Munin

Monitoring and Notifications

James Pinkster

31 Oct 2016 / MLUG

Outline

Introduction

Introduction

- Munin is a monitoring tool written in perl that graphs all the resource information it collects and visualizes this information via a web interface.
- It uses a master/node architecture. The master periodically connects to the nodes retrieving resource data via plugins enabled on the nodes.
- The master stores the retrieved information in time series database files using RRDTool.

Munin Master

Munin Master - Installation

Arch Linux (2.0.26):

```
# pacman -S munin nginx
```

Debian Family (2.0.25):

```
# apt-get install munin nginx
```

Munin Master - Image Generation

Graph image generation uses one of two strategies:

- **Static** (cron) - images generated every 5 minutes when `munin-cron` runs
- Dynamic (cgi) - images are generated upon page request to the web server running on the master.

Munin Master - Configure Static Image Generation

/etc/munin/munin.conf

```
dbdir /var/lib/munin
htmldir /usr/share/munin/www
logdir /var/log/munin
rundir /run/munin
tmpldir /etc/munin/templates
staticdir /etc/munin/static
includedir /etc/munin/munin-conf.d

graph_strategy cron
html_strategy cron

[xmen.jp-it.net.au;profx]
    address 127.0.0.1

[xmen.jp-it.net.au;vagrant]
    address 192.168.2.200
```

Munin Master - Permissions

```
# chown munin:  /usr/share/munin/www
```

Munin Master - Cron

The `munin-cron` command is scheduled to run every 5 minutes[1] and handles running the perl scripts `munin-update`, `munin-limits`, `munin-html`,and `munin-graph` in `/usr/lib/munin`.

There are two options for scheduling the `munin-cron` command:

- crontab - The default on Debian. Arch package includes a cron entry file.
- **Systemd Timer** - Uses a `.service` and `.timer` file.

[1] The RRD files are constructed to create 5 minutes averages.

Munin Master - Cron

/etc/systemd/system/munin-cron.service

```
[Unit]
Description=Munin Master Cron Service
After=network.target

[Service]
User=munin
ExecStart=/usr/bin/munin-cron
```

Munin Master - Cron

/etc/systemd/system/munin-cron.timer

```
[Unit]
Description=Munin Master Cron runs every five minutes

[Timer]
OnCalendar=*-*-* *:00/5:00

[Install]
WantedBy=multi-user.target
```

Munin Master - Cron

Reload systemd configuration

```
# systemctl daemon-reload
```

Enable the timer

```
# systemctl enable --now munin-cron.timer
```

Munin Master - Notifications

Add line to allow notification via an external script

/etc/munin/munin.conf

```
...
html_strategy cron

contact.rocket.command >/usr/local/bin/munin_notify_rocket ${var:group} \
${var:host} ${var:graph_category} "${var:graph_title}" ${var:worst}

[xmen.jp-it.net.au;profx]
...
```

Munin Master - Notifications - External Notification Script

/usr/local/bin/munin_notify_rocket

```
#!/bin/bash
LOG_FILE="/tmp/munin_notify_rocket.log"
WEBHOOK_URL="http://172.17.0.3:3000/hooks/KX5jLvSAPkrNF3iSx/HJxhtnauw6RWgHnDNA3YhTo5n4d9ZnxzeH9CmoMzHMAAEt
ICON_EMOJI=":robot:"
GROUP="$1"
HOST="$2"
CATEGORY="$3"
GRAPH_TITLE="$4"
SERVICE_STATE="$5"
INPUT=$(cat)

if [ "${SERVICE_STATE}" = "CRITICAL" ]
then
    COLOR="danger"
elif [ "${SERVICE_STATE}" = "WARNING" ]
then
    COLOR="warning"
elif [ "${SERVICE_STATE}" = "ok" ]
then
    COLOR="good"
fi

PAYLOAD="{\"icon_emoji\":\":robot:\",\"text\":\"${GRAPH_TITLE}\",\"attachments\":[{\"title\":\"Munin: ${SE
curl -sX POST -o /dev/null --data "payload=${PAYLOAD}" ${WEBHOOK_URL} 2>&1
```

Munin Master - Troubleshooting

Watch Munin update log

```
# tail -f /var/log/munin/munin-update.log
```

Run Munin perl scripts as the `munin` user

```
# su - munin -s /bin/bash
$ munin-cron
$ cd /usr/lib/munin
$ ./munin-limit --always-send critical,warning
```

With static image generation if the master is running on a desktop with a browser the html files can be browsed directly from the filesystem.

`file:///usr/share/munin/www/profx/profx/index.html`

Munin Master - Troubleshooting

Master is able to connect to the node on port 4949
Manually test the master can connect to the node.

```
telnet 192.168.10.42 4949
```

Munin Node

Munin Node - Installation

Install the Munin Node package on all computers to be monitored.
Arch Linux:

```
# pacman -S munin-node
```

Debian Family:

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

The node package can also be installed on the Master itself.

Munin Node - Configuration

/etc/munin/munin-node.conf

```
log_level 4
allow ^127\.0\.0\.1$
allow ^::1$
allow ^192\.168\.2\.7$
ignore_file [\#~]$
ignore_file DEADJOE$
ignore_file \.bak$
ignore_file %$
ignore_file \.dpkg-(tmp|new|old|dist)$
ignore_file \.rpm(save|new)$
ignore_file \.pod$
host *
setsid 1
user root
background 1
pid_file /var/run/munin/munin-node.pid
group root
log_file /var/log/munin/munin-node.log
port 4949
host_name profx
```

Munin Node - Plugins

Plugins are enabled on the node by symlinking files in directory `/usr/lib/munin/plugins` to `/etc/munin/plugins` and then restarting the node service.

Regular plugin

```
# ln -s /usr/lib/munin/plugins/cpu /etc/munin/plugins/cpu
```

Wildcard plugin

```
# ln -s /usr/lib/munin/plugins/if_ /etc/munin/plugins/if_enp3s0
```

Restart the `munin-node` service

```
# systemctl restart munin-node
```

Munin Node - Plugins

The helper script `munin-node-configure` can be used to view and install plugins.
Show list of installed plugins:

```
# munin-node-configure
```

Suggest a list of plugins that may work

```
# munin-node-configure --suggest
```

Display the shell commands for the list of plugins that will work.
The output can be piped directly into a shell to run the commands.

```
# munin-node-configure --shell
# munin-node-configure --shell | sh
```

Munin Node - Writing Plugins

Plugins can be written in any language. With the expectation that:

- If run without any parameters the plugin is expected to output values.
- Specifying the `config` parameter prints out configuration information for the plugin. This is used by the Munin Master to configure options for images and the time series database files.

Munin Node - Writing Plugins

/etc/munin/plugins/how_many_beers

```
#!/bin/bash
if [ ! -e /tmp/beers ]
then
    touch /tmp/beers
    echo "0" > /tmp/beers
fi

case $1 in
   config)
        cat <<'EOM'
graph_title How Many Beers
graph_vlabel count
graph_category mlug
beers.label beers
beers.warning 6
beers.critical 10
EOM
        exit 0;;
esac
```

Munin Node - Writing Plugins

Show plugin information

```
# munin-run how_many_beers config
```

Show plugin value

```
# munin-run how_many_beers
```

Webserver

Nginx - Configuration

/etc/nginx/sites-available/munin

```
server {
    location /munin/static/ {
        alias /etc/munin/static/;
        expires modified +1w;
    }

    location /munin/ {
        alias /usr/share/munin/www/;
        expires modified +310s;
    }
}
```

Nginx - Configuration

```
# ln -s /etc/nginx/sites-available/munin \
/etc/nginx/sites-enabled/munin

# systemctl restart nginx
```

`http://localhost/munin/`

Demo

Further Reading

Further Reading

- http://guide.munin-monitoring.org/en/latest
- https://wiki.archlinux.org/index.php/Munin
- RRDTool http://oss.oetiker.ch/rrdtool/
- Topics not covered
    - SNMP
    - Tunneled SSH connections
    - TLS secured connections
    - cgi graph strategy
    - Script environment variables