

## QOS\_WATCHER V1.5

### *Release history*

<b>Version</b>	<b>Author</b>	<b>Comments</b>
1.0	Gijsbert Wiesenekker	Initial release.
1.1	Gijsbert Wiesenekker	The alarm now also shows the message with the largest delay.
1.2	Gijsbert Wiesenekker	The alarm now uses the description of the QoS watcher as the suppression key to auto-clear alarms.
1.3	Gijsbert Wiesenekker	QoS messages are now sent for the number of delayed QoS messages.
1.4	Gijsbert Wiesenekker	Added an option to track all QoS messages that match a QoS watcher.
1.5	Gijsbert Wiesenekker	A watcher will at most send one alarm: the QoS message with the largest delay. QoS messages are matched to watchers last-to-first. QoS messages that match a QoS watcher are tracked by default, and all tracked QoS messages are logged to csv files.

### *Description*

This probe uses QoS messages as a heartbeat check: it monitors if certain QoS messages have been received and generates alarms when not or have been received with a delay. If you have not received an alert from a probe (or from a profile within a probe) for some time either everything is fine, but there could also be a problem. Perhaps someone has disabled the probe or the profile in the configuration, or perhaps the probe or the profile is not working correctly. If you are still receiving QoS messages it is likely that everything is working as it should. There could still be a problem receiving the alarm messages of course, so the probe also generates a heartbeat 'hello from qos\_watcher' alarm.

Good use cases for this probe are monitoring of QoS heartbeats from critical url\_response and e2e\_appmon profiles.

### *Installation*

Ensure SDK\_Perl 5.04 or greater is deployed to the robot that will run cmd.

Ensure ActiveState Perl version 5.14 is installed on the robot system if it is a Windows robot.

Deploy the probe.

### *Usage*

Double click the probe in Infrastructure Manager to raw configure it or edit the configuration file with a text-editor (recommended):



## qos\_watcher 1.5

Name	Optional/Required	Description
interval	Optional. The default is 300.	The interval at which the probe should check if QoS messages have not been received for any of the defined QoS watchers. The interval should be smaller than any of the maximum allowed delays in the QoS watchers (see below). This is checked by the probe on startup.

The probe subscribes to QoS messages and matches received QoS messages against the QoS watchers defined in the <watchers> section of the configuration file in the order last first, so more specific QoS watchers should be defined *after* the more general watchers in the configuration file. For each QoS watcher you specify the following fields. The field names are the names shown by Dr. Nimbus when sniffing QoS messages:

Name	Value	Type
nimid	XK19749018-27610	string
nimts	1394646949	integer
tz_offset	-3600	integer
source	172.16.4.123	string
hop	0	integer
hop0	hubint123	string
md5sum		void
robot	centos64int123	string
domain	domint123	string
origin	hubint123	string
pri	1	integer
subject	QOS_MESSAGE	string
prid	cdm	string
dev_id	D1A1A480E808858BE9DD37C1C737246F	string
met_id	MAA5189EE6887EFAC0B7ED9B18B999457	string
udata	-	PDS
qos	QOS_INODE_USAGE_PERC	string
source	centos64int123	string
target	/	string
sampletime	1394646949	integer
sampletype	0	integer
samplevalue	2.800284	float
samplestdev	0	float
samplerate	300	integer
samplemax	100	integer
qsize	12	integer

Name	Optional or required	Description
description	Required	A description of the QoS watcher. The description is used as the suppression id in the alarm message, but spaces will be replaced by the '-' sign.
robot	Optional, default '.'	The name of the robot in the QoS message. You can use any Perl regular expression without the leading and trailing '/
prid	Optional, default '.'	The name of the probe in the QoS message. You can use any Perl regular expression without the leading and trailing '/



## qos\_watcher 1.5

Name	Optional or required	Description
qos	Optional, default ‘.*’	The name of the QoS in the QoS message. You can use any Perl regular expression without the leading and trailing ‘/’.
source	Optional, default ‘.*’	The name of the source in the QoS message. You can use any Perl regular expression without the leading and trailing ‘/’.
target	Optional, default ‘.*’	The name of the target in the QoS message. You can use a Perl regular expression without the leading and trailing ‘/’.
delay	Optional, default 900	The maximum allowed delay for QoS messages for this watcher.
track_qos	Optional, default ‘yes’	If ‘yes’ the probe will track the individual delay of all QoS messages that match the watcher. Messages that exceed the delay are logged to the file ‘description.csv’ to avoid cluttering the console.

Multiple QoS messages could match a watcher when using wildcards. In such case the oldest QoS message is retained.

The supplied configuration file shows examples how to watch for

- any QoS;
- any QoS from a remote Hub;
- QoS CPU\_USAGE from the cdm probe on the portal server;
- QoS Portal.\* from the e2e\_appmon Portal profile monitoring the portal server. The name of the profile is not available in the e2e\_appmon QoS message, so you are advised to start your QoS name in the e2e\_appmon script with the name of the profile:  
Script\$ = nimGetEnv\$("APPMON\_PROFILE\_NAME", "APPMON\_PROFILE\_NAME")  
QoS\$(1) = Script\$ + " Login"  
etc.