The Nagios Windows plugins

Revision	Date	By	Changes
01	5th of August 05	T.F. Sluyter	Initial creation
02	9th of Sept 05	T.F. Sluyter	Switched to NRPEnt.
03	12th of June 06	T.F. Sluyter	Converted for use on website.

Copyright and such

Summary

This document is meant as a companion to the document titled "NAGIOS configuration". Whereas that document describes the general configuration and usage of the Nagios monitoring software, this document provides additional instructions on the usage of the following:

- The NRPEnt daemon, which is installed on the Windows client.
- The NSClient binary, which is installed on the Windows client.
- The **`check_nt`** plugin, which is used on the server, with NSClient.

We decided to switch to NRPEnt for our monitoring needs in September, since NSClient was deemed too complicated for our daily use (I still include the information to be as complete as possible).

Acquiring and installing NRPEnt

For those who would like to get a fresh start, the binary for NRPEnt may be downloaded from http://www.miwi-dv.org/nrpent.

< original proprietary instructions removed >

Please install according to the instructions included with the NRPEnt binary.

Acquiring and installing NSClient

For those who would like to get a fresh start, the binary for NSClient may be downloaded from http://nsclient.ready2run.nl.

< original proprietary instructions removed >

Please install according to the instructions included with the NRPEnt binary.

The check_nt plugin

In order to have Nagios read information from the NSClient software you will use the check_nt plugin which is stored with the rest of the plugins in **/usr/local/nagios/libexec**. The plugin will be run on the Nagios master server[1].

Running **check_nt -h** will show you all of the available options for using the plugin. I will not go over all of the options, but one thing I will do is explain all of the different metrics that may be read from NSClient.

Variable (-v)	Metric
CLIENTVERSION	Not very interesting, this variable returns the version number of the NSClient binary used.
CPULOAD	The average CPU load over the last X minutes. In the case of multi-CPU systems an average is taken. Using the **-l** parameter you provide the parameters for **$timeframe**, **$warning** and **$critical**. For example: **-l 1,80,90**. You may also request multiple measurements. For example: **-l 1,80,90,5,75,85**.
UPTIME	Naturally this returns the uptime of the system in question. There is no option to set warning or critical levels.
USEDDISKSPACE	Reports the size and usage percentage of a disk. Use **-l** to provide a drive letter and **-w** and **-c** to set warning and critical levels.
MEMUSE	Reports the total amount of memory, and the amounts and percentages of used and free memory.
SERVICESTATE	Checks the state of one or several services. Using the **-l** parameter you can provide a comma separated list of service names. When supplying the service name you should not use the displayed name, but the real one stored in **HKEY_LOCAL\SYSTEM\CurrentControlSet\Services**.
PROCSTATE	Check the state of one or several processes. Using the **-l** parameter you can provide a comma separated list of process names, which can be found in the Task Manager.
COUNTER	Windows keeps a lot of counters and you can specify their exact name (enclosed in double quotes) using **-l**. Using **-w** and **-c** you can, yet again, set warning and critical levels.

[1] Actually, it is entirely possible to have another Nagios host poll the Windows systems. In that case the server will use check_nrpe to run commands on the remote host. The remote host in turn will have all kinds of command definitions that will make use of check_nt.

FILEAGE	This should report on the last modification date for the file specified with **-l**, but right now it does not seem to be working correctly.
INSTANCES	Aside from the aforementioned counters, Windows also keeps track of so-called Permon Counters, which can be read using this variable. Once again the object name should be enclosed in double quotes and and is passed using **-l**.

Defining monitors with NRPEnt

NRPEnt uses the same principles as NRPE, but instead requires version 2.0 (instead of <2.0) of `check_nrpe`. Hence we made a new check-command. An exemplary service definition:

```
Define service{
      use                     generic-service
      host_name               win-pub01
      service_description     LOAD
      check_period            24x7
      contact_groups          admins
      check_command           check_nrpe2!check_load
}
```

Defining monitors with NSClient

NSCLient is a tad more complicated:

```
Define service{
      use                     generic-service
      host_name               win-pub01
      service_description     LOAD
      check_period            24x7
      contact_groups          admins
      check_command           check_nt_load!1,80,90,5,75,85
}
```

The command definition for **check_nt_load** would then look like this.

```
Define command{
      command_name     check_nt_load
      command_line     $USER1$/check_nt -H $HOSTADDRESS -v CPULOAD
-l $ARG1$
}
```