Option 1: Utilisation de Machines Virtuelles (VM)

Pré-requis

- Un hyperviseur comme VirtualBox, VMware, ou KVM
- ISO d'une distribution Linux (par exemple, Ubuntu Server)

Étapes de Création et de Configuration des Machines Virtuelles

1. Création des Machines Virtuelles

Machine Virtuelle 1 (Serveur)

1. **Créer une nouvelle machine virtuelle** :
   - Nom : `Server`
   - Type : Linux
   - Version : Ubuntu (64-bit)
2. **Allouer de la mémoire** (au moins 1 Go)
3. **Créer un disque dur virtuel** (10 Go suffisent)
4. **Configurer le réseau** : Utilisez un réseau interne ou réseau NAT pour la communication entre les VM.

Machine Virtuelle 2 (Client)

1. **Créer une nouvelle machine virtuelle** :
   - Nom : `Client`
   - Type : Linux
   - Version : Ubuntu (64-bit)
2. **Allouer de la mémoire** (au moins 1 Go)
3. **Créer un disque dur virtuel** (10 Go suffisent)
4. **Configurer le réseau** : Utilisez le même type de réseau que pour la VM `Server`.

2. Installation du Système d'Exploitation

Pour chaque machine virtuelle :

1. **Démarrer la VM avec l'ISO d'Ubuntu Server**
2. **Suivre les instructions d'installation** :
   - Choisir l'option "Installer Ubuntu Server"
   - Configurer le réseau (DHCP ou manuel selon vos besoins)
   - Créer un utilisateur et définir un mot de passe
   - Installer OpenSSH pour faciliter l'accès à distance

3. Configuration de la VM Server

1. **Mettre à jour le système** :

```
sudo apt-get update && sudo apt-get upgrade -y
```

2. **Installer Apache et OpenSSL** :

```
sudo apt-get install apache2 openssl -y
```

3. **Activer le module SSL dans Apache** :

```
sudo a2enmod ssl
sudo systemctl restart apache2
```

4. **Générer un certificat auto-signé** :

```
sudo openssl req -x509 -newkey rsa:2048 -keyout /etc/ssl/private/apache-selfsigned.key -out /etc/ssl/certs/apache-selfsigned.crt -days 365 -nodes
```

5. **Configurer un hôte virtuel HTTPS** :

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

Ajoutez la configuration suivante :

```
<VirtualHost *:443>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
    SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key

    <Directory /var/www/html>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

6. **Activer le site et redémarrer Apache** :

```
sudo a2ensite 000-default.conf
sudo systemctl reload apache2
```

4. Configuration de la VM Client

1. **Mettre à jour le système** :

```
sudo apt-get update && sudo apt-get upgrade -y
```

Option 2: Utilisation de Conteneurs avec Docker

Pré-requis

- Docker installé sur votre machine hôte (Windows, macOS, ou Linux).

Étapes de Création et de Configuration des Conteneurs Docker

1. Création du Réseau Docker

```
docker network create mynetwork
```

2. Configuration du Serveur (Conteneur Docker)

Dockerfile pour le Serveur

Créez un `Dockerfile` pour Apache et OpenSSL :

```
FROM ubuntu:20.04

RUN apt-get update && apt-get install -y apache2 openssl

RUN a2enmod ssl

RUN openssl req -x509 -newkey rsa:2048 -keyout /etc/ssl/private/apache-
selfsigned.key -out /etc/ssl/certs/apache-selfsigned.crt -days 365 -
nodes \
    && mkdir -p /etc/ssl/private && mkdir -p /etc/ssl/certs

COPY 000-default.conf /etc/apache2/sites-available/000-default.conf
```

```
EXPOSE 80 443

CMD ["apache2ctl", "-D", "FOREGROUND"]
```

Fichier de Configuration Apache `000-default.conf`

Créez `000-default.conf` dans le même répertoire que le `Dockerfile` :

```
<VirtualHost *:443>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
    SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key

    <Directory /var/www/html>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Construction et Exécution du Conteneur

```
docker build -t apache-ssl .
docker run -d --name apache-server --network mynetwork -p 80:80 -p
443:443 apache-ssl
```

3. Configuration du Client (Conteneur Docker)

Dockerfile pour le Client VPN

Créez un `Dockerfile` pour StrongSwan :

```
FROM ubuntu:20.04

RUN apt-get update && apt-get install -y strongswan

COPY ipsec.conf /etc/ipsec.conf
COPY ipsec.secrets /etc/ipsec.secrets
```

```
CMD ["sleep", "infinity"]
```

Fichiers de Configuration `ipsec.conf` et `ipsec.secrets`

Créez `ipsec.conf` :

```
config setup
    charondebug="ike 2, knl 2, cfg 2"

conn %default
    keyexchange=ikev2
    ike=aes256-sha1-modp1024!
    esp=aes256-sha1!
    dpdaction=clear
    dpddelay=300s
    rekey=no

conn myvpn
    right=apache-server
    rightauth=pubkey
    rightid="C=US, O=MyOrg, CN=server"
    rightsubnet=0.0.0.0/0
    leftauth=pubkey
    leftcert=clientCert.pem
    leftsourceip=%config
    auto=start
```

Créez `ipsec.secrets` :

```
: RSA clientKey.pem
```

Construction et Exécution du Conteneur

```
docker build -t strongswan-client .
docker run -d --name vpn-client --network mynetwork strongswan-client
```

Vérification de la Configuration

Connexion HTTPS

Depuis le navigateur de votre hôte, accédez à `https://localhost` pour voir le site sécurisé par HTTPS.

Connexion VPN

Depuis le conteneur client VPN, vérifiez la connexion VPN avec :

```
docker exec -it vpn-client bash
sudo ipsec statusall
ping 10.10.10.1
```

Conclusion

En suivant ces étapes, vous pouvez configurer et tester des configurations de sécurité telles que HTTPS et VPN IPsec localement en utilisant des machines virtuelles ou des conteneurs Docker. Cela vous permet de créer un environnement sécurisé pour vos tests ou développements.