## Exercice 6 – Nginx (Corrigé)

## Remarque : La couleur de police rouge ou les surlignements gris indiquent le texte qui apparaît uniquement dans la copie corrigée.

Informations

**Évaluation** : formative

**Type de travail** : individuel

**Durée** : 3 heures

**Système d’exploitation** : Ubuntu serveur

**Environnement** : Docker

Objectifs

Cet exercice a pour objectifs :

* Installer le serveur Web Nginx.
* Faire une configuration de base et avec des hôtes virtuels.

Dans cet exercice, vous allez installer un environnement de serveur Web (HTTP) nginx, regarder à sa configuration de base et configurer des hôtes virtuels.

**Note** : si vous fermez votre VM, n’oubliez pas d’arrêter vos conteneurs :

docker container stop *nom\_conteneur*

docker container start *nom\_conteneur*

#### Partie 1 : Installation de nginx

Dans cette partie, vous allez installer le serveur Web nginx selon les spécifications données.

Étape 1 : Installation

1. Pour débuter, allez à l’image officielle sur le site de docker hub et prenez en note la commande pour lancer le conteneur.
2. Avant de lancer notre serveur, on va se créer un répertoire pour nos fichiers HTML de ce projet. À partir du répertoire Dev, créer un répertoire SiteNginx/html. Vous devriez avoir la structure suivante :

/home/user01/Dev/SiteNginx/html

1. Copiez le fichier index.html, que vous avez créé précédemment, dans html. Adapter le contenu au nouveau serveur.
2. Entrez dans le répertoire SiteNginx et exécutez la commande recommandée pour lancer l’image nginx. Ajuster la commande pour que le répertoire html soit monté sur le répertoire de documents du conteneur, en lecteur-écriture, et que le port 80 soit redirigé vers le port 8080.

docker run --name some-nginx -p 8080:80 -v "$PWD"/html:/usr/share/nginx/html

-d nginx

1. Vérifiez que votre image s’exécute bien.

docker container ls

ss -tnap | grep 8080

Ouvrez un navigateur et allez au site Web http://www.monsite.com:8080.

Avez-vous une page Web? Sinon, pourquoi?

Oui.

#### Partie 2 : Configurations

Dans cette partie, vous allez regarder la configuration par défaut et récupérer le fichier de configuration par défaut.

Étape 1 : Consultation des configurations par défaut

1. Connectez-vous sur votre serveur en ligne de commande.

docker container exec -it some-nginx bash

1. Déplacez-vous dans le répertoire de configuration et récupérez le fichier de configuration par défaut. Vous pouvez juste le copier dans le répertoire monté. Ouvrez le fichier de configuration et répondez aux questions suivantes :

Consulter les sites de documentations de nginx pour vous aider :

<https://nginx.org/en/docs/>

<https://docs.nginx.com/nginx/admin-guide/>

* 1. Quelle est l’utilité de la directive user ? L’utilisateur sous lequel s’exécute gninx.
  2. Quelle est l’utilité de la directive worker\_process ? . The number of worker processes is defined by the worker\_processes directive in the nginx.conf configuration file and can either be set to a fixed number or configured to adjust automatically to the number of available CPU cores.
  3. À quel endroit se trouvent les logs et sont-ils gérés par docker ?

error\_log /var/log/nginx/error.log notice;

access\_log /var/log/nginx/access.log main;

Oui, ils sont gérés quand même par docker, si l'on fait la commande docker container logs some-nginx, on les voit.

* 1. Pourquoi ne voit-on pas de contexte server dans le contexte http? Car nous avons include /etc/nginx/conf.d/\*.conf;. Dans conf.d/ nous avons le fichier default.conf qui contient le contexte server.
  2. Suite à la réponse de la question précédente, comment pouvons-nous implanter des hôtes virtuels avec cette version de nginx ? En créant un fichier *hote*.conf par hôte virtuel dans conf.d/.

1. Récupérer le fichier /etc/nginx/conf.d/default.conf.
2. Arrêter votre conteneur.

Étape 2 : Hôtes virtuels

1. À cette étape, avec ce que vous avez appris à l’étape précédente, vous allez refaire votre site avec des hôtes virtuels avec nginx. Votre structure de répertoire devrait ressembler à ça :

/SitesNginx/

|--docker-compose.yml

|--Dockerfile

|--conf/

| |--nginx.conf

| |--default.conf

| |--site02.com.conf

|--site01/

| |--html/

| |--index.html

|--site02/

| |--html/

| |--index.html

* 1. Créer les fichiers nécessaires dans SitesNginx pour créer une image et un conteneur avec les spécifications suivantes (vos fichiers index.html doivent indiquer le nom du site et que le serveur est nginx) :

**Image à utiliser** : nginx:alpine.

**Dans l’image** : copier les fichiers de configurations nécessaires et exposer les ports 80 et 443. L’hôte par défaut est site01.com et l’on veut 2 processus worker pour le serveur.

**Répertoires à créer dans l’image** : /usr/share/nginx/html/site01.com et /usr/share/nginx/html/site02.com.

**Nom du conteneur** : nginx.

**Ports à exposer** : le port 80 au port 80 du conteneur.

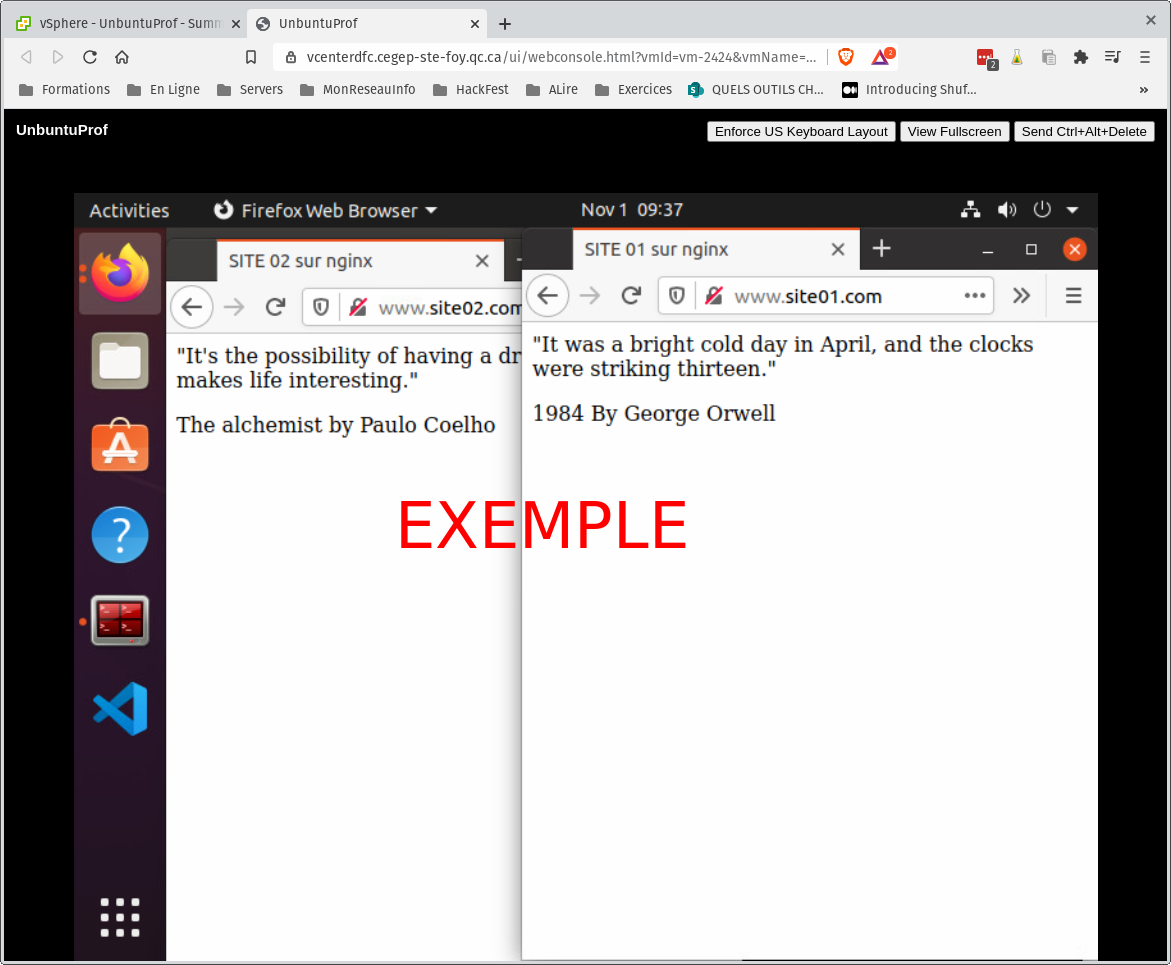
**Point de montage** : site01/html <-> /usr/share/nginx/html/site01.com et site02/html <-> /usr/share/nginx/html/site02.com. En lecture seulement.

**Nom de domaine** : les sites doivent être accessibles sur votre VM avec site01.com, www.site01.com site02.com et www.site02.com.

1. Lancer votre conteneur et vérifier que les deux sites fonctionnent.
2. Arrêter et effacer votre conteneur nginx.

Pour vérification

Remettre une capture d’écran de votre Linux avec Firefox ouvert sur les deux sites de votre serveur Web.



Compétences développées

|  |  |
| --- | --- |
| **00SJ** – Effectuer le déploiement de serveurs intranet. | **00SJ # 1** – Analyser le projet de déploiement.  **00SJ # 2** – Monter les serveurs intranet.  **00SJ # 3** – Installer les services intranet.  **00SJ # 5** – Participer à la mise en service des serveurs intranet. |

**Note** : les compétences sont développées en partie.

Références

<https://nginx.org/en/docs/>

<https://docs.nginx.com/nginx/admin-guide/>

<https://hub.docker.com/_/nginx>

Solution partie 5.2

**# docker-compose.yml**

version: "3.2"

services:

nginx:

build:

context: .

ports:

- "80:80"

volumes:

- ./site01/html:/usr/share/nginx/html/site01.com

- ./site02/html:/usr/share/nginx/html/site02.com

**# Dockerfile**

# Creer une image pour notre serveur nginx

# 420-W45-SF - 20201101 - CR

FROM nginx:alpine

# Utilise la derniere image

RUN mkdir -p /usr/share/nginx/html/site01.com

# Ajoute le repertoire pour site01

RUN mkdir -p /usr/share/nginx/html/site02.com

# Ajoute le repertoire pour site02

EXPOSE 80 443

# Exposer les ports 80 et 443

WORKDIR /etc/nginx/

# Change dans le repertoire de configuration

COPY conf/nginx.conf nginx.conf

# Remplacer nginx.conf par le mien

COPY conf/default.conf conf.d/default.conf

# Remplacer default.conf par le mien, pour site01.com

COPY conf/site02.com.conf conf.d/site02.conf

# Copier le fichier de configuration pour site02.com

**# nginx.conf**

user nginx;

worker\_processes 2;

error\_log /var/log/nginx/error.log warn;

pid /var/run/nginx.pid;

events {

worker\_connections 1024;

}

http {

include /etc/nginx/mime.types;

default\_type application/octet-stream;

log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

access\_log /var/log/nginx/access.log main;

sendfile on;

#tcp\_nopush on;

keepalive\_timeout 65;

#gzip on;

include /etc/nginx/conf.d/\*.conf;

}

**# default.conf**

server {

listen 80;

listen [::]:80;

server\_name site01.com www.site01.com;

#charset koi8-r;

#access\_log /var/log/nginx/host.access.log main;

location / {

root /usr/share/nginx/html/site01.com;

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 /usr/share/nginx/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;

#}

}

**# site02.conf**

server {

listen 80;

listen [::]:80;

server\_name site02.com www.site02.com;

#charset koi8-r;

#access\_log /var/log/nginx/host.access.log main;

location / {

root /usr/share/nginx/html/site02.com;

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 /usr/share/nginx/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;

#}

}