NAGIOS MONITORING COOKBOOK

Level up your IT Infrastructure Monitoring

GABRIEL CANEPA

SYSTEM CODE GEEKS
SYSADMINS RESOURCE CENTER

Nagios Monitoring Handbook

Contents

Preface

Nagios, now known as Nagios Core, is a free and open source computer-software application that monitors systems, networks and infrastructure. Nagios offers monitoring and alerting services for servers, switches, applications and services. It alerts users when things go wrong and alerts them a second time when the problem has been resolved.

Nagios was originally designed to run under Linux, but it also runs well on other Unix variants. It is free software licensed under the terms of the GNU General Public License version 2 as published by the Free Software Foundation. (Source: https://en.wikipedia.org/-wiki/Nagios)

In this ebook, we provide a compilation of Nagios tutorials that will help you set up your own monitoring infrastructure. We cover a wide range of topics, from installation and configuration, to plugins and NRPE. With our straightforward tutorials, you will be able to get your own projects up and running in minimum time.

About the Author

Gabriel Canepa is a Linux Foundation Certified System Administrator (LFCS-1500-0576-0100) and web developer from Villa Mercedes, San Luis, Argentina.

He works for a worldwide leading consumer product company and takes great pleasure in using FOSS tools to increase productivity in all areas of his daily work.

When he's not typing commands or writing code or articles, he enjoys telling bedtime stories with his wife to his two little daughters and playing with them, the great pleasure of his life.

Chapter 1

Core Installation and Configuration on Ubuntu Server

Whether you are an engineer in charge of monitoring a large and complex network infrastructure, a system administrator of a relatively small number of machines, or a just a regular user who needs to check on the availability of a couple of machines and important services in your home network, it is critical to understand the importance of and implement a monitoring solution.

As such, Nagios is the answer for you. In addition to providing monitoring of hardware resources (processor load, disk usage, etc) and the availability of network services (HTTP, FTP, SMTP, SSH, etc), this open source tool also offers alerting (via SMS or email through the use of plugins) in the wake of undesired events, and allows you to identify potential problems before they occur.

On top of it, through its web interface Nagios also provides access to availability data at a quick glance to share with the leaders of your organization. This same feature can also help you plan in advance for necessary upgrades to your infrastructure.

In this tutorial we will show you how to install and use Nagios Core in an Ubuntu 14.04 server. We will then demonstrate how to monitor the availability (whether the host is up or down) in a CentOS 7 system, and the status of the web service running therein. In a future guide, we will also add more advanced monitoring features for services on the same host.

1.1 Installing Nagios Core

Nagios Core has a free-of-cost solution that features complete infrastructure monitoring, hundreds of addons, and forum support. Other versions (Student VM, Pro, and Business) include these and other features as well, but they are all paid options. However, for our present purposes, the Free DIY (Do-It-Yourself) edition provides the functionality that we need, so we will show you how to download and install it on your Ubuntu 14.04 server.

Although Nagios can be installed from the Ubuntu repositories, the available version (3.5.1) is a bit outdated. For that reason, we will install the application using the code package from https://www.nagios.org/downloads/core-stay-informed/ (you can skip the form that asks for details of your implementation by clicking on *Skip to download* as you can see in Fig. 1.1).

Figure 1.1: Downloading the Free DIY version of Nagios Core

Step 1 - Before we download and install Nagios from the project's website, we will need to install and configure some additional dependencies. These consists of a complete LAMP stack and several development libraries that will assist us in building Nagios from source. Also, we will install an email service (`postfix`) to handle notifications and additional utilities (`mailutils`) to check those notifications on the local machine:

```
sudo aptitude update && sudo aptitude install apache2 php5-mysql mysql-server libapache2- ↩
```

```
    mod-php5 php5-mcrypt php5-gd php5-curl build-essential libgd2-xpm-dev openssl libssl-dev ↩
     xinetd apache2-utils unzip wget postfix mailutils
```

While installing `mysql-server`, you will be prompted to set a password for the MySQL root user. Make sure you choose a strong password which is easy to remember. When prompted to choose a mail server type for `postfix`, choose *Local Only*, as we will be delivering notifications to a local user. If you want to learn more about each of the above dependencies in detail, you can use `aptitude show dependency`, where you will need to replace dependency with one of the package names listed previously.

Step 2 - Once you have installed the dependencies listed above, download the source code for the latest Nagios stable release (at the time of this writing it's 4.1.1):

```
wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.1.1.tar.gz">https:// ↩
    assets.nagios.com/downloads/nagioscore/releases/nagios-4.1.1.tar.gz
```

and untar it:

```
tar xzvf nagios-4.1.1.tar.gz
```

We will later change directory into the folder where we just extracted the contents of the tarball. By now, proceed with the next step.

Step 3 - Create a user and group for Nagios-related processes to run, then add the nagios and www-data (Apache) users to the Nagios group (nagioscmd):

```
sudo useradd nagios
sudo groupadd nagioscmd
sudo usermod -aG nagioscmd nagios
sudo usermod -aG nagioscmd www-data
```

Particularly, the **nagioscmd** group will be needed to run commands via the web interface.

Step 4 - Change directory to the folder where you unpacked the Nagios source code earlier:

```
cd nagios-4.1.1
```

Find out where the mail binary is located:

```
which mail
```

Most likely, the above command will return `/usr/bin/mail`. You will use the `--with-mail` configure option followed by this absolute path below. Then do:

```
sudo ./configure --with-nagios-group=nagios --with-command-group=nagioscmd --with-mail=/usr ↩
    /bin/mail
```

You will be given the chance to take a second look at the configuration options before proceeding, as shown in Fig. 1.2:

```
Creating sample config files in sample-config/ ...


*** Configuration summary for nagios 4.1.1 08-19-2015 ***:

 General Options:
 -------------------------
        Nagios executable:  nagios
        Nagios user/group:  nagios,nagios
       Command user/group:  nagios,nagioscmd
             Event Broker:  yes
        Install ${prefix}:  /usr/local/nagios
    Install ${includedir}:  /usr/local/nagios/include/nagios
                Lock file:  ${prefix}/var/nagios.lock
   Check result directory:  ${prefix}/var/spool/checkresults
           Init directory:  /etc/init.d
  Apache conf.d directory:  /etc/httpd/conf.d
             Mail program:  /usr/bin/mail
                  Host OS:  linux-gnu
          IOBroker Method:  epoll

 Web Interface Options:
 ------------------------
                 HTML URL:  http://localhost/nagios/
                  CGI URL:  http://localhost/nagios/cgi-bin/
 Traceroute (used by WAP):


Review the options above for accuracy.  If they look okay,
type 'make all' to compile the main program and CGIs.

gacanepa@ubuntu:~/nagios-4.1.1$
```


Figure 1.2: Checking the --configure options before compiling

If everything looks correct, proceed with Step 5. Otherwise, correct the corresponding option in the configure statement above and try again.

Step 5 - To compile Nagios and install auxiliary files and extra features, run the following commands. Keep in mind that only sample configuration files will be installed, and you will still need to go through the documentation for more information on how to actually define entities (devices, hosts, services, etc) to fit your particular needs.

```
sudo make all
sudo make install # The main program, CGIs, and HTML files
sudo make install-init # The init script in /etc/init.d
sudo make install-config # Sample config files in /usr/local/nagios/etc
sudo make install-commandmode # Fix permissions on the directory for the external command file
sudo /usr/bin/install -c -m 644 sample-config/httpd.conf /etc/apache2/sites-enabled/nagios.conf # The Apache config file for the Nagios web interface
sudo make install-exfoliation # Exfoliation theme for the user interface
```

Step 6 - Create an admin user (and set password, see Fig. 1.3) to access the web interface and enable the Apache rewrite and cgi modules:

```
gacanepa@ubuntu:~/nagios-4.1.1$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
gacanepa@ubuntu:~/nagios-4.1.1$
```

Figure 1.3: Creating an user account for the web interface

```
sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
sudo a2enmod rewrite cgi && sudo service apache2 restart
```

The username / password pair that you will use to access the web interface will be stored in **/usr/local/nagios/etc/htpasswd.users**. To restrict permissions, we will change the group owner to nagioscmd and only allow read permissions for the members of that group:

```
sudo chgrp nagioscmd /usr/local/nagios/etc/htpasswd.users
sudo chmod 640 /usr/local/nagios/etc/htpasswd.users
```

Step 7 - Finally, let's add the necessary symbolic link to the sites-enabled directory, restart Apache, and start nagios. Please note that the actual file in sites-available was created in Step 5.

```
sudo ln -s /etc/apache2/sites-available/nagios.conf /etc/apache2/sites-enabled/
sudo service apache2 restart
sudo service nagios start
```

At this point, Nagios and Apache should be running. It is time to launch the web interface to check.

Step 8 - Verify that you can access the Nagios web interface at **https://ServerIP/nagios**. In our case, the ServerIP is 192.168.0.32. Use nagiosadmin as username and the password you chose in Step 6. If everything goes as expected, you should see the user interface as shown in Fig. 1.4:

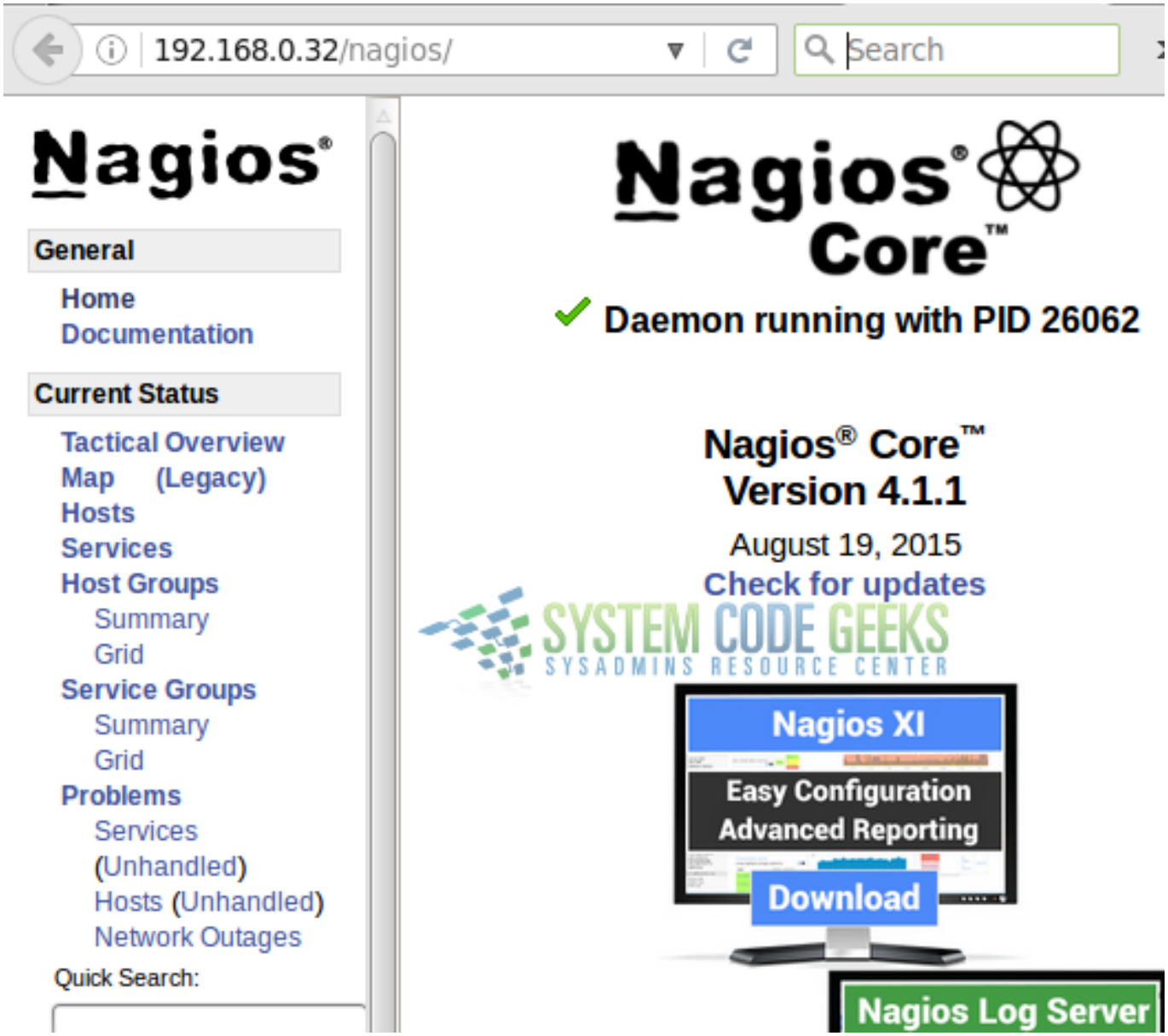

Figure 1.4: Launching the Nagios web interface

If you experience any issues or the web interface does not display correctly, check the Apache logs in /var/log/apache2. Particularly, the error.log file will point you in the right direction to troubleshoot.

Step 9 - Download and install Nagios plugins (we will dive more deeply into this topic in the next tutorial).

You can think of a Nagios Core plugin as an extension that that processes command-line arguments, performs specific checks, and then return the results to the main program. Plugins exist in the form of compiled binaries or executable scripts.

To begin, find the latest version from https://nagios-plugins.org/download/ (2.1.1 at the time of this writing) and download it:

```
wget https://nagios-plugins.org/download/nagios-plugins-2.1.1.tar.gz
tar xzvf nagios-plugins-2.1.1.tar.gz
cd nagios-plugins-2.1.1
sudo ./configure --with-nagios-user=nagios --with-nagios-group=nagios --with-openssl
sudo make
sudo make install
```

Step 10 - Finally, we need to modify the default contact that will receive alerts. Open /usr/local/nagios/etc/objects/contacts.cfg and replace the address in the email directive for an administrative account (gacanepa@ubuntu in the below example), as indicated in Fig. 1.5:

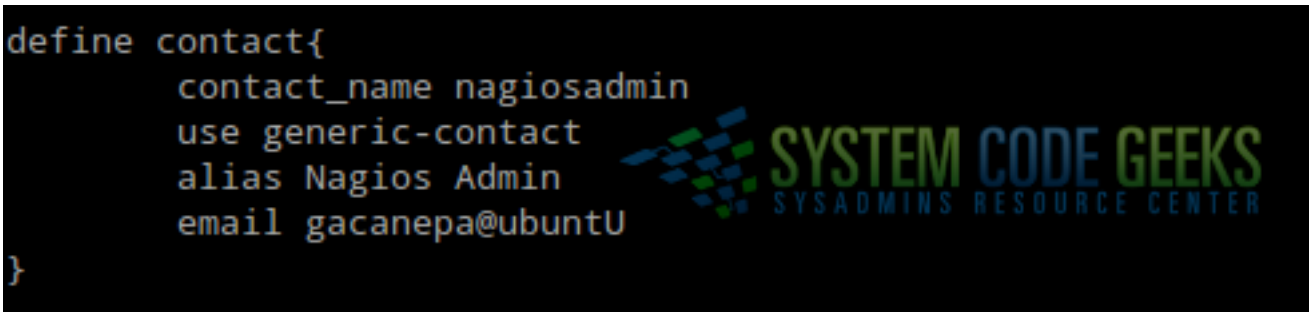

Figure 1.5: Setting the email address for the preferred contact

Once the above 10 steps have been completed successfully, we can proceed to further configure Nagios.

1.2 Configuring Nagios

To organize the hosts definitions and monitoring configuration, we will create a directory named **/usr/local/nagios/etc/servers**:

```
mkdir /usr/local/nagios/etc/servers
```

and will tell Nagios to process all configuration files (*.cfg inside this directory) by uncommenting the following line in /usr/local/nagios/etc/nagios.cfg:

```
cfg_dir=/usr/local/nagios/etc/servers
```

Inside /usr/local/nagios/etc/servers we will add a basic configuration file (centos7.cfg) to monitor the availability of the remote CentOS 7 system (IP address: 192.168.0.29) and the status of the HTTP service in that host by checking the index.html file inside the DocumentRoot directory:

```
define host {
host_name               centos7
alias                   My CentOS 7 server
address                 192.168.0.29
max_check_attempts      3
check_period            24x7
check_command           check-host-alive
contacts                nagiosadmin
notification_interval   60
notification_period     24x7
}

define service {
use                     local-service
host_name               centos7
service_description     HTTP
check_command           check_http!-I 192.168.0.29 -u /index.html
notifications_enabled   1
}
```

As per the above configuration, Nagios will attempt to reach the host 3 times before raising an alert (**max_check_attempts**). Once it detects an anomaly, it will send notifications every 60 minutes (**notification_interval**).

Before restarting Nagios, you can check the configuration file for errors (this will also check all other files invoked by nagios.cfg) as follows:

```
sudo sh -c "/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg"
```

Then let's restart Nagios:

```
sudo service nagios restart
```

and refresh the web interface. The newly added host should now show up. Refer to Fig. 1.6 for more details.

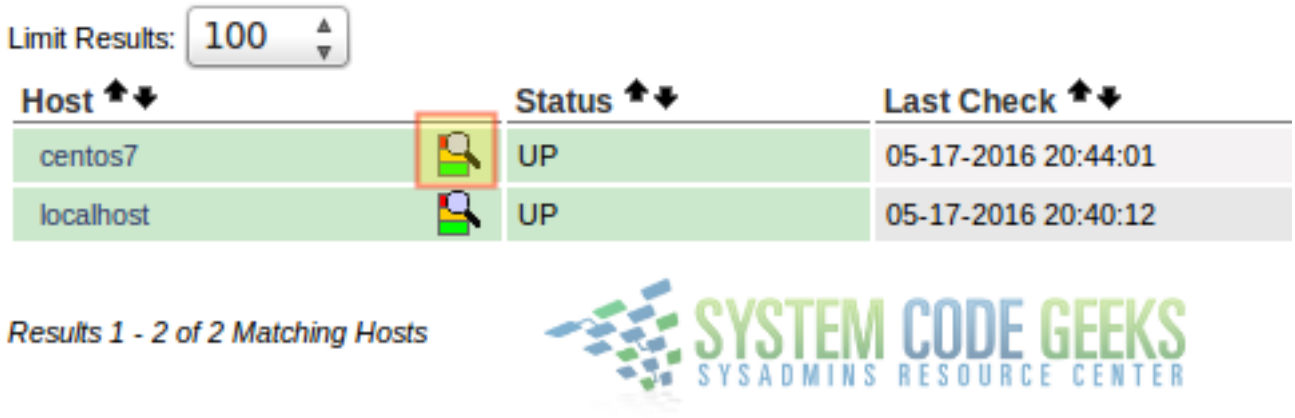

Figure 1.6: Viewing the list of monitored hosts in the Nagios Core web interface

If you click on the magnifying glass icon next to **centos7**, we will see the list of services defined for this host. We can then click on the service name to display the corresponding stats (see Fig. 1.7). Using the same interface, you can force a check of this service and perform other operations (**Service Commands**).

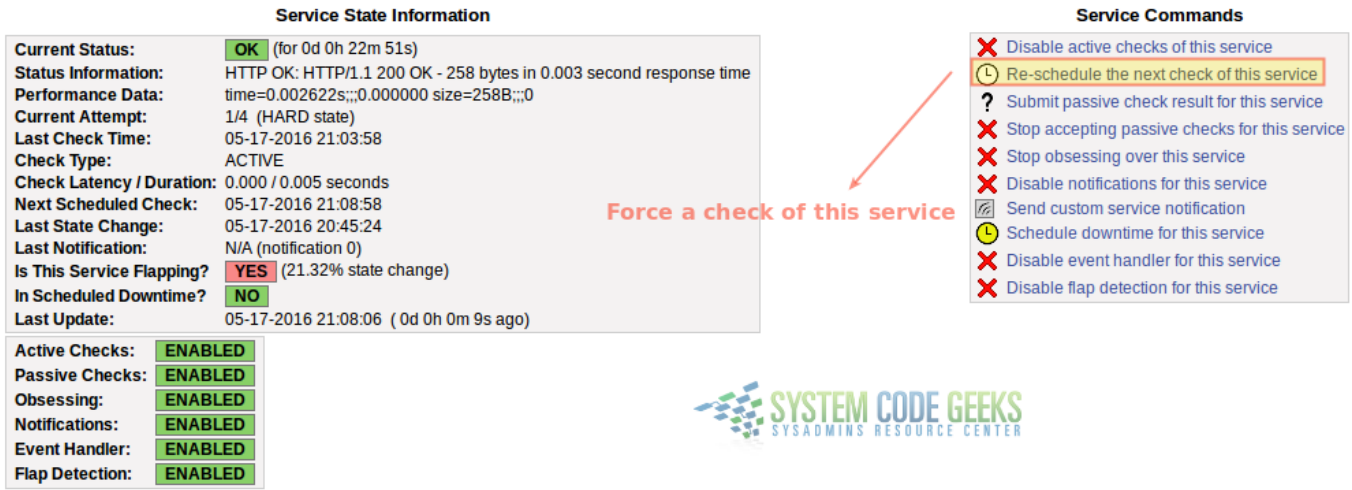

Figure 1.7: Viewing the details of a monitored service

If we now stop the service in the remote host (`systemctl stop httpd`) or shut the machine down (`poweroff`), we should receive notifications in the email account we defined in Step 10. To read your emails, you will use the `mail` command (see Fig 1.8):

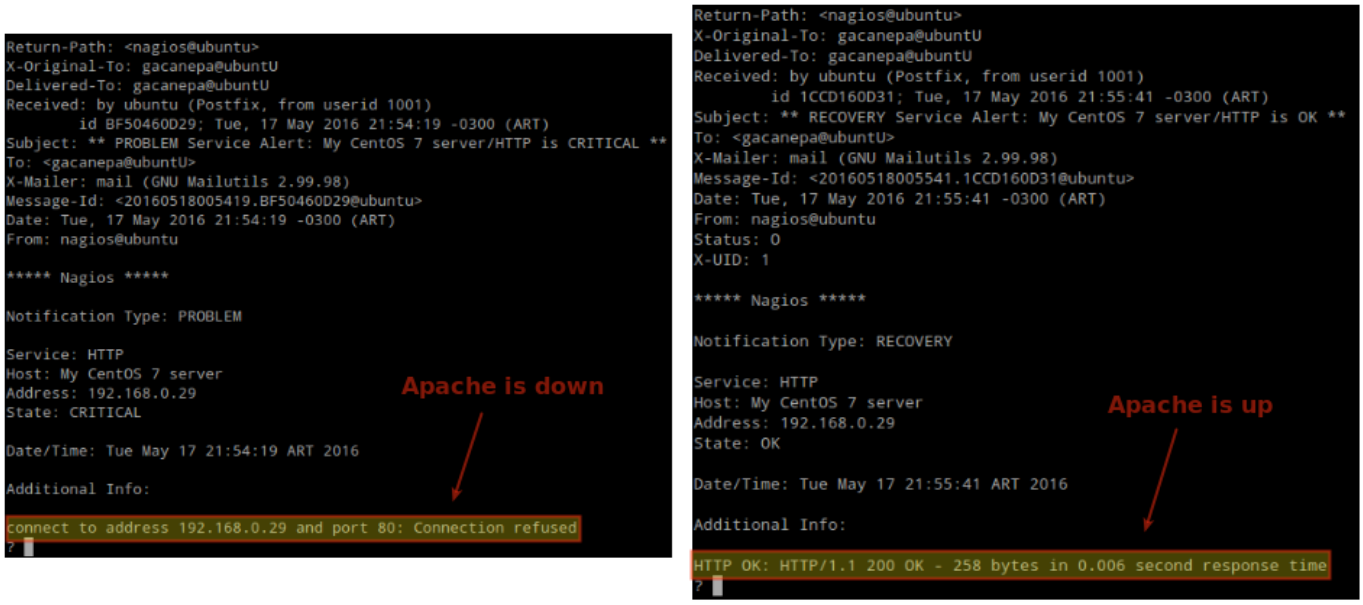

Figure 1.8: Viewing notification emails

By default, Nagios disables notifications if a host or service is found to be flapping (toggling between states). You may want to temporarily disable flapping detection by setting the `enable_flap_detection` flag to 0 in /usr/local/nagios/etc/nagios.cfg.

Last, but not least, remember to enable Nagios to start on boot:

```
sudo ln -s /etc/init.d/nagios /etc/rcS.d/S99nagios
```

Congratulations! You have successfully installed Nagios Core in your Ubuntu 14.04 server and set up monitoring for the availability of a CentOS 7 machine and the Apache service running therein.

Remember to restart Nagios each time you make changes to the associated configuration files. Otherwise, the changes will not be applied and the service will continue running with the old configuration.

1.3 Summary

In this article we have explained how to install the latest version of Nagios Core and plugins from source, and how to configure monitoring for a host and a specific service. In addition, we set up email alerts to receive notifications when the host or the chosen service experience issues. In the next guide we will dive more deeply into the use of plugins for checking a wide variety of common services.

Chapter 2

Using plugins and NRPE to check network services

In our previous article ("Nagios Core Installation and Configuration on Ubuntu Server"), we defined Nagios Core plugins as extensions (either in the form of compiled binaries or executable scripts) that process command-line arguments, perform specific checks, and then return the results to the main program. Under the hood, plugins are an abstraction layer that link the Nagios daemon and the hosts and services being monitored.

In this guide we will dive more deeply into Nagios Core plugins and what you can do with them.

Additionally, we will introduce you to NRPE (Nagios Remote Plugin Executor), an utility that will allow you to run Nagios Core plugins remotely both in Linux and Microsoft Windows machines to check metrics such as disk space usage and CPU load.

To better be able to go through this tutorial and understand the concepts and practices presented here, it is highly recommended that you have installed Nagios Core and the plugins package as explained earlier. To do so, make sure you have followed Steps 1 through 10 in the previous guide before proceeding.

2.1 A closer look at Nagios Core plugins

In the first guide, we also downloaded and installed the official Nagios Core plugins, which consists of ~50 binary files (located in `/usr/local/nagios/libexec`) that are officially developed and maintained by the project. With these plugins you can check the status of common services like SMTP, SSH, HTTP, DNS (to name a few examples), plus some other machine-specific information such as uptime and disk space usage.

In addition, there are around 3000 plugins that have been developed and are maintained by the community under the name of Nagios Exchange. Together with the official plugins, they can be used to monitor "just about everything that runs on electricity", as the Nagios project advertises.

Remember that without Nagios plugins, you can only monitor whether a host or device connected to the network is up or down. We need plugins to check on the status of services and metrics as explained in the above paragraph.

Let's take a closer look at the configuration file for the CentOS 7 system we defined previously. The service definition is of particular interest:

```
define service {
use                     local-service
host_name               centos7
service_description     HTTP
check_command           check_http!-I 192.168.0.29 -u /index.html
notifications_enabled   1
}
```

Let's examine the line beginning with `check_command`. This directive is followed by the command Nagios will use to check the status of the service in the remote host (`check_http`) and the required parameters after the exclamation sign (`-I 192.168.0.29 -u /index.html` in this case).

Fortunately, plugins are highly customizable in that you can pass parameters to indicate exactly **what** you need to monitor and **where**. For example, what happens if the remote HTTP service is listening on a port other than the default 80? What if you need to check whether the certificate in a SSL enabled server will still be valid after a given number of days? The `check_http` plugin allows to perform these checks and many more.

Let's find out more about the usage of the plugin:

```
cd /usr/local/nagios/libexec
./check_http --usage
```

A more detailed help guide is available at https://nagios-plugins.org/doc/man/index.html or via the command line (`./check_plugin --help`, where plugin can be dns, ftp, ssh, load, ups, or any of the plugins inside /usr/local/nagios/libexec).

The help guide indicates that you can use the -p option (followed by the port number) to indicate that the HTTP server is running on a different port. Likewise, the -S option tells Nagios to connect via SSL. To point out the version you can optionally use a number between 1 and 3 (1 = TLSv1, 2 = SSLv2, and 3 = SSLv3).

Let's change the port where Apache is listening on the CentOS 7 host to 8080. This will require the following change in the service definition:

```
check_command           check_http!-I 192.168.0.29 -u /index.html -p 8080
```

You can refer to the Apache HTTP server tutorial if you feel you could use a little help to change the default port or to set up a SSL enabled server. In addition, make sure that connections from the Nagios server to your remote host / new port are not blocked by an active firewall rule.

The first 2 notifications in Fig. 2.1 shows the results of the checks **BEFORE** and **AFTER** port 8080 was enabled in the remote host's firewall:

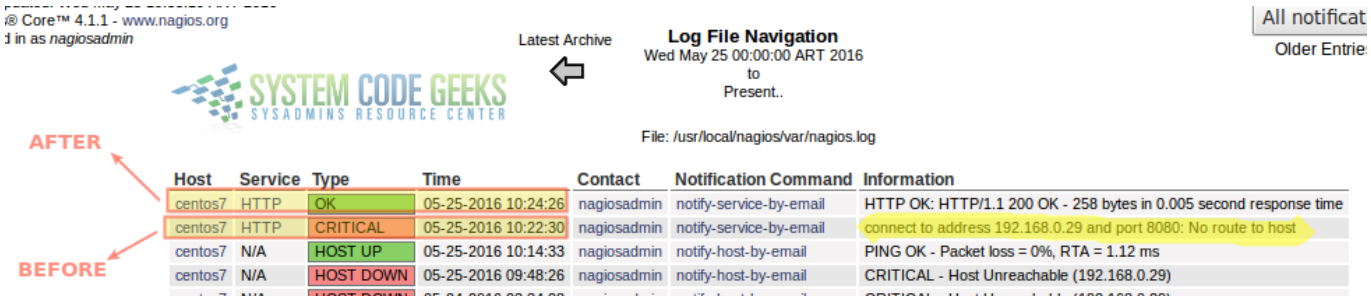

Figure 2.1: Viewing Nagios notifications BEFORE and AFTER enabling traffic through the remote port in the firewall

If you later enable SSL on the HTTP server, modify the `check_command` directive in the service definition as follows:

```
check_command           check_http!-I 192.168.0.29 -u /index.html -S 1
```

Fig. 2.2 shows what you can expect in the Apache logs in the remote host under this scenario:

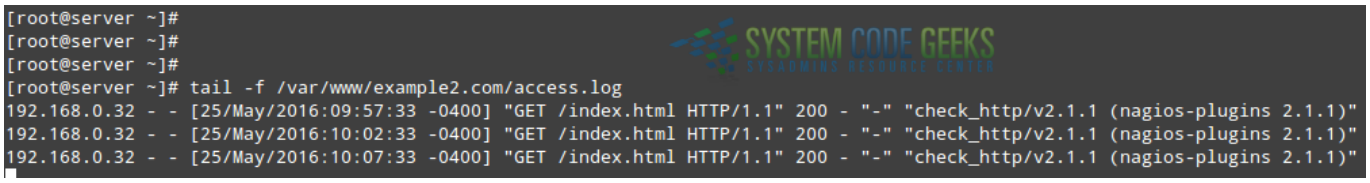

Figure 2.2: The check_http plugin in action: viewing the Apache logs on the remote host

As you can see in Fig. 2.2 above, checks are performed at 5-minute intervals by default. If you want to change this setting, add the `check_interval` directive to the service definition followed by the desired number of minutes.

We can even go one step further and check if the SSL certificate in the remote host will still be valid for the next 366 days (it won't if we followed the instructions given in Apache enable SSL / TLS tutorial). Feel free to choose a higher number if you wish.

To do this, change the check_command line as follows (please note that, according to the help guide, the URL is not checked when verifying the validity of a certificate - that is why we removed the -u option):

```
check_command           check_http!-I 192.168.0.29 -C 366
```

Nagios will perform the check and return a warning, as you can see in Fig. 2.3:

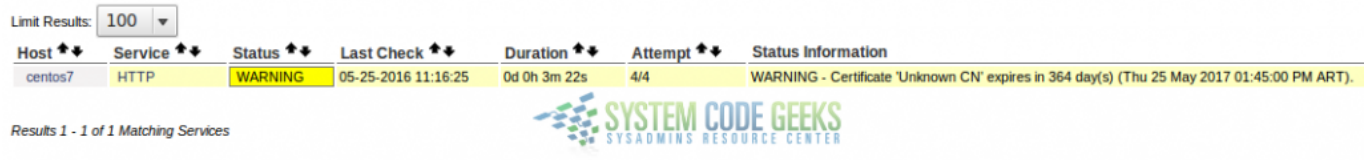

Figure 2.3: A warning message is issued when the certificate validity is not within the specified limit

After you're done with the above test, you should change the certificate expiration check to a more reasonable value (60 days, for example). This will warn you to renew the certificate when its expiration date is less than 2 months ahead:

```
check_command           check_http!-I 192.168.0.29 -C 60
```

2.2 Introducing Nagios Remote Plugin Executor (NRPE)

Up to this point you have learned how to use plugins to check the status of network services on remote hosts from a centralized Nagios server, without having to install any agents on those remote hosts. If, additionally, you need to verify other metrics such as disk usage, CPU load, number of logged-on users, you will need to consider using NRPE.

This tool consists of a service that allows the Nagios server to execute plugins on remote machines and report the results back to the server. In short, the `check_nrpe` plugin on the Nagios server communicates with the NRPE service running on the remote host, which in turn runs the desired plugin (check_load, for example) locally.

In order to use NRPE (the package name is **nrpe** in Fedora-based distributions and **nagios-nrpe-server** in Debian and derivatives), let's set up the remote CentOS 7 host:

Step 1 - Download and compile NRPE and configure it to run under xinetd:

```
yum install xinetd
wget https://sourceforge.net/projects/nagios/files/nrpe-2.x/nrpe-2.15/nrpe-2.15.tar.gz
tar xvf nrpe-2.15.tar.gz
cd nrpe-2.15
./configure
make all
make install-plugin
make install-daemon
make install-daemon-config
make install-xinetd
```

Edit /etc/xinetd.d/nrpe and allow connections from the Nagios server (192.168.0.32), as shown in Fig. 2.4:

```
only_from       = 127.0.0.1 192.168.0.32
```

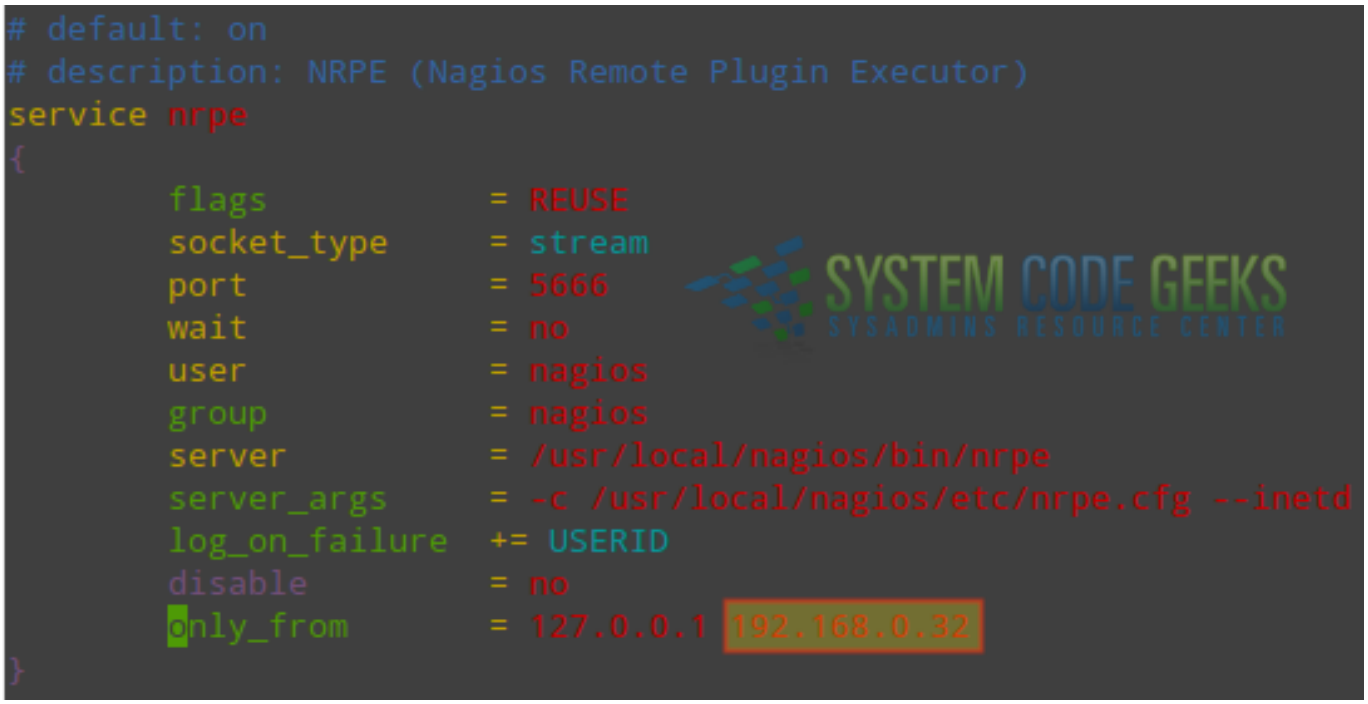

Figure 2.4: Allowing connections to the xinetd daemon from the Nagios server (192.168.0.32)

Step 2 - Create an user account and group (nagios)

```
useradd nagios
```

Step 3 - Define the commands to be used in the service declaration in the server:

Open `/usr/local/nagios/etc/nrpe.cfg` and make sure the following lines are present (see Fig. 2.5). You will want to replace the partition to be checked (/dev/mapper/centos_centos7—2-root) with the one that applies to your case.

```
command[check_users]=/usr/local/nagios/libexec/check_users -w 5 -c 10
command[check_load]=/usr/local/nagios/libexec/check_load -w 15,10,5 -c 30,25,20
command[check_disk]=/usr/local/nagios/libexec/check_disk -w 20% -c 10% -p /dev/mapper/ ↩
    centos_centos7--2-root
```

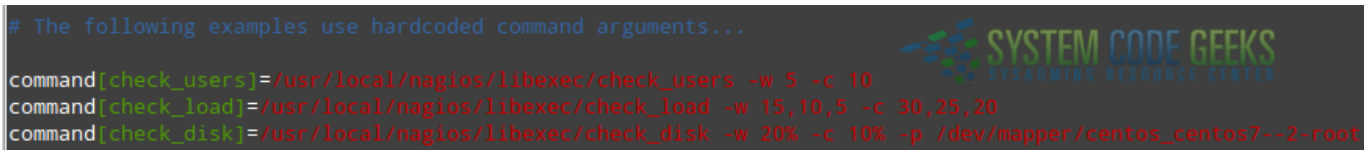

Figure 2.5: Viewing command definitions in the NRPE configuration file

In each case, the -w and -c flags indicate the warning and critical thresholds, respectively. In other words, when the free disk space reaches the 20% threshold, NRPE will raise a warning message, whereas if the available space is 10% or less, a critical message will be issued. Likewise, you will receive warning and critical notifications when the CPU load reaches 15,10,5 and 30,25,20. The same is true for the number of logged-on users - warning and critical messages for 5 and 10 users.

Step 4 - Enable port 5666/tcp (the default port where NRPE listens on) in the built-in firewall:

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

And add the service description at the bottom of /etc/services:

```
echo "nrpe              5666/tcp                # NRPE" >> /etc/services
```

Finally, start / enable xinetd to start on boot(this will manage the NRPE daemon):

```
systemctl start xinetd
systemctl enable xinetd
```

In this example, we are using xinetd to provide access control based on the IP address of the Nagios server, as we can see in Figure 2.4 above.

Step 5 - Set up the service definitions in the Nagios server (`/usr/local/nagios/etc/servers/centos7.cfg`):

```
define service {
use                     generic-service
host_name               centos7
service_description     Check load
check_command           check_nrpe!check_load
notifications_enabled   1
}
define service {
use                     generic-service
host_name               centos7
service_description     Check disk
check_command           check_nrpe!check_disk
notifications_enabled   1
}
define service {
use                     generic-service
host_name               centos7
service_description     Check logged-on users
check_command           check_nrpe!check_users
notifications_enabled   1
}
```

Please note how the parameters being passed to `check_nrpe` match the command definition in the remote host, as shown in Step 3.

2.3 Testing NRPE

As always, don't forget to restart Nagios for the changes to take effect. Then go to the Nagios web interface and check the status of the metrics being monitored (refer to Fig. 2.6 for details):

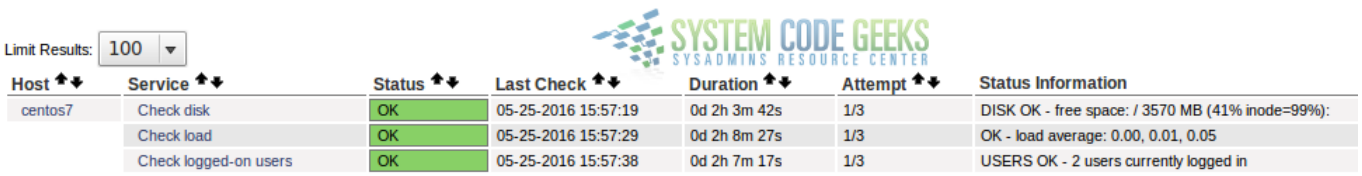

Figure 2.6: Checking metrics in the remote hosts via the Nagios web interface

At this point, your Nagios server should be monitoring the disk usage, CPU load, and number of logged-on users in the remote CentOS 7 server.

If you face any issues, you can refer to the Nagios log (`/usr/local/nagios/var/nagios.log`) in the server and the generic message log (/var/log/messages) in the remote host as the first source of information for troubleshooting.

2.4 Summary

In this article we explained how to use Nagios plugins to monitor network services on remote hosts, and NRPE to monitor several machine-specific metrics on those hosts. How do you decide which one (generic plugins or NRPE) you should use? For simplicity, use the answers to the following questions to determine which approach applies to your case:

- Do you need to verify the status of network services running on remote hosts? Use plugins running on the central Nagios server.
- Do you need to check machine-specific metrics on the remote hosts? Use NRPE to run plugins on the remote hosts.

Last but not least, remember that Nagios is a monitoring tool and not Aladdin's lamp. It does still require user intervention to prevent hardware damage when metrics go beyond the established limits, and human brains in action to troubleshoot issues when they occur. As with any other tool, you need to learn how to use it in order to make the most of it. We hope that this series will provide you with the necessary starting skills to accomplish that goal.

Chapter 3

Monitoring through SNMP

In the previous two articles (Nagios Core Installation and Configuration on Ubuntu Server and Using Nagios plugins and NRPE to check network services and metrics on remote hosts) we discussed how to install Nagios Core, plugins and NRPE to monitor host status (up / down), several network services running on Linux servers, and machine-specific metrics such as the number of logged-on users, processes, and CPU load, to name a few examples.

Additionally, you can use the Simple Network Management Protocol (SNMP) with Nagios to manage other types of network devices, such as printers, routers, and switches. This will be the topic that we will address in this guide.

SNMP not only allows to collect information about a network device, but also to modify the behavior of such device. Under the hood, this protocol exposes device data in the form of variables, which can then be queried and / or set by controlling applications. A classic example of modifying device data consists of changing the date and time on network printers, and retrieving print counts. However, not all variables are rw (read and write), and most of them are only read-only. If in doubt, refer to the device documentation.

3.1 Prerequisites

As explained earlier, you will typically use SNMP with Nagios to monitor network devices, as opposed to using plugins or NRPE to check services and system information associated with a Linux system. However, you can still use SNMP in the latter case as well. For simplicity, we will use the same CentOS 7 box we have been utilizing so far.

In order to use the CentOS 7 box to simulate a regular network device, we will need to perform some preliminary work before proceeding. This prework will consist of the following steps:

Step 0 - Review basic SNMP concepts. Although a thorough discussion about SNMP is out of the scope of this article, we will try to point out the basics as we go. However, should you need a more detailed explanation, feel free to take a look at this excellent question (with answers) in the Ubuntu forums. Bookmark that page in case you need to refer to it for clarifications later.

Our test environment consists of the following key components (essential in SNMP monitoring):

- Managed device: CentOS 7 box.
- Agent: the snmpd service running on the CentOS machine.
- A Network Management System: Nagios running on the Ubuntu box.

Step 1 - Stop and disable xinetd (which will prevent the service from starting automatically on subsequent boots) on the managed device:

```
systemctl stop xinetd && systemctl disable xinetd
```

Step 2 - Install the SNMP packages on the Ubuntu box (this is required before proceeding with Step 3):

```
sudo apt-get install snmp snmpd libsnmp-dev
```

and in the CentOS 7 machine (necessary to monitor this host via SNMP):

```
yum install net-snmp net-snmp-utils
```

Finally, make sure that the snmpd service is started in the current session and on subsequent boots (CentOS):

```
systemctl start snmpd && systemctl enable snmpd
```

Step 3 - Recompile the Nagios plugins on the Ubuntu box. Since the SNMP packages were not installed at the time when we first compiled the plugins (see Step 9 in Nagios Core Installation and Configuration on Ubuntu Server for your reference), the check_snmp plugin could not be added to /usr/local/nagios/libexec.

After completing Steps 2 and 3, verify that the check_snmp plugin is now present inside /usr/local/nagios/libexec, as indicated in Fig. 3.1:

```
ls -l /usr/local/nagios/libexec | grep snmp
```

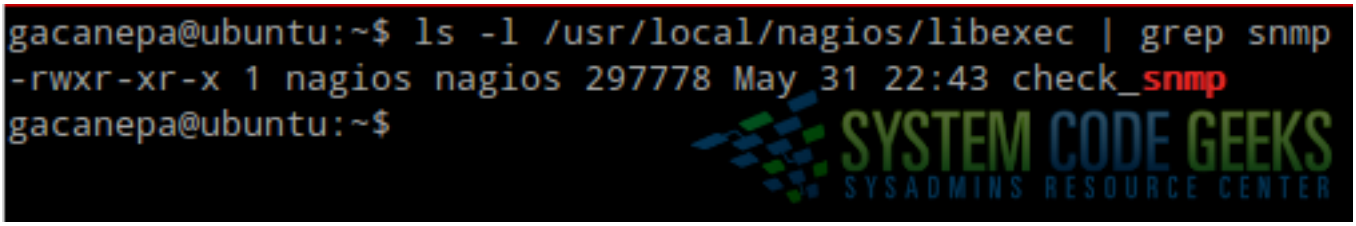

Figure 3.1: Checking the presence of the check_snmp plugin

Step 4 - Remove (or comment out) the lines in the /usr/local/nagios/etc/servers/centos7.cfg file except for the host definition (we will later add a couple of service definitions).

```
define host {
host_name               centos7
alias                   My CentOS 7 server
address                 192.168.0.29
max_check_attempts      3
check_period            24x7
check_command           check-host-alive
contacts                nagiosadmin
notification_interval   60
notification_period     24x7
}
```

Step 5 - Open port 161/udp on the CentOS 7 host (for your information, this is the port where SNMP traffic will be directed to):

```
firewall-cmd --add-port=161/udp
firewall-cmd --add-port=161/udp --permanent
```

Steps 0 through 5, as outlined above, represent the essential preparations in order for Nagios to monitor the CentOS 7 system via SNMP. In the following sections we will get to the nitty-gritty of the corresponding configurations.

3.2 Configuring SNMP on the managed device

Once you have installed and started snmpd in the CentOS 7 box, the variables are only accessible from that host. We need to allow the Ubuntu box to query those variables. To do that, rename /etc/snmp/snmpd.conf to /etc/snmp/snmpd.conf.orig, and -for simplicity- create a new snmpd.conf file with only the following lines in it. The last three lines (beginning with disk) represent the mount points of existing root, projects, and backups logical volumes, where we will want to check the percentage of disk usage via SNMP:

```
rocommunity public 192.168.0.0/24
disk /
disk /home/projects
disk /home/backups
```

After restarting snmpd, hosts in the 192.168.0.0/24 network will be allowed to query (ro: read-only) the SNMP variables from the CentOS 7 machine.

Please keep in mind that this is a basic configuration. You can also restrict access to the SNMP variables by host using the IP of the allowed machine. To explore further options, run the snmpconf command on the managed device after making a copy of snmpd.conf.

3.3 Configuring Nagios for SNMP

To read or set variables via SNMP, object identifiers (OIDs) are used. A list of common OIDs are available in https://www.oid-info.com/basic-search.htm.

Here are some sample OIDs that we are going to use in this guide:

- System uptime: 1.3.6.1.2.1.25.1.1.0
- Percentage of disk space usage (**first** mount point indicated in snmpd.conf): 1.3.6.1.4.1.2021.9.1.9.**1**
- Percentage of disk space usage (**second** mount point indicated in snmpd.conf): 1.3.6.1.4.1.2021.9.1.9.**2**
- Percentage of disk space usage (**third** mount point indicated in snmpd.conf): 1.3.6.1.4.1.2021.9.1.9.**3**
- Total RAM installed: 1.3.6.1.4.1.2021.4.5.0

Object IDentifiers are unique across devices and vendors. In other words, the same information is accessible using the same OID. However, some vendors may have specific OIDs for their devices.

In the Nagios server, make sure the following block is present in /usr/local/nagios/etc/objects/commands.cfg:

```
define command{
command_name    check_snmp
command_line    $USER1$/check_snmp -H $HOSTADDRESS$ $ARG1$
}
```

And append the following command definitions to the same file. They will be used to check the uptime, the percentage of disk usage, and the total RAM installed.

```
# Uptime via SNMP
define command{
command_name    SNMP-Uptime
command_line    $USER1$/check_snmp -o 1.3.6.1.2.1.25.1.1.0 -H $HOSTADDRESS$ $ARG1$
}
# Percentage of disk usage (/)
define command{
command_name    SNMP-DiskUsagePercentageRoot
command_line    $USER1$/check_snmp -o 1.3.6.1.4.1.2021.9.1.9.1 -H $HOSTADDRESS$ $ARG1$ -w  ↩
    60 -c 80
}
# Percentage of disk usage (/home/projects)
define command{
command_name    SNMP-DiskUsagePercentageProjects
command_line    $USER1$/check_snmp -o 1.3.6.1.4.1.2021.9.1.9.2 -H $HOSTADDRESS$ $ARG1$ -w  ↩
    60 -c 80
}
# Percentage of disk usage (/home/backups)
```

```
define command{
command_name    SNMP-DiskUsagePercentageBackups
command_line    $USER1$/check_snmp -o 1.3.6.1.4.1.2021.9.1.9.3 -H $HOSTADDRESS$ $ARG1$ -w  ↩
    60 -c 80
}
# Total RAM installed
define command{
command_name    SNMP-TotalRAMInstalled
command_line    $USER1$/check_snmp -o 1.3.6.1.4.1.2021.4.5.0 -H $HOSTADDRESS$ $ARG1$
}
```

Finally, we will add the corresponding service definitions to apply the above commands to our CentOS box. To do that, insert the following lines in /usr/local/nagios/etc/servers/centos7.cfg:

```
define service{
use                     generic-service
host_name               centos7
service_description     System uptime
check_command           SNMP-Uptime!-C public
}
define service{
use                     generic-service
host_name               centos7
service_description     Disk used percentage of /
check_command           SNMP-DiskUsagePercentageRoot!-C public
}
define service{
use                     generic-service
host_name               centos7
service_description     Disk used percentage of /home/projects
check_command           SNMP-DiskUsagePercentageProjects!-C public
}
define service{
use                     generic-service
host_name               centos7
service_description     Disk used percentage of /home/backups
check_command           SNMP-DiskUsagePercentageBackups!-C public
}
define service{
use                     generic-service
host_name               centos7
service_description     System uptime
check_command           SNMP-Uptime!-C public
}
```

Once Nagios is restarted, we can open the web user interface and check the status of the services that we just defined as we can see in Fig. 3.2:

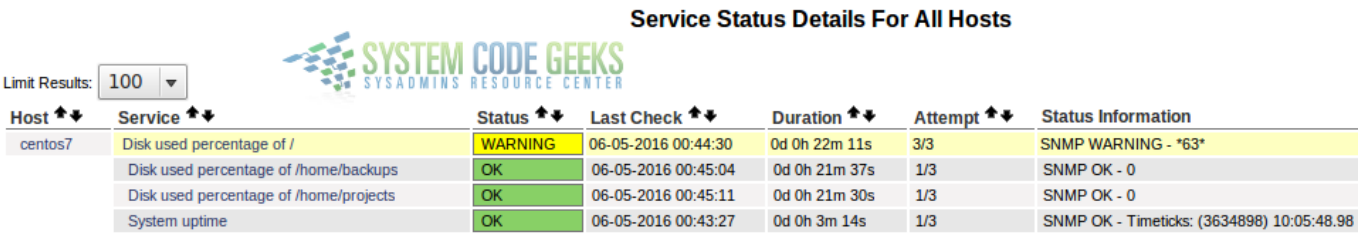

Figure 3.2: Displaying system variables acquired through SNMP in Nagios

The **WARNING** status in the percentage of disk usage corresponding to the root partition is caused by the -w flag followed by

60 in the SNMP-DiskUsagePercentageRoot command definition; that is, raise a warning message if the disk usage is above 60%.

3.4 Summary

In this article we have reviewed some essential concepts about SNMP and explained how to configure Nagios to monitor system metrics in the managed device using that protocol. To check other types of network devices, consult the specific documentation. The only difference is that you will not need to set up a SNMP agent in a network printer, or a router. The rest of this guide should apply to such cases without major modifications.

Chapter 4

Alternatives: Centreon and Icinga

During the last 3 articles ("Nagios Core Installation and Configuration on Ubuntu Server", "Using Nagios plugins and NRPE to check network services and metrics on remote hosts", and "Nagios monitoring through SNMP"), we introduced you to Nagios Core and explained how to monitor network devices and servers via plugins and the Simple Network Management Protocol (SNMP).

In the ecosystem of monitoring tools, there are other heavyweights you may want to consider using in your environment. Centreon and Icinga, which we will explore in this guide, are two examples. Both being open source applications, they allow you to lower the Total Cost of Ownership (TCO) for your business while still providing effective solutions for system administrators.

We will begin by introducing Icinga and Centreon, to later highlight some of the similarities and differences between them and Nagios. However, please note that this article is not intended to be an exhaustive installation guide nor provide a detailed configuration help for either Centreon or Icinga. Our expectation is to give you enough info so as to help you make an informed decision when choosing a network monitoring system.

4.1 Introducing and installing Icinga

Originally started as a fork of Nagios in 2009, Icinga aimed to overcome certain flaws in the then-current development process of Nagios, fix bugs, and add new features required by the community as well, including a more modern web interface, an improved Service Level Agreement (SLA) reporting module, and connectors for several Relational Database Management Systems (Oracle, MySQL / MariaDB, and PostgreSQL).

Although the first version of Icinga (Icinga 1) was a Nagios fork, Icinga 2 was written from scratch in an attempt to eliminate the issues that existed because of the inherited Nagios code base.

You can choose to install Icinga either from the Ubuntu default repositories (v1) or from the Icinga Personal Package Archives (PPA, v2). Icinga 1 is only being currently maintained for security and bug fixes, but other than that it is not under active development anymore.

Before installing Icinga on our Ubuntu box (192.168.0.32), we will stop the Nagios service and prevent it from starting automatically on subsequent boots. Although this is not specifically required in order to set up Icinga, having more than one monitoring tool running on our server may end up returning misleading results.

```
sudo service nagios stop
sudo rm /etc/rcS.d/S99nagios
```

If you choose to undo this after trying Icinga, you can restart and enable Nagios again as explained in Nagios Core Installation and Configuration on Ubuntu Server.

As explained above, the safest choice for a new Icinga installation is version 2, which you can install as follows:

```
sudo add-apt-repository ppa:formorer/icinga
sudo apt-get update
sudo apt-get install icinga2 icinga2-classicui icinga2-doc icinga2-ido-mysql mysql-server  ↩
    libdbd-mysql mysql-client
```

During the installation process, you will be prompted to enter the following information:

- The password for the Icinga administrator user (icingaadmin).
- Configure a database for Icinga. This will allow to import Icinga status messages into a SQL (MySQL or PostgreSQL) database.
- Enter the desired password for the MySQL administrative account.
- Enter a password for icinga2-ido-mysql to register with the database server. If you leave this field blank, a random password will be generated.

Once the installation is complete, Icinga will be configured automatically on subsequent boots (a link to `/etc/init.d/icinga2` was created as `/etc/rc2.d/S20icinga2`) and the web interface is available at https://192.168.0.32/icinga2-classicui. To login, use the password you chose for icingaadmin earlier. If you now take a couple of minutes to browse the web interface, you will realize it resembles the Nagios UI, with many of its menus and features present (see Fig. 4.1 for more details).

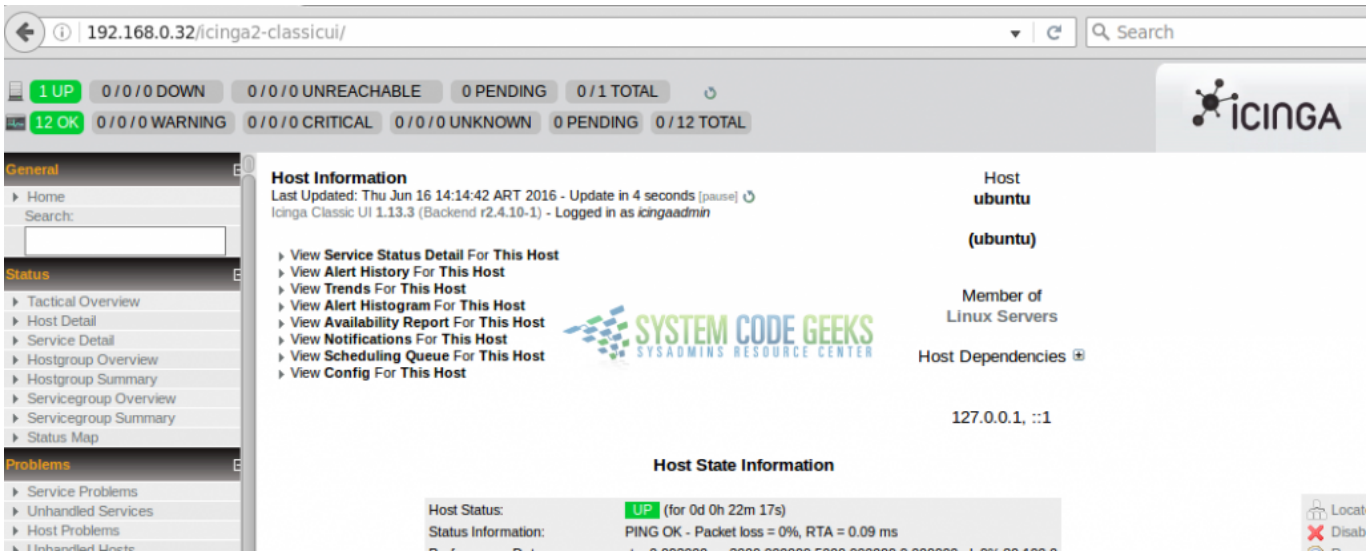

Figure 4.1: Exploring the web interface of Icinga 2

Refer to the Icinga 2 documentation if you are interested in setting up hosts and services to be monitored (by default, it will only monitor the local system and classic metrics such as disk space usage, number of processes, logged on users, and system packages for which a newer version is available.

4.2 Introducing and installing Centreon

Perhaps what makes Centreon stand out from the crowd of the plethora of monitoring tools is the fact that it is distributed as an ISO. In addition, it is installed as a standalone Linux distribution -instead of a package- as opposed to Nagios and Icinga.

To install Centreon, download the ISO from the project's website (click on Get Centreon 3.3 → Direct download), and burn it to a DVD. Alternatively, you may want to install it on Virtualbox as you would with a regular Linux distribution. This is, in fact, the case, as Centreon 3.3 (the latest version) is built on a CentOS 6.7 operating system.

If you don't know how to set up a virtual machine in VirtualBox, or need to refresh your memory, feel free to check our Virtualbox series, beginning with Virtualization with VirtualBox: Installation and Configuration.

During the installation process you will be prompted to decide which kind of Centreon server you want to configure. Choose the default (Central server with database) as shown in Fig. 4.2.

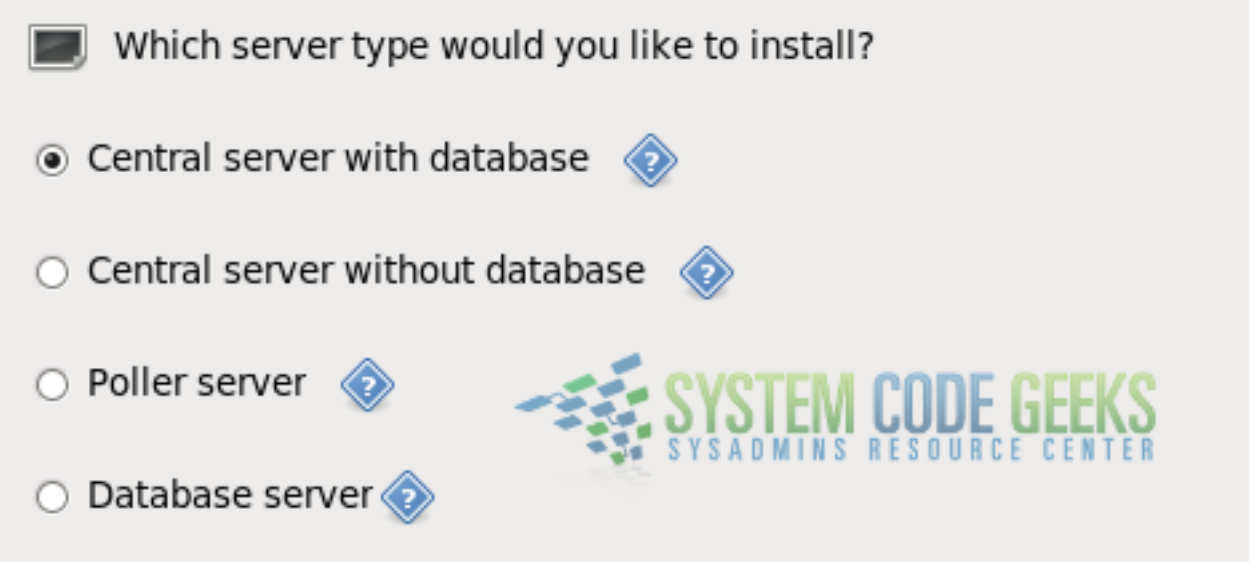

Figure 4.2: Choosing the desired Centreon installation type

After installing Centreon, you will need to:

1) Set your timezone (if you didn't do so earlier) in the `/etc/php.ini` file. Fig. 4.3 shows the relevant line and my choice of timezone (America/Argentina/San_Luis). You can pick the timezone that matches your location from the List of Supported Timezones:

Figure 4.3: Setting the timezone in the php.ini file for Centreon

2) Run mysql_secure_installation to set a password for the MariaDB root user (you will need this later to complete the configuration of Centreon). Other than this, you may either accept the default settings presented by the procedure, or use your own.

3) Launch a web browser and point it to the IP of the Centreon server followed by /centreon (https://192.168.0.50/centreon for example) and you will be taken through a series of steps to complete the configuration. Among other things, you will be asked to enter

- Password, full name, and email of the Centreon admin.
- MariaDB root password, and an additional password for the MariaDB centreon user, as shown in Fig. 4.4:

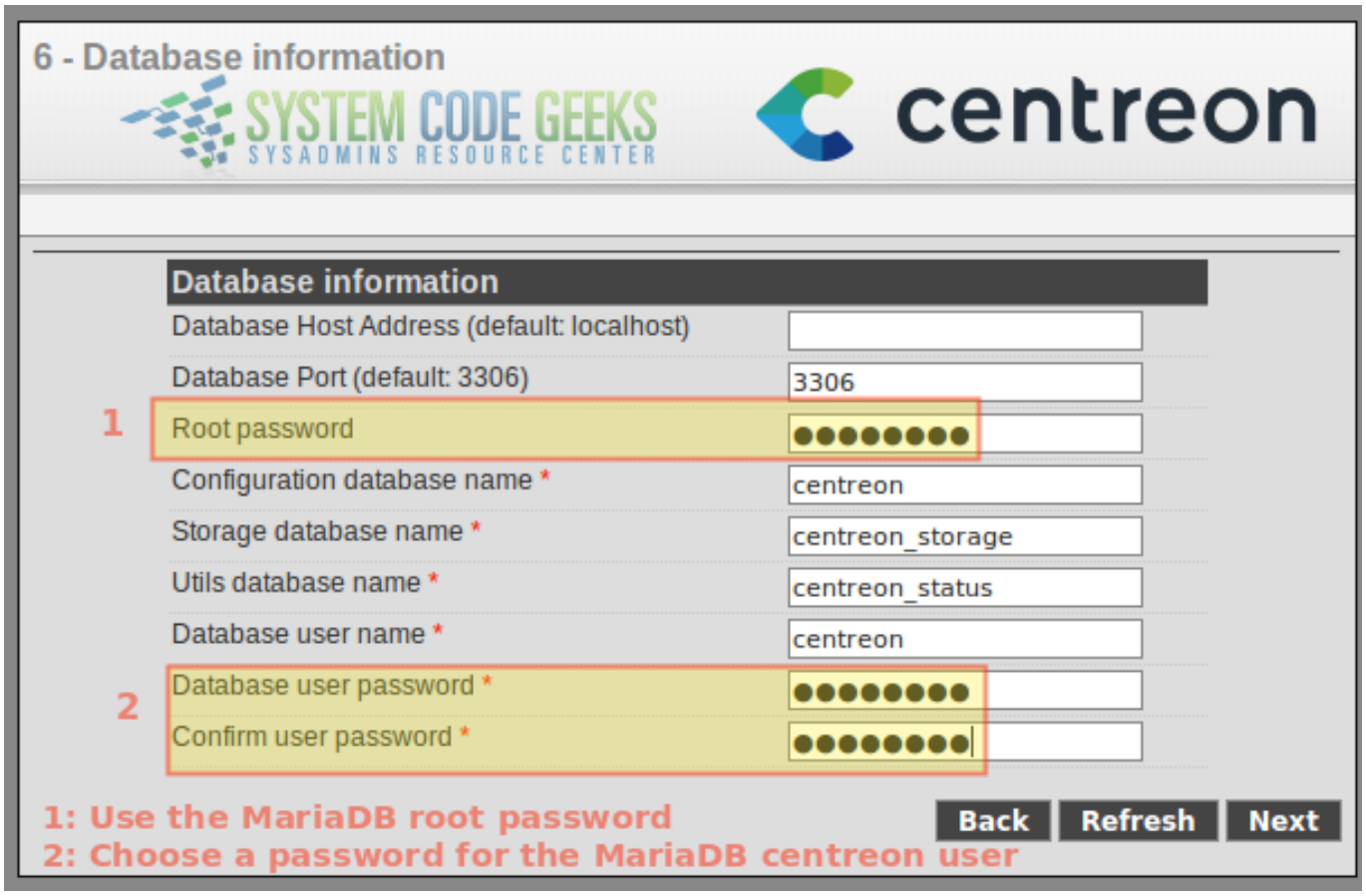

Figure 4.4: Database configuration for Centreon

If you encounter required fields (marked with an asterisk) where the corresponding textbox has already been filled in, you can safely leave the default value as it is. Otherwise, provide the information listed above.

After this, you will be taken to the login page, where you will need to use the credentials of the Centreon admin user (not the MariaDB one). Upon successful login, you will see the initial dashboard where no hosts or services have been added yet.

There seems to be a bug that prevents Centreon from becoming aware of itself unless you set up ssh access to the localhost for the centreon admin user. Before proceeding to add hosts and services to be monitored, run the following commands as root:

a) Set a password for the centreon Unix user account:

```
passwd centreon
```

b) Set access over ssh to the localhost for the centreon user:

```
ssh 127.0.0.1 -l centreon
```

Now proceed to add a host as explained here.

Finally, restart Apache and the Centreon-related services:

```
service httpd restart
service centengine restart
service cbd restart
```

When you're done, click on Hosts or Services at the top of the screen to view the corresponding information (see Fig. 4.5):

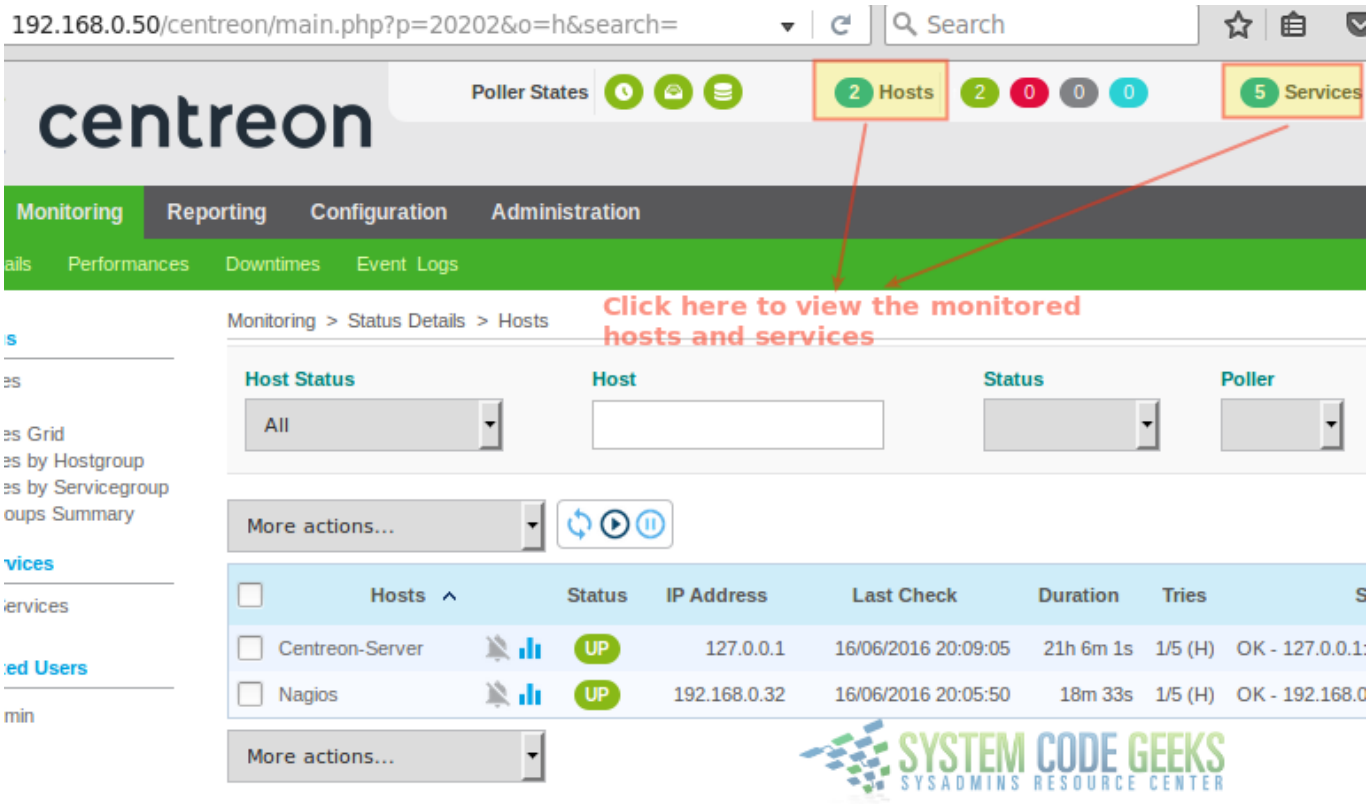

Figure 4.5: Viewing the status of monitored hosts and services in Centreon

Feel free to examine the interface and play with it a little bit. Refer to the Centreon official docs for more details.

4.3 A comparison between Nagios, Centreon, and Icinga

After installing Icinga and Centreon, we are in a better position to determine which is the most appropriate solution for our monitoring needs. Here are some facts that may help you to take a decision:

- Icinga and Nagios require you to define hosts and services using plain text files, whereas in Centreon you can use the web interface directly. Chances are that you, as a sysadmin, feel comfortable using command line editors; if so, you won't have a problem configuring hosts and services in Nagios or Icinga. But keep in mind that Centreon allows you to do the same job with only a few clicks.
- At least in my experience, Centreon is somewhat buggy and you have to dig around in forums a lot in order to get things up and running. Another way to put this is saying that it has a steeper learning curve than Nagios or Icinga. For example, it took me a lot of searching before I found out you have to generate the configuration and export it to the monitoring engine each time you add / update / delete a new host or service before changes are put into effect.
- Another downside of Centreon when compared to Nagios and Icinga is that the former does not allow to export data in the form of reports or support authentication via Active Directory, while the latter two do.
- On the bright side -and not that it actually matters- Centreon's web interface is more polished than Nagios' or Icinga's.
- Last, but not least, is the licensing terms and available support. For paid versions, Nagios and Centreon provide user support and training, with Nagios being the least expensive alternative (< $2500 vs. +$3000 per year for 1000 monitored hosts and one central server). Icinga, on the other hand, is supported by the community but has the advantage of providing all the features of Nagios and more.

4.4 Summary

In this article we have explored Icinga and Centreon as two alternatives to Nagios. With the pros and cons of each solution, you will be able to make an informed decision when choosing a monitoring tool for your network. Hope it helps.