UNIVERSITÉ DE ROUEN	ANNÉE 2024-2025
FACULTÉ DES SCIENCES	ADMINISTRATION RÉSEAUX
MASTER 2 SSI	FICHE DE TD N° 3

Autour de Nginx

Exercice 1

L'objectif de cet exercice est d'acquérir une expérience pratique de l'installation et de la configuration de Nginx à partir des sources, tout en comprenant les différents paramètres de configuration.

1) Installez les paquets requis pour la compilation de Nginx, y compris les outils de développement et les bibliothèques nécessaires :
- `build-essential` : ensemble d'outils de compilation GNU (GCC).
- `libpcre3-dev` : bibliothèque pour la prise en charge des expressions régulières compatibles avec Perl.
- `zlib1g-dev` : bibliothèque de compression.
- `libssl-dev` : bibliothèques pour le support SSL/TLS.

2) Téléchargez la dernière version de Nginx depuis le site officiel et extrayez le code source dans le répertoire `/usr/local/src`
3) Explorer les options de configuration du script `configure` en l'exécutant avec l'option `-help`, puis donner une description des otptions suivantes :
`-prefix`, `-sbin-path`, `-conf-path`, `-pid-path`, `-lock-path`, `-http-log-path`, `-error-log-pa`
`-with-pcre`, `-with-http_ssl_module`.
4) Lancez le script de configuration avec les options appropriées que vous avez explorées.
5) Compilez le binaire de Nginx et installez-le.
6) Confirmez que Nginx a été installé correctement en vérifiant sa version, puis vérifiez les options de construction utilisées pour compiler Nginx.
7) Créez un fichier de service nommé `nginx.service` dans `/etc/systemd/system/` pour permettre à Nginx de démarrer automatiquement au démarrage du système.
7) Une fois le fichier de service créé, rechargez le gestionnaire de services et démarrez Nginx :
8) Pour vous assurer que Nginx fonctionne correctement, créez une page web `index.html` contenant le texte "`<h1>Bienvenue en SSI!</h1>`" dans `/var/www/html`,
puis en utilisant l'utilitaire `curl`, lancez une requête au serveur Nginx en accédant à `http://localhost`.

9) Donner les étape nécessaires pour ajouter un module comme `ngx_http_stub_status_module` à nginx.

10) Activer le module, puis ajouter la configuration suivante dans le fichier de configuration de Nginx

```
location /status {
  stub_status on;
  allow 127.0.0.1;
  deny all;
}
```

11) Redémarrez Nginx pour appliquer les modifications, puis lancez une requête au serveur Nginx en accédant à `http://localhost/status`.

Exercice 2

L'objectif de cet exercice est de se familiariser avec la configuration de Nginx en créant des hôtes virtuels, en gérant les journaux et en configurant des redirections.

1. Créer deux hôtes virtuels pour les domaines `*.ssi.edu` et `*.gil.edu` écoutant sur le port 8080.
2. Que se passe-t-il si deux hôtes virtuels partagent le même `server_name`, mais présentent des configurations différentes ?
3. Mette en place des journaux d'accès et d'erreurs pour les hôtes virtuels que vous avez configurés.
4. Quelle est la différence entre les niveaux de journalisation dans `error_log`, tels que `error`, `warn` et `info` ?
5. Créer un bloc `location` pour servir des fichiers statiques à partir du répertoire `/var/www/ssi/static`.
6. Donner la signification des directives suivantes :

```
location ~* \.(css|js|jpg|png)$ {
  access_log off;
  add_header Cache-Control "public, max-age=2592000";
  add_header Pragma public;
  add_header Vary Accept-Encoding;
  expires 30d;
}
```

   Testez cette configuration à l'aide de `curl -I`
7. Quelle est la distinction entre `location /adminrezo/` et `location /adminrezo` ?
8. Comment effectuer une réécriture d'URL de `/ancien` vers `/nouveau` de manière permanente ? Effectuez un test pour vérifier son bon fonctionnement.
9. Quelle est la différence entre une réécriture (utilisant `rewrite`) et une redirection (utilisant `return`) ?

```
location /ancien {
  rewrite ^/ancien$ /nouveau permanent;
}

location /ancien {
  return 301 http://www.ssi.edu/nouveau;
}
```

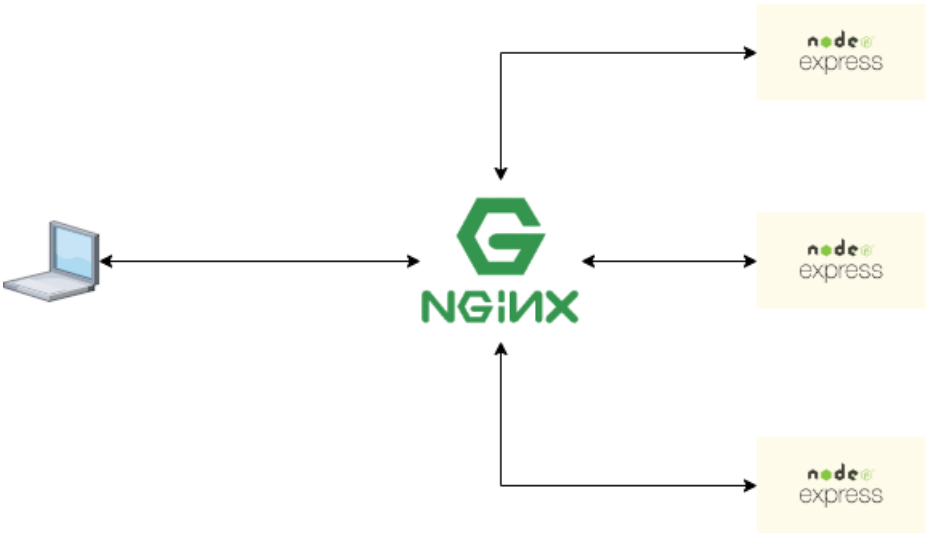
10. Définissez une variable `$racine` pour représenter le chemin racine du site `ssi` et utilisez cette variable dans un bloc `location`.
11. Comment implémenter `try_files` pour tenter plusieurs fichiers avant de retourner une erreur 404 ?
12. Configurez une location nommée `nontrouve` pour gérer les requêtes qui ne correspondent à aucun fichier ou ressource existant.
13. Expliquer les directives suivantes :

```
http {
  gzip on;
  gzip_types text/plain text/css application/json application/javascript text/xml
      application/xml application/xml+rss text/javascript;
  gzip_min_length 256;
  gzip_comp_level 5;
}
```

14. Ajouter les types images à la compression.
15. Quel entête est envoyé par un client web pour informer le serveur qu'il accepte la compression gzip ? Comment vérifier si gzip est activé et fonctionne correctement en utilisant `curl` ?
16. Configurez Nginx pour traiter les fichiers PHP en utilisant `php-fpm`.

Objectif

L'objectif de ce TP est la mise en place d'une infrastructure complète de load balancing avec Nginx et des serveurs Express.js dans un environnement Docker.



1) Configurez un serveur `Nginx` en tant que load balancer pour répartir les requêtes vers un backend constitué de trois serveurs `Express`. Chaque serveur Node.js fonctionne sur les ports suivants :

- `localhost:3001`
- `localhost:3002`
- `localhost:3003`

Le load balancer doit utiliser la stratégie de répartition par défaut de Nginx.

2) Écrivez une application `Express` qui écoute sur le port 3001 et retourne un message indiquant le serveur qui traite la requête.

3) Comment démarrer plusieurs serveurs `Express` sur les ports 3002 et 3003 pour compléter la configuration de load balancing Nginx ?

3) Comment configurer `Nginx` pour activer le protocole `HTTP/2` uniquement entre le client et le serveur Nginx, tout en gardant la communication entre Nginx et les serveurs backend `Express` en `HTTP/1.1` ?

4) Comment pouvez-vous utiliser Docker pour conteneuriser le serveur Nginx ainsi que les trois serveurs `Express`, et orchestrer le tout avec Docker Compose ?