Nagios XI - NRPE Troubleshooting and Common Solutions

Nagios®

The Industry Standard in IT Infrastructure Monitoring

Purpose

This document describes how to troubleshoot Nagios Remote Plugin Executor (NRPE) agent issues in Nagios® XI™. NRPE is most commonly used to monitor other Linux or Unix servers. This document will cover solutions to common problems/errors and general troubleshooting tips. NRPE is also used to communicate with NSClient++, which is a Microsoft Windows monitoring client. This document also provides a simple framework to systematically drill down problems with NRPE.

Target Audience

This document is intended for Nagios administrators who are having issues with the NRPE agent on Linux platforms. It assumes that you have some basic knowledge of how NRPE works, and that you have installed the NRPE agent on a remote server.

Prerequisites

There are a number of prerequisites and defaults that need to be explained before running through this document:

- This document will assume that NRPE and the plugins are installed in the default directories. If your corporate build or repo installed NRPE or the Nagios-plugins to a different directory than the default location, you will have to make note and substitute your paths for those in this document. The directories referenced most in this document are `/usr/local/nagios/libexec/` and `/usr/local/nagios/etc/`.

- The remote system being checked by the NRPE agent will be referred as “remote host” while the Nagios server will be referred to as the “XI Server” or “Nagios Server”.

- The command line editor “vi” will be used throughout this document. When using the vi editor:
  - To make changes press **`i`** on the keyboard first to enter insert mode
  - Press **`Esc`** to exit insert mode
  - When you have finished, save the changes in vi by typing **`:wq`** and press Enter

- Any packages that must be installed for certain sections of this document will use the standard Red Hat/CentOS package manager “yum”. If you are using a different distribution, you most likely will have to install the packages through your package manager (apt-get for Debian/Ubuntu, pacman for Archlinux, etc).

Nagios XI - NRPE Troubleshooting and Common Solutions

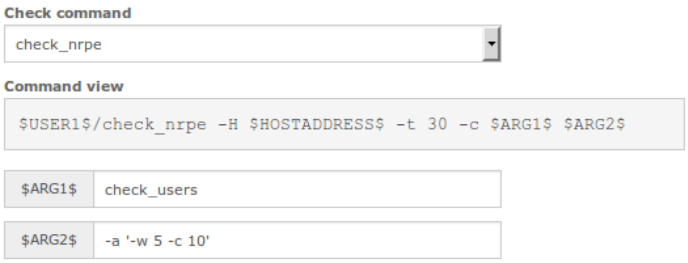
What Types Of Problems Can An Administrator Expect To Encounter With NRPE?

The most common problems are found in the initial setup of NRPE. Connection and communication issues are some of the easiest to troubleshoot and resolve. Other problems that people run into involve specific errors reported by Nagios XI in reference to an agent or remote host.

NRPE Explained

A lot of this documentation does require you to understand how the NRPE configuration files work and how it receives requests and executes the plugins. This brief introduction explains how NRPE works, this will help you understand the information in this document.

Here's a screenshot from a service definition in Nagios XI Core Configuration Manager (CCM).



You can see how the **Command view** shows the Nagios XI command being used. The variables `$USER1$`, `$HOSTADDRESS$`, `$ARG1$` and `$ARG2$` are dynamically inserted into the command when Nagios XI executes the check.

Here is that same command after the variables have been replaced with the real values:

```
/usr/local/nagios/libexec/check_nrpe -H 10.25.13.30 -t 30 -c check_users -a '-w 5 -c 10'
```

What does this mean? The two important options explained here are:

- `-c check_users`
  - `-c` is how Nagios XI tells the remote server which command it wants executed
  - In this example `check_users` is the name of the command
- `-a '-w 5 -c 10'`
  - `-a` is how Nagios XI sends a list of arguments to the remote server
  - In this example, the arguments being sent are enclosed in `'single quotes'` and hence will be received as ONE argument ($ARG1$). This will be explained in more detail shortly.

On the remote server, where is the `check_users` command defined?

Commands are defined in the `nrpe.cfg` file, this is commonly located at `/usr/local/nagios/etc/nrpe.cfg`.

They may also be defined in an additional `.cfg` file referenced in the `nrpe.cfg` file, for example:

```
include_dir=/usr/local/nagios/etc/nrpe
```

What does a command definition look like? Using the example above, the `check_users` command is defined as follows:

```
command[check_users]=/usr/local/nagios/libexec/check_users $ARG1$
```

- The line begins with the word `command`, this is how the config file knows this line is a command definition
- The word between the `[square brackets]` is the actual command "name"
- Everything after the `=` is the actual plugin that is being executed, along with any arguments
- This example only accepts one argument. Arguments are defined as `$ARG`**`x`**`$` where **`x`** is the argument number received

Using the example of the request sent by the Nagios XI server, the remote server will execute the following command:

```
/usr/local/nagios/libexec/check_users -w 5 -c 10
```

You can see that the `$ARG1$` variable has been replaced with `-w 5 -c 10`.

It's a simple as that, however there was something mentioned earlier that requires further explanation.

- `-a '-w 5 -c 10'`
  - the arguments being sent are enclosed in `'single quotes'` and hence will be received as ONE argument ($ARG1$). This will be explained in more detail shortly.
- If the command was defined as `-a -w 5 -c 10` (without the single quotes), then NRPE on the remote server would receive the following variables:
  - `$ARG1$` **=** `-w`
  - `$ARG2$` **=** `5`
  - `$ARG3$` **=** `-c`
  - `$ARG4$` **=** `10`
- This means that NRPE on the remote server will execute the following command:
  - `/usr/local/nagios/libexec/check_users -w`
- Clearly this is an invalid command, the point being made here is that the NRPE agent will only use the variables that have been defined in the command definition. If you wanted to NOT use `'single quotes'` then your NRPE command definition would need to be:
  - `command[check_users]=/usr/local/nagios/libexec/check_users -w $ARG1$ -c $ARG2$`
- You then define the arguments as `-a 5 10`
  - Before NRPE v3 it was not possible to send `-c` as an "argument", hence you defined these in the command and only send the values

This completes the brief introduction on how NRPE works.

Error Codes And Other Issues Covered In This Document

This is by no means a definitive list, but these are the most common problems associated with the NRPE agent. Although it is not entirely in the scope of this document, there are a few tips for troubleshooting NSClient++ when using the NRPE handler.

Errors:

NSClient++ NRPE Specific Errors:

General

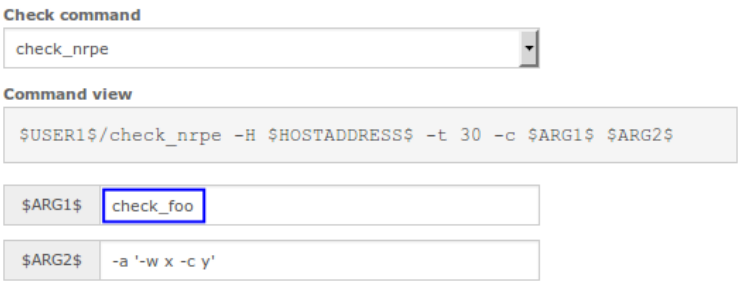
If you are experiencing an error with NRPE that is not listed here, you are encouraged to contact us at the Nagios Support Forum for possible resolutions:

https://support.nagios.com/forum

I. Return Code Of 127 Is Out Of Bounds - Plugin May Be Missing

This error is usually experienced when the plugin referenced by the command directive in `nrpe.cfg` is either missing from the libexec folder or the command directive is named incorrectly. It could also imply that the command name passed through NRPE from the Nagios XI server is not defined in the `nrpe.cfg` file on the remote host.

The first troubleshooting step is to know the name of the command you are requesting NRPE to execute. From the screenshot you can see the command is called:




```
check_foo
```

On your remote host make sure that the command is defined in `nrpe.cfg`, for example:

```
command[check_foo]=/usr/local/nagios/libexec/check_foo.sh $ARG1$
```

Now that you have identified the command definition, verify the spelling of `check_foo` in `$ARG1$` (in CCM) matches the exact spelling of the command directive name `command[`**`check_foo`**`]`.

Next, make sure the plugin being executed on the remote host actually exists. In this example, the name of the plugin is:

```
/usr/local/nagios/libexec/check_foo.sh
```

Execute the following command on the remote host to see if it exists:

```
ls -la /usr/local/nagios/libexec/check_foo.sh
```

If it does not exist, then the output would be something like:

```
ls: cannot access /usr/local/nagios/libexec/check_foo.sh: No such file or directory
```

If this is your problem, you will have to copy the plugin to the `/usr/local/nagios/libexec/` folder.

Make sure the filename `check_foo.sh` has the correct file extension. Different plugins have different file extensions (`.sh`, `.bin`, `.pl`, `.py`, etc.). The path must include the extension of the plugin, but the command directive name, wrapped in `'command[]'` does not need an extension.

II. Return Code Of 126 Is Out Of Bounds - Plugin May Not Be Executable

Many times when a plugin is downloaded from the exchange and copied to the remote host, it will not have executable permissions. You can verify this by getting a long-listing of the libexec plugin directory. For this example will will be using the command `check_foo.sh`. On the remote host execute the following command:

```
ls -l /usr/local/nagios/libexec
```

You should see a listing similar to:

```
-rwxr-xr-x. 1 root   root     4173 Nov 21 01:39 check_bl
-rw-r--r--. 1 root   root     2289 Nov 21 01:39 check_foo.sh
```

The far left column of the listing are the permissions for each file. If you noticed, `check_foo.sh` is missing an "`x`" in a few places. These are executable permissions and can easily be added to the file using the following command:

```
chmod +x /usr/local/nagios/libexec/check_foo.sh
```

Remember that `check_foo.sh` is just an example, you will change `/usr/local/nagios/libexec/check_foo.sh` to the actual name and path to your plugin that is missing executable permissions.

III. CHECK_NRPE: Error - Could Not Complete SSL Handshake

Allowed hosts:

This is probably the most common of all error messages and one of the first you will experience when new to NRPE. There are a few different causes of this, though the most likely one is that the Nagios XI server's IP address is not defined as being allowed to communicate with NRPE. This can be defined in one of two locations, depending if you are using xinetd as the daemon to run NRPE or if NRPE has it's own dedicated daemon. To identified which one is being used, execute the following command on the remote host:

```
ls -la /etc/xinetd.d/nrpe
```

If you get output like this, **xinetd** is being used and you should follow the **xinetd** steps below:

```
-rw-r--r--. 1 root root 483 Feb 22 12:23 /etc/xinetd.d/nrpe
```

If you get output like this, NRPE has it's own Daemon and you should follow the **nrpe.cfg** steps below:

```
ls: cannot access /etc/xinetd.d/nrpe: No such file or directory
```

XINETD

If you use xinetd for controlling the NRPE daemon (most people do), then you need to add the Nagios server's IP address to the xinetd NRPE configuration file `/etc/xinetd.d/nrpe`. Edit the file by executing the following command:

```
vi /etc/xinetd.d/nrpe
```

In this file you will find the line:

```
only_from = 127.0.0.1
```

This list is a **space-delimited** list. Change it to:

```
only_from = 127.0.0.1 <Nagios XI server ip>
```

Remember to change `<Nagios XI server ip>` to your actual Nagios XI server IP address. One thing to note is that `127.0.0.1` should remain as it allows you to troubleshoot NRPE issues locally. After you have made the following changes, restart the `xinetd` service on the remote host:.

```
service xinetd restart
```

NRPE.CFG

If you use a dedicated daemon for NRPE, then you need to add the Nagios server's IP address to the `nrpe.cfg` configuration file `/usr/local/nagios/etc/nrpe.cfg`. Edit the file by executing the following command:

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

In this file you will find the line:

```
allowed_hosts=127.0.0.1
```

This list is a **comma-delimited** list. Change it to:

```
allowed_hosts=127.0.0.1,<Nagios XI server ip>
```

Remember to change `<Nagios XI server ip>` to your actual Nagios XI server IP address. One thing to note is that `127.0.0.1` should remain as it allows you to troubleshoot NRPE issues locally. After you have made the following changes, restart the `xinetd` service on the remote host: (this example may be different to your operating system distribution):

```
service nrpe restart
```

SSL Not Compiled In:

Another cause of SSL issues is that NRPE was not compiled with ssl enabled. To recompile NRPE with ssl support, browse to your NRPE source directory (usually in /tmp/nrpe-2.14 if you followed the compiling NRPE from source document) and re-compile using the `--enable-ssl` flag:

```
cd /tmp/nrpe-2.14
./configure --enable-ssl
make all
make install
```

Understand that if you installed from a corporate build or from a package repo, you may have either uninstall the current NRPE package and install from source. You may need to pursue support on the specific distribution's forums or through Nagios support. The following link has detailed instructions on compiling NRPE from source:

https://support.nagios.com/kb/article.php?id=515

Xinetd Per Source Limit:

This cause is rare, but worth mentioning. If you use your remote host's NRPE server as a NRPE node proxy (sending all checks for the network segment to a single NRPE enabled server behind a firewall), or if you are doing a large number of NRPE checks in relatively short time period on one remote host, you may hit the maximum connection limit of NRPE. This is technically an xinetd setting and can be uncapped by editing the file `/etc/xinetd.d/nrpe` on your remote host:

```
nano /etc/xinetd.d/nrpe
```

Add the following line to the file inside the closing "`}`":

```
per_source = UNLIMITED
instances = UNLIMITED
```

And then restart XINETD with the following command:

```
service xinetd restart
```

IV. CHECK_NRPE: Error - Could not connect to xxx.xxx.xxx.xxx: Connection reset by peer

This is the same as the **Allowed host** issue described in the previous chapter. Please follow the steps in the previous chapter to define the Nagios XI server's IP address as being allowed to communicate with NRPE.

V. CHECK_NRPE: Socket Timeout After n Seconds

Increase Socket Timeout:

This is one of the harder to pin down errors. More often than not, following the steps from **part III** will be enough to solve this problem. However sometimes it is not related to SSL or your allowed hosts. In these instances, it can either be that a plugin is taking longer than "n" seconds to return the check, or there is a firewall/port issue.

You can increase the timeout on the check, though you will have to alter the check in XI and the command and connection timeout in the `nrpe.cfg` file on the remote host.

Nagios XI check_nrpe Timeout

This timeout is how long the `check_nrpe` command on the Nagios XI server will wait for a response from the NRPE agent. By default the timeout is set to 10 seconds, which is too short for certain checks (disk/filesystem/database checks among others) however in Nagios XI the default has been defined at 30.

In the Nagios XI web interface navigate to **Configure** > **Core Config Manager** > **Commands**. This brings up the Commands page, use the **Search** field to search for **nrpe** and click **Search**.

Click the `check_nrpe` command.

You can change the timeout in Nagios XI with the switch `-t` in the `check_nrpe` command.

In the **Command Line**, change `-t xx` to a higher value, in the screenshot above you can see it is set to `30` seconds.

Save your changes and then click the **Apply Configuration** button.

NRPE Client Timeout

This timeout is how long the **NRPE** client on the Nagios XI server will wait for a response from the plugin it executes before returning a result to Nagios XI. You may need to change a couple settings in the remote host's `/usr/local/nagios/etc/nrpe.cfg` file depending on how high you set the timeout in Nagios XI. Edit the file with the following command:

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

Search for the `command_timeout=` and `connection_timeout=` settings which may need to be altered. Set both of these, at minimum, to the value of the timeout in Nagios XI. Usually the `connection_timeout=300` is more than enough, as is the `command_timeout` which defaults to 60 seconds. If you do set your timeout in Nagios XI higher, increase the `command_timeout` to match.

Plugin Timeout

You may also find that certain plugins also have their own timeout argument, if this does exist you would need to define your NRPE command to also take this into account.

Nagios XI Global Timeout

Nagios XI by default has a global timeout for host (30 seconds) and service (60 seconds) check commands. This means if you were to change the `check_nrpe` command timeout in Nagios XI with the switch `-t` to `120`, Nagios XI will not wait for 120 seconds to pass, the global timeout will stop at 60 seconds.

To adjust the global timeout, navigate to **Configure** > **Core Config Manager** > **CCM Admin** > **Core Configs**. This brings up the Core Configs page and by default the **General [nagios.cfg]** tab is selected. The two directives to change are:

```
host_check_timeout=30
service_check_timeout=60
```

Click Save Changes to update these settings and the **Apply Config** via **Quick Tools**.

A Realistic Discussion On Timeouts

After reading all of that you might think to yourself "*I'm going to go and change all the timeouts to 120 seconds*". It's not as simple as that, you need to take into account that each layer of timeout needs to take into account the previous layer. If Nagios XI global timeout was set to 120 seconds and the NRPE was `command_timeout=120` then it may take a whole second before it gets to NRPE, you will need to take that into account, here's an example of the "layers":

- Nagios XI Global Timeout
  - 120
- `check_nrpe` timeout on Nagios XI server
  - 119
- `connection_timeout=` on NRPE Client
  - 118
- `command_timeout=` on NRPE Client
  - 117
- Plugin specific timeout (if any)
  - 116

This completes the section on timeouts. The remaining part of this chapter helps identify other reasons why **Socket Timeout After n Seconds** may be occurring.

Check the NRPE Service Status:

You may receive this error if the NRPE daemon is not running on the remote host. If you are using xinetd, you can check the status of the service by logging onto the remote host as root and running the following command:

```
service xinetd status
```

You should see output similar to the following:

```
xinetd (pid  1260) is running...
```

If you are using the init-script method, or if your distribution does not use the "service" command, you can always grep a process listing:

```
ps -aef | grep nrpe
```

You should see output similar to the following (important bits in bold):

```
nagios   53213     1  0 Feb26 ?    00:00:07 /usr/libexec/nrpe -c /etc/nagios/nrpe.cfg --daemon
```

If NRPE/xinetd is not running, start it with the following command:

```
service xinetd start
```

Or if you are not using xinetd:

```
/path/to/init/script start
```

Check Firewall and Port Settings:

The last of the probable causes of this error is associated with firewalls and ports. If the NRPE traffic is not traversing a firewall, you will see the checks timeout. Additionally, if port 5666 is not open on the remote host's firewall, you may receive a timeout error as well. Usually xinetd will open the ports automatically, as long as the `/etc/xinetd.d/nrpe` file is configured correctly, and NRPE's port settings have been added to `/etc/services`.

First, we should make sure that port 5666 is open on the remote host. The easiest way to do this, is to just run `check_nrpe` from the remote host to itself. This will also double as a good way to check that NRPE is functioning as expected. Log into the remote host as root and execute:

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

You should get something similar to the following output:

```
NRPE v2.15
```

If not, make sure the that port 5666 is open on the remote host's firewall. If you are using xinetd go back to previous step (check the NRPE service status) as it should automatically open the port for you.

Checking Remote Host's Ports and Configuring iptables:

You may have to open port 5666 on your firewall, which in the case of most Linux distributions, is iptables. To get a listing of the current iptables rules, run the following on the remote host as root:

```
iptables -L
```

The expected output is similar to:

```
ACCEPT - tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:5666
```

or

```
ACCEPT     tcp  -  anywhere             anywhere            state NEW tcp dpt:nrpe
```

If the port is not open, you will have to add an iptables rule for it using the following commands.

```
iptables -I INPUT -p tcp --destination-port 5666 -j ACCEPT
service iptables save
```

Those commands were for TCP/IP v4. If you need TCP/IP **v6** the commands are similar:

```
ip6tables -L
```

The expected output is similar to:

```
ACCEPT     tcp  -  anywhere             anywhere            state NEW tcp dpt:nrpe
```

If the port is not open, you will have to add an ip6tables rule for it using the following commands.

```
Ip6tables -I INPUT -p tcp --destination-port 5666 -j ACCEPT
service ip6tables save
```

Checking Remote Host's Ports and Configuring firewalld:

Firewalld is present on Enterprise Linux 7 and higher. To get a listing of the current firewalld rules, run the following on the remote host as root:

```
firewall-cmd --list-all
```

The expected output is similar to:

```
ports: 5666/tcp
```

If the port is not open, you will have to add a firewalld rule for it using the following commands.

```
firewall-cmd --zone=public --add-port=5666/tcp
firewall-cmd --zone=public --add-port=5666/tcp --permanent
```

firewalld applies to both TCP/IP v4 and TCP/IP v6.

Checking Port 5666 From the Nagios XI Server with nmap:

You can use nmap (among other port scanners) to check the remote host's ports. If you do not have nmap installed, it can be installed using the following commands (with yum for RHEL/CentOS systems):

```
yum install -y nmap
```

Once installed, test the connection on port 5666 from the Nagios XI server to the remote host by logging in as root on your Nagios XI server and running the following command:

```
nmap <remote host ip> -Pn -p 5666
```

Remember to replace your remote host server ip address above. The expected output should be similar to:

```
PORT     STATE SERVICE
5666/tcp open  nrpe
```

VI. CHECK_NRPE: Received 0 Bytes From Daemon. Check The Remote Server Logs For Error Messages

First, make sure that NRPE is running as this is a common cause of this error. For instructions on how to do so, refer to section V of this document under **Check the NRPE Service Status**.

The other causes all deal with arguments. If you are passing arguments to the remote host through NRPE, the argument usage should be consistent between the Nagios XI service check and the arguments declared in the command directive in the remote host's `nrpe.cfg`.

Additionally, check the remote host's `nrpe.cfg` for the `dont_blame_nrpe` directive. Take this command executed on the Nagios XI server as an example:

```
/usr/local/nagios/libexec/check_nrpe -H 10.25.13.30 -t 30 -c check_users -a '-w 5 -c 10'
```

The `-a` is the arguments being sent to the NRPE agent.

- When `dont_blame_nrpe=1`, arguments will be accepted by the NRPE agent
  - This allows for dynamic commands, for example thresholds for warning and critical arguments can be adjusted on the Nagios XI server and nothing needs to be updated on the NRPE agent
  - However there is a risk that a potential security hole could be exploited (for example a malicious user could identify a way to delete all the files on the remote server)
- When `dont_blame_nrpe=0`, arguments will be rejected by the NRPE agent
  - This means commands need to be specifically defined on the NRPE agent. If you wanted to adjust the thresholds for warning and critical arguments, these would need to updated on each the NRPE agent
  - This limits the potential of a malicious user being able to do bad things to the remote server, as "unknown arguments" are not accepted.

To identify what setting your NRPE server has defined, log into the remote host as the root user and execute:

```
grep -e ^dont /usr/local/nagios/etc/nrpe.cfg
```

The expected output will be something like:

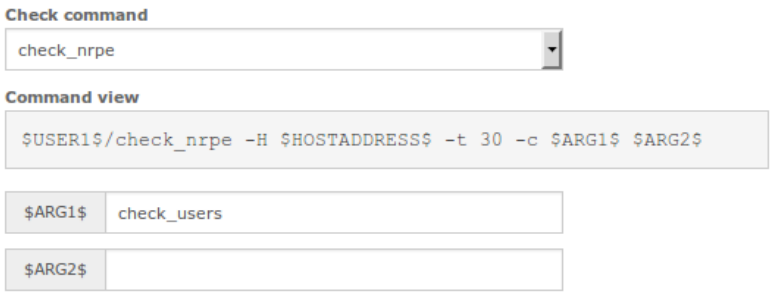
```
dont_blame_nrpe=1
```

No Arguments

To verify if your argument usage is consistent, compare the check in Nagios XI to the command directive in the remote host's `/usr/local/nagios/etc/nrpe.cfg` file. If you have declared all the arguments for a check in the `nrpe.cfg` file, then Nagios XI should pass no arguments other than the command itself. In the example below, the command directive check_users is defined to not pass any arguments:

```
command[check_users]=/usr/local/nagios/libexec/check_users -w 5 -c 10
```

We can verify the the arguments which are sent to the remote host from Nagios XI by navigating to **Configure** > **Core Config Manager** > **Services** and select the check_user service (in this example).



As you can see the service check is created to send no arguments other than the command name in `$ARG1$`:

```
check_command = check_nrpe
$ARG1$ = check_users
$ARG2$+ <blank>
```

Combined Arguments

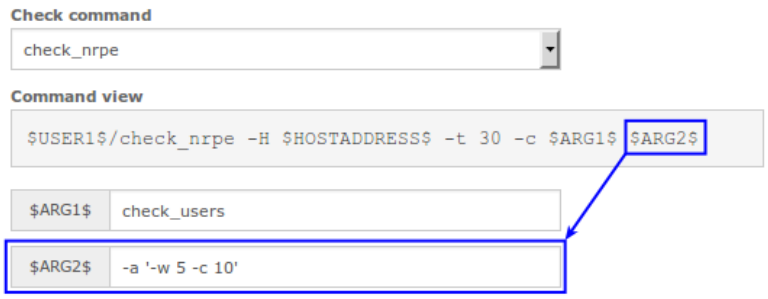
This format encapsulates all of the arguments into one field in Nagios XI and one $ARG1$ in the remote host's `nrpe.cfg` file. This is how Nagios sets up checks configured through the Linux Server and NRPE wizards. If you compiled NRPE from source for the remote host but are using the XI wizards to create checks, you will have to edit the command directive in the remote host's `nrpe.cfg` file.

`nrpe.cfg` command directive:

```
command[check_users]=/usr/local/nagios/libexec/check_users $ARG1$
```

The service check is set up in Nagios XI:

```
check_command: check_nrpe
$ARG1$ check_users
$ARG2$ -a '-w 5 -c 10'
$ARG3$+ <blank>
```



You can see that `check_nrpe` sends everything in the `$ARG2$` field. Remember the `-a` in the `$ARG2$` field is how `check_nrpe` sends arguments to the NRPE client. Because the arguments are enclosed in `'single quotes'` this is how the NRPE client receives this as `$ARG1$`.

Separate Arguments

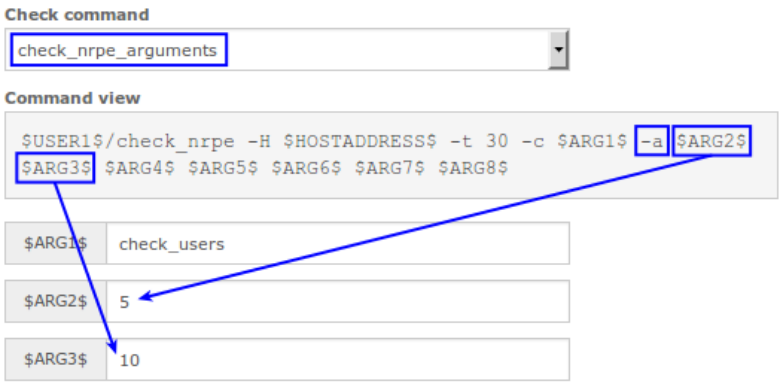
If you have setup multiple arguments for each threshold/option, Nagios XI will pass them in the same order:

`nrpe.cfg` command directive:

```
command[check_users]=/usr/local/nagios/libexec/check_users -w $ARG1$ -c $ARG2$
```

The service check is set up in Nagios XI:

```
check_command: check_nrpe_arguments
$ARG1$ check_users
$ARG2$ 5
$ARG3$ 10
$ARG4$+ <blank>
```



Notice the command directive expects `$ARG1$` for `-w` and `$ARG2$` for `-c` even though in Nagios XI they are `$ARG2$` and `$ARG3$`. This trips up beginners, this is because `check_nrpe` sends arguments after the `-a`, these are received by the NRPE client as `$ARG1$`, `$ARG2$` and so on.

Take note that the Check Command in the screenshot above is a custom check command that has been defined. In the previous example the `-a` was part of the `$ARG1$` variable. However in this example it was defined as part of the check command, which means you can put just the threshold values in `$ARG2$` and `$ARG3$`.

All of these argument configuration methods are valid, though it is best to choose one method and stick to it for consistency and ease of troubleshooting.

VII. CHECK_NRPE: Error Receiving Data From Daemon

This error is not to be confused with the error “CHECK_NRPE: Received 0 bytes from daemon” as they have separate causes. Most often, this error is experienced when using the no ssl switch (`-n`) with `check_nrpe` even though NRPE on the remote host was compiled with ssl enabled. There are very few instances where NRPE is best run without ssl, so if you added the `-n` switch to your check for testing reasons, make sure to remove the switch before deploying the check. If you have a reason for not using ssl, do note that you will have to compile NRPE without ssl to avoid this error when using the `-n` switch.

The other general cause of this error, though rare, happens when your check's `check_nrpe` timeout is set too low. To increase the timeout, refer to section **V. CHECK_NRPE: Socket Timeout After n Seconds** under the subsection **Increase Socket Timeout**.

VIII. NRPE: Unable To Read Output

This error implies that NRPE did not return any character output. Common causes are incorrect plugin paths in the `nrpe.cfg` file or that the remote host does not have NRPE installed. There are also cases where the wrong interpreter is invoked when running the remote plugin. Rarely, it is caused by trying to run a plugin that requires root privileges.

Incorrect Plugin Paths

Log onto the remote host as root and check the plugin paths in `/usr/local/nagios/etc/nrpe.cfg`. Try to browse to the plugin folder and make sure the plugins are listed. Sometimes when installing from a package repo, the commands in `nrpe.cfg` will have a path to a distribution specific location. If the nagios-plugins package was installed from source or moved over from another remote host, they me be located in a different directory.

The default location for the nagios-plugins can be found at `/usr/local/nagios/libexec/`. Open up your `nrpe.cfg` file on the remote host and take note of the path for the command directives (in bold):

```
command[check_users]=/usr/local/nagios/libexec/check_users $ARG1$
```

Change directory to this location and get a listing of this directories contents – you should see a large list of available plugins:

```
cd /usr/local/nagios/libexec/
ls
```

If the directory is blank or altogether missing, you are either missing the nagios-plugins, or they are in a different directory. You will need to change your `nrpe.cfg` file to reflect the location of your plugins.

Is NRPE Installed?

Next, make sure that NRPE is indeed installed on the remote host. Log onto the remote host as root and execute the following command:

```
find / -name nrpe
```

The results should be similar to the following:

```
/usr/local/nagios/bin/nrpe
/usr/local/nagios/etc/nrpe
---- Truncated --------
```

If NRPE is installed, refer to part V of this document **CHECK_NRPE: Socket Timeout After n Seconds**, under the section **Check The NRPE Service Status** to make sure that NRPE is actually running.

If the remote host does not have NRPE, you will have to install it. This can be done in a few different ways. We suggest installing NRPE via the Linux agent provided by Nagios XI. Please reference the below link for instructions:

Installing the Linux NRPE Monitoring Agent:

https://assets.nagios.com/downloads/nagiosxi/docs/Installing_The_XI_Linux_Agent.pdf

However if you need to compile NRPE from source, please reference the link below for instructions:

Installing and Configuring NRPE from Source:

https://support.nagios.com/kb/article.php?id=515

The wrong interpreter is used when running the remote plugin

Every plugin written in a non-compiled (interpreted) language should have a line at the top similar to any of the following:

```
#/bin/bash
#/bin/sh
#/usr/bin/python
```

This is appropriate based on the plugin's language and the system on which it is running. This line tells the system what interpreter to use when running the plugin, and is called a "shebang". This line allows one to run a plugin/script without explicitly declaring the interpreter to use, like so:

```
./check_example
```

If the shebang is missing, one must directly invoke an interpreter like so:

```
bash ./check_example
```

Problems in NRPE can arise from any of the following situations regarding the shebang:

1. The shebang references the wrong path (`/usr/bin/bash` instead of `/bin/bash`)
2. The shebang references a missing interpreter (`/bin/zsh` instead of `/bin/bash`)
3. The shebang references `/bin/sh` or some other symlink to a missing interpreter

In the last case, a simple `ls -l` against the symlink will reveal the ultimate interpreter being used:

```
[root@localhost libexec]# ls -l /bin/sh
lrwxrwxrwx 1 root root 4 Dec  8 07:12 /bin/sh -> bash
```

Ensure that the symlink points to the correct location of an existing interpreter.

The Plugin Requires "sudo" Privileges

It may be that your specific plugin requires root access. Depending on the Linux distribution on the remote host, you may have to consult the specific distribution's forums for instructions on how to give permission to the plugin and the user "`nagios`". This example will use `sudo` and the `/etc/sudoers` file.

You will need to create a rule in `/etc/sudoers` for the user `nagios` and the plugin script/binary requiring root access. Additionally, if the plugin script calls another system binary that requires root access, you will need to specify a rule for that binary as well (this problem is most often found with raid array plugins that require an access to a third party utility that requires root access). Log into the remote host as root and edit the sudoers file using the following command:

```
visudo
```

The visudo command opens the `/etc/sudoers` file in vi, and when you save your changes and exit it will validate you have a valid sudoers file. You will need to add the following line (replace <plugin> with the file name of your plugin):

```
nagios    ALL = NOPASSWD:/usr/local/nagios/libexec/<plugin>
```

If your plugin requires another binary on the system that is restricted to root, you will have to create an additional rule (replace `/path/to/binary` with the actual path to the required binary):

```
nagios    ALL = NOPASSWD:/path/to/binary
```

This will allow the user "nagios" (the user that NRPE runs as) to run the specified plugin as root (through sudo) without a password. You should be very careful with these settings, as incorrectly configuring it will lead to LARGE security vulnerabilities.

The final step is to add "sudo" to the command in the remote host's nrpe.cfg:

```
command[check_raid]=sudo /usr/local/nagios/libexec/check_raid
```

Now restart NRPE and verify the plugin works correctly. An additional test you can perform is to become the nagios user and then try and execute the plugin:

[**root**@centos12 ~]# `su nagios`
[**nagios**@centos12 root]$ `sudo /usr/local/nagios/libexec/check_raid`
output from plugin
[**nagios**@centos12 root]$ `exit`
[**root**@centos12 ~]#

Testing the plugin this way can help you see problems that you don't see through NRPE, like being prompted for a password.

IX. Command '[Your Plugin]' Not Defined

This error is very straight forward. Usually this is caused by a mismatch between the command name declared in Nagios XI to be check through NRPE and the actual command name of the command directive in the remote host's nrpe.cfg file. For more information see section **I. Return Code Of 127 Is Out Of Bounds - Plugin May Be Missing**.

However you may also receive a similar error when trying to pass the -c argument, for example:

```
./check_nrpe -H 10.25.13.30 -t 30 -c check_users -a -w 5 -c 10
NRPE: Command '10' not defined
```

This problem will occur in versions of `check_nrpe` before v3. What is happening here is that the initial `-c check_users` is being overwritten by the `-a -w 5` **`-c 10`**, as `check_nrpe` thinks the `-c 10` argument is the command argument, not one of the `-a` arguments.

If the arguments were enclosed in single quotes `-a '-w 5 -c 10'` then everything in the single quotes is sent as a string and not interpreted by the `check_nrpe` plugin. However your goal may be to have a dynamic command in your NRPE client and let Nagios XI send the arguments as multiple arguments. The solution is to upgrade the `check_nrpe` command on your Nagios XI server to the latest version, you don't specifically need up update the NRPE clients to fix the problem. Please refer to the following documentation on how to upgrade the `check_nrpe` command, it's at the end of the documentation under **Install check_nrpe Plugin Only**:

https://support.nagios.com/kb/article.php?id=515

The other solution is to change how you are sending the arguments to the NRPE client and also change the command definition on your NRPE client. Please refer to the section **VI. CHECK_NRPE: Received 0 Bytes From Daemon** in this document, there are several examples that show how.

X. Connection Refused By Host

This error usually relates to port/firewall issues or improperly configured "`allowed_hosts`" directives. See the following sections of this document for the pertinent troubleshooting steps:

III. CHECK_NRPE: Error - Could Not Complete SSL Handshake
IV. CHECK_NRPE: Socket Timeout After n Seconds

XI. No Output Returned From Plugin

There are a few causes of this error, two of which have solutions that have been covered other places in this document.

Permissions

The most common solution is to check the permissions on the check_nrpe binary on the Nagios XI server:

```
ls -la /usr/local/nagios/libexec/check_nrpe
```

The expected permissions should resemble:

```
-rwxrwxr-x. 1 nagios nagios  75444 Nov 21 01:38 check_nrpe
```

If not, change ownership to user/group “nagios” and fix up the permissions:

```
chown nagios:nagios /usr/local/nagios/libexec/check_nrpe
chmod u+rwx /usr/local/nagios/libexec/check_nrpe
chmod u+rx /usr/local/nagios/libexec/check_nrpe
```

This should be setup by default during the install process, but enough people have had the issues that it was worth noting here.

Missing Plugin

Another cause is a missing plugin file, though, in order to receive this error, you usually have to also be experiencing a secondary configuration issue. In order to resolve issues relating to missing plugins, see the section **I. Return Code of 127 Is Out Of Bounds - Plugin May Be Missing** for possible solutions.

Mismatch of Arguments between Nagios XI and nrpe.cfg

The final cause, and usually the secondary issue for those who found their plugin missing from the expected location, is an argument usage mismatch between the remote host's `nrpe.cfg` command directive and the arguments passed by Nagios XI through `check_nrpe`. This was covered in this document under the section **VI. CHECK_NRPE: Received 0 Bytes From Daemon**.

XII. Error While Loading Shared Libraries: libssl.so.0.9.8: Cannot Open Shared Object File: No Such File Or Directory

You are probably missing the ssl libraries on the remote host. This is an easy fix, as all you need to do is install openssl from the host's distribution repos. For example, in CentOS/RHEL, log onto your remote host and execute the following command:

```
yum install -y openssl
```

You can verify that it installed correctly with:

```
which openssl
```

The output should be similar to:

```
/usr/bin/openssl
```

If you use another distribution other than CentOS or RHEL, you may need to consult with their forums or run a search with the distribution's package manager to locate the correct package.

XIII. Warning: This Plugin Must Be Either Run As Root Or Setuid

This error is usually plugin specific and is most commonly experienced when trying to use a third-party hardware check plugin (most often disk smart checks and raid health plugins). You need to setup the sudoers file and associated config changes mentioned in this document earlier in the section **VII. NRPE: Unable To Read Output**, subsection: **The Plugin Requires 'sudo' Privileges**.

Sticky Bit

Alternatively, you could set the sticky bit on the plugin's permissions. Sudoers is considered safer, so only use this option if you understand the consequences:

```
chmod u+s /usr/local/nagios/libexec/<plugin>
```

XIV. Connection Refused Or Timed Out

This error is most often experienced when using the remote host as an NRPE proxy server to a network segment. It can also be caused by using an incorrect IP address or hostname in the check_nrpe command (rare in Nagios XI configurations). If you do use the remote host as an NRPE proxy, you may need to increase the maximum number of concurrent connections through xinetd. You need to add per_source = UNLIMITED to `/etc/xinetd.d/nrpe`.

Log onto your remote host at root and execute:

```
vi /etc/xinetd.d/nrpe
```

Add the following line to the file inside the closing "}":

```
per_source=UNLIMITED
```

Restart xinetd:

```
service xinetd restart
```

NSClient++ NRPE Specific Errors:

XV. NRPE Configuration

The following documentation explains how to ensure NSClient++ is correctly configured for NRPE:

Enabling The NRPE Listener In NSClient++ 0.3.x
https://assets.nagios.com/downloads/nagiosxi/docs/Enabling-The-NRPE-Listener-In-NSClient-0.3.x.pdf

Enabling The NRPE Listener In NSClient++ 0.4.x
https://assets.nagios.com/downloads/nagiosxi/docs/Enabling-the-NRPE-Listener-in-NSClient-0.4.x-for-Nagios-XI.pdf

Please check the relevant document to ensure your NSClient++ is correctly configured NRPE as it will help resolve most common problems.

XVI. UNKNOWN: No Handler For That Command

This is usually caused by a missing or incorrectly spelled handler (external alias) in the remote host's `nsc.ini` (v0.3.x) or `nsclient.ini` (v0.4.x). This file is typically found in `C:\Program Files\NSClient++\`. Check the spelling of the `check_nrpe` command for the service check in Nagios XI (the name of the command after the "`-c`"). It should match the spelling of the external alias in the nsclient config file. For example:

```
[External Alias]
alias_cpu=checkCPU warn=80 crit=90 time=5m time=1m time=30s
...[truncated]...
```

In the example above, the bolded "**`alias_cpu`**" is the handler and therefore the service check in Nagios should specify the check_nrpe command as "alias_cpu".

XVII. ERROR: Missing Argument Exception

This is usually due to clashing handler names (more than 1 of the same external alias name). It can also be caused by an argument mismatch as well. Read over the section **VI. CHECK_NRPE: Received 0 Bytes From Daemon** of this document, specifically the **No Arguments** section for an in depth *explanation* of this problem.

Instead of editing the command directives in your `nrpe.cfg` file (which does not exist as this is a windows remote host), edit the "`[External Alias]`" section of C:\Program Files\NSClient++\`NSC.ini` (v0.3.x) or `nsclient.ini` (v0.4.x). Make sure your argument usage is consistent between the `NSC.ini/nsclient.ini` and the Nagios XI service check.

XVIII. General Troubleshooting Tips

When Troubleshooting NRPE issues, there is a general order of procedures for drilling down the problem. Start with the plugin itself, and then move to NRPE, and finally check your argument usage. If you follow the general steps below before dealing with support, your issue may be solved faster than expected as these are always the first steps a Nagios XI support representative will ask you to perform:

1. Test The Plugin Locally First. Log onto your remote server as root and copy the plugin to your plugins directory (`/usr/local/nagios/libexec/`) on the remote host and run it:

```
/usr/local/nagios/libexec/<name of plugin>
```

If it does not work as expected, you may want to check the plugin's usage as you may find some hints to why it is not working:

```
/usr/local/nagios/libexec/<name of plugin> -h
```

You may have to set some thresholds, usually warning (`-w`) and critical (`-c`) for a large number of plugins before they will work correctly. Once the plugin has been tested and working locally from the remote host, create a command directive for it in the `nrpe.cfg` file. Take a mental note of how you setup your arguments.

2. Verify That NRPE Is Working Locally And Open To Requests From The XI Server:
On the remote host, run:

```
service xinetd status
```

Or (for init script systems):

```
service nrpe status
```

If NRPE is not running, follow the steps in **Part III** of this document. If NRPE is running, move on to testing the connection to the remote host from the XI server with `check_nrpe`. Log onto the Nagios XI server as root and run the following command inserting the actual remote host IP address:

```
/usr/local/nagios/libexec/check_nrpe -H <remote host ip>
```

The command above should return the NRPE version of the remote host. If not, follow the steps in **Part IV** of this document. If the version of NRPE is returned successfully, move on to step 3.

An important step in the above command was to use the IP ADDRESS of the remote host. If your Nagios XI host object uses a DNS record, use that in the command above. It's possible that the DNS record is incorrect.
Another similar mistake is when you use CCM to copy an existing host to provision a new host. The newly copied host still has the address of the host it was copied from (your forgot to change it). Your command line tests work OK because you are forced to type the address at the command line. A good troubleshooting technique is to make sure when you do your testing at the command line, make sure the arguments you type in match the values in the Nagios XI objects.

3. Try The Full Command From The Command Line Interface On The XI Server:
From the Nagios XI server command line interface, run the following command:

```
/usr/local/nagios/libexec/check_nrpe -H <remote host ip> -c <command and arguments>
```

You will need to replace the remote host IP address and match your command and arguments to your command directives in your remote host nrpe.cfg.

If you do not get the expected output, check the plugin usage again to make sure your syntax is correct. As mentioned in the last step, a good troubleshooting technique is to make sure when you do your testing at the command line, make sure the arguments you type in match the values in the Nagios XI objects. Refer to **Part IX** of this document for information on argument usage. If the plugin does output the expected data, move on to step 4.

4. Setup The Service Check In XI:
Create a new service for the check by navigating within the Nagios XI web interface **Configure** > **Core Config Manager** > **Monitoring** > **Services** > **Add New.** Specify the **Config Name** and **Description** for the check. Use *check_nrpe* in the **Check_command** drop-down.

Next set up the command arguments under **Command view**.

$ARG1$ is the remote command to be sent to the remote host through NRPE. This must match the command directive in the nrpe.cfg.

$ARG2$ is used for extra command arguments. Again, if you have defined any in the remote host's nrpe.cfg..

The check needs to be applied to a host, so click the **Manage Hosts** button. Select a host from the list and click **Add Selected**. You should see the host appear in the right hand pane under **Assigned**. Now click **Close**.

Click the **Check Settings** tab. At minimum, we need to setup check intervals, attempts, and a period.

- **Check interval** specifies how often the check is run
- **Retry interval** specifies the time between check retries when the service check has failed (SOFT STATE)
- **Max check attempts** specifies the number of retries a check will attempt before it is marked as a HARD STATE fail
- The last required setting to set on this tab is the **Check period**
- This specifies what “time period” the check should run and can be configured for certain days and time frames. **xi_timeperiod_24x7** will be fine for this example.

Last, click the **Alert Settings** and set the **Notification period** to "xi_timeperiod_24x7", or to the time period of your choice. This specifies the time period for notifications. (emails, SMS, etc.) Click **Manage Contacts** and add a contact to the check if you want.

Finally, click **Save** and **Apply Configuration**.

Alternatively you can run the **NRPE** or **Linux Server** configuration wizards which do all this for you.

Now when you navigate to **Service Detail** you will see your service check listed. It may take a minute for the service to change from pending to a STATE. From this page you can verify that your plugin is executing as expected.

Final Thoughts

Hopefully you have been able to resolve your issues related to NRPE using the information within this document. If you have any additional support related questions please visit us at our Nagios Support Forums:

https://support.nagios.com/forum