# Netwatch

## Summary

Netwatch monitors the state of hosts on the network. Monitoring can be done with the following probe types:
1) ICMP - pings to a specified IP address - hosts, with an option to adjust threshold values
2) Simple - uses ping, without use of advanced metrics
3) TCP conn, to test the TCP connection
4) HTTP GET/HTTPS GET, request against a server you are monitoring

For each entry in the Netwatch table, you can specify an IP address, ping interval, and console scripts. The main advantage of Netwatch is its ability to issue arbitrary console commands on host state changes.

Since 7.4, Netwatch functionality has been expanded, prior versions only support simple ICMP probes. While upgrading to the new version, old Netwatch entries will be unchanged, reporting probe type "simple" - preserving the same functionality.

Default Netwatch values are always used - **even if they were not defined** by the user. Make sure to check the "status" page of the probe to see if the default thresholds are appropriate for your use case. Default threshold values can be found under the "probe options" section on this page.

## Properties

**Sub-menu:** `/tool/netwatch`

| Property | Description |
|---|---|
| **host** (Default:"") | The IP address of the server to be probed. Formats:<br><br>- *ipv4*<br>- *ipv4@vrf*<br>- *ipv6*<br>- *ipv6@vrf*<br>- *ipv6-linklocal%interface* |
| **type** (icmp/ *tcp-conn\|http-get\|simple;* D efault: **simple**) | Type of the probe:<br>- icmp - (ping-style) series of ICMP request-response with statistics<br>- tcp-conn - test TCP connection (3-way handshake) to a server specified by IP and port<br>- http-get - do an HTTP Get request and test for a range of correct replies<br>- simple - simplified ICMP probe, with fewer options than "ICMP" type, used for backward compatibility with the older Netwatch version |
| **interval** (Default: **10s**) | The time interval between probe tests |

| | |
|---|---|
| **timeout** (Default: **3s**) | Max time limit to wait for a response |
| **src-address** (Default:**""**) | Source IP address which the Netwatch will try to use in order to reach the host. If address is not present, then the host will be considered as "down". |
| **start-delay** (Default: **3s**) | Time to wait before starting probe (on add, enable, or system start) |
| **startup-delay** (Default: **5m**) | Time to wait until starting Netwatch probe after system startup |
| **up-script** (Default:"") | Script to execute on the event of probe state change 'fail' --> 'OK' |
| **down-script** (Default:"") | Script to execute on the event of probe state change 'OK' --> 'fail' |
| **test-script** (Default:"") | Script to execute at the end of every probe test |

Netwatch executes scripts as *sys user, so any defined global variable in the Netwatch script will not be readable by for an example a scheduler or other users

> Netwatch is limited to *read,write,test,reboot* script policies. If the owner of the script does not have enough permissions to execute a certain command in the script, then the script will not be executed. If the script has greater policies than *read,write,test,reboot* - then the script will not be executed as well, make sure your scripts do not exceed the mentioned policies.

It is possible to disable permission checking for RouterOS scripts under */system/scripts* menu. This is useful when Netwatch does not have enough permissions to execute a script, though this decreases overall security. It is recommended to assign proper permissions to a script instead.

# Type-specific options

All config options specific to one probe type (e.g. icmp's packet-interval) are ignored for other probe types (tcp-conn, http-get).

## ICMP probe options

| **Property** | **Description** |
|---|---|
| **packet-interval** (Default: **50ms**) | The time between ICMP-request packet send |
| **packet-count** (Default: **10**) | Total count of ICMP packets to send out within a single test |
| **packet-size** (Default: **54** (IPv4) or **54** (IPv6)) | The total size of the IP ICMP packet |
| **thr-rtt-max** (Default**: 1s**) | Fail threshold for rtt-max (a value above thr-max is a probe fail) |
| **thr-rtt-avg** (Default: **100ms**) | Fail threshold for rtt-avg |
| **thr-rtt-stdev** (Default: **250ms**) | Fail threshold for rtt-stdev |
| **thr-rtt-jitter** (Default: **1s**) | Fail threshold for rtt-jitter |
| **thr-loss-percent** (Default: **85.0%**) | Fail threshold for loss-percent |
| **thr-loss-count** (Default: **4294967295**(max)) | Fail threshold for loss-count |

## TCP-CONNECT/HTTP-GET probe options

| **Property** | **Description** |
|---|---|
| **port** (Default: **80**) | TCP port (for both tcp-conn and http-get probes) |

## TCP-CONNECT pass-fail criteria

| **Property** | **Description** |
|---|---|

| | |
|---|---|
| **thr-tcp-conn-time** (Default: **00:05...00:30**) | Fail threshold for tcp-connect-time, the configuration uses microseconds, if the time unit is not specified (s/m/h), log and status pages display the same value in milliseconds. |

## HTTP-GET probe pass/fail criteria

| Property | Description |
|---|---|
| **thr-http-time** (Default: **10s**) | Fail threshold for http-resp-time |
| **http-code-min** (Default: **100**) | OK/fail criteria for HTTP response code. |
| **http-code-max** (Default: **299**) | Response in the range [`http-code-min` , `http-code-max`] is a probe pass/OK; outside - a probe fail. See mozilla-http-status or rfc7231 |

# Probe statistics/variables

You can view statistics and use these variables in scripting, keep in mind that variables containing "-" must be written like this, for example, "done-tests" would be $"done-tests"

## Generic:

| Property | Description |
|---|---|
| **name** | user added name for Netwatch entry |
| **comment** | user added comment |
| **host** | host that was probed |
| **type** | probe type |
| **interval** | interval |
| **timeout** | timeout |
| **since** | The last time the status change happened |
| **status** | current status of probe |
| **done-tests** | total count of probe tests already done so far |
| **failed-tests** | count of failed probe tests |

## ICMP:

| Property | Description |
|---|---|
| **sent-count** | ICMP packets sent out |
| **response-count** | Matching/valid ICMP packet responses received |
| **loss-count** | number of lost packets |
| **loss-percent** | number of lost packets in percent |
| **rtt-avg** | mean value of rtt (packet roundtrip time) |
| **rtt-min** | min rtt |
| **rtt-max** | max rtt |

| | |
|---|---|
| **rtt-jitter** | jitter ( = max - min) of rtt |
| **rtt-stdev** | standard deviation of rtt |

### TCP:

| Property | Description |
|---|---|
| **tcp-connect-time** | time taken to establish a TCP connection |

### HTTP:

| Property | Description |
|---|---|
| **http-status-code** | HTTP response status code (200 OK, 404 Not Found, etc.). See mozilla-http-status or RFC7231 |

### HTTPS:

| Property | Description |
|---|---|
| **http-status-code** | HTTP response status code (200 OK, 404 Not Found, etc.). See mozilla-http-status or RFC7231 |

## Logs

On each probe's OK/fail state change:

- probe identification info and OK->fail or fail->OK is printed to info level
- detailed probe stats and config is printed to debug level

## Status

Command */tool/netwatch/print* will show the current status of Netwatch and **read-only** properties:

- since - Indicates when a state of the host changed last time;
- status - Shows the current status of the host;
- host - address being monitored

## Quick Example

Here we will use a simple ICMP check to host with IP 8.8.8.8:

```
[admin@MikroTik] > /tool/netwatch add host=8.8.8.8 interval=30s up-script=":log info \"Ping to 8.8.8.8
successful\""
```

Afterward, in the logging section we can see Netwatch executed script:

```
[admin@MikroTik] > log print where message~"8.8.8.8"
08:03:26 script,info Ping to 8.8.8.8 successful
```