## Overview

There are large numbers of WMI classes on Windows hosts and this plugin package can be used to retrieve almost any parameter of interest in the default WMI namespace. With a small change to each plugin they can be used to retrieve parameters from any WMI namespace. There is no need for agents on the remote hosts.

The plugin package comes with an NRPE configuration file that can be included from the base nrpe.cfg file. Some of the plugins retrieve specific properties such as CPU load percentage, disk utilization, disk and network I/O, etc. while others are for retrieving arbitrary properties from WMI. The provided NRPE configuration file defines NRPE commands using the plugins to get both specific and arbitrary properties and there are documented syntax examples for each command.

Plugins are configured to return performance data where it makes sense which facilitates easy graphing of results on a Nagios server.

## Description of plugins

The following table lists plugins in the package with a brief description of each.

| Plugin | Description |
|---|---|
| check_100nsec_timer.vbs | checks WMI counters of type 'PERF_100NSEC_TIMER' |
| check_counter_bulk_count.vbs | checks WMI counters of type 'PERF_COUNTER_BULK_COUNT' |
| check_counter_counter.vbs | checks WMI counters of type 'PERF_COUNTER_COUNTER' |
| check_counter_large_rawcount.vbs | checks WMI counters of type 'PERF_COUNTER_LARGE_RAWCOUNT' |
| check_counter_rawcount.vbs | checks WMI counters of type 'PERF_COUNTER_RAWCOUNT' |
| check_cpu_load_percentage.vbs | checks load percentage of one or more CPUs |
| check_disks_io.vbs | checks disk I/O of one or more logical disks |
| check_disks_percentage_space_used.vbs | checks disk usage of one or more logical disks |
| check_memory_percentage_space_used.vbs | checks RAM, page file, or total memory usage |
| check_network_io.vbs | checks network I/O of one or more TCP/IP network interfaces |
| check_raw_fraction.vbs | checks WMI counters of type 'PERF_RAW_FRACTION' |
| check_proc_num.vbs | checks number of running processes matching a search expression |
| check_services_states.vbs | checks the state of an installed service |
| get_computer_info.vbs | enumerates information about processors, installed services, running processes, network interfaces, and logical disks for use in configuring other plugins |
| get_counter_type.vbs | enumerates information about WMI classes and properties to determine which counter plugin to use |
| verify_wmi_status.vbs | checks OS version to verify WMI is working |

## Installation

*Prerequisites*

1.  NRPE_NT version 0.8b. The remainder of this document assumes that NRPE_NT has been installed under `C:\NRPE_NT` with the executable and configuration file under `C:\NRPE_NT\bin`. It also assumes that you have tested correct operation of NRPE_NT by calling it with the `check_nrpe` Nagios plugin from your Nagios server. Version 0.8b is necessary in order to fully support specification of some special characters when calling NRPE_NT from `check_nrpe`. NRPE_NT can be downloaded from [nagiosexchange.org](nagiosexchange.org).
2.  Nagios 2.x with `check_nrpe` plugin compatible with NRPE_NT version 0.8b. Nagios can be found at [www.nagios.org](www.nagios.org).

*Installation*

1.  Once you have installed NRPE_NT on a host (designated as `<host>`) and tested that it can be called from your Nagios server with a `check_nrpe -H <host>` create the following directories:

        C:\NRPE_NT\Plugins
        C:\NRPE_NT\Plugins\V2

2.  Copy all of the VBS programs from this directory into `C:\NRPE_NT\Plugins\V2`.
3.  Copy the file `V2_nrpe_commands.cfg` into `C:\NRPE_NT\bin`.
4.  Edit `C:\NRPE_NT\bin\nrpe.cfg` and append the following line to the bottom:

        include=C:\NRPE_NT\bin\V2_nrpe_commands.cfg

5.  Restart the NRPE_NT service.

*Testing*

After completing the installation procedure you should be able to call each of the installed scripts with the commands defined in `V2_nrpe_commands.cfg` either from the Windows server itself or from the Nagios server using `check_nrpe`.

1.  On the Windows server you installed NRPE_NT and the plugins on, open a command window and type:

        cscript //nologo C:\NRPE_NT\Plugins\V2\verify_wmi_status.vbs -h
        127.0.0.1

    The output you get should look something like:

        OK - Microsoft Windows XP Professional, SP 2.0

    This indicates that the plugin was able to successfully talk with the WMI service on 127.0.0.1 and retrieve the OS version.

2.  On the Nagios server, run type the following command in a shell once you have changed directory to the location of the check_nrpe plugin:

        ./check_nrpe -H <host> -c show_os -a 127.0.0.1

**GROUNDWORK**
**OPEN SOURCE**

The output you get should look something like:

```
OK - Microsoft Windows XP Professional, SP 2.0
```

You have just executed the same `verify_wmi_status.vbs` plugin but through the NRPE_NT service from your Nagios server.

## Examples

The following example Nagios command and service definitions assume you have Nagios 2.x installed with a version of check_nrpe compatible with NRPE_NT v0.8b.  They also assume that $USER1$ points to the directory containing check_nrpe, and that $USER21$ is the address of the host on which NRPE_NT is installed!  Download information about both Nagios 2.x and NRPE_NT v0.8b is available in the install document referenced below.

```
define command {
      command_name check_wmi_mssql_transactions
      command_line $USER1$/check_nrpe -t 60 -H $USER21$ -c \\
      get_mssql_transactions -a "$HOSTADDRESS$" "$ARG1$" \\
      "$ARG2$" "$ARG3$"
}

define service {
      name wmi_mssql_transactions
      check_command
check_wmi_mssql_transactions!Name=_Total!10!20
      #.
      #.  other  parameters  as  desired  for  service  definition
      #.

}


define command {
      command_name check_wmi_cpu
      command_line $USER1$/check_nrpe -t 60 -H "$USER21$" -c \\
      get_cpu -a "$HOSTADDRESS$" "$ARG1$" "$ARG2$"
}

define service {
      name wmi_cpu
      check_command check_wmi_cpu!_Total!80,90
      #.
      #.  other  parameters  as  desired  for  service  definition
      #.
}


define command {
      command_name check_wmi_mem
      command_line $USER1$/check_nrpe -t 60 -H $USER21$ -c \\
```

**GROUNDWORK**
**OPEN SOURCE**

```
        get_mem -a "$HOSTADDRESS$" "$ARG1$" "$ARG2$"
}

define service {
        name wmi_mem_ram
        check_command check_wmi_mem!RAM!80,90
        #.
        #. other parameters as desired for service definition
        #.
}


define command {
        command_name check_wmi_net_io
        command_line $USER1$/check_nrpe -t 60 -H $USER21$ -c \\
        get_netio -a "$HOSTADDRESS$" "$ARG1$" "$ARG2$" "$ARG3$"
}

define service {
        name wmi_net_io_loopback
        check_command  check_wmi_net_io!"MS  TCP  Loopback  Interface"
\\
        !BytesReceivedPerSec,BytesSentPerSec!300,1000
        #.
        #. other parameters as desired for service definition
#.
}
```

For a list of further examples refer to the NRPE_NT configuration file available for download below.  One key point is that you can reverse warning and critical thresholds and have match in the reverse sense.  E.g. if you set -w 80 -c 50 and the returned value is 90 the return code will be 0 (OK).


## Downloads

All plugins, NRPE_NT configuration files, and this document are located in the following Subversion repository:

WMI plugins HEAD revision:
http://archive.groundworkopensource.com/groundwork-exchange/trunk/plugins/nagios/wmi/

WMI plugins tagged revisions:
http://archive.groundworkopensource.com/groundwork-exchange/tags/


## NRPE_NT and WMI troubleshooting

WMI is installed on newer versions of Windows and can also be downloaded and installed on Windows NT 4.  However, sometimes there are issues in getting WMI functioning.  Most often these are easy to solve by ensuring authentication is set up correctly but in other cases there are corrupt or missing data structures on the remote host.  There are a number of references on the Internet to assist with troubleshooting WMI

**GROUNDWORK**
**OPEN SOURCE**

problems.  The plugins in this package assist with this troubleshooting by returning the WMI error code and (if possible) a description for that error.

If you are using the plugins without supplying -user and -pass arguments then the NRPE_NT service must be run as a user with administrative privileges on the host that the plugins are targeting.  If you are using the plugins with -user and -pass arguments then that user must have administrative privileges on the target host.

In the case where you aren't using the -user and -pass arguments the use of a domain administrator account and then making sure that domain administrators have local administrative rights on each target host is the most efficient and maintainable solution. Another method involves defining a local account on each target host that uses the same account name and password as that passed to the plugin or as that being used to run the NRPE_NT service.

We list the following URLs on WMI and WMI troubleshooting for your convenience and will make every attempt to keep the links up to date:

- WMI Classes – A list of Microsoft WMI classes and their properties.  Note that other vendors extend the WMI schema with their own classes, e.g. Cisco AVVID products.
- Using WMI to monitor performance - A reference from Microsoft.
- Repairing WMI – A reference from Microsoft.
- WMI Tools – Microsoft's WMI Tools ActiveX applications for browsing WMI schema and instances on a computer.
- WMI Diagnosis Utility – Microsoft's WMIDiag Tool to troubleshoot the state of WMI on a computer.

## Further assistance

If you get stuck with using these plugins then please visit the GroundWork Community Support Forums site. There you will find topics on the WMI plugins as well as monitoring in general.

## Change log

*1.3 December 5, 2008*
- updated documentation.
- added check for `StartMode` in `check_services_states.vbs` so that if the service is set to Manual or Disabled then the return code will be `OK`.

*1.2 August 28, 2008*
- no changes, but tagged with version 1.2

*HEAD February 29, 2008*
- moved initialization of `intReturnTemp1` out of subroutines to global area allowing for correct exit code determination based on the worst case code determined for all queried parameters.

*HEAD October 16, 2007*
- added ability to specify `~DS~` in the service name and have it substituted for a `$` symbol in the `check_services_states.vbs` script.

*HEAD October 15, 2007*
- updated `f_error` in all scripts to return `UNKNOWN` for all errors.

**GROUNDWORK**
**OPEN SOURCE**

- updated `check_disks_percentage_space_used.vbs` to probe for local fixed disks (type 3) only
- updated `check_disks_percentage_space_used.vbs` to correctly return the right exit code if any of the disks checked are above the specified thresholds
- updated to support property name in performance data output of the `check_disks_io.vbs` script.

*HEAD July 17, 2007*
- updated `check_cpu_load_percentage.vbs` to correctly return an UNKNOWN state when CPU utilization cannot be determined via WMI (an issue with WMI that is observed).
- updated `check_cpu_load_percentage.vbs` to correctly loop through available CPUs so that one can specify something other than the first in the list or 'total'.

*1.1 June 7, 2007*
- Updated `f_Error` function in all VBS scripts:
  462 error exits critical with the error text Timeout connecting to WMI on this host
  -2147023174 exists critical with the error text Timeout connecting to WMI on this host
- Updated `f_ExecQuery` in all VBS scripts:
  If no instance is returned from the WMI query then plugin exits in an UNKNOWN state with the error text Unknown - No data received from WMI
- Updated threshold specification in the following way:
  Specifying a warning threshold lower than a critical threshold
        if value > warning then exit state is warning
        if value > critical then exit state is critical
  Specifying a warning threshold higher than a critical threshold
        if value < warning then exit state is warning
        if value < critical then exit state is critical
  Specifying an inclusive range for a threshold, e.g. 90:95
        if value is within the range then the threshold is not triggered

*1.0 November 22, 2006*
- Initial Release

**GROUNDWORK**
**OPEN SOURCE**