Nagios®

NRPE Documentation

Last Updated: May 1, 2007

CONTENTS

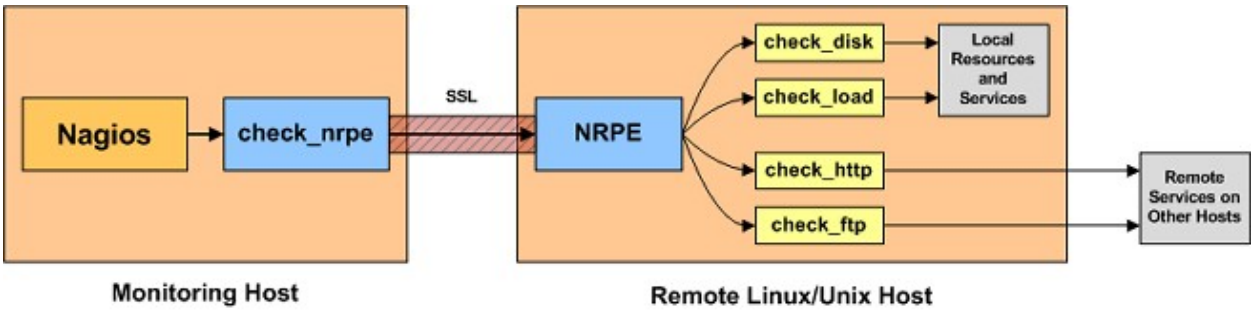
1. INTRODUCTION

a) Purpose

The NRPE addon is designed to allow you to execute Nagios plugins on remote Linux/Unix machines. The main reason for doing this is to allow Nagios to monitor "local" resources (like CPU load, memory usage, etc.) on remote machines. Since these public resources are not usually exposed to external machines, an agent like NRPE must be installed on the remote Linux/Unix machines.

Note: It is possible to execute Nagios plugins on remote Linux/Unix machines through SSH. There is a *check_by_ssh* plugin that allows you to do this. Using SSH is more secure than the NRPE addon, but it also imposes a larger (CPU) overhead on both the monitoring and remote machines. This can become an issue when you start monitoring hundreds or thousands of machines. Many Nagios admins opt for using using the NRPE addon because of the lower load it imposes.

b) Design Overview



The NRPE addon consists of two pieces:

- The *check_nrpe* plugin, which resides on the local monitoring machine
- The *NRPE* daemon, which runs on the remote Linux/Unix machine

When Nagios needs to monitor a resource of service from a remote Linux/Unix machine:

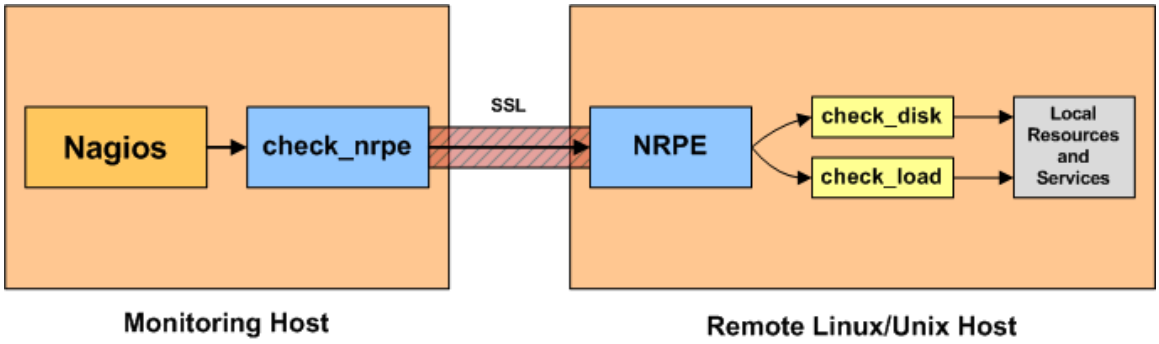
- Nagios will execute the *check_nrpe* plugin and tell it what service needs to be checked
- The *check_nrpe* plugin contacts the *NRPE* daemon on the remote host over an (optionally) SSL-protected connection
- The *NRPE* daemon runs the appropriate Nagios plugin to check the service or resource
- The results from the service check are passed from the *NRPE* daemon back to the *check_nrpe* plugin, which then returns the check results to the Nagios process.

Note: The NRPE daemon requires that Nagios plugins be installed on the remote Linux/Unix host. Without these, the daemon wouldn't be able to monitor anything.

2. EXAMPLE USES

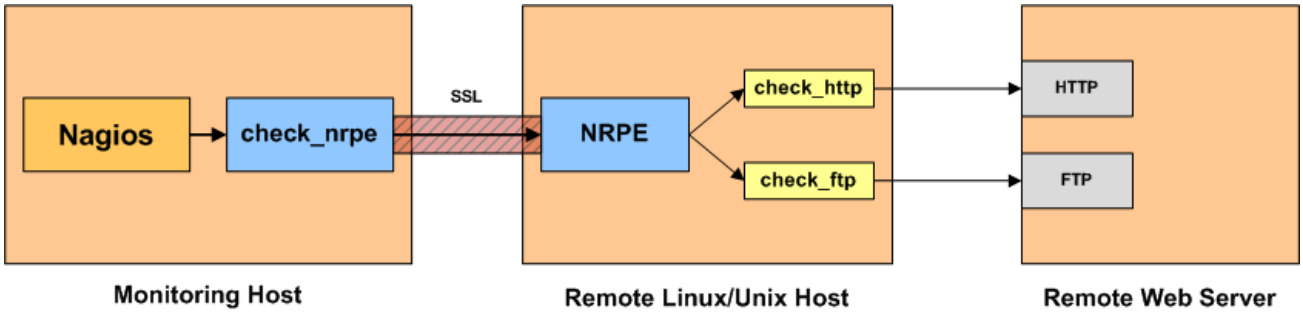
a) Direct Checks

The most straight forward use of the NRPE addon is to monitor "local" or "private" resources on a remote Linux/Unix Machine. This includes things like CPU load, memory usage, swap usage, current users, disk usage, process states, etc.



b) Indirect Checks

You can also use the NRPE addon to indirectly check "public" services and resources of remote servers that might not be reachable directly from the monitoring host. For instance, if the remote host that the NRPE daemon and plugins are installed on can talk to the a remote web server (but the monitoring host cannot), you can configure the NRPE daemon to allow you to monitor the remote web server indirectly. The NRPE daemon is essentially acting as proxy in this case.



3. INSTALLATION

In order to use the NRPE addon, you'll need to perform some tasks on both the monitoring host and the remote Linux/Unix host that the NRPE daemon is installed on. I'll cover both of these tasks separately.

Note: These instructions are based on a remote host running Fedora Core 6. Naming conventions, commands, etc. vary across different Linux distros and UNIX variants, so the instructions provided here may have to be altered a bit for your situation. If you're having trouble using these instructions, you can find OS/distribution-specific installation guides, HOWTOs, and other helpful installation documentation on the Nagios Community wiki at http://www.nagioscommunity.org/wiki/.

a) Prerequisites

In order to complete these installation instructions, you'll need:

- Root access on the remote Linux/Unix host
- Access to the nagios user account on the monitoring host

b) Assumptions

These instructions assume that you are installing the NRPE addon on a system that supports TCP wrappers and has the xinetd superserver installed. Most modern Linux distributions and Unix variants have these installed by default. If you have an older system that runs inetd (instead of xinetd) and/or does not support TCP wrappers, or you want to run the NRPE daemon without inetd or xinetd, read the following files for information on installing the addon (both are found in the main directory of the NRPE distribution):

- README
- SECURITY

[Continued on next page]

c) Remote Host Setup

These instructions should be completed on the remote Linux/Unix host that the NRPE daemon will be installed on. You'll be installing the Nagios plugins and the NRPE daemon...

i. Create Account Information

Become the root user. You may have to use `sudo -s` on Ubuntu and other distros.

```
su -l
```

Create a new nagios user account and give it a password.

```
/usr/sbin/useradd nagios
passwd nagios
```

ii. Install the Nagios Plugins

Create a directory for storing the downloads.

```
mkdir ~/downloads
cd ~/downloads
```

Download the source code tarball of the Nagios plugins (visit http://www.nagios.org/download/ for links to the latest versions). At the time of writing, the latest stable version of the Nagios plugins was 1.4.6.

```
wget http://osdn.dl.sourceforge.net/sourceforge/nagiosplug/nagios-plugins-1.4.6.tar.gz
```

Extract the Nagios plugins source code tarball.

```
tar xzf nagios-plugins-1.4.6.tar.gz
cd nagios-plugins-1.4.6
```

Compile and install the plugins.

```
./configure
make
make install
```

The permissions on the plugin directory and the plugins will need to be fixed at this point, so run the following commands.

```
chown nagios.nagios /usr/local/nagios
chown -R nagios.nagios /usr/local/nagios/libexec
```

iii. Install xinetd

Fedora Core 6 doesn't ship with xinetd installed by default, so install it with the following command:

```
yum install xinetd
```


iv. Install the NRPE daemon

Download the source code tarball of the NRPE addon (visit http://www.nagios.org/download/ for links to the latest versions). At the time of writing, the latest version of NRPE was 2.8.

```
cd ~/downloads
wget http://osdn.dl.sourceforge.net/sourceforge/nagios/nrpe-2.8.tar.gz
```

Extract the NRPE source code tarball.

```
tar xzf nrpe-2.8.tar.gz
cd nrpe-2.8
```

Compile the NRPE addon.

```
./configure
make all
```

Install the NRPE plugin (for testing), daemon, and sample daemon config file.

```
make install-plugin
make install-daemon
make install-daemon-config
```

Install the NRPE daemon as a service under xinetd.

```
make install-xinetd
```

Edit the */etc/xinetd.d/nrpe* file and add the IP address of the monitoring server to the *only_from* directive.

```
only_from       = 127.0.0.1 <nagios_ip_address>
```

Add the following entry for the NRPE daemon to the */etc/services* file.

```
nrpe            5666/tcp        # NRPE
```

Restart the xinetd service.

```
service xinetd restart
```


v. Test the NRPE daemon locally

Its time to see if things are working properly...

Make sure the nrpe daemon is running under xinetd.

```
netstat -at | grep nrpe
```

The output out this command should show something like this:

```
tcp        0          0 *:nrpe   *:*        LISTEN
```

If it does, great! If it doesn't, make sure of the following:

- You added the nrpe entry to your */etc/services* file
- The *only_from* directive in the */etc/xinetd.d/nrpe* file contains an entry for "127.0.0.1"
- xinetd is installed and started
- Check the system log files for references about xinetd or nrpe and fix any problems that are reported

Next, check to make sure the NRPE daemon is functioning properly. To do this, run the *check_nrpe* plugin that was installed for testing purposes.

```
/usr/local/nagios/libexec/check_nrpe -H localhost
```

You should get a string back that tells you what version of NRPE is installed, like this:

```
NRPE v2.8
```

vi. Open firewall rules

Make sure that the local firewall on the machine will allow the NRPE daemon to be accessed from remote servers. To do this, run the following iptables command. Note that the *RH-Firewall-1-INPUT* chain name is Fedora-specific, so it will be different on other Linux distributions.

```
iptables -I RH-Firewall-1-INPUT -p tcp -m tcp –dport 5666 -j ACCEPT
```

Save the new iptables rule so it will survive machine reboots.

```
service iptables save
```


vii. Customize NRPE commands

The sample NRPE config file that got installed contains several command definitions that you'll likely use to monitor this machine. The command definitions are used to (surprise) define commands that the NRPE daemon will run to monitor local resources and services. The sample command definitions run some of the plugins that were installed in step 2. You can edit the command definitions, add new commands, etc, by editing the NRPE config file:

```
vi /usr/local/nagios/etc/nrpe.cfg
```

More information on customizing the commands can be found on page 13 in the section titled "Customizing Your Configuration".

For the time being, I'll assume you're using the sample commands that are defined. You can test some of these by running the following commands:

```
/usr/local/nagios/libexec/check_nrpe -H localhost -c check_users
/usr/local/nagios/libexec/check_nrpe -H localhost -c check_load
/usr/local/nagios/libexec/check_nrpe -H localhost -c check_hda1
/usr/local/nagios/libexec/check_nrpe -H localhost -c check_total_procs
/usr/local/nagios/libexec/check_nrpe -H localhost -c check_zombie_procs
```

At this point, you are done installing and configuring NRPE on the remote host. Now its time to install a component and make some configuration entries on your monitoring server...

[Continued on next page]

d) Monitoring Host Setup

On the monitoring host (the machine that runs Nagios), you'll need to do just a few things:

- Install the check_nrpe plugin
- Create a Nagios command definition for using the check_nrpe plugin
- Create Nagios host and service definitions for monitoring the remote host

These instructions assume that you have already installed Nagios on this machine according to the quickstart installation guide. The configuration examples that are given reference templates that are defined in the sample *localhost.cfg* and *commands.cfg* files that get installed if you follow the quickstart.

i. Install the check_nrpe plugin

Become the root user. You may have to use `sudo -s` on Ubuntu and other distros.

```
su -l
```

Create a directory for storing the downloads.

```
mkdir ~/downloads
cd ~/downloads
```

Download the source code tarball of the NRPE addon (visit http://www.nagios.org/download/ for links to the latest versions). At the time of writing, the latest version of NRPE was 2.8.

```
wget http://osdn.dl.sourceforge.net/sourceforge/nagios/nrpe-2.8.tar.gz
```

Extract the NRPE source code tarball.

```
tar xzf nrpe-2.8.tar.gz
cd nrpe-2.8
```

Compile the NRPE addon.

```
./configure
make all
```

Install the NRPE plugin.

```
make install-plugin
```


ii. Test communication with the NRPE daemon

Make sure the check_nrpe plugin can talk to the NRPE daemon on the remote host. Replace "192.168.0.1" in the command below with the IP address of the remote host that has NRPE installed.

/usr/local/nagios/libexec/check_nrpe -H 192.168.0.1

You should get a string back that tells you what version of NRPE is installed on the remote host, like this:

```
NRPE v2.8
```

If the plugin returns a timeout error, check the following:

- Make sure there isn't a firewall between the remote host and the monitoring server that is blocking communication
- Make sure that the NRPE daemon is installed properly under xinetd
- Make sure the remote host doesn't have local (iptables) firewall rules that prevent the monitoring server from talking to the NRPE daemon

iii. Create a command definition

You'll need to create a command definition in one of your Nagios object configuration files in order to use the check_nrpe plugin. Open the sample *commands.cfg* file for editing...

```
vi /usr/local/nagios/etc/commands.cfg
```

and add the following definition to the file:

```
define command{
        command_name    check_nrpe
        command_line    $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$
        }
```

You are now ready to start adding services that should be monitored on the remote machine to the Nagios configuration...

iv. Create host and service definitions

You'll need to create some object definitions in order to monitor the remote Linux/Unix machine. These definitions can be placed in their own file or added to an already exiting object configuration file.

First, its best practice to create a new template for each different type of host you'll be monitoring. Let's create a new template for linux boxes.

```
define host{
          name                            linux-box            ; Name of this template
          use                             generic-host         ; Inherit default values
          check_period                    24x7
          check_interval                  5
          retry_interval                  1
          max_check_attempts              10
          check_command                   check-host-alive
          notification_period             24x7
          notification_interval           30
          notification_options            d,r
          contact_groups                  admins
          register                        0          ; DONT REGISTER THIS - ITS A TEMPLATE
          }
```

Notice that the linux-box template definition is inheriting default values from the generic-host template, which is defined in the sample *localhost.cfg* file that gets installed when you follow the Nagios quickstart installation guide.

Next, define a new host for the remote Linux/Unix box that references the newly created linux-box host template.

```
define host{
          use                     linux-box            ; Inherit default values from a template
          host_name               remotehost           ; The name we're giving to this server
          alias                   Fedora Core 6        ; A longer name for the server
          address                 192.168.0.1          ; IP address of the server
          }
```

Next, define some services for monitoring the remote Linux/Unix box. These example service definitions will use the sample commands that have been defined in the nrpe.cfg file on the remote host.

The following service will monitor the CPU load on the remote host. The "check_load" argument that is passed to the check_nrpe command definition tells the NRPE daemon to run the "check_load" command as defined in the nrpe.cfg file.

```
define service{
          use                     generic-service
          host_name               remotehost
          service_description     CPU Load
          check_command           check_nrpe!check_load
          }
```

The following service will monitor the the number of currently logged in users on the remote host.

```
define service{
          use                     generic-service
          host_name               remotehost
          service_description     Current Users
          check_command           check_nrpe!check_users
          }
```

The following service will monitor the free drive space on /dev/hda1 on the remote host.

```
define service{
          use                     generic-service
          host_name               remotehost
          service_description     /dev/hda1 Free Space
          check_command           check_nrpe!check_hda1
          }
```


The following service will monitor the total number of processes on the remote host.

```
define service{
          use                     generic-service
          host_name               remotehost
          service_description     Total Processes
          check_command           check_nrpe!check_total_procs
          }
```

The following service will monitor the number of zombie processes on the remote host.

```
define service{
          use                     generic-service
          host_name               remotehost
          service_description     Zombie Processes
          check_command           check_nrpe!check_zombie_procs
          }
```

Those are the basic service definitions for monitoring the remote host. If you would like to add additional services to be monitored, read the "Customizing Your Configuration" section starting on page 13.

v. Restart Nagios

At this point you've installed the check_nrpe plugin and addon host and service definitions for monitoring the remote Linux/Unix machine. Now its time to make those changes live...

Verify your Nagios configuration files.

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

If there are errors, fix them. If everything is fine, restart Nagios.

```
service nagios restart
```

That's it! You should see the host and service definitions you created in the Nagios web interface. In a few minutes Nagios should have the current status information for the remote Linux/Unix machine.

Since you might want to monitor more services on the remote machine, I would suggest you read the next section as well. :-)

Also, when it comes time to upgrade the version of NRPE you're running, its pretty easy to do. The initial installation was the toughest, but upgrading is a snap.

4. CUSTOMIZING YOUR CONFIGURATION

You'll might want to monitor more services on the remote Linux/Unix box than those that are described in this documentation. This is easy to accomplish.

Anytime you want to monitor a new service on a remote host using the NRPE addon, you have to do two things:

1. Add a new command definition to the nrpe.cfg file on the remote host
2. Add a new service definition to your Nagios configuration on the monitoring host

Let's say you want to monitor the swap usage on the remote Linux/Unix host. Here are the steps you'd need to follow...

a) Remote Host Configuration

You can use the *check_swap* plugin to monitor swap usage on the machine. Assuming you followed the installation instructions in this document, the *check_swap* plugin should already be installed in */usr/local/nagios/libexec*.

Login as the nagios user on the remote host.

Run the *check_swap* plugin manually and tweak the command line options to specify the desired warning and critical free swap space thresholds. Make sure the full command line returns the expected output you want from the plugin. For this example, let's say you want a critical alert if swap free space is less than 10% and a warning if free space is less than 20%. Here's the command line that would accomplish that:

```
/usr/local/nagios/libexec/check_swap -w 20% -c 10%
```

Now that you know the command line that should be execute, open the NRPE configuration file.

```
vi /usr/local/nagios/etc/nrpe.cfg
```

Add a new *check_swap* command definition that uses the command line from above and save the file.

```
command[check_swap]=/usr/local/nagios/libexec/check_swap -w 20% -c 10%
```

If you're running the NRPE daemon as a standalone daemon you'll need to restart it. If you're running it under the inetd/xinetd superserver you don't need to do anything more.

b) Monitoring Host Configuration

On the monitoring host, you need to define a new service for monitoring the swap usage on the remote host. Add the following entry to one of your object configuration files.

```
define service{
          use                     generic-service
          host_name               remotehost
          service_description     Swap Usage
          check_command           check_nrpe!check_swap
          }
```

Notice that the check commands is passing "check_swap" to the *check_nrpe* command definition. This will cause the NRPE daemon to run the *check_swap* command that was defined in the nrpe.cfg file on the remote host in the previous step.

Next, verify your Nagios configuration files and restart Nagios.

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

service nagios restart
```

That's it! You are now monitoring a new service on the remote host using the NRPE addon.

5. UPGRADING

At some point in time you'll want to upgrade the version of the NRPE addon that you're running. The upgrade process is fairly pain free. Here are the details...

a) Monitoring Host Upgrade

On the monitoring host (the machine that runs Nagios), you only have to upgrade the check_nrpe plugin. Here's how...

Login as the nagios user and create a directory for storing the downloads.

```
mkdir ~/downloads
cd ~/downloads
```

Download the source code tarball of the NRPE addon (visit http://www.nagios.org/download/ for links to the latest versions). At the time of writing, the latest version of NRPE was 2.8.

```
wget http://osdn.dl.sourceforge.net/sourceforge/nagios/nrpe-2.8.tar.gz
```

Extract the NRPE source code tarball.

```
tar xzf nrpe-2.8.tar.gz
cd nrpe-2.8
```

Compile the NRPE addon.

```
./configure
make all
```

Install the NRPE plugin.

```
make install-plugin
```

You're done on the monitoring server!

[Continued on Next Page]

b) Remote Host Upgrades

You'll need to upgrade the NRPE daemons on all remote hosts that are being monitored using the NRPE addon. Here's how...

Login as the nagios user and create a directory for storing the downloads.

```
mkdir ~/downloads
cd ~/downloads
```

Download the source code tarball of the NRPE addon (visit http://www.nagios.org/download/ for links to the latest versions). At the time of writing, the latest version of NRPE was 2.8.

```
wget http://osdn.dl.sourceforge.net/sourceforge/nagios/nrpe-2.8.tar.gz
```

Extract the NRPE source code tarball.

```
tar xzf nrpe-2.8.tar.gz
cd nrpe-2.8
```

Compile the NRPE addon.

```
./configure
make all
```

Install the NRPE daemon.

```
make install-daemon
```

If you're running the NRPE daemon as a standalone daemon, first kill the old daemon process and then start the (new) daemon up again.

You're done with the upgrade on the remote host!

6. TROUBLESHOOTING

Here are some tips for troubleshooting some of the more common errors with the NRPE addon. If you encounter problems that aren't covered here, send an email to the *nagios-users* mailing list. More information on the mailing lists can be found at: http://www.nagios.org/support/

The check_nrpe plugin returns "CHECK_NRPE: Socket timeout after 10 seconds" or "Connection refused or timed out"

This error can indicate several things:

- The command that the NRPE daemon was asked to run took longer than 10 seconds to execute. This is the most likely cause if the error message was "CHECK_NRPE: Socket timeout after 10 seconds". Use the *-t* command line option to specify a longer timeout for the check_nrpe plugin. The following example will increase the timeout to 30 seconds:

```
/usr/local/nagios/check_nrpe -H localhost -c somecommand -t 30
```

- The NRPE daemon is not installed or running on the remote host. Verify that the NRPE daemon is running as a standalone daemon or under inetd/xinetd with one of the following commands:

```
ps axuw | grep nrpe

netstat -at | grep nrpe
```

- There is a firewall that is blocking the communication between the monitoring host (which runs the check_nrpe plugin) and the remote host (which runs the NRPE daemon). Verify that the firewall rules (e.g. iptables) that are running on the remote host allow for communication and make sure there isn't a physical firewall that is located between the monitoring host and the remote host.

The check_nrpe plugin returns "CHECK_NRPE: Received 0 bytes from daemon. Check the remote server logs for an error message."

First thing you should do is check the remote server logs for an error message. Seriously. :-) This error could be due to the following problem:

- The check_nrpe plugin was unable to complete an SSL handshake with the NRPE daemon. An error message in the logs should indicate whether or not this was the case. Check the versions of OpenSSL that are installed on the monitoring host and remote host. If you're running a commercial version of SSL on the remote host, there might be some compatibility problems.

The check_nrpe plugin returns "NRPE: Unable to read output"

This error indicates that the command that was run by the NRPE daemon did not return any character output. This could be an indication of the following problems:

- An incorrectly defined command line in the command definition. Verify that the command definition in your NRPE configuration file is correct.

- The plugin that is specified in the command line is malfunctioning. Run the command line manually to make sure the plugin returns some kind of text output.

The check_nrpe plugin returns "NRPE: Command 'x' not defined"

This error means that you didn't define command *x* in the NRPE configuration file on the remote host. On the remote host, add the command definition for *x*. See the existing command definitions in the NRPE configuration file for more information on doing this. If you're running the NRPE daemon as a standalone daemon (and not under inetd or xinetd), you'll need to restart it in order for the new command to be recognized.

The check_nrpe plugin returns "NRPE: Command timed out after x seconds"

This error indicates that the command that was run by the NRPE daemon did not finish executing within the specified time. You can increase the timeout for commands by editing the NRPE configuration file and changing the value of the *command_timeout* variable. If you're running the NRPE daemon as a standalone daemon (and not under inetd or xinetd), you'll need to restart it in order for the new timeout to be recognized.

How to go about debugging other problems...

When debugging problems it may be useful to edit the NRPE configuration file and change the *debug=0* entry to *debug=1*. Once you do that, restart the NRPE daemon if it is running as a standalone daemon. After you try using the check_nrpe plugin again, you should be able to see some debugging information in the log files of the remote host. Check your logs carefully – they should be able to help provide clues as to where the problem lies...

