

Integrating the Bisccheck Plugin Extension with Nagios XI

Purpose

This document describes how to install and configure the Bisccheck plugin extension for use with Nagios XI to dynamically monitor Linux machines with time-based thresholds. The Bisccheck extension is an application that sends passive checks to a Nagios server, specifically relating to dynamic and adaptive thresholds that change over time.

Bisccheck Plugin Extension Overview

Bisccheck executes checks on a schedule, using a mixture of Nagios plugins and various other polling methods to gather information. It caches the results and uses this cache to perform calculations on historical data. This allows Bisccheck to follow trends and (in future releases) predict future metrics. What this means for a user is that you now have the ability to have one check in Nagios report different data throughout the day depending on previous results, as opposed to current status.

Bisccheck is designed to follow trends, not static values, and it compares the current status of a check to the recent historical data from the check. Because of this design, any check which returns a Warning or Critical status may eventually change back to an OK status without any external action. If you do not set up Nagios XI to email when a warning or critical status occurs, you may not benefit from the power of Bisccheck.

This documentation will configure Bisccheck to monitor the Nagios XI localhost root partition growth over time.

Using this example will teach you how you can use Bisccheck for your needs.

Editing Files

In many 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**

Prerequisites

Bisccheck will run on RHEL/CentOS 6/7. Bisccheck requires Java version 6+, NSCA and REDIS to be configured on your Nagios XI server in order to function. Establish a terminal session to your Nagios XI server as the root user and execute the following commands to install the prerequisites:

Integrating the Bisccheck Plugin Extension with Nagios XI

RHEL | CentOS | Oracle Linux

```
yum install -y java redis
```

Debian | Ubuntu

```
sudo apt install default-jre redis-server
```

You will need to configure NSCA in Nagios XI, this guide will be using the Decryption Method set to XOR and the password (this guide is using `Str0ngP@sswr0d`). The following documentation explains how to configure NSCA:

[How To Use The NSCA Addon](#)

Note: By default the `/etc/xinetd.d/nsca` has the `only_from` line configured with `127.0.0.1`, this is all that is required.

Bisccheck Installation

In your terminal session execute the following commands to install Bisccheck:

```
mkdir -p /opt/socbox/addons
cd /tmp
wget https://github.com/thenodon/bisccheck/releases/download/bisccheck_rel-1.1.2/bisccheck-1.1.2.tar.gz tar xzf bisccheck-1.1.2.tar.gz cd bisccheck-1.1.2
chmod +x install ./install
```

Note: If any problems occur, you can run the following command to view the command-line options supported by the installer:

```
./install -u
```

It has been identified that on RHEL 8.x the installer fails as it does not correctly detect the distribution.

This can be rectified by adding the word **red** to line 224 of the **install** file:

```
red|rh|rhel|redhatenterpriseserver|centos|debian|ubuntu)
```

This command will make that change for you:

```
sed -i 's/rh|/red|rh|/g' install
```

Integrating the Bisccheck Plugin Extension with Nagios XI

Create check_root_used Check

In the configuration step below you will define a check to be used for the Bisccheck service. This part of the documentation creates that check and tests that it works. In a terminal session execute the following command to open vi to the new file `/usr/local/nagios/libexec/check_root_used`:

```
vi /usr/local/nagios/libexec/check_root_used
```

Paste the following code into the `check_root_used` file:

```
#!/bin/bash USED=`df -Ph | grep "/"$ | awk '{print $5;}' | sed 's/\%//'\`  
echo "used=$USED" exit 0
```

Save the file as you do not need to make any additional changes. You will need to set proper permissions on the file in order for it to run. Execute the following command:

```
chmod +x /usr/local/nagios/libexec/check_root_used
```

Now run the command to test it works:

```
/usr/local/nagios/libexec/check_root_used
```

Your output will look something like this:

```
used=34
```

This is the percentage of your root partition which is used. Bisccheck will grab the 34 and use it to determine whether or not the disk is filling up too fast (this is explained later in this document).

Configuring The Bisccheck Plugin Extension

Three main files make up the configuration of Bisccheck and are located by default in

```
/opt/socbox/addons/bisccheck/etc/:
```

- `bisccheck.xml`
 - holds scheduling and check command configuration
- `24thresholds.xml`
 - describes time profiles and thresholds used to evaluate measured data

Integrating the Bisccheck Plugin Extension with Nagios XI

- `servers.xml`
 - parameters for connecting to the Nagios server

You will need to configure Bisccheck with the password you entered when setting up NSCA within Nagios XI.

Edit the file `/opt/socbox/addons/bisccheck/etc/servers.xml` and locate the following lines:

```
<class>NSCAServerNOSEND</class>
<key>password</key>
<value>nscapassword</value>
```

Change `NSCAServerNOSEND` to `NSCAServer` (remove the `NOSEND` portion). Change `nscapassword` in the value field to the password you entered when setting up NSCA. Here are those changes:

```
<class>NSCAServer</class>
<key>password</key>
<value>Str0ngP@sswr0d</value>
```

Save the file, this completes the changes required.

Define `bisccheck.xml` Configuration File

The next step is to configure the `bisccheck.xml` file. The default file provided with the installation is going to be purged, you will define the contents of this file using the code on this page. Execute the following commands:

```
cd /opt/socbox/addons/bisccheck/etc
echo "" > bisccheck.xml
vi bisccheck.xml
```

Paste the following code into the `bisccheck.xml` file (lines in bold will be discussed below):

```
<?xml version='1.0' encoding='UTF-8'?>
<b>bisccheck</b>
  <host>
    <name>localhost</name>
    <desc>My localhost</desc>
```

Integrating the Bisccheck Plugin Extension with Nagios XI

```
<service>
  <name>root</name>
  <desc>Monitor the root space used</desc>
  <schedule>30S</schedule>
  <url>shell://localhost</url>
  <serviceitem>
    <name>used</name>
    <desc>Percent of root used</desc>
    <execstatement>
      {"check": "/usr/local/nagios/libexec/check_root_used", "label": "used"}
    </execstatement>
    <thresholdclass>Twenty4HourThreshold</thresholdclass>
    <serviceitemclass>CheckCommandServiceItem</serviceitemclass>
  </serviceitem>
</service>
</host>
</bisccheck>
```

The indentation may be lost when you copy and paste, but this is not important. Save the file as you do not need to make any additional changes, the lines in bold are explained as follows.

- `<name>localhost</name>`
 - This is the name of the host which must have a matching host entry in Nagios. In our example we are running Bisccheck and Nagios XI on the same host, on a default install of Nagios XI localhost is an existing host.
 - If you were running this on a remote host named mailserv1, you would have to create a host in Nagios XI with the name mailserv1.
- `<name>root</name>`
 - This is the name of the service you are checking with Bisccheck. This needs a matching service entry in Nagios XI and that service must be attached to the host above (localhost). Creating this service is discussed later in this document.
 - **Note:** Don't confuse this root service with the Root Partition service that is part of the Nagios XI default installation.
- `<schedule>30S</schedule>`

Integrating the Bisccheck Plugin Extension with Nagios XI

- This is how often the check will be performed. You can enter either a specific interval such as 10M, or in this case 30S, or you can use a cron-style expression such as "0 15 10 ? * MON-FRI" to run every week day at 10:15AM.
- **Note:** When using intervals you must use capital letters for the Second, Minute, and Hour specifier.
- `<name>used</name>`
 - This line is just a subname of the service you are checking. It is arbitrary and does not need to be configured in Nagios XI however it is mentioned here as it is reference in the next setting.
- `{"check":"/usr/local/nagios/libexec/check_root_used", "label":"used"}`
 - This is the check that will be run to gather the required metrics. In our case it is running `/usr/local/nagios/libexec/check_root_used`, this is the script you created earlier on.
 - The check is looking for the label used.

Note: For the name, the following characters are valid: a-z, A-Z, 0-9, dash (-), period (.), and underscore (_). The **service** and **serviceitem** entries have all the above valid characters, plus the at sign (@) and spaces.

Define 24thresholds.xml File

The next step is to configure the `24thresholds.xml` file. The default file provided with the installation is going to be purged, you will define the contents of this file using the code on this page. Execute the following commands:

```
cd /opt/socbox/addons/bisccheck/etc echo "" > 24thresholds.xml
vi 24thresholds.xml
```

Paste the following code into the `24thresholds.xml` file (lines in bold will be discussed below):

```
<?xml version='1.0' encoding='UTF-8'?>
<twenty4threshold xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

Integrating the Bisccheck Plugin Extension with Nagios XI

```
<servicedef>
  <hostname>localhost</hostname>
  <servicename>root</servicename>
  <serviceitemname>used</serviceitemname>
  <period>
    <calcmethod>&lt;</calcmethod>
    <warning>0</warning>
    <critical>0</critical>
    <hoursIDREF>1</hoursIDREF>
  </period>
</servicedef>

<hours hoursID="1">
  <hourinterval>
    <from>00:00</from>
    <to>23:00</to>
    <threshold>avg(localhost-root-used[0:3])+5</threshold>
  </hourinterval>
</hours>
</twenty4threshold>
```

The indentation may be lost when you copy and paste, but this is not important. Save the file as you do not need to make any additional changes, the lines in bold are explained as follows.

- The `<hostname>`, `<servicename>`, and `<serviceitemname>` fields
 - These need to match what was entered in the `bisccheck.xml` file's `<host><name>`, `<service><name>`, and `<serviceitem><name>` fields
 - `<calcmethod>`
 - This defines how Bisccheck will test the measured metrics.
 - You want to verify that the percent of your root partition that is used, is less than a certain threshold.
 - In this case, `<` is being used for the less than character (`<`) because it will break the XML file.
- `<warning>` and `<critical>`

Integrating the Bisccheck Plugin Extension with Nagios XI

- These fields are percentage variances from the threshold, they are not used in this example.
- <hoursIDREF>
 - Refers to a time period that the check will be performed, and this refers to the
- <hours hoursID="1"> field
- <hours hoursID="1">
 - Within this section there is a <from> and a <to> field which are used to define start and end times for running the check, they must be supplied in 24-hour time.
- <threshold>
 - In this field a formula is used to average out the last 4 checks (starting at 0 and going to 3) and the it adds 5. This value is what the current value must be less than in order to return OK

How To Start Bisccheck

At this point things should be ready to go. Execute the following commands to begin the necessary services:

RHEL | CentOS | Oracle Linux

```
systemctl start redis
systemctl start bisccheckd
```

Debian | Ubuntu

```
systemctl start redis-server
systemctl start bisccheckd
```

The Bisccheck installation should start making checks immediately.

Execute the following command to confirm:

```
tail -f /var/tmp/bisccheck.log
```


Integrating the Bisccheck Plugin Extension with Nagios XI

You should see output similar to the following:

```
2017-02-09 05:05:33.489 ; INFO ; pool-3-thread-1 ;
com.ingby.socbox.bisccheck.servers.NSCAWorker ; NSCA-1:localhost:root:OK used
= 28 (33 < 33 < W < 33 < C < ) | used=28;33;33;0; threshold=33;0;0;0;
avgexec-time=5ms
```

This is just some informational output showing what the check is returning, and the values being compared. In our example, Bisccheck is checking used root partition space every 30 seconds, averaging the last 4 results, and alerting as CRITICAL if the latest used space percentage has risen 5 or more in the last 30 seconds. The values used in this example were chosen somewhat randomly, but a 5% increase in used space over a 30-second period should be an indication of a problem in just about any system.

Don't be concerned with warnings like the following, they exist because the check hasn't executed enough times to gather data to calculate.

```
2017-02-09 05:02:33.729 ; WARN ; DefaultQuartzScheduler_Worker-1 ;
com.ingby.socbox.bisccheck.jepext.ExecuteJEP ; Math jep parse error for
expression - avg(null)+5 : Error during evaluation: Invalid parameter type
```

Execute these commands to ensure these services start on boot:

RHEL | CentOS | Oracle Linux

```
systemctl enable redis
systemctl enable bisccheckd
```

Debian | Ubuntu

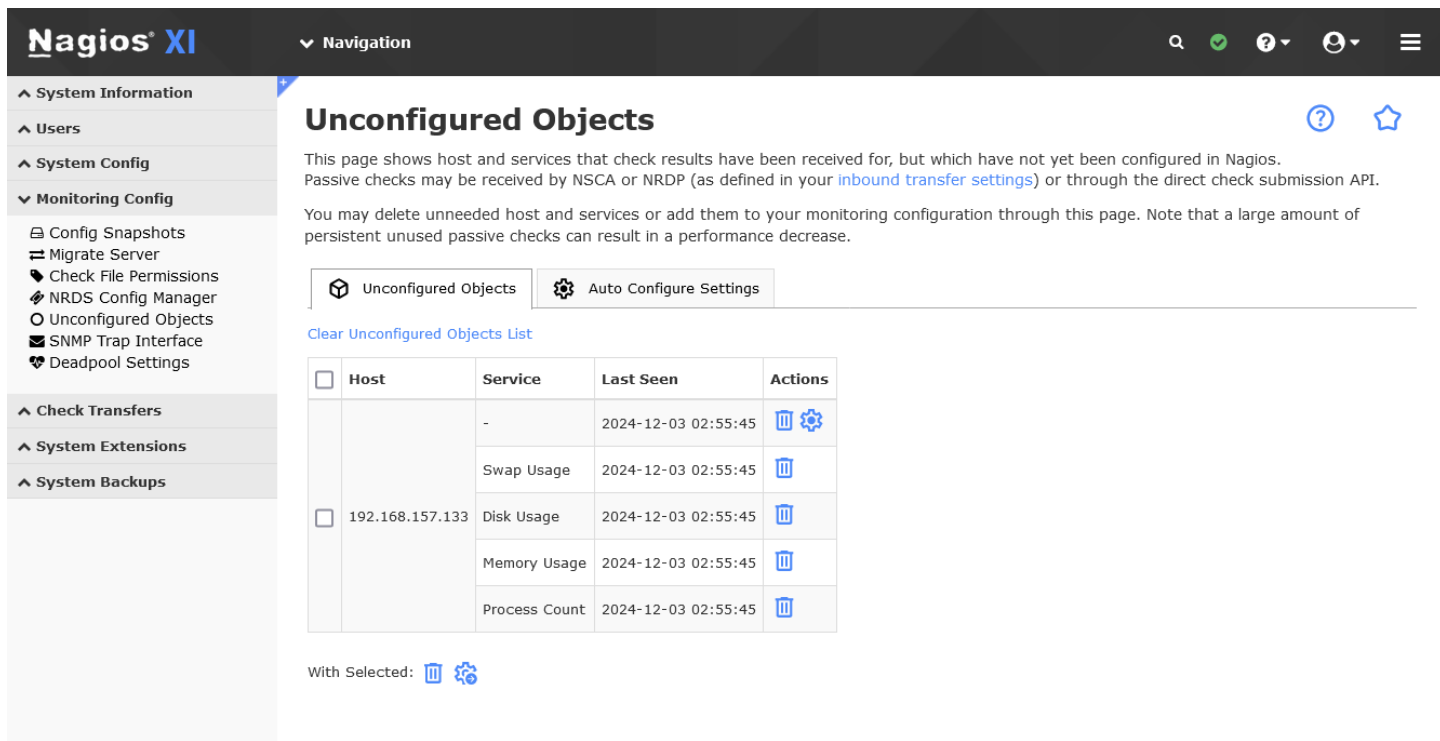
```
systemctl enable redis-server
systemctl enable bisccheckd
```

Configuring Bisccheck In Nagios XI

Once Bisccheck is running, it will start sending passive checks to the Nagios XI server. These will show up as **Unconfigured Objects** in Nagios XI until you set up a service for your local host with the name you specified in `bisccheck.xml`.

In the Nagios XI web interface, navigate to **Admin > Monitoring Config > Unconfigured Objects**.

Integrating the Bisccheck Plugin Extension with Nagios XI



The screenshot shows the Nagios XI interface. The left sidebar contains navigation menus for System Information, Users, System Config, Monitoring Config, Check Transfers, System Extensions, and System Backups. The main content area is titled "Unconfigured Objects" and includes a description of passive checks and a table of unconfigured objects. The table has columns for Host, Service, Last Seen, and Actions. The "Actions" column contains a trash icon and a blue gear icon. Below the table, there is a "With Selected:" label and icons for trash and settings.

Unconfigured Objects

This page shows host and services that check results have been received for, but which have not yet been configured in Nagios. Passive checks may be received by NSCA or NRDP (as defined in your [inbound transfer settings](#)) or through the direct check submission API. You may delete unneeded host and services or add them to your monitoring configuration through this page. Note that a large amount of persistent unused passive checks can result in a performance decrease.

Unconfigured Objects | Auto Configure Settings

[Clear Unconfigured Objects List](#)

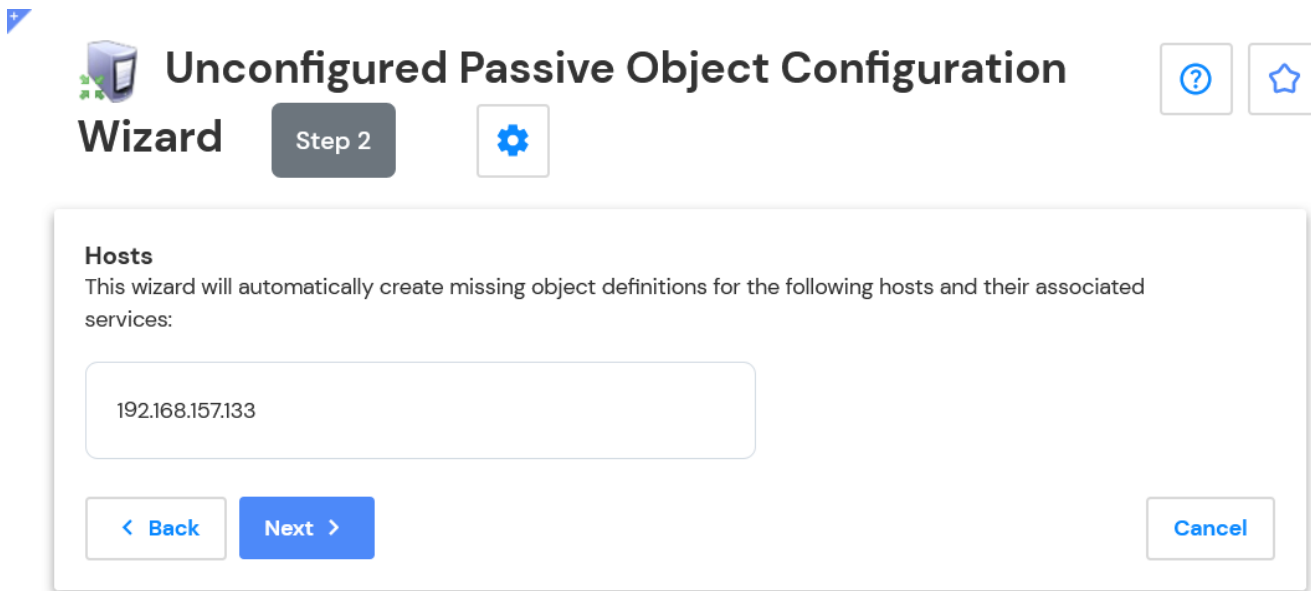
<input type="checkbox"/>	Host	Service	Last Seen	Actions
		-	2024-12-03 02:55:45	
		Swap Usage	2024-12-03 02:55:45	
<input type="checkbox"/>	192.168.157.133	Disk Usage	2024-12-03 02:55:45	
		Memory Usage	2024-12-03 02:55:45	
		Process Count	2024-12-03 02:55:45	

With Selected:

These are results from passive checks that do not have an associated service check in Nagios XI. To turn these check results into a service, click the **blue gear** under **Actions**.

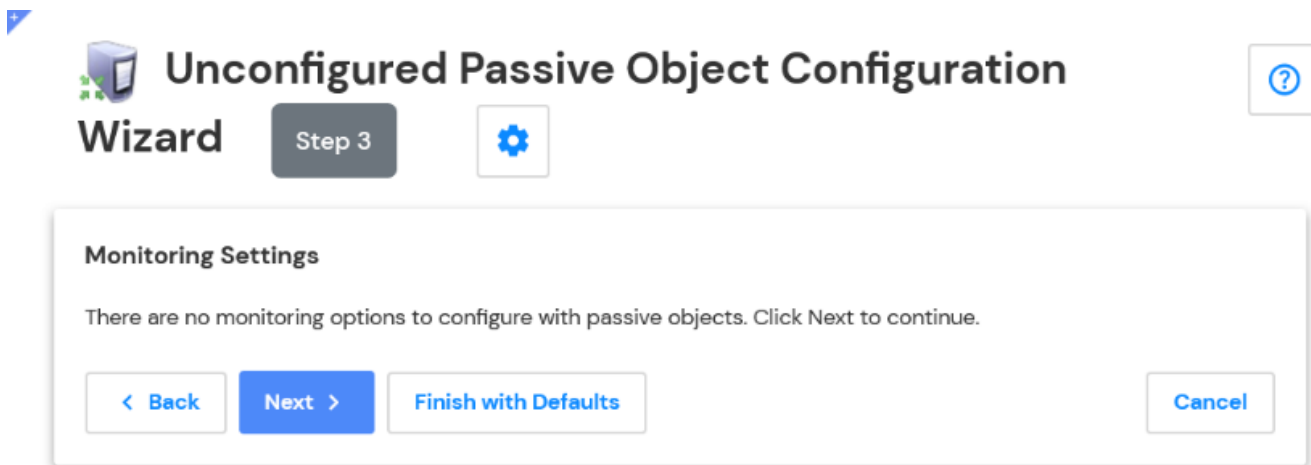
Integrating the Bischeck Plugin Extension with Nagios XI

When the Unconfigured Passive Object wizard starts, you will be presented with a screen like the one to the right.



The screenshot shows the 'Unconfigured Passive Object Configuration Wizard' at Step 2. The title bar includes a question mark icon and a star icon. Below the title, there is a 'Step 2' indicator and a gear icon. The main content area is titled 'Hosts' and contains the text: 'This wizard will automatically create missing object definitions for the following hosts and their associated services:'. Below this text is a text input field containing the IP address '192.168.157.133'. At the bottom of the wizard, there are three buttons: '< Back', 'Next >', and 'Cancel'.

Click **Next**.



The screenshot shows the 'Unconfigured Passive Object Configuration Wizard' at Step 3. The title bar includes a question mark icon. Below the title, there is a 'Step 3' indicator and a gear icon. The main content area is titled 'Monitoring Settings' and contains the text: 'There are no monitoring options to configure with passive objects. Click Next to continue.'. At the bottom of the wizard, there are four buttons: '< Back', 'Next >', 'Finish with Defaults', and 'Cancel'.

On the next step, there are no monitoring options, just click **Next**.

Integrating the Bisccheck Plugin Extension with Nagios XI

Run through the wizard selecting your preferred settings such as notifications, groups, parents. Finish the wizard and you will now have a service set up in Nagios for your Bisccheck results. When the configuration is successfully applied, click the **View status details for localhost** link.

That's it, you now have Bisccheck up and running.

Finishing Up

This completes the documentation on integrating the bisccheck plugin extension 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](#)