

Nagios Installation and Configuration

Contents

Introduction	2
Goals	2
Notes	2
Exercises	2
PART IV - Adding Parent Relationships	2
1. Adding Parents to switches.cfg	2
2. Adding Parents to routers.cfg	3
3. Adding Parents to pcs.cfg	4
4. Restart Nagios and See the Updated Status Map	5
PART V - Create More Host Groups	6
0. Prep	6
1. Update /etc/nagios3/conf.d/hostgroups__nagios2.cfg	6
2. Go back to the web interface and look at your new Host Groups in Nagios.	7
PART VI - Extended Host Information (“making your graphs pretty”)	7
1. Update extinfo__nagios2.cfg	7
PART VII - Create Service Groups	8
1. Create service groups for ssh and http for each set of pcs.	8

PART VIII - Configure Guest Access to the Nagios Web Interface 9

1. You will edit the file `/etc/nagios3/cgi.cfg` to give read-only guest user access to the Nagios web interface. 9
2. Enable External commands in `nagios.cfg` 10

Introduction

Goals

- Install and configure Nagios

Notes

- Commands preceded with “\$” imply that you should execute the command as a general user - not as root.
- Commands preceded with “#” imply that you should be working as root.
- Commands with more specific command lines (e.g. “`rtrX>`” or “`mysql>`”) imply that you are executing commands on remote equipment, or within another program.

Exercises

PART IV - Adding Parent Relationships

Each item is a child of either a switch or a router in our classroom, EXCEPT for your gateway router (`rtrX`) and the other members of your group. We are now going to add a “parents” statement for each device we have configured.

If you are unsure of the parent relationships you can look at our classroom Network Diagram. Remember, the parent relationships are from the point of view of your Nagios instance running on your pc.

1. Adding Parents to `switches.cfg`

```
$ cd /etc/nagios3/conf.d
$ sudo editor switches.cfg
```

Update the entry:

```

define host {
    use          generic-host
    host_name    sw
    alias        Backbone Switch
    address      10.10.0.253
}

```

to be

```

define host {
    use          generic-host
    host_name    sw
    alias        Backbone Switch
    address      10.10.0.253
    parents rtrX
}

```

Where “rtrX” is the gateway router for your group. I.E., for group 1 you would use “rtr1”, for group 2, “rtr2” and so forth.

Save and exit from the file.

2. Adding Parents to routers.cfg

```
$ sudo editor routers.cfg
```

For each entry we will add a “parents” line. So, for the gw definition at the top of the file this should now look like:

```

define host {
    use          generic-host
    host_name    gw
    alias        Classroom Gateway Router
    address      10.10.0.254
    parents      sw
}

```

For all the remaining rtrX entries, except for your group, you should add a line that says:

```
parents sw
```

There should be NO PARENTS entry for your group. If you have an entry for “ap1” (classroom wireless access point), then the parents entry is, also, “sw” - same as the other routers.

So, if you are in group 2, then the entries for groups 1, 2 and 3 would look like:

```
define host {
    use          generic-host
    host_name    rtr1
    alias        Group 1 Router
    address      10.10.1.254
    parents      sw
}

define host {
    use          generic-host
    host_name    rtr2
    alias        Group 2 Router
    address      10.10.2.254
}

define host {
    use          generic-host
    host_name    rtr3
    alias        Group 3 Router
    address      10.10.3.254
    parents      sw
}
```

Update the rest of the file correctly and then save and exit from the file.

3. Adding Parents to pcs.cfg

For all the PC entries you should add a “parents” line that has the router for that PC’s group, EXCEPT FOR YOUR GROUP - there should be no parents entry for the pcs in your group.

For the noc the parent is the core switch or “sw”

```
#
# Classroom NOC
#

define host {
    use          generic-host
```

```

        host_name    noc
        alias        Workshop NOC machine
        address      10.10.0.250
        parents      sw
    }

```

For PCs in Group 1 entries look like:

```

#
# Group 1
#

define host {
    use          generic-host
    host_name    pc1
    alias        pc1
    address      10.10.1.1
    parents      rtr1
}

```

```

define host {
    use          generic-host
    host_name    pc2
    alias        pc2
    address      10.10.1.2
    parents      rtr1
}

```

etc

Do this for all the PCs in the remaining groups. I.E., pc5 in Group 2 has a parents statement of:

```

        parents      rtr2

```

Just remember NOT to add a parents line for the PCs in your group.

Save and exit from the file.

4. Restart Nagios and See the Updated Status Map

```
$ sudo service nagios3 restart
```

If you have errors, fix these and try restarting again.

Open a web browser to <http://pcN.ws.nsrc.org/nagios3> and click on the “Map” link on the left. Your map should now look quite different. You should see a map that represents the Nagios world point of view from your machine.

PART V - Create More Host Groups

0. Prep

In the web view, look at the pages “Hostgroup Overview”, “Hostgroup Summary”, “Hostgroup Grid”. This gives a convenient way to group together hosts which are related (e.g. in the same site, serving the same purpose).

1. Update `/etc/nagios3/conf.d/hostgroups_nagios2.cfg`

For the following exercises it will be very useful if we have created or update the following hostgroups:

```
debian-servers
routers
switches
```

If you edit the file `/etc/nagios3/conf.d/hostgroups_nagios2.cfg` you will see an entry for `debian-servers` that just contains `localhost`. Update this entry to include all the classroom PCs, including the `noc` (this assumes that you created a “`noc`” entry in your `pcs.cfg` file). Remember to skip your PC entry as it is represented by the `localhost` entry.

```
$ sudo editor /etc/nagios3/conf.d/hostgroups_nagios2.cfg
```

Update the entry that says:

```
# A list of your Debian GNU/Linux servers
define hostgroup {
    hostgroup_name  debian-servers
                    alias          Debian GNU/Linux Servers
                    members        localhost
}
```

So that the “members” parameter contains something like this. Use your classroom network diagram to confirm the exact number of machines and names in your workshop.

```

members    localhost,pc1,pc2,pc3,pc4,pc5,pc6,pc7,pc8,pc9,pc10,pc11,pc12, \
           pc13,pc14,pc15,pc16,pc17,pc18,pc19,pc20,pc21,pc22,pc23,pc24,pc25,\
           pc26,pc27,pc28,pc29,pc30,pc31,pc32,pc33,pc34,pc35,pc36

```

Be sure that the end of the line has a “\” to indicate a new line. Otherwise you will get an error when you go to restart Nagios. Remember that your own PC is “localhost”, so skip your pc entry.

Once you have done this, add one more host group for our classroom switch(es). If there is more than just one switch (sw.ws.nsrc.org) include this on the members line below, otherwise the entry at the end of the `hostgroups_nagios2.cfg` file should look like (COPY and PASTE):

```

# A list of our switches
define hostgroup {
    hostgroup_name switches
    alias          Classroom Switches
    members        sw
}

```

When you are done be sure to verify your work and restart Nagios.

2. Go back to the web interface and look at your new Host Groups in Nagios.

PART VI - Extended Host Information (“making your graphs pretty”)

1. Update `extinfo_nagios2.cfg`

If you would like to use appropriate icons for your defined hosts in Nagios this is where you do this. We have the three types of devices:

- Cisco routers
- Cisco switches
- Ubuntu servers

There is a fairly large repository of icon images available for you to use located here:

```
/usr/share/nagios/htdocs/images/logos/
```

these were installed by default as dependent packages of the nagios3 package in Ubuntu. In some cases you can find model-specific icons for your hardware, but to make things simpler we will use the following icons for our hardware:

```
/usr/share/nagios/htodocs/images/logos/base/debian.*  
/usr/share/nagios/htodocs/images/logos/cook/router.*  
/usr/share/nagios/htodocs/images/logos/cook/switch.*
```

The next step is to edit the file `/etc/nagios3/conf.d/extinfo_nagios2.cfg` and tell nagios what image you would like to use to represent your devices.

```
$ sudo editor /etc/nagios3/conf.d/extinfo_nagios2.cfg
```

Here is what an entry for your routers looks like (there is already an entry for debian-servers that will work as is). Note that the router model (3600) is not all that important. The image used represents a router in general.

```
define hostextinfo {  
    hostgroup_name    routers  
    icon_image        cook/router.png  
    icon_image_alt    Cisco Routers (7200)  
    vrml_image        router.png  
    statusmap_image   cook/router.gd2  
}
```

Note how we can simply use “hostgroup_name routers” as this has already been defined in the file `hostgroups_nagios2.cfg`. This makes configuring multiple, like items much simpler.

Now add an entry for your switches. Once you are done check your work and restart Nagios. Take a look at the Status Map in the web interface (Map link on the left). It should be much nicer, with real icons instead of question marks for most items.

PART VII - Create Service Groups

1. Create service groups for ssh and http for each set of pcs.

The idea here is to create three service groups. Each service group will be for a quarter of the classroom. We want to see these PCs grouped together and include status of their ssh and http services. To do this edit and create the file:

```
$ cd /etc/nagios3/conf.d          (just to be sure)
$ sudo editor servicegroups.cfg
```

Here is a sample of the service group for group 1:

```
define servicegroup {
    servicegroup_name    group1-services
    alias                group 1 services
    members    pc1,SSH,pc1,HTTP,pc2,SSH,pc2,HTTP,pc3,SSH,pc3,HTTP,pc4,SSH,pc4,HTTP
}
```

Note that if the members line is too long you can use the “\” at the end to create a new line of members just below.

Note that “SSH” and “HTTP” need to be uppercase as this is how the service_description is written in the file /etc/nagios3/conf.d/services_nagios2.cfg

You should create an entry for other groups of servers too.

CRITICAL - When you create an entry for your group remember to use “localhost” instead of your “pcN” name since you have only defined your pc as localhost in the file hostgroups_nagios2.cfg.

Save your changes, verify your work and restart Nagios. Now if you click on the Service Groups menu item in the Nagios web interface you should see this information grouped together.

PART VIII - Configure Guest Access to the Nagios Web Interface

1. You will edit the file /etc/nagios3/cgi.cfg to give read-only guest user access to the Nagios web interface.

By default Nagios is configured to give full r/w access via the Nagios web interface to the user nagiosadmin. You can change the name of this user, add other users, change how you authenticate users, what users have access to what resources and more via the cgi.cfg file.

First, lets create a “guest” user and password in the htpasswd.users file.

```
$ sudo htpasswd /etc/nagios3/htpasswd.users guest
```

You can use any password you want (or none). A password of “guest” is not a bad choice.

Next, edit the file `/etc/nagios3/cgi.cfg` and look for what type of access has been given to the `nagiosadmin` user. By default you will see the following directives (note, there are comments between each directive):

```
authorized_for_system_information=nagiosadmin
authorized_for_configuration_information=nagiosadmin
authorized_for_system_commands=nagiosadmin
authorized_for_all_services=nagiosadmin
authorized_for_all_hosts=nagiosadmin
authorized_for_all_service_commands=nagiosadmin
authorized_for_all_host_commands=nagiosadmin
```

Now let's tell Nagios to allow the "guest" user some access to information via the web interface. You can choose whatever you would like, but what is pretty typical is this:

```
authorized_for_system_information=nagiosadmin,guest
authorized_for_configuration_information=nagiosadmin,guest
authorized_for_system_commands=nagiosadmin
authorized_for_all_services=nagiosadmin,guest
authorized_for_all_hosts=nagiosadmin,guest
authorized_for_all_service_commands=nagiosadmin
authorized_for_all_host_commands=nagiosadmin
```

Note we do not give the guest user access to system commands, service commands nor host commands.

Once you make the changes, save the file `cgi.cfg`, verify your work and restart Nagios.

To see if you can log in as the "guest" user you will need to clear the cookies in your web browser or open an alternate web browser if you have one. You will not notice any difference in the web interface. The difference is that a number of items that are available via the web interface (forcing a service/host check, scheduling checks, comments, etc.) will not work for the guest user.

2. Enable External commands in `nagios.cfg`

This change is required in order to allow users to "Acknowledge" problems with hosts and services in the Web interface. The default file permissions are set up in a secure way to prevent the web interface from updating nagios, so you need to make them slightly more permissive.

First, edit the file `/etc/nagios3/nagios.cfg`, and change the line:

```
check_external_commands=0
```

to

```
check_external_commands=1
```

Save the file and exit.

Then, perform the following commands to change directory permissions and to make the changes permanent:

```
$ sudo /etc/init.d/nagios3 stop
$ sudo dpkg-statoverride --update --add nagios www-data 2710 /var/lib/nagios3/rw
$ sudo dpkg-statoverride --update --add nagios nagios 751 /var/lib/nagios3
$ sudo /etc/init.d/nagios3 start
```

Once this is done, go to “Problems” > “Services (Unhandled)” and find a service in the red (critical) or yellow (warning) state. Click on the service name. Then under “Service commands” click on “Acknowledge this service problem”.

The problem should disappear from the list of unhandled problems.