Monitoring et Enrichissement des Données de Films

Brief 7 – 9/04/2025, Manel

Livrables

* Fichier de configuration Logstash :

```
input {
  file {
    path => "/usr/share/logstash/data/movies_elastic.json"
    start_position => "beginning"
    sincedb_path => "/dev/null"
    codec => "json"
  }
}

filter {
  mutate {
    remove_field => ["@version", "host"]
  }

  # # Add 'decade' field based on 'year'
  if [fields][year] {
    ruby {
      code => "
        year = event.get('[fields][year]')
        if year.is_a?(Integer)
          decade = (year / 10) * 10
          event.set('[fields][decade]', decade.to_s + 's')
        end
      "
    }
  }

  # Convert 'running_time_secs' to 'running_time_mins'
  ruby {
    code => "
      if event.get('[fields][running_time_secs]')
        secs = event.get('[fields][running_time_secs]').to_i
        event.set('[fields][running_time_mins]', secs.to_i / 60.0)
      end
    "
  }

  # Drop events where rating is less than 5
  if [fields][rating] and [fields][rating] < 5 {
```

```
    drop { }
  }
}

output {
  elasticsearch {
    hosts => ["http://elasticsearch:9200"]
    index => "movies"
  }
  stdout { codec => rubydebug }
}
```

****

Mapping pour l'index - j'ai utilisé un method PUT sur le Kibana Dev Tools :

```
PUT /movies
{
  "mappings": {
    "properties": {
      "fields": {
        "properties": {
          "title": {
            "type": "text",
            "fields": {
              "keyword": {
                "type": "keyword"
              }
            }
          },
          "directors": {
            "type": "keyword"
          },
          "release_date": {
            "type": "date",
            "format": "dd MMM yyyy||strict_date_optional_time||epoch_millis"
          },
          "decade": {
            "type": "keyword"
          },
          "rating": {
            "type": "float"
          },
          "running_time_mins": {
            "type": "float"
          },
          "genres": {
            "type": "keyword"
          },
          "plot": {
```

```
          "type": "text"
        },
        "actors": {
          "type": "keyword"
        },
        "year": {
          "type": "integer"
        }
      }
    },
    "@timestamp": {
      "type": "date"
    }
  }
 }
}
```

****

Rapport de synthèse du travail accompli :

1. Input – Lecture des données

Source des données :
Le pipeline lit en continu un fichier JSON situé à /usr/share/logstash/data/movies_elastic.json.
Dans un contexte de streaming, on pourra aussi connecter Logstash à une source en continu (par exemple, via Kafka, Beats ou une autre API) de manière similaire.

Position de départ :
L'option start_position => "beginning" assure que Logstash lit le fichier dès le début à chaque démarrage (avec sincedb_path => "/dev/null", la position n'est pas conservée).
→ Pour une solution de production en streaming, on utilise habituellement un sincedb persistant (ou un autre mécanisme de suivi), afin d'éviter de reprocesser les mêmes données.

Codec JSON :
Les données sont interprétées comme des objets JSON grâce au codec "json".

2. Filters – Transformation et enrichissement des données

Suppression de champs inutiles :
Le plugin **mutate** retire les champs @version et host qui ne sont pas utiles dans notre cas.

Extraction de la décennie :
Un **filtre ruby** vérifie si le champ [fields][year] est présent et est un entier.

Calcul : La décennie est obtenue en divisant l'année par 10 puis en multipliant par 10.

Ajout du champ : Le résultat est converti en chaîne de caractères et suffixé avec « s » (ex. : "2010s") puis stocké dans **[fields][decade]**.

Conversion de la durée en minutes :
Un autre filtre ruby récupère le champ [fields][running_time_secs], le convertit en entier et calcule la durée en minutes en divisant ce nombre par 60.

Ajout du champ : Le résultat est enregistré dans **[fields][running_time_mins].**

Filtrage des films mal notés :
Un test conditionnel vérifie si le champ [fields][rating] existe et que sa valeur est inférieure à 5.

Action : Pour ces cas, le plugin drop supprime l'événement du flux, afin de ne garder que les films ayant un rating supérieur ou égal à 5.

Ces filtres sont appliqués en temps réel à chaque document ingéré, ce qui est particulièrement adapté pour un traitement en flux continu.

3. Output – Envoi vers Elasticsearch et affichage en console

Sortie vers Elasticsearch :
Les documents transformés sont envoyés vers l'instance Elasticsearch, dans l'index nommé movies.

Indexation :
Tous les documents traités sont enregistrés dans l'index nommé "movies".
Cette approche permet de disposer d'un index structuré et enrichi, prêt à être utilisé par Kibana pour la visualisation et l'analyse **en temps réel.**

Sortie standard (stdout) :
Le codec rubydebug est utilisé pour afficher, en mode console, le contenu des événements transformés à des fins de débogage.

Nouveaux documents indexés dans Elasticsearch après ingestion avec Logstash :

```
"max_score" : 1.0,
"hits" : [
  {
    "_index" : "movies",
    "_type" : "_doc",
    "_id" : "dMOtGZYBiH2H4T7Uat6w",
    "_score" : 1.0,
    "_source" : {
      "fields" : {
        "year" : 2012,
        "rating" : 5.1,
        "plot" : "A couple's wedding day turns horrific as some of the guests start showing signs of a strange illness.",
        "image_url" : "http://ia.media-imdb.com/images/M/MV5BNDM3OTUzMTA1OF5BMl5BanBnXkFtZTcwNTkwMTE3Nw@@._V1_SX400_.jpg",
        "directors" : [
          "Paco Plaza"
        ],
        "rank" : 3690,
        "actors" : [
          "Leticia Dolera",
          "Diego Martín",
          "Ismael Martínez"
        ],
        "title" : "[REC]³ Génesis",
        "running_time_mins" : 80.0,
        "decade" : "2010s",
        "release_date" : "2012-03-09T00:00:00Z",
        "running_time_secs" : 4800,
        "genres" : [
          "Comedy",
          "Horror"
        ]
      },
      "type" : "add",
      "@timestamp" : "2025-04-09T08:31:57.201Z",
      "id" : "tt1649444",
      "path" : "/usr/share/logstash/data/movies_elastic.json"
    }
  },
```


Dashboard Kibana

1. Choix des visualisations

- Carte «Top Rated Movie» + Directors + Actors + Duration

Affiche le film le mieux noté pour la décennie sélectionnée. Permet de repérer rapidement la référence de la période choisie. Rassemble en un coup d'œil les métadonnées essentielles pour l'utilisateur.

- Liste déroulante «Choice of Decade» et bar de carrés = meme fonctionnalité :

Propose un filtre par décennie (1920s, 1930s, etc.), permettant à l'utilisateur de n'explorer qu'une période donnée. Toutes les autres visualisations se mettent à jour en fonction de cette sélection.

- Pie Chart (Répartition par Genre du film)

Montre la proportion de chaque genre dans l'ensemble de la sélection courante (décennie, ou tous les films). Permet de voir quels genres dominent et de comparer leur popularité relative.

- Bar Chart (Films by Rating)

Affiche la distribution des films par note (rating). Donne un aperçu du niveau qualitatif des films conservés (notés ≥ 5), avec leur répartition sur différentes tranches de rating.

- Bar Chart (Films by Decade)

Sert à visualiser l'évolution du nombre de films sur les différentes décennies.

- Metric «Total Movies»

Montre combien de films au total (avec rating ≥ 5) correspondent au filtre actuel. Indique la densité de contenu et donne un repère pour l'analyse globale.

Interprétation des résultats : Ce dashboard permet de découvrir les tendances par décennie (quantité de films, genres dominants), d'identifier rapidement les meilleurs films (en fonction du rating), et de visualiser les informations clés (réalisateurs, acteurs, durée). Les filtres interactifs rendent l'analyse très flexible et aident à se focaliser sur la période ou la valeur recherchée.

Monitoring des logs

1 - Configuration de Metricbeat :

```
# config/metricbeat.yml
metricbeat.modules:
  - module: elasticsearch
    metricsets: ["node", "node_stats", "cluster_stats"]
    period: 10s
    hosts: ["http://elasticsearch:9200"]

  - module: logstash
    metricsets: ["node", "node_stats"]
    period: 10s
    hosts: ["http://logstash_stream:9600"]
```

```
  - module: docker
    metricsets:
      - "container"
      - "cpu"
      - "diskio"
      - "healthcheck"
      - "info"
      - "memory"
      - "network"
    hosts: ["unix:///var/run/docker.sock"]
    period: 10s

output.elasticsearch:
  hosts: ["http://elasticsearch:9200"]

setup.kibana:
  host: http://kibana-stream:5601
```

Explications :

2 - Docker-compose.yaml (il faut ajouter le module de metricbeat) :

```
 metricbeat:
    image: docker.elastic.co/beats/metricbeat:7.17.0
    container_name: metricbeat_stream
    user: root
    volumes:
      - ./config/metricbeat.yml:/usr/share/metricbeat/metricbeat.yml
      - /var/run/docker.sock:/var/run/docker.sock:ro
      - /sys/fs/cgroup:/hostfs/sys/fs/cgroup:ro
      - /proc:/hostfs/proc:ro
      - /:/hostfs:ro

    depends_on:
      - elasticsearch
      - logstash
      - kibana
```

3 – on télécharge les dashboards à Kibana via terminal :

```
docker exec -it metricbeat_stream metricbeat setup –dashboards
```

4 – verification de l’index via Kabana Dev Tools :

```
GET metricbeat-*/_search
{
  "query": {
    "match_all": {}
  }
```

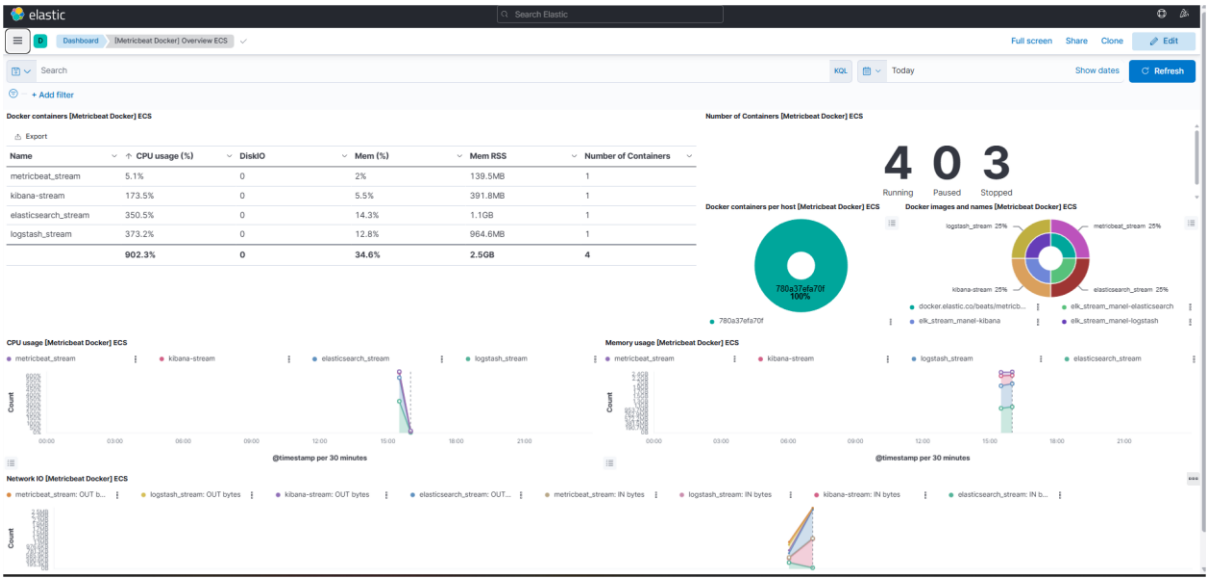
```
}

output :
{
  "took" : 3,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 3380,
      "relation" : "eq"
    },
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "metricbeat-7.17.0-2025.04.09-000001",
        "_type" : "_doc",
        "_id" : "tVDlGpYBcaxfll21ceeg",
        "_score" : 1.0,
        "_source" : {
          "@timestamp" : "2025-04-09T14:12:45.612Z",
          "metricset" : {
            "period" : 10000,
            "name" : "container"
          },
          "service" : {
            "address" : "unix:///var/run/docker.sock",
            "type" : "docker"
          },
          "event" : {
            "dataset" : "docker.container",
            "module" : "docker",
            "duration" : 9986789
          },
          "host" : {
            "name" : "780a37efa70f"
          },
          "agent" : {
            "ephemeral_id" : "e51c1037-968b-444d-929e-04c31f2038d8",
            "id" : "c5894f85-9ad0-4ce6-bafb-fe8343072f5e",
            "name" : "780a37efa70f",
            "type" : "metricbeat",
            "version" : "7.17.0",
```

```
      "hostname" : "780a37efa70f"
    },
    "ecs" : {
      "version" : "1.12.0"
    },
    "container" : {
      "name" : "metricbeat_stream",
      "runtime" : "docker",
      "id" : "780a37efa70f6f301c3ec2739ec603e291ff0b1f795b8aa90a29e590070fe7d6",
      "image" : {
        "name" : "docker.elastic.co/beats/metricbeat:7.17.0"
      }
    },
    "docker" : {
      "container" : {
        "labels" : {
          "desktop_docker_io/binds/1/Source" : "/proc",
          "io_k8s_description" : "Metricbeat is a lightweight shipper for metrics.",
          "com_docker_compose_version" : "2.15.1",
          "org_label-schema_name" : "metricbeat",
          "com_docker_compose_oneoff" : "False",
          "org_opencontainers_image_title" : "Metricbeat",
          "desktop_docker_io/binds/0/Target" : "/hostfs/sys/fs/cgroup",
          "desktop_docker_io/binds/1/Target" : "/hostfs/proc",
          "org_label-schema_vcs-url" : "github.com/elastic/beats/v7",
          "com_docker_compose_project_config_files" :
"""C:\Users\daria\Documents\ELK_stream_Manel\docker-compose.yml""",
          "org_opencontainers_image_created" : "2022-01-28T10:05:33Z",
          "org_label-schema_schema-version" : "1.0",
          "org_label-schema_build-date" : "2022-01-28T10:05:33Z",
          "desktop_docker_io/binds/3/SourceKind" : "hostFile",
          "com_docker_compose_container-number" : "1",
          "desktop_docker_io/binds/3/Target" : "/usr/share/metricbeat/metricbeat.yml",
          "license" : "Elastic License",
          "org_label-schema_url" : "https://www.elastic.co/beats/metricbeat",
          "com_docker_compose_project" : "elk_stream_manel",
          "io_k8s_display-name" : "Metricbeat image",
          "summary" : "metricbeat",
          "release" : "1",
          "name" : "metricbeat",
          "desktop_docker_io/binds/0/Source" : "/sys/fs/cgroup",
          "com_docker_compose_service" : "metricbeat",
          "org_label-schema_vendor" : "Elastic",
          "desktop_docker_io/binds/2/Source" : "/",
          "com_docker_compose_config-hash" :
"38400323b24de20c09099ff50271da50894a329800c5d61c5ed02f5e6c324d15",
          "com_docker_compose_image" :
"sha256:ec23a1f53a960681d75b0cfa36b615bdf3b2125e8c204bea595f7a837d2fb175",
```

```
            "url" : "https://www.elastic.co/beats/metricbeat",
            "desktop_docker_io/binds/1/SourceKind" : "hostFile",
            "com_docker_compose_depends_on" :
"logstash:service_started,kibana:service_started,elasticsearch:service_started",
            "desktop_docker_io/binds/4/SourceKind" : "dockerSocketProxied",
            "org_opencontainers_image_licenses" : "Elastic License",
            "desktop_docker_io/binds/4/Target" : "/var/run/docker.sock",
            "desktop_docker_io/binds/2/SourceKind" : "hostFile",
            "org_opencontainers_image_vendor" : "Elastic",
            "maintainer" : "infra@elastic.co",
            "description" : "Metricbeat is a lightweight shipper for metrics.",
            "desktop_docker_io/binds/3/Source" :
"""C:\Users\daria\Documents\ELK_stream_Manel\config\metricbeat.yml""",
            "desktop_docker_io/binds/2/Target" : "/hostfs",
            "desktop_docker_io/binds/4/Source" : "/var/run/docker.sock",
            "org_label-schema_vcs-ref" : "93708bd74e909e57ed5d9bea3cf2065f4cc43af3",
            "org_label-schema_version" : "7.17.0",
            "vendor" : "Elastic",
            "desktop_docker_io/binds/0/SourceKind" : "hostFile",
            "org_label-schema_license" : "Elastic License",
            "com_docker_compose_project_working_dir" :
"""C:\Users\daria\Documents\ELK_stream_Manel""",
            "version" : "7.17.0"
          },
          "created" : "2025-04-09T13:57:56.000Z",
          "command" : "/usr/bin/tini -- /usr/local/bin/docker-entrypoint /bin/sh -c 'chmod go-w
/usr/share/metricbeat/metricbeat.yml && metricbeat -e -c
/usr/share/metricbeat/metricbeat.yml'",
          "ip_addresses" : [
            "172.20.0.5"
          ],
          "size" : {
            "rw" : 0,
            "root_fs" : 0
          },
          "status" : "Up 14 minutes"
        }
      }
    }
  },
```


Dashboard de logs metrics via Kibana :



Interprétation : Le dashboard « [Metricbeat Docker] Overview ECS » mis en place permet un suivi précis et continu de l'état des conteneurs Docker, ainsi que de leurs performances en temps réel.

Objectifs du monitoring :

- Assurer une visibilité continue sur l'état des conteneurs Docker
- Surveiller la consommation des ressources critiques : CPU, mémoire et trafic réseau
- Identifier rapidement d’éventuels problèmes de performances ou anomalies.

Principales visualisations et indicateurs

Tableau détaillé des conteneurs :

- Nom du conteneur
- Consommation CPU (%) (charge du processeur)
- Consommation mémoire (%) (RAM utilisée)
- Mémoire RSS (quantité réelle de mémoire consommée)
- Opérations Disk IO
- Nombre total de conteneurs actifs

Ce tableau de bord constitue désormais une base solide pour assurer un monitoring régulier et efficace de l’infrastructure Docker avec Elastic Stack.
Le monitoring est opérationnel et pleinement fonctionnel.