Nagios XI – How To Monitor JMX With Nagios XI

Nagios®

The Industry Standard in IT Infrastructure Monitoring

Purpose

This document will cover how to monitor Java application servers using the **`check_jmx`** plugin within Nagios® XI™, in order for users to be notified when java applications are not functioning properly.

Target Audience

This document is intended for use by Nagios XI Administrators who wish to monitor JMX applications.

Prerequisites

This document assumes you have the following:

- A remote server running JMX
- Nagios XI server with a network route to the JMX server

The monitoring plugin used in this documentation will be executed on the remote server. This means that the NRPE agent will need to be installed on your remote server. Follow our Linux-Agent installation document below:

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

The documentation will assume that you have installed NRPE before continuing.

Editing Files

In steps of this documentation you will be required to edit files. This documentation will use the **vi** text editor. 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

check_jmx Plugin Overview

This document will specifically cover the configuration of the **`check_jmx`** plugin, which can be downloaded from:

https://exchange.nagios.org/directory/Plugins/Java-Applications-and-Servers/check_jmx/details

The **check_jmx** plugin can:

- Monitor standard Java JMX implementation by exposing memory, threads, OS, and garbage collector parameters
- Monitor Tomcat's multiple parameters such as requests, processing time, threads, etc.
- Monitor Spring framework by exposing Java beans parameters to JMX
- Expose any attributes to JMX by declaration or explicitly
- Monitor any java object/attribute accessible through JMX, including jboss objects and beans

Installing The Plugin

Establish a terminal session to your JMX server and execute the following commands to download and install the plugin:

```
cd /tmp
```

The next command is all one line, it's too long for the document so it wraps onto two lines. It's important that the "double quotes" are used and the `-O` is a capital "oh":

```
wget "https://exchange.nagios.org/components/com_mtree/attachment.php?link_id=1274&cf_id=24"
-O check_jmx.tgz

tar xzf check_jmx.tgz
chmod -R +x /tmp/check_jmx/nagios/plugin/*
chown -R nagios:nagios /tmp/check_jmx/nagios/plugin/*
cp /tmp/check_jmx/nagios/plugin/* /usr/local/nagios/libexec
```

Test The Plugin

In your terminal session execute the following commands to test the plugin:

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

./check_jmx -U service:jmx:rmi:///jndi/rmi://localhost:<port>/jmxrmi -O java.lang:type=Memory
-A HeapMemoryUsage -K used -I HeapMemoryUsage -J used -vvvv -w 4248302272 -c 5498760192
```

The expected output of the above command should be similar to:

```
JMX OK HeapMemoryUsage.used=97989080{committed=954204160;init=964689920;max=954204160;used=97989080}
```

You will have noticed that the command had the following:

```
localhost:<port>
```

You will need to replace the port your JMX application uses, in all future examples port **7199** will be used.

Configure NRPE

In order for Nagios XI to execute this plugin, you need to define a command for this plugin in the `nrpe.cfg` on the JMX server. Edit the `/usr/local/nagios/etc/nrpe.cfg` file by executing the following command:

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

Add the following line to the end of the file:

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

After saving these changes, restart the `xinetd` service on the JMX Server (or the nrpe service if you compiled from source) by running the following command.

```
service xinetd restart
```

Now to test the check from the Nagios XI server. Establish a terminal session to your Nagios XI server and execute the following command, making sure to replace **<jmx_server_ip>** with the IP address of your JMX server:

```
/usr/local/nagios/libexec/check_nrpe -H <jmx_server_ip> -c check_jmx -a '-U
service:jmx:rmi:///jndi/rmi://127.0.0.1:7199/jmxrmi -O java.lang:type=Memory -A
HeapMemoryUsage -K used -I HeapMemoryUsage -J used -vvvv -w 4248302272 -c 5498760192'
```

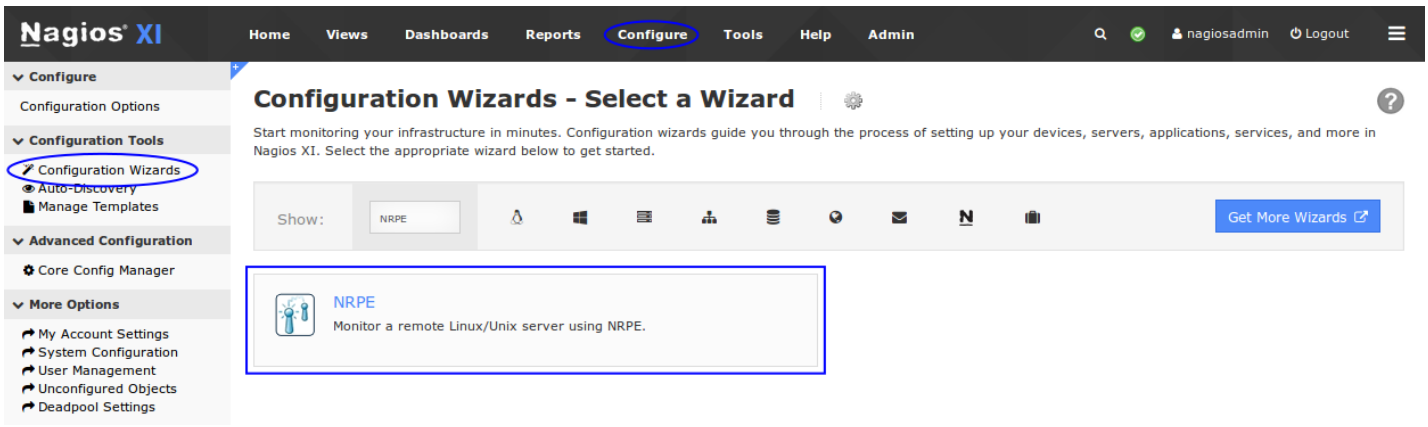
You should see output similar to:

```
JMX OK HeapMemoryUsage.used=125412576{committed=954204160;init=964689920;max=954204160;used=125412576}
```

Now that NRPE has been configured correctly on your JMX server the next step is to create the monitoring configuration in Nagios XI.

Create Nagios Monitoring Objects

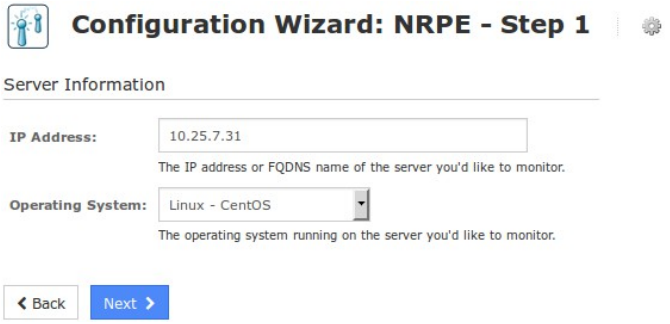
In this example you will use the **NRPE Configuration Wizard** which will create the host and service objects. Navigate via the top menu bar to **Configure** > **Run a configuring wizard**, and select the **NRPE** wizard. In the following screenshot you can see how the search field allows you to quickly find a wizard.



On Step 1 you will be asked to supply the **address** of the JMX server.

You will also have to select the **Operating System**, in this case it is CentOS.

Click Next to progress to step 2.



On step 2 you will configure all of the options for monitoring.

To start off with make sure a valid **Host Name** has been entered.

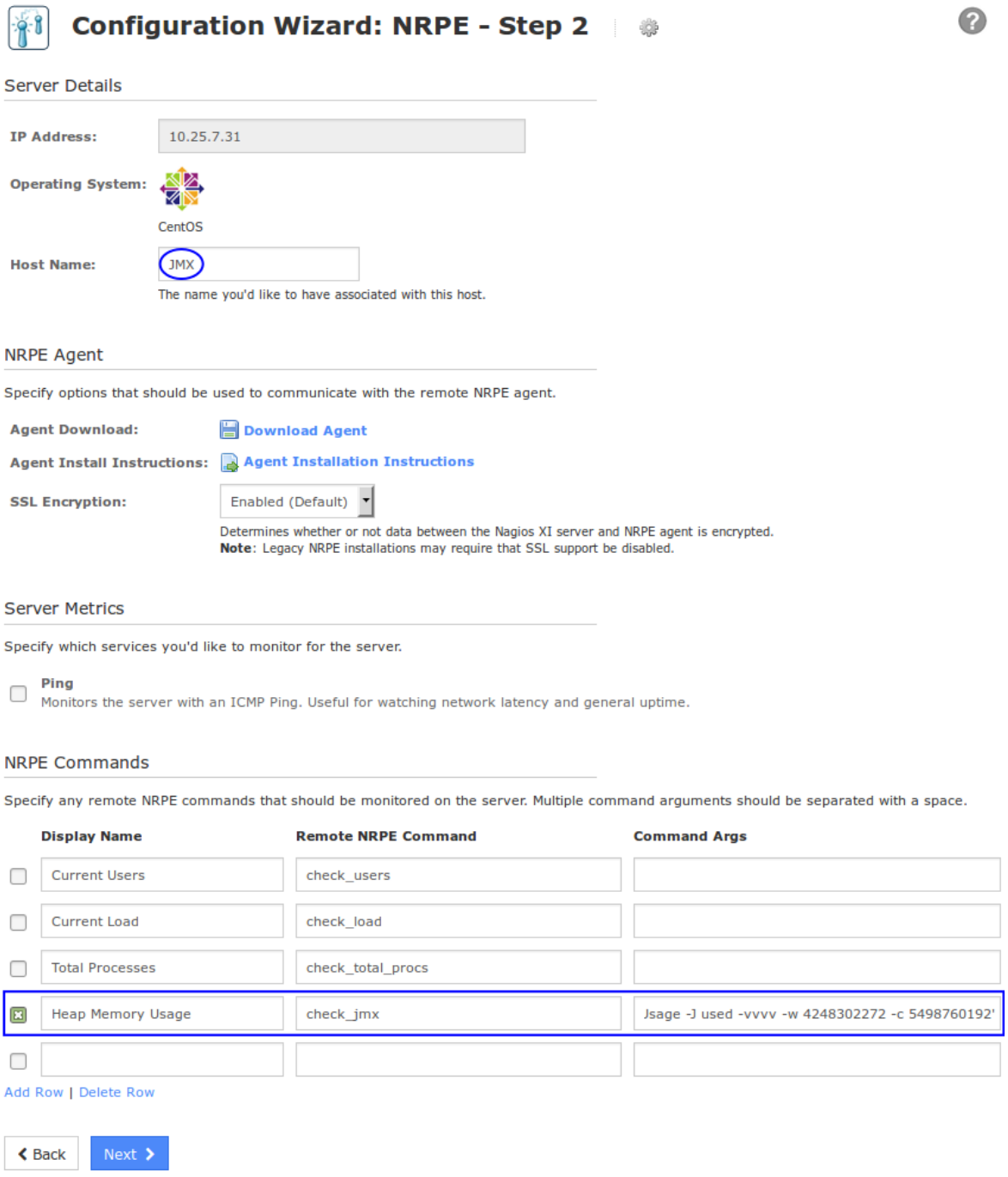
The NRPE Agent section can be ignored because you have already installed it.

The NRPE wizard allows you to specify which NRPE commands should be executed and monitored and what display name (service description) should be associated with each command.

In the screenshot on the next page you can see the command has been defined for the `check_jmx` check.

In the **Command Args** field the following has been typed (the field is too short to display it all):

```
'-U service:jmx:rmi:///jndi/rmi://127.0.0.1:7199/jmxrmi -O java.lang:type=Memory -A
HeapMemoryUsage -K used -I HeapMemoryUsage -J used -vvvv -w 4248302272 -c 5498760192'
```



Click **Next** and then complete the wizard by choosing the required options in Step 3 – Step 5. To finish up, click on **Finish** in the final step of the wizard. This will create the new hosts and service and begin monitoring.

Once the wizard applies the configuration, click the **View status details for xxxxx** link to see the new host and service that was created.

Host	Service	Status	Duration	Attempt	Last Check	Status Information
JMX	Heap Memory Usage	Ok	54s	1/5	2017-02-15 11:48:46	JMX OK HeapMemoryUsage.used=106150032{committed=954204160:init=964689920:max=954204160:used=106150032}

You are now monitoring your JMX server, it's as simple as that.

Note: In the wizard, in the **Command Args** field pay special attention to how the entire argument is surrounded by `'single quotes'`:

```
'-U service:jmx:rmi:///jndi/rmi://127.0.0.1:7199/jmxrmi -O java.lang:type=Memory -A
HeapMemoryUsage -K used -I HeapMemoryUsage -J used -vvvv -w 4248302272 -c 5498760192'
```

Additional check_jmx Argument Configurations

Below you will find a few different examples of the types of checks that can be done with `check_jmx`. Your JMX server may or may not support some of the examples but will most likely support many more options.

Note: The **`-w xxx`** and **-c yyy** threshold values need to be replaced with values relative to the object being monitored.

Garbage Collection:

```
-U service:jmx:rmi:///jndi/rmi://127.0.0.1:7199/jmxrmi -O
java.lang:type=GarbageCollector,name=ConcurrentMarkSweep -A LastGcInfo -K duration -u ms
-vvvv -w xxx -c yyy
```

Thread Count:

```
-U service:jmx:rmi:///jndi/rmi://127.0.0.1:7199/jmxrmi -O java.lang:type=Threading -A
ThreadCount -w xxx -c yyy
```

Available Connections in Pool:

```
-U service:jmx:rmi:///jndi/rmi://127.0.0.1:7199/jmxrmi -O
jboss.jca:name=JmsXA,service=ManagedConnectionPool -A AvailableConnectionCount -w xxx -c yyy
```

System Load:

```
-U service:jmx:rmi:///jndi/rmi://127.0.0.1:7199/jmxrmi -O java.lang:type=OperatingSystem -A
SystemLoadAverage -w xxx -c yyy
```

Regular Expression Matching:

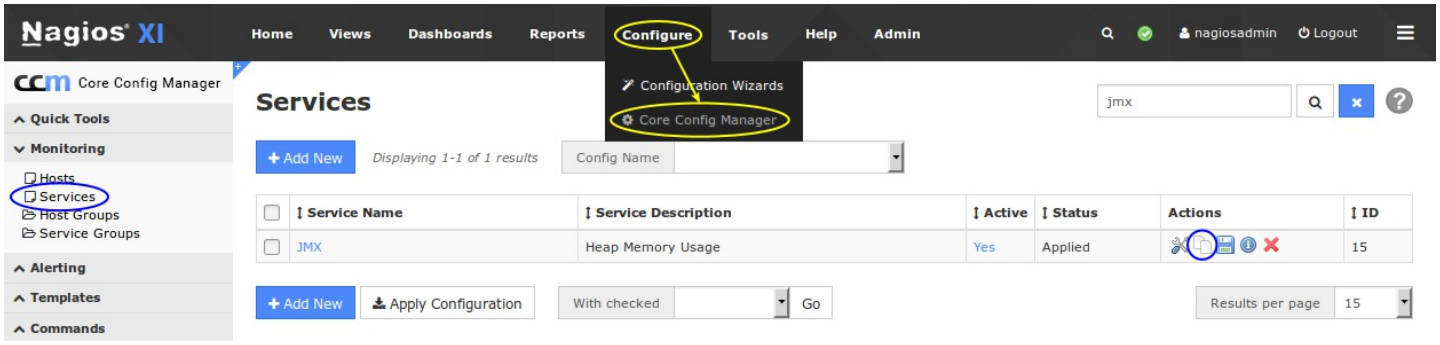
```
-U service:jmx:rmi:///jndi/rmi://127.0.0.1:7199/jmxrmi -O <object>:name=<Bean name> -A Status
-w <regular expression> -c <regular expression>
```

To add additional services you can:

1. Run the NRPE Wizard again
2. Copy an existing service and change the required options

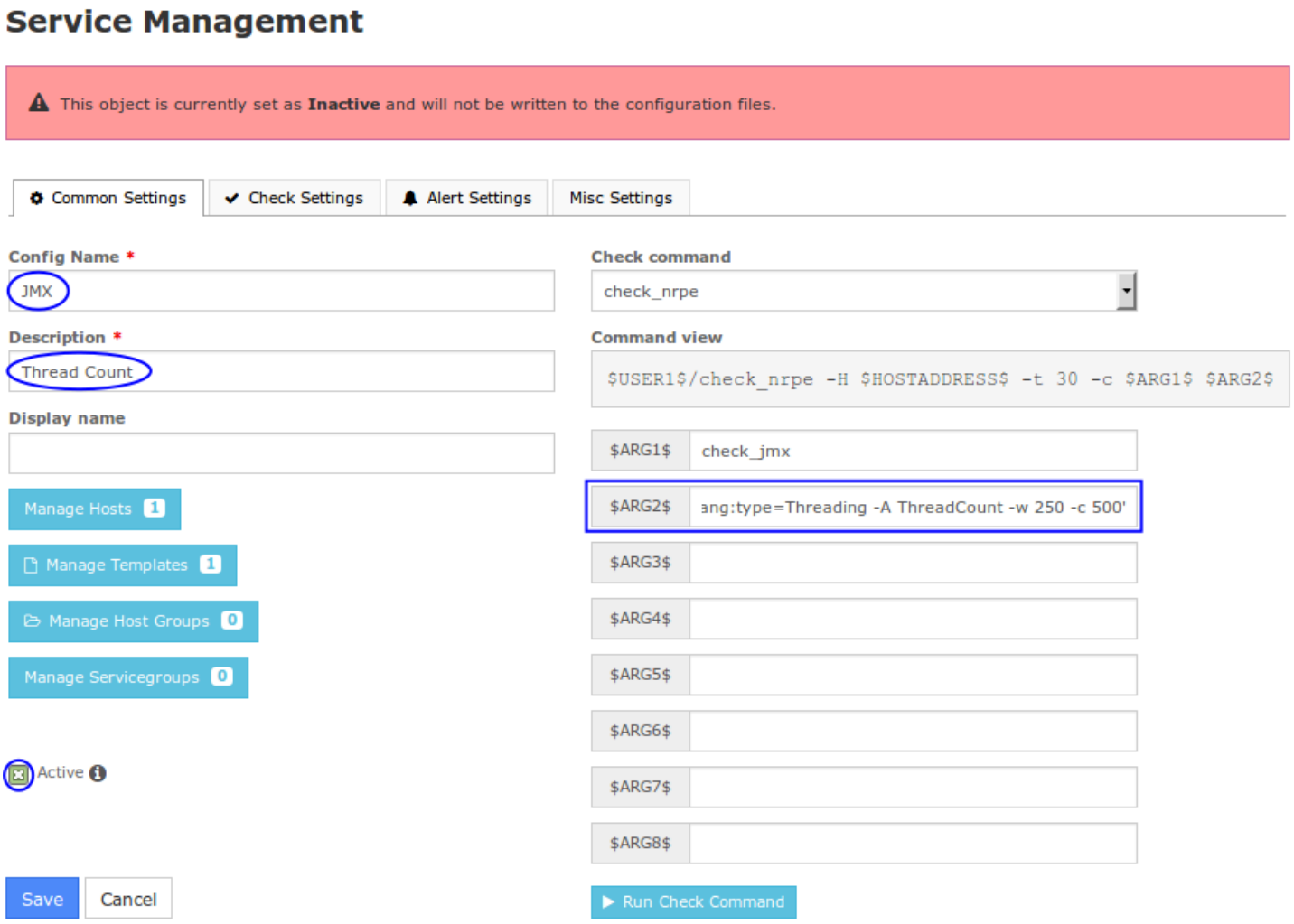
The following steps show you how to copy an existing service to create an additional service. This example will add the **Thread Count** check shown above.

In Nagios XI, navigate via the top menu bar to **Configure** > **Core Configuration Manager** and then **Monitoring** > **Services**.



Click the **Copy** icon for the existing service. This will create a copy of the existing service with it's name appended with `_copy_1`.

Click the **Modify** icon to edit the copied service.



In the **Config Name** field remove the **`_copy_1`**.

Change the **Description** field remove to **`Thread Count`**.

In the **`$ARG2$`** field change the text so that it correctly matches the check being performed, this is what is being used in this example:

```
-a '-U service:jmx:rmi:///jndi/rmi://127.0.0.1:7199/jmxrmi -O java.lang:type=Threading -A
ThreadCount -w 250 -c 500'
```

Finally make sure the **Active** checkbox is **checked**.

Click the **Save** button and then **Apply Configuration**.

Once the configuration has been applied you will have a new **Thread Count** service:

Host	Service	Status	Duration	Attempt	Last Check	Status Information
JMX	Heap Memory Usage	Ok	1h 54m 56s	1/5	2017-02-15 13:38:45	JMX OK HeapMemoryUsage.used=140624720{committed=954204160:init=964689920:max=954204160:used=140624720}
	Thread Count	Ok	10s	1/5	2017-02-15 13:43:32	JMX OK ThreadCount=81

That is how you use Core Configuration Manager to copy an existing service to create a new service.

Finishing Up

If you experience difficulties implementing `check_jmx`, please post your questions in the Nagios Support Forums at:

https://support.nagios.com/

You may also find help from the original developer on the Nagios Exchange at:

https://exchange.nagios.org/directory/Plugins/Java-Applications-and-Servers/check_jmx/details