Integrating AutoIT With Nagios XI 2024

Purpose

This document describes how to automate and monitor Windows actions, tasks, and tests using AutoIt and Nagios. AutoIt is an extremely powerful scripting recorder and editor. This document is an example of how it can be used.

Prerequisites

You must have NSClient++ installed on the Windows machine. NSClient++ must be configured to allow NRPE checks from the Nagios XI server. Information on installing and configuring NSClient++ can be found in the following documents:

- Installing The XI 2024 Windows Agent
- Configuring The XI Windows Agent
- Enabling The NRPE Listener In NSClient++ 0.4.x For Nagios XI 2024

This guide is specifically aimed at NSClient++ v 0.4.x or newer, the previous 0.3.x version of NSClient++ is no longer supported by the application developer.

In addition to this you will need to install the **Firefox** web browser, which can be downloaded from:

http://www.mozilla.org

Installing AutoIt

Both of the following packages must be installed on the remote Windows system(s) where the automated tasks will be executed.

- AutoIt – http://www.autoitscript.com/site/autoit/downloads/
- SciTE Script Editor – http://www.autoitscript.com/site/autoit-script-editor/downloads/

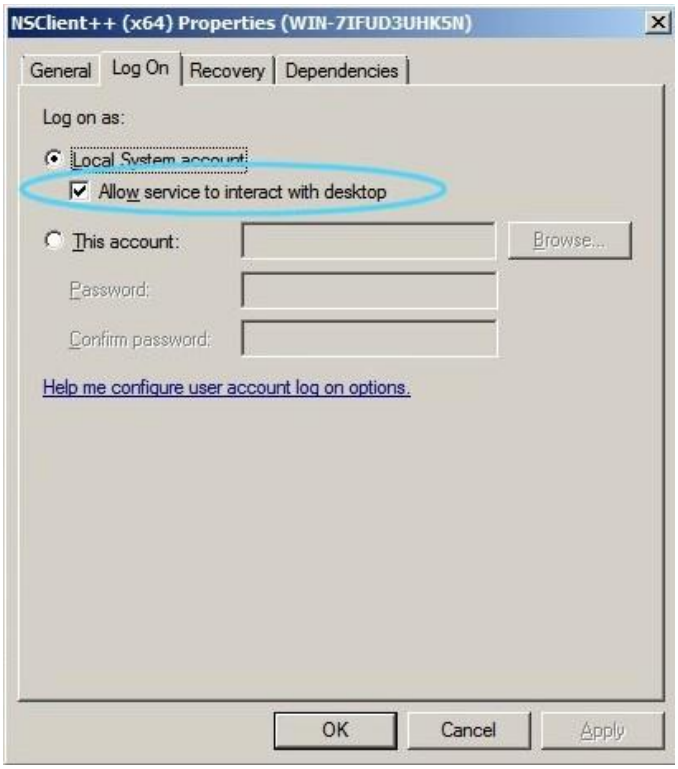
Once both applications are downloaded and installed on the remote Windows system, automation scripts can either be recorded or written by hand. More information, along with guides and tutorials on writing and recording scripts can be found at the main AutoIt site (www.autoitscript.com).

Configure NSClient++ Service

AutoIT requires control of the Windows workspace, so turning on **Allow service to interact with desktop** for the NSClient++ service is recommended.

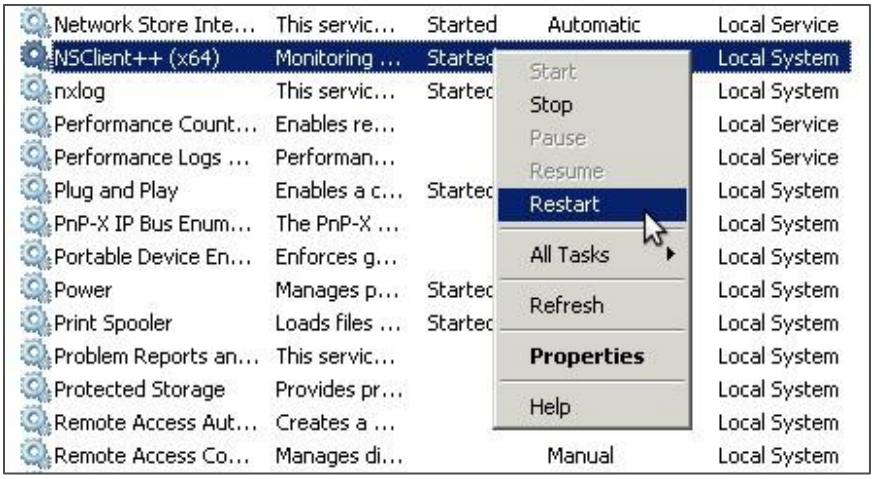
In Windows open the **Services** console under **Administrative Tools**. Running `services.msc` will open the Services console.

- Find the **NSClient++** service in the list.
- Right click on **NSClient++** and select **Properties**
- Click the **Log On** tab
- Check the box **Allow service to interact with desktop**
- Click **OK**



Restart NSClient++ Service

Right click the NSClient++ service and select **Restart**.



Create Script

This example shows how to create a script that will open firefox.exe, select the URL box and enter a URL. After the URL is entered, a timer starts which runs until the web page has completely loaded.

After the page is loaded and the timer ends, the time required to load the page is output to the console.

Open the **SciTE Editor** and paste this example script into the editor.

```
Func _WinWaitActivate($title,$text,$timeout=0)
   WinWait($title,$text,$timeout)
   If Not WinActive($title,$text) Then WinActivate($title,$text)
   WinWaitActive($title,$text,$timeout)
EndFunc
$title_string = "Yahoo"
Local $begin = TimerInit()
Run("C:\Program Files (x86)\Mozilla Firefox\firefox.exe www.yahoo.com")
_WinWaitActivate($title_string,"")
Local $dif = TimerDiff($begin)
$time_string = $dif
WinClose($title_string,"")
ConsoleWrite($time_string)
```

The script directs Firefox to open the **www.yahoo.com** web page, see the address (line 9).

Once the page has loaded, the title of the web browser becomes **Yahoo**. The script is looking for the string **Yahoo**, see variable declaration on line 7 of the script **$title_string = "Yahoo"**.

Note: If the script does not match the string, open the URL in a web browser to make sure the title of the page matches the expected title "Yahoo", because Yahoo may change the title (also different countries show different versions of the web page).

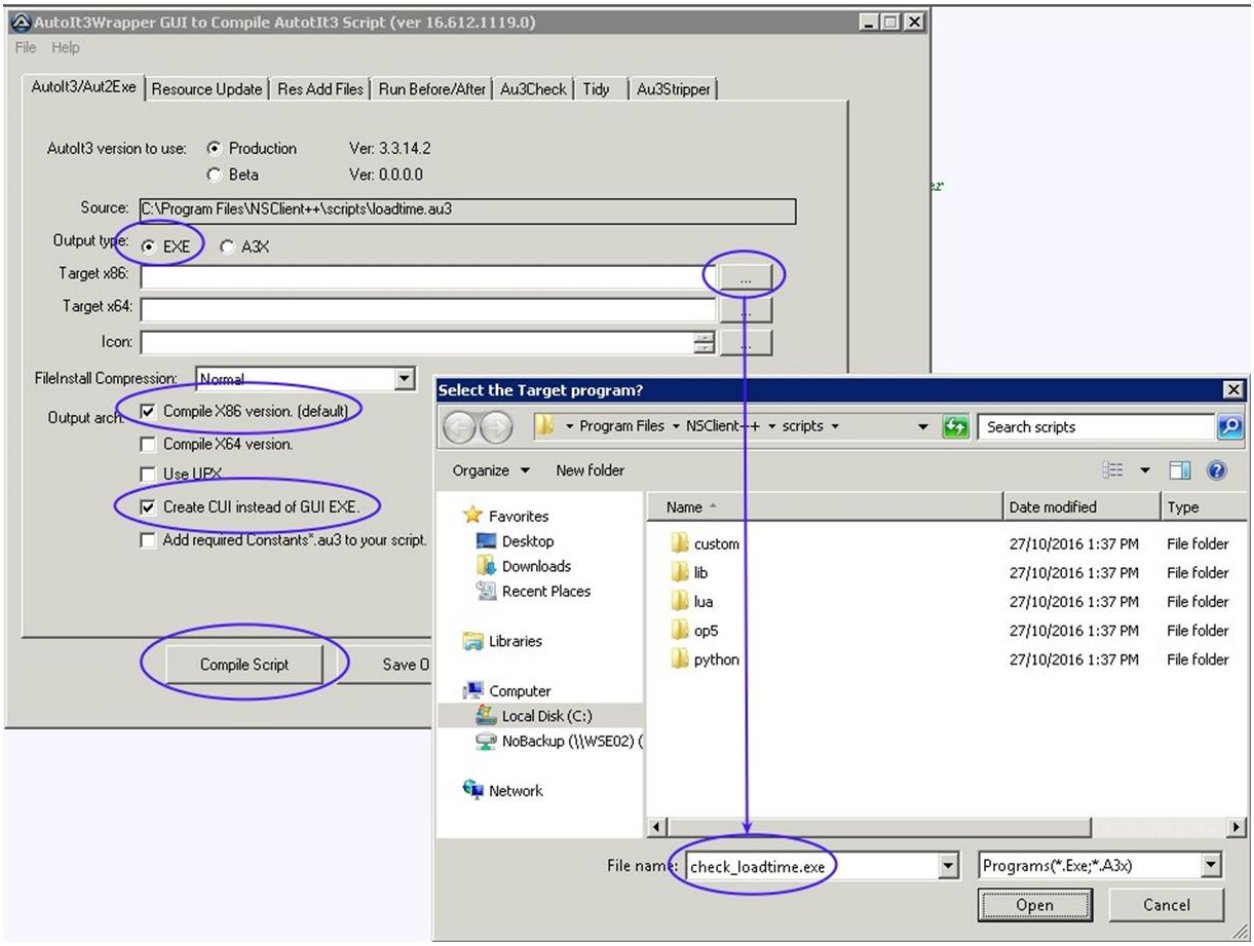
Once it detects the string, the time it took is calculated (line 11) and the time difference is stored in the variable `$time_string` (line 12). Finally, the script outputs the variable `$time_string` to the console.

For reuse, save this script to the `C:\Program Files\NSClient++\scripts\` directory. This example saves the file as `loadtime.au3`. Saving the script is not a requirement, but it may be useful, later.

The next step is to compile this script into an EXE file, this is necessary, so the output is correctly written to the console.

Compile Script

- To compile the script, click the drop-down menu **Tools** and select **Compile**.
- For the **Target x86** field click the **"…"** button.
- Browse to `C:\Program Files\NSClient++\scripts\`
- Name it, `check_loadtime.exe`
- Click **Open**
- Compile X86 version = **Ticked**
- Create CUI instead of GUI.EXE = **Ticked**
- Click the **Compile Script** button to create `check_loadtime.exe`



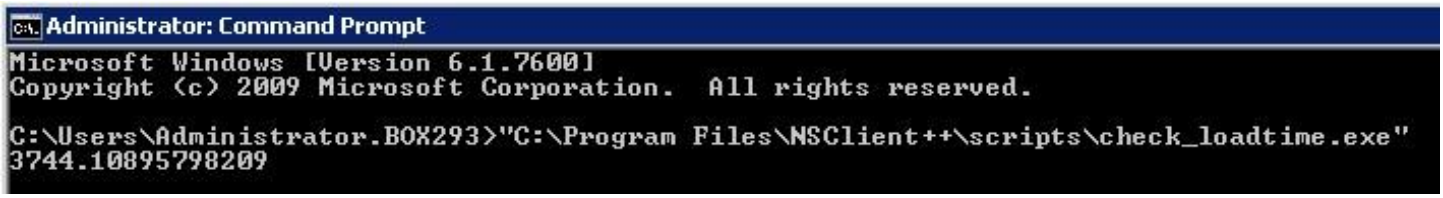
A dialogue window will appear while the script is being compiled. When the compile finishes you will be returned to the **SciTE Editor**.

At this point we have a simple AutoIt script created. You can test it by opening a command prompt and typing the following command:

```
"C:\Program Files\NSClient++\scripts\check_loadtime.exe"
```

Do not move the mouse or type on the keyboard until all the actions have been performed and the console output appears.

The output should look like the following:



The next step will be to configure NSClient++ to execute `check_loadtime.exe` so that Nagios XI server can execute it.

Configure NSClient++

Edit the NSClient++ configuration file, by opening the file `C:\Program Files\NSClient++\nsclient.ini` in Notepad.

Locate the External Scripts section. If NRPE checks are already in use, there should be commands listed. If not add this directly below `[/settings/external scripts/scripts]`:

```
[/settings/external scripts/scripts]
check_loadtime = scripts\check_loadtime.exe
```

Once this has been added, save the nsclient.ini file and restart NSClient++ (the same way as earlier in section Restart NSClient++ Service.

Configuration of the Windows machine is complete.

Next, combine this check on the windows side with a plugin called **check_autoit_timer**. This plugin will take the data the **AutoIt** script outputs and turn it into something readable, that can be used to trigger warning and critical thresholds.

Add A Plugin To Nagios

To automate the **AutoIt** script and monitor its output with Nagios, a plugin will need to be uploaded to Nagios XI.

The **check_autoit_timer.sh** plugin used in this example **AutoIt** script can be downloaded from the Nagios Exchange:

https://exchange.nagios.org/directory/Plugins/Operating-Systems/Windows/NRPE/autoIT-Timer-plugin/details

Once the plugin has been downloaded onto the Nagios XI server, open **Nagios XI** and navigate to **Admin** > **System Extensions** > **Manage Plugins**.

Upload the plugin using the **Browse** and **Upload Plugin** buttons.

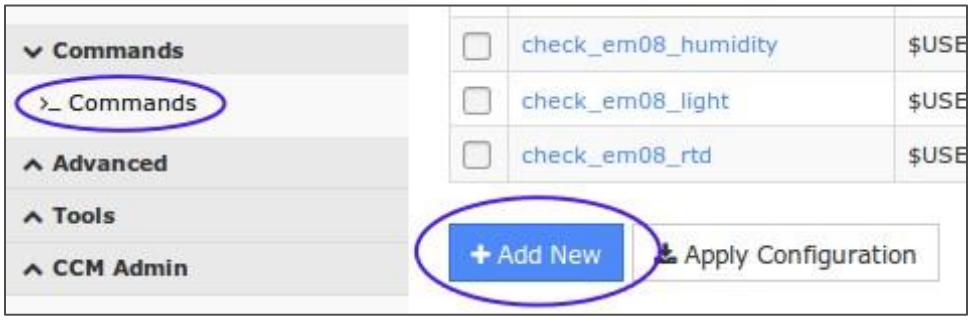
Creating The Check In Nagios XI

Now the check must be configured in the Nagios XI Web Interface using Core Configuration Manager (CCM).

The first step will be to create a custom command for this check.

Create Check Command

1. Navigate to **Configure > Core Config Manager (CCM)**
2. In the left pane expand **Commands** and then click **>_ Commands**.
3. Click the **Add New** button



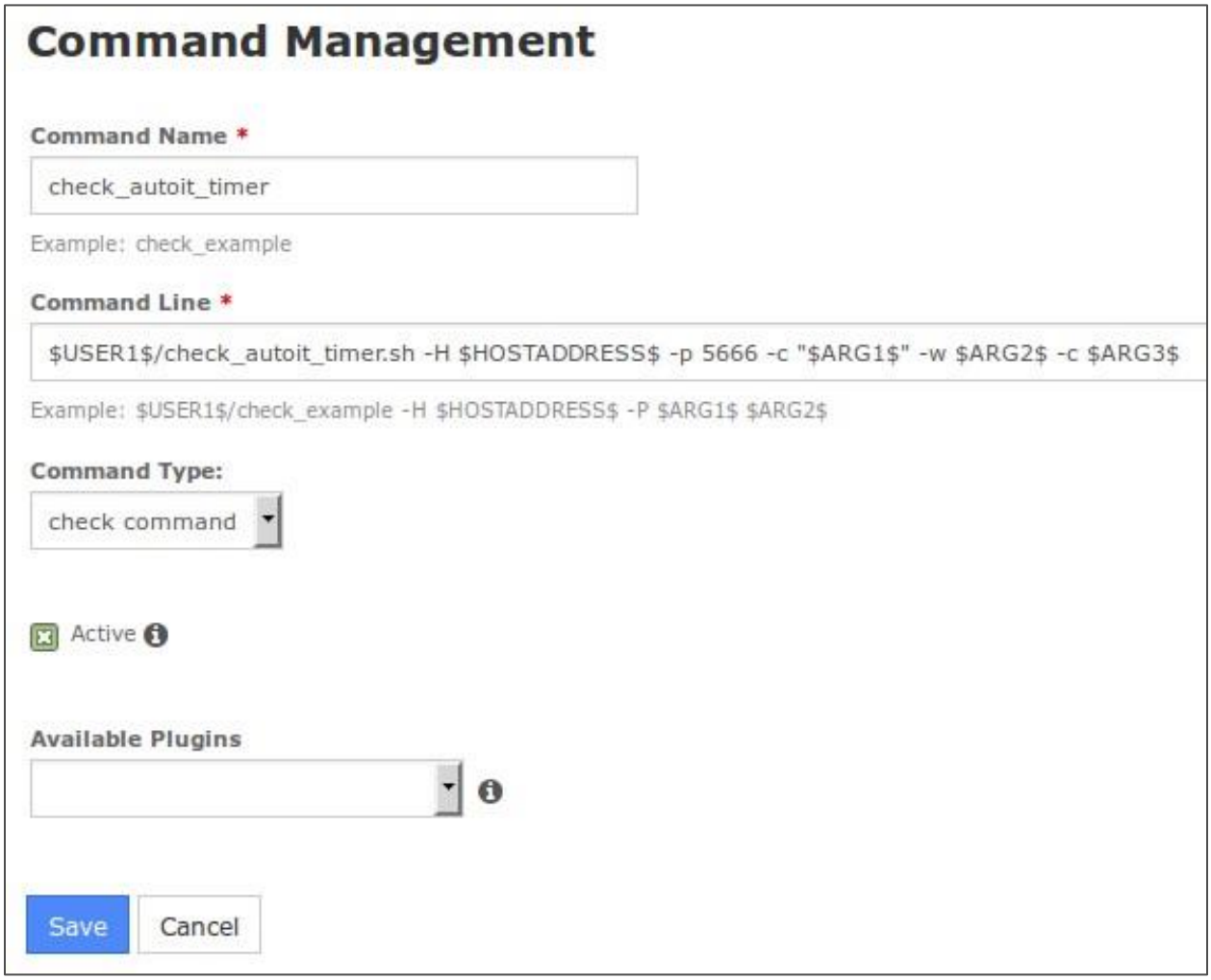
4. The Command Management page will open. Populate the fields with the following values:

   Command Name: `check_autoit_timer`

   Command Line: `$USER1$/check_autoit_timer.sh -H $HOSTADDRESS$ -p 5666 -c "$ARG1$" -w $ARG2$ -c $ARG3$`

Command Type: `check command`

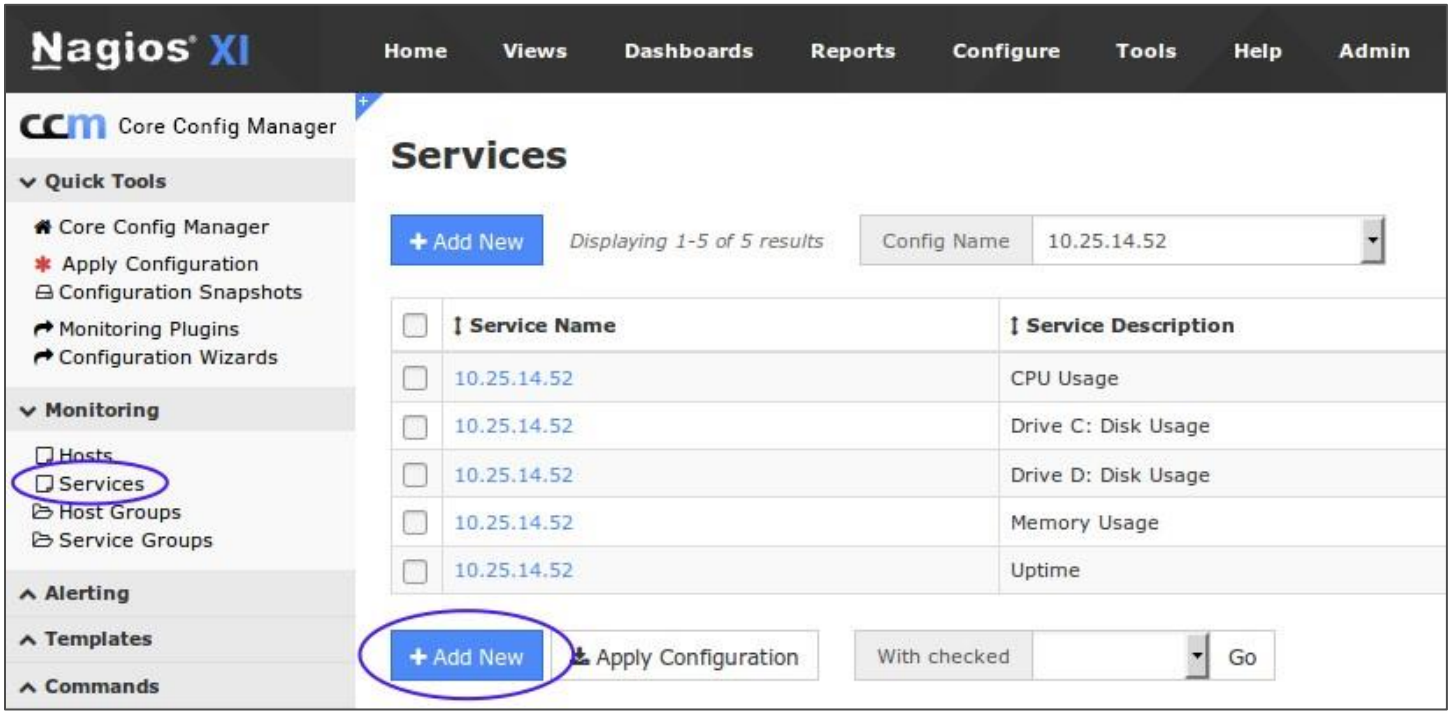
Active: `checked`



5. Click the **Save** button to create this new command.

Create Service

The final step is to create a new service definition that is associated with the remote windows host. This example assumes that this Windows host is already being monitored, so a HOST object already exists. If not, run the **Windows Server Configuration Wizard** and then return to this step. This guide uses the host **10.25.14.52** as an example.



1. In the left pane expand **Monitoring** and then click **Services**.
2. Click the **Add New** button.

   Common Settings tab

   Config Name: `10.25.14.52`

   Description: `AutoIt Timer`
3. Click the **Manage Hosts** button
4. Select `10.25.14.52` in the left pane and click the **Add Selected >** button
5. Click the **Close** button

   For this service use the **generic-service** template as it has a lot of the required directives already configured

6. Click the **Manage Templates** button
7. Select **generic-service** in the left pane and click the **Add Selected >** button
8. Click the **Close** button

Check command (drop down list)

check_autoit_timer

`$ARG1$:` check_loadtime

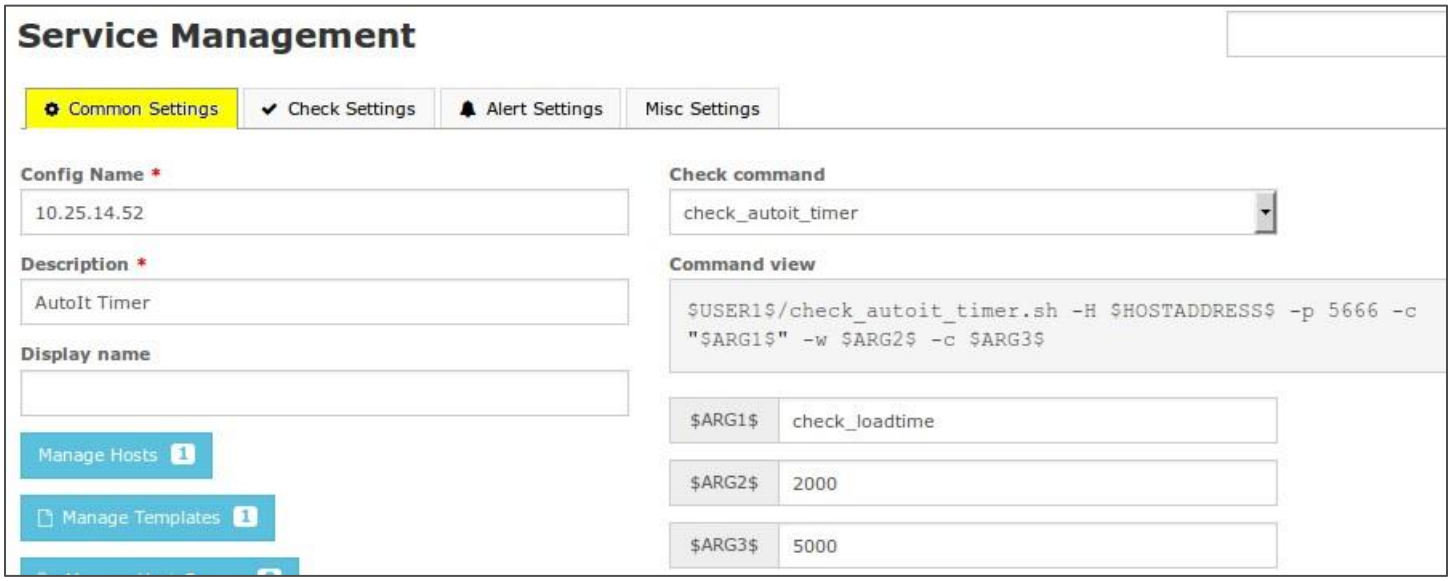
`$ARG2$:` 2000

This is the warning threshold (in milliseconds), if the number returned from the AutoIt script exceeds this number then the service will go into a WARNING state.

`$ARG3$:` 5000

This is the critical threshold (in milliseconds), if the number returned from the AutoIt script exceeds this number then the service will go into a CRITICAL state.

`Active:` Checked



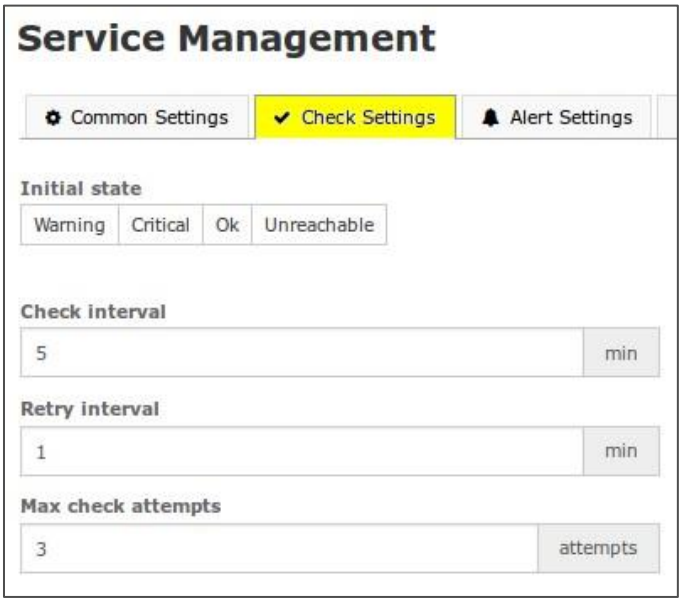
Check Settings tab

Check interval: 5

Retry interval: 1

Max check attempts: 3

9. Click the **Save** button and then click the **Apply Configuration** button at the bottom of the screen.



Final Result

Once the service has been created, navigate to **Home** > **Service Detail** and search for the service. If the check has been configured correctly, it will appear like the example below.

The Warning status shows that the **AutoIt** script is running more than 2000 milliseconds to load the test webpage.

Host	Service	Status	Duration	Attempt	Last Check	Status Information
10.25.14.52	AutoIt Timer	Warning	23s	1/3	2016-11-01 12:36:01	WARNING: Script took 4.2690 seconds to complete: time=4269ms:2000:5000:0

Finishing Up

This completes the documentation on Integrating AutoIT with Nagios XI. If you have additional questions or other support-related questions, please visit us at our Nagios Support Forum, Nagios Knowledge Base, or Nagios Library:

Visit Nagios Support Forum Visit Nagios Knowledge Base Visit Nagios Library