Le monitoring (ou surveillance en français) est le processus de collecte, d'analyse et de visualisation des données sur les systèmes et les applications informatiques afin de s'assurer de leur bon fonctionnement et de leur performance. Cela permet de détecter et de diagnostiquer rapidement les problèmes, d'optimiser les performances et de garantir la disponibilité des services.

Dans le contexte des applications basées sur les architectures microservices, le monitoring peut inclure plusieurs aspects :

**1. Santé des services individuels**

Chaque microservice doit être surveillé individuellement pour garantir sa disponibilité et sa performance :

* **Disponibilité :** Vérifier régulièrement si le service répond aux requêtes.
* **Health checks :** Des endpoints dédiés peuvent être utilisés pour effectuer des tests automatisés afin de vérifier l'état du service (par exemple, /health endpoint).

**2. Performance des services**

La performance de chaque microservice est essentielle pour maintenir une expérience utilisateur optimale :

* **Temps de réponse :** Mesurer le temps nécessaire pour qu'un microservice réponde à une requête. Idéalement, ces mesures incluent les temps de traitement et les temps de latence réseau.
* **Taux de requêtes :** Le nombre de requêtes que chaque microservice reçoit et traite sur une période donnée. Cela permet de détecter les variations de charge et d'identifier les potentiels goulets d'étranglement.

**3. Utilisation des ressources**

Surveiller l'utilisation des ressources est crucial pour optimiser l'efficacité et prévenir les pannes :

* **CPU et mémoire :** Monitorer l'utilisation du processeur et de la mémoire par chaque instance de microservice pour identifier les besoins en ressources et les éventuels problèmes de performance.
* **Réseau :** Surveiller la bande passante utilisée par chaque microservice, ainsi que les délais et la latence réseau.
* **Stockage :** Vérifier l'utilisation et la disponibilité du stockage utilisé par chaque microservice, surtout dans les environnements où le stockage est partagé.

**4. Erreurs et exceptions**

La détection rapide des erreurs est essentielle pour minimiser les temps d'arrêt et améliorer la fiabilité du système :

* **Logs :** Collecter, analyser et alerter sur les erreurs et les exceptions rencontrées par chaque microservice. L'agrégation de logs permet de retracer les problèmes et de diagnostiquer les causes profondes des incidents.
* **Monitoring des erreurs :** Utiliser des outils qui permettent de surveiller les taux d'erreurs, les types d'erreurs les plus fréquents, et les tendances d'erreur au fil du temps.

**5. Tracing distribué**

Dans un environnement microservices, le tracing distribué aide à comprendre le parcours des requêtes à travers différents microservices :

* **Corrélation des requêtes :** Suivre le parcours d'une requête à travers plusieurs microservices pour identifier les goulots d'étranglement et optimiser les performances.
* **Latence entre services :** Mesurer le temps passé dans chaque microservice lors du traitement d'une requête pour identifier les microservices ou les appels à des services externes qui ralentissent le traitement.

**6. Sécurité**

Assurer la sécurité des microservices et des données échangées est primordial :

* **Monitoring des accès et des identités :** Surveiller les tentatives d'accès non autorisées et les anomalies de comportement qui pourraient indiquer une violation de la sécurité.
* **Intégrité des données :** Vérifier que les données manipulées par les microservices ne sont pas altérées ou compromises.

**7. Monitoring des intégrations externes**

Les microservices peuvent dépendre d'intégrations avec des services externes :

* **Monitoring des APIs externes :** Surveiller les performances et la disponibilité des API externes utilisées par les microservices.
* **Latence des appels externes :** Mesurer le temps de réponse des appels aux services externes pour identifier les dépendances critiques et les possibles points de défaillance.

**8. Expérience utilisateur**

Assurer une bonne expérience utilisateur est l'objectif ultime du monitoring des microservices :

* **End-User Monitoring (EUM) :** Mesurer les temps de chargement des pages, les interactions utilisateur, et les métriques de performance depuis le point de vue des utilisateurs finaux pour identifier les problèmes qui impactent directement leur expérience.

**Outils et technologies recommandés :**

Pour mettre en œuvre un monitoring efficace dans un environnement de microservices, les outils suivants sont souvent utilisés :

* **Prometheus :** Pour la collecte de métriques et le monitoring des performances.
* **Grafana :** Pour la visualisation et l'analyse des données de monitoring.
* **ELK Stack (Elasticsearch, Logstash, Kibana) :** Pour la gestion des logs et le troubleshooting.
* **Zipkin :** Pour le tracing distribué des requêtes.
* **Kubernetes Dashboard :** Pour surveiller les clusters Kubernetes et les ressources utilisées par les microservices.