5 What's new in Zabbix 4.2.0

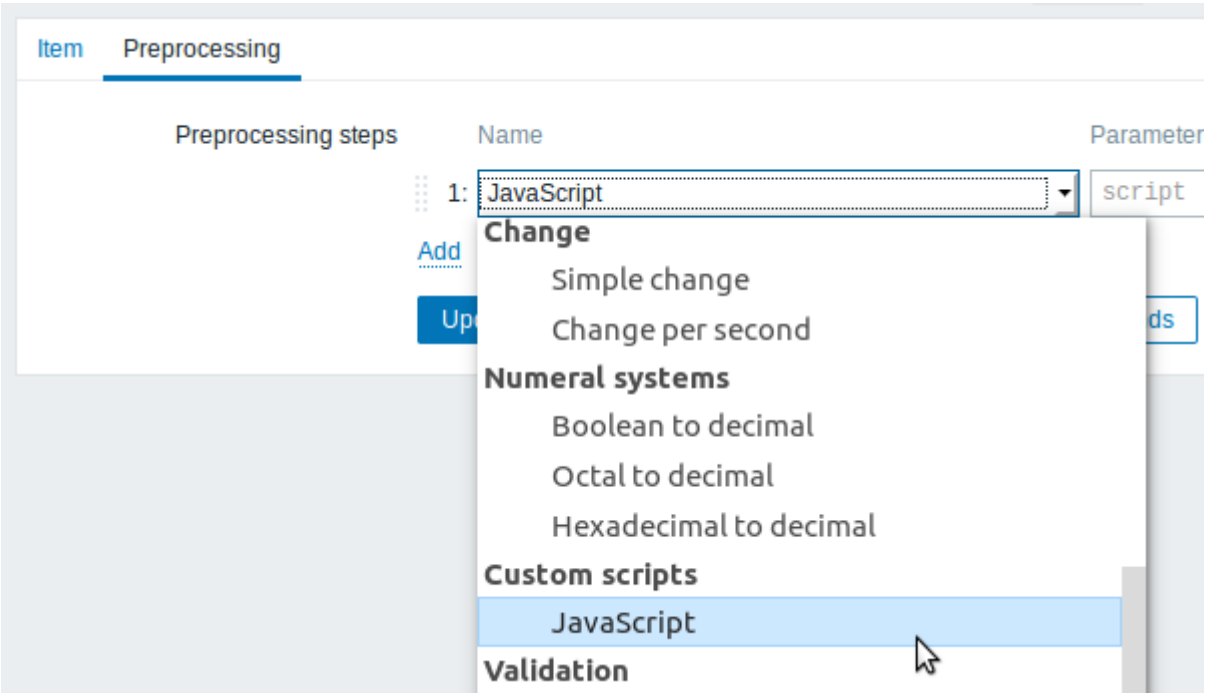
Zabbix 4.2.0 is not released yet.

This section introduces new and updated features of Zabbix 4.2.

Extended item value preprocessing

JavaScript

It is now possible to apply preprocessing to item value or low-level discovery values using JavaScript.



The JavaScript step accepts one parameter - a non-empty JavaScript code block.

Clicking in the parameter field or on the view/edit button next to the parameter field will open a window for entering the code block.

JavaScript

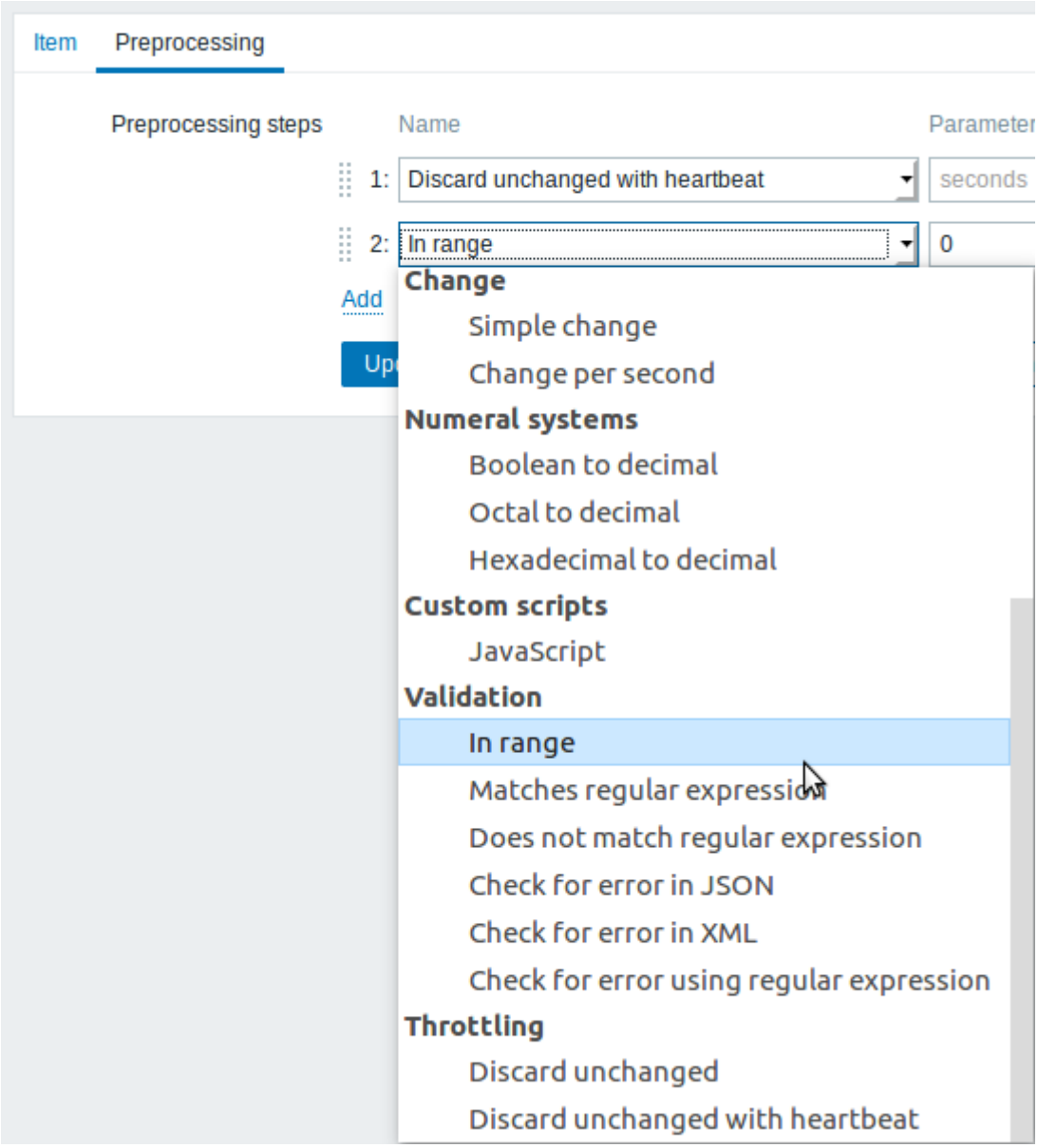
```
function (value) {
    var stats = {};
    var lines = value.split("\n");
    var i = 0;

    var reVersion = new RegExp("Server Version: ([^$]+)");
    var reUptime = new RegExp("Server uptime: ([0-9]+) days? ([0-9]+) hours? ([0-9]
    var reLoad = new RegExp("Server load: ([^ ]+) ([^ ]+) ([^ ]+)");
    var reHits = new RegExp("Total accesses: ([^ ]+) \- Total Traffic: ([^ ]+) ([^
    var reCpu = new RegExp("CPU Usage: u([^ ]+) s([^ ]+)");
    var lines_num = lines.length;

    for (i = 0; i < lines_num; i++)
    {
        var match = lines[i].match(reVersion);
        if (match)
        {
            stats.version = match[1];
            break;
        }
    }
```

Validation and throttling rules

New options have been added to item value preprocessing focusing on validation and throttling.

New validation options such as defining the acceptable range or some regular expression match allow to remove clearly incorrect or noise values which may happen when, for example, a counter is not initialized yet (empty or 'N/A') or a temperature sensor returns +999°C when it is off, the normal range being -100°C up to +100°C. Validation and custom error handling allows to discard these noise values or replace them with some default value or simply replace them with a human-readable error message.

Extracting an error message provided in incoming data is another new feature. For example, a JSON object may contain a field "error". If the field exists preprocessing is able to extract its value and set it as an error message for the item.



Throttling rules allow to process value only if it is changed, thus reducing the amount of incoming data when it's mostly static. For example, dependent items may be updated less frequently as the master item, provided there's no change to the values. If the master is updated every 10 seconds, a dependent item may be updated every minute.

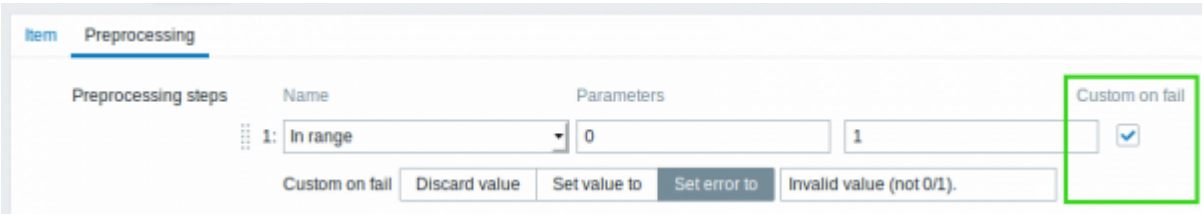
For more details, see: Item value preprocessing.

Custom error handling

Previously an item would become unsupported if any of the preprocessing steps failed. Now a custom error handling option has been added to the following preprocessing steps:

- Regular expression
- XML XPath
- JSON Path
- Custom multiplier
- Simple change
- Change per second

- Boolean to decimal
- Octal to decimal
- Hexadecimal to decimal
- In range (new validation option)
- Matches regular expression (new validation option)
- Does not match regular expression (new validation option)
- Prometheus pattern
- Prometheus to JSON



If you mark the *Custom on fail* checkbox, the item will not become unsupported in case of a failed preprocessing step and it is possible to specify a custom error handling option, e.g.:

- discard the value
- set to a specified value
- set a specified error message

Testing preprocessing steps

As new item preprocessing steps are being added the preprocessing pipeline has become more complex and with increased chance of mistakes. To help with getting it right Zabbix provides a way for the user to test the configured preprocessing steps.

Each step can be tested individually as well as all steps can be tested together using the new *Test* or *Test all steps* buttons respectively in the Actions block.

See also: Testing preprocessing steps.

Preprocessing support on Zabbix proxy

For hosts monitored by proxies, all item (including low-level discovery rules, dependent items) value preprocessing will now be done on the proxy.

Given new preprocessing options such as Javascript, extensive validation and throttling, preprocessing may become a server bottleneck thus preprocessing by proxies offers the necessary scalability. As this is not a configurable setting it is important to be aware of the performance implications - while preprocessing by proxies is beneficial for Zabbix server performance, as data from proxies will no longer go through the preprocessing manager, the affected proxies will require more resources than before.

The change affects the upgrade process to Zabbix 4.2. Please see the upgrade notes.

Extended error messages

Previously, in case of a preprocessing error, the item would become unsupported and the error message would report the number of the failed preprocessing step along with the error description.

```
Item preprocessing step #<N> failed: <error description>
```

This was not always enough as the real problem could lie in the step before the one that failed. Therefore, now the error message is extended with a log of the executed steps, reporting on the result or failure of each preprocessing step.

```
Preprocessing failed for: <value>

1. Result (<custom on fail action>): <value>
2. Result (<custom on fail action>): <value>
3. Failed: <error description>
```

Note that if the summary error message is larger than 2048 bytes, some of the initial steps may be replaced with '...'. If the header plus the last step error description exceeds 2048 character limit the error message will be truncated.

Low-level discovery

Separate processing for low-level discovery

Processing low-level discovery rules has been split from data gathering processes into its own processing.

See: Daemon improvements.

Expected format change in low-level discovery

In order to support preprocessing of low-level discovery result and custom paths to low-level

discovery (LLD) macro values in a JSON document, the format of JSON returned by low-level discovery rules has been changed. It is no longer expected that the JSON will contain the "data" object:

```
{
  "data": [
    {<discovery row>},
    {<discovery row>}
  ]
}
```

Low-level discovery will now accept a normal JSON containing an array.

Built-in discovery keys have been updated to return an array of LLD rows at the root of JSON document. Zabbix will automatically extract a macro and value if an array field uses the {#MACRO} syntax as a key. Any new native discovery checks will use the new syntax without the "data" elements. When processing a low-level discovery value first the root is located (array at `$.` or `$.data`).

While the "data" element has been removed from all native items related to discovery, for backward compatibility Zabbix will still accept the JSON notation with a "data" element, though its use is discouraged. If the JSON contains an object with only one "data" array element, then it will automatically extract the content of the element using JSONPath `$.data`.

Low-level discovery as dependent item

Low-level discovery rule now can also be a dependent item, depending on a regular item. Low-level discovery rule cannot be dependent on another low-level discovery rule.

To configure it as a dependent item, just select *Dependent item* for 'Type' and specify the master item in the 'Master item' field in discovery rule configuration.

Preprocessing and custom macros in low-level discovery

Low-level discovery configuration has gained additional options of preprocessing the discovery result and extracting custom macros from the preprocessed result. To better understand the changes in the discovery data flow, let's compare it with the previous versions:

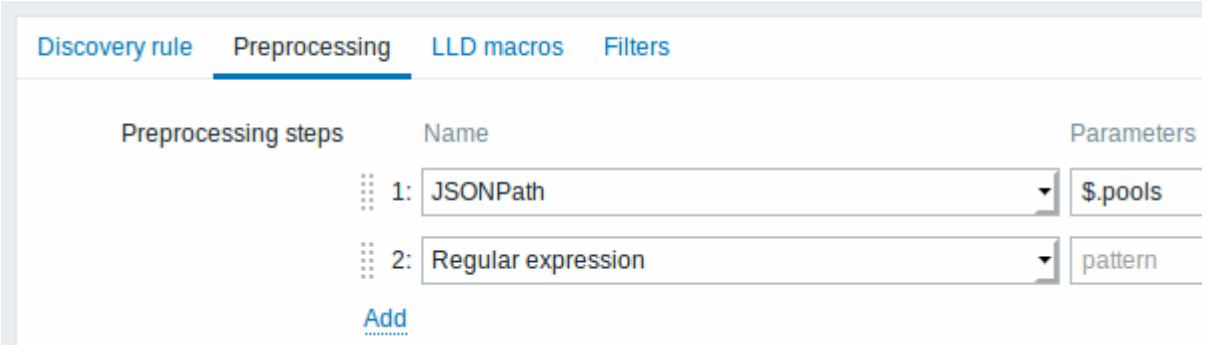
Version	**LLD data flow**
Before Zabbix 4.2	*Discovery rule → Discovery rule filter*
In Zabbix 4.2	*Discovery rule → Preprocessing → Custom macros → Discovery rule filter*

It's important to understand that this logic works in the order from left to right, the same as when defining the rule: first the discovery happens, then optionally preprocessing is applied to it, then custom macros may be extracted and then the filter conditions are applied to the result.

Preprocessing

Low-level discovery rules now have a preprocessing step, containing the following options:

- Regular expression match
- JSONPath (structured data)
- Javascript
- Does not match regular expression (validation)
- Check for error in JSON (validation)
- Discard unchanged with heartbeat (throttling)
- Prometheus to JSON

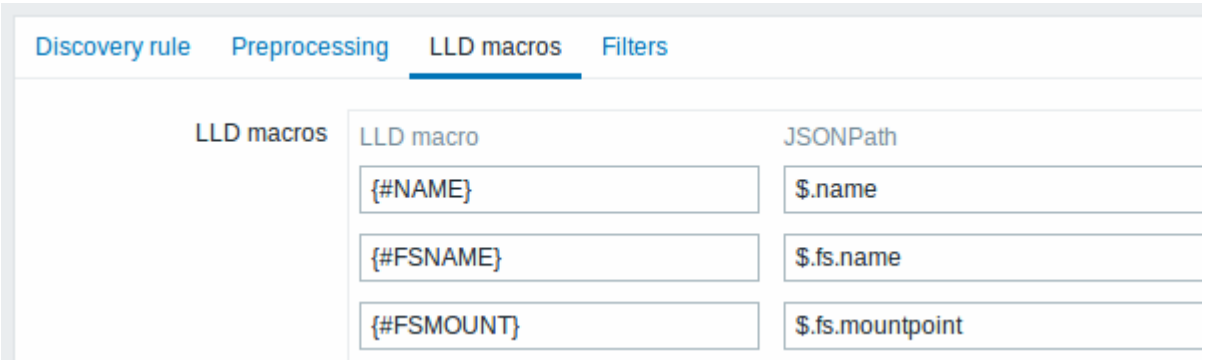


For more details, see: Preprocessing in low-level discovery rules.

LLD macro extraction with JSONPath

Low-level discovery rules now also accept extracting user-defined low-level discovery (LLD) macro values using a custom path specified in JSONPath syntax.

LLD macro = JSONPath mappings can now be defined where each macro is defined with a custom JSON path to the value location. The values pointed to by the defined JSON path are used to replace the macros in item, trigger, etc prototype fields.



For more details, see: Custom macros in low-level discovery rules.

TimescaleDB support

In the new version Zabbix adds support of a time-series database in the form of **experimental** support of TimescaleDB, a PostgreSQL-based database solution of automatically partitioning data into

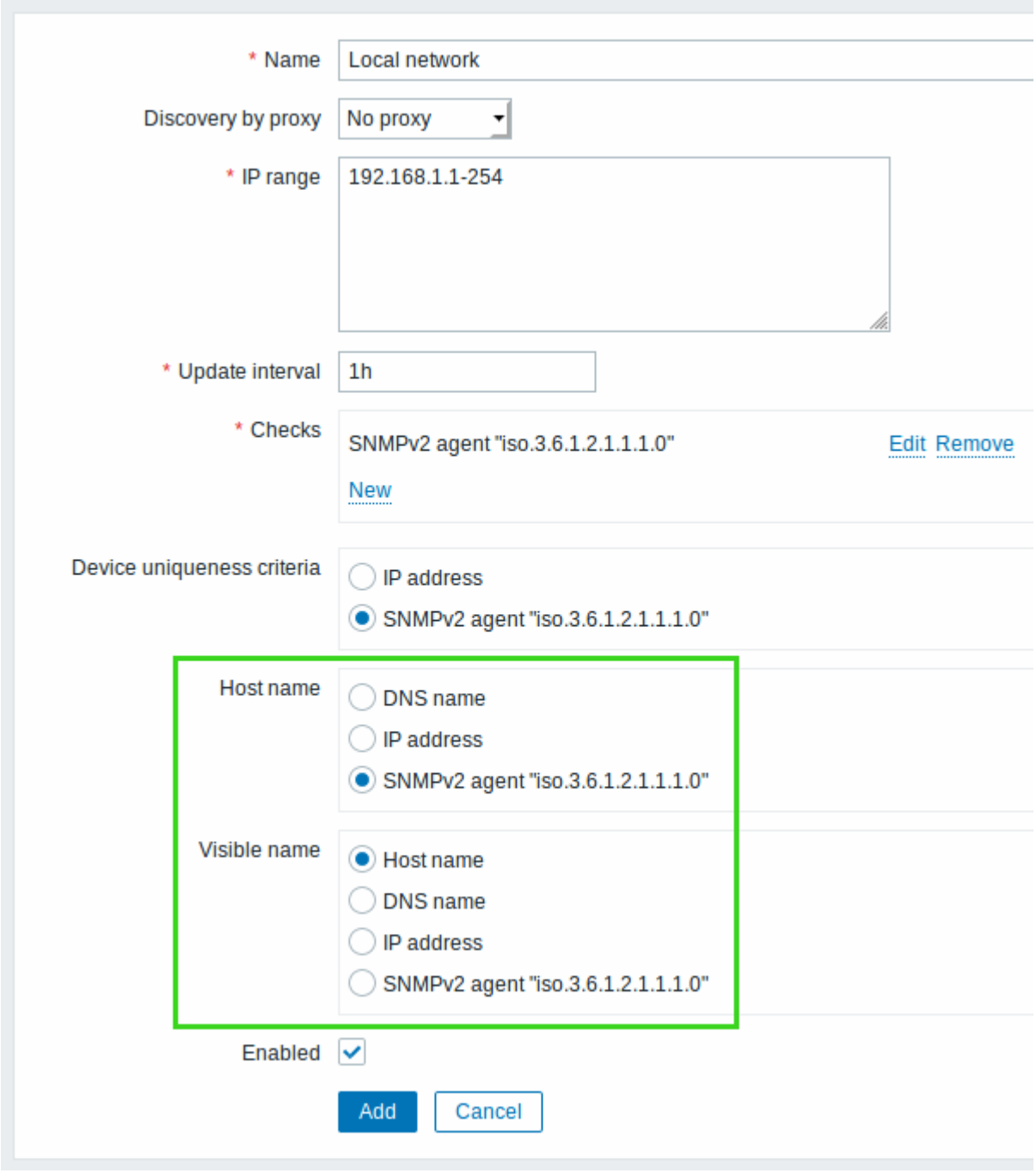
time-based chunks to support faster performance at scale.

A migration script is available to migrate from existing PostgreSQL tables to a TimescaleDB layout. For more details see: Migration to TimescaleDB.

Currently TimescaleDB is not supported by Zabbix proxy.

Using item value to name discovered hosts

Previously names of new hosts added as a result of network discovery were using either the DNS name or, in the absence of it, the IP address. In the new version it is possible to give much more descriptive naming to discovered hosts using discovery item values collected either from Zabbix agent or SNMP agent items:



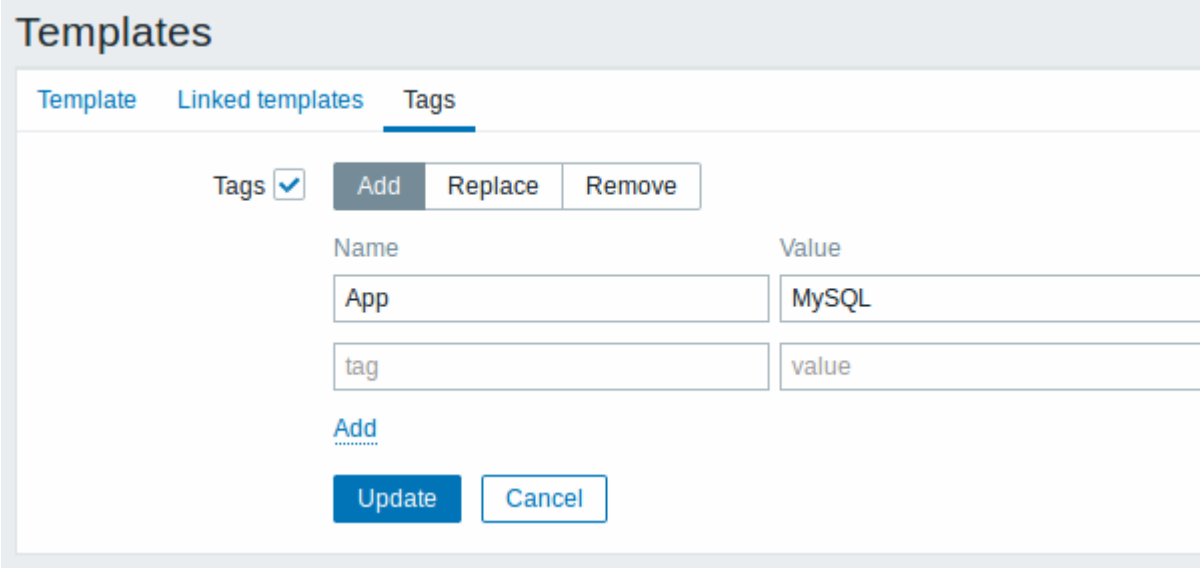
Zabbix agent or SNMP agent checks are listed as naming options in the new *Host name* and *Visible name* fields, in addition to the ability to specify DNS name or IP address as the name.

Template and host-level tags

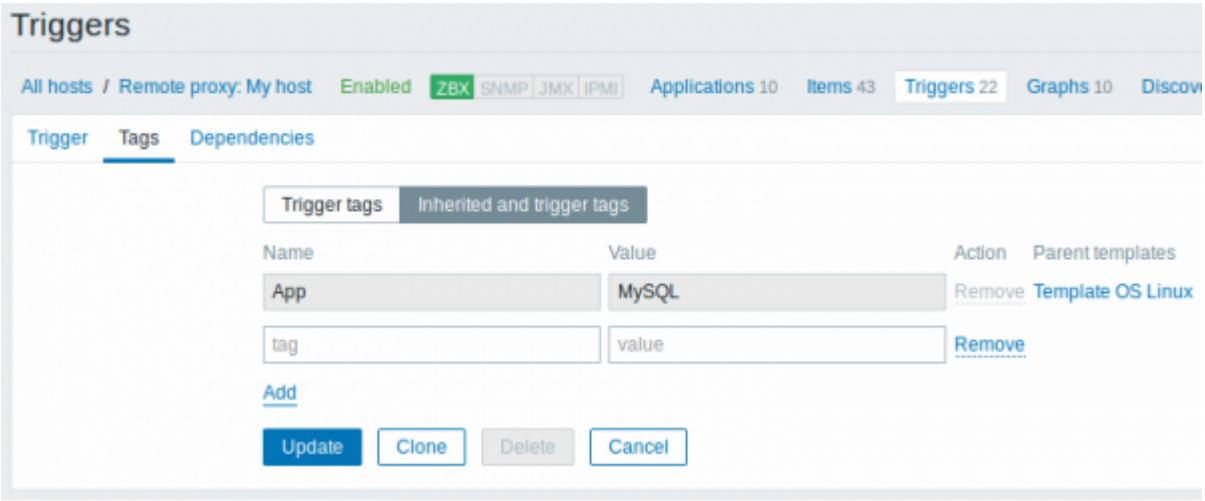
The possibility to tag events has been a feature of Zabbix since 3.2.0, however, previously, tagging was confined to individual triggers. In the new version tagging can be applied on template and host levels:

- On template level that means all problems of host triggers inherited from these templates will be tagged accordingly
- On host level that means that all host problems will be tagged accordingly

Tags can be defined in a new tab in template and host configuration forms, for example, for a template:



In trigger configuration, tags are now also separated into their own tab. In addition the *Inherited and trigger tags* option allows to view tags defined on template level (but not on host level), if the trigger comes from that template:



An event when it happens inherits all tags from the whole chain of templates, hosts, triggers.

Completely identical `tag:value` combinations are merged into one rather than being duplicated, when marking the event.

See more on tags.

Remote monitoring of Zabbix stats

It is now possible to make some internal metrics of Zabbix server and proxy accessible remotely by another Zabbix instance or a third party tool. This can be useful so that supporters/service providers can monitor their client Zabbix servers/proxies remotely or, in organizations where Zabbix is not the main monitoring tool, that Zabbix internal metrics can be monitored by a third party system in an umbrella-monitoring setup.

Zabbix internal stats are exposed to a configurable set of addresses listed in the new 'StatsAllowedIP' server/proxy parameter. Requests will be accepted only from these addresses.

To configure querying of internal stats on another Zabbix instance, you may use two new items:

- `zabbix[stats,<ip>,<port>]` internal item - for direct remote queries of Zabbix server/proxy. <ip> and <port> are used to identify the target instance.
- `zabbix.stats[<ip>,<port>]` agent item - for agent-based remote queries of Zabbix server/proxy. <ip> and <port> are used to identify the target instance.

To make sure that the target instance allows querying it by the external instance, list the address of the external instance in the 'StatsAllowedIP' parameter on the target instance.

These items gather statistics in bulk and return a JSON which can be used as the master item for dependent items that get their data from. A selected set of internal metrics (i.e. not all) is returned by either of these two items.

There are also another two new items allowing to specifically remotely query internal queue stats:

- `zabbix[stats,<ip>,<port>,queue,<from>,<to>]` internal item - for direct internal queue queries to remote Zabbix server/proxy
- `zabbix.stats[<ip>,<port>,queue,<from>,<to>]` agent item - for agent-based internal queue queries to remote Zabbix server/proxy

For more details, see:

- Remote monitoring of Zabbix stats
- Internal items
- Zabbix agent items

New templates

New templates are also available for remote Zabbix server or proxy internal metric monitoring:

- Template App Remote Zabbix server

- Template App Remote Zabbix proxy

Note that in order to use a template for remote monitoring of multiple external instances, a separate host is required for each external instance monitoring.

More intuitive dashboard editing

Several improvements have been made to the way widgets behave in dashboard editing mode:

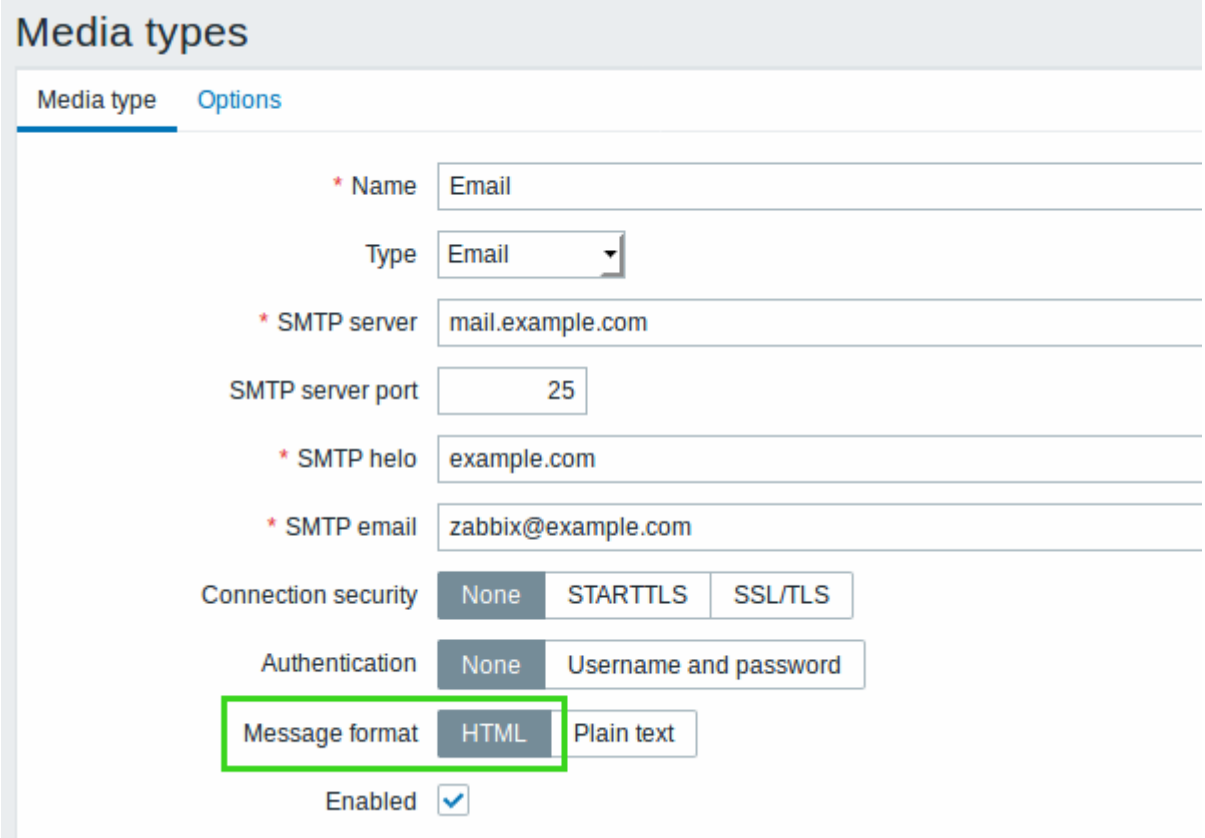
- When a widget is made larger, it no longer “throws down” other widgets. Instead the other widgets in any direction try to shrink accordingly using all the available space;
- When a widget is moved down, the widgets below it try to take its place if possible, instead of being dragged along downwards;
- When a widget is moved up, directly adjoining widgets below it are dragged up as well, instead of being left behind;
- If previously it was possible to add a widget only at the end and only in its default size, now it is now possible to add a widget in any open slot and size. See Adding widgets for details on the changes.

Animated GIF support in maps

It is now supported to upload animated GIF images to use in Zabbix maps. To support animated GIF upload, the required minimum GD library version is now 2.0.28.

HTML notifications

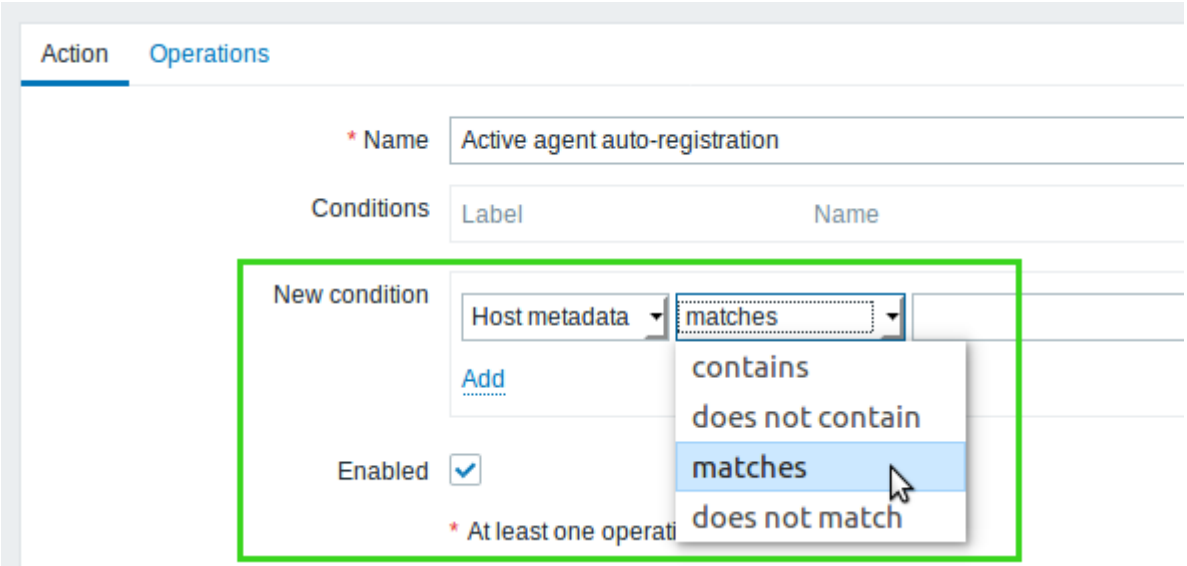
Email notifications can now be sent in HTML format. HTML format allows for more feature-rich emails containing font colouring, clickable links, interactive elements, company style, etc.



HTML format can be selected in the *Message format* option when configuring email as the media type in *Administration* → *Media types*.

Regular expression-based matching in auto-registration conditions

Regular expression match is now supported when configuring conditions for active agent auto-registration actions. It is supported if using *matches* and *does not match* operators for host name and host metadata conditions.



Note that *contains* and *does not contain* operators that were supported before perform only a

substring match.

Web monitoring

Content retrieval based on regular expressions also in headers

While previously it was possible to retrieve headers only, it was not possible to seek for matching content in those. Now it is possible to search headers as well for a regular expression match either in variables or the required string.

Additionally, a *Retrieve mode* option has been added to web monitoring steps, allowing to:

- Retrieve body only
- Retrieve headers only
- Retrieve both body and headers

Retrieve mode replaces the *Retrieve only headers* option. Thus it is now possible to seek matching content in body only, headers only or in both.

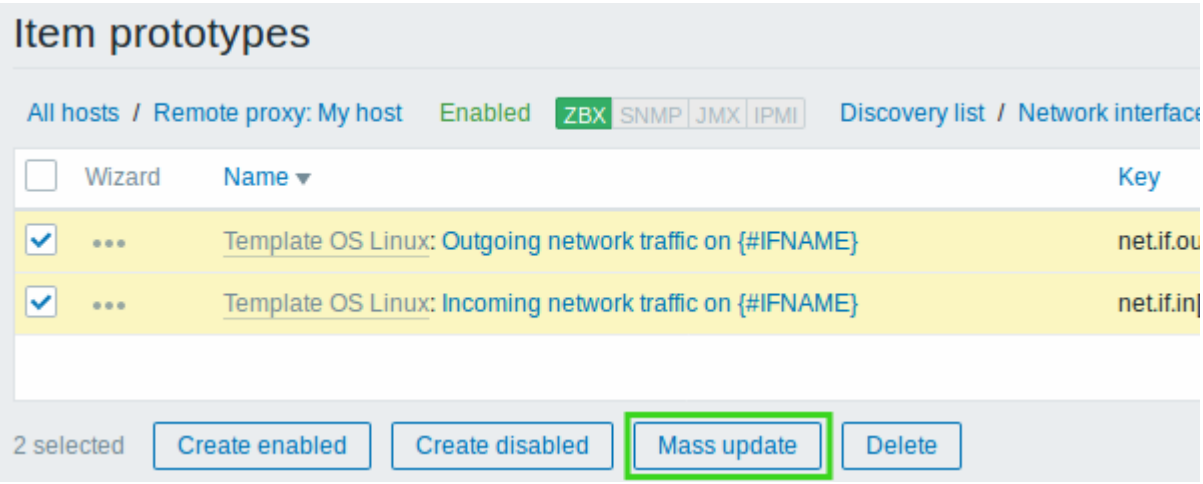
Frontend

This section introduces miscellaneous frontend improvements:

Mass update

Mass update for item prototypes

It is now possible to also mass update item prototypes used in low-level discovery rules. To update properties of several item prototypes at once, mark the checkboxes before the items and click on *Mass update*:



Then update the properties you need in the mass update form that is identical to that of regular items.

Mass update for templates

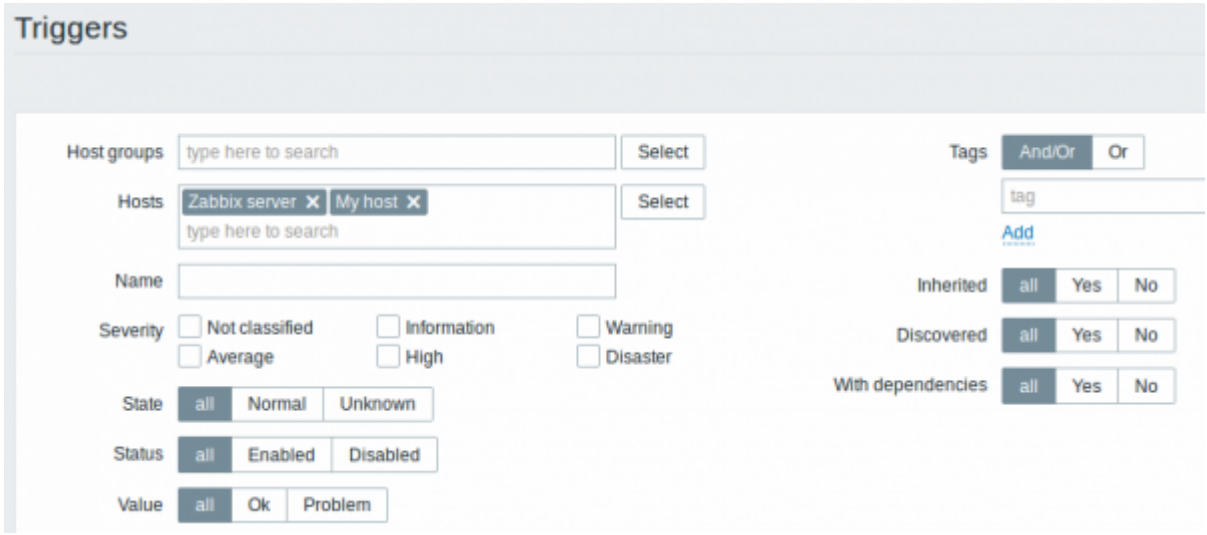
Mass update is now available in the list of templates.

Filtering

Trigger filtering

Several new options have been added to the filter in the trigger list. Importantly:

- it is now possible to specify multiple hosts and host groups in the filter allowing to view triggers of more than one host, greatly increasing the usability of such functions as mass update. Also, the dropdowns of host and host group selection have been removed as these selections now can be made in the filter;
- it is also possible to filter by several severities:



See trigger filter details for more information.

Item filtering

The possibility to specify multiple hosts and host groups in the filter, introduced in the trigger filter (see above), is also available in the item filter.

If no host or host group is specified in the item filter, now all items will be displayed instead of none and a message to select something.

See item filter details for more information.

Macros

New macros supported in maps

More macros are now supported in map element labels, URL names and URL values:

New supported location in maps	Macros			
	Host element	**Host group element**	**Trigger element**	**Map element**
Element label	{HOST.ID}, {INVENTORY.*} macros	{HOSTGROUP.ID}, {INVENTORY.*} macros	{TRIGGER.ID}, {INVENTORY.*} macros	{MAP.ID}, {MAP.NAME}
URL name	{HOST.ID}	{HOSTGROUP.ID}	{TRIGGER.ID}	{MAP.ID}
URL name and value	{HOST.CONN}, {HOST.DNS}, {HOST.HOST}, {HOST.IP},{HOST.NAME}, {INVENTORY.*} macros	{HOST.CONN}, {HOST.DNS}, {HOST.HOST}, {HOST.IP},{HOST.NAME}, {INVENTORY.*} macros	{HOST.CONN}, {HOST.DNS}, {HOST.HOST}, {HOST.ID},{HOST.IP},{HOST.NAME}, {INVENTORY.*} macros	{MAP.NAME}

Note that this table lists only the new locations. So, even though {HOST.ID}, {HOSTGROUP.ID}, {TRIGGER.ID} and {MAP.ID} are listed as now supported in map URL names, they were supported in the URL values already. Similarly, the many {HOST.*} macros were already supported in element labels.

See also: All supported macros.

Daemons

Separate processing for low-level discovery

Processing low-level discovery rules has been split from data gathering processes into its own processing, consisting of a configurable number of low-level discovery workers and a low-level discovery manager. This change greatly reduces low-level discovery impact on data gathering (poller, trapper processes) and proxy communications (trapper, proxypoller processes). Before data gathering could be delayed because the corresponding data gathering processes were busy with low-level discovery.

See also: StartLLDProcessors parameter

See also

- Template changes