System monitoring with Open Monitoring Distribution (OMD), hands-on tutorial

Autor: Iñigo Aldazabal Mensa <inigo_aldazabal@ehu.es>

Date: 2014/01/14 - HPC Knowledge Meeting'14, Barcelona

Step-by-step guide for a system monitoring installation and initial configuration using Nagios and Check_MK collection of extensions for Nagios. We will use the pre-packaged Open Monitoring Distribution (OMD) system which bundles both Nagios and Check_MK, as well as many other Nagios extensions into a single, pre-configured package and brings the setup, configuration and maintenance of the monitoring system to a new level of simplicity.

We will use two CentOS virtual machines (*VMs*) to be able follow this tutorial, but the same procedure should be applicable with minimal changes to any of the distributions supported by OMD.

Also basic guidelines are given on how to proceed in order to setup a similar system monitoring computing nodes in a typical HPC cluster.

Contents

1 Required software

- VirtualBox, version 4.3.6, has been used througout the tutorial:

  Virtualization software https://www.virtualbox.org/

  CentOS Virtual Machines from http://virtualboxes.org/images/centos/
- Virtual Machine for the OMD server: CentOS 6.3 with GNOME desktop graphical environment

  http://sourceforge.net/projects/virtualboximage/files/CentOS/6.3/CentOS-6.3-x86.7z
- Virtual Machine to be monitored: CentOS 5.7 base

  http://sourceforge.net/projects/virtualboximage/files/CentOS/5.7/CentOS-5.7-i386.7z
- Open Monitoring Distribution - OMD, version 1.10

  http://omdistro.org/
- Check_MK

  Collection of Nagios extensions / plugins, already integrated in OMD.

  Check_MK monitoring and inventory agent to be installed in the monitored systems http://mathias-kettner.com/check_mk.html, version 1.2.2p3.

2 VirtualBox configuration

The two virtual machines we are using are (see http://virtualboxes.org/images/centos/)

CentOS 5.7 base x86

Size:	(compressed/uncompressed) 173 MBytes / 1.3 GBytes
Link:	http://sourceforge.net/projects/virtualboximage/files/CentOS/5.7/CentOS-5.7-i386.7z
Active user account(s):	(username/password) root/reverse.
Notes:	text mode installed, no graphics

CentOS 6.3 Gnome Desktop x86

Size:	(compressed/uncompressed) 492 MBytes / 2.2 GBytes
Link:	http://sourceforge.net/projects/virtualboximage/files/CentOS/6.3/CentOS-6.3-x86.7z
Active user account(s):	(username/password) root/reverse, centos/reverse.
Notes:	GNOME desktop environement, install from LiveCD; Guest Additions NOT installed.

Now we download the virtual machines, extract them and open the corresponding `.vbox` files from the VirtualBox Manager (`Machine | Add`).

> ***Note***
>
> If we get an error about the disc UUID already being used (eg. because we just copied this virtual machine for another test) we have to change the `.vdi` virtual disk UUID with the command:
>
> ```
> VBoxManage internalcommands sethduuid CentOS-5.7.vdi
> ```

and update the "HardDisk uuid" section in the configuration file `.vbox`.

2.1 Internal network configuration

We want to set up an internal network for the virtual machines to be able to communicate each other.

First we make sure we have an internal network configured in the VirtualBox server (`VirtualBox Manager -> File | Preferences | Network | Host-only Networks`). Make sure you have:

PC VirtualBox Host

IP: 192.168.56.1

We also have to add a `Host-only Adapter` to each virtual machine (`VirtualBox Manager: select the VM -> settings | network | Adapter 2 | Enable + attached to "Host-only Adapter"`).

From the "Advanced" section we write down the network "card" MAC address in order to later set up static IP addresses whithin the internal network. In this case the MACs we have and IPs we will use are:

CentOS 6.3 - OMD monitoring server

MAC: 08:00:27:C1:99:2D
IP: 192.168.56.10

CentOS 5.7 - monitored system

MAC: 08:00:27:42:79:DF
IP: 192.168.56.11

3 Monitoring Server configuration and OMD install

3.1 Network configuration

After booting the virtual machine first enable ssh access as it is disabled by default:

```
chkconfig ssh on
service sshd on
```

Then setup the static IP by creating the `/etc/sysconfig/network-scripts/ifcfg-eth1` file:

```
#/etc/sysconfig/network-scripts/ifcfg-eth1
DEVICE=eth1
BOOTPROTO=none
IPADDR=192.168.56.10
NETMASK=255.255.255.0
ONBOOT=yes
HWADDR=08:00:27:C1:99:2D
DEFROUTE=yes
NAME="eth1"
```

and restart the network:

```
service network restart
```

3.2 email sending configuration

First lets check whether we can already send emails straigth from postfix over port 25:

```
echo "Test mail from postfix" | mail -s "Test Postfix" user@domain
```

If we do not get the message at user@domain check the postfix log at `/var/log/maillog`. In this case it may be necessary to set up a relay host for postfix in `/etc/ postfix/main.cf`. We can eg. use Google SMTP servers for testing.

> ***Note***
>
> Use `tail -f /var/log/maillog` while testing to see the postfix behaviour. In order to check/clean the postfix queue use `mailq` and `postsuper -d ALL` commands.

3.2.1 Postfix relay using Gmail

We follow the guide at http://blog.earth-works.com/2013/05/14/postfix-relay-using-gmail-on-centos/, with a summarized version reproduced here just for completeness.

Install SASL nedded modules:

```
yum install cyrus-sasl-plain
```

Create `/etc/postfix/sasl_passwd` with just one line (adapt to your gmail user data):

```
smtp.gmail.com     GmailUsername:GmailPassword
```

Secure the thing:

```
chown postfix /etc/postfix
postmap hash:/etc/postfix/sasl_passwd
chown root:root /etc/postfix/sasl_passwd*
chmod 640 /etc/postfix/sasl_passwd*
```

Edit the `/etc/postfix/main.cf` configuration file, and add the following lines at the end:

```
#Set the relayhost to the Gmail SMTP server
relayhost = smtp.gmail.com:587

#Set the required TLS options
smtp_tls_security_level = secure
smtp_tls_mandatory_protocols = TLSv1
smtp_tls_mandatory_ciphers = high
smtp_tls_secure_cert_match = nexthop

#Check that this path exists -- these are the certificates used by TLS
smtp_tls_CAfile = /etc/pki/tls/certs/ca-bundle.crt
```

```
#Set the sasl options
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_sasl_security_options = noanonymous
```

Restart postfix service:

```
service postfix restart
```

Test:

```
echo "Test email from postfix with Gmail relay" | mail -s "Gmail-postfix test" user@domain
```

> ***Warning***
>
> Beware of the 500 email/day limits for the regular Google accounts!

3.3 OMD installation

We follow the quickstart CentOS installation instructions straigth from the OMD web page at http://omdistro.org/doc/quickstart_redhat just adapting everything to our CentOS version (6) and architectura (i386).

First install the `epel` repository configuration

```
rpm -Uvh http://download.fedoraproject.org/pub/epel/6/i386/epel-release-6-8.noarch.rpm
```

and then download and install the ~100MB OMD rpm package:

```
wget http://files.omdistro.org/releases/centos_rhel/omd-1.10-rh61-31.i386.rpm
yum install --nogpgcheck omd-1.10-rh61-31.i386.rpm
```

In our case this installs 36 packages and upgrades 4, with a total download size of 24MB.

> ***Note***
>
> We could have instead used the Consol* Labs OMD repository in order to have the latest version available at hand. Setting it up is trivial, just follow the guidelines at https://labs.consol.de/repo/stable.

3.4 OMD initial setup

The `omd` command is used to manage OMD *sites*. OMD sites are completely independent instances of OMD which allow us, if so desired, to have different sites for different purposes as testing, production, upgrading, etc. (see http://mathias-kettner.com/checkmk_install_with_omd.html)

The `omd` command can be executed as the site user to modify just that site, or as root user. As the root user `omd` offers more options such as copying, renaming, disabling or uninstalling sites. Calling `omd` alone provides a list of options with a brief description of them.

3.4.1 OMD site creation and access

To create and start a new OMD `test` site instance just:

```
omd create test
omd start test
```

When creating a new site OMD, amongst other things, creates a new user in the system which will be used to manage this specific site.

In order to manage our site we just `"su -"` to the site/user, in this case:

```
su - test
```

The `test` user home directory is `/omd/sites/test`. Here all the local configurations, caches, performance data, etc. for this site will be kept, specifically in the `tmp`, `var` and `etc` directories (the rest of the directories are symlinked to your OMD version. Again, see http://mathias-kettner.com/checkmk_install_with_omd.html for a detailled description of the file/folder structure and contents.

3.4.2 Multisite web interface access configuration

> ***Note***
>
> Default user/password for the OMD interface is **omdadmin/omd**

Once the test site is up we try to access to it through the *Multisite* web interface from within the own machine first at http://localhost/test. In our case we get a error "OMD: Site not started". This is documented in the OMD FAQ specifically for CentOS and related systems and it has to do to with the selinux configuration. Just run:

```
/usr/sbin/setsebool -P httpd_can_network_connect 1
```

The `-P` option makes the change persistent and the command may take a while to run, even some minutes, so be patient. Once it's done we can access the web interface from the localhost without problems.

If we want to access to the web interface from remote machines (as the VirtualBox physical host in this case) we have to enable the service in the CentOS firewall, activated by default. Just run:

```
/usr/bin/system-config-firewall-tui
```

go to **"Customise"** (<TAB> moves between fields), scroll down the list up to **"WWW (HTTP)"** and enable the service with <SPACE>. Then select **"Close"**, **"OK"** and **"YES"**.

Now you can access the OMD web interface at http://192.168.56.10 eg. from your VirtualBox physical host.

4 Monitored system configuration

After booting the machine (CentOS-5.7) up we just set the static IP and then install the `check_mk` agent.

4.1 Network configuration

As before, we create the `/etc/sysconfig/network-scripts/ifcfg-eth1` file in order to set up a static IP:

```
#/etc/sysconfig/network-scripts/ifcfg-eth1
DEVICE=eth1
BOOTPROTO=none
IPADDR=192.168.56.11
NETMASK=255.255.255.0
ONBOOT=yes
HWADDR=08:00:27:42:79:DF
DEFROUTE=yes
NAME="eth1"
```

and restart the network:

```
service network restart
```

> ***Note***
>
> This is not the case here, but if you are using a firefwall you have to enable accees to the monitoring server at port 6556.

4.2 check_mk agent installation

Download and install the `check_mk` monitoring agent from the check_mk webpage without further complications, the only needed dependence being `xinetd`:

```
wget http://mathias-kettner.com/download/check_mk-agent-1.2.2p3-1.noarch.rpm
wget http://mathias-kettner.com/download/check_mk-agent-logwatch-1.2.2p3-1.noarch.rpm
yum install --nogpgcheck check_mk-agent-1.2.2p3-1.noarch.rpm \
    check_mk-agent-logwatch-1.2.2p3-1.noarch.rpm
```

If desired we can restrict the access to the agent execution in this machine to the OMD monitoring service so we have a more secure setup. In order to do this we just add to the `/etc/xinetc.d/check_mk` file the line:

```
$> vim /etc/xinetc.d/check_mk
...
only_from = 192.168.56.10
...
```

and we reload the `xinetd` daemon configuration:

```
$>/etc/init.d/xinetd reload
```

4.3 Hard disk monitoring with S.M.A.R.T.

When monitoring a physical host we will be interested in monitoring their hard disk health status. Check_mk does not includes S.M.A.R.T. checking by default, but provides a `plugin` that has to be explicitly installed in the remote host.

The plugin is called `smart` and it already is in the OMD server, we just have to copy it over to the desired host:

```
# su - test
# scp ~/share/check_mk/agents/plugins/smart  \
      user@remote-host:/usr/lib/check_mk_agent/plugins/smart
```

If the host has not been inventorized yet in Check_MK, the `smart` check will be present amongst the detected checks when doing it, otherwise you will have to reinventorize it and the new check will appear. Wee will see later how inventorizing hosts works.

5 Basic Check_MK configuration

To setup the basic monitoring system setup we will be using at first *WATO - Check_MK's Web Administrator Tool* through the *Multisite* web interface, both part of the Check_MK ecosystem. This will make our first steps into the Check_MK monitoring world much easier.

We will first setup a new user who will get the test alerts and after this we will add the hosts to be monitored and force some alerts in order to test the notification system.

5.1 User creation

Every user (*contact* in the Nagios nomenclature) belongs to a *contact group*, which are the ones which are really assigned to host and services notifications. In the default OMD/check_MK configuration we have only one contact group, **"all**" or **"Everybody"** (alias), so we will add the new contact to this group ("Contact Groups" section), also making sure that we check the **"Administrator"** role in the "Security" section and that we **"enable notifications"** in the "Notifications" section:

```
( WATO-Configuration | Users & Contacts | New User )
```

We save the changes (**"Save"** in the lower part of the new user creation form) and we are brougth back to the "User & Contacts" main section, where we have a notice about the **"1 Changes"** done. In order to propagate the change to the Check_MK/Nagios configuration click on the **"1 Changes"** button and then on the **"Activate Changes!"** one. We can now see the newly created user in the "Users & Contacts " WATO section and also can checks that the user is a member of the "Everybody" group in the "Contact Goups" section.

5.2 Integration (inventory) of the new *host* to be monitored

In order to add/inventorize a new host (in which of course we already have installed the check_mk agent), we go to:

```
( WATO-Configuration | Hosts & Folders | New host )
```

and there we just add the **"Hostname"** (CentOS-5.7), **"IP"** if needed (192.168.56.11 in this case), **"Permissions" -> "Everybody"** and **"Alias"** (if desired). Clicking on **"Save & go to Services"** brings us

to the autodetected host services list, where we can choose to ignore some of the automatically detected checks. We then **"Save manual check configuration"** and as we did before we **"Activate Changes!"**.

Going to the main web interface page (Check_MK logo in the upper left or `( Views | Dashboards | Main Overview )` we see that we have one host and 19 services monitored.

> ***Note***
>
> It is convenient to set up the own monitoring server to monitor itself. For this we just install the check_mk agent in the server and add the host *"localhost"* in WATO. Do it!

5.3 Reinventoring

If we add new checks to a host through check_mk plugins, legacy nagios checks, NRPE nagios checks, etc., we can make Check_MK to scan this host for new, not inventorized services. Just go to `( WATO-Configuration | Hosts & Folders )`, click on the desired host and then select **"Services"** and **"Full Scan"**. New services will be detected and you can enable them at will, as well as disable existing checks if wanted.

> ***Note***
>
> When reinventoring a host all previously inventorized checks, performance data, graphs, etc. are kept.

5.4 email notification test

In order to test email notifications go to a host `( Views | Hosts | All hosts )` and click on a service name. In the service information page click on the hammer icon in order to run commands over this service. Then go to **"Various Commands" -> "Fake check results"** and eg. click **"Critical**". Confirm the action and see eg. in the `( Dashboard | Main Overview )` the service being Critical for a while and the notifications being sent. Check you email for the Critical State notification and the Recovery one a minute later, when the service comes back to normal state!

6 Advanced Check_MK configuration

Automatic inventoring and Check_MK managing with WATO is OK if we add machines one by one or want to monitor certain very specific machines: a few workstations, some storage server(s), a HPC cluster head node, etc. But, what if we want to monitor some HPC cluster computing nodes? Should we add say 100 nodes one by one? Not indeed.

WATO is really not more than a front end that does part of the job for us, but in the background, as you may have suspected, everything is in fact done trough configuration files with a very clean, documented interface.

Check_MK configuration files lay under `~/etc/check_mk`, being `~` the home of the user corresponding to the OMD *site*. Check_MK reads the `.mk` configuration files there and generates the corresponding nagios configuration files. When requested to do it, of course!

Check_MK first reads the `~/etc/check_mk/main.mk` file, and then all the `.mk` files under `~/etc/check_mk/conf.d`. The configuration files syntax is plain python systax.

See http://mathias-kettner.com/checkmk_configfiles.html for information about configuration files reading and parsing and http://mathias-kettner.com/checkmk_configvars.html for a detailled description of the configuration variables and how to use them.

The typicall steps when working with configuration files consist are:

1. add some hosts or modify some settings in some of the configuration files,
2. optionally do a reinventory if needed, and
3. recompile nagios configuration and reload/restart nagios service.

The command used to do all this is `check_mk` or its alias `cmk`. Calling just `cmk` provides a sumary of the options and a sparse summary of its behaviour. See http://mathias-kettner.com/checkmk_calling.html

6.1 Manually configure VirtualBox host monitoring

As a very simple example we will set up the VirtualBox host itself for monitoring, all from the command line. We will call this host *"VB-host"*. See next section for a more complex example.

First, in the monitoring server, we go into the OMD `test` user/site:

```
su - test
```

and there we create the `vb-host.mk` file in `~/etc/check_mk/conf.d` with the following content:

```
# ~/etc/check_mk/conf.d/vb-host.mk

all_hosts += [
    'VB-host',
]

ipaddresses['VB-host'] = '192.168.56.1'
```

After installing the check_mk agent in the *"VB-host"* we manually inventorize it and update the Nagios core with the `check_mk` command:

```
check_mk -I VB-host
check_mk -R
```

And we are done! The new host is added to the default `all` (`Everyone`) contact group and we can see all detected services in the multisite interface.

6.2 Two node HPC cluster configuration

Let's see a slightly more complex example of a real configuration file for a two nodes test rocks cluster. It should be work the same for a more general HPC cluster.

We have installed OMD in the head node, inventorized the own head node with WATO (both just as described in the previous sections), and want to add the compute nodes using a configuration file so that we can script the process for any number of them. Of course we have also installed the check_mk agent and the "smart" check_mk plugin in the compute nodes (you can include them in you nodes master image, post install it with pdsh/pdcp, set in up in your cfengine/puppet/salt or whatever configuration management system you may be using, etc.).

6.2.1 Configuration file

So lets write the configuration file `~/etc/check_mk/conf.d/conpute.mk` and dissect it:

```
 1 # ~/etc/check_mk/conf.d/compute.mk
 2 # Configuration for test rocks cluster compute nodes
 3
 4 all_hosts = all_hosts + [
 5     'compute-0-0|compute',
 6     'compute-0-1|compute',
 7 ]
 8
 9 ignored_services += [
10      ( [ "compute" ], ALL_HOSTS, [ "fs_/var" ] ),
11  ]
12
13 ignored_checks += [
14     ( [ "postfix_mailq" ], [ "compute" ], ALL_HOSTS  ),
15 ]
16
17 host_contactgroups += [
18     ( "Everybody", [ "compute" ], ALL_HOSTS ),
19 ]
20
21 check_parameters += [
22     ( (45, 55), [ 'compute' ], ALL_HOSTS, [ "Temperature SMART" ] ),
23 ]
```

Hosts setup

We fist setup the hosts. Note how we *add* the new hosts to the `all_hosts` variable:

```
4 all_hosts = all_hosts + [
5     'compute-0-0|compute',
6     'compute-0-1|compute',
7 ]
```

The first part of every field is the hostname and the second the check_mk host *tag*, `compute` in this case (see http://mathias-kettner.com/checkmk_hosttags.html). The last comma (,) is superfluous, but python allows it and makes the scripting much easier. This is the only part we would have to script in order to include our 10's or 100's of nodes in a general case.

Ignored services

If we want to ignore some services, we add them up to the `ignored_services` variable:

```
 9 ignored_services += [
10      ( [ "compute" ], ALL_HOSTS, ["fs_/var"] ),
11  ]
```

You can get the exact name of the service you want to ignore from the own service name as it is shown in the multisite web interface or get it from inspecting the `cmk -D` command output. In this case all services whose name *begings* with `fs_/var` on hosts with the host tag `compute` will be ignored. See http://mathias-kettner.com/checkmk_inventory.html.

Ignored checks

We can also ignore checks, in this case the `postfix_mailq` check:

```
13 ignored_checks += [
14     ( [ "postfix_mailq" ], [ "compute" ], ALL_HOSTS  ),
15 ]
```

See also http://mathias-kettner.com/checkmk_inventory.html.

Contact groups

We also add the new hosts to a contact group, in this case the OMD default `Everybody` group as an example:

```
17 host_contactgroups += [
18     ( "Everybody", [ "compute" ], ALL_HOSTS ),
19 ]
```

Special check parameters

And finally we adjust some of the checks default warning/critical levels:

```
21 check_parameters += [
22     ( (45, 55), [ 'compute' ], ALL_HOSTS, [ "Temperature SMART" ] ),
23 ]
```

In this case the default smart check temperature levels are too low for our system (35C and 40C, see `cmk --man smart.temp`) so we raise them a bit. All services called "Temperature SMART<whatever>" will have as new levels (45,55). See http://mathias-kettner.com/checkmk_check_parameters.html.

Host groups

You can also, although we have not done it in this example, create a Nagios *host group* in WATO **Host Groups** section and add these hosts to the group:

```
host_groups += [
    ( 'computenodes', [ 'compute' ], ALL_HOSTS ),
]
```

You can find a list of all the configuration variables that mey be used at http://mathias-kettner.com/checkmk_configvars.html.

6.2.2 Inventoring and Nagios configuration update

After we have written down the new configuration file for the compute nodes we have to first scan (inventorize) the new hosts for services and then propagate the new hosts configuration and the newly found services to Nagios.

So we (re)inventorize all hosts with the tag `compute` so that Check_MK finds the new hosts and the corresponding services:

```
check_mk -II @compute
```

In order to propagate the updated configuration to Nagios and restart the monitoring core we just:

```
check_mk -R
```

And ready! We have the compute nodes slightly more under control :-)

6.2.3 Post-configuration

It quite likely that after a very quick initial setup some adjustments have to be done on the system in order to reflect your real life situation. Eg. in a computer cluster hard disk temperatures will likely be more that 35C, CPU loads will be very high and so on. So expect a bit of playing specially with the ignored checks/services and the check parameters options.

After this adjustment period you will only get notified when something undesired is really going on in your cluster (or in your data center, or in your backups, or...).

6.3 Adding custom checks

Let see how to add a custom check to the monitorized machine: the bread and butter of nagios!

Just go to the machine, eg. `VB-host`, and add a script/program to `/usr/lib/check_mk_agent/local/` generating a check_mk plugin output, very similar to the nagios ones (see http://mathias-kettner.com/checkmk_localchecks.html):

```
#!/bin/bash
# /usr/lib/check_mk_agent/local/check_filecount_tmp
# Counts number of files in /tmp. Harcoded levels w=50, c=100.

count=$(ls -1 /tmp | wc --lines)

if [ $count -lt 50 ] ; then
    echo "0 filecount_tmp /tmp=$count;50;100 OK - $count files in /tmp "
    exit 0
elif [ $count -lt 100 ] ; then
    echo "1 filecount_tmp /tmp=$count;50;100 WARNING - $count files in /tmp "
    exit 1
elif [ $count -ge 100 ] ; then
    echo "2 filecount_tmp /tmp=$count;50;100 CRITICAL - $count files in /tmp"
    exit 2
else
    echo "3 filecount_tmp /tmp=$count;50;100 UNKNOWN - $count files in /tmp"
    exit 3
fi
```

Make it executable:

```
chmod a+x /usr/lib/check_mk_agent/local/check_filecount_tmp
```

test:

```
# ./check_filecount_tmp
0 filecount_tmp /tmp=15;50;100 OK - 15 files in /tmp
```

and reinventorize the checks in the monitoring server:

```
su - test
cmk -I VB-host
cmk -R
```

And done!

7 References

Virtual Machines

- Oracle VirtualBox, multiplatform virtualization system: https://www.virtualbox.org/
- CentOS preinstalled VirtualBox virtual machines: http://virtualboxes.org/images/centos/

Nagios

- Web: http://www.nagios.org/
- Official Documentation: http://nagios.sourceforge.net/docs/nagioscore/3/en/toc.html
- Nagios Exchange: Nagios extension and checks open repository http://exchange.nagios.org/
- *"Building a Monitoring Infrastructure With Nagios"*, David Josephsen, Prentice Hall 2007

Check_MK

- Web: http://mathias-kettner.com/check_mk.html
- Official Documentation: http://mathias-kettner.com/checkmk.html

OMD

- Web: http://omdistro.org/