots that are part of the name with a backslash:

```
jmx[com.example:Type=Hello,all\.fruits.apple.weight]
```

Same way, if your hash key contains a dot you escape it:

```
jmx[com.example:Type=Hello,all\.fruits.apple.total\.weight]
```

Other issues

A backslash character should be escaped as well:

```
jmx[com.example:type=Hello,c:\\documents]
```

If the object name or attribute name contains spaces or commas double-quote it:

```
jmx["com.example:Type=Hello","fruits.apple.total weight"]
```

This is actually all there is to it. Happy JMX monitoring!

2.0/manual/config/items/itemtypes/jmx_monitoring.txt · Last modified: 2013/10/04 15:48 by Heilig

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

15 ODBC monitoring

15.1 Overview

ODBC monitoring corresponds to the *Database monitor* item type in the Zabbix frontend.

ODBC is a C programming language middle-ware API for accessing database management systems (DBMS). The ODBC concept was developed by Microsoft and later ported to other platforms.

Zabbix may query any database, which is supported by ODBC. To do that, Zabbix does not directly connect to the databases, but uses the ODBC interface and drivers set up in ODBC. This function allows for more efficient monitoring of different databases for multiple purposes - for example, checking specific database queues, usage statistics and so on. Zabbix supports unixODBC and iODBC, which are the two most commonly used open source ODBC API implementations.

15.2 Installing unixODBC

The suggested way of installing unixODBC is to use the Linux operating system default package repositories. In the most popular Linux distributions unixODBC is included in the package repository by default. If it's not available, it can be obtained at the unixODBC homepage: http://www.unixodbc.org/download.html [http://www.unixodbc.org/download.html].

unixODBC installation using the *yum* package manager:

```
shell> yum -y install unixODBC unixODBC-devel
```

The unixODBC-devel package is needed to compile Zabbix with unixODBC support.

15.3 Installing unixODBC drivers

A unixODBC database driver should be installed for the database, which will be monitored. unixODBC has a list of supported databases and drivers: http://www.unixodbc.org/drivers.html [http://www.unixodbc.org/drivers.html]. In some Linux distributions database drivers are included in package repositories.

For example, a MySQL database driver can be installed using the *yum* package manager:

```
shell> yum install mysql-connector-odbc
```

15.4 Configuring unixODBC

ODBC configuration is done by editing the **odbcinst.ini** and **odbc.ini** files. To verify the configuration file location, type:

```
shell> odbcinst -j
```

odbcinst.ini is used to list the installed ODBC database drivers:

```
[mysql]
Description = ODBC for MySQL
Driver      = /usr/lib/libmyodbc5.so
```

Parameter details:

Attribute	Description
mysql	Database driver name.
Description	Database driver description.
Driver	Database driver library location.

odbc.ini is used to define data sources:

```
[test]
Description = MySQL test database
Driver      = mysql
Server      = 127.0.0.1
User        = root
Password    =
Port        = 3306
Database    = zabbix
```

Parameter details:

Attribute	Description
test	Data source name (DSN).
Description	Data source description.
Driver	Database driver name - as specified in odbcinst.ini
Server	Database server IP/DNS.
User	Database user for connection.
Password	Database user password.
Port	Database connection port.
Database	Database name.

To verify if ODBC connection is working successfully, a connection to database should be tested. That can be done with the **isql** utility (included in the unixODBC package):

```
shell> isql test
+---------------------------------------+
| Connected!                            |
|                                       |
| sql-statement                         |
| help [tablename]                      |
| quit                                  |
|                                       |
+---------------------------------------+
SQL>
```

15.5 Compiling Zabbix with ODBC support

To enable ODBC support, Zabbix should be compiled with one of following flags:

```
  --with-iodbc[=ARG]      use odbc driver against iODBC package [default=no],
  --with-unixodbc[=ARG]   use odbc driver against unixODBC package
```

See more about Zabbix installation from the source code.

15.6 Item configuration in Zabbix frontend

Configure a database monitoring item:

Specifically for database monitoring items you must enter:

Type	Select *Database monitor* here.
Key	Enter **db.odbc.select**[unique_description] The unique description will serve to identify the item in triggers etc.
Additional parameters	**DSN** - data source name (as specified in odbc.ini) **user** - database user name (optional if user is specified in odbc.ini) **password** - database user password (optional if password is specified in odbc.ini) **sql** - SQL query
Type of information	It is important to know what type of information will be returned by the query, so that it is selected correctly here. With an incorrect *type of information* the item will turn unsupported.

15.7 Important notes

- The query must not be executing longer than the Timeout parameter on the server. Starting from Zabbix 2.0.8 the Timeout parameter value is also used as a ODBC login timeout (note that depending on ODBC drivers the login timeout setting might be ignored).
- The query must return one value only.
- If a query returns more than one column, only the first column is read.
- If a query returns more than one line, only the first line is read.
- The SQL command must begin with `select`.
- The SQL command mustn't contain any line breaks.

15.8 Error messages

Starting from Zabbix 2.0.8 the ODBC error messages are structured into fields to provide more detailed information. Example:

```
Cannot execute ODBC query:[SQL_ERROR]:[42601][7][ERROR: syntax error at or near ";"; Error while executing the query]|
-------------------------  ----------  -----  |  ------------------------------------------------------------------------ |
            |                   |         |    `- Native error code            `- error message.                      `- Record separator
            |                   |         `-SQLState
            `- Zabbix message   `- ODBC return code
```

Note that the error message length is limited to 128 bytes, so the message can be truncated. If there is more than one ODBC diagnostic record Zabbix tries to concatenate them as far as the length limit allows.

2.0/manual/config/items/itemtypes/odbc_checks.txt · Last modified: 2013/08/01 17:07 by wiper
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 History and trends

Overview

History and trends are the two ways of storing collected data in Zabbix.

Whereas history keeps each collected value, trends keep averaged information on hourly basis and therefore are less resource-hungry.

Keeping history

You can set for how many days history will be kept:

- in the item properties form
- when mass-updating items

Any older data will be removed by the Housekeeper.

The general strong advice is to keep history for the smallest possible number of days and that way not to overload the database with lots of historical values.

Instead of keeping a long history, you can keep longer data of trends. For example, you could keep history for 14 days and trends for 5 years.

You can get a good idea of how much space is required by history versus trends data by referring to the database sizing page.

While keeping shorter history, you will still be able to review older data in graphs, as graphs will use trends values for displaying older data.

If history is set to '0', Zabbix will only be able to calculate triggers that check the last value. Historical values would not be stored in the database at all, except for the last value for the item itself.

Keeping trends

Trends is a built-in historical data reduction mechanism where for every hour minimum, maximum and average values are stored, as well as the total number of values in that hour.

You can set for how many days trends will be kept:

- in the item properties form
- when mass-updating items

Trends usually can be kept for much longer than history. Any older data will be removed by the Housekeeper.

If trends are set to '0', Zabbix server does not calculate or store trends at all.

2.0/manual/config/items/history_and_trends.txt · Last modified: 2011/11/09 12:41 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

4 User parameters

Overview

Sometimes you may want to run an agent check that does not come predefined with Zabbix. This is where user parameters come to help.

You may write a command that retrieves the data you need and include it in the user parameter in the agent configuration file ('UserParameter' configuration parameter).

A user parameter has the following syntax:

```
UserParameter=<key>,<command>
```

As you can see, a user parameter also contains a key. The key will be necessary when configuring an item. Enter a key of your choice that will be easy to reference (it must be unique within a host). Restart the agent.

Then, when configuring an item, enter the key to reference the command from the user parameter you want executed.

User parameters are commands executed by Zabbix agent. Up to 512KB of data can be returned (it used to be 64KB before Zabbix 2.0.5). **/bin/sh** is used as a command line interpreter under UNIX operating systems. This way you can enhance the functionality of Zabbix agents.

See a step-by-step tutorial on making use of user parameters.

Examples of simple user parameters

A simple command:

```
UserParameter=ping,echo 1
```

The agent will always return '1' for an item with 'ping' key.

A more complex example:

```
UserParameter=mysql.ping,mysqladmin -uroot ping|grep -c alive
```

The agent will return '1', if MySQL server is alive, '0' - otherwise.

Flexible user parameters

Flexible user parameters accept parameters with the key. This way a flexible user parameter can be the basis for creating several items.

Flexible user parameters have the following syntax:

```
UserParameter=key[*],command
```

Parameter	Description
Key	Unique item key. The [*] defines that this key accepts parameters within the brackets.

Command	Command to be executed to evaluate value of the key. Zabbix parses the content of [] and substitutes $1,...,$9 in the command accordingly. $0 will be substituted by the original command (prior to expansion of $0,...,$9) to be run.

To use positional references unaltered, specify double dollar sign - for example, awk '{print $$2}'.

Unless UnsafeUserParameters agent daemon configuration option is enabled, it is not allowed to pass flexible parameters containing these symbols: \ ' " ` * ? [ ] { } ~ $! & ; ( ) < > | # @

User parameters that return text (character, log, text types of information) now can return whitespace only as well, setting the return value to an empty string (supported since 2.0). If non-valid value is returned, ZBX_NOTSUPPORTED will be sent back by the agent.

Example 1

Something very simple:

```
UserParameter=ping[*],echo $1
```

We may define unlimited number of items for monitoring all having format ping[something].

- ping[0] - will always return '0'
- ping[aaa] - will always return 'aaa'

Example 2

Let's add more sense!

```
UserParameter=mysql.ping[*],mysqladmin -u$1 -p$2 ping | grep -c alive
```

This parameter can be used for monitoring availability of MySQL database. We can pass user name and password:

```
mysql.ping[zabbix,our_password]
```

Example 3

How many lines matching a regular expression in a file?

```
UserParameter=wc[*],grep -c "$2" $1
```

This parameter can be used to calculate number of lines in a file.

```
wc[/etc/passwd,root]
wc[/etc/services,zabbix]
```

2.0/manual/config/items/userparameters.txt · Last modified: 2013/05/09 16:59 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Extending Zabbix agents

This tutorial provides step-by-step instructions on how to extend the functionality of Zabbix agent with the use of a user parameter.

Step 1

Write a script or command line to retrieve required parameter.

For example, we may write the following command in order to get total number of queries executed by a MySQL server:

```
mysqladmin -uroot status|cut -f4 -d":"|cut -f1 -d"S"
```

When executed, the command returns total number of SQL queries.

Step 2

Add this command to agent's configuration file.

Add the command to zabbix_agentd.conf:

```
UserParameter=mysql.questions,mysqladmin -uroot status|cut -f4 -d":"|cut -f1 -d"S"
```

mysql.questions is an unique identifier. It can be any string, for example, queries.

Test this parameter by using zabbix_get utility.

Step 3

Restart Zabbix agent.

Agent will reload configuration file.

Step 4

Add new item for monitoring.

Add new item with Key=mysql.questions to the monitored host. Type of the item must be either Zabbix Agent or Zabbix Agent (active).

Be aware that type of returned values must be set correctly on Zabbix server. Otherwise Zabbix won't accept them.

2.0/manual/config/items/userparameters/extending_agent.txt · Last modified: 2012/05/28 09:37 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

5 Windows performance counters

Overview

You can effectively monitor Windows performance counters using the *perf_counter[]* key.

For example:

```
perf_counter["\Processor(0)\Interrupts/sec"]
```

or

```
perf_counter["\Processor(0)\Interrupts/sec", 10]
```

For more information on using this key, see WIN32-specific item keys.

In order to get a full list of performance counters available for monitoring, you may run:

```
typeperf -qx
```

Numeric representation

As the naming of performance counters may differ on different Windows servers, depending on local settings, it introduces a certain problem when creating a template for monitoring several Windows machines having different locales.

At the same time every performance counter can also be referred to by its numeric form, which is unique and exactly the same regardless of language settings, so you might use the numeric representation instead of strings.

To find out the numeric equivalents, run **regedit**, then find *HKEY_LOCAL_MACHINE|SOFTWARE|Microsoft|Windows NT|CurrentVersion|Perflib|009*.

The registry entry contains information like this:

```
1
1847
2
System
4
Memory
6
% Processor Time
10
File Read Operations/sec
12
File Write Operations/sec
14
File Control Operations/sec
16
File Read Bytes/sec
18
File Write Bytes/sec
....
```

Here you can find the corresponding numbers for each string part of the performance counter, like in '\System\% Processor Time':

```
System -> 2
% Processor Time -> 6
```

Then you can use these numbers to represent the path in numbers:

```
\2\6
```

User parameters

You can deploy some user parameters for the monitoring of Windows performance counters.

For example, you can add these to the Zabbix agent configuration file:

```
   PerfCounter=UserPerfCounter1,"\Memory\Page Reads/sec",30
   or
   PerfCounter=UserPerfCounter2,"\4\24",30
```

With such parameters in place, you can then simply use *UserPerfCounter1* or *UserPerfCounter2* as the keys for creating the respective items.

Remember to restart Zabbix agent after making changes to the configuration file.

2.0/manual/config/items/perfcounters.txt · Last modified: 2012/05/25 09:50 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

6 Mass update

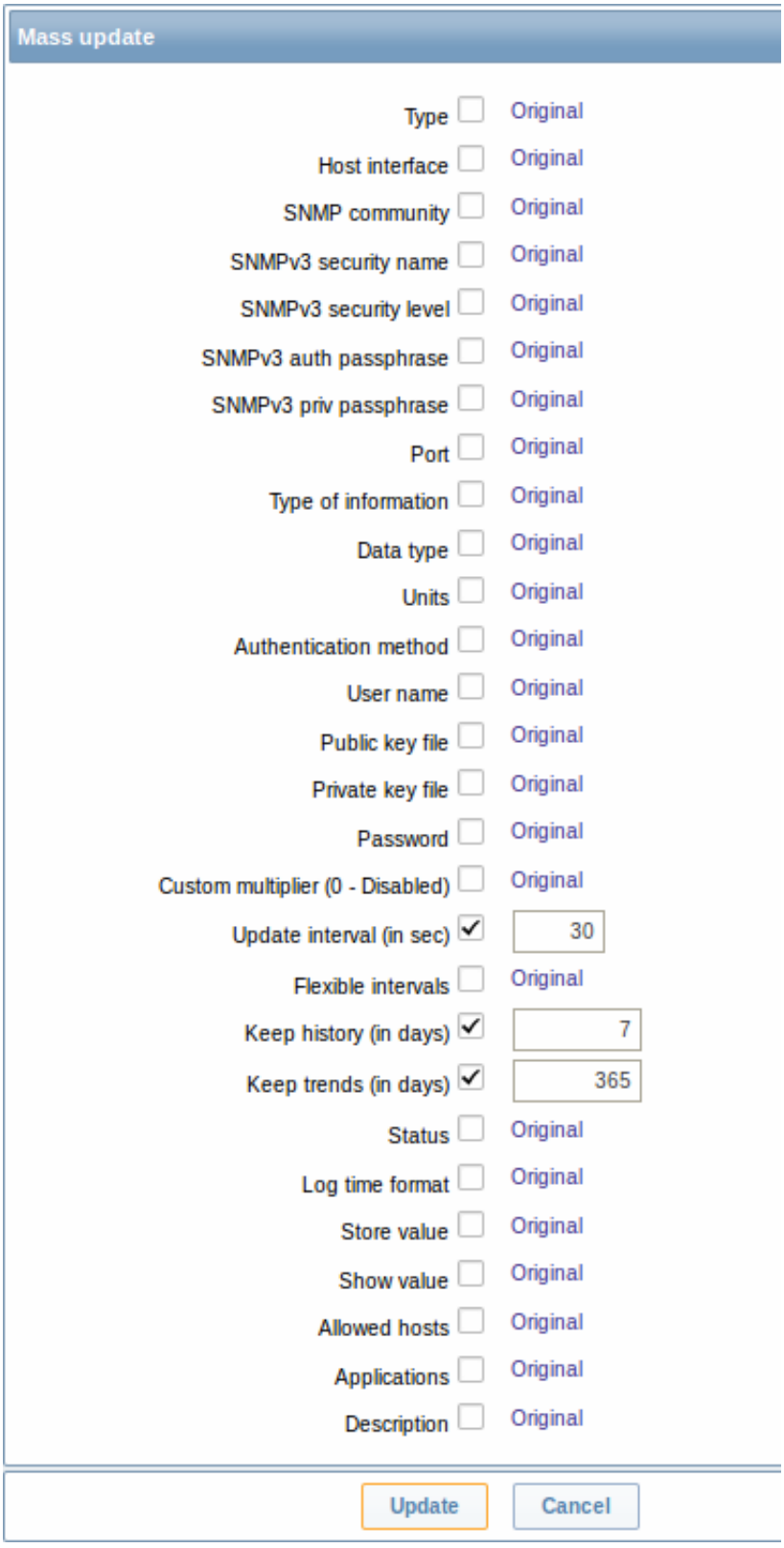
Overview

Sometimes you may want to change some attribute for a number of items at once. Instead of opening each individual item for editing, you may use the mass update function for that.

Using mass update

To mass-update some items, do the following:

- Mark the checkboxes of the items to update in the list
- Select *Mass update* from the dropdown below and click on *Go*
- Mark the checkboxes of the attributes to update
- Enter new values for the attributes and click on *Update*



2.0/manual/config/items/itemupdate.txt · Last modified: 2012/07/11 09:33 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

7 Value mapping

Overview

For a more “human” representation of received values, you can use value maps that contain the mapping between numeric values and string representations.

Value mappings can be used in both the Zabbix front-end and notifications sent by email/SMS/jabber etc.

For example, an item which has value '0' or '1' can use value mapping to represent the values in a human-readable form:

- '0' ⇒ 'Not Available'
- '1' ⇒ 'Available'

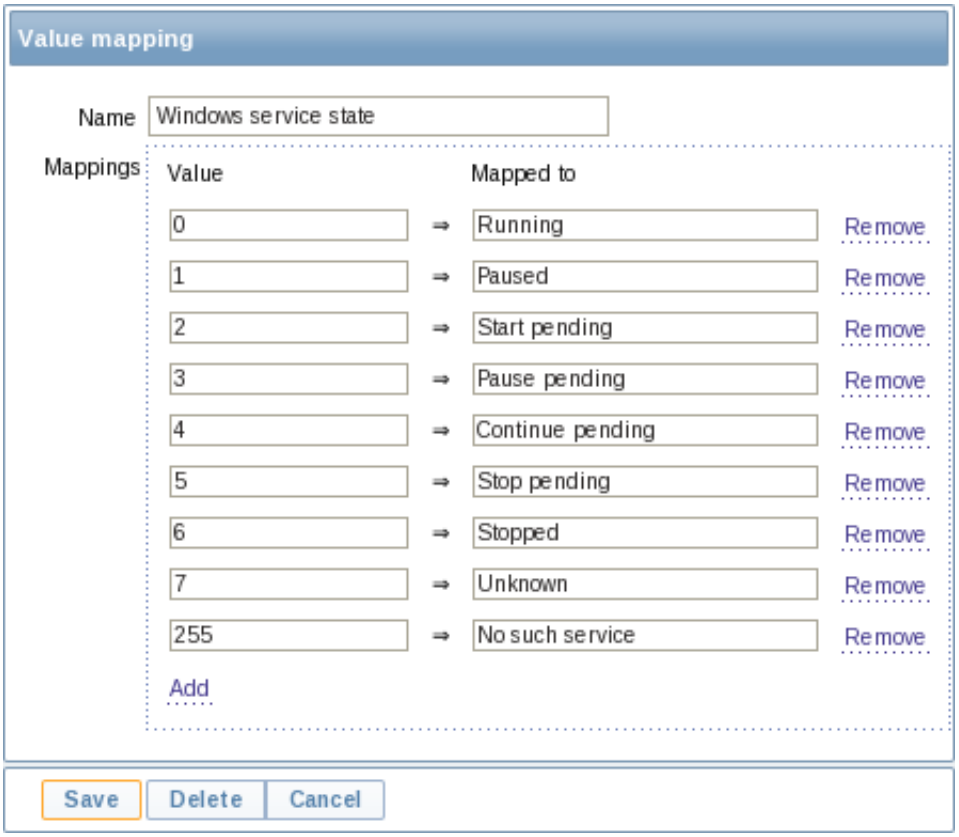
Thus, when configuring items you can use a value map to “humanize” the way an item value will be displayed. To do that, you refer to the name of a previously defined value map in the *Show value* field.

Value mapping can only be used with items having type *Numeric (unsigned)*.

Configuration

To define a value map:

- Go to: *Administration → General*
- Select *Value mapping* from the dropdown
- Click on *Create value map* (or on the name of an existing map)

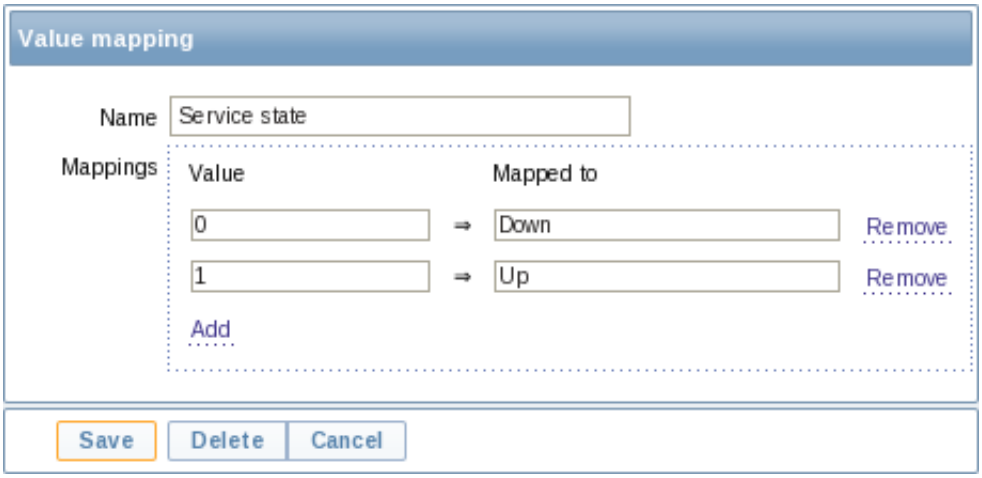


Parameters of a value map:

Parameter	Description
Name	Unique name of a set of value mappings.
Mapping	Individual mappings - pairs of numeric values and their string representations.
New mapping	A single mapping for addition.

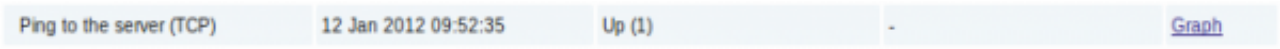
How this works

For example, one of the predefined agent items 'Ping to the server (TCP)' uses an existing value map called 'Service state' to display its values.



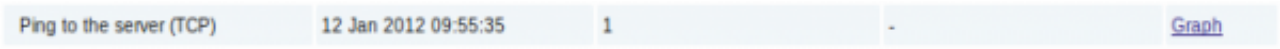
In the item configuration form you can see a reference to this value map in the *Show value* field:

So in *Monitoring → Latest data* the mapping is put to use to display 'Up' (with the raw value in parentheses).



A value being displayed in a human-readable form is also easier to understand when receiving notifications.

Without a predefined value map you would only get this:



So in this case you would either have to guess what the '1' stands for or do a search of documentation to find out.

2.0/manual/config/items/mapping.txt · Last modified: 2012/09/26 21:01 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

8 Applications

Overview

Applications are used to group items in logical groups.

For example, the *MySQL Server* application can hold all items related to the MySQL server: availability of MySQL, disk space, processor load, transactions per second, number of slow queries, etc.

Applications are also used for grouping web scenarios.

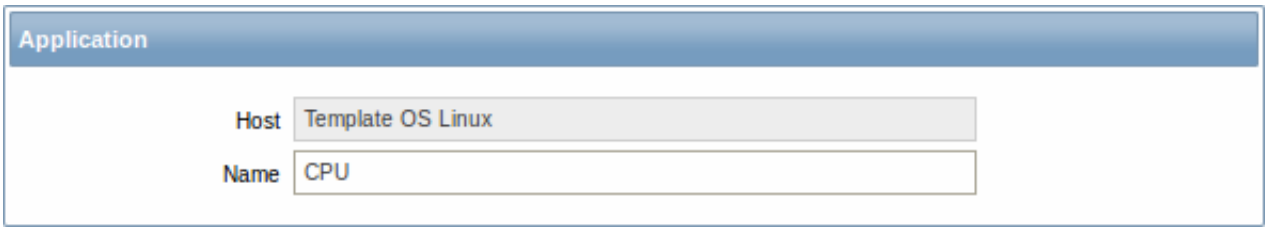
If you are using applications, then in *Monitoring → Latest data* you will see items and web scenarios grouped under their respective applications.

Configuration

To work with applications you must first create them and then link items or web scenarios to them.

To create an application, do the following:

- Go to *Configuration → Hosts* or *Templates*
- Click on *Applications* next to the required host or template
- Click on *Create application*
- Enter the application name and save it



You can also create a new application directly in the item properties form.

Items are linked to applications in the item properties form. Select one or more applications the item will belong to.

Web scenarios are linked to applications in the web scenario definition form. Select the application the scenario will belong to.

2.0/manual/config/items/applications.txt · Last modified: 2012/10/30 11:17 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

9 Queue

Overview

The queue displays items that are waiting for a refresh. The queue is just a **logical** representation of data from the database. There is no IPC queue or any other queue mechanism in Zabbix.

Statistics shown by the queue is a good indicator of the performance of Zabbix server.

Reading the queue

To read the queue, go to *Administration* → *Queue*. *Overview* should be selected in the dropdown to the right.

Items	5 seconds	10 seconds	30 seconds	1 minute	5 minutes	More than 10 minutes
Zabbix agent	0	1	0	0	1	0
Zabbix agent (active)	0	0	0	0	0	0
Simple check	0	0	0	0	0	0
SNMPv1 agent	0	0	0	0	0	0
SNMPv2 agent	0	0	0	0	0	0
SNMPv3 agent	0	0	0	0	0	0
Zabbix internal	0	0	0	0	0	0
Zabbix aggregate	0	0	0	0	0	0
External check	0	0	0	0	0	0
Database monitor	0	0	0	0	0	0
IPMI agent	0	0	0	0	0	0
SSH agent	0	0	0	0	0	0
TELNET agent	0	0	0	0	0	0
JMX agent	0	0	0	0	0	0
Calculated	0	0	0	0	0	0

The picture here is generally “green” so we may assume that the server is doing fine.

The queue shows one item waiting for 10 seconds and one for 5 minutes. Nice, it would be great to know what items these are.

To do just that, select *Details* in the dropdown in the upper right corner. Now you can see a list of those delayed items.

Next check	Delayed by	Host	Name
20 Jan 2012 10:44:07	7m 44s	Zabbix server	Version of zabbix_agent(d) running
20 Jan 2012 10:51:37	14s	Zabbix server	Ping to the server (TCP)

With these details provided it may be possible to find out why these items might be delayed.

With one or two delayed items there perhaps is no cause for alarm. They might get updated in a second. However, if you see a bunch of items getting delayed for too long, there might be a more serious problem.

Items	5 seconds	10 seconds	30 seconds	1 minute	5 minutes	More than 10 minutes
Zabbix agent	0	1	0	1	0	45
Zabbix agent (active)	0	0	0	0	0	0

Is the agent down?

Delay for remote node items

Queue information from a child node is not up-to-date. The master node receives historical data with a certain delay (normally, up-to 10 seconds for inter-node data transfer), so the information is delayed.

The information from a child node also depends on:

- performance of the child node
- communications between master and child nodes
- possible local time difference between master and child nodes

Queue item

A special internal item **zabbix[queue,<from>,<to>]** can be used to monitor the health of the queue in Zabbix. It will return the number of items delayed by the set amount of time. For more information see Internal items.

2.0/manual/config/items/queue.txt · Last modified: 2012/09/27 13:04 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 Triggers

Overview

Triggers are logical expressions that “evaluate” data gathered by items and represent the current system state.

While items are used to gather system data, it is highly impractical to follow these data all the time waiting for a condition that is alarming or deserves attention. The job of “evaluating” data can be left to trigger expressions.

Trigger expressions allow to define a threshold of what state of data is “acceptable”. Therefore, should the incoming data surpass the acceptable state, a trigger is “fired” - or changes status to PROBLEM.

A trigger may have the following status:

VALUE	DESCRIPTION
OK	This is a normal trigger state. Called FALSE in older Zabbix versions.
PROBLEM	Normally means that something happened. For example, the processor load is too high. Called TRUE in older Zabbix versions.

Trigger status (the expression) is recalculated every time Zabbix server receives a new value that is part of the expression.

If time-based functions (**nodata()**, **date()**, **dayofmonth()**, **dayofweek()**, **time()**, **now()**) are used in the expression, the trigger is recalculated every 30 seconds by a zabbix *timer* process.

You can build trigger expressions with different degrees of complexity.

2.0/manual/config/triggers.txt · Last modified: 2012/10/25 20:12 by zalex_ua
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Configuring a trigger

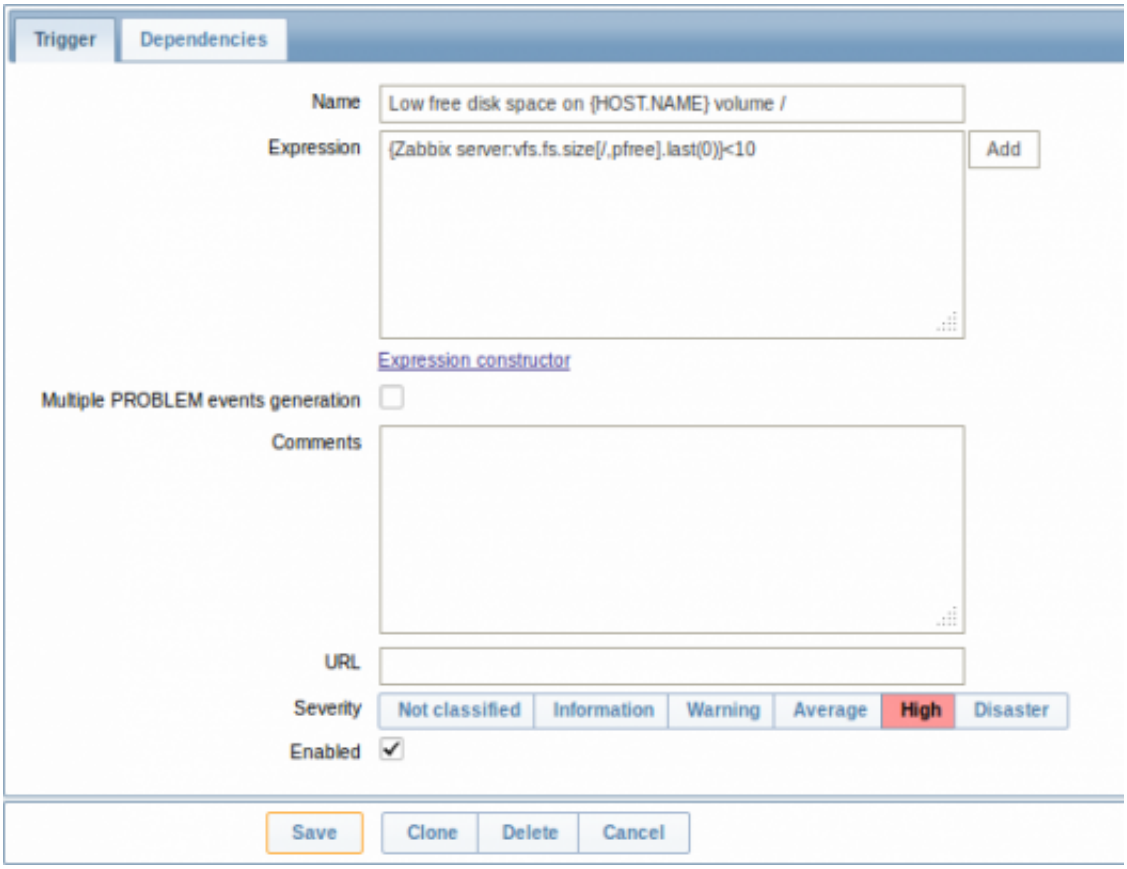
Overview

To configure a trigger, do the following:

- Go to: *Configuration* → *Hosts*
- Click on *Triggers* in the row of the host
- Click on *Create trigger* to the right (or on the trigger name to edit an existing trigger)
- Enter parameters of the trigger in the form

Configuration

The **Trigger** tab contains all the essential trigger attributes.



Parameter	Description
Name	Trigger name. The name may contain macros. **$1, $2**...**$9** macros can be used to refer to the first, second...ninth constant of the expression. *Note*: $1-$9 macros will resolve correctly if referring to constants in relatively simple, straightforward expressions. For example, the name "Processor load above $1 on {HOST.NAME}" will automatically change to "Processor load above 5 on New host" if the expression is {New host:system.cpu.load[percpu,avg1].last(0)}>5
Expression	Logical expression used for calculating the trigger state.
Multiple PROBLEM	

events generation	By checking this option you can set that an event is generated upon *every* 'Problem' evaluation of the trigger.
Comments	Text field used to provide more information about this trigger. May contain instructions for fixing specific problem, contact detail of responsible staff, etc.
URL	If not empty, the URL entered here is available as a link when clicking on the trigger name in *Monitoring → Triggers*. One macro may be used in the trigger URL field - {TRIGGER.ID}.
Severity	Set the required trigger severity by clicking the buttons.
Enabled	Unchecking this box will disable the trigger if required.

The **Dependencies** tab contains all the dependencies of the trigger.

Click on *Add* to add a new dependency.

You can also configure a trigger by opening an existing one, pressing the *Clone* button and then saving under a different name.

2.0/manual/config/triggers/trigger.txt · Last modified: 2013/08/06 11:26 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Trigger expression

Overview

The expressions used in triggers are very flexible. You can use them to create complex logical tests regarding monitored statistics.

A simple useful expression might look like:

```
{<server>:<key>.<function>(<parameter>)}<operator><constant>
```

1 Functions

Trigger functions allow to reference the collected values, current time and other factors.

A complete list of supported functions is available.

2 Function parameters

Most of numeric functions accept the number of seconds as a parameter.

You may use the prefix # to specify that a parameter has a different meaning:

FUNCTION CALL	MEANING
sum(600)	Sum of all values within 600 seconds
sum(#5)	Sum of the last 5 values

The function **last** uses a different meaning for values when prefixed with the hash mark - it makes it choose the n-th previous value, so given the values 3, 7, 2, 6, 5 (from most recent to least recent), **last(#2)** would return *7* and **last(#5)** would return *5*.

A parameter must be given even for those functions which ignore it. Example: last(0)

avg, **count**, **last**, **min** and **max** functions support an additional, second `time_shift` parameter. This parameter allows to reference data from a period of time in the past. For example, **avg(1h,1d)** will return the average value for an hour one day ago.

Triggers only evaluate history information. If history is not available (especially relevant for time shift), trend information is not used, thus history must be kept for at least the period trigger functions expect it.

You can use the supported unit symbols in trigger expressions, for example '5m' (minutes) instead of '300' seconds or '1d' (day) instead of '86400' seconds. '1K' will stand for '1024' bytes.

3 Operators

The following operators are supported for triggers **(in descending priority of execution)**:

PRIORITY	OPERATOR	DEFINITION
1	/	Division
2	*	Multiplication
3	-	Arithmetical minus
4	+	Arithmetical plus
5	<	Less than. The operator is defined as: A<B ⇔ (A<=B-0.000001)
6	>	More than. The operator is defined as: A>B ⇔ (A>=B+0.000001)
7	#	Not equal. The operator is defined as: A#B ⇔ (A<=B-0.000001) | (A>=B+0.000001)
8	=	Is equal. The operator is defined as: A=B ⇔ (A>B-0.000001) & (A<B+0.000001)
9	**&**	Logical AND
10	|	Logical OR

4 Examples of triggers

Example 1

Processor load is too high on www.zabbix.com [http://www.zabbix.com]

```
{www.zabbix.com:system.cpu.load[all,avg1].last(0)}>5
```

'www.zabbix.com:system.cpu.load[all,avg1]' gives a short name of the monitored parameter. It specifies that the server is 'www.zabbix.com' and the key being monitored is 'system.cpu.load[all,avg1]'. By using the function 'last()', we are referring to the most recent value. Finally, '>5' means that the trigger is in the PROBLEM state whenever the most recent processor load measurement from www.zabbix.com is greater than 5.

Example 2

www.zabbix.com [http://www.zabbix.com] is overloaded

```
{www.zabbix.com:system.cpu.load[all,avg1].last(0)}>5|{www.zabbix.com:system.cpu.load[all,avg1].min(10m)}>2
```

The expression is true when either the current processor load is more than 5 or the processor load was more than 2 during last 10 minutes.

Example 3

/etc/passwd has been changed

Use of function diff:

```
{www.zabbix.com:vfs.file.cksum[/etc/passwd].diff(0)}>0
```

The expression is true when the previous value of checksum of /etc/passwd differs from the most recent one.

Similar expressions could be useful to monitor changes in important files, such as /etc/passwd, /etc/inetd.conf, /kernel, etc.

Example 4

Someone is downloading a large file from the Internet

Use of function min:

```
{www.zabbix.com:net.if.in[eth0,bytes].min(5m)}>100K
```

The expression is true when number of received bytes on eth0 is more than 100 KB within last 5 minutes.

Example 5

Both nodes of clustered SMTP server are down

Note use of two different hosts in one expression:

```
{smtp1.zabbix.com:net.tcp.service[smtp].last(0)}=0&{smtp2.zabbix.com:net.tcp.service[smtp].last(0)}=0
```

The expression is true when both SMTP servers are down on both smtp1.zabbix.com and smtp2.zabbix.com.

Example 6

Zabbix agent needs to be upgraded

Use of function str():

```
{zabbix.zabbix.com:agent.version.str("beta8")}=1
```

The expression is true if Zabbix agent has version beta8 (presumably 1.0beta8).

Example 7

Server is unreachable

```
{zabbix.zabbix.com:icmpping.count(30m,0)}>5
```

The expression is true if host “zabbix.zabbix.com” is unreachable more than 5 times in the last 30 minutes.

Example 8

No heartbeats within last 3 minutes

Use of function nodata():

```
{zabbix.zabbix.com:tick.nodata(3m)}=1
```

'tick' must have type 'Zabbix trapper'. In order to make this trigger work, item 'tick' must be defined. The host should periodically send data for this parameter using zabbix_sender. If no data is received within 180 seconds, the trigger value becomes PROBLEM.

Example 9

CPU activity at night time

Use of function time():

```
{zabbix:system.cpu.load[all,avg1].min(5m)}>2&{zabbix:system.cpu.load[all,avg1].time(0)}>000000&{zabbix:system.cpu.load[all,avg1].time(0)}<060000
```

The trigger may change its status to true, only at night (00:00-06:00) time.

Example 10

Check if client local time is in sync with Zabbix server time

Use of function fuzzytime():

```
{MySQL_DB:system.localtime.fuzzytime(10)}=0
```

The trigger will change to the problem state in case when local time on server MySQL_DB and Zabbix server differs by more than 10 seconds.

Example 11

Comparing average load today with average load of the same time yesterday (using a second `time_shift` parameter).

```
{server:system.cpu.load.avg(1h)}/{server:system.cpu.load.avg(1h,1d)}>2
```

This expression will fire if the average load for the last hour tops the average load of the same hour yesterday more than two times.

5 Hysteresis

Sometimes a trigger must have different conditions for different states. For example, we would like to define a trigger which would become PROBLEM when server room temperature is higher than 20C while it should stay in the state until temperature will not become lower than 15C.

In order to do this, we define the following trigger:

Example 1

Temperature in server room is too high

```
({TRIGGER.VALUE}=0&{server:temp.last(0)}>20)|
({TRIGGER.VALUE}=1&{server:temp.last(0)}>15)
```

Note the use of a macro {TRIGGER.VALUE}. The macro returns current trigger value.

Example 2

Free disk space is too low

Problem: it is less than 10GB for last 5 minutes

Recovery: it is more than 40GB for last 10 minutes

```
({TRIGGER.VALUE}=0&{server:vfs.fs.size[/,free].max(5m)}<10G) |
({TRIGGER.VALUE}=1&{server:vfs.fs.size[/,free].min(10m)}<40G)
```

Note use of macro {TRIGGER.VALUE}. The macro returns current trigger value.

2.0/manual/config/triggers/expression.txt · Last modified: 2013/06/19 00:22 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 Trigger dependencies

Overview

Sometimes the availability of one host depends on another. A server that is behind some router will become unreachable if the router goes down. With triggers configured for both, you might get notifications about two hosts down - while only the router was the guilty party.

This is where some dependency between hosts might be useful. With dependency set notifications of the dependants could be withheld and only the notification for the root problem sent.

While Zabbix does not support dependencies between hosts directly, they may be defined with another, more flexible method - trigger dependencies. A trigger may have one or more triggers it depends on.

So in our simple example we open the server trigger configuration form and set that it depends on the respective trigger of the router. With such dependency the server trigger will not change state as long as the trigger it depends on is in 'Problem' state - and thus no dependant actions will be taken and no notifications sent.

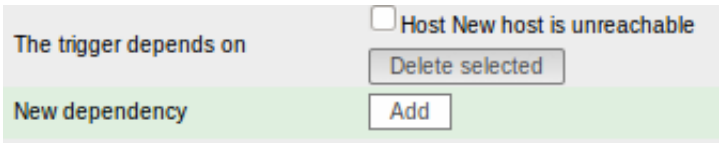
If both the server and the router are down and dependency is there, Zabbix will not execute actions for the dependant trigger.

Also:

- Trigger dependency may be added from any host trigger to any other host trigger, as long as it wouldn't result in a circular dependency.
- Trigger dependency may be added from a template to a template. If a trigger from template A depends on a trigger from template B, template A may only be linked to a host (or another template) together with template B, but template B may be linked to a host (or another template) alone.
- Trigger dependency may be added from template trigger to a host trigger. In this case, linking such a template to a host will create a host trigger that depends on the same trigger template trigger was depending on. This allows to, for example, have a template where some triggers depend on router (host) triggers. All hosts linked to this template will depend on that specific router.
- Trigger dependency from a host trigger to a template trigger may not be added.

Configuration

To define a dependency, open the trigger configuration form. Click on *Add* next to 'New dependency' and select one or more triggers that our trigger will depend on.



Click *Save*. Now the trigger has an indication of its dependency in the list.

Example of several dependencies

For example, a Host is behind a Router2 and the Router2 is behind a Router1.

```
Zabbix - Router1 - Router2 - Host
```

If Router1 is down, then obviously Host and Router2 are also unreachable yet we don't want to receive three notifications about Host, Router1 and Router2 all being down.

So in this case we define two dependencies:

```
'Host is down' trigger depends on 'Router2 is down' trigger
'Router2 is down' trigger depends on 'Router1 is down' trigger
```

Before changing the status of the 'Host is down' trigger, Zabbix will check for corresponding trigger dependencies. If found, and one of those triggers is in 'Problem' state, then the trigger status will not be changed and thus actions will not be executed and notifications will not be sent.

Zabbix performs this check recursively. If Router1 or Router2 is unreachable, the Host trigger won't be updated.

2.0/manual/config/triggers/dependencies.txt · Last modified: 2012/01/19 16:28 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

4 Trigger severity

Trigger severity defines how important a trigger is. Zabbix supports the following trigger severities:

SEVERITY	DEFINITION	COLOUR
Not classified	Unknown severity.	Grey
Information	For information purposes.	Light green
Warning	Be warned.	Yellow
Average	Average problem.	Orange
High	Something important has happened.	Red
Disaster	Disaster. Financial losses, etc.	Bright red

The severities are used for:

- visual representation of triggers. Different colours for different severities.
- audio in global alarms. Different audio for different severities.
- user media. Different media (notification channel) for different severities. For example, SMS - high severity, email - other.
- limiting actions by conditions against trigger severities

It is possible to customise trigger severity names and colours.

2.0/manual/config/triggers/severity.txt · Last modified: 2012/05/02 09:39 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

5 Customising trigger severities

Trigger severity names and colours for severity related GUI elements can be configured in *Administration → General → Trigger severities*. Colours are shared among all GUI themes.

Translating customised severity names

If Zabbix frontend translations are used, custom severity names will override translated names by default.

Default trigger severity names are available for translation in all locales. If a severity name is changed, custom name is used in all locales and additional manual translation is needed.

Custom severity name translation procedure:

1. set required custom severity name, for example 'Important'
2. edit <fronent_dir>/locale/<required_locale>/LC_MESSAGES/frontend.po
3. add 2 lines:

```
msgid "Important"
msgstr "<translation string>"
```

   and save file

4. create .mo files as described in <fronent_dir>/locale/README

Here **msgid** should match the new custom severity name and **msgstr** should be the translation for it in the specific language.

This procedure should be performed after each severity name change.

2.0/manual/config/triggers/customseverities.txt · Last modified: 2012/01/19 12:42 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

6 Unit symbols

Overview

Having to use some large numbers, for example '86400' to represent the number of seconds in one day, is both difficult and error-prone. This is why you can use some appropriate unit symbols (or suffixes) to simplify Zabbix trigger expressions or item keys.

Instead of '86400' you can simply enter '1d'. Suffixes function as multipliers.

Time unit suffixes

For time you can use:

- **s** - seconds (when used, works the same as the raw value)
- **m** - minutes
- **h** - hours
- **d** - days
- **w** - weeks

Time unit suffixes are supported in:

1. trigger expressions (constants and function parameters)
2. parameters of the **zabbix[queue,<from>,<to>]** internal item
3. last parameter of aggregate checks

Prefix symbols

In both Zabbix server and frontend these prefix symbols are supported for both display and usage in trigger expressions (constants and function parameters):

- **K** - kilo
- **M** - mega
- **G** - giga
- **T** - tera

When item values other than in B, Bps are displayed in the frontend, a base of 10 is used (1K = 1000). Apart from that, base 2 is applied (1K = 1024) everywhere.

Additionally the frontend also supports the display of:

- **P** - peta
- **E** - exa
- **Z** - zetta
- **Y** - yotta

Usage examples

By using some appropriate suffixes you can write trigger expressions that are easier to understand and maintain, for example these expressions:

```
{host:zabbix[proxy,zabbix_proxy,lastaccess]}>120
{host:system.uptime[].last(0)}<86400
{host:system.cpu.load.avg(600)}<10
```

could be changed to:

```
{host:zabbix[proxy,zabbix_proxy,lastaccess]}>2m
{host:system.uptime.last(0)}<1d
{host:system.cpu.load.avg(10m)}<10
```

2.0/manual/config/triggers/suffixes.txt · Last modified: 2012/07/06 14:46 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

4 Events

Overview

Events in Zabbix are generated by three sources:

- triggers - whenever a trigger changes its status
- discovery - upon detection of hosts or services
- auto registration - when active agents are auto-registered by server

Events are time-stamped and can be the basis of actions such as sending notification e-mail etc.

To view details of events in the frontend, go to *Monitoring* | *Events*. There you can click on the event date and time to view details of an event.

More information is available on each event source.

2.0/manual/config/events.txt · Last modified: 2011/10/10 15:13 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Event sources

1.1 Trigger events

Change of trigger status is the most frequent and most important source of events.

Each time the trigger changes its state, an event is generated. The event contains details of the trigger state's change - when did it happen and what the new state is.

1.2 Discovery events

Zabbix periodically scans the IP ranges defined in network discovery rules. Frequency of the check is configurable for each rule individually. Once a host or a service is discovered, a discovery event (or several events) are generated.

Zabbix generates the following events:

Event	When generated
Service Up	Every time Zabbix detects active service.
Service Down	Every time Zabbix cannot detect service.
Host Up	If at least one of the services is UP for the IP.
Host Down	If all services are not responding.
Service Discovered	If the service is back after downtime or discovered for the first time.
Service Lost	If the service is lost after being up.
Host Discovered	If host is back after downtime or discovered for the first time.
Host Lost	If host is lost after being up.

1.3 Active agent auto-discovery events

Active agent auto-registration creates events in Zabbix.

If configured, active agent auto-registration can happen when a previously unknown active agent asks for checks. The server adds a new auto-registered host, using the received IP address and port of the agent.

For more information, see the active agent auto-registration page.

2.0/manual/config/events/sources.txt · Last modified: 2011/12/08 15:20 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

5 Visualisation

2.0/manual/config/visualisation.txt · Last modified: 2011/11/01 13:50 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Graphs

Overview

With lots of data flowing into Zabbix, it becomes much easier for the users if they can look at a visual representation of what is going on rather than only numbers.

This is where graphs come in. Graphs allow to grasp the data flow at a glance, correlate problems, discover when something started or make a presentation of when something might turn into a problem.

Zabbix provides users with built-in simple graphs as well as with the possibility to create more complex customised graphs.

2.0/manual/config/visualisation/graphs.txt · Last modified: 2011/10/25 16:18 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

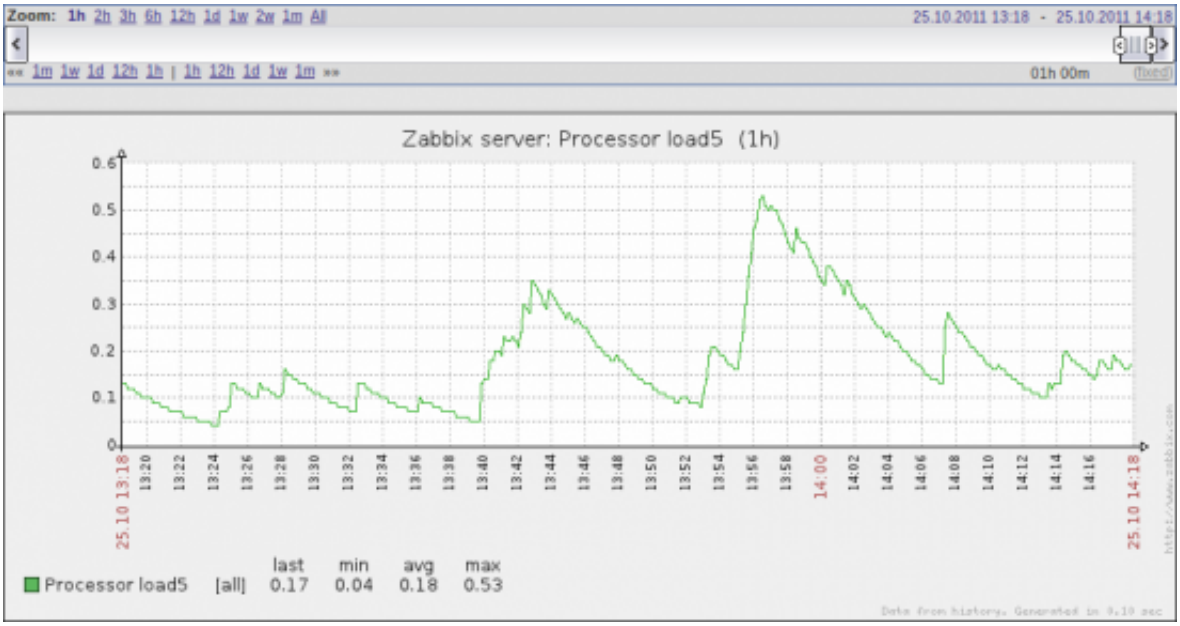
1 Simple graphs

Overview

Simple graphs are provided for the visualization of data gathered by items.

No configuration effort is required on the user part to view simple graphs. They are freely made available by Zabbix.

Just go to *Monitoring* → *Latest data* and click on the Graph link for the respective item and a graph will be displayed.



Time period selector

Take note of the time period selector above the graph. It allows you to select the desired time period easily.

The slider within the selector can be dragged back and forth, as well as resized, effectively changing the time period displayed. Links on the left hand side allow to choose some often-used predefined periods (above the slider area) and move them back and forth in time (below the slider area). The dates on the right hand side actually work as links, popping up a calendar and allowing to set a specific start/end time.

The **fixed/dynamic** link in the lower right hand corner has the following effects:

- controls whether the time period is kept constant when you change the start/end time in the calendar popup.
- when *fixed*, time moving controls (« 6m 1m 7d 1d 12h 1h | 1h 12h 1d 7d 1m 6m ») will move the slider, while not changing its size, whereas when *dynamic*, the control used will enlarge the slider in the respective direction.
- when *fixed*, pressing the larger < and > buttons will move the slider, while not changing its size, whereas when *dynamic*, < and > will enlarge the slider in the respective direction. The slider will move by the amount of its size, so, for example, if it is one month, it will move by a month; whereas the

slider will enlarge by 1 day.

Another way of controlling the displayed time is to highlight an area in the graph with the left mouse button. The graph will zoom into the highlighted area once you release the left mouse button.

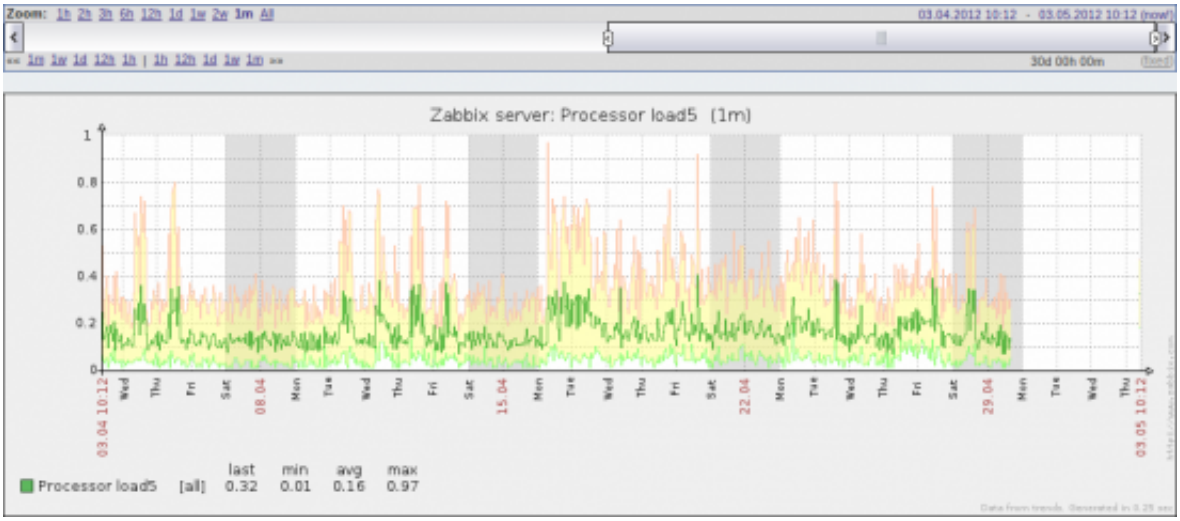
Simple graphs are provided for all numeric items. For textual items, a link to History is available in *Monitoring → Latest data*.

Recent data vs longer periods

For very recent data a **single** line is drawn connecting each received value. The single line is drawn as long as there is at least one horizontal pixel available for one value.

For data that show a longer period **three lines** are drawn - a dark green one shows the average, while a light pink and a light green line shows the maximum and minimum values at that point in time. The space between the highs and the lows is filled with yellow background.

Working time (working days) is displayed in graphs as a white background, while non-working time is displayed in grey (with the *Original blue* default frontend theme).



Working time is always displayed in simple graphs, whereas displaying it in custom graphs is a user preference.

Working time is not displayed if the graph shows more than 3 months.

Generating from history/trends

Graphs can be drawn based on either item history or trends. A grey caption at the bottom right of a graph indicates where the data come from.

Two factors influence whether history of trends is used:

- longevity of item history. For example, item history can be kept for 14 days. In that case, any data older than the fourteen days will be coming from trends.
- data congestion in the graph. If the amount of seconds to display in a horizontal graph pixel exceeds 3600/16, trend data are displayed (even if item history is still available for the same period).

Switching to raw values

A dropdown on the upper right above the graph allows to switch from the simple graph to the *Values/500 latest values* listings. This can be useful for viewing the numeric values making up the graph.

The values represented here are raw, i.e. no units or postprocessing of values is used. Value mapping, however, is applied.

2.0/manual/config/visualisation/graphs/simple.txt · Last modified: 2013/06/07 11:46 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Custom graphs

Overview

Custom graphs, as the name suggests, offer customisation capabilities.

While simple graphs are good for viewing data of a single item, they do not offer configuration capabilities.

Thus, if you want to change graph style or the way lines are displayed or compare several items, for example incoming and outgoing traffic in a single graph, you need a custom graph.

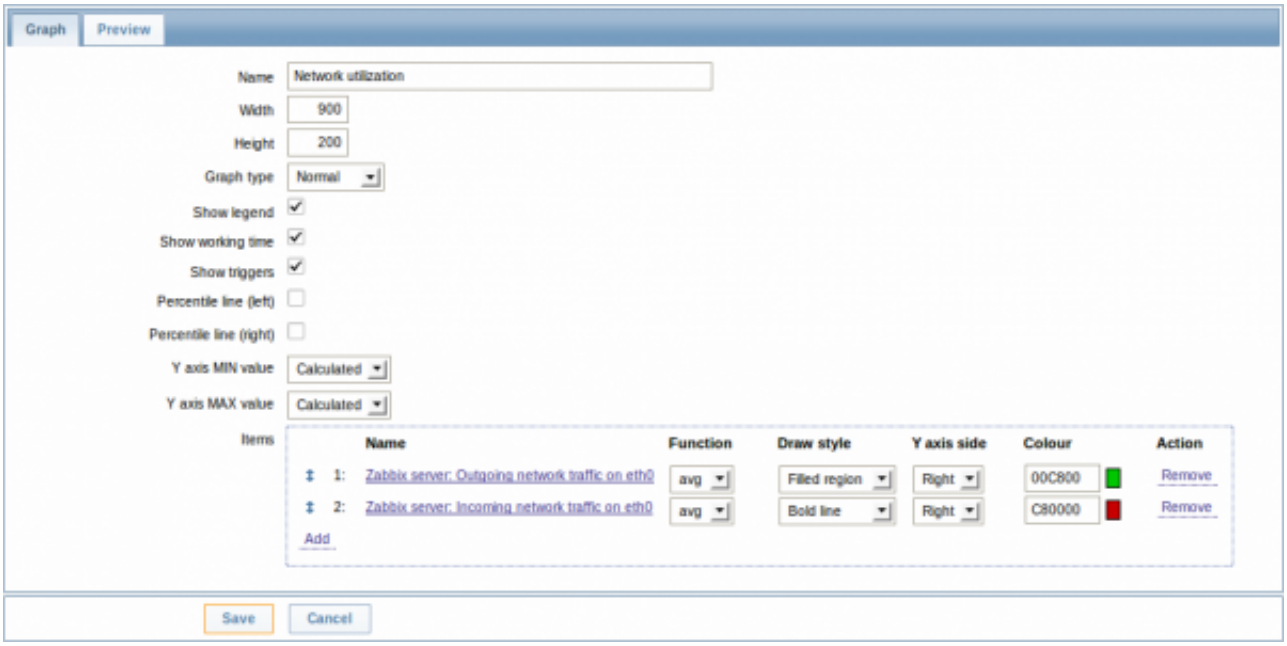
Custom graphs are configured manually.

They can be created for a host or several hosts or for a single template.

Configuring custom graphs

To create a custom graph, do the following:

- Go to *Configuration → Hosts (or Templates)*
- Click on *Graphs* in the row next to the desired host or template
- In the Graphs screen click on *Create graph*
- Edit graph attributes



Graph attributes:

Parameter	Description
Name	Unique graph name.
Width	Graph width in pixels (for preview and pie/exploded graphs only).
Height	Graph height in pixels.

Graph type	Graph type: **Normal** - normal graph, values displayed as lines. **Stacked** - stacked graph. **Pie** - pie graphs. **Exploded** - exploded pie graph.
Show legend	Checking this box will set to display the graph legend.
Show working time	If selected, non-working hours will be shown with gray background. Not available for pie and exploded pie graphs.
Show triggers	If selected, simple triggers will be displayed as red lines. Not available for pie and exploded pie graphs.
Percentile line (left)	Display percentile for left Y axis. If, for example, 95% percentile is set, then the line will be at the level where 95 per cent of the values fall under. Only available for normal graphs.
Percentile line (right)	Display percentile for right Y axis. If, for example, 95% percentile is set, then the line will be at the level where 95 per cent of the values fall under. Only available for normal graphs.
Y axis MIN value	Mimimum value of Y axis: **Calculated** - Y axis minimum value will be automatically calculated **Fixed** - fixed minimum value for Y axis. Not available for pie and exploded pie graphs. **Item** - last value of the selected item will be the minimum value
Y axis MAX value	Maximum value of Y axis: **Calculated** - Y axis maximum value will be automatically calculated **Fixed** - fixed maximum value for Y axis. Not available for pie and exploded pie graphs. **Item** - last value of the selected item will be the maximum value
3D view	Enable 3D style. For pie and exploded pie graphs only.
Items	Items, data of which are to be displayed in this graph.

Configuring graph items

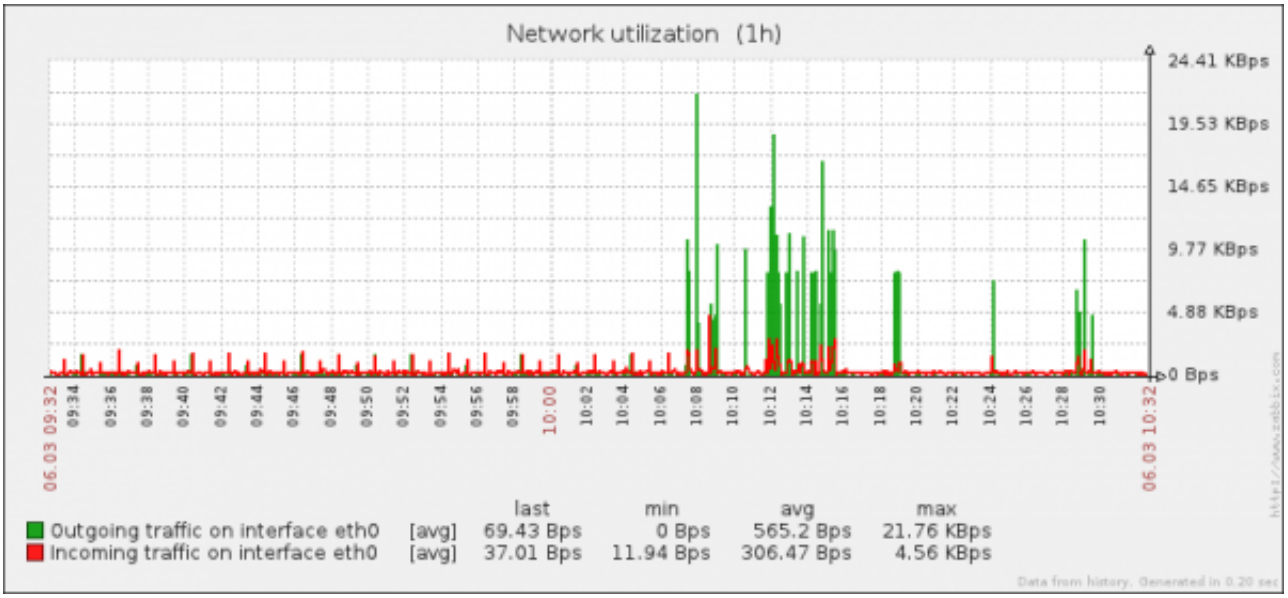
To add items, data of which are to be displayed in the graph, click on *Add* in the *Items* block, select items and then set attributes for the way item data will be displayed.

Item display attributes:

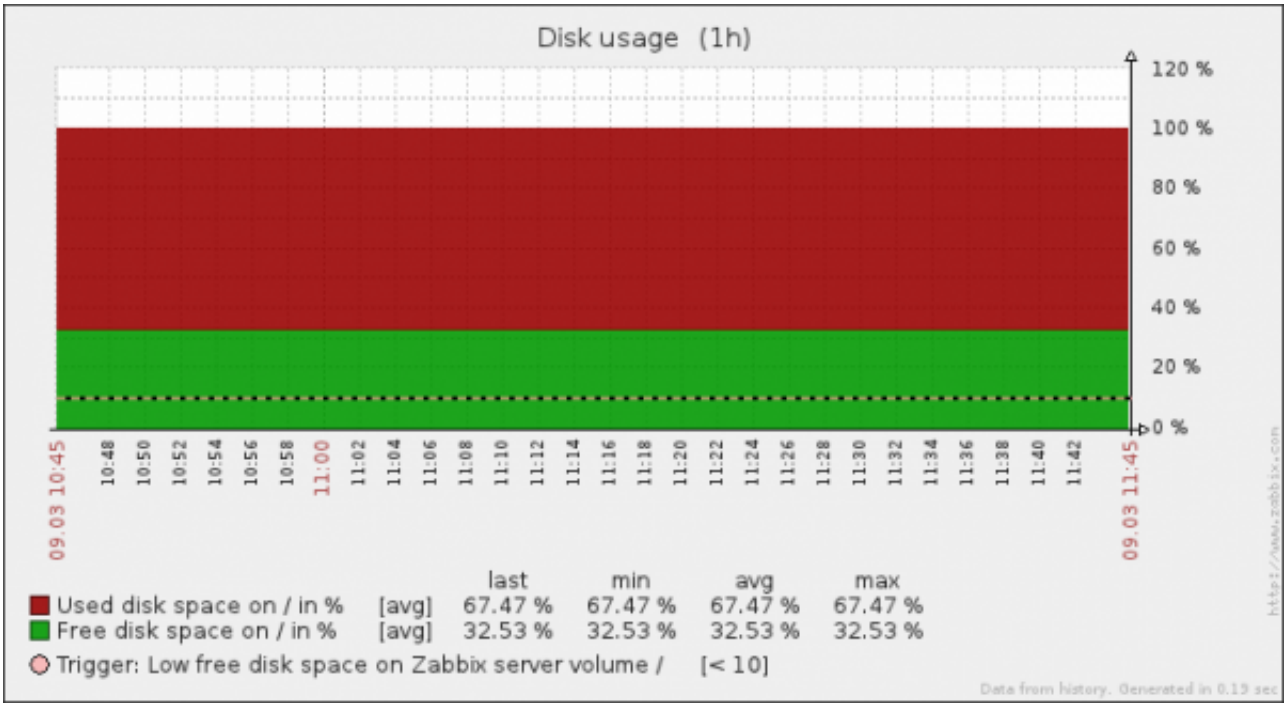
Parameter	Description
Sort order (0→100)	Draw order. 0 will be processed first. Can be used to draw lines or regions behind (or in front of) another. You can drag and drop items by the arrow in the beginning of line to set the sort order or which item is displayed in front of the other.
Name	Name of item, data of which will be displayed.
Type	Type (only available for pie and exploded pie graphs): **Simple** **Graph sum**
Function	What values will be displayed when more than one value exists for an item: **all** - all (minimum, average and maximum) **min** - minimum only **avg** - average only **max** - maximum only
Draw style	Draw style (only available for normal graphs; for stacked graphs filled region is always used): **Line** - draw lines **Filled region** - draw filled region **Bold line** - draw bold lines **Dot** - draw dots **Dashed line** - draw dashed line
Y axis side	Which Y axis side the element is assigned to.
Colour	RGB colour in HEX notation.

Graph preview

In the *Preview* tab, a preview of the graph is displayed so you can immediately see what you are creating.



Note that the preview will not show any data for template items.



In this example, pay attention to the dashed bold line displaying the trigger level and the trigger information displayed in the legend.

3 triggers is the hard-coded limit for the number of triggers displayed in the legend.

If graph height is set as less than 120 pixels, no trigger will be displayed in the legend.

2.0/manual/config/visualisation/graphs/custom.txt · Last modified: 2013/06/11 09:34 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-

Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Network maps

Overview

If you have a network to look after, you may want to have an overview of your infrastructure somewhere. For that purpose you can create maps in Zabbix - of networks and of anything you like.

Proceed to configuring a network map.

2.0/manual/config/visualisation/maps.txt · Last modified: 2011/10/26 16:43 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Configuring a network map

Overview

Configuring a map in Zabbix requires that you first create a map by defining its general parameters and then you start filling the actual map with elements and their links.

You can populate the map with elements that are a host, a host group, a trigger, an image or another map.

Icons are used to represent map elements. You can define the information that will be displayed with the icons and set that recent problems are displayed in a special way. You can link the icons and define information to be displayed on the links.

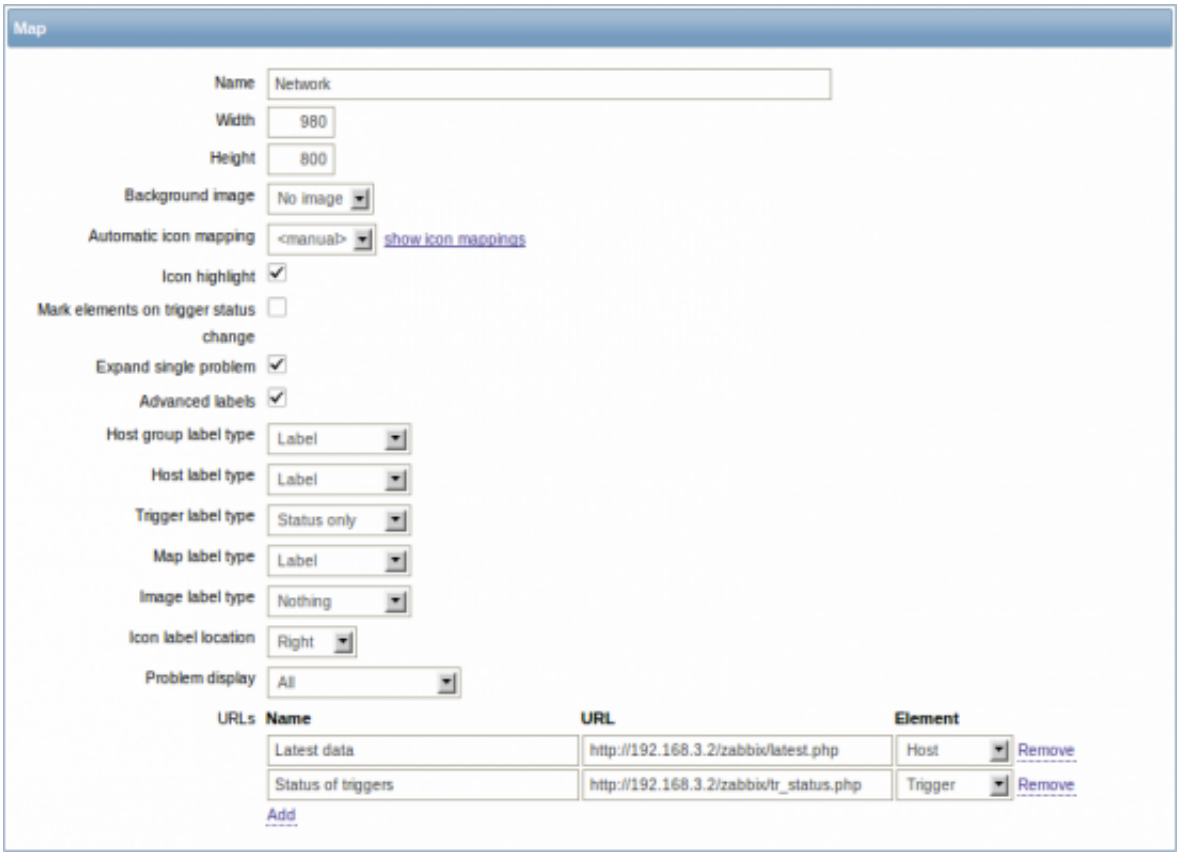
Maps that are ready can be viewed in *Monitoring → Maps*. In the monitoring view you can click on the icons and take advantage of the links to some scripts and URLs.

You can add custom URLs to be accessible by clicking on the icons. Thus you may link a host icon to host properties or a map icon to another map.

Creating a map

To create a map, do the following:

- Go to *Configuration → Maps*
- Click on *Create map*
- Edit general map attributes



General map attributes:

Parameter	Description
Name	Unique map name.
Width	Map width in pixels.
Height	Map height in pixels.
Background image	Use background image: **No image** – no background image (white background) **Image** – selected image to be used as a background image. No scaling is performed. You may use a geographical map or any other image to enhance your map.
Automatic icon mapping	You can set to use an automatic icon mapping, configured in *Administration → General → Icon mapping*. Icon mapping allows to map certain icons against certain host inventory fields.
Icon highlighting	If you check this box, icons will receive highlighting. If the element has an active trigger, it will receive a round background, having the same colour as the highest severity trigger. If element status is “disabled” or “in maintenance”, square background will be used.
Mark elements on trigger status change	A recent change of trigger status (recent problem or resolution) will be highlighted with markers (inward-pointing red triangles) on the three sides of the element icon that are free of the label. Markers are displayed for 30 minutes.
Expand single problem	If a map element (host, host group or another map) has one single problem, this option controls whether the problem (trigger) name is displayed, or problem count. If marked, problem name is used.
Advanced labels	If you check this box you will be able to define separate label types for separate element types.
Icon label type	Label type used for icons: **Label** – icon label **IP address** – IP address **Element name** – element name (for example, host name) **Status only** – status only (OK or PROBLEM) **Nothing** – no labels are displayed

Icon label location	Label location in relation to the icon: **Bottom** - beneath the icon **Left** - to the left **Right** - to the right **Top** - above the icon
Problem display	Display problem count as: **All** - full problem count will be displayed **Separated** - unacknowledged problem count will be displayed separated as a number of the total problem count **Unacknowledged only** - only the unacknowledged problem count will be displayed
URLs	URLs for each element type can be defined (with a label). These will be displayed as links when a user clicks on the element in the monitoring section. Macros that can be used in map URLs: {MAP.ID}, {HOSTGROUP.ID}, {HOST.ID}, {TRIGGER.ID}

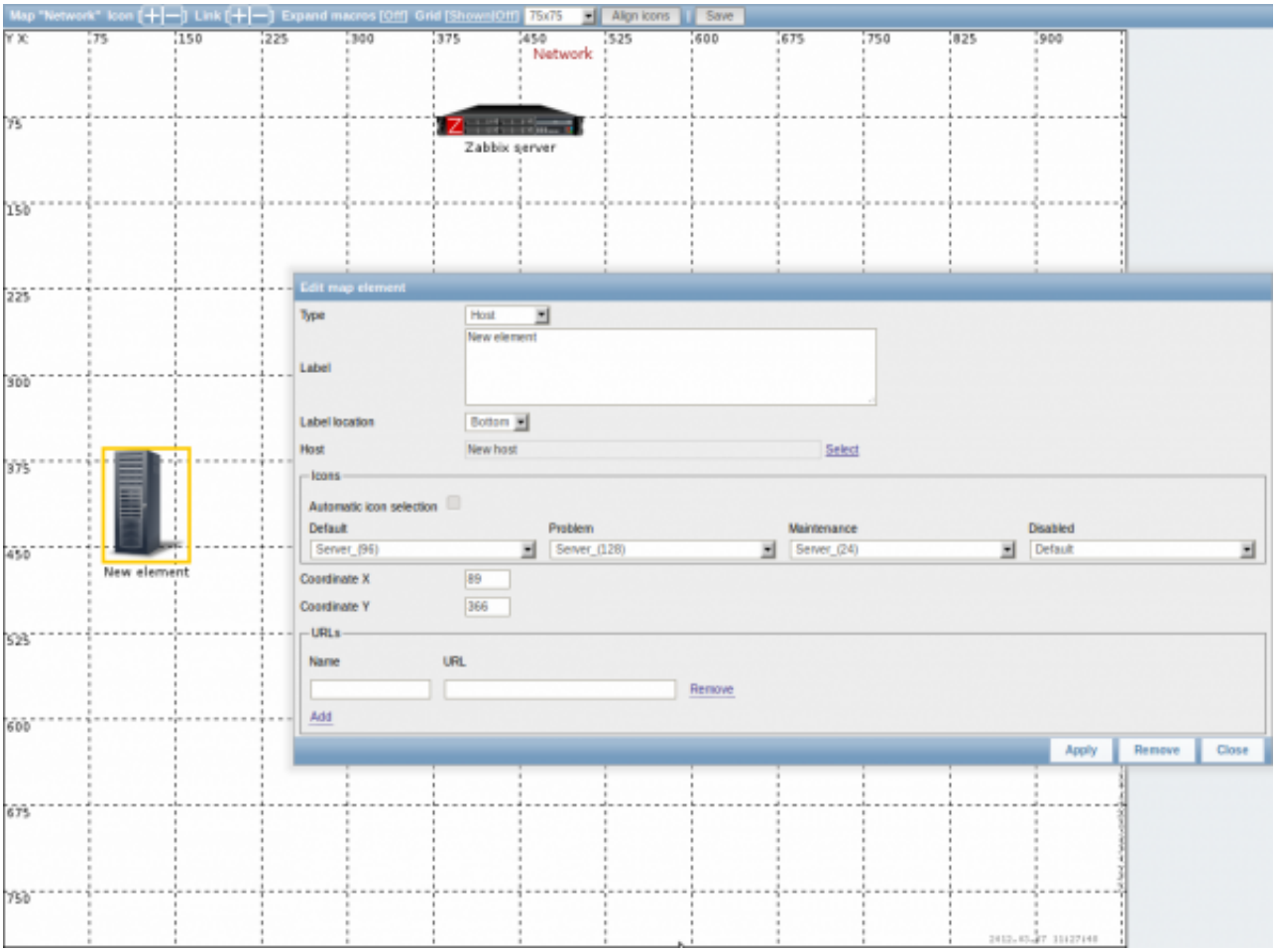
When you save this, you have created an empty map with a name, dimensions and certain preferences. Now you need to add some elements. For that, click on the map name in the list to open the editable area.

Adding elements

To add an element, click on the "+" next to Icon. The new element will appear at the top left corner of the map. Drag and drop it wherever you like.

Note that with the Grid option "On", elements will always align to the grid (you can pick various grid sizes from the dropdown, also hide/show the grid). If you want to put elements anywhere without alignment, turn the option to "Off". (Random elements can later again be aligned to the grid with the *Align icons* button.)

Now that you have some elements in place, you may want to start differentiating them by giving names etc. By clicking on the element, a form is displayed and you can set the element type, give a name, choose a different icon etc.



Map element attributes:

Parameter	Description
Type	Type of the element: **Host** - icon representing status of all triggers of the selected host **Map** - icon representing status of all elements of a map **Trigger** - icon representing status of a single trigger **Host group** - icon representing status of all triggers of all hosts belonging to the selected group **Image** - an icon, not linked to any resource
Label	Icon label, any string. Macros and multi-line strings can be used in labels.
Label location	Label location in relation to the icon: **Default** - map's default label location **Bottom** - beneath the icon **Left** - to the left **Right** - to the right **Top** - above the icon
Host	Select the host, if the element type is 'Host'.
Map	Select the map, if the element type is 'Map'.
Trigger	Select the trigger, if the element type is 'Trigger'.
Host group	Select the host group, if the element type is 'Host group'.
Icon (default)	Icon to be used.

Automatic icon selection	In this case an icon mapping will be used to determine which icon to display.
Icons	You can choose to display different icons for the element in these cases: default, problem, maintenance, disabled.
Coordinate X	X coordinate of the map element.
Coordinate Y	Y coordinate of the map element.
URLs	Element-specific URLs can be set for the element. These will be displayed as links when a user clicks on the element in the monitoring section. If the element has its own URLs and there are map level URLs for its type defined, they will be combined in the same menu. Macros that can be used in map URLs: {MAP.ID}, {HOSTGROUP.ID}, {HOST.ID}, {TRIGGER.ID}

Added elements are not automatically saved. If you navigate away from the page, all changes may be lost.

Therefore it is a good idea to click on the **Save** button in the top right corner. Once clicked, the changes are saved regardless of what you choose in the following popup.

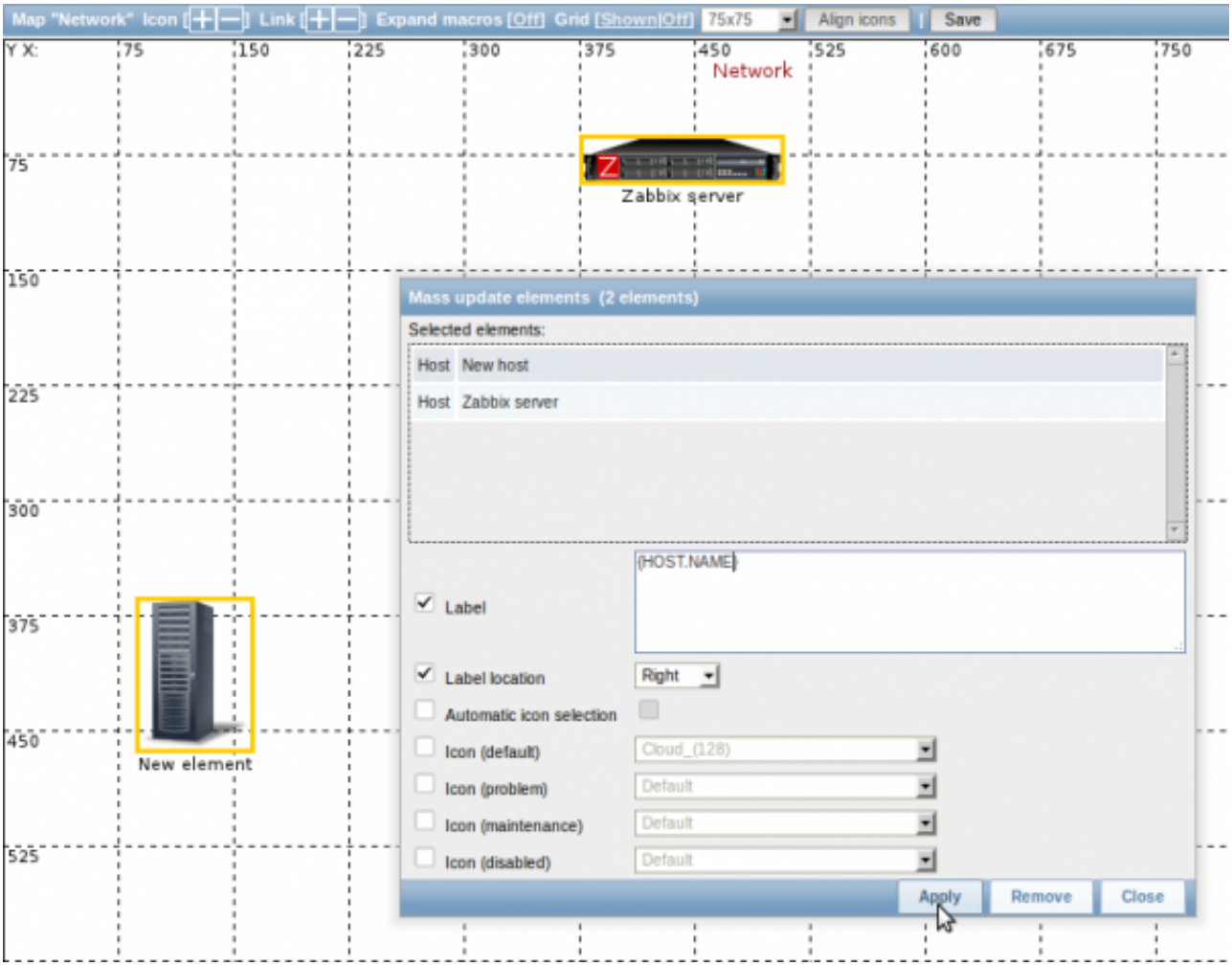
Selected grid options are also saved with each map.

Selecting elements

To select elements, select one and then hold down *Ctrl* (or *Shift*) to select the others.

You can also select multiple elements by dragging a rectangle in the editable area and selecting all elements in it (option available since Zabbix 2.0).

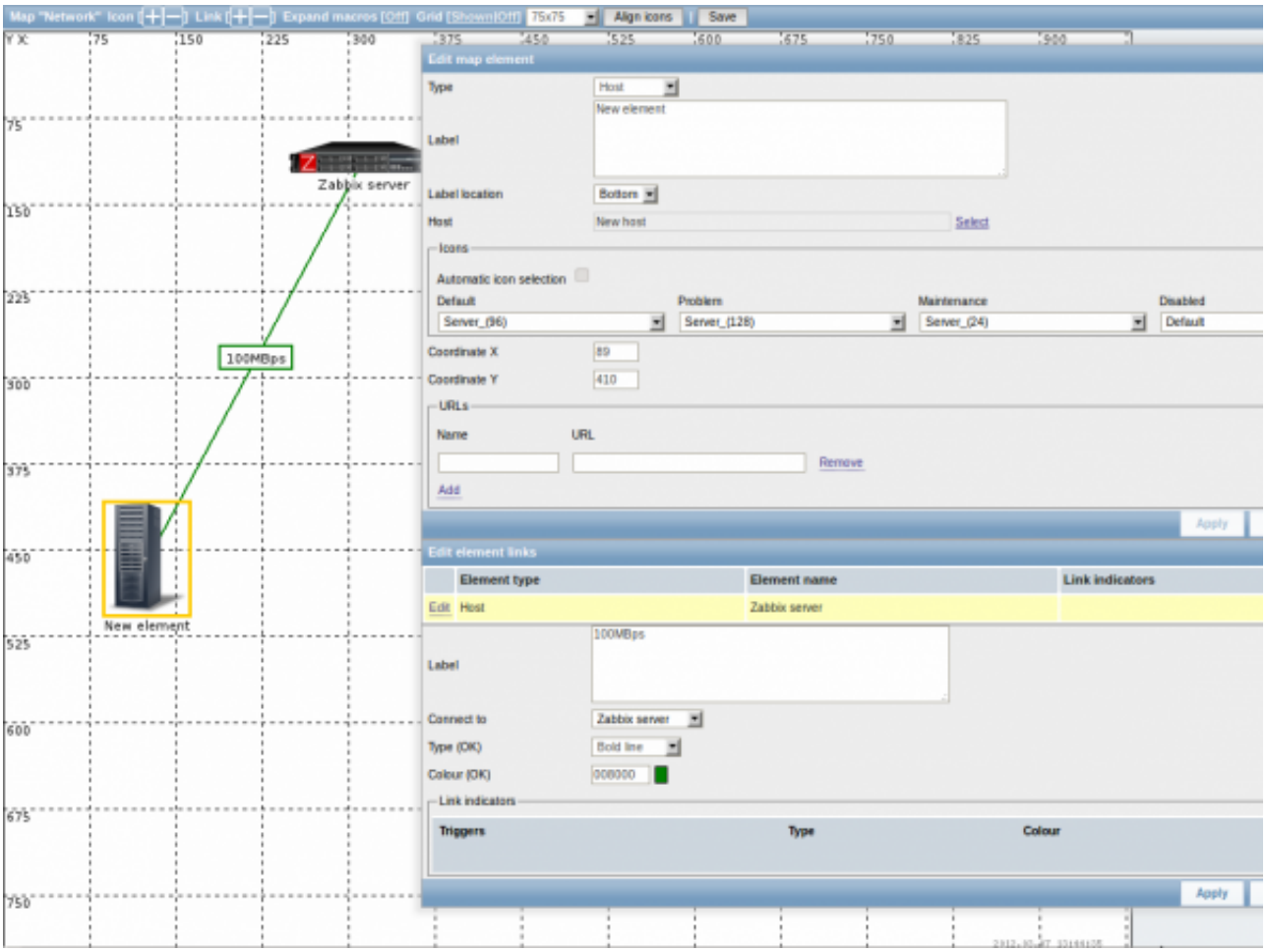
Once you select more than one element, the element property form shifts to the mass-update mode so you can change attributes of selected elements in one go. To do so, mark the attribute using the checkbox and enter a new value for it. You may use macros here (such as, say, {HOSTNAME} for the element label).



Linking elements

Once you have put some elements on the map, it is time to start linking them. To link two elements you must first select them. With the elements selected, click on the "+" next to Link.

With a link created, the single element form now contains an additional *Edit element links* section. Click on *Edit* before the link to edit its attributes.



Link attributes:

Parameter	Description
Label	Label that will be rendered on top of the link. You can use macros here.
Connect to	The element that the link connects to.
Type (OK)	Default link style: **Line** - single line **Bold line** - bold line **Dot** - dots **Dashed line** - dashed line
Colour (OK)	Default link colour.
Link indicators	List of triggers linked to the link. In case a trigger has status PROBLEM, its style is applied to the link.

2.0/manual/config/visualisation/maps/map.txt · Last modified: 2012/09/18 14:30 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Link indicators

Overview

You can assign some triggers to a link between elements in a network map. When these triggers go into a problem state, the link can reflect that.

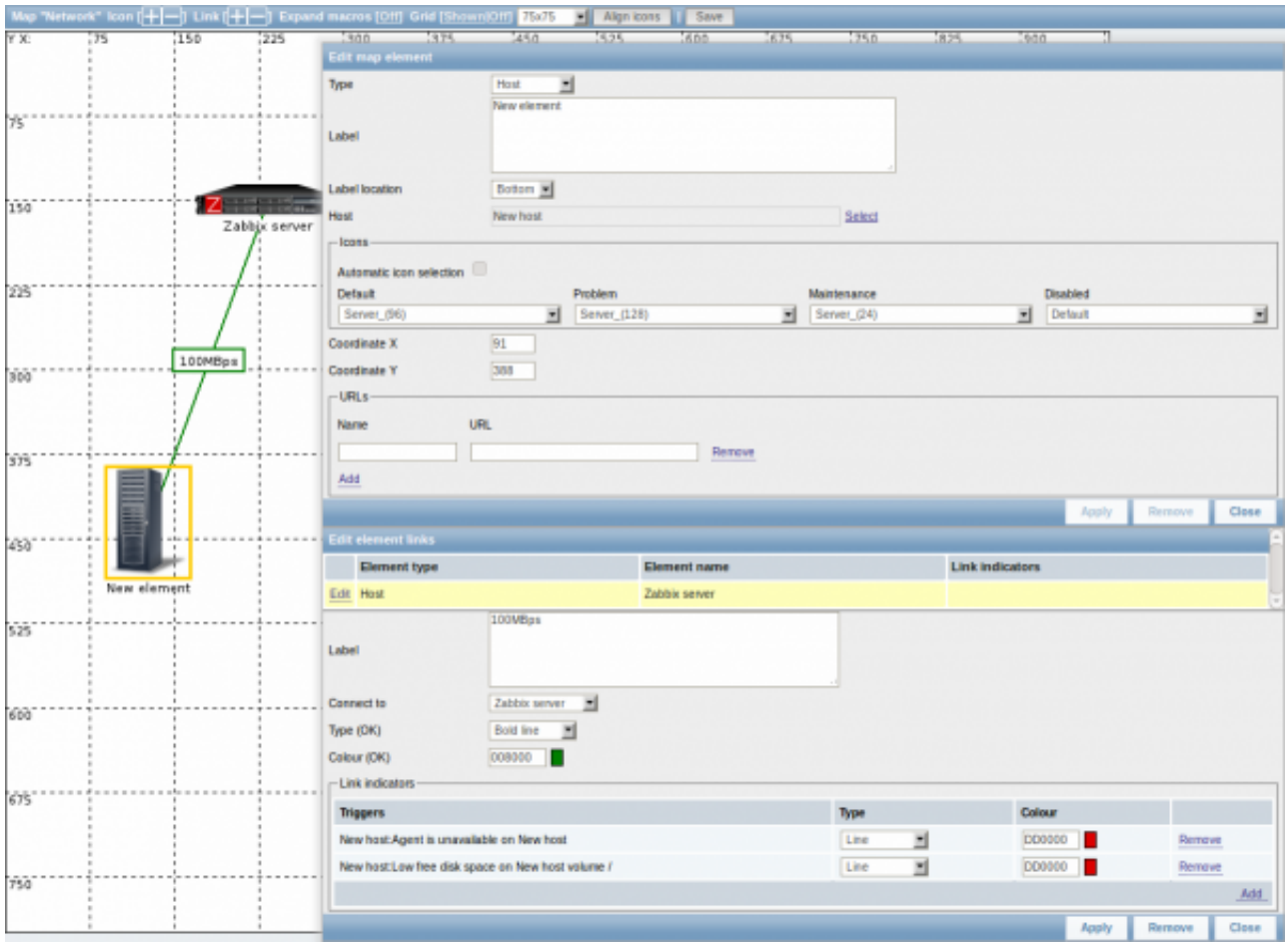
When you configure a link, you set the default link type and color. When you assign triggers to a link, you can assign different link types and colors with these triggers.

Should any of these triggers go into a problem state, their link style and color will be displayed on the link. So maybe your default link was a green line. Now, with the trigger in problem state, your link may become bold red (if you have defined it so).

Configuration

To assign triggers as link indicators, do the following:

- select a map element
- click on *Edit* in the *Edit element links* section before the appropriate link
- click on *Add* in the *Link indicators* block and select one or more triggers

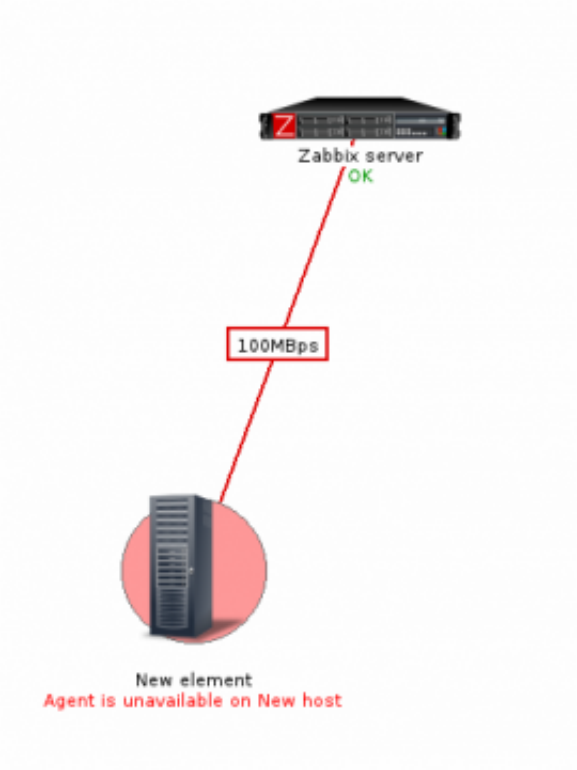


Added triggers can be seen in the *Link indicators* list.

You can set the link type and color for each trigger directly from the list. When done, click on *Apply*, close the form and save the map.

Display

In *Monitoring → Maps* the respective color will be displayed on the link if the trigger goes into a problem state.



If multiple triggers go into a problem state, the one with the highest severity will determine the link style and color. If multiple triggers with the same severity are assigned to the same map link, the one with the lowest ID takes precedence.

2.0/manual/config/visualisation/maps/links.txt · Last modified: 2012/03/08 10:25 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 Screens

Overview

On Zabbix screens you can group information from various sources for a quick overview on a single screen. Building the screens is quite easy and intuitive.

Essentially a screen is a table. You choose how many cells per table and what elements to display in the cells. The following elements can be displayed:

- simple graphs
- user-defined custom graphs
- maps
- other screens
- plain text information
- server information (overview)
- hosts information (overview)
- trigger information (overview)
- status of triggers (by host or hostgroup)
- system status
- data overview
- clock
- history of events
- history of actions
- URL (data taken from another location)

Browsers might not load an HTTP page included in a screen (using URL element), if Zabbix frontend is accessed over HTTPS.

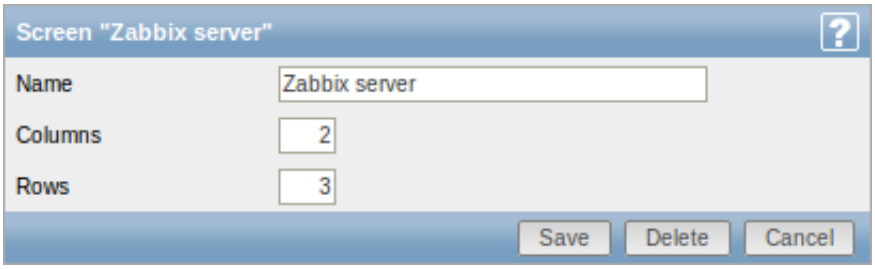
Screens that are ready can be viewed in *Monitoring → Screens*. They can also be added to the favourites section of the Dashboard.

To configure a screen you must first create it by defining its general properties and then add individual elements in the cells.

Creating a screen

To create a screen, do the following:

- Go to *Configuration → Screens*
- Click on *Create Screen*
- Edit general screen attributes



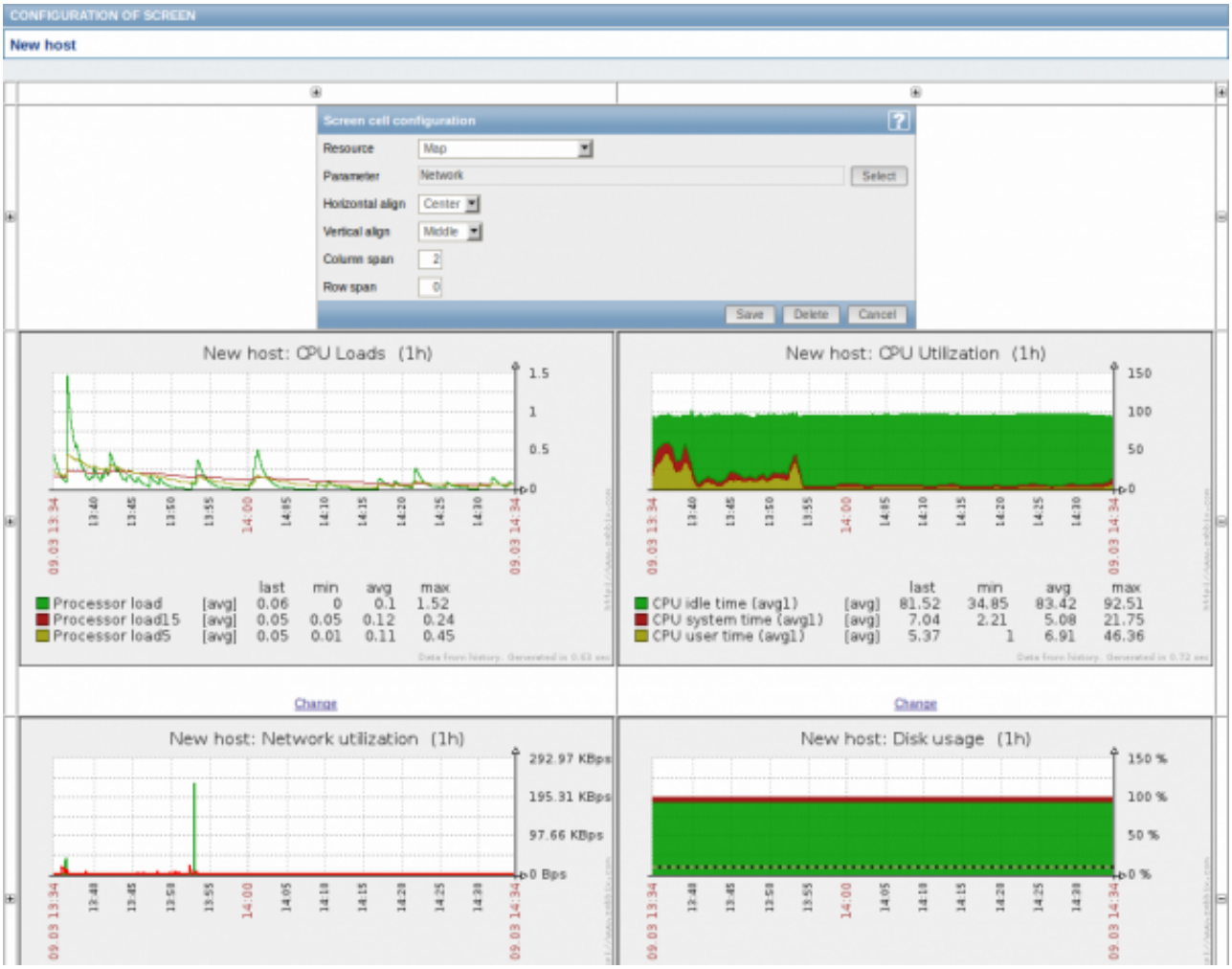
Give your screen a unique name and set the number of columns (vertical cells) and rows (horizontal cells). Click Save.

Now you can click on the screen name in the list to be able to add elements.

Adding elements

On a new screen you probably only see links named *Change*. Clicking those links opens a form whereby you set what to display in each cell.

On an existing screen you click on the existing elements to open the form whereby you set what to display.



Screen element attributes:

Parameter	Description
Resource	Information displayed in the cell: **Clock** - digital or analog clock displaying current server or local time **Data overview** - latest data for a group of hosts **Graph** - single custom graph **History of actions** - history of recent actions **History of events** - latest events **Hosts info** - high level host related information **Map** - single map **Plain text** - plain text data **Screen** - screen (one screen may contain other screens inside) **Server info** - server high-level information **Simple graph** - single simple graph **Status of hostgroup triggers** - status of triggers filtered by the hostgroup **Status of host triggers** - status of triggers filtered by the host **System status** - displays system status (similar to the Dashboard) **Triggers info** - high level trigger related information **Triggers overview** - status of triggers for a host group **URL** - include content from an external resource
Horizontal align	Possible values: **Center** **Left** **Right**
Vertical align	Possible values: **Middle** **Top** **Bottom**
Column span	Extend cell to a number of columns, same way as HTML column spanning works.
Row span	Extend cell to a number of rows, same way as HTML row spanning works.

Take note of the '+' and '-' controls on each side of the table.

Clicking on '+' above the table will add a column. Clicking on '-' beneath the table will remove a column.

Clicking on '+' on the left side of the table will add a row. Clicking on '-' on the right side of the table will remove a row.

If graph height is set as less than 120 pixels, no trigger will be displayed in the legend.

Dynamic elements

For some of the elements there is an extra option called *Dynamic item*. Checking this box at first does not to seem to change anything.

However, once you go to *Monitoring → Screens*, you may realize that now you have extra dropdowns there for selecting the host. Thus you have a screen where some elements display the same information while others display information depending on the currently selected host.

The benefit of this is that you do not need to create extra screens just because you want to see the same graphs containing data from various hosts.

Dynamic item option is available for several screen elements:

- Graphs (custom graphs)
- Simple graphs
- Plain text

Clicking on a dynamic graph opens it in full view; although with custom graphs that is currently supported with the default host selected only.

2.0/manual/config/visualisation/screens.txt · Last modified: 2013/08/31 03:11 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

4 Slide shows

Overview

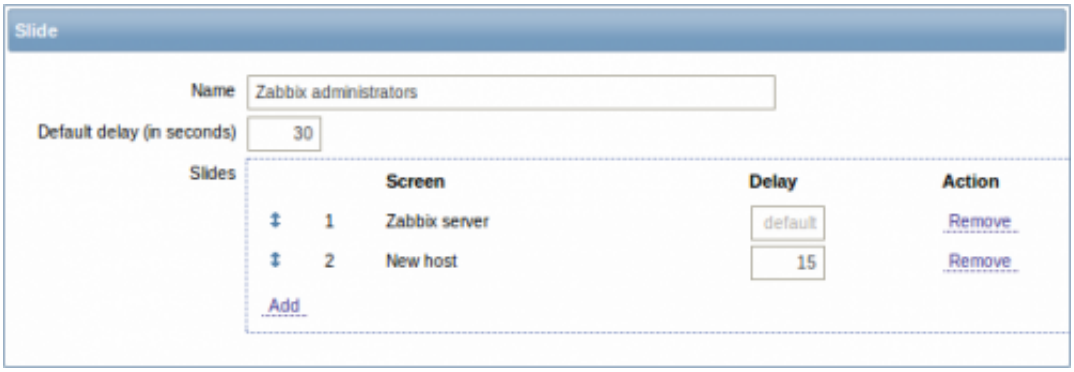
In a slide show you can configure that a number of screens are displayed one after another at set intervals.

Sometimes you might want to switch between some configured screens. While that can be done manually, doing that more than once or twice may become very tedious. This is where the slide show function comes to rescue.

Configuration

To create a slide show, do the following:

- Go to *Configuration → Slide shows*
- Click on *Create slide show*
- Edit slide show attributes



PARAMETER	Description
Name	Name of the slide show.
Default delay (in seconds)	How long one screen is displayed by default, before rotating to the next, in seconds.
Slides	List of screens to be rotated. Click on *Add* to select screens. The *Up/Down* arrow before the screen allows to drag a screen up and down in the sort order of display. If you want to display only, say, a single graph in the slide show, create a screen containing just that one graph.
Screen	Screen name.
Delay	A custom value for how long the screen will be displayed, in seconds. If set to 0, the *Default delay* value will be used.
Action	Click on *Remove* to remove a screen from the slide show.

The slide show in this example consists of two screens which will be displayed in the following order:

Zabbix server ⇒ Displayed for 30 seconds ⇒ New host ⇒ Displayed for 15 seconds ⇒ Zabbix server ⇒ Displayed for 30 seconds ⇒ New host ⇒ ...

Display

Slide shows that are ready can be viewed in *Monitoring* → *Screens* and then choosing *Slide shows* from the dropdown.

With the Menu option next to the dropdown, you can accelerate or slow down the display by choosing a slide delay multiplier:

Refresh time multiplier
x0.25
x0.5
x1
x1.5
x2
x3
x4
x5

If a delay ends up as being less than 5 seconds (either by having entered a delay less than 5 seconds or by using the slide delay multiplier), a 5-second minimum delay will be used.

2.0/manual/config/visualisation/slides.txt · Last modified: 2013/05/03 11:33 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

6 Templates

Overview

A template is a set of entities that can be conveniently applied to multiple hosts.

The entities may be:

- items
- triggers
- graphs
- applications
- screens (*since Zabbix 2.0*)
- low-level discovery rules (*since Zabbix 2.0*)

As many hosts in real life are identical or fairly similar so it naturally follows that the set of entities (items, triggers, graphs,...) you have created for one host, may be useful for many. Of course, you could copy them to each new host, but that would be a lot of manual work. Instead, with templates you can copy them to one template and then apply the template to as many hosts as needed.

When a template is linked to a host, all entities (items, triggers, graphs,...) of the template are added to the host. Templates are assigned to each individual host directly (and not to a host group).

Templates are often used to group entities for particular services or applications (like Apache, MySQL, PostgreSQL, Postfix...) and then applied to hosts running those services.

Another benefit of using templates is when something has to be changed for all the hosts. Changing something on the template level once will propagate the change to all the linked hosts.

Thus, the use of templates is an excellent way of reducing one's workload and streamlining the Zabbix configuration.

Proceed to creating and configuring a template.

2.0/manual/config/templates.txt · Last modified: 2012/04/02 13:29 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Configuring a template

Overview

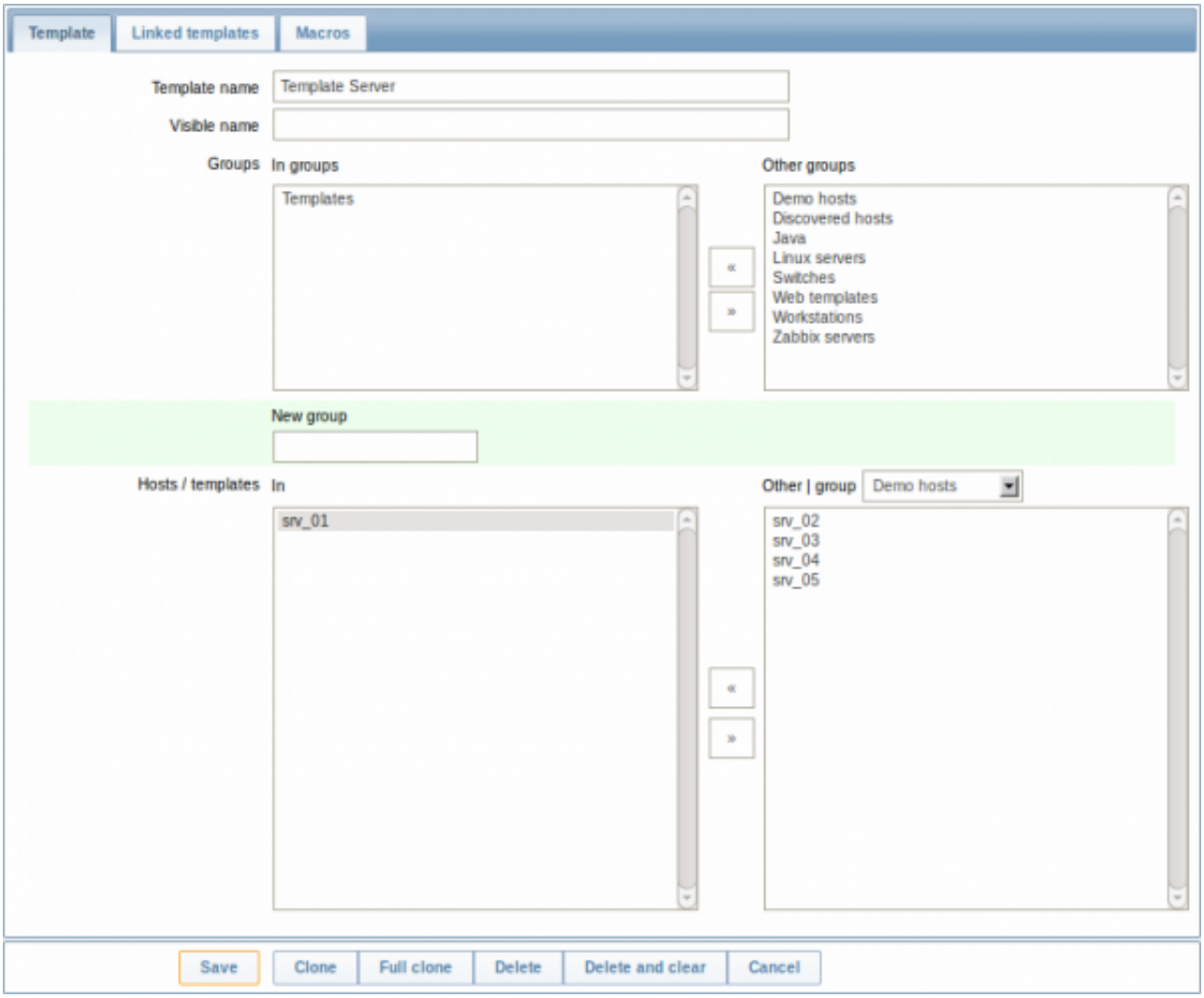
Configuring a template requires that you first create a template by defining its general parameters and then you add entities (items, triggers, graphs etc.) to it.

Creating a template

To create a template, do the following:

- Go to *Configuration → Templates*
- Click on *Create template*
- Edit template attributes

The **Template** tab contains general template attributes.



Template attributes:

Parameter	Description
Template name	Unique template name.
Visible name	If you set this name, it will be the one visible in lists, maps, etc.
Groups	Host/template groups the template belongs to.
New group	A new group can be created to hold the template. Ignored, if empty.
Hosts/Templates	List of hosts/templates the template is applied to.

The **Linked templates** tab allows you to link one or more "nested" templates to this template. All entities (items, triggers, graphs etc.) will be inherited from the linked templates.

To link a new template, click on *Add*. To unlink a template, use one of the two options:

- *Unlink* - unlink the template, but preserve its items, triggers and graphs
- *Unlink and clear* - unlink the template and remove all its items, triggers and graphs

The **Macros** tab allows you to define template-level user macros.

Buttons:

Save	Save the template. The saved template should appear in the list.
Clone	Create another template based on the properties of the current template, including the entities (items, triggers, etc) inherited from linked templates.
Full clone	Create another template based on the properties of the current template, including the entities (items, triggers, etc) both inherited from linked templates and directly attached to the current template.
Delete	Delete the template; entities of the template (items, triggers, etc) remain with the linked hosts.
Delete and clear	Delete the template and all its entities from linked hosts.
Cancel	Cancel the editing of template properties.

With a template created, it is time to add some entities to it.

Items have to be added to a template first. Triggers and graphs cannot be added without the corresponding item.

Adding items, triggers, graphs

To add items to the template, do the following:

- Go to *Configuration* → *Hosts* (or *Templates*)
- Click on *Items* in the row of the required host/template
- Mark the checkboxes of items you want add to the template
- Select *Copy selected to...* below the item list and click on *Go*
- Select the template (or group of templates) the items should be copied to and click on *Copy*

All the selected items should be copied to the template.

Adding triggers and graphs is done in similar fashion (from the list of triggers and graphs respectively), again, keeping in mind that they can only be added if the required items are added first.

Adding screens

To add screens to a template in *Configuration → Templates*, do the following:

- Click on *Screens* in the row of the template
- Configure a screen following the usual method of configuring screens

The elements that can be included in a template screen are: simple graph, custom graph, clock, plain text, URL.

Configuring low-level discovery rules

See the low-level discovery section of the manual.

2.0/manual/config/templates/template.txt · Last modified: 2012/12/06 17:50 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Linking/unlinking

Overview

Linking is a process whereby templates are applied to hosts, whereas unlinking removes the association with the template from a host.

Templates are linked directly to individual hosts and not to host groups. Simply adding a template to a host group will not link it. Host groups are used only for logical grouping of hosts and templates.

Linking a template

To link a template to the host, do the following:

- Go to *Configuration → Hosts*
- Click on the required host and switch to the *Templates* tab
- Click on *Add*
- Select one or several templates in the popup window
- Click on *Save* in the host attributes form

The host will now have all the items, triggers, graphs, applications, screens and low-level discovery rules of the template.

Linking multiple templates to the same host will fail if in those templates there are items with the same item key or applications with the same name. And, as triggers and graphs use items, they cannot be linked to a single host from multiple templates either, if using identical item keys.

When entities (items, triggers, graphs etc.) are added from the template:

- previously existing identical entities on the host are updated as entities of the template
- entities from the template are added
- any directly linked entities that, prior to template linkage, existed only on the host remain untouched

In the lists, all entities from the template now are prefixed by the template name, indicating that these belong to the particular template. The template name itself (in grey text) is a link allowing to access the list of those entities on the template level.

If some entity (item, trigger, graph etc.) is not prefixed by the template name, it means that it existed on the host before and was not added by the template.

Entity uniqueness criteria

When adding entities (items, triggers, graphs etc.) from a template it is important to know what of those entities already exist on the host and need to be updated and what entities differ. The uniqueness criteria for deciding upon the sameness/difference are:

- for items - the item key
- for triggers - trigger name and expression
- for custom graphs - graph name and its items

- for applications - application name

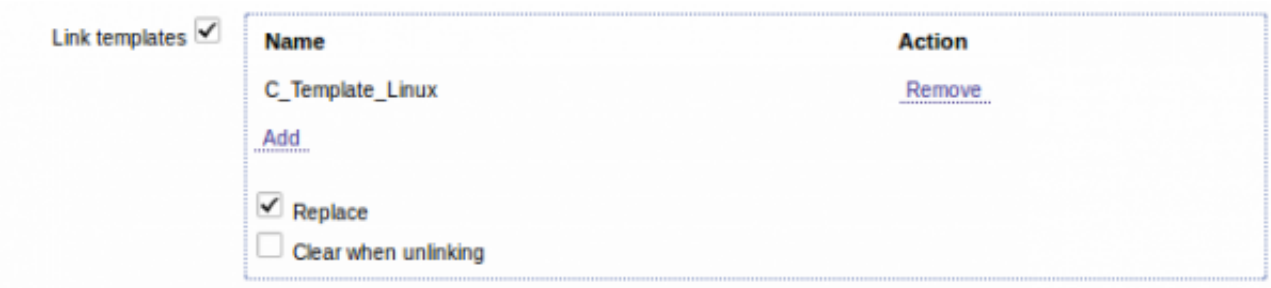
Linking templates to several hosts

There are some ways of mass-applying templates (to many hosts at once):

- To link a template to many hosts, in *Configuration* → *Templates*, click on the template, then select hosts from the respective group in the *Other* box, click on **«** and save the template.

Vice versa, if you select the linked hosts in the *In* box, click on **»** and save the template, you unlink the template from these hosts (while the hosts will still inherit the items, triggers, graphs etc. from the template).

- To update template linkage of many hosts, in *Configuration* → *Hosts* select some hosts by marking their checkboxes, then choose **Mass update** below the list, click on *Go* and then from all the options select to link additional templates:



Click on *Add* to select the templates to link. The *Replace* option allows to unlink the previously linked templates, before linking the new ones. When choosing to replace, there is also the *Clear when unlinking* option. Marking that will remove all the entities (items, triggers, graphs, etc) previously linked to the hosts by the linked templates.

Zabbix offers a sizable set of predefined templates. You can use these for reference, but beware of using them unchanged in production as they may contain too many items and poll for data too often. If you feel like using them, finetune them to fit you real needs.

Editing linked entities

If you try to edit an item or trigger that was linked from the template, you may realize that many key options are disabled for editing. This makes sense as the idea of templates is that things are edited in one-touch manner on the template level. However, you still can, for example, enable/disable an item on the individual host and set the update interval, history length and some other parameters.

If you want to edit the entity fully, you have to edit it on the template level (template level shortcut is displayed in the form name), keeping in mind that these changes will affect all hosts that have this template linked to them.

Unlinking a template

To unlink a template from a host, do the following:

- Go to *Configuration* → *Hosts*
- Click on the required host and switch to the *Templates* tab

- Click on *Unlink* or *Unlink and clear* next to the template to unlink
- Click on *Save* in the host attributes form

Choosing the *Unlink* option will simply remove association with the template, while leaving all its entities (items, triggers, graphs etc.) with the host.

Choosing the *Unlink and clear* option will remove both the association with the template and all its entities (items, triggers, graphs etc.).

2.0/manual/config/templates/linking.txt · Last modified: 2012/12/07 11:54 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 Nesting

Overview

Nesting is a way of one template encompassing one or more other templates.

As it makes sense to separate out on individual templates entities for various services, applications etc. you may end up with quite a few templates all of which may need to be linked to quite a few hosts. To simplify the picture, it is possible to link some templates together, in one “nested” template.

The benefit of nesting is that then you have to link only the one template to the host and the host will inherit all entities of the linked templates automatically.

Configuring a nested template

If you want to link some templates, to begin with you can take an existing template or a new one, then:

- Open the template properties form
- Look for the *Linked templates* tab
- Click on *Add*, select the templates to link in the popup window
- Click on *Save* in the template properties form

Now the template should have all the entities (items, triggers, custom graphs etc.) of the linked templates.

To unlink any of the linked templates, in the same form use the *Unlink* or *Unlink and clear* buttons and click on *Save*.

Choosing the *Unlink* option will simply remove the association with the other template, while not removing all its entities (items, triggers, graphs etc).

Choosing the *Unlink and clear* option will remove both the association with the other template and all its entities (items, triggers, graphs etc).

2.0/manual/config/templates/nesting.txt · Last modified: 2012/03/26 10:43 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

7 Notifications upon events

Overview

Assuming that we have configured some items and triggers and now are getting some events happening as a result of triggers changing state, it is time to consider some actions.

To begin with, we would not want to stare at the triggers or events list all the time. It would be much better to receive notification if something significant (such as a problem) has happened. Also, when problems occur, we would like to see that all the people concerned are informed.

That is why sending notifications is one of the primary actions offered by Zabbix. Who and when should be notified upon a certain event can be defined.

To be able to send and receive notifications from Zabbix you have to:

- define some media
- configure an action that sends a message to one of the defined media

Actions consist of *conditions* and *operations*. Basically, when conditions are met, operations are carried out. The two principal operations are sending a message (notification) and executing a remote command.

For discovery and auto-registration created events, some additional operations are available. Those include adding or removing a host, linking a template etc.

2.0/manual/config/notifications.txt · Last modified: 2011/11/11 15:58 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Media types

Overview

Media are the delivery channels used for sending notifications and alerts in Zabbix.

You can configure several media types:

- E-mail
- SMS
- Jabber
- Ez Texting
- Custom alertscripts

2.0/manual/config/notifications/media.txt · Last modified: 2011/12/29 09:30 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 E-mail

Overview

To configure e-mail as the delivery channel for messages, you need to configure e-mail as the media type and assign specific addresses to users.

Configuration

To configure e-mail as the media type:

- Go to *Administration→Media types*
- Click on *Create media type* (or click on *E-mail* in the list of pre-defined media types).

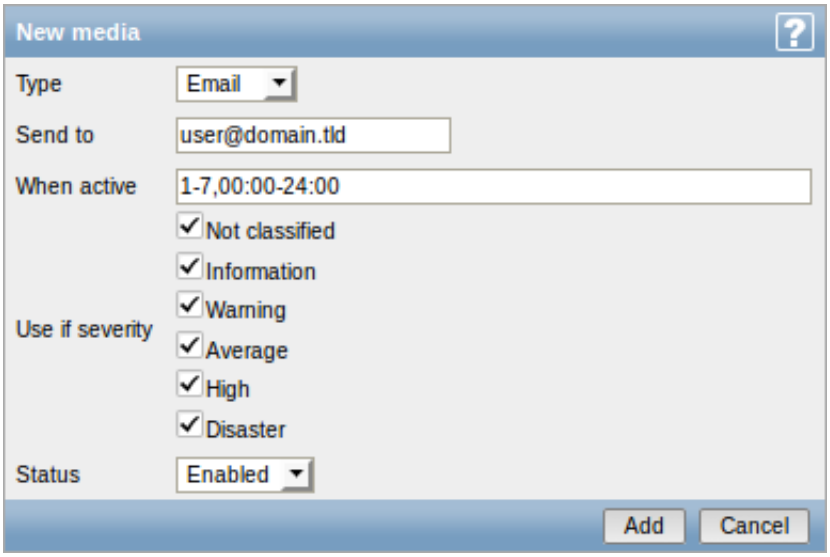
Media type attributes:

Parameter	Description
Description	Name of the media type.
Type	Select *Email* as the type.
SMTP server	Set an SMTP server to handle outgoing messages.
SMTP helo	Set a correct SMTP helo value, normally a domain name.
SMTP email	The address entered here will be used as the **From** address for the messages sent.

User media

To assign a specific address to the user:

- Go to *Administration→Users*
- Open the user properties form
- In Media tab, click on *Add*



User media attributes:

Parameter	Description
Type	Select *Email* as the type.
Send to	Specify the e-mail address to send the messages to.
When active	You can limit the time when messages are sent, for example, the working days only (1-5,09:00-18:00). See the Time period specification page for description of the format.
Use if severity	Mark the checkboxes of trigger severities that you want to receive notifications for.
Status	Status of the user media. **Enabled** - is in use. **Disabled** - is not being used.

2.0/manual/config/notifications/media/email.txt · Last modified: 2012/04/24 10:23 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 SMS

Overview

Zabbix supports the sending of SMS messages using a serial GSM modem connected to Zabbix server's serial port.

Make sure that:

- The speed of the serial device (normally /dev/ttyS0 under Linux) matches that of the GSM modem. Zabbix does not set the speed of the serial link. It uses default settings.
- The 'zabbix' user has read/write access to the serial device. Run the command ls -l /dev/ttyS0 to see current permissions of the serial device.
- The GSM modem has PIN entered and it preserves it after power reset. Alternatively you may disable PIN on the SIM card. PIN can be entered by issuing command AT+CPIN="NNNN" (NNNN is your PIN number, the quotes must be present) in a terminal software, such as Unix minicom or Windows HyperTerminal.

Zabbix has been tested with these GSM modems:

- Siemens MC35
- Teltonika ModemCOM/G10

To configure SMS as the delivery channel for messages, you also need to configure SMS as the media type and enter the respective phone numbers for the users.

Configuration

To configure SMS as the media type:

- Go to *Administration→Media types*
- Click on *Create media type* (or click on *SMS* in the list of pre-defined media types).

Media type attributes:

Parameter	Description
Description	Name of the media type.
Type	Select *SMS* as the type.
GSM modem	Set the serial device name of the GSM modem.

User media

To assign a phone number to the user:

- Go to *Administration→Users*
- Open the user properties form
- In Media tab, click on *Add*

User media attributes:

Parameter	Description
Type	Select *SMS* as the type.
Send to	Specify the phone number to send messages to.
When active	You can limit the time when messages are sent, for example, the working days only (1-5,09:00-18:00). See the Time period specification page for description of the format.
Use if severity	Mark the checkboxes of trigger severities that you want to receive notifications for.
Status	Status of the user media. **Enabled** - is in use. **Disabled** - is not being used.

2.0/manual/config/notifications/media/sms.txt · Last modified: 2012/04/24 10:24 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 Jabber

Overview

Zabbix supports sending Jabber messages.

When sending notifications, Zabbix tries to look up the Jabber SRV record first, and if that fails, it uses an address record for that domain. Among Jabber SRV records, the one with the highest priority and maximum weight is chosen. If it fails, other records are not tried.

To configure Jabber as the delivery channel for messages, you need to configure Jabber as the media type and enter the respective addresses for the users.

Configuration

To configure Jabber as the media type:

- Go to *Administration→Media types*
- Click on *Create media type* (or click on *Jabber* in the list of pre-defined media types).

Media type attributes:

Parameter	Description
Description	Name of the media type.
Type	Select *Jabber* as the type.
Jabber identifier	Enter Jabber identifier.
Password	Enter Jabber password.

User media

To assign a Jabber address to the user:

- Go to *Administration→Users*
- Open the user properties form
- In Media tab, click on *Add*

User media attributes:

Parameter	Description
Type	Select *Jabber* as the type.
Send to	Specify the address to send messages to.
When active	You can limit the time when messages are sent, for example, the working days only (1-5,09:00-18:00). See the Time period specification page for description of the format.
Use if severity	Mark the checkboxes of trigger severities that you want to receive notifications for.
Status	Status of the user media. **Enabled** - is in use. **Disabled** - is not being used.

2.0/manual/config/notifications/media/jabber.txt · Last modified: 2012/04/24 10:25 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

4 Ez Texting

Overview

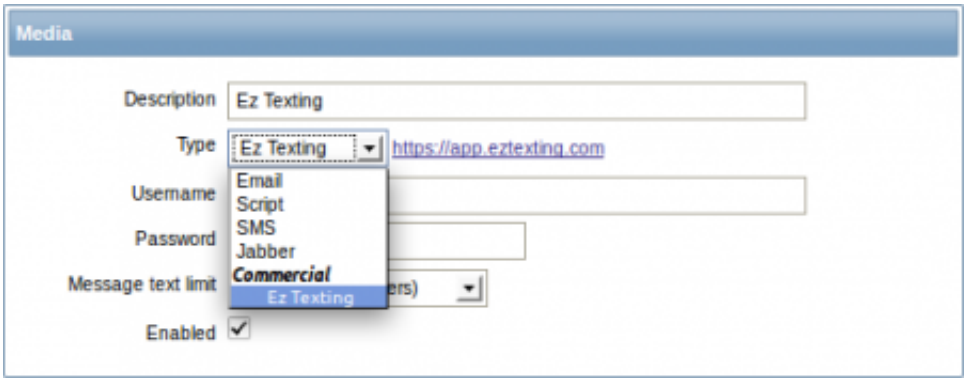
You can use Zabbix technological partner [http://www.zabbix.com/partners.php#Technology_Partners] Ez Texting for message sending.

To configure Ez Texting as the delivery channel for messages, you need to configure Ez Texting as the media type and assign recipient identification to the users.

Configuration

To configure Ez Texting as the media type:

- Go to *Administration→Media types*
- Click on *Create media type*



Media type attributes:

Parameter	Description
Description	Name of the media type.
Type	Select *Ez Texting* as the type.
Username	Enter the Ez Texting username.
Password	Enter the Ez Texting password.
Message text limit	Select the message text limit. **USA (160 characters)** **Canada (136 characters)**

User media

To assign Ez Texting recipient identification to the user:

- Go to *Administration→Users*
- Open the user properties form
- In Media tab, click on *Add*

User media attributes:

Parameter	Description
Type	Select the Ez Texting media type.
Send to	Specify the recipient to send the messages to.
When active	You can limit the time when messages are sent, for example, the working days only (1-5,09:00-18:00). See the Time period specification page for description of the format.
Use if severity	Mark the checkboxes of trigger severities that you want to receive notifications for.
Status	Status of the user media. **Enabled** - is in use. **Disabled** - is not being used.

2.0/manual/config/notifications/media/ez_texting.txt · Last modified: 2012/04/24 10:26 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

5 Custom alertscripts

Overview

If you are not satisfied with existing media types for sending alerts there is an alternative way to do that. You can create a script that will handle the notification your way. These scripts are located in the directory defined in the Zabbix server configuration file **AlertScriptsPath** variable. When alert script is executed it gets 3 command-line variables (as $1, $2 and $3 respectively):

- To
- Subject
- Message

The recipient ("To") is specified in user media properties. Here is an example alert script:

```
#!/bin/bash

to=$1
subject=$2
body=$3

cat <<EOF | mail -s "$subject" "$to"
$body
EOF
```

Environment variables are not preserved or created for the script, so they should be handled explicitly.

Configuration

To configure custom alertscripts as the media type:

- Go to *Administration→Media types*
- Click on *Create media type*

Media type attributes:

Parameter	Description
Description	Name of the media type.
Type	Select *Script* as the type.
Script name	Enter the name of the script.

User media

To assign custom alertscripts to the user:

- Go to *Administration→Users*
- Open the user properties form
- In Media tab, click on *Add*

User media attributes:

Parameter	Description
Type	Select the custom alertscripts media type.
Send to	Specify the recipient to receive the alerts.
When active	You can limit the time when alertscripts are executed, for example, the working days only (1-5,09:00-18:00). See the Time period specification page for description of the format.
Use if severity	Mark the checkboxes of trigger severities that you want to activate the alertscript for.
Status	Status of the user media. **Enabled** - is in use. **Disabled** - is not being used.

2.0/manual/config/notifications/media/script.txt · Last modified: 2012/08/02 14:30 by dimir

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Actions

Overview

If you want some operations taking place as a result of events (for example, notifications sent), you need to configure actions.

Actions can be defined for the events of all three sources:

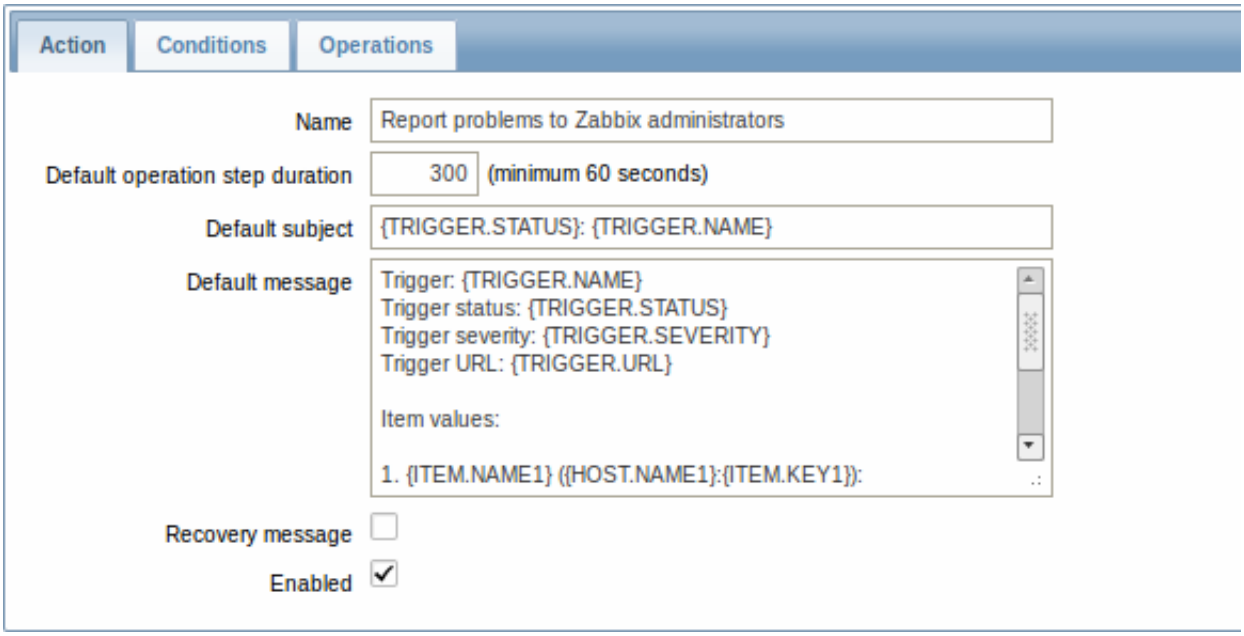
- Triggers - when trigger status changes
- Discovery - when discovery takes place
- Auto registration - when new active agents auto-register

Configuring an action

To configure an action, do the following:

- Go to *Configuration → Actions*
- From the *Event source* dropdown select the required source
- Click on *Create action*
- Set general action attributes
- Choose the operation to carry out, in Operations tab
- Choose the conditions upon which the operation is carried out, in Conditions tab

General action attributes:



Parameter	Description
Name	Unique action name.
Default operation	

step duration	Time period of one escalation step, by default.
Default subject	Default message subject. The subject may contain macros.
Default message	Default message. The message may contain macros.
Recovery message	If enabled, Zabbix will send a recovery message after the original problem is resolved. The messages will be sent only to those who received any messages regarding the problem before.
Recovery subject	Recovery message subject. It may contain macros.
Recovery message	Recovery message. It may contain macros. Starting with Zabbix 2.0.4, recovery message can also be left empty.
Enabled	Mark the checkbox to enable the action. Otherwise it will be disabled.

2.0/manual/config/notifications/action.txt · Last modified: 2013/05/24 15:43 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Operations

Overview

You can define the following operations for all events:

- send a message
- execute a remote command (including IPMI)

For discovery events, there are additional operations available:

- add host
- remove host
- enable host
- disable host
- add to group
- delete from group
- link to template
- unlink from template

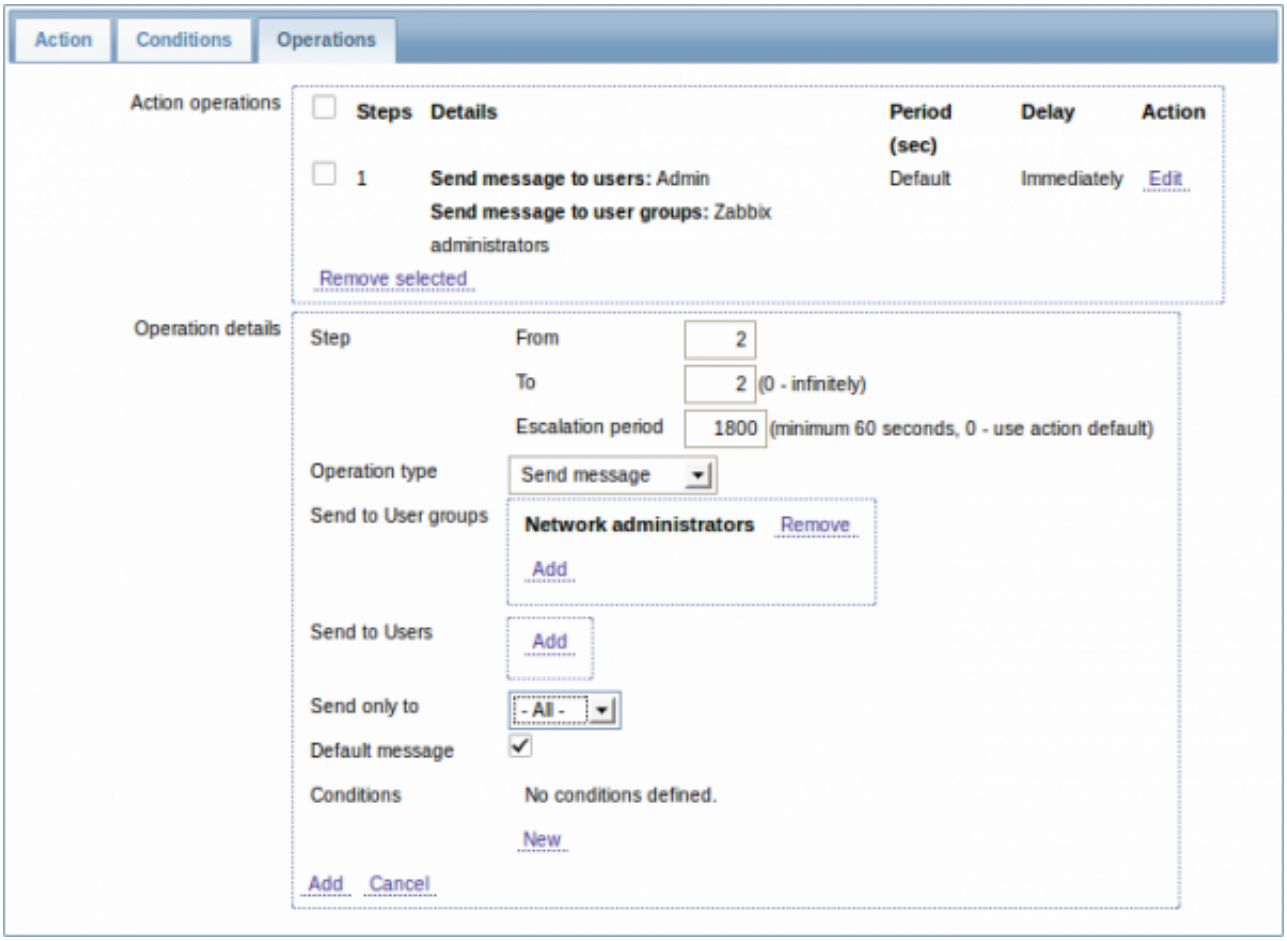
The additional operations available for auto-registration events are:

- add host
- disable host
- add to group
- link to template

Configuring an operation

To configure an operation, go to *Operations* tab in the action properties form and click on *New*. Edit the operation step and click on *Add* to add to the list of *Action operations*.

Operation attributes:



Parameter	Description
Step	Escalation schedule: **From** - execute starting with this step **To** - execute until this step (0=infinity, execution will not be limited) **Escalation period** - custom interval between these steps (0=use default period).
Operation type	Two operation types available for all events: **Send message** - send message to user **Remote command** - execute a remote command More operations are available for discovery and auto-registration based events (see above).
Conditions	Condition for executing operation: **Not ack** - only when the event is unacknowledged **Ack** - only when the event is acknowledged
Operation type: send message	
Send to user groups	Click on *Add* to select user groups to send the message to.
Send to users	Click on *Add* to select users to send the message to.
Send only to	Send message to all defined media types or a selected one only.
Default message	If selected, the default message will be used (see general action attributes).
Subject	Subject of the custom message. The subject may contain macros.
Message	The custom message. The message may contain macros.
Operation type: remote command	
Target list	Select current host, other hosts or host groups as targets to execute the command on.

Type	Select the command type: **IPMI** - execute an IPMI command **Custom script** - execute a custom set of commands. You can select to execute the command on Zabbix agent or Zabbix server. **SSH** - execute an SSH command **Telnet** - execute a Telnet command **Global script** - execute one of the global scripts defined in *Administration→Scripts*
Commands	Enter the command(s).

2.0/manual/config/notifications/action/operation.txt · Last modified: 2012/04/24 10:59 by martins-v

Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Sending message

Overview

Sending a message is one of the best ways of notifying people about a problem. That is why it is one of the primary actions offered by Zabbix.

Configuration

To be able to send and receive notifications from Zabbix you have to:

- define the media to send a message to
- configure an action operation that sends a message to one of the defined media

Zabbix sends notifications only to those users that have at least 'read' permissions to the host that generated the event. At least one host of a trigger expression must be accessible.

You can configure custom scenarios for sending messages using escalations.

To successfully receive and read e-mails from Zabbix, e-mail servers/clients must support standard 'SMTP/MIME e-mail' format since Zabbix sends UTF-8 data (If the subject contains ASCII characters only, it is not UTF-8 encoded.). The subject and the body of the message are base64-encoded to follow 'SMTP/MIME e-mail' format standard.

Tracking messages

You can view the status of messages sent in *Monitoring → Events*.

In the *Actions* column you can see summarized information about actions taken. In there green numbers represent messages sent, red ones - failed messages. *In progress* indicates that an action is initiated. *Failed* informs that no action has executed successfully.

If you click on the event time to view event details, you will also see the *Message actions* block containing details of messages sent (or not sent) due to the event.

In *Administration → Audit*, if you select *Actions* from the dropdown, you will see details of all actions taken for those events that have an action configured.

2.0/manual/config/notifications/action/operation/message.txt · Last modified: 2012/09/14 22:51 by zalex_ua
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Remote commands

Overview

With remote commands you can define that a certain pre-defined command is automatically executed on the monitored host upon some condition.

Thus remote commands are a powerful mechanism for smart pro-active monitoring.

In the most obvious uses of the feature you can try to:

- Automatically restart some application (web server, middleware, CRM) if it does not respond
- Use IPMI 'reboot' command to reboot some remote server if it does not answer requests
- Automatically free disk space (removing older files, cleaning /tmp) if running out of disk space
- Migrate a VM from one physical box to another depending on the CPU load
- Add new nodes to a cloud environment upon insufficient CPU (disk, memory, whatever) resources

Configuring an action for remote commands is similar to that for sending a message, the only difference being that Zabbix will execute a command instead of sending a message.

Remote commands are not supported on Zabbix proxies, so for commands from Zabbix server to agent a direct connection is required.

Remote commands are limited to 255 characters. Multiple commands can be executed one after another by placing them on e new line. Remote commands may contain macros!

This tutorial provides step-by-step instructions on how to set up remote commands.

Configuration

Those remote commands that are executed on Zabbix agent (custom scripts) must be first enabled in the respective zabbix_agentd.conf.

Make sure that the **EnableRemoteCommands** parameter is set to **1** and uncommented. Restart agent daemon if changing this parameter.

Remote commands do not work with active Zabbix agents.

Then, when configuring a new action in *Configuration→Actions*:

- In the *Operations* tab, select the **Remote command** operation type
- Select the remote command type (IPMI, Custom script, SSH, Telnet, Global script)
- Enter the remote command

For example:

```
sudo /etc/init.d/apache restart
```

In this case, Zabbix will try to restart an Apache process. With this command, make sure that the command is executed on Zabbix agent (mark the respective radio button against *Execute on*).

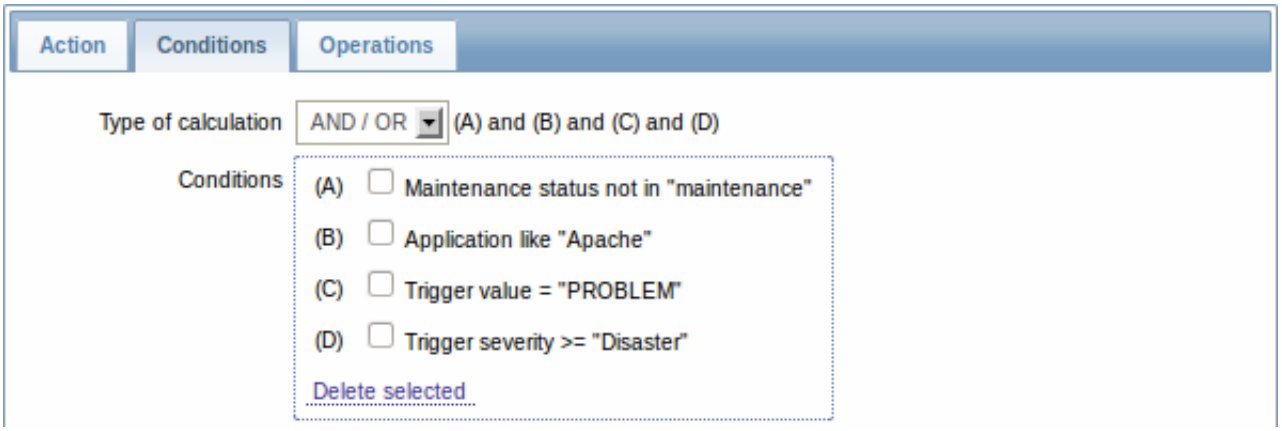
Note the use of **sudo** - Zabbix user does not have permissions to restart system services by default. See below for hints on how to configure **sudo**.

Zabbix agent should run on the remote host and accept incoming connections. Zabbix agent executes commands in background.

Zabbix does not check if a command has been executed successfully.

Remote commands on Zabbix agent are executed without timeout by the system.run[,nowait] key. On Zabbix server remote commands are executed with timeout as set in the TrapperTimeout parameter of zabbix_server.conf file.

- In the *Conditions* tab, define the appropriate conditions. In this example, set that the action is activated upon any disaster problems with one of Apache applications.



Access permissions

Make sure that the 'zabbix' user has execute permissions for configured commands. One may be interested in using **sudo** to give access to privileged commands. To configure access, execute as root:

```
# visudo
```

Example lines that could be used in *sudoers* file:

```
# allows 'zabbix' user to run all commands without password.
zabbix ALL=NOPASSWD: ALL
```

```
# allows 'zabbix' user to restart apache without password.
zabbix ALL=(ALL) NOPASSWD: /etc/init.d/apache restart
```

On some systems *sudoers* file will prevent non-local users from executing commands. To change this, comment out **requiretty** option in */etc/sudoers*.

Remote commands with multiple interfaces

If the target system has multiple interfaces of the selected type (Zabbix agent or IPMI), remote commands will be executed on the default interface.

Examples

Example 1

Restart of Windows on certain condition.

In order to automatically restart Windows upon a problem detected by Zabbix, define the following actions:

PARAMETER	Description
Operation type	'Remote command'
Type	'Custom script'
Command	c:\windows\system32\shutdown.exe -r -f

Example 2

Restart the host by using IPMI control.

PARAMETER	Description
Operation type	'Remote command'
Type	'IPMI'
Command	reset on

Example 3

Power off the host by using IPMI control.

PARAMETER	Description
Operation type	'Remote command'
Type	'IPMI'
Command	power off

2.0/manual/config/notifications/action/operation/remote_command.txt · Last modified: 2013/05/24 09:09 by richlv
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 Additional operations

Overview

For discovery events, there are additional operations available:

- add host
- remove host
- enable host
- disable host
- add to group
- delete from group
- link to template
- unlink from template

The additional operations available for auto-registration events are:

- add host
- disable host
- add to group
- link to template

Adding host

Hosts are added during the discovery process, as soon as a host is discovered, rather than at the end of the discovery process.

As network discovery can take some time due to many unavailable hosts/services having patience and using reasonable IP ranges is advisable.

When adding a host, its name is decided by the standard **gethostbyname** function. If the host can be resolved, resolved name is used. If not, the IP address is used. Besides, if IPv6 address must be used for a host name, then all ":" (colons) are replaced by "_" (underscores), since colons are not allowed in host names.

If performing discovery by a proxy, currently hostname lookup still takes place on Zabbix server.

If a host already exists in Zabbix configuration with the same name as a newly discovered one, versions of Zabbix prior to 1.8 would add another host with the same name. Zabbix 1.8.1 and later adds **_N** to the hostname, where **N** is increasing number, starting with 2.

2.0/manual/config/notifications/action/operation/other.txt · Last modified: 2012/08/28 11:32 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

4 Using macros in messages

Overview

In message subjects and message text you can use macros for more efficient problem reporting.

A full list of macros supported by Zabbix is available.

Examples

Examples here illustrate how you can use macros in messages.

Example 1

Message subject:

```
{TRIGGER.NAME}: {TRIGGER.STATUS}
```

When you receive the message, the message subject will be replaced by something like:

```
Processor load is too high on server zabbix.zabbix.com: PROBLEM
```

Example 2

Message:

```
Processor load is: {zabbix.zabbix.com:system.cpu.load[,avg1].last(0)}
```

When you receive the message, the message will be replaced by something like:

```
Processor load is: 1.45
```

Example 3

Message:

```
Latest value: {{HOST.HOST}:{ITEM.KEY}.last(0)}
MAX for 15 minutes: {{HOST.HOST}:{ITEM.KEY}.max(900)}
MIN for 15 minutes: {{HOST.HOST}:{ITEM.KEY}.min(900)}
```

When you receive the message, the message will be replaced by something like:

```
Latest value: 1.45
MAX for 15 minutes: 2.33
MIN for 15 minutes: 1.01
```

2.0/manual/config/notifications/action/operation/macros.txt · Last modified: 2013/07/19 11:08 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Conditions

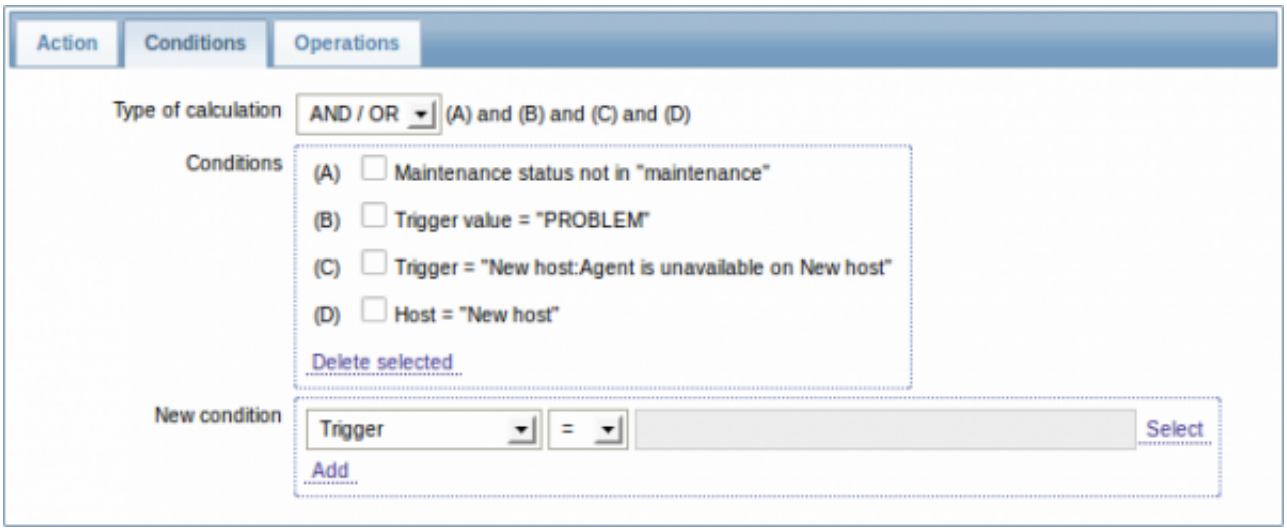
Overview

An action is executed only in case an event matches a defined set of conditions.

Configuration

To set a condition:

- Go to *Conditions* tab in the action properties form
- Select conditions from the *New condition* dropdowns and click on *Add*
- Select the type of calculation (with more than one condition)



The following conditions can be set for trigger-based actions:

Condition type	Supported operators	Description
Application	= like not like	Specify an application or an application to exclude. = - event belongs to a trigger of the item that is linked to the specified application. **like** - event belongs to a trigger of the item that is linked to an application containing the string. **not like** - event belongs to a trigger of the item that is linked to an application not containing the string.
Host group	= <>	Specify a host group or a host group to exclude. = - event belongs to this host group. <> - event does not belong to this host group.
Host template	= <>	Specify a host template or a template to exclude. = - event belongs to a trigger inherited from this host template. <> - event does not belong to a trigger inherited from this host template.
Host	= <>	Specify a host or a host to exclude. = - event belongs to this host. <> - event does not belong to this host.
Trigger	= <>	Specify a trigger or a trigger to exclude. = - event is generated by this trigger. <> - event is generated by any other trigger, except this one.

Trigger name	like not like	Specify a string in the trigger name or a string to exclude. **like** - event is generated by a trigger, containing this string in the name. Case sensitive. **not like** - this string cannot be found in the trigger name. Case sensitive. *Note*: Entered value will be compared to trigger name with all macros expanded.
Trigger severity	= <> >= <=	Specify trigger severity. = - equal to trigger severity <> - not equal to trigger severity >= - more or equal to trigger severity <= - less or equal to trigger severity
Trigger value	=	Specify a trigger value. = - equal to trigger value (OK or PROBLEM)
Time period	in not in	Specify a time period or a time period to exclude. **in** - event time is within the time period. **not in** - event time is not within the time period. See Time period specification page for description of the format.
Maintenance status	in not in	Specify a host in maintenance or not in maintenance. **in** - host is in maintenance mode. **not in** - host is not in maintenance mode. *Note*: If several hosts are involved in the trigger expression, the condition matches if at least one of the hosts is/is not in maintenance mode.

Starting with Zabbix 2.0.6 if any object (host, template, trigger, etc) that is used in the action condition is deleted, the condition is deleted and the action is disabled to avoid incorrect execution of the action. The action can be re-enabled by the user.

Before 2.0.6 the missing object is displayed as *unknown* and the condition remains in place.

Trigger value:

- if a trigger changes status from OK to PROBLEM, trigger value is PROBLEM
- if a trigger changes status from PROBLEM to OK, trigger value is OK

When a new action for triggers is created, it gets two automatic conditions (both can be removed by the user):

- *"Trigger value = PROBLEM"* - so that only problem notifications are sent. This means that if you configure an action without setting any more specific conditions, messages will be sent for all problems. This makes the recovery message checkbox operate in a more intuitive way.
- *"Maintenance status = not in maintenance"* - so that notifications are not sent for hosts in maintenance.

The following conditions can be set for discovery-based events:

Condition type	Supported operators	Description
Host IP	= <>	Specify an IP address range or a range to exclude for a discovered host. = - host IP is in the range. <> - host IP is not in the range.
Service type	= <>	Specify a service type of a discovered service or a service type to exclude. = - matches the discovered service. <> - does not match the discovered service.
Service port	= <>	Specify a TCP port range of a discovered service or a range to exclude. = - service port is in the range. <> - service port is not in the range.
Discovery rule	=	Specify a discovery rule or a discovery rule to exclude. = - using this discovery rule.

	<>	<> - using any other discovery rule, except this one.
Discovery check	= <>	Specify a discovery check or a discovery check to exclude. = - using this discovery check. <> - using any other discovery check, except this one.
Discovery object	=	Specify the discovered object. = - equal to discovered object (a device or a service).
Discovery status	=	**Up** - matches 'Host Up' and 'Service Up' events **Down** - matches 'Host Down' and 'Service Down' events **Discovered** - matches 'Host Discovered' and 'Service Discovered' events **Lost** - matches 'Host Lost' and 'Service Lost' events
Uptime/Downtime	>= <=	Uptime for 'Host Up' and 'Service Up' events. Downtime for 'Host Down' and 'Service Down' events. >= - is more or equal to. Parameter is given in seconds. <= - is less or equal to. Parameter is given in seconds.
Received value	= <> >= <= like not like	Specify the value received from an agent (Zabbix, SNMP). String comparison. = - equal to the value. <> - not equal to the value. >= - more or equal to the value. <= - less or equal to the value. **like** - contains the substring. Parameter is given as a string. **not like** - does not contain the substring. Parameter is given as a string.
Proxy	= <>	Specify a proxy or a proxy to exclude. = - using this proxy. <> - using any other proxy except this one.

Type of calculation

The following options of calculating conditions are available:

- AND - all conditions must be met
- OR - enough if one condition is met
- AND/OR - combination of the two: AND with different condition types and OR with the same condition type, for example:

Host group = Oracle servers
Host group = MySQL servers
Trigger name like 'Database is down'
Trigger name like 'Database is unavailable'

is evaluated as

(Host group = Oracle servers **or** Host group = MySQL servers**) and (**Trigger name like 'Database is down' **or** Trigger name like 'Database is unavailable'**)**

2.0/manual/config/notifications/action/conditions.txt · Last modified: 2013/05/02 13:27 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 Escalations

Overview

With escalations you can create custom scenarios for sending notifications or executing remote commands.

In practical terms it means that:

- Users can be informed about new problems immediately
- Notifications can be repeated until the problem is resolved
- Sending a notification can be delayed
- Notifications can be escalated to another “higher” user group
- Remote commands can be executed immediately or when a problem is not resolved for a lengthy period
- Recovery messages can be sent

Actions are escalated based on the **escalation step**. Each step has a duration in time.

You can define both the default duration and a custom duration of an individual step. The minimum duration of one escalation step is 60 seconds.

You can start actions, such as sending notifications or executing commands, from any step. Step one is for immediate actions. If you want to delay an action, you can assign it to a later step. For each step, several actions can be defined.

The number of escalation steps is not limited.

Escalations are defined when configuring an operation.

If different escalations follow in close succession and overlap, the execution of each new escalation supersedes the previous escalation, but for at least one escalation step that is always executed on the previous escalation. This behavior is relevant in actions upon events that are created with EVERY problem evaluation of the trigger.

If an action is disabled during an escalation in progress (like a message being sent), the message in progress will be sent and then one more message on the escalation will be sent. The follow-up message will have the following text at the beginning of the message body: *NOTE: Escalation cancelled: action '<Action name>' disabled*. This way the recipient is informed that the escalation is cancelled and no more steps will be executed.

Escalation examples

Example 1

Sending a repeated notification once every 30 minutes (5 times in total) to a 'MySQL administrators' group. To configure:

- Set the *Default escalation period* to '1800' seconds (30 minutes) in general action attributes
- in Operations tab, set the escalation steps to be *From* '1' *To* '5'
- Select the 'MySQL administrators' group as the recipients of message

Notifications will be sent at 0:00, 0:30, 1:00, 1:30, 2:00 hours after the problem starts (unless, of course, the problem is resolved sooner).

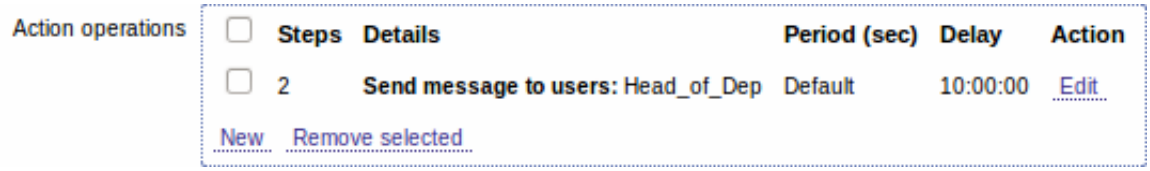
If the problem is resolved and a recovery message is configured, it will be sent to those who received at least one problem message within this escalation scenario.

If the trigger that generated an active escalation is disabled, Zabbix sends an informative message about it to all those that have already received notifications.

Example 2

Sending a delayed notification about a long-standing problem. To configure:

- Set the *Default escalation period* to '36000' seconds (10 hours) in general action attributes
- In Operations tab, set the escalation steps to be *From* '2' *To* '2'



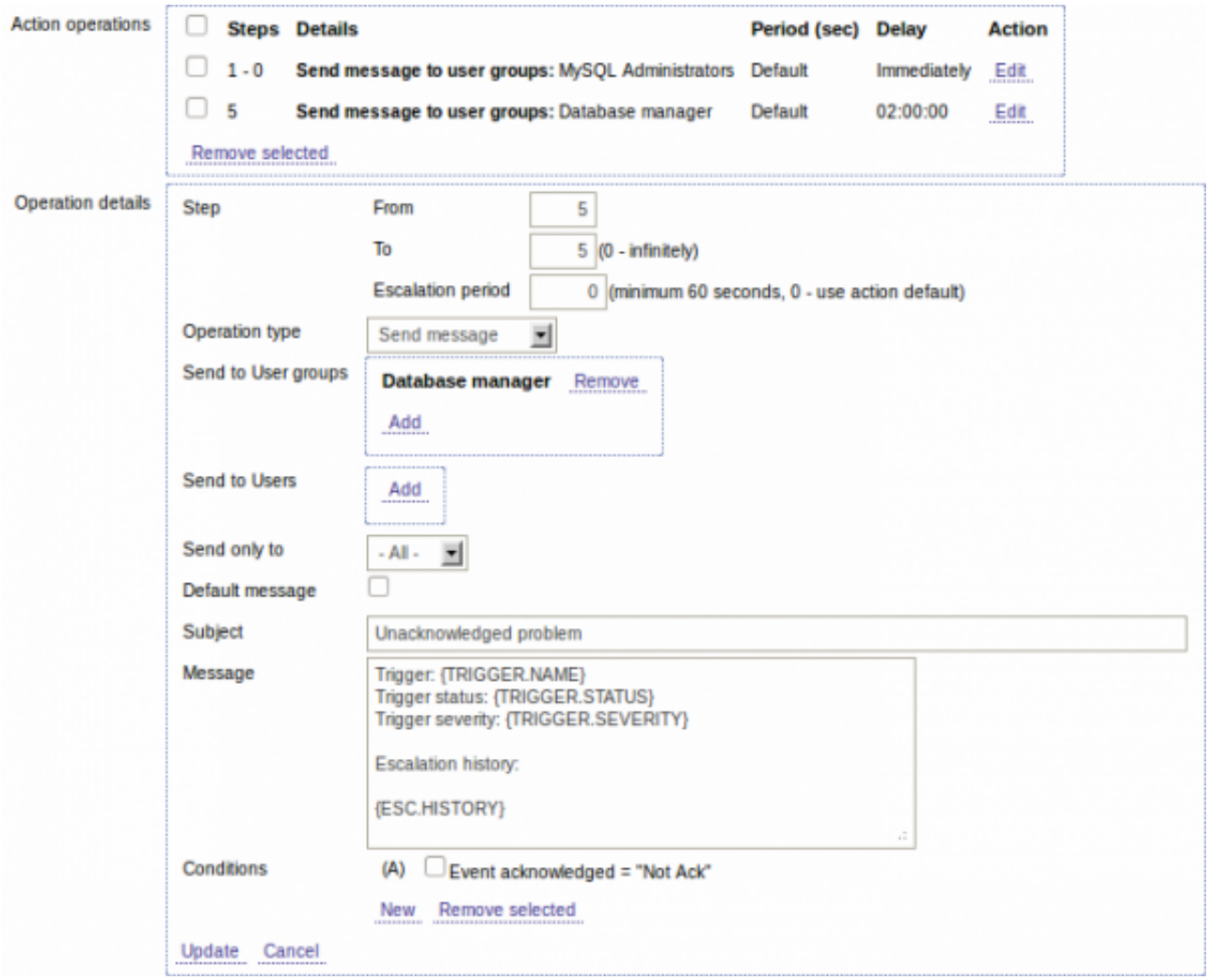
A notification will only be sent at Step 2 of the escalation scenario, or 10 hours after the problem starts.

You can customize the message text to something like 'The problem is more than 10 hours old'.

Example 3

Escalating the problem to the Boss.

In the first example above we configured periodical sending of messages to MySQL administrators. In this case, the administrators will get four messages before the problem will be escalated to the Database manager. Note that the manager will get a message only in case the problem is not acknowledged yet, supposedly no one is working on it.



Note the use of {ESC.HISTORY} macro in the message. The macro will contain information about all previously executed steps on this escalation, such as notifications sent and commands executed.

Example 4

A more complex scenario. After multiple messages to MySQL administrators and escalation to the manager, Zabbix will try to restart the MySQL database. It will happen if the problem exists for 2:30 hours and it hasn't been acknowledged.

If the problem still exists, after another 30 minutes Zabbix will send a message to all guest users.

If this does not help, after another hour Zabbix will reboot server with the MySQL database (second remote command) using IPMI commands.

Action operations

Steps	Details	Period (sec)	Delay	Action
1 - 0	**Send message to user groups:** MySQL Administrators	Default	Immediately	Edit
5	**Send message to user groups:** Database manager	Default	02:00:00	Edit
6	**Run remote commands on current host**	Default	02:30:00	Edit
7	**Send message to user groups:** Guests	Default	03:00:00	Edit
9	**Run remote commands on current host**	Default	04:00:00	Edit

New Remove selected

Example 5

An escalation with several operations assigned to one step and custom intervals used. The default escalation period is 30 minutes.

Action operations

Steps	Details	Period (sec)	Delay	Action
1 - 4	**Send message to user groups:** MySQL Administrators	Default	Immediately	Edit
5 - 8	**Send message to user groups:** Zabbix administrators	600	02:00:00	Edit
5 - 6	**Send message to user groups:** Database manager	3600	02:00:00	Edit
11	**Send message to user groups:** Guests	Default	04:00:00	Edit

New Remove selected

Notifications will be sent as follows:

- to MySQL administrators at 0:00, 0:30, 1:00, 1:30 after the problem starts
- to Zabbix administrators at 2:00, 2:10, 2:20, 2:30 after the problem starts (the custom step duration of 600 seconds setting in)
- to Database manager at 2:00 and 2:10 (the shorter custom step duration above of 600 seconds overriding the longer custom step of 3600 seconds tried to set here)
- to guest users at 4:00 hours after the problem start (the default interval of 30 minutes returning between the step 8 and 11)

2.0/manual/config/notifications/action/escalations.txt · Last modified: 2013/06/26 10:47 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

8 Macros

Overview

Zabbix supports a number of macros which may be used in various situations. Effective use of macros allows to save time and make Zabbix configuration more transparent.

See a full list of supported macros.

You can also configure your own user macros.

2.0/manual/config/macros.txt · Last modified: 2011/10/17 15:04 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 User macros

Overview

For greater flexibility, Zabbix supports user macros, which can be defined on global, template and host level. These macros have a special syntax: **{$MACRO}**.

The macros can be used in:

- item keys and descriptions
- trigger names and expressions (see examples 2 and 3)
- several other locations

The following characters are allowed in the macro names: **A-Z** , **0-9** , **_** , **.**

Zabbix substitutes macros according to the following precedence:

1. host level macros (checked first)
2. macros defined for first level templates of the host (i.e., templates linked directly to the host), sorted by template ID
3. macros defined for second level templates of the host, sorted by template ID
4. macros defined for third level templates of the host, sorted by template ID
5. ...
6. global macros (checked last)

In other words, if a macro does not exist for a host, Zabbix will try to find it in the host templates of increasing depth. If still not found, a global macro will be used, if exists.

If Zabbix is unable to find a macro, the macro will not be substituted.

To define user macros, go to the corresponding locations in the frontend:

- for global macros, visit *Administration → General → Macros*
- for host and template level macros, open host or template properties and look for the *Macros* tab

If a user macro is used in items or triggers in a template, it is suggested to add that macro to the template even if it is defined on a global level. That way, exporting the template to XML and importing it in another system will still allow it to work as expected.

Most common use cases of global and host macros:

1. taking advantage of templates with host specific attributes: passwords, port numbers, file names, regular expressions, etc
2. global macros for global one-click configuration changes and fine tuning

Examples

Example 1

Use of host-level macro in the “Status of SSH daemon” item key:

net.tcp.service[ssh,{$SSH_PORT}]

This item can be assigned to multiple hosts, providing that the value of **{$SSH_PORT}** is defined on those hosts.

Example 2

Use of host-level macro in the “CPU load is too high” trigger:

{ca_001:system.cpu.load[,avg1].last(0)}>{$MAX_CPULOAD}

Such a trigger would be created on the template, not edited in individual hosts.

If you want to use amount of values as the function parameter (for example, **max(#3)**), include hash mark in the macro definition like this: SOME_PERIOD ⇒ #3

Example 3

Use of two macros in the “CPU load is too high” trigger:

{ca_001:system.cpu.load[,avg1].min({$CPULOAD_PERIOD})}>{$MAX_CPULOAD}

Note that a macro can be used as a parameter of trigger function, in this example function **min()**.

User macros in a trigger expression will be expanded if referencing a parameter or a constant. They are NOT supported for referencing the host name, item key, function or operator.

2.0/manual/config/macros/usermacros.txt · Last modified: 2012/09/07 16:21 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

9 Users and user groups

Overview

All users in Zabbix access the Zabbix application through the web-based front end. Each user is assigned a unique login name and a password.

All user passwords are encrypted and stored in the Zabbix database. Users cannot use their user id and password to log directly into the UNIX server unless they have also been set up accordingly to UNIX. Communication between the web server and the user browser can be protected using SSL.

With a flexible user permission schema you can restrict and differentiate access to:

- administrative Zabbix frontend functions
- monitored hosts in hostgroups

The initial Zabbix installation has two predefined users - 'Admin' and 'guest'. The 'guest' user is used for unauthenticated users. Before you log in as 'Admin', you are 'guest'. Proceed to configuring a user in Zabbix.

2.0/manual/config/users_and_usergroups.txt · Last modified: 2011/12/01 14:54 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

1 Configuring a user

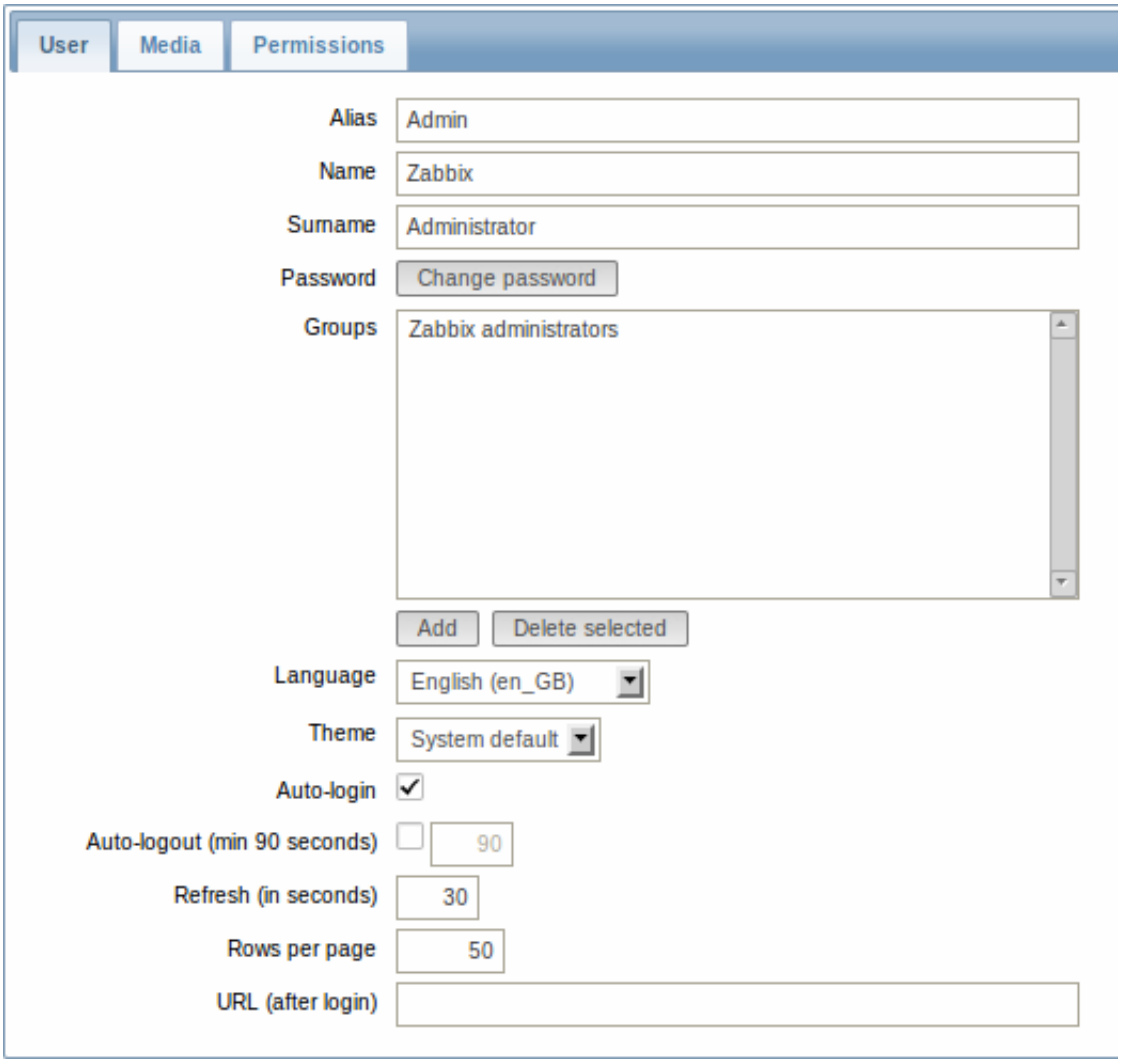
Overview

To configure a user:

- Go to *Administration* → *Users*
- Select *Users* from the dropdown to the right
- Click on *Create user* (or on the user name to edit an existing user)
- Edit user attributes in the form

General attributes

The *User* tab contains general user attributes:



Parameter	Description
Alias	Unique username, used as the login name.

Name	User first name (required).
Surname	User second name (required).
Password	Two fields for entering the user password. With an existing password, contains a *Password* button, clicking on which opens the password fields.
Groups	List of all user groups the user belongs to. Adherence to user groups determines what host groups and hosts the user will have access to. Click on *Add* to add groups.
Language	Language of Zabbix GUI.
Theme	Defines how the GUI looks like: **System Default** - use default system settings **Original Blue** - standard blue theme **Black & Blue** - alternative theme **Dark orange** - alternative theme
Auto-login	Enable if you want Zabbix to remember you and log you in automatically for 30 days. Browser cookies are used for this.
Auto-logout (min 90 seconds)	Mark the checkbox to enable automatic user logout after the set seconds of inactivity (minimum value = 90 seconds).
Refresh (in seconds)	Set the refresh rate used for graphs, screens, plain text data, etc. Can be set to 0 to disable.
Rows per page	You can determine how many rows per page will be displayed in lists.
URL (after login)	You can make Zabbix to transfer you to a specific URL after successful login, for example, the status of triggers page.

User media

The *Media* tab contains a listing of all media defined for the user. Media are used for sending notifications. Click on *Add* to assign media to the user.

See the Media types section for details on configuring media types.

Permissions

The *Permissions* tab contains information on:

- the user type (Zabbix User, Zabbix Admin, Zabbix Super Admin). Users cannot change their own type.
- host groups and hosts the user has access to. 'Zabbix User' and 'Zabbix Admin' users do not have access to any host groups and hosts by default. To get access they need to be included in user groups that have access to respective host groups and hosts.

See the User permissions page for details.

2.0/manual/config/users_and_usergroups/user.txt · Last modified: 2013/08/21 10:21 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

2 Permissions

Overview

You can differentiate user permissions in Zabbix by defining the respective user type and then by including the unprivileged users in user groups that have access to host group data.

User type

The user type defines the level of access to administrative menus and the default access to host group data.

User type	Description
Zabbix User	The user has access to the Monitoring menu. The user has no access to any resources by default. Any permissions to host groups must be explicitly assigned.
Zabbix Admin	The user has access to the Monitoring and Configuration menus. The user has no access to any host groups by default. Any permissions to host groups must be explicitly given.
Zabbix Super Admin	The user has access to everything: Monitoring, Configuration and Administration menus. The user has a read-write access to all host groups. Permissions cannot be revoked by denying access to specific host groups.

Permissions to host groups

Access to any host data in Zabbix are granted to user groups on host group level only.

That means that an individual user cannot be directly granted access to a host (or host group). It can only be granted access to a host by being part of a user group that is granted access to the host group that contains the host.

2.0/manual/config/users_and_usergroups/permissions.txt · Last modified: 2013/04/26 15:30 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]

3 User groups

Overview

User groups allow to group users both for organizational purposes and for assigning permissions to data. Permissions to monitoring data of host groups are assigned to user groups, not individual users.

It may often make sense to separate what information is available for one group of users and what - for another. This can be accomplished by grouping users and then assigning varied permissions to host groups.

A user can belong to any amount of groups.

Configuration

To configure a user group:

- Go to *Administration → Users*
- Select *User groups* from the dropdown to the right
- Click on *Create group* (or on the group name to edit an existing group)
- Edit group attributes in the form

The *User group* tab contains general group attributes:

Parameter	Description
Group name	Unique group name.
Users	The **In group** block contains a listing of the members of this group. To add users to the group select them in the *Other groups* block and click on **«**.
Frontend access	How the users of the group are authenticated. **System default** - use default authentication **Internal** - use Zabbix authentication. Ignored if HTTP authentication is set **Disabled** - access to Zabbix GUI is forbidden
Users status	Status of group members: **Enabled** - users are active **Disabled** - users are disabled
Debug mode	Mark this checkbox to activate debug mode for the users.

The *Permissions* tab allows you to specify user group access to host group (and thereby host) data:

Composing permissions	Click on *Add* beneath the respective list to specify the host groups that the user group will have access to on the level of: **Read-write** - read-write access to a host group **Read only** - read-only access to a host group **Deny** - access to a host group denied
Calculated permissions	Depending on the permissions set above, *Calculated permissions* will display all host groups and all hosts that the user group has access to on the level of: **Read-write** - host groups with read-write access **Read only** - host groups with read-only access **Deny** - host groups with access denied

2.0/manual/config/users_and_usergroups/usergroup.txt · Last modified: 2013/04/26 15:30 by martins-v
Except where otherwise noted, content on this wiki is licensed under the following license:CC Attribution-

Noncommercial-Share Alike 3.0 Unported [http://creativecommons.org/licenses/by-nc-sa/3.0/]