Documentation sur nginx

Sommaire

Présentation

Nginx [engine x] est un serveur HTTP(S) écrit par Igor Sysoev, dont le développement a débuté en 2002 pour les besoins d'un site russe à très fort trafic. Une partie de la documentation a été traduite du russe vers l'anglais. Ses sources sont disponibles sous une licence de type BSD. Il existe en plus du site principale un wiki proposant une documentation complète sur l'installation, la mise en production et détails sur les différentes options possibles.

Pourquoi Nginx ?

Depuis maintenant plusieurs années Apache règne dans le monde libre et est devenu **LE** serveur HTTP de référence. Cependant beaucoup de fonctionnalitées inutiles pour une utilisation dite *classique* ont été rajouté au cours du développement au détriment de la légerté de l'application. C'est pour cela que depuis quelques mois certaines alternatives plus optimisées et légères comme Lighttpd ou Nginx ont le vent en poupe.

Pour le cas de Nginx ses points fort sont:

- Système Asynchrone (Meilleurs gestion de requêtes simultanées)
- Code ultra modulaire (Noyau minimal + modules complémentaires)
- Gestion des machines multi-processeurs
- Gestion optimisé d'utilisation en mode proxy (Ce qui nous intéresse dans le cas d'un zope)

En fouillant sur le net ont peut trouver quelques benchmarks assez convaincants. Bref si on met de coté mon coté geek qui aime bien tester les nouvelles applis sur lequel on buzz pas mal, je me suis dit que ces atouts seraient les bienvenues pour une mise en production de service Zope/zwook. En effet dans un tel cas le serveur HTTP se contente d'être utilisé en tant que proxy et redirige les requêtes sur le serveur Zope. Bref pas besoin d un serveur HTTP qui fait le café mais plutôt d'un truc light avec un système de proxy qui tient la route. On remonte 5 6 lignes plus haut et on relit que c'est l'un des points forts de Nginx. CQFD !

Installation

Je ne vais pas refaire une documentation sur l'install de nginx, pour plusieurs raisons. Tout d'abord car elle est très classique si on connaît le traditionnel configure, make, make install et surtout que le wiki en fournit une plus que détaillée. A noter qu'il existe aussi un package debian, mais son créateur a décidé d'arrêter pour a peut près les mêmes raisons[1]. De plus une page regroupe une liste de tutoriels d'installation selon le système d'exploitation et sa version. Pas besoin de re-inventer la roue ...

Post-Installation

Hint

Ce qui suit n'est pas obligatoire mais a mon avis améliore le confort d'utilisation et d'administration du logiciel:

Séparation des configurations de domaine à la apache2 debian

Par défaut la configuration des domaines/virtual host se fait dans un seul et unique fichier: conf/nginx.conf. ceci dit cela peut devenir vite brouillon dans le cas d'une configuration avec plusieurs domaines. Nous allons donc séparer les fichiers par domaine.

- Allez dans le répertoire de configuration:

```
> cd /usr/local/nginx/conf
```

- Créez deux répertoires site-available et site-enabled:

```
> sudo mkdir sites-available
> sudo mkdir sites-enabled
```

- Supprimez du fichier nginx.conf ce passage:

```
server {
    listen       80;
    server_name  localhost;

    #charset koi8-r;

    #access_log  logs/host.access.log  main;

    location / {
        root   html;
        index  index.html index.htm;
    }
```

```
        #error_page  404              /404.html;

        # redirect server error pages to the static page /50x.html
        #
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }

        # proxy the PHP scripts to Apache listening on 127.0.0.1:80
        #
        #location ~ \.php$ {
        #    proxy_pass   http://127.0.0.1;
        #}

        # pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000
        #
        #location ~ \.php$ {
        #    root           html;
        #    fastcgi_pass   127.0.0.1:9000;
        #    fastcgi_index  index.php;
        #    fastcgi_param  SCRIPT_FILENAME  /scripts$fastcgi_script_name;
        #    include        fastcgi_params;
        #}

        # deny access to .htaccess files, if Apache's document root
        # concurs with nginx's one
        #
        #location ~ /\.ht {
        #    deny  all;
        #}
    }
```

- Le coller dans le fichier site-available/default.conf
- Faites un lien symbolique de default.conf vers site-enabled:

```
> sudo ln -s site-available/default.conf site-enabled/default.conf
```

- Ensuite toujours dans le fichier nginx.conf mettez à la place du texte supprimé:

```
include sites-enabled/*;
```

Voila je pense que vous aurez compris le système. Chaque domaine aura son fichier dans le répertoire sites-available et pour l'activer il suffira de faire un lien symbolique de ce fichier dans le répertoire sites-enabled. Si jamais cela semble encore trouble vous pouvez directement aller voir une illustration dans le paragraphe Configuration d'une instance zwook

Activation du logrotate

Par défaut les logs sont stocké dans un seul et unique fichier. Nous allons donc configurer le logiciel Logrotate pour effectuer une rotation/compression automatique des logs.

Pour cela créez un fichier nginx dans logrotate.d et y insérer:

```
/usr/local/nginx/logs/*.log {
    daily
    missingok
    rotate 52
    compress
    delaycompress
    notifempty
    create 640 root adm
    sharedscripts
    postrotate
            [ ! -f /usr/local/nginx/run/nginx.pid ] || kill -
USR1 ‘cat /usr/local/nginx/run/nginx.pid‘
    endscript
}
```

Warning

Si jamais votre distibution n'utilise pas des fichiers de confs séparés pour logrotate, coller cette configuration dans le fichier /etc/logrotate.conf

Installation d'un script init.d

Maintenant nous allons rajouter un script de contrôle au fichier init.d, cela permettra d'harmoniser démarrage, redémarrage du service et un démarrage automatique au lancement de la machine.

- Créez un fichier /etc/init.d/nginx et y insérer:

```
#! /bin/sh

### BEGIN INIT INFO
# Provides:          nginx
# Required-Start:    $all
# Required-Stop:     $all
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: starts the nginx web server
# Description:       starts nginx using start-stop-daemon
### END INIT INFO

PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
DAEMON=/usr/local/nginx/sbin/nginx
NAME=nginx
DESC=nginx

test -x $DAEMON || exit 0

# Include nginx defaults if available
if [ -f /etc/default/nginx ] ; then
        . /etc/default/nginx
fi

set -e
```

```
case "$1" in
  start)
        echo -n "Starting $DESC: "
        start-stop-daemon --start --quiet --
pidfile /usr/local/nginx/run/$NAME.pid \
                --exec $DAEMON -- $DAEMON_OPTS
        echo "$NAME."
        ;;
  stop)
        echo -n "Stopping $DESC: "
        start-stop-daemon --stop --quiet --
pidfile /usr/local/nginx/run/$NAME.pid \
                --exec $DAEMON
        echo "$NAME."
        ;;
  restart|force-reload)
        echo -n "Restarting $DESC: "
        start-stop-daemon --stop --quiet --pidfile \
                /usr/local/nginx/run/$NAME.pid --exec $DAEMON
        sleep 1
        start-stop-daemon --start --quiet --pidfile \
                /usr/local/nginx/run/$NAME.pid --exec $DAEMON --
$DAEMON_OPTS
        echo "$NAME."
        ;;
  reload)
      echo -n "Reloading $DESC configuration: "
      start-stop-daemon --stop --signal HUP --quiet --
pidfile /usr/local/nginx/run/$NAME.pid \
          --exec $DAEMON
      echo "$NAME."
      ;;
  *)
N=/etc/init.d/$NAME
echo "Usage: $N {start|stop|restart|force-reload}" >&2
exit 1
;;
esac

exit 0
```

- Rendre ce script exécutable:

```
> sudo chmod +x /etc/init.d/nginx
```

- Maintenant que le script est prêt il n'y a plus qu'a l'ajouter au système de boot:

```
> sudo /usr/sbin/update-rc.d -f nginx defaults
```

- La sortie à l'écran devrait être similaire à ça:

```
Adding system startup for /etc/init.d/nginx ...
   /etc/rc0.d/K20nginx -> ../init.d/nginx
   /etc/rc1.d/K20nginx -> ../init.d/nginx
   /etc/rc6.d/K20nginx -> ../init.d/nginx
```

```
/etc/rc2.d/S20nginx -> ../init.d/nginx
/etc/rc3.d/S20nginx -> ../init.d/nginx
/etc/rc4.d/S20nginx -> ../init.d/nginx
/etc/rc5.d/S20nginx -> ../init.d/nginx
```

- Si nginx est toujours lancé à la main l'éteindre:

```
> sudo kill `cat /usr/local/nginx/logs/nginx.pid`
```

- Puis relancer via le script:

```
> sudo /etc/init.d/nginx start
```

Voila nginx est lancé via init.d, a noter que pour restart, reload la configuration ou stopper le service il suffit d'utiliser les commandes suivantes:

```
> sudo /etc/init.d/nginx restart
...
> sudo /etc/init.d/nginx reload
...
> sudo /etc/init.d/nginx stop
...
```

Configuration pour une machine multi-processeurs

Nginx peut se configurer pour une machine multi-proc ou avec un processeur multi-core[2].

En gros il suffit de préciser dans le fichier de configuration global (conf/nginc.conf) le nombre de process que lancera Nginx et a quels processeurs les attribuer.

Exemple avec une machine a 4 processeurs:

```
worker_processes     4;
worker_cpu_affinity 0001 0010 0100 1000;
```

Nous lancerons 4 process et chacun d'eux sera attribué à un processeur.

Autre exemple:

```
worker_processes     2;
worker_cpu_affinity 0101 1010;
```

Nous créerons cette fois ci 2 process, le premier sera attribué aux processeurs CPU0/CPU2 et le deuxieme au CPU1/CPU3.

Configuration d'une instance zwook

Il ne nous reste plus qu'a regarder comment configurer un domaine et le re-diriger sur une instance zope. Nous allons prendre l'exemple d'un site zwook avec le domaine monsitezwook.com. Le folder se nomme *monsitezwook* et se trouve dans la racine du zope. Le zope tourne sur le port 8080 de la machine.

- Créez un fichier monsitezwook.com.conf[3] dans le répertoire conf/site-available et y insérer cette configuration:

```
server {
    listen       80;
    server_name  monsitezwook.com *.monsitezwook.com;
```

```
        access_log  logs/monsitezwook.access.log;

        location / {
            proxy_pass http://localhost:8080/VirtualHostBase/http/www.monsitezwook.com:80/monsi
        }
    }
```

Note

Pour cette configuration tous les sous domaines de monsitezwook.com seront mappés sur cette configuration. Si jamais pour une raison ou une autre vous ne voulez pas il suffit de supprimer de la ligne *server_name* le *.monsitezwook.com et mettre a la place tous les sous domaines qu'on désire utiliser (exemple www.monsitezwook.com) De même la redirection sera automatiquement faite sur www.monsitezwook.com, si vous voulez garder le domaine rentré par l'utilisateur mettre a la place dans la ligne *proxy_pass* $host:80

- Maintenant faire un lien symbolique du fichier dans site-enabled:

```
> sudo ln -s /usr/local/nginx/conf/sites-available/monsitezwook.com.conf /usr/local/nginx/c
enabled/monsitezwook.com.conf
```

- Reloadez nginx:

```
> sudo /etc/init.d/nginx reload
```

et voila !!!

Authors: Philippe Lafaye aka RAGE2000

Contact: lafaye at emencia.com

Organization: Emencia http://www.emencia.com

Version: 1.1

Date: 2007-07-24

[1] I am no longer going to be making or updating these packages. Compiling nginx from source is very easy, and is still the recommened way to run the current version.

[2] http://wiki.codemongers.com/NginxMainModule#worker_cpu_affinity

[3] Le nom du fichier est totalement libre, ceci dit l'utilisation de la norme domaine.conf permet de garder les choses claires !!