Commandes basiques en mode terminal : A réaliser aussi dans un script !

Alerting : tests

Version synthétique :
cd : curl http://localhost:9091/metrics

Quelques **commandes alternatives filtrées** selon les besoins :

🔹 **1. N'afficher que les 20 premières lignes (aperçu global)**
cd : curl -s http://localhost:9091/metrics | head -n 20

🔹 **2. Ne récupérer que les lignes contenant un mot-clé (ex. : cpu)**
cd : curl -s http://localhost:9091/metrics | grep 'cpu'

🔹 **3. Lister uniquement les noms des métriques (sans valeurs)**
cd : curl -s http://localhost:9091/metrics | grep -v '^#' | cut -d' ' -f1 | sort -u

🔹 **4. Afficher les métriques contenant une valeur spécifique (ex. : node_memory ou up)**
cd : curl -s http://localhost:9091/metrics | grep 'node_memory'
ou :
cd : curl -s http://localhost:9091/metrics | grep '^up'

🔹 **5. Nombre total de métriques exposées**
cd : curl -s http://localhost:9091/metrics | grep -v '^#' | wc -l

Ceux-sont des métriques Prometheus, majoritairement liées à l'environnement **Go Runtime** (métriques internes à Prometheus). Il faut des métriques plus pertinentes sur les **services supervisés** (comme **node_exporter, blackbox_exporter, etc.**).

Commandes ciblées :

🔸 **1. Pour vérifier l'état de tes cibles (via up)**
cd : curl -s http://localhost:9091/metrics | grep '^up'

- Résultat : up{job="node_exporter", instance="localhost:9100"} 1
- Signification : 1 = OK, 0 = HS

🔸 **2. Pour extraire des métriques node_exporter (CPU, mémoire, disque)**
cd : curl -s http://localhost:9091/metrics | grep 'node_cpu_seconds_total' | head -n 5
cd : curl -s http://localhost:9091/metrics | grep 'node_memory' | head -n 5

🔸 **3. Pour tester les résultats d'un module Blackbox (HTTP par ex.)**
cd : curl -s http://localhost:9091/metrics | grep 'probe_success'

- Cela te dira si les sondes (HTTP/ICMP/TCP) renvoient 1 (succès) ou 0 (échec).

🔸 **4. Pour n'afficher que les noms des métriques disponibles**

cd : curl -s http://localhost:9091/metrics | grep -v '^#' | awk '{print $1}' | sort -u

🔸 **5. Version synthétique personnalisée dans un fichier**

cd :²a²curl -s http://localhost:9091/metrics | grep -E 'up|probe_success|node_memory|node_cpu_seconds' > mini-metrics.txt

Affichage des alertes en console ou via Grafana (mail/SMS à activer) voir annexe Monitoring

7. Résultats observés

- Toutes les sondes actives
- Réception des alertes sur latence ou coupure réseau
- Bonne visibilité des métriques dans Grafana

8. Problèmes rencontrés

- Conflit utilisateur prometheus non système → solution : suppression manuelle + purge dpkg.

9. Outils CLI utilisés

curl http://localhost:9091/metrics
curl -s http://localhost:9091/api/v1/targets | jq
promtool check config /etc/prometheus/prometheus.yml

10. Bonnes pratiques

- Toujours vérifier les fichiers avec **promtool**
- Redémarrer avec **systemctl daemon-reexec** après modification de services
- Grouper les cibles par **job_name** explicite
- Annoter les **dashboards** pour suivi opérationnel