Voici un guide rapide pour mettre en place LiveKit avec Docker, en distinguant bien l'approche "bac à sable" locale et la configuration robuste pour la production.

---

1. Installation en Local (Développement)

Pour tester rapidement sur votre machine, on utilise le mode `--dev` qui ignore les vérifications TLS complexes et utilise des clés par défaut.

Étape A : Créer un `docker-compose.yml` minimal

Créez un dossier `livekit-local` et placez-y ce fichier :

YAML

```
services:
  livekit:
    image: livekit/livekit-server:latest
    command: --dev --config /etc/livekit.yaml
    ports:
      - "7880:7880" # API & Signal
      - "7881:7881" # WebRTC over TCP
      - "7882:7882/udp" # WebRTC over UDP
    volumes:
      - ./livekit.yaml:/etc/livekit.yaml
```

Étape B : Configurer `livekit.yaml`

YAML

```
port: 7880
keys:
  devkey: secret  # Vos identifiants de test
```

Lancement

Bash

```
docker compose up -d
```

> **Accès :** Votre serveur est disponible sur `ws://localhost:7880` avec la clé `devkey` et le secret `secret`.

---

2. Installation en Production (Docker)

En production, WebRTC nécessite impérativement du **HTTPS/WSS** et une gestion fine des ports UDP pour la vidéo/audio. LiveKit fournit un outil pour générer toute la configuration (Docker + SSL via Caddy).

Étape A : Prérequis

- Un **nom de domaine** pointant vers l'IP de votre serveur.

- Ports ouverts sur votre pare-feu :
    - `80`, `443` (TCP) : SSL et Signal.
    - `3478` (UDP) : TURN (indispensable pour passer les pare-feu).
    - `50000-60000` (UDP) : Flux média WebRTC.

Étape B : Utiliser le générateur officiel

La méthode la plus fiable est d'utiliser l'image de génération de LiveKit qui va créer vos fichiers de config, votre Docker Compose et gérer SSL automatiquement :

Bash

```
docker run --rm -it -v$PWD:/output livekit/generate
```

L'assistant va vous poser plusieurs questions :

1. **Domain name** : votre sous-domaine (ex: `livekit.mondomaine.com`).
2. **TURN domain** : souvent le même ou un dédié (ex: `turn.mondomaine.com`).
3. **SSL** : Choisissez "Let's Encrypt".
4. **Redis** : Choisissez "Yes" (recommandé pour la prod).

Étape C : Déploiement

L'outil va générer un dossier portant le nom de votre domaine. Allez dedans et lancez le script d'initialisation :

Bash

```
cd livekit.mondomaine.com
sudo ./init_script.sh
```

Ce script installe Docker (si besoin) et lance le serveur avec une configuration optimisée pour la performance (utilisation du réseau `host` pour éviter le surcoût de NAT de Docker).

Comparatif des configurations

Caractéristique	Local (Dev)	Production
Protocole	`ws://` (Non sécurisé)	`wss://` (SSL obligatoire)
Réseau Docker	Bridge (ports mappés)	`host` (performance WebRTC)
Auth	Clés statiques simples	JWT via clés sécurisées
Traversée NAT	Non requise	Serveur TURN configuré

Exporter vers Sheets

Prochaine étape

Souhaitez-vous que je vous aide à générer un **Token JWT** en Node.js ou Python pour permettre à vos utilisateurs de se connecter à votre nouveau serveur ?