Ateliers
« ElasticSearch»

Pré-requis :

- Bonne Connexion Internet
- OS : Linux, MacOs, Windows 10
- YAML Editor : (VSCode, Atom, ...)
- Optionnel : Docker, Git

Atelier 0 : Installation

0.1 Premier démarrage

Vérifier votre espace disque libre (Au moins 20 % de libre)

Télécharger et décompresser la dernière release d' Elastic Search

Démarrer le serveur avec
`$ES_HOME/bin/elasticsearch`

Si la release est supérieure à 8, Visualiser les traces et sauvegarder toutes les informations relative au password et à l'*enrollement token*

Accéder à http(s)://localhost:9200

0.2 Fichier de configuration et traces

Editer le fichier de configuration principal et modifier les propriétés suivantes :

- Nom du Cluster
- Nom du nœud
- Adresse d'écoute (Indiquer votre adresse publique)

Essayer de démarrer le serveur et observer les vérifications de démarrage (bootstrap checks), effectuer les corrections si nécessaires.
(voir https://www.elastic.co/guide/en/elasticsearch/reference/current/bootstrap-checks.html)

Changer le niveau de trace à WARN

Positionner la propriété ***node.name*** via la CLI lors du démarrage du serveur

0.3 Installation de Kibana

Télécharger une distribution de Kibana avec le même numéro de version que celui d'ElasticSearch .

Décompresser l'archive et démarrer Kibana (si 8.x +, utiliser l'enrollment token)

Accéder à http://localhost:5601 et retrouver la *Dev Console*.

Exécuter les requêtes suivantes :

```
GET /_cluster/health
GET /_search
GET /_cat/nodes
```

Atelier1 : Cluster, noeuds, shards, replica

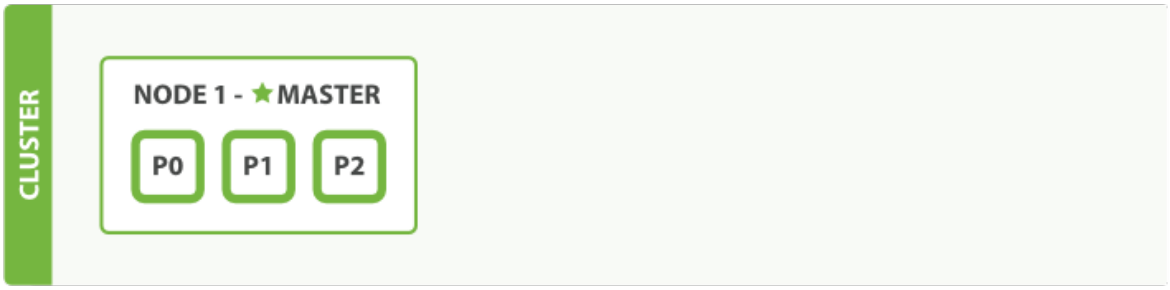
1.1 Nœuds, cluster et shard replica

- Créer un enrollment token with :
  bin/elasticsearch-create-enrollment-token -s node
- Décommenter la propriété ***transport.host*** à la fin de *config/elasticsearch.yml*.
- Redémarrer Elasticsearch.

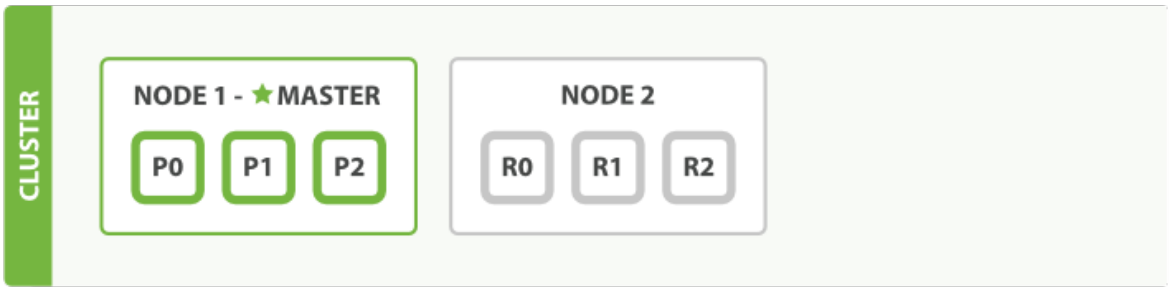
- Exécuter :
- `GET /_cluster/health?pretty`
  Combien de nœuds de shards sont disponibles ?
- Créer un index nommé *blogs*

```
PUT /blogs {
 "settings" : {
 "number_of_shards" : 3,
 "number_of_replicas" : 1
 }
}
```

- Re-executer *_cluster/health?pretty*
  Couleur du statut de santé ? Combien de disponibles, actifs ?



- Dézipper la distribution dans un autre emplacement
- Démarrer un second noeud avec l'enrollment token précédent :
  bin/elasticsearch --enrollment-token <token>
- Re-executer `_cluster/health?pretty`
  Couleur du statut de santé ? Combien de disponibles, actifs ?



- Démarrer un 3ème noeud
  Couleur du statut de santé ? Combien de disponibles, actifs ?
- Augmenter le nombre de répliques
  ```
  PUT /blogs/_settings
   { "number_of_replicas" : 2 }
  ```
- Arrêter le premier noeud
- Couleur du statut de santé ?
- Redémarrer le premier noeud

1.2 Désactivation de la sécurité

Pour les laboratoires suivants, nous désactivons la sécurité. Dans 8.x+, la sécurité est activée par défaut.

Pour désactiver, il faut :

- Positionner xpack.security.enabled: false
- Commenter toutes les propriétés relatives à *xpack.security* dans *elasticsearch.yml*
- Supprimer les propriétés stocker dans le keystore d'elasticsearch.
  ```
  bin/elastisearch-keystore remove <name-of-the-setting>
  ```

Dans *kibana.yml* commenter toutes les propriétés relatives à *xpack.security*

Atelier2 : Document API et SearchLite

2.1 Document API

- Indexer les 3 documents fournis via la commande *curl* suivante ou via Kibana
  `curl -XPOST /blogs/_doc/ --data-binary "@entry1.json"`
- Noter les ids et récupérer les document via un ID
- Mettre à jour 1 document en ajoutant de nouveau champs (Remarquer l'incrémentation de la version) :
  - *tags* : Array
  - *views* : Initialisé à 0
- Effectuer des mises à jour avec un script:
  - Incrémenter un champ numérique
  - Ajouter un élément au tableau *tags*
  - Supprimer le champ *tags*
- Supprimer l'index et effectuer toutes les opérations suivantes en une seule commande Bulk API

2.2 Search Lite

Effectuer les recherches suivantes en utilisant la query string et en utilisant les paramètres suivants :

- *size, from*
- *q, df*
- *_source*

Atelier3 : Beats

3.1 Metrics beats

Dans cette partie, nous mettons en place *metricbeats* afin qu'il alimente directement ElasticSearch.

- Récupérer une distribution de *metricbeats*
- Configurer :
  - Activer uniquement le module système dans la configuration
  - Configurer le cluster ELS et la répartition de charge
  - Exclure des informations particulières (voir doc en ligne)
  - Activer le mode DEBUG des traces
  - Configurer les index créés afin qu'il aient 2 shards et 1 réplique
- Démarrer *metricbeats*
- Vérifier dans ElasticSearch que des données apparaissent et que le mapping est satisfaisant
- Vérifier dans Kibana que les tableaux de bord ont été importés

Atelier 5 : Démarrer avec Logstash

Vérification de l'installation

- Dézipper l'archive fournie et vérifier qu'il s'agit de la même version qu'ElasticSearch
- Placer vous dans *bin* et exécuter :
  ```
  logstash -e 'input { stdin { } } output { stdout {} }'
  ```

Traitement de logs Apache

Connexion FileBeat / Logstash

- Récupérer la distribution de *FileBeat* et dézipper
- Récupérer l'archive contenant les logs Apache et dézipper
- Dans le fichier de configuration ***filebeat.yml***, mettre à jour la configuration avec les lignes suivantes :

```
filebeat.inputs:

- type: filestream

id: my-filestream-id

enabled: true

# Paths that should be crawled and fetched. Glob based paths.
paths:
# - /var/log/*.log
- /path/to/file/logstash-tutorial.log

output.logstash:
hosts: ["localhost:5044"]
```

- Démarrer *filebeat* avec la commande suivante :
  ```
  ./filebeat -e -c filebeat.yml -d "publish"
  ```

- Créer un fichier de configuration de pipeline ***pipeline.conf*** comme suit :

```
input {
    beats {
        port => "5044"
    }
}
# The filter part of this file is commented out to indicate that it is
# optional.
# filter {
#
# }
output {
    stdout { codec => rubydebug }
}
```

Tester la configuration :

```
bin/logstash -t -f ./pipeline.conf
```

Démarrer logstash et observer la console :

```
bin/logstash -r -f ./pipeline .conf
```

Ajout filtre grok

- Modifier la configuration afin d'appliquer un filtre grok parsant les messages Apache

```
filter {
    grok {
        match => { "message" => "%{COMBINEDAPACHELOG}"}
    }
}
```

La configuration est automatique par contre il faut demander à *Filebeat* de retraiter le fichier Apache.

- Arrêter *filebeat* et supprimer son registre avec la commande :

```
sudo rm data/registry
```

- Relancer *filebeat* et observer la sortie sur la console de logstash.
  Quels sont les changement par rapport à tout à l'heure ?

Ajouter ensuite le filtre GeoIP

```
geoip {
      source => "[source][address]"
      ecs_compatibility => "disabled"
}
```

- Recommencer le traitement et observer les changements

Alimentation d'ElasticSearrch

- Modifier la config afin que la sortie soit dirigée vers votre cluster ElasticSearch

```
output {
   elasticsearch {
        hosts => [ "localhost:9200" ]
    }
}
```

- Recommencer le traitement, vérifier la console d'ElasticSearch et trouver l'index créé.
- Effectuer une recherche afin de trouver des documents ElasticSearch

Atelier6 : Configuration Pipeline Logstash (Traitement de fichiers de traces)

6.1 Traitement des logs apache

Nous voulons améliorer la pipeline logstash du TP précédent.
Le répertoire de logs Apache contient différents fichiers en fonction des virtual host configuré sur Apache.
La pipeline doit :

- Utiliser un input file traitant tous les fichiers d'un répertoire
- Alimenter des types de documents différents en fonction des noms des fichiers de trace
- Convertir le champ « bytes » en un type long
- Alimenter la géo-localisation à partir de l'IP du client
- Affecter le champ *timestamp* au champ *@timestamp*
- Créer des champs day,month,year qui isolera respectivement le jour, le mois et l'année de l'événement

Modifier le nom d el'index ElasticSerach vers lequel les documents sont acheminés.

6.2 Traitement des logs Jboss

L'objectif est de garder toutes les lignes :

- WARNING, ERROR ou FATAL
- Et les exceptions

6.3 Exécution simultanée des 2 pipelines

Mettre au point le fichier pipelines.yml qui configure les 2 pipelines précédentes.

Utiliser un modèle d'exécution différent pour chaque pipeline

Atelier6-b : Configuration Pipeline Logstash (Ingestion via JDBC)

Nécessite une base mysql

Alimenter la BD :

Récupérér l'archive fournie et la dézipper.

Démarrer une CLI mysql

mysql -u root -p

Importer les données via :

source mysqlsampledatabase.sql

Mise au point de la pipeline

Mettre au point une pipeline utilisant le plugin d'input jdbc permettant d'ingérer tous les enregistrements fournis par la requête suivante :

```
select * from customers,orders,orderdetails,products where
orders.customernumber=customers.customernumber and
orderdetails.orderNumber=orders.orderNumber and
orderdetails.productcode=products.productcode;
```

Les données seront ingérer dans un index nommé ***classicmodels*** et l'ID des documents correspondera au champ ***orderLineNimber***

Atelier7: Ingestion de documents bureautiques

Installation du plugin ingest-attachment

- Installer le plugin ***ingest-attachment*** plugin
  ./elasticsearch-plugin install ingest-attachment
- Redémarrer le noeud

Création de pipeline

- Avec l'API, créer une pipeline utilisant le processeur *attachment*

Indexation

- Utiliser le programme Java fourni pour indexer tous les documents fournis :

- Vérifier le nombre de documents indexés

Atelier8 : Mapping et analyzers

8.1 Contrôle du Mapping

- Visualiser le mapping de l'index contenant les documents bureautiques. Quels sont les champs de texte intégral et quels analyseurs sont utilisés ?
- Tester les différences entre l'analyseur standard et l'analyseur français sur des textes français avec des requêtes REST. Quels sont les mots vides utilisés ? Comment se comportent les mots avec des accents, avec des apostrophes, des mots composés ?
- Définir le mapping le plus approprié pour les documents bureautiques. Le champ de contenu à analyser en français et en anglais.
- Effectuer des recherches identiques sur les 2 index et visualiser les différences

8.2 Analyseurs dédiés

- Créer un nouvel index pour la base bureautique qui utilise des analyseurs spécifiques pour le contenu anglais et français :
  - augmente la liste des stop-words
  - ◦ ajoute des synonymes
- Refaire l'indexation dans ce nouvel index et effectuer des recherches

8.3 Reindexation API

Créez un nouvel index avec de nouveaux paramètres pour les partitions et utilisez l'API de réindexation pour alimenter le nouvel index.

Atelier 9 : Gabarit d’index

Regarder les gabarits d’index existants

Regarder les index créés par les pipelines logstash précédentes

Sont-ils optimisés ?

Les supprimer et faire en sorte que la pipeline *jboss* crée des index ayant les caractéristiques suivantes :

- Nombre de shards = 1
- Tous les champs string sauf le champ *message* sont exclusivement de type *keyword*

Atelier 10 : Recherche via DSL

10.1 Syntaxe DSL

Effectuez des requêtes DSL basées sur l'index du bureau :

- Documents PDF triés par date
- Documents dont le champ de contenu répond à "administration"
- Documents dont le champ contenu ou titre répond à "administration"
- Les documents dont le champ contenu ou titre répond à "administration" et dont la date de création se situe dans une fourchette
- Documents PDF dont le champ de contenu ou de titre répond à "administration" et dont la date de création se situe dans une fourchette
- Documents dont le champ de contenu répond à "Administration" ou "Oracle"
- Documents dont le champ de contenu répond à "Administration" et éventuellement "Oracle"

10.2 Contrôle de la pertinence

Effectuer des recherches avancées avec le paramètre EXPLAIN :

- A l'aide du boosting, récupérer les Documents dont le champ contenu ou titre répond à "administration". Documents avec "administration" dans le champ titre apparaissant en premier
- Même requête utilisant des clauses should, comparer les scores
- Utiliser un autre mode de calcul avec dis_max

10.3 Match partiel

- Préparez un nouvel index qui utilise l'indexation multiple pour le champ attachment.title :
  - mot-clé DataType
  - texte avec analyseur standard
  - texte avec l'analyseur edge_ngram
- Effectuez des requêtes d'appariement partiel en utilisant l'un des 3 champs de mappage. Comparez les résultats et les temps de réponse

10.4 Phrase

Effectuez des requêtes avec des phrases :

- Récupérer les documents contenant l'expression "cadre java", permettre une distance de 5 mots
- Récupérer les documents dont le titre commence par "administration j"

10.5 Fuzzy search et langage naturel

- Effectuez des recherches floues avec des fautes de frappe
- Facultatif : préparer un nouvel index avec un filtre phonétique, effectuer des recherches avec des fautes d'orthographe

6.5 Mise en surbrillance

Effectuer les requêtes précédentes en ajoutant une surbrillance sur le champ de contenu (limiter la taille des fragments)

Atelier 11: Agrégations et géo-localisation

11.1 Agrégations

Sur l’index Apache

- Effectuer des agrégations de comptage
  - par type de code retour
  - par verbe
  - par les 2

- Effectuer des regroupements par intervalles de taille
- Trouver la taille moyenne d'une requête
- Trouver la taille moyenne des requêtes groupées par code retour
- Trouver la somme des octets transférés par trimestre
- Trouver le client ayant un taux anormal de code réponse 400

11.2 Géo-localisation

Sur l’index Apache

- Effectuer des agrégation de type *geo_hash_grid* sur les localisation des clients en augmentant progressivement la précision

Atelier 12 : Kibana et tableaux de bord

12.1 Traitement logs d'accès Apache

Le but de cet exercice est de construire un tableau de bord sur les accès Apache.

Ce tableau de bord affichera plusieurs visualisations :

- Les KPI
  - Total de hits
  - Nombre d'Ips différentes
  - Taille moyenne d'une requête
- Le profil type des connexions sur une journée : un histogramme des requêtes par heure en distinguant les code retour des requêtes (200, 404, …)
- Une répartition géographique des clients

Les visualisations seront effectués après avoir filtrer les ip clients correspondant à des adresses internes

12.2 : Timelion

Tutorial

https://www.elastic.co/guide/en/kibana/current/timelion.html

Le résultat final consiste à afficher 3 graphiques :

- L’usage CPU en temps réel comparée à celle de la dernière heure
- Le traffic réseau en entrée et en sortie
- La consommation mémoire avec l’affichage d’alerte si elle dépasse un certain seuil

Atelier 13 : Architecture

Définir un cluster logstash, utiliser persistent queue

Un cluster ElasticSearch avec des spécialisations de nœud :

- Data node
- Master nodes
- Ingstion nodes

Activer des beats (MetricBeats, Packetbeats) sur toutes les machines qui load-balanced vers les nœuds logstash,

Atelier 14 : *Exploitation*

13.1 Monitoring

- Utiliser les APIs de monitoring d'ELS
- Sans redémarrer le serveur, augmenter le niveau de log à DEBUG
- Activer le slowlog et effectuer des requêtes provoquant des écritures dans le fichier

13.2 Exploitation

- Désactiver la suppression d’index par wildcards
- Effectuer un backup d’index puis restore
- Créer des alias d’index
- Réindexation d’un index existant vers un autre index avec un nombre de shards différents