Nagios Installation and Configuration

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. "RTR-GW>" or "mysql>")
  imply that you are executing commands on remote equipment, or within
  another program.

Exercises

PART I
------

0. Log in to your PC as the sysadm user.

1. Install Nagios

    Install Nagios version 3:

        $ sudo apt-get install nagios3

   Unless you already have an MTA installed, nagios3 will install
   postfix as a dependency. If you are prompted for this, select
   "Internet Site" option. (If you had wanted to use a different MTA like
   exim you'd install it before nagios3)

   You will be prompted to choose a nagiosadmin password. Give it the
normal
   workshop password.

   To get the documentation in /usr/share/doc/nagios3-doc/html/ (which
   can also be read via the nagios web interface), do:

       $ sudo apt-get install nagios3-doc

3. You should already have a working Nagios!

    - Open a browser, and go to

    http://pcX/nagios3/

        Check with the instructor or your neighbor if you are in doubt.

- At the login prompt, login as:

    user: nagiosadmin
    pass: <workshop password>

Browse to the "Host Detail" page to see what's already configured.

4. Let's look at the configuration layout...

```
# cd /etc/nagios3
# ls -l

-rw-r--r-- 1 root root    1882 2008-12-18 13:42 apache2.conf
-rw-r--r-- 1 root root   10524 2008-12-18 13:44 cgi.cfg
-rw-r--r-- 1 root root    2429 2008-12-18 13:44 commands.cfg
drwxr-xr-x 2 root root    4096 2009-02-14 12:33 conf.d
-rw-r--r-- 1 root root      26 2009-02-14 12:36 htpasswd.users
-rw-r--r-- 1 root root   42539 2008-12-18 13:44 nagios.cfg
-rw-r----- 1 root nagios  1293 2008-12-18 13:42 resource.cfg
drwxr-xr-x 2 root root    4096 2009-02-14 12:32 stylesheets

# cd conf.d
# ls -l

-rw-r--r-- 1 root root 1695 2008-12-18 13:42 contacts_nagios2.cfg
-rw-r--r-- 1 root root  418 2008-12-18 13:42 extinfo_nagios2.cfg
-rw-r--r-- 1 root root 1152 2008-12-18 13:42 generic-host_nagios2.cfg
-rw-r--r-- 1 root root 1803 2008-12-18 13:42 generic-
service_nagios2.cfg
-rw-r--r-- 1 root root  210 2009-02-14 12:33 host-gateway_nagios3.cfg
-rw-r--r-- 1 root root  976 2008-12-18 13:42 hostgroups_nagios2.cfg
-rw-r--r-- 1 root root 2167 2008-12-18 13:42 localhost_nagios2.cfg
-rw-r--r-- 1 root root 1005 2008-12-18 13:42 services_nagios2.cfg
-rw-r--r-- 1 root root 1609 2008-12-18 13:42 timeperiods_nagios2.cfg
```

Notice that the package installs files with "nagios2" in their name. This is because they are the same files as were used for the Nagios version 2 Debian package. However there was a change made to the host-gateway configuration file, so this has a new name.

5. You have a config which is already monitoring your own system (localhost_nagios2.cfg) and your upstream default gateway (host-gateway_nagios3.cfg).

Have a look at the config file for the default gateway: it's very simple. (Note: tab completion is useful here. Type "cat host-g" then hit tab; the filename will be filled in for you)

```
    # cat host-gateway_nagios3.cfg
```

It should look something like this:

```
    # a host definition for the gateway of the default route
    define host {
            host_name   gateway
            alias       Default Gateway
            address     10.10.X.254
            use         generic-host
            }
```

It is monitoring the virtual Cisco router which is upstream of your VM.

PART II
Configuring Equipment

--

0. Order of configuration

Conceptually we will build our configuration files from the "nearest" device
then the further away ones.

By going in this order you will have defined the devices that act as parents
for other devices.

Your upstream Cisco virtual router (your PC's gateway) is already defined.

1. The three PCs in your group are directly connected to you with nothing in
between. So there are no dependencies.

Create a new file, 'pcs.cfg', to list the three other PCs in your group. The
example below is ONLY for pc1, which has pc2/pc3/pc4 in its group, so modify
it for your neighbours.

```
    # cd /etc/nagios3/conf.d/
    # editor pcs.cfg
```

```
define host {
    use         generic-host
    host_name   pc1
```

```
    alias       pc1 in group 1
    address     pc1.ws.nsrc.org
}

define host {
    use         generic-host
    host_name   pc2
    alias       pc2 in group 1
    address     pc2.ws.nsrc.org
}

define host {
    use         generic-host
    host_name   pc3
    alias       pc3 in group 1
    address     pc3.ws.nsrc.org
}
```

THE FOLLOWING STEPS 2a - 2c SHOULD BE REPEATED WHENEVER YOU UPDATE THE CONFIGURATION!

2a. Verify that your configuration files are OK:

```
    # nagios3 -v /etc/nagios3/nagios.cfg

    ... You should get something like this:
Warning: Host 'pc2' has no services associated with it!
Warning: Host 'pc3' has no services associated with it!
Warning: Host 'pc4' has no services associated with it!
...
Total Warnings: 3
Total Errors:   0

Things look okay - No serious problems were detected during the check.
Nagios is saying that it's unusual to monitor a device just for its
existence on the network, without also monitoring some service.
```

2b. Reload/Restart Nagios

```
    # service nagios3 restart
```

HINT: You will be doing this a lot. If you do it all on one line, like this,
then you can hit cursor-up and rerun all in one go:

```
    # nagios3 -v /etc/nagios3/nagios.cfg && service nagios3 restart
```

The '&&' ensures that the restart only happens if the config is valid.

2c. Go to the web interface (http://pcX/nagios3) and check that the hosts
   you just added are now visible in the interface. Click on the "Host
Detail" item
   on the left of the Nagios screen to see this. You may see it in
"PENDING"
   status until the check is carried out.

3. Let's configure Nagios to start monitoring the classroom switch and then the backbone router.

Add the switch in a new file:

```
    # cd /etc/nagios3/conf.d
    # editor switches.cfg
```

```
define host {
    use         generic-host
    host_name   bb-sw
    alias       backbone switch
    address     10.10.0.253
    parents     gateway
}
```

And let's create a file for routers:

```
        # editor routers.cfg
```

```
define host {
    use         generic-host
    host_name   bb-gw
    alias       backbone gw
    address     10.10.0.254
    parents     bb-sw
}
```

Notice the "parents" entry. This must point at a device or devices which are
also defined somewhere else in the configuration.

From a topology point of view, pcX cannot reach the switch 'bb-sw' if its gateway is down; so the parent of bb-sw is gateway. Similarly, you cannot reach bb-gw if bb-sw is down, so the parent of bb-gw is bb-sw.

We end up with this relationship from the point of view of Nagios:

```
    [Nagios]
       |
       |
    gateway   ==>    host-gateway_nagios3.cfg
       |
       |
     bb-sw    ==>    switches.cfg (parent is gateway)
       |
       |
     bb-gw    ==>    routers.cfg (parent is sw)
```

Once you have created these files, validate the config and restart nagios (by repeating steps 2a - 2c above) and check the web interface.

Try the "Status Map" option: it gives you a graphical view of the parent-child relationships you have just defined.

4. Create an entry for the classroom NOC

Open the existing pcs.cfg and add a new entry to the end:

```
        # editor pcs.cfg

# Our classroom NOC

define host {
    use         generic-host
    host_name   noc
    alias       Workshop NOC machine
    address     10.10.0.250
    parents     bb-sw
}
```

Question: why is the parent 'bb-sw?'

As usual, validate configuration and restart nagios.

PART III
Configure Service checks for the classroom NOC

--

0. Configuring

Now that we have our hardware configured we can start telling Nagios what services to monitor
on the configured hardware.

The most basic way is to define individual service checks.

1. Edit pcs.cfg and add the following service check near the definition for the 'noc' host

```
    # cd /etc/nagios3/conf.d
    # editor pcs.cfg
```

```
define service {
        host_name                       noc
        service_description             HTTP
        check_command                   check_http
        use                             generic-service
        notification_interval           0
}
```

2. Validate the config, restart, and via the nagios web interface check that
the http service is being monitored (go to "service detail" page)

However, when you are checking many identical services, this approach quickly becomes tedious. For example, you may have many hosts which are running an ssh server and you wish to monitor that service. So you create a single service definition, and link it to a group of hosts.

3. Look inside the file 'services_nagios2.cfg':

```
    # cat services_nagios2.cfg
```

... it should include a section like this:

```
# check that ssh services are running
define service {
        hostgroup_name                  ssh-servers
        service_description             SSH
        check_command                   check_ssh
        use                             generic-service
        notification_interval           0 ; set > 0 if you want to be
        renotified
}
```

4. Open the hostgroups file

    # editor hostgroups_nagios2.cfg

    - Find the hostgroup named "ssh-servers". In the members section of the
    definition change the line:

```
members                 localhost
```

    to

```
members                 localhost,noc
```

Exit and save the file.

5. Verify that your changes are OK:

        # nagios3 -v /etc/nagios3/nagios.cfg

Restart Nagios to see the new service assocation with your host:

        # service nagios3 restart

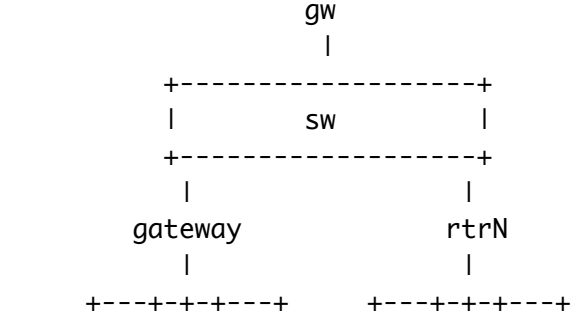
Click on the "Service Detail" link in the Nagios web interface to see your
new entry.

PART IV
Defining more devices

--

1. Create entries for some other routers and PCs in the classroom

Now that we have our routers and switches defined it is quite easy to
create
entries for another group's router and PCs. Think about the parent
relationships:



```
     |   |   |   |     |   |   |   |
    pcA pcB pcC pcD   pcW pcX pcY pcZ
```

The parent of one of you neighbour's PCs is THEIR router. The parent of their router is the switch.

If you are in doubt: DRAW this on paper!

So: pick a group to monitor - this example assumes you decided to pick group 2. Edit routers.cfg to add their router:

```
define host {
    use         generic-host
    host_name   rtr2
    alias       group 2 router
    address     rtr2.ws.nsrc.org
    parents     bb-sw
}
```

And edit pcs.cfg to add their PCs:

```
define host {
    use         generic-host
    host_name   pc5
    alias       pc5 outside interface
    address     pc5.ws.nsrc.org
    parents     rtr2
}
define host {
    use         generic-host
    host_name   pc6
    alias       pc6 outside interface
    address     pc6.ws.nsrc.org
    parents     rtr2
}
define host {
    use         generic-host
    host_name   pc7
    alias       pc7 outside interface
    address     pc7.ws.nsrc.org
    parents     rtr2
}
define host {
    use         generic-host
    host_name   pc8
    alias       pc8 outside interface
    address     pc8.ws.nsrc.org
    parents     rtr2
}
```

You can review the Network Diagram for the class linked off the classroom wiki
main page.

As before, repeat steps 2a-2c to verify your configuration, correct any errors, and activate it.

PART V
Defining more services
---

0. For services, the default normal_check_interval is 5 (minutes) in
   generic-service_nagios2.cfg. You may wish to change this to 1 to speed up
   how quickly service issues are detected, at least in the workshop.

1. Determine what services to define for what devices

   - In this particular class we have:

     routers: running ssh and snmp
     switches: running telnet and possibly ssh as well as snmp
     pcs: All PCs are running ssh and http and should be running snmp
               The NOC is currently running an snmp daemon

     So, let's configure Nagios to check for these services for these
     devices.

2.) Verify that SSH is running on the routers and workshop PCs images

   - In the file services_nagios2.cfg there is already an entry for the SSH
     service check, so you do not need to create this. Instead, you
     simply need to re-define the "ssh-servers" entry in the file
     /etc/nagios3/conf.d/hostgroups_nagios2.cfg. The initial entry in the file
     looked like:

```
# A list of your ssh-accessible servers
define hostgroup {
        hostgroup_name  ssh-servers
                alias           SSH servers
                members         localhost,noc
        }
```

     What do you think you should change? Correct, the "members" line. You should
     add the other group's router and PCs that you defined above. You can
     also add "bb-sw" and "bb-gw" since they are also running SSH servers.

The entry will look something like this:

```
define hostgroup {
        hostgroup_name  ssh-servers
                alias           SSH servers
                members         localhost,rtr2,pc5,pc6,pc7,pc8,bb-sw,bb-gw
        }
```

Note: leave in "localhost" - This is your PC and represents Nagios' network point of view.

The "members" entry will be a long line and might wrap on the screen.

- Once you are done, run the pre-flight check:

```
# nagios3 -v /etc/nagios3/nagios.cfg
```

If everything looks good, then restart Nagios

```
# service nagios3 restart
```

and view your changes in the Nagios web interface.

3.) Check that http is running on all the classroom PCs.

- This is almost identical to the previous exercise. There is already a hostgroup called 'http-servers' in the hostgroups_nagios2.cfg file, so you just need to add the new router and PCs there as members of the http-servers group.

PART VI
Create More Host Groups

---

0. 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 you are monitoring, including the NOC, but not including the routers.

```
# 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,noc,pc5,pc6,pc7,pc8
```

Be sure that the line wraps and is not on two separate lines. Otherwise you will get an error when you go to restart Nagios. Remember that your own PC is "localhost".

- Once you have done this, add in two more host groups, one for routers and one for switches. Call these entries "routers" and "switches". Include the routers and switches you are monitoring.

- 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 hostgroups.

PART VII
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/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.

    # 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 (3600)
        vrml_image       router.png
        statusmap_image  cook/router.gd2
}
```

      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.
      It should be much nicer, with real icons instead of question marks.

PART VIII
Create Service Groups

--

1. Create service groups for ssh for your group's PCs.

   - The idea is to create groups of services for display; one for the
     HTTP servers in your own group, and one for the HTTP servers in the
     other group you are monitoring. To do this create a new file:

   # editor /etc/nagios3/conf.d/servicegroups.cfg

```
# My group (example is for group 1)
define servicegroup {
        servicegroup_name       group1-http
        alias                   group 1 HTTP services
        members                 localhost,HTTP,pc2,HTTP,pc3,HTTP,pc4,HTTP
        }

# Another group (example is for group 2)
define servicegroup {
        servicegroup_name       group2-http
        alias                   group 2 HTTP services
        members                 pc5,HTTP,pc6,HTTP,pc7,HTTP,pc8,HTTP
        }
```

        - Note that "SSH" needs 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 restart Nagios. Now if you
click on
      the Servicegroup menu items in the Nagios web interface you should
see
      this information grouped together.

    - If you like you can also create service groups for SSH between
      the groups.

PART IX
Configure Guest Access to the Nagios Web Interface

--

1. Edit /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.

    # 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

    - 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 may need to clear
      the cookies in your web browser. 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.

OPTIONAL

You can now look at configuring different plugins for monitoring services.

* As opposed to just checking that a web server is running on the classroom PCs, you could also check that the nagios3 service is available, by requesting the /nagios3/ path. This means passing extra options to the check_http plugin.

  For a description of the available options, type this:

  ```
  # /usr/lib/nagios/plugins/check_http
  # /usr/lib/nagios/plugins/check_http --help
  ```

  and of course you can browse the online nagios documentation or google for information on check_http. You can even run the plugin by hand to perform a one-shot service check:

  ```
  # /usr/lib/nagios/plugins/check_http -H localhost -u /nagios3/
  ```

  So the goal is to configure nagios to call check_http in this way.

There is no suitable plugin definition available, so we need to create one.

```
# editor /etc/nagios-plugins/config/local.cfg
define command{
        command_name    check_http_arg
        command_line    /usr/lib/nagios/plugins/check_http -H '$HOSTADDRESS
$' $ARG1$
        }

# editor /etc/nagios3/conf.d/services_nagios2.cfg
define service {
        hostgroup_name                  nagios-servers
        service_description             NAGIOS
        check_command                   check_http_arg!-u /nagios3/
        use                             generic-service
}
```

  and of course you'll need to create a hostgroup called nagios-servers (in hostgroups_nagios2.cfg) to link to this service check.

Once you have done this, check that Nagios warns you about failing authentication (because it's trying to fetch the page without providing the username/password). There's an extra parameter you can pass to check_http_arg to provide that info, see if you can find it.

WARNING: in the tradition of "Debian Knows Best", their definition of the check_http command in /etc/nagios-plugins/config/http.cfg is *not* the same as that recommended in the nagios3 documentation. It is missing $ARG1$, so any parameters to pass to check_http are ignored. So you might think you are monitoring /nagios3/ but actually you are monitoring root!

This is why we had to make a new command definition "check_http_arg". You could make a more specific one like "check_nagios", or you could modify the Ubuntu check_http definition to fit the standard usage.

* Check that SNMP is running on the classroom NOC

- First you will need to add in the appropriate service check for SNMP in the file /etc/nagios3/conf.d/services_nagios2.cfg. This is where Nagios is impressive. There are hundreds, if not thousands, of service checks available via the various Nagios sites on the web. You can see what plugins are installed by Ubuntu in the nagios3 package that we've installed by looking in the following directory:

```
# ls /usr/lib/nagios/plugins
```

As you'll see there is already a check_snmp plugin available to us. If you are interested in the options the plugin takes you can execute the plugin from the command line by typing:

```
# /usr/lib/nagios/plugins/check_snmp
# /usr/lib/nagios/plugins/check_snmp --help
```

to see what options are available, etc. You can use the check_snmp plugin and Nagios to create very complex or specific system checks.

- Now to see all the various service/host checks that have been created using the check_snmp plugin you can look in /etc/nagios-plugins/config/snmp.cfg. You will

see that there are a lot of preconfigured checks using snmp, including:

snmp_load
snmp_cpustats
snmp_procname
snmp_disk
snmp_mem
snmp_swap
snmp_procs
snmp_users
snmp_mem2
snmp_swap2
snmp_mem3
snmp_swap3
snmp_disk2
snmp_tcpopen
snmp_tcpstats
snmp_bgpstate
check_netapp_uptime
check_netapp_cupuload
check_netapp_numdisks
check_compaq_thermalCondition

And, even better, you can create additional service checks quite easily.
For the case of verifying that snmpd (the SNMP service on Linux) is running we
need to ask SNMP a question. If we don't get an answer, then Nagios can assume
that the SNMP service is down on that host. When you use service checks such as
check_http, check_ssh and check_telnet this is what they are doing as well.

- In our case, let's create a new service check and call it "check_system". This
service check will connect with the specified host, use the private community
string we have defined in class and ask a question of snmp on that ask - in this
case we'll ask about the System Description, or the OID "sysDescr.0" -

- To do this start by editing the file /etc/nagios-plugins/config/snmp.cfg:

editor /etc/nagios-plugins/config/snmp.cfg

At the top (or the bottom, your choice) add the following entry to

the file:

```
# 'check_system' command definition
define command{
       command_name    check_system
       command_line    /usr/lib/nagios/plugins/check_snmp -H '$HOSTADDRESS
$' -C
'$ARG1$' -o sysDescr.0
        }
```

You may wish to copy and paste this vs. trying to type this out.

Note that "command_line" is a single line. If you copy and paste in joe the line
may not wrap properly and you may have to manually add the part:

'$ARG1$' -o sysDescr.0

to the end of the line.

- Now you need to edit the file /etc/nagios3/conf.d/services_nagios2.cfg and add
in this service check. We'll run this check against all our servers in the
classroom, or the hostgroup "debian-servers"

- Edit the file /etc/nagios3/conf.d/services_nagios2.cfg

editor /etc/nagios3/conf.d/services_nagios2.cfg

At the bottom of the file add the following definition:

```
# check that snmp is up on all servers
define service {
        hostgroup_name                  snmp-servers
        service_description             SNMP
        check_command                   check_system!xxxxxx
        use                             generic-service
        notification_interval           0 ; set > 0 if you want to be
renotified
}
```

The "xxxxxx" is the community string previously (or to be) defined in class.

Note that we have included our private community string here vs. hard-coding
it in the snmp.cfg file earlier. You must change the "xxxxx" to be the snmp
community string given in class or this check will not work.

- Now we must create the "snmp-servers" group in our hostgroups_nagios2.cfg file.
Edit the file /etc/nagios3/conf.d/hostgroups_nagios2.cfg and go to the end of the
file. Add in the following hostgroup definition:

```
# A list of snmp-enabled devices on which we wish to run the snmp service check
define hostgroup {
           hostgroup_name       snmp-servers
                   alias        snmp servers
                   members      noc
          }
```

- Note that for "members" you could, also, add in the switches and routers for
group 1 and 2. But, the particular item (MIB) we are checking for "sysDescr.0"
may not be available on the switches and/or routers, so the check would then fail.

- Now verify that your changes are correct and restart Nagios.

- If you click on the Service Detail menu choice in web interface you should see
the SNMP check appear for the noc host.

- After we do the SNMP presentation and exercises in class, then you could come
back to this exercise and add in all the classroom PCs to the members list in the
hostgroups_nagios2.cfg file, snmp-servers hostgroup definition. Remember to list
your PC as "localhost".