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>
```

Functions

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

A complete list of supported functions is available.

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 in no more than the latest 600 seconds
sum(#5)	Sum of all values in no more than 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*.

Several 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.

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.

Numbers with a '+' sign are not supported.

Operators

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

PRIORITY	OPERATOR	DEFINITION
1	**-**	Unary minus

PRIORITY	OPERATOR	DEFINITION
2	**not**	Logical NOT
3	*	Multiplication
	/	Division
4	**+**	Arithmetical plus
	-	Arithmetical minus
5	**<**	Less than. The operator is defined as: A<B ⇔ (A<B-0.000001) since Zabbix 3.0.17 A<B ⇔ (A≤B-0.000001) before Zabbix 3.0.17
	<=	Less than or equal to. The operator is defined as: A<=B ⇔ (A≤B+0.000001) since Zabbix 3.0.17 A<=B ⇔ (A<B+0.000001) before Zabbix 3.0.17
	>	More than. The operator is defined as: A>B ⇔ (A>B+0.000001) since Zabbix 3.0.17 A>B ⇔ (A≥B+0.000001) before Zabbix 3.0.17
	>=	More than or equal to. The operator is defined as: A>=B ⇔ (A≥B-0.000001) since Zabbix 3.0.17 A>=B ⇔ (A>B-0.000001) before Zabbix 3.0.17
6	**=**	Is equal. The operator is defined as: A=B ⇔ (A≥B-0.000001) and (A≤B+0.000001) since Zabbix 3.0.17 A=B ⇔ (A>B-0.000001) and (A<B+0.000001) before Zabbix 3.0.17
	<>	Not equal. The operator is defined as: A<>B ⇔ (A<B-0.000001) or (A>B+0.000001) since Zabbix 3.0.17 A<>B ⇔ (A≤B-0.000001) or (A≥B+0.000001) before Zabbix 3.0.17
7	**and**	Logical AND
8	**or**	Logical OR

not, **and** and **or** operators are case-sensitive and must be in lowercase. They also must be surrounded by spaces or parentheses.

All operators, except unary **-** and **not**, have left-to-right associativity. Unary **-** and **not** are non-associative (meaning **-(-1)** and **not (not 1)** should be used instead of **--1** and **not not 1**).

Evaluation result:

- **<**, **<=**, **>**, **>=**, **=**, **<>** operators shall yield '1' in the trigger expression if the specified relation is true and '0' if it is false;
- **and** shall yield '1' if both of its operands compare unequal to '0'; otherwise, it yields '0';
- **or** shall yield '1' if either of its operands compare unequal to '0'; otherwise, it yields '0';
- The result of the logical negation operator **not** is '0' if the value of its operand compares unequal to '0'; '1' if the value of its operand compares equal to '0'.

Value caching

Values required for trigger evaluation are cached by Zabbix server. Because of this trigger evaluation causes a higher database load for some time after the server restarts. The value cache is not cleared when item history values are removed (either manually or by housekeeper), so the server will use the cached values until they are older than the time periods defined in trigger functions or server is restarted.

Examples of triggers

Example 1

Processor load is too high on www.zabbix.com

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

```
{www.zabbix.com:system.cpu.load[all,avg1].last()}>5 or
{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()}=1
```

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

To make use of this trigger, 'tick' must be defined as a Zabbix trapper item. The host should periodically send data for this item using zabbix_sender. If no data is received within 180 seconds, the trigger value becomes PROBLEM.

Note that 'nodata' can be used for any item type.

Example 9

CPU activity at night time

Use of function time():

```
{zabbix:system.cpu.load[all,avg1].min(5m)}>2 and
{zabbix:system.cpu.load[all,avg1].time()}>000000 and
{zabbix:system.cpu.load[all,avg1].time()}<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 of the last hour tops the average load of the same hour yesterday more than two times.

Example 12

Using the value of another item to get a trigger threshold:

```
{Template PfSense:hrStorageFree[{#SNMPVALUE}].last()}<{Template
PfSense:hrStorageSize[{#SNMPVALUE}].last()}*0.1
```

The trigger will fire if the free storage drops below 10 percent.

Example 13

Using evaluation result to get the number of triggers over a threshold:

```
({server1:system.cpu.load[all,avg1].last()}>5) +
({server2:system.cpu.load[all,avg1].last()}>5) +
({server3:system.cpu.load[all,avg1].last()}>5)>=2
```

The trigger will fire if at least two of the triggers in the expression are over 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 and {server:temp.last()}>20) or
({TRIGGER.VALUE}=1 and {server:temp.last()}>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 and {server:vfs.fs.size[/,free].max(5m)}<10G) or
({TRIGGER.VALUE}=1 and {server:vfs.fs.size[/,free].min(10m)}<40G)
```

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