Tips For Customizing Configuration Wizards

ver 2010-06-22

Contents

Overview

Nagios XI comes with built in Configuration Wizards (CW) that allow you to monitor a host with little effort.

You may have a handful of services that you would like to add to a CW so you do not have to manually create them after the CW has been run. It is easy to modify an existing CW to include your customizations.

You can also create your own CW once you have understood the how a configuration wizard works.

This document is a guide of tips to help you achieve this. This document is a work in progress so some sections may not be complete or 100% correct.

Requirements

My background experience is Microsoft Windows so this guide is focused on performing all these actions from a Windows 7 PC.

I will direct you to save files to C:\NagiosXI\ as we progress through the guide.

This document is based on a running virtual machine of Nagios XI downloaded from the Nagios Enterprise website. It is recommended to be running the most up to date release version of Nagios XI.

Applications

There are three main programs I use throughout this guide:

- Notepad++ is used as the editor ○ http://notepad-plus-plus.org/
- WinSCP is used to transfer files to / from your Nagios XI server ○ http://winscp.net/eng/download.php
- Putty is used to remotely access your Nagios XI server to perform console commands ○ http://www.chiark.greenend.org.uk/~sgtatham/putty/

WinSCP and Putty

The first time you connect to the Nagios XI server it will present a warning dialog box about the Nagios XI host not being in the local cache, click Yes and you will not be prompted again.

All the steps I use with WinSCP are done using the Commander view. To enable this do the following:

 Click **Options > Preferences** ○ Click **Interface** on the left menu ○ **Select** the **Commander** radio button ○ Click **OK**

Adding A Service To An Existing Wizard

- Name: Windows Server
- File name: windowsserver.inc.php
- Location: /usr/local/nagiosxi/html/includes/configwizards/windowsserver

Goal

We'll modify the existing Windows Server wizard. I want to add disk read and write services for my Windows hosts. I do not want to use warning or critical values or receive notifications; all I want is the statistical data that will generate pretty graphs.

But first let's back it up

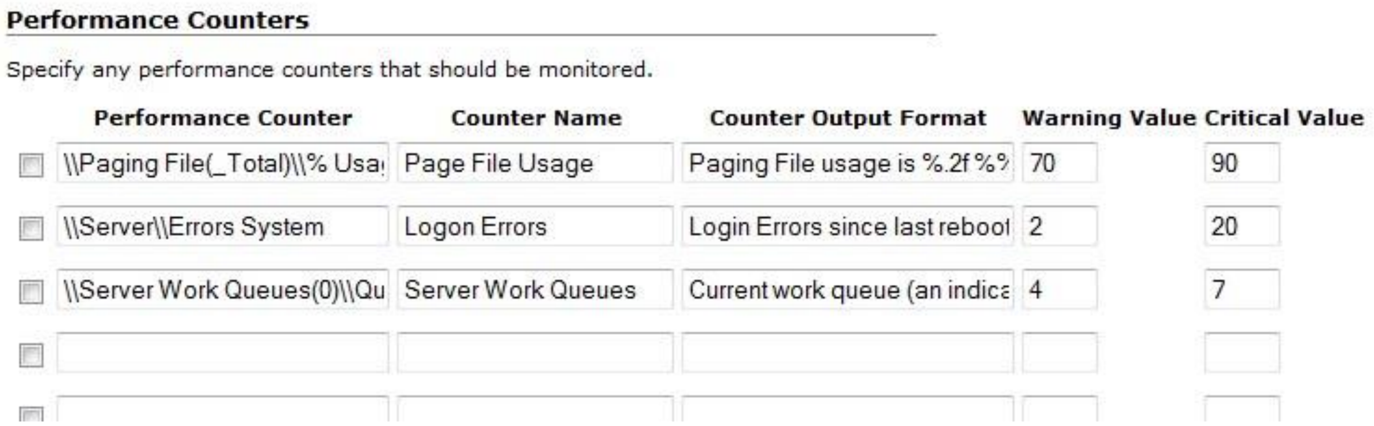
Backup Original CW + Create Copy

Please create the directory **C:\NagiosXI\Backup**.

- Open **WinSCP**
- □ Hostname: IP address of your Nagios XI server
- □ Username: **root**
- □ Password: Your root Password (nagiosxi by default)
- **Click** the **Login** button
- **Click** in the **right** pane
- **Click** the **Open directory/bookmark** icon *[Remote > Goto > Open directory/bookmark]* ○ In the remote directory type in **/usr/local/nagiosxi/html/includes/configwizards/windowsserver** ○ Click **OK**
- In the left pane **browse** to **C:\NagiosXI\Backup**
- **Drag** the **windowsserver.inc.php** file from the **right** pane to the **left** pane
- **Click** the **Copy** button
- *Wait for the file to copy*
- In the left pane **double** click on **.. Parent directory**
- **Drag** the **windowsserver.inc.php** file from the **right** pane to the **left** pane
- **Click** the **Copy** button
- *Wait for the file to copy*
- **Close** WinSCP

Windows Server CW

Figure 1 is a screenshot of the Windows Server CW.



Performance Counters

Specify any performance counters that should be monitored.

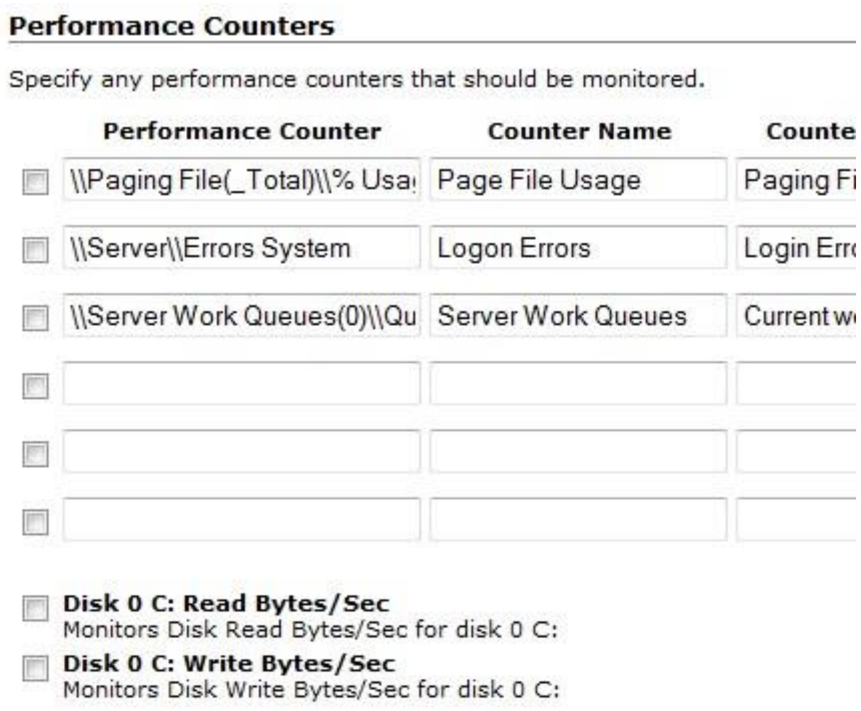
	Performance Counter	Counter Name	Counter Output Format	Warning Value	Critical Value
□	\\Paging File(_Total)\\% Usa	Page File Usage	Paging File usage is %.2f %%	70	90
□	\\Server\\Errors System	Logon Errors	Login Errors since last reboot	2	20
□	\\Server Work Queues(0)\\Qu	Server Work Queues	Current work queue (an indica	4	7
□					
□					

Figure 1

We will add our disk read and write services to this section of the CW.

Note: in Figure 1 the fields are displayed and can be altered.

We don’t require the user to change the fields for disk read/write so we’ll just not show them. Here is an example:



Performance Counters

Specify any performance counters that should be monitored.

	Performance Counter	Counter Name	Counte
□	\\Paging File(_Total)\\% Usa	Page File Usage	Paging Fi
□	\\Server\\Errors System	Logon Errors	Login Err
□	\\Server Work Queues(0)\\Qu	Server Work Queues	Current w
□			
□			
□			

□ **Disk 0 C: Read Bytes/Sec**
Monitors Disk Read Bytes/Sec for disk 0 C:

□ **Disk 0 C: Write Bytes/Sec**
Monitors Disk Write Bytes/Sec for disk 0 C:

Figure 2

Disk Read Write Services

We are using Windows Performance Counters (WPC) to obtain the data that will be turned into performance graphs.

There is a separate section in this guide that explains the use of WPC.

- \PhysicalDisk(0 C:)\Disk Read Bytes/sec
- \PhysicalDisk(0 C:)\Disk Write Bytes/sec

Note: for every backslash (\) we need to use an escape character (\) so that the backslash is treated as text instead of being an escape character. This means we will end up with:

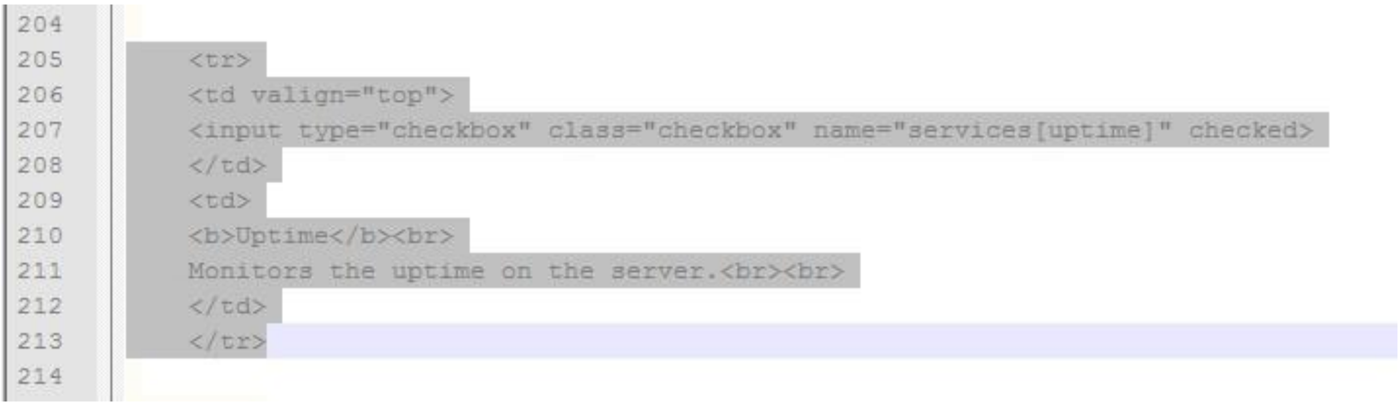
- \\PhysicalDisk(0 C:)\\Disk Read Bytes/sec
- \\PhysicalDisk(0 C:)\\Disk Write Bytes/sec

Add Read Service To CW Stage2

Open the file C:\NagiosXI\windowsserver.inc.php in *Notepad++*

We are going to copy and paste an existing service

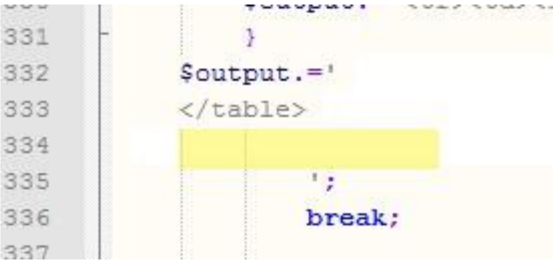
- Press **Ctrl** + **F** ○ Type **[uptime]** and click **Find Next** ○ Click **Close**
- Highlight the entire uptime service *(<tr> to </tr>)*
- **Right** click your **selection** and select **Copy**



```
    <tr>
    <td valign="top">
    <input type="checkbox" class="checkbox" name="services[uptime]" checked>
    </td>
    <td>
    <b>Uptime</b><br>
    Monitors the uptime on the server.<br><br>
    </td>
    </tr>
```

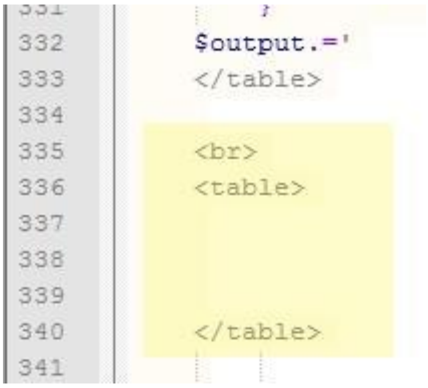
Figure 3

- Press **Ctrl** + **F** ○ Type **Performance Counter** and click **Find Next** ○ *You will be taken to the Performance Counter section of Stage 2*
- Click **Close**
- After the end of the performance counters section the table ends *[see Figure 4]*
- We are going to add a line break and then create a new table for our disk read service *[see* Figure 5*]*



```
        }
    $output.='
    </table>

            ';
            break;
```

Figure 4

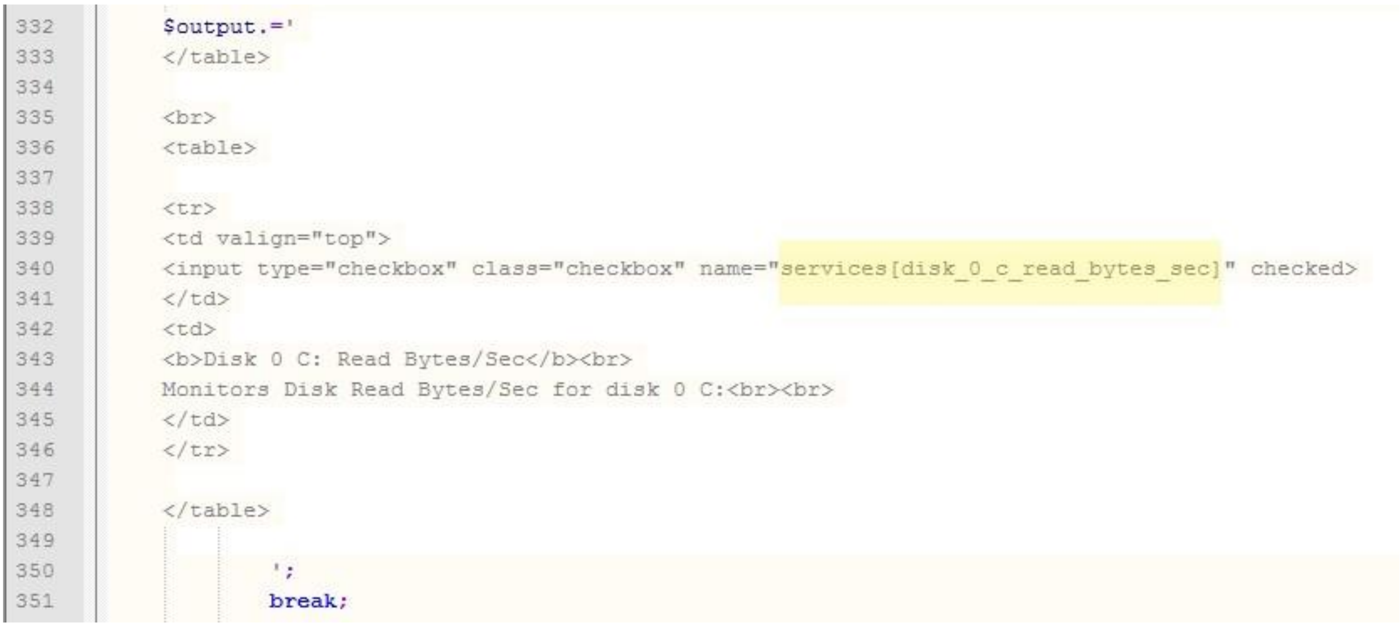


```
        }
    $output.='
    </table>

    <br>
    <table>



    </table>
```

Figure 5

- Now paste the contents of your clipboard in between the table and make some changes Change: ○ <input type="checkbox" class="checkbox" name="services[**uptime**]" checked> To: ○ <input type="checkbox" class="checkbox" name="services[**disk_0_c_read_bytes_sec**]" checked> Change: ○ <b>**Uptime**</b>
  To: ○ <b>**Disk 0 C: Read Bytes/Sec**</b><br> Change: ○ Monitors the uptime on the server.<br>
  To: ○ **Monitors Disk Read Bytes/Sec for disk 0 C:**<br>

- This should leave you with this:



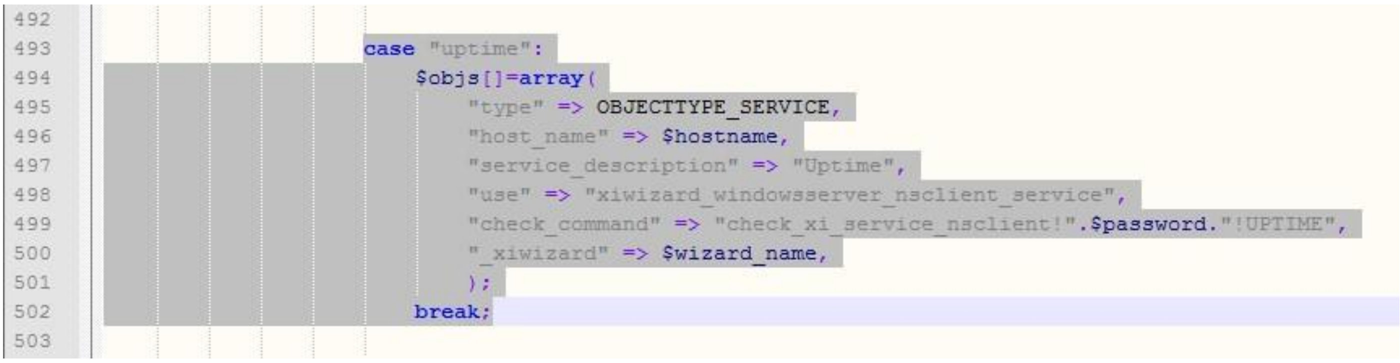
```
        $output.='
        </table>

        <br>
        <table>

        <tr>
        <td valign="top">
        <input type="checkbox" class="checkbox" name="services[disk_0_c_read_bytes_sec]" checked>
        </td>
        <td>
        <b>Disk 0 C: Read Bytes/Sec</b><br>
        Monitors Disk Read Bytes/Sec for disk 0 C:<br><br>
        </td>
        </tr>

        </table>

                ';
                break;
```

Figure 6

- On Figure 6 you can see that I highlighted **services[disk_0_c_read_bytes_sec]**
- This is the name that this service is called by the configuration wizard □ We will link this up in the end of the CW

- *Press **Crtl** + **S** to save our changes so far*

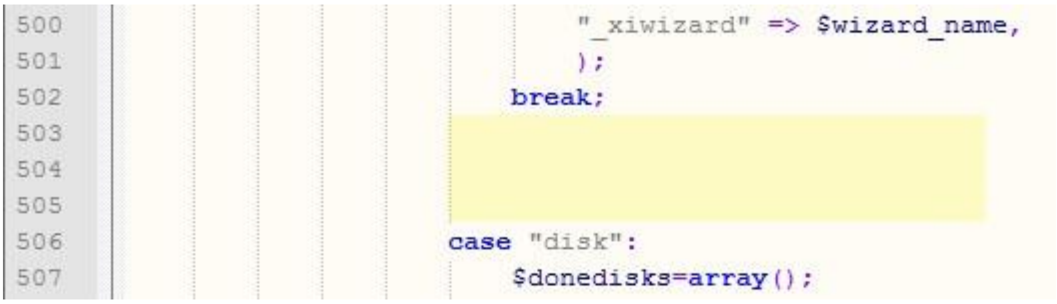
- Press **Ctrl** + **F** ○ Type **case "uptime":** and click **Find Next** ○ *You will be taken to the service creation section for the uptime service*
- Click **Close**
- Highlight the entire uptime service *(case to break;) Figure 7*
- **Right** click your **selection** and select **Copy**



```
                    case "uptime":
                        $objs[]=array(
                            "type" => OBJECTTYPE_SERVICE,
                            "host_name" => $hostname,
                            "service_description" => "Uptime",
                            "use" => "xiwizard_windowsserver_nsclient_service",
                            "check_command" => "check_xi_service_nsclient!".$password."!UPTIME",
                            "_xiwizard" => $wizard_name,
                            );
                        break;

```

Figure 7

□ **Paste** this **between** the **uptime** service and the **disk** service **Error! Reference source not found.**



```
                            "_xiwizard" => $wizard_name,
                            );
                        break;



                    case "disk":
                        $donedisks=array();
```

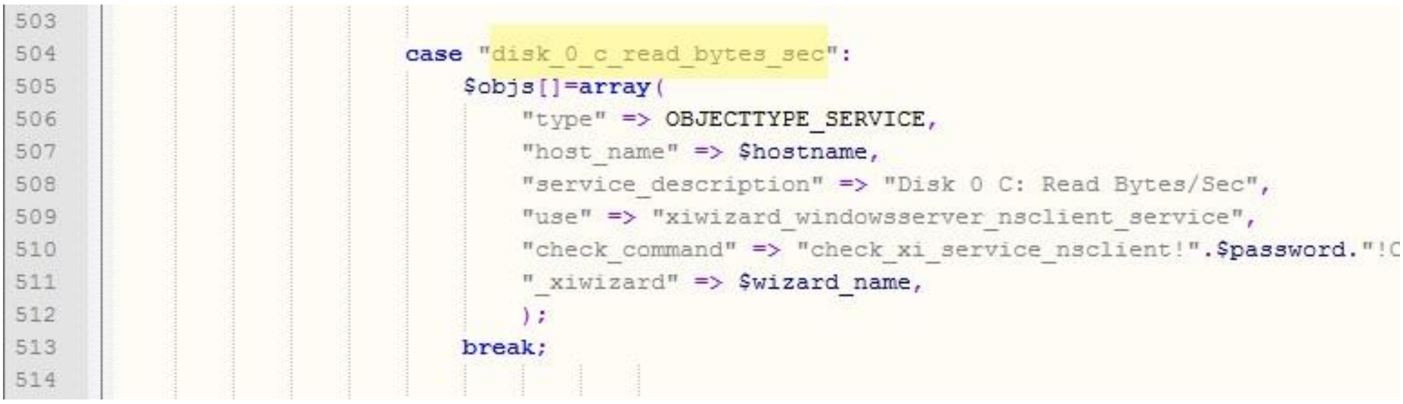
Figure 8

- Change: ○ case "**uptime**" To: ○ case "**disk_0_c_read_bytes_sec**":
- Change:

  ○ "service_description" => "**Uptime**", To: ○ "service_description" => "**Disk 0 C: Read Bytes/Sec**", Change: ○ "check_command" => "check_xi_service_nsclient!".$password."**!UPTIME**", To:

  ○ "check_command" => "check_xi_service_nsclient!".$password."**!COUNTER! -l "\\PhysicalDisk(0 C:)\\Disk Read Bytes/sec","Disk 0 C: Read Bytes/Sec is %.f""**,

- You should now have a screen that looks like Figure 9:



```
                case "disk_0_c_read_bytes_sec":
                    $objs[]=array(
                        "type" => OBJECTTYPE_SERVICE,
                        "host_name" => $hostname,
                        "service_description" => "Disk 0 C: Read Bytes/Sec",
                        "use" => "xiwizard_windowsserver_nsclient_service",
                        "check_command" => "check_xi_service_nsclient!".$password."!C
                        "_xiwizard" => $wizard_name,
                        );
                    break;
```

Figure 9

- You can see that I've highlighted **disk_0_c_read_bytes_sec**
- This is how we link Stage 2 of the CW with the case section that does the service creation ○ In Figure 6, line 380 ends with
  - **" checked>** ○ This defines that the check box will be checked when the wizard runs ○ If you want the user to make this choice, remove the word **checked**
  - **">** ○ When the checkbox **disk_0_c_read_bytes_sec** is checked, this tells the CW to process the case section later in the script.

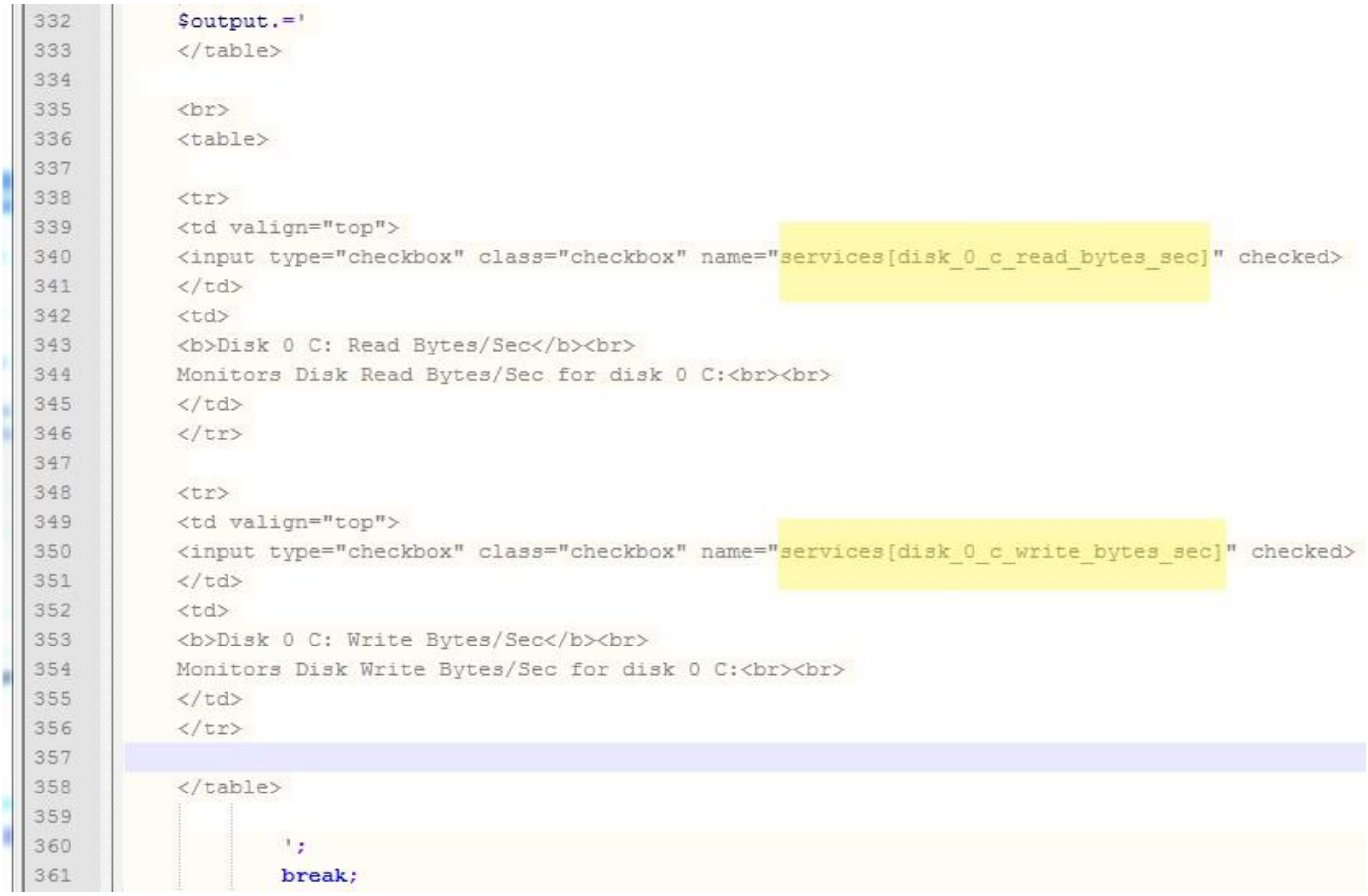
  ○ The case section is the bit that actually defines the services that are going to be created

- *Press **Crtl** + **S** to save our changes so far*

Add Write Service To CW Stage2

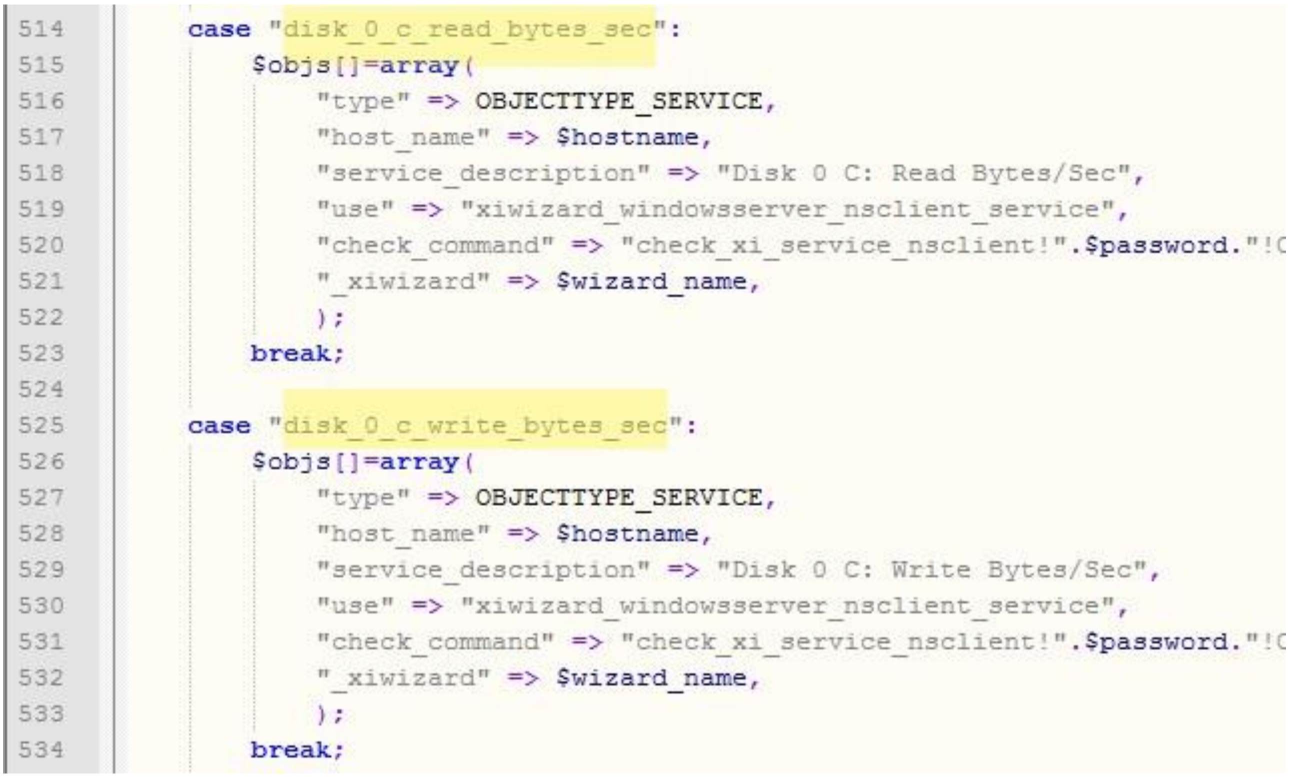
You can follow the same steps in Add Read Service To CW Stage2, simply replace the work Read with Write throughout the steps.

You will end up with something similar to Figure 10 and Figure 11.



```
    $output.='
    </table>

    <br>
    <table>

    <tr>
    <td valign="top">
    <input type="checkbox" class="checkbox" name="services[disk_0_c_read_bytes_sec]" checked>
    </td>
    <td>
    <b>Disk 0 C: Read Bytes/Sec</b><br>
    Monitors Disk Read Bytes/Sec for disk 0 C:<br><br>
    </td>
    </tr>

    <tr>
    <td valign="top">
    <input type="checkbox" class="checkbox" name="services[disk_0_c_write_bytes_sec]" checked>
    </td>
    <td>
    <b>Disk 0 C: Write Bytes/Sec</b><br>
    Monitors Disk Write Bytes/Sec for disk 0 C:<br><br>
    </td>
    </tr>

    </table>

            ';
            break;
```

Figure 10



```
    case "disk_0_c_read_bytes_sec":
        $objs[]=array(
            "type" => OBJECTTYPE_SERVICE,
            "host_name" => $hostname,
            "service_description" => "Disk 0 C: Read Bytes/Sec",
            "use" => "xiwizard_windowsserver_nsclient_service",
            "check_command" => "check_xi_service_nsclient!".$password."!C
            "_xiwizard" => $wizard_name,
            );
        break;

    case "disk_0_c_write_bytes_sec":
        $objs[]=array(
            "type" => OBJECTTYPE_SERVICE,
            "host_name" => $hostname,
            "service_description" => "Disk 0 C: Write Bytes/Sec",
            "use" => "xiwizard_windowsserver_nsclient_service",
            "check_command" => "check_xi_service_nsclient!".$password."!C
            "_xiwizard" => $wizard_name,
            );
        break;
```

Figure 11

Upload The Custom CW

- Open **WinSCP**
- Hostname: IP address of your Nagios XI server

- Username: **root**
- Password: Your root Password (nagiosxi by default)
- **Click** the **Login** button
- **Click** in the **right** pane
- **Click** the **Open directory/bookmark** icon *[Remote > Goto > Open directory/bookmark]* ○ In the remote directory type in **/usr/local/nagiosxi/html/includes/configwizards/windowsserver** ○ Click **OK**
- In the left pane **browse** to **C:\NagiosXI\**
- **Drag** the **windowsserver.inc.php** file from the **left** pane to the **right** pane
- **Click** the **Copy** button
- *Wait for the file to copy*
- **Close** WinSCP

Test The Custom CW

- Login to Nagios XI and run your Custom CW
- You will now have the disk read and write checks available for selection
- Finish the wizard and you'll see your new services you've created for the host you ran the CW against

End Of Chapter

This completes this section on adding a service to an existing configuration wizard.

Windows Performance Counters (WPC)

We can use WPC as a way of obtaining point in time data from a Windows host. NSClient++ is the agent on the Windows host that Nagios XI communicates through.

Source Information

A performance object is any resource, application, or service that can be measured. You can select **performance objects**, **counters** and **instances** to collect data about the performance of system components or installed software.

Identifying WPC

How do you know what WPC are available on your Windows host? We use Performance Monitor to browse the currently available counters. I am looking for a *disk read bytes* counter.

- Log onto your Windows host
- Start > Run > **perfmon** > **OK** ○ **Click** the **+** icon *[Add]*
    - ○ We are looking for disk counters, in the list you'll find **PhysicalDisk**

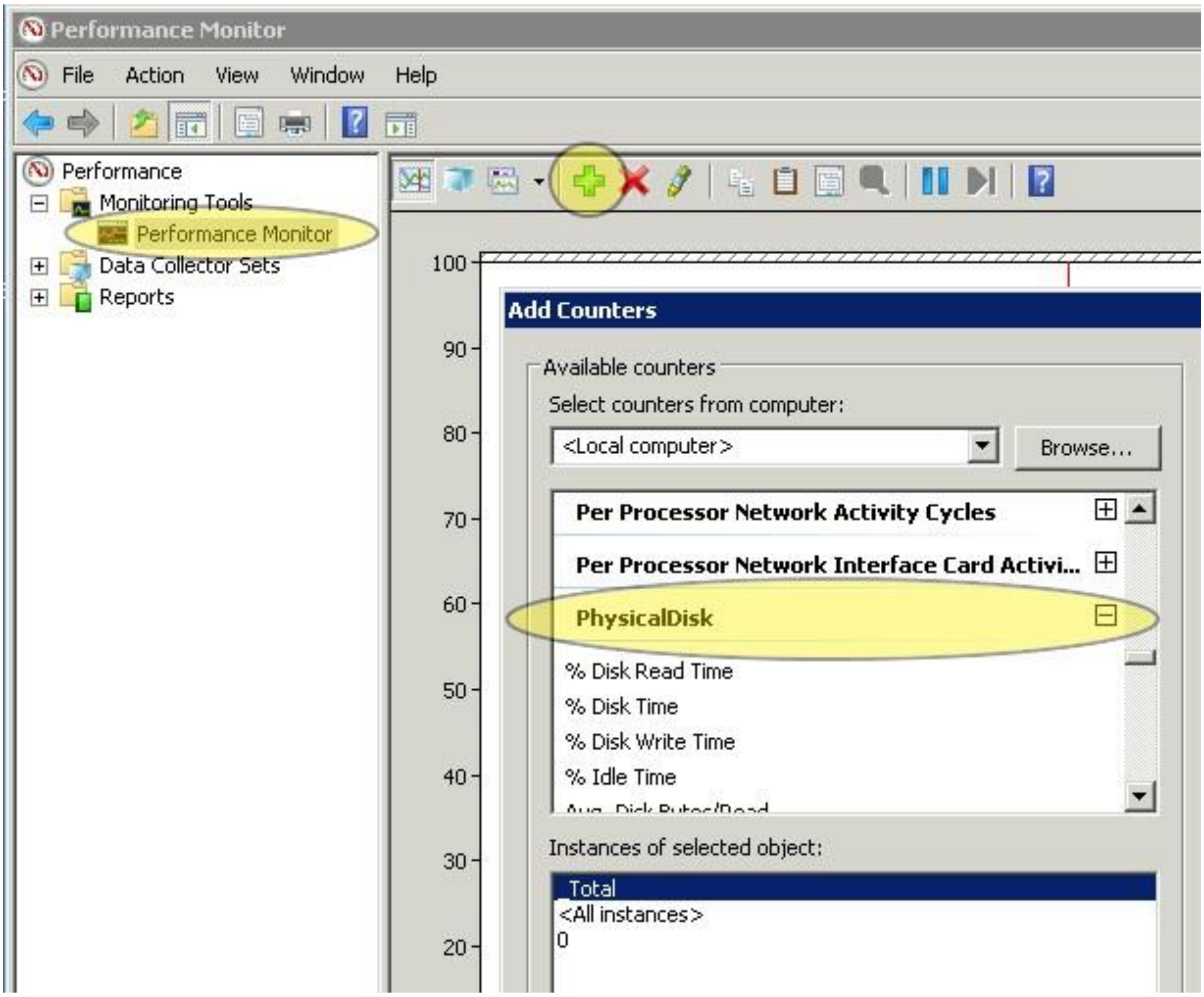
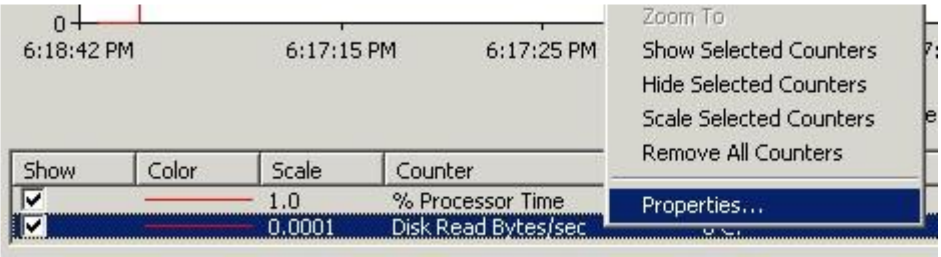


Figure 12

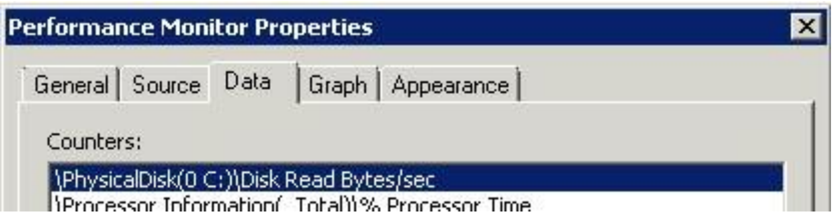
○ Select **Disk Read Bytes/sec**
○ Select **0 C:** ○ **Click** the **Add**
>> button ○ Click **OK**

- **Right** click **Disk Read Bytes/sec** and select **Properties**



Show	Color	Scale	Counter
☑		1.0	% Processor Time
☑		0.0001	Disk Read Bytes/sec

Figure 13



\PhysicalDisk(0 C:)\Disk Read Bytes/sec
\Processor Information(_Total)\% Processor Time

Figure 14

○ You can see here the counter is expressed as:

  ▪ \PhysicalDisk(0 C:)\Disk Read Bytes/sec

- ○ Click **Cancel**

Extra Backslashes

When we use WPC in Nagios XI we need to add extra characters. For every backslash (\) we need to use an escape character (\) so that the backslash is treated as plain text. This means we will end up with:

- ▪ **\\PhysicalDisk(0 C:)\\Disk Read Bytes/sec**

I need a link here to some documentation that explains this correctly.

Understanding How WPC Are Used In Nagios XI

There are several components that make your service checks possible. I will attempt to explain how these components work together in Nagios XI. It is assumed that you will already have used NSClient++.

Nagios XI Service Relationship Map

Please refer to the "Nagios XI Service Relationship Map" diagram that shows a graphical representation of how these components work together. *This is a separate PDF file not included with this guide.*

The relationship map and the topics below should hopefully give you a basic understanding of how WPC are used in Nagios XI.

Plugins

- A plugin is a script that is responsible for communicating with devices to gather the information you require
- A plugin could make use of SNMP ○ Gather information from a switch or router using SNMP ○ Requires no agent ○ SNMP is widely used in enterprise hardware / software
- Some plugins use an agent to perform the checks ○ NSClient++ is an example of an agent

  ○ The agent is installed on a windows host ○ Nagios XI talks to the agent and issues the request ○ The agent then responds back with the results of the request ○ **check_nt** is a plugin that is included with NSClient++ ○ We use **check_nt** for our WPC checks

  - ▪ *More information about check_nt can be found here*

Command Line / Commands

- Every check you perform on your Nagios XI server is issued from the **command line** The plugins used on the command line vary for each type of check Here is an example of the check_nt command for WPC:

  ```
  check_nt -H hostname -s password -p 12489-v COUNTER -l "\\<performance
  object>\\counter","<description>"
  ```

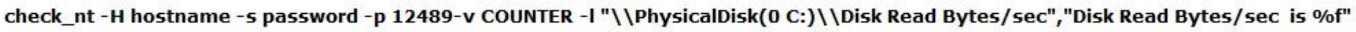
- Now lets translate that command into \PhysicalDisk(0 C:)\Disk Read Bytes/sec

  ```
  check_nt -H hostname -s password -p 12489-v COUNTER -l "\\PhysicalDisk(0 C:)\\Disk Read Bytes/sec","Disk
  Read Bytes/sec  is %f"
  ```

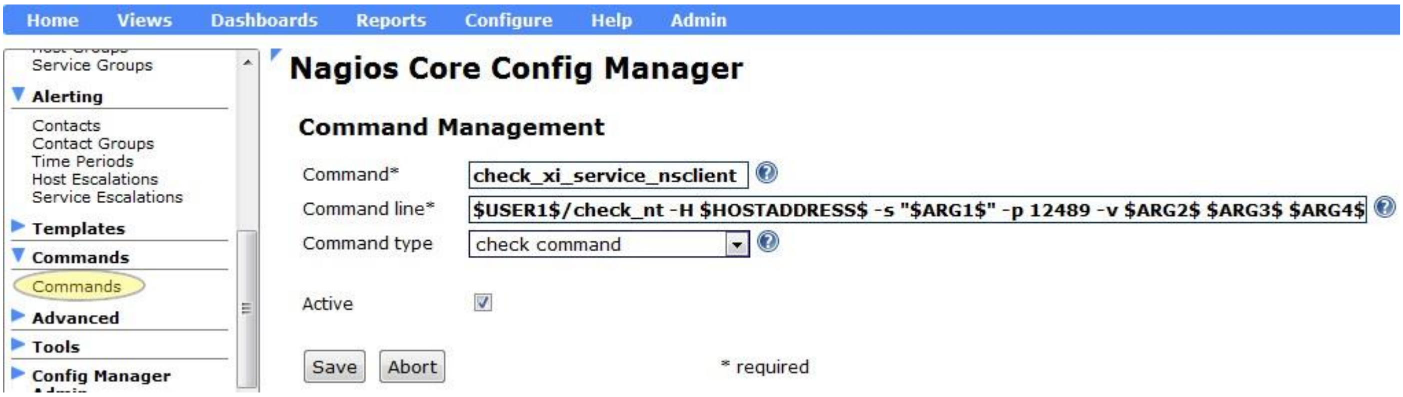
- You can see that I added extra backslashes to the WPC as this is what is required by NSClient++
- I also added the description field that will return the result in the format I require, %f is the returning value from the check_nt command

Commands In Nagios XI

 Here is an example of how Nagios XI uses the check_nt plugin: ○ Figure 15 is the check_xi_service_nsclient command ○ Figure 16 is from an existing service using a WPC



```
check_nt -H hostname -s password -p 12489-v COUNTER -l "\\PhysicalDisk(0 C:)\\Disk Read Bytes/sec","Disk Read Bytes/sec  is %f"
```



Nagios Core Config Manager

Command Management

Command* check_xi_service_nsclient

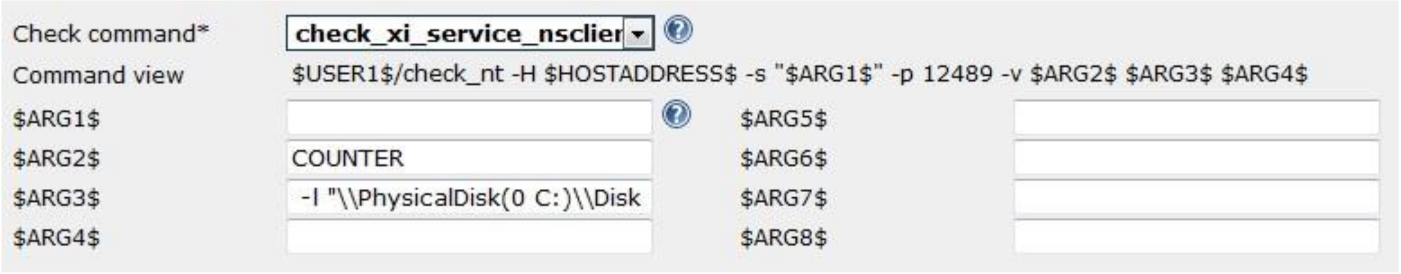
Command line* $USER1$/check_nt -H $HOSTADDRESS$ -s "$ARG1$" -p 12489 -v $ARG2$ $ARG3$ $ARG4$

Command type check command

Active ☑

* required

Figure 15



Check command* check_xi_service_nsclier

Command view $USER1$/check_nt -H $HOSTADDRESS$ -s "$ARG1$" -p 12489 -v $ARG2$ $ARG3$ $ARG4$

$ARG1$		$ARG5$	
$ARG2$	COUNTER	$ARG6$	
$ARG3$	-l "\\PhysicalDisk(0 C:)\\Disk	$ARG7$	
$ARG4$		$ARG8$	

Figure 16

- The service is using the check command check_xi_service_nsclient
- The check command check_xi_service_nsclient is using the check_nt plugin
- The $ARG1$ $ARG2$ ... fields in the service is how we pass the information we require through to the check_nt command
- $ARG1$ is not populated as I was not using a password to connect to this host

You can add your own commands to Nagios XI

- Upload your plugin via the **Manage Plugins** page
- Using Core Configuration Manager go to the **Commands** section Add your own command that runs your uploaded plugin

End Of Chapter

This completes this section on WPC.

Troubleshooting

White Screen

Quite often when there is a mistake in your custom CW code, you will get a blank screen when you try to access the Configure menu as seen in Figure 17.

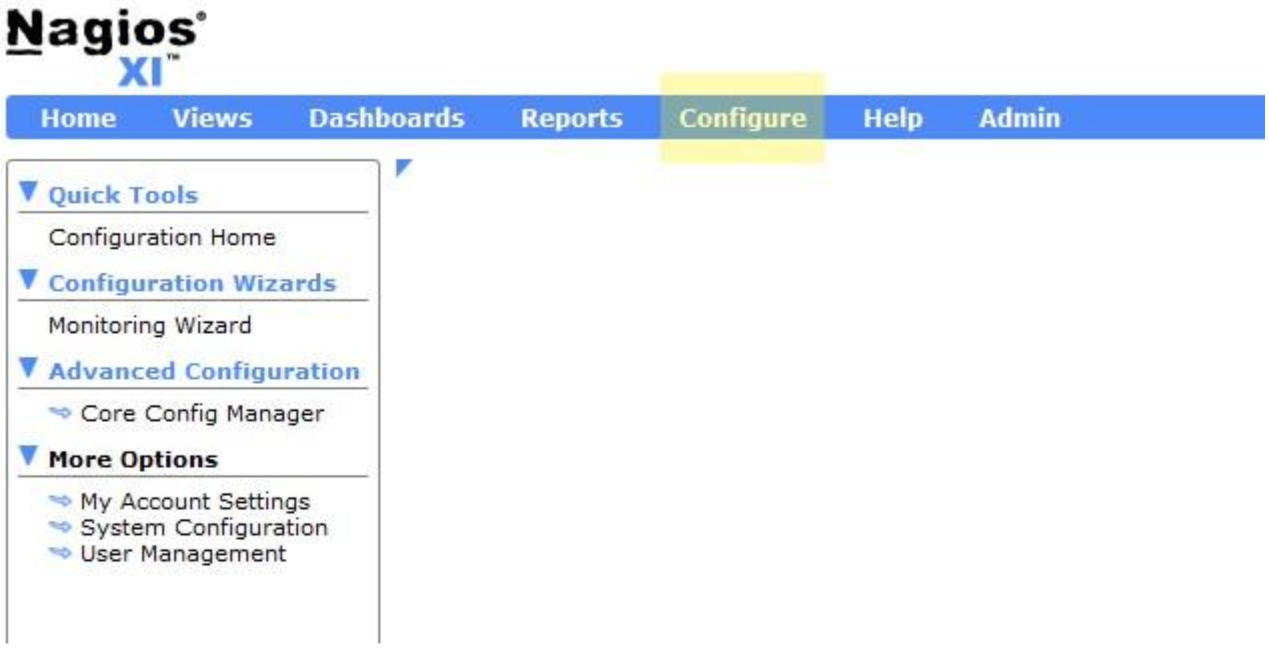
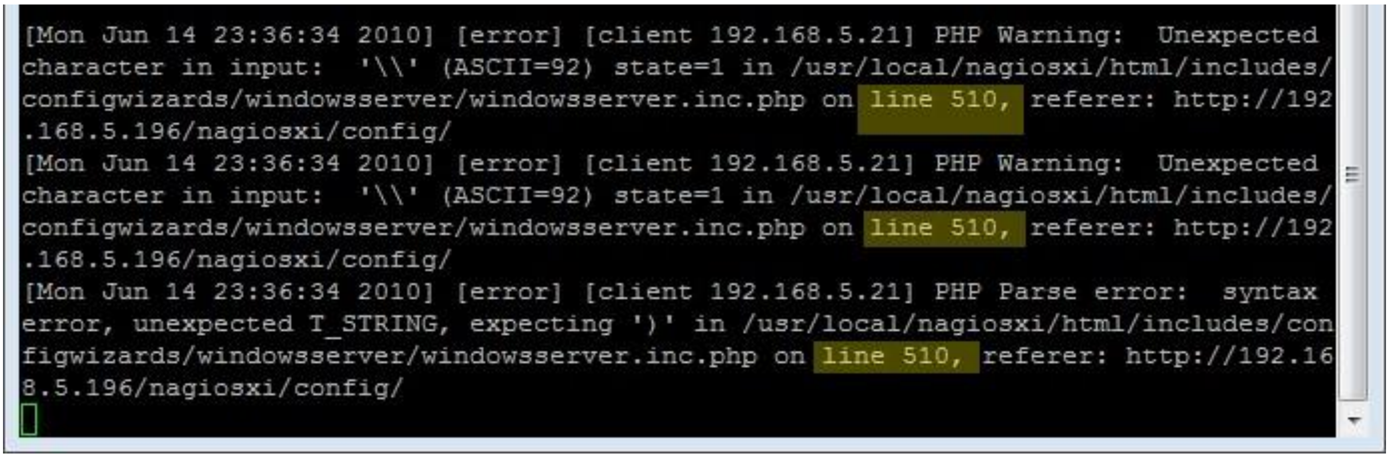


Figure 17

One way to diagnose is to watch the web server log file in real time.

- Access the Console of your Nagios XI server or use Putty to connect to it remotely
- Username: **root**
- Password: Your root Password (nagiosxi by default)
- Type **tail -f /var/log/httpd/error_log** and press **Enter**
- Initially you will be shown the most recent errors
- Press **Enter** a **couple** of times to move the text up the screen
- Now go back to your Nagios XI session and click **Configure** on the top blue menu
- Your Putty session should have updated with some reference to the error, this should help you to determine where in code your mistake lies



```
[Mon Jun 14 23:36:34 2010] [error] [client 192.168.5.21] PHP Warning:  Unexpected
character in input:  '\\' (ASCII=92) state=1 in /usr/local/nagiosxi/html/includes/
configwizards/windowsserver/windowsserver.inc.php on line 510, referer: http://192
.168.5.196/nagiosxi/config/
[Mon Jun 14 23:36:34 2010] [error] [client 192.168.5.21] PHP Warning:  Unexpected
character in input:  '\\' (ASCII=92) state=1 in /usr/local/nagiosxi/html/includes/
configwizards/windowsserver/windowsserver.inc.php on line 510, referer: http://192
.168.5.196/nagiosxi/config/
[Mon Jun 14 23:36:34 2010] [error] [client 192.168.5.21] PHP Parse error:  syntax
error, unexpected T_STRING, expecting ')' in /usr/local/nagiosxi/html/includes/con
figwizards/windowsserver/windowsserver.inc.php on line 510, referer: http://192.16
8.5.196/nagiosxi/config/
```

Figure 18

You can see in Figure 18 that it is pointing out an error on line 510.