

Extending SNMPd for custom monitoring

NMIS relies primarily on SNMP and ping for collecting measurements of your infrastructure's health, but there are situations where this is insufficiently flexible: not every interesting device out there is SNMP-capable, and there are concepts and services that do not directly fit into the SNMP universe.

In the [Managing Servers and Services with NMIS8](#) document we describe what mechanisms NMIS itself provides for managing non-SNMP services: e.g. checking ports, looking at process status, checking the DNS and checking textual protocols with send/expect scripts - and the recently added capability to run external programs for getting a service's status.

This page describes a more generic approach to this kind of problem which doesn't rely on custom features programmed into NMIS: instead we show how to extend the standard Net-SNMP snmpd with a script or program of your choice, to make an arbitrarily non-standard 'thing' accessible via SNMP (and thus available to NMIS).

Use-cases for this infrastructure include collecting statistics from services that don't offer SNMP (e.g. the bind DNS server), capturing the status of multi-component services (e.g. email end-to-end) and so on.

snmpd and pass_persist programs

[snmpd's manual page](#) describes a number of extensibility mechanisms, one of them called "pass_persist programs": snmpd starts that program and delegates an OID subtree to it. Whenever it is queried for variables in that subtree it forwards the request to the pass_persist program which provides an answer. As the communication is very simple (write to the program's STDIN, read from its STDOUT) it's a very flexible way of capturing custom things; It's also very efficient because the pass_persist program is running permanently and there is no repeated startup overhead, and the program can do whatever it needs to do, whenever and however it wants to.

There are a few caveats:

- The documentation for this snmpd-to-program communication isn't complete - a blank 'command' is meant to tell your program that it should terminate.
- You'll need to pick an unused OID subtree to attach your script at; the [Net-SNMP documentation](#) recommends you use .1.3.6.1.4.1.8072.2.255 or .1.3.6.1.4.1.2021.255.
- The snmpd will block until the program has responded. This means your program needs to perform its operations in a non-blocking fashion, or your snmp infrastructure will suffer badly.

An example pass_persist program

Here is an example program in perl, which reads `/proc/loadavg` every 42 seconds and makes this information available at .1.3.6.1.4.1.2021.255.1 to .3, without blocking your snmpd.

Download: [passpersist-example.pl](#)

You would use it by adding the following line to your `snmpd.conf`:

```
pass_persist .1.3.6.1.4.1.2021.255 /wherever/you/put/your/passpersist-example.pl
```

It's quite simple, commented and less than 200 lines of code. Feel free to use it as a template for your own extensions.

Where to go from here

Once you have captured your custom measurements and you've tested the snmpd-pass_persist interaction with snmpwalk or snmpget, the next step would be to extend the most appropriate model with your new measurements. This part is a straightforward modelling exercise, and you will find ample documentation in the [NMIS section of this site](#), and lots of examples in the `models-install/` directory of your NMIS installation.