NMIS 8 - Configuration Options for Server Performance Tuning

- Related Articles
- NMIS 8 Processes
  - Configurations that affect performance
  - Gaps in Graphs

Related Articles

- Scaling NMIS Polling
- Scaling NMIS polling - how NMIS handles long running processes
- Configuration Options for Server Performance Tuning

NMIS 8 Processes

The main NMIS 8 process is called from different cron jobs to run different operations: collect, update, summary, clean jobs, etc. As an example:

```
* * * * * root /usr/local/nmis8/bin/nmis.pl type=collect abort_after=60 mthread=true ignore_running=true;
```

The cron configuration can be found in /etc/crond.d/nmis.

For a collect or an update, the main thread is set up by default to fork worker processes to perform the requested operations using threads and improving performance. One of each operation will run every minute (by default), and will process as many nodes as the collect polling cycle is set up to process.

Configurations that affect performance

There are some important configurations that affect performace:

- **abort_after**: From NMIS 8.6.8G there is a new command line option, abort_after, that prevents the main thread to run for a long time, preventing it to collide with the next cron job. By default, this parameter is 60 seconds, as the cron job is set to run every 60 minutes by default.

  Also, this option needs to always have also the option mthreads=true.

  ```
  nmis8/bin/nmis.pl type=collect abort_after=60 mthread=true ignore_running=true;
  ```

- **nmis_maxthreads or maxthreads**: The other important configuration option is nmis_maxthreads, which is maxthreads on the command line, that will prevent the number of children of the main process to grow too big. Considerations:
  - If the collect operation has a lot of nodes to process, the number of children won't reach the limit instantly. While the main thread is forking, the children complete their jobs and will exit. Also, the main process will wait for them to change their state so the number will increase slowly.
  - NMIS can have more than one instance of the main process running, and the number of children could be higher that nmis_maxthreads, as the limit is only per instance.

- **sort_due_nodes:** When NMIS decides what to poll it can do so in a pseudo random order which is the default, if your server is overloaded you will likely see some nodes never getting polled, hence pseudo random, so for heavily loaded servers, enable sort_due_nodes, in the NMIS configuration add with the value set to 1.

Gaps in Graphs

If the server takes a long time to collect and cannot complete any operation, an useful tool is nmis8/admin/polling_summary. Here we can see how many nodes have any late collect, and a summary of nodes being collected and not collected:

```
nmis8/admin> ./polling_summary.pl
```

An example output:

```
node                    attempt   status    ping  snmp  policy     delta  snmp avgdel  poll   update
pollmessage
u18_poller              23:55:02  pingonly  down  down  default    ---    300  0.00    0.00   0.03    no snmp
collect
uburnto                 13:14:03  pingonly  down  down  default    ---    300  0.00    0.00   0.03    no snmp
collect
unreachablenode         23:55:02  demoted   down  down  default    ---    300  0.00    0.00   0.01    snmp
polling demoted
virtual_elf             23:56:03  pingonly  down  down  default    ---    300  0.00    0.00   0.02    no snmp
collect
vrouter-host            16:44:01  ontime    up    up    default    299    300  300.04  1.44
1.59
vyos-p1                 16:44:02  ontime    up    up    default    299    300  300.04  1.27
2.79
vyos-p2                 16:44:01  ontime    up    up    default    299    300  300.04  1.99
1.91
vyos-p3                 16:44:04  ontime    up    up    default    299    300  300.05  1.79
1.86
vyos-p4                 16:44:03  ontime    up    up    default    300    300  300.04  1.81
1.84
vyos-pe1                16:44:04  ontime    up    up    default    300    300  300.05  1.81
1.91
vyos-pe2                16:44:04  ontime    up    up    default    299    300  300.05  1.78
1.90
vyos-rr1                16:47:02  ontime    up    up    default    300    300  300.00  2.23
2.22
vyos-rr2                16:44:01  ontime    up    up    default    299    300  300.05  1.95
1.79
wifi                    16:46:02  ontime    up    up    default    300    300  300.00  0.56
0.34

totalNodes=59 totalPoll=52 ontime=38 pingOnly=14 1x_late=0 3x_late=0 12x_late=0 144x_late=0
```

A symptom of an overloaded server can be gaps in the graphs.

Below is an example about how these parameters can impact in the performance of the server, in a server with 64 CPUs and more than 3700 nodes:

When	abort_after (seconds)	demote_faulty_nodes	CPU	Nodes Not Collected	Other
Initial Configuration	Default (60)	false	<50% (Aprox.)	1100 ~	`totalPoll=3713 ontime=891 1x_late=1460 3x_late=41 12x_late=56 144x_late=1265`
Test 1	120	true	<50% (Aprox.)	500 ~	N/A
Test 2	240	true	<60% (Aprox.)	240 ~	`totalPoll=1229 ontime=998 no_snmp=14 demoted=0 1x_late=217 3x_late=0 12x_late=0 144x_late=0`
Test 3	0 (Disabled)	true	Around 100% (Aprox.)	0	Took 7 minutes. Processed >3000 nodes. Disabled cron
Test 4	0 (Disabled)	true	100% (Aprox.)	N/A	Commented while (wait for children) in nmis.pl
Test 5	0 (Disabled)	false	100% (Aprox.)	N/A	N/A

Note that problems in the modelling that throw errors in the logs can also make the system slow. The polling time for each node will be increased, hence the polling cycle will take longer to run, and depending on the configuration options, the process can be aborted with some nodes not being polled.

(Internal case reference: SUPPORT-6976)