Nagios lab parts 4 to 8

Nagios Installation and Configuration

Introduction

Goals

- Nagios Configuration Continued

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

If you look at the Nagios interface for your server and select Status Maps you will see your campus servers and devices centered around your Nagios instance. In order for Nagios to work efficiently you need to include parent relationships for each device defined.

Go to http://oob.hostX.campusY.ws.nsrc.org/nagios3 and click on the “Map” link.

Now we will add parent relationships for router, switch and server.

Adding Parents to routers.cfg

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

And, update bdr1.campusY and core1.campusY entries with a parents line as shown below. The transit1.nren entry can be left alone.

```
define host {
    use         generic-host
    host_name   transit1.nren
    alias       Campus Y Transit Router
    address     transit1.nren.ws.nsrc.org
}

define host {
    use         generic-host
    host_name   bdr1.campusY
    alias       Border Router 1, Campus Y
    address     bdr1.campusY.ws.nsrc.org
    parents     transit1.nren
}

define host {
    use         generic-host
    host_name   core1.campusY
    alias       Core Router Campus Y
    address     core1.campusY.ws.nsrc.org
    parents     bdr1.campusY
}
```

Save and exit from the file.

Adding Parents to switches.cfg

```
$ sudo editor switches.cfg
```

```
define host {
    use         generic-host
    host_name   dist1-b1.campusY
    alias       Distribution Switch 1, Building 1, Campus Y
    address     dist1-b1.campusY.ws.nsrc.org
    parents     core1.campusY
}

define host {
    use         generic-host
    host_name   dist1-b2.campusY
    alias       Distribution Switch 1, Building 2, Campus Y
    address     dist1-b2.campusY.ws.nsrc.org
    parents     core1.campusY
}
```

Do you understand why both switches have the same parent? If not, ask your instructor or assistant to explain.

Save and exit from the file.

Adding Parents to pcs.cfg

We will leave this exercise up to you. It should be fairly simple. All your campus servers have the same parent: core1.campusY.

Edit the pcs.cfg file:

```
$ sudo editor pcs.cfg
```

and at the of each entry add the line:

```
    parents     core1.campusY
```

Be sure to change “Y” to be your campus number.

Be sure that you do not add core1.campusY as a parent for your *own* host. If you don’t understand why ask your instructor.

Save and exit from the file when you are done with all entries.

Reload the Nagios Configuration and See the Updated Status Map

```
$ sudo systemctl reload nagios3
```

If you see errors, fix them and try reloading again.

Open a web browser to http://oob.hostX.campusY.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, but with everything in proper hierarchy based on the “parents” entries that you have just done.

PART V - Create More Host Groups

Preparation

In the web view, look at the pages “Hostgroups”, “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).

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:

```
       ubuntu-servers
       routers
       switches
```

If you edit the file /etc/nagios3/conf.d/hostgroups_nagios2.cfg you will see an entry for ubuntu-servers that just contains localhost. Update this entry to include all the your campus servers.

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

Update the entry that says:

```
# A list of your Ubuntu Linux servers
define hostgroup {
        hostgroup_name  ubuntu-servers
                alias           Ubuntu 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
host1.campusY,host2.campusY,host3.campusY,host4.campusY, \
             host5.campusY,host6.campusY,srv1.campusY
```

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 reload the Nagios configuration.

Once you have done this, add one more host group for our classroom switch(es).

```
# A list of our switches
define hostgroup {
        hostgroup_name  switches
                alias           Classroom Switches
                members         dist1-b1.campusY,dist1-
b2.campusY
        }
```

When you are done be sure to verify your work and reload the Nagios configuration.

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

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

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/htodcs/images/logos/base/debian.*
- /usr/share/nagios/htdocs/images/logos/cook/router.*
- /usr/share/nagios/htdocs/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 reload 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

Create service groups for ssh and http for your campus servers.

The idea is to create service groups for your 7 campus servers. Servicegroups consider the service defined by the combined services to be down if any of the services in a group are down.

In this case we’ll group together ssh and http. In real life you might do msyql, imap, smtp, http and your mta (postfix, mail, exim) if those were services required to deliver a mail interface to your users.

We start by editing the file:

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

For campus 1 this service group would look like:

```
define servicegroup {
    servicegroup_name   campus1-ssh-http
    alias           Campus 1 SSH and Web
    members
host1.campus1,SSH,host1.campus1,HTTP,host2.campus1,SSH,host2.cam
pus1,HTTP, \

host3.campus1,SSH,host3.campus1,HTTP,host4.campus1,SSH,host4.cam
pus1,HTTP, \

host5.campus1,SSH,host5.campus1,HTTP,host6.campus1,SSH,host6.cam
pus1,HTTP, \
             srv1.campus1,SSH,srv1.campus1,HTTP
            }
```

We used “\” to indicate a new line. Without this you will see errors.

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

Save your changes, verify your work and reload 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

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 if you plan for this to be a r/o account.

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 and lines 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 that 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 reload 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.

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 systemctl stop nagios3
$ 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 systemctl start nagios3
```

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.