Documentation Technique : Module Monitoring & Alerting

Projet : Système de Détection de Fraude Bancaire (DigitalBank)

Version	1.0
Date	22 Janvier 2026

1. Vue d'ensemble

Ce module assure la surveillance en temps réel de l'API de détection de fraude. Il collecte des métriques techniques et métier, visualise l'état du système via des tableaux de bord, et notifie l'équipe de sécurité instantanément via Discord en cas d'attaque détectée.

Stack Technologique

Composant	Description
Prometheus	Collecte et stockage des métriques (Time Series Database)
Alertmanager	Gestion des alertes et routage vers Discord
Grafana	Visualisation des données (Dashboards)
Docker Compose	Orchestration des conteneurs
cAdvisor / Node Exporter	Surveillance des ressources système (CPU, RAM)

2. Architecture du Pipeline

Le flux de surveillance suit les étapes suivantes :

1. **Génération** : L'API Flask expose des métriques sur `/metrics` (nombre de requêtes, score de fraude moyen, latence).

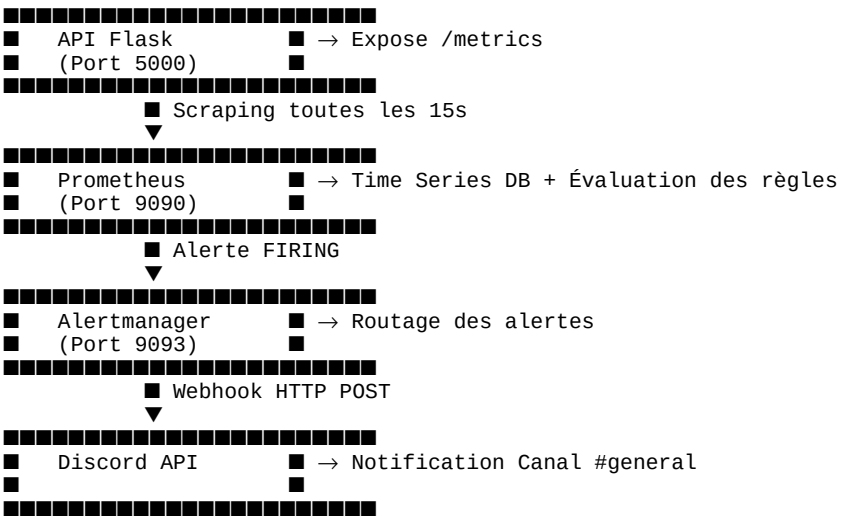
2. **Scraping** : Prometheus interroge cette page toutes les 15 secondes.

3. **Évaluation** : Prometheus compare les données aux règles définies dans `alerts.yml`.

4. **Déclenchement** : Si une règle est violée (ex: Score > 0.5), l'état passe à `FIRING`.

5. **Notification** : Alertmanager reçoit l'alerte et l'envoie via Webhook au canal Discord `#general`.

Vue d'ensemble du flux


API Flask
(Port 5000)
→ Expose /metrics
Scraping toutes les 15s
Prometheus
(Port 9090)
→ Time Series DB + Évaluation des règles
Alerte FIRING
Alertmanager
(Port 9093)
→ Routage des alertes
Webhook HTTP POST
Discord API
→ Notification Canal #general

3. Configuration des Alertes

Le cœur du système repose sur la règle de détection d'attaque massive.

Fichier : `monitoring/prometheus/alerts.yml`

Cette règle est configurée pour une réactivité maximale (démo) :

```
groups:
  - name: fraud_detection_alerts
    rules:
      - alert: HighFraudActivity
        # Déclenchement si le score moyen de fraude dépasse 0.5
        expr: fraud_score_average > 0.5
        # Durée de confirmation (rapide pour la démo)
        for: 5s
        labels:
          severity: critical
        annotations:
          summary: "■ FRAUDE DÉTECTÉE SUR DIGITALBANK !"
          description: "Le modèle détecte une attaque en cours (Score > 0.5)."
```

Fichier : `monitoring/alertmanager/alertmanager.yml`

Configuration de l'intégration native Discord (évitant les erreurs de format Slack) :

```
route:
  receiver: 'discord_webhook'

receivers:
- name: 'discord_webhook'
  discord_configs:
  - webhook_url: 'https://discord.com/api/webhooks/YOUR_WEBHOOK_URL'
    title: '■ ALERTE BANQUE : Fraude Détectée'
```

4. Guide d'Utilisation et Tests

Démarrage de la Stack

Le système est conteneurisé. Pour lancer l'environnement complet :

```
cd monitoring
sudo docker compose up -d
```

Vérification de l'État

• **Prometheus** : `http://localhost:9090` (Vérifier onglet "Alerts")
• **Grafana** : `http://localhost:3000` (Login: `admin` / `admin`)

Simulation d'Attaque (Script de Test)

Pour valider le déclenchement des alertes, nous utilisons un script Bash qui simule une vague de transactions frauduleuses (Gros montant, Crypto, Nigeria).

Script `test_attack.sh` :

```
#!/bin/bash
echo "■ Démarrage de la simulation d'attaque..."

# Envoi de 30 transactions frauduleuses en boucle
for i in {1..30}; do
  echo "■■ Envoi transaction suspecte n°$i"
  curl -X POST http://localhost:5000/predict \
       -H "Content-Type: application/json" \
       -d '{"features": [9500, "Cryptocurrency", "Nigeria", 3]}'
  sleep 1.5
done

echo "■ Simulation terminée."
```

5. Résultats Attendus

Lors de l'exécution du script de test, le comportement suivant est observé :

1. **Dans Prometheus** : L'alerte `HighFraudActivity` passe de l'état INACTIVE à PENDING (jaune), puis FIRING (rouge).

2. **Dans Discord** : Une notification est reçue instantanément avec le titre "■ ALERTE BANQUE : Fraude Détectée".

3. **Dans Grafana** : Le graphique "Fraud Probability" montre un pic atteignant le seuil critique.

Dépannage Rapide (Troubleshooting)

• **Erreur** `Permission denied` **sur Docker** : Redémarrer le service (`sudo systemctl restart docker`).

• **Pas de notification Discord ?** Vérifier les logs (`sudo docker compose logs alertmanager`) et s'assurer que la configuration utilise `discord_configs` et non `slack_configs`.