proxy public proxy interne

D'après votre description, vous avez une architecture avec :
1. Un serveur frontal Nginx qui fait office de reverse proxy public
2. Un serveur LAN qui héberge des applications Docker
3. Les applications sont accessibles en interne via `nommachine.local:port_container`

Si vous rencontrez des erreurs lors de l'accès depuis Internet, voici mes recommandations :

Solution recommandée : Oui, installez un second Nginx sur le serveur d'applications

Pourquoi :
1. **Meilleure gestion du routage** : Le proxy frontal devrait parler à un seul endpoint propre (le nouveau Nginx) plutôt qu'à divers ports de containers
2. **Centralisation de la config** : Vous pourrez gérer toutes les règles de routage LAN au même endroit
3. **Résolution des erreurs** : Beaucoup de problèmes viennent des headers HTTP mal propagés (Host, X-Forwarded-For, etc.)

Configuration suggérée :
```
[Internet]
→ Nginx frontal (public IP)
→ Nginx interne (sur le serveur Docker, écoute sur un port fixe)
→ Containers (sur leurs ports respectifs)
```

Avantages :
- Vous pouvez configurer proprement les headers HTTP
- Meilleure gestion du SSL (terminaison au proxy frontal ou interne selon votre besoin)
- Plus simple à maintenir (une seule entrée/sortie sur le serveur Docker)
- Possibilité d'ajouter des règles de routage complexes côté LAN

Alternative :
Vous pourriez tout configurer sur le Nginx frontal, mais cela devient vite complexe si vous avez beaucoup de services et moins sécurisé (exposition des ports internes).

Les erreurs que vous voyez sont probablement dues soit :
1. À des problèmes de résolution DNS interne/externe
2. À des headers HTTP mal propagés
3. À des timeouts entre le proxy frontal et les containers

Une configuration à deux niveaux de proxy est une pratique courante et recommandée dans ce type d'architecture.

La réponse dépend de votre architecture et de vos besoins en sécurité, mais voici les meilleures pratiques :

1. **Solution la plus simple et recommandée (Terminaison SSL unique)**
  - **Seul le proxy frontal gère le SSL**
    - Le trafic entre les deux Nginx (frontal et interne) reste en **HTTP** (LAN)
    - Configuration :
      ```nginx
      # Sur le proxy frontal (public)
      server {
          listen 443 ssl;
          server_name app.votre-domaine.com;
          ssl_certificate /chemin/vers/cert.pem;
          ssl_certificate_key /chemin/vers/key.pem;

          location / {
              proxy_pass http://nommachine.local:80;  # Communication interne non chiffrée
              proxy_set_header Host $host;
              proxy_set_header X-Real-IP $remote_addr;
          }
      }
      ```
    - **Avantages** :
      - Pas besoin de Certbot sur le serveur interne
      - Moins de maintenance (un seul certificat à renouveler)
      - Performances accrues (pas de double chiffrement/déchiffrement)

  - **Quand l'utiliser ?**
    - Si votre LAN est sécurisé (réseau privé fiable, pas besoin de SSL interne).

---

2. **Solution plus sécurisée (SSL end-to-end)**
  - **Proxy frontal ET interne ont leur propre SSL**
    - Le frontal fait une terminaison SSL, puis **reverse proxy vers le second Nginx en HTTPS**
    - Configuration :
      ```nginx
      # Sur le proxy frontal
      location / {
          proxy_pass https://nommachine.local:443;  # Communication interne chiffrée
          proxy_ssl_verify off;  # Désactive la vérification du certificat (optionnel)
      }
      ```

```
    # Sur le proxy interne
    server {
        listen 443 ssl;
        server_name nommachine.local;
        ssl_certificate /chemin/vers/cert-interne.pem;
        ssl_certificate_key /chemin/vers/key-interne.pem;

        location / {
            proxy_pass http://conteneur:port;  # Vers le container Docker
        }
    }
    ```
```

- **Avantages** :
  - Chiffrement même sur le LAN (utile si le réseau n'est pas 100% sécurisé)
  - Meilleure conformité pour certains standards (HIPAA, PCI DSS)

- **Quand l'utiliser ?**
  - Si votre LAN n'est pas totalement sécurisé.
  - Si vous avez des données ultra-sensibles (médical, bancaire, etc.).

---

### 3. **Alternative hybride (SSL entre proxies + HTTP vers les containers)**
- **Proxy frontal en HTTPS → Proxy interne en HTTPS → Containers en HTTP**
  - Moins de charge que le SSL end-to-end, mais sécurise le trafic inter-serveurs.

---

### **Faut-il utiliser Certbot sur le second serveur ?**
- **Oui** si :
  - Vous voulez une sécurité renforcée (HTTPS entre proxies).
  - Le nom de domaine interne (`nommachine.local`) est accessible publiquement (peu probable).
- **Non** si :
  - Le trafic interne est en HTTP (solution 1).
  - Vous utilisez un certificat auto-signé pour la communication interne.

### **Recommandation finale**
Dans **90% des cas**, la **Solution 1 (terminaison SSL unique)** suffit, car :
- Le LAN est considéré comme un environnement de confiance.
- Évite la complexité de gestion de plusieurs certificats.

Si vous optez pour HTTPS interne, un certificat auto-signé peut suffire (pas besoin de Certbot).
Utilisez Certbot uniquement si le second Nginx doit servir un domaine public.

□□ **Vérifiez aussi** :
- Que les headers (`Host`, `X-Forwarded-For`, `X-Forwarded-Proto`) sont bien transmis.
- Que les timeouts (`proxy_read_timeout`, `proxy_connect_timeout`) sont adaptés.
```

- Que le DNS interne (`nommachine.local`) fonctionne correctement entre les serveurs.

Si les erreurs persistent, examinez les logs (`sudo tail -f /var/log/nginx/error.log`) des deux proxies.

Revision #1
Created 2025-08-15 23:00:49 UTC by Nicolas Farrie
Updated 2025-08-15 23:02:02 UTC by Nicolas Farrie