*By Falko Timme*
Published: 2007-05-13 18:21

# Server Monitoring With munin And monit On Debian Etch

Version 1.0
Author: Falko Timme <ft [at] falkotimme [dot] com>
Last edited 04/24/2007

In this article I will describe how you can monitor your Debian Etch server with munin and monit. munin produces nifty little graphics about nearly every aspect of your server (load average, memory usage, CPU usage, MySQL throughput, eth0 traffic, etc.) without much configuration, whereas monit checks the availability of services like Apache, MySQL, Postfix and takes the appropriate action such as a restart if it finds a service is not behaving as expected. The combination of the two gives you full monitoring: graphics that lets you recognize current or upcoming problems (like "We need a bigger server soon, our load average is increasing rapidly."), and a watchdog that ensures the availability of the monitored services.

Although munin lets you monitor more than one server, we will only discuss the monitoring of the system where it is installed here.

This tutorial was written for Debian Etch, but the configuration should apply to other distributions with little changes as well.

I want to say first that this is not the only way of setting up such a system. There are many ways of achieving this goal but this is the way I take. I do not issue any guarantee that this will work for you!

## 1 Preliminary Note

Our system's hostname is *server1.example.com*, and we have a web site *www.example.com* on it with the document root */var/www/www.example.com/web*.

## 2 Install And Configure munin

To install munin on Debian Etch, we do this:

```
apt-get install munin munin-node
```

Next, we must edit the munin configuration file */etc/munin/munin.conf*. We want munin to put its output into the directory */var/www/www.example.com/web/monitoring*, therefore we change the value of *htmldir*, and we want it to use the name *server1.example.com* instead of *localhost.localdomain* in the HTML output, therefore we replace *localhost.localdomain* with *server1.example.com*. Without the comments, the changed file looks like this:

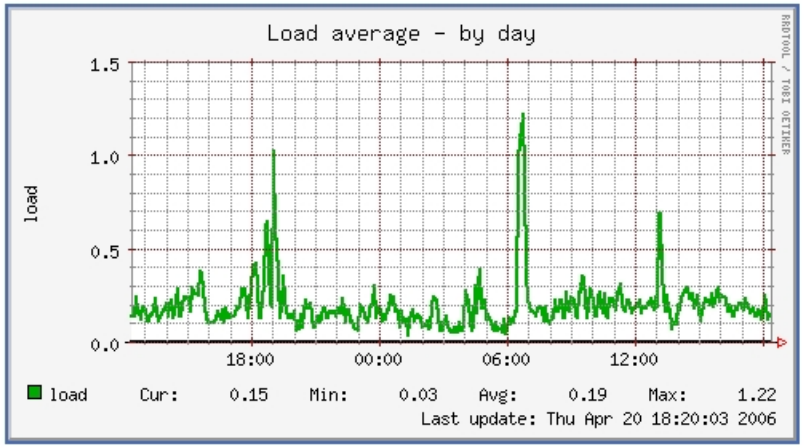
```
vi /etc/munin/munin.conf
```

```
dbdir   /var/lib/munin
htmldir /var/www/www.example.com/web/monitoring
logdir  /var/log/munin
rundir  /var/run/munin

tmpldir /etc/munin/templates

[server1.example.com]
    address 127.0.0.1
    use_node_name yes
```

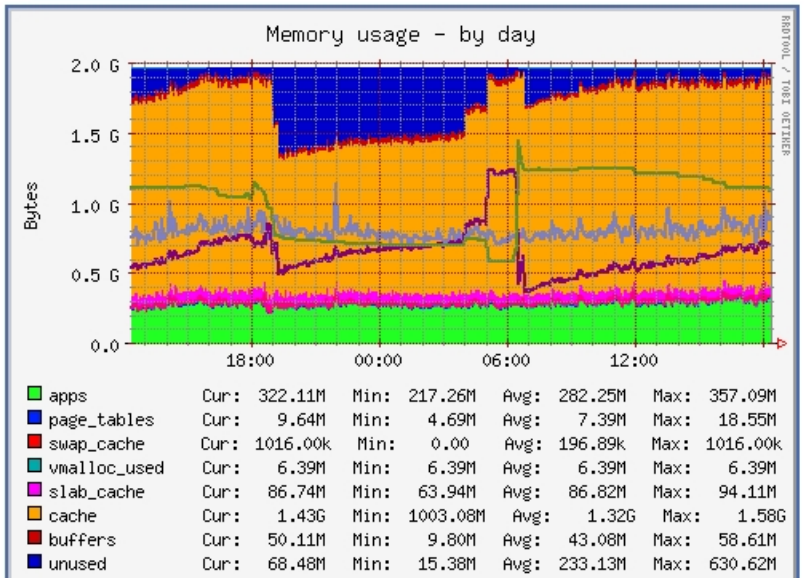
Next we create the directory */var/www/www.example.com/web/monitoring* and change its ownership to the user and group *munin*, otherwise munin cannot place its output in that directory. Then we restart munin:

```
mkdir -p /var/www/www.example.com/web/monitoring

chown munin:munin /var/www/www.example.com/web/monitoring

/etc/init.d/munin-node restart
```

Now wait a few minutes so that munin can produce its first output, and then go to *http://www.example.com/monitoring/* in your browser, and you see the first statistics. After a few days this could look like this:




:: Memory usage




(This is just a small excerpt of the many graphics that munin produces...)

## 3 Password-Protect The munin Output Directory (Optional)

Now it is a good idea to password-protect the directory */var/www/www.example.com/web/monitoring* unless you want everybody to be able to see every little statistic about your server.

To do this, we create an *.htaccess* file in */var/www/www.example.com/web/monitoring*:

```
vi /var/www/www.example.com/web/monitoring/.htaccess
```

```
AuthType Basic
AuthName "Members Only"
AuthUserFile /var/www/www.example.com/.htpasswd
<limit GET PUT POST>
require valid-user
</limit>
```

Then we must create the password file */var/www/www.example.com/.htpasswd*. We want to log in with the username *admin*, so we do this:

```
htpasswd -c /var/www/www.example.com/.htpasswd admin
```

Enter a password for *admin*, and you're done!

# 4 Install And Configure monit

To install monit, we do this:

```
apt-get install monit
```

Now we must edit */etc/monit/monitrc*. The default */etc/monit/monitrc* has lots of examples, and you can find more configuration examples on

http://www.tildeslash.com/monit/doc/examples.php. However, in my case I want to monitor *proftpd*, *sshd*, *mysql*, *apache*, and *postfix*, I want to enable the monit web interface on port 2812, I want a https web interface, I want to log in to the web interface with the username *admin* and the password *test*, and I want monit to send email alerts to *root@localhost*, so my file looks like this:

```
cp /etc/monit/monitrc /etc/monit/monitrc_orig

cat /dev/null > /etc/monit/monitrc

vi /etc/monit/monitrc
```

```
set daemon  60
set logfile syslog facility log_daemon
set mailserver localhost
set mail-format { from: monit@server1.example.com }
set alert root@localhost
set httpd port 2812 and
    SSL ENABLE
    PEMFILE  /var/certs/monit.pem
    allow admin:test

check process proftpd with pidfile /var/run/proftpd.pid
  start program = "/etc/init.d/proftpd start"
  stop program  = "/etc/init.d/proftpd stop"
  if failed port 21 protocol ftp then restart
  if 5 restarts within 5 cycles then timeout

check process sshd with pidfile /var/run/sshd.pid
  start program  "/etc/init.d/ssh start"
  stop program  "/etc/init.d/ssh stop"
  if failed port 22 protocol ssh then restart
  if 5 restarts within 5 cycles then timeout
```

```
check process mysql with pidfile /var/run/mysqld/mysqld.pid
   group database
   start program = "/etc/init.d/mysql start"
   stop program = "/etc/init.d/mysql stop"
   if failed host 127.0.0.1 port 3306 then restart
   if 5 restarts within 5 cycles then timeout

check process apache with pidfile /var/run/apache2.pid
   group www
   start program = "/etc/init.d/apache2 start"
   stop program  = "/etc/init.d/apache2 stop"
   if failed host www.example.com port 80 protocol http
      and request "/monit/token" then restart
   if cpu is greater than 60% for 2 cycles then alert
   if cpu > 80% for 5 cycles then restart
   if totalmem > 500 MB for 5 cycles then restart
   if children > 250 then restart
   if loadavg(5min) greater than 10 for 8 cycles then stop
   if 3 restarts within 5 cycles then timeout

check process postfix with pidfile /var/spool/postfix/pid/master.pid
   group mail
   start program = "/etc/init.d/postfix start"
   stop  program = "/etc/init.d/postfix stop"
   if failed port 25 protocol smtp then restart
   if 5 restarts within 5 cycles then timeout
```

(Please make sure that you check processes only that really exist on your server - otherwise monit won't start. I.e., if you tell monit to check Postfix, but Postfix isn't installed on the system, monit won't start.)

The configuration file is pretty self-explaining; if you are unsure about an option, take a look at the monit documentation: **http://www.tildeslash.com/monit/doc/manual.php**

In the *apache* part of the monit configuration you find this:

```
if failed host www.example.com port 80 protocol http
   and request "/monit/token" then restart
```

which means that monit tries to connect to *www.example.com* on port 80 and tries to access the file */monit/token* which is */var/www/www.example.com/web/monit/token* because our web site's document root is */var/www/www.example.com/web*. If monit doesn't succeed it means Apache isn't running, and monit is going to restart it. Now we must create the file */var/www/www.example.com/web/monit/token* and write some random string into it:

```
mkdir /var/www/www.example.com/web/monit

echo "hello" > /var/www/www.example.com/web/monit/token
```

Next we create the pem cert (*/var/certs/monit.pem*) we need for the SSL-encrypted monit web interface:

```
mkdir /var/certs

cd /var/certs
```

We need an OpenSSL configuration file to create our certificate. It can look like this:

```
vi /var/certs/monit.cnf
```

```
# create RSA certs - Server

RANDFILE = ./openssl.rnd

[ req ]
default_bits = 1024
encrypt_key = yes
distinguished_name = req_dn
x509_extensions = cert_type

[ req_dn ]
countryName = Country Name (2 letter code)
countryName_default = MO

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Monitoria

localityName                    = Locality Name (eg, city)
localityName_default            = Monittown

organizationName                = Organization Name (eg, company)
organizationName_default        = Monit Inc.

organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default  = Dept. of Monitoring Technologies

commonName                      = Common Name (FQDN of your server)
commonName_default              = server.monit.mo

emailAddress                    = Email Address
emailAddress_default            = root@monit.mo

[ cert_type ]
```

```
nsCertType = server
```

Now we create the certificate like this:

```
openssl req -new -x509 -days 365 -nodes -config ./monit.cnf -out /var/certs/monit.pem -keyout /var/certs/monit.pem
```

```
openssl gendh 512 >> /var/certs/monit.pem
```

```
openssl x509 -subject -dates -fingerprint -noout -in /var/certs/monit.pem
```

```
chmod 700 /var/certs/monit.pem
```

Afterwards we edit */etc/default/monit* to enable the monit daemon. Change *startup* to *1* and set *CHECK_INTERVALS* to the interval in seconds that you would like monit to check your system. I choose 60 (seconds) so my file looks like this:

```
vi /etc/default/monit
```

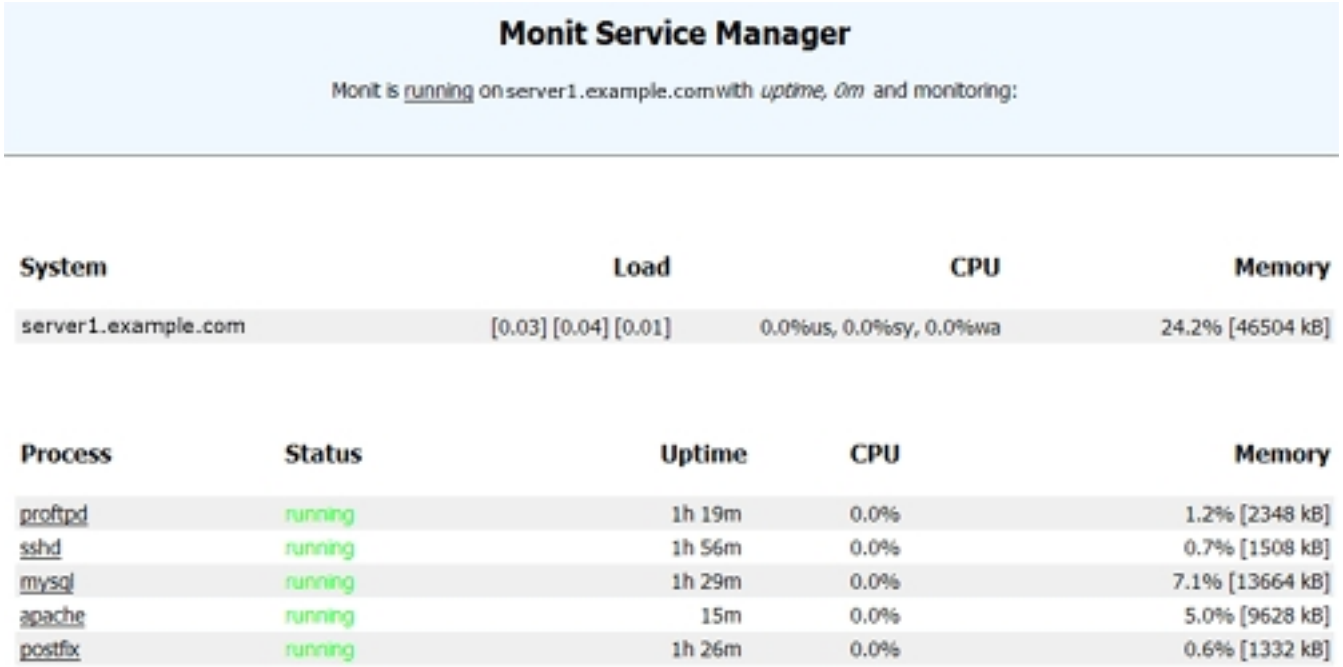
```
# Defaults for monit initscript
# sourced by /etc/init.d/monit
# installed at /etc/default/monit by maintainer scripts
# Fredrik Steen <stone@debian.org>

# You must set this variable to for monit to start
startup=1
```

```
# To change the intervals which monit should run uncomment
# and change this variable.
CHECK_INTERVALS=60
```

Finally, we can start monit:

```
/etc/init.d/monit start
```

Now point your browser to *https://www.example.com:2812/* (make sure port 2812 isn't blocked by your firewall), log in with *admin* and *test*, and you should see the monit web interface. It should look like this:



**Monit Service Manager**

Monit is running on server1.example.com with *uptime, 0m* and monitoring:

| System | Load | CPU | Memory |
|---|---|---|---|
| server1.example.com | [0.03] [0.04] [0.01] | 0.0%us, 0.0%sy, 0.0%wa | 24.2% [46504 kB] |

| Process | Status | Uptime | CPU | Memory |
|---|---|---|---|---|
| proftpd | running | 1h 19m | 0.0% | 1.2% [2348 kB] |
| sshd | running | 1h 56m | 0.0% | 0.7% [1508 kB] |
| mysql | running | 1h 29m | 0.0% | 7.1% [13664 kB] |
| apache | running | 15m | 0.0% | 5.0% [9628 kB] |
| postfix | running | 1h 26m | 0.0% | 0.6% [1332 kB] |

(Main Screen)

**Process status**

| Parameter | Value |
|---|---|
| Name | apache |
| Pid file | /var/run/apache2.pid |
| Status | running |
| Group | www |
| Monitoring mode | active |
| Monitoring status | monitored |
| Start program | /etc/init.d/apache2 start |
| Stop program | /etc/init.d/apache2 stop |
| Check service | every 1 cycle |
| Timeout | If 3 restart within 5 cycles then unmonitor else if recovered then alert |
| Data collected | Fri Oct 21 16:35:16 2005 |
| Port Response time | 0.008s to 127.0.0.1:80/monit/token [HTTP] |
| Process id | 7561 |
| Parent process id | 1 |
| Process uptime | 18m |
| CPU usage | 0.0% |
| Memory usage | 5.0% [9628kB] |
| Children | 8 |
| Total CPU usage (incl. children) | 0.0% |
| Total memory usage (incl. children) | 46.8% [89976kB] |
| Port | If failed 127.0.0.1:80/monit/token [HTTP] with timeout 5 seconds ther |
| Pid | If changed then alert |
| Ppid | If changed then alert |

(Apache Status Page)

Depending on your configuration in `/etc/monit/monitrc` monit will restart your services if they fail and send notification emails if process IDs of

services change, etc.

Have fun!

## 5 Links

- munin: **http://munin.projects.linpro.no**
- monit: **http://www.tildeslash.com/monit/index.php**