

2. Icinga Configuration

[Page 1](#) [Page 2](#) [Page 3](#) [Page 4](#)

The main Icinga configuration file is `/etc/icinga/icinga.cfg`, additional configurations are stored in `/etc/icinga/commands.cfg` and `/etc/icinga/resource.cfg`. Usually the default configuration is ok, so you don't have to change these files.

The first thing you should change is the contact details in `/etc/icinga/objects/contacts_icinga.cfg` so that notifications are sent to the correct email address:

```
vi /etc/icinga/objects/contacts_icinga.cfg
```

```
[...]
define contact{
    contact_name          root
    alias                 Falko Timme
    service_notification_period 24x7
    host_notification_period  24x7
    service_notification_options w,u,c,r
    host_notification_options  d,r
    service_notification_commands notify-service-by-email
    host_notification_commands  notify-host-by-email
    email                   me@myself.com
}
[...]
```

The service checks for localhost are defined in `/etc/icinga/objects/localhost_icinga.cfg` - take a look at that file:

```
cat /etc/icinga/objects/localhost_icinga.cfg
```

```
# A simple configuration file for monitoring the local host
# This can serve as an example for configuring other servers;
# Custom services specific to this host are added here, but services
# defined in icinga-common_services.cfg may also apply.
#

define host{
    use          generic-host          ; Name of host
    template to use
    host_name    localhost
    alias        localhost
    address      127.0.0.1
}

# Define a service to check the disk space of the root partition
```

```
# on the local machine. Warning if <20% free, critical if
# <10% free space on partition.
```

```
define service{
    use                generic-service        ; Name of
service template to use
    host_name          localhost
    service_description Disk Space
    check_command       check_all_disks!20%!10%
}
```

```
# Define a service to check the number of currently logged in
# users on the local machine. Warning if> 20 users, critical
# if> 50 users.
```

```
define service{
    use                generic-service        ; Name of
service template to use
    host_name          localhost
    service_description Current Users
    check_command       check_users!20!50
}
```

```
# Define a service to check the number of currently running procs
# on the local machine. Warning if> 250 processes, critical if
#> 400 processes.
```

```
define service{
    use                generic-service        ; Name of
service template to use
    host_name          localhost
    service_description Total Processes
    check_command       check_procs!250!400
}
```

```
# Define a service to check the load on the local machine.
```

```
define service{
    use                generic-service        ; Name of
service template to use
    host_name          localhost
    service_description Current Load
    check_command
check_load!5.0!4.0!3.0!10.0!6.0!4.0
}
```

As you see, the check_all_disks command is defined as /usr/lib/nagios/plugins/check_disk -w '\$ARG1\$' -c '\$ARG2\$' -e. If you take a look at the /etc/icinga/objects/localhost_icinga.cfg file again, you see that we have the line check command check_all_disks!20%!10% in it. Icinga allows us to pass command line arguments to service checks by separating them with an exclamation mark (!), so check_all_disks!20%!10% means we pass 20% as the first command line argument and 10% as the second command line argument to the /usr/lib/nagios/plugins/check_disk -w '\$ARG1\$' -c '\$ARG2\$' -e command so that it finally translates to /usr/lib/nagios/plugins/check_disk -w '20%' -c '10%' -e.

'check_all_disks' command definition
 define command{
 command_name check_all_disks
 command_line /usr/lib/nagios/plugins/check_disk -w '\$ARG1\$' -c '\$ARG2\$' -e
 }
 To find out what command line arguments a plugin can take, call that plugin with the -help switch. For example, to find out how the check_disk plugin can be used, run

```
/usr/lib/nagios/plugins/check_disk --help
```

'ssh_disk' command definition
 define command{
 command_name ssh_disk
 command_line /usr/lib/nagios/plugins/check_by_ssh -H
 }
 With this knowledge you can modify the service checks in /etc/icinga/objects/localhost_icinga.cfg to your likings, and you can add/modify plugin configurations in the /etc/nagios-plugins/config directory.

'\$HOSTADDRESS\$' -C GESHI_QUOT /usr/lib/nagios/plugins/check_disk
 Now let's assume we want to add a service check for MySQL, we first take a look at the appropriate plugin configuration:

```
cat /etc/nagios-plugins/config/mysql.cfg
#####
# use these c
# 'check_mysql' command definition
define command{
  command_name check_mysql
  command_line /usr/lib/nagios/plugins/check_mysql -H '$HOSTADDRESS$'
}
#####
# 'check_mysql_cmdlinecred' command definition
define command{
  command_name check_mysql_cmdlinecred
  command_line /usr/lib/nagios/plugins/check_mysql -H '$HOSTADDRESS$'
}
# 'check_mysql_database' command definition
define command{
  command_name check_mysql_database
  command_line /usr/lib/nagios/plugins/check_mysql -d '$ARG3$' -H '$HOSTADDRESS$' -u '$ARG1$' -p '$ARG2$'
}
```

The command I want to use is check_mysql_cmdlinecred - this takes a MySQL username and a password as arguments (besides the host address which is taken from the host_name parameter of the service check definition. I want to use the MySQL user nagios with the password howtoforge here, so I add the following section to /etc/icinga/objects/localhost_icinga.cfg :

```
vi /etc/icinga/objects/localhost_icinga.cfg
```

```
[...]
define service{
    use                generic-service
    host_name          localhost
    service_description MySQL
    check_command
check_mysql_cmdlinecred!nagios!howtoforge
}
```

Before we restart Icinga, we must create the MySQL user nagios with the password howtoforge :

```
mysql -u root -p

GRANT USAGE ON *.* TO nagios@localhost IDENTIFIED BY 'howtoforge';
GRANT USAGE ON *.* TO nagios@localhost.localdomain IDENTIFIED BY 'howtoforge';
FLUSH PRIVILEGES;
```

quit;

(The USAGE privilege is a synonym for 'no privileges', i.e., the nagios user can connect to MySQL, but not alter or read any data.)

Now we restart Icinga so that our changes take effect:

```
/etc/init.d/icinga restart
```

If you check localhost 's services in the Icinga web interface now, you should see that a check for MySQL has been added:



http://static.howtoforge.com/images/icinga_monitoring_ubuntu_11.10/big/4.png Likewise, we can add checks for SMTP, POP3, and IMAP - these are just connection checks, so we don't need any arguments:

```
vi /etc/icinga/objects/localhost_icinga.cfg
```

```
[...]
define service{
    use                generic-service
    host_name          localhost
    service_description SMTP
    check_command
check_smtp
}
define service{
    use                generic-service
    host_name          localhost
    service_description POP3
    check_command
check_pop
}
```

```
}  
define service{  
    use                generic-service  
    host_name          localhost  
    service_description IMAP  
    check_command       check_imap  
}
```

Restart Icinga...

/etc/init.d/icinga restart

... and a few moments later you should see the new checks in the Icinga web interface:



http://static.howtoforge.com/images/icinga_monitoring_ubuntu_11.10/big/5.png You might have noticed the SSH and HTTP checks for localhost which are not defined in /etc/icinga/objects/localhost_icinga.cfg . These are defined in hostgroup s in the /etc/icinga/objects/hostgroups_icinga.cfg file. A hostgroup allows us to run a service check for multiple servers and define it only once. Take a look at that file:

cat /etc/icinga/objects/hostgroups_icinga.cfg

```
# Some generic hostgroup definitions  
  
# A simple wildcard hostgroup  
define hostgroup {  
    hostgroup_name  all  
        alias      All Servers  
        members    *  
}  
  
# A list of your Debian GNU/Linux servers  
define hostgroup {  
    hostgroup_name  debian-servers  
        alias      Debian GNU/Linux Servers  
        members    localhost  
}  
  
# A list of your web servers  
define hostgroup {  
    hostgroup_name  http-servers  
        alias      HTTP servers  
        members    localhost  
}  
  
# A list of your ssh-accessible servers  
define hostgroup {
```

```
    hostgroup_name  ssh-servers
    alias           SSH servers
    members         localhost
}
```

As you see, we have a hostgroup called http-servers and a hostgroup called ssh-servers , and localhost is a member of each of these groups. The service checks for the hostgroup s are defined in /etc/icinga/objects/services_icinga.cfg . This file contains service checks and refers to the hostgroup s to which these checks should be applied by using the hostgroup_name parameter:

```
cat /etc/icinga/objects/services_icinga.cfg
```

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

# 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
}
```

As you see, the SSH and HTTP service checks are defined here.

[Page 1](#) [Page 2](#) [Page 3](#) [Page 4](#)

From:

<http://vlired.cu/dokuwiki/> - **ICT Network Project**

Permanent link:

http://vlired.cu/dokuwiki/icinga:icinga_configuration_page2

Last update: **2015/06/29 12:10**

