

Déploiement sur Microsoft Azure

Application Web PHP & SQL

Par : El Akhal Mohamad

Notre Mission Commune

- **Le Projet** : Déployer une application web (PHP/SQL) en production.
- **Le Contexte** : Une comparaison entre 3 grands fournisseurs cloud.
- **Mon Choix : Microsoft Azure** ☁️

Aujourd'hui, je vous montre comment j'ai construit une infrastructure **sécurisée, haute-disponibilité** et **entièrement automatisée** pour notre application.

La Philosophie : DevOps & CI/CD

Pour un déploiement moderne, on ne clique plus sur des boutons, on automatise !

- **DevOps** : Une culture pour livrer de la valeur plus **vite** et de manière plus **fiable**.
- **CI/CD (Intégration et Déploiement Continu)** : Le moteur de cette culture.
 1. **CI (Intégration)** : Le code est testé et "empaqueté" (image Docker) automatiquement.
 2. **CD (Déploiement)** : L'application est mise en production sans intervention manuelle.

L'Architecture Cible sur Azure

Voici le plan de bataille. Chaque composant a été choisi pour une raison précise : **Sécurité**, **Scalabilité** et **Résilience**.

1. Les Fondations : Réseau & Sécurité

- **VNet (CloudProject-VNet)**
Notre réseau privé et 100% isolé dans le cloud.
- **Segmentation en Subnets**
 - **Public-Subnet** : Pour les composants face à Internet (nos serveurs web).
 - **Private-Subnet** : Pour les joyaux de la couronne (la

2. Le Web : Scalable & Résilient

- **VM Scale Sets (VMSS)**
Plutôt que des VMs isolées, on utilise un groupe de VMs identiques.
 - **Auto-scaling** : Ajoute des serveurs si le trafic augmente, les retire si ça se calme. **Optimisation des coûts et performance garantis !**

3. Le Trafic et les Données

- **Application Gateway (Le diamant 💎)**
Notre portier intelligent. Il reçoit le trafic (port 80) et le distribue aux VMs saines. Il gère la répartition de charge.
- **Azure SQL Database (PaaS)**
Une base de données managée. **Pourquoi ?**
✅ Pas d'OS à gérer

4. L'Accès : Sécurisé par le Bastion




- **Le Problème** : Comment accéder à nos VMs privées pour la maintenance sans ouvrir de ports (SSH) sur Internet ?
- **La Solution : Azure Bastion**



C'est un "pont" sécurisé. Je me connecte en HTTPS au portail Azure et **Bastion**

5. L'Automatisation : Le Pipeline en Action

Mon pipeline `.gitlab-ci.txt` est le chef d'orchestre :

1.  **Scan de Sécurité** : `gitleaks` cherche des secrets et `SAST` des vulnérabilités dans le code. **La sécurité d'abord !**
2.  **Build** : Le `Dockerfile` crée une image de notre application (PHP, Apache, drivers SQL) et la stocke dans le registre GitLab.
3.  **Deploy** :
 - Le Runner GitLab se connecte de manière sécurisée au **Bastion**.
 - Il utilise le Bastion comme **relais** pour envoyer un script de déploiement à chaque VM du Scale Set.

Synthèse : Pourquoi Azure ?

Objectif	Solution Implémentée
Sécurité Maximale	VNet, Subnets, Azure Bastion , Private Link
Haute Disponibilité	Availability Zones , VM Scale Sets, Azure SQL
Scalabilité	VM Scale Sets (Auto-scaling)
Automatisation	GitLab CI/CD via le Bastion, Docker
Coûts Optimisés	Auto-scaling (on ne paie que ce qu'on utilise)
Maintenance Réduite	Services PaaS (Azure SQL, Bastion, App Gateway)

Merci !
Questions ?