Vidéos

20 November 2023 14:53

Déploiement site html Apache

(syll chap 6)

1. Installer le paquet du server web (apache)
   apt-get install apache2
   apt-get install lynx

2. Transférer le code du site web sur le serveur

3. Créer un virtualHost
   cd /etc/apache2/sites-available
   nano test.conf
   <VirtualHost *:80>
   ServerName test
   ServerAdmin webmaster@localhost
   DocumentRoot /var/www/test
   ErrorLog ${APACHE_LOG_DIR}/test_error.log
   CustomLog ${APACHE_LOG_DIR}/test_access.log combined

   <Directory /var/www/test>
   Require all granted
   AllowOverride All
   </Directory>
   </VirtualHost>

4. Activer le site
   a2ensite test.conf
   systemctl reload apache2

5. Faire correspondre la Directive ServerName et /etc/hosts
   nano /etc/hosts
   127.0.0.1 test

6. Tester le site web en local
   lynx http://test

Déploiement terraform
Syllabus chap 13

Putty

1. **Générer une paire de clés avec PuTTYgen** :
   1. Ouvrez PuTTYgen (vous pouvez le télécharger depuis le site officiel de PuTTY si vous ne l'avez pas déjà).
   2. Sélectionnez le type de clé à générer. Pour SSH-2 RSA, sélectionnez "RSA" dans la section "Type de clé à générer".
   3. Cliquez sur le bouton "Générer".
   4. Déplacez votre souris dans la zone vide pour générer de la randomité.
   5. Une fois la clé générée, vous pouvez entrer une passphrase pour sécuriser votre clé dans le champ "Key passphrase". C'est optionnel mais recommandé.
   6. Cliquez sur "Save private key" pour sauvegarder votre clé privée. Vous pouvez également sauvegarder la clé publique en cliquant sur "Save public key".
   7. Copiez la clé publique affichée dans la zone "Public key for pasting into OpenSSH authorized_keys file". Vous l'utiliserez dans la prochaine étape.
2. **Installer la clé publique sur votre machine Debian** :
   1. Connectez-vous à votre machine Debian.
   2. Ouvrez le fichier ~/.ssh/authorized_keys avec un éditeur de texte (créez-le s'il n'existe pas).
   3. Collez la clé publique que vous avez copiée à l'étape précédente à la fin du fichier.
   4. Enregistrez et fermez le fichier.
3. **Se connecter à la machine Debian avec PuTTY** :
   1. Ouvrez PuTTY.
   2. Dans la section "Host Name (or IP address)", entrez l'adresse IP de votre machine Debian.
   3. Dans la section "Connection -> SSH -> Auth", cliquez sur "Browse..." et sélectionnez votre clé privée que vous avez sauvegardée à l'étape 1.
   4. Cliquez sur "Open" pour vous connecter à votre machine Debian.

Déploiement reverse proxy https apache

1. Créez un utilisateur Jetty
   adduser jetty

2. Copier les fichiers du siteJetty dans la home directory de jetty
   mettre le site dans la home directory de l'utilisateur azureuser
   mv siteJetty /home/jetty
   cd /home/jetty
   chown -R jetty siteJetty/
   chgrp -R jetty siteJetty
   su jetty
   cd siteJetty

3. java -jar NoDBRunTest.jar & <= & pour faire tourner le serveur java en arrière plan
   (si java pas installé : sudo apt-get install default-jdk)
4. sur portalAzure, <= si pas Reverse Proxy
   paramètres réseau
   => créer une règle de port
   => Règle de port d'entrée
   => ports de destination : 8080

5. Installer le module ssl
   apt-get install openssl && a2enmod ssl && systemctl restart apache2

6. Installer le module proxy
   a3enmod proxy proxy_http && systemctl restart apache2

7. Créer certificat
   (commande syllabus 6.9.2)

   mkdir /etc/apache2/ssl &&
   openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/apache2/ssl/apache.key -out /etc/apache2/ssl/apache.crt

8. exit (pour retourner en root)
   cd /etc/apache2/sites-available
   nano siteJetty.conf
   <VirtualHost *:443>
   ServerName siteJetty
   ServerAdmin webmatser@localhost
   ProxyPass / http://localhost:8080/
   ProxyPassReverse / http://localhost:8080/
   ErrorLog ${APACHE_LOG_DIR}/monsite_error.log
   CustomLog ${APACHE_LOG_DIR}/monsite_access.log combined
   SSLEngine on
   SSLCertificateFile /etc/apache2/ssl/apache.crt
   SSLCertificateKeyFile /etc/apache2/ssl/apache.key
   </VirtualHost>

   A2ensite siteJetty.conf
   Systemctl reload apache2

9. Lynx https://localhost/

10. Créer une règle de port pour le 443

11. Sur machine distante : https://98.71.237.21

Créer un utilisateur
useradd username
créer un utilisateur avec un repertory /home/username/
useradd -m username
Si on veut des fichier dans le home directory de l'utilisateur
ajouté, on les met au préalable dans le dossier /etc/skel/

Déploiement application docker

1. Documentation

2. Création du Dockerfile
   créer un fichier : Dockerfile
   FROM debian
   RUN apt-get update
   RUN apt-get install apache2 -y
   COPY siteHTML.conf /etc/apache2/sites-available/000-default.conf
   COPY siteHTML /var/www/siteHTML
   EXPOSE 80
   CMD apachectl -D FOREGROUND

3. Création de l'image de conteneur
   cd siteHTMLDocker/
   docker build -t sitehtml .

4. Création d'un conteneur
   docker run -d -p 8090:80 sitehtml

5. Tests- Debug

Déploiement application Docker Compose

1. Documentation
   > models > db_config : charge config.js
   config.js : info connection db
   > intdb : fichier db

2. Création du(des) Dockerfile(s) et docker-compose.yml
   a. Dockerfile
      FROM node:current-alpine3.16
      COPY exoplanets /var/www/exoplanets
      WORKDIR /var/www/exoplanets
      RUN npm install
      EXPOSE 3000
      CMD ["node", "app.js"]

   b. docker-compose.yml
      version: "3.9"
      services:
      app:
      build: .
      ports:
      - 3000:3000
      depends_on:
      postgres_db:
      condition: service_healthy
      postgres_db:
      image: postgres
      environment:
      POSTGRES_PASSWORD: ipl
      POSTGRES_DB: exoplanets
      Volumes:
      - pgdata: /var/lib/postgresql/data
      - ./initdb: /docker-entrypoint-initdb.d
      healthcheck:
      test: ["CMD-SHELL", "pg_isready -U postgres"]
      interval: 30s
      timeout: 30s
      retries: 3
      volumes:
      pgdata:

3. Exécuter
   docker-compose up -d

4. Docker-compose ps

5. Lynx http://localhost:3000