DOCUMENTATION TECHNIQUE

Déploiement d'une Solution Cloud Privée

sur Oracle Cloud Infrastructure

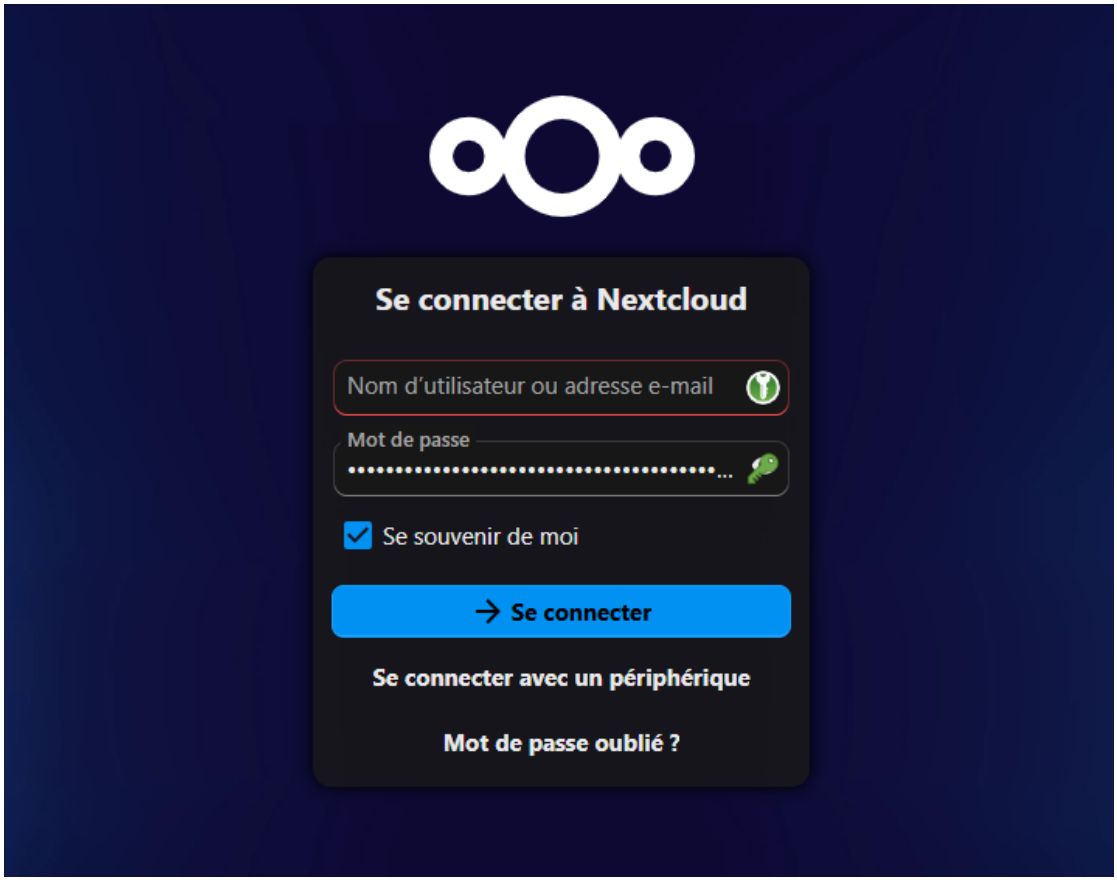
Andrea hardy

contact@andreahardy.fr

Profil	**Administrateur Systèmes & Réseaux**
Projet	Infrastructure Cloud Privée (nextcloud)
Environnement	Oracle Cloud Infrastructure (OCI)
Technologies	Docker, Nginx, MariaDB, Let's Encrypt

VUE D'ENSEMBLE DU PROJET

Déploiement d'une infrastructure cloud privée hautement disponible utilisant une architecture conteneurisée sur Oracle Cloud Infrastructure. La solution intègre un reverse proxy avec gestion automatique des certificats SSL, une base de données relationnelle, et une application web accessible via HTTPS.



Architecture technique

- Frontend : Reverse proxy Nginx avec renouvellement automatique SSL
- Application : Solution de stockage cloud containerisée (nextcloud)
- Base de données : MariaDB 10.6 avec volumes persistants
- Orchestration : Docker Compose pour la gestion multi-conteneurs

1. CONFIGURATION RÉSEAU CLOUD

1.1 Virtual Cloud Network (VCN)

Création d'un réseau virtuel isolé avec subnet public pour l'exposition des services.

Configuration de sécurité (Security Lists)

Règles d'entrée configurées pour autoriser le trafic vers l'infrastructure :

Protocole	Port	Source	Description
TCP	22	0.0.0.0/0	Accès SSH (Administration)
TCP	80	0.0.0.0/0	Trafic HTTP (Redirection SSL)
TCP	443	0.0.0.0/0	Trafic HTTPS (Application)
TCP	81	0.0.0.0/0	Interface Nginx Proxy Manager

1.2 Configuration Pare-feu Local (Firewalld)

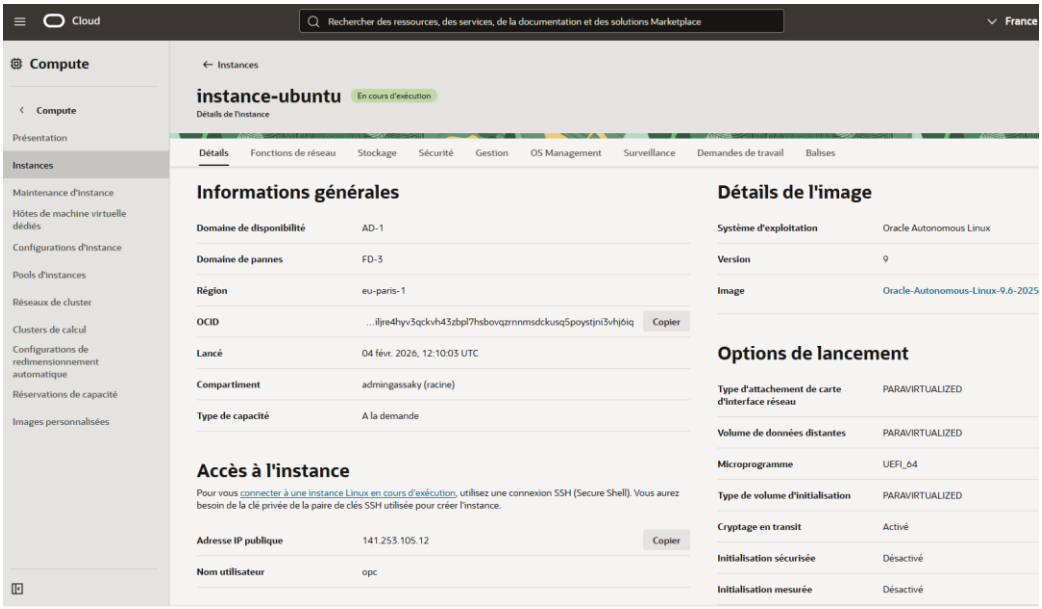
Sur l'instance Oracle Autonomous Linux, le pare-feu interne doit refléter ces ouvertures. Commandes exécutées :

```
sudo firewall-cmd --permanent --add-port=80/tcp

sudo firewall-cmd --permanent --add-port=443/tcp

sudo firewall-cmd --permanent --add-port=81/tcp

sudo firewall-cmd --reload
```

2. PROVISIONING DE L'INSTANCE

Configuration matérielle et système de l'instance cloud :

Shape	**VM.Standard.E2.1.Micro (AMD) / VM.Standard.A1.Flex (ARM)**
Système d'exploitation	Oracle Autonomous Linux 9
Kernel	6.12.0-105.51.5.el9uek.x86_64
Stockage	50 Go (Boot Volume)
Authentification	Paire de clés SSH (RSA)



3. CONFIGURATION DNS

La résolution de nom est gérée via un service DNS externe. Configuration d'un enregistrement de type A :

Type d'enregistrement	**A Record**
Hostname	nextcloud.example.com
Adresse IP	XXX.XXX.XXX.XXX
Vérification	dig cloud.example.com ou ping cloud.example.com

4. STACK APPLICATIVE (DOCKER)

L'installation repose sur Docker pour assurer l'isolation, la portabilité et la reproductibilité de l'environnement.

4.1 Installation des Dépendances Système

Ajout du dépôt Docker officiel et installation des composants :

```
# Ajout du dépôt Docker CE

sudo dnf config-manager --add-repo \

  https://download.docker.com/linux/centos/docker-ce.repo


# Installation de Docker et Docker Compose

sudo dnf install -y docker-ce docker-ce-cli \

  containerd.io docker-compose-plugin


# Activation du service Docker

sudo systemctl enable --now docker


# Ajout de l'utilisateur au groupe docker

sudo usermod -aG docker $USER
```

4.2 Architecture Docker Compose

Fichier de configuration docker-compose.yml définissant l'orchestration des conteneurs :

```
services:

  npm:

    image: 'jc21/nginx-proxy-manager:latest'

    restart: always

    ports:

      - '80:80'   # HTTP

      - '443:443' # HTTPS

      - '81:81'   # Interface Web

    volumes:
```

```
      - ./npm_data:/data
      - ./npm_letsencrypt:/etc/letsencrypt


  db:
    image: mariadb:10.6
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: ${DB_ROOT_PWD}
      MYSQL_DATABASE: cloudapp
      MYSQL_USER: appuser
      MYSQL_PASSWORD: ${DB_PWD}
    volumes:
      - ./db_data:/var/lib/mysql


  app:
    image: nextcloud
    restart: always
    environment:
      MYSQL_HOST: db
      MYSQL_DATABASE: cloudapp
      MYSQL_USER: appuser
      MYSQL_PASSWORD: ${DB_PWD}
      NEXTCLOUD_TRUSTED_DOMAINS: cloud.example.com
    volumes:
      - ./app_data:/var/www/html
```

Démarrage de l'infrastructure

```
# Création du répertoire de travail
```

```
mkdir -p ~/cloudapp && cd ~/cloudapp



# Démarrage de la stack

sudo docker compose up -d
```

5. REVERSE PROXY & SÉCURISATION SSL

La gestion du certificat SSL Let's Encrypt est déléguée à Nginx Proxy Manager pour un renouvellement automatique.

5.1 Accès à l'interface d'administration

- URL : http://XXX.XXX.XXX.XXX:81
- Identifiants par défaut : admin@example.com / changeme

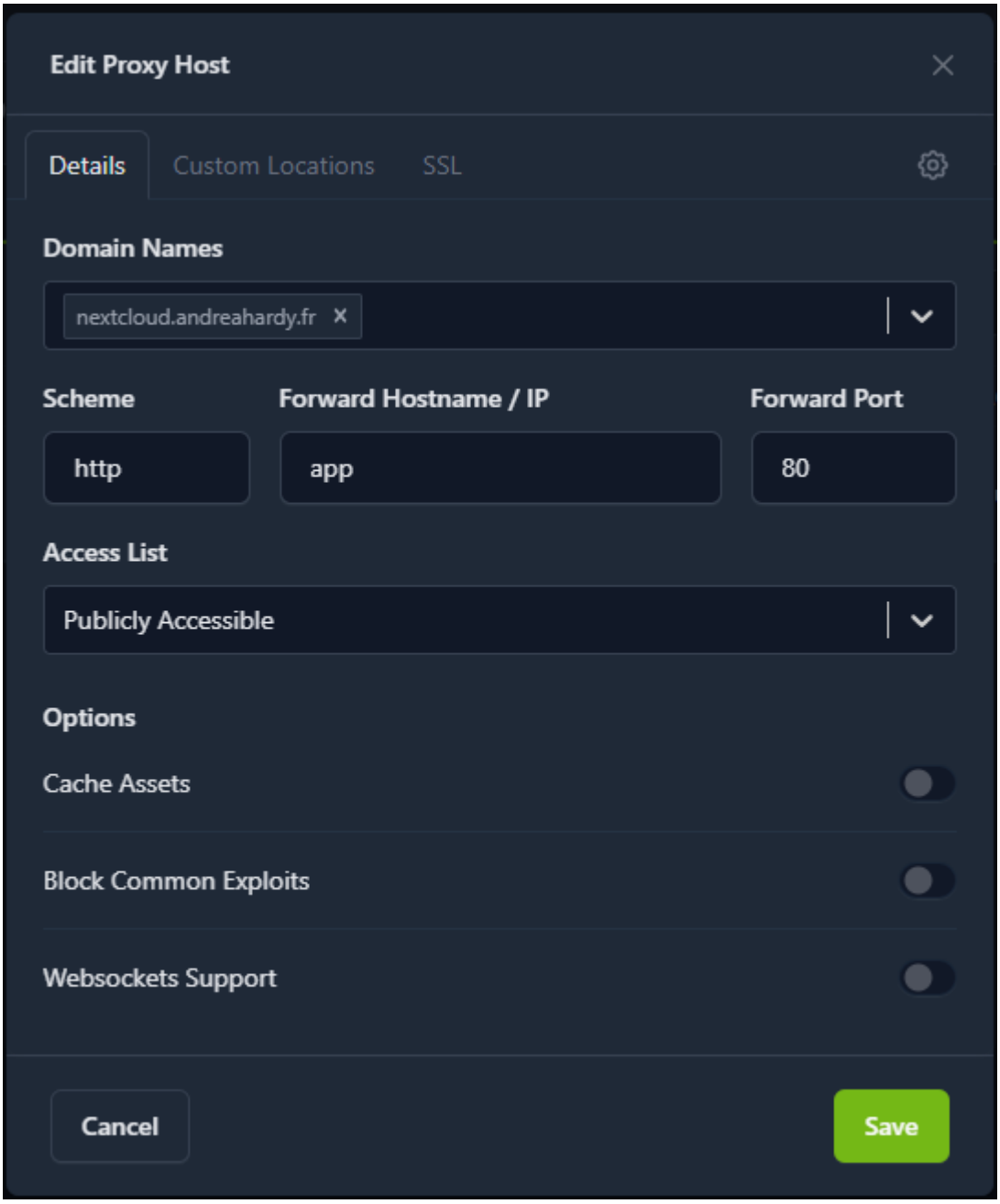
5.2 Configuration du Proxy Host

Domain Names	**cloud.example.com**
Scheme	http
Forward Hostname	app (nom du service Docker)
Forward Port	80

5.3 Configuration SSL

- Activer Request a new SSL Certificate
- Activer Force SSL (redirection HTTP → HTTPS)
- Activer HTTP/2 Support

- Le renouvellement automatique est géré par Let's Encrypt (validité 90 jours)



6. MAINTENANCE & SUPERVISION

6.1 Commandes de gestion

```
# Vérifier l'état des conteneurs

docker compose ps



# Consulter les logs en temps réel

docker compose logs -f app



# Redémarrer la stack complète

docker compose restart



# Arrêter et supprimer les conteneurs

docker compose down



# Mettre à jour les images

docker compose pull

docker compose up -d
```

6.2 Sauvegardes

Les données persistantes sont stockées dans des volumes Docker locaux. Il est recommandé de mettre en place une stratégie de sauvegarde régulière :

- Base de données : ./db_data
- Données applicatives : ./app_data
- Configuration Nginx : ./npm_data
- Certificats SSL : ./npm_letsencrypt

7. POINTS CLÉS

- Architecture conteneurisée assurant portabilité et isolation des services
- Sécurisation HTTPS avec certificats Let's Encrypt à renouvellement automatique

- Infrastructure cloud gratuite optimisée pour Oracle Cloud Infrastructure (Always Free tier)
- Haute disponibilité grâce aux politiques de redémarrage automatique des conteneurs
- Facilité de maintenance via Docker Compose pour la gestion unifiée des services

— Fin du document —