Nagios

Nagios (http://www.nagios.org/) is an open source host, service and network monitoring program. It monitors specified hosts and services, alerting you to any developing issues, errors or improvements. This article describes the installation and configuration of Nagios.

Contents

Features

Some of Nagios' features **include (http://nagios.sourceforge.net/docs/3_0/about.html#whatis)**:

- Monitoring of network services (SMTP, POP3, HTTP, NNTP, PING, etc.)
- Monitoring of host resources (processor load, disk usage, etc.)
- Simple plugin design that allows users to easily develop their own service checks
- Parallelized service checks
- Ability to define network host hierarchy using "parent" hosts, allowing detection of and distinction between hosts that are down and those that are unreachable
- Contact notifications when service or host problems occur and get resolved (via email, pager, or user-defined method)
- Ability to define event handlers to be run during service or host events for proactive problem resolution
- Automatic log file rotation
- Support for implementing redundant monitoring hosts

- Optional web interface for viewing current network status, notification and problem history, log file, etc.

The following installation and configuration were tested using nagios 3.2.0-1, **Apache** web server 2.2.14-2, and **PHP**5 5.3.1-3 by **awayand (https://bbs.archlinux.org/viewtopic.php?id=88461)**.

Webserver

According to the **official documentation (http://nagios.sourceforge.net/docs/3_0/about.html)** a webserver is not required, but if you wish to use any of the CGI features then a webserver (apache preferred), PHP (**php-apache**) for it and the gd library are required. This is assumed for this installation

Installation

Before installation, it's a good idea to make sure you have prerequisites installed, e.g. if you're using nginx then: nginx, php, php-fpm, fcgiwrap might be a good start.

Install `nagios (https://aur.archlinux.org/packages/nagios/)`AUR from the **AUR**.

Users may also want to install monitoring-plugins (https://www.archlinux.org/packages/?name=monitoring-plugins). When you do, make sure to edit `/etc/nagios/resource.cfg` later to reflect the new paths:

```
#$USER1$=/usr/share/nagios/libexec
$USER1$=/usr/lib/monitoring-plugins
```

Nagios Configuration

Copy the sample config files as root:

```
cp /etc/nagios/cgi.cfg.sample /etc/nagios/cgi.cfg
cp /etc/nagios/resource.cfg.sample /etc/nagios/resource.cfg
cp /etc/nagios/nagios.cfg.sample /etc/nagios/nagios.cfg
cp /etc/nagios/objects/commands.cfg.sample /etc/nagios/objects/commands.cfg
cp /etc/nagios/objects/contacts.cfg.sample /etc/nagios/objects/contacts.cfg
cp /etc/nagios/objects/localhost.cfg.sample /etc/nagios/objects/localhost.cfg
cp /etc/nagios/objects/templates.cfg.sample /etc/nagios/objects/templates.cfg
cp /etc/nagios/objects/timeperiods.cfg.sample /etc/nagios/objects/timeperiods.cfg
```

Make owner/group for all the files you just copied and belong to root equal to nagios/nagios:

```
# chown -R nagios:nagios /etc/nagios
```

Create htpasswd.users file with a username and password, eg. nagiosadmin and secretpass

```
# htpasswd -c /etc/nagios/htpasswd.users nagiosadmin
```

If you do not want to install apache-tools, you can run following command

```
# echo -e "nagiosadmin:`perl -le 'print crypt("your-password","salt")'`" > /etc/nagios/htpasswd.users
```

You can also add a different user, but before you can do anything with it in Nagios, you will need to edit `/etc/nagios/cgi.cfg`. You can replace 'nagiosadmin' with the desired user, or, you can append it with comma: nagiosadmin,yourusername,yournextusername etc.

If the owner/group of the nagios-plugins you installed are root:root, the following needs to be done:

```
# chown -R nagios:nagios /usr/share/nagios
```

Once Nagios is configured, it is time to configure the webserver.

Apache Configuration

Edit `/etc/httpd/conf/httpd.conf`, add the following to the end of the file:

```
LoadModule php5_module modules/libphp5.so
Include "conf/extra/nagios.conf"
Include "conf/extra/php5_module.conf"
```

Note: For CGI scripts to work, you'll need to uncomment the following.

```
LoadModule cgi_module modules/mod_cgi.so
```

Copy configure file:

```
# cp /etc/webapps/nagios/apache.example.conf /etc/httpd/conf/extra/nagios.conf
```

Add the apache user http to the group nagios, otherwise you will get the following error when using nagios:

```
Could not open command file '/var/nagios/rw/nagios.cmd' for update!:
```

```
# usermod -G nagios -a http
```

If you are still getting this error, you might need to change the rights on the file:

```
 # chmod 666 /var/nagios/rw/nagios.cmd
```

Nginx Configuration

Apart from php and php-fpm, You should have **fcgiwrap** installed or else CGI scripts will not run.

Example configuration:

```
server {
    server_name     nagios.yourdomain.tld;
    root            /usr/share/nagios/share;
    listen          80;
    index           index.php index.html index.htm;
    access_log      nagios.access.log;
    error_log       nagios.error.log;

    auth_basic            "Nagios Access";
    auth_basic_user_file  /etc/nagios/htpasswd.users;

    # Fixes frames not working
    add_header X-Frame-Options "ALLOW";

    location ~ \.php$ {
        try_files       $uri = 404;
        fastcgi_index   index.php;
        fastcgi_pass    unix:/run/php-fpm/php-fpm.sock;
        include         fastcgi.conf;
    }

    location ~ \.cgi$ {
        root            /usr/share/nagios/sbin;
        rewrite         ^/nagios/cgi-bin/(.*)\.cgi /$1.cgi break;
        fastcgi_param   AUTH_USER $remote_user;
        fastcgi_param   REMOTE_USER $remote_user;
        include         fastcgi.conf;
        fastcgi_pass    unix:/run/fcgiwrap.sock;
    }

    # Fixes the fact some links are expected to resolve to /nagios, see here (http://serverfault.com/questions/653960/nagios-nginx-css-and-image-issues).
    location /nagios {
        alias /usr/share/nagios/share;
    }

}
```

Lighttpd Configuration

Example for lighttpd:

```
$HTTP["url"] =~ "^/nagios" {
        alias.url = (
                "/nagios/cgi-bin" => "/usr/share/nagios/sbin",
                "/nagios" => "/usr/share/nagios/share"
        )

        $HTTP["url"] =~ "^/nagios/cgi-bin" {
                cgi.assign = ( "" => "" )
        }

        auth.backend = "htpasswd"
        auth.backend.htpasswd.userfile = "/etc/nagios/passwd"
        auth.require = ( "" => (
                "method" => "basic",
                "realm" => "nagios",
                "require" => "user=nagiosadmin"
                )
        )
}
```

note that mod_setenv, mod_cgi, mod_alias and mod_auth must be allowed.

PHP Configuration

Edit /etc/php/php.ini to include /usr/share/nagios in the open_basedir directive.

Example configuration:

```
open_basedir = /srv/http/:/home/:/tmp/:/usr/share/pear/:/usr/share/webapps:/etc/webapps:/usr/share/nagios
```

Final Steps

Start/Restart nagios:

```
# systemctl restart nagios
```

Start/Restart apache:

```
# systemctl restart httpd
```

Or Start/Restart nginx:

```
# systemctl restart nginx
```

Now you should be able to access nagios through your webbrowser using the username and password you have created above using htpasswd:

```
http://localhost/nagios
```

Monitor an Archlinux host

You will need `monitoring-plugins (https://www.archlinux.org/packages/?name=monitoring-plugins)` and either `nrpe (https://www.archlinux.org/packages/?name=nrpe)` or use check_by_ssh along with passwordless ssh to monitor your host.

The nrpe configuration is done in /etc/nrpe/nrpe.cfg and the interesting files to monitor will be in /usr/share/nagios/libexec/ . Do not forget to edit nrpe.cfg as it is mostly empty after install.

Quick notes on check_by_ssh: On the monitoring system, su to the user account that Nagios/Icinga/whatever runs as, run ssh-keygen. Create a user on the Arch system to be monitored with the same name and a temporary password eg: # useradd -m -d /home/icinga -s /bin/bash -p icinga icinga. From the monitoring system run this: $ ssh-copy-id ip.ad.dr.ess (ie the IP of the client). Back on the client: # passwd -d icinga (ie clear the temporary password). Verify you can login from the server eg $ ssh icinga@ip.ad.dr.ess.

Many non Arch systems install the monitoring plugins to /usr/lib/nagios/plugins but Arch installs them to /usr/lib/monitoring-plugins/. It may be helpful to create /usr/lib/nagios and symlink ../monitoring-plugins to plugins from that directory.

Here's an example of a command invocation run from the command line as the monitoring system's user. Given the note on paths mentioned above this should work on nearly any Linux (and probably BSD - it does on FreeNAS) distro:

```
$ /usr/lib/nagios/plugins/check_by_ssh -E -H 192.168.100.11 -C "/usr/lib/nagios/plugins/check_disk -w 10 -c 5 --path=/ --units=GB"
```

Plugin check_rdiff

A small guide on monitoring rdiff-backups using a plugin called check_rdiff.

Download and Install

You will need perl installed.

```
cd
wget http://www.monitoringexchange.org/attachment/download/Check-Plugins/Software/Backup/check_rdiff/check_rdiff
cp check_rdiff /usr/share/nagios/libexec
chown nagios:nagios /usr/share/nagios/libexec/check_rdiff
chmod 755 /usr/share/nagios/libexec/check_rdiff
```

Enable sudo for user nagios

Since the perl script check_rdiff needs to run as root, you will have to enable sudo for the nagios user:

```
# visudo
```

This will open the /etc/sudoers file for editing (and protect it from saving errors). then paste the following at the end of the file (you should know how to use the vi editor):

```
nagios  ALL=(root)NOPASSWD:/usr/share/nagios/libexec/check_rdiff
```

Integrate check_rdiff plugin into nagios

Edit /etc/nagios/objects/commands.cfg to include the following command definition:

```
# check rdiff-backup
define command{
        command_name    check_rdiff
        command_line    sudo $USER1$/check_rdiff -r $ARG1$ -w $ARG2$ -c $ARG3$ -l $ARG4$ -p $ARG5$
        }
```

Edit /etc/nagios/objects/localhost.cfg to include checking of rdiff-backup on localhost, for example:

```
define service{
        use                             local-service         ; Name of service template to use
        host_name                       localhost
        service_description             rdiff-backup
        check_command                   check_rdiff!/home/x/rdiffbackup!8!10!500!24
        }
```

Quote from the check_rdiff script content:

The above command checks the repository (-r) which is defined as the destination of the backup, or more specifically, the directory above the rdiff-backup-data directory. It will return warning if the backup has not finished by 8am and critical by 10am. It will also return warning if the TotalDestinationSizeChange is greater than 500Mb. It also get the period set to 24hrs (-p). This is important as the plugin will throw a critical if the backup does not start in time.

Finally, restart nagios:

```
# systemctl restart nagios
```

You can now see the rdiff-backup status by clicking on Services on the left side of the nagios web interface control panel.

Forks

- **Icinga** is a Nagios fork. More details about the fork can be found at **Icinga FAQ: Why a fork? (https://www.icinga.org/icinga/faq/icinga-vs-nagios/)**

- **Naemon** is the new monitoring suite that aims to be faster and more stable, while giving you a clearer view of the state of your network. **Naemon FAQ: Why a fork? (http://www.naemon.org/project.html)**

See also

- **nagios.org (http://www.nagios.org/)** Official website
- **Nagios Plugins (http://www.nagiosplugins.org/)** the home of the official plugins
- **wikipedia.org** Wikipedia article
- **NagiosExchange (http://www.nagiosexchange.org)** overview of plugins, addons, mailing lists for Nagios
- **NagiosForge (http://www.nagiosforge.org/)** a repository for ad

Retrieved from "https://wiki.archlinux.org/index.php?title=Nagios&oldid=475040"

- This page was last edited on 24 April 2017, at 14:07.