ELK

The Elastic Stack

CSP Informatique

Daniel Lavoie - dlavoie@live.ca

Plan du cours

Introduction et vue d’ensemble
Elasticsearch
Logstash
Kibana

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Introduction et vue d’ensemble

L’écosystème d'Elastic Stack

Cas d’utilisations

Exemples d’architectures

The Elastic Stack V5

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

L’écosystème d’Elastic Stask

elasticsearch

logstash

beats

kibana

CSP Informatique

Daniel Lavoie - dlavoie@live.ca

L’écosystème d’Elastic Stack

Elasticsearch

Stockage des données

Moteur de recherche distribué Full-Text

Logstash / Beats

Pipeline d'intégration des données.

Kibana

Plateforme graphique de visualisation des données.

CSP Informatique

Daniel Lavoie - dlavoie@live.ca

Cas d’utilisations

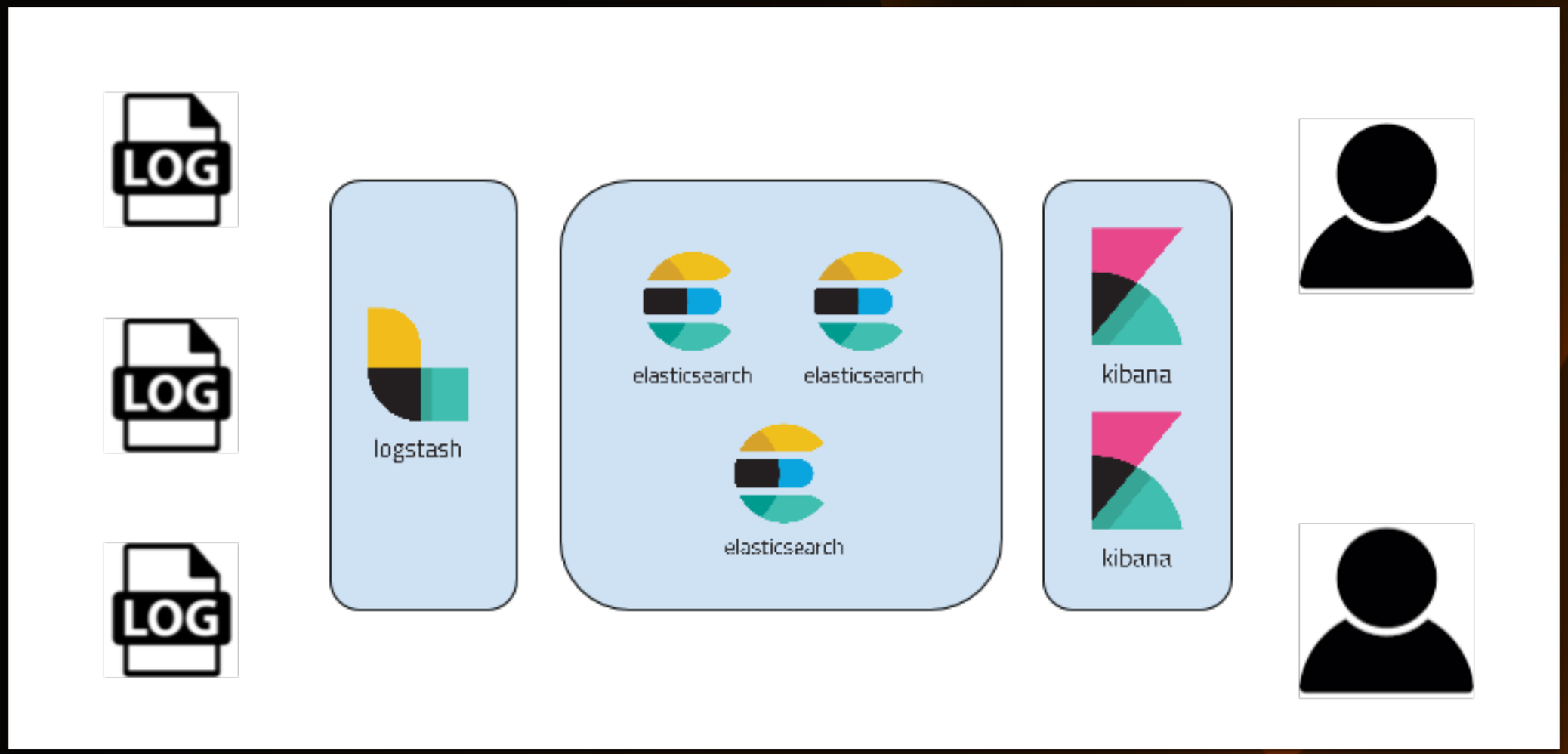
Centralisation des logs d’une plateforme.

Analytique temps réel sur un jeu de donnée dé-normalisé.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Exemple d’architecture

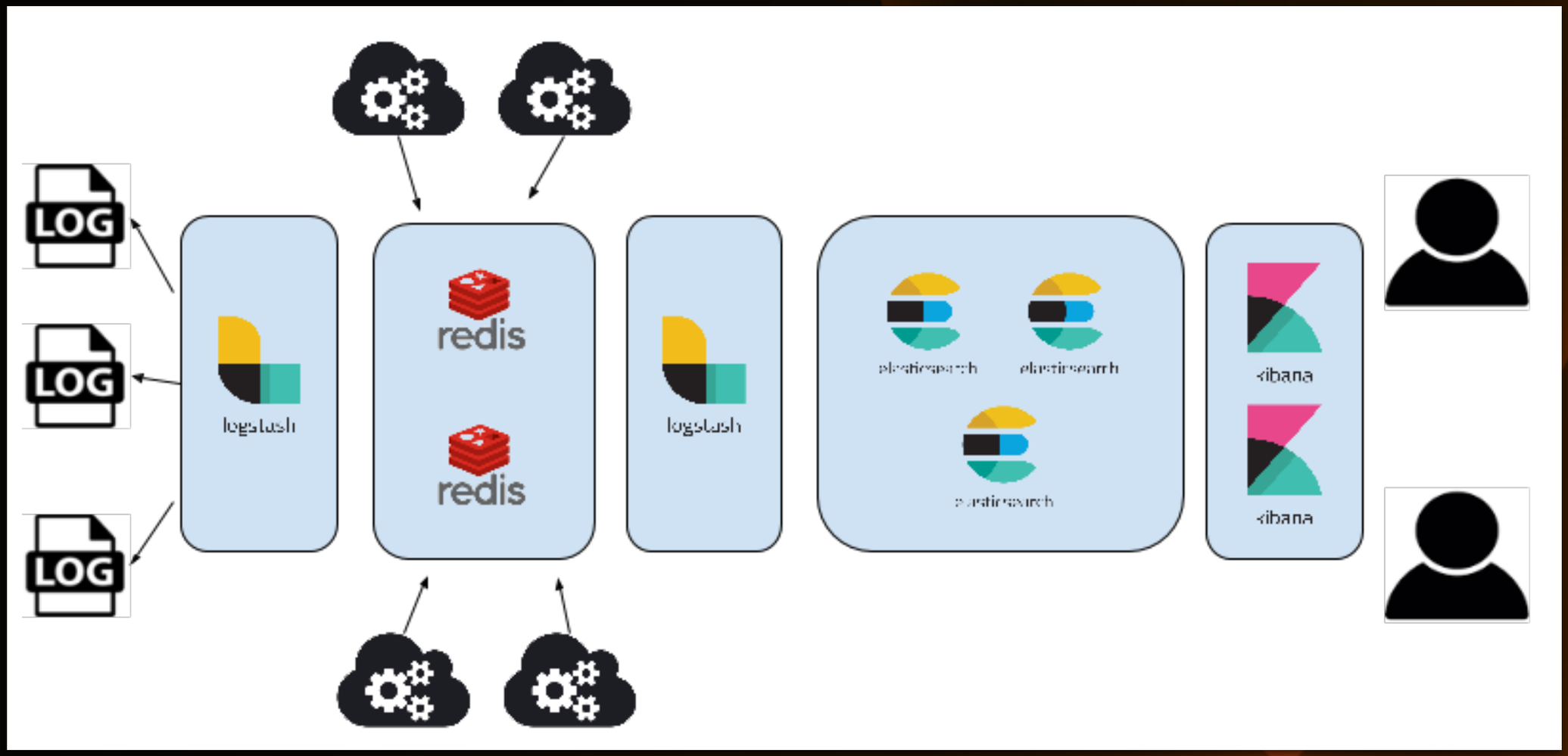
Architecture simple



CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Exemple d’architecture

Architecture découplé



CSP Informatique
Daniel Lavoie - dlavoie@live.ca

The Elastic Stack V5

Mise à niveau de la nomenclature de tous les composants.

Publication simultané de tous les composants de la stack.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Elasticsearch

Introduction à Elasticsearch

Pratique - Installation

Configurations à connaitre

Indexation et recherche

Pratique - Recherche

Insertions

Pratique - Insertions

Mappings et analyse

Agrégations

Pratique - API Java

Prêt pour la production

Gestion de la sécurité

CSP Informatique

Daniel Lavoie - dlavoie@live.ca

Introduction à Elasticsearch

- Orienté document
- Coeur d’Elasticsearch
- Les API disponibles
- Composition d’un cluster

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Orienté document

Document stocké dénormalisé au format JSON.

Tous les champs de la donnée JSON sont indexé et trié par défaut.

Documents stockés immuables.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Coeur d'Elasticsearch

Basé sur la librairie Lucene.

Cache la complexité de Lucene pour offrir un système distribué simple et efficace

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Elasticsearch - API

Deux protocoles de communication

Natif : Client Java avec 2 modes de communication

Node Client : Client se joignant au cluster

Transport Client : Client se reposant sur 1 noeud pour toute communication

HTTP : Api polyglot RESTfull

Utilisé par les clients non Java. Les outils d'administrations se repose généralement sur l'API HTTP.

CSP Informatique

Daniel Lavoie - dlavoie@live.ca

Node Client vs Transport Client

Le Node Client est membre à part entière du Cluster. Par conséquent, il connait la topologie complète de la plateforme. Son coup en resource est plus élevé par contre ses requêtes sont optimisé car elles sont dirigé directement vers les noeuds qui contiennent les données qui intéressent la requête exécuté. Ce type de connexion est à privilégier pour des applications qui seront connecté en permanence et qui exécuteront énormément de requêtes.

Le Transport Client quant à lui, s'adresse à un noeud pour chacune de ses requêtes. Le noeud qui reçoit sa requête aura la responsabilité de redirigé vers les noeuds concernés. Les resources nécessaire à ce client sont moindre car il n'a pas a connaitre toute la composition du cluster. A privilégier pour des clients qui se connecte de manière irrégulière sur demande.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Composition d’un cluster

Noeud (Node)

- Un noeud correspond à une instance d’Elasticsearch.
- Dans un cluster, chaque noeud est égal.
- Aux yeux d'un client, tout noeud peut répondre à tout type de demande.
- L'ajout ou la suppression de noeud est transparent. Il suffit de démarrer ou d’arrêter les noeuds en questions.

Daniel Lavoie - dlavoie@live.ca

Composition d’un cluster

Index

- Unité logique dans laquelle on va insérer les données à indexer.
- Divisé en shard qui sont répartis à travers les différents noeuds.
- Supporte des données n’ayant pas la même composition.
- Equivalent de la table SQL.

Daniel Lavoie - dlavoie@live.ca

Composition d’un cluster

Shard

- Représente un bloc des données d’un index.
- Deux type de shard : *Primary* et *Réplica*
- Techniquement, un shard représente une instance de Lucene.
- Par défaut, un shard primary se doit toujours d’avoir un réplica présent sur un autre noeud.
- Les Réplicas peuvent répondre aux requêtes de lecture afin d’alléger les IO sur le shard primaire.

Démonstration

Création d’un Cluster avec ajout de noeud progressif

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Pratique - Installation

https://github.com/daniellavoie/elk-training/install/download-install.md

Daniel Lavoie - dlavoie@live.ca

Pratique - Installation d’un cluster

https://github.com/daniellavoie/elk-training/install/install-cluster.md

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Pratique - Installation du plugin Kopf

https://github.com/daniellavoie/elk-training/install/install-kopf.md

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Gestion de la configuration

Les configurations d'Elasticsearch sont applicable de deux manières :

Arguments au lancement

Toutes les configurations prévus pour le système peuvent être passé par paramètre de JVM.

Ex : elasticsearch -Dcluster.name=superb-cluster

Fichier de configuration

Le fichier **conf/elasticsearch.yml** permet de centraliser toutes les configurations possible sur le noeud du cluster.

Note importante

Il est impératif de prévoir un outil de provisioning type Chef, Puppet ou Ansible pour le maintient des configurations. Une gestion manuelle peut vite devenir ingérable avec un Cluster contenant beaucoup de noeud.

CSP Informatique Daniel Lavoie - dlavoie@live.ca

Configurations à connaitre

cluster.name
Nom de cluster recherché par le noeud à son démarrage. Tous les noeuds tentant de former un cluster doivent utiliser le même nom de cluster.

node.name
Nom du noeud. Un nom aléatoire sera généré si non spécifié.

path.data
Par défaut, Elasticsearch persiste ses données parmi le répertoire **data** de la distribution. Il faut donc changer cette configuration pour éviter de perdre les données sur une monté de version d'Elasticsearch.

path.logs
Répertoire dans lequel les logs d'Elasticsearch seront écrit.

path.plugins
Tout comme les données, les plugins sont persisté parmi les binaires de la distribution. Il est donc utile de configurer ce paramètre pour pouvoir garder ses plugins après une monté de version.

discovery.zen.ping.unicast.hosts
Liste des noeuds appartenant au cluster. C'est avec cette liste que sera construit le Cluster.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Indexation et recherche

Quelques termes :

Index : Représente une collection de document de la même famille (l'équivalent d'une table SQL)

Type : Détermine les champs disponible dans l'index (l'équivalent d'une structure de table SQL).

Document : Représente une donnée stocké dans un index (équivalent d'une ligne de table SQL).

L'action d'insérer un nouveau document dans un index se nomme **indexer**.

CSP Informatique Daniel Lavoie - dlavoie@live.ca

Types de recherche

Recherche par identifiant

Recherche Lite

Recherche avec Query DSL

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Recherche par identifiant

GET /***index***/***type***/***id***

Exemple

GET /address/v1/ADRNIVX_0000000285616515

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Recherche Lite

GET /***index***/***type***/_search?q=**mot-clef**

Exemples

GET /address/address/_search?q=nom_comm:seine
GET /address/address/_search?q=+nom_comm:seine%2B+numero:23

Très puissant mais difficile à lire de par l'encodage HTTP.
/person/_search?q=%2Bname%3A(mary+john)+%2Bdate%3A%3E2014-09-10

Pour toute les options de recherche :
https://www.elastic.co/guide/en/elasticsearch/guide/current/search-lite.html

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Recherche avec Query DSL

```
GET /_search
{
    "query": {
        fonction: {
            argument1: valeur1,
            argument2: valeur2
        }
    }
}
```

Exemple

```
GET /_search
{
    "query": {
        "match": {
        "nom_comm": "seine"
        }
    }
}
```

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Query et Filter

Query

Critères sur laquelle sera basé notre requête de recherche. Les résultats de recherche obtiendront un score de recherche qui indique la pertinence du résultat par rapport aux critère de la Query construite.

Filter

Critère de filtrage. Les données qui ne correspondent pas au filtre seront complètement ignoré. Le score de recherche ne sera pas calculé avec les paramètres de la Query.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Critères de requête

match_all

{ "match_all": {}}

Fonction de recherche par défaut si aucune fonction est spécifié.

match

{ "match": { "nom_comm": "Seine" }}

Applique une recherche Full-Text sur les champs spécifié. Les champs seront analysés.

Daniel Lavoie - dlavoie@live.ca

Critères de requête

multi_match

```
{
    "multi_match": {
        "query":    "napoléon",
        "fields":   [ "voie", "nom_comm" ]
    }
}
```

Permet d’appliquer la fonction **match** à plusieurs champs simultanément.

Daniel Lavoie - dlavoie@live.ca

Critères de requête

range

```
{
    "range": {
        "age": {
            "gte":  20,
            "lt":   30
        }
    }
}
```

Propose plusieurs opérateurs **gt**, **gte**, **lt** et **lte** permettent de déclarer l'intervalle de recherche.

Daniel Lavoie - dlavoie@live.ca

Combinaison de critères

```
{
  "query": {
    "bool": {
      "must": {
        "match_all": {}
      },
      "filter": {
        "match": {
          "numero": 110
        }
      }
    }
  }
}
```

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Pratique - Recherche par identifiant

https://github.com/daniellavoie/elk-training/search/search-identifier.md

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Pratique - Recherche lite

https://github.com/daniellavoie/elk-training/search/search-lite.md

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Pratique - Recherche QueryDSL

https://github.com/daniellavoie/elk-training/search/search-querydsl.md

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Insertions

Insertion avec identifiant

```
PUT /{index}/{type}/{id}
{
  "field": "value",
  ...
}
```

Insertion avec Identifiant Auto-Généré

```
POST /{index}{type}
{
  "field": "value",
  ...
}
```

CSP Informatique

Daniel Lavoie - dlavoie@live.ca

Insertions

Insertion bulk

{ ***action***: { ***metadata*** }}\n
{ ***request body*** }\n
{ ***action***: { ***metadata*** }}\n
{ "message" : "Test d'insertion builk 1" }
{ "create": { "_index": "test-bulk", "_type": « test-bulk",}}
{ "message" : "Test d'insertion builk 2" }
{ "create": { "_index": "test-bulk", "_type": "test-bulk"}}
{ "message" : "Test d'insertion builk 3" }

{ ***request body*** }\n

Actions supporté

create : Créer un document seulement s'il n'existe pas.
index : Créer ou remplace un document.
update : Mise à jour partiel d'un document
delete : Suppression du document.

metadata doit contenir l'index, le type et l'id du document à insérer.

request body correspond au document à insérer.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Insertions

Exemple d’insertion bulk

```
POST /_bulk
{ "create": { "_index": "test-bulk", "_type": "test-bulk"}}
{ "message" : "Test d'insertion bulk 1 ‘’nom_serveur’’ : ‘’serveur-toto’’ }
{ "delete": { "_index": "test-bulk", "_type": "test-bulk"}}
{ "message" : "Test d'insertion bulk 2" }
{ "create": { "_index": "test-bulk", "_type": "test-bulk"}}
{ "message" : "Test d'insertion bulk 3" }
```

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Pratique - Insertions

https://github.com/daniellavoie/elk-training/insert/insert.md

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Mapping et analyse

Toutes les données stocké dans Elasticsearch sont systématiquement indexé. C’est cet index qui sera interrogé lors des demandes de recherche.

Les informations qui sont stockés dans cet index sont donc cruciaux.

Elasticsearch permet d’influencer comment nos données seront représenté dans l’index. Ceci permet donc d’agir sur le comportement attendu en fonction des mots clefs saisis pour la recherche.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Analyse

Action de prendre une chaine de recherche et de la décomposer en élément de comparaison pour l’index.

L’analyse se produit lors de l’indexation d’une donnée dans l’index et lors de la comparaison d’un mot clef de recherche avec le contenu d’un index.

L’action d’analyse permet de mettre les documents stocké et ceux recherché sur le même pied de comparaison.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Mapping

Le mapping correspond aux instructions d’analyse pour chaque champs à comparer lors de la recherche.

Le mapping détermine comment un champs sera stocké dans l’index.

Il est impossible de modifier un mapping sans devoir ré indexer toutes les données de l’index impacté.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Agrégations - Concepts

Bucket

Résultat d’un groupe agrégé.

Metrics

Valeur résultant d’un calcul d’agrégation.
Applicable sur une requête globale ou sur un bucket.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Agrégations - Exemple

```
GET /address/address/_search
{ "size": 0,
  "aggs": {
    "adresse_par_ville": {
      "terms": {
        "field": "nom_comm"
      }
    }
  }
}
```

Les agrégations sont sensible aux définitions du mapping. Pour obtenir une agrégation sur un champs sans qu'il soit découpé, il faut prévoir désactiver l'analyse sur le champ à agréger.

Ne pas hésiter à se référer à la documentation de référence Elasticsearch pour identifier toutes les fonctions d'agrégations.

https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations.html

Pratique - Agrégations

https://github.com/daniellavoie/elk-training/aggregations/aggregations.md

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Prêt pour la production - Mémoire

Mémoire

Très important, suffisamment de mémoire signifie moins d'IO. Ultimement, tout finit sur disque, mais plus il y a de mémoire, mon il a d'allé retour avec les disques.

64Gb représente un juste milieu pour une instance.

8Gb est un minimum. Créer un cluster basé sur des machines plus petite se traduit en perte d'efficacité.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Prêt pour la production -CPU et Réseau

CPU

Les processeurs modernes suffisent amplement à répondre aux besoin d'Elasticsearch. Il est très peu gourmand en temps CPU. Il est préférable d'opter sur un maximum de coeur possible car la capacité de traitement concurrentiel est préférable à la puissance de calcul brute.

Réseau

Un cluster construit sur un réseau 1Gb ou 10Gb suffit amplement. Il est a noter qu'Elasticsearch ne gère pas nativement la localisation des données par Datacenter. Des techniques avancé mais plus complexes permettent de dupliquer les données sur différents cluster élasticsearch déployé dans différent Datacenter.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Prêt pour la production - Et la JVM dans tout ça ?

Il est recommandé de toujours utilisé la dernière version du JDK (Oracle ou OpenJDK). Lucene est une librairie très complexe qui a dévoilé énormément de bug sur les dernières versions du JDK.

Il est recommandé que les clients Java qui utilisent les API native (Transport Client ou Node Client) tournent avec une JVM identique à celle des noeuds du cluster. Oracle a tendance à impacter les mécanismes de sérialisation même sur des versions mineures.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Et si je veux ajuster la JVM ?

Les paramètres de lancement de la JVM sont spécialement optimisé pour les besoins d'Elasticsearch. Les paramètres utilisé par Elasticsearch viennent de cas d'études sur des cluster gigantesques.

Un problème de performance sur Elasticsearch se résume très souvent à ajouter des noeuds ou à revoir la structure des données indexés.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

On touche pas à la JVM, sinon…

Daniel Lavoie - dlavoie@live.ca

Gestion de la mémoire par Elasticsearch

L'utilisation de la mémoire vive par Elasticsearch se gère en deux temps. L'instance applicative d'Elasticsearch travaille avec la heap tandis que les instances Lucene qu'elle utilisent, elles, travaillent avec la non heap. Par conséquent, il est normalement conseillé de configurer une heap qui permet de laisser libre autant de mémoire pour la non heap.

La heap peut être configuré par la variable d'environment **ES_HEAP_SIZE**.

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Que reste t'il pour optimiser le cluster ?

SSD ! SSD ! SSD !!!!

Insertion en lots (attention à la taille des requêtes).

Pas de RAID mirroring

Augmenter le Refresh Interval lorsque l'on peut se permettre un décalage entre les insertions et les écritures

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Logstash

- Concepts
- Lancement de logstash
- Composition d'un fichier de configuration
- Input plugins
- Filter plugins
- Output plugins

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Concepts

Logstash a originalement été conçu pour récolter des logs systèmes. Cependant, son système de plugins en fond un pipeline de traitement de données très flexible.

Le système modulaire composé de plugins d'entrés, de sorties et de filtres permet d'ingérer des données de n'importe quel source, de les transformer puis de les envoyé vers n'importe quel système en sortie.

La déclaration des plugins prend en considération un méta langage, propre à Logstash, qui permet de définir des conditions sur les différents plugins déclaré dans le pipe line.

Lancement de logstash

Logstash s’exploite sous forme d’outil en ligne de commande. Il suffit de le lancer en spécifiant le fichier contenant la définition et les configurations du pipeline d’intégration de donnée que l’on désire construire.

Exemple : bin/logstash -f logstash-logparser.conf

Composition d’un fichier de configuration

```
input {
  # Endroit où l’on déclare les plugins de consommation.
}

filter {
  # Déclaration des plugins pour la transformation des
  # données
}

output {
  # Définition des plugins d’exportation des données
  # transformés.
}
```

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Input plugin

Un input filter s’occupe d’écouter une source de donnée. Voici quelques exemple de source :

Liste complete et documentation de Référence pour tous les plugins :

https://www.elastic.co/guide/en/logstash/current/input-plugins.html

Daniel Lavoie - dlavoie@live.ca

File input plugin

```
input {
  file {
    path => "/tmp/access_log"
    start_position => "beginning"
  }
}
```


Daniel Lavoie - dlavoie@live.ca

JDBC input plugin

```
input {
  jdbc {
    jdbc_driver_library => "mysql-connector-java-5.1.36-bin.jar"
    jdbc_driver_class => "com.mysql.jdbc.Driver"
    jdbc_connection_string => "jdbc:mysql://localhost:3306/mydb"
    jdbc_user => "mysql"
    parameters => { "favorite_artist" => "Beethoven" }
    schedule => "* * * * *"
    statement => "SELECT * from songs where artist = :favorite_artist"
  }
}
```

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

D’autres Input Plugin

elasticsearch
exec
eventlog
file
github
http
irc
imap
jdbc
jmx
kafka
rss
rabbitmq
redis
s3
syslog
tcp
twitter
unix
udp
websocket
zeromq

CSP Informatique
Daniel Lavoie - dlavoie@live.ca

Filter plugin

Les plugins de type Filter recevront toutes les données provenant des différents Input Plugin.

Les filtres ont sont généralement utilisé pour formater certaines données avant leur insertions

Références technique sur tous les filtres disponible :

https://www.elastic.co/guide/en/logstash/current/filter-plugins.html

Les filtres classiques

csv

geoip

throttle

mutate

anonymize

xml

Daniel Lavoie - dlavoie@live.ca

Output Plugin

Correspond à la dernière partie du pipeline d'intégration de données. Les plugins output prennent les sorties des filtres et les exporteront vers les destinations configurés dans cette section.

Références technique sur tous les Output Plugin :

https://www.elastic.co/guide/en/logstash/current/output-plugins.html