Création d'un Docker Registry Sécurisé

Pour sécuriser un Docker Registry privé, l'une des étapes les plus importantes consiste à activer le protocole HTTPS, ce qui garantit que la communication entre le client Docker et le démon Docker est chiffrée. Cela permet d'éviter les risques d'écoute ou de manipulation de données sensibles transmises en clair.

Voici les étapes détaillées pour créer un Docker Registry sécurisé en utilisant HTTPS avec un certificat SSL.

---

1. Utilisation du Protocole HTTPS

Pourquoi utiliser HTTPS ?

Le protocole HTTPS (HyperText Transfer Protocol Secure) est essentiel pour sécuriser les communications réseau. Il chiffre les données envoyées et reçues entre le client Docker et le registry Docker, protégeant ainsi les informations sensibles comme les identifiants de connexion et les images Docker.

En utilisant HTTPS avec un certificat SSL, vous vous assurez que :

- Les connexions sont chiffrées.
- Les clients sont authentifiés.
- Les attaques par interception (man-in-the-middle) sont évitées.

Étapes pour activer HTTPS sur un Docker Registry

1. **Préparer le serveur Docker Registry** :
    - Vous devez disposer d'un Docker Registry privé en cours d'exécution. Si ce n'est pas déjà fait, vous pouvez suivre les étapes précédentes pour créer un registry Docker privé de base.
    - Dans cette étape, nous allons configurer le registry pour qu'il écoute en HTTPS.
2. **Générer ou obtenir un certificat SSL**

    Vous pouvez soit acheter un certificat SSL auprès d'une autorité de certification (CA), soit générer un certificat auto-signé pour des tests internes.

---

2. Générer un Certificat SSL pour Docker Registry

Option 1 : Générer un certificat auto-signé avec OpenSSL

Un certificat auto-signé est suffisant pour un usage local ou pour des environnements de développement. Cependant, pour des environnements de production, il est préférable d'obtenir un certificat SSL d'une autorité de certification reconnue.

Étapes pour générer un certificat SSL auto-signé avec OpenSSL :

1. **Créer un répertoire pour stocker les certificats** :
   Vous pouvez stocker le certificat et la clé dans un répertoire dédié, par exemple `/etc/docker/certs`.

   ```
   sudo mkdir -p /etc/docker/certs
   ```

2. **Générer la clé privée** :
   Utilisez OpenSSL pour générer une clé privée RSA.

   ```
   sudo openssl genpkey -algorithm RSA -out
   /etc/docker/certs/domain.key -pkeyopt rsa_keygen_bits:4096
   ```

3. **Générer le certificat SSL auto-signé** :
   Créez un certificat SSL qui correspond à la clé générée. Ce certificat est valide pour 365 jours.

   ```
   sudo openssl req -new -x509 -sha256 -key
   /etc/docker/certs/domain.key -out /etc/docker/certs/domain.crt -
   days 365
   ```

   Pendant cette étape, vous devrez entrer des informations pour le certificat, telles que le pays, l'organisation, etc. Veuillez remplir ces informations en conséquence.

4. **Vérification du certificat** :
   Assurez-vous que votre certificat et votre clé sont correctement générés.

   ```
   ls -l /etc/docker/certs/
   ```

   Vous devriez voir les fichiers `domain.key` et `domain.crt` dans le répertoire `/etc/docker/certs/`.

Option 2 : Utiliser Let's Encrypt pour un certificat SSL gratuit

Si vous souhaitez un certificat SSL valide pour un usage en production, vous pouvez utiliser **Let's Encrypt**, qui propose des certificats SSL gratuits.

1. **Installer Certbot** (outil pour générer un certificat Let's Encrypt) :
   Suivez les instructions de la documentation de Let's Encrypt pour installer **Certbot** sur votre serveur.

Par exemple, sur Ubuntu :

```
sudo apt install certbot
```

2. **Obtenir un certificat SSL** :
   Utilisez Certbot pour générer un certificat SSL valide pour votre domaine. Par exemple :

   ```
   sudo certbot certonly --standalone -d yourdomain.com
   ```

   Cette commande génère un certificat SSL pour votre domaine (remplacez `yourdomain.com` par votre propre nom de domaine). Certbot stocke les fichiers dans `/etc/letsencrypt/live/yourdomain.com/`.

3. **Configurer Docker pour utiliser le certificat SSL** :
   Utilisez les fichiers générés par Let's Encrypt (`fullchain.pem` et `privkey.pem`) pour configurer Docker Registry.

---

3. Configurer Docker Registry pour utiliser HTTPS

Une fois que vous avez généré votre certificat SSL, vous devez configurer votre registry Docker pour qu'il utilise ce certificat et écoute sur le port HTTPS (443).

1. **Créer un fichier de configuration pour Docker Registry**

   Créez un fichier de configuration `config.yml` pour votre registry, si ce n'est pas déjà fait, dans `/etc/docker/registry/` :

   ```
   http:
     secret: a_random_secret_key
     addr: :5000
     headers:
       X-Content-Type-Options: nosniff
     # Configure SSL
     tls:
       certificate: /etc/docker/certs/domain.crt
       key: /etc/docker/certs/domain.key
   ```

   Explication :

   - `certificate` : Chemin vers le certificat SSL.
   - `key` : Chemin vers la clé privée.
   - `addr` : Adresse et port sur lequel le registry écoute (vous pouvez le configurer pour écouter sur le port 443 pour HTTPS).

2. **Lancer le Docker Registry avec HTTPS**

   Maintenant, vous pouvez lancer votre registry Docker avec la configuration HTTPS :

   ```
   docker run -d -p 443:5000 --name registry \
     -v /etc/docker/certs:/certs \
     -v
   /etc/docker/registry/config.yml:/etc/docker/registry/config.yml \
     --restart=always \
     registry:2
   ```

   Ce conteneur écoute maintenant sur le port 443 et utilise HTTPS pour sécuriser les communications avec les clients Docker.

3. **Vérification**

   Pour vérifier que votre registry fonctionne en HTTPS, essayez de vous connecter à l'adresse `https://localhost:5000` (ou `https://<votre-ip>` si vous l'avez configuré sur un autre serveur) via un navigateur ou en utilisant `curl` :

   ```
   curl -k https://localhost:5000/v2/_catalog
   ```

   Le `-k` est utilisé pour ignorer les erreurs liées à un certificat auto-signé (vous pouvez omettre cette option si vous avez un certificat valide).

---

4. Configurer les Clients Docker pour utiliser HTTPS

Si vous utilisez un certificat SSL auto-signé, vous devrez peut-être ajouter le certificat à la configuration Docker pour que le client Docker accepte les connexions HTTPS vers votre registry.

1. **Ajouter le certificat sur le client Docker** :

   - Copiez votre certificat (`domain.crt`) dans le répertoire `/etc/docker/certs.d/localhost:5000/` sur le client Docker.

   Exemple pour un certificat auto-signé :

   ```
   sudo mkdir -p /etc/docker/certs.d/localhost:5000
   sudo cp /etc/docker/certs/domain.crt
   /etc/docker/certs.d/localhost:5000/ca.crt
   ```

2. **Vérification de l'accès sécurisé** :
   Vous pouvez maintenant exécuter les commandes Docker comme `docker pull` ou `docker push` avec une connexion sécurisée au registry.

Résumé

- **HTTPS** sur un Docker Registry est une méthode de sécurisation essentielle pour garantir des communications chiffrées.
- Vous pouvez générer un **certificat SSL auto-signé** pour des environnements de développement ou utiliser **Let's Encrypt** pour obtenir un certificat valide en production.
- Une fois le certificat généré, vous devez configurer votre Docker Registry pour qu'il utilise ce certificat SSL et écoute sur HTTPS.
- N'oubliez pas de configurer les clients Docker pour qu'ils reconnaissent et fassent confiance au certificat SSL utilisé par le registry.

En suivant ces étapes, vous pouvez créer un Docker Registry privé sécurisé qui permet de stocker et de distribuer des images Docker en toute sécurité.