Munin : l'outil de supervision sans fioritures

Installation sous Debian 8.0 Jessie

- __ Explications__

L'application munin se compose de 2 parties :

1. Munin (master pour les systèmes BSD qui sont plus explicites) qui se charge de générer les pages web et de réaliser les graphiques à partir des données...
2. fournies par Munin node qui se charge à l'aide de “sondes” écrites en perl ou en shell script de générer les fichiers RRD qui vont bien. Le démon écoute par défaut sur le port TCP 4949.

- __ Prérequis__

Il faut posséder une Debian Jessie à jour. Veuillez noter que, si vous n'avez pas modifié le fichier *Preferences* pour APT, installer munin revient à installer munin (master) et munin-node. Pour ce bloc-notes, nous nous positionnerons sur un schéma de fonctionnement le plus simple possible : munin et munin-node sont sur la même machine (physique ou virtuelle).

Installation des paquets nécessaires :

```
sudo apt install munin
```

Des plugins en plus (optionnel) :

```
sudo apt install munin-plugins-extra
```

Veuillez noter que certains plugins tiers peuvent nécessiter des programmes additionnels (bien souvent en perl tels ceux requis pour surveiller nginx).

- __ Configuration basique__

On édite **/etc/munin.conf** par exemple, sachant que la page web affichera monfqdn.demondomaine.grd :

```
[monfqdn.demondomaine.grd]
     address ::1
     use_node_name yes
```

Oui, IPv6 roulaize.

Petit moment sécurité

La configuration par défaut du paquet Debian fait écouter le démon munin-node sur toutes les interfaces réseaux. Il serait peut-être intéressant de limiter l'écoute à la boucle locale en éditant **/etc/munin-node.conf** :

```
host ::1
```

```
   #host *
```

- __ Configuration du CGI__

Depuis la version 2.0 (et même un peu avant mais passons), munin sait générer des graphiques de façon dynamique mais également des pages web de la même façon en lieu et place d'une bête tache cron. Par la suite, je détaillerai seulement la génération dynamique des graphiques et pas celles des pages web dont le comportement parfois aléatoire (au moins avec la version de Wheezy) était désespérant.

On repère les lignes intéressantes dans **/etc/munin.conf** :

```
   graph_strategy cgi
   #graph_strategy cron
```

```
   cgiurl_graph /munin-cgi/munin-cgi-graph
   #Important pour le serveur web
```

On installe spawn-fcgi :

apt once more :

```
apt install spawn-fcgi
```

On récupère les fichiers d'init (pour Jessie, il est aussi possible de créer une unité adéquate mais je n'ai pas modifié ces fichiers depuis Wheezy) :

A coller dans **/etc/init.d** avec les droits d'exécution

```
sudo wget
http://files.julienschmidt.com/public/cfg/munin/spawn-fcgi-munin-html
sudo wget
http://files.julienschmidt.com/public/cfg/munin/spawn-fcgi-munin-graph
```

Le premier est optionnel mais si vous êtes suffisamment malin, vous pourrez mettre en place une génération dynamique des pages web (c'est le même principe).

A modifier dans le fichier spawn-fcgi-munin-graph

```
   PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
   NAME=spawn-fcgi-munin-graph
   PID_FILE=/var/run/munin/$NAME.pid
   SOCK_FILE=/var/run/munin/$NAME.sock
   SOCK_USER=www-data
   FCGI_USER=www-data
   FCGI_GROUP=www-data
   FCGI_WORKERS=2
   DAEMON=/usr/bin/spawn-fcgi
   DAEMON_OPTS="-s $SOCK_FILE -F $FCGI_WORKERS -U $SOCK_USER -u $FCGI_USER -
g $FCGI_GROUP -P $PID_FILE -- /usr/lib/munin/cgi/munin-cgi-graph"
```

Veuillez noter que l'utilisateur de script doit être le même que celui de votre serveur web et la

configuration de munin.conf doit être en accord avec la dernière ligne.

N'oubliez pas d'activer le ou les scripts et rendre persistant leur démarrage.

- __ Configuration du vhost nginx__

Je suppose que vous accéder à l'adresse suivante : http://munin.fqdn.grd

```
 server {
      server_name munin.fqdn.grd;
      listen 80;
      listen [::]:80;
      if ($ssl_protocol = "") {
          rewrite ^   https://$server_name$request_uri? permanent;
      }
}
server {
      server_name munin.fqdn.grd;
      root /var/cache/munin/www;
      listen 443 ssl spdy;
      listen [::]:443 ssl spdy;
      ssl_certificate /etc/ssl/private/certificate.crt;
      ssl_certificate_key /etc/ssl/private/certificate.key;
      index index.html;
      rewrite ^([^.]*[^/])$ $1/ permanent;
  location /munin/static/ {
      alias /etc/munin/static/;
  }
  location ~* ^.+.(jpg|jpeg|gif|css|js|ico|txt|xml)$ {
      access_log /var/log/nginx/static.access.log;
      error_log /var/log/nginx/static.errors.log;
      expires 30d;
  }
  location ^~ /munin-cgi/munin-cgi-graph/ {
       fastcgi_split_path_info ^(/munin-cgi/munin-cgi-graph)(.*);
       fastcgi_param PATH_INFO $fastcgi_path_info;
       fastcgi_pass unix:/run/munin/spawn-fcgi-munin-graph.sock;
       include fastcgi_params;
  }
  location ~ /\.ht {
      allow 127.0.0.1;
      deny all;
  }
}
```

En cas de génération dynamique des pages web :

```
  location /munin/ {
fastcgi_split_path_info ^(/munin)(.*);
fastcgi_param PATH_INFO $fastcgi_path_info;
fastcgi_pass unix:/var/run/munin/fcgi-html.sock;
```

```
include fastcgi_params;
  }
```

- __ Installation du client__
- __ Installation des fichiers de configuration__

Important, les droits de l'utilisateur tomcat8 sur le fichier user-mapping.xml sont indispensables.

Copie des fichiers de configuration :

```
sudo mkdir {/etc/guacamole,/usr/share/tomcat8/.guacamole}
sudo cp guacamole-
client-0.9.6/guacamole/doc/example/{guacamole.properties,user-mapping.xml}
/etc/guacamole/
sudo ln -s /etc/guacamole/guacamole.properties
/usr/share/tomcat8/.guacamole/guacamole.properties
sudo chown tomcat8 /etc/guacamole/user-mapping.xml
sudo chmod 600 /etc/guacamole/user-mapping.xml
```

- __ Configuration__

Le fichier guacamole.properties à **éditer ou vérifier avant de redémarrer les services** :

```
     # Hostname and port of guacamole proxy
     guacd-hostname: localhost
     guacd-port:     4822
     # Auth provider class (authenticates user/pass combination, needed if
using the provided login screen)
     # Le chemin vers user-mapping.xml doit être un chein complet et non
relatif !
     auth-provider:
net.sourceforge.guacamole.net.basic.BasicFileAuthenticationProvider
     basic-user-mapping: /etc/guacamole/user-mapping.xml
```

Pour le fichier user-mapping.xml, référez-vous à la documentation officielle.

- Démarrage

Démarrage des services :

```
sudo systemctl tomcat8.service restart
sudo /etc/init.d/guacd restart
```

L'url http://fqdn:8080/guacamole doit vous permettre d'accéder à l'application. Toutefois, tout transite en clair sur le réseau et rien n'est sécurisé.

- Reverse proxy avec nginx

Le reverse proxy en question, je vous épargne la partie liée à **TLS qui est indispensable**.

```
     location / {
             proxy_pass http://localhost:8080/guacamole;
```

```
                proxy_buffering off;
                proxy_cookie_path /guacamole/ /; #indispensable pour se
connecter
                proxy_http_version  1.1;
                include conf.d/proxy.conf;
        }
```

Accès et utilisation

From:
https://wiki.mirtouf.fr/ - **Da mirtouf wiki**

Permanent link:
https://wiki.mirtouf.fr/doku.php?id=munin&rev=1435090730

Last update: **2015/06/23 20:18**