Étude de cas - OpenSearch

Étudiant : Samuel Richard

Cours : LOG430

Groupe : 01

Professeur : Fabio Petrillo

Remis le 2025-05-12

1. Introduction

1.1 Contexte industriel d'un moteur de recherche distribué

Les moteurs de recherche distribués constituent une composante essentielle dans le domaine de l'analyse et de l'exploration de données à grande échelle. Ces systèmes permettent de stocker, d'indexer et d'interroger efficacement d'importants volumes de données réparties sur plusieurs serveurs, en assurant à la fois rapidité d'accès et résilience.

Ils sont aujourd'hui déployés dans divers secteurs où les enjeux liés à la recherche rapide, à la scalabilité et à la tolérance aux pannes sont cruciaux. Parmi ceux-ci, on retrouve la santé (analyse de données cliniques massives), le commerce électronique (moteurs de recherche internes), ou encore les infrastructures de calcul haute performance, où ils facilitent l'exploration de traces d'exécution et de journaux applicatifs. Ces moteurs permettent ainsi aux organisations de mieux comprendre, diagnostiquer et optimiser leurs systèmes et leurs services.

1.2 Système distribué choisi

OpenSearch est un moteur de recherche et d'analyse distribué et libre, issu d'un projet de fourche d'Elasticsearch. Il présente des fonctionnalités de recherche de texte intégral, d'analyse de performance, d'analyse de données structurées et non-structurées, ainsi que de détection d'anomalies. La version 3.0 d'OpenSearch vient d'ailleurs d'être publiée en mai 2025, apportant de nombreuses améliorations en lien avec les bases de données vectorielles et la gestion de données dans l'optique de répondre aux besoins des applications d'intelligence artificielle modernes.

Selon Amazon, on retrouve parmi les utilisateurs du *Amazon OpenSearch Service* des entreprises telles que Atlassian, Autodesk, Prime Video, Pearson et AirBnb, pour ne nommer que celles-ci. Bien que le déploiement d'OpenSearch soit généralement interne à l'infrastructure au service de l'entreprise, son utilisation peut effectivement atteindre des utilisateurs finaux. Par exemple, le moteur de recherche d'Amazon.com utilise OpenSearch pour fournir des résultats de recherche pertinents et rapides aux clients.

Dans un autre contexte, OpenSearch peut être utilisé pour l'analyse de journaux et la surveillance des performances dans des applications critiques, comme dans un centre de calcul haute performance, ou encore être intégré dans des systèmes d'authentification et de sécurité pour analyser les comportements des utilisateurs et détecter les anomalies.

2. Motivation pour le choix du système

Critères de choix du système :

- Pertinence dans le cadre du cours LOG430 - Architecture logicielle
- Attributs de qualité pertinents qui puissent être évalués et explorés

L'intérêt immédiat d'OpenSearch est parti du fait qu'il est implémenté en Java, un langage que je maîtrise bien. De plus, OpenSearch est une solution implantée chez Calcul Québec, où je travaille, ce qui me permet d'avoir accès à des ressources et à une expertise dans le cadre de mon étude de cas.

Le projet est pertinent pour le cours LOG430, puisqu'il s'agit d'un système distribué éprouvé, doté d'une architecture bien définie et abondamment documentée. Cela permet d'analyser en profondeur plusieurs attributs de qualité, essentiels à l'architecture d'un moteur de recherche distribué.

Parmi ceux-ci, on pourrait évaluer :

- La performance (*performance*) : OpenSearch est conçu pour traiter des requêtes complexes sur de grands volumes de données en temps réel, ce qui est essentiel pour les applications nécessitant une réponse rapide.
- La graduation (*scalability*) : OpenSearch est conçu pour être déployé dynamiquement sur plusieurs nœuds, permettant ainsi de gérer des volumes de données croissants sans compromettre les performances.
- La flexibilité (*flexibility*) : le système peut être déployé sur des infrastructures variées, allant des ordinateurs portables aux clusters de serveurs, facilitant d'ailleurs son déploiement dans le cadre de l'étude de cas.
- La résistance aux pannes (*fault tolerance*) : OpenSearch utilise des mécanismes de réplication et de partitionnement pour garantir que les données restent accessibles même en cas de défaillance d'un ou plusieurs nœuds.
- La sécurité (*security*) : le système intègre des fonctionnalités avancées telles que l'authentification, l'autorisation, et le chiffrement des données, tant au repos qu'en transit. Il convient de noter que ces aspects relèvent de l'infrastructure logicielle, et ne doivent pas être confondus avec les fonctionnalités d'analyse de sécurité (telles que la détection d'anomalies), qui relèvent davantage du traitement des données.

3. Objectifs de l'étude de cas

Cette étude de cas vise à explorer l'architecture d'OpenSearch à travers deux axes applicatifs émergents : l'intégration de l'apprentissage machine, notamment via les bases de données vectorielles, et l'analyse automatisée de la sécurité, incluant la détection d'anomalies et la journalisation d'événements.

L'un des usages récents mis de l'avant par OpenSearch est la possibilité d'employer la recherche vectorielle dans un contexte d'intelligence artificielle. Le système propose une bibliothèque de cas d'utilisation illustrant ces fonctionnalités, avec des modèles préentraînés, mais aussi la possibilité d'intégrer des modèles tiers.

Un autre domaine d'application inattendue lors de l'étude préliminaire est l'analyse de la sécurité. OpenSearch propose des fonctionnalités avancées pour la détection d'anomalies, la surveillance des journaux et l'analyse des menaces. Des cas d'études existent déjà à cet effet, comme celui de Pearson, démontrant un usage concret et éprouvé dans un contexte industriel.

Il semble également que ces deux concepts puissent être combinés, permettant ainsi d'utiliser des modèles d'apprentissage machine pour détecter des anomalies dans les données de sécurité. Cela semble cependant être un cas peu documenté, et je n'ai pas trouvé d'exemple concret d'implémentation.

À partir de ces constats, mon objectif est de comprendre dans quelle mesure l'architecture d'OpenSearch supporte ces cas d'usage avancés, à la fois du point de vue des capacités logicielles (ex. : ingestion de vecteurs, extension de modèles ML) et de la résilience et de la graduation de l'architecture distribuée.

Deux questions principales guideront mon étude :

- De quelle manière OpenSearch intègre-t-il des modèles d'apprentissage machine tiers dans son architecture ?
- Quels mécanismes d'ingestion et de traitement rendent possible l'analyse de sécurité à grande échelle dans un système distribué comme OpenSearch ?