ZABBIX 5.0

Certified Specialist Training

Day 2

COPYRIGHT NOTICE

Rules

It is prohibited to make any video and/or audio recordings during the whole period of this course.

This course is intended only for the officially enrolled student. Subject to the Copyright Notice below, the student is not allowed to share his credentials for attending this course, to allow others to join and take part, or otherwise make use of these Materials.

Copyright notice

AGENDA

Zabbix agent installation

Zabbix agent checks

Command line utilities

Monitoring Windows

Macros

Problem detection

Advanced problem detection

Templates

User parameters

Zabbix Agent installation

Zabbix agent is a process deployed on monitoring targets.

A native Zabbix agent:

- Is developed in the C language
- May run on various supported platforms (Linux, UNIX, macOS, Windows etc.)
- Collects data from a device or an application
- Has very low memory footprint and resource usage
- Can work in the passive and active modes
- Supports native communication encryption
- Uses a JSON based protocol to communicate with Zabbix server

https://www.zabbix.com/documentation/5.0/manual/concepts/agent

Zabbix agents starting from version 1.4 are compatible with:

Install Zabbix agent:

```
# dnf install zabbix-agent
```

Configure zabbix_agentd.conf

- Server=<IP/DNS>
- ServerActive=<IP/DNS>
- Hostname=<name of host>

Start Zabbix agent:

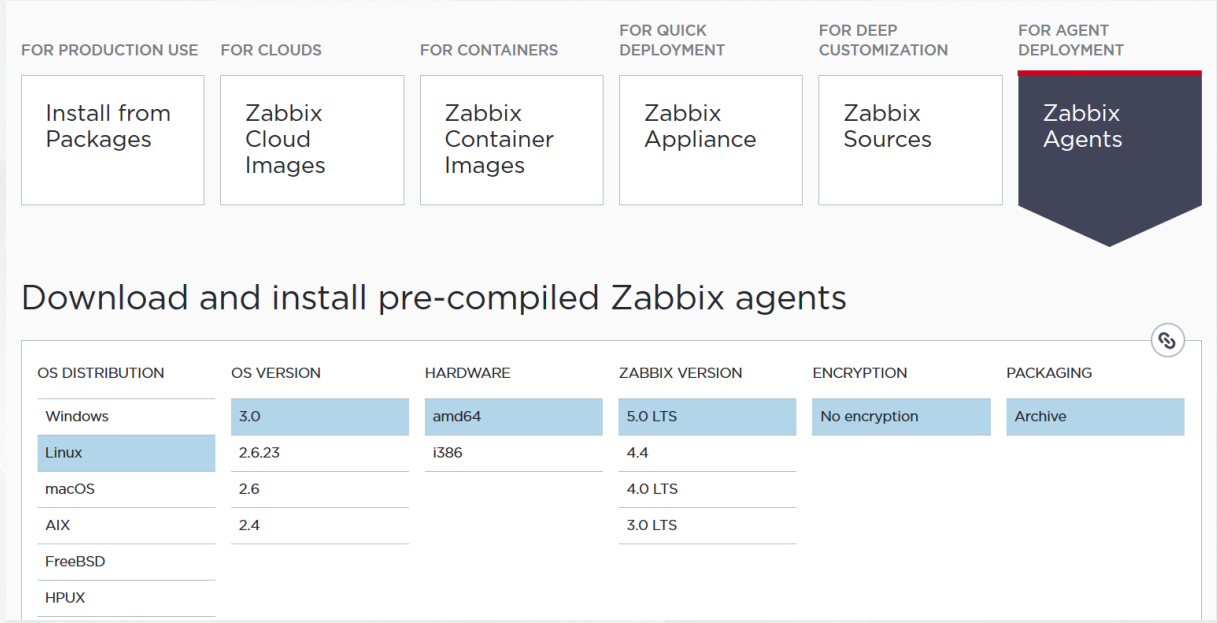
```
# systemctl start zabbix-agent
```

Enable autostart:

```
# systemctl enable zabbix-agent
```

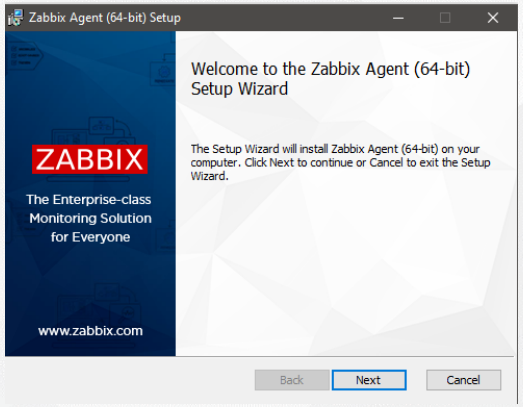
Often pre-compiled

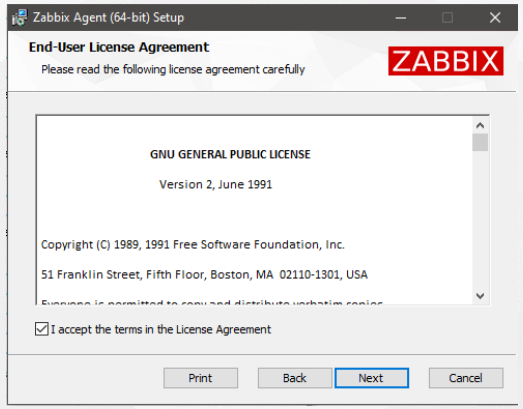
Configure and compile sources

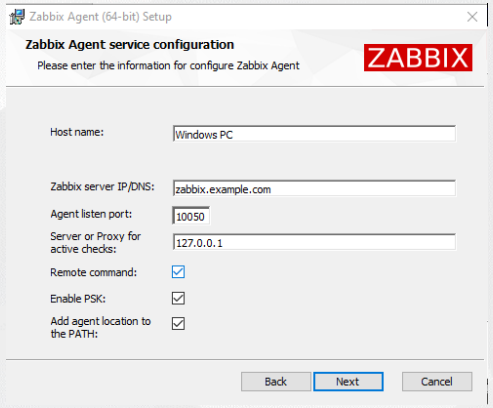


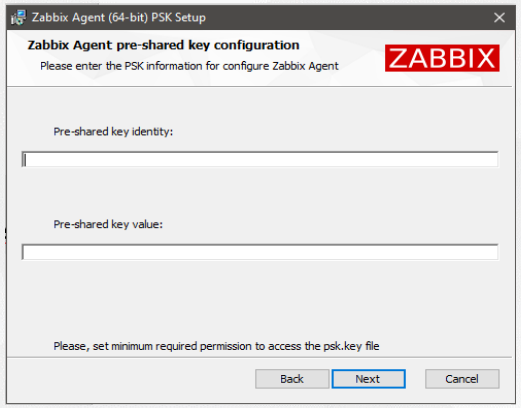
INSTALLING WINDOWS AGENT

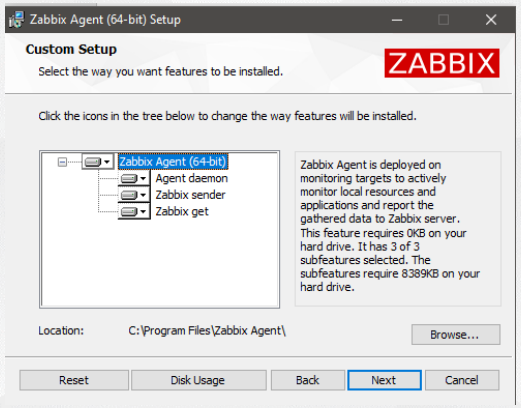
- Download an agent msi install package from zabbix.com
- Run the zabbix_agent-5.0.X-windows-amd64-openssl.msi
- Follow the steps of the setup wizard.











AGENT FOR MACOS

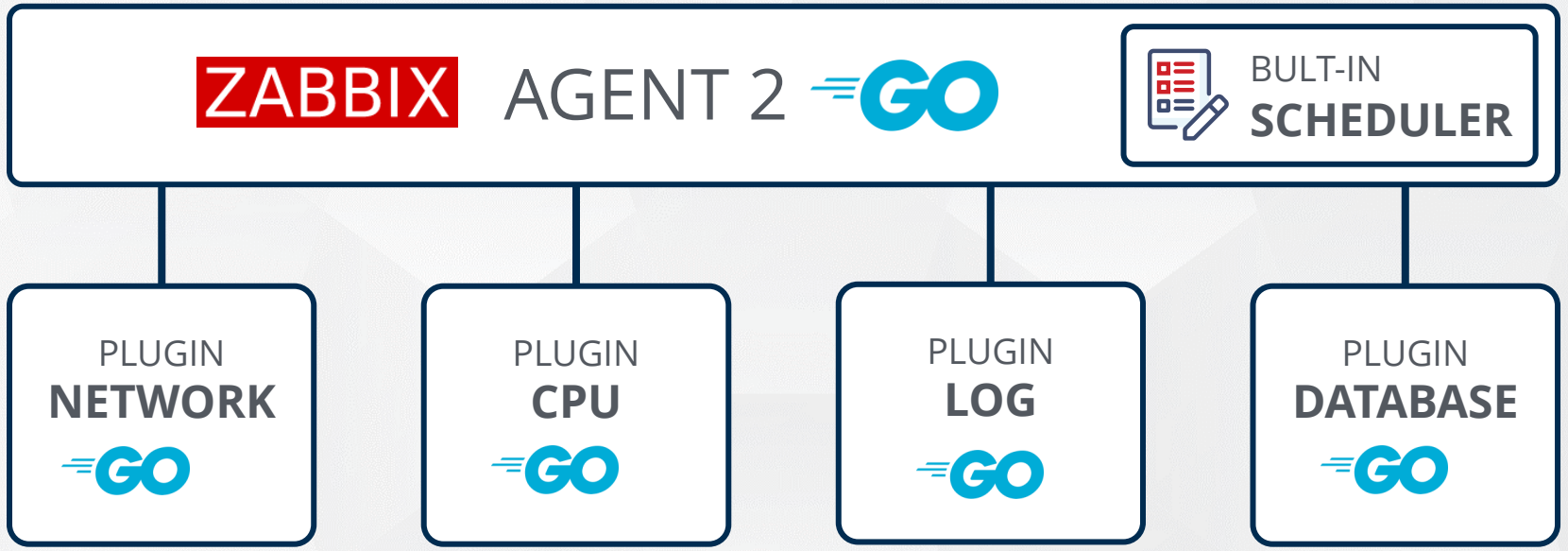
FOR PRODUCTION USE	FOR CLOUDS	FOR CONTAINERS	FOR QUICK DEPLOYMENT	FOR DEEP CUSTOMIZATION	FOR AGENT DEPLOYMENT
Install from Packages	Zabbix Cloud Images	Zabbix Container Images	Zabbix Appliance	Zabbix Sources	**Zabbix Agents**

Download and install pre-compiled Zabbix agents

OS DISTRIBUTION	OS VERSION	HARDWARE	ZABBIX VERSION	ENCRYPTION	PACKAGING
Windows	Any	amd64	5.0 LTS	GnuTLS	PKG
Linux		i386	4.4	OpenSSL	Archive
macOS			4.2	No encryption	

...version=5.0+LTS&release=5.0.0&os=macOS...

ZABBIX AGENT 2



scheduled/flexible intervals

written in Go

less TCP connections

older configuration file support

out-of-the-box systemd monitoring

easily extendable

Go version 1.13+ environment is required for building Zabbix agent 2 from sources

Zabbix agent 2 is written in Go (Golang).

- Drop-in replacement for Zabbix agent
  - Supports all previous functionality (same item keys)
  - Supports old configuration file format
- Agent 2 currently has limited systemd support
- Can be installed as a Windows service (since 5.0.4)
- SQLite can be used to buffer data
- Has been developed to
  - Reduce the number of TCP connections
  - Be easily extendible with plugins.
- Improved active checks
  - Active checks support scheduled/flexible intervals
  - Parallel execution of multiple active checks for each ServerActive

https://www.zabbix.com/documentation/5.0/manual/concepts/agent2

Q&A
Why do we need Zabbix agent on a Zabbix server?

PRACTICAL SETUP

1. Install Zabbix agent
2. Check status of the "zabbix_agentd" daemon
3. Login to frontend and find the host "Zabbix server"
4. Check Latest data section to make sure the metrics are collected
    1) Enable detailed
    2) Find Zabbix agent items
    3) Filter CPU related items only
    4) Select all load averages and display on a simple graph

Zabbix agent Passive checks

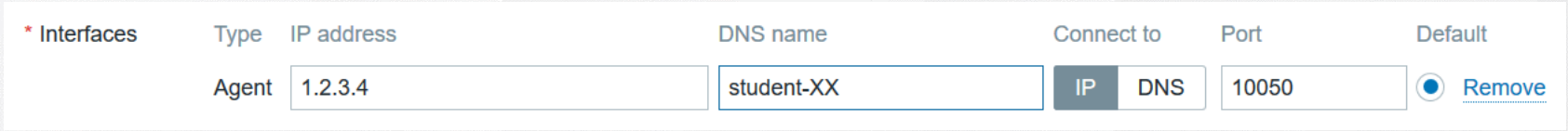
ZABBIX AGENT PASSIVE CHECKS: SETTINGS

A passive check is a simple data request.

- Zabbix server or proxy asks for some data (for example, CPU load) and Zabbix agent sends back the result to the server/proxy

Settings for host in the frontend:

- Configuration > Hosts > {host} > Interfaces IP/DNS and Port



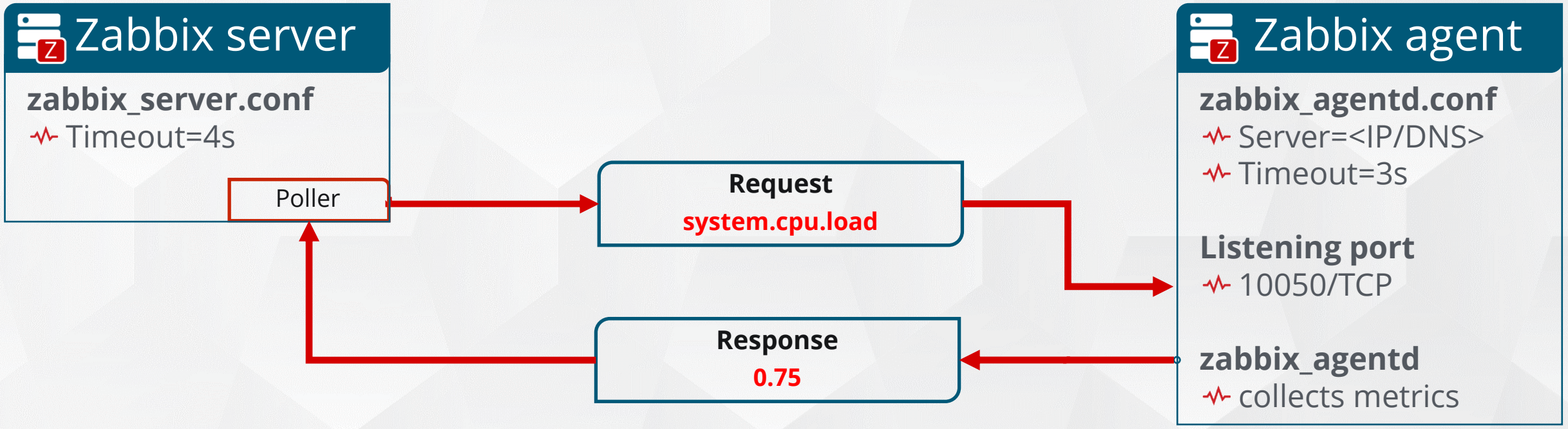
zabbix_agentd.conf

```
## Server=<IP/DNS>
Server=127.0.0.1,192.168.1.0/24,::1,2001:db8::/32,zabbix.example.com

## Optional: Default port (TCP/10050) can be changed in agent configuration file:
ListenPort=10050

## Number of pre-forked instances of zabbix_agentd that process passive checks.
## If set to 0, disables passive checks
StartAgents=3
```

Zabbix server connects to Zabbix agent and requests a metric.



- Zabbix Server timeout is a connection timeout
  - How long Zabbix server will wait for agent to give a response to request
- Zabbix Agent timeout is a service timeout
  - How long it may take for agent to complete a check

Both timeouts for server and agent must be adjusted for passive checks to work properly.

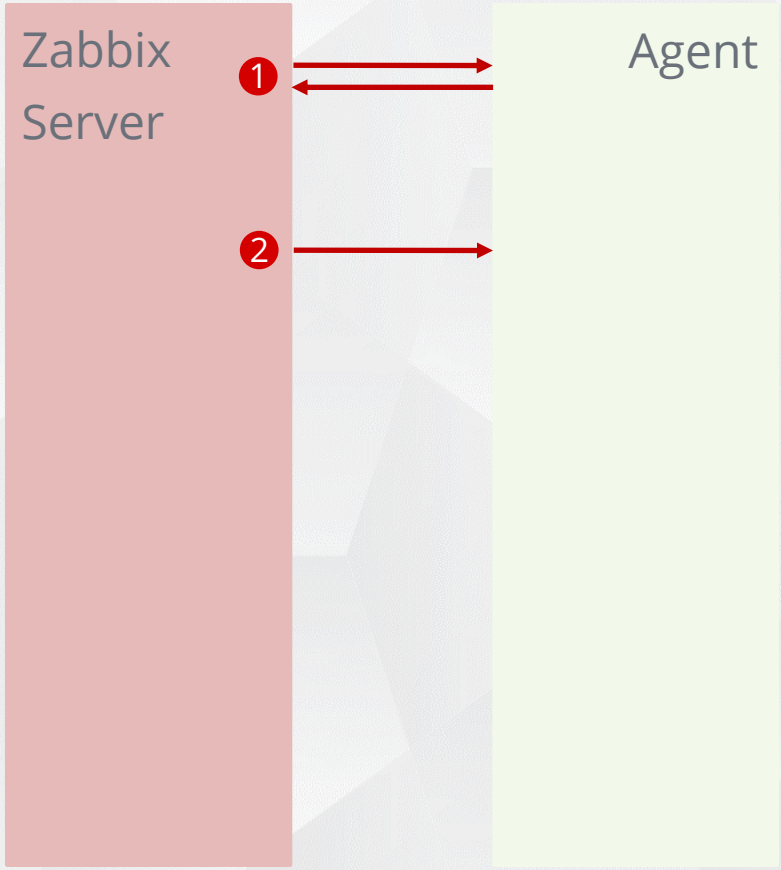
Several configuration parameters define how Zabbix server should behave when an agent check (Zabbix, SNMP, IPMI, JMX) fails and a host becomes unreachable.

The call-outs below identify the Zabbix server behavior found in the diagram

1 Agent check with response ZBX

2 Failed check (network error, timeout).

The host is treated as unreachable ZBX



UNREACHABLE/UNAVAILABLE SETTINGS

3 – UnreachableDelay:

- Defines how often an unreachable host is rechecked using one of the items

5 – UnreachablePeriod:

- Defines maximum total length of the unreachability period

The call-outs below identify the Zabbix server behavior found in the diagram

1 Agent check with response ZBX

2 Failed check (network error, timeout).

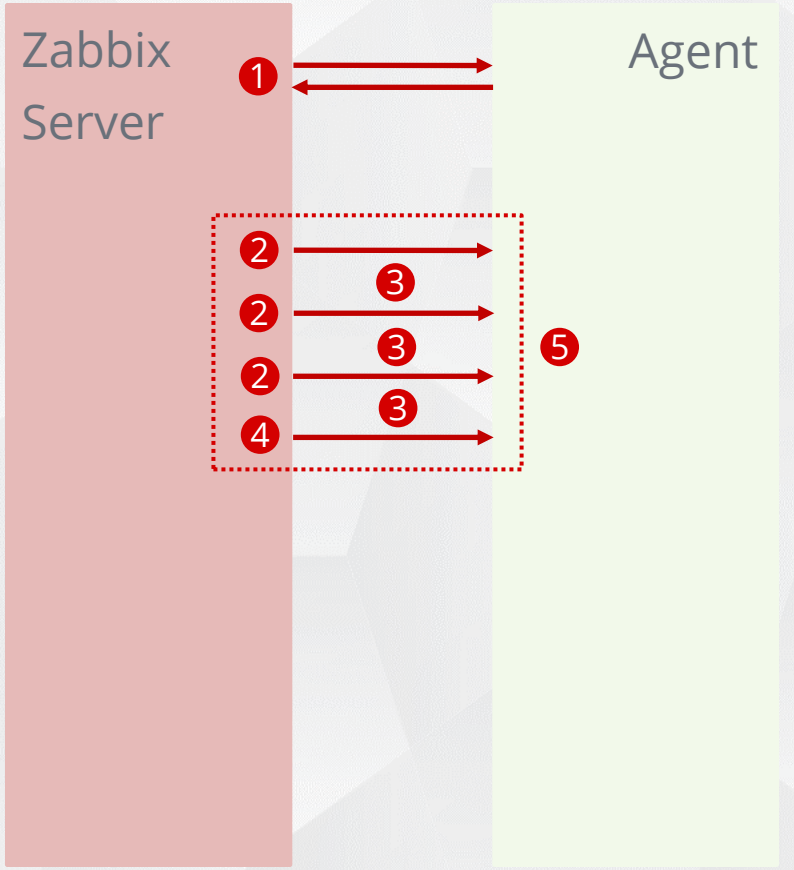
The host is treated as unreachable ZBX

3 15 seconds delay before the next check

4 Failed check

5 Unreachable Period is 45 seconds

The host is treated as unavailable ZBX



- 5 – If the UnreachablePeriod ends and the host has not reappeared:
  - The host is treated as unavailable
- 6 – UnavailableDelay:
  - Defines how often a host is checked during host unavailability

The call-outs below identify the Zabbix server behavior found in the diagram

1 Agent check with response

2 Failed check (network error, timeout).

The host is treated as unreachable

3 15 seconds delay before the next check

4 Failed check

5 Unreachable Period is 45 seconds

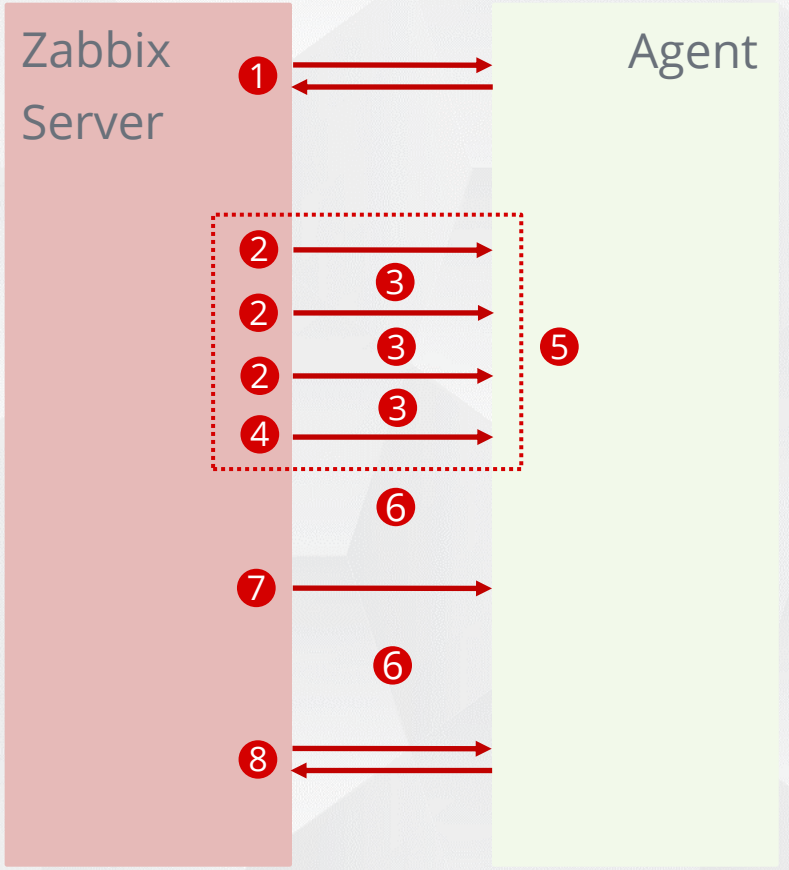
The host is treated as unavailable

6 60 seconds delay before the next check

7 Failed check

8 Agent check with response

The host is treated as available



UNAVAILABILITY CHECKS

- First 45 seconds, when a host is unreachable:
  - Different item keys are checked each time
  - Host availability icon is still green
- After 60 seconds a host is marked as unavailable:
  - Host availability icon turns red
- Can be tracked in Zabbix server log file

```
20200824:103500 Zabbix agent item "system.localtime" failed: first network error, wait for 15 sec
20200824:103515 Zabbix agent item "system.sw.arch" failed: another network error, wait for 15 sec
20200824:103530 Zabbix agent item "kernel.maxproc" failed: another network error, wait for 15 sec
20200824:103545 temporarily disabling Zabbix agent checks on host "PROD Server": host unavailable
20200824:103745 enabling Zabbix agent checks on host "PROD Server": host became available
```

COMMON ITEM KEYS

Area	Suggested key
Host name	system.hostname[<type>]
Availability	agent.ping
Remote services	net.tcp.service[service,<ip>,<port>]
Processes	proc.num[<name>,<user>,<state>,<cmdline>]
Disk space availability	vfs.fs.size[fs,<mode>]
Directory entry count	vfs.dir.count[dir]
Network	net.if.in/out/total[interface]
Memory availability	vm.memory.size[<mode>]
CPU load (Unix)	system.cpu.load[<cpu>,<mode>]
CPU utilization (Win)	system.cpu.util[<cpu>,<type>,<mode>]

Command-line utilities

Help and Runtime control:

```
# zabbix_agentd -h
## Supports runtime control:
# zabbix_agentd -R log_level_increase="active checks"
## Start multiple agent instances
# zabbix_agentd.exe -m -c zabbix_agentd.conf
## Run in foreground:
# zabbix-agentd.exe -f -c zabbix_agentd.conf
```

List built-in items:

```
# zabbix_agentd -p
```

- Does not display active-only keys
- Does not display calculated information

```
# zabbix_agentd -t system.cpu.load
# zabbix_agentd -t "vfs.file.regexp[/etc/passwd,root]"
# zabbix_agentd -t "mysql.ping" -c /etc/zabbix/zabbix_agentd.conf
```

https://www.zabbix.com/documentation/5.0/manual/concepts/agent

COMMAND-LINE UTILITIES - ZABBIX GET

Retrieves information from the remote passive Zabbix agent daemons.

- Can be used in custom scripts
- Can be used to check agent availability or user parameters
- Can specify source IP
- Supports encryption

To get a list of supported command-line attributes:

```
# zabbix_get -h
```

Usage:

```
# zabbix_get -s 127.0.0.1 -k system.cpu.load
# zabbix_get -s server.local.lan -p 10050 -k vfs.file.exists[/etc/zabbix/zabbix_agentd.conf]
```

- The agent must be configured to accept incoming connections from the machine executing zabbix_get

It is recommended to use zabbix_get or telnet for testing zabbix agents.

https://www.zabbix.com/documentation/5.0/manual/concepts/get

PRACTICAL SETUP

1. Create three items on the Training-VM-XX host:
    1) CPU load
    2) Interface eth0: Incoming traffic
    3) Interface eth0: Outgoing traffic
2. Force Zabbix server to reload configuration cache
3. Make sure that the items receive data
4. Test your item keys using:
    1) zabbix_get -s <IP/DNS> -k <key>
    2) zabbix_agentd -t <key>

Advanced task: Get bits per second for your network items

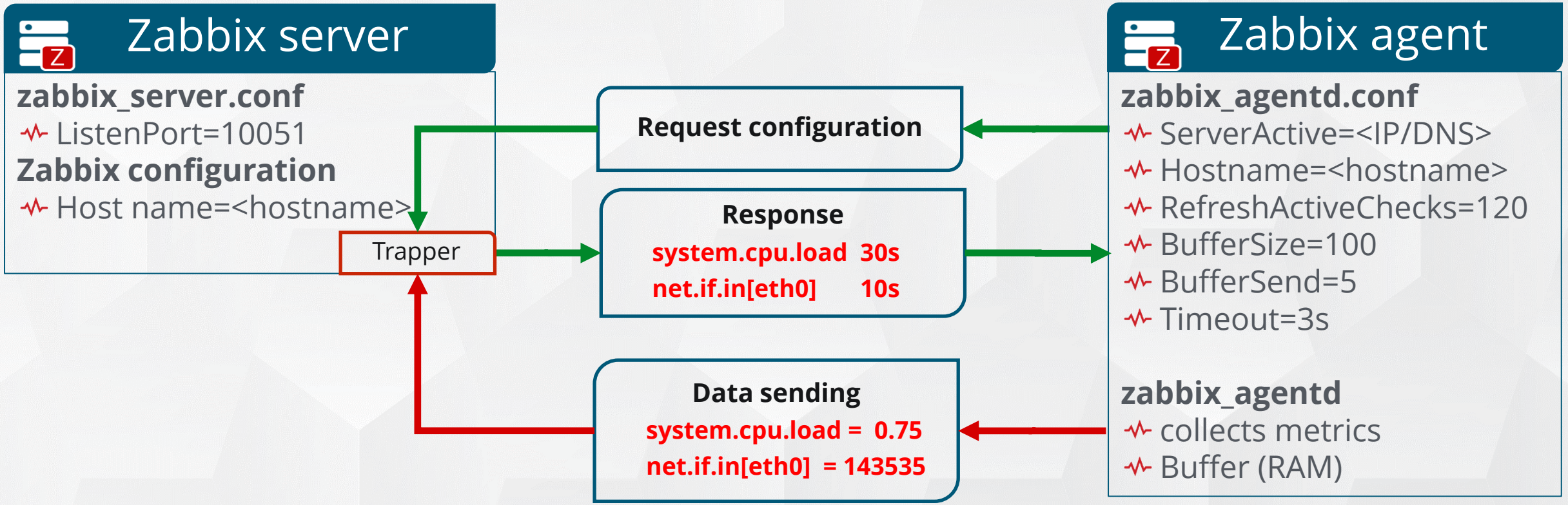
Zabbix agent active checks

Active checks require more complex processing.

- Agent retrieves a list of items from Zabbix server for independent processing:
  - On agent start
  - Every 120 seconds
- Periodically sends new values to the server in bulk:
  - Every 5 seconds and clears buffer
  - When a buffer is full (if this happens in less than 5 seconds)
  - Local system timestamp is used for collected values
- Can buffer data in memory if the connection is lost.
  - Default buffer size is 100 values

"Execute now" can not be used for active items in frontend

ACTIVE CHECK - DATA FLOW



Timeout in zabbix_server.conf does not affect active checks.

Host settings in the frontend:

- Configuration > Hosts > {host}
  - Explicitly set: "Host name" = student-XX

zabbix_agentd.conf

```
ServerActive=<IP/DNS>
Hostname=student-XX
#HostnameItem=system.hostname
```

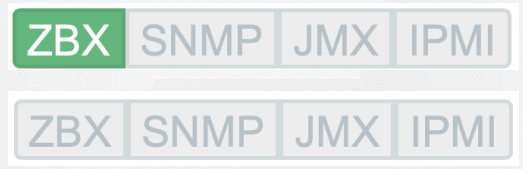
- If both are not set, the agent will automatically use "system.hostname"

"Visible name" is not used to identify a host.

PASSIVE VS ACTIVE CHECKS

Passive (polling)

Active (trapping)



Agents support both modes simultaneously.

PRACTICAL SETUP

1. Configure Zabbix agent for active checks
2. Full Clone the host "Training-VM-XX"
   - Host name: student-XX (training VMs hostname)
   - Visible name: Training-VM-XX active checks
3. Use Mass update button to change the type to Zabbix agent (active)
4. Force Zabbix server to reload configuration cache
5. Make sure that the items receive data

Advanced task: Get a hostname from the system using the agent key.

Q&A
Passive VS Active checks

Monitoring Windows

Zabbix agent on MS Windows:

- Runs as a service (Local system account)
- Possible to run single or multiple instances
- Single instance can use the default configuration file:
  - C:\Program Files\Zabbix\zabbix_agentd.conf
  - Configuration file specified in the command line
- Each agent instance must have its own configuration file.

Can be installed and controlled using msi packages or from command line:

```
# zabbix_agentd.exe --install --config <your_configuration_file>
# zabbix_agentd.exe --start
# zabbix_agentd.exe --stop
# zabbix_agentd.exe --uninstall
```

See the configuration file for configuring options details.

https://www.zabbix.com/documentation/5.0/manual/appendix/install/windows_agent

The key to monitor all Windows event logs:

- eventlog[name,<regexp>,<severity>,<source>,<eventid>,<maxlines>,<mode>]
  - name of the event log (System, Security etc)
  - regexp - regular expression describing the required pattern
  - severity - regular expression describing severity (“Information”, “Warning”, “Error”, “Critical”, etc.)
  - source - regular expression describing source identifier
  - eventid - regular expression describing the event identifier, for example (529|680)

Additional settings for event log items:

- Item must be configured as an active check
- Type of information: Log (retrieves timestamp of original event)

Minimum permission level for Windows agent items.

https://www.zabbix.com/documentation/5.0/manual/appendix/items/win_permissions_

Windows services items:

- Discovery of Windows services:
  - service.discovery - reports back a JSON object containing Windows services;
  - used in the low-level discovery, which is discussed later.
- Windows service monitoring item:
  - Key: service.info[service,<param>]
    - **param** values: display name, state, path, user, startup or description
    - if the **param** is not specified (service.info[service]), the default value state is used.
  - Two value maps are available:
    - Windows service state
    - Windows service startup type.
- Listing of services:
  - Key: services[<type>,<state>,<exclude>]
    - type - all (default), automatic, manual or disabled
    - state - all (default), stopped, started etc.
    - exclude - services to exclude from the result.
  - Example: services[automatic, stopped] - a list of stopped services, that should be running.

...manual/config/items/itemtypes/zabbix_agent/win_keys#service.discovery#service.discovery

WINDOWS PERFORMANCE COUNTERS

You can effectively monitor Windows performance counters with Zabbix.

- Value of any performance counter:
  - perf_counter[counter,<interval>]
- Value of any performance counter in English (recommended):
  - perf_counter_en[counter,<interval>]
    - This item is only supported on Windows Server 2008/Vista and above
    - List of English strings in the registry:
      - "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Perflib\009"
- Example:
  - perf_counter_en["\Processor(0)\Interrupts/sec"]

In order to get a full list of performance counters run from the command line:

- typeperf -qx

Possible to use numerical form for counters (check the documentation):

- perf_counter_en[10] (% Processor Time=10)
- perf_counter_en[12] (File Read Operations/sec=12)

https://www.zabbix.com/documentation/5.0/manual/config/items/perfcounters

WMI queries are performed with WQL:

- Windows Management Instrumentation Query Language

- Agent keys:
  - wmi.get[<namespace>,<query>]
    - Single metric
  - wmi.getall[<namespace>,<query>]
    - All metrics in JSON format

- Examples:
  - Status of the first physical disk:
    - wmi.get["root\cimv2,select status from Win32_DiskDrive where Name like '%PHYSICALDRIVE0%'"]
  - Status information of all physical disks:
    - wmi.getall["root\cimv2,select * from Win32_DiskDrive where Name like '%PHYSICALDRIVE%'"]

{MACRO}
{$MACRO}

Zabbix has built-in macros.

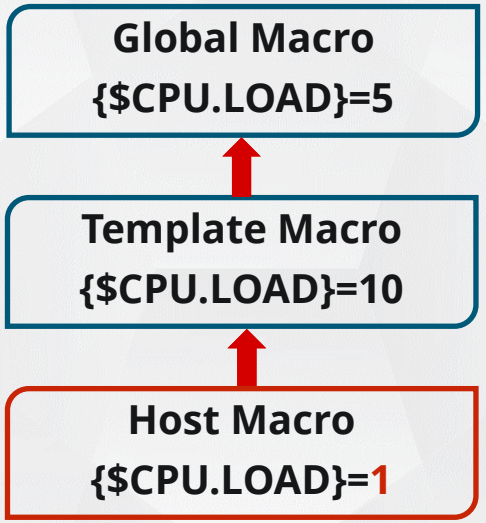
- Macros resolve to a specific value depending on the context.
- Syntax: {MACRO.NAME}
- Used in various places:
  - Hosts and templates: {HOST.NAME}, {HOST.CONN}
  - Triggers: {ITEM.VALUE}, {ITEM.LASTVALUE}
  - Tags
  - Notifications: {EVENT.DATE}
  - Web monitoring
  - etc.
- Macros allow to save time and make Zabbix configuration more transparent

See full list of supported macros in the documentation.

https://www.zabbix.com/documentation/5.0/manual/appendix/macros/supported_by_location

User macros:

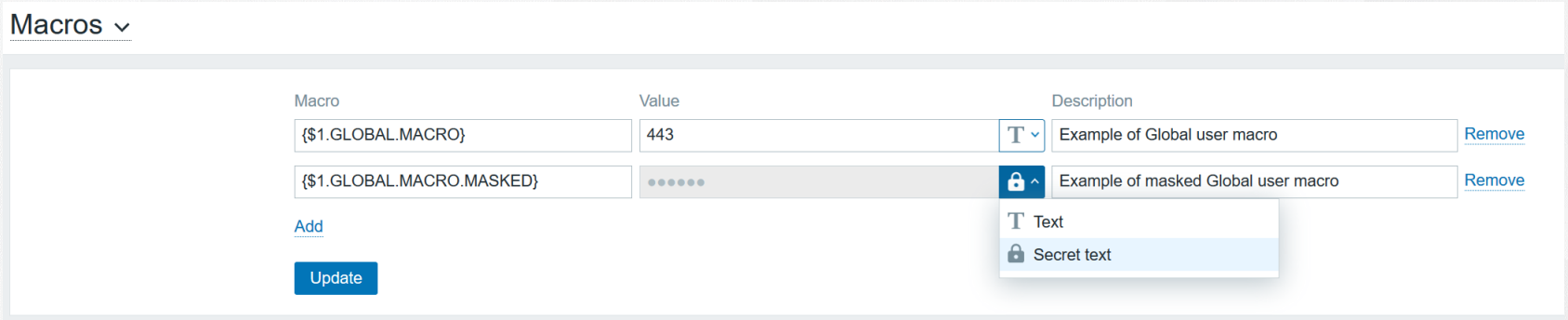
- Variables for greater flexibility;
- Syntax (A-Z 0-9 _ .): {$MACRO.NAME}
- Two options:
  - "Text" - displays field data as is (default);
  - "Secret text" - masks sensitive data with asterisks.
- Three levels:
  - Global
  - Template
  - Host
- If a macro is not defined on a host level:
  - templates level(of increasing depth) is used;
  - if still not found, the global macro will be used, if exists.



If Zabbix is unable to find a macro, the macro will not be resolved!

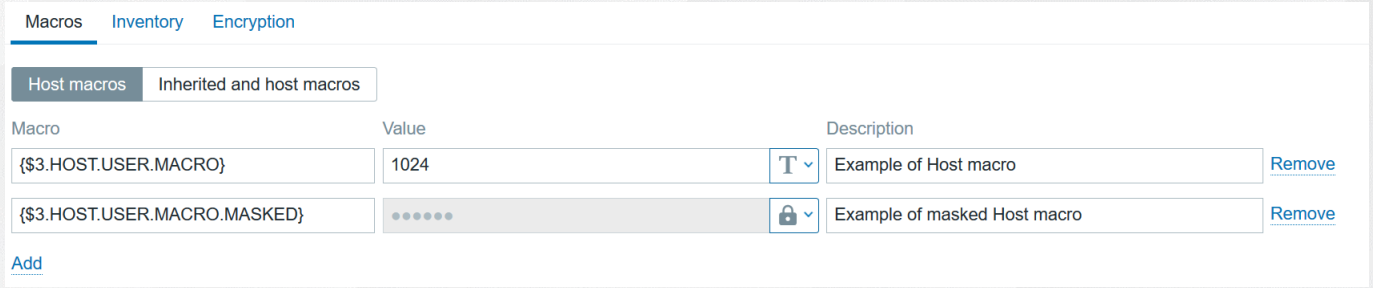
USER MACRO CONFIGURATION

Global macros: Administration > General > Macros



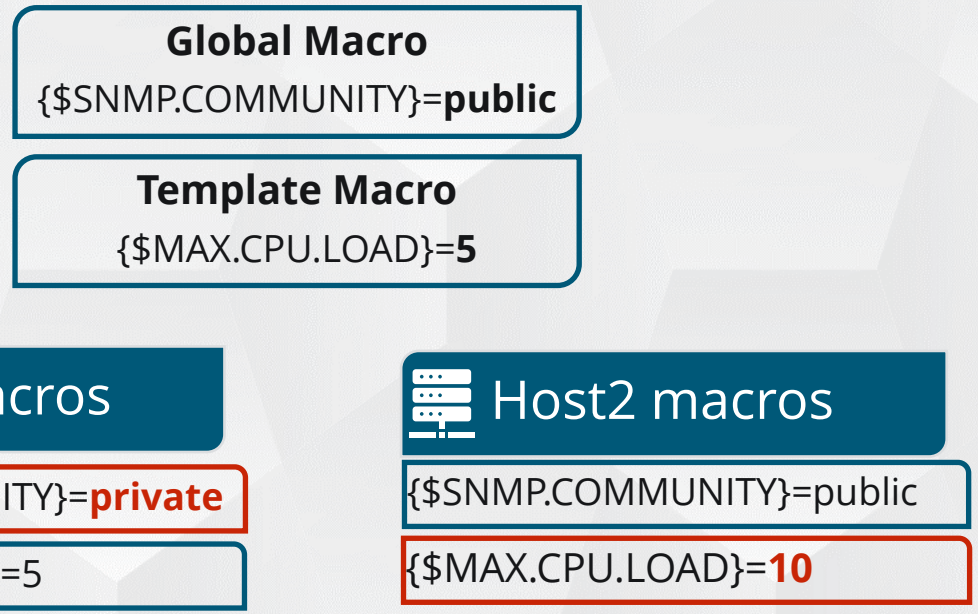
Template macros: Configuration > Templates > {template} > Macros

Host level macros: Configuration > Hosts > {host} > Macros

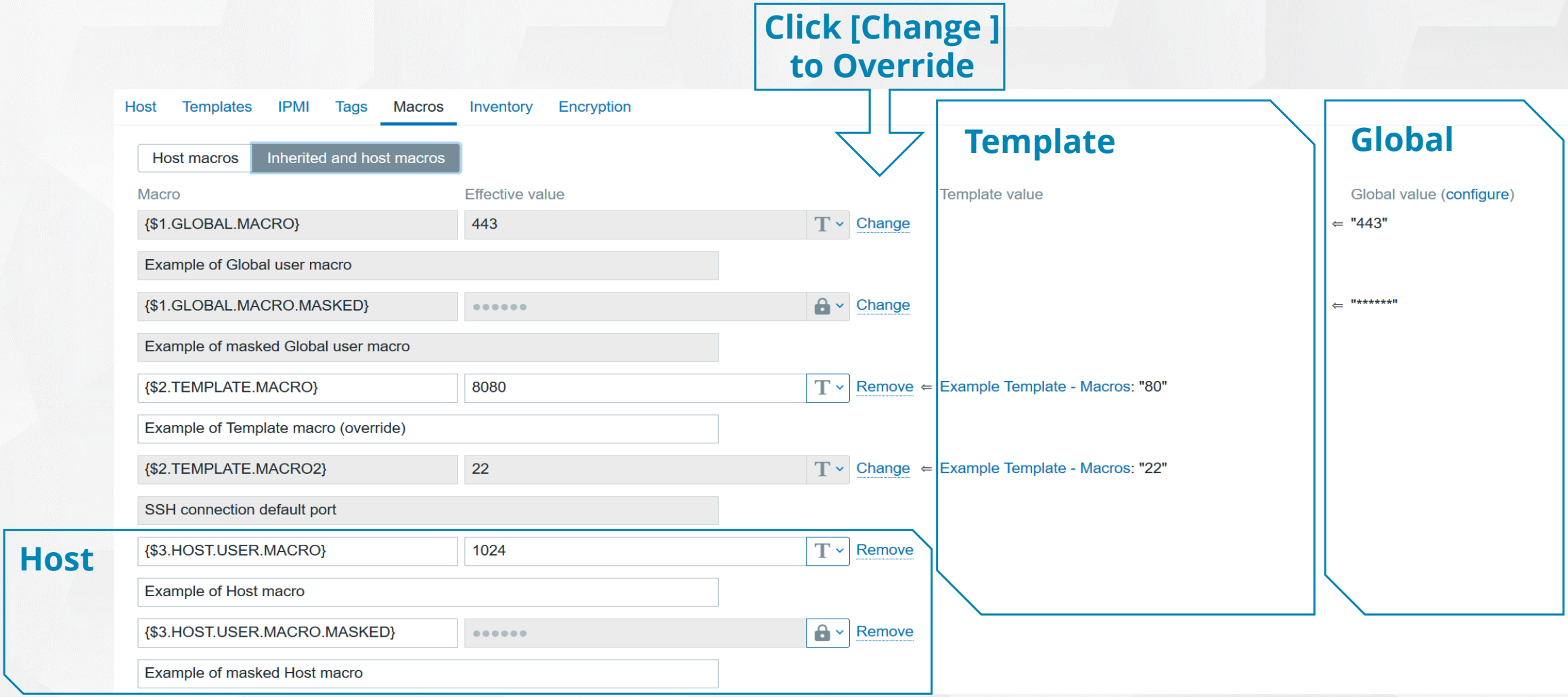


One template, different parameters:

- Different item key parameters:
  - net.tcp.service[ssh,{$SSH.PORT}]
- Different trigger expression values:
  - {server:system.cpu.load.last()} > {$MAX.CPU.LOAD}
  - {server:system.cpu.load[,avg1].min({$CPU.LOAD.PERIOD})}>{$MAX.CPU.LOAD}
- Different credentials:
  - ssh.run[remote.command]
    - {$SSH.USERNAME}
    - {$SSH.PASSWORD}



Inherited macros can be seen and modified on templates and hosts:



Restricting users from viewing a global macro value on a host/template level.

Problem detection
Triggers

Triggers are logical expressions that “evaluate” the data gathered by items.

A trigger:

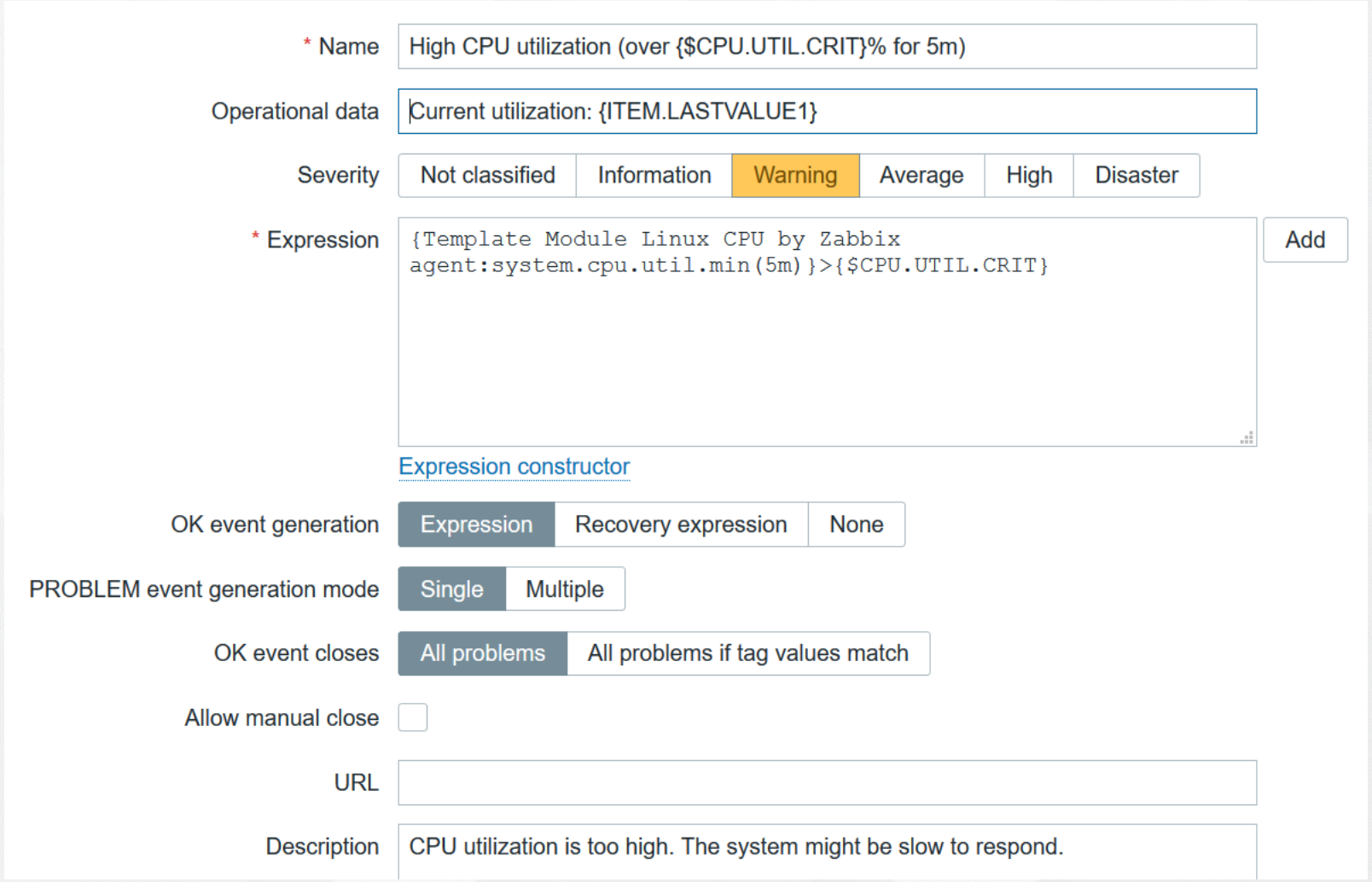
- represents current system state;
- allows to define a threshold (state of data is "acceptable"/"not acceptable");
- may have the following statuses:

OK	normal trigger state.
PROBLEM	something happened. For example, the processor load is too high.

- When a trigger changes state, a new event is created.
- If a trigger goes into PROBLEM state, it's also called - “fired”.
- Trigger status (expression) is recalculated every time Zabbix server receives a new value - that is part of the expression.
- Triggers are evaluated based on history data only; trend data are never considered.
- You can build trigger expressions with different degrees of complexity.

https://www.zabbix.com/documentation/5.0/manual/config/triggers

Configuration > Hosts > Triggers > Create trigger



Name: Trigger name

- Supported macros:
  - {HOST.HOST}, {HOST.NAME}, {HOST.CONN}, {HOST.DNS}, {HOST.IP}, {ITEM.VALUE}
  - User macros {$MACRO}
- Examples:
  - Server mail.example.com is not reachable
  - High CPU utilization on db2.example.com
  - Service "DHCP" (DHCP Client) is not running
- Use of macros makes names dynamic - representing state or info of items/values
- $1, $2...$9 macros can be used to refer to the constant of the expression

Trigger name:	Processor load above $1 on {HOST.NAME}
Trigger expression:	{Linux server:system.cpu.load[percpu,avg1].last()}>5}
Problem Name:	Processor load above 5 on Linux server

TRIGGERS - OPERATIONAL DATA AND SEVERITY

Operational data:

- Allows to define arbitrary strings along with macros (informative).
- Macros will resolve dynamically to real time data in Monitoring> Problems.
- If not configured, latest values of all items from the expression are displayed.

Severity - select by clicking the buttons:

- Visual representation of triggers (Different colors);
- Filtering based on severities (in Problems, Maps, Dashboards etc.);
- Audio in global alarms (different sound for different severities);
- Different user media (notification channel) for different severities:
  - SMS - High and Disaster
  - Email – All;
- Limiting actions by conditions against trigger severities.

TRIGGERS - EXPRESSION SYNTAX

A simple expression can look like this:

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

Operators: **- + / * < > = <> >= <= not or and**

Examples:

{	<host>	:	<key>	.	<function>(<parameters>)	}	<operator>	<constant>
{	Zabbix	:	system.uptime	.	last()	}	<	10m
{	Template CPU	:	system.cpu.load[all]	.	min(5m)	}	>=	{$CPU.LOAD}

Refering to several items:

{host1:item1.func(5m)}>10 and {host1:item2.func(5m)}>5

Refering to items from several hosts:

{host1:item.func(5m)}>10 and {host2:item.func(5m)}>5 and {host3:item.func(5m)}<3

https://www.zabbix.com/documentation/5.0/manual/config/triggers/expression

Functions: **min(), max(), avg(), last(), diff(), count(), delta(), time(),** etc.

- Syntax: function(parameter1,<parameter2>)
- Most of functions have parameters (mandatory and optional):
  - Example: count (sec|#num,<pattern>,<operator>,<time_shift>)

Parameters:

- sec maximum evaluation period
  - Supported suffixes: s, m, h, d, w
  - In seconds, if suffix is not specified
- #num number of latest collected values (preceded by a hash mark)
- <str> optional parameters

{zabbix:system.cpu.load.last()}>5	Evaluate most recent value, "fire" if more than 5
{zabbix:system.cpu.load.min(10m)}>5	Evaluate values for last 10 minutes, "fire" if all are more than 5
{zabbix:system.cpu.load.max(#10)}<5	Evaluate last 10 values, "fire" if all values are less than 5 (IDLE)

https://www.zabbix.com/documentation/5.0/manual/appendix/triggers/functions

Compare two item values (numerical or string) using function last():

{host1:item1.last()}={host1:item2.last()}

{host1:item1.last()}={host2:item1.last()}

{host1:item1.last()}={host2:item2.last()}

{host1:item1.last(#1)}={host1:item1.last(#5)}

{Linux server:vfs.file.contents[/etc/os-release].last()}	<>	"CentOS Linux release 8.1.1911 (Core)"
{Server1:vm.memory.size.last()}	<>	{Server2:vm.memory.size.last()}
{Server1:system.hw.macaddr[eth0,short].last()}	=	{Server2:system.hw.macaddr[eth0,short].last()}
{Server1:system.hw.macaddr[eth0,short].last(#1)}	=	{Server1:system.hw.macaddr[eth1,short].last(#1)}

last() = last(#1)

Examples:

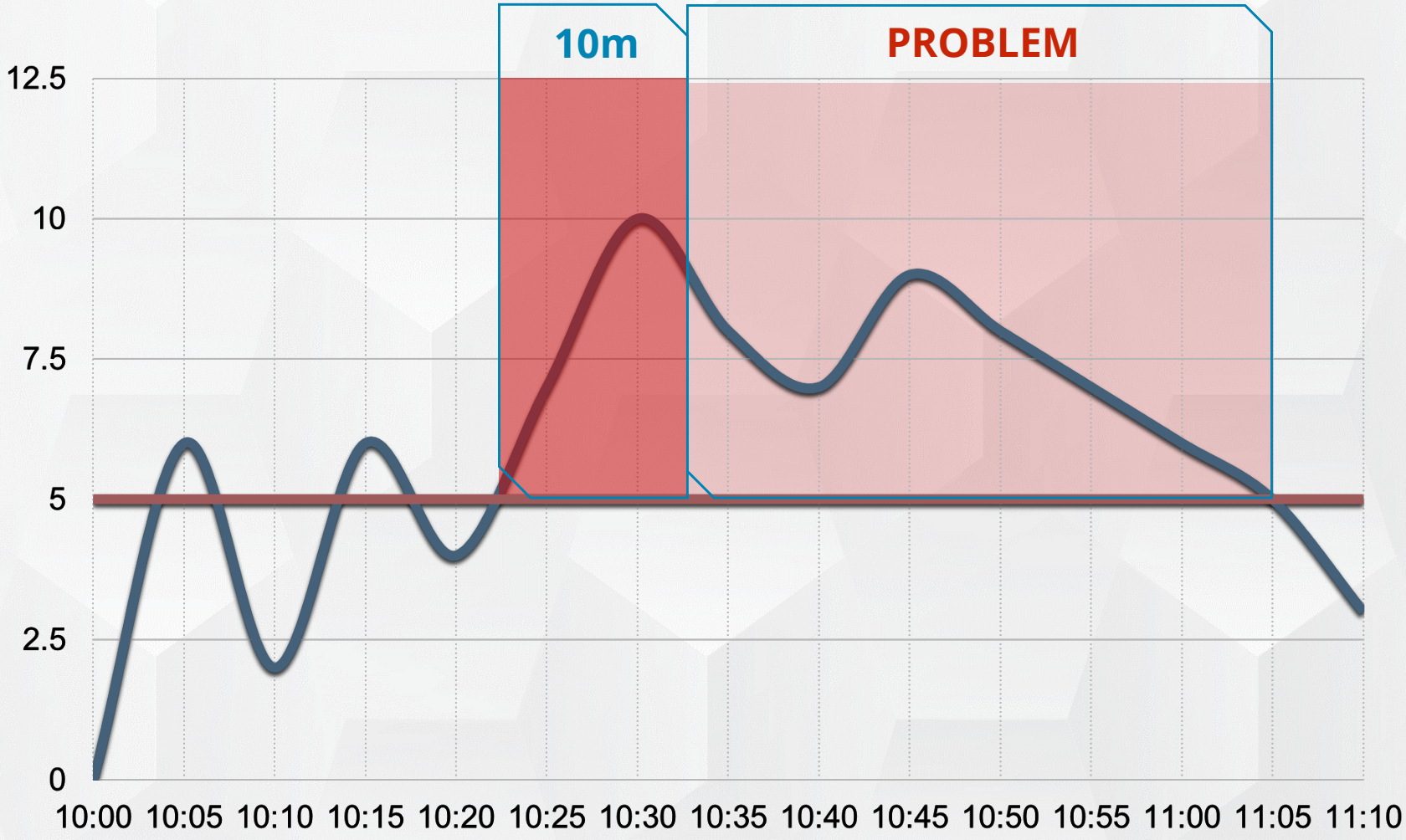
max(10m) = 0	Availability check - usually all values are 1; the moment we receive 0 - trigger fires
avg(#5) > 1	Problem if average of last 5 values is more than 1
str(error) = 1	Problem if value contains the string "error"
diff = 1	Problem if the last and previous values differ
nodata(5m) = 1	Problem if an item has not received any data for at least 5m

Common misconfiguration:

{zabbix:system.cpu.load.last(#3)}>5	last(#3) - third most recent value (not three latest values)
{zabbix:system.cpu.load.last(5m)}>5	last(5m) - the most recent value (5m is ignored)
{zabbix:system.cpu.load.last(5)}>5	last(5) - the most recent value (not last 5 values)
{zabbix:agent.ping.nodata(10s)}=1	the nodata() function is evaluated every 30s (set proper intervals)

Time difference on a server and agent will affect time functions and cause false positives.

TRIGGERS - LESS SENSITIVE

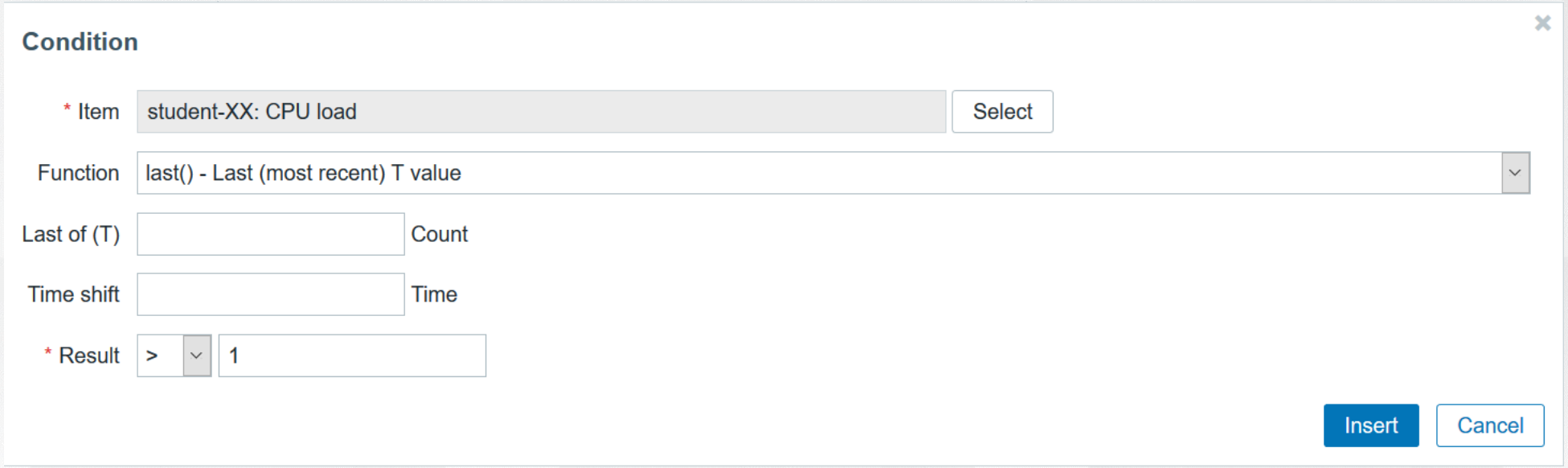


{server:system.cpu.load.min(10m)} > 5

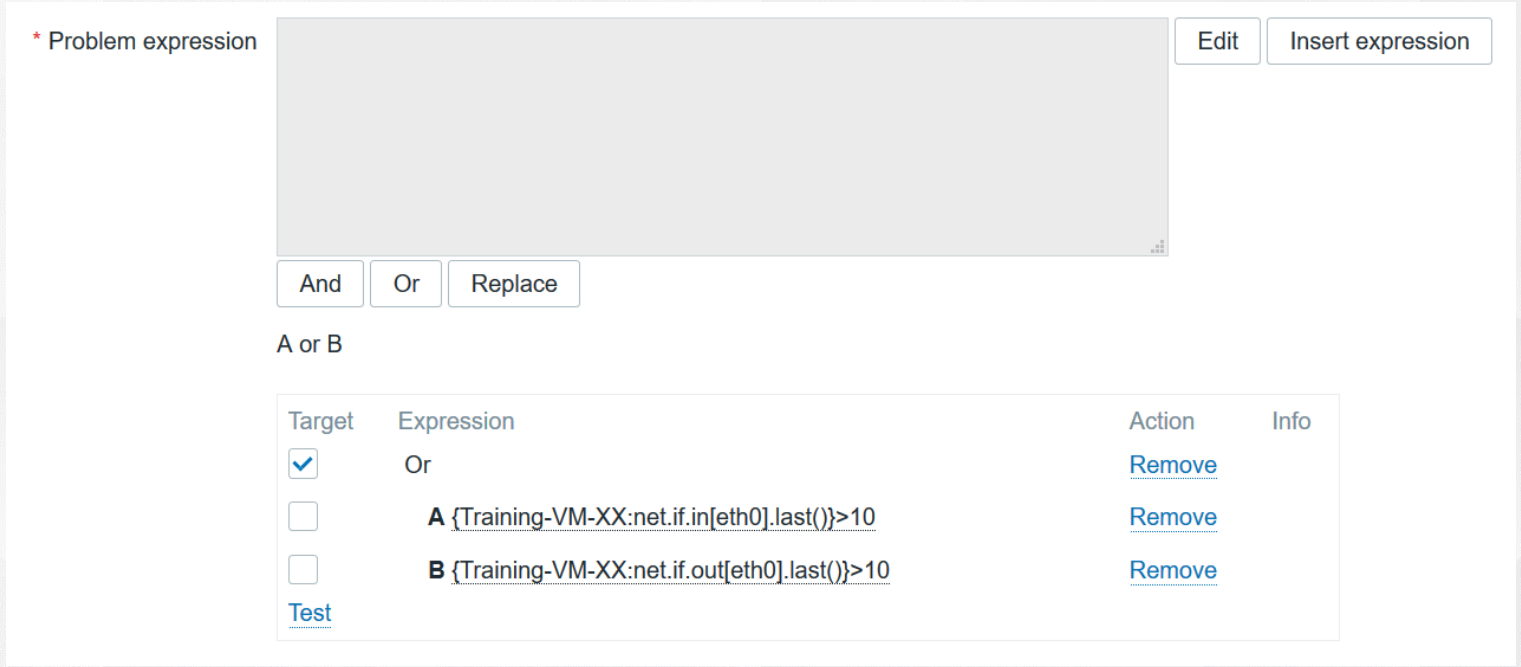
Expression can be added/edited:

- Manually (type into the Expression field):

- Using wizard (press the Add button):

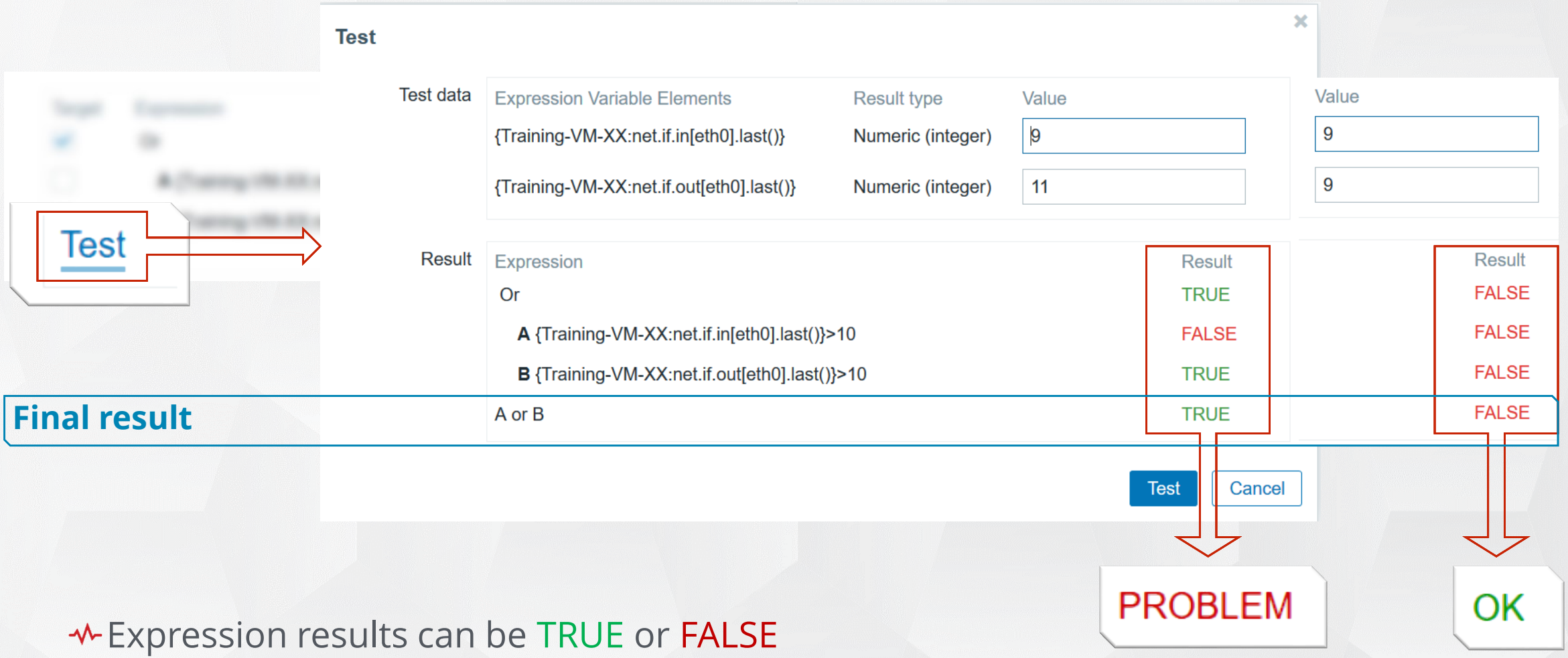


Use the Expression constructor to create complex triggers.



- An easy way to construct multiple combinations of items using **And**, **Or** scenarios
- Use brackets to combine multiple expressions: (A or B) and (C or D)

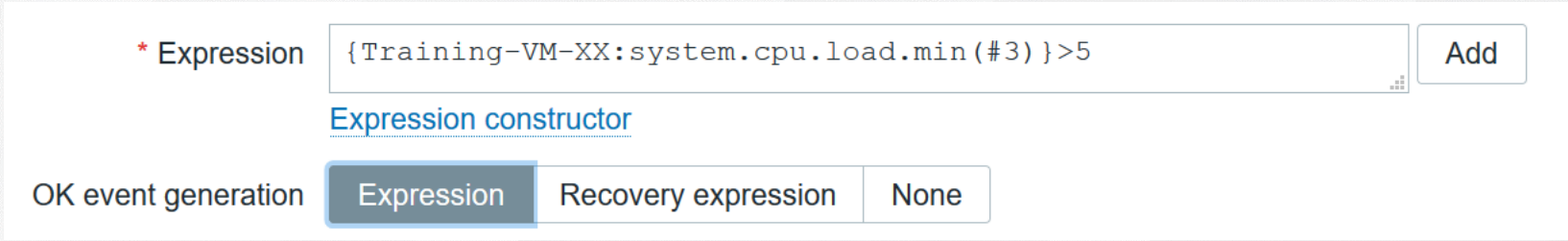
You can enter sample values and check the expression result:



- Expression results can be TRUE or FALSE
- A PROBLEM is generated, if the final result is TRUE

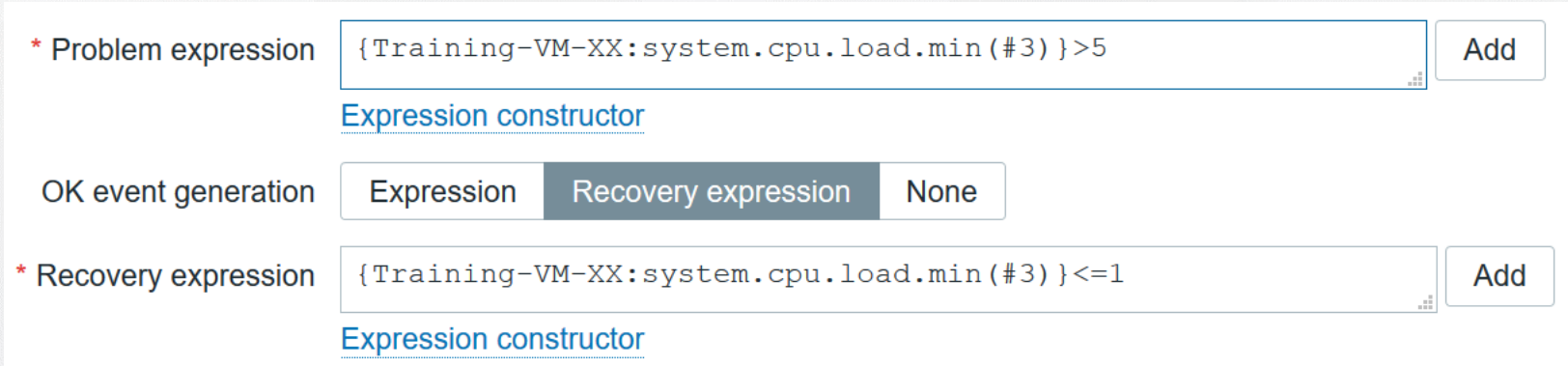
Expression:

- OK events are generated based on the problem expression.



Recovery expression:

- OK events are generated, if both conditions are met:
  - Problem expression evaluates to FALSE
  - Recovery expression evaluates to TRUE



None:

- the trigger will never return to the OK state on its own.

TRIGGERS - PROBLEM EVENT GENERATION

PROBLEM event generation mode: **Single** | Multiple

Single:

- Only one event is generated, when a trigger goes into the Problem state

Time ▾	Recovery time	Status	Info	Host	Problem • Severity	Operational data	Duration	Ack	Actions
18:11:26		PROBLEM		Training-VM-XX	CPU Load is very high on Training-VM-XX	2.33/2.18	1m 36s	No	

Multiple:

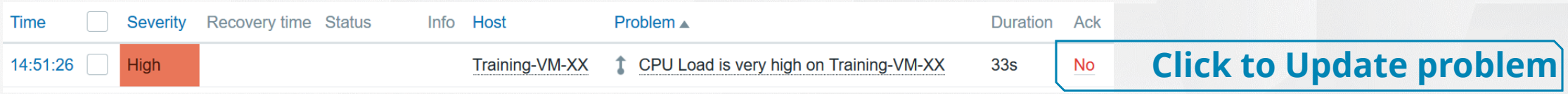
- New event is generated upon every evaluation of the trigger
  - useful for eventlog, log or SNMP traps monitoring, and some other cases

Time ▾	Recovery time	Status	Info	Host	Problem • Severity	Operational data	Duration	Ack	Actions
18:14:46		PROBLEM		Training-VM-XX	CPU Load is very high on Training-VM-XX	2.4/2.4	6s	No	
18:14:36		PROBLEM		Training-VM-XX	CPU Load is very high on Training-VM-XX	2.47/2.4	16s	No	
18:14:26		PROBLEM		Training-VM-XX	CPU Load is very high on Training-VM-XX	2.56/2.4	26s	No	
18:14:16		PROBLEM		Training-VM-XX	CPU Load is very high on Training-VM-XX	2.48/2.4	36s	No	
18:14:06		PROBLEM		Training-VM-XX	CPU Load is very high on Training-VM-XX	2.1/2.4	46s	No	

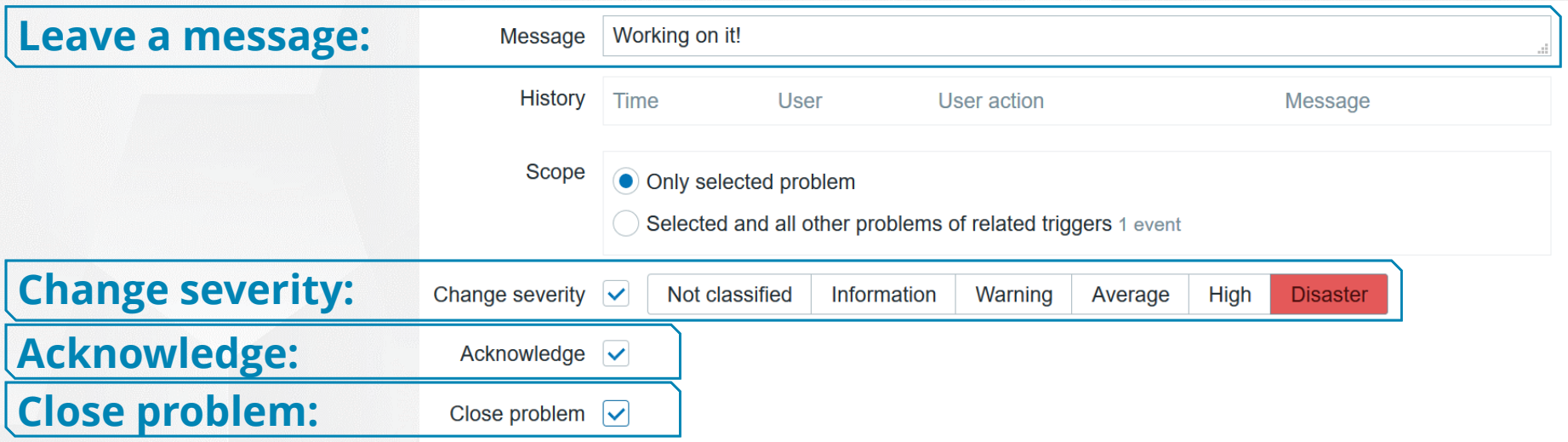
Be very careful with Multiple problem generation! Why?

TRIGGERS - PROBLEM UPDATE, ACK, MANUAL CLOSE

To acknowledge a problem, click Yes / No in Ack column.

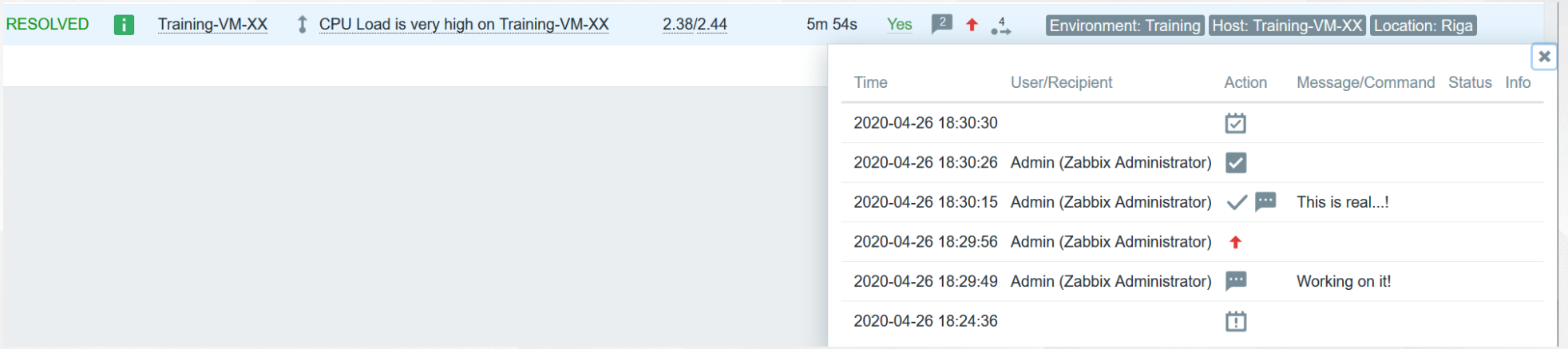


When acknowledging problem users can:

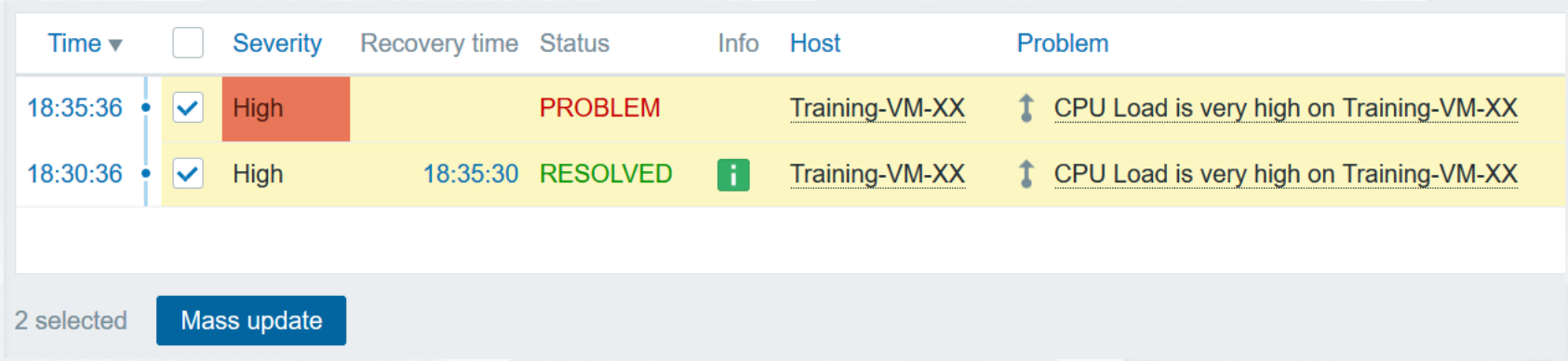


- Manual closing is allowed only if enabled in the trigger's configuration
  - It is useful, if "OK event generation: None" is set
  - Zabbix will create a new problem if the issue is still not fixed

Problem history stores all actions (changes, message, script execution).



Select multiple events and press the [Mass update] button:

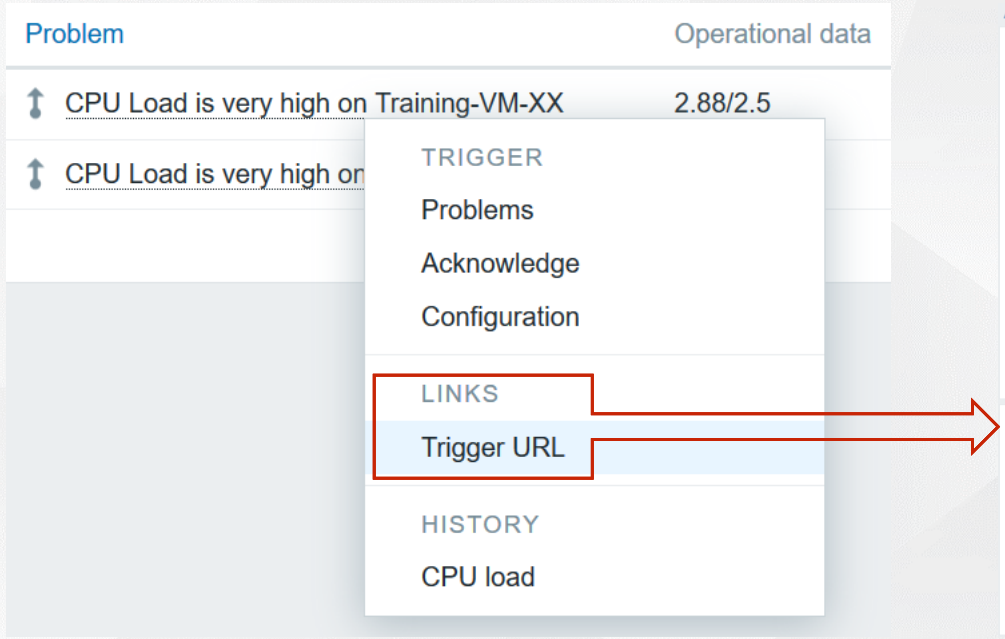


If the URL field is populated with a link

URL zabbix.php?action=map.view&sysmapid=2

The URL is available as a link when clicking on the problem name in:

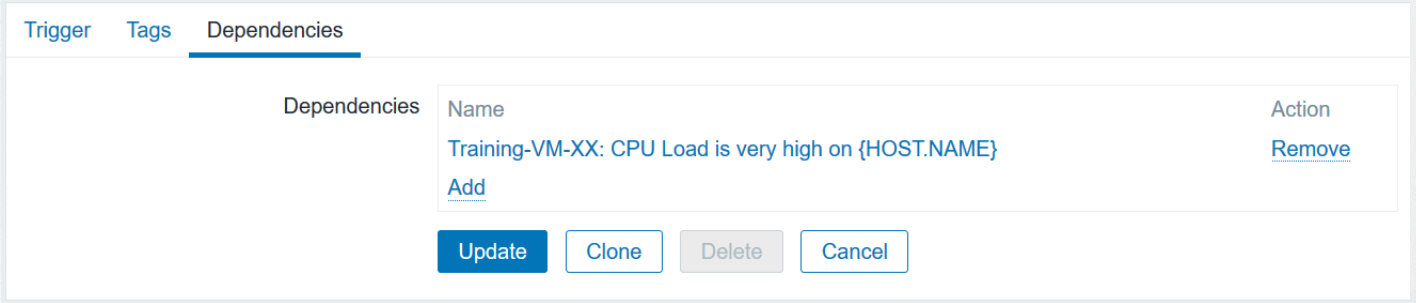
- Monitoring > Problems
- Dashboards

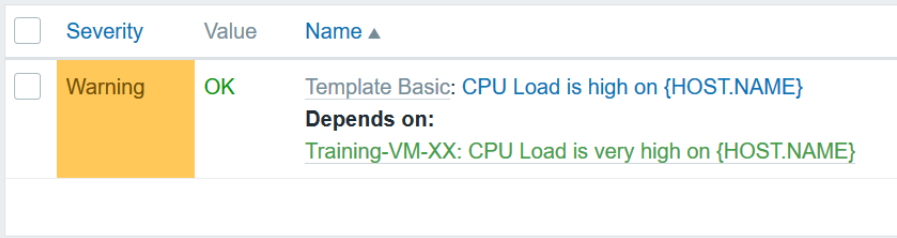


TRIGGERS - DEPENDENCIES

Dependencies are used to:

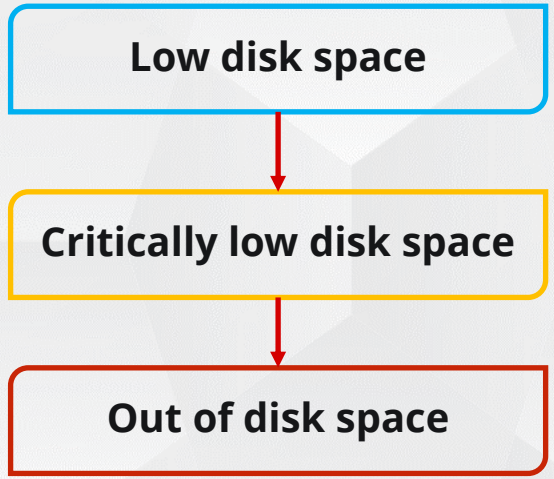
- Show only root problems
- Avoid notifications





Define dependencies between triggers:

- Same host:
  - Problem level (different severities)
- Different hosts:
  - Network devices
  - Applications
  - Other resources



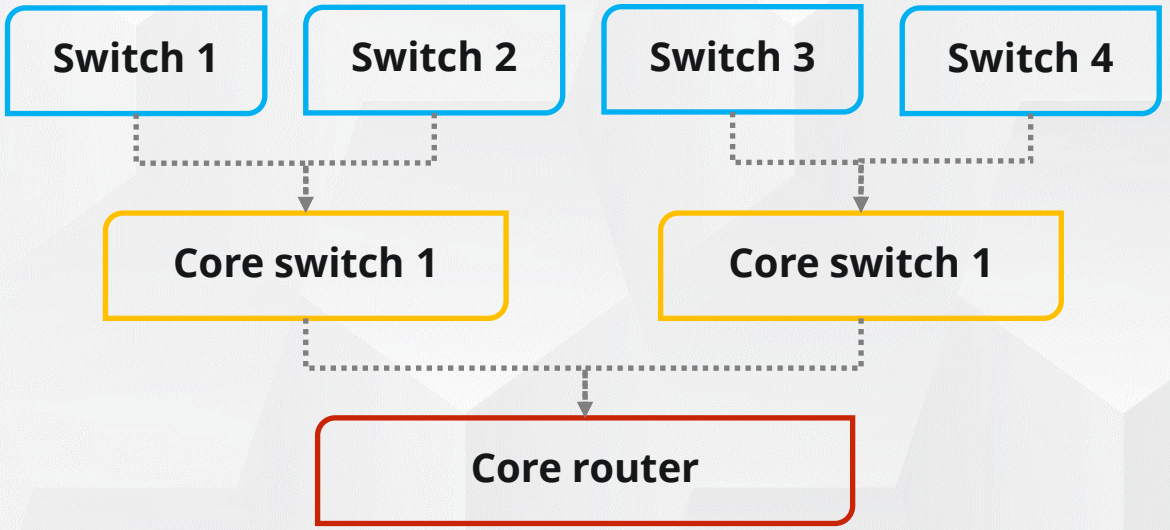
https://www.zabbix.com/documentation/current/manual/config/triggers/dependencies

Multiple levels:

- Host > Switch 1 > Switch 2

Multiple dependencies:

- Host > Router1
- Host > Router2



If both the switch and the router are down, and a dependency is defined:

- The problem, generated by the dependent trigger, will be suppressed and hidden
- Zabbix will not execute actions for the dependent trigger
- The dependent trigger will be re-evaluated and will change its state only after the parent trigger returns to the OK state and the new metrics are received.

Problems are not generated if:

- A trigger is in the "Unknown" state:
  - an item used in the expression becomes "Not supported”
  - cannot evaluate expression
  - a host or an item is disabled
  - no data exists for the evaluation period in history

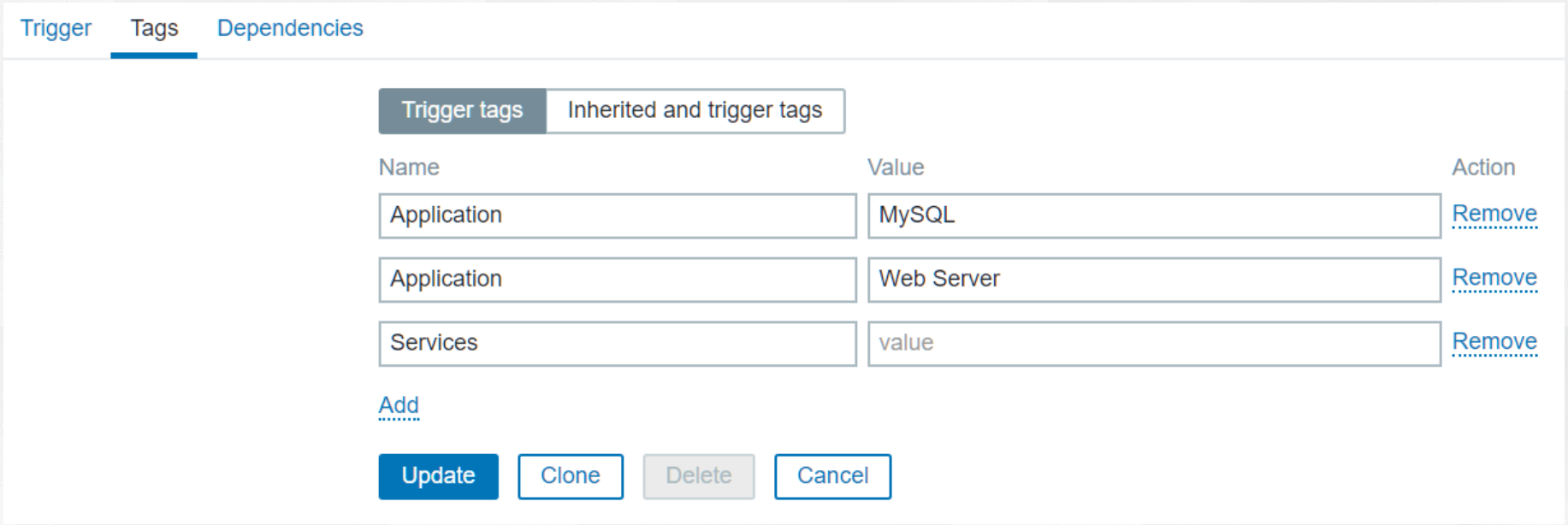
- To discard values during preprocessing steps:
  - Set Custom on fail, then Discard value
  - Use Throttling
    - Discard unchanged
    - Discard unchanged with heartbeat

Introducing throttling may cause existing triggers to change their behavior.

Event tags

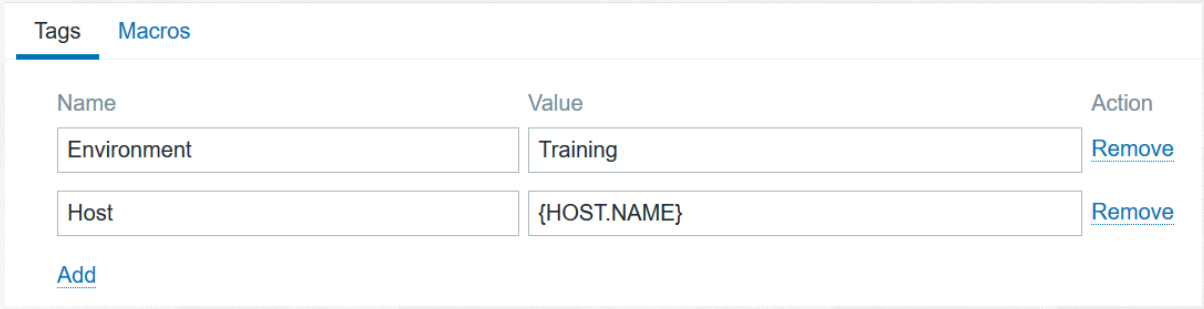
Triggers support an option to define custom event tags:

- After the tags are defined, new problems will get marked with tag data.
- Event tags are realized as a pair of the tag name and value.
  - You can use only the name or pair it with a value.
  - Trigger may have several tags with the same name, but different values.
  - Tag without a value and the same tag with a value can be used simultaneously.



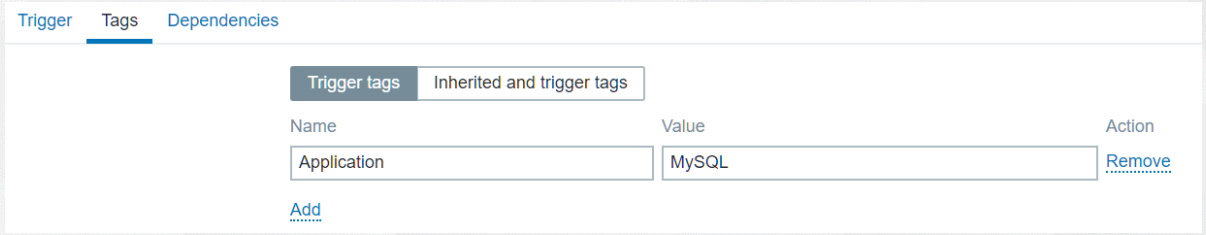
Event tags can be defined on multiple levels:

- Template level - affects all triggers from the template, when linked to hosts.

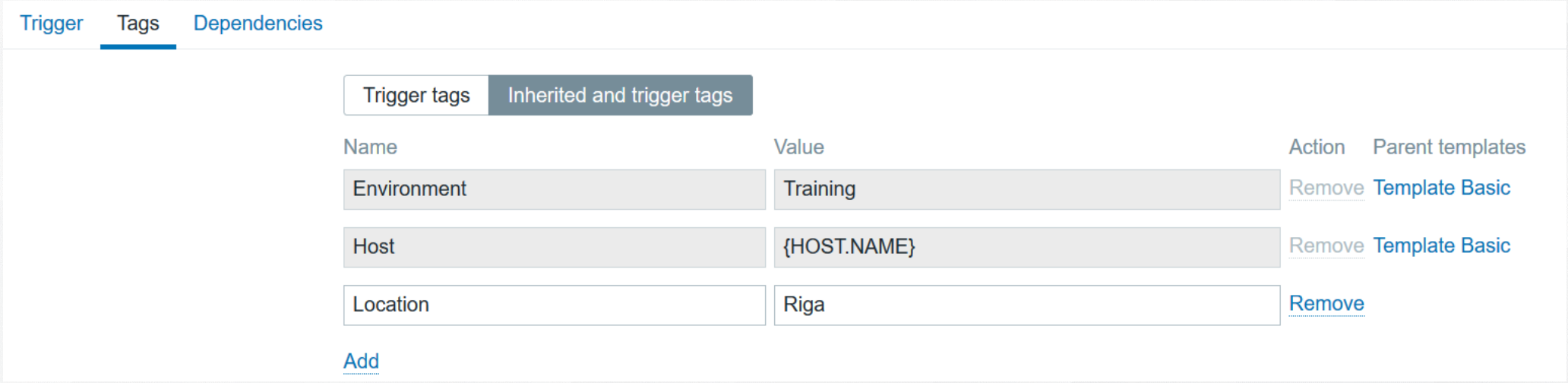


- Host level - affects all triggers of the host.

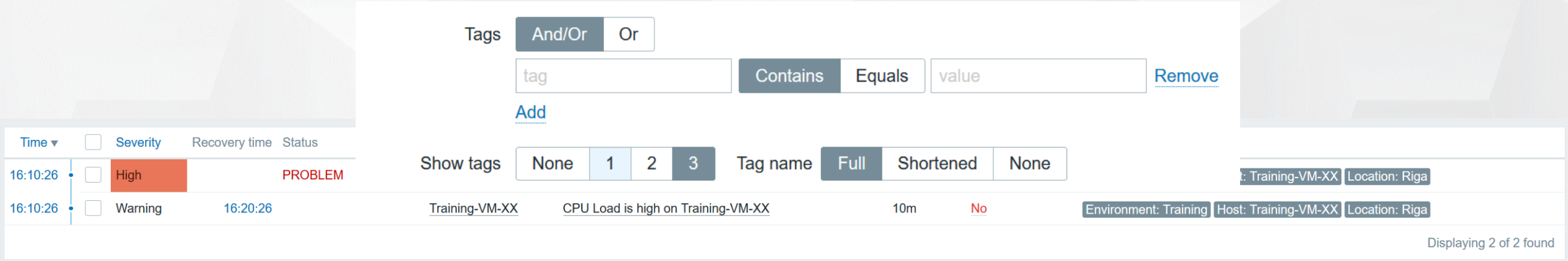
- Individual trigger level - only problems created by this trigger will be marked.



Inherited tags can be viewed in the Trigger configuration form > Tags tab



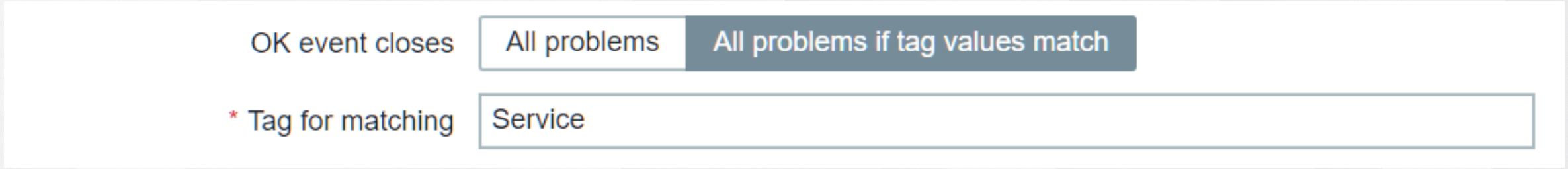
In filters to find specific problems by tags



https://www.zabbix.com/documentation/5.0/manual/config/event_correlation/trigger/event_tags

Some use cases:

- Mark trigger events in the frontend:
  - Problems are marked with these tags in Monitoring>roblems
  - Filtering can be used to display only problems matched by tags
- Filter notifications based on tags in actions
- Define global event correlation rules
- Identify problems in a log file and close them separately based on the tag values
  - Information extracted from item value can be used as a tag value
  - Works with multiple problem event generation mode enabled:
    - Only problems where tag values match are resolved
    - Other problems generated by the same trigger are left unresolved



PRACTICAL SETUP

Create three triggers on the host "Training resources":

1) NTP server is down on Training Resources server
   - Severity: Warning
   - Function: last()
2) ICMP Ping is down on Training Resources server
   - Severity: Average
   - Function: max()
   - Evaluate last 3 values
3) Web service has problems on {Use macro to get host name}
   - Severity: High
   - Add operational data and show item values
   - Use two function s with the operand OR: max(#3) min(5m)

Advanced task: Create a trigger to monitor availability for all 3 services using operand AND

Advanced problem detection

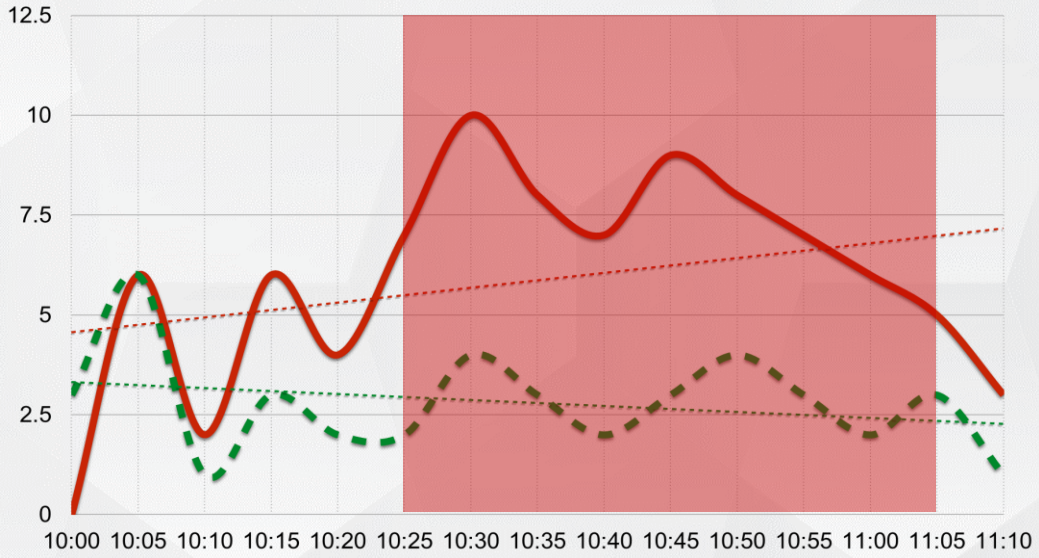
Anomalies can be detected:

Using time shift in functions.

- function(...,<time_shift>)
  - min(10m,1d), max(5m,1d), last(,1w), count(10m,"error",eq,1h)

Example:

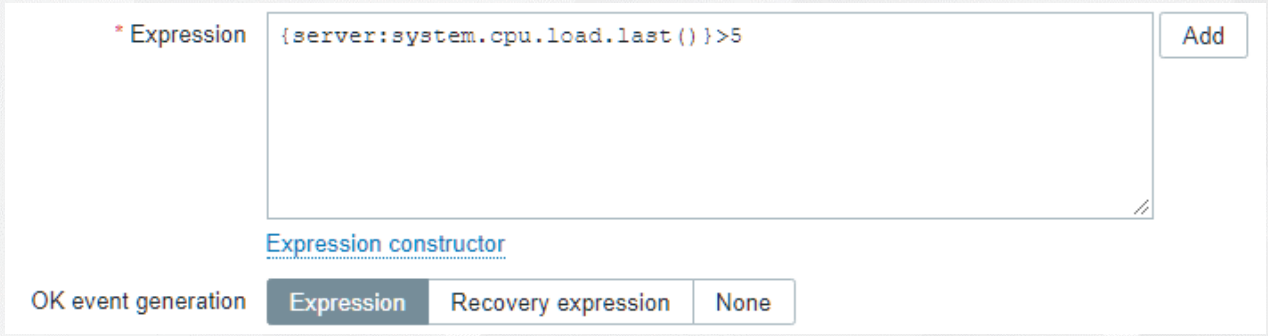
- If load average today exceeds average load of the same hour yesterday 2 times = Problem
  - {host:system.cpu.load.avg(1h)} / {host:system.cpu.load.avg(1h,1d)} >2



Unknown status if no history is stored.

Simple expression may cause flapping

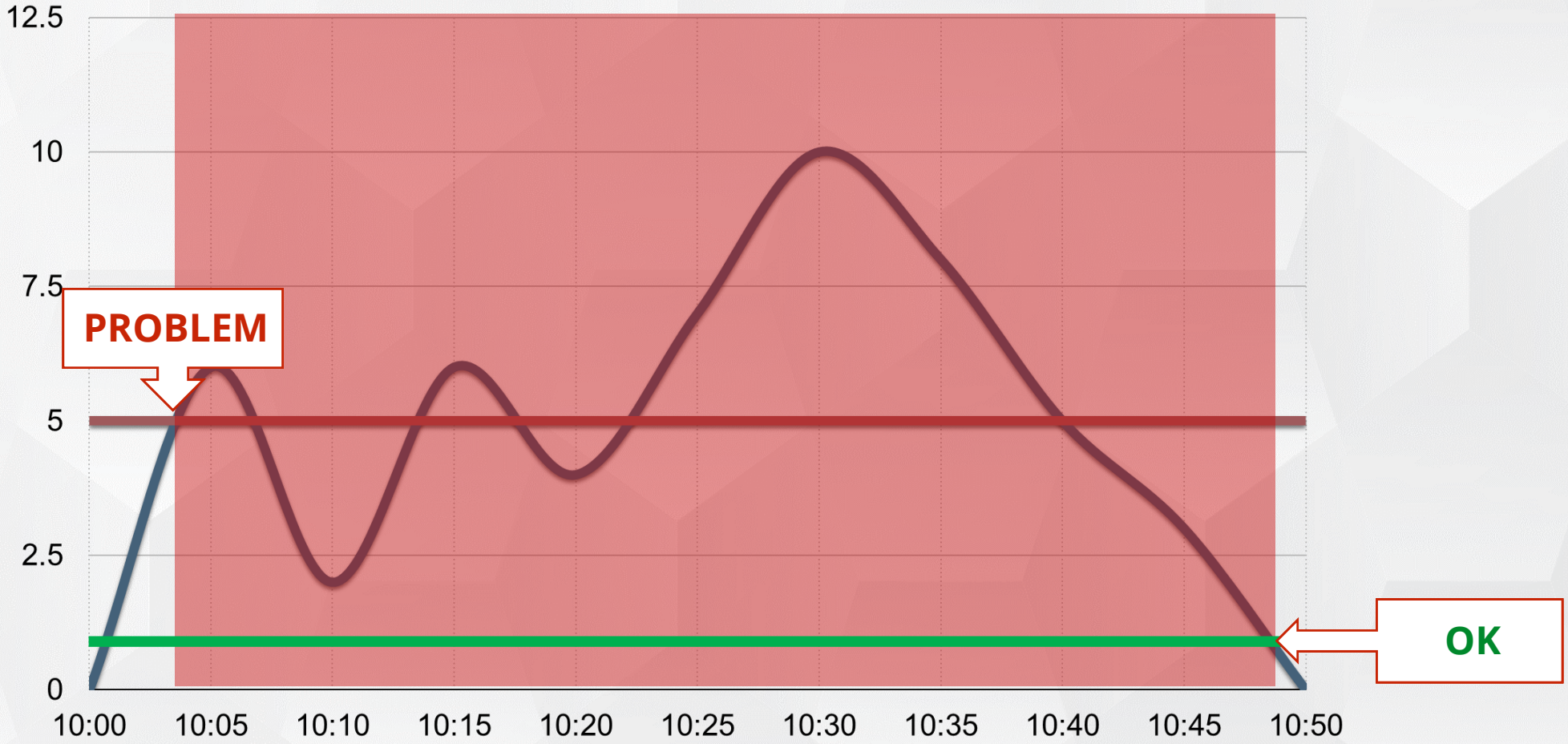
- Simple and very sensitive trigger
  - {server:system.cpu.load.last()}>5



Different conditions for PROBLEM and OK states may prevent flapping

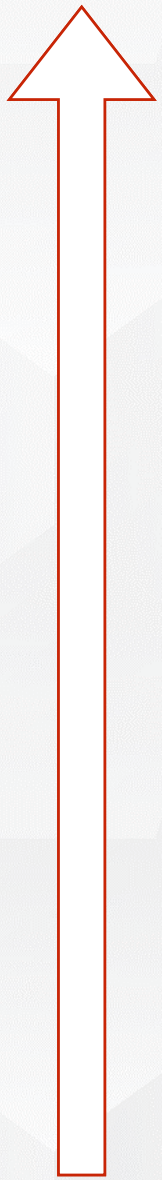
- Problem expression
  - {server:system.cpu.load.last()}>5
- Recovery expression
  - {server:system.cpu.load.last()}<=1

DIFFERENT CONDITIONS



{server:system.cpu.load.last()} > 5 ... {server:system.cpu.load.last()} <= 1

DIFFERENT CONDITIONS



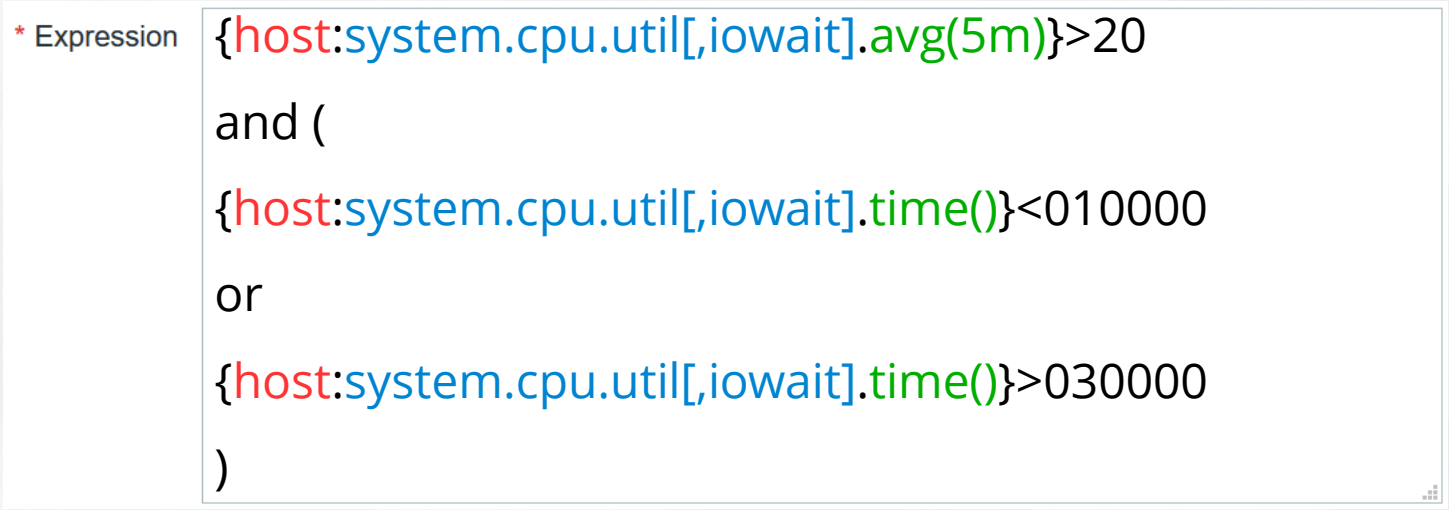
Simple

	TIMESTAMP	VALUE	
	2020-01-30 10:55:50	0.2	
	2020-01-30 10:55:20	0.8	OK
	2020-01-30 10:54:50	2.5	
OK	2020-01-30 10:54:20	4.7	-
PROBLEM	2020-01-30 10:53:50	10	-
OK	2020-01-30 10:53:20	4.6	-
PROBLEM	2020-01-30 10:52:50	5.2	-
OK	2020-01-30 10:52:20	2.4	-
PROBLEM	2020-01-30 10:51:50	5.3	PROBLEM
	2020-01-30 10:51:20	2.5	
	2020-01-30 10:50:50	1.2	
	2020-01-30 10:50:20	0.5	

Different conditions

The IO monitoring excluding backup hours (01:00-03:00)

- Flexible intervals (no values are collected at all)
  - 1-7,00:00-01:00
  - 1-7,03:00-24:00
- Zabbix maintenance (problems are detected, but no notifications are generated)
  - Maintenance is discussed in the upcoming topics
- Time trigger function (no problems generated)



Zabbix server time is used for calculation.

PRACTICAL SETUP

1. On the host "Training-VM-XX" create a new trigger:
    1) "CPU Load is high on {HOST.NAME}"
        - Problem: if load is more than 1
        - Severity: Warning
        - Operational data: use a macro to display threshold and problematic value
        - Use macros to show a host name and an IP address
        - Use {$USER.MACRO} for the threshold
        - Add a tag "Value" and use macro {ITEM.VALUE}
    2) "CPU Load is very high”
        - Problem: if load is more than 2
        - Severity: High
        - Other options the same as in the trigger 1.
        - Create a recovery expression to make trigger less sensitive
    3) Create a dependency: if trigger 2 fires, trigger 1 should be suppressed
2. Use "cat /dev/urandom | md5sum" command to test this setup.

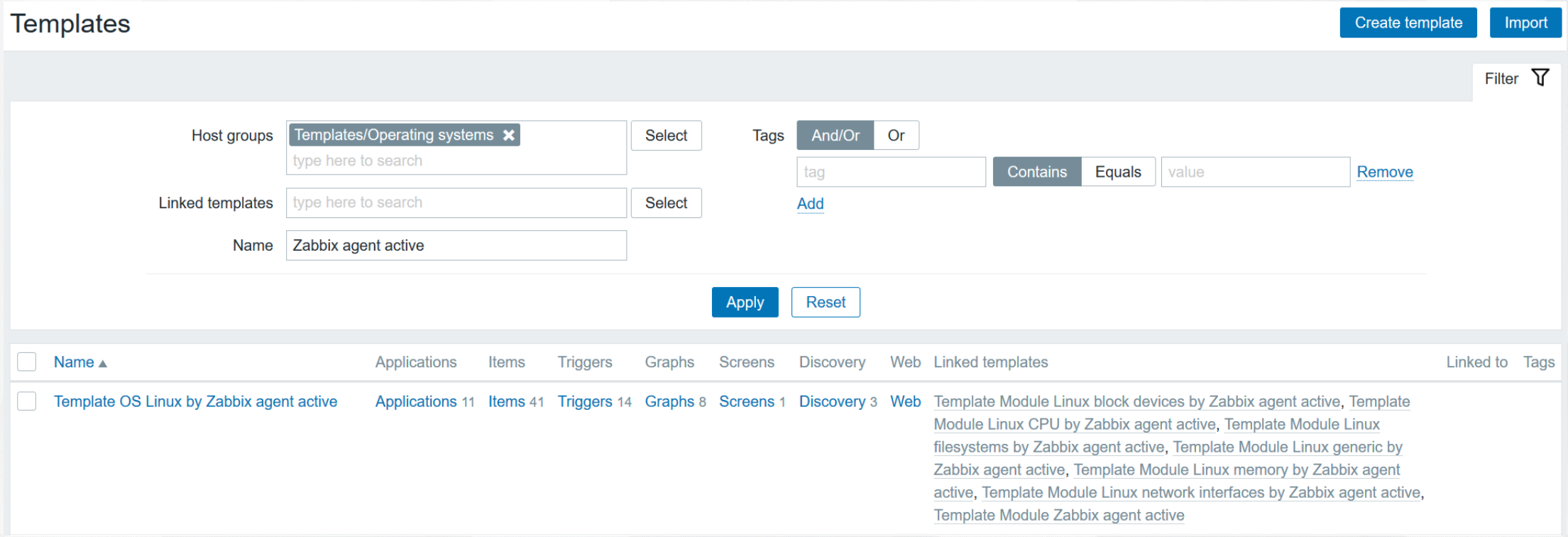
Working with templates

TEMPLATES - PROPERTIES AND CONTENTS

Template is a set of entities that can be applied to multiple hosts.

- Used to manage configuration
- Allows to easily set up and manage monitoring of hundreds or thousands of hosts

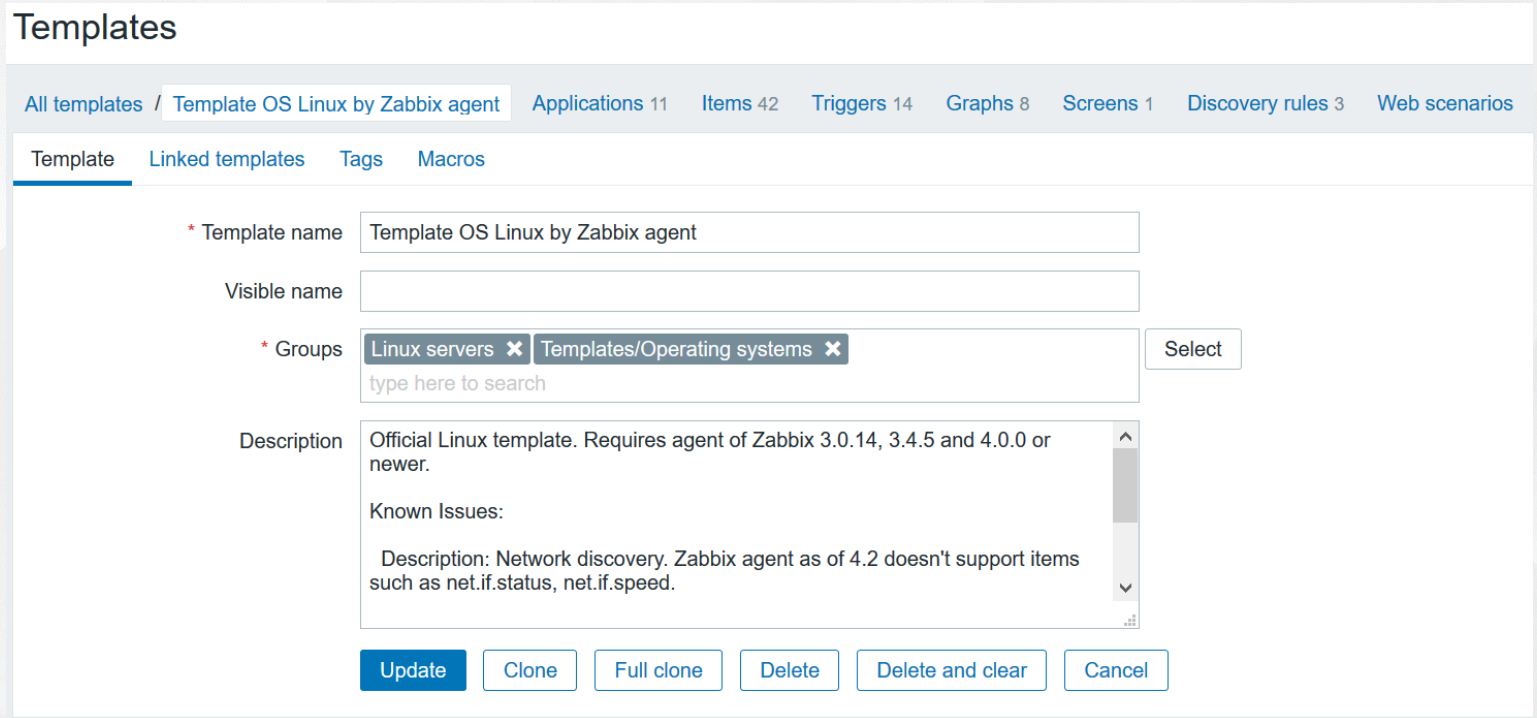
Configuration > Templates



https://www.zabbix.com/documentation/current/manual/config/templates

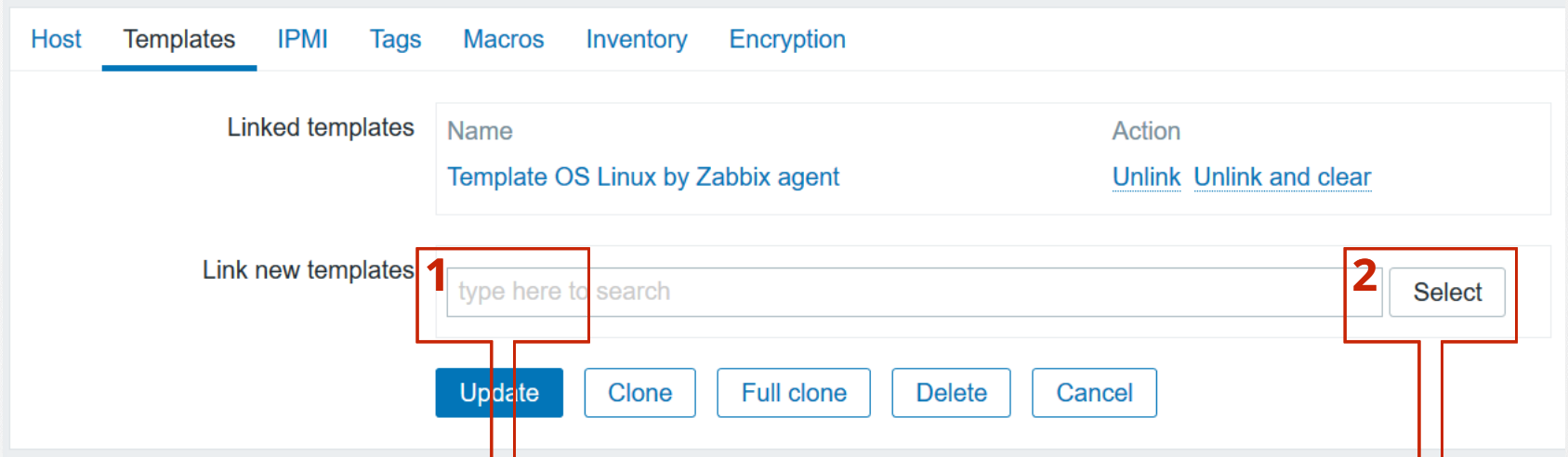
Configuration > Templates > [Create template]

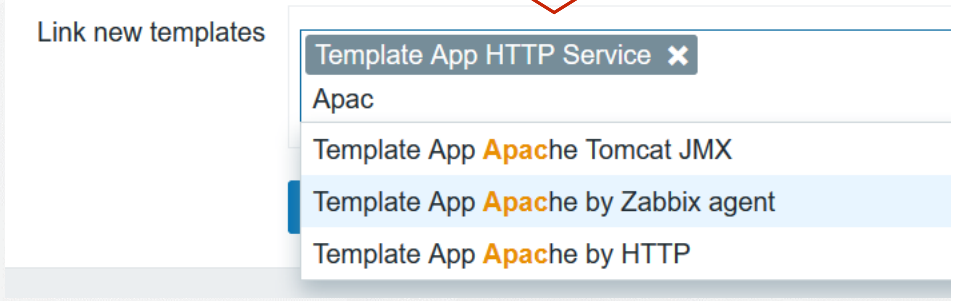
- Properties:
  - Name
  - Groups
  - Linked templates
  - Macros
  - Tags
- Entities:
  - Applications
  - Items
  - Triggers
  - Graphs
  - Templated screens
  - Discovery rules
  - Web scenarios

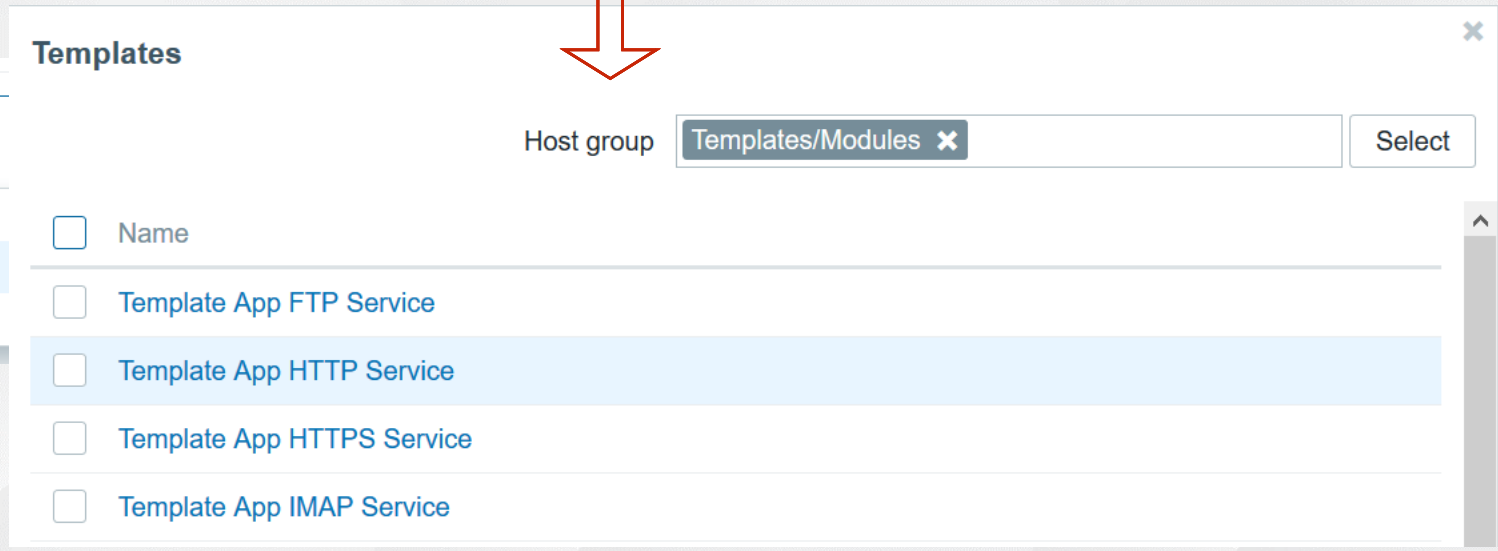


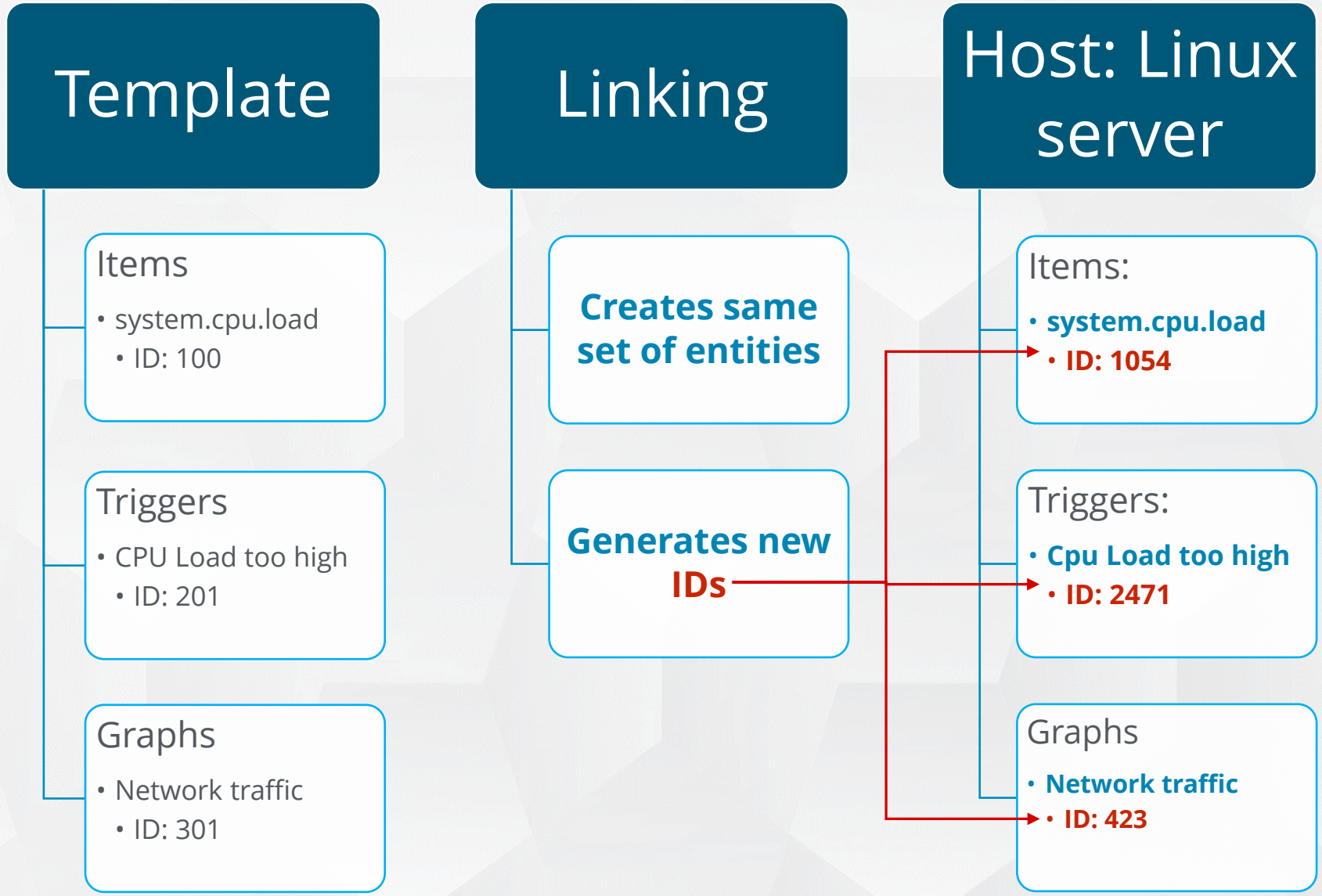
Some entity properties can be overridden on a host level!

Templates can be linked by using automatic lookup field (1) or a list selector (2)









If an entity exists on a host, linking will reuse and change it.

Visible in the list of hosts (the first two levels only):

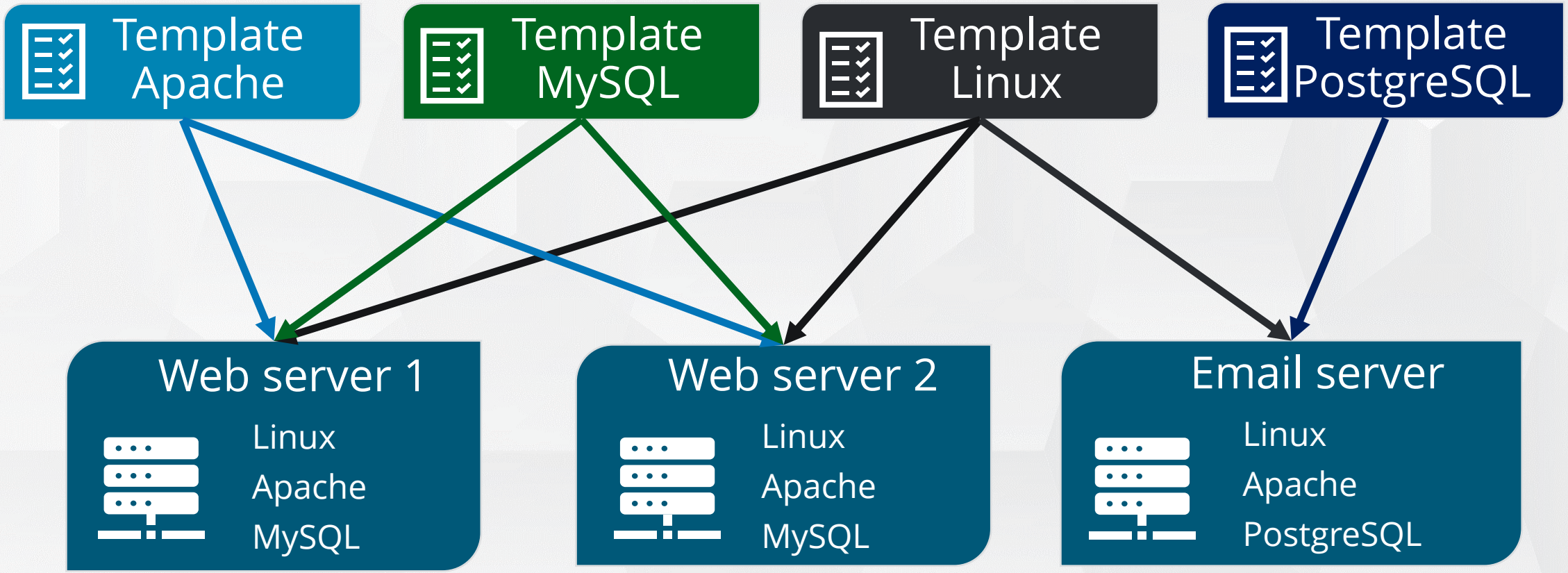
Name ▾	Applications	Items	Triggers	Graphs	Discovery	Web	Interface	Proxy	Templates
Zabbix server	Applications 18	Items 124	Triggers 58	Graphs 25	Discovery 3	Web	127.0.0.1: 10050		Template App Zabbix Server, Template OS Linux by Zabbix agent (Template Module Linux block devices by Zabbix agent, Template Module Linux CPU by Zabbix agent, Template Module Linux filesystems by Zabbix agent, Template Module Linux generic by Zabbix agent, Template Module Linux memory by Zabbix agent, Template Module Linux network interfaces by Zabbix agent, Template Module Zabbix agent)

Visible in the list of templates:

Template Net Mikrotik SNMPv2	Applications 8	Items 19	Triggers 11	Graphs 1	Screens	Discovery 4	Web	Template Module Generic SNMPv2, Template Module Interfaces SNMPv2	net.mikrotik.450g, net.mikrotik.912UAG-5HPnD, net.mikrotik.941-2nD, net.mikrotik.951G-2HnD, net.mikrotik.1100ahx2, net.mikrotik.CCR1016-12G, net.mikrotik.CCR1036-12G-4S, net.mikrotik.rb1100ah, net.mikrotik.rb2011uas-2hnd
Template Net Netgear Fastpath SNMPv2	Applications 9	Items 18	Triggers 8	Graphs 2	Screens	Discovery 4	Web	Template Module Generic SNMPv2, Template Module Interfaces SNMPv2	net.netgear_M5300-28G

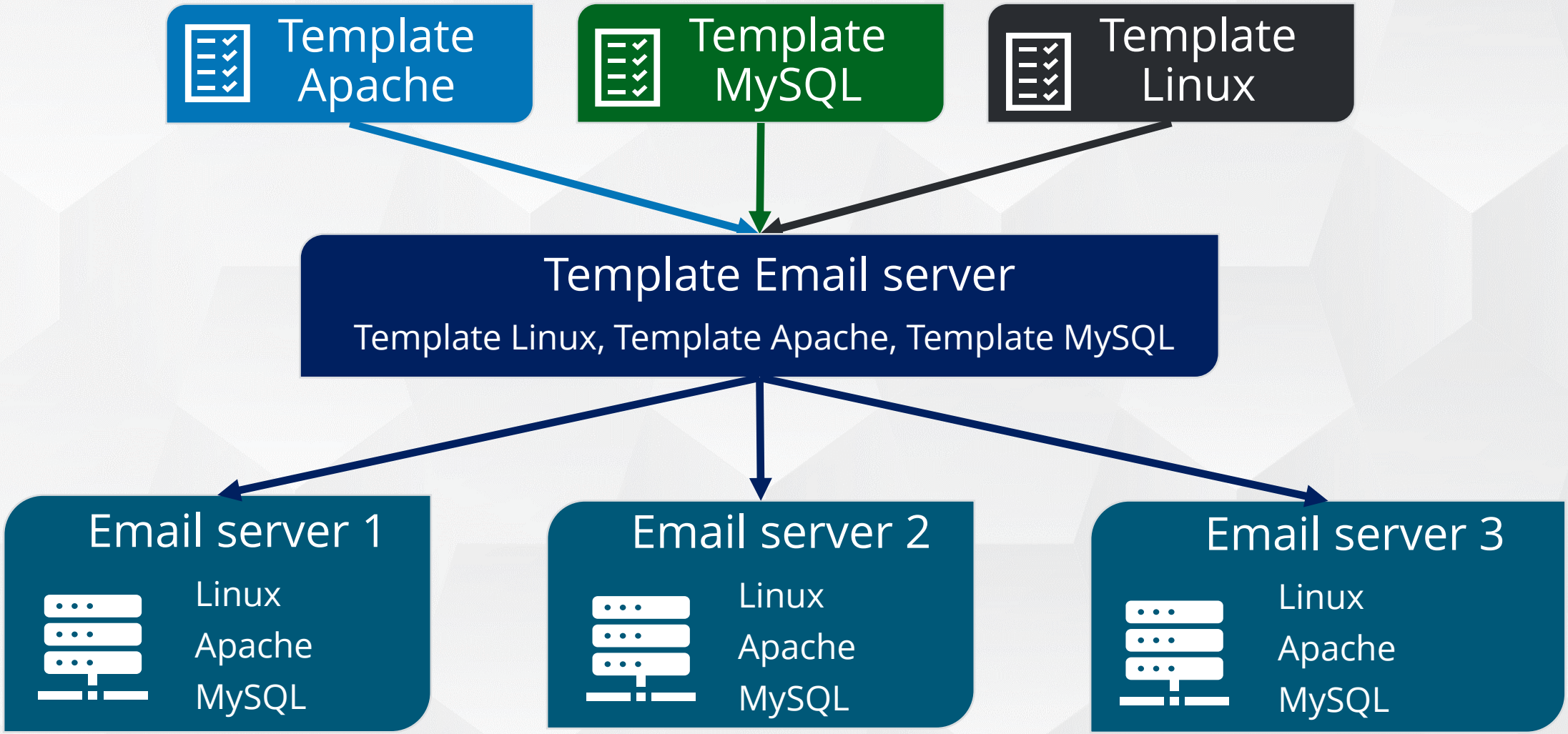
Multiple templates can be linked to a single host.

One template can be linked to many hosts.

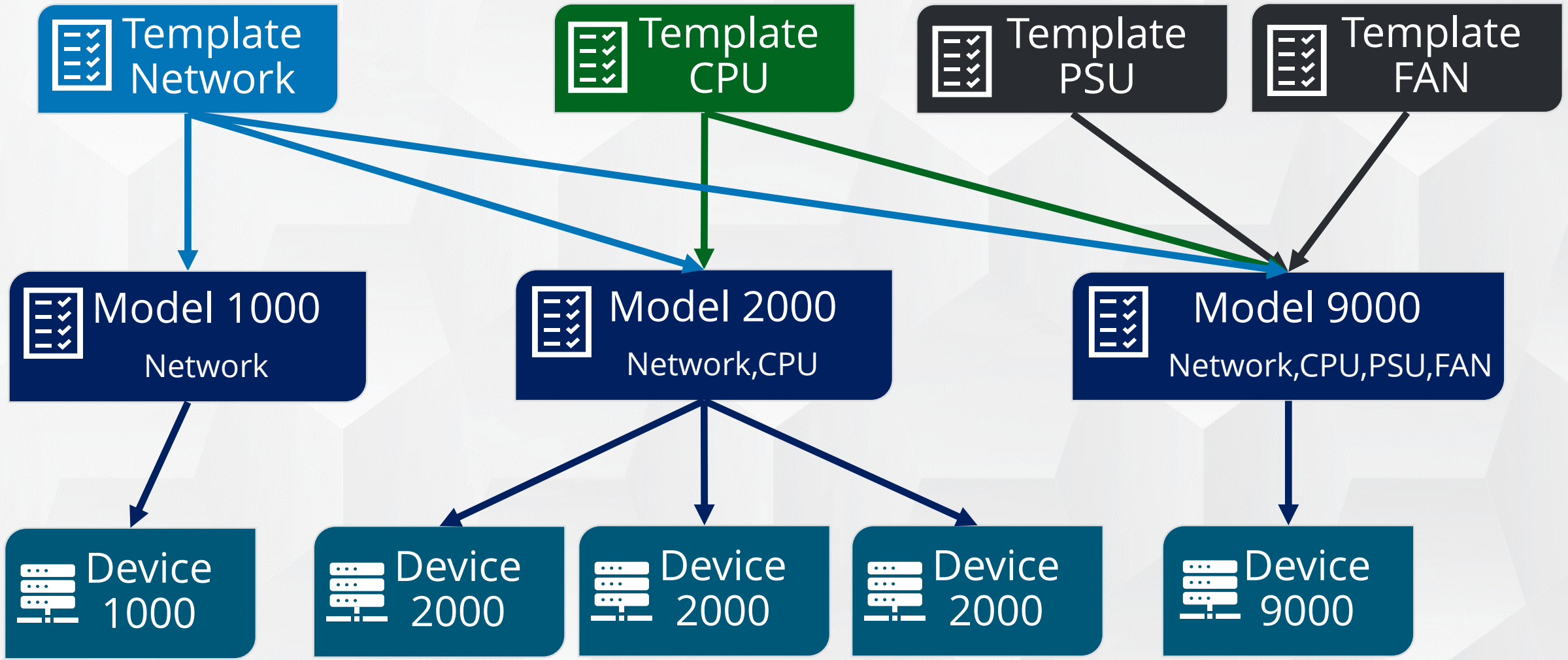


Templates are assigned to hosts directly, not to a host group.

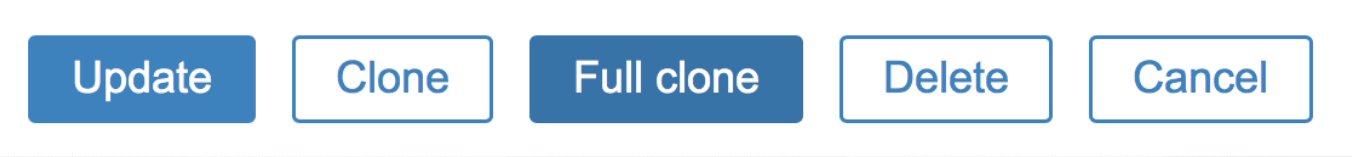
Container templates can be created to simplify management of similar hosts.



Templates can be used as building blocks to build larger templates.



You can use Clone and Full clone buttons in the form of an existing template or host to create a new one.



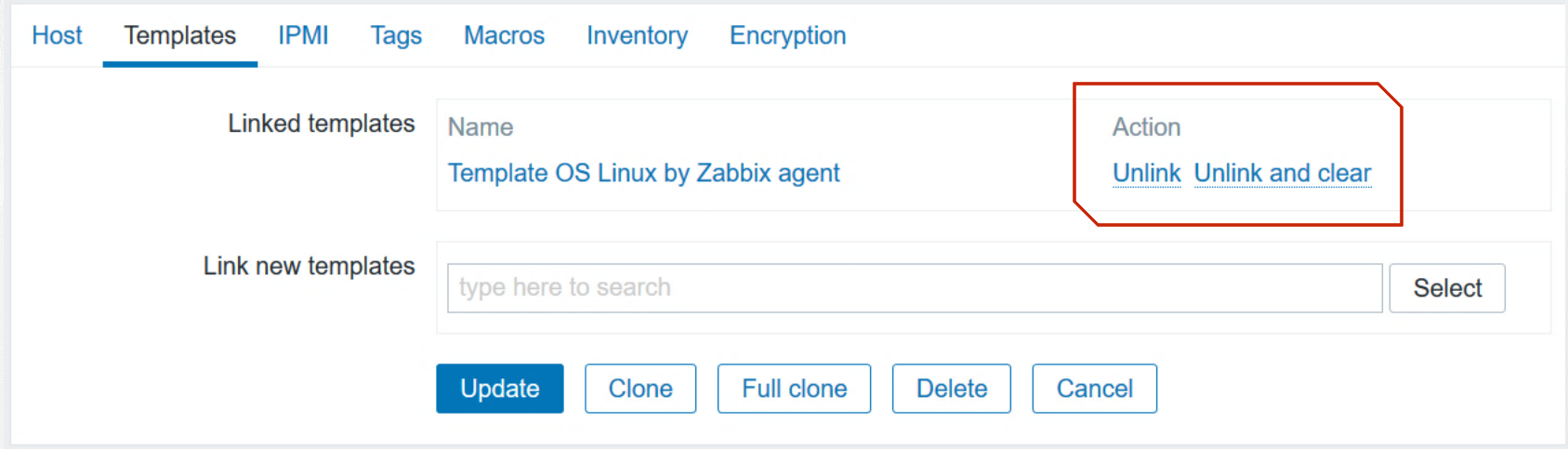
[Clone]	Will retain all parameters and linkage (e.g. keeping all entities from the templates)
[Full clone]	Full clone will additionally retain directly attached entities (applications, items, triggers, etc.)

When a host or template is cloned:

- It will retain all template/host entities as they are originally defined.
- Any changes made on a host level to entities coming from a template will not be cloned to a new host:
  - Update interval
  - History/trends storage period
  - etc.

When cloning macros with type "Secret text", value and type are reset.

There are two options to unlink templates:

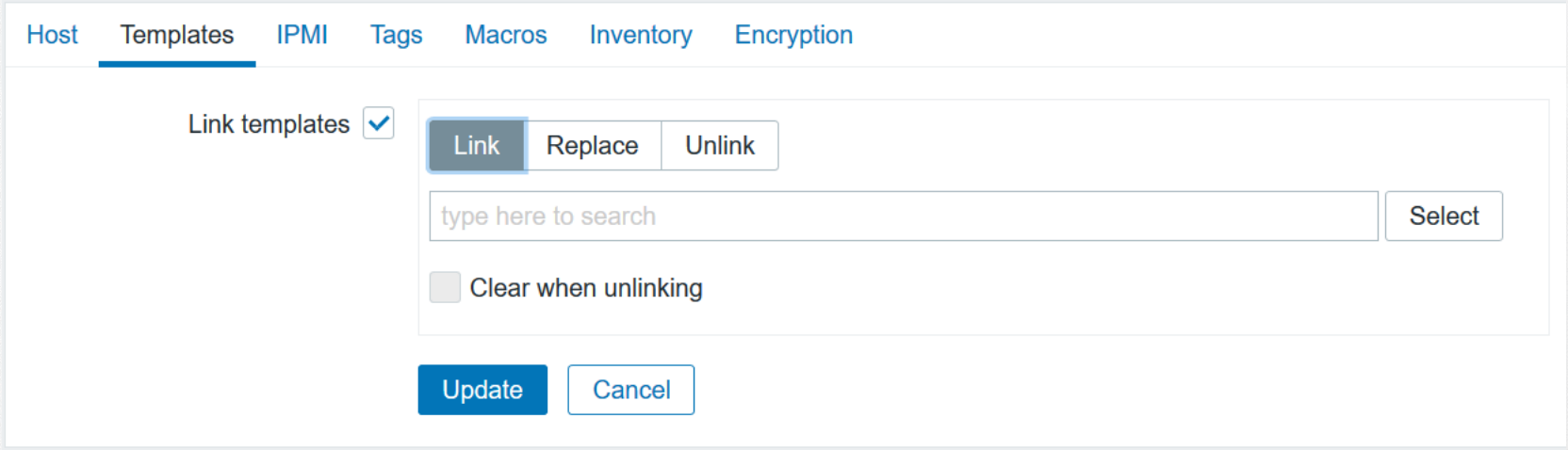


Unlink	Removes association with the template, but keeps all its entities (items, triggers, graphs etc.) with the host.
Unlink and clear	Removes both the association with the template and all its entities (items, triggers, graphs etc.).

TEMPLATES - MULTIPLE HOSTS LINKING/UNLINKING

Using [Mass update] it is possible to modify many hosts at once.

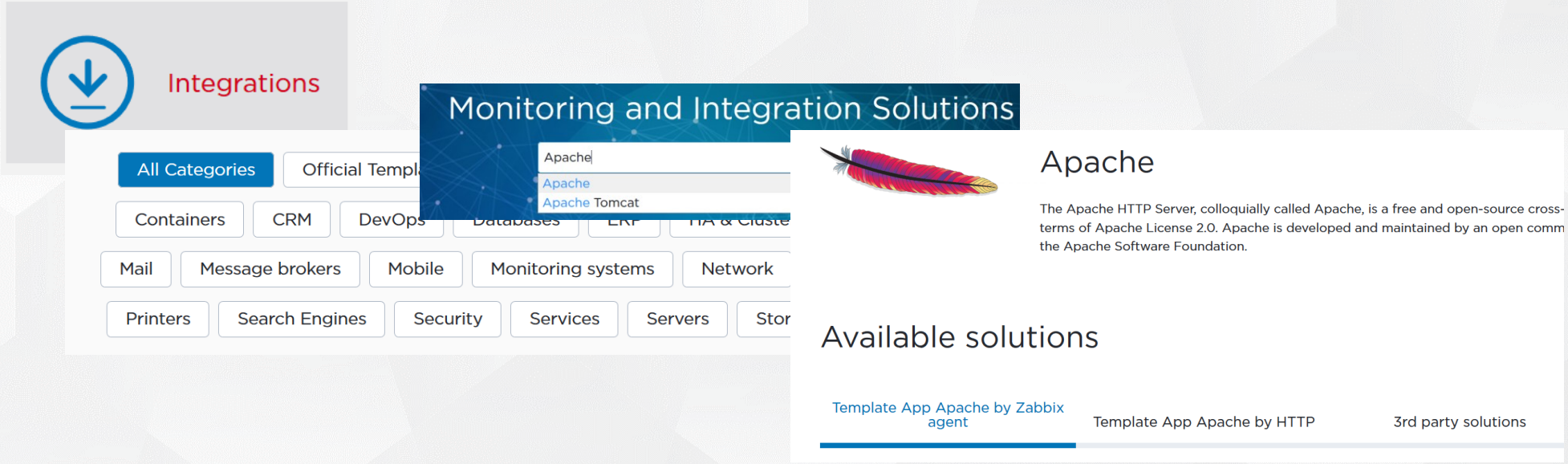
- Configuration > Hosts > Select some hosts by marking their checkboxes.



Link	Same as for a single host: auto-lookup field or selector to find templates and link.
Replace	Link a new template while unlinking any template that was linked to the hosts before.
Replace + Clear	Unlinks all templates and removes entities, links new one. If no template name provided - unlinks & removes all.
Unlink	Remove association with the template leaving all its entities with the host.
Unlink + Clear	Remove both the association with the template and all its entities.

WHERE TO GET TEMPLATES

- https://www.zabbix.com/integrations
  - Dedicated integration team



- https://share.zabbix.com/
- https://git.zabbix.com/projects/ZBX/repos/zabbix/browse/templates
- From a fresh Zabbix server installation
- Search the internet

PRACTICAL SETUP

1. Create a new template.
   - Host group: "Training/Templates"
   - Template: "Template Basic"
     - Tag: Environment Value: Training
2. Create two additional hosts for other trainees VMs.
   - Name Training-VM-XY and Training-VM-XZ (use neighbor student numbers)
   - Host group: "Training/Servers"
3. Copy all items and triggers from Training-VM-XX to the template.
4. Link the template to your own host and the new hosts.
5. Recreate applications and user macros.
6. Configure Zabbix agents to allow passive connections from other hosts and the trainer's host.

Advanced task: Create "Template Basic active" using active agent mode, monitor trainer host

User parameters

USER PARAMETERS ON ZABBIX AGENT

A simple way to run/check a script that does not come predefined out-of-the-box

- Must be configured for every Zabbix agent
  - Directly in zabbix_agentd.conf file
  - Included from zabbix_agentd.d directory (recommended)
- Syntax UserParameter=item.key[*],command
  - Simple
  - Flexible ([*] defines that key accepts parameters)

		item key	,	executed command
Simple	UserParameter=	mysql.qps	,	mysqladmin status | cut -f9 -d":"
Flexible	UserParameter=	calc[*]	,	echo "$1*$2" | bc

Escape variables in flexible parameters: awk '{print $$2}'

https://www.zabbix.com/documentation/5.0/manual/config/items/userparameters

USER PARAMETERS - NOTES

- Restart of agent is required when changing user parameters
- The return value of the command is standard output together with standard error
- User parameter must work fast, otherwise it timeouts
- Executed with "zabbix" user permissions
- Environment may not be preserved on some Unix systems
- Can be used as passive or active check
- Can be used to collect related metrics in bulk
- Preprocessing can be used to execute transformation rules for the received item values
- Certain symbols can not be passed as arguments by default
  - \ ' " ` * ? [ ] { } ~ $! & ; ( ) < > | # @
  - Additionally, newline characters are not allowed
  - Can be allowed by setting UnsafeUserParameters=1 in zabbix_agentd.conf file

PRACTICAL SETUP

1. Add simple and flexible user parameters:

```
UserParameter=mysql.uptime.s,mysqladmin -uzabbix -pP455w0RD status 2>/dev/null

UserParameter=mysql.uptime.f[*],mysqladmin -u$1 -p$2 status 2>/dev/null | grep Uptime | awk '{print $$2}'
```

2. Add corresponding items to the "Template Basic":
   - MySQL Uptime (simple) with preprocessing steps to extract Uptime
   - MySQL Uptime (flexible)
3. Use units:
   - uptime
   - !uptime
4. Make sure that the items receive data.

QUESTIONS?

Time for a break :)