Servir plusieurs sites web avec le même serveur

Note : Pour pouvoir faire ce TP, il faut avoir fait les TP « héberger une page web » et « enregistrer un nom de domaine pour votre conteneur ». Les objectifs opérationnels et pédagogiques de ce TP sont décrits en bas de la page.

Repartir avec les fichiers originaux

Le TP de découverte d'`nginx` était un TP de navigation dans l'arborescence et d'observation des fichiers de configuration de nginx, que vous n'étiez pas censé·e modifier.

Plusieurs étudiant·es ont modifié le fichier `/etc/nginx/sites-available/default` sur lequel vous allez vous baser dans ce TP.

Pour savoir si c'est votre cas, vous pouvez regarder la page de suivi individuel ou vérifier que le hash `SHA256` du fichier est celui du fichier d'origine :

```
$ sha256sum /etc/nginx/sites-available/default
ce0901350a021608139b5639cf4ccd7717bef8c3a9e4f79031eb46386b67b03f  /etc/nginx/sites-available/default
```

Ce fichier est fourni par le paquet `nginx-common`, qui est une dépendance du paquet `nginx`.

Si le fichier a été modifié, il suffit de réinstaller le paquet `nginx-common` et de redeployer ses fichiers :

```
# apt -o Dpkg::Options::="--force-confask" reinstall nginx-common
```

La version modifiée de ce fichier sera sauvegardée sous le nom `/etc/nginx/sites-available/default.dpkg-old`, vous pouvez voir vos modifications apportées au fichier grâce à la commande `diff`.

Introduction aux « hôtes virtuels »

Lorsqu'on entre une `URL` `http://domaine/page.html` dans un client web (un navigateur graphique comme `firefox` ou le logiciel `wget` par exemple), le nom de domaine de l'URL est utilisé deux fois :

- une première fois lors de la résolution de nom (DNS) qui permet de connaître l'adresse IP de la machine qui héberge le serveur web qui sert la page web correspondant à l'URL.
- une seconde fois, lors de l'échange `HTTP` avec le serveur web, afin d'identifier le site web dont vous demandez la page web.

En effet, un serveur web peut servir plusieurs sites web, ainsi `/page.html` n'est pas suffisant pour déterminer une page web fournie par un serveur web, car selon le nom de domaine, il peut s'agir de pages différentes.

Par exemple, si vous visitez les sites https://rdos.lipn.univ-paris13.fr/, https://ask.sagemath.org/ et https://sysadmin2023.netlib.re/ vous tombez sur des pages différentes, alors que c'est le même serveur web qui les sert, et à chaque fois, vous avez demandé la page qui se trouve à la racine du site :

```
$ dig +short A rdos.lipn.univ-paris13.fr
194.254.163.53
```

```
$ dig +short A ask.sagemath.org
194.254.163.53
$ dig +short A sysadmin2023.netlib.re
194.254.163.53
```

Par ailleurs, souvenez-vous de la partie réseau du cours, lorsqu'on simule un client web avec `socat` :

```
$ socat STDIO TCP:example.com:80,crnl << EOF
GET /index.html HTTP/1.1
Host: example.com

EOF
```

le nom de domaine `example.com` apparaît une première fois pour établir la connexion TCP et une seconde fois lors de l'échange avec le serveur web pour renseigner la valeur du champ `Host`, qui est obligatoire dans le protocole `HTTP`.

On peut connaître l'adresse IPv4 de la machine dont le nom de domaine est `example.com` :

```
$ dig +short A example.com
93.184.216.34
```

Si on remplace `example.com` par son adresse IPv4 dans la commande `socat`, ça va encore marcher :

```
$ socat STDIO TCP:93.184.216.34:80,crnl << EOF
GET /index.html HTTP/1.1
Host: example.com

EOF
```

Par contre, si on remplace `example.com` par son adresse IPv4 dans l'échange `HTTP`, le serveur ne va pas savoir de quel site il doit servir la page `/index.html` :

```
$ socat STDIO TCP:93.184.216.34:80,crnl << EOF
GET /index.html HTTP/1.1
Host: 93.184.216.34

EOF
```

La technique permettant à un même serveur web se servir des sites web différents selon le nom de domaine (donnant l'illusion qu'il s'agit de machines différentes) s'appelle *name based virtual host* (fr: hôte virtuel basé sur le nom), avec les dérivés *virtualhost*, *vhost*, etc.

L'objectif (opérationnel) de ce TP est de vous faire héberger plusieurs sites web sur votre conteneur. C'est aussi une étape nécessaire pour le chiffrement des connexions entre le client et le serveur web à l'aide d'un certificat TLS (anciennement SSL), car celui-ci est lié à un ou plusieurs noms de domaine.

Préparation de trois sites web

On suppose que vous avez enregistré le nom de domaine `exemple.netlib.re` et que son enregistrement `AAAA` pointe vers l'adresse IPv6 de votre conteneur (cf TP précédent sur le DNS).

1. Configurez votre fichier de zone pour que les deux noms de domaine `serieux.exemple.netlib.re` et `marrant.exemple.netlib.re` pointent eux aussi vers l'adresse IPv6 de votre conteneur. Vous pouvez aussi choisir `pro` et `perso` si vous préférez, ou autre chose, l'essentiel est d'avoir deux sous-domaines supplémentaires, soit un total de trois domaines configurés au total.

   Pour alléger la configuration, au lieu de recopier l'IPv6 dans l'enregistrement `AAAA` pour ces sous-domaines, utilisez l'enregistrement `CNAME`. Vous pouvez aussi utiliser des chemins relatifs au domaine principal (l'*apex* de la zone), et désigner celui-ci par `@`.

2. Créez trois répertoires dans `/var/www/` ou dans `/srv/` (au choix) pour accueillir le contenu de vos trois sites web.
3. Dans chacun de ces répertoires, créez une page web d'accueil basique qui y corresponde.
4. Pour le site du nom de domaine principal, faites en sorte que la page d'accueil ait un hyperlien vers les deux autres sites.

   Astuce : en `HTML` pour faire un hyperlien, on utilise une balise de type `<a>` (ancre, en: anchor) :

   ```
   <a href="URL_DU_LIEN">TEXTE_DU_LIEN</a>
   ```

Configuration du serveur web

Lors du TP sur l'hébergement d'une page web, vous avez observé le fonctionnement du serveur web `nginx`, sans modifier ses fichiers de configuration.

Le fichier de configuration par défaut, `/etc/nginx/sites-available/default` contient la ligne :

```
server_name _;
```

Le underscore `_` signifie que ce bloc `server` s'applique à tous les noms de domaine qui n'auraient pas été définis ailleurs.

Ce fichier est fourni par le paquet `nginx-common` qui est une dépendance du paquet `nginx` installé lors d'un TP précédent. Ainsi, si vous modifiez ce fichier, il pourra être écrasé lors d'une mise à jour du paquet `nginx-common`. Il est donc recommandé de ne pas modifier ce fichier, mais plutôt d'en créer d'autres.

Ce fichier a un rôle de configuration par défaut, mais aussi un rôle de documentation par l'exemple. Dans le bloc de configuration principal, on peut voir des exemples (commentés) de configuration de SSL ou d'interprétation de scripts PHP.

Le dernier paragraphe de ce fichier donne un exemple de configuration autonome pour un virtual host.

5. Copiez ce bloc dans trois fichiers qui correspondent aux trois domaines définis plus haut et décommentez-le (c'est dans ces situations que vous verrez l'intérêt des éditeurs puissants comme `vim` ou `emacs` avec lesquels il est possible de supprimer une colone).
6. Adaptez chaque fichier à vos trois sites web.
7. Passez vos sites en production un par un et vérifiez à chaque fois que tout fonctionne comme prévu. Pour passer un site en production, relisez les question 7, 8 et 9 du TP « Héberger une page web ».

   Astuce : avant de recharger la configuration de nginx avec `systemctl`, vous pouvez vérifier que vous n'avez pas fait d'erreur de syntaxe avec la commande `nginx -t`.
8. Ajoutez les URL de vos trois sites web sur votre page personnelle du wiki.

Lorsque vous avez vos trois sites fonctionnels, vous pouvez ajouter le tag `vhost` à votre page wiki.

Objectifs du TP :

- objectifs opérationnels du TP :
  - héberger plusieurs sites web indépendants par un même serveur web.
  - prendre le contrôle de la configuration de nginx et préparer le terrain pour le chiffrement TLS.
- objectifs pédagogiques du TP :
  - configurer un service en modifiant un template existant.