Ecole Supérieure Privée des Technologies et de l'Ingénierie

SPECIALITE : SSIR-2D

RAPPORT DE PROJET CLOUD

Implémentation d'une architecture ELK

Réalisée par : Wissem nasri

Encadré par : mourad melliti

Année Universitaire : 2023 – 2024

Table de matière

1. Introduction 1
1.1. Contexte et Objectifs 1
2. Présentation de l'Architecture ELK 1
2.1. Description des Composants ELK (Elasticsearch, Logstash, Kibana, Metricbeat, Filebeat, Setup Certificates) 1
2.2. Avantages et Limitations de l'Architecture ELK 2
2.2.1. Avantages : 2
2.2.2. Limitations : 2
3. Analyse des Besoins et Objectifs 2
3.1. Identification des Besoins Techniques et Fonctionnels 2
3.2. Spécifications Non-Fonctionnelles (Sécurité, Performances, Évolutivité) 3
3.3. Enjeux et Contraintes du Projet 3
4. Conception de l'Architecture 3
4.1. Schéma d'Architecture ELK 3
5. Mise en Œuvre de l'Architecture ELK 4
5.1. Préparation de l'Environnement (Docker et Docker Compose) 4
5.2. Deploiement des services 5
5.2.1. Déploiement de Setup Certificates 6
5.2.2. Déploiement de Elasticsearch 6
5.2.3. Déploiement de Kibana 6
5.2.4. Déploiement de Logstash, Filebeat et Metricbeat 6
6. Tests et Validation de l'Architecture 7

Table des figure

Figure 1 : Schéma d'Architecture ELK 4
Figure 2:installation docker & docker-compose 5
Figure 3:fichier de configuration des services 5
Figure 4:déploiement des services 5
Figure 5:vérification du lancement des services 6
Figure 6:: certificats générer par Setup certificates 7
Figure 7:fonctionnement du metricbeat 7
Figure 8:teste de fonctionnement de filebeat 8
Figure 9:este de fonctionnement de logstash 9

1. Introduction

1.1. Contexte et Objectifs

Les systèmes modernes génèrent de grandes quantités de logs, contenant des informations précieuses pour le suivi, la sécurité, et l'optimisation des performances. Gérer ces données efficacement est un défi majeur pour les administrateurs et les équipes de sécurité. Dans ce contexte, l'architecture ELK (Elasticsearch, Logstash, Kibana) est une solution largement utilisée pour centraliser, traiter et visualiser ces données de manière rapide et accessible.

L'objectif de ce rapport est de documenter la mise en œuvre d'une architecture ELK, en partant de l'analyse des besoins jusqu'à l'évaluation des résultats, afin de faciliter l'analyse des logs et la détection d'anomalies.

2. Présentation de l'Architecture ELK

2.1. Description des Composants ELK (Elasticsearch, Logstash, Kibana, Metricbeat, Filebeat, Setup Certificates)

L'architecture ELK repose sur trois composants principaux : Elasticsearch, Logstash, et Kibana. Elle est souvent étendue avec d'autres outils comme Metricbeat et Filebeat pour la collecte de données et l'analyse.

- **Elasticsearch** : Un moteur de recherche et d'analyse distribué utilisé pour stocker et indexer les données de logs. Il permet de rechercher rapidement des informations spécifiques dans de grandes quantités de données.
- **Logstash** : Un pipeline de traitement de données qui collecte, filtre et envoie les logs vers Elasticsearch. Il est hautement configurable et peut traiter les données issues de plusieurs sources.
- **Kibana** : Un outil de visualisation qui permet d'afficher les données stockées dans Elasticsearch sous forme de graphiques, tableaux de bord, et alertes. Kibana rend les données plus accessibles pour une analyse en temps réel.
- **Metricbeat** : Un outil léger qui collecte les métriques des systèmes et services (CPU, mémoire, réseau, etc.) et les envoie à Elasticsearch pour surveillance et analyse.
- **Filebeat** : Un collecteur léger qui envoie les logs des fichiers directement vers Elasticsearch ou Logstash, facilitant l'ingestion des données de journaux sans surcharge du système.

- **Setup Certificates** : La configuration des certificats SSL/TLS pour sécuriser les échanges de données entre les composants de l'architecture ELK, garantissant ainsi la confidentialité et l'intégrité des informations transmises.

2.2. Avantages et Limitations de l'Architecture ELK

2.2.1. Avantages :

- **Centralisation** : Facilite la collecte et la gestion des logs de différentes sources en un seul endroit.
- **Analyse en Temps Réel** : Avec Kibana, les utilisateurs peuvent surveiller et analyser les données en temps réel.
- **Scalabilité** : Elasticsearch est conçu pour gérer des volumes importants de données et peut être étendu pour répondre aux besoins croissants.

2.2.2. Limitations :

- **Consommation de Ressources** : Les composants ELK peuvent être gourmands en mémoire et CPU, surtout avec de gros volumes de données.
- **Configuration Complexe** : La configuration et le déploiement d'ELK peuvent nécessiter des compétences techniques et du temps, surtout pour des fonctionnalités avancées comme la sécurité.
- **Sécurisation** : Nécessite des configurations supplémentaires, comme les certificats SSL/TLS, pour assurer la sécurité des données en transit.

3. Analyse des Besoins et Objectifs

3.1. Identification des Besoins Techniques et Fonctionnels

Dans cette partie, nous identifions les besoins techniques et fonctionnels que l'architecture ELK doit satisfaire pour réussir. Ces besoins incluent :

- **Collecte de Logs** : Le système doit pouvoir collecter des logs de différents serveurs, applications et dispositifs réseau.
- **Analyse des Données** : Une capacité d'analyse avancée pour extraire des informations pertinentes à partir des logs collectés.
- **Visualisation** : Les utilisateurs doivent pouvoir visualiser les données sous forme de tableaux de bord interactifs et graphiques dans Kibana.
- **Alertes et Notifications** : L'architecture doit être capable de générer des alertes en fonction d'événements ou d'anomalies détectées dans les données.

3.2. Spécifications Non-Fonctionnelles (Sécurité, Performances, Évolutivité)

Les spécifications non-fonctionnelles de l'architecture ELK comprennent des exigences telles que :

- **Sécurité** : Le système doit assurer la protection des données, en particulier via des protocoles sécurisés comme SSL/TLS pour le chiffrement des communications.
- **Performances** : La capacité de traiter et d'analyser un grand volume de logs en temps réel tout en maintenant des temps de réponse optimaux.
- **Évolutivité** : L'architecture doit être capable de s'adapter à une augmentation du volume de données collectées et analysées, et de s'étendre en fonction des besoins futurs.

3.3. Enjeux et Contraintes du Projet

Plusieurs enjeux et contraintes doivent être pris en compte pour assurer la réussite du projet, notamment :

- **Volume de Données** : La gestion efficace d'un grand nombre de logs en temps réel, sans nuire aux performances du système.
- **Sécurité** : Garantir l'intégrité des données collectées et le respect des meilleures pratiques de sécurité.
- **Scalabilité** : S'assurer que l'architecture puisse évoluer pour s'adapter à la croissance future des données ou du nombre d'utilisateurs.

4. Conception de l'Architecture

4.1. Schéma d'Architecture ELK

L'architecture ELK a été déployée sur une machine unique fonctionnant sous **RedHat 9.3**, avec l'utilisation de **Docker** et **Docker Compose** pour orchestrer les différents services. Les composants suivants ont été installés et configurés sur cette machine :

- **Elasticsearch** : Pour l'indexation, le stockage et la recherche des données.
- **Kibana** : Pour la visualisation et l'analyse des données indexées par Elasticsearch.
- **Logstash** : Pour la collecte, la transformation et l'envoi des logs vers Elasticsearch.
- **Filebeat** : Pour la collecte des logs à partir des systèmes cibles et leur envoi à Logstash ou directement à Elasticsearch.

- **Metricbeat** : Pour la collecte des métriques systèmes (CPU, mémoire, réseau, etc.) et leur envoi vers Elasticsearch ou Logstash.
- **Setup Certificates**: Pour sécuriser les communications entre tous les composants ELK (Elasticsearch, Logstash, Kibana, et les Beats) grâce à des certificats.

Le schéma d'architecture montre comment ces composants communiquent entre eux, avec **Filebeat** et **Metricbeat** collectant les logs et métriques des systèmes distants, **Logstash** traitant et envoyant ces données vers **Elasticsearch** pour l'indexation, et **Kibana** permettant de visualiser les résultats. Le **setup certificates** est utilisé pour sécuriser ces communications entre les services à travers des canaux chiffrés.

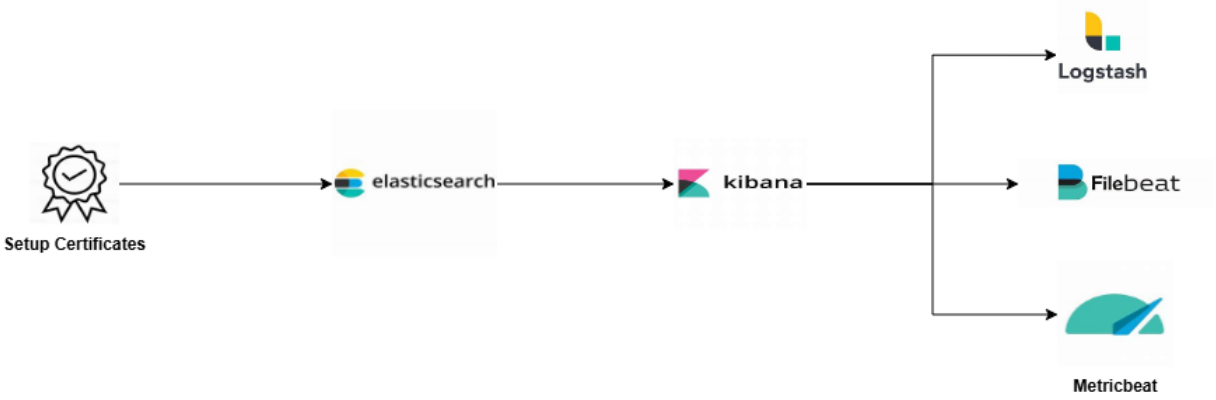


Figure 1 : Schéma d'Architecture ELK

5. Mise en Œuvre de l'Architecture ELK

5.1. Préparation de l'Environnement (Docker et Docker Compose)

La mise en œuvre de l'architecture ELK repose sur une machine virtuelle RedHat 9.3 où Docker et Docker Compose sont utilisés pour déployer les différents services de l'architecture. Voici les étapes principales de la préparation de l'environnement :

- **Installation de Docker** : Docker a été installé sur la machine virtuelle pour permettre l'exécution des conteneurs des différents services de l'architecture ELK (Elasticsearch, Logstash, Kibana, Filebeat, Metricbeat).
- **Installation de Docker Compose** : Docker Compose est utilisé pour simplifier l'orchestration des services et leur interconnexion. Il permet de définir tous les services nécessaires dans un seul fichier de configuration (docker-compose.yml), facilitant ainsi leur gestion et déploiement.

```
[root@localhost elk]# docker --version
Docker version 27.3.1, build ce12230
[root@localhost elk]# docker-compose --version
Docker Compose version v2.30.1
[root@localhost elk]#
```

Figure 2:installation docker & docker-compose

Un fichier docker-compose.yml a été préparé pour définir et orchestrer l'ensemble des services (setup certificates,Elasticsearch, Kibana, Logstash, metricbeat,filebeat)

```
[root@localhost elk]# ls
docker-compose.yml    filebeat.yml  log            logstash_ingest_data
filebeat_ingest_data  kibana.yml    logstash.conf  metricbeat.yml
```

Figure 3:fichier de configuration des services

5.2. Deploiement des services

Le déploiement de l'architecture ELK a été effectué sur une machine virtuelle RedHat 9.3, utilisant Docker et Docker Compose pour orchestrer les différents services nécessaires. Ce processus a impliqué le déploiement successif des composants de l'architecture, en commençant par la configuration des certificats SSL/TLS pour sécuriser les communications entre les services. Ensuite, **Elasticsearch**, **Kibana**, **Logstash**, **Filebeat**, et **Metricbeat** ont été déployés, chacun étant configuré pour fonctionner de manière sécurisée et efficace au sein de l'architecture ELK.

```
[root@localhost elk]# docker-compose up --build -d
WARN[0000] /home/nasri/elk/docker-compose.yml: the attribute `version` is obsole
te, it will be ignored, please remove it to avoid potential confusion
[+] Running 7/7
 ✔ Network elastic              Created                                    0.5s
 ✔ Container es-setup-1         Healthy                                    2.4s
 ✔ Container es-es01-1          Healthy                                   46.6s
 ✔ Container es-filebeat01-1    Started                                   45.9s
 ✔ Container es-kibana-1        Healthy                                   98.5s
 ✔ Container es-metricbeat01-1  Started                                  100.6s
 ✔ Container es-logstash01-1    Started                                  100.6s
[root@localhost elk]#
```

Figure 4:déploiement des services

5.2.1. Déploiement de Setup Certificates

Le service **setup certificates** a été déployé pour générer et fournir des certificatsnécessaires à la sécurisation des communications entre les différents services de l'architecture ELK. Ces certificats permettent à Elasticsearch, Kibana, Logstash, Metricbeat et Filebeat de communiquer de manière sécurisée.

5.2.2. Déploiement de Elasticsearch

Grâce aux certificats fournis par le service **setup certificates**, Elasticsearch a été déployé et configuré pour accepter les connexions sécurisées. Le service a été configuré pour indexer et stocker les données provenant des autres composants.

5.2.3. Déploiement de Kibana

Kibana a été configuré pour se connecter à Elasticsearch de manière sécurisée, en utilisant les certificats fournis. Il permet ainsi de visualiser les données collectées par Elasticsearch dans des dashboards interactifs.

```
[root@localhost elk]# docker ps
CONTAINER ID   IMAGE                                                  COMMAND                  CREATED          STATUS
               PORTS                                                  NAMES
e7b4da5840b2   docker.elastic.co/logstash/logstash:8.8.2              "/usr/local/bin/dock…"   35 minutes ago   Up 34
minutes             5044/tcp, 9600/tcp                                     es-logstash01-1
57d9ced66667   docker.elastic.co/beats/metricbeat:8.8.2               "/usr/bin/tini -- /u…"   35 minutes ago   Up 34
minutes                                                                    es-metricbeat01-1
d31b2a76ff15   docker.elastic.co/beats/filebeat:8.8.2                 "/usr/bin/tini -- /u…"   35 minutes ago   Up 34
minutes                                                                    es-filebeat01-1
5d13fe956b69   docker.elastic.co/kibana/kibana:8.8.2                  "/bin/tini -- /usr/l…"   35 minutes ago   Up 34
minutes (healthy)   0.0.0.0:5601->5601/tcp, :::5601->5601/tcp              es-kibana-1
aabbf8fac1e4   docker.elastic.co/elasticsearch/elasticsearch:8.8.2    "/bin/tini -- /usr/l…"   35 minutes ago   Up 35
minutes (healthy)   0.0.0.0:9200->9200/tcp, :::9200->9200/tcp, 9300/tcp    es-es01-1
[root@localhost elk]#
```

Figure 5:vérification du lancement des services

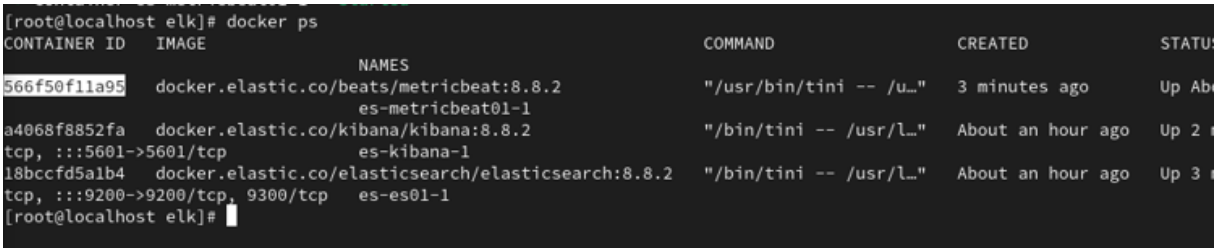
5.2.4. Déploiement de Logstash, Filebeat et Metricbeat

Logstash, Filebeat et Metricbeat ont été déployés pour collecter et envoyer les données vers Elasticsearch, avec une communication sécurisée via les certificats fournis par **setup certificates**. Ces services assurent la collecte des logs et des métriques, qui sont ensuite traités et stockés dans Elasticsearch.

6. Tests et Validation de l'Architecture

Pour garantir le bon fonctionnement et la fiabilité de l'architecture ELK déployée, une série de tests a été effectuée.

Figure 6:: certificats générer par Setup certificates



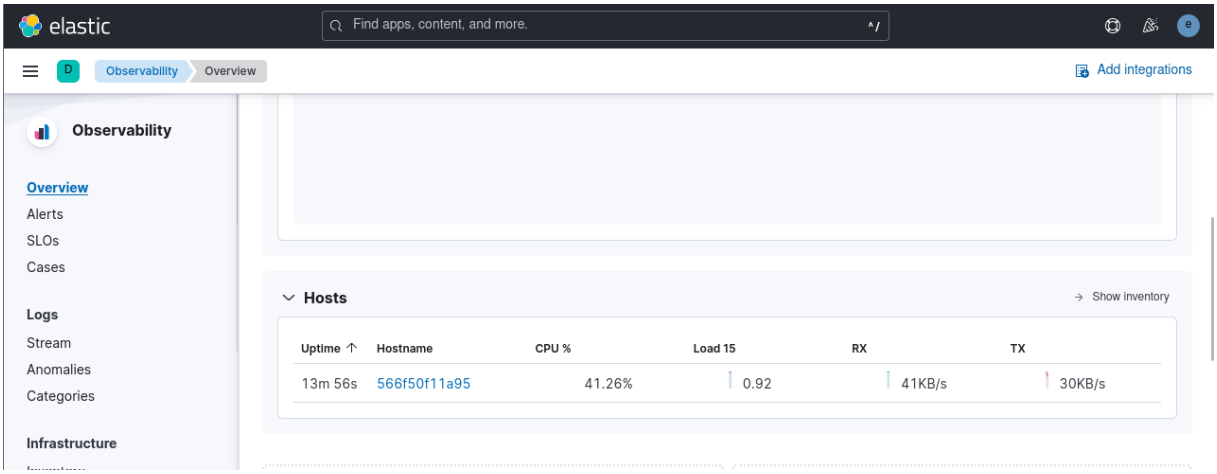
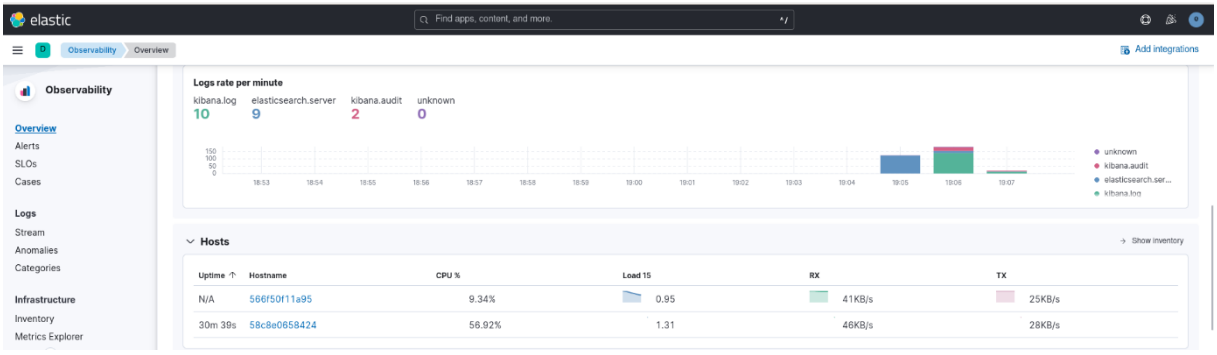
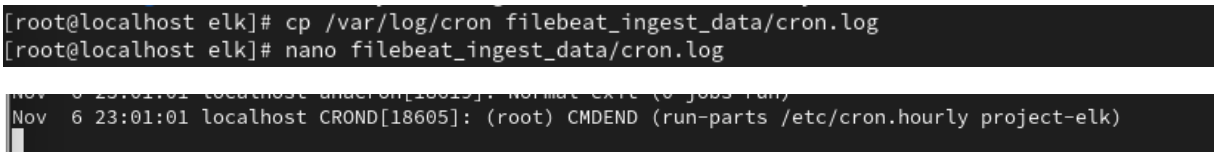


Figure 7:fonctionnement du metricbeat





```
Nov  6 23:01:01 localhost CROND[18605]: (root) CMDEND (run-parts /etc/cron.hourly project-elk)
```

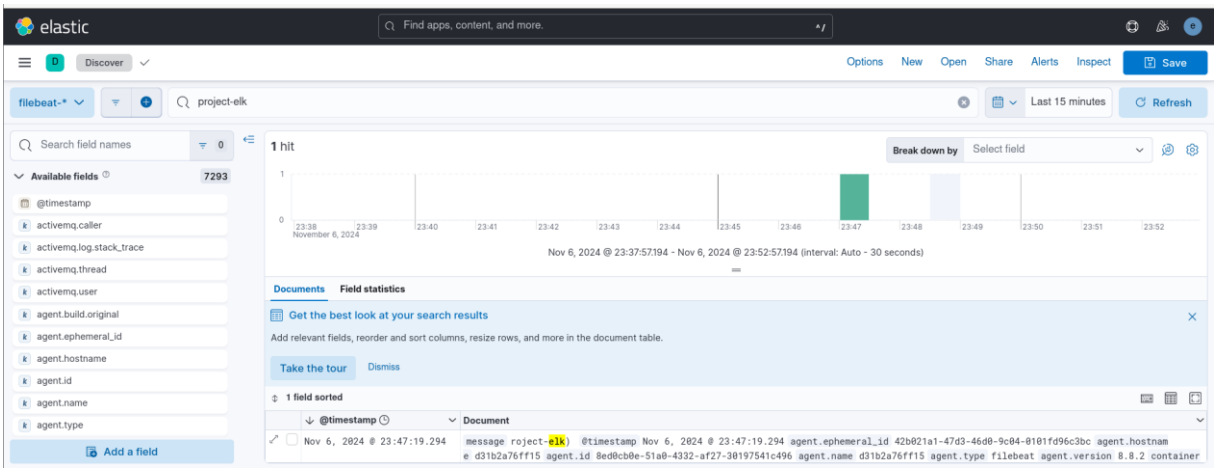
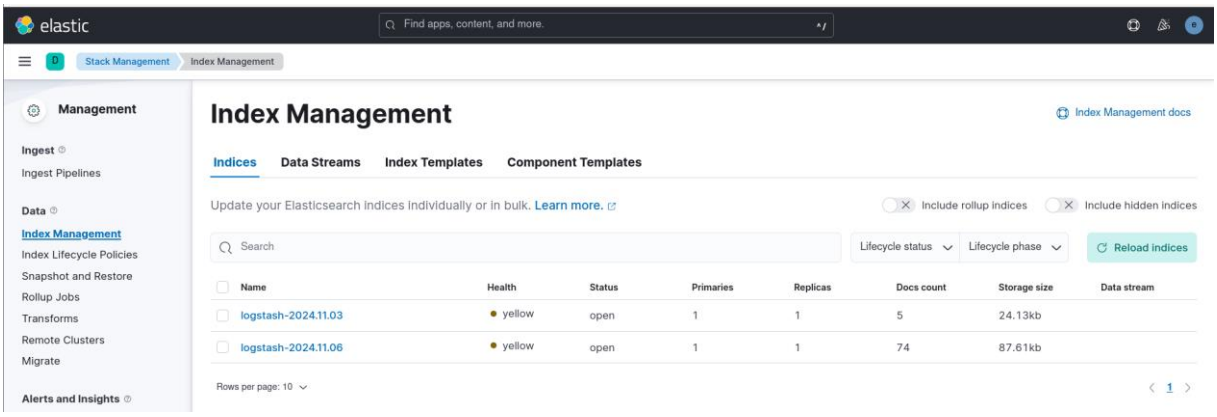
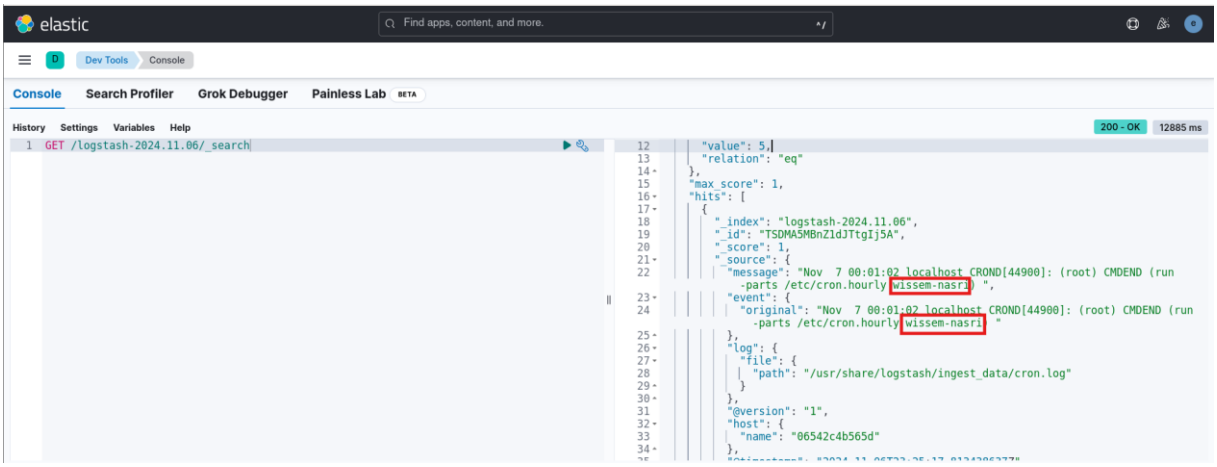


Figure 8:teste de fonctionnement de filebeat

```
  GNU nano 5.6.1                          logstash_ingest_data/cron.log
Nov  3 15:27:24 localhost crond[1407]: (CRON) STARTUP (1.5.7)

Nov  6 23:01:01 localhost CROND[18605]: (root) CMDEND (run-parts /etc/cron.hourly project-elk)
Nov  6 23:01:01 localhost CROND[18605]: (root) CMDEND (run-parts /etc/cron.hourly nasri-wissem)
```





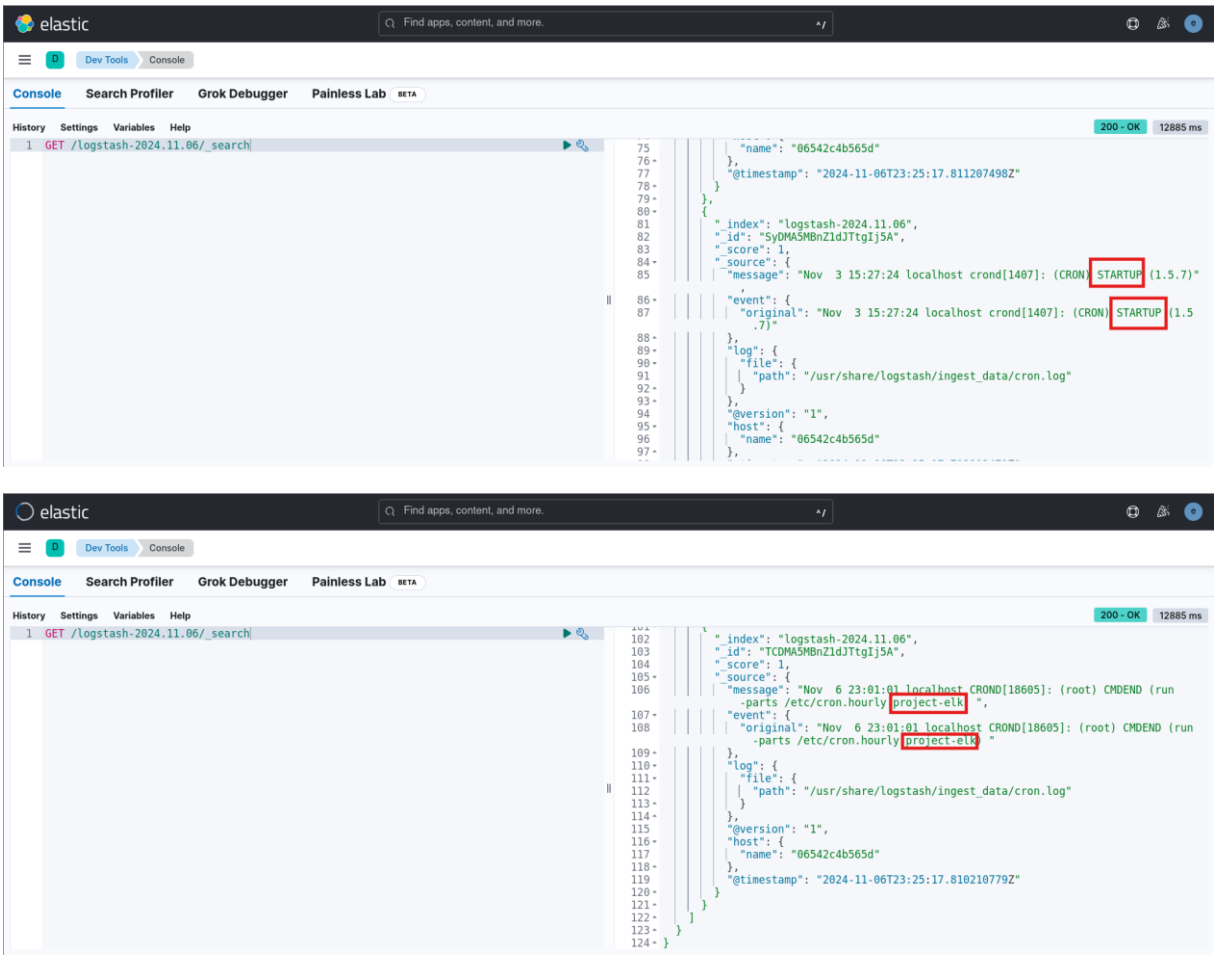


Figure 9:este de fonctionnement de logstash