**III/ Automatisation**

Nous allons maintenant voire comment automatiser le déploiement de notre solution de manière sécurisée. En effet actuellement vous spécifiez le mot de passe du compte administrateur du webmin de votre DNS directement dans le Dockerfile. Si vous publiez votre solution sur un dépôt comme github n'importe qui y ayant accès connaitra votre mot de passe, ce n'est donc pas sécurisé. Nous allons donc modifier notre compose.

Nous indiquerons dans notre compose la variable d'environnement du mot de passe administrateur, nous ne l'indiquerons pas directement mais nous indiquerons au fichier docker compose d'utiliser un certain fichier d'environnement. Ce fichier s'appelera .env . Nous indiquerons également au compose d'utiliser l'image sameersbn/bind (et non pas notre image personnalisée vue que l'étape de définition de la variable d'environnement se fera directement dans le compose).

Si nous rédigeons directement le fichier .env cela ne sera pas plus sécurisé que d'écrire le mot de passe administrateur dans le docker-compose.yml. Nous allons donc rédigez un script qui demandera à l'utilisateur le mot de passe administrateur, puis l'écrira dans le fichier .env et qui démarrera pour finir notre solution.

Pour rédiger ce script nous aurons besoin du paquet make. Ce script portera le nom de Makefile. Dans ce Makefile nous indiquerons des catégories dans lesquels nous mettrons des commande linux à exécuter. Voici à quoi va ressembler notre script:

run:  
 @read -p "Saisissez le mot de passe admin: " PASS; echo "ROOT\_PASSWORD="$$PASS > .env  
 docker compose up -d --build

Quelques explications sur la première ligne :

* L'instruction read -p permet d'afficher du texte, de laisser l'utilisateur entrer une information et de l'enregistrer dans une variable, en l'occurrence PASS.
* L'instruction echo "du texte" > fichier permet d'écrire tout ce qu'il y a avant le chevron (>) dans le fichier indiqué après. Si le fichier n'existe pas il est créé.

Avec ce fichier rédigé vous pouvez taper la commande make run (en étant dans le dossier où figure votre Makefile) pour que linux exécute les commandes de votre section run.

Rédigez maintenant des sections permettant de stopper votre solution et également de démarrer la solution sans redéfinir la variable d'environnement.

**IV/ Déploiement sur gitlab**

Maintenant que votre solution est complète vous pouvez la déployer sur un dépôt distant qui vous permettra d'y avoir accès de n'importe où. Nous allons utiliser la solution gitlab (qui est une solution alternative à github mais fonctionnant exactement de la même manière) pour héberger notre solution. Commencez par vous créer un compte gitlab et de démarrer un nouveau projet.

Sur la page de création du projet donnez-lui un nom, une description, cochez la case publique et décochez la case vous proposant d'ajouter un fichier README.

Lorsque vous valider la création du projet vous arrivez sur la page principale de votre dépôt git où figure des instructions pour pouvoir envoyer vos données dessus.

Sur votre VM installez le paquet git et créez dans votre dossier personnel un dossier dépôt. Dans ce dossier dépôt mettez tous les fichiers et dossiers de votre solution docker compose (vous devriez avoir le fichier docker-compose.yml, le Makefile, le fichier .env si vous avez démarré votre makefile, le dossier nginx et le dossier bind). Rajoutez dans ce dossier un fichier que vous appelerez .gitignore qui permettra d'indiquer à git quels fichiers nous ne voulons pas publier. Dans ce fichier indiquez simplement .env, étant donnée qu'il y a notre mot de passe administrateur dedans).

Placez-vous dans le dossier dépôt et tappez les commandes des sections Git global setup et Push existing folder.

git config --global user.name "przybylo maxime"

git config --global user.email "przybylo.maxime@gmail.com"

Cette section permet d'indiquer à votre serveur linux votre identité sur git (votre mot de passe vous sera demandé quand vous voudrez envoyer des fichiers sur votre dépôt)

git init

git remote add origin https://gitlab.com/m.przybylo/2104.git

git add .

git commit -m "Initial commit"

git push -u origin master

git init permet d'initialiser votre dossier comme dépôt git local.  
  
git remote add origin <url> permet de créer un alias pour votre url de dépôt git. Au lieu de tapper l'url vous pourrez juste indiquer origin à partir de maintenant.   
  
(/!\remplacez l'url https://gitlab.com/m.przybylo/2104.git par celle de votre dépôt !)  
  
git add . indique qu'il faudra ajouter tous le répertoire courant et ses sous dossier lors du prochain envoie sur le dépôt git.  
  
git commit -m "message" permet de préparer l'envoi en indiquant un message précisant quelles modifications ont étés effectués.  
  
git push origin master permet d'envoyer notre dossier sur la branche master de notre dépôt dont l'alias est origin.

Une fois que vous avez saisi votre mot de passe vous pouvez actualiser la page gitlab pour constater que les fichiers y figurent bien.

Maintenant n'importe qui, y compris vous, peut récupérer le contenu de votre dépôt gitlab simplement avec la commande

git clone <url de votre dépôt.git>

Replacez-vous dans votre dossier personnel et tappez la commande :

git clone https://gitlab.com/m.przybylo/cours\_docker.git

Cette commande vous permettra de télécharger mon dépôt git contenant ma solution docker compose correspondante à votre solution avec l'ajout d'un service reverse proxy permettant d'attribuer une url par port exposé de votre serveur web.