



ZABBIX 5.0

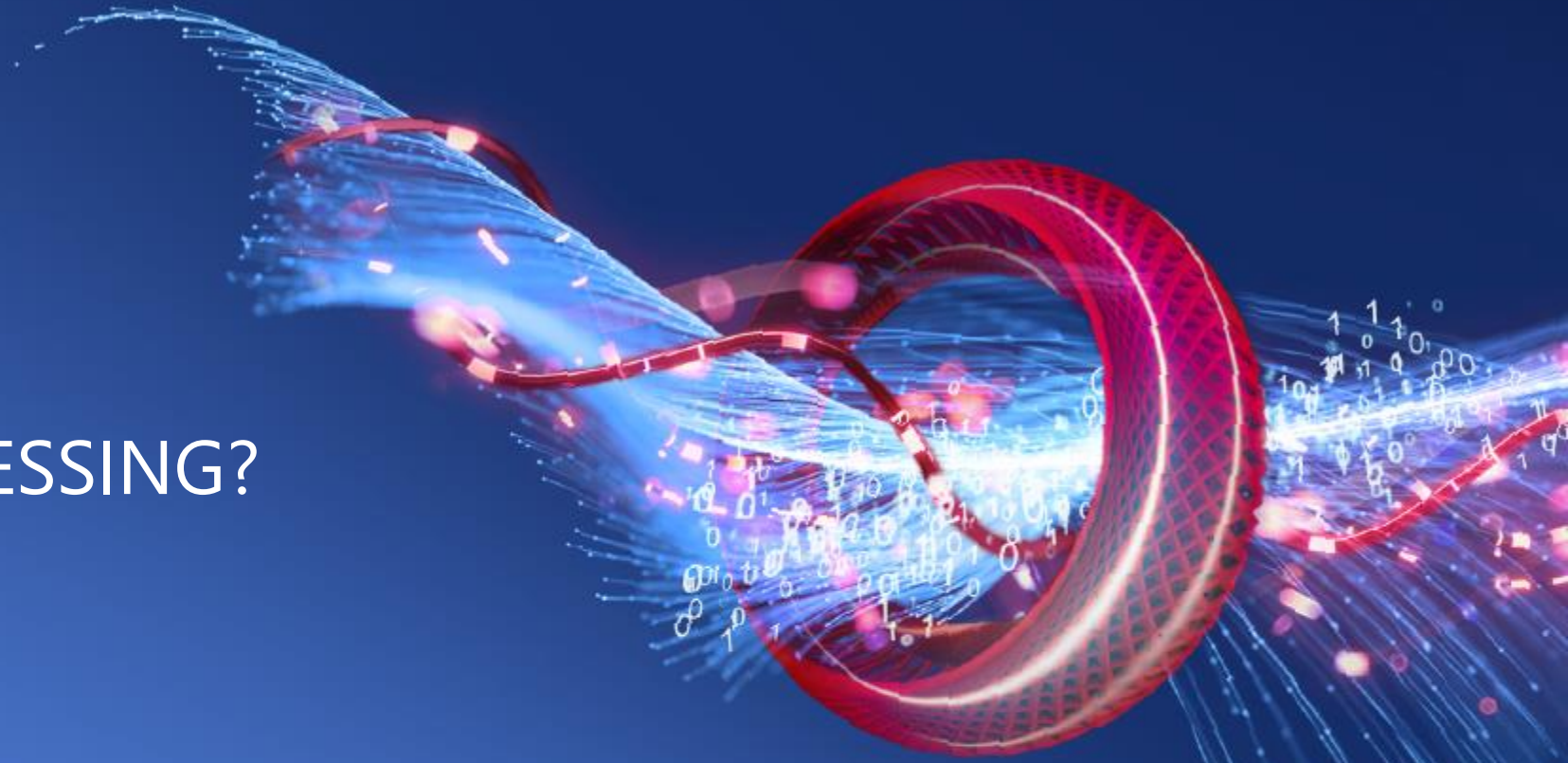
PREPROCESSING - HOW TO GET WHAT
YOU WANT TO GET WITH 5.0



Aleksandrs Petrovs-Gavrilovs
Technical Support Engineer

01

WHAT IS PREPROCESSING?



ZABBIX PREPROCESSING



WHAT IS PREPROCESSING?

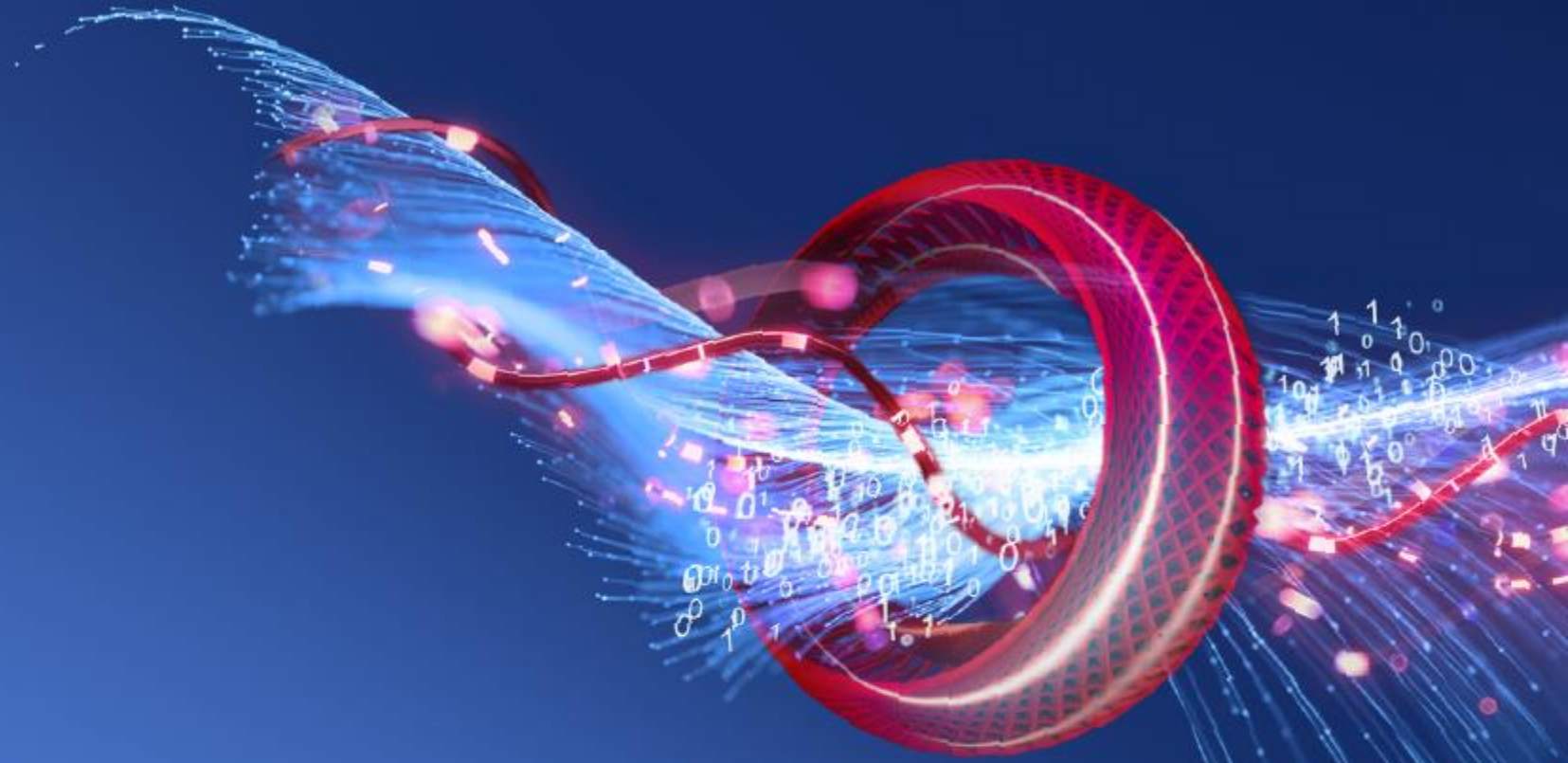
- ⊗ In computer science, a preprocessor is a program that processes its input data to produce output that is used as input to another program Both require tuning (e.g. buffer and transaction log) and is important.
- ⊗ Data preprocessing, used in learning and data collecting to make input data easier to work with.

WHAT IS PREPROCESSING?

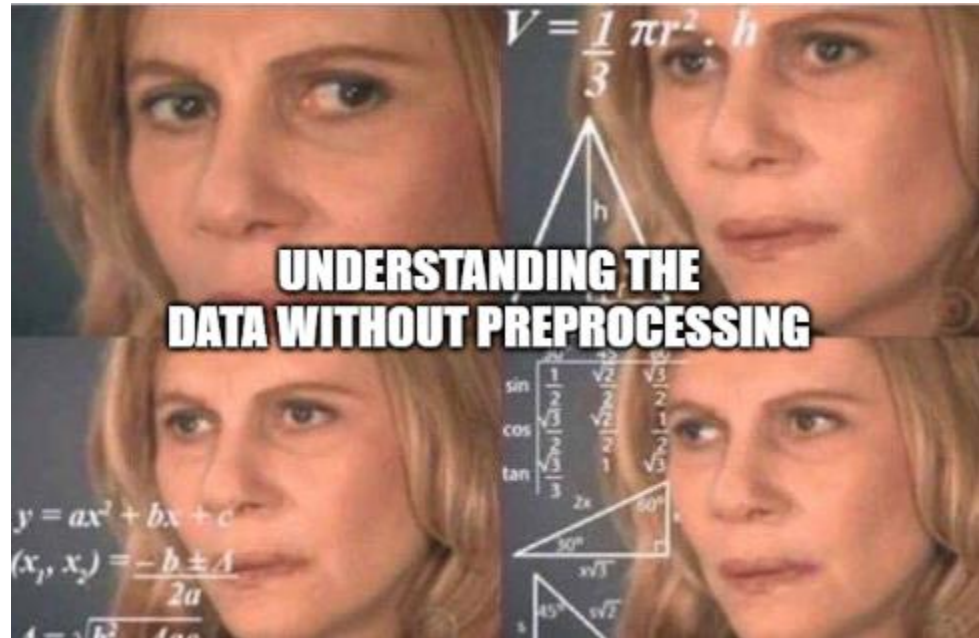
- ⊗ Preprocessing allows to define transformation rules for the received item values.
- ⊗ One or several transformations are possible before saving values to the database.
- ⊗ Transformations are executed in the order in which they are defined.
- ⊗ Preprocessing is done either by Zabbix server or by Zabbix proxy (for items monitored by proxy).

02

WHY DO WE NEED
PREPROCESSING?



WHY DO WE NEED **PREPROCESSING**?



WHY DO WE NEED **PREPROCESSING**?

- ④ Data cleansing
- ④ Data editing
- ④ Data reduction
- ④ Data wrangling

DATA CLEANSING?

- ④ Define a range by specifying minimum/maximum values (inclusive) .
- ④ Specify a regular expression that will look for an error or a value that must match or not match.
- ④ Check for an application-level error message located at JSONpath.
- ④ Check for an application-level error message located at xpath.

DEFINING A RANGE

Item Preprocessing

Preprocessing steps	Name	Parameters	Custom on fail	Actions
1:	In range	1	100	<input checked="" type="checkbox"/> Test Remove

Custom on fail **Discard value** Set value to Set error to

[Add](#) [Test all steps](#)

[Add](#) [Test](#) [Cancel](#)

Item Preprocessing

Preprocessing steps	Name	Parameters	Custom on fail	Actions
1:	In range	1	999	<input checked="" type="checkbox"/> Test Remove

Custom on fail **Discard value** Set value to Set error to 999

[Add](#) [Test all steps](#)

[Add](#) [Test](#) [Cancel](#)

USE A REGULAR EXPRESSION

Item Preprocessing

Preprocessing steps	Name	Parameters	Custom on fail	Actions
1:	Matches regular expression	(?i)error	<input checked="" type="checkbox"/>	Test Remove

Custom on fail **Discard value** Set value to Set error to

[Add](#) [Test all steps](#)

Add Test Cancel

Item Preprocessing

Preprocessing steps	Name	Parameters	Custom on fail	Actions
1:	Does not match regular expression	(?i)error	<input checked="" type="checkbox"/>	Test Remove

Custom on fail **Discard value** Set value to Set error to

[Add](#) [Test all steps](#)

Add Test Cancel

USE MORE **REGULAR EXPRESSIONS**

☑ `^(automatic|automatic delayed|manual|disabled)$`

Windows service startup states

☑ `([1-9]+)`

collect only numbers

☑ `((?>[a-z\-\0-9]{2,}\.){1,}[a-z]{2,8})(?:\s|/)`

domain name

☑ `([0-9A-Za-z'\&\-\.\V\(\)=:;]+)|((?:::|;|=)(?:-)?(?:\)|D|P))`

smiley faces

USE JSONPATH

Item Preprocessing

Preprocessing steps	Name	Parameters	Custom on fail	Actions
1:	Check for error in JSON	\$.myjson.error	<input checked="" type="checkbox"/>	Test Remove

Custom on fail: Discard value | Set value to | Set error to

[Add](#) [Test](#) [Cancel](#) [Test all steps](#)

Item Preprocessing

Preprocessing steps	Name	Parameters	Custom on fail	Actions
1:	Check for error in XML	/address/message/error	<input checked="" type="checkbox"/>	Test Remove

Custom on fail: Discard value | Set value to | Set error to | There is an error

[Add](#) [Test](#) [Cancel](#) [Test all steps](#)

DATA EDITING?

- ✓ Replace.
- ✓ Trim.
- ✓ Use regex.
- ✓ Multiply.
- ✓ Javascript.

REPLACE

Item Preprocessing

Preprocessing steps	Name	Parameters	Custom on fail	Actions	
1:	Replace	error	0	<input type="checkbox"/>	Test Remove

[Add](#)

[Add](#) [Test](#) [Cancel](#)

[Test all steps](#)

Item Preprocessing

Preprocessing steps	Name	Parameters	Custom on fail	Actions	
1:	Regular expression	^(error)	\1	<input type="checkbox"/>	Test Remove
2:	Replace	error	0	<input type="checkbox"/>	Test Remove

[Add](#)

[Add](#) [Test](#) [Cancel](#)

[Test all steps](#)

TRIM

Removes provided symbols on both sides (right/left) with **Trim**

t 36 C

Trim: tC

36

Remove provided symbols on right side with **Right trim**

t 36 F

Right trim: F

t 36

Remove provided symbols on left side with **Left trim**

t 36 K

Left trim: t

36 K

USE EVEN MORE **REGULAR EXPRESSIONS**

✓ `^(eth[0-9]$)|(^eth[0-9]:[1-9]$)`

Look for network adapters

✓ `^(ntfs|fat32|zfs)$`

Short filesystem regex

✓ `^(Physical memory|Virtual memory|Memory buffers|Cached memory|Swap space)$`

SNMP storage devices

MULTIPLY

Kilobytes to bytes

The screenshot shows the 'Preprocessing' configuration interface in Zabbix. It features a table with columns for 'Preprocessing steps', 'Name', 'Parameters', 'Custom on fail', and 'Actions'. A single step is configured with the name 'Custom multiplier' and a parameter value of '1024'. Below the table are several control buttons: 'Update', 'Clone', 'Execute now', 'Test', 'Clear history and trends', 'Delete', and 'Cancel'. There are also links for 'Add', 'Test', 'Remove', and 'Test all steps'.

Preprocessing steps	Name	Parameters	Custom on fail	Actions
1:	Custom multiplier	1024	<input type="checkbox"/>	Test Remove

[Add](#)

[Test all steps](#)

[Update](#) [Clone](#) [Execute now](#) [Test](#) [Clear history and trends](#) [Delete](#) [Cancel](#)

Milliseconds to seconds

The screenshot shows the 'Preprocessing' configuration interface in Zabbix, similar to the one above. In this case, the parameter value is '0.001'. The layout and controls are identical to the previous screenshot.

Preprocessing steps	Name	Parameters	Custom on fail	Actions
1:	Custom multiplier	0.001	<input type="checkbox"/>	Test Remove

[Add](#)

[Test all steps](#)

[Update](#) [Clone](#) [Execute now](#) [Test](#) [Clear history and trends](#) [Delete](#) [Cancel](#)

Note that if the item type of information is *Numeric (unsigned)*, incoming values with a fractional part will be trimmed (i.e. '0.9' will become '0') before the custom multiplier is applied.

USE JAVASCRIPT

Get time till the end of certificate in seconds

```
var split = value.split(' '),
MONTHS_LIST = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct',
'Nov', 'Dec'],
month_index = ('0' + (MONTHS_LIST.indexOf(split[0]) + 1)).slice(-2),
ISOdate = split[3] + '-' + month_index + '-' + split[1] + 'T' + split[2],
now = Date.now();
return parseInt((Date.parse(ISOdate) - now) / 1000);
```

Test item

Value Time

Previous value Prev. time

End of line sequence

Name	Result
1: JavaScript	44380233

USE MORE JAVASCRIPT

Convert SNMP DateandTime to Timestamp

```
'use strict';
var str = value;
alert("str: " + str);
var y256 = str.slice(0,2); var y = str.slice(3,5); var m = str.slice(6,8);
var d = str.slice(9,11); var h = str.slice(12,14); var min = str.slice(15,17);
var y256Base10 = +("0x" + y256);
var yBase10 = +("0x" + y);
var Year = 256*y256Base10 + yBase10;
var mBase10 = +("0x" + m);
var dBase10 = +("0x" + d);
var hBase10 = +("0x" + h);
var minBase10 = +("0x" + min);
var YR = String(Year); var MM = String(mBase10); var DD = String(dBase10);
var HH = String(hBase10);
var MIN = String(minBase10);
if (mBase10 < 10) MM = "0" + MM; if (dBase10 < 10) DD = "0" + DD;
if (hBase10 < 10) HH = "0" + HH; if (minBase10 < 10) MIN = "0" + MIN;
var Date = YR + "-" + MM + "-" + DD + " " + HH + ":" + MIN;
return Date;
```

USE MORE JAVASCRIPT

Convert SNMP DateandTime to Timestamp

Test item

Value Time

Previous value Prev. time

End of line sequence

Preprocessing steps

Name	Result
1: JavaScript	2020-09-17 13:22

DATA REDUCTION?

- ✓ XML XPath.
- ✓ JSON Path.
- ✓ Use regex.
- ✓ Throttling.
- ✓ Replace

XML XPATH

Prepare an XML

```
<rhs_check>
  <network>
    <ping host="cdn.redhat.com">passed</ping>
    <reverse_dns>passed</reverse_dns>
    <port address="80">passed</port>
    <port address="8080">passed</port>
    <port address="5671">passed</port>
    <port address="443">passed</port>
    <port address="8140">passed</port>
    <port address="9090">passed</port>
  </network>
  <services>
    <service name="mongod">active</service>
    <service name="qpidd">active</service>
    <service name="qdrouterd">active</service>
    <service name="tomcat">active</service>
    <service name="pulp_resource_manager">active</service>
    <service name="pulp_workers">active</service>
    <service name="httpd">active</service>
    <service name="puppet">inactive</service>
    <service name="elasticsearch">unknown</service>
  </services>
</rhs_check>
```

XML XPATH

Find a service you are looking for:

```
/rhs_check/services/service[1]
```

Add some more steps:

The screenshot shows a 'Test item' configuration window with the following details:

- Value:** `<rhs_check>...`
- Time:** `now`
- Previous value:** (empty)
- Prev. time:** (empty)
- End of line sequence:** `LF` (selected), `CRLF`
- Preprocessing steps:**

Name	Result
1: XML XPath	<code><service name="mongod">active</service></code>
2: Regular expression	<code>active</code>
3: Replace	<code>1</code>
- Result:**

Result converted to Numeric (unsigned)	<code>1</code>
Result with value map applied	<code>Up</code>

Buttons: `Test`, `Cancel`

THE PATH OF THE JSON

```
{ "obj2testobj2Connection": "OK", "objam.testHttpConnection": "OK",  
  "obj2.testobj2ServerComponents": "OK", "tomcat.freememory.threshold": "2132131000",  
  "tomcat.freememory": "OK", "tomcat.freememory.actual": "13123131121",  
  "objam.testNFSSStore": "OK" }
```

The screenshot shows the 'Preprocessing' configuration window. It features a table with columns for 'Preprocessing steps', 'Name', 'Parameters', 'Custom on fail', and 'Actions'. A single step is configured with the name 'JSONPath' and the parameter '\$.["tomcat.freememory.actual"]'. Below the table are buttons for 'Update', 'Clone', 'Test', 'Delete', and 'Cancel'. To the right of the table are links for 'Test', 'Remove', and 'Test all steps'.

Preprocessing steps	Name	Parameters	Custom on fail	Actions
1:	JSONPath	\$.["tomcat.freememory.actual"]	<input type="checkbox"/>	Test Remove

[Add](#)

[Update](#) [Clone](#) [Test](#) [Delete](#) [Cancel](#)

[Test all steps](#)

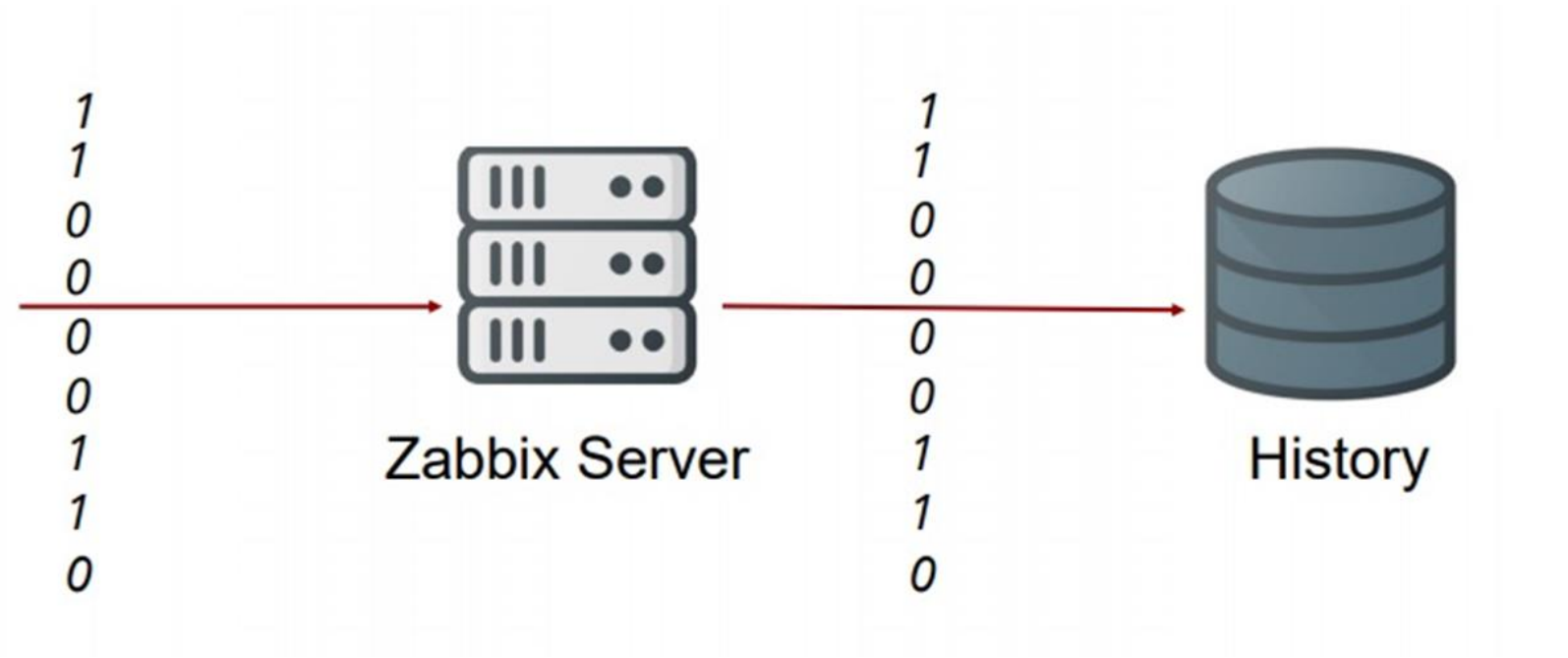
The screenshot shows the 'Test item' dialog box. It contains fields for 'Value' (with a truncated JSON string), 'Time' (set to 'now'), 'Previous value', and 'Prev. time'. There are buttons for 'End of line sequence' (LF and CRLF) and a 'Preprocessing steps' table. At the bottom are 'Test' and 'Cancel' buttons.

Value: { "obj2testobj2Connection": "OK", "objam.testHttpConnection": "OK", "obj2.t...
Time: now
Previous value:
Prev. time:
End of line sequence: LF CRLF
Preprocessing steps:

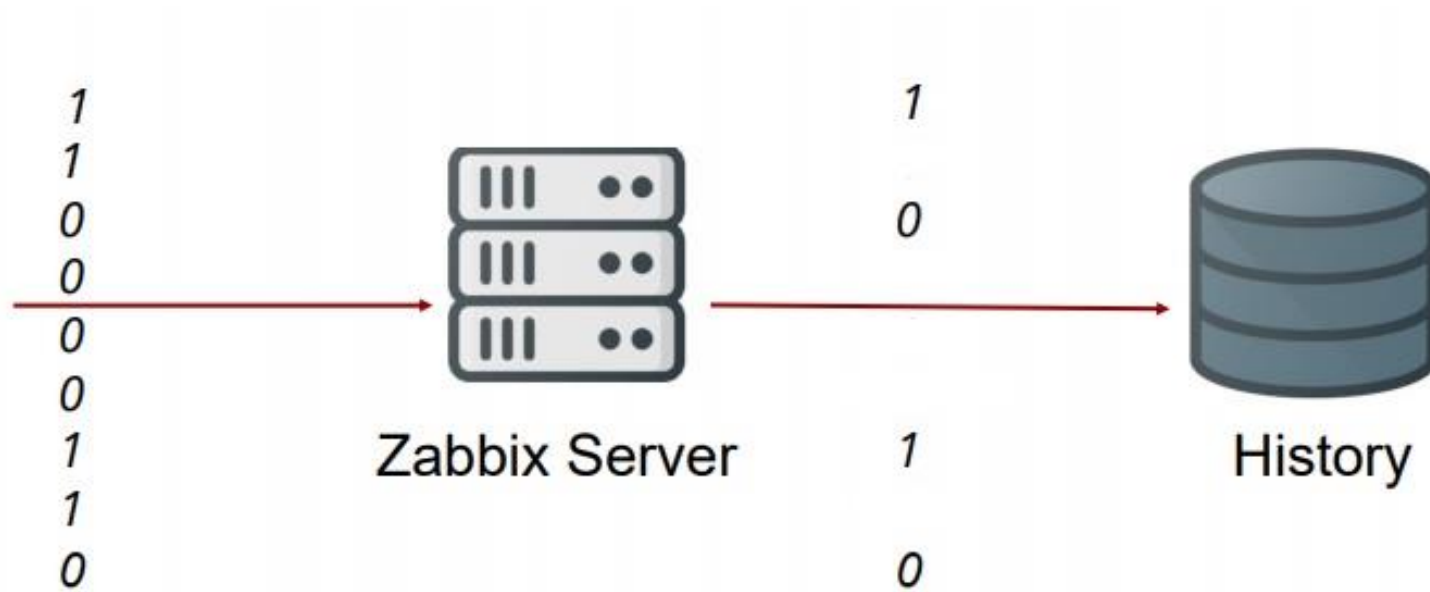
Name	Result
1: JSONPath	13123131121

[Test](#) [Cancel](#)

BEFORE THROTTLING



BEFORE THROTTLING



REPLACE

Item Preprocessing

Preprocessing steps	Name	Parameters	Custom on fail	Actions
1:	Replace	Up	<input type="checkbox"/>	Test Remove
2:	Replace	Down	<input type="checkbox"/>	Test Remove
Add				Test all steps

[Update](#) [Clone](#) [Test](#) [Delete](#) [Cancel](#)

REPLACE

Test item

Value Time

Previous value Prev. time

End of line sequence

Preprocessing steps	
Name	Result
1: Replace	1
2: Replace	1

Result

Test item

Value Time

Previous value Prev. time

End of line sequence

Preprocessing steps	
Name	Result
1: Replace	Down
2: Replace	0

Result

DATA WRANGLING?

- ✓ Boolean/Octal/Hexadecimal to decimal.
- ✓ CSV to JSON.
- ✓ Prometheus pattern
- ✓ Simple change
- ✓ Change per second

BOOLEN TO DECIMAL

The image displays two screenshots of a 'Test item' configuration window, illustrating the 'Boolean to decimal' preprocessing step.

Top Screenshot:

- Value:** true
- Time:** now
- Previous value:** (empty)
- Prev. time:** (empty)
- End of line sequence:** LF (selected), CRLF
- Preprocessing steps:**

Name	Result
1: Boolean to decimal	1
- Buttons:** Test, Cancel

Bottom Screenshot:

- Value:** false
- Time:** now
- Previous value:** (empty)
- Prev. time:** (empty)
- End of line sequence:** LF (selected), CRLF
- Preprocessing steps:**

Name	Result
1: Boolean to decimal	0
- Buttons:** Test, Cancel

TRUE - true, t, yes, y, on, up, running, enabled, available

FALSE - false, f, no, n, off, down, unused, disabled, unavailable

OCTAL/HEXADECIMAL TO DECIMAL

Test item

Value Time

Previous value Prev. time

End of line sequence

Preprocessing steps

Name	Result
1: Octal to decimal	1024

Test item

Value Time

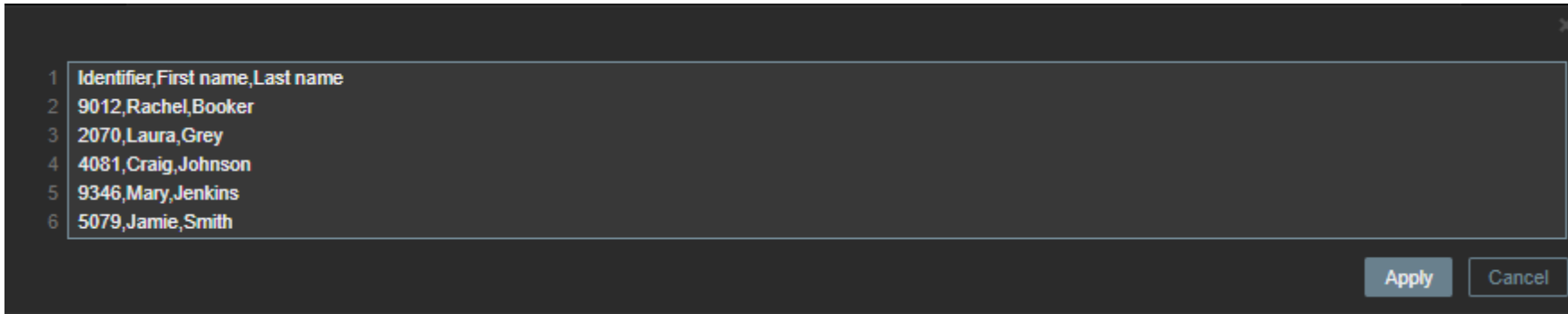
Previous value Prev. time

End of line sequence

Preprocessing steps

Name	Result
1: Hexadecimal to decimal	2020

CSV TO JSON



```
[{"Identifier":"9012","First name":"Rachel","Last name":"Booker"},  
{"Identifier":"2070","First name":"Laura","Last name":"Grey"},  
{"Identifier":"4081","First name":"Craig","Last name":"Johnson"},  
{"Identifier":"9346","First name":"Mary","Last name":"Jenkins"},  
{"Identifier":"5079","First name":"Jamie","Last name":"Smith"}]
```

CSV TO JSON

Test item

Value Time

Previous value Prev. time

End of line sequence LF CRLF

Preprocessing steps

Name	Result
1: CSV to JSON	<code>{{"Identifier":"9012","First name":"Rachel","Last name":"Booker"},"{"Identifier":"2070"...</code>
2: JSONPath	9012

Result 9012

PROMETHEUS PATTERN

```
# HELP wmi_logical_disk_free_bytes Free space in bytes
(LogicalDisk.PercentFreeSpace)
# TYPE wmi_logical_disk_free_bytes gauge
wmi_logical_disk_free_bytes{volume="C:"} 3.5180249088e+11
wmi_logical_disk_free_bytes{volume="D:"} 2.627731456e+09
wmi_logical_disk_free_bytes{volume="HarddiskVolume4"}
4.59276288e+08

wmi_service_state{name="dhcp",state="continue pending"} 0
wmi_service_state{name="dhcp",state="pause pending"} 0
wmi_service_state{name="dhcp",state="paused"} 0
wmi_service_state{name="dhcp",state="running"} 1
wmi_service_state{name="dhcp",state="start pending"} 0
wmi_service_state{name="dhcp",state="stop pending"} 0
wmi_service_state{name="dhcp",state="stopped"} 0
wmi_service_state{name="dhcp",state="unknown"} 0
```

PROMETHEUS PATTERN

Item Preprocessing

Preprocessing steps	Name	Parameters	Custom on fail	Actions
1:	Prometheus pattern	wmi_service_state{name="dhcp"}	<label name>	<input type="checkbox"/>

[Add](#)

[Update](#) [Clone](#) [Test](#) [Delete](#) [Cancel](#)

[Test](#) [Remove](#)
[Test all steps](#)

Test item

Value: # HELP wmi_logical_disk_free_bytes Free space in bytes (LogicalDisk.Perc... [↗](#) Time: now

Previous value: [↗](#) Prev. time:

End of line sequence: LF CRLF

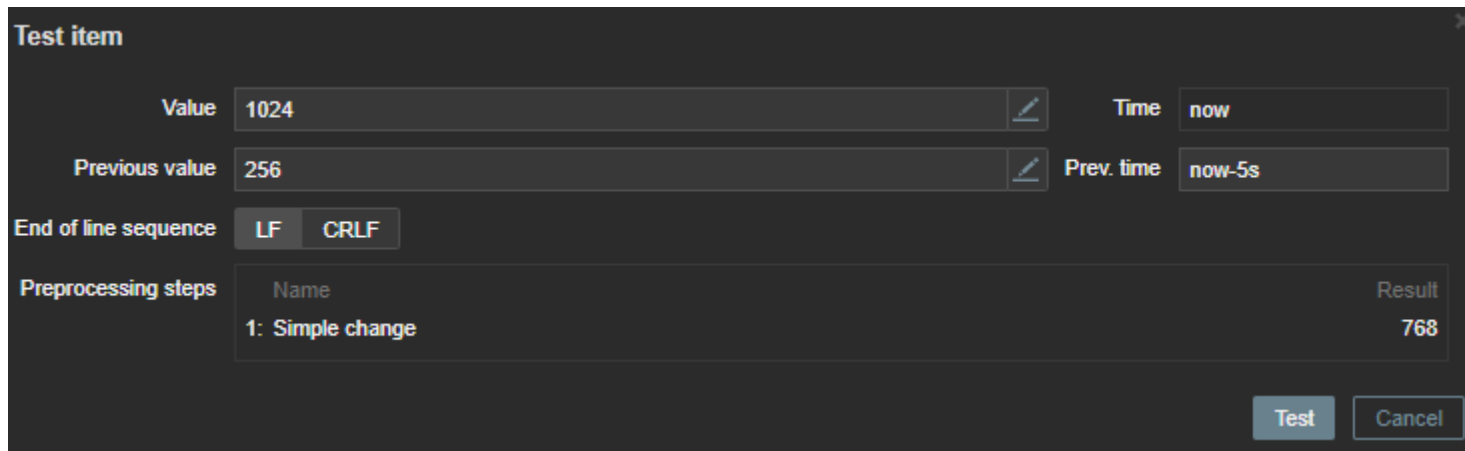
Preprocessing steps

Name	Result
1: Prometheus pattern	1

[Test](#) [Cancel](#)

SIMPLE CHANGE

- Simple change - Calculate difference between the current and previous value. Evaluated as **value-prev_value**, where value - current value; prev_value - previously received value. This setting can be useful to measure a constantly growing value.



The screenshot shows a 'Test item' configuration window with the following fields and values:

- Value: 1024
- Time: now
- Previous value: 256
- Prev. time: now-5s
- End of line sequence: LF (selected), CRLF
- Preprocessing steps table:

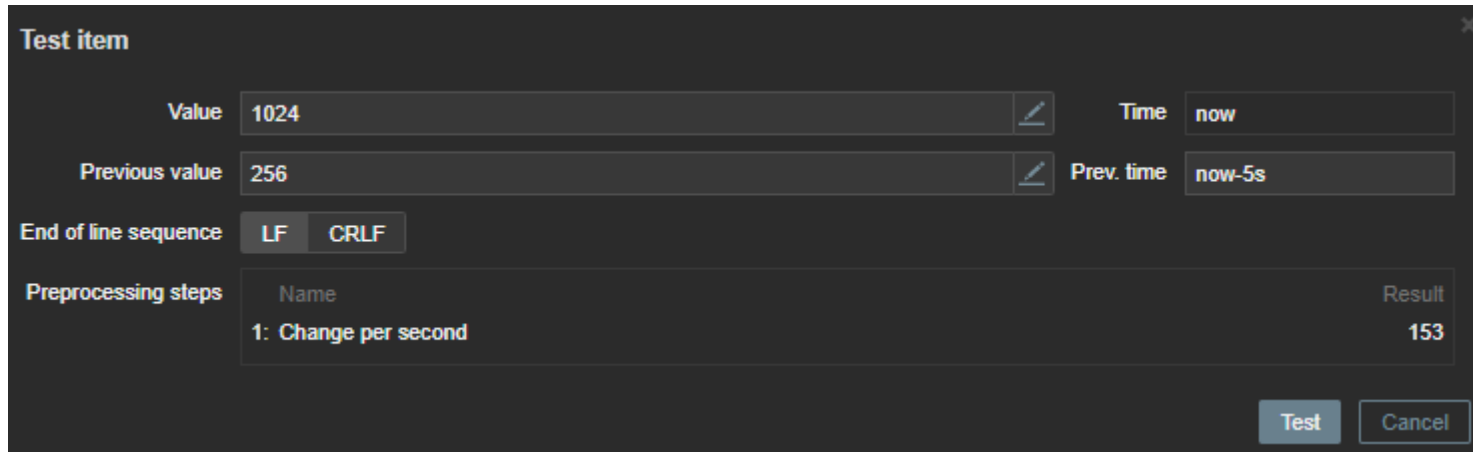
Name	Result
1: Simple change	768

Buttons: Test, Cancel

CHANGE PER SECOND

- ⊗ Change per second - Calculate the value change (difference between the current and previous value) speed per second.

Evaluated as **(value-prev_value)/(time-prev_time)**, where value - current value; prev_value - previously received value; time - current timestamp; prev_time - timestamp of previous value.



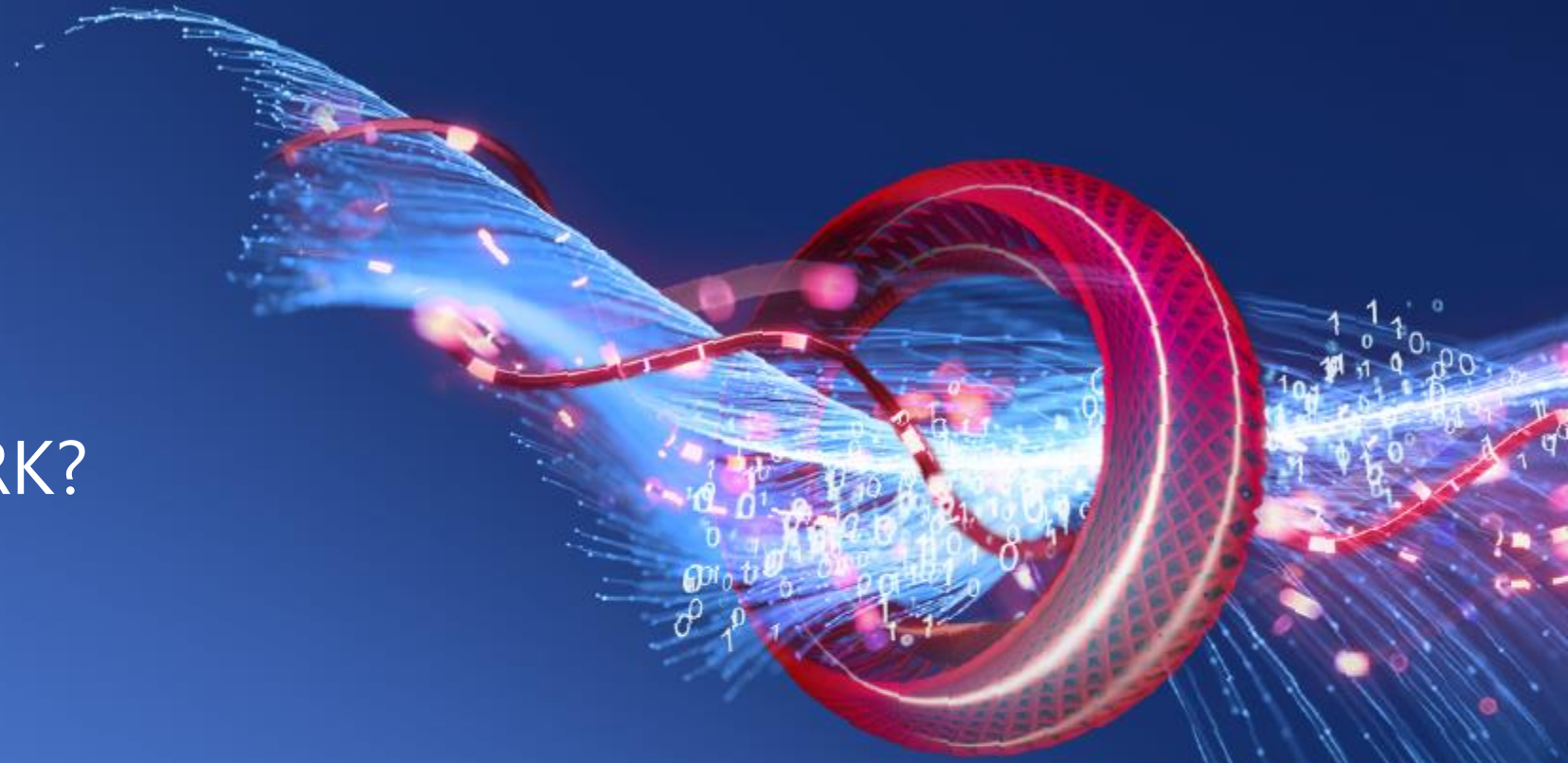
The screenshot shows a 'Test item' configuration window with the following fields and values:

Field	Value
Value	1024
Previous value	256
Time	now
Prev. time	now-5s
End of line sequence	LF CRLF
Preprocessing steps	1: Change per second
Result	153

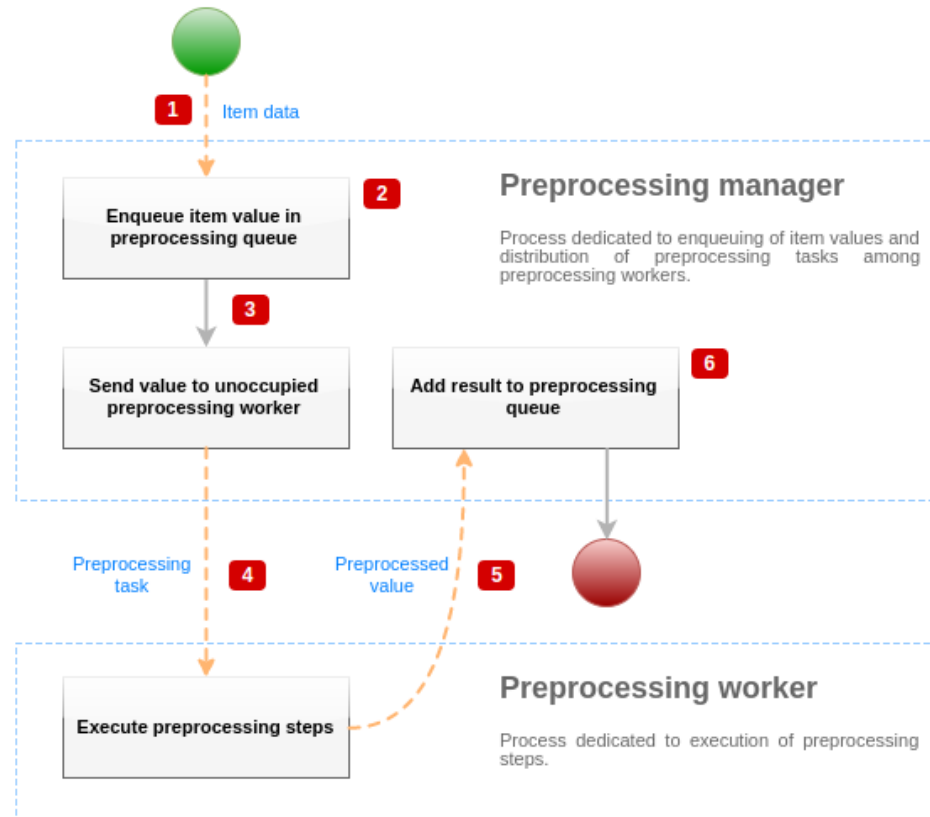
Buttons: Test, Cancel

03

HOW DOES IT WORK?



HOW IT WORKS?



Item can be placed at the end or at the beginning of the preprocessing queue. Zabbix internal items are always placed at the beginning of preprocessing queue, while other item types are enqueued at the end.

5.0 IMPROVEMENTS

- ✔ Maximum count of dependent items for one master item is now 29999 (was 999)
- ✔ [\[ZBX-17694\]](#) Memory usage became significantly lower
15K dependent items with master worth of 209 KB needed around 3 GB RAM with total busy time in manager of 2.115940 seconds, but after changes 4.6 MB is the peak and busy time is 0.688623 seconds.
- ✔ [\[ZBX-17720\]](#) Exclude disabled dependant items from preprocessing configuration
- ✔ [\[ZBX-17548\]](#) Don't store history in Proxy DB If History storage period is 0

Thank you!

The background features a vibrant blue gradient. A complex, glowing digital structure is composed of blue and red lines, resembling a fiber optic network or data stream. A prominent feature is a large, red, ring-like structure that appears to be a torus or a similar geometric shape, with a grid-like texture. Binary code (0s and 1s) is scattered throughout the scene, particularly around the red ring and in the upper right quadrant. The overall aesthetic is high-tech and futuristic.